A General Approach to Circuit Equations

T. Tuma and F. Bratkovič
University of Ljubljana,
Faculty of Electrical Engineering,
1000 Ljubljana, Tržaška 25, Slovenia.
Phone: 386-61-1768-329, Fax: 386-61-1264-630,
Email: tadej.tuma@fe.uni-lj.si

Paper originally published:
The International Journal of Circuit Theory and Applications,
vol. 22, page 431–445, 1994

Abstract

Computer aided circuit analysis is a complex procedure, which may be roughly divided into three steps. The first stage is device modelling, where quantitative properties of the participating electronic devices are determined. In the second step, the circuit's topology in conjunction with the device equations is being exploited, in order to formulate a linearized nonsingular equation set. Eventually, the equation set is iteratively solved in the third phase, yielding a solution vector of circuit variables.

This paper deals with the second and third step, where diverse sparse matrix methods are employed. Its ambition is to provide a general mathematical form, by which most matrix manipulating techniques can be described, thus enabling their classification on a theoretical level. Usually, these methods would be presented descriptively rather than strictly mathematically.

The paper introduces a matrix reduction operator, from which a general matrix operator equation is deduced. This operator equation can be used to describe entire analysis approaches. After some necessary definitions, the purpose of this paper is illustrated by studying several classic examples rather than giving lots of mathematical proof. The selected examples involve some well known linear equation set solution methods as well as several typical equation set transformations.

1 Introduction

In order for an analysis approach to be efficient, it is most important that the equation formulation principle is in harmony with its subsequent solution technique. A famous example of such a successful coupling is the modified nodal equation set in combination with a fine tuned LU decomposition as implemented in SPICE. Both methods are well known and are usually described by schemes, symbolic programmes or graphs, accompanied with lots of rules and recommendations. Various approaches to circuit analysis have been proposed in the past two decades and benchmarked against each other. In most cases the definitions of the methods indicate only weak relations between them. Looking at their coded computer implementations, however, they obviously have much in common. Hence, there should be a way of expressing these relations on a higher level.

This paper is based on a very simple idea. It is a fact, that eventually even very sophisticated methods can be decomposed to only four basic determinant preserving matrix manipulations: 1) Any pair of rows may be exchanged. 2) Analogously, any pair of columns may be exchanged. 3) Any linear combination of rows may be added to another row. 4) Also, any linear combination of columns may be added to another column. Bearing this in mind, it should be possible to construct a general operator equation which can reflect any analysis approach. In the following sections we will try to find this 'common ancestor', which should enable us to formally compare even very faintly related techniques.

2 The Permutation

Let $\boldsymbol{P}$ denote a square permutation matrix. A permutation matrix consists of exactly one unit element in each row and column. All other elements are zero. Thus, every permutation matrix can be obtained from an identity matrix $\boldsymbol{I}$ by interchanging its rows. Obviously, permutation matrices are orthogonal

$$\boldsymbol{P}^T\boldsymbol{P} = \boldsymbol{P}^{-1}\boldsymbol{P} = \boldsymbol{I}. \tag{1}$$

Permutation matrices may be used to alter the succession of rows and columns in any matrix $\boldsymbol{A}$ by premultiplication and postmultiplication, respectively. This process can be looked upon as a transformation of $\boldsymbol{A}$ by a row permutation $\boldsymbol{P}$ and a column permutation $\boldsymbol{Q}$

$$\boldsymbol{P}_{r\times r}\boldsymbol{A}_{r\times c}\boldsymbol{Q}^T_{c\times c} = \hat{\boldsymbol{A}}. \tag{2}$$

Throughout this paper, permuted matrices shall be denoted by a hat $\hat{\boldsymbol{A}}$. The rules of matrix multiplication impose the only restrictions on the equation above requiring the matrices' dimensions to match.

3 The Reduction Operator

Let R_n be a reduction operator of rank n. By definition, R_n maps a given matrix $\boldsymbol{A}_{r\times c}$ from the reduction operator's domain vector space $\boldsymbol{A} \in \mathcal{D}(R_n)$ to an upper block triangular matrix $\tilde{\boldsymbol{A}}_{U_{r\times c}}$ in its range vector space $\tilde{\boldsymbol{A}}_U \in \mathcal{R}(R_n)$

$$R_n(\boldsymbol{A}) \stackrel{\triangle}{=} \tilde{\boldsymbol{A}}_U. \tag{3}$$

Unless the contrary is specifically stated, a tilde shall denote elements from the reduction operator's range. The domain of R_n is a subset of the vector space of real matrices satisfying

$$\mathcal{D}(R_n) = \{\forall \boldsymbol{A}_{r\times c} : r, c > n\} \cap \{\forall \boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11_{n\times n}} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} : \exists \boldsymbol{A}_{11}^{-1}\}. \tag{4}$$

The reduction operator's range on the other hand, is a subset of its domain, namely

$$\mathcal{R}(R_n) = \mathcal{D}(R_n) \cap \{\forall \boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11_{n\times n}} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} : \boldsymbol{A}_{21} = 0\}. \tag{5}$$

The reduction operator associates each given matrix from the operators domain with a corresponding image in the operators range by premultiplying the given matrix with the following square matrix

$$R_n(\boldsymbol{A}) \stackrel{\triangle}{=} \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ -\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{I} \end{bmatrix}_{r\times r} \begin{bmatrix} \boldsymbol{A}_{11_{n\times n}} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix}_{r\times c} = \begin{bmatrix} \boldsymbol{A}_{11_{n\times n}} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix}_{r\times c} \stackrel{\triangle}{=} \tilde{\boldsymbol{A}}_U, \tag{6}$$

where

$$\tilde{\boldsymbol{A}}_{22} \stackrel{\triangle}{=} \boldsymbol{A}_{22} - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}. \tag{7}$$

The 'reduced' matrix $\tilde{\boldsymbol{A}}_U$ contains an $(r-n)\times n$ zero submatrix in its south–west corner. Actually, the linear equation set associated with $\boldsymbol{A}$ is reduced rather than the matrix itself, which will be seen later. Nevertheless, we prefer to use the term 'reduced matrix', suggesting that the matrix contains reduced information.

Reduction operators are obviously nonlinear operators. Furthermore, it can be proved, that reduction operators are singular operators, since their domain and range vector spaces, $\mathcal{D}(R_n)$ and $\mathcal{R}(R_n)$ respectively are not isomorphic. Hence inverse reduction operators do not exist. The transformation they perform is destructive in some sense, as certain information is lost. Yet, they have some characteristic properties

$$\begin{aligned} R_n(\boldsymbol{U}) &= \boldsymbol{U} && (8)\\ R_n(\alpha\boldsymbol{A}) &= \alpha\, R_n(\boldsymbol{A}) && (9)\\ R_n(R_n(\boldsymbol{A})) &= R_n(\boldsymbol{A}) && (10)\\ \boldsymbol{A}\boldsymbol{B} = 0 &\Rightarrow R_n(\boldsymbol{A})\boldsymbol{B} = 0 && (11)\\ \det(R_n(\boldsymbol{A})) &= \det(\boldsymbol{A}) && (12)\\ R_m(R_n(\boldsymbol{A})) &= R_n(R_m(\boldsymbol{A})). && (13) \end{aligned}$$

It is quite obvious, that upper triangular matrices $\boldsymbol{U}$ are not affected by reduction operators (8) neither are diagonal or identity matrices. Furthermore, the matrix' determinant is preserved by any reduction operator (12). Also, reduction operators are not sensitive to scalar multiplications (9). Once a reduction operator has been applied to a matrix, any further application of the same operator has no effect whatsoever. In other words, reduction operators are idempotent (10). A series of arbitrary reduction operators performs a transformation of the domain independent of the succession in which the individual operators have been applied (13). Finally, if the product of two matrices is a null matrix, then any reduction operator may be applied to the left–hand matrix (11). The proofs for (8...13) are in Appendix A.

