Some practical remarks in solving partial differential equations using reduced order schemes obtained through the POD method

Alexandru SOLOMON[1], Valentin Claudiu OLTEI[1], Alina BOGOI*[,1]

*Corresponding author
[1]Department of Aerospace Sciences, POLITEHNICA University Bucharest, Splaiul Independentei 313, 060042, Bucharest, Romania,
alexandru.solomon@stud.aero.upb.ro, valentin.oltei@stud.aero.upb.ro, alina.bogoi@upb.ro*

DOI: 10.13111/2066-8201.2022.14.1.15

Received: 09 November 2021/ Accepted: 25 January 2022/ Published: March 2022

Abstract: *In this paper we address the subject of mathematical modelling, more precisely the optimization of algorithms for numerically solving partial differential equations. The problem proposed to be tackled in this paper is the implementation of an algorithm for solving partial differential equations in a significantly faster way than that obtained through applying finite difference schemes. The proper orthogonal decomposition (POD) method is a modern and efficient method of reducing the number of variables that occur as a result of applying centred difference schemes to partial differential equations, thus reducing the running time of the algorithm and the accumulation of truncation errors. Therefore, the POD method has been implemented to obtain a reduced order scheme applied to different partial differential equations, with some practical applications and comparisons with the analytical solutions.*

Key Words: *Reduced order method (ROM), Proper orthogonal decomposition (POD), Singular value decomposition (SVD)*

1. INTRODUCTION

Today, as is well known, we are living in an age of speed in which information is flowing faster and faster. Each of us uses at least one device that has an internet connection to navigate through the millions and millions of data circulating around the globe every second.For such a high processing speed and such a large volume of information, the problem arises quickly: processing time and storage space. The intuitive action of engineers and scientists is to build faster and more efficient computers, more capable of storing information. Another method that is starting to be implemented more and more nowadays is that of optimizing computational algorithms, the mathematical models on which they are based.

In this paper we address the subject of engineering, to improve algorithms that solves numerically partial differential equations which model physical phenomena studied in this part of science. In the following, we will present a topic that has been talked about more and more lately: reducing the number of freedom degrees for a faster computational solution of partial differential equation. In order to do this, an intense study of the specialized works was needed, but also a better knowledge of the mathematical tools necessary to solve such problems. Over the years, several authors have dealt with this subject, of which we would like to mention those presented in the reference chapter. We tried to take the best of the mentioned works and make some improvements where necessary. The present paper focuses on the field of numerical analysis, mainly on solving numerically, higher order partial differential equations. A Reduced Order Central Difference Scheme (ROECD) will be considered to solve such equations, the results of which will be analyzed and compared with "classical" numerical schemes (CD).

2. MATHEMATICAL MODEL

The problem proposed for solving in this paper is the implementation of an algorithm for solving partial differential equations significantly faster than that obtained by applying finite difference schemes. As presented in the work mentioned in the reference chapter, orthogonal decomposition method (POD) is a modern and effective method of reducing the number of unknowns that occur as a result of applying centered difference schemes, reducing in this way the running time of the algorithm and the accumulation of truncation errors. The steps to be followed for the implementation of the reduced order scheme are briefly presented below, and their implementation on a particular case is illustrated after the theoretical part.

In the first step we implement the finite difference scheme [6] and extract the snapshots [1], [2], [3]. The scheme with finite differences in vector form will be implemented for the first L time steps and the solutions of the equation will be stored in the columns of the snapshot matrix ($\mathbf{A}$). For the case of a 2D parabolic equation, the solution of the equation at a time n can be written as follows:

$$u_{i,j}^n = u(x_i, y_j, t^n) = u_m^n, \qquad 0 \le n \le N \tag{1}$$

$$\mathbf{u}^n = (u_1^n, u_2^n, u_3^n, \dots, u_m^n, \dots, u_M^n)^T, m = (j-1)\cdot I + i \longrightarrow \begin{Bmatrix} 1 \le i \le I+1 \\ 1 \le j \le J+1 \\ 1 \le m \le M = (I+1)(J+1) \end{Bmatrix} \tag{2}$$

$$\mathbf{A} = \begin{pmatrix} u_1^1 & u_1^2 & \dots & u_1^L \\ u_2^1 & u_2^2 & \dots & u_2^L \\ \vdots & \vdots & \ddots & \vdots \\ u_M^1 & u_M^2 & \cdots & u_M^L \end{pmatrix} = (\mathbf{u}^1 \quad \mathbf{u}^2 \quad \dots \quad \mathbf{u}^L) \tag{3}$$

