Applying Proper Orthogonal Decomposition to Parabolic Equations: A Reduced Order Numerical Approach

Mircea MORARU*

*Corresponding author
Department of Aerospace Sciences, POLITEHNICA University Bucharest,
Splaiul Independentei 313, 060042, Bucharest, Romania,
mircea.moraru@stud.aero.upb.ro

DOI: 10.13111/2066-8201.2024.16.1.7

Received: 05 December 2023/ Accepted: 23 January 2024/ Published: March 2024
Copyright © 2024. Published by INCAS. This is an "open access" article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

Abstract: *In this paper we present a low-order numerical scheme developed using the Proper Orthogonal (POD) method to address non-homogeneous parabolic equations in both one and two dimensions. The proposed schemes leverage the POD technique to reduce the computational complexity associated with solving these equations while maintaining accuracy. By employing POD, the high-dimensional problem is approximated by a reduced set of models, allowing for a more efficient representation of the system dynamics. The application of this method to non-homogenous parabolic equations offers a promising approach for enhancing the computational efficiency of simulations in diverse fields, such as fluid dynamics, heat conduction, and reaction-diffusion processes. The presented numerical scheme demonstrates its efficacy in achieving accurate results with significantly reduced computational costs, making it a valuable tool for applications demanding efficient solutions to non-homogeneous parabolic equations in one and two dimensions.*

Key Words: *Proper orthogonal decomposition (POD), Singular value decomposition (SVD), Reduced order method (ROM), Forward time centered space (FTCS)*

1. INTRODUCTION

We live in an era where the size of data represents a space with an enormous exponential growth. Parabolic equations describe phenomena such as gas proliferation, thermal conductivity, and liquid infiltration, events that occur in nature and daily life. Exact solutions for these engineering problems are not easily found, especially concerning numerical solutions. Among the methods mentioned above, the finite difference scheme is the simplest, as it is easier to understand and program, being considered the most efficient method. However, regarding finite difference schemes for the parabolic equation, especially in the case of a large dimension, there are too many degrees of freedom. Thus, an important problem arises: how to simplify computational resources and save computation time in a way that guarantees the highest accuracy and efficiency of the numerical solution [1].

The Proper Orthogonal Decomposition (POD) method is a technique that offers the possibility of obtaining the most suitable approximation in terms of fluid flow representation with a reduced number of degrees of freedom and reduced-size models to alleviate the computational burden and not to overload the memory requirements for data processing. The method essentially provides an orthogonal basis to represent processed data in the optimal sense for the processed dataset. Combined with the Galerkin projection method, POD represents a powerful method for generating reduced-size models of dynamic systems that have an extremely large or even infinite domain of definition [11].

Modal decomposition is necessary for the following reasons [2, 4]:

- Recognition of distinct structures in databases.
- Conducting a statistical analysis of processed data.
- Obtaining a set of models to generate a basis. This basis is subsequently used to approximate solutions to numerical problems with characteristics similar to those used in constructing the modal basis.
- Reconstruction of data states using a reduced-order model (ROM) with a lower order, ensuring that the informational quantity remains relatively the same or close.

2. THE GENERATION OF THE OPTIMAL BASIS THROUGH THE SVD-POD METHOD

We generate the set of instantaneous moments $\left\{u_j^{n_i}\right\}_{i=1,L}$ $(j = 1,2,\dots,J-1, 1 \leq n_1 < n_2 <\cdot \cdot< n_L \leq N)$ that can be expressed as a matrix $\boldsymbol{A}_{m\times L}(m = J-1)$ written as:

$$\boldsymbol{A}_{m\times L} = \begin{pmatrix} u_1^{n_1} & u_1^{n_2} & \dots & u_1^{n_L} \\ u_2^{n_1} & u_2^{n_2} & \dots & u_2^{n_L} \\ \vdots & \vdots & \ddots & \vdots \\ u_m^{n_1} & u_m^{n_2} & \cdots & u_m^{n_L} \end{pmatrix} \tag{1}$$

Using the SVD (**S**ingular **V**alue **D**ecomposition) method for the $\boldsymbol{A}_{m\times L}$ matrix we obtain:

$$\boldsymbol{A} = \boldsymbol{U\Sigma V}^{\mathrm{T}}, \tag{2}$$

where $\boldsymbol{U} = \boldsymbol{U}_{m\times m}$ and $\boldsymbol{V} \in M_{L\times L}(\mathbb{R})$ are orthogonal matrices, and the matrix $\boldsymbol{\Sigma}_{m\times L}$ is:

$$\boldsymbol{\Sigma}_{m\times L} = \begin{pmatrix} \boldsymbol{\Sigma}_{l\times l} & \boldsymbol{O}_{l\times(L-l)} \\ \boldsymbol{O}_{(m-l)\times l} & \boldsymbol{O}_{(m-l)\times(L-l)} \end{pmatrix}, \tag{3}$$

and

$$\boldsymbol{\Sigma}_{l\times l} = \begin{pmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_l \end{pmatrix} = diag(\sigma_1, \sigma_2, \dots, \sigma_l) \tag{4}$$

The values on the main diagonal of the matrix $\boldsymbol{\Sigma}_{l\times l}$ are called singular values of the matrix $\boldsymbol{A}$, which respects the following condition: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_l > 0$ [10, 11].

