Proceedings of the 1st international Symposium on AI and Fluid Mechanics
Paper No S4 P3

A Brief Survey on Data-driven Convergence Acceleration Methods in Computational Fluid Dynamics

[1]**Xukun Wang**; [2,3]**Chenjia Ning**; [2,3]**Zikun Liu**; [2,3]**Weiwei Zhang***

[1]*ETSIAE-UPM-School of Aeronautics, Universidad Politécnica de Madrid, Madrid, 28040, Spain*

[2]*School of Aeronautics, Northwestern Polytechnical University, Xi'an, 710072, China*

[3]*International Joint Institute of Artificial Intelligence on Fluid Mechanics, Northwestern Polytechnical University, Xi'an, 710072, China*

**Correspondence:aeroelastic@nwpu.edu.cn*

Abstract. With the growing complexity and scale of fluid-related problems, the overhead of computational fluid dynamics (CFD) simulations has becoming heavier than ever before. Reducing computational costs has become a critical priority, especially for steady-state simulations which has been used extensively in industrial applications. Traditional convergence acceleration methods focus on enhancing iterative schemes itself, but advances in data-driven techniques have opened new pathways for more efficient computation, including not only Machine Learning (ML) integrated methods but also other attempts to fully leverage the historic data generated during the simulation. This paper briefly reviews three categories of data-driven convergence acceleration methods: Vector Extrapolation Methods (VEMs), Reduced Order Models (ROMs)-based, and Neural Network (NN)-based techniques. These methods aim to reduce the total number of iterations by restarting computations with flow fields reconstructed from historic snapshots. The lower residual levels of these reconstructed fields accelerate convergence, thereby minimizing iteration steps and improving solver performance. For each class of methods, fundamental idea and essential formulation are summarized, and brief comparison is also given. Through an analysis of state-of-the-art developments, this survey underscores the great potential of data-driven convergence acceleration methods in CFD and presents an optimistic outlook for future research in this domain.

Keywords: Computational fluid dynamics, convergence acceleration, steady state flow simulation, data-driven methods.

1. Introduction

To obtain steady-state flow solutions, pseudo-time marching is commonly used in CFD to simulate compressible flows. This approach iterates pseudo-unsteady governing equations from an initial state until the residual decreases sufficiently. While explicit time-marching schemes are straightforward, they are constrained by the Courant–Friedrichs–Lewy (CFL) condition [1], which limits the time step size and increases the total number of iterations. Since the final steady-state solution is the primary goal, the precision of intermediate non-convergent solutions is less critical. This observation leads to the idea of increasing the time step size as much as possible, disregarding temporal accuracy. Implicit time marching scheme breaks the limitation of CFL number by evaluating the residual of equations at the newest time step. Despite the enlarged time step and thus the reduced total iteration steps, implicit scheme needs more computation overhead in each step, where a large linear system has to be solved. Successful solving strategy includes the Lower-Upper Symmetric Gauss–Seidel (LU-SGS) method [2,3] and Generalized Minimal Residual (GMRES) method [4,5]. With these advanced methods, implicit scheme still has advantages over explicit scheme and is widely used in steady CFD solver. Many classical methods are coupled with implicit scheme to accelerate the convergence, such as local time step [6], multigrid (MG) methodology [7], and large time step schemes [8]. Apart from increasing size of time step, another fundamental idea of accelerating convergence is to eliminate or damp the error during iteration. One of the most widely used method is residual smoothing [9], which substitutes the weight-averaged residuals of adjacent grid elements for the original residual of the current control volume and effectively damps the error of high-frequency. The MG methodology [7,10,11,12] combines the both ideas mentioned above, where low-wavenumber errors are damped via transferring solution between refined and coarsen grids while high-wavenumber errors are eliminated by smoother, i.e. typical iteration. What's more, larger time steps can be allowed on the coarsen grids. Combining these two aspects, MG method has outstanding effects of accelerating convergence. As for these classical acceleration methods, Mavriplis [13] conducted a comprehensive research. More recently, more

advanced methods have been used to damp the error, such as eigenvalue analysis[14],Selective Frequency Damping(SFD) method[15,16]and residual recombination[17,18,19].

All the convergence acceleration methods mentioned above is focused on improving the iteration scheme itself, e.g. increasing the rate of convergence in each iteration step. They can be termed 'One-step' method, as the updating of flow field only depends on the most recent solution while more previous solutions are not used. Considering the fact that a large amount of intermediate non-convergent solutions are generated during iteration, abundant information contained in the historic solutions are wasted. Therefore, other acceleration methods aim to reduce the total number of iteration steps based on the thought of making full use of the historic solutions during iteration. Taking this feature into account, this type of methods can be termed 'data-driven' convergence acceleration method because the data ,i.e. snapshots of historic solutions, are stored first and used afterwards. Another feature of data-driven convergence acceleration method is that the acceleration is applied periodically as it contains the phase of collecting snapshots from the original iteration. The general framework is shown in Fig.., under which a large amount of acceleration methods can be summarized and compared each other. Through wide literature research and deep analysis, data-driven acceleration methods can be roughly classified into three categories. The first class is the classical Vector Extrapolation Method (VEM), such as Minimal Polynomial Extrapolation (MPE) and Reduced Rank Extrapolation (RRE) [20,21]. Base on the property of exponential decay of error, approximation of convergent solution can be expressed as a linear combination of historic snapshots. The second class focuses on establishing data-driven Reduced Order Model (ROM) based on snapshots to reconstruct a better solution via the superposition of several modes. Some popular ROMs, such as Proper Orthogonal Decomposition (POD) [22] and Dynamic Mode Decompostion (DMD) [23, 24] are used to accelerate the convergence. The third class of methods merged recently with the rise of Machine Learning (ML) methods, especially Neural Network (NN), to couple with traditional CFD solver and to accelerate the convergence. Typically, the mapping between the initial flow field or the intermediate non-convergent solutions and final convergent solution is established by the powerful approximation ability of deep NN. After the NN is trained based on the data prepared before, it can be used as a surrogate model to skip parts of the iteration steps so as to accelerate the convergence. As a brief survey on data-driven convergence acceleration methods in CFD, all the three classes of methods and their application in CFD will be introduced in the following parts. It's worth mentioning that we restrict our scope on accelerating steady Euler or RANS simulation throughout this article.

