Incremental SVD-Based Compression for Unsteady Adjoint Fields

C Brunelli[1,2,3], B Janssens[1], K Hillewaert[2] and M Runacres[3]

[1]Department Mechanics, Royal Military Academy, Belgium
[2]Environmental and Applied Fluid Dynamics, von Karman Institute for Fluid Dynamics, Belgium
[3]Department of Thermodynamics and Fluid Mechanics, Vrije Universiteit Brussel, Belgium

Abstract

Efficient storage of high-dimensional unsteady flow solutions is critical for optimization using adjoint frameworks, particularly using Variational MultiScale methods for low-Reynolds airfoils, as they retain information linked to time-dependent phenomena like Laminar Separation Bubbles. To address the challenges posed by the high dimensionality and time-dependent structure of unsteady adjoint fields, a compression approach leveraging Incremental Singular Value Decomposition combined with a modified Gram-Schmidt re-orthogonalization strategy is proposed. The performance of the method is evaluated on the unsteady flow on the NACA0012 at angle of attack 2.5° and Reynolds number 1000. Results demonstrate a substantial reduction in storage requirements, enabling scalable handling of flow field data without compromising the fidelity of optimization outcomes, avoiding performing an SVD at each time step. This work underscores the potential of advanced compression techniques to enhance the feasibility of unsteady-adjoint-based design optimization.

Keywords: AI, Fluid Mechanics, Compression, Incremental SVD, Variational MultiScale

1. Introduction

The efficient storage of unsteady flow fields represents a significant challenge in adjoint-based optimization frameworks in aerodynamic shape optimization. Resolving and accurately storing the flow fields at each time step is crucial for capturing the unsteady flow dynamics. Unsteady adjoint methods, critical for gradient-based optimizations, require storing flow information at each timestep, creating a bottleneck in memory and computational efficiency. Traditional approaches, such as full storage or checkpointing [1], [2], often fail to scale effectively for complex unsteady problems, highlighting the need for innovative solutions. This work employs a Large Eddy Simulation (LES) framework based on the Variational MultiScale (VMS) method to solve both the primal and adjoint flow equations while exploring the use of Incremental Singular Value Decomposition (ISVD) for compressing unsteady flow in adjoint optimization. The reduced-order storage methodology achieves real-time compression with minimal storage overhead. Unlike standard SVD techniques, the incremental approach updates singular value decompositions dynamically as new data become available, avoiding the prohibitive cost of storing large snapshot vectors, one for each time-step. Furthermore, integrating a modified Gram-Schmidt re-orthogonalization strategy ensures the numerical stability of the evolving basis vectors, mitigating the accumulation of round-off errors—a critical factor for long-term unsteady simulations. The proposed approach is validated on a 2D airfoil test case, demonstrating that ISVD achieves substantial compression ratios while preserving the accuracy of optimization outcomes. By efficiently capturing dominant spatiotemporal modes, the method aligns with the computational requirements of high-fidelity adjoint solvers. This study enhances the feasibility of using compressed representations in adjoint-based optimization frameworks, enabling scalable and memory-efficient aerodynamic design techniques.

2. Basic Definitions

This work begins by presenting key definitions and concepts essential for the analysis. For notational consistency, one-based indexing square brackets notation is adopted. A vector $\boldsymbol{u} \in \mathbb{R}^n$ is represented such that $\boldsymbol{u}[1:r]$ denotes the first r components of $\boldsymbol{u}$, where $r \leq n$. Similarly, a matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is represented such that $\boldsymbol{A}[p:q, r:s]$ represents a sub-matrix, with rows from p to q and columns from r to s. A sparse, symmetric, positive definite matrix $\boldsymbol{W} \in \mathbb{R}^{n \times n}$ is introduced to define the inner product, [3]. In this study, it is going to be the identity matrix. The weighted norm of a vector is also used $||\boldsymbol{a}||_W = (\boldsymbol{a}^\top \boldsymbol{W} \boldsymbol{a})^{1/2}$.

