MASTER

Factorization of indefinite systems associated with RLC circuits

Rosen Esquivel, P.I.

Award date:
2008

Link to publication
Factorization of Indefinite Systems Associated with RLC Circuits

Patricio Rosen

Supervisors:

Prof. Dr. W.H.A. Schilders
Dr. Joost Rommes

A Thesis submitted for the degree of Master in Science

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN
TECHNISCHE UNIVERSITEIT EINDHOVEN
NXP SEMICONDUCTORS

Eindhoven, August 2008
Contents

1 Introduction 1

2 Circuit Equations 4
   2.1 Circuit Equations 4
   2.2 Kirchhoff’s Laws 5
   2.3 Branch Constitutive Relations 5
   2.4 System Formulation 6
   2.5 Complex Phasor Analysis 7
   2.6 Eigenvalue Properties 8

3 Matrix Decompositions 12
   3.1 Linear Solvers 12
   3.2 LU Factorization 13
   3.3 Cholesky Decomposition 14
   3.4 Nonsymmetric Matrix Iterations 15
   3.5 Saddle Point Problems 18
   3.6 Schilders’ Factorization 19

4 Factorization of RL Circuits 21
   4.1 Incidence Matrix Decomposition 21
   4.2 RL Factorization 23
   4.3 Dependences on \( w \) 26
   4.4 Final Factorization 29

5 Factorization RLC Circuits 31
   5.1 Invertibility of the RLC system 31
   5.2 Circuit Equations (Revisited) 33
   5.3 RLC Factorization 35

6 Time Complexity 39
   6.1 Small RL example 39
   6.2 RL Complexity 42
   6.3 RL Study Case 44
   6.4 Example: RLC Ladder Circuit 48
   6.5 RLC Complexity 52

7 Conclusions 57
   7.1 Contribution of this work 57
   7.2 Future Work 58
# List of Figures

1.1 Moore’s Law, source Wikipedia ........................................... 2  
1.2 Microprocessor manufactured by photographic process. Wikipedia Commons, Picture by Angeloleithold 2004, Dec ............... 3  

2.1 Three Nodes Circuit ......................................................... 4  
2.2 Circuit with one current source and two RL branches ......... 8  
2.3 Spectrum of the system in complex form with $\omega = 0$ .......... 9  
2.4 Spectrum trajectories of the system in complex form with $\omega \in (0, 2.2)$ .............................................................. 9  
2.5 Spectrum of the system in stable complex form with $\omega = 0$ . 9  
2.6 Spectrum trajectories of the system in stable complex form with $\omega \in (0, 2.2)$ .............................................................. 9  
2.7 Spectrum of the system in real form with $\omega = 0$ ............ 10  
2.8 Spectrum trajectories of the system in real form with $\omega \in (0, 2.2)$ .............................................................. 10  
2.9 Spectrum of the system in stable real form with $\omega = 0$ .... 10  
2.10 Spectrum trajectories of the system in stable real form with $\omega \in (0, 2.2)$ .............................................................. 10  

3.1 Example with Identity Matrix. All three methods converge in one step. (Source [4]) .................................................. 16  
3.2 Example with Matrix $C$. GMRES and CGS converge till iteration $N$. CGN converges after one single iteration. (Source [4]) .... 17  
3.3 Example with Random Matrix. All three methods require $N$ iterations. (Source [4]) ..................................................... 17  

4.1 Original incidence matrix ($47 \times 29$) .................................. 23  
4.2 Incidence matrix in lower trapezoidal form ($47 \times 29$) ......... 23  
4.3 Original incidence matrix ($452 \times 256$) ............................ 24  
4.4 Incidence matrix in lower trapezoidal form ($452 \times 256$) .... 24  

6.1 Simple RL Circuit ............................................................. 39  
6.2 RL Circuit ........................................................................ 44  
6.3 Solution of the System and Control Toolbox for the voltage at node 3 ................................................................. 45  
6.4 Solution with RL algorithm at node 3 ............................... 45  
6.5 Magnitude plot for different coupling factors ..................... 45  
6.6 Phase plot for different coupling factors ........................... 45  
6.7 Matrix $P_1^{-1}$ ................................................................. 46  
6.8 Product $P_1^{-1}P_1^{-T}$ ..................................................... 46
<table>
<thead>
<tr>
<th>Section Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.9</td>
<td>Solution running times for RL circuits without coupling ( p = 0 )</td>
<td>47</td>
</tr>
<tr>
<td>6.10</td>
<td>Solution running times for RL circuits with coupling ( p = 0.1 )</td>
<td>47</td>
</tr>
<tr>
<td>6.11</td>
<td>Executions time for the modified RL circuits without coupling</td>
<td>47</td>
</tr>
<tr>
<td>6.12</td>
<td>Executions time for the modified RL circuits with coupling</td>
<td>47</td>
</tr>
<tr>
<td>6.13</td>
<td>RLC Ladder Circuit with N Blocks</td>
<td>48</td>
</tr>
<tr>
<td>6.14</td>
<td>Magnitude and Phase of the Voltage at node 2 in a RLC Ladder</td>
<td>51</td>
</tr>
<tr>
<td></td>
<td>with 1 block</td>
<td></td>
</tr>
<tr>
<td>6.15</td>
<td>RLC Ladder Circuit with N Blocks</td>
<td>53</td>
</tr>
<tr>
<td>6.16</td>
<td>Matrix ( \mathbf{P}_T^{-1} )</td>
<td>54</td>
</tr>
<tr>
<td>6.17</td>
<td>Product ( \mathbf{P}_1^{-1} \mathbf{P}_1^{-T} )</td>
<td>54</td>
</tr>
<tr>
<td>6.18</td>
<td>RLC Ladder Circuit with N Blocks</td>
<td>55</td>
</tr>
<tr>
<td>6.19</td>
<td>RLC Ladder Circuit with N Blocks</td>
<td>55</td>
</tr>
</tbody>
</table>

iii
# List of Algorithms

<table>
<thead>
<tr>
<th></th>
<th>Algorithm</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Gaussian Elimination Algorithm</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>LU Decomposition Algorithm</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>Cholesky Algorithm</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>Incidence Matrix Algorithm</td>
<td>23</td>
</tr>
<tr>
<td>5</td>
<td>RL Solution Algorithm</td>
<td>42</td>
</tr>
<tr>
<td>6</td>
<td>RLC Solution Algorithm</td>
<td>52</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The semiconductor industry is concerned with many different areas. One of those areas is circuit simulation. Circuit simulation helps to understand the way in which circuits work. Besides it is also useful for predicting errors and for optimizing the design of the circuits before actually constructing them. Circuit simulation uses mathematical models describing the behavior of both the individual components and the interactions between them. Predicting the behavior of a circuit before building it improves the efficiency and provides useful information to the circuit designers. Moreover, in the case of integrated circuits for example, the simulation of circuits is one of the main tools in which circuit design relies. This is the case, because performing experiments is usually very expensive and unpractical.

We can summarize this by saying that, circuit simulation is a key tool in the electronics industry. One of the characteristics in this industry is the always increasing complexity of the circuits. The number of components in a single chip keeps increasing day after day. For example, in Figure 1.2 we can observe a typical integrated circuit manufactured by photographic process. In figure 1.1, we can observe a graph showing the growth of the number of transistors in different commercial integrated circuits. This is known as Moore’s Law.

Nowadays it is very common to find circuit designs involving a very large number of components. Consequently the mathematical models describing the circuits also contain many variables. This characteristic makes direct solvers, like Gaussian elimination for instance, to be very time consuming. This happens specially when extra interactions between the components, like inductive coupling, are taken into account. The usage of iterative solvers is also restricted, due to the indefiniteness and the poor spectral properties of the correspondent linear systems. We will discuss this more precisely in further chapters.

The problem at hand can be classified as saddle point problem. Solution techniques have been proposed to tackle this kind of systems. In our particular case we will concentrate on developing exact factorizations of the RLC circuit equations. The key component for constructing these factorizations will be the Schilders factorization. In this work we present how to achieve this goal. To that extend we organized this thesis as follows.
This first chapter contains the introduction. In the second chapter we review the basics of circuit simulation. We focus on constructing the equations describing the behavior of RLC circuits. We also analyze the properties of the derived system of equations. In the third chapter we try to give an overview of the strategies for solving linear equations, algorithms and their corresponding complexities are included. Later, the saddle point problems are introduced and we present some of the difficulties that arise when using iterative solvers with non symmetric and indefinite matrices. In this same chapter we also present the Schilders factorization; this idea will lead us later to the factorizations of the circuit equations.

Chapter 4 develops the particular RL case. First we discuss in detail the step of rearranging the incidence matrix. We include a proof of the existence of such rearrangement and provide an algorithm for finding the involved permutation matrices. Then we present the RL factorization and discuss the frequency dependencies of the factor matrices. The firstly presented factorization is then reorganized to take the form of a factorization of type LDU.

In chapter 5, we treat the more general case of the RLC circuits. First we prove the invertibility of the RLC system. Then we reorganize the circuit equations and write in a more suitable form for developing the factorization. With this information in hand we proceed to construct the factorization. We provide proofs for the existence of the factorization.

In order to show how the factorizations work, we present some small illus-
Figure 1.2: Microprocessor manufactured by photographic process. *Wikimedia Commons, Picture by Angeloleithold 2004, Dec*

Illustrative examples in chapter 6. Then we formalize the usage of the factorizations for developing direct solvers. This is done by stating it in terms of algorithms for each case. We derive expressions for the time complexity of each of the algorithms and compare it with the time complexity of direct LU decomposition. Furthermore, we provide two circuit examples, one for the RL case and one for the RLC case, in which we are able to increase the number of components arbitrarily. This allows us to compare the performance of the algorithms in practice. We include and discuss the numerical results.

Finally, we include a last chapter containing the conclusions and some ideas for future work.
Chapter 2

Circuit Equations

In this chapter we review the basics of circuit simulation. We start by handling the topology of the circuit. Then we introduce Kirchhoff’s Laws and the Branch constitutive relations. Finally these are put together and written in terms of a differential algebraic equation. The properties of the system are discussed as well. We took [2] and [17] as main references for this chapter.

2.1 Circuit Equations

Theoretically speaking a circuit is a set of interconnected nodes. These connections are either simple ideal wires or more complicated components like resistors $R$, inductors $L$, and capacitors $C$. Additional to these components we consider the current sources. In order to formulate the circuit equations we first need to describe its topology. This is done by means of the so-called incidence matrix $A$. Each row of the matrix $A$ is associated with one branch of the circuit and the columns correspond to the circuit nodes. By convention, a row has $+1$ in the corresponding source node, $-1$ in the destination node and 0 everywhere else. Notice that one of the columns (ground) needs to be removed in order to avoid redundancy. As an example let us consider the circuit with three nodes in Figure 2.1.

![Three Nodes Circuit](image)

Figure 2.1: Three Nodes Circuit

The incidence matrix for this circuit is given in (2.1). Note that the rows
were ordered according to the labeling of the branches in Figure 2.1. With the topological information in hand we can proceed to include Kirchhoff’s laws.

\[
\begin{pmatrix}
N_1 & N_2 & N_3 \\
1 & -1 & 0 \\
1 & 0 & -1 \\
0 & 1 & -1
\end{pmatrix}
\begin{pmatrix}
B_1 \\
B_2 \\
B_3
\end{pmatrix}
\] (2.1)

### 2.2 Kirchhoff’s Laws

Kirchhoff’s current law states that the sum of all branch currents entering any closed surface is zero. In our case we are dealing with nodes, for this particular case, Kirchhoff’s current law says that all currents entering a node add to zero.

Kirchhoff’s voltage law states that the sum of all branch voltages along any closed loop in a circuit add to zero. This can be formulated in the following way. Let \( i_b, v_b \) and \( v_n \) be the vectors of branch currents, branch voltages and node voltages respectively. Then Kirchhoff’s current and voltage laws read

\[
A^T i_b = 0, \quad Av_n = v_b.
\] (2.2)

In order to fully describe the system, the only thing missing are equations describing the behavior of the branch elements. We will consider four kinds of elements. Namely resistors, inductors, capacitors and current sources. These kind of circuits are known as RLC circuits. With this idea in mind it is useful to decompose the matrix \( A \) and the corresponding vectors as follows:

\[
A = \begin{bmatrix}
A_i & A_g & A_c & A_l
\end{bmatrix}, \quad v_b = \begin{bmatrix}
v_i \\
v_g \\
v_c \\
v_l
\end{bmatrix}, \quad i_b = \begin{bmatrix}
i_i \\
i_g \\
i_c \\
i_l
\end{bmatrix},
\] (2.3)

where the subscripts \( i, g, c \) and \( l \) denote current source, conductance, capacitor and inductor respectively. With this decomposition Kirchhoffs current law (2.4) appears in the following way:

\[
A_i^T i_i + A_g^T i_g + A_c^T i_c + A_l^T i_l = 0.
\] (2.4)

Kirchhoffs voltage law (2.5) simply says

\[
A_i v_n = v_i, \quad A_g v_n = v_g, \quad A_c v_n = v_c, \quad A_l v_n = v_l.
\] (2.5)

### 2.3 Branch Constitutive Relations

We now need to include the branch constitutive relations, for each set of components in our circuit. These equations are:

\[
i_i = I_i(t), \quad i_g = G v_g, \quad i_c = C \frac{d}{dt} v_c, \quad v_l = \frac{d}{dt} i_l.
\] (2.6)
where \( \mathbf{i}(t) \) denotes the vector of current-source values, \( \mathcal{G} \) and \( \mathcal{C} \) denote the conductances and capacitances and are diagonal matrices. The matrix \( \mathcal{L} \) denotes the inductances. \( \mathcal{L} \) is diagonal in the absence of inductive coupling. Inductive coupling adds off-diagonal terms, but the matrix remains symmetric and positive definite. The whole system can be described by using only \( \mathbf{i} \) and \( \mathbf{v}_n \). Thus equations (2.4),(2.5) and (2.6) can be written as:

\[
\mathbf{A}_l \mathbf{v}_n - \mathcal{L} \frac{d}{dt} \mathbf{i} = 0 \quad (2.7a)
\]

\[
\mathbf{A}_l^T \mathbf{i}(t) + \mathbf{A}_g^T \mathcal{G} \mathbf{A}_g \mathbf{v}_n + \mathbf{A}_c^T \mathcal{C} \mathbf{A}_c \frac{d}{dt} \mathbf{v}_n + \mathbf{A}_l^T \mathbf{i} = 0 \quad (2.7b)
\]

and in system form,

\[
\begin{pmatrix}
0 & \mathbf{A}_l \\
\mathbf{A}_l^T & \mathbf{A}_g^T \mathcal{G} \mathbf{A}_g
\end{pmatrix}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} +
\begin{pmatrix}
-\mathcal{L} & 0 \\
0 & \mathbf{A}_c^T \mathcal{C} \mathbf{A}_c
\end{pmatrix}
\frac{d}{dt}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} =
\begin{pmatrix}
0 \\
\mathbf{A}_l^T
\end{pmatrix}
\mathbf{i}(t)
\quad (2.8)
\]

### 2.4 System Formulation

In the previous derivation it was assumed that a branch can contain a pure inductor. In practice it is very natural to model an inductor in series with a resistor. If we consider this, then the first equation in (2.7) would include an extra term \( R \mathbf{i} \) accumulating the resistances. This leads to the system:

\[
\begin{pmatrix}
-R & \mathbf{A}_l \\
\mathbf{A}_l^T & \mathbf{A}_g^T \mathcal{G} \mathbf{A}_g
\end{pmatrix}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} +
\begin{pmatrix}
-\mathcal{L} & 0 \\
0 & \mathbf{A}_c^T \mathcal{C} \mathbf{A}_c
\end{pmatrix}
\frac{d}{dt}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} =
\begin{pmatrix}
0 \\
\mathbf{A}_l^T
\end{pmatrix}
\mathbf{i}(t)
\quad (2.9)
\]

Now multiplying by \(-1\) in all the equations we get the system:

\[
\begin{pmatrix}
-R & \mathbf{A}_l \\
\mathbf{A}_l^T & \mathbf{A}_g^T \mathcal{G} \mathbf{A}_g
\end{pmatrix}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} +
\begin{pmatrix}
\mathcal{L} & 0 \\
0 & -\mathbf{A}_c^T \mathcal{C} \mathbf{A}_c
\end{pmatrix}
\frac{d}{dt}
\begin{pmatrix}
\mathbf{i} \\
\mathbf{v}_n
\end{pmatrix} =
\begin{pmatrix}
0 \\
\mathbf{A}_l^T
\end{pmatrix}
\mathbf{i}(t)
\quad (2.10)
\]

Notice that now \( \mathbf{i} \) stands for the current of the inductor-resistor branches. It is common to write this system in general form:

\[
\mathbf{G} \mathbf{x}(t) + \mathbf{C} \frac{d}{dt} \mathbf{x}(t) = \mathbf{B} \mathbf{u}(t) \quad (2.11)
\]

This system is formulated as a symmetric but indefinite system. This is due to the form of the first matrix \( \mathbf{G} \) of our system. Matrices with that structure can be decomposed in the following way:

\[
\begin{pmatrix}
\mathbf{A} & \mathbf{P} \\
\mathbf{P}^T & -\mathbf{D}
\end{pmatrix} = \begin{pmatrix}
\mathbf{I} & \mathbf{0} \\
\mathbf{0} & \mathbf{S}
\end{pmatrix} \begin{pmatrix}
\mathbf{I} & \mathbf{0} \\
\mathbf{0} & \mathbf{S}^{-1} \mathbf{P}
\end{pmatrix}
\quad (2.12)
\]

Where \( \mathbf{S} = -(\mathbf{D} + \mathbf{P}^T \mathbf{A}^{-1} \mathbf{P}) \) is the Schur complement of \( \mathbf{A} \) in the system matrix. Then using Silvester’s Law of Inertia [14, p.403] we conclude that
2.5 Complex Phasor Analysis

Often we are interested in the sinusoidal steady-state behavior of the circuit, i.e. we want to know how the circuit behaves when applying excitations with fixed frequency. This is known as the Alternating Current or AC analysis. In order to find this behavior we can make use of the so-called complex phasor analysis. We consider the circuit equations in general form (2.11), and an input at a fixed frequency $u(t) = \theta \cos(\omega t + \phi)$. Here $\omega$ and $\theta$ stand for the frequency and the amplitude respectively.