The second step refers to decomposition into singular values (SVD) of the $\mathbf{A}^T\mathbf{A}$ matrix. The **SVD** will be applied to the matrix $\mathbf{A}^T\mathbf{A}$, obtaining the following equality according to [3]: $\mathbf{A}^T\mathbf{A} = \mathbf{U\Sigma V}^T$. According to the relations between the decomposition into singular values and the decomposition into eigenvalues illustrated in [1], [2], [3], [4], the matrix $\mathbf{V} = \mathbf{U}$ is the matrix of eigenvectors of $\mathbf{A}^T\mathbf{A}$ and is also the matrix of singular vectors to the right of $\mathbf{A}$, and is the diagonal matrix containing the non-zero eigenvalues of $\mathbf{A}^T\mathbf{A}$.

$$\mathbf{A}^T\mathbf{A} = \mathbf{U}\begin{pmatrix} \mathbf{\Sigma}_{d\times d} & \mathbf{0}_{d\times(L-d)} \\ \mathbf{0}_{(M-d)\times d} & \mathbf{0}_{(M-d)\times(L-d)} \end{pmatrix}\boldsymbol{V}^T \tag{4}$$

where $\mathbf{U} = \mathbf{V} = (\phi_1 \quad \phi_2 \quad \dots \quad \phi_L)$, and the diagonal matrix is:

$$\mathbf{\Sigma}_{d\times d} = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_d \end{pmatrix}$$

The third step consists in creating the POD basis. The matrix $\mathbf{\Sigma}$ contains on the main diagonal, in descending order, the eigenvalues of $\mathbf{A}^T\mathbf{A}$, denoted by $\boldsymbol{\lambda}$. According to [1], [2], [3] and [4] the condition that must be met in the formulation of the POD basis for the reduced order scheme to be convergent is that the expression:

$$|u_{exact} - u_{calculated}| = (1+\delta)^{N-L} \cdot \sqrt{\lambda_{k+1}} \tag{5}$$

does not exceed the order of magnitude of the errors obtained by implementing the finite difference scheme. δ is a strictly positive term that depends on the implementation of the finite difference scheme, N is the number of time steps that are intended to be calculated, and k + 1 is the index of the eigenvalue for which the condition is met. To ensure the convergence of the reduced order scheme, we choose $k = d = rank(\mathbf{A})$, so that $\lambda_{d+1} = 0$. If the rank of the matrix is maximum (d = L) and λ_{d+1} does not exist, return to: First step and choose another L (number of snapshots).

According to [2], the POD base has the following form: $\mathbf{\Phi}_{d\times d} = (\boldsymbol{\varphi}_1 \quad \boldsymbol{\varphi}_2 \quad \cdots \quad \boldsymbol{\varphi}_d)$ where the vectors are given by:

$$\boldsymbol{\varphi}_j = \mathbf{A}\phi_j / \sqrt{\lambda_j}, 1 \le j \le d \tag{6}$$