3.1 The Transposed Reduction Operator

A reduction operator which maps the transposed domain to a transposed codomain shall be – by definition – called transposed reduction operator and denoted by

$$\begin{aligned} R_n^T(\boldsymbol{A}) &\stackrel{\triangle}{=} \left(R_n(A^T)\right)^T = \left(\begin{bmatrix} \boldsymbol{I} & 0 \\ -\boldsymbol{A}_{12}^T\boldsymbol{A}_{11}^{T\,-1} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{11}^T & \boldsymbol{A}_{21}^T \\ \boldsymbol{A}_{12}^T & \boldsymbol{A}_{22}^T \end{bmatrix}\right)^T = \\ &= \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12} \\ 0 & \boldsymbol{I} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}_{11} & 0 \\ \boldsymbol{A}_{21} & \tilde{\boldsymbol{A}}_{22} \end{bmatrix} \stackrel{\triangle}{=} \tilde{\boldsymbol{A}}_L, \end{aligned} \tag{14}$$

where $\tilde{\boldsymbol{A}}_{22}$ is identically equal to (7).

Please notice the similarity in expressions (6) and (14). In both cases submatrix $\boldsymbol{A}_{22}$ in the south–east corner is transformed to $\tilde{\boldsymbol{A}}_{22}$. The only difference is, that this time the north–east submatrix $\boldsymbol{A}_{12}$ is put to zero instead of the south–west one $\boldsymbol{A}_{21}$. The transposed reduction operator inherits all six basic properties (8...13) with minor modifications of (8) and (11)

$$R_n^T(\boldsymbol{L}) = \boldsymbol{L} \tag{15}$$

$$\boldsymbol{A}\boldsymbol{B} = 0 \;\Rightarrow\; \boldsymbol{A}\, R_n^T(\boldsymbol{B}) = 0. \tag{16}$$

Once the transposed reduction operator is defined, we have to mention its relation to the original reduction operator. At this time, we shall only state the relation

$$R_n^T(\boldsymbol{A})\, R_n(R_n^T(\boldsymbol{A}))^{-1}\, R_n(\boldsymbol{A}) = \boldsymbol{A}, \tag{17}$$

and comment on it later.

3.2 Multiple Reduction Operators of Ascending Rank

Before we continue, let us review some consequences of applying ascending reduction operators. For this purpose we shall use R_m and R_n $(n > m)$ on matrix $\boldsymbol{A}$, which embraces two nonsingular diagonal matrices $\boldsymbol{A}_{11}$ and $\boldsymbol{A}_{22}$ of order m and $n-m$ respectively (18)

$$R_n(R_m(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ \boldsymbol{A}_{31} & \boldsymbol{A}_{32} & \boldsymbol{A}_{33} \end{bmatrix})) = R_n(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ 0 & \tilde{\boldsymbol{A}}_{22} & \tilde{\boldsymbol{A}}_{23} \\ 0 & \tilde{\boldsymbol{A}}_{32} & \tilde{\boldsymbol{A}}_{33} \end{bmatrix}). \tag{18}$$

According to (6) $\boldsymbol{A}$ is transformed to hold a zero submatrix in its south–west corner. In addition to this, the four submatrices in the south–east corner are modified, which is denoted by tildes

$$\tilde{\boldsymbol{A}}_{ij} = \boldsymbol{A}_{ij} - \boldsymbol{A}_{i1}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{1j}. \tag{19}$$

Now, let us perform the second reduction of rank n in (18). By definition, we get another zero submatrix, and again, the south–east submatrix is transformed, which is denoted by double tildes

$$R_n(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ 0 & \tilde{\boldsymbol{A}}_{22} & \tilde{\boldsymbol{A}}_{23} \\ 0 & \tilde{\boldsymbol{A}}_{32} & \tilde{\boldsymbol{A}}_{33} \end{bmatrix}) = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ 0 & \tilde{\boldsymbol{A}}_{22} & \tilde{\boldsymbol{A}}_{23} \\ 0 & 0 & \tilde{\tilde{\boldsymbol{A}}}_{33} \end{bmatrix}. \tag{20}$$

Only this time the transformation of $\tilde{\boldsymbol{A}}_{33}$ appears more complex, namely

$$\tilde{\tilde{\boldsymbol{A}}}_{33} = \tilde{\boldsymbol{A}}_{33} - [\,0 \quad \tilde{\boldsymbol{A}}_{32}\,] \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{A}_{12} \\ \tilde{\boldsymbol{A}}_{21} \end{bmatrix}. \tag{21}$$

Fortunately, the inverse of a block triangular matrix can be stated in form of

$$\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{A}_{11}^{-1} & \cdots \\ 0 & \tilde{\boldsymbol{A}}_{22}^{-1} \end{bmatrix} \tag{22}$$

thereby simplifying expression (21) to

$$\tilde{\tilde{\boldsymbol{A}}}_{33} = \tilde{\boldsymbol{A}}_{33} - \tilde{\boldsymbol{A}}_{32}\tilde{\boldsymbol{A}}_{22}^{-1}\tilde{\boldsymbol{A}}_{31}. \tag{23}$$

In other words, after R_m has been applied, R_n $(n > m)$ operates only on the south–east corner. This is equal to using R_{n-m} on the four bottom–right submatrices

$$R_n(R_m(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ \boldsymbol{A}_{31} & \boldsymbol{A}_{32} & \boldsymbol{A}_{33} \end{bmatrix})) = \begin{bmatrix} \boldsymbol{A}_{11} & [\, \boldsymbol{A}_{12} \quad \boldsymbol{A}_{13} \,] \\ \begin{bmatrix} 0 \\ 0 \end{bmatrix} & R_{n-m}(\begin{bmatrix} \tilde{\boldsymbol{A}}_{22} & \tilde{\boldsymbol{A}}_{23} \\ \tilde{\boldsymbol{A}}_{32} & \tilde{\boldsymbol{A}}_{33} \end{bmatrix}) \end{bmatrix}. \tag{24}$$

Conclusively, we can say that by applying reduction operators of increasing rank, the operation area is limited by earlier reduction operators. Hence, the operation area is contracted with each subsequent reduction operator. Outside the reduction area, the matrix remains untouched. Fill–in, numerical error and operation count are strictly confined to the shrinking reduction area. Exactly the same goes for transposed reduction operators.

It is important to point out that, in a mathematical sense, a series of ascending, descending or any arbitrary succession of the same reduction operators produces exactly the same result (13). The number and nature of numerical operations however is different, since the ascending case involves the inversion of smaller matrices as the descending one. Although the mathematical outcome is identical in any case, fill–in, numerical error and operation count may heavily depend on the succession of reduction operators. Thus, analytical mathematics is concerned only with the ranks of reduction operators, whether numerical (finite mantissa) mathematics will also take interest in their succession.

Now, let us observe two special cases of ascending reduction operators. The first will describe the triangularization of a square matrix, where the second shall perform a subsequent diagonalization.

Let $\boldsymbol{A}$ denote a nonsingular square coefficient matrix of order n. $\boldsymbol{A}$ is transformed to an upper triangular matrix $\boldsymbol{U}$ by applying $n-1$ successive reduction operators

$$\boldsymbol{U}_{n\times n} = R_{n-1}(\ldots R_2(R_1(\boldsymbol{A}_{n\times n}))\ldots). \tag{25}$$

Analogously to (25), the coefficient matrix $\boldsymbol{A}$ can be mapped to a lower triangular matrix $\boldsymbol{L}$ by $n-1$ successive transposed reduction operators

$$\boldsymbol{L}_{n\times n} = R_{n-1}^T(\ldots R_2^T(R_1^T(\boldsymbol{A}_{n\times n}))\ldots). \tag{26}$$

In both cases the determinant of matrix $\boldsymbol{A}$ is preserved due to (12) and is equal to the product of the pivots in the triangular form: $\det(\boldsymbol{A}) = \det(\boldsymbol{U}) = \det(\boldsymbol{L}) = \prod_{i=1}^n u_{ii} = \prod_{i=1}^n l_{ii}$. Actually (6,14), the pivots of the two triangular forms are equal $u_{ii} = l_{ii}, \forall i$.