Thus, relation (2) becomes:

$$\boldsymbol{A} = \boldsymbol{U} \begin{pmatrix} \boldsymbol{\Sigma}_{l\times l} & \boldsymbol{O}_{l\times(L-l)} \\ \boldsymbol{O}_{(m-l)\times l} & \boldsymbol{O}_{(m-l)\times(L-l)} \end{pmatrix} \boldsymbol{V}^{\mathrm{T}}, \tag{5}$$

Matrix $\boldsymbol{U} = (\boldsymbol{\psi}_1 \quad \boldsymbol{\psi}_2 \quad \dots \quad \boldsymbol{\psi}_m)$ contains the orthogonal vectors specific of the matrix $\boldsymbol{AA}^{\mathrm{T}}$. While the singular values σ_i $(i = 1,2,\dots,l)$ satisfy the relationship $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_l > 0$. A column of the matrix $\boldsymbol{A} \in \mathbb{R}_{m\times L}$ is written as $\boldsymbol{w}^{n_i} = (u_1^{n_i}, u_2^{n_i}, \dots, u_m^{n_i})^T$, $(i = 1,2,\dots,L)$ [3]. We denote by $\boldsymbol{\Phi} = (\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \dots, \boldsymbol{\psi}_M)$ the matrix of the first M vectors of the orthogonal matrix $\boldsymbol{U} = (\boldsymbol{\psi}_1 \quad \boldsymbol{\psi}_2 \quad \dots \quad \boldsymbol{\psi}_m)$ [3]. Therefore, we have for any $M \leq l$ defines the projection P_M by

$$P_M(\boldsymbol{w}^n) = \boldsymbol{w}^{*n} = \sum_{j=1}^{M} (\boldsymbol{\psi}_j, \boldsymbol{w}^n)\, \boldsymbol{\psi}_j \tag{6}$$

where $(\cdot,\cdot)$ represents the dot product between two vectors. We will have the following relation:

$$\|\boldsymbol{w}^n - P_M(\boldsymbol{w}^n)\|_2 \leq \sigma_{M+1} \tag{7}$$

where $\|.\|_2$ represents the standard Euclidean norm of the vector. Thus, $\{\boldsymbol{\psi}_i\}_{i=1,M}$ represents an optimal basis, and $\boldsymbol{\Phi}$ is a matrix $\boldsymbol{\Phi} = (\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \dots, \boldsymbol{\psi}_M)$ constructed with orthogonal vectors so that $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} = \boldsymbol{I}$ ($\boldsymbol{I} \in \mathbb{R}_{M\times M}$ is the identity matrix) [3]. The relationships describing the initial and boundary conditions can be written in the following vectorial form:

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n + \frac{a^2 \Delta t}{\Delta x^2} \boldsymbol{K}\boldsymbol{w}^n + \Delta t \boldsymbol{F}^n, n = 0,1,\dots,N-1 \tag{8}$$

where $\boldsymbol{F}^n = (f_1^n, f_2^n, \dots, f_m^n)^{\mathrm{T}}$ and

$$\boldsymbol{K} = \begin{pmatrix} -1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & -2 & 1 & \ddots & 0 & 0 & 0 \\ 0 & 1 & -2 & \ddots & 0 & 0 & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \ddots & -2 & 1 & 0 \\ 0 & 0 & 0 & \ddots & 1 & -2 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & -1 \end{pmatrix} \tag{9}$$

If $\boldsymbol{w}^n$ is replaced in equation (8) by his approximation $\boldsymbol{w}^{*n}$ which is determined by the following relationship:

$$\boldsymbol{w}^{*n} = \boldsymbol{\Phi}\boldsymbol{\alpha}^n = \boldsymbol{\Phi}_{m\times M}(\boldsymbol{\alpha}^n)_{M\times 1},\ \ n = 0,1,2,\dots,N, \tag{10}$$

We are going to obtain:

$$\boldsymbol{w}^{*n+1} = \boldsymbol{w}^{*n} + \frac{a^2 \Delta t}{\Delta x^2} \boldsymbol{K}\boldsymbol{w}^{*n} + \Delta t \boldsymbol{F}^n,\ n = 0,1,\dots,N-1, \tag{11}$$

We obtain a system equivalent to [6]

$$\boldsymbol{\Phi}\boldsymbol{\alpha}^{n+1} = \boldsymbol{\Phi}\boldsymbol{\alpha}^n + \frac{a^2 \Delta t}{\Delta x^2} \boldsymbol{K}\boldsymbol{\Phi}\boldsymbol{\alpha}^n + \Delta t \boldsymbol{F}^n, n = 0,1,\dots,N-1 \tag{12}$$