In the last step we solve the reduced order scheme that will be implemented as described in [2], so that the $\boldsymbol{u}_n$ will be replaced with the $\mathbf{u}_\mathbf{n}^* = \mathbf{\Phi}_{\mathbf{d}\times\mathbf{d}} \times \boldsymbol{\alpha}^\mathbf{n}$ and the newly obtained scheme in the variable $\boldsymbol{\alpha}^\mathbf{n} = (\boldsymbol{\alpha}_1^\mathbf{n} \quad \boldsymbol{\alpha}_2^\mathbf{n} \quad \dots \quad \boldsymbol{\alpha}_\mathbf{d}^\mathbf{n})^\mathbf{T}$ will be solved. This is possible due to the property:

$$\mathbf{\Phi}^\mathbf{T}\mathbf{\Phi} = \mathbf{I}_\mathbf{d} \tag{7}$$

*Note: According to the references there is also a step 5, which involves recalculating the POD basis if the convergence condition is not met. However, if the formulation of the POD basis in step 3 is followed, this step of renewing the base is no longer necessary, as the reduced order scheme is thus convergent, whatever the number of time steps.

In the next pages we will present two tests that solve a parabolic problem (in 2D) and a hyperbolic problem of order 4 (in 2D).

The 2D parabolic equation is given in a general form:

$$\begin{cases} \dfrac{\partial u}{\partial t} - \Delta u = f(x,y,t), & (x,y,t) \in \Omega \times [0,T_f) \subseteq \mathbf{R}^2 \times \mathbf{R} \\ u|_{\partial\Omega}(x,y,t) = g(x,y,t), & t \in [0,T_f) \\ u(x,y,0) = h(x,y), & (x,y) \in \Omega \end{cases} \tag{8}$$

The 2D 4-th order hyperbolic equation is given in a general form:

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} + \mu\Delta^2 u = f(x,y,t), & (x,y,t) \in \Omega \times [0,T_f) \subseteq \mathbf{R}^2 \times \mathbf{R} \\ u|_{\partial\Omega}(x,y,t) = g(x,y,t), & t \in [0,T_f) \\ \left.\dfrac{\partial u}{\partial x}\right|_{\partial\Omega} = h_1(x,y,t); \quad \left.\dfrac{\partial u}{\partial y}\right|_{\partial\Omega} = h_2(x,y,t) & \\ u(x,y,0) = h(x,y), & (x,y) \in \Omega \\ \dfrac{\partial u(x,y,0)}{\partial t} = h_3(x,y) & (x,y) \in \Omega \end{cases} \tag{9}$$

The classical schemes with centered differences expressed in a matrix formulation are for parabolic equation,

$$\boldsymbol{u}^{n+1}=\boldsymbol{u}^{n}+\frac{dt}{dx^2}\boldsymbol{B}\boldsymbol{u}^{n}+\frac{dt}{dy^2}\boldsymbol{C}\boldsymbol{u}^{n}+dt\boldsymbol{T} \tag{10}$$

and respectively for hyperbolic equation

$$\mathbf{u}^{n+1}=2\mathbf{u}^{n}-\mathbf{u}^{n-1}-\mu\frac{(dt)^2}{(dx)^4}\mathbf{K}\mathbf{u}^{n}-\mu\frac{(dt)^2}{(dy)^4}\mathbf{P}\mathbf{u}^{n}-2\mu\frac{(dt)^2}{(dx)^2(dy)^2}\mathbf{D}\mathbf{u}^{n}+(dt)^2\mathbf{T}^{n} \tag{11}$$

In the preparatory step we define the computational domain which has a rectangular shape and we divide it in equal intervals, then the space step, corresponding to the chosen space division, is calculated. The time step is chosen according to [1], [2] and [3] as follows:

$$dt\le\frac{1}{4\left(\frac{1}{dx^2}+\frac{1}{dy^2}\right)} \tag{12}$$

After calculating the values at the initial moment and entering them in the linearized form on the first column of the matrix **A**, as presented by relation (2), the matrices in equation (10) (for the parabolic equation) are implemented as:

