GPPS | JOURNAL

JOURNAL OF THE GLOBAL POWER AND PROPULSION SOCIETY
journal.gpps.global

A high-performance code for the use of spectral POD on the analysis of turbomachinery high fidelity simulations

Original article

Article history:
Submission date: 22 November 2024
Acceptance date: 6 June 2025
Publication date: 9 December 2025

Check for updates

Daniele Biassoni[1], Matteo Russo[1], Davide Lengani[1,*]

[1]*Department of Mechanical, Energy, Management and Transport Engineering (DIME), University of Genoa, Via Montallegro 1, Genoa i-16145, Italy*

*Correspondence:
DL: davide.lengani@edu.unige.it

Peer review:
Single blind

Keywords:
Turbomachinery; spectral POD; HPDA; large Eddy simulations

Citation:
Biassoni D., Russo M., and Lengani D. (2025). A high-performance code for the use of spectral POD on the analysis of turbomachinery high fidelity simulations. Journal of the Global Power and Propulsion Society. 9: 330–340.
https://doi.org/10.33737/jgpps/205987

Abstract

This paper presents a High-Performance Data Analytics procedure for computing Spectral Proper Orthogonal Decomposition (SPOD) from high-fidelity simulations. The code is implemented in Python, leveraging the Dask library for ease of programming and efficient parallel computing. The method is applied to Large Eddy Simulation data of a low-pressure turbine blade, showing the ability to extract the fundamental dynamics and their link to loss generation mechanisms. The mathematical framework behind the SPOD implementation in Python and the decomposition of the total pressure transport equations are discussed. Scalability tests using an exemplary dataset demonstrate a very high efficiency of the implementation. Results from SPOD are compared with a classical decomposition framework such as proper orthogonal decomposition and the paper discusses the limitations and advantages of SPOD. The procedure is open-access and available in an online repository, promoting accessibility and collaboration within the scientific community.

Introduction

The use of high-fidelity simulations to analyze turbomachinery flow fields is nowadays a common practice, even at industrial level. Hi-fi simulations may provide a wealth of data for detailed investigation of the flow physics and unveil the dynamics that generate losses. As discussed in the review paper of Sandberg and Michelassi (2022), high-fidelity simulations may be used to extract local information to pinpoint local loss causes. To this end, advanced post-processing methods have been introduced in the last years.

Wheeler et al. (2016) were among the first to show the integral entropy rise in a high pressure turbine blade solved with Direct Numerical Simulation (DNS). They used the entropy transport equation by directly computing the contribution to viscous dissipation by the time-mean strain field and by the turbulence field. Similarly, Miki and Ameri (2022) showed the instantaneous dissipation obtained from Large Eddy Simulation (LES) of a transonic turbine blade. Further detailed analysis of the transport equations of entropy, total pressure or mechanical work potential have been provided by several research groups (e.g., Leggett et al., 2020; Przytarski and Wheeler, 2020; Zhao and Sandberg, 2020; Przytarski et al., 2024b). In these works the losses are evaluated by means of an integral approach within the control volume rather than the classical inlet to outlet loss estimation. With these methods, it is easy to identify where losses are generated. However, it is rather difficult to decouple the dynamical features that lead to the loss generation. In order to overcome this limit, Lengani et al. (2019) applied the Proper Orthogonal Decomposition to LES data of a low pressure turbine blade to identify the most energetic dynamical features. They showed that total pressure transport equation in the Reynolds average sense may be re-written to highlight the contribution of each mode (i.e., dynamical feature). This method