Now we realize that the input can be seen as the real part of a complex exponential, namely
\[
\theta \cos(\omega t + \phi) = \text{Re}(\theta e^{i\omega t + \phi}) = \text{Re}(\Theta e^{i\omega t}).
\]  
(2.13)

In order to include sinusoidal inputs we consider a more general input, namely $u = \Theta e^{i\omega t}$. This leads to the system in complex variable $z(t) = x(t) + iy(t)$.

Now replacing $z(t)$ by a complex exponential we get:
\[
Gz(t) + \frac{d}{dt}z(t) = B\Theta e^{i\omega t},
\]  
(2.14)

Now reducing $z(t)$ by a complex exponential we get:
\[
G(Z(\omega)e^{i\omega t}) + C \frac{d}{dt}(Z(\omega)e^{i\omega t}) = B\Theta e^{i\omega t},
\]  
(2.15)

after differentiation and canceling $e^{i\omega t}$ we find:
\[
GZ(\omega) + i\omega CZ(\omega) = B\Theta
\]  
(2.16)

Thus $Z(\omega)$ is obtained after solving this linear system of complex matrices:
\[
Z(\omega) = (G + i\omega C)^{-1}B\Theta.
\]  
(2.17)

For the special case of our RLC circuit, i.e. substituting $G$ and $C$ we get the system:

\[
\begin{bmatrix}
\hat{R} & \hat{P} \\
\hat{P}^T & -\hat{C}
\end{bmatrix}
+ i\omega
\begin{bmatrix}
\hat{L} & 0 \\
0 & -\hat{C}
\end{bmatrix}
\begin{bmatrix}
i_{r} & i_{i} \\
i_{r} + i_{i} & v_{r} + iv_{i}
\end{bmatrix}
= \begin{bmatrix}0 \\ A_{i}^{T}\end{bmatrix} \Theta.
\]  
(2.18)

Here $i_{r}$ represents the real part of the current vector, and $i_{i}$ the imaginary part (the same for the vector of voltages). $\Theta$ is a vector with the amplitudes of the source signals. Since these values only act as scaling factors one usually takes $\Theta = 1$. Additionally we have $\hat{P} = -A_{i}$, $\hat{R} = R$, $\hat{L} = L$, $\hat{C} = A_{g}^{T}GA_{g}$ and $\hat{G} = A_{c}^{T}CA_{c}$ (see (2.10).
2.6 Eigenvalue Properties

This system has the advantage of being symmetric but it is complex and some of the properties of the submatrices are lost, for instance $\hat{R} + i\omega \hat{L}$ is not positive definite anymore. This system can also be rewritten as a real system of equations.

$$
\begin{pmatrix}
\hat{R} & -\omega \hat{L} & \hat{P} & 0 \\
\omega \hat{L} & \hat{R} & 0 & \hat{P} \\
\hat{P}^T & 0 & -\hat{C} & \omega \hat{C} \\
0 & \hat{P}^T & -\omega \hat{C} & -\hat{G} \\
\end{pmatrix}
\begin{pmatrix}
i_r \\
i_i \\
v_r \\
v_i \\
\end{pmatrix}
=
\begin{pmatrix}
0 \\
0 \\
A_i^T \\
0 \\
\end{pmatrix}
\Theta 
\tag{2.19}
$$

By using the same argument as above we can see that the system is indefinite and also non symmetric. The spectral properties of the system in complex form and in real form are different. In what follows we will analyze these properties briefly.

### 2.6 Eigenvalue Properties

To discuss the spectral properties of the system formulations we will consider the following example.

![Circuit with one current source and two RL branches](image)

**Figure 2.2:** Circuit with one current source and two RL branches

For this simple circuit, after setting the node number 3 as ground we get:

$$
A_t = \begin{pmatrix} 1 & -1 \\ 1 & 0 \end{pmatrix}, \quad A_i = \begin{pmatrix} 0 & -1 \end{pmatrix} \quad \text{and} \quad P = \begin{pmatrix} -1 & 1 \\ -1 & 0 \end{pmatrix}
\tag{2.20}
$$

Now let us take the following values for resistances and inductances. Notice that we assume the inductors to be coupled, the off diagonal terms represent the coupling.

$$
\mathcal{R} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}, \quad \mathcal{L} = \begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix}
\tag{2.21}
$$

For the system written in complex form (2.18) we check that the system is in fact indefinite. In Figure 2.3 we observe the location of the eigenvalues of the complex matrix in (2.18), the four eigenvalues are real (since $\omega = 0$). Two eigenvalues are negative while the other two are positive. In Figure 2.4 we can observe the paths of the eigenvalues till the value $\omega = 2$. We realize that the eigenvalues on the left half plane tend to zero. The real part of the
2.6 Eigenvalue Properties

Circuit Equations

The complex system (2.18) can be transformed into a complex system with the property of being positive semistable. This means that for all \( \lambda \in \sigma(A) \) we have \( \text{Re}(\lambda) \geq 0 \). This can be achieved by multiplying the second set of equations by -1. Thus we get the equivalent system:

\[
\begin{bmatrix}
    \hat{R} & \hat{P} \\
    \hat{P}^T & -\hat{G}
\end{bmatrix} + i\omega
\begin{bmatrix}
    \hat{L} & 0 \\
    0 & \hat{C}
\end{bmatrix}
\begin{bmatrix}
    i_r + i_i \\
    v_r + v_i
\end{bmatrix} = \begin{bmatrix}
    0 \\
    -A_i^T
\end{bmatrix} \Theta, \quad (2.22)
\]

The spectrum of this system can be seen in Figures 2.5 and 2.6. In the first Figure, the eigenvalues appear in conjugate pairs and their real part is in fact positive. In Figure 2.6 we observe that the eigenvalues stay in the right half plane, two of the eigenvalues tend to zero and the real part of the other two remains bounded by one. Their imaginary part grows together with \( \omega \).
2.6 Eigenvalue Properties

The system in real form (2.19) is indefinite as it was stated before. Since the systems is now real, then the eigenvalues appear in conjugate pairs. First in Figure 2.7 we observe the position of the eigenvalues when \( \omega = 0 \). In this Figure all the eigenvalues have multiplicity two. When \( \omega > 0 \) the eigenvalues stop being real and appear in conjugate pairs (see Figure 2.8). We find the same behavior as in the case with the complex system. The eigenvalues on the left half plane tend to zero while the real part of the other ones is bounded from below by 1. In fact we can observe that the spectrum of the real system can be obtained by including the conjugates of the eigenvalues found in the complex form.

As it was done before for the system in complex form the real system can also be reformulated in a positive semistable form. This can be achieved by multiplying by -1 the last two set of equations of the system (2.19). In Figures 2.9 and 2.10 we can observe the path of the eigenvalues of the system in this form. Again we can observe that the profile of the spectrum of the system in real form can be obtained from the profile for the complex stable form by adding the "missing" conjugate pairs.

This description provides us with useful information. First we know that the systems are in fact indefinite. A more detailed description regarding the
problems that arise when dealing with indefinite and nonsymmetric systems will be treated with more detail in the next chapter. Nonetheless, this formulation can be changed to a positive semidefinite form. This property might be useful when applying iterative techniques, but we will concentrate more in developing factorizations and their use for implementing direct solvers. We will be able to construct a factorization of the circuit equations in real form.
Chapter 3

Matrix Decompositions

In this chapter we review some strategies for the solution of linear systems namely, direct and iterative solvers. We briefly analyze their properties and we discuss some of the problems one might face when dealing with indefinite and non symmetric systems. We go back to our special kind of systems, the saddle point problem, and discuss some of the solution methods heading finally toward Schilders factorization, which is the path we will follow later to solve the circuit equations. As a main source for the algorithms presented in this chapter we refer to the book by Demmel [13].

3.1 Linear Solvers

In order to solve a linear system we have the choice between direct and iterative solvers. The usual arguments to use iterative methods are mainly computer storage and computation time. But it is important to notice that iterative solvers require some expertise and if CPU time and storage are not really at stake, then it is simpler to apply a direct solver.

Algorithm 1 Gaussian Elimination Algorithm

INPUT: $n \times n$ non singular matrix $A = a_{ij}$, vector $b$
OUTPUT: solution vector $x$ s.t $Ax = b$
1. for $i = 1, ..., n-1$ (for each row $i$)
2. for $j = i + 1, ..., n$ (subtract a multiple of row $i$ through $n$)
3. $b_j = b_j - \frac{a_{ji}}{a_{ii}} b_i$ (actualizing vector $b$)
4. for $k = i, ..., n$ (in columns $i$ through $n$)
5. $a_{jk} = a_{jk} - \frac{a_{ji}}{a_{ii}} a_{ik}$ (to zero out column $i$ below the diagonal)
6. end for
7. end for
8. end for
9. Solve resulting upper triangular system with backward substitution

A direct method leads, in the absence of rounding errors, in a finite and fixed amount of work, to the exact solution of the given linear system. These rounding errors can be handled very well by means of pivoting strategies. The
main problem of direct solvers is that their complexity, in the general case, is of order $O(n^3)$, which for very big systems might become too expensive.

In scientific computing lots of computational time is spent solving systems of linear equations. These systems can be really large, for instance in computational fluid flow problem and in circuit simulation. In the latter case some of the designs might involve variables to the order of one million. The process of solving the unknowns from these large linear systems implies lots of computational work. The standard direct approach is Gaussian elimination (Algorithm 1).

Now we will analyze the complexity of this algorithm. For a system with $n$ equations and $n$ unknowns we need $2(n - 1)^2$ operations for creating zeros in the first column. Then, for the second column we need $2(n - 2)^2$ operations. Thus to eliminate all the elements in the lower triangular part we require $\frac{2}{3}n^3$ operations. Finally the resulting system comes in upper triangular form:

$$
\begin{pmatrix}
  a_{11} & a_{12} & \ldots & a_{1n} \\
  0 & a_{22} & \ldots & a_{2n} \\
  \vdots & \ddots & \ddots & \vdots \\
  0 & 0 & \ldots & a_{nn}
\end{pmatrix}
\begin{pmatrix}
  x_1 \\
  x_2 \\
  \vdots \\
  x_n
\end{pmatrix}
= 
\begin{pmatrix}
  b_1 \\
  b_2 \\
  \vdots \\
  b_n
\end{pmatrix}
$$

(3.1)

This system can be easily solved by means of backward substitution. Leading to a complexity of order $n^2$, which is a relatively minor cost for large values of $n$. Altogether we see that the cost of performing Gaussian elimination to a system is proportional to $n^3$.

For very large systems Gaussian Elimination is not very attractive and in some cases the computation time might be just not affordable. Nonetheless, once Gaussian Elimination has been performed, the resulting lower triangular system can be solved fast (order $n^2$). A question comes out naturally. How to improve the performance of the algorithms in order to find a factorization of the system?

A factorization of a matrix $A$ is a representation of $A$ as a product of several simpler matrices. The solution to the original system can be found faster by solving a sequence of systems involving these matrices. In this sense the matrices are simpler.

### 3.2 LU Factorization

Usually Gaussian Elimination is carried out by first factorizing the matrix $A$ and then solving the simpler systems. To do so we recall the following theorem.

**Theorem 1. LU Decomposition** If the $n$ by $n$ matrix $A$ is nonsingular, there exist a permutation matrix $P$, a nonsingular lower triangular matrix $L$, and a nonsingular upper triangular matrix $U$ such that $A = PLU$. The system $Ax = b$ is then solved by solving $PLUx = b$ as follows:

- $LUx = P^Tb$ (permuting entries of $b$)
- $Ux = L^{-1}(P^Tb)$ (solving with forward substitution)
- $x = U^{-1}(L^{-1}P^Tb)$ (solving with backward substitution)

There is a condition for having the factorization without needing a permutation matrix. This result is stated as follows.
Theorem 2. The following statements are equivalent:
1. There exists a unique unit lower triangular $L$ and nonsingular upper triangular $U$ such that $A = LU$.
2. All leading principal submatrices of $A$ are nonsingular.

Without much effort similar theorems can be proved to deliver LDU factorizations.

Theorem 3. If $A$ is a $n \times n$ nonsingular matrix then, there exist a permutation matrix $P$, a nonsingular lower triangular matrix $L$ a nonsingular diagonal matrix $D$ and a nonsingular upper triangular matrix $U$ such that $A = PLDU$.

The algorithm for computing these factorizations (LU or LDU) is just Gaussian elimination. Pivoting strategies can lead to some extra properties of the factors. For example by using "partial pivoting" we can assure that all entries of $L$ are bounded by 1 in absolute value. This is the most common implementation of Gaussian elimination. There is some other pivoting strategy that is almost never used in practice, the Gaussian elimination with complete pivoting. In this case the permutations are chosen in such a way that $a_{11}$ is the largest entry in absolute value in the whole matrix. The following algorithm shows how to perform a PLU decomposition.

**Algorithm 2 LU Decomposition Algorithm**

**INPUT:** $n \times n$ non singular matrix $A = a_{ij}$

**OUTPUT:** permutation matrix $P$, unit lower triangular matrix $L$ and a nonsingular upper triangular matrix $U$ such that $A = PLU$ (overwriting $L$ and $U$ on $A$)

1. for $i = 1, \ldots, n - 1$
2. apply permutations so $a_{ii} \neq 0$ (permute $L$ and $U$ too)
   (for partial pivoting, swap rows $j$ and $i$ of $A$ and of $L$ where $|a_{ij}|$ is the largest entry in $|A(i : n, j)|$)
3. $A(i + 1 : n, i) = A(i + 1 : n, i)/A(i, i)$,
   
   $A(i + 1 : n, i + 1 : n) = A(i + 1 : n, i + 1 : n) - A(i + 1 : n, i) \ast A(i, i + 1 : n)$
4. end for

In the last line of the algorithm, $A(i + 1 : n, i) \ast A(i, i + 1 : n)$ is the product of an $(n - i) \times 1$ matrix $(L_{21})$ by a $1 \times (n - i)$ matrix $(U_{12})$, which yields an $(n - i) \times (n - i)$ matrix. This is the general setting for Gaussian Elimination. In order to increase speed of solution and decrease storage it is important to exploit any special structure of the matrix. For example if the matrix is symmetric and positive definite (s.p.d.) the computation time as well as the storage, for solving $Ax = b$ can be decreased by a half.

### 3.3 Cholesky Decomposition

Let us recall that a matrix $A$ is said to be symmetric if and only if $A = A^T$ and positive definite if and only if $x^T Ax > 0$ for all $x \neq 0$. Now we recall the following theorem from the literature.
Theorem 4. 1. If \( X \) is nonsingular, then \( A \) is s.p.d if and only if \( X^TAX \) is s.p.d.
2. If \( A \) is s.p.d. and \( H \) is any principal submatrix of \( A \) (\( H = A(j : k, j : k) \), \( j \leq k \)), then \( H \) is s.p.d.
3. \( A \) is s.p.d if and only if \( A = A^T \) and all of its eigenvalues are positive.
4. \( A \) is s.p.d. if and only if there is a unique lower triangular nonsingular matrix \( L \), with positive diagonal entries, such that \( A = LL^T \). \( A = LL^T \) is called the Cholesky factor of \( A \).

The next algorithm delivers the Cholesky factorization of a symmetric positive definite matrix \( A \).

Algorithm 3 Cholesky Algorithm

**INPUT:** \( n \times n \) s.p.d. matrix \( A = a_{ij} \)

**OUTPUT:** lower triangular nonsingular matrix \( L \) such that \( A = LL^T \)

1. for \( i = 1, \ldots, n \)
2. \( l_{jj} = (a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2)^{1/2} \)
3. for \( i = j + 1, \ldots, n \)
4. \( l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}) / l_{jj} \)
5. end for
6. end for

If \( A \) is not positive definite, then this algorithm will fail by attempting to compute the square root of a negative number or by dividing by zero; this is the cheapest way to test if a symmetric matrix is positive definite.

As with Gaussian elimination, \( L \) can overwrite the lower half of \( A \). Only the lower half of \( A \) is referred to by the algorithm, so in fact only \( n(n+1)/2 \) storage is needed instead of \( n^2 \). The number of flops is just half the flops of Gaussian elimination. Pivoting is not necessary for Cholesky to be numerically stable.

### 3.4 Nonsymmetric Matrix Iterations

So far we have seen that direct Gaussian Elimination might be too expensive for very large linear systems. We also reviewed some results regarding matrix decompositions as LU decompositions and the Cholesky factorization. Some of these results will be used later in further chapters. Now we will discuss the iterative schemes.

For the application of iterative schemes one usually has sparse patterns in mind, for instance linear systems arising from finite element or finite difference approximations of a partial differential equation. However the structure of the operator plays no explicit role in any of these schemes, and they may also successfully be used to solve certain large dense linear systems. Some iterative methods may be much faster for special problems, like for Poisson Solvers to mention an example. For matrices that are not positive definite nor symmetric the situation can be more problematic. It is often difficult to find a proper iterative method or a suitable preconditioner. In our special case (Circuit equations) we are dealing with a so called saddle point system. Since these systems are
3.4 Nonsymmetric Matrix Iterations

Matrix Decompositions

indefinite and in our special case also non symmetric, they represent a significant challenge for solver developers. To make this clear we recall a paper by Nachtigal [4].