$$\boldsymbol{B}=\begin{pmatrix}\mathbf{B}' & \cdots & \cdots & \cdots & 0\\ \vdots & \mathbf{B}'' & & & \vdots\\ \vdots & & \ddots & & \vdots\\ \vdots & & & \mathbf{B}'' & \vdots\\ 0 & \cdots & \cdots & \cdots & \mathbf{B}'\end{pmatrix}\mathbf{C}=\begin{pmatrix}\mathbf{C}' & \cdots & \cdots & \cdots & 0\\ \vdots & \mathbf{C}'' & & & \vdots\\ \vdots & & \ddots & & \vdots\\ \vdots & & & \mathbf{C}'' & \vdots\\ 0 & \cdots & \cdots & \cdots & \mathbf{C}'\end{pmatrix} \tag{13}$$

where $\mathbf{B}^{'},\mathbf{B}^{''},\mathbf{C}^{'},\mathbf{C}^{''}\in M_{(N_x+1)\times(N_x+1)}$are the following:

$$\mathbf{B}'=\mathbf{C}'=(-2)\mathbf{I}_{(N_x+1)\times(N_x+1)} \tag{14}$$

$$\mathbf{B}''=\begin{pmatrix}-2 & 0 & 0 & \cdots & 0 & 0 & 0\\ 1 & -2 & 1 & \ddots & 0 & 0 & 0\\ 0 & 1 & -2 & \ddots & 0 & 0 & 0\\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots\\ 0 & 0 & 0 & \ddots & -2 & 1 & 0\\ 0 & 0 & 0 & \ddots & 1 & -2 & 1\\ 0 & 0 & 0 & \cdots & 0 & 0 & -2\end{pmatrix} \tag{15}$$

$$\mathbf{C}''=\begin{pmatrix}-2 & \overbrace{\cdots}^{deN_x ori 0} & 0 & & & & \\ & -2 & & 1 & & & \\ 1 & & -2 & & 1 & & \\ & 1 & & \ddots & & \ddots & \\ & & \ddots & & -2 & & 1\\ & & & 1 & & -2 & \\ & & & & 0 & \underbrace{\cdots}_{deN_x ori 0} & -2\end{pmatrix} \tag{16}$$

For the hyperbolic equation, the $\mathbf{K},\mathbf{P},\mathbf{D}\in M_{(Nx+1)\times(Ny+1)}$matrices have the following form:

$$\mathbf{K}=\begin{pmatrix}\mathbf{K}' & \cdots & \cdots & \cdots & 0\\ \vdots & \mathbf{K}'' & & & \vdots\\ \vdots & & \ddots & & \vdots\\ \vdots & & & \mathbf{K}'' & \vdots\\ 0 & \cdots & \cdots & \cdots & \mathbf{K}'\end{pmatrix}\quad \mathbf{P}=\begin{pmatrix}\mathbf{P}' & \cdots & \cdots & \cdots & 0\\ \vdots & \mathbf{P}'' & & & \vdots\\ \vdots & & \ddots & & \vdots\\ \vdots & & & \mathbf{P}'' & \vdots\\ 0 & \cdots & \cdots & \cdots & \mathbf{P}'\end{pmatrix} \tag{17}$$

where $\mathbf{K}', \mathbf{K}'', \mathbf{P}', \mathbf{P}''$ are auxiliary matrices

$$\mathbf{K}'=\mathbf{P}'=6\mathbf{I}_{2(N_x+1)\times 2(N_x+1)} \tag{18}$$

$$\mathbf{K}''=\begin{pmatrix}6 & 0 & 0 & \cdots & 0 & 0 & 0\\ 0 & 6 & 0 & \ddots & 0 & 0 & 0\\ 1 & -4 & 6 & -4 & 1 & 0 & 0\\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots\\ 0 & 0 & 1 & -4 & 6 & -4 & 1\\ 0 & 0 & 0 & \ddots & 0 & 6 & 0\\ 0 & 0 & 0 & \cdots & 0 & 0 & 6\end{pmatrix}_{(Nx+1)\times(Nx+1)} \tag{19}$$