2.1 SVD

The SVD decomposition for a matrix is defined as $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ as $\boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top = \boldsymbol{A}$, where $\boldsymbol{U} \in \mathbb{R}^{n \times m}$, $\boldsymbol{S} \in \mathbb{R}^{m \times m}$, $\boldsymbol{V} \in \mathbb{R}^{m \times m}$, where for CFD applications $m << n$. $\boldsymbol{S}$ is a diagonal matrix, storing the singular values of the decomposition in descending order $S_{ii} \geq S_{jj} \quad \forall \quad 1 \geq j > 1$. The truncated SVD rank-k SVD decomposition is defined as $\tilde{\boldsymbol{U}}\tilde{\boldsymbol{S}}\tilde{\boldsymbol{V}}^\top = \tilde{\boldsymbol{A}}$, where $\tilde{\boldsymbol{U}} \in \mathbb{R}^{n \times k}, \tilde{\boldsymbol{S}} \in \mathbb{R}^{k \times k}, \tilde{\boldsymbol{V}} \in \mathbb{R}^{m \times k}$ and the new matrices are defined as $\tilde{\boldsymbol{U}} = \boldsymbol{U}[:, 1:k]$, $\tilde{S} = \boldsymbol{S}[1:k, 1:k]$, $\tilde{\boldsymbol{V}} = \boldsymbol{V}[1:k, :]$, with $k < \min(m, n)$

2.2 ISVD

The ISVD algorithm was first introduced in [4]. Consider a matrix $\overset{A}{\boldsymbol{A}} \in \mathbb{R}^{n \times m}$ with an existing SVD decomposition given by $\boldsymbol{A} = \boldsymbol{U}_0 \boldsymbol{S}_0 \boldsymbol{V}_0^\top$. When a new column vector $\boldsymbol{u}_1$ is introduced, the SVD can be efficiently updated using Algorithm 1. The latent representation of the new column is computed as $\boldsymbol{w} = \boldsymbol{U}_0^\top \boldsymbol{u}_1$, and the auto-encoding reconstruction error vector is given by $\boldsymbol{r}_t = \boldsymbol{u}_1 - \boldsymbol{U}_0 \boldsymbol{w}$. A key advantage of ISVD is that it eliminates the need to store the entire matrix $\boldsymbol{A}$. Instead, only the SVD decomposition is required, which can be a k-rank approximation. This significantly reduces storage requirements while maintaining the ability to perform efficient updates.

Algorithm 1 Incremental SVD

Input: $\boldsymbol{U}_0 \in \mathbb{R}^{n \times k}$, $\boldsymbol{S}_0 \in \mathbb{R}^{k \times k}$, $\boldsymbol{V}_0 \in \mathbb{R}^{m \times k}$, $\boldsymbol{r}_t \in \mathbb{R}^n$ reconstruction error, $\boldsymbol{w} \in \mathbb{R}^k$ latent representation of the new snapshot

$$\boldsymbol{MAT} \leftarrow \begin{matrix} \boldsymbol{S}_0 & \boldsymbol{w} \\ \boldsymbol{0} & ||\boldsymbol{r}_t||_W \end{matrix}$$

$$\hat{\boldsymbol{U}}, \boldsymbol{S}_1, \hat{\boldsymbol{V}} \leftarrow \text{svd}(\boldsymbol{MAT})$$

$$\boldsymbol{U}_1 \leftarrow \boldsymbol{U}_0 \quad \frac{\boldsymbol{r}_t}{||\boldsymbol{r}_t||_W} \quad \hat{\boldsymbol{\iota}}$$

$$\boldsymbol{V}_1 \leftarrow \begin{matrix} \boldsymbol{V}_0 & \boldsymbol{0} \\ \boldsymbol{0} & 1 \end{matrix} \hat{\boldsymbol{V}}$$

return $\boldsymbol{U}_1, \boldsymbol{S}_1, \boldsymbol{V}_1$ ▷ Updating SVD matrices

2.3 Re-Orthogonalization

In practical applications, small numerical errors can lead to a loss of orthogonality, [5]. To address this, re-orthogonalization is performed when the weighted inner product between the first and last left singular vectors exceeds a specified tolerance. This is achieved by applying a modified Gram-Schmidt procedure with re-orthogonalization to the columns of $\boldsymbol{U}$ ensuring that the resulting matrix has columns that are orthonormal, algorithm 2.