The paper is organised as follows. The general iteration formulation and acceleration framework are given in Section 2. Then, in Section 3,4 and 5, we will introduce the VEM-based, ROM-based and NN-based convergence acceleration methods respectively. In each section, basic formulas are listed briefly. The state-of-the-art and application in CFD of each kind of methods are also summarized. Finally, in Section 6, a future outlook for data-driven convergence acceleration methods is given.

2.General iterative formula and acceleration framework

Considering a discretized nonlinear problem:

$$\mathrm{N}(x) = 0 \tag{1}$$

a general iteration formula can be constructed for both explicit and implicit time-marching scheme as:

$$U^{k+1} = g(U^k) = U^k + B^k R^k \tag{2}$$

where $U^k \in \mathbb{R}^n$ is the solution vector, g denotes the iteration function, $R^k \in \mathbb{R}^n$ is the residual and $B^k \in \mathbb{R}^n$ depends on the chosen iteration scheme. $n = n_i^d$ is the DOFs of the problem, n_i is the DOFs in each dimension and d is dimension number. When solving a linear system, B^k is stationary with k and presents the effect of classical iteration, e.g. Jacobi and Gauss-Seidel iteration. While in case of solving steady-state flow, B^k depends on the pseudo-time marching scheme.

In CFD fields, the iteration scheme usually couples with acceleration methods, e.g. local time step and residual smoothing, which makes the iterations in CFD nonstationary globally but stationary locally. With the general basic iterative formulation, different acceleration methods can be compared and analyzed conveniently.

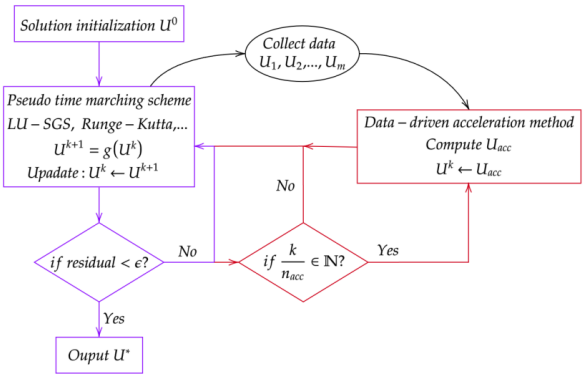


Figure 1 General framework of data-driven acceleration methods.

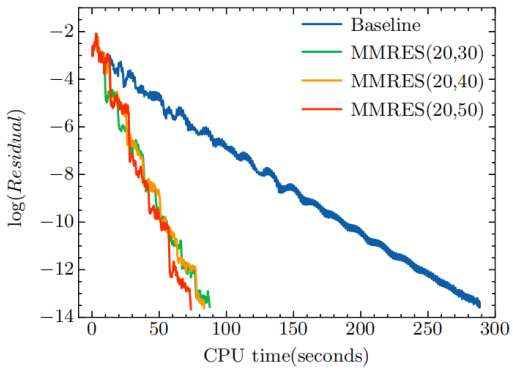


Figure 2 Typical convergence history of data-driven acceleration methods

Holding the basic iteration scheme in hands, data-driven acceleration methods can be easily coupled by inserting a additional module to the original CFD solver. The general acceleration framework is shown in Figure 1. The blue flow diagram on the lefthand side is the typical solution flow of a steady CFD solver, which iterates the solution until the residual is reduced below a tolerance ϵ, and the red flow diagram on the righthand sideshows the acceleration module which is executed periodically, every n_{acc} steps. Finally, the black flow diagram on the top illustrates the process of collecting data (flow field snapshots) from original CFD solver and feeding them as input into the acceleration module. Based on the collected data, a better approximation solution U_{acc} for acceleration can be computed and brought back to the original iteration. The differences of various acceleration methods lie in the approach of computing the new flow field U_{acc}, based on which data-driven acceleration methods can be classified into three categories roughly and all of them will be introduced and summarized briefly in the following section. To give readers an intuitional impression, a typical residual convergence history is given in Figure 1 from [25], from which it can be seen clearly that data-driven convergence acceleration method makes the residual decrease in a 'cliff-like' style. The example shown here illustrates a speed-up of $4\times$ in terms of CPU time.

3 VEM-based convergence acceleration methods

The first category of convergence acceleration methods is the direct application of vector extrapolation method (VEM), whose origin is closely linked to the fix-point (FP) iteration. VEM considers the FP iteration as a 'sequence generator' and the explicit information about iteration is not needed. Based on the sequence generated, VEM approximates the convergent solution by extrapolated value. Since its first appearance in mid-1950s, large amounts of methods have been proposed and researched comprehensively. Roughly speaking, VEMs can be divided into two families: epsilon methods and polynomial methods. Originate in scalar case, epsilon algorithms were extended to vector case, resulting in the Wynn's vector epsilon algorithm (VEA) [26] and Brezinski's topological epsilon algorithm (TEA) [27]. As for the polynomial methods, minimal polynomial extrapolation (MPE) of Cabay and Jackson [20] and reduced rank extrapolation (RRE) of Mešina [28] and Eddy [21] are two very important methods and further developed by Pugachev [29] to modified minimal polynomial extrapolation (MMPE). However, as MPE and RRE are most popular in early research on accelerating CFD, we will only focus on these two methods in this section. As for comprehensive comparison of VEMs, which is beyond the scope of this paper, please reader refer to reviews by Smith et. al. [30],

Jbilou and Sadok [31], Sidi [32] and Brezinski and Redivo-Zaglia [33]. Moreover, another important method called Anderson Acceleration (AA) [34] proposed by Anderson in 1960s, or Anderson mixing, is also briefly introduced considering its close relation with RRE and wide application in accelerating iteration. All the methods, including MPE, RRE, AA and quasi-Newton method, have deeper mathematical meaning and can be summarized under the framework of Shanks sequence transformation [35,36].