Recalling that $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} = \boldsymbol{I}$, we multiply to the left by $\boldsymbol{\Phi}^{\mathrm{T}}$, in equation (12) we obtain a system with a reduced number of equations ($M \ll m$) associated with the classical Crank-Nicolson scheme with m equations [1]:

$$\boldsymbol{\alpha}^{n+1} = \boldsymbol{\alpha}^n + \frac{a^2 \Delta t}{\Delta x^2} \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{\Phi}\boldsymbol{\alpha}^n + \Delta t \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{F}^n,\ \ n = 0,1,\dots,N-1, \tag{13}$$

where $\boldsymbol{\alpha}^0 = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{w}^0 = \boldsymbol{\Phi}^{\mathrm{T}}(u_1^0, u_2^0, \dots, u_m^0)^{\mathrm{T}}$.

After we obtain $\boldsymbol{\alpha}^n$ from equation (13), we deduct the optimized solutions for the PODROEFD scheme which are $\boldsymbol{w}^{*n} = \boldsymbol{\Phi}\boldsymbol{\alpha}^n$. The low-order finite difference Crank-Nicolson scheme includes only $M \times N$ equations, but the classical finite difference Crank-Nicolson includes $m \times N$ equations, but usually $m \gg M$, therefore this means that time will be saved in future numerical simulations, [9].

The 1D parabolic equation is given in a general form:

$$\frac{\partial u}{\partial t} - a^2 \frac{\partial^2 u}{\partial x^2} = f(t,x), \quad 0 < t < T, 0 < x < l, \tag{14}$$

with the following Dirichlet-type boundary conditions

$$\begin{aligned} u(0,t) &= h_1(t), 0 \le t \le T \\ u(l,t) &= h_2(t), 0 \le t \le T \end{aligned} \tag{15}$$

and the initial condition

$$u(x,0) = \varphi(x), \quad\quad 0 \le x \le l, \tag{16}$$

where a is a constant, and $f(t,x)$ and $\varphi(x)$ are two smooth functions. We consider the spatial step Δx and the time step Δt, we generate the grid $x_j = j\Delta x (j = 0,1,2,\dots,J)$, $t_n = n\Delta t (n = 0,1,2,\dots,N)$ and we denote by u_j^n an approximation of the exact solution $u(x_j,t_n)$ and by $f_j^{n+\frac{1}{2}} = \frac{1}{2}\left[f(x_j,t_n) + f(x_j,t_{n+1})\right]$, [5].

Applying the Crank-Nicolson finite difference scheme to the parabolic equation yields:

$$\begin{aligned} w_j^{n+1} = w_j^n + \frac{a^2 \Delta t}{\Delta x^2}\left[(1-\theta)\left(w_{j+1}^{n+1} - 2w_j^{n+1} + w_{j-1}^{n+1}\right) + \theta\left(w_j^n - 2w_j^n + w_j^n\right)\right] \\ + \Delta t F_j^{n+\frac{1}{2}}, \quad\quad j = \overline{1, N_x - 1} \end{aligned} \tag{17}$$

to which Dirichlet-type conditions are attached:

$$\begin{aligned} w_0^n &= h_1(t^n), \\ w_{N_x}^n &= h_2(t^n), \\ w_j^0 &= f(x_j), \quad\quad j = \overline{0, N_x} \end{aligned} \tag{18}$$

or, explicitly written, we obtain:

$$\begin{cases} w_0^{n+1} = h_1(t^{n+1}) \\ w_1^{n+1} = w_1^n + \theta \dfrac{a^2 \Delta t}{\Delta x^2}(w_2^n - 2w_1^n + w_0^n) + \\ \quad\quad + (1-\theta)\dfrac{a^2 \Delta t}{\Delta x^2}(w_0^{n+1} - 2w_1^{n+1} + w_2^{n+1}) + \Delta t \dfrac{F_1^n + F_1^{n+1}}{2} \\ w_2^{n+1} = w_2^n + \theta \dfrac{a^2 \Delta t}{\Delta x^2}(w_3^n - 2w_2^n + w_1^n) + \\ \quad\quad + (1-\theta)\dfrac{a^2 \Delta t}{\Delta x^2}(w_1^{n+1} - 2w_2^{n+1} + w_3^{n+1}) + \Delta t \dfrac{F_2^n + F_2^{n+1}}{2} \\ \vdots \\ w_{N_x-1}^{n+1} = w_{N_x-1}^n + \dfrac{\theta a^2 \Delta t}{\Delta x^2}(w_{N_x}^n - 2w_{N_x-1}^n + w_{N_x-2}^n) + \\ \quad\quad + \dfrac{(1-\theta) a^2 \Delta t}{\Delta x^2}(w_{N_x}^{n+1} - 2w_{N_x-1}^{n+1} + w_{N_x-2}^{n+1}) + \Delta t \dfrac{F_{N_x-1}^n + F_{N_x-1}^{n+1}}{2} \\ w_{N_x}^{n+1} = h_2(t^{n+1}) \end{cases} \tag{19}$$