In this paper [4] the speed of convergence of nonsymmetric matrix iterations is studied. The iterative schemes that are analyzed in that paper are CGN (Conjugate Gradient Iteration Applied to the Normal Equations), GMRES (Generalized Minimal Residual Method), and CGS (a Biorthogonalization Algorithm adapted from the Biconjugate Gradient iteration).

![Figure 3.1: Example with Identity Matrix. All three methods converge in one step. (Source [4])](image)

First in figure 3.1 we observe how the three methods behave when applied to the identity matrix. As expected, all methods converge after one single iteration. For the experiment the identity matrix of size 40 was used. A random real initial vector $x_0$ and right hand side $r$ with independent normally distributed elements of mean 0 and variance 1 were taken. On the other hand, we will see what happens when the system is non symmetric and indefinite. For instance, if we consider the next system, involving a $n \times n$ matrix:

$$C x = r,$$

where $C$ is

$$
\begin{pmatrix}
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
\vdots & \vdots & \vdots & \ddots & \ddots \\
1 & 0 & \ldots & 0 & 1
\end{pmatrix}
$$

(3.2)

The matrix $C$ can be obtained from a simple permutation of the identity matrix. Nonetheless, this matrix is already nonsymmetric and indefinite, the elements of the right hand side $r$, were taken to be independent normally distributed with mean 0 and variance 1. After this simple change we can observe (Figure 3.2) that both GMRES and CGS perform in a very bad way. During the first $N - 1$ (recall that for this example it is considered $N = 40$) iterations no improvement is observed for both GMRES and CGS. Only till the last iteration both methods converge. Meanwhile, since $C^T C = I$, the CGN method converges after one single step, but we will see that this is not always the case.

All this happens after applying a simple permutation to the identity matrix. In a more general case that it is also explored in the same paper by Nachtigal.
3.4 Nonsymmetric Matrix Iterations

Matrix Decompositions

Figure 3.2: Example with Matrix C. GMRES and CGS converge till iteration \( N \). CGN converges after one single iteration. (Source [4])

They show that the three methods converge in a slow way when a random matrix \( R \) of size \( 40 \times 40 \) was taken. The entries of the matrix \( R \) were independent normally distributed random numbers with mean 0 and variance 1. Such matrix has condition number \( O(N) \) on average and smoothly distributed singular values [18], so that CGN will require \( N \) steps for convergence. The eigenvalues are approximately uniformly distributed in a disk of radius \( \sqrt{N} \) about the origin, thus GMRES and CGS will also require \( N \) steps to converge. In Figure 3.3 we can observe that the convergence of CGS is oscillating, while the other two are monotonic. It is important to notice that only GMRES exhibits the convergence in \( N \) steps that would be achieved by all three methods in exact arithmetic.

Figure 3.3: Example with Random Matrix. All three methods require \( N \) iterations. (Source [4])

At this point it is time to mention that CGN, GMRES and CGS are the best nonsymmetric matrix iterations. This in the sense that, for calculations in exact arithmetic measured by the residual norm, no other iteration ever outperforms these three by more than a constant factor, except in certain examples involving special initial residuals. Roughly speaking, the convergence of CGN is determined by the singular values of \( A \). If \( A \) is normal or close to normal, then
the convergence of GMRES is determined by the eigenvalues of $A$; the singular values and in particular the condition number, have nothing to do with it.

If $A$ is far from normal the convergence of GMRES becomes slower by a potentially unbounded factor. To make this clear we cite the paper by Greenbaum [3]. In this paper it is shown that any nonincreasing convergence curve is possible for GMRES. The convergence of CGS is very close related to that of GMRES, but the convergence of CGS is additionally affected by instabilities that are not yet fully understood.

3.5 Saddle Point Problems

From the previous examples we can conclude that nonsymmetric and indefinite systems are indeed a challenging problem for solver developers. In our case it is important to notice that we are dealing with a system with this characteristics. Nonetheless we still have some special structure.

\[ \mathbf{A} \mathbf{x} = \mathbf{b}, \quad \text{or} \quad \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & -\mathbf{C} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \mathbf{c} \end{pmatrix} \quad (3.3) \]

These kind of systems are known as saddle point type, and they arise not only in circuit simulation but in very different kinds of applications. In recent years there has been a surge of interest in saddle point problems, and different solution techniques have been proposed for solving this type of systems. A direct attempt for solving this system is direct eliminations of the unknowns $\mathbf{x}$.

This can be done for the more general case when $\mathbf{C} \neq 0$, but only in order to illustrate the idea of the method we will assume $\mathbf{C} = 0$. Assuming this we can solve for the vector $\mathbf{x}$ and get:

\[ \mathbf{x} = \mathbf{A}^{-1}\mathbf{b} - \mathbf{A}^{-1}\mathbf{B}\mathbf{y}, \quad (3.4) \]

and substituting this in the other set of equations we get:

\[ \mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}\mathbf{y} = \mathbf{B}^T\mathbf{A}^{-1}\mathbf{b} - \mathbf{c}. \quad (3.5) \]

This approach is known as the range space method or the Schur complement method [5]. This approach is attractive if the order $m$ of the reduced system is small and if linear systems with coefficient matrix $\mathbf{A}$ can be solved efficiently. The main disadvantages are the need for $\mathbf{A}$ to be nonsingular (which is not a necessary condition, for the “whole” system matrix $\mathbf{A}$, to be nonsingular), and the fact that the Schur complement $\mathbf{S} = -\mathbf{B}^T\mathbf{A}^{-1}\mathbf{B}$ may be completely full and too expensive to compute and factor. However, when $\mathbf{B}$ has no dense columns and $\mathbf{A}$ is such that $\mathbf{A}^{-1}$ is sparse, then the Schur complement is usually quite sparse. In these cases this approach performs well.

Another method that is in some sense complementary to the previous method is the null space method. This method only works for the case when $\mathbf{C} = 0$. Here the variables $\mathbf{y}$ are eliminated from the system as follows. It is assumed that there is a matrix $\mathbf{Z}$ which columns are a basis for the null space of $\mathbf{B}^T$, i.e. $\mathbf{B}^T\mathbf{Z} = 0$. Let us further assume that $\hat{\mathbf{y}}$ is particular solution of $\mathbf{B}^T\mathbf{B}\hat{\mathbf{y}} = \mathbf{c}$. Then the solution set of $\mathbf{B}^T\mathbf{x} = \mathbf{c}$ can be written as:

\[ \mathbf{x} = \mathbf{B}\hat{\mathbf{y}} + \mathbf{Z}\mathbf{z}, \quad (3.6) \]
Substituting this expression for $x$ in the first set of equations, we obtain:

$$AZz + By = b - AB\gamma.$$  \hfill (3.7)

Multiplying this by $Z^T$ and using the fact that $Z^TB = 0$, we find

$$Z^T AZz = Z^T b - Z^T AB\gamma.$$  \hfill (3.8)

The null space method has the advantage of not requiring $A^{-1}$. And the
coefficient matrix looks more attractive than the one obtained with the range
method, provided $A$ is a sparse matrix. However, in order not to perturb the
sparsity, one will have to take care that the matrix $Z$ is also rather sparse. This
means that a sparse basis for the null space has to be used. For certain problems
this is certainly possible (for our special case we will prove that this can be done).

With the discussion so far it should be clear that the solution of indefinite and
nonsymmetric systems is a difficult task, and that their treatment is far from
being uniform. In our case we will focus on developing solution strategies based
in the so called Schilders Factorization.

### 3.6 Schilders’ Factorization

The Schilders factorization has been developed for systems of the form [1], [5]:

$$A = \begin{pmatrix} \hat{A} & \hat{B} \\ B^T & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}.$$  \hfill (3.9)

where $\hat{A}$ is a $n \times n$ nonsingular and symmetric matrix, and $\hat{B}$ is a $n \times m$ full rank
matrix. It is assumed that $m \leq n$. The Schilders factorization is constructed
in the following way. First we perform the $LQ$ of the matrix $\hat{B}$. This means that
we construct an $n \times n$ permutation matrix $\Pi$ and an $m \times m$ orthogonal matrix $Q$ such that

$$\Pi \hat{B} = BQ$$  \hfill (3.10)

where $B$ is of lower trapezoidal form. Furthermore we require the top $m \times m$ part of $B$ to be nonsingular. Now we can define:

$$Q = \begin{pmatrix} \Pi & 0 \\ 0 & Q \end{pmatrix}$$

and let $A = \Pi \hat{A} \Pi^T$, then

$$QAQ^T = \begin{pmatrix} A & B \\ B^T & 0 \end{pmatrix}.$$  \hfill (3.11)

The Schilders factorization can be performed in this final form of the system
as follows.

**Lemma 1.** Let $A$ and $B$ be as in the foregoing, and write $B^T = (B_1, B_2)^T$, where $B_1$ is the $m \times m$ top part of $B$. Then there exist $D_1$ an $m \times m$ diagonal
matrix, $D_2$ an $(n-m) \times (n-m)$ diagonal matrix, $L_1$ an $m \times m$ strictly lower triangular matrix, $L_2$ an $(n-m) \times (n-m)$ strictly lower triangular matrix and
3.6 Schilders’ Factorization

Matrix Decompositions

$M$ an $(n - m) \times m$ matrix, such that:

$$
\begin{pmatrix}
A & B \\
B^T & 0
\end{pmatrix} =
\begin{pmatrix}
B_1 & 0 \\
B_2 & I_{n-m} + L_2
\end{pmatrix}
\begin{pmatrix}
L_1 & M \\
0 & I_m
\end{pmatrix}
\begin{pmatrix}
D_1 & 0 & I_m \\
0 & D_2 & 0 \\
I_m & 0 & 0
\end{pmatrix}
\begin{pmatrix}
B_1^T & B_2^T & 0 \\
0 & I_{n-m} + L_2^T & 0 \\
L_1^T & M^T & I_m
\end{pmatrix}
$$

(3.12)

Using the previous lemma. The following theorem can be proved.

**Theorem 5.** Let $A$ be an $n \times n$ symmetric, positive definite matrix, $B$ an $n \times m$ matrix of full rank, $m \leq n$, and set

$$
A = \begin{pmatrix}
\hat{A} \\
\hat{B}
\end{pmatrix},
$$

Then there exist an $n \times n$ permutation matrix $\Pi$, an $m \times m$ orthogonal matrix $Q$, an $m \times m$ diagonal matrix $D_1$, an $(n - m) \times (n - m)$ diagonal nonsingular matrix $D_2$, an $m \times m$ strictly lower triangular matrix $L_1$, an $(n - m) \times (n - m)$ strictly lower triangular matrix $L_2$, and an $(n - m) \times m$ matrix $M$, such that $\Pi BQ^T$ is lower trapezoidal and

$$
A = QCQ^T L^T Q^T,
$$

where

$$
Q = \begin{pmatrix}
0 & I_{n} \\
Q^T & 0
\end{pmatrix},
$$

$$
L = \begin{pmatrix}
I_m & 0 \\
L_1 & B_1 \\
M & B_2 & I_{n-m} + L_2
\end{pmatrix},
$$

$$
D = \begin{pmatrix}
0 & I_m & 0 \\
I_m & D_1 & 0 \\
0 & 0 & D_2
\end{pmatrix}
$$

We can observe that with this factorization the original matrix is decomposed into simpler matrices, for which the solution can be calculated by simple backward or forward substitution, which are of order $O(n^2)$. Thus we will attempt to use Schilders factorization as a basis for developing a decomposition of the circuit equations in the next chapters.
Chapter 4

Factorization of RL Circuits

As we saw in the previous chapter certain symmetric indefinite systems can be decomposed by means of the Schilders factorization. The system of equations for RLC circuits is no longer symmetric and the structure is also different from the one for which the Schilders factorization can be directly performed. In this chapter we first start by constructing a factorization for RL circuits. In this special case no capacitive effects are taken into account (i.e. \( C = 0 \)). We also assume that there are no pure conductances in the circuit (i.e. \( G = 0 \)). Under this assumptions the matrix has a structure for which a Schilders type factorization can be performed. Moreover this factorization also provides us with a proof of the existence and uniqueness of solutions of the AC analysis of RL circuits.

4.1 Incidence Matrix Decomposition

As we saw above one of the main ingredients in the Schilders Factorization is to rearrange the matrix \( \hat{B} \). In the general case this is done by means of the LQ decomposition. This means we construct an \( n \times n \) permutation matrix \( \Pi \) and an \( m \times m \) orthogonal matrix \( Q \) such that,

\[
\Pi \hat{B} = BQ
\]

where, \( B \) is of lower trapezoidal form.

For the case of circuit equations our matrix \( \hat{B} \) has more properties. It is an incidence matrix. For such matrices it is possible to make the decomposition by means of only permutations. This means that the equation above holds but with both \( \Pi \) and \( Q \) being permutation matrices. Of course \( B \) remains in lower trapezoidal form. This is stated in the following theorem.

Theorem 6. Incidence Matrix Decomposition Let \( \hat{P} \) be the \( n \times m \) (njm) incidence matrix associated with a connected network, after deleting one column (grounded node). Then there exist \( \Pi_1 \) an \( n \times n \) and \( \Pi_2 \) an \( m \times m \) permutations matrices such that:

\[
\Pi_1 \hat{P} = P \Pi_2
\]

where \( P^T = (P_1, P_2)^T \) with \( P_1 \) is an \( m \times m \) lower triangular and nonsingular matrix.
4.1 Incidence Matrix Decomposition

Proof: We will proceed by induction over \( m \) (remember that the number of nodes of the network is \( m + 1 \)).

When \( m = 1 \) we have \( \hat{P} = v \), where \( v \) is a vector. Since there are only 2 nodes we have \( v_i \neq 0 \) for all \( i = 1, \ldots, n_1 \), where \( n_1 \) is the number of branches of the graph with only 2 nodes. For this case it is enough to take \( \Pi_1 = I_n \) and \( \Pi_2 = I_m \). Now assume the theorem is true for any graph with \( m \) nodes (with an \( n_2 \times m - 1 \) incidence matrix), and let \( \hat{P} \) be an arbitrary \( n_3 \times m \) incidence matrix.

Let us remember that \( \hat{P} \) was obtained after deleting one of the original columns (the ground node). Since the graph is assumed to be connected, there exists at least one branch which either starts or ends at the ground node. Let this branch be in position \( i \) with corresponding node in position \( j \), and define \( \Pi_r (n \times n) \) and \( \Pi_c (m \times m) \) the row permutation matrices that interchange the rows number 1 and \( i \) and the rows 1 and \( j \) respectively. Then we have:

\[
\Pi_r \hat{P} \Pi_c = \begin{pmatrix} v & M \end{pmatrix}.
\] (4.3)

Since the branch lies between the node \( j \) and the grounded node we have first that \( v_1 = x \neq 0 \). Furthermore this also implies that the first row of \( M \) has to be zero. Thus we have:

\[
\Pi_r \hat{P} \Pi_c = \begin{pmatrix} x & 0 \\ v & \hat{P} \end{pmatrix}
\] (4.4)

where \( \hat{P} \) is an incidence matrix associated with the original network after deleting the node \( j \). Thus, by the inductive step there exist permutation matrices \( \Pi_{r1}, \Pi_{c1} \) such that \( \Pi_{r1} \hat{P} = \hat{P}_1 \Pi_{c1} \) with \( \hat{P}_1 \) being lower trapezoidal with invertible top. Finally by defining

\[
\Pi_{r2} = \begin{pmatrix} 1 & 0 \\ 0 & \Pi_{r1} \end{pmatrix}, \quad \Pi_{c2} = \begin{pmatrix} 1 & 0 \\ 0 & \Pi_{c1} \end{pmatrix}
\] (4.5)

we have,

\[
P = \begin{pmatrix} x & 0 \\ * & \hat{P}_1 \end{pmatrix} = \Pi_{r2} \begin{pmatrix} x & 0 \\ v & \hat{P} \end{pmatrix} \Pi_{c2}^T = \Pi_{r2} \Pi_r \hat{P} \Pi_c \Pi_{c2}^T
\] (4.6)

where \( * \) is a permutation of the vector \( v \). Clearly the matrix \( P \) is lower trapezoidal and since \( x \neq 0 \), then the top part of the matrix is invertible. The required permutation matrices are given as follows.

\[
\Pi_1 = \Pi_{r2} \Pi_r, \quad \Pi_2 = \Pi_{c2} \Pi_c^T
\] (4.7)

QED.

The decomposition can be performed by means of algorithm 4. In this algorithm, when looking at the third step, a natural question comes out. Is it always possible to find such \( k \) and \( l \)? The answer is yes and it is provided by the argument given in theorem 6. The complexity of this algorithm is \( O(n^2) \), making use of sparse structures to store the incidence matrix \( \hat{P} \).

In Figure 4.1 and 4.2 we can see the shape of an original \( 47 \times 29 \) incidence matrix and its rearranged lower trapezoidal form after applying algorithm 4.