The diagonalization of a nonsingular coefficient matrix is a logic extension of its triangularization and is accomplished by appending a series of transposed reduction operators to the triangularizing one

$$\boldsymbol{D}_{n\times n} = R_{n-1}(\ldots R_2(R_1(\boldsymbol{L}_{n\times n}))\ldots) \tag{27}$$

and

$$\boldsymbol{D}_{n\times n} = R_{n-1}^T(\ldots R_2^T(R_1^T(\boldsymbol{U}_{n\times n}))\ldots) \tag{28}$$

respectively. The resulting diagonal matrix $\boldsymbol{D}$ is equal in both cases. At first sight, the diagonalization process seems useless since the pivots have not changed $u_{ii} = l_{ii} = d_{ii}, \forall i$. In this case, the same result would be achieved by simply throwing away all off–diagonal elements of the triangular form. As we shall see later however, the transformations (27) and (28) are very useful in case of a nonsquare coefficient matrix $\boldsymbol{A}$, where a part of matrix $\boldsymbol{A}$ is being diagonalized.

4 A General Operator Equation

In this section we shall consider a general transformation of matrix equation

$$\boldsymbol{AB} = 0. \tag{29}$$

Since the right hand side is a null matrix, we may premultiply or postmultiply the left hand side by any nonsingular matrix. Permutation matrices certainly are nonsingular, hence $(\boldsymbol{PA})(\boldsymbol{BR}^T) = 0$. The expressions in brackets denote a row permutation of $\boldsymbol{A}$ and column permutation of $\boldsymbol{B}$ respectively. By employing (1) we can also change the succession of columns in $\boldsymbol{A}$, as long as it is accompanied by corresponding row permutations in $\boldsymbol{B}$, namely

$$(\boldsymbol{PAQ}^T)(\boldsymbol{QBR}^T) \stackrel{\triangle}{=} \hat{\boldsymbol{A}}\hat{\boldsymbol{B}} = 0. \tag{30}$$

Now we introduce the general operator equation by reducing $\hat{\boldsymbol{A}}$ with reduction operator R_m according to (11) and $\hat{\boldsymbol{B}}$ with R_n^T according to (16)

$$R_m(\boldsymbol{PAQ}^T)\, R_n^T(\boldsymbol{QBR}^T) = R_m(\hat{\boldsymbol{A}})\, R_n^T(\hat{\boldsymbol{B}}) \stackrel{\triangle}{=} \tilde{\hat{\boldsymbol{A}}}\tilde{\hat{\boldsymbol{B}}} = 0. \tag{31}$$

The original matrix equation (29) is mapped by the general operator equation (31) without affecting the equality to the null matrix on the right hand side. Also, the determinants of both matrices $\tilde{\hat{\boldsymbol{A}}}$ and $\tilde{\hat{\boldsymbol{B}}}$ are preserved. Please note, that the resulting matrix equation $\tilde{\hat{\boldsymbol{A}}}\tilde{\hat{\boldsymbol{B}}} = 0$ has the same form as the original one, therefore several successive general operator equations may be applied. In the following section we shall illustrate the use of the reduction operator and its associated operator equation.

5 The Formulation of Circuit Equations

The most typical way to analyze a circuit is to linearize it within a small time interval and about a certain bias point. Then its modified nodal equation set is formulated and permuted, in order to minimize fill-in and avoid bad pivots during the Gaussian elimination. Once the elimination is complete, the unknowns are obtained by back substitution. With respect to the solution vector, a new bias is computed and the equation set of the next iteration is produced. After convergence has been reached, the entire procedure is repeated for the next time interval.

In this section we shall be concerned with the second step of computer aided circuit analysis, that is the formulation of the appropriate equation set. The standard notation for a linear algebraic equation system of n–th order with c constant vectors would be

$$\boldsymbol{A}_{n\times n}\boldsymbol{X}_{n\times c} = \boldsymbol{B}_{n\times c}. \tag{32}$$

In this equation, $\boldsymbol{A}$ represents a square coefficient matrix, $\boldsymbol{B}$ is composed of c columns denoting constant excitation vectors (the equation's right hand side) and $\boldsymbol{X}$ contains c columns describing the solution vectors to the corresponding excitation vectors (the equation's unknowns). For our purposes however, we shall rephrase the above equation as follows

$$[\,\boldsymbol{A}_{n\times n} \quad \boldsymbol{B}_{n\times c}\,] \begin{bmatrix} \boldsymbol{X}_{n\times c} \\ -\boldsymbol{I}_{c\times c} \end{bmatrix} = 0. \tag{33}$$

When applying an operator equation (31) to equation set (33), we are usually interested neither in the succession of the constant vectors in $\boldsymbol{B}$ nor in the succession of the corresponding variable vectors in $\boldsymbol{X}$. These facts are formally denoted by dropping the transposed operator from (31), by not using column permutations of the variable vector ($\boldsymbol{R} = \boldsymbol{I}$) and by restricting column permutations to the coefficient matrix ($\boldsymbol{Q}$ contains an identity submatrix in its south–east corner)

$$R_m(\boldsymbol{P}\,[\,\boldsymbol{A}_{n\times n} \quad \boldsymbol{B}_{n\times c}\,]\,\boldsymbol{Q}^T)(\boldsymbol{Q}\begin{bmatrix}\boldsymbol{X}_{n\times c}\\ -\boldsymbol{I}_{c\times c}\end{bmatrix}) = 0 \tag{34}$$

where

$$\boldsymbol{Q} = \begin{bmatrix}\boldsymbol{Q}' & 0\\ 0 & \boldsymbol{I}_{c\times c}\end{bmatrix}. \tag{35}$$

5.1 The Circuit's Tableau

Let us begin by formulating the circuit's general tableau [4, 6] with a given excitation vector ($c = 1$). We prefer a simple version of the tableau, so we assume that energy storing elements have been replaced by adequate time invariant linear models. This way, the tableau's unknowns are only nodal voltages, branch voltages and branch currents (36)

$$\begin{bmatrix}\boldsymbol{A}^T & -\boldsymbol{I} & 0 & 0\\ 0 & 0 & \boldsymbol{A} & 0\\ 0 & \boldsymbol{Y} & \boldsymbol{Z} & \boldsymbol{e}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_n\\ \boldsymbol{v}_b\\ \boldsymbol{i}_b\\ -1\end{bmatrix} = 0 \tag{36}$$

where

$n+1$	number of nodes
b	number of branches
$\boldsymbol{v}_n$	nodal voltage vector
$\boldsymbol{v}_b$	branch voltage vector
$\boldsymbol{i}_b$	branch current vector
$\boldsymbol{e}$	excitation vector
$\boldsymbol{A}_{n\times b}$	reduced incidence matrix
$\boldsymbol{Y}_{b\times b}$	admittance coefficient matrix
$\boldsymbol{Z}_{b\times b}$	impedance coefficient matrix.

The first two rows of (36) represent the Kirchoff's voltage ($\boldsymbol{A}^T\boldsymbol{v}_n = \boldsymbol{v}_b$) and current ($\boldsymbol{A}\boldsymbol{i}_b = 0$) laws. Matrix $\boldsymbol{A}$ is the reduced node versus branch incidence matrix. The third row of (36) states the branch constitutive relations of the circuit.