3.1 Introduction of MPE and RRE

MPE and RRE are derived from the fact of exponential evolution of error e and increments between iteration Δx:

$$e_{k+j} = M^j e_k, \Delta x_{k+j} = M^j \Delta x_k, j = 1,2,... \tag{1}$$

where the subscript denotes the iteration index and M is the iteration matrix. However, if the spectral radius of matrix $\rho(M)$ is very close to 1, then the sequence generated by FP iteration converges to the exact solutions slowly. This situation will deteriorate as the dimension of the problem becomes large. To conquer the problem, VEMs were developed to accelerate the convergence rate of the sequence. The basic idea is to approximate the exact solution by a linear combination of non-convergent solutions:

$$s^{(k,m)} = \sum_{j=1}^{m} \gamma_j^{(k,m)} x_{k-j-1}, \sum_{j=0}^{m-1} \gamma_j^{(k,m)} = 1 \tag{2}$$

where the superscript (k,m) means that the extrapolation is executed at k^{th} iteration step and m previous solutions are used. Using the minimal polynomial of M, the coefficients $\gamma_j^{(k,m)}, j = 1,2,...$ for MPE and RRE can be computed as:

$$\gamma_j^{MPE} = \frac{c_j}{\sum_{i=1}^{m} c_i}; \gamma_j^{RRE} = \min_{\gamma_1,\gamma_2,...,\gamma_m} \left\| \sum_{j=1}^{m} \gamma_j \Delta x_j \right\| \tag{3}$$

where $c' = [c_1, c_2, ..., c_{m-1}] = \min_{c' \in \mathbb{R}^{m-1}} \left\| U_{k-1} c' + \Delta x_m \right\|$.

3.2 Application in CFD

VEMs had been used in fields of CFD pretty soon after the emerge of VEMs because the close relationship between FP iteration and CFD. As far as authors known, in 1977, Hafez [37] first applied the Aitken's Δ^2 method, a special version of RRE where only three snapshots are used to execute extrapolation, to accelerate the line successive over-relaxation (SOR) iteration for solving steady Euler transonic flow under the FDM framework. Afterwards, he extended these methods to accelerate solving potential equations [38] and Jameson's Runge-Kutta multigrid method [39] in FVM. In 1987, Reddy and Jacocks [40] successfully accelerate the implicit scheme by RRE to solve Euler equations combined with other acceleration methods. However, due to the limited storage, the number of snapshots, m, is restricted to only 2 to 4. In [41], Sidi and Mark compared MPE with RRE when applying to the Runge-Kutta scheme and first took the strategy of collecting snapshots at intervals. Cheung et. al. [42] applied VEM to accelerate Richardson iteration for solving 3D hypersonic flow. More recently, Aleksandar and Joseph [43] researched the acceleration effect of RRE in both density- and pressure-based solver.

4 ROM-based convergence acceleration methods

With the boosting development of computer science and numerical simulation, large amount of data are generated and thus pacing the way of development data science and data-driven reduced order models (ROM). In fields of fluid mechanics, proper orthogonal decomposition (POD) and dynamic mode decomposition (DMD) are two of the most popular ROM when analysing the flow field. Due to its property of purely data-driven and easiness to conduct, researchers started to apply them to accelerate the convergence of iteration. Different from their usage in flow analysis, where the decomposition is

conducted 'offline' after the computation is finished, ROMs are applied 'online' to the history of iteration periodically. The basic idea of ROM-based convergence acceleration methods is to project the solution into a low-dimensional subspace and find a better solution in it. If this solution has a smaller residual when brought back into the original iteration, then it accelerates the convergence because of the drop of residual. The schematic diagram is given in Figure 3, and the solving process is composed by the following three basic steps:

Step 1 Original iteration: Update the solution using time-marching (iteration) scheme, during which m snapshots of flow fields are collected as $\mathbf{U}_1, \mathbf{U}_2, ..., \mathbf{U}_m$;

Step 2 ROM establishment: Based on these m snapshots, a ROM is established by decomposing the flow into several ROM modes $\phi_1, \phi_2, ...$;

Step 3 Reconstruction: Reconstruct the solution in the reduced-dimensional subspace $\tilde{\mathbf{U}} = \sum_i c_i \phi_i$;

Step 4 Back substitution: Bring $\tilde{\mathbf{U}}$ back to the solver and substitute the newest solution. Go back to Step 1 and repeat this cycle until achieving the convergence.

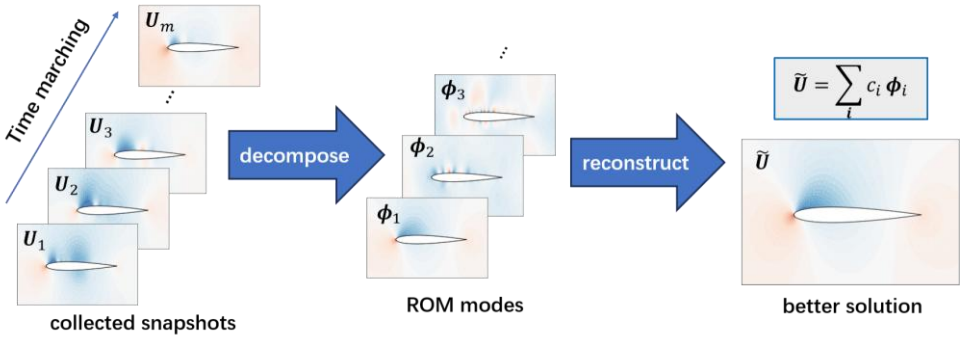


Figure 3 One cycle of acceleration of ROM-based methods. $\mathbf{U}_1, \mathbf{U}_2, ..., \mathbf{U}_m$ are collected snapshots; $\phi_1, \phi_2, ...$ are ROM modes; c_i are the mode coefficients; $\tilde{\mathbf{U}}$ is reconstructed by several ROM modes.