22
4.2 RL Factorization

Algorithm 4 Incidence Matrix Algorithm

INPUT: \( n \times m \) incidence matrix \( \hat{B} \)

OUTPUT: matrix \( B \) in lower trapezoidal form, \( \Pi_1, \Pi_2 \) row and column permutation matrices s.t. \( \Pi_1 \hat{B} = B \Pi_2 \)

1. Define \( B = \hat{B}, \Pi_1 = I_{n,n}, \Pi_2 = I_{m,m} \)
2. for \( i = 1, 2, ..., m \) do
3. Find \( k, l \) such that \( \sum_{j=i}^m |B_{k,j}| = 1 \) and \( B_{k,l} \neq 0 \)
4. Permute rows \( i \) and \( k \) of matrices \( B \) and \( \Pi_1 \)
5. Permute columns \( i \) and \( l \) of matrix \( B \), permute rows \( i \) and \( l \) of matrix \( \Pi_2 \)
6. end for

In Figure 4.1 we can observe a \( 452 \times 256 \) incidence matrix in its original form. After applying our algorithm we get the matrix in lower trapezoidal form (Figure 4.2). This provides us with an important tool to factorize the circuit equations.

4.2 RL Factorization

In this chapter we are working with RL circuits, this means that, we consider circuits without capacitors and without pure conductances (i.e. \( \hat{C} = \hat{G} = 0 \)). Now let \( m + 1 \) be the number of nodes and \( n \) the number of branches in the circuit. Thus equation (2.19) takes the following form:
4.2 RL Factorization

Factorization of RL Circuits

Figure 4.3: Original incidence matrix (452 × 256)

Figure 4.4: Incidence matrix in lower trapezoidal form (452 × 256)

\[ A = \begin{bmatrix} \hat{R} & -\omega \hat{L} & \hat{P} & 0 \\ \omega \hat{L} & \hat{R} & 0 & \hat{P} \\ \hat{P}^T & 0 & 0 & 0 \\ 0 & \hat{P}^T & 0 & 0 \end{bmatrix} \begin{bmatrix} i_r \\ i_i \\ v_r \\ v_i \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ A^T \end{bmatrix} \Theta \] (4.8)

where \( \hat{R} \) is an \( n \times n \) diagonal positive definite matrix, \( \hat{L} \) is an \( n \times n \) symmetric positive definite matrix and \( \hat{P} \) is an \( n \times m \) incidence matrix. Since \( \hat{P} \) is an incidence matrix we can then apply algorithm 4 and find permutation matrices \( \Pi_1 (n \times n) \), \( \Pi_2 (m \times m) \) such that \( \Pi_1 \hat{P} = P \Pi_2 \). Then defining

\[ Q = \begin{pmatrix} \Pi_1 \\ \Pi_1 \\ \Pi_2 \\ \Pi_2 \end{pmatrix} \] (4.9)

and letting \( R = \Pi_1 \hat{R} \Pi_1^T \) and \( L = \Pi_1 \hat{L} \Pi_2^T \) we get:

\[ QAQ^T = \begin{bmatrix} R & -\omega L & P & 0 \\ \omega L & R & 0 & P \\ P^T & 0 & 0 & 0 \\ 0 & P^T & 0 & 0 \end{bmatrix} \] (4.10)

where \( P^T = (P_1, P_2)^T \). Moreover, due to the way the algorithm was constructed we have that \( P_1 \) is an \( m \times m \) nonsingular lower triangular matrix and \( P_2 \) is an \( n - m \times m \) containing the rest of the incidence matrix. Now let us subdivide the matrices \( R \) and \( L \) as follows.
4.2 RL Factorization

Factorization of RL Circuits

\[ R = \begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{pmatrix}, \quad L = \begin{pmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{pmatrix} \]  

(4.11)

then, defining the permutation matrix \( \Pi_3 \) as follows:

\[ \Pi_3 = \begin{pmatrix} I_m & & & \\ & I_{n-m} & & \\ & & I_{n-m} & \\ & & & I_{2m} \end{pmatrix} \]  

(4.12)

We can rearrange the equations by multiplying with \( \Pi_3 \), we have that:

\[ \Pi_3 Q A Q^T \Pi_3^T = \begin{pmatrix} R_{11} - \omega L_{11} & R_{12} - \omega L_{12} & P_1 & 0 \\ \omega L_{11} & R_{11} & R_{12} & 0 & P_1 \\ R_{21} - \omega L_{21} & R_{22} - \omega L_{22} & P_2 & 0 \\ \omega L_{21} & R_{21} & R_{22} & 0 & P_2 \\ P_1^T & 0 & P_2^T & 0 & 0 \\ 0 & P_1^T & 0 & P_2^T & 0 & 0 \end{pmatrix} \]  

(4.13)

Let us organize this matrix in the following blocks.

\[ \Pi_3 Q A Q^T \Pi_3^T = \begin{pmatrix} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ B_1^T & B_2^T & 0 \end{pmatrix} \]  

(4.14)

here

\[ B_1 = \begin{pmatrix} P_1 \\ P_1 \end{pmatrix}, \quad B_2 = \begin{pmatrix} P_2 \\ P_2 \end{pmatrix} \]

and

\[ A_{ij} = \begin{pmatrix} R_{ij} & -\omega L_{ij} \\ \omega L_{ij} & R_{ij} \end{pmatrix}, \quad i, j \in \{1, 2\}. \]  

(4.15)

Once our system equation have been set in this form we can perform a Schilders type factorization. Such kind of factorizations have been developed to construct preconditioners for nonsymmetric saddle point matrices [7]. We construct the factorization as it is stated in the next theorem.

**Theorem 7.** The matrix for the RL circuit arranged as in 4.14 can be decomposed in the following way:

\[ \begin{pmatrix} A_{11} & A_{12} & B_1 \\ A_{21} & A_{22} & B_2 \\ B_1^T & B_2^T & 0 \end{pmatrix} = \begin{pmatrix} D_1 & 0 & L_1 \\ B_2 & L_2 & M \\ 0 & 0 & I_{2m} \end{pmatrix} \begin{pmatrix} D_1 & 0 & I_{2m} \\ D_2 & 0 & 0 \\ I_{2m} & 0 & 0 \end{pmatrix} \begin{pmatrix} B_1^T & B_2^T & 0 \\ 0 & U_2 & 0 \\ U_1 & F & I_{2m} \end{pmatrix} \]  

(4.16)

where \( M \) is an \( 2(n-m) \times 2m \) matrix, \( F \) is an \( 2(n-m) \times 2m \) matrix. \( D_1 \) is \( 2m \times 2m \) diagonal, \( L_1 \) is \( 2m \times 2m \) strictly lower triangular and \( U_1 \) is \( 2m \times 2m \) strictly upper triangular. Meanwhile the matrices \( D_2 \), \( L_2 \) and \( U_2 \) are \( 2(n-m) \times 2(n-m) \) nonsingular matrices with \( D_2 \) diagonal, \( L_2 \) lower triangular and \( U_2 \) upper triangular.

**Proof:** First notice that the last matrix from the decomposition is not necessarily the transpose of the first one, in contrast with the original Schilders factorization. We now observe that the factorization holds if and only if the following equations are satisfied.
4.3 Dependences on $w$

Factorization of RL Circuits

\begin{align*}
B_1D_1B_1^T + B_1U_1 + L_1B_1^T &= A_{11} \quad (4.17) \\
B_1D_1B_2^T + B_1F + L_1B_2^T &= A_{12} \quad (4.18) \\
B_2D_1B_1^T + B_2U_1 + MB_1^T &= A_{21} \quad (4.19) \\
L_2D_2U_2 + B_2D_1B_2^T + B_2F + MB_2^T &= A_{22} \quad (4.20)
\end{align*}

Using (4.17) we can find $D_1$, $U_1$ and $L_1$. First let us recall that $B_1$ is invertible, in fact we have:

\begin{equation}
B_1^{-1} = \begin{pmatrix} P_1^{-1} \\ p_1^{-1} \end{pmatrix}.
\end{equation}

(4.21)

Now we can multiply equation (4.17) by $B_1^{-1}$ from the left and by $B_1^{-T}$ from the right:

\begin{equation}
D_1 + U_1B_1^{-T} + B_1^{-1}L_1 = B_1^{-1}A_{11}B_1^{-T}.
\end{equation}

(4.22)

From this equation, using the fact that $U_1$ is upper triangular, $L_1$ is lower triangular and that $P_1^{-1}$ is also lower triangular we get:

\begin{equation}
L_1 = B_1 \text{strlow} (B_1^{-1}A_{11}B_1^{-T}), \quad U_1 = \text{strupp} (B_1^{-1}A_{11}B_1^{-T}) B_1^T,
\end{equation}

(4.23)

and

\begin{equation}
D_1 = \text{diag} (B_1^{-1}A_{11}B_1^{-T}),
\end{equation}

(4.24)

where “diag” means to take only the diagonal elements of the matrix, “strlower” means to take the strictly lower part of the matrix, and “strupp” means to take the strictly upper part of a matrix. We can now proceed to compute the matrices $F$ and $M$ from equations (4.18) and (4.19) respectively. In fact we find,

\begin{equation}
F = B_1^{-1} (A_{12} - B_1D_1B_2^T - L_1B_2^T), \quad M = (A_{21} - B_2D_1B_1^T - B_2U_1) B_1^{-T}.
\end{equation}

(4.25)

We are left with the problem of finding the matrices $D_2$, $L_2$ and $U_2$. To do so we write equation (4.20) as follows,

\begin{equation}
L_2D_2U_2 = A_{22} - B_2D_1B_2^T - B_2F - MB_2^T.
\end{equation}

(4.26)

By performing an $LDU$ decomposition on the matrix in the right hand side we can find the last matrices. The remaining step is to show that in fact this last matrix can be decomposed. This can be achieved by applying Cholesky decomposition to some submatrices. We will prove this fact in the following section (see lemma 2). This completes the proof. QED.

4.3 Dependences on $w$

Now we proceed to show that the matrices $D_2$, $L_2$ and $U_2$ can be found. In order to do so we will first explore how $w$ plays a role in the factorization. By doing so we will also prove that the matrices $D_2$, $L_2$ and $U_2$ exist and are nonsingular. In the previous section we found expressions for the matrices $D_1, U_1$ and $L_1$.
in terms of $B_1^{-1}$ and $A_{11}$. By substituting the expressions for $B_1^{-1}$ and $A_{11}$, namely

$$ B_1^{-1} = \begin{pmatrix} P_1^{-1} \\ P_1^{-1} \end{pmatrix}, \quad A_{11} = \begin{pmatrix} R_{11} & -\omega L_{11} \\ \omega L_{11} & R_{11} \end{pmatrix} \quad (4.27) $$

we find that,

$$ D_1 = \begin{pmatrix} \text{diag}(P_1^{-1} R_{11} P_1^{-T}) \\ \text{diag}(P_1^{-1} R_{11} P_1^{-T}) \end{pmatrix}, \quad (4.28) $$

$$ L_1 = \begin{pmatrix} \text{strupp}(P_1^{-1} R_{11} P_1^{-T}) \\ \omega L_{11} P_1^{-T} \\ P_1 \text{strlow}(P_1^{-1} R_{11} P_1^{-T}) \end{pmatrix} \quad (4.29) $$

and

$$ U_1 = \begin{pmatrix} \text{strupp}(P_1^{-1} R_{11} P_1^{-T}) P_1^{-T} \\ 0 \\ -\omega P_1^{-1} L_{11} \\ \text{strupp}(P_1^{-1} R_{11} P_1^{-T}) P_1^{-T} \end{pmatrix}. \quad (4.30) $$

Using these previous expressions we can describe more precisely the matrices $F$ and $M$. In fact we get

$$ F = \begin{pmatrix} P_1^{-1} R_{12} - \text{low}(P_1^{-1} R_{11} P_1^{-T}) P_2^T \\ \omega(P_1^{-1} L_{12} - P_1^{-1} L_{11} P_1^{-T} P_2^T) \\ P_1^{-1} R_{12} - \text{low}(P_1^{-1} R_{11} P_1^{-T}) P_2^T \end{pmatrix} \quad (4.31) $$

$$ M = \begin{pmatrix} R_{21} P_1^{-T} - P_2 \text{upp}(P_1^{-1} R_{11} P_1^{-T}) \\ \omega L_2 P_1^{-T} \\ -\omega (L_{21} P_1^{-T} + P_2 P_1^{-1} L_{11} P_1^{-T}) \\ R_{21} P_1^{-T} - P_2 \text{upp}(P_1^{-1} R_{11} P_1^{-T}) \end{pmatrix}. \quad (4.32) $$

Here upp and low stand for taking the lower and the upper part of a matrix, i.e. including the diagonal.

The only terms remaining to be determined are $L_2, D_2$ and $U_2$. These matrices are to be taken out from the $LDU$ decomposition of $\tilde{W} := A_{22} - B_2 D_1 B_2^T - B_2 F - M B_2^T$ (see (4.26)). This is stated in the next lemma.

**Lemma 2.** Define $\hat{W} := A_{22} - B_2 D_1 B_2^T - B_2 F - M B_2^T$ with the expression given above for the matrices involved. Then $\hat{W}$ can be factorized as follows:

$$ \hat{W} = \begin{pmatrix} L_{2,1} & 0 \\ \omega L_{2,2} \\ L_{2,3}(\omega) \end{pmatrix} \begin{pmatrix} D_{2,1} \\ D_2(\omega) \end{pmatrix} \begin{pmatrix} L_{2,1}^T & 0 \\ -\omega L_{2,2}^T \\ L_{2,3}^T(\omega) \end{pmatrix} \quad (4.33) $$

where $L_{2,1}, L_{2,2}$ and $D_{2,1}$ are matrices independent of $\omega$ and $L_{2,3}$ and $D_{2,2}$ are functions of $\omega$. Additionally we have that $L_{2,1}$ and $L_{2,3}$ are nonsingular lower triangular matrices, $D_{2,1}$ and $D_{2,2}$ are diagonal and nonsingular as well.
4.3 Dependences on $w$

Factorization of RL Circuits

**Proof:** using the previous expressions we see that the matrix $\hat{W}$ has the following $2 \times 2$ block structure,

\[
\hat{W}_{11} = R_{22} - R_{21}P_1^{-T}P_2^T - P_2P_1^{-1}R_{12} + P_2P_1^{-1}R_{11}P_1^{-T}P_2^T
\]
\[
\hat{W}_{12} = -\omega(L_{22} - P_2P_1^{-1}L_{11}P_1^{-T}P_2^T + L_{21}P_1^{-T}P_2^T + P_2P_1^{-1}L_{12})
\]
\[
\hat{W}_{21} = \omega(L_{22} - P_2P_1^{-1}L_{11}P_1^{-T}P_2^T + L_{21}P_1^{-T}P_2^T + P_2P_1^{-1}L_{12})
\]
\[
\hat{W}_{22} = R_{22} - R_{21}P_1^{-T}P_2^T - P_2P_1^{-1}R_{12} + P_2P_1^{-1}R_{11}P_1^{-T}P_2^T
\]

(4.34)

From here we first realize that these blocks can be written as follows.

\[
\hat{W}_{11} = \hat{W}_{22} = \begin{pmatrix} -P_2P_1^{-1} & I \\ R_{21} & R_{22} \end{pmatrix} \begin{pmatrix} -P_1^{-T}P_2^T \\ I \end{pmatrix}
\]

(4.35)

\[
-\hat{W}_{12} = \hat{W}_{21} = \omega \begin{pmatrix} -P_2P_1^{-1} & I \\ L_{21} & L_{22} \end{pmatrix} \begin{pmatrix} -P_1^{-T}P_2^T \\ I \end{pmatrix}
\]

(4.36)

It is important to observe that both matrices (in the right hand side of equations (4.35) and (4.36)) are symmetric and positive definite. Thus we first realize that, $L_{2,1}, D_{2,1}$ can be computed from the $LDU$ decomposition of the matrix $\hat{W}_{11}$. Having $\hat{W}_{11}$ invertible is a sufficient condition (theorem 3) for decomposing the matrix as an LDU product. In our particular case we have even more properties, $\hat{W}_{11}$ is symmetric and positive definite and thus it can be decomposed as an $L D L^T$ product (theorem 4). Additionally we know that the matrices $L_{2,1}$ and $D_{2,1}$ are invertible. With this information we find out that the matrix $\hat{W}_{2,2}$ has the following expression.

\[
L_{2,2} = \hat{W}_{22}L_{2,1}^{-T}D_{2,1}^{-1}
\]

(4.37)

After doing some computations we are left finally with the problem of determining $L_{2,3}$ and $D_{2,2}$ such that,

\[
L_{2,3}D_{2,2}L_{2,3}^T = \hat{W}_{11} + \omega^2\hat{W}_{22}\hat{W}_{11}^{-1}\hat{W}_{21}.
\]

(4.38)

The matrix on the right hand side of the equation appears as the sum of two matrices. $\hat{W}_{11}$ is clearly symmetric and positive definite, the second matrix is also symmetric, and since the inverse of a positive definite matrix is also positive definite, we can conclude that $\omega^2\hat{W}_{22}\hat{W}_{11}^{-1}\hat{W}_{21}$ is also positive definite (despite the case when $\omega = 0$ which leads to a positive semidefinite matrix). Finally we use that the sum of two symmetric and positive definite matrices is symmetric and positive definite. Thus the matrix can be decomposed as a $L D L^T$ product by means of Cholesky (see algorithm 3) and consequently nonsingular matrices $L_{2,3}$ and $D_{2,2}$ can be found. These matrices are clearly dependent of $\omega$. This concludes the argument. QED

It is important to notice that all the matrices, excluding $L_{2,2}$ and $D_{2,2}$, are either independent or linearly dependent of $\omega$. This plays an important role because this matrices need to be calculated only once.
4.4 Final Factorization

The previous results can be summarized in the following theorem.