In the following we will present the variant of introducing the boundary conditions in the matrix equation of the system. This variant consists in solving a system of $(N_x - 1) \times (N_x -$

1) equations instead of $(N_x+1)\times(N_x+1)$. Basically, the vector of unknowns at western and eastern border will be dropped because they will be introduced in the other equations. If we replace the first and the last condition in relation (17) the following relationship is obtained explicitly:

$$\begin{cases} w_0^{n+1} = h_1(t^{n+1}) \\ w_1^{n+1} = w_1^n + \theta\dfrac{a^2\Delta t}{\Delta x^2}(w_2^n - 2w_1^n) + \\ \qquad + (1-\theta)\dfrac{a^2\Delta t}{\Delta x^2}(w_0^{n+1} - 2w_1^{n+1} + w_2^{n+1}) + \theta\dfrac{a^2\Delta t}{\Delta x^2}w_0^n + \\ \qquad + \Delta t\dfrac{F_1^n + F_1^{n+1}}{2} \\ w_2^{n+1} = w_2^n + \theta\dfrac{a^2\Delta t}{\Delta x^2}(w_3^n - 2w_2^n + w_1^n) + \\ \qquad + (1-\theta)\dfrac{a^2\Delta t}{\Delta x^2}(w_1^{n+1} - 2w_2^{n+1} + w_3^{n+1}) + \\ \qquad + \Delta t\dfrac{F_2^n + F_2^{n+1}}{2} \\ \vdots \\ w_{N_x-1}^{n+1} = w_{N_x-1}^n + \dfrac{\theta a^2\Delta t}{\Delta x^2}(w_{N_x-2}^n - 2w_{N_x-1}^n) + \\ \qquad + \dfrac{(1-\theta)a^2\Delta t}{\Delta x^2}(w_{N_x}^{n+1} - 2w_{N_x-1}^{n+1} + w_{N_x-2}^{n+1}) + \dfrac{\theta a^2\Delta t}{\Delta x^2}w_{N_x}^n + \\ \qquad + \Delta t\dfrac{F_{N_x-1}^n + F_{N_x-1}^{n+1}}{2} \\ w_{N_x}^{n+1} = h_2(t^{n+1}) \end{cases} \tag{20}$$

Using $\boldsymbol{w}^n = (w_1^n, w_2^n, \ldots, w_{N_x-1}^n)^{\mathrm{T}}$, relation (17) can be written in the following matrix formulation:

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n + \theta\frac{a^2\Delta t}{\Delta x^2}\boldsymbol{K}\boldsymbol{w}^n + (1-\theta)\frac{a^2\Delta t}{\Delta x^2}\boldsymbol{K}\boldsymbol{w}^{n+1} + \Delta t\widetilde{\boldsymbol{F}}^{n+\frac{1}{2}}, \quad n = 0,1,\ldots,N-1, \tag{21}$$

or

$$\left[\boldsymbol{I} - (1-\theta)\frac{a^2\Delta t}{\Delta x^2}\boldsymbol{K}\right]\boldsymbol{w}^{n+1} = \left(\boldsymbol{I} + \theta\frac{a^2\Delta t}{\Delta x^2}\boldsymbol{K}\right)\boldsymbol{w}^n + \Delta t\widetilde{\boldsymbol{F}}^{n+\frac{1}{2}}, \quad n = 0,1,\ldots,N-1, \tag{22}$$

where

$$\widetilde{\boldsymbol{F}}^{n+\frac{1}{2}} = \begin{bmatrix} \theta\dfrac{a^2}{\Delta x^2}h_1(t^n) + (1-\theta)\dfrac{a^2}{\Delta x^2}h_1(t^{n+1}) + \dfrac{F_1^n + F_1^{n+1}}{2} \\ \dfrac{F_2^n + F_2^{n+1}}{2} \\ \vdots \\ \theta\dfrac{a^2}{\Delta x^2}h_2(t^n) + (1-\theta)\dfrac{a^2}{\Delta x^2}h_2(t^{n+1}) + \dfrac{F_{N_x-1}^n + F_{N_x-1}^{n+1}}{2} \end{bmatrix} \tag{23}$$

and

$$
\boldsymbol{K}=\begin{pmatrix}
-2 & 1 & 0 & \cdots & 0 & 0 & 0\\
1 & -2 & 1 & \ddots & 0 & 0 & 0\\
0 & 1 & -2 & \ddots & 0 & 0 & 0\\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots\\
0 & 0 & 0 & \ddots & -2 & 1 & 0\\
0 & 0 & 0 & \ddots & 1 & -2 & 1\\
0 & 0 & 0 & \cdots & 0 & 1 & -2
\end{pmatrix} \tag{24}
$$

where we consider the vector $\boldsymbol{w}^{n+1}$ equals to:

$$
\boldsymbol{w}^{n+1}=\begin{bmatrix} w_1^{n+1}\\ w_2^{n+1}\\ \vdots\\ w_{N_x-2}^{n+1}\\ w_{N_x-1}^{n+1}\end{bmatrix} \tag{25}
$$

Solving equation (22) involves solving a tridiagonal system of form:

$$
\boldsymbol{A}\boldsymbol{w}^{n+1}=\boldsymbol{B}\boldsymbol{w}^{n}+\Delta t\widetilde{\boldsymbol{F}}^{n+\frac{1}{2}} \tag{26}
$$

at each iteration for the time advance [7].