$$\mathbf{P}''=\left(\begin{array}{ccccccccc}6 & \overbrace{\cdots}^{Nxtimes0} & 0 & \overbrace{\cdots}^{Nxtimes0} & 0 & & & & \\ \vdots & 6 & & 0 & & 0 & & & \\ -4 & & 6 & & -4 & & \overbrace{\ddots}^{1} & & \\ \vdots & -4 & & 6 & & \ddots & & 1 & \\ 1 & & \ddots & & \ddots & & \ddots & & 1\\ & 1 & & \ddots & & 6 & & -4 & \vdots\\ & & \ddots & & -4 & & 6 & & -4\\ & & & 0 & & 0 & & 6 & \vdots\\ & & & & 0 & \cdots & 0 & \cdots & 6\end{array}\right)_{(Nx+1)\times(Nx+1)} \tag{20}$$

$$\mathbf{D}=\begin{pmatrix}\mathbf{D}' & \cdots & \cdots & \cdots & 0\\ \vdots & \mathbf{D}'' & & & \vdots\\ \vdots & & \ddots & & \vdots\\ \vdots & & & \mathbf{D}'' & \vdots\\ 0 & \cdots & \cdots & \cdots & \mathbf{D}'\end{pmatrix} \tag{21}$$

where $\mathbf{D}', \mathbf{D}''$ are auxilary matrices

$$\mathbf{D}'=4\mathbf{I}_{2(N_x+1)\times 2(N_x+1)} \tag{22}$$

$$\mathbf{D}''=\left(\begin{array}{ccccccccc}4 & \overbrace{\cdots}^{Nx-1times0} & 0 & 0 & 0 & & & & \\ \vdots & 4 & & 0 & 0 & 0 & & & \\ 1 & & 4 & & 1 & -2 & 1 & & \\ -2 & 1 & & 4 & & 1 & -2 & 1 & \\ 1 & -2 & 1 & & \ddots & & \ddots & \ddots & \ddots\\ & \ddots & \ddots & \ddots & & 4 & & 1 & -2\\ & & 1 & -2 & 1 & & 4 & & 1\\ & & & 0 & 0 & 0 & & 4 & \vdots\\ & & & & 0 & 0 & 0 & \cdots & 4\end{array}\right)_{(Nx+1)\times(Nx+1)} \tag{23}$$

In first step from the implementation process the matrix **A** of snapshots is formed by calculating the first L moments of time using the classical scheme (10). That L is the smallest natural number for which the rank of the matrix **A** is less than L. For each time moment, calculate the vector of free terms and, taking into account the boundaries, calculate the values of the function for each next time with the lines code specific to each type of equation:

Parabolic equation

$$\mathbf{a}_i = \mathbf{a}_{i-1} + \mu_x \mathbf{B}\mathbf{a}_{i-1} + \mu_y \mathbf{C}\mathbf{a}_{i-1} + dt \cdot \mathbf{T} \tag{24}$$

where $\mathbf{a}_i$ is one entire column in the matrix **A** and $\mu_x = c_x \frac{dt}{dx^2}$, $\mu_y = c_y \frac{dt}{dy^2}$ and $i \in \{1,2,\dots,L\}$.

Hyperbolic equation

$$\mathbf{a}_i = 2\mathbf{a}_{i-1} - \mathbf{a}_{i-2} - \mu_x \mathbf{K}\mathbf{a}_{i-1} - \mu_y \mathbf{P}\mathbf{a}_{i-1} - \mu_{xy} \mathbf{D}\mathbf{a}_{i-1} + dt^2 \cdot \mathbf{T} \tag{25}$$

where $\mu_x = \delta \frac{(dt)^2}{(dx)^4}$, $\mu_y = \delta \frac{(dt)^2}{(dy)^4}$, $\mu_{xy} = 2\delta \frac{(dt)^2}{(dx)^2(dy)^2}$ and $i \in \{1,2,\dots,L\}$.

After forming the snapshot matrix **A** and after checking the conformity of the errors with the chosen scheme, we proceed to the next step.

For the next step we created in the program the function POD which, having as input parameter the matrix **A**, applies the procedure from comercial computing softwaare, svd, for the quadratic matrix $\mathbf{A}^T\mathbf{A}$. At the end of this step we will have the matrices from the relation (4) calculated by means of the above mentioned procedure.