**Theorem 8.** Let $A$ be the matrix associated with an RL circuit (see 4.8). Then there exist an $n \times n$ permutation matrix $\Pi_1$ and an $m \times m$ permutation matrix $\Pi_2$ such that $\Pi_1 B \Pi_2^T$ is lower triangular. Moreover there also exist, an $2(n - m) \times 2m$ matrix $M$, an $2(n - m) \times 2m$ matrix $F$, an $2m \times 2m$ diagonal matrix $D_1$, an $2m \times 2m$ strictly lower triangular matrix $L_1$, an $2m \times 2m$ strictly upper triangular matrix $U_1$, and $2(n - m) \times 2(n - m)$ invertible matrices $D_2$, $L_2$ and $U_2$, with $D_1$ diagonal, $L_2$ lower triangular and $U_2$ upper triangular, such that:

$$A = \Pi \tilde{L} \tilde{D} \tilde{U} \Pi^T,$$

where

$$\Pi = \begin{pmatrix}
0 & 0 & \Pi_{1,1}^T & \Pi_{1,2}^T & 0 \\
0 & 0 & \Pi_{2,1}^T & 0 & \Pi_{1,1}^T \\
\Pi_{2,2}^T & 0 & 0 & 0 & 0 \\
0 & \Pi_{2,2}^T & 0 & 0 & 0
\end{pmatrix},$$

$$\tilde{L} = \begin{pmatrix}
I_{2m} & 0 & 0 \\
L_1 & B_1 & 0 \\
M & B_2 & L_2
\end{pmatrix}, \quad \tilde{U} = \begin{pmatrix}
I_{2m} & U_1 & F \\
0 & B_1^T & B_2^T \\
0 & 0 & U_2
\end{pmatrix},$$

$$\tilde{D} = \begin{pmatrix}
I_{2m} & 0 \\
0 & D_1 & 0 \\
0 & 0 & D_2
\end{pmatrix},$$

where $B_1$ is the top $2m \times m$ part of $\Pi_1 B \Pi_2^T$, and $B_2$ is the lower $2(n - m) \times m$ part. $\Pi_{1,1}$ is the matrix formed by the first $m$ columns of $\Pi_1$, $\Pi_{1,2}$ is the matrix formed by the last $n - m$ columns of $\Pi_1$.

**Proof:** To find the matrices $\Pi_1$ and $\Pi_2$ we use theorem 6. Then we arrange the matrix as in (4.14), and apply the decomposition given in theorem 7. Let $\Pi_3$ and $Q$ be as in (5.24) and (4.9), respectively. Then let $\Pi_4$ be the following permutation matrix:

$$\Pi_4 = \begin{pmatrix}
0 & 0 & I_{2m} \\
I_{2m} & 0 & 0 \\
0 & I_{2(n-m)} & 0
\end{pmatrix}.$$  \hfill (4.39)

Using this permutation we observe that:

$$Q^T \Pi_2^T \Pi_4^T = \begin{pmatrix}
0 & 0 & \Pi_{1,1}^T & \Pi_{1,2}^T & 0 \\
0 & 0 & \Pi_{2,1}^T & 0 & \Pi_{1,1}^T \\
\Pi_{2,2}^T & 0 & 0 & 0 & 0 \\
0 & \Pi_{2,2}^T & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}.$$  \hfill (4.40)
4.4 Final Factorization

Factorization of RL Circuits

Then we observe that:

\[
\begin{pmatrix}
0 & 0 & I_{2m} \\
I_{2m} & 0 & 0 \\
0 & I_{2(n-m)} & 0
\end{pmatrix}
\begin{pmatrix}
B_1 & 0 & L_1 \\
B_2 & L_2 & M \\
0 & 0 & I_{2m}
\end{pmatrix}
\begin{pmatrix}
0 & I_{2m} & 0 \\
0 & 0 & I_{2(n-m)} \\
I_{2m} & 0 & 0
\end{pmatrix}
= \begin{pmatrix}
I_{2m} & 0 & 0 \\
0 & L_1 & B_1 \\
M & B_2 & L_2
\end{pmatrix}
\]

and

\[
\Pi_4 \begin{pmatrix} B_1^T & B_2^T & 0 \\ 0 & U_2 & 0 \\ U_1 & F & I_{2m} \end{pmatrix} \Pi_4^T = \begin{pmatrix} I_{2m} & U_1 & F \\ 0 & B_1^T & B_2^T \\ 0 & 0 & U_2 \end{pmatrix},
\]

(4.41)

The final step is to realize that:

\[
\Pi_4 \begin{pmatrix} D_1 & 0 & I_{2m} \\ 0 & D_2 & 0 \\ I_{2m} & 0 & 0 \end{pmatrix} \Pi_4^T = \begin{pmatrix} 0 & I_{2m} & 0 \\ I_{2m} & D_1 & 0 \\ 0 & 0 & D_2 \end{pmatrix},
\]

(4.43)

and this concludes the proof. QED

An important remark can be done at this point. The previous theorem does not assume that the original matrix \( A \) is invertible. Thus theorem 8, provides us not only with a factorization of the matrix but it also tells us that the matrix \( A \) is invertible. This follows after realizing that,

\[
\begin{pmatrix}
0 & I_{2m} & 0 \\
I_{2m} & D_1 & 0 \\
0 & 0 & D_2
\end{pmatrix}^{-1} = \begin{pmatrix}
-D_1 & I_{2m} & 0 \\
I_{2m} & 0 & 0 \\
0 & 0 & D_2^{-1}
\end{pmatrix},
\]

(4.44)

and since \( D_2 \) is invertible it follows that \( A \) is also invertible.

This means that a solution to the Alternating Current Analysis of an RL circuit always exist and moreover this solution is unique. In the next chapter we will first have to prove the invertibility of the system matrix and then proceed to develop the factorization.
Chapter 5

Factorization RLC Circuits

In this chapter we develop a factorization for the general case of circuits, i.e. including capacitive effects. In contrast with the RL case, for the RLC case we first have to prove that the system associated with circuit is nonsingular, this result will be used later to develop the factorization. As a second step we rewrite the circuit equations in a new form. This new formulation involves more variables than the original one, nonetheless we are able to factorize the system of equation written in this new form.

5.1 Invertibility of the RLC system

Let us recall the equation for the RLC circuit from (2.19). The system has the form:

\[
\begin{pmatrix}
\hat{R} & -\omega \hat{L} & \hat{P} & 0 \\
\omega \hat{L} & \hat{R} & 0 & \hat{P} \\
\hat{P}^T & 0 & -\hat{G} & \omega \hat{C} \\
0 & \hat{P}^T & -\omega \hat{C} & -\hat{G}
\end{pmatrix}
\begin{pmatrix}
i_r \\
i_i \\
v_r \\
v_i
\end{pmatrix}
= \begin{pmatrix}0 \\ 0 \\ A^T \\ 0\end{pmatrix} \Theta
\] (5.1)

This system can be rewritten as:

\[
A = \begin{pmatrix}A & B \\ -B^T & C\end{pmatrix},
\] (5.2)

where

\[
A = \begin{pmatrix}\hat{R} & -\omega \hat{L} \\ \omega \hat{L} & \hat{R}\end{pmatrix},
B = \begin{pmatrix}\hat{P} & 0 \\ 0 & \hat{P}\end{pmatrix}
\] (5.3)

and

\[
C = \begin{pmatrix}\hat{G} & -\omega \hat{C} \\ \omega \hat{C} & \hat{G}\end{pmatrix}.
\] (5.4)

Where \(\hat{R}\) is a positive definite diagonal matrix, \(\hat{P}\) is of full column rank. Then we can make the following observations. First it is easy to realize that the symmetric part of \(A\) \((H = \frac{1}{2}(A + A^T))\) is:

\[
H = \begin{pmatrix}\hat{R} & 0 \\ 0 & \hat{R}\end{pmatrix}.
\] (5.5)
5.1 Invertibility of the RLC system

Factorization RLC Circuits

thus \( H \) is positive definite and consequently also invertible. A second observation is that, since \( \hat{P} \) has full column rank then also \( B \) has full column rank. Additionally we can check that \( C \) is positive semidefinite. With these properties we can prove that the matrix \( A \) is invertible. We do it by means of the next lemma. The proof is based in a more general result by Benzi and Golub [6].

**Lemma 3.** Let \( A \) be a matrix with the structure given in (5.2). Now let \( H = \frac{1}{2}(A + A^T) \) be positive definite, \( D = \frac{1}{2}(C + C^T) \) be positive semidefinite and \( B \) be of full rank. Then \( A \) is nonsingular.

**Proof:** Suppose that \( Av = 0 \) with \( v^T = (x^T y^T) \). Then we have \( e^T Av = 0 \), thus by performing the following calculations:

\[
0 = v^T Av = x^T Ax + x^T By^T - y^T B^T x + y^T Cy = x^T H x + y^T D y, \tag{5.6}
\]

since \( H \) is positive definite and \( D \) is positive semidefinite, we can conclude that \( x^T H x = 0 \) and \( y^T D y = 0 \). Now since \( H \) is positive definite then \( x = 0 \). Now using the first block equation of \( Av = 0 \) we have:

\[
Ax + By = 0, \tag{5.7}
\]

since \( x = 0 \) then we have \( By = 0 \), and finally using that \( B \) has full column rank we conclude that \( y = 0 \) as well. This means \( v = 0 \), which implies that \( A \) is invertible. QED

Now we will proceed to rewrite the system in a different form, the system in this new form will be later factorized. First let us recall that our system consist of four kinds of elements. Namely resistors (pure resistors), inductors(in series with a resistor), capacitors and current sources. We will further assume the input frequency is always strictly positive (i.e. \( \omega > 0 \)). This is acceptable because when \( \omega = 0 \) the system has a rather simple form, namely:

\[
\begin{pmatrix}
\hat{R} & 0 & \hat{P} & 0 \\
0 & \hat{R} & 0 & \hat{P} \\
\hat{P}^T & 0 & -\hat{G} & 0 \\
0 & \hat{P}^T & 0 & -\hat{G}
\end{pmatrix}
\begin{pmatrix}
i_r \\
i_i \\
v_r \\
v_i
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
A_i^T \\
0
\end{pmatrix}
\Theta. \tag{5.8}
\]

This system can be easily factorized, to this purpose it is enough to observe that:

\[
\begin{pmatrix}
\hat{R} & \hat{P} \\
\hat{P}^T & -\hat{G}
\end{pmatrix}
= \begin{pmatrix}
I & 0 \\
\hat{P}^T \hat{R}^{-1} & I
\end{pmatrix}
\begin{pmatrix}
\hat{R} & 0 \\
0 & -(\hat{G} + \hat{P}^T \hat{R}^{-1} \hat{P})
\end{pmatrix}
\begin{pmatrix}
I & -\hat{R}^{-1} \hat{P} \\
0 & I
\end{pmatrix} \tag{5.9}
\]

The factorization can be completed after observing that \( \hat{G} + \hat{P}^T \hat{R}^{-1} \hat{P} \) is positive definite, then we can apply Cholesky to decompose it as an \( LDL^T \) product. This provides us with the factorization. It is important to notice that \( \hat{R}^{-1} \) is diagonal and that the Schur complement \( \hat{G} + \hat{P}^T \hat{R}^{-1} \hat{P} \) remains sparse.
5.2 Circuit Equations (Revisited)

In the previous section we saw that we can assume \( \omega > 0 \). With this assumption we will get an equivalent formulation of the circuit equations. As it was done before we consider the involved matrices arranged as follows.

\[
A = \begin{bmatrix}
A_i \\
A_g \\
A_c \\
A_l
\end{bmatrix},
\quad v_b = \begin{bmatrix}
v_i \\
v_g \\
v_c \\
v_l
\end{bmatrix},
\quad i_b = \begin{bmatrix}
i_i \\
i_g \\
i_c \\
i_l
\end{bmatrix},
\quad (5.10)
\]

where the subscripts \( i, g, c \) and \( l \) denote current source, resistor, capacitor and inductor (in series with a resistor) respectively. With this decomposition Kirchhoff currents law (2.4) appears in the following way.

\[
A_T i + A_T g i + A_T c i + A_T l i = 0,
\quad (5.11)
\]

Kirchhoff’s voltage law (2.5) simply says

\[
A_i v_n = v_i,
\quad A_g v_n = v_g,
\quad A_c v_n = v_c,
\quad A_l v_n = v_l.
\quad (5.12)
\]

We consider the following branch constitutive relations:

\[
i_i = I_t(t),
\quad i_g = G v_g,
\quad i_c = C \frac{d}{dt} v_c,
\quad v_l = (L \frac{d}{dt} + R) i_l,
\quad (5.13)
\]

where \( I_t(t) \) denotes the vector of current-source values, \( G \) and \( C \) denote the conductances (of the pure resistors) and capacitances and hence are diagonal matrices. The matrix \( R \) denote the resistances that are in series with the inductors hence it is also diagonal. The matrix \( L \) denotes the inductances and it is diagonal in the absence of inductive coupling. Inductive coupling adds off-diagonal terms but the matrix remains symmetric and positive definite.

Now we will write the system using \( i_g, i_c, i_l \) and \( v_n \). Thus we get the following equations:

\[
G^{-1} i_g - A_g v_n = 0 \quad (5.14a)
\]

\[
C^{-1} i_c - A_c \frac{d}{dt} v_n = 0 \quad (5.14b)
\]

\[
(L \frac{d}{dt} + R) i_l - A_l v_n = 0 \quad (5.14c)
\]

\[
A_T^T i_g + A_T^T i_c + A_T^T i_l = -A_T^T I_t(t) \quad (5.14d)
\]

We can rewrite the equations in system form (note that Kirchhoff equations was multiplied by minus one):

\[
\begin{pmatrix}
G^{-1} & -A_g \\
C^{-1} & -A_c \\
-L \frac{d}{dt} & -A_l
\end{pmatrix}
\begin{pmatrix}
i_g \\
i_c \\
v_n
\end{pmatrix}
+ \begin{pmatrix}
0 & 0 & -A_c \\
0 & L & 0
\end{pmatrix}
\begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix}
I_t(t)
\quad (5.15)
\]

Then using the complex phasor analysis the system takes the following form:
5.2 Circuit Equations (Revisited)  
Factorization RLC Circuits

\[
\begin{pmatrix}
G^{-1} & \mathcal{C}^{-1} & -A_g \\
-A_g^T & -A_c^T & -A_l^T & 0 \\
\end{pmatrix} + i\omega
\begin{pmatrix}
0 & 0 & -A_c \\
0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
i_{gr} + i_{gi} \\
i_{cr} + i_{ci} \\
0 \\
i_{nr} + i_{ni} \\
0 \\
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
\end{pmatrix}
\Theta
\]

(5.16)

This system can be rewritten as a real system:

\[
\begin{pmatrix}
G^{-1} & \mathcal{C}^{-1} & -A_g \\
-A_g^T & -A_c^T & -A_l^T & 0 \\
\end{pmatrix} + i\omega
\begin{pmatrix}
0 & 0 & -A_c \\
0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
i_{gr} + i_{gi} \\
i_{cr} + i_{ci} \\
0 \\
i_{nr} + i_{ni} \\
0 \\
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
\end{pmatrix}
\Theta
\]

(5.17)

We multiply by minus one the third blocks row. And since we assume \( \omega > 0 \) then we can divide by \( \omega \) in the third and fourth block rows. We also permute these rows and get the following:

\[
\begin{pmatrix}
G^{-1} & \mathcal{C}^{-1} & -A_g \\
-A_g^T & -A_c^T & -A_l^T & 0 \\
\end{pmatrix} + \omega \mathcal{C}^{-1}
\begin{pmatrix}
0 & 0 & -A_c \\
0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
i_{gr} + i_{gi} \\
i_{cr} + i_{ci} \\
0 \\
i_{nr} + i_{ni} \\
0 \\
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
\end{pmatrix}
\Theta
\]

(5.18)

Now we further permute rows and get:

\[
\begin{pmatrix}
\mathcal{C}^{-1} & \mathcal{C}^{-1} & -A_g \\
-A_g^T & -A_c^T & -A_l^T & 0 \\
\end{pmatrix} + \mathcal{C}^{-1}
\begin{pmatrix}
0 & 0 & -A_c \\
0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
i_{gr} + i_{gi} \\
i_{cr} + i_{ci} \\
0 \\
i_{nr} + i_{ni} \\
0 \\
\end{pmatrix}
= \begin{pmatrix}
0 \\
0 \\
\end{pmatrix}
\Theta
\]

(5.19)

Finally permuting columns we get:
5.3 RLC Factorization

In the previous section we reformulated the RLC circuit equations. Let \( m+1 \) be the number of nodes of the circuit and \( n \) be the number of components. Then matrix of the system is of the form:

\[
A = \begin{pmatrix}
\tilde{X} & \tilde{Y} & \hat{P} \\
-\hat{Y} & \tilde{X} & 0 \\
0 & 0 & 0
\end{pmatrix},
\]

where

\[
\tilde{X} = \begin{pmatrix}
G^{-1} & 0 \\
0 & \mathcal{R}
\end{pmatrix}, \quad \text{and} \quad \tilde{Y} = \begin{pmatrix}
0 \\
\frac{1}{2} C^{-1} \\
-\omega L
\end{pmatrix}
\]

are \( n \times n \) matrices. The incidence matrix \( \hat{P} \) is of size \( n \times m \) and is given by:

\[
\hat{P} = \begin{pmatrix}
-A_g \\
-A_c \\
-A_l
\end{pmatrix}.
\]

Since \( \hat{P} \) is an incidence matrix then we can again apply algorithm 4 and find an \( n \times n \) permutation matrix \( \Pi_1 \) and an \( m \times m \) permutation matrix \( \Pi_2 \) such that \( \Pi_1 \hat{P} = P \Pi_2 \). Here \( P \) is an incidence matrix in lower trapezoidal form, i.e. \( P^T = (P_1^T P_2^T) \), where \( P_1 \) is lower triangular and invertible. Thus by defining \( Q \) and \( \Pi_3 \) as follows,

\[
Q = \begin{pmatrix}
\Pi_1 \\
\Pi_1 \\
\Pi_2
\end{pmatrix}, \quad \Pi_3 = \begin{pmatrix}
I_m & I_m \\
I_{n-m} & I_{n-m} \\
I_{2m}
\end{pmatrix}.
\]

We find that:
The matrix associated with the RLC circuit (5.27), can be decomposed strictly upper triangular. The matrix
where

\[ M = \begin{pmatrix}
X_{11} & Y_{11} & X_{12} & Y_{12} & P_1 & 0 \\
-1 & X_{11} & -Y_{11} & X_{12} & 0 & P_1 \\
X_{21} & Y_{21} & X_{22} & Y_{22} & P_2 & 0 \\
-1 & X_{21} & -Y_{21} & X_{22} & 0 & P_2 \\
P_1^T & 0 & P_2^T & 0 & 0 & 0 \\
0 & P_1^T & 0 & P_2^T & 0 & 0 \\
\end{pmatrix}, \quad (5.25)
\]

where the matrices \( X \) and \( Y \) are defined and organized in blocks as follows. Observe that \( X_{11} \) and \( Y_{11} \) are of size \( m \times m \), and that \( X_{22} \) and \( Y_{22} \) are of size \((n-m) \times (n-m)\).

\[ \Pi_1 \hat{X} \Pi_1^T =: X = \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix}, \quad \Pi_1 \hat{Y} \Pi_1^T =: Y = \begin{pmatrix} Y_{11} & Y_{12} \\ Y_{21} & Y_{22} \end{pmatrix}. \quad (5.26) \]

In order to perform the decomposition, we will arrange the matrix in the same way in which it was done in the previous chapter.

\[ \Pi_3 Q A Q^T \Pi_3^T = \begin{pmatrix} A_{11} & A_{12} & B_1 \\
A_{21} & A_{22} & B_2 \\
B_1^T & B_2^T & 0 \end{pmatrix}, \quad \text{where } A_{ij} = \begin{pmatrix} X_{ij} \\ Y_{ij} \end{pmatrix}. \quad (5.27) \]

Now we are ready to prove the following theorem that will lead us to the final factorization.