We consider the general form of a parabolic equation in 2D:

$$
\begin{cases}
\dfrac{\partial u}{\partial t}(x,y,t)-\dfrac{\partial^2 u}{\partial x^2}(x,y,t)-\dfrac{\partial^2 u}{\partial y^2}(x,y,t)=f(x,y,t),\\
u(x,y,t)=g(x,y,t),(x,y,t)\in\partial\Omega\times[0,T)\\
u(x,y,0)=s(x,y),(x,y)\in\Omega
\end{cases} \tag{27}
$$

where $f(x,y,t), g(x,y,t), s(x,y)$ represents the source term, the boundary function, and the initial function, respectively, while T is the total duration of time for the simulation [11].

We consider Δx and Δy the spatial steps for x and y directions, and Δt the time step, also we denote by $u_{j,k}^n=u(x_j,y_k,t^n)$ the value of function [8].

Using the FTCS scheme in the x direction and also in the y direction, we obtain:

$$
\begin{aligned}
w_{i,j}^{n+1}=w_{i,j}^{n}&+\frac{\Delta t}{(\Delta x)^2}\left(w_{i+1,j}^n-2w_{i,j}^n+w_{i-1,j}^n\right)+\\
&+\frac{\Delta t}{(\Delta y)^2}\left(w_{i,j+1}^n-2w_{i,j}^n+w_{i,j-1}^n\right)+\Delta t F_{i,j}^n
\end{aligned} \tag{28}
$$

To which we attach the initial conditions:

$$
w(x_i,y_j,0)=f(x_i,y_j), i=\overline{1,N_x+1}, j=\overline{1,N_y+1} \tag{29}
$$

Dirichlet-type boundary conditions on the western and eastern boundaries are [12]:

$$
\begin{aligned}
w_{1,j}^n&=h_w(x_{1,}y_j,t^n), j=\overline{1,N_y+1}\\
w_{N_x+1,j}^n&=h_e(x_{N_x+1,}y_j,t^n), j=\overline{1,N_y+1}
\end{aligned} \tag{30}
$$

The boundary conditions on the southern and northern boundaries are, respectively, [12]:

$$
\begin{aligned}
w_{i,1}^n&=h_s(x_{1,}y_j,t^n), i=\overline{1,N_x+1}\\
w_{i,N_y+1}^n&=h_n(x_{N_x+1,}y_j,t^n), i=\overline{1,N_x+1}
\end{aligned} \tag{31}
$$

We will transition from the index pair (i,j) to the multi-index defined by [12]:

$$m = (j-1)(N_x+1)+i, i = \overline{1, N_x+1}, j = \overline{1, N_y+1} \tag{32}$$

Equation (28) will be rewritten using the multi-index [12]:

$$\begin{aligned} w_m^{n+1} = w_m^n + {c_x}^2 \frac{\Delta t}{\Delta x^2}(w_{m-1}^n - 2w_m^n + w_{m+1}^n) + \\ + {c_y}^2 \frac{\Delta t}{\Delta y^2}(w_{m+(N_x+1)}^n - 2w_m^n + w_{m-(N_x+1)}^n) + \Delta t F_m^n \end{aligned} \tag{33}$$

We will write the attached system in matrix form for m equations:

$$\boldsymbol{w}^{n+1} = \boldsymbol{w}^n + \frac{\Delta t}{\Delta x^2}\boldsymbol{B}\boldsymbol{w}^n + \frac{\Delta t}{\Delta y^2}\boldsymbol{C}\boldsymbol{w}^n + \Delta t \boldsymbol{F}^n, \tag{34}$$

or explicitly, written in block matrices:

$$\begin{cases} \widetilde{\boldsymbol{w}}_1^{n+1} = \widetilde{\boldsymbol{w}}_1^n + \frac{\Delta t}{\Delta x^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_1^n + \frac{\Delta t}{\Delta y^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_1^n + \Delta t \boldsymbol{F}_1^n, \\ \widetilde{\boldsymbol{w}}_2^{n+1} = \widetilde{\boldsymbol{w}}_2^n + \frac{\Delta t}{\Delta x^2}\boldsymbol{B}_1\widetilde{\boldsymbol{w}}_2^n + \frac{\Delta t}{\Delta y^2}(\boldsymbol{I}\widetilde{\boldsymbol{w}}_1^n - 2\boldsymbol{I}\widetilde{\boldsymbol{w}}_2^n + \boldsymbol{I}\widetilde{\boldsymbol{w}}_3^n) + \Delta t \boldsymbol{F}_2^n, \\ \vdots \\ \widetilde{\boldsymbol{w}}_{N_y+1}^{n+1} = \widetilde{\boldsymbol{w}}_{N_y+1}^n + \frac{\Delta t}{\Delta x^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_{N_y+1}^n + \frac{\Delta t}{\Delta y^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_{N_y+1}^n + \Delta t \boldsymbol{F}_{N_y+1}^n, \end{cases} \tag{35}$$