The first question is to choose a suitable ROM to reduce the dimension of solution. Both POD and DMD have been used to accelerate the convergence. Other simplified ROM, such as mean-based ROM, are also applied without losing effectiveness. Once the ROM is determined, the second question is how to choose the mode coefficients c_i so as to reconstruct a better solution. Different approaches are adopted to determine these coefficients as follows:

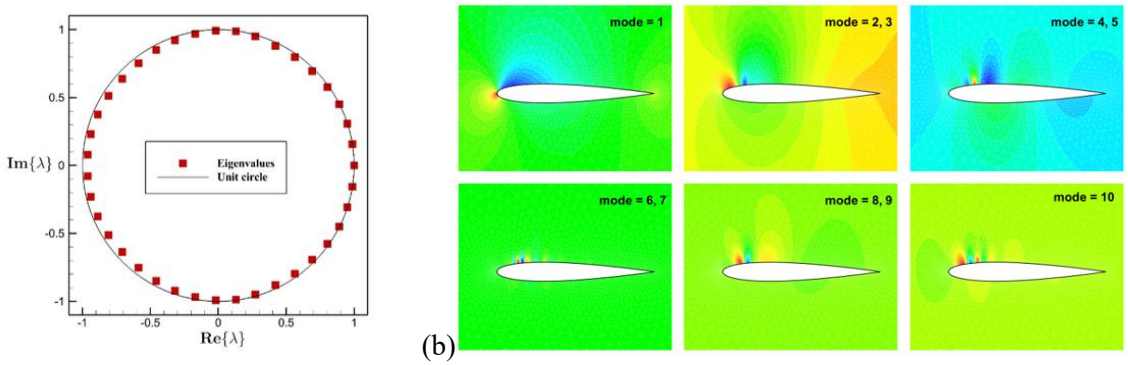
- Mode truncation: $\tilde{\mathbf{U}}_m = \sum_{i=1}^{t} c_i \phi_i$ and setting the number of modes to be retained, $t < r$
- Linear minimal residual: (1) $\mathbf{c} = -[\mathbf{\Psi}^T \mathbf{\Psi}]^{-1} \mathbf{\Psi}^T \bar{\mathbf{R}}$ (minimizes the 2-norm of residual) ; (2) $\mathbf{c} = -[\mathbf{\Phi}^T \mathbf{J} \mathbf{\Phi}]^{-1} \mathbf{\Phi}^T \bar{\mathbf{R}}$ (minimizes the $\mathbf{J}$ - norm of solution error) where $\mathbf{J}\mathbf{\Phi} = \mathbf{\Psi}$, $\mathbf{J} = \partial \mathbf{R} / \partial \mathbf{U}$ and $\bar{\mathbf{R}}$ is the residual of a base flow
- Nonlinear optimization: solve the optimization problem: $\mathbf{c} = \arg\min_{\mathbf{c} \in \mathbb{R}^r} \left\| \mathbf{R}(\tilde{\mathbf{U}}) \right\|$

4.1 POD

Using different ROMs and reconstruction approaches, researchers developed a large amount of ROM-based convergence acceleration methods. Ekici et. al. [44] proposed a POD-based stabilization methods for explicit flow solvers and extended it to accelerate the convergence [45] and continuous adjoint solvers [46] using the approach 2 mentioned above. The results proved that the proposed method can effectively accelerate the convergence of both pressure- and density-based solvers. More recently, Cao et.al. [47] went along the method of combining POD and nonlinear optimization and proposed Online Dimension Reduction Optimization (ODRO) method, with which they successfully accelerate the convergence of RANS simulation and compute the unstable steady solution of NS equations.

4.2 DMD

Apart from POD, Liu et. al. [48] developed Mode Multigrid (MMG) to accelerate the convergence of inviscid flow. The results of numerical experiments proved that DMD is highly suitable for mode truncation and only the single mode with largest amplitude should be retained and all the other high-order modes are recommended to be truncated. It is based on the observation that the eigenvalue of first DMD mode equals to one exactly, which means this mode represents the invariant components of the flow field, while other eigenvalues are either complex or smaller than one, corresponding to the oscillating and damped modes. A typical example of distribution of DMD eigenvalues is plotted in Figure 4 (a). What's more, it is also found that the high-order DMD modes represents high-wave number error during the iteration, as shown in Figure 4 (b). Truncating these high-order modes has the equivalent effects of filtering oscillating errors existing in the flow field. Because of its similar acceleration mechanics compared to traditional Multigrid method and the filtering is conducted in mode space, this method is termed as Mode Multigrid. Chen et. al. [49] uses it to accelerate the convergence of steady adjoint equations. Furthermore, Tan et. al. [50] extends this method to solve three-dimensional complex turbulent. Wang et. al. [51] use it to stabilize the iteration process of implicit time-marching scheme. More recently, Amirhossein et. al. [52] extends the classic over-relaxation scheme by using DMD to identify and relax the dominant slow-converging modes of the solution towards their steady state, showing acceleration effects when solving steady flows. Different from previous works, they do not truncate any mode but reconstruct a new solution by means of relaxing the slow dominant modes, which is beyond the mentioned above.