**Theorem 9.** The matrix associated with the RLC circuit (5.27), can be decomposed as follows:

\[ \begin{pmatrix}
A_{11} & A_{12} & B_1 \\
A_{21} & A_{22} & B_2 \\
B_1^T & B_2^T & 0 \\
\end{pmatrix} = \begin{pmatrix} B_1 & 0 & L_1 \\
B_2 & I_{2(n-m)} & M \\
0 & \hat{W} & 0 \\
\end{pmatrix} \begin{pmatrix} D_1 & 0 & I_{2m} \\
0 & \hat{W} & 0 \\
I_{2m} & 0 & 0 \\
\end{pmatrix} \begin{pmatrix} B_1^T & B_2^T & 0 \\
0 & I_{2(n-m)} & 0 \\
U_1 & F & I_{2m} \\
\end{pmatrix} \quad (5.28) \]

where \( M \) is an \( 2(n-m) \times 2m \) matrix, \( F \) is an \( 2(n-m) \times 2m \) matrix. \( D_1 \) is \( 2m \times 2m \) diagonal, \( L_1 \) is \( 2m \times 2m \) strictly lower triangular and \( U_1 \) is \( 2m \times 2m \) strictly upper triangular. The matrix \( \hat{W} \) is of size \( 2(n-m) \times 2(n-m) \), and moreover since the original system is invertible then \( \hat{W} \) is also invertible.

**Proof:** Just as it was done in the proof of theorem 7, we realize that the statement is true if the next equations are satisfied.

\[ B_1 D_1 B_1^T + B_1 U_1 + L_1 B_1^T = A_{11} \quad (5.29) \]
\[ B_1 D_1 B_2^T + B_1 F + L_1 B_2^T = A_{12} \quad (5.30) \]
\[ B_2 D_1 B_1^T + B_2 U_1 + M B_1^T = A_{21} \quad (5.31) \]
\[ \hat{W} + B_2 D_1 B_2^T + B_2 F + M B_2^T = A_{22} \quad (5.32) \]

again multiplying (5.29) by \( B_1^{-1} \) from the left and by \( B_1^{-T} \) from the right we find expressions for \( L_1, U_1 \) and \( D_1 \):

\[ L_1 = B_1 \text{strlow} (B_1^{-1} A_{11} B_1^{-T}), \quad U_1 = \text{strupp} (B_1^{-1} A_{11} B_1^{-T}) B_1^T, \quad (5.33) \]
\[ D_1 = \text{diag} (B_1^{-1} A_{11} B_1^{-T}). \quad (5.34) \]
Using (5.30) and (5.31) we find expressions for \( M \) and \( F \),
\[
F = B_1^{-1} \left( A_{12} - B_1 D_1 B_2^T - L_1 B_2^T \right), \quad M = (A_{21} - B_2 D_1 B_1^T - B_2 U_1) B_1^{-T}.
\]

Finally from (5.32) we get:
\[
\hat{W} = A_{22} - B_2 D_1 B_2^T - B_2 F - MB_1^T.
\]

This concludes the decomposition. Now, since the original matrix \( A \) is invertible it follows that the product of the three matrices is invertible. After some permutations, one can check that the first and the last matrices in the product are invertible. Thus the matrix in the middle is invertible. This matrix is invertible if and only if \( \hat{W} \) is also invertible, because:
\[
\begin{pmatrix}
D_1 & 0 & I_{2m} \\
0 & \hat{W} & 0 \\
I_{2m} & 0 & 0
\end{pmatrix}^{-1} =
\begin{pmatrix}
0 & 0 & I_{2m} \\
0 & \hat{W}^{-1} & 0 \\
I_{2m} & 0 & -D_1
\end{pmatrix}.
\]

this finishes the argument. QED

This provides us with all the ingredients to find factorization for the RLC circuit equations. The only step left is to use an \( LDU \) decomposition of the matrix \( \hat{W} \). This can be done because \( \hat{U} \) is nonsingular (see theorem 3). Thus there exist matrices of size \( 2(n-m) \times 2(n-m) \), \( L_2, D_2, U_2 \) and \( \Pi_e \) such that:
\[
\Pi_e L_2 D_2 U_2 = \hat{W}
\]

where \( L_1 \) is lower triangular, \( D_2 \) is diagonal, \( U_2 \) is upper triangular and \( \Pi_e \) is permutation matrix.

Using this \( LDU \) decomposition and theorem 9 we get the final decomposition for the RLC circuit equations. This is stated in the following theorem.

**Theorem 10.** Let \( A \) be the matrix associated with an RLC circuit as it appears in (5.20). Then there exist permutation matrices \( Q, \Pi_3, \Pi_E, \) and invertible matrices \( \tilde{L}, \tilde{D}, \tilde{U} \), with \( \tilde{L} \) lower triangular, \( \tilde{D} \) "almost" diagonal, and \( \tilde{U} \) upper triangular. Such that:

\[
A = Q^T \Pi_1^T \Pi_2^T \Pi_3^T \tilde{L} \tilde{D} \tilde{U} \Pi_4 \Pi_3 Q
\]

where:

\[
Q = \begin{pmatrix}
\Pi_1 & \\
\Pi_1 & \\
\Pi_2 & \\
\Pi_2 & \\
\end{pmatrix}, \quad \Pi_3 = \begin{pmatrix}
I_m & \\
I_{n-m} & \\
& I_{n-m} & \\
& I_{2m} & \\
\end{pmatrix},
\]

\[
\Pi_E = \begin{pmatrix}
I_{2m} & \\
\Pi_e & \\
I_{2m} & \\
\end{pmatrix}, \quad \Pi_4 = \begin{pmatrix}
0 & 0 & I_{2m} \\
I_{2m} & 0 & 0 \\
0 & I_{2(n-m)} & 0
\end{pmatrix},
\]

\[
\tilde{D} = \begin{pmatrix}
0 & I_{2m} & 0 \\
I_{2m} & \tilde{D}_1 & 0 \\
0 & 0 & D_2
\end{pmatrix}
\]
5.3 RLC Factorization

\[
\hat{L} = \begin{pmatrix}
I_{2m} & 0 & 0 \\
L_1 & B_1 & 0 \\
\Pi_4^T M & \Pi_4^T B_2 & L_2
\end{pmatrix}, \quad \hat{U} = \begin{pmatrix}
I_{2m} & U_1 & F \\
0 & B_1^T & B_2^T \\
0 & 0 & U_2
\end{pmatrix}
\]

**Proof:** The proof is analogous to the one in theorem 8. Here it is only important to take care with the next calculations:

\[
\Pi_4^T \begin{pmatrix}
B_1 & 0 & L_1 \\
B_2 & \Pi_4 L_2 & M \\
0 & 0 & I_{2m}
\end{pmatrix} = \begin{pmatrix}
B_1 & 0 & L_1 \\
\Pi_4^T B_2 & L_2 & \Pi_4^T M \\
0 & 0 & I_{2m}
\end{pmatrix}, \quad (5.39)
\]

and

\[
\Pi_4 \begin{pmatrix}
B_1 & 0 & L_1 \\
\Pi_4^T B_2 & L_2 & \Pi_4^T M \\
0 & 0 & I_{2m}
\end{pmatrix} \Pi_4^T = \hat{L}. \quad (5.40)
\]

QED
Chapter 6

Time Complexity

In this chapter we analyze how the RL and the RLC factorizations work. First we show some illustrative examples for each of the cases. Then we make a derivation of the time complexity of both factorizations. Finally we provide some examples with circuits of arbitrary size which allows us to compare the time complexity quantitatively. The execution time of the solving using the factorizations is compared to the one obtained when solving by means of LU decomposition.

6.1 Small RL example

In order to show how the RL factorization works we will consider the circuit in Figure 6.1. Here we have \( m + 1 = 3 \), the number of nodes, and \( n_l = 3 \), the number of inductors, which is also the number of total branches because in this case there are no current sources, because this is only an example to show how the factorization works.

![Simple RL Circuit](image)

Then constructing the incidence matrix we have:

\[
A_l = \begin{pmatrix} -1 & 1 \\ 0 & -1 \\ 1 & 0 \end{pmatrix} \quad \text{Thus} \quad \hat{P} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \\ -1 & 0 \end{pmatrix}.
\] (6.1)
6.1 Small RL example

Time Complexity

Furthermore let:

\[ R = \begin{pmatrix} 5 & 6 \\ 7 \end{pmatrix}, \quad L = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 4 \end{pmatrix}. \]  

(6.2)

In this case we consider inductors with coupling, that is why the matrix \( L \) has nonzero entries in the off-diagonal. Now we proceed to construct the RL factorization of the matrix as it was done in chapter 4, we will consider \( \omega = 1 \).

First we need to construct the permutations matrices \( \Pi_1 \) and \( \Pi_2 \), by applying algorithm 4 we find:

\[ \Pi_1 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \quad \Pi_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{pmatrix} \]

and define

\[ Q = \begin{pmatrix} \Pi_1 & \Pi_2 \\ \Pi_2 & \Pi_1 \end{pmatrix}. \]  

(6.3)

Then we have:

\[ \Pi_1 \hat{R} \Pi_1^T =: R = \begin{pmatrix} 7 & 6 \\ 5 \end{pmatrix}, \quad \Pi_1 \hat{L} \Pi_1^T =: L = \begin{pmatrix} 4 & 1 & 1 \\ 1 & 3 & 1 \\ 1 & 1 & 2 \end{pmatrix} \]  

(6.4)

and

\[ \Pi_1 \hat{P} \Pi_2 = \begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 1 & -1 \end{pmatrix}, \quad \text{and identify } P_1 = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \quad P_2 = \begin{pmatrix} 1 & -1 \end{pmatrix}. \]  

(6.5)

Furthermore we first compute \( P_1^{-1} \) and by defining (remember that \( m = 2 \) and \( n_l = 3 \)):

\[ P_1^{-1} = \begin{pmatrix} -1 & 1 \\ 1 & 1 \end{pmatrix}, \quad \Pi_3 = \begin{pmatrix} I_2 & I_2 \\ I_1 & I_2 \\ I_1 & I_2 \end{pmatrix}. \]  

(6.6)

And we find

\[ \Pi_3 Q A Q \Pi_3^T = \begin{pmatrix} 7 & 0 & -4 & -1 & 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 6 & -1 & -3 & 0 & -1 & 0 & 1 & 0 & 0 \\ 4 & 1 & 7 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 1 & 3 & 0 & 6 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & -1 & 5 & -2 & 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 2 & 5 & 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \end{pmatrix}. \]  

(6.7)
6.1 Small RL example

We recognize the structures (see theorem 7):

\[
A_{11} = \begin{pmatrix}
7 & 0 & -4 & -1 \\
0 & 6 & -1 & -3 \\
4 & 1 & 7 & 0 \\
1 & 3 & 0 & 6
\end{pmatrix},
A_{12} = \begin{pmatrix}
0 & -1 \\
0 & -1 \\
1 & 0 \\
1 & 0
\end{pmatrix},
B_1 = \begin{pmatrix}
-1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

\[
A_{21} = \begin{pmatrix}
0 & 0 & -1 & -1 \\
1 & 1 & 0 & 0
\end{pmatrix},
A_{22} = \begin{pmatrix}
5 & 2 \\
2 & 5
\end{pmatrix},
B_2 = \begin{pmatrix}
1 & -1 & 0 & 0 \\
0 & 0 & 1 & -1
\end{pmatrix}
\]  

(6.8)

Then we can calculate:

\[
B_1^{-1}A_{11}B_1^{-T} = \begin{pmatrix}
7 & 0 & -4 & 1 \\
0 & 6 & 1 & -3 \\
4 & -1 & 7 & 0 \\
-1 & 3 & 0 & 6
\end{pmatrix},
\]  

(6.9)

from here we conclude (see again theorem 7),

\[
D_1 = \begin{pmatrix}
7 & 0 & 0 & 0 \\
0 & 6 & 0 & 0 \\
0 & 0 & 7 & 0 \\
0 & 0 & 0 & 6
\end{pmatrix},
L_1 = \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
-4 & 1 & 0 & 0 \\
-1 & 3 & 0 & 0
\end{pmatrix},
U_1 = \begin{pmatrix}
0 & 0 & 4 & 1 \\
0 & 0 & -1 & -3 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{pmatrix}
\]  

(6.10)

With the information above we find:

\[
M = \begin{pmatrix}
-7 & 6 & 6 & -5 \\
1 & -7 & 6 & 0
\end{pmatrix},
F = \begin{pmatrix}
-7 & 1 \\
6 & -1 \\
-6 & -7 \\
5 & 6
\end{pmatrix}
\]  

(6.11)

Finally we find that:

\[
\hat{W} = A_{22} - B_2D_1B_2^T - B_2F - MB_2^T = (18 - 151518)
\]

(6.12)

Now we need to decompose this matrix as stated in lemma 2. Thus we realize that:

\[
L_{2,1} = 1 \quad \text{and} \quad D_{2,1} = 18
\]

(6.13)

hence we can compute

\[
L_{2,2} = W_{21}L_{2,1}^{-T}D_{2,1}^{-1} = 15(1)\frac{1}{18} = \frac{15}{18}
\]

(6.14)

and it is only left to find

\[
L_{2,3}D_{2,2}L_{2,3}^T = \hat{W}_{11} + \omega^2\hat{W}_{21}\hat{W}_{11}^{-1}\hat{W}_{21} = 18 + (-15)(\frac{1}{18})(-15) = \frac{549}{18}
\]

(6.15)

Thus we have,

\[
L_2 = \begin{pmatrix}
\frac{1}{15} & 0 \\
\frac{15}{18} & 1
\end{pmatrix},
D_2 = \begin{pmatrix}
18 & \frac{549}{18} \\
0 & \frac{15}{18}
\end{pmatrix},
U_2 = \begin{pmatrix}
1 & -\frac{15}{18} \\
0 & 1
\end{pmatrix}
\]

(6.16)

Despite some extra permutations, this completes the factorization.
6.2 RL Complexity

Now we will analyze the time complexity of solving the circuit equations with the RL factorization. For this purpose let $m + 1$ be the number of nodes in the circuit, $n_l$ the number of inductors (in series with a resistor) and $n_t$ the number of current sources. Then the circuit equations have the form:

$$A = \begin{pmatrix}
\hat{R}_t & -\omega \hat{L} & 0 & 0 \\
\omega \hat{L} & \hat{R} & 0 & \hat{P} \\
\hat{P}^T & 0 & 0 & 0 \\
0 & \hat{P}^T & 0 & 0
\end{pmatrix}
\begin{pmatrix}
i_r \\
i_i \\
v_r \\
v_i
\end{pmatrix} =
\begin{pmatrix}0 \\
0 \\
A_t^T \\
0
\end{pmatrix}
(6.17)$$

The size of the matrix $A$, is then $2n_l + 2m$, the number of current sources does not play a role in the size of the system actively. Then we can conclude the complexity of solving the equations via the LU decomposition is:

$$O \left( (2(n_l + m))^3 + (2(n_l + m))^2 \right)$$

(6.18)

The cubic term comes from computing the LU decomposition the quadratic term comes from the solving the respective upper triangular and lower triangular systems.

Now we analyze the complexity of solving the circuit equations using the RL factorization. In this case the size of the system equation remains $2(n_l + m)$. The algorithm to solve the equations using the RL factorization is conformed roughly as follows.