where the vector of unknowns is denoted as blocks of vectors:

$$\boldsymbol{w}^{n+1} = \begin{bmatrix} \widetilde{\boldsymbol{w}}_1^{n+1} \\ \widetilde{\boldsymbol{w}}_2^{n+1} \\ \vdots \\ \widetilde{\boldsymbol{w}}_{N_y-1}^{n+1} \\ \widetilde{\boldsymbol{w}}_{N_y+1}^{n+1} \end{bmatrix} \tag{36}$$

We note vector $\widetilde{\boldsymbol{w}}_s^{n+1}$ (where s is an arbitrary line) as being:

$$\widetilde{\boldsymbol{w}}_s^{n+1} = \begin{bmatrix} w_{1+(s-1)(N_x+1)}^{n+1} \\ w_{2+(s-1)(N_x+1)}^{n+1} \\ \vdots \\ w_{N_x+(s-1)(N_x+1)}^{n+1} \\ w_{s(N_x+1)}^{n+1} \end{bmatrix} \tag{37}$$

Matrix **B** is

$$\boldsymbol{B} = \begin{pmatrix} -2\boldsymbol{I} & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & \boldsymbol{B}_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \boldsymbol{B}_1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{B}_1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & \boldsymbol{B}_1 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & -2\boldsymbol{I} \end{pmatrix}_{m\times m} \tag{38}$$

Submatrix $\mathbf{B_1}$ is

$$\boldsymbol{B}_1 = \begin{pmatrix} -2 & 0 & 0 & 0 & \dots & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & -2 & 1 & \dots & 0 & 0 \\ 0 & 0 & 1 & -2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -2 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & -2 \end{pmatrix}_{(N_x+1)\times(N_x+1)} \tag{39}$$

Matrix **C** is

$$\boldsymbol{C} = \begin{pmatrix} -2\boldsymbol{I} & 0 & 0 & 0 & \dots & 0 & 0 \\ \boldsymbol{I} & -2\boldsymbol{I} & \boldsymbol{I} & 0 & \dots & 0 & 0 \\ 0 & \boldsymbol{I} & -2\boldsymbol{I} & \boldsymbol{I} & \dots & 0 & 0 \\ 0 & 0 & \boldsymbol{I} & -2\boldsymbol{I} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & -2\boldsymbol{I} & \boldsymbol{I} \\ 0 & 0 & 0 & 0 & \dots & 0 & -2\boldsymbol{I} \end{pmatrix}_{m\times m}, \quad \boldsymbol{I} \in \mathbb{R}_{(N_x+1)\times(N_x+1)} \tag{40}$$

where $\boldsymbol{I}$ is the identity matrix.

Special attention is given to the right-hand side term $\boldsymbol{F}^n$. Thus, for example, we have:

$$\boldsymbol{F}_1^n = \frac{1}{\Delta t}\left[\widetilde{\boldsymbol{w}}_1^{n+1} - \left(\widetilde{\boldsymbol{w}}_1^n + \frac{\Delta t}{\Delta x^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_1^n + \frac{\Delta t}{\Delta y^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_1^n\right)\right] \tag{41}$$

$$\boldsymbol{F}_{N_y+1}^n = \frac{1}{\Delta t}\left[\widetilde{\boldsymbol{w}}_{N_y+1}^{n+1} - \left(\widetilde{\boldsymbol{w}}_1^n + \frac{\Delta t}{\Delta x^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_{N_y+1}^n + \frac{\Delta t}{\Delta y^2}(-2\boldsymbol{I})\widetilde{\boldsymbol{w}}_{N_y+1}^n\right)\right] \tag{42}$$

3. NUMERICAL EXAMPLES AND RESULTS

We consider the next following problem of the parabolic equation for the 1D case:

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = xe^t - 6x, \quad 0 < t < T, 0 < x < 1, \tag{43}$$

With the following Dirichlet-type boundary conditions:

$$\begin{gathered} u(0,t) = 0, 0 \le t \le T \\ u(1,t) = 1 + e^t, 0 \le t \le T \end{gathered} \tag{44}$$

Along with the exact solution:

$$u(x,t) = x(x^2 + e^t), \quad 0 \le x \le 1, \tag{45}$$

To draw valid conclusions, we have chosen the following computation parameters: the number of intervals on x equals to $N_x = 100$ and the final simulation time $T_{final} = 0.1s$. For the selected time moment, the numerical solution of the equation will be calculated using three methods: the classical Crank-Nicolson method, the POD method without basis update, and the POD method with basis update.