(a) (b)

Figure 4: (a)Distribution of eigenvalues of DMD modes in complex plane and (b) DMD modes of pressure from inviscid NACA0012 simulation (taken from [61]).

Table 1: Summary of ROM-based convergence acceleration methods

literature	ROM	reconstruction approach
Djeddi et.al. [44]	POD/meam-based ROM	linear minimization (1)
Andrew et.al. [46]	POD/meam-based ROM	linear minimization (1)
Liu et.al. [48]	DMD	mode truncation: $c_1 = 1; c_i = 0, i = 2,...,r$
Chen et.al. [49]	DMD	mode truncation: $c_1 = 1; c_i = 0, i = 2,...,r$
Tan et.al. [50]	DMD	mode truncation: $c_i = \begin{cases} 0 & for \quad i = 1,...,t \\ 1 & for \quad i = t+1,...r \end{cases}$
Cao et.al. [47]	POD	mode truncation ($r = 5$)+nonlinear optimization
Wang et. al.[25]	mean-based ROM	linear minimization (2)
Amirhossein et. al.[52]	DMD	relaxation

4.3 Mean-based ROM

In [44,25], mean-based ROM is applied but using different minimal residual approaches. In [44], linear minimal residual method (2) is used while in [25] Wang et. al. proposed the mean-based minimal residual (MMRES) method by using the method (2). The significant reduction of time complexity from

$O(n^2)$ to $O(n)$ is proved mathematically and numerically. Its acceleration effects are achieved from simple inviscid flow around airfoil to complex three-dimensional transonic turbulent flow past wing.

Finally, all these works are concluded in the **Table 1**, where applied ROM and reconstruction approaches are listed. It is also worth mentioning that POD and mean-based ROM is mathematically equivalent if the finial target is to minimizing the residual, as the subspace spanned by POD modes are the same as that generated by mean-based ROM and the solution with minimal residual is also the same.

5 NN-based convergence acceleration methods

With the rapid development of artificial intelligence (AI) technology, machine learning (ML) technology with neural network (NN) as the core method, also injects new vitality into acceleration methods for CFD and has shown unprecedented potential. This kind of AI-based method has gradually become a hotspot in the research field of CFD due to its powerful nonlinear fitting ability and high-dimensional data processing ability, which is expected to break the bottleneck of traditional numerical simulation and lead the development of computational fluid dynamics into the era of intelligence. It is applied in a flexible form, which not only shows excellent performance in dealing with complex flow problems, but also plays a key role in several CFD simulation procedures. By optimizing the numerical solver, improving the prediction of initial conditions, accelerating the iterative process, and enhancing the accuracy of turbulence models, ML-driven techniques are gradually replacing the traditional numerical methods. However, it is not the purpose of this article to review comprehensively all the ML-enhanced CFD as [53], but to concentrate on the works on accelerating the convergence in steady CFD computation using ML techniques. Considering the fact that overwhelming number of these methods are based on NN technique, we term them as NN-based convergence acceleration method.

5.1 Flow field modeling

Relying on the powerful nonlinear and data mining exploits of NNs, a number of researches have focused on the direct data-driven NN mapping models to replace CFD solvers completely for specific problems. These data-driven models are trained using a large amount of CFD data and are designed to rapidly predict the evolution of complex flow fields, thus significantly reducing the computational time of traditional CFD simulations. Typically, these methods take parameters of PDE as input, e.g. AOA, Mach number, Reynolds number and parameters describing the geometry, and outputs the desired flow field, therefore, we term these methods as flow field modeling. Duru et. al. [54] achieved pressure field prediction for airfoils with different thicknesses and curvatures by means of a convolutional self-encoder structure, which reduced the computational time by nearly four orders of magnitude compared to CFD simulations. Hu et. al. [55, 56] carried out a lot of works on flow field modeling based on convolutional neural networks (CNN) by adding self-attention mechanism, constructing input features for flow field modeling, and transforming the input features using Jacobi matrix from the perspective of transformation domain, etc. The two-dimensional and three-dimensional lifting configurations were used as validation examples, which greatly improved the accuracy of convolutional flow field modeling and achieved efficient and high-precision flow field prediction. In addition, there are a number of flow field modeling methods based on a variety of typical NN structures such as Pointnet [57], long short-term memory (LSTM) [58], and graph convolutional neural network [59].

Despite great speed-up achieved, the accuracy of flow field modeling can not be guaranteed and the unfairness of comparison are widely existed in published papers as pointed in [60]. The accuracy of the flow field obtained by the NN-based mapping model is far less than that of the CFD numerical simulation. In the study of CFD accelerated convergence, if we follow the idea of flow field modeling, it is an intuitive idea to replace the uniform incoming flow by the output of NN as the initial CFD flow field for calculation. This substitution follows the idea of providing a better initial guess with CFD solver to accelerate its convergence. Since the CFD solver will update the solution till the required accuracy is achieved, i.e., with further guidance from the physical equations, the results will obtain a accuracy

guaranteed by CFD solver. Hajgató et. al. [61] and Tsunoda et. al. [62] applied the above idea to the aerodynamic optimization design problems of a U-shaped duct and a 2D airfoil, respectively. Among them, [62] constructed a flow field prediction model with a CNN with grid point coordinates and distances from the wall as inputs and flow field as outputs, and provided better initial field results for CFD, which achieved more than three times acceleration.