3. ISVD for Unsteady Field Compression

Let $n \in \mathbb{N}$ denote the number of degrees of freedom, representing the large-dimensional space to be compressed. In the context of CFD, n is proportional to the number of computational nodes. For a 2D simulation, each node contributes three degrees of freedom (two velocity components and pressure), while in 3D, each node contributes four degrees of freedom. Since velocity and pressure are typically

Algorithm 2 Modified W-weighted Gram–Schmidt with reorthogonalization

Input: $\boldsymbol{U} \in \mathsf{R}^{n\times k}$ matrix to re-orthogonalize by columns, $\boldsymbol{R} \in \mathsf{R}^{k\times k}$ zero matrix
for $j \leftarrow 1$ **to** n **do**
  for $i \leftarrow 1$ **to** $j-1$ **do**
    $E \leftarrow \boldsymbol{U}[:, i]^{\top} W \boldsymbol{U}[:, j]$ ▷ Projection coefficient
    $\boldsymbol{U}[:, j] \leftarrow \boldsymbol{U}[:, j] - E \cdot \boldsymbol{U}[:, i]$ ▷ Orthogonalize
    $U[i, j] \leftarrow R[i, j] + E$
  end for
  $U[j, j] \leftarrow \|\boldsymbol{U}[:, j]\|_W$
  $\boldsymbol{U}[:, j] \leftarrow \dfrac{\boldsymbol{U}[:, j]}{R[j, j]}$ ▷ Normalize column
end for
return $\boldsymbol{U}$

stored in separate SVD decompositions, the compression is applied independently to each field. Let $n_t \in \mathsf{N}$ represent the total number of time steps to be saved.

The parameter $k \in \mathsf{N}$ indicates the number of independent basis vectors used in the compression. Starting with $k = 1$, it increases dynamically as additional eigenfields are appended as columns to the matrix $\boldsymbol{U}_0$. Typically, $k \ll n_t \ll n$, where the effectiveness of compression improves as the ratio k/n_t decreases. When $k = n_t$, no compression occurs, and it is equivalent to storing all snapshots at full resolution.

Let $\boldsymbol{LS} \in \{\boldsymbol{l}_i \in \mathsf{R}^k \mid i = 1, \ldots, n_{ti}\}$ be the Latent Space (LS). It contains n_{ti} vectors of dimension k, corresponding to the snapshots acquired up to a specific point, where $n_{ti} \leq n_t$. If a maximum number of basis vectors $k_{\max}$ is prescribed, $\boldsymbol{LS}$ can be pre-allocated. Otherwise, it is updated dynamically during the process (see algorithm 4).

The procedure begins by acquiring the first snapshot, $\mathbf{u}_1 \in \mathsf{R}^n$, and initializing the ISVD using algorithm 3. This results in the initial SVD decomposition $\boldsymbol{U}_0, \boldsymbol{S}_0, \boldsymbol{V}_0$ and the first latent space vector $\boldsymbol{LS}_0[1] = \boldsymbol{U}_0^{\top}\mathbf{u}_1$, which encodes $\mathbf{u}_1$ using $\boldsymbol{U}_0$ as the basis representation.

Algorithm 3 Initialize incremental SVD

Input: $\boldsymbol{u}_1 \in \mathsf{R}^n$ first snapshot
$\boldsymbol{S}_0 \leftarrow ||\boldsymbol{u}_1||_W$
$\boldsymbol{U}_0 \leftarrow \dfrac{\boldsymbol{u}_1}{\boldsymbol{S}_0}$
$\boldsymbol{V}_0 \leftarrow \begin{bmatrix} 1 \end{bmatrix}$
return $\boldsymbol{U}_0, \boldsymbol{S}_0, \boldsymbol{V}_0$

After the initialization, it is evident that $k = 1$. For any new snapshot $\boldsymbol{u}_t$ the algorithm 4 is executed, where $2 \leq t \leq n_t$. Two tolerances are introduced: one for controlling the update of the basis space $\boldsymbol{U}_0$, an another for the re-orthogonalization. The first one, tol_v represents the velocity error tolorance - *default value:* 1×10^{-3}. It is the absolute maximum reconstruction error allowed. The second tolerance, $\mathrm{tol}_{\mathrm{orth}}$ non-orthogonalization tolerance - *default value:* 1×10^{-8}. It represents the threshold scalar value beyond which re-orthogonalization is performed.

Algorithm 4 Iterative Update SVD

Input: $\boldsymbol{U_0} \in \mathbb{R}^{n\times k}$, $\boldsymbol{S_0} \in \mathbb{R}^{k\times k}$, $\boldsymbol{V_0} \in \mathbb{R}^{m\times k}$, $\boldsymbol{u_t} \in \mathbb{R}^n$ new snapshot, $\boldsymbol{LS_0} \in \mathbb{R}^{n_t}\{\mathbb{R}^k\}$, $\text{tol}_v \in \mathbb{R}^+$, $\text{tol}_{\text{orth}} \in \mathbb{R}^+$

$k \leftarrow \text{size}(\boldsymbol{U_0}, 2)$ ▷ number of basis used for the representation, equal to the number of columns of $\boldsymbol{U_0}$