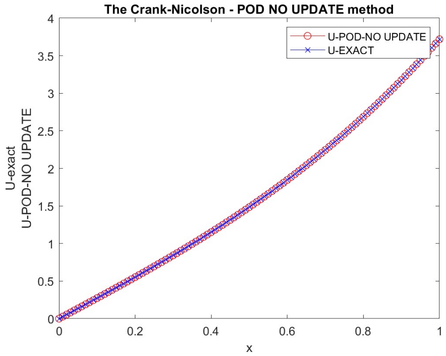

Fig. 1 – Comparison between the exact solution and the solution calculated using the Crank-Nicolson-POD method without basis update

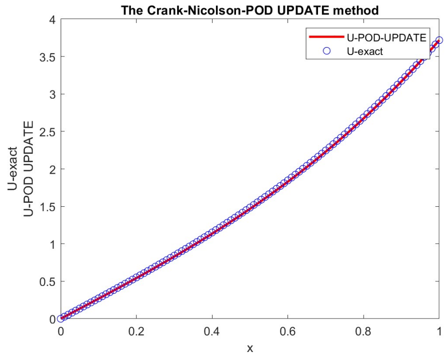

Fig. 2 – Comparison between the exact solution and the solution calculated using the Crank-Nicolson-POD method with basis update

Method	Time-CPU	POD basis	$\|e_n\|_2 = \|\boldsymbol{u}_{exact} - \boldsymbol{w}^{*n}\|_2$	$RMSE = \sqrt{\|\boldsymbol{e}_n\|_2^2/(N-1)}$
CN-classic	9.76	0	0.073	7.3529e-4
POD NO UPDATE	4.14	1	0.0412	0.0042
POD UPDATE	5.72	21	0.9188	0.0928

We consider the next following problem of the parabolic equation for the 2D case:

$$\frac{\partial u}{\partial t} - 2\frac{\partial^2 u}{\partial x^2} - 2\frac{\partial^2 u}{\partial y^2} = 1 + 2\,sin(x) + 2\,cos(y)\,, \quad 0 \le t \le T, 0 \le x \le \frac{\pi}{2}, 0 \le y \le \frac{\pi}{2} \tag{46}$$

With the following Dirichlet-type boundary conditions:

$$u(0,y,t) = t + cos(y)\,, u\left(\frac{\pi}{2},y,t\right) = t + 1 + cos(y)\,, 0 \le t \le T \quad u(x,0,t) = t + 1 + sin(x)\,, u\left(x,\frac{\pi}{2},t\right) = t + sin(x)\,, 0 \le t \le T \tag{47}$$

Along with the exact solution:

$$u(x,y,t) = t + sin(x) + cos(y)\,,\ \ 0 \le t \le T, 0 \le x \le \frac{\pi}{2}, 0 \le y \le \frac{\pi}{2} \tag{48}$$

To draw valid conclusions, we have chosen the following computation parameters: the number of intervals on x equals to $N_x = 40$, the number of intervals on y equals to $N_y = 50$ and the final simulation time $T = t_f = 2\ s$. For the selected time moment, the numerical solution of the equation will be calculated using three methods: the classical FTCS method, the POD method without basis update, and the Crank-Nicolson-POD method with basis update.

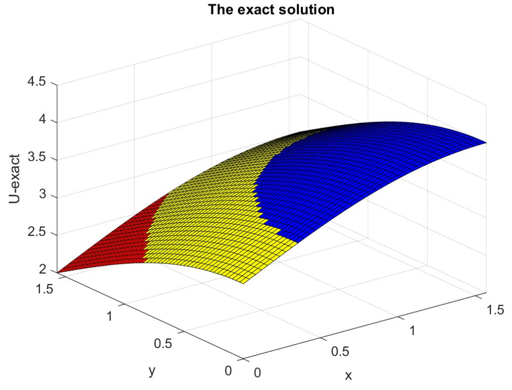

Fig. 3 – The exact solution

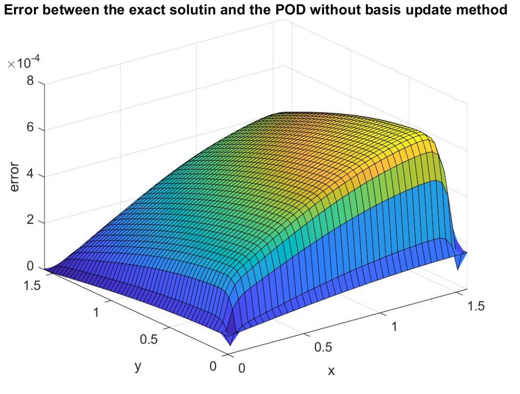

Fig. 4 – The absolute error between the exact solution and the solution calculated with the POD method without basis update