**Algorithm 5 RL Solution Algorithm**

**INPUT:** circuit equations matrices: $A_l, A_i, R, L$ and a frequency $\omega$

**OUTPUT:** solution vector $x$ for the frequency $\omega$

1. Determine permutation matrices $\Pi_1$ and $\Pi_2$, transforming $\hat{P}$ into lower trapezoidal form (see algorithm 4)
2. Transform the system matrix $A$ with $Q$ leading to $R, L$ and $P$
3. Arrange the system matrix in blocks $A_{ij}$ (see (4.14))
4. Determine $P^{-1}$ and thus find the matrices $D_1, L_1$ and $U_1$ (see theorem 7)
5. Perform the Cholesky decomposition on $\hat{W}_{11}$ and $\hat{W}_{11} + \omega^2 \hat{W}_{21} \hat{W}_{11}^{-1} \hat{W}_{21}$, leading to the matrices $D_2, L_2$ and $U_2$ (see lemma 2)
6. Apply the correspondent permutations, solve with backward substitution the upper triangular system $\tilde{U}$, solve with forward substitution the lower triangular system $\tilde{L}$ and apply $\tilde{D}^{-1}$ (see theorem 8)

The first step of the algorithm is performed by using algorithm 4. Thus we incur in $n_l^2$ operations. Multiplying by the permutation matrix $Q$ as well as the arrangement of the system matrix, are basically read and write operations. Performing Cholesky decomposition needs $\frac{1}{3} (n_l - m)^3$ operations, in our case we need to apply this algorithm twice. Solving the resulting upper and lower triangular systems requires $O((2n_l + m)^2)$ operations.

So far we have discussed most of the steps in the algorithm, now we go a bit deeper into step number 4. This step seems to be rather simple but it deserves more discussion. The easy part is the computation of $P^{-1}$, this involves $O(m^2)$
6.2 RL Complexity

Time Complexity

operations, because \( P_1 \) is lower triangular and sparse (it contains at most 2 nonzero entries per row). The most delicate issue here, is the computation of the product \( P_1^{-1}A_{11}P_1^{-T} \). The computation of \( P_1^{-1}L_{11}P_1^{-T} \) acts in the same way. The matrix \( P_1 \) was set to be lower triangular and very sparse, hence the inverse is also lower triangular, but it might be full though. This might happen, for example, when

\[
P_1 = \begin{pmatrix}
  1 & 1 & & & \\
  -1 & 1 & & & \\
  & & \ddots & & \\
  & & & -1 & 1
\end{pmatrix}.
\]  

(6.19)

In this special case we have that the inverse is full, in fact:

\[
P_1^{-1} = \begin{pmatrix}
  1 & 1 & & & \\
  1 & 1 & & & \\
  \vdots & \vdots & \ddots & & \\
  1 & 1 & 1 & & \\
\end{pmatrix}.
\]  

(6.20)

From this we can conclude that performing the products \( P_1^{-1}R_{11}P_1^{-T} \) and \( P_1^{-1}L_{11}P_1^{-T} \), depends on the properties of \( P_1 \), in bad cases it might take up to the order of \( O(m^3) \) operations. In some other cases it might happen that we require no operations, for instance if \( P_1 = I_m \). Adding altogether and omitting factors, we can conclude that the worst case complexity of the algorithm is:

\[
O \left( n_l^2 + (n_l - m)^3 + m^3 + (2(n_l + m))^2 \right)
\]  

(6.21)

In the previous expression some factors were left in order to identify the problem from which they come from. For instance, the term \( m^3 \) comes from the problem of calculating the product associated with \( P_1^{-1} \). Provided that \( P_1^{-1} \) remains sparse, the complexity of solving by using the RL decomposition,

\[
O \left( n_l^2 + (n_l - m)^3 + (2(n_l + m))^2 \right)
\]  

(6.22)

the other terms can not be avoided. From this, we can conclude that the RL solution algorithm performs the best when \( P_1^{-1} \) remains sparse and \( n_l - m \) is small. In any case we can conclude that, for systems big enough, solving with the RL factorization is better than solving by using direct LU decomposition.

An extra comment deserves to be made here. In the discussion before we assumed to be interested in solving the system only for a fixed frequency \( \omega \). In general one is interested not only in one specific value of \( \omega \) but rather in a whole frequency range. In chapter 4 it was shown how many of the matrices of the factorization are either independent or linearly dependent of \( \omega \). This means that all these matrices need to be computed only once. The only matrices that need to be recalculated are \( L_{2,3} \) and \( D_{2,3} \) (lemma 2). Meanwhile, with the LU decomposition everything needs to be recomputed again. This provides one extra argument for solving with the RL factorization, specially when one is interested in finding the solution for many different values of \( \omega \).
6.3 RL Study Case

Above we discussed the time complexity of solving the circuit equations with direct LU decomposition and with the RL factorization. Now we will analyze this more quantitatively. In order to do so we will consider the circuit in Figure 6.2.

![RL Circuit Diagram](image)

The size of this system can be measured with the number $N$. Our circuit has $2N + 1$ nodes and hence $m = 2N$. The number of RL branches is $n_l = 3N - 2$. The matrices corresponding to this circuit take the form.

$$A_l = \begin{pmatrix} M_1 & 0 \\ 0 & M_2 \\ I_N & -I_N \end{pmatrix},$$  

(6.23)

where

$$M_1 = \begin{pmatrix} 0 & -1 & -1 \\ 1 & \ddots & \ddots \\ \vdots & \ddots & \ddots & \ddots \\ 1 & -1 & \cdots & \cdots & \cdots \end{pmatrix}, \quad M_2 = \begin{pmatrix} 1 & -1 & -1 \\ \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots \\ 1 & -1 & \cdots \end{pmatrix}.$$  

(6.24)

Here $A_l$ is of size $(3N - 2) \times 2N$, the blocks $M_1$ and $M_2$ are of size $(N_1) \times N$. Additionally we have $A_i = (-1, 0, \ldots, 0)$. All the resistances are assumed to have the same value $r = 2.5$, the inductances are set to $l = 1.25 \times 10^{-9}$. Furthermore we assume a coupling between parallel inductors that are next to each other (see Figure 6.2). The coupling will be denoted by a a factor $0 \leq p \leq 1$, i.e. a proportional value to the value of the inductances. Thus $\mathcal{R} = rI_{3N-2}$ and

$$\mathcal{L} = \begin{pmatrix} I_{N-1} & pI_{N-1} \\ pI_{N-1} & I_{N-1} \\ M_3 \end{pmatrix}, \quad \text{where} \quad M_3 = \begin{pmatrix} 1 & p & \ldots \\ p & 1 & p \\ \ddots & \ddots & \ddots \\ p & 1 \end{pmatrix}.$$  

(6.25)

We selected this circuit because we can vary its size arbitrarily with the parameter $N$. Furthermore we can also observe how the introduction of coupling
affect the performance of the methods. For the case $N = 2$ we computed the solution for the voltage at node 3, using the RL decomposition and using Systems and Control toolbox of MatLab. In Figure 6.3 we observe the result obtained with Systems and Control Toolbox, meanwhile in Figure 6.4 we can see the results obtained after solving with the RL factorization. Since the RL factorization is exact (despite roundoff error), both graphs coincide as expected.

The previous observation validate the results obtained with the RL factorization, now we will briefly explore the effect of the coupling factor. In Figure 6.5 we can observe how the curve of the voltage in node 3 changes with different coupling factors. The higher the coupling factor is, the less steep the magnitude plot is. The phase changes in a similar way, for higher coupling factors the phase curve is "pulled" to the right (Figure 6.6).

In order to analyze the time complexity, we implemented both solution meth-
6.3 RL Study Case

Time Complexity

ods in MatLab. We increased the size of the circuit (see Figure 6.2) by increasing the parameter $N$. First we can express the time complexity in terms of the variable $N$. Using that for our circuit $m = 2N$ and $n_l = 3N - 2$, we have that the complexity of solving, with direct LU decomposition, the RL circuit in Figure 6.2 for a fixed frequency is:

$$O((2(5N - 2))^3 + [2(5N - 2)]^2).$$  \hspace{1cm} (6.26)

When solving with the RL algorithm we have the following. For this special circuit, the generated $P^{-1}$ is unfortunately not sparse enough. For instance in Figure 6.7 we can observe the sparsity structure of the matrix for the case when $N = 20$. This structure is not sparse enough to avoid expensive computation time for the calculation of the product $P^{-1}R_{11}P^{-T}$. In Figure 6.8 we can check that in fact the product becomes a full matrix, incurring in $O(m^3)$ flops.

From the discussion above we can conclude that for this special circuit the time complexity of solving the circuit equation with the RL algorithm is:

$$O((N - 2)^3 + [2N]^3 + [3N - 2]^2 + [2(5N - 2)]^2).$$  \hspace{1cm} (6.27)

This will reflect in the numerical results. The following computations were performed in MATLAB on a PC with processor IntelPentiumCoreDuo 1.86GHz with 1GB RAM. In Figure 6.9 we can observe the computation times for different values of $N$. Here we consider $p = 0$, i.e. no inductive coupling. As expected, for small values of $N$, we can see that the RL Algorithm takes more time that solving with direct LU decomposition. This happens because, for small values the terms that determine the running time are the second order terms. The RL Algorithm performs more procedures of order 2 than the LU decomposition this leads to more operations for small values of $N$. Nonetheless, once $N$ is bigger that $10^2$, the dominant terms determining the computation time are the terms of order 3. Consequently the computation time of the RL algorithm starts to be shorter than the one of the direct LU decomposition. The difference might not look very large, but this is only due to the logarithmic scale. For instance, when $N = 512$ direct LU decomposition uses 147.2sec, while the RL algorithm needs only 55.98sec, i.e. almost one third of the time.

When we include a coupling factor $p = 0.1$ (Figure 6.10), the computation times for the RL algorithm remain almost the same. This is not the case for
6.3 RL Study Case

Time Complexity

Figure 6.9: Solution running times for RL circuits without coupling $p = 0$

Figure 6.10: Solution running times for RL circuits with coupling $p = 0.1$

direct LU decomposition. For instance, when $N = 256$, the computation time of the RL algorithm is 0.861 sec for the problem without coupling and 0.995 sec for the problem with coupling. Meanwhile, for the same value of $N$ the computation time of direct LU decomposition is 1.129 sec for the problem without coupling and 3.962 sec for the problem with coupling. This effect becomes more clear for higher values of $N$. For $N = 1024$, the computation time of direct LU decomposition increases from 147.2 sec to 294.3 sec, while the computation time of the RL algorithm changes only from 55.98 sec to 56.8 sec.

Figure 6.11: Executions time for the modified RL circuits without coupling

Figure 6.12: Executions time for the modified RL circuits with coupling

In order to explore how the structure of the network affects the complexity of the RL algorithm, we will consider a modified version of the circuit in Figure 6.2. We will keep the parameter $N$, but we will consider only two vertical branches instead of $N$. This leads to a circuit with $m = 2N$ nodes and $n_l = 2N$ branches. Thus solving with direct LU decomposition leads to the order of
6.4 Example: RLC Ladder Circuit

Time Complexity

\[ 2(2N)^3 + [2(4N)]^2 \] flops. Using the RL algorithms needs about \([2N]^2 + [2N]^3 + [2(4N)]^2\) operations. It is important to notice that the the term “\(n_l - m\)”, in the complexity of the RL algorithm (6.21), vanishes. This means that for this special case Cholesky decomposition does not need to be applied.

In Figure 6.11 we can see the computation times for the problem without coupling. Figure 6.12 displays the results for the problem with a coupling factor of \(p = 0.1\). For the case without coupling the RL Algorithm is more expensive when \(N \leq 256\), for bigger values of \(N\) the RL algorithm it is already cheaper. Moreover, it might be a bit difficult to observe it in the Figure, but the slope of the curve of the RL algorithm is smaller than the one for direct LU. This means that if we increase the size of \(N\), then the difference in time will also increase. This effect can be observed more clearly, for the case with coupling (see Figure 6.12), here for \(N = 256\) the computation time with the RL algorithm is of 0.49 sec, while with direct LU is 1.08 sec.

The difference in computation times keeps increasing together with \(N\). For example when \(N = 1024\) the direct LU needs 85.12 sec, while RL algorithm takes 22.76. The RL algorithm changes from being 2 times cheaper to be 4 times cheaper than direct LU decomposition. We can check again that the introduction of the coupling factor does not greatly affects the performance of the RL algorithm. For example for \(N = 256\), the computation time of the RL algorithm moves from 0.38 sec for the case without coupling, to 0.49 for the problem including coupling, meanwhile the time of the direct LU changes from 0.46 sec to 1.08. This fact is more clear when \(N = 1024\), in this case direct LU changes from 26.88 sec, for the problem without coupling, to 85.12 sec to the problem with coupling. This is not the case for the RL algorithm, the time here moves from 16.02 sec, for the case with no coupling to 22.76 sec for the case with coupling. If this property holds in general, then this will be an extra pro for choosing the RL algorithm to find the solution of RL circuits. In any case, due to the analytic analysis of the complexity, we can conclude that the RL algorithm will perform better, for circuits large enough.

6.4 Example: RLC Ladder Circuit

Above we discussed the RL factorization in detail, now we will do the same with the RLC factorization. To this extend we will use a circuit that is used as basic model for transmission lines, the so called RLC Ladder circuit (see Figure 6.13). We will use \(N\), the number of blocks, as a variable to measure the size of the circuit.

![Figure 6.13: RLC Ladder Circuit with N Blocks](image)

The RLC ladder with \(N\) blocks, contains \(m + 1 = N + 2\) nodes and \(3N\)
6.4 Example: RLC Ladder Circuit

To be more precise we have, \( n_l = N \), \( n_g = N \) and \( n_c = N \), where \( n_l \), \( n_g \), \( n_c \) denote the number of inductors, conductances and capacitors in the circuit. The matrices \( A_l, A_g, A_c \) are of size \( N \times N + 1 \) and have the form:

\[
A_l = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ \cdots & \cdots & \cdots \\ 0 & 0 & 1 \end{pmatrix}, \quad A_g = A_c = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & 0 & 1 \end{pmatrix}
\]

(6.28)

Additionally we have \( A_i = (-10, \ldots, 0) \). For us to show the RLC factorization works we will first consider the case \( N = 1 \). For appearance purposes we will consider \( R = 2 \), \( L = 3 \), \( C = \frac{1}{4} \) and \( \omega = 1 \). The matrix of the circuit equations in standard form (2.19) is:

\[
\begin{pmatrix}
\hat{R} & -\omega \hat{L} & \hat{P} & 0 \\
\omega \hat{L} & \hat{R} & 0 & \hat{P} \\
\hat{P}^T & 0 & -\hat{G} & \omega \hat{C} \\
0 & \hat{P}^T & -\omega \hat{C} & -\hat{G}
\end{pmatrix}
\]

(6.29)

and substituting the chosen values we have:

\[
\begin{pmatrix}
2 & -3 & -1 & 1 & 0 & 0 \\
3 & 2 & 0 & 0 & -1 & 1 \\
-1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & -\frac{1}{4} & 0 & \frac{1}{4} \\
0 & -1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & -\frac{1}{4} & 0 & -\frac{1}{4}
\end{pmatrix}
\]

(6.30)

On the other hand, the reformulation of the circuit equation (5.20), which we need in order to apply the RLC factorization, are:

\[
\begin{pmatrix}
G^{-1} & 0 & 0 & -A_g \\
0 & \mathcal{R} & -\omega \mathcal{L} & -A_c \\
0 & -\omega \mathcal{L}^{-1} & G^{-1} & -A_g \\
-A_g^T & -A_c^T & \omega \mathcal{L} & 0 \\
-A_c^T & -A_g^T & \mathcal{R} & 0 \\
-A_l^T & -A_l^T & 0 & 0
\end{pmatrix}
\]

(6.31)

and substituting the correspondent values we get:

\[
\begin{pmatrix}
5 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & -1 & 0 \\
0 & 0 & 2 & 0 & 0 & 0 & -3 & -1 & 1 & 0 \\
0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & -4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 3 & 0 & 0 & 2 & 0 & 0 & -1 & 1 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & 1 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

(6.32)
6.4 Example: RLC Ladder Circuit

We can clearly observe how the size of the matrix increases by 4 = 2(n_g + n_c).