5.2 Mapping from un-convergent to convergent flow

Similar to the VEM- and ROM-based convergence acceleration methods, there are also several research devoted to constructing a mapping between the un-convergent flow field and the convergent flow field. Obiols-Sales et. al. [63] proposed a CNN-based flow field accelerated convergence method, CFD-Net. Unlike directly inputting the solution of flow field modeling into the CFD solver as an initial guess, this method uses the NN model to construct a mapping from the un-converged flow field to the converged flow field. The method is divided into three phases: the first phase is the 'warm up', in which the CFD solver iterates K steps from the uniform incoming flow as a warm up ($K \ll$total iteration steps N). The second stage utilizes the NN for 'inference', where the input to the model is the flow field obtained from the K^{th} iteration step in the warm up stage, and output is the final converged flow field predicted by NN. The third stage is 'optimization', i.e., the flow field obtained from inference of NN is re-input into the CFD solver for iteration. When constructing the mapping between the un-convergent and convergent flow fields, the authors add the snapshots of the flow fields throughout the convergence process as training samples, which mitigates the failure and accuracy degradation of the input-output mapping relationship due to inconsistent convergence accuracy. Consistent with the idea of CFD-Net, Rojek et. al. [64] used OpenFOAM as a solver and test the convergence acceleration for chemical doping problem. They combined attention, affine transformation and encoder-decoder to form a mapping model with Pointnet as the basic structure. Instead of using the flow field snapshots of the whole convergence process as training samples, the method introduces the grid node features into the mapping process of the model in an encoded way, which achieves more than 30× acceleration from 2-dimensional to 3-dimensional cases. This method of using NN to construct the mapping from non-convergence to convergence does not need to take into account the complex structure in CFD simulation, and the algorithm is more stable. More recently, Zhang et. el. [65] extends this idea to unstructured mesh using a residual graph convolutional network (Res-GCN), which enlarges the flexibility of this method greatly.

5.3 NN-enhanced turbulence model

In RANS simulation, the construction of data-driven turbulence models using NN also enables to accelerate the convergence of numerical simulations. Zhu et. al. [66] directly reconstructed the mapping function between the turbulent eddy viscosity and the mean flow variable by means of an RBF-NN, which completely replaced the original partial differential equation model. Since this eddy viscosity model omits the partial differential equations for calculating the turbulent eddy viscosity, it has some time optimization and acceleration in the convergence iteration of CFD. Maulik et. al. [67] used a NN model for the on-the-fly prediction of steady state turbulent eddy viscosity profiles. In agreement with the idea of [66], due to the fact that the eddy viscosity model of NN excludes a set of additional equations for the turbulent eddy viscosity calculation, the time to solve the RANS problem is greatly reduced. From the results provided by the authors, utilizing the NN model obtained by training on the SA equations instead makes the simulation more than 5 × faster than normal RANS.

6 Conclusion

This review has provided an overview of three major categories of data-driven convergence acceleration methods in CFD: Vector Extrapolation Methods (VEMs), Reduced Order Models (ROMs)-based methods, and Neural Network (NN)-based techniques. Each of these approaches offers unique advantages and has been successfully applied to various CFD problems, demonstrating significant potential for reducing computational costs and improving solver performance.

- VEMs are well-established and have been widely used in early CFD applications, particularly for their simplicity and effectiveness in accelerating fixed-point iterations.

- ROMs provide a powerful framework for reducing the dimensionality of flow fields, enabling faster convergence by focusing on the most significant modes of the solution.
- NN-based methods leverage the power of machine learning to model complex flow behaviors and accelerate simulations.

While each method has its strengths, the choice of technique depends on the specific problem and computational resources available. Future research in this domain should focus on integrating these methods, exploring hybrid approaches, and further optimizing their performance for large-scale, real-world applications. As the development of high-performance computing, the parallelism of method is becoming an essential point. What's more, as computational power and data availability continue to grow, data-driven convergence acceleration methods are poised to play an increasingly important role in the evolution of CFD, paving the way for more efficient and accurate simulations in fluid dynamics.

References

[1] R. Courant, K. Friedrichs, and H. Lewy. 1967 On the partial difference equations of mathematical physics. *IBM Journal of Research and Development*, **11**(2):215–234, 1967
[2] Antony Jameson and Seokkwan Yoon. 1987 Lower-upper implicit schemes with multiple grids for the euler equations. *AIAA Journal*, **25**(7):929–935
[3] S. Yoon and A. Jameson. 1988 Lower-upper symmetric-gauss-seidel method for the Euler and Navier-Stokes equations. *AIAA Journal*, **26**(9):1025–1026
[4] Youcef Saad and Martin H. Schultz. 1986 GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, **7**(3):856–869
[5] Max Blanco and David W. Zingg. 1998 Fast newton-krylov method for unstructured grids. *AIAA Journal*, **36**(4):607–612
[6] A. Jameson, Wolfgang Schmidt, and E. L. I. Turkel. 1981 Numerical solution of the Euler equations by finite volume methods using Runge Kutta time stepping schemes. *Fluid Dynamics and Co-located Conferences* (American Institute of Aeronautics and Astronautics)
[7] Antony Jameson and Seokkwan Yoon. 1986 Multigrid solution of the Euler equations using implicit schemes. *AIAA Journal*, **24**(11):1737–1743
[8] ZhanSen Qian and Chun-Hian Lee. 2011 A class of large time step Godunov schemes for hyperbolic conservation laws and applications. *Journal of Computational Physics*, **230**(19):7418–7440
[9] R. Enander and A. Karlsson. 1995 Implicit explicit residual smoothing in multigrid cycle. *Aerospace Sciences Meetings* (American Institute of Aeronautics and Astronautics)
[10] D. J. Mavriplis. 1988 Multigrid solution of the two-dimensional Euler equations on unstructured triangular meshes. *AIAA Journal* **26**(7):824–831
[11] D. J. Mavriplis and A. Jameson. 1990 Multigrid solution of the Navier-Stokes equations on triangular meshes. *AIAA Journal*, **28**(8):1415–1425
[12] Mavriplis, D. J., & Venkatakrishnan, V. 1996 A 3D agglomeration multigrid solver for the Reynolds–averaged Navier–Stokes equations on unstructured meshes. *International Journal for Numerical Methods in Fluids*, **23**(6), 527-544.
[13] D. Mavriplis. 1998 On convergence acceleration techniques for unstructured meshes. *Fluid Dynamics and Co-located Conferences* (American Institute of Aeronautics and Astronautics)
[14] Sinan Eyi. 2016 Convergence Acceleration Based on Eigenvalue Analysis. *AIAA AVIATION Forum* (American Institute of Aeronautics and Astronautics)
[15] Espen Akervik, Luca Brandt, Dan S. Henningson, Jero Hœpffner, Olaf Marxen et. al. 2006 Steady solutions of the Navier-Stokes equations by selective frequency damping. *Physics of Fluids*, **18**(6), 2006.
[16] Frédéric Plante and Éric Laurendeau. 2018 Acceleration of Euler and rans solvers via selective frequency damping. *Computers and Fluids*, **166**:46–56