This 3rd step, theoretically presented previously, is also contained in the newly created function, POD. In short, at the current step the proper orthogonal basis for determining the solutions of the function **u** is determined. Thus, using those calculated in the previous step and the matrix **A**, the POD basis is formed with the following relation.

$$\boldsymbol{\varphi}_j = \mathbf{A}\phi_j / \sqrt{\lambda_j} 1 \le j \le rank(\mathbf{A}) \tag{26}$$

The number of columns in the base is equal to the rank of the matrix **A**, as explained previously in order to ensure the stability of the reduced order scheme whitch will be presented. The POD basis, which will be used in the implementation of the reduced order scheme, is obtained at the end of the current step, through the procedures discussed.

At this step, four, considering the properties (7) of the POD_basis matrix and ***u***'s decomposition as presented in [2], but also in the previous chapter, we implement the new reduced order scheme, specific to each type of equation, with the instructions:

The reduced order sheme for the parabolic equation becomes:

$$\boldsymbol{\alpha}_i = \boldsymbol{\alpha}_{i-1} + \mathbf{B}_1\boldsymbol{\alpha}_{i-1} + \mathbf{C}_1\boldsymbol{\alpha}_{i-1} + dt\boldsymbol{\Phi}^T T \tag{27}$$

where $\mathbf{B}_1$ and $\mathbf{C}_1$ represent the newly created matrices corresponding to the reduced order scheme and $i \in \{L, L+1,\dots,N\}$.

$$\mathbf{B}_1 = \frac{dt}{dx^2}\mathbf{\Phi}^T\mathbf{B}\mathbf{\Phi} \qquad \mathbf{C}_1 = \frac{dt}{dy^2}\mathbf{\Phi}^T\mathbf{C}\mathbf{\Phi} \tag{28}$$

Similarly, for hyperbolic equation we get:

$$\boldsymbol{\alpha}_i = 2\boldsymbol{\alpha}_{i-1} - \boldsymbol{\alpha}_{i-2} - K_1\boldsymbol{\alpha}_{i-1} - P_1\boldsymbol{\alpha}_{i-1} - D_1\boldsymbol{\alpha}_{i-1} + dt\boldsymbol{\Phi}^T T \tag{29}$$

where $\mathbf{B_1}$ and $\mathbf{C_1}$ represent the newly created matrices corresponding to the reduced order scheme and $i \in \{L, L+1, \dots, N\}$.

$$\mathbf{K}_1 = \mu \frac{(dt)^2}{(dx)^4} \mathbf{\Phi}^T \mathbf{K} \mathbf{\Phi}, \ \ \mathbf{P}_1 = \mu \frac{(dt)^2}{(dy)^4} \mathbf{\Phi}^T \mathbf{P} \mathbf{\Phi}, \ \ \mathbf{D}_1 = \mu \frac{(dt)^2}{(dx)^2 (dy)^2} \mathbf{\Phi}^T \mathbf{D} \mathbf{\Phi} \tag{30}$$

By multiplying the alpha vector by the POD_basis matrix, the solution of the reduced order system of the FTCS scheme for the parabolic equation is obtained.

3. NUMERICAL RESULTS AND CONCLUSIONS

In order to draw valid conclusions, the computation parameters will be selected as follows:

Table 1. Parabolic/Hyperbolic equation parameters

Name of the parameter	Value
The number of intervals in the x direction	50
The number of intervals in the y direction	50
The domain on x	[0,2]
The domain on y	[0,2]
Exact function	e^{-2txy}
Final time	1
Final time	3
Number of snapshots	10

Other important parameters derived from the above are:

Table. 2 Derived parameters for parabolic/hiperbolic equation

Name of the parameter	Value
Time step, dt	0.0002
Space step on x	0.04
Space step on y	0.04
Number of points in the domain	2601

For the two selected time moments $t = 1s$ and $t = 3s$ the numerical solution of the equation will be calculated by two methods: the ROFTCS (reduced order forward time central space scheme) method and the classic FTCS (forward time central space scheme) method.