$w \leftarrow \boldsymbol{U_0}^\top \boldsymbol{W} \cdot \boldsymbol{u_t}$ ▷ encode $\boldsymbol{u_t}$ in the latent space

$\boldsymbol{p_t} \leftarrow \boldsymbol{U_0} \cdot w$ ▷ autoencoding of $\boldsymbol{u_t}$ in the current basis space

$\boldsymbol{r_t} \leftarrow \boldsymbol{u_t} - \boldsymbol{p_t}$ ▷ reconstruction error

if $\text{maximum}(|\boldsymbol{r_t}|) > \text{tol}_v$ **then**

  $\boldsymbol{U_1}, \boldsymbol{S_1}, \boldsymbol{V_1}$ update using algorithm 1

  if $|\boldsymbol{U_1}[:, 1]^\top \boldsymbol{W} \boldsymbol{U_1}[:, \text{end}]| > \text{tol}_{\text{orth}}$ **then**

    re-orthogonalize U_1 using algorithm 2

  end if

  $\boldsymbol{RLS} \leftarrow \boldsymbol{U_1}^\top \boldsymbol{W} \boldsymbol{U_0}$ $\quad \boldsymbol{RLS} \in \mathbb{R}^{k+1\times k}$

  ▷ Decode the latent space with the old basis space, and re-encode it with the new one

  for $i \leftarrow 1$ **to** k **do**

    $\boldsymbol{LS_1}[i] \leftarrow \boldsymbol{RLS} \cdot \boldsymbol{LS_0}[i]$

  end for

  ▷ Update Space and Latent Space

  $\boldsymbol{U_0} \leftarrow \boldsymbol{U_1}, \ \boldsymbol{S_0} \leftarrow \boldsymbol{S_1}, \ \boldsymbol{V_0} \leftarrow \boldsymbol{V_1}, \boldsymbol{LS_0} \leftarrow \boldsymbol{LS_1}$

end if

$\boldsymbol{LS_0}[k+1] \leftarrow \boldsymbol{U_0}^\top \boldsymbol{W} \boldsymbol{u_t}$ ▷ Encode in the LS the new vector

4. Numerical Results

In this example, the NACA0012 airfoil is simulated at an angle of attack of 2.5° with a Reynolds number of 1000, in 2D. The velocity vector at each time step consists of 88574 components. The simulation uses the VMS as LES, on a structured mesh composed of first-order quadrangular elements, similarly as [6]. The inflow velocity is constant at 1 m/s in the x-direction, and the first 10 s of the simulation are excluded to allow the flow to develop. Throughout the simulation, a time step of $\Delta t = 0.05$ s is employed. The initial SVD decomposition is constructed with the first snapshot, and subsequent snapshots are analyzed and stored sequentially using the ISVD algorithm 4. Using the default tolerance, the algorithm retains the leading 10 singular values, providing an efficient low-rank approximation of the data. Comparisons are conducted between the incremental singular values obtained during the simulation and those computed from the full data set, demonstrating the accuracy and storage efficiency of the ISVD approach.