[17] V. Citro, P. Luchini, F. Giannetti, and F. Auteri. 2017 Efficient stabilization and acceleration of numerical simulation of fluid flows by residual recombination. *Journal of Computational Physics*, **344**:234–246
[18] Vincenzo Citro. Simple and efficient acceleration of existing multigrid algorithms. AIAA Journal, 57(6):2244– 2247, 2019.
[19] A. Dicholkar, F. Zahle, and N. N. Sorensen. 2022 Convergence enhancement of simple-like steady-state rans solvers applied to airfoil and cylinder flows. *Journal of Wind Engineering and Industrial Aerodynamics*, **220**
[20] S. Cabay and L. W. Jackson. 1976 A polynomial extrapolation method for finding limits and antilimits of vector sequences. *SIAM Journal on Numerical Analysis* **13**(5):734–752
[21] R. P. Eddy. 1979 EXTRAPOLATING TO THE LIMIT OF A VECTOR SEQUENCE, pages 387–396. Academic Press
[22] G. Berkooz, P. Holmes, and J. L. Lumley. 1993 The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, **25**:539–575
[23] P. J. Schmid. 2010 Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28
[24] P. J. Schmid, L. Li, M. P. Juniper, and O. Pust. 2010 Applications of the dynamic mode decomposition. *Theoretical and Computational Fluid Dynamics*, **25**(1-4):249–259
[25] Wang, Xukun and Liu, Yilang and Zhang, Weiwei 2024 A Data-Driven Convergence Booster of Steady-State Solvers: Mean-Based Minimal Residual. SSRN: https://ssrn.com/abstract=4728103
[26] P. Wynn. 1962 Acceleration techniques for iterated vector and matrix problems. *Mathematics of Computation*, **16**(79):301–322
[27] C. Brezinski. 1975 Generalisations dela transformation de shanks, dela table de pade et del'-algorithme. *CALCOLO*, **12**(4):317–360, 1975.
[28] M. Mešina. 1977 Convergence acceleration for the iterative solution of the equations x = ax + f. *Computer Methods in Applied Mechanics and Engineering*, **10**(2):165–173
[29] B. P. Pugachev. 1977 Acceleration of the convergence of iterative processes and a method of solving systems of non-linear equations. *USSR Computational Mathematics and Mathematical Physics*, **17**(5):199–207
[30] David A. Smith, William F. Ford, and Avram Sidi. 1987 Extrapolation methods for vector sequences. *SIAM Review*, **29**(2):199–233
[31] K. Jbilou and H. Sadok. 2000 Vector extrapolation methods. applications and numerical comparison. *Journal of Computational and Applied Mathematics*, **122**(1-2):149–165
[32] Avram Sidi. 2012 Review of two vector extrapolation methods of polynomial type with applications to large-scale problems. *Journal of Computational Science*, **3**(3):92–101
[33] C. Brezinski and M. Redivo-Zaglia. 2022 Extrapolation and prediction of sequences in a vector space. *Journal of Computational and Applied Mathematics*, **409**
[34] D. G. Anderson. 1965 Iterative procedures for nonlinear integral equations. *Journal of the Acm*, **12**(4):547
[35] Daniel Shanks. 1955 Non-linear transformations of divergent and slowly convergent sequences. *Journal of Mathematics and Physics*, **34**(1-4):1–42
[36] Claude Brezinski, 2018 Michela Redivo-Zaglia, and Yousef Saad. Shanks sequence transformations and anderson acceleration. *SIAM Review*, **60**(3):646–669
[37] Mohamed M. Hafez and H. K. Cheng. 1977 Convergence acceleration of relaxation solutions for transonic flow computations. *AIAA Journal*, **15**(3):329–336
[38] M. Hafez, S. Palaniswamy, G. Kuruvila, and M. Salas. 1987 Applications of Wynn's epsilon-algorithm to transonic flow- calculations. *Fluid Dynamics and Co-located Conferences* (American Institute of Aeronautics and Astronautics)
[39] Antony Jameson. 1983 Solution of the Euler equations for two dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation*, **13**(3):327–355
[40] K. Reddy and J. Jacocks. 1987 A locally implicit scheme for the Euler equations. *Fluid Dynamics and Co-located Conferences* (American Institute of Aeronautics and Astronautics)