Table 3. Computation result for parabolic equation after 1s

	ROFTCS	ROFTCS(+step 5)	FTCS
Computation time (seconds)	17	33	256
Norm of the poinwise relative error (%)	0.0122	0.0115	0.2314

In tab. 3 are presented the computational times and the errors corresponding to the results.

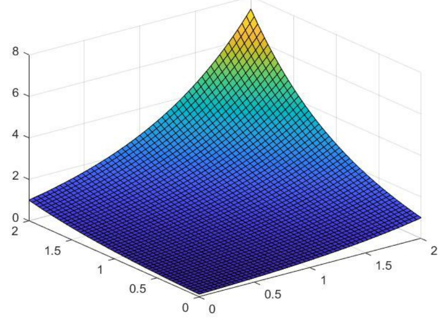

Fig. 1 The ROFTCS solution of the parabolic equation at time t = 1s

Table 4. Computation result for parabolic equation after 3s

	ROFTCS	ROFTCS(+step5)	FTCS
Computation time (seconds)	23	45	760
Norm of the pointwise relative error divided by the number of points in the grid	0.01216	0.0102	0.2464

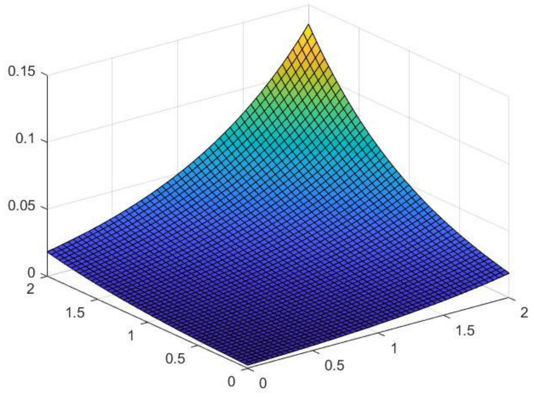

Fig. 2 The ROFTCS solution of the parabolic equation in time t = 3s

For the hyperbolic equation, as it is described in [1] and [5], we will consider the same input parameters as were taken in the case of the parabolic equation, minus the exact function which is equal to:

Table 5. Hiperbolic equation parameters

Numele parametrului	Valoarea
Exact function	$2e^{-\pi t}\, sin(\pi x)\, sin(\pi y)$
Coefficient of space derivatives(μ)	0.01

For testing the application consider the same two time points (1s, 3s).

Table 6. Computation result for hiperbolic equation after 1s

	ROFTCS	ROFTCS(+step5)	FTCS
Computation time (seconds)	10	34	210
Norm of the pointwise relative error divided by the number of points in the grid	0.0091	0.0083	0.0088

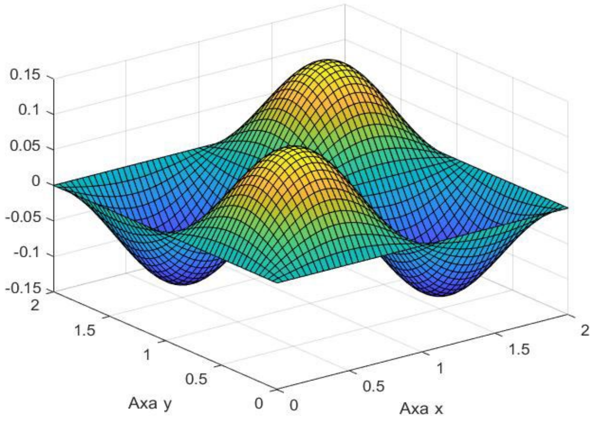

Fig. 3 The ROFTCS solution of the hiperbolic equation in time t = 1s

Table 7. Computation result for hiperbolic equation after 3s

	ROFTCS	**ROFTCS(+step5)**	**FTCS**
Computation time (seconds)	52	74	1451
Norm of the pointwise relative error divided by the number of points in the grid	0.01216	0.0118	0.02464