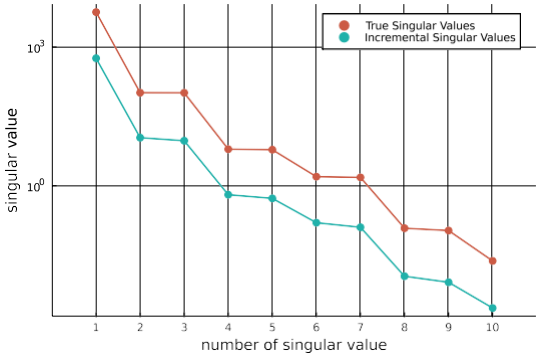

Figure 1: Comparison of singular values of ISVD and SVD

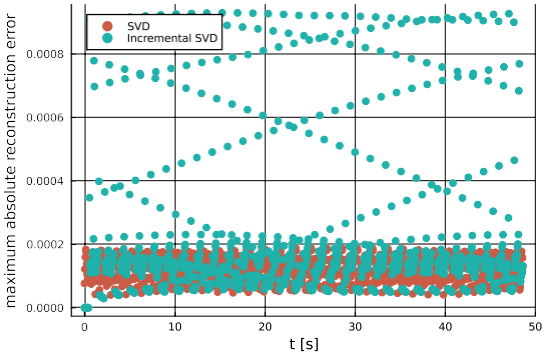

Figure 2: Maximum absolute reconstruction error for ISVD and SVD

The singular value distribution, figure 1, shows important insights into its underlying dynamics. In this case, the presence of a dominant first singular value, followed by pairs of nearly identical singular values suggests the presence of periodicity in the flow. Such coupling in singular values is a characteristic feature of a periodic system. The dominance of the first singular value indicates a primary structure or energy-dominant mode, while the coupled singular values reflect pairs of oscillatory modes that are shifted in phase but with the same energetic content. This pattern aligns with the expected behaviour of flows with coherent structures recurring at regular intervals. The analysis reveals that the flow exhibits a periodicity of $\approx$ 2 Hz, figure 6. For this specific case, the most efficient strategy would be to preserve all snapshots at full resolution over a single period, which, given a time step of 0.05 s, results in 10 snapshots. Interestingly, this aligns with the number of singular values automatically identified by the algorithm.

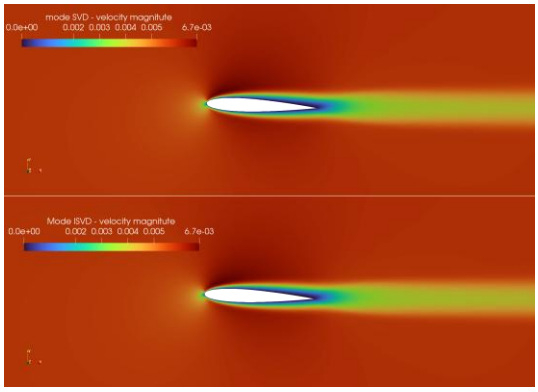

Figure 3: 1st velocity mode

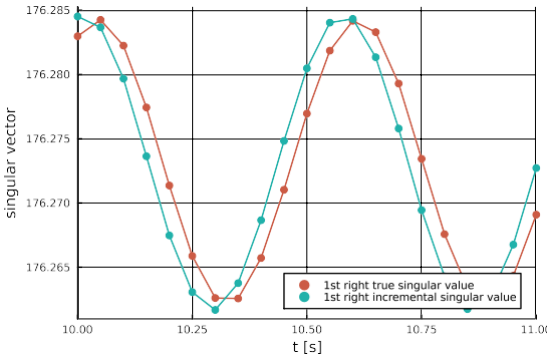

Figure 4: 1st right singular vectors

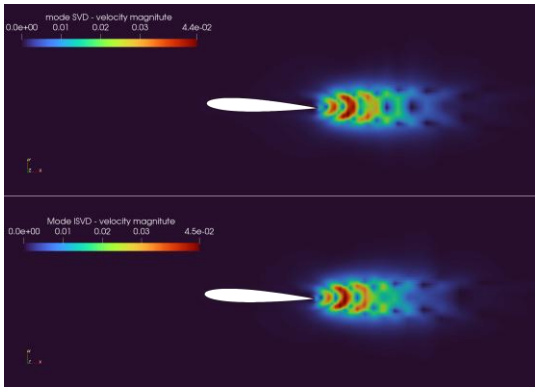

Figure 5: 2nd velocity mode

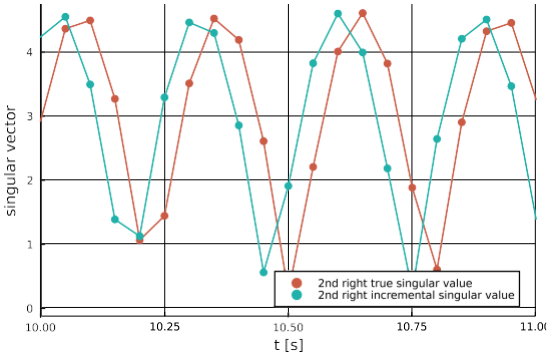

Figure 6: 2nd right singular vectors

In general, relying on sequential snapshots for storage is not advisable, because they are highly correlated. Furthermore, periodicity and frequency characteristics are often unknown in advance, especially in unsteady or turbulent flows. Thus, adopting the proposed strategy is a more robust approach to ensure efficient and accurate representation of diverse flow behaviours.