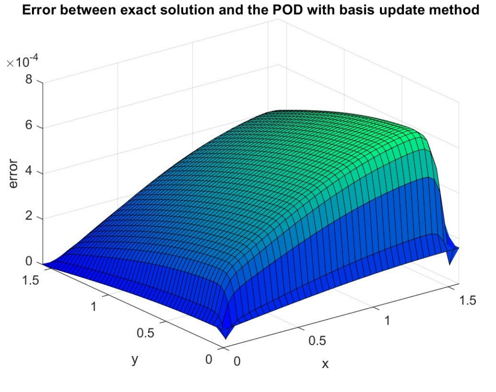

Fig. 5 – The absolute error between the exact solution and the solution calculated with the POD method with basis update

Method	Time-CPU	POD basis	$\|\boldsymbol{e}_n\|_2 = \|\boldsymbol{u}_{exact} - \boldsymbol{w}^{*n}\|_2$	$RMSE = \sqrt{\|\boldsymbol{e}_n\|_2^2/(N-1)}$
FTCS-classic	591.38	0	0.060035	0.00184
POD-NO UPDATE	46.12	1	0.02707	8.3009e-04
POD UPDATE	22.67	30	1.24149	0.03806

4. CONCLUSIONS

In conclusion, in this work, we utilized the POD basis to derive low-order finite difference schemes for the parabolic equation. We analyzed the errors between the exact solution and the solution obtained based on the implementation of finite difference schemes, demonstrating that our current method has improved and innovated existing approaches. We validated the correctness of our theoretical results with numerical examples.

When solving engineering problems, data can be interpolated or assimilated by acquiring information from experiments to organize instantaneous moments (i.e., snapshots) and the POD basis. As a result, computational efficiency is significantly increased, and time and resource-consuming calculations in the computation process are greatly reduced. The aim of future research in this field is to develop and apply the low-order POD method for numerical simulations of a realistic system to estimate, in a relatively short time with relatively small errors, the behavior of various flows around obstacles. This method can be implemented to calculate numerical solutions for more complex partial differential equations, such as the Navier-Stokes equations.

The use of the POD-Galerkin method on finite difference schemes led to increased computational efficiency. Dimensionality reduction through the use of POD basis allowed a significant acceleration of calculations, enabling the solution of larger and more complex problems in a shorter time.

This approach significantly enhanced computational efficiency, reducing the time and resources required in the calculation process. Thus, there is a great potential for many people to utilize the techniques presented in this work.

REFERENCES

[1] A. Solomon, V. Oltei, A. Bogoi, Some practical remarks in solving partial differential equations using reduced order schemes obtained through the POD method, *INCAS BULLETIN*, vol. **14**, issue 1, 187-196, (print) ISSN 2066–8201, (online) ISSN 2247–4528, ISSN–L 2066–8201, https://doi.org/10.13111/2066-8201.2022.14.1.15, 2022.

[2] S. Dănăilă, D. Isvoranu, C. Levențiu, A. Bogoi, A Reduced Order Model based on Large Eddy Simulation of Turbulent Combustion in the Hybrid Rocket Engine MATEC Web Conf., *9th EASN International Conference on "Innovation in Aviation & Space"*, Volume **304**, 2019.

[3] Z. Luo, R. Wang, J. Zhu, Finite difference scheme based on proper orthogonal decomposition for the nonstationary Navier-Stokes equations, *Science in China Series A: Mathematics*, Vol. **50**, No. 8, 1186–1196, 2007.

[4] A. Bogoi, *Solved and proposed applications to differential equations*, Editura Monitorul Oficial, 2014, (440 pag), ISBN: 978-973-567-893-7.

[5] N. Padmanabhan, *Reduced order model of a convection-diffusion equation using Proper Orthogonal Decomposition*, 2023.

[6] J. Weiss, *A Tutorial on the Proper Orthogonal Decomposition*, 2019.

[7] A. Bogoi, S. Dănăilă, D. Isvoranu, *Ecuațiile generale de transport ale dinamicii fluidelor*, Editura Monitorul Oficial, (560 pag), 2021.

[8] M. C. Brindise, P. P. Vlachos, Proper orthogonal decomposition truncation method for data denoising and order reduction,*Exp Fluids* **58**, 28, 2017.
[9] S. Volkwein, *Proper Orthogonal Decomposition: Theory and Reduced-Order Modelling*, University of Konstanz Department of Mathematics and Statistics, Lecture Notes, 2013.
[10] E. Al-taai & A. Abdali, Improved Alternating Direction Implicit Method, *Journal of Physics: Conference Series*, 2020.
[11] Zhendong Luo, Hong Li, Ping Sun, A reduced-order Crank–Nicolson finite volume element formulation based on POD method for parabolic equations, *Applied Mathematics and Computation*, Volume **219**, Issue 11,2013.
[12] Yukihito Suzuki, Masashi Ohnawa, GENERIC formalism and discrete variational derivative method for the two-dimensional vorticity equation, *Journal of Computational and Applied Mathematics*, Volume **296**, 2016.