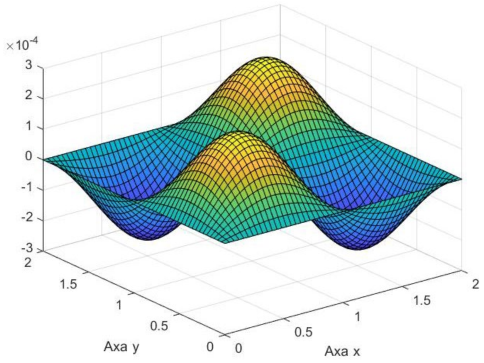

Fig. 4 The ROFTCS solution of the hiperbolic equation in time t = 3s

In conclusion, the current paper presents the steps of implementing a reduced order scheme for solving partial differential equations and applying them to a concrete example. Within the application, the way of solving a 2D parabolic equation was exemplified, and the results were compared with those obtained by implementing the classic FTCS scheme. As shown in the previous illustrations (Fig. 1, 2, 3, 4), the results obtained by implementing the reduced order scheme are significantly closer to the exact value than those obtained by implementing the FTCS scheme. At the same time, an essential thing is that the running time of the ROFTCS algorithm is very short compared to that for FTCS, and the discrepancy between the two programs increases as the number of discretization points increases. This is due to the fact that the number of unknowns reached by implementing ROFTCS depends only on the number of snapshots chosen, which is of the order of tens and, compared to the FTCS scheme, does not depend on the number of points chosen to discretize the spatial dimensions. To illustrate this with an extreme case, we chose Nx = Ny = 100, Tfinal = 2s and $dt = 5 \cdot 10^{-5}s$: the ROFTCS program ran in 830s (<14 min), while FTCS ran in about 32400s (9h), ROFTCS having a much smaller error than FTCS for t = 2s.

Our improvement for the aleardy existing reduced order shemes is the removal of step 5, and we demosntrate how this thing is possible for these types of schemes. If we remove step 5, presented in the literature, there is a significant decrease in computation time, while the error has only an insignificant increase.

In terms of future plans, these reduced order schemes can be implemented to solve various types of equations, starting from classical numerical schemes. FTCS (4th order 2D parabolic and hyperbolic equations), Crank-Nickolson and ADI are the schemes we have implemented. The literature on these schemes is very new, so there are certainly ways to improve it.

ACKNOWLEDGEMENT

This article is an extension of the paper presented at ***The 39th "Caius Iacob" Conference on Fluid Mechanics and its Technical Applications***, 28 – 29 October 2021, Bucharest, Romania, Virtual Conference, Section 4 – Mathematical Modeling.

REFERENCES

[1] L. Zhendong, J. Shiju, C. Jing, A reduced order extrapolation central difference scheme based on POD for two-dimensional fourth-order hyperbolic equations, *Applied Mathematics and Computation*, vol **289**, issue C, pp. 396-408, DOI:10.1016/j.amc.2016.05.032, October 2016.

[2] L. Zhendong, C. Goong, *Proper Orthogonal Decomposition Methods for Partial Differential Equations*, Mathematics in Science and Engineering, ISBN: 978-0-12-816798-4, 2019.

[3] L. Zhendong, X. Zhenghui, S. Yueqiang, C. Jing, A reduced finite volume element formulation and numerical simulations based on POD for parabolic problems, Elsevier Journals, *Journal of Computational and Applied Mathematics*, **235**, 2098–2111, 2011.

[4] S. Volkwein, *Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling*, Lecture Notes, University of Konstanz, Department of Mathematics and Statistics, August 27, 2013.

[5] Q. Li, Q. Yang, Compact difference scheme for two-dimensional fourth-order hyperbolic equation, *Advances in Difference Equations*, **2019**(1), DOI:10.1186/s13662-019-2094-4, August 2019.

[6] A. Bogoi, An introduction to differential equations, ISBN: 9789735678746 9735678748, Ed. Monitorul Oficial, Bucharest, 2014.