5. Conclusions

An approach for efficient storage of high-dimensional unsteady flow solutions has been presented to overcome the storage bottleneck in adjoint frameworks. The proposed method leverages Incremental Singular Value Decomposition and a modified Gram-Schmidt re-orthogonalization strategy to compress the adjoint fields without compromising their fidelity. The performance of the method has been evaluated on the unsteady flow on the NACA0012 airfoil at an angle of attack of 2.5° and Reynolds number of 1000, with promising results demonstrating a substantial reduction in storage requirements without sacrificing the accuracy of optimization outcomes. This work highlights the potential of advanced compression

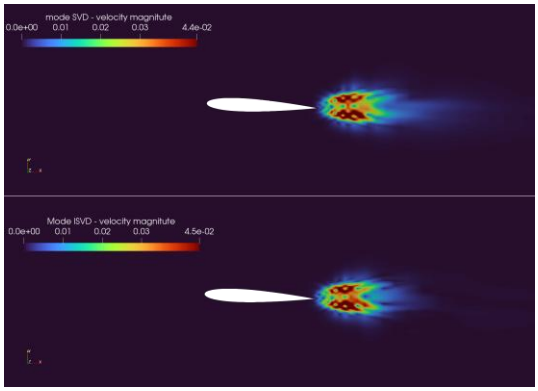

Figure 7: 4^{th} velocity mode

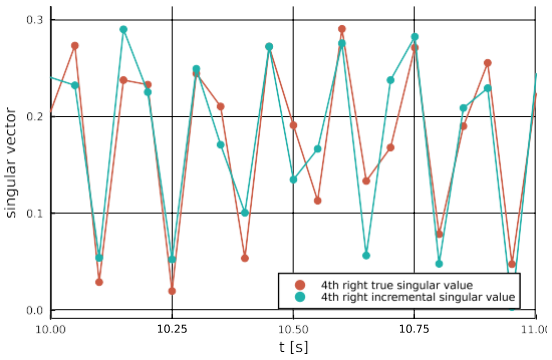

Figure 8: 4^{th} right singular vectors

techniques to enhance the feasibility of unsteady-adjoint-based design optimization for complex flow problems. Furthermore, the analysis reveals important insights into the underlying dynamics of the flow, including the presence of periodicity and frequency characteristics. This approach ensures a robust representation of diverse flow behaviours, particularly in unsteady or turbulent flows where frequency and periodicity are often unknown in advance. In summary, the proposed method offers a practical solution for efficient storage of high-dimensional unsteady flow solutions, opening up new possibilities for optimization and design exploration in fluid dynamics.

References

[1] Q. Wang, P. Moin, and G. Iaccarino, "Minimal repetition dynamic checkpointing algorithm for unsteady adjoint calculation," *SIAM Journal on Scientific Computing*, vol. 31, pp. 2549–2567, 4 2009, ISSN: 10648275. DOI: 10.1137/080727890.

[2] A. Griewank and A. Walther, "Algorithm 799: Revolve: An implementation of checkpointing for the reverse or adjoint mode of computational differentiation," *ACM Transactions on Mathematical Software*, vol. 26, pp. 19–45, 1 Mar. 2000, ISSN: 0098-3500. DOI: 10.1145/347837.347846. [Online]. Available: https://dl.acm.org/doi/10.1145/347837.347846.

[3] H. Fareed, J. R. Singler, Y. Zhang, and J. Shen, "Incremental proper orthogonal decomposition for pde simulation data," *Computers & Mathematics with Applications*, vol. 75, pp. 1942–1960, 6 Mar. 2018, ISSN: 08981221. DOI: 10.1016/j.camwa.2017.09.012. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0898122117305606.

[4] M. Brand, "Fast low-rank modifications of the thin singular value decomposition," *Linear Algebra and its Applications*, vol. 415, pp. 20–30, 1 May 2006, ISSN: 00243795. DOI: 10.1016/j.laa.2005.07.021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0024379505003812.

[5] G. M. Oxberry, T. Kostova-Vassilevska, W. Arrighi, and K. Chand, "Limited-memory adaptive snapshot selection for proper orthogonal decomposition," *International Journal for Numerical Methods in Engineering*, vol. 109, pp. 198–217, 2 Jan. 2017, ISSN: 0029-5981. DOI: 10.1002/nme.5283. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/nme.5283.

[6] C. Brunelli, M. Avirovic, B. Janssens, B. G. Marinus, G. May, and M. Runacres, "Prediction of a Laminar Separation Bubble on Low-Reynolds Airfoils Using Variational Multiscale Method," in *AIAA AVIATION FORUM AND ASCEND 2024*, Las Vegas, Nevada: American Institute of Aeronautics and Astronautics, Jul. 29, 2024, ISBN: 978-1-62410-716-0. DOI: 10.2514/6.2024-4261. [Online]. Available: https://arc.aiaa.org/doi/10.2514/6.2024-4261.