[41] Avram Sidi and Mark Celestina. 1990 Convergence acceleration for vector sequences and applications to computational fluid dynamics. *Aerospace Sciences Meetings*. (American Institute of Aeronautics and Astronautics)
[42] S. Cheung, A. Cheer, M. Hafez, and J. Flores. 1991 Convergence acceleration of viscous and inviscid hypersonic flow calculations. *AIAA Journal*, **29**(8):1214–1222
[43] Aleksandar Jemcov and Joseph Maruszewski. 2008 Nonlinear Flow Solver Acceleration by Reduced Rank Extrapolation. *Aerospace Sciences Meetings*. (American Institute of Aeronautics and Astronautics)
[44] Reza Djeddi, Andrew Kaminsky, and Kivanc Ekici. 2017 Convergence acceleration of fluid dynamics solvers using a reduced-order model. *AIAA Journal*, **55**(9):3059–3071
[45] Kivanc Ekici, Kenneth C. Hall, Huang Huang, and Jeffrey P. Thomas. 2013 Stabilization of explicit flow solvers using a proper-orthogonal-decomposition technique. *AIAA Journal*, **51**(5):1095–1104
[46] Andrew L. Kaminsky, Reza Djeddi, and Kivanc Ekici. 2018 Convergence acceleration of continuous adjoint solvers using a reduced-order model. *International Journal for Numerical Methods in Fluids*, **86**(9):582–606
[47] W. B. Cao, Y. L. Liu, X. L. Shan, C. Q. Gao, and W. W. Zhang. 2023 A novel convergence enhancement method based on online dimension reduction optimization. *Physics of Fluids*, **35**(3)
[48] Yilang Liu, Weiwei Zhang, and Jiaqing Kou. 2019 Mode multigrid - a novel convergence acceleration method. *Aerospace Science and Technology*, **92**:605–619
[49] Wengang Chen, Weiwei Zhang, Yilang Liu, and Jiaqing Kou. 2020 Accelerating the convergence of steady adjoint equations by dynamic mode decomposition. *Structural and Multidisciplinary Optimization*, **62**(2):747–756
[50] S. L. Tan, Y. L. Liu, J. Q. Kou, and W. W. Zhang. 2021 Improved mode multigrid method for accelerating turbulence flows. *AIAA Journal*, **59**(8):3012–3024
[51] Wang, X., Liu, Y., & Zhang, W. 2023 Novel approach to improve stability and convergence of flowfield solution processes: Mode multigrid. *AIAA Journal,* **61**(8), 3496-3506
[52] Amirhossein Mirshahi and Carl F. Ollivier Gooch. 2024 Automated Convergence Acceleration of Flow Solvers Using Dynamic Mode Decomposition.
[53] D. Kochkov, J. A. Smith, A. Alieva, Q. Wang, M. P. Brenner, and S. Hoyer. 2021 Machine learning-accelerated computational fluid dynamics. *Proc Natl Acad Sci USA*, **118**(21)
[54] Cihat Duru, Hande Alemdar, and Ozgur Baran. 2021 Cnnfoil: convolutional encoder decoder modeling for pressure fields around airfoils. *Neural Computing and Applications*, **33**:1–15
[55] J. W. Hu and W. W. Zhang. 2022 Mesh-conv: Convolution operator with mesh resolution independence for flow field modeling. *Journal of Computational Physics*, **452**
[56] Jiawei Hu and Weiwei Zhang. 2023 Flow field modeling of airfoil based on convolutional neural networks from transform domain perspective. *Aerospace Science and Technology*, **136**:108198
[57] Ali Kashefi and Tapan Mukerji. 2022 Physics-informed pointnet: A deep learning solver for steady-state incompressible flows and thermal fields on multiple sets of irregular geometries. *Journal of Computational Physics*, **468**:111510
[58] Kazuto Hasegawa, Kai Fukami, Takaaki Murata, and Koji Fukagata. 2020 Cnn-lstm based reduced order modeling of two-dimensional unsteady flows around a circular cylinder at different reynolds numbers. *Fluid Dynamics Research*, **52**
[59] Francis Ogoke, Kazem Meidani, Amirreza Hashemi, and Amir Barati Farimani. 2021 Graph convolutional networks applied to unstructured flow field data. *Machine Learning: Science and Technology*, **2**
[60] McGreivy, N., & Hakim, A. 2024 Weak baselines and reporting biases lead to overoptimism in machine learning for fluid-related partial differential equations. *Nature Machine Intelligence*, **6**(10), 1256-1269
[61] Gergely Hajgató, Bálint Gyires-Tóth, and György Paál. 2019 Accelerating convergence of fluid dynamics simulations with convolutional neural networks. *Mechanical Engineering* **63**(3):230–239
[62] Yukito Tsunoda, Toshihiko Mori, Tsuguchika Tabaru, and Akira Oyama. Accuracy improvement technique of DNN for accelerating CFD simulator. *AIAA SciTech 2022 Forum*, page 1437.

[63] Obiols-Sales, O., Vishnu, A., Malaya, N., Chandramowliswharan, A. 2020 CFDNet: A deep learning-based accelerator for fluid simulations. In *Proceedings of the 34th ACM international conference on supercomputing* (pp.1-12).
[64] Krzysztof Rojek, Roman Wyrzykowski, and Pawel Gepner. Ai-accelerated CFD simulation based on OpenFoam and cpu/gpu computing. Computational Science – ICCS 2021, pages 373–385. Springer International Publishing.
[65] Xiaoyuan Zhang, Guopeng Sun, Peng Zhang, Yueqing Wang, Jian Zhang, Liang Deng, Jie Lin, and Jianqiang Chen. 2024 A residual graph convolutional network for setting initial flow field in computational fluid dynamics simulations. *Physics of Fluids*, **36**(3)
[66] Linyang Zhu, Weiwei Zhang, Jiaqing Kou, and Yilang Liu. 2019 Machine learning methods for turbulence modeling in subsonic flows around airfoils. *Physics of Fluids*, **31**(1):015105
[67] Romit Maulik, Himanshu Sharma, Saumil Patel, Bethany Lusch, and Elise Jennings. 2021 A turbulent eddy-viscosity surrogate modeling framework for reynolds-averaged navier-stokes simulations. *Computers Fluids*, **227**:104777