In the following three subsections we will demonstrate how different equation sets can be obtained by transforming the basic tableau due to general operator equation (31).

5.2 Nodal Equations

An alternative to the so called 'sparse tableau approach', proposed by [4, 6] is of course nodal analysis. In the past, many applications of either have been benchmarked against each other yielding only insubstantial differences. In this subsection we shall attempt to analyze these two approaches by using the general operator equation to establish some relations between them.

Let us consider the following second order operator equation on the tableau

$$R_{2b}(R_b(\boldsymbol{P}\begin{bmatrix}\boldsymbol{A}^T & -\boldsymbol{I} & 0 & 0\\ 0 & 0 & \boldsymbol{A} & 0\\ 0 & \boldsymbol{Y} & \boldsymbol{Z} & \boldsymbol{e}\end{bmatrix}\boldsymbol{Q}^T))(\boldsymbol{Q}\begin{bmatrix}\boldsymbol{v}_n\\ \boldsymbol{v}_b\\ \boldsymbol{i}_b\\ -1\end{bmatrix}) = 0 \tag{37}$$

where

$$\boldsymbol{P} = \begin{bmatrix}\boldsymbol{I} & 0 & 0\\ 0 & 0 & \boldsymbol{I}\\ 0 & \boldsymbol{I} & 0\end{bmatrix}, \qquad \boldsymbol{Q} = \begin{bmatrix}0 & \boldsymbol{I} & 0 & 0\\ 0 & 0 & \boldsymbol{I} & 0\\ \boldsymbol{I} & 0 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}. \tag{38}$$

By evaluating (37) we get the block triangular form (39). The last row in (39) is easily identified as the circuit's well known nodal equation set, provided of course, that the inverse of $\boldsymbol{Z}$ exists

$$\begin{bmatrix}-\boldsymbol{I} & 0 & \boldsymbol{A}^T & 0\\ 0 & \boldsymbol{Z} & \boldsymbol{Y}\boldsymbol{A}^T & \boldsymbol{e}\\ 0 & 0 & -\boldsymbol{A}\boldsymbol{Z}^{-1}\boldsymbol{Y}\boldsymbol{A}^T & -\boldsymbol{A}\boldsymbol{Z}^{-1}\boldsymbol{e}\end{bmatrix}\begin{bmatrix}\boldsymbol{v}_b\\ \boldsymbol{i}_b\\ \boldsymbol{v}_n\\ -1\end{bmatrix} = 0. \tag{39}$$

Actually, (39) is a more complete way to express the nodal equations, since the nodal voltage vector $\boldsymbol{v}_n$ must be expanded to satisfy the branch relations, which are controlled by branch voltages and/or branch currents rather than by nodal voltages. Hence, the trivial operations of the back substitution in the first row of (39) must be performed in any case. The inversion of the second block ($\boldsymbol{Z}$) is also inevitable, since $\boldsymbol{Z}^{-1}$ appears in the nodal equation set below.

Hereby we have formally established the connection between the tableau and the nodal equations. Precisely the same transformation can be used without the assumption of $\boldsymbol{Z}$ being nonsingular, rendering a modified nodal equation set, very much like the one used in SPICE [7].

5.3 Cutset and Loop Equations

Cutset equations and their dual, the loop equations are another very basic approach to the problem of equation formulation. These two methods require the selection of a tree and are therefore considered unattractive for computer implementations. Instead of employing Kirchoff's current law to each node, in cutset and loop analysis the tree branches define the fundamental cutset, to which Kirchoff's current law is applied. Also, each link branch in combination with tree branches produces one fundamental loop to which Kirchoff's voltage law is enforced. This makes the nodal voltages of the tableau (36) superfluous.

We shall now attempt to transform the 'nodal' tableau (36) to a 'fundamental' tableau using reduction operator equations. The tree selection can be replaced by a permutation algorithm which splits the circuit's incidence matrix into a lower triangular tree submatrix $\boldsymbol{L}_t$ and the remaining link incidence matrix $\boldsymbol{A}_l$

$$\boldsymbol{P}\boldsymbol{A}\boldsymbol{Q}^T = [\,\boldsymbol{L}_t \quad \boldsymbol{A}_l\,]. \tag{40}$$

In a normal circuit, the triangular echelonization of its incidence matrix by permutation is always possible. Consequently, the entire tableau (36) is split in two ways

$$\begin{bmatrix} \boldsymbol{Q} & 0 & 0 \\ 0 & \boldsymbol{P} & 0 \\ 0 & 0 & \boldsymbol{Q} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}^T & -\boldsymbol{I} & 0 & 0 \\ 0 & 0 & \boldsymbol{A} & 0 \\ 0 & \boldsymbol{Y} & \boldsymbol{Z} & \boldsymbol{e} \end{bmatrix} \begin{bmatrix} \boldsymbol{P} & 0 & 0 & 0 \\ 0 & \boldsymbol{Q} & 0 & 0 \\ 0 & 0 & \boldsymbol{Q} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} \boldsymbol{P} & 0 & 0 & 0 \\ 0 & \boldsymbol{Q} & 0 & 0 \\ 0 & 0 & \boldsymbol{Q} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{v}_n \\ \boldsymbol{v}_b \\ \boldsymbol{i}_b \\ -1 \end{bmatrix} = 0. \quad (41)$$

By evaluating (41) and by using indices t and l for tree and link split matrices respectively, we arrive at

$$\begin{bmatrix} \boldsymbol{L}_t^T & -\boldsymbol{I} & 0 & 0 & 0 & 0 \\ \boldsymbol{A}_l^T & 0 & -\boldsymbol{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{L}_t & \boldsymbol{A}_l & 0 \\ 0 & \boldsymbol{Y}_t & 0 & \boldsymbol{Z}_t & 0 & \boldsymbol{e}_t \\ 0 & 0 & \boldsymbol{Y}_l & 0 & \boldsymbol{Z}_l & \boldsymbol{e}_l \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{v}}_n \\ \boldsymbol{v}_{bt} \\ \boldsymbol{v}_{bl} \\ \boldsymbol{i}_{bt} \\ \boldsymbol{i}_{bl} \\ -1 \end{bmatrix} = 0, \quad (42)$$

where $\hat{\boldsymbol{v}}_n = \boldsymbol{P}\boldsymbol{v}_n$. In (42) we have already imposed some restrictions upon the branch relation coefficient matrices $\boldsymbol{Y}$ and $\boldsymbol{Z}$, that is, controlled sources are not allowed between tree and link branches. At this point some trivial rearrangements of (42) and a series of reduction operators suffice to obtain the desired cutset equations

$$R_{2b}(R_{n+b}(R_{2n}(\dots R_{n+1}(R_n(\begin{bmatrix} \boldsymbol{L}_t^T & 0 & 0 & 0 & -\boldsymbol{I} & 0 \\ 0 & \boldsymbol{L}_t & \boldsymbol{A}_l & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{Z}_l & \boldsymbol{Y}_l & 0 & \boldsymbol{e}_l \\ \boldsymbol{A}_l^T & 0 & 0 & -\boldsymbol{I} & 0 & 0 \\ 0 & \boldsymbol{Z}_t & 0 & 0 & \boldsymbol{Y}_t & \boldsymbol{e}_t \end{bmatrix}))))) \begin{bmatrix} \hat{\boldsymbol{v}}_n \\ \boldsymbol{i}_{bt} \\ \boldsymbol{i}_{bl} \\ \boldsymbol{v}_{bl} \\ \boldsymbol{v}_{bt} \\ -1 \end{bmatrix} = 0. \quad (43)$$

For the sake of simplicity let us agree upon another simplification before evaluating (43). Cutset analysis is usually derived with the assumption that all branches in the circuit are current–defined and voltage–controlled, meaning $\boldsymbol{Z}_t = -\boldsymbol{I}$ and $\boldsymbol{Z}_l = -\boldsymbol{I}$. The evaluation of (43) also yields several occurrences of the expression $\boldsymbol{L}_t^{-1}\boldsymbol{A}_l$, which is indeed equal to the circuit's fundamental cutset matrix $\boldsymbol{F}$, usually obtained by heuristic methods. With this in mind

$$\left[\begin{array}{c|ccccc} \boldsymbol{L}_t^T & 0 & 0 & 0 & -\boldsymbol{I} & 0 \\ \hline 0 & \boldsymbol{D}_t & \boldsymbol{D}_t\boldsymbol{F} & 0 & 0 & 0 \\ 0 & 0 & -\boldsymbol{I} & \boldsymbol{Y}_l & 0 & \boldsymbol{e}_l \\ 0 & 0 & 0 & -\boldsymbol{I} & \boldsymbol{F}^T & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{Y}_t + \boldsymbol{F}\boldsymbol{Y}_l\boldsymbol{F}^T & \boldsymbol{e}_t + \boldsymbol{F}\boldsymbol{e}_l \end{array}\right] \left[\begin{array}{c} \hat{\boldsymbol{v}}_n \\ \hline \boldsymbol{i}_{bt} \\ \boldsymbol{i}_{bl} \\ \boldsymbol{v}_{bl} \\ \boldsymbol{v}_{bt} \\ -1 \end{array}\right] = 0. \quad (44)$$

The first matrix equation in (44) determines the relationship between the permuted nodal voltage vector and the tree branch voltage vector. Actually, this relation is not needed any longer and may be dropped from (44), which is marked by the horizontal and vertical lines. The remaining south eastern part of (44) is the sought fundamental tableau. Its first and third lines state the fundamental cutset and loop equation, respectively. Its second row gives the link branch constitutive relations, while the last matrix equation denotes the circuit's final cutset equations. At the same time, it is apparent, that (44) consists of a $2b$ upper triangular submatrix in its north–west corner, thus the first four equations present us with a

substitutional relation between the tree branch voltages $\boldsymbol{v}_{bt}$ (the independent variables) and all the other (dependent) circuit variables.

Let us put down the cutset equation system separately without the previously imposed restriction of $\boldsymbol{Z}_t$ and $\boldsymbol{Z}_l$

$$(\boldsymbol{Y}_t + \boldsymbol{Z}_t\boldsymbol{F}\boldsymbol{Z}_l^{-1}\boldsymbol{Y}_l\boldsymbol{F}^T)\,\boldsymbol{v}_{bt} = \boldsymbol{e}_t + \boldsymbol{Z}_t\boldsymbol{F}\boldsymbol{Z}_l^{-1}\boldsymbol{e}_l. \tag{45}$$

It is certainly not easy to recognize cutset equations in the expression above. In order to clarify (45), let us use the tableau's original notations (36) and expand (45)

$$\boldsymbol{Z}_t\,[\,\boldsymbol{I}\quad\boldsymbol{F}\,]\,\boldsymbol{Z}^{-1}\boldsymbol{Y}\,[\,\boldsymbol{I}\quad\boldsymbol{F}\,]^T\,\boldsymbol{v}_{bt} = \boldsymbol{Z}_t\,[\,\boldsymbol{I}\quad\boldsymbol{F}\,]\,\boldsymbol{Z}^{-1}\boldsymbol{e}. \tag{46}$$

Although it is not apparent from (46), please observe, that cutset equations are less restrictive by nature than nodal equations, since they demand nonsingularity only of $\boldsymbol{Z}_l$ in (45), where nodal equations require the entire $\boldsymbol{Z}$ to be non–singular (39).

As to loop analysis, it is needless to say, that it can be expressed by using exactly the same principles. We may start directly with (43), by using a different permutation of the tableau and a modified reduction operator series

$$R_{b+2n}(\dots R_{b+n+1}(R_{b+n}(R_b(R_n(\begin{bmatrix} \boldsymbol{L}_t^T & 0 & -\boldsymbol{I} & 0 & 0 & 0 \\ \boldsymbol{A}_l^T & -\boldsymbol{I} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{Y}_t & \boldsymbol{Z}_t & 0 & \boldsymbol{e}_t \\ 0 & 0 & 0 & \boldsymbol{L}_t & \boldsymbol{A}_l & 0 \\ 0 & \boldsymbol{Y}_l & 0 & 0 & \boldsymbol{Z}_l & \boldsymbol{e}_l \end{bmatrix}))))) \begin{bmatrix} \hat{\boldsymbol{v}}_n \\ \boldsymbol{v}_{bl} \\ \boldsymbol{v}_{bt} \\ \boldsymbol{i}_{bt} \\ \boldsymbol{i}_{bl} \\ -1 \end{bmatrix} = 0. \tag{47}$$

Eventually, the evaluation of (47) delivers loop equations (48), where (49) is their more commonly known notation

$$(\boldsymbol{Z}_l + \boldsymbol{Y}_l\boldsymbol{F}^T\boldsymbol{Y}_t^{-1}\boldsymbol{Z}_t\boldsymbol{F})\,\boldsymbol{i}_{bl} = \boldsymbol{e}_l - \boldsymbol{Y}_l\boldsymbol{F}^T\boldsymbol{Y}_t^{-1}\boldsymbol{e}_t \tag{48}$$

$$\boldsymbol{Y}_l\,[-\boldsymbol{F}^T\quad\boldsymbol{I}\,]\,\boldsymbol{Y}^{-1}\boldsymbol{Z}\,[-\boldsymbol{F}^T\quad\boldsymbol{I}\,]^T\,\boldsymbol{i}_{bl} = \boldsymbol{Y}_l\,[-\boldsymbol{F}^T\quad\boldsymbol{I}\,]\,\boldsymbol{Y}^{-1}\boldsymbol{e}. \tag{49}$$

The duality of expressions (45), (46) and (48), (49) respectively is apparent. Even the permutations which relate (43) to (47) are symmetric, so the relationship between cutset and loop analysis is hereby demonstrated most evidently.

5.4 The Circuit's Reduced Equations

As we have seen in all three basic equation formulations, nodal, cutset, and loop equations, the large variable vector of the tableau (36) has been divided into a small 'independent' variable vector ($\boldsymbol{v}_n$, $\boldsymbol{v}_{bt}$ and $\boldsymbol{i}_{bl}$ respectively) and a relatively large 'dependent' variable vector, which would include all the remaining variables. The solution of the actual equation set produces each time the values of the independent variables. If the values of the dependent variables are needed, they have to be obtained by back substitution in (39), (44) and (47) respectively, which is why we prefer to call them dependent variables.

In this section we shall follow this concept to the extreme edge and employ the sparse matrix reduction technique (SMaRT), which was introduced by M. Vehovec and F. Bratkovič in 1977 [9]. The underlying idea is to select a subset of circuit variables many times smaller than the number of nodes in the circuit. This is done by permutation of the tableau (36)

according to the pattern of nonzero elements in the coefficient matrix, yielding a large upper triangular submatrix $\boldsymbol{U}_{m\times m}$ in its north–west corner. Then a single reduction operator of rank m is applied. The permutation is denoted by $\boldsymbol{P}$ and $\boldsymbol{Q}$

$$R_m(\boldsymbol{P}\begin{bmatrix} \boldsymbol{A}^T & -\boldsymbol{I} & 0 & 0 \\ 0 & 0 & \boldsymbol{A} & 0 \\ 0 & \boldsymbol{Y} & \boldsymbol{Z} & \boldsymbol{e} \end{bmatrix}\boldsymbol{Q}^T)(\boldsymbol{Q}\begin{bmatrix} \boldsymbol{v}_n \\ \boldsymbol{v}_b \\ \boldsymbol{i}_b \\ -1 \end{bmatrix}) = 0 \tag{50}$$

$$R_m(\begin{bmatrix} \boldsymbol{U} & \boldsymbol{A}_{1I} & \boldsymbol{b}_1 \\ \boldsymbol{A}_{2D} & \boldsymbol{A}_{2I} & \boldsymbol{b}_2 \end{bmatrix})\begin{bmatrix} \boldsymbol{x}_D \\ \boldsymbol{x}_I \\ -1 \end{bmatrix} = 0. \tag{51}$$

Unlike the previously mentioned cases, the independent variables of this equation set are not predictable, since the permutation does not keep track of the individual coefficient types. The only permutation criterion is to achieve the largest possible upper triangular submatrix. For this reason we use a different notation in (51). The coefficient matrix is split in two ways, namely, the dependent part (denoted by index D) and the independent portion (denoted by index I). We cannot anticipate, which circuit variables are going to end up in the independent variable vector $\boldsymbol{x}_I$ and which are the ones to become dependent variables $\boldsymbol{x}_D$.

Numerous tests [12, 14] have shown, that it is possible to permute a circuit's tableau to accommodate a triangular submatrix as large as 98% of the tableau's size. In other words: the independent variable vector $\boldsymbol{x}_I$ is as up to 10 times smaller than the number of nodes in the circuit. With this in mind let us execute the reduction operator of rank m on the permuted equation set mapping it into

$$\begin{bmatrix} \boldsymbol{U} & \boldsymbol{A}_{1I} & \boldsymbol{b}_1 \\ 0 & \boldsymbol{A}_{2I}-\boldsymbol{A}_{2D}\boldsymbol{U}^{-1}\boldsymbol{A}_{1I} & \boldsymbol{b}_2-\boldsymbol{A}_{2D}\boldsymbol{U}^{-1}\boldsymbol{b}_1 \end{bmatrix}\begin{bmatrix} \boldsymbol{x}_D \\ \boldsymbol{x}_I \\ -1 \end{bmatrix} = 0. \tag{52}$$

The first row of (52) is a simple back substitution relating the dependent variables to the independent ones. Please remember, that there were no arithmetic operations involved so far, hence there has been no numerical error and certainly no fill–in.

The second row in (52) is a strongly reduced equation set, its formulation necessitating the inversion of the large triangular submatrix $\boldsymbol{U}$. Fortunately, the inverse is premultiplied by the extremely 'thin' matrix $\boldsymbol{A}_{2D}$. This situation resembles (54) very much and is actually computed via a set of residual vectors [9], which are obtained by multiple back substitution. The resulting equation set is extremely compact

$$\boldsymbol{C}\boldsymbol{x}_I = \boldsymbol{d}, \tag{53}$$

yet the coefficient matrix $\boldsymbol{C}$ is still sparse. Furthermore, $\boldsymbol{C}$ is quasi upper triangular, meaning its lower triangular portion is significantly sparser than the upper triangular one.

6 Solving Sparse Equation Sets

In the following subsections we shall see, how operator equation (34) can be used to express equation set solution techniques. Let us start with the principles of variable substitution.

6.1 The Substitution

In case where the coefficient matrix is a lower triangular matrix $\boldsymbol{L}$ of rank n, the variable vector can be obtained by successive variable substitution. The first component is determined by dividing the right hand side by the first pivot. All subsequent components are calculated by replacing all unknowns with the values of the previously computed variables. This way we have one equation with one unknown in each step. This process can be formally denoted by a series of successive reduction operators

$$[\,\boldsymbol{L}_{n\times n} \quad \boldsymbol{B}\,]\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0 \tag{54}$$

$$R_{n-1}(\ldots R_2(R_1([\,\boldsymbol{L} \quad \boldsymbol{B}\,]))\ldots)\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0 \tag{55}$$

$$[\,\boldsymbol{D} \quad \tilde{\boldsymbol{B}}\,]\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0. \tag{56}$$

By applying operator equation (55), the coefficient matrix is diagonalized according to (27), at the same time the right hand side $\boldsymbol{B}$ is transformed to $\tilde{\boldsymbol{B}}$. The resulting equation set (56) is only a trivial step away from the final solution $\boldsymbol{X} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{B}}$.

Please consider that the substitution as presented in (55) is proceeding in a 'column–wise' rather than the usual 'row–wise' manner. This distinction may have some influence on a computer implementation regarding the data structure and the memory organization, but has no effect whatsoever on the numerical outcome of the substitution. Operation count and numerical error are not affected by the succession, in which the arithmetic operations are carried out.

Since the back substitution is very similar to the forward substitution, let us just briefly mention its representation by reduction operators. The back substitution implies, that the coefficient matrix is upper triangular $\boldsymbol{U}$. A simple permutation however suffices to transform any upper triangular matrix to a lower triangular one. Then the operator series (55) can be used.

6.2 The Gaussian Elimination

The Gaussian elimination consists of two courses: the forward course (sometimes also called the elimination course), in which the coefficient matrix is triangularized and the back substitution, which is used to obtain the solution vectors. It is very interesting, that the forward course is described exactly by the same operator series, as it was used for the substitution (55)

$$[\,\boldsymbol{A}_{n\times n} \quad \boldsymbol{B}\,]\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0 \tag{57}$$

$$R_{n-1}(\ldots R_2(R_1([\,\boldsymbol{A} \quad \boldsymbol{B}\,]))\ldots)\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0 \tag{58}$$

$$[\,\boldsymbol{U} \quad \tilde{\boldsymbol{B}}\,]\begin{bmatrix}\boldsymbol{X}\\ -\boldsymbol{I}\end{bmatrix} = 0. \tag{59}$$

The only difference is the coefficient matrix being nontriangular. The transformed equation set consists of an upper triangular coefficient matrix (59) and is solved by the back

substitution course

$$[\boldsymbol{U}_{n\times n} \quad \tilde{\boldsymbol{B}}] \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0 \tag{60}$$

$$R_{n-1}(\ldots R_2(R_1(\boldsymbol{P}[\boldsymbol{U} \quad \tilde{\boldsymbol{B}}]\boldsymbol{Q}^T))\ldots)\boldsymbol{Q} \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0 \tag{61}$$

$$\begin{bmatrix} \boldsymbol{D} & \tilde{\tilde{\boldsymbol{B}}} \end{bmatrix} \boldsymbol{Q} \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0, \tag{62}$$

where $\boldsymbol{P}$ and $\boldsymbol{Q}$ are antidiagonal unit matrices, which are used to reverse the row and column order of $\boldsymbol{U}$, making it lower triangular.

Many authors [5] have proposed different modifications of the basic elimination course. These modifications however suggest only a different order, in which exactly the same arithmetic operations are performed. It is not the issue of this article to deal with storage use, we are only concerned with modifications which affect numerical results.

6.3 Direct Triangular Decomposition

By triangular decomposition the coefficient matrix $\boldsymbol{A}$ is expressed as the product of a lower triangular matrix $\boldsymbol{L}$ with an upper triangular matrix $\boldsymbol{U}$

$$\boldsymbol{LU} = \boldsymbol{A}. \tag{63}$$

The factored form (63) is then used to solve associated linear equation sets by forward and backward substitution. In this subsection we will be concerned only with the determination of $\boldsymbol{L}$ and $\boldsymbol{U}$.

For this purpose expression (17) is very convenient, for $\boldsymbol{A}$ is equal to the product of three matrices. The first one is a lower block triangular matrix $R_n^T(\boldsymbol{A})$, the second is a block diagonal matrix $R_n^T(R_n(\boldsymbol{A}))^{-1}$ and finally there is an upper block triangular matrix $R_n(\boldsymbol{A})$. If equality (17) is true for all n, then it is true for any reduction operator series as well. Let us therefore apply the triangularization series (26), the diagonalization (27) and the triangularization (25) to the first, second and third factor of (17) respectively,

$$\begin{aligned} \boldsymbol{L}' &= R_{n-1}^T(\ldots R_2^T(R_1^T(\boldsymbol{A}))\ldots) \\ \boldsymbol{D} &= R_{n-1}(\ldots R_2(R_1(\boldsymbol{L}'))\ldots) \\ \boldsymbol{U} &= R_{n-1}(\ldots R_2(R_1(\boldsymbol{A}))\ldots). \end{aligned} \tag{64}$$

Relation (17) is thereby rephrased to

$$(\boldsymbol{L}'\boldsymbol{D}^{-1})\boldsymbol{U} = \boldsymbol{A}, \tag{65}$$

where the expression in brackets is the sought lower triangular factor of (63), namely $\boldsymbol{L} = \boldsymbol{L}'\boldsymbol{D}^{-1}$.

6.4 The Gauss–Jordan Elimination

The Gauss–Jordan elimination consists of only one forward course rather than the usual forward and backward course of the basic Gaussian elimination. The coefficient matrix is diagonalized in a single forward course, which is very convenient for practical applications,

although not necessary, since the operations on the upper triangular part are perfectly independent of the ones in the lower triangular part.

In order to understand better the relations to other methods, we prefer to split this forward course into two separate forward courses, from which the first one shall be identical with the forward course of the basic Gaussian elimination (58). The second course, on the other hand, is somehow similar to the back substitution (61), but proceeds in the opposite direction so, the reduction operator series shall be simply reversed

$$[\boldsymbol{U}_{n\times n} \quad \tilde{\boldsymbol{B}}] \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0 \tag{66}$$

$$R_1(R_2(\dots R_{n-1}(\boldsymbol{P}\,[\boldsymbol{U} \quad \tilde{\boldsymbol{B}}]\,\boldsymbol{Q}^T)\dots))\boldsymbol{Q} \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0 \tag{67}$$

$$\begin{bmatrix} \boldsymbol{D} & \tilde{\tilde{\boldsymbol{B}}} \end{bmatrix} \boldsymbol{Q} \begin{bmatrix} \boldsymbol{X} \\ -\boldsymbol{I} \end{bmatrix} = 0. \tag{68}$$

As we well know from (13), a reversed order of reduction operators has no effect on the outcome of the transformation, hence, in a mathematical sense, (68) is identical to (62). The arithmetic operations involved, however, are very different in quality as well as in quantity. The back substitution constitutes of $n-1$ ascending reduction operators and consequently (24) requires the inversion of $n-1$ scalars. The application of the same reduction operators in a descending succession (67) on the other hand, demands the inversion of $n-1$ triangular submatrices. It is clear, that the inversion of an m–dimensional triangular matrix takes considerably more arithmetic operations than the inversion of m scalars. So, operator equation (67) distinctly shows a well known disadvantage of the Gauss–Jordan elimination scheme.

7 Conclusions

The four matrix manipulations, which are mentioned in the introduction, have been combined in a single general operator equation (31). This general operator equation is composed of row and column permutations, a reduction operator (6) and a transposed reduction operator (14). The two operators actually describe a generalization of the second couple of manipulations, that is, the addition of a linear row combination and column combination respectively.

It has been established, that this combination suffices to describe most analysis methods, which is demonstrated on two groups of practical examples. First, equation set formulation techniques are classified in section 5 and secondly equation set solution methods are investigated in section 6. In order to validate the quality of the presented mathematical tools, classic methods have been selected, as their solution is more or less known in advance [1, 2, 3, 5, 6].

Once the basic matrix methods are classified, it is easy to express entire circuit analysis concepts. Let us try to phrase the SPICE–approach in terms of reduction operators.

First, an adaptation of operator equation (37) is used to produce the circuits modified nodal equations. In fact, SPICE composes its nodal equation set 'directly' from the input, nevertheless exactly the same arithmetical operations are carried out, as they would be compulsory to evaluate the last row of (39). So, just for the sake of argument, we could well say, that the two reduction operators from (37) are applied. Next, the coefficient matrix is LU decomposed due to operator series (64) and the independent solution vector is obtained by one forward and one backsubstitution (55). Finally, the dependent circuit variables are computed

by backsubstitution in (39). Again, SPICE documentation never uses the expression 'backsubstitution' in this connection, since the dependent variables are obtained implicitly during output processing and/or during the bias point calculations of nonlinear devices [7, 11]. Yet, the operations involved are exactly the same as the ones required by operator series (55), when enforced upon the first two rows of (39).

It is common knowledge, that the equation formulation and equation solution methods must be well tuned to each other. A reordering algorithm during LU decomposition, for example, should try to preserve as much of the nodal equation's symmetry as possible. At this point, we dare to go one step further in this direction by stating, that equation formulation and its subsequent solution are just two names for two phases of a single process. In order to support this thesis we have silently introduced sparse matrix reduction technique as an equation formulation principle in section 5, in spite of the fact, that SMaRT was originally introduced as an equation solution technique [9]. Another example speaking in favour of this thesis is presented in [6, 15] and demonstrates, how Gaussian elimination can be used to transform the circuit tableau to nodal equations.

The formalisms proposed in this paper can be under certain circumstances also recommended for educational purposes. Though it is certainly easier for a novice student to understand e.g. the Gaussian elimination, if it is presented in form of a descriptive algorithm rather than as a reduction operator equation (58). Still, it might be worth the extra effort, when a comparison or classification of different methods is attempted later.

8 Appendix A

Proof 1 (equation 8) *Let $\boldsymbol{U}$ denote any upper triangular matrix $\boldsymbol{u}_{ij} = 0\ \forall i > j$ from the domain of the reduction operator $\boldsymbol{U} \in \mathcal{D}(R_n)$. Then, by definition (6), the application of R_n to $\boldsymbol{U}$ yields*

$$R_n(\boldsymbol{U}) = R_n(\begin{bmatrix} \boldsymbol{U}_{11} & \boldsymbol{U}_{12} \\ 0 & \boldsymbol{U}_{22} \end{bmatrix}) \triangleq \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{U}_{11} & \boldsymbol{U}_{12} \\ 0 & \boldsymbol{U}_{22} \end{bmatrix} = \boldsymbol{U} \tag{69}$$

■

Proof 2 (equation 9) *Consider any $\boldsymbol{A}$ from the domain $\boldsymbol{A} \in \mathcal{D}(R_n)$ premultiplied by any nonzero scalar $\alpha \neq 0$. Reducing the product by R_n we get*

$$R_n(\alpha\boldsymbol{A}) \triangleq \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ -\alpha\boldsymbol{A}_{21}(\alpha\boldsymbol{A}_{11})^{-1} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \alpha\boldsymbol{A}_{11} & \alpha\boldsymbol{A}_{12} \\ \alpha\boldsymbol{A}_{21} & \alpha\boldsymbol{A}_{22} \end{bmatrix} = \alpha \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix} = \alpha\, R_n(\boldsymbol{A}). \tag{70}$$

■

Proof 3 (equation 10) *Let $\boldsymbol{A}$ be in the domain of R_n. Since the range of reduction operators is a subset of their domain $\mathcal{R}(R_n) \subset \mathcal{D}(R_n)$, it is possible to apply two identical reduction operators to $\boldsymbol{A}$*

$$R_n(R_n(\boldsymbol{A})) \triangleq R_n(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix}) \triangleq \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ 0 & \tilde{\boldsymbol{A}}_{22} \end{bmatrix} = R_n(\boldsymbol{A}). \tag{71}$$

The idempotence of reduction operators is hereby established.

■

Proof 4 (relation 11) *Please observe matrix equation $\boldsymbol{AB} = 0$, where $\boldsymbol{A}$ is in the domain of R_n and $\boldsymbol{B}$ is an arbitrary matrix (72). Any matrix equation may be premultiplied by an arbitrary nonsingular matrix without disturbing the equality [2]. The application of the reduction operator R_n to $\boldsymbol{A}$ actually implies a premultiplication of the equation's left–hand side. The right–hand side constitutes of a null matrix, ergo it is not sensitive to multiplication (73).*

$$\boldsymbol{AB} = 0 \tag{72}$$

$$\begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ -\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \boldsymbol{B} = \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ -\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{I} \end{bmatrix} 0 \tag{73}$$

$$R_n(\boldsymbol{A})\boldsymbol{B} = 0 \tag{74}$$

In order to establish the truth of (74) it is necessary to show, that the premultiplication matrix in (73) is regular (nonsingular). A necessarily square matrix is nonsingular iff its necessarily unique reciprocal exists. In our case, all these conditions are met for

$$\begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ -\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_{n\times n} & 0 \\ \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} & \boldsymbol{I} \end{bmatrix} = \boldsymbol{I}. \tag{75}$$

■

Proof 5 (equation 12) *Let $\boldsymbol{A}$ be a square matrix belonging to the domain of R_n. The determinant of $\boldsymbol{A}$ remains unchanged, if a linear combination of some rows is added to another row [2]. We will now show, that the reduction operator just adds a specific linear combination of some rows to the remaining rows.*

Consider (6) and substitute $-\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}$ for $\boldsymbol{X}$. The upper part of $\boldsymbol{A}$ is premultiplied by $\boldsymbol{X}$ and added to the lower part. Obviously, we are facing a linear combination on a 'matrix level'. In order to prove that this is a linear combination of rows, we use a row–wise notation of $\boldsymbol{A}$

$$\boldsymbol{A} = [\, \boldsymbol{a}_1^T \quad \boldsymbol{a}_2^T \quad \dots \quad \boldsymbol{a}_n^T \quad \boldsymbol{a}_{n+1}^T \quad \dots \quad \boldsymbol{a}_r^T \,]^T, \tag{76}$$

where $\boldsymbol{a}_j^T$ is a vector, equal to the j–th row of $\boldsymbol{A}$. According to (6) the reduction of $\boldsymbol{A}$ yields

$$\begin{aligned} R_n(\boldsymbol{A}) &= [\, \boldsymbol{a}_1^T \quad \boldsymbol{a}_2^T \quad \dots \quad \boldsymbol{a}_n^T \quad \tilde{\boldsymbol{a}}_{n+1}^T \quad \dots \quad \tilde{\boldsymbol{a}}_r^T \,]^T \\ \tilde{\boldsymbol{a}}_{n+i}^T &= \boldsymbol{a}_{n+i}^T + \sum_{j=1}^{n} \boldsymbol{a}_j^T x_{i,j} \quad \forall i = 1, 2 \dots (r-n). \end{aligned} \tag{77}$$

In (77), the reduction operator is decomposed to a series of linear row–combinations. Hence the matrix's determinant is not changed by reduction operators.

■

Proof 6 (equation 13) *Let $\boldsymbol{A}$ belong to $\mathcal{D}(R_m) \cap \mathcal{D}(R_n)$, where $n > m$. The reduction $R_n(R_m(\boldsymbol{A}))$ has already been dealt with in section 3.2, and leads to the right–hand side of (20). We shall now show, that $R_m(R_n(\boldsymbol{A}))$ produces the same result, starting with a similar notation as in (18)*

$$R_m(R_n(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ \boldsymbol{A}_{31} & \boldsymbol{A}_{32} & \boldsymbol{A}_{33} \end{bmatrix})) = R_m(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ 0 & 0 & \tilde{\boldsymbol{A}}_{33} \end{bmatrix}), \tag{78}$$

where $\tilde{\tilde{\boldsymbol{A}}}_{33}$*, according to (7), is*

$$\tilde{\tilde{\boldsymbol{A}}}_{33} = \boldsymbol{A}_{33} - [\, \boldsymbol{A}_{31} \quad \boldsymbol{A}_{32} \,] \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{A}_{13} \\ \boldsymbol{A}_{23} \end{bmatrix}. \tag{79}$$

By fully evaluating the right–hand sides of (21) and (79), their equality can be confirmed. As to R_m*, in (78), it is evident from (7) that it will affect only* $\boldsymbol{A}_{22}$ *and* $\boldsymbol{A}_{23}$

$$R_m(R_n(\begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \boldsymbol{A}_{23} \\ \boldsymbol{A}_{31} & \boldsymbol{A}_{32} & \boldsymbol{A}_{33} \end{bmatrix})) = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} & \boldsymbol{A}_{13} \\ 0 & \tilde{\boldsymbol{A}}_{22} & \tilde{\boldsymbol{A}}_{23} \\ 0 & 0 & \tilde{\tilde{\boldsymbol{A}}}_{33} \end{bmatrix}), \tag{80}$$

where $\tilde{\boldsymbol{A}}_{22}$ *and* $\tilde{\boldsymbol{A}}_{23}$ *are defined according to (19).*

■

References

[1] H. M. Markowitz, "The Elimination Form of Inverse and its Application to Linear Programming", *Management Sci.*, vol. 3, pp. 255-269, 1957.

[2] G. A. Korn, T. M. Korn, *Mathematical Handbook for Scientists and Engineers*, McGraw–Hill Book Company, London, 1961.

[3] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

[4] G. D. Hachtel, R. K. Brayton, F. G. Gustavson, "The Sparse Tableau Approach to Network Analysis and Design", *IEEE Transactions on Circuit Theory*, vol. CT-18, no. 1, 1971.

[5] R. P. Tewarson, *Sparse Matrices*, Academic Press, New York, NY, 1973.

[6] L. O. Chua, P. M. Lin, *Computer–Aided Analysis of Electronic Circuits*, Prentice–Hall, Englewood Cliffs, NJ, 1975.

[7] L. W. Nagel, "A Computer Program to Simulate Semiconductor Circuits", *Memorandum no. ERL-M5520*, University of California, Berkeley, CA, 1975.

[8] U. Schendel, *Sparse–Matrizen*, R. Oldenbourg Verlag, München, Wien, 1977.

[9] M. Vehovec, F. Bratkovič, "On Methods of Network Analysis with Small Number of Independent Variables", *Proceedings 20th Midwest Symposium on Circuits and Systems*, Lubbock, TX, 1977.

[10] M. J. C. Grover, *Applications of Matrix Theory*, Clarendon Press, Oxford, 1989.

[11] T. L. Quarles, "The Spice3 Implementation Guide", *Memorandum no. ERL-M89/44*, University of California, Berkeley, CA, 1989.

[12] F. Bratkovič, T. Tuma, "Applicability of Sparse Matrix Reduction to Different Types of Circuit Equations", *Proceedings ISYNT-89*, page 178–181, Zagreb, 1989.

[13] T. A. Grandine, *The Numerical Methods Programming Projects Book*, Oxford University Press, New York, NY, 1990.

[14] T. Tuma, F. Bratkovič, "Improved Permutation and Fast Back Substitution in Sparse Matrix Reduction", *Proceedings ECCTD-91*, page 294–303, Copenhagen, 1991.

[15] T. Tuma, F. Bratkovič, "A Mathematical Model for Network Analysis Methods", *Proceedings ECCTD-93*, page 397–402, Davos, 1993.