We also recognize the matrix arranged in blocks $\hat{P}$, $\hat{X}$ and $\hat{Y}$,

$$
\hat{P} = \begin{pmatrix}
0 & -1 \\
0 & -1 \\
-1 & 1
\end{pmatrix},
\hat{X} = \begin{pmatrix}
5 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 2
\end{pmatrix},
\hat{Y} = \begin{pmatrix}
0 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & -3
\end{pmatrix}.
$$

(6.33)

After applying algorithm 4 to $\hat{P}$ we find:

$$
\Pi_1 = \begin{pmatrix}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 0
\end{pmatrix},
\Pi_2 = \begin{pmatrix}
0 & 1 \\
1 & 0
\end{pmatrix},
$$

then

$$
\Pi_1 \hat{P} \Pi_2^T = \begin{pmatrix}
-1 & 0 \\
1 & -1 \\
-1 & 0
\end{pmatrix} =: P
$$

(6.34)

and identify

$$
P_1 = \begin{pmatrix}
-1 & 0 \\
1 & -1
\end{pmatrix},
P_2 = (-1, 0),
$$

then

$$
P_1^{-1} = \begin{pmatrix}
-1 & 0 \\
1 & -1
\end{pmatrix}
$$

(6.35)

Furthermore, as it was done for the general case, we define the following permutation matrices:

$$
Q = \begin{pmatrix}
\Pi_1 & \Pi_2 \\
\Pi_2 & \Pi_1
\end{pmatrix},
\Pi_3 = \begin{pmatrix}
I_2 & I_2 \\
I_1 & I_4
\end{pmatrix}
$$

(6.36)

Then we have:

$$
\Pi_1 \hat{X} \Pi_1^T =: X = \begin{pmatrix}
5 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 0
\end{pmatrix},
\Pi_1 \hat{Y} \Pi_1^T =: Y = \begin{pmatrix}
0 & 0 & 0 \\
0 & -3 & 0 \\
0 & 0 & 4
\end{pmatrix}
$$

(6.37)

and thus

$$
\Pi_3 Q A Q^T \Pi_3^T = \begin{pmatrix}
A_{11} & A_{12} & B_1 \\
A_{21} & A_{22} & B_2 \\
B_1^T & B_2^T & 0
\end{pmatrix},
$$

where

(6.38)

$$
A_{11} = \begin{pmatrix}
5 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 5
\end{pmatrix},
A_{12} = \begin{pmatrix}
0 & 0 \\
0 & 0 \\
0 & 0
\end{pmatrix},
B_1 = \begin{pmatrix}
-1 & 0 & 0 \\
1 & -1 & 0 \\
0 & 0 & -1
\end{pmatrix}
$$

(6.39)

$$
A_{21} = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix},
A_{22} = \begin{pmatrix}
0 & 4 \\
0 & 0 \\
0 & 0
\end{pmatrix},
B_2 = \begin{pmatrix}
-1 & 0 & 0 \\
0 & 0 & -1 \\
0 & 0 & 0
\end{pmatrix}
$$

(6.40)

Then we have:

$$
B_1^{-1} A_{11} B_1^T = \begin{pmatrix}
5 & 5 & 0 & 0 \\
5 & 7 & 0 & -3 \\
0 & 0 & 5 & 5 \\
0 & 3 & 5 & 7
\end{pmatrix},
D_1 = \begin{pmatrix}
5 & 0 & 0 & 0 \\
0 & 7 & 0 & 0 \\
0 & 0 & 5 & 0 \\
0 & 0 & 0 & 7
\end{pmatrix}
$$

(6.41)
6.4 Example: RLC Ladder Circuit

**Time Complexity**

\[ L_1 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ -5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -3 & -5 & 0 \end{pmatrix}, \quad U_1 = \begin{pmatrix} 0 & -5 & 0 & 0 \\ 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & -5 \\ 0 & 0 & 0 & 0 \end{pmatrix} \]  

(6.42)

and

\[ M = \begin{pmatrix} 5 & 5 & 0 & 0 \\ 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 0 \\ 0 & -5 & 0 \end{pmatrix}, \quad F = \begin{pmatrix} 5 & 0 \\ 5 & 0 \\ 0 & 5 \\ 0 & 5 \end{pmatrix} \]  

(6.43)

The matrix which need to be LDU decomposed (see theorem 9) is:

\[ \hat{W} = \begin{pmatrix} 5 & 4 \\ -4 & 5 \end{pmatrix} \]  

(6.44)

and after applying the LDU decomposition we find:

\[ \Pi_e = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad L_2 = \begin{pmatrix} 1 & 0 \\ -4 & 5 \end{pmatrix}, \quad D_2 = \begin{pmatrix} 5 & 0 \\ 0 & 41 \end{pmatrix}, \quad U_2 = \begin{pmatrix} 1 & 4 \\ 0 & 5 \end{pmatrix}. \]  

(6.45)

This finishes the factorization, just some reordering, to find the matrices \( \tilde{L} \), \( \tilde{D} \), \( \tilde{U} \), is left to the reader. In Figure 6.14 we can observe the magnitude and the phase plots of the voltage at node 2, for the RLC circuit with \( N = 1 \). The curve of the magnitude starts to bend for frequencies higher that \( \omega = 10^{-2} \text{ (rad/sec)} \).

![Figure 6.14: Magnitude and Phase of the Voltage at node 2 in a RLC Ladder with 1 block](image-url)
6.5 RLC Complexity

In the previous section we saw more clearly how the RLC factorization can be computed. This RLC factorization can be used to solve the equations of an RLC circuit. But first let us make a comment on the complexity of solving the equation with direct LU decomposition. For this case we can use the normal formulation (2.19). The size of the matrix is then \(2(n_l + m)\), thus the complexity is:

\[ O((2(n_l + m))^3 + (2(n_l + m))^2), \]  

(6.46)

where, \(n_l\) is the number of inductors (in series with a resistor) branches and \(m + 1\) is the number of nodes in the RLC circuit. Now we will discuss the complexity of using the RLC factorization for a direct solver. The steps to do this are put together in algorithm 6.

Algorithm 6 RLC Solution Algorithm

INPUT: circuit matrices: \(A,A_c, A_g, A_i, R, L, G, C\) and a frequency \(\omega\)

OUTPUT: solution vector \(x\) for the frequency \(\omega\)

\[ x^T = (i_{gr}, l_{ir}, l_{ir}, i_{pi}, l_{ri}, i_r, v_r, v_i)^T \]

1. Determine \(G^{-1}, C^{-1}\), write the equations as in (5.20) and find permutation matrices \(\Pi_1\) and \(\Pi_2\), transforming \(\hat{P}\) into lower trapezoidal form (see algorithm 4)

2. Transform the system matrix \(A\) with \(Q\) leading from the matrices \(\hat{X}, \hat{Y}\) and \(\hat{P}\) to \(X, Y\) and \(P\)

3. Arrange the system matrix in blocks \(A_{ij}\) (see (5.27))

4. Determine \(P^{-1}_1\) and thus find the matrices \(D_1, L_1\) and \(U_1\) (see theorem 7)

5. Perform an LDU decomposition on \(W\) (see theorem 9), leading to the matrices \(\Pi_1, D_2, L_2\) and \(U_2\)

6. Apply the correspondent permutations, solve with backward substitution

Since \(G\) and \(C\) are diagonal, the computation of \(G^{-1}\) and \(C^{-1}\) is a problem of linear complexity. Finding the permutation matrices \(\Pi_1\) and \(\Pi_2\) takes about \((n_l + n_g + n_e)^2\) flops with algorithm 4. Multiplying by \(Q\) and \(\Pi_3\) is cheap because both matrices are permutations. Since \(P_1\) is lower triangular and sparse (at most two nonzero values per row), then the calculation of \(P_{-1}^{-1}\) takes only \(m^2\) flops. 

Computing the products \(P_{-1}^{-1}X_{11}P_1^{-T}\) and \(P_{-1}^{-1}Y_{11}P_1^{-T}\) might take up to \(O(m^3)\) operations, in case \(P_{-1}^{-1}\) is not sparse enough. If the matrix is sparse then then computing the products is cheap. Performing an LDU decomposition on the matrix \(\hat{W}\) costs about \([2(n_l + n_g + n_e - m)]^3\) operations. Finally solving the resulting lower and upper triangular systems needs \([2(n_l + n_g + n_e + m)]^2\) operations. Altogether the complexity of the RLC algorithm is:

\[ O((n_l + n_g + n_e)^2 + m^2 + m^3 + [2(n_l + n_g + n_e - m)]^3 + [2(n_l + n_g + n_e + m)]^2). \]  

(6.47)

Provided \(P_{-1}^{-1}\) is sparse enough the complexity reduces to:

\[ O((n_l + n_g + n_e)^2 + m^2 + [2(n_l + n_g + n_e - m)]^3 + [2(n_l + n_g + n_e + m)]^2). \]  

(6.48)
It should be clear that the RLC algorithm might be more expensive than applying direct LU decomposition. This happens mainly because the RLC algorithm increases the size of the system equations in order to provide the factorization. Nonetheless, in some cases, the RLC algorithm can be faster than direct LU decomposition. We will show this by analyzing the case of the RLC ladder circuit.

The results displayed in Figures 6.15, 6.18, and 6.19 were performed in MatLab in a PC with processor AMD Turion64 X2 1.60GHz with 2GB RAM. In Figure 6.15 we see the running times for solving the RLC ladder with different number of blocks. For this case direct LU decomposition is always faster than the RLC algorithm. We observe how the curve for the RLC algorithm always lies above the one for direct LU decomposition.

We can explain this in an analytic form as well. We just need to remember that \( n_l = n_g = n_c = m = N \) then using direct LU needs:

\[
O([4N]^3 + [4N]^2) \quad (6.49)
\]

operations. To calculate the complexity of the RLC algorithm we can use expression (6.48). We can in fact use this expression, because for this case \( P^{-1}_1 \) is sparse enough, the sparsity structure of \( P^{-1}_1 \) and \( P^{-1}_1 P^{-T}_1 \) can be seen in Figures 6.16 and 6.17 respectively. Using the expression we get the complexity:

\[
O([3N]^2 + N^2 + [4N]^3 + [8N]^2). \quad (6.50)
\]

This expression is clearly bigger than the one for direct LU decomposition, because of the extra terms of order 2.

At first glance it might seem not such a good an idea to use the RLC algorithm, but luckily this is not always the case. For instance, if we considered a modified version of the RLC ladder circuit the RLC algorithm performs better than direct LU decomposition. We will consider the RLC ladder with the same block structure, but we will “delete” some of the vertical branches. The way in which we will remove the vertical branches is as follows.

Given a proportion of vertical branches to be removed \( 1 - p \), we first compute \( \lfloor pN \rfloor \), where \( \lfloor x \rfloor \) denotes the biggest integer smaller or equal than \( x \). Then we
6.5 RLC Complexity

Time Complexity

start by removing capacitors branches from left to right (see Figure 6.13), till we have removed \( \lceil pN \rceil \) capacitor branches in total. In the same, but this time from right to left we remove the same quantity of conductances. Thus the modified version of the RLC ladder satisfies \( n_l = N, m = N \) and \( n_g = n_c = pN \), where \( N \) is the number of blocks in the circuit.

In Figure 6.18 we can see the computation times for the modified version of the RLC ladder circuit with factor \( p = 0.5 \). For small values of \( N \) direct LU is still faster that the RLC algorithm, but in contrast to the previous case, this time the RLC algorithm reaches the performance of direct LU for \( N = 256 \). For difference in performance increases with \( N \). When \( N = 1024 \) direct LU needs 12.83 sec, while the RLC algorithm needs 10.35 sec. Later for \( N = 2054 \) LU needs 96.53 and the RLC algorithm needs 76.13, the difference in computation times changes from about 2 sec to about 20 sec. For this case the RLC algorithm behaves better than direct LU decomposition. We can again explain this, using the expressions for the time complexity.

Since for the modified RLC the number of resistor-inductor branches \( n_l \) and the number of nodes \( m + 1 \) do not change, then the complexity of direct LU remains the same. Meanwhile, \( n_g = n_c = \lfloor 0.5N \rfloor \), thus \( n_l + n_g + n_c = 2N \) and \( n_l + n_g + n_c - m = N \). Then the expression for the complexity of the RLC algorithm now reads:

\[
O([2N]^2 + N^2 + [0.8N]^3 + [2(2.4N)]^2),
\]

(6.51)

the only term of third order in the previous expressions is \([2N]^3\), which is smaller compared to \([4N]^3\), the leading term in the complexity of solving via direct LU decomposition.

This effect is more clear when we study the running times for solving the modified version of the RLC ladder with factor \( p = 0.2 \). For this case the complexity of the RLC factorization is:

\[
O([1.4N]^2 + N^2 + [0.8N]^3 + [2(2.4N)]^2),
\]

(6.52)

while the complexity of solving with direct LU stays the same. Numerical results appear in Figure 6.19. The difference between the computation times is now evident. Already for \( N = 64 \) the RLC algorithm performs better than direct LU. For larger values of \( N \) the RLC algorithm is much faster. For instance for
6.5 RLC Complexity

Time Complexity

\[ N = 256, \text{ the elapsed time for RLC is } 0.03\text{sec} \text{ while the one for direct LU is } 0.24. \]

As another example, when \( N = 2048 \) direct LU needs 95.89 sec while the RLC algorithm needs only 2.58 sec, this is a time reduction factor of 36. Confirming, that for this case, the RLC algorithm performs much faster than direct LU.

With these examples we can get some useful information to know when it is advisable to use the RLC algorithm instead of LU decomposition for solving directly the circuit equations. First it is very important to remark, that for RL circuit circuit case, the RL algorithm is always better than direct LU. Provided that the circuit is big enough.

For the case of RLC circuits, we can say the following. If the circuit contains just some few capacitors and few conductances, then the RLC algorithm should perform better than direct LU. We can give a reason for this by examining the expression for the complexity of the RLC algorithm. We immediately observe that we require \( n_g + n_c \leq 2m \) for the RLC algorithm to be faster than direct LU decomposition. If this condition is not fulfilled then there is no chance for the RLC algorithm to perform better than direct LU decomposition. For this
cases is better to discard the RLC algorithm. On the other hand, if the condition is fulfilled, the RLC algorithm might be much faster than direct LU decomposition, as it was shown in the example above. This finishes our discussion of the usage of the RL and RLC factorization for implementing direct solvers. The kind of factorization developed here can also be used to find preconditioners for iterative schemes, but we will not discuss it here. Such kind of preconditioner matrices has already been addressed by Schilders [1] and also by Cao [7], leading to very interesting results.
Chapter 7

Conclusions

In mathematical terms, the main problem we were concerned along this thesis is the solution of indefinite and non symmetric linear equations. Such kind of systems arise in different areas. We showed how these systems appear in electronics, in the simulation of RLC circuits for our particular case. In this area, there is a special need for fast solvers, mainly because nowadays the circuits contain a very large number of components. This last condition make direct solvers very expensive and thus not affordable. From the point of view of iterative solvers, the poor spectral properties of the matrix, delay the convergence of the iterative schemes. In simple words, developing efficient solvers for these kind of systems is not a trivial issue.

The matrix associated with the RLC circuit falls in the class of saddle point matrices. Many efforts have been done to develop solution methods for this kind of problem. In this thesis we reviewed some of these methods, as well as the basics of linear equations, in order to make the work as self contained as possible. The Schilders factorization is a method in which we made special emphasis. Based on this factorization, we were able to construct factorizations for both, first for the case of RL circuits and, after some extra effort rewriting the original equations, for the general case of RLC circuits as well.

7.1 Contribution of this work

It is important to mention that in this thesis, we constructed and provided proofs for the existence of the factorizations mentioned above. We achieved this in the following way. First we discussed the problem of rearranging the incidence matrix in to lower trapezoidal form, we also included an algorithm to perform such rearrangement. Then we treated the case RL, and by proving the existence of this factorization we were also able to conclude that the matrix associated with the RL circuit is invertible. In contrast, for the RLC case we first had to prove the invertibility of the original matrix, then reformulate the circuit equations in to a slightly different form and finally we constructed the factorization.
The presented factorizations can be used for implementing direct solvers or to construct preconditioners in order to apply iterative schemes. This was already mentioned in the literature [1] [7]. In this text we were more concerned analyzing the use of these factorizations for implementing direct solvers. We stated this in terms of an algorithm for each case. Furthermore we provided a comparison, in terms of the time complexity, between these algorithms and applying the direct LU decomposition. This was done analytically and also numerically to give an idea of how the algorithms will behave in practice.

From the discussion regarding the time complexity of the algorithms we were able to conclude the following. Provided that the circuit is big enough, the RL factorization is always faster that applying direct LU decomposition. Unfortunately, this not always the case for the general RLC factorization. In this case we found that sometimes applying the RLC factorization is definitely not recommended. A condition when this happens, was stated in terms of the number of capacitors, conductances and nodes in the RLC circuit. Despite this fact, by means of the example of the RLC ladder circuit we were able to show that there are still some cases when applying the RLC algorithm is much cheaper than applying direct LU factorization. One extra advantage that was appointed but not yet exploited here, is the fact that for different input frequencies only a small part of the factorizations needs to be recomputed. This could lead to more advantages.

7.2 Future Work

We already mentioned that the frequency dependences of the factor matrices should be further explored and exploited. This problem has a lot to do with rearranging the incidence matrix into lower trapezoidal form. In general the reordering is not unique and consequently there is room for trying to find an optimal one. In terms of graph theory, the problem of rearranging the matrix is equivalent to finding an spanning tree in the circuit network. We already mentioned and gave references for the possibility of constructing preconditioners using the factorizations as basis. Another possibility would be to try to use them, together with a model order reduction technique, in order to find model reduction algorithms, better suited for RLC circuits; which is very important in the industry.
Bibliography


BIBLIOGRAPHY


Acknowledgements

First of all I would like to thank my thesis supervisor prof.dr. Wil Schilders, for giving me the opportunity of making this project. He shared lots of his expertise with me, he gave me much freedom to work, an always provided an easy going working atmosphere. His contribution to this work was very important, just to mention that the main ingredient for completing this work is a factorization named after him.

I thank my second advisor dr. Joost Rommes. He was also very helpful and always provided me with many comments and ideas for improving my work. Besides that he also encouraged insightful discussions.

I would like to thank prof.dr. Henk van der Vorst for the time he took to have, what for me was, a very constructive discussion. I also thank him for pointing me to very important references for doing my work. I thank my colleagues at NXP, where I carried out this project. I specially mention Maria Uglyumova, for helping me to find very useful literature. I also thank Charles Chilaka for reviewing this text.

I would like to thank all the persons I met at both universities, the Technische Universitaet Kaiserlatuern and the Technische Universiteit Eindhoven. The staff members, my colleague students, and specially all the friends I now have, which are so many that I can not name all of them here.

I would like to acknowledge the committee members, dr. Sorin Pop and dr. Jos Maubach, for investing their time reading this thesis.

I am very grateful to the European Commission. My complete studies were supported with a grant under the ERASMUS MUNDUS scheme.

Last but definitely not least, I would like to thank family, my brothers Abril, Karol and Omar. Most importantly my parents Patricio and Imelda, who have always supported me in all my projects. To them I dedicate this thesis.