Solving the circuit equations using tensor decompositions

V N Lantsov
Computer Engineering and System Control Department, Vladimir State University named after Alexander and Nikolay Stoletovs, Vladimir, Gorky Str., 87, 600000, Russian Federation

E-mail: lantsov@ieee.org

Abstract. Tensor decomposition methods are widely used for solving high-dimensional and multi-dimensional problems. These approaches have a huge potential for reducing computational costs in the simulation of electronic circuits and in the CAD systems for electronics. Simulation of modern radio-frequency (RF) circuits, especially for RF integrated circuits, is one of the areas of extensive scientific research and development. The main problem of analysis methods for circuit design of RF electronics is the significant computational costs associated with the high dimensions to be solved. This paper shows that the computational costs of circuit design are largely determined by the costs of solving a system of linear algebraic equations (SLAE). The problems of the solving of SLAE are indicated in the paper. Further development of methods for solving and reducing computational costs requires the use of new approaches, one of which is proposed in this paper. The article presents the developed algorithm for converting a second-order matrix into a higher-order tensor. In this paper algorithms and software tools for tensor decompositions, as well as methods for solving SLAE in the representation of equations in tensor format are investigated. The results of numerical experiments and the results of comparison with traditional methods are presented.

1. Introduction
In recent years, tensor decomposition methods have been widely used for solving high-dimensional and multi-dimensional problems [1]. This is primarily due to the success in developing numerical methods for tensor decompositions [2]. Tensor decomposition methods can significantly reduce the memory for storing model equations, as well as reduce the calculation time [3, 4]. One of the first developments on the use of tensors for circuit simulation was the work [4], where a method for analysing nonlinear circuits based on tensor decomposition was presented. This method made it possible to use the higher-order nonlinearities, and the problem of modelling with higher accuracy while reducing the calculation time and significantly reducing the required computer memory was solved [4].

Simulation of modern radio-frequency (RF) circuits, especially for RF integrated circuits, is one of the areas of extensive scientific research and development [5]. The main problem of circuit simulation methods in CAD for electronics is the significant computational costs (calculation time and required memory) associated with the high dimensions of the problems to be solved [6].

Modern software tools for circuit design of CAD electronics include the following main types of circuit analysis [6]: static mode analysis (direct currents analysis, DC), small-signal analysis (linear analysis), transient analysis (time domain analysis), nonlinear distortion analysis (based on Volterra series), harmonic balance analysis, and a number of others [6].
It is important to note that in all of the above types of analyses, the main operation \([5, 6]\) is the solution of a system of linear algebraic equations (SLAE). Therefore, the computational cost of circuit design is largely determined by the cost of solving the SLAE in the form (1)

\[
A \mathbf{x} = \mathbf{b},
\]

where the dimension of the matrix \(A\) is \([N \times N]\). Number of equations (unknowns) \(N\) can reach hundreds of thousands or more.

Two classes of methods have been used to calculate SLAE in RF CAD systems for circuit design: direct and iterative methods \([5, 6]\). For classical direct methods of solving such as the Gaussian elimination method (software tools often use its modification, LU decomposition), the costs are proportional to \(N^3\). In recent years, iterative methods based on Krylov subspaces have been used for large dimensional tasks, and the GMRES algorithm is widely known \([7]\). The computational cost for iterative methods is proportional to \(N^2\). Unfortunately, for complex circuits with almost densely filled matrices (especially for the Jacobian), Krylov method have many problems with orthogonalization \([5]\). Therefore, further development of methods for solving and reducing computational costs requires the use of new approaches, one of which is proposed in this paper and is associated with the use of tensor decompositions.

2. Tensor decompositions

A \(d\)-rank tensor defined as \(\mathbf{A} \in R^{n_1 \times n_2 \times \cdots \times n_d}\) \([4]\). Figure 1 illustrates an example of a third-rank tensor for \(3 \times 4 \times 2\).

![Figure 1. Example of a third-rank tensor \((3 \times 4 \times 2)\) \([4]\).](image)

There are many methods of tensor decomposition, the most commonly used are the Tucker decomposition method and the "tensor train" decomposition method \([3]\). In our work, we used the tensor train decomposition (or as it is often called “tensor train format") \([8]\).

**Tensor Train Decomposition** (TT decomposition). In this method, a tensor of order \(d\) is approximated by two second-order tensors (matrices) and \((d - 2)\) 3-dimensional tensors connected in series or in the form of a train \([8, 9]\). In the TT-format, the tensor \(\mathbf{A}\) will be determined by the following formula

\[
\mathbf{A}(i_1, ..., i_d) = G_1(i_1) ... G_d(i_d),
\]

where \(G_k(i_k)\) is a matrix (the kernel of the TT representation) of dimension \(r_k \times r_k\) for each fixed value of \(i_k\), with \(1 \leq i_k \leq n_k\). In order to obtain matrix-vector products in the form of a scalar, the following boundary conditions are necessary \(r_0 = r_d = 1\). If \(r_k \leq r\), \(n_k \leq n\) then the memory required to store the tensor in TT-format will be \(\leq d n r^2\). If \(r\) is small enough, then the required memory will be significantly less than for storing the full matrix, \(n^d\).
3. **Algorithm for converting the matrices to tensors**

To solve SLAE using tensor decompositions, we need to develop an algorithm for converting a high-dimensional matrix into a low-rank tensor, then we need to develop or choose an existing method for solving SLAE when presenting equations will be in tensor format.

To solve the problem of converting matrices to tensors, we used the binary representation of the matrix indices in different number systems. So, when binary system is used the square matrix $A$ with $N^2$ elements changed to matrix $B$ with $2^{2D}$ elements, where $D$ is the order of the tensor. To find the order of the tensor, we can use the equation

$$D = \log_2 N.$$  

(3)

To convert the original matrix $A$ to the tensor format, we need to use the binary decomposition algorithm. Any element of the original matrix $A$ has two indices $i$ and $j$. The performance of this element in the binary system will be:

$$A(i,j) = A_{bin}(i_1, i_2, ..., i_D; j_1, j_2, ..., j_D),$$  

(4)

where $A_{bin}(i,j)$ element of the original matrix; $i$ is the index of the row; $j$ is the column index; $i_1, i_2, ..., i_D$ – binary index of the row; $j_1, j_2, ..., j_D$ – binary index of the column.

In the new structure (tensor), the location of any element will now be determined by binary indexes. Below is the algorithm for implementing the matrix-to-tensor transformation procedure in the MATLAB system.

**Algorithm 1** (binary transformation class of the matrix Ten2)

1. function $obj = Ten2(A)$
2. % Initial settings
3. $N = size(A);$ % Matrix size
4. $Nm = max(N);$ % Maximum size value
5. % Binary translation
6. $Nmb = de2bi(Nm);$ % Returns an array
7. $Cv = zeros(1, length(Nmb));$
8. $Cv(length(Nmb)) = 1;$
9. if ($Cv == Nmb$);
10. obj.$X = Nm;$
11. else
12. obj.$X = 2^{length(Nmb)}$;
13. end
14. % Calculating the function
15. obj.$B = zeros(obj.X, obj.X);$  
16. obj.$B(1:N(1), 1:N(2)) = A;$
17. obj.$B = reshape(obj.$B', 1, []).$  
18. function $r = getvalue(obj, i)$
19. $r = obj.B(bin2dec(mat2str(i)) + 1)$;
20. end

4. **Algorithms of solving SLAE in tensor format**

After we transform (decompose) the tensor into simpler elements, we get a linear system in tensor format. At the same time, in the tensor format, we must also represent the right part of the SLAE. One of the first works on solving SLAE in tensor format was the paper [8], where an algorithm for solving a linear system based on the so-called DMRG (Density Matrix Renormalization Group) scheme, which was used earlier in solid-state physics for solving eigenvalue problems, was proposed.
The problem of solving the representation of SLAE in the TT-format is formulated as the problem of minimizing the next functional

\[ \| y - Ax \| \rightarrow \min . \] (5)

Unfortunately, the reliability of the DMRG method was not so high and not all problems were solved. In [9], another more reliable computational method, AMEn (Alternating Minimal Energy), was proposed.

The AMEn computational method combines the strengths of both the DMRG, algorithm of variable direction for tensors optimization, and the classical iterative methods [10]. In the AMEn method, during the DMRG step, the tensor format of the solution is expanded using the tensor format of the approximate residual iterative methods. This provides mutual support for classical Krylov’s type iterative methods and the DMRG method of variable directions. The AMEn method has proven geometric convergence with respect to the elements of the tensor [10].

The AMEn method is implemented in the TT Toolbox package developed by the Russian team [11]. The TT-Toolbox package is implemented as a Toolbox for MATLAB system and includes all the basic operations with tensors, is freely distributed, and is well documented.

To demonstrate the methods and test them, we first considered a simple example of SLAE of the form \( Ax = b \), where the matrix \( A \) with dimension \([16 \times 16]\) is shown in figure 2. The vector of the right side of \( b \) was given as a unit vector (i.e. all values were equal to 1). Matrix \( A \) and vector \( b \) were then converted to TT-format. The SLAE solution in TT-format is performed using the “amen_solve2” procedure of the TT-Toolbox package, which implements the AMEn method. The result of the solution \( x \) is shown in figure 3.

![Figure 2. Matrix \( A \) of the system of linear equations.](image)

![Figure 3. The result of the solution \( x \).](image)

The methodology and algorithms are tested using standard procedures built into the MATLAB package that implement the Gaussian elimination method and GMRES procedures. The results were completely consistent. To determine the effectiveness of the new algorithms in solving systems of linear equations of higher dimension, an example of solving SLAE with the number of equations 512 is considered. The comparison was also performed with the Gaussian procedure and GMRES. The experimental study showed the effectiveness of the proposed algorithms and methods in terms of computational costs.
5. Conclusion

Computational cost of circuit design is largely determined by the cost of solving a system of linear algebraic equations (SLAE). The new algorithms are suitable for solving large-dimensional SLAE were proposed. Results of numerical experiments and results of comparison with traditional methods are presented.

References

[1] Zare A, Ozdemir A, Iwen M A and Aviyente S 2018 Proc. IEEE 106 1341
[2] Zhang Z, Batselier K, Liu H, Daniel L and Wong N 2017 IEEE Trans. Comp.-Aided Design Int, Circ. Systems 36 521
[3] Lantsov V N 2019 Dynamic Compl. Syst. 13 53 (in Russian)
[4] Liu H, Danial L and Wong N 2015 Trans. Comp.-Aided Design Int, Circ. Systems 34 1059
[5] Zhang Q J, Gad E, Nouri B, Na W and Nakhla M 2021 IEEE J. Microw. 1 494
[6] Lantsov V N 2012 Design Tehn. Electr. Syst. 2 (in Russian)
[7] Saad Y 2013 SIAM J. Sci. Comp.
[8] Oseledets I V 2011 SIAM J. Sci. Comp. 33 2295
[9] Oseledets I 2011 Comp. Methods Appl. Mathem. 11 382
[10] Dolgov S V 2014 Algorithms and applications of tensor decompositions for numerical solving of multilevel tasks (Moscow, PhD Dissertation) (in Russian)
[11] TT-Toolbox. Https://github.com/oseledets/TT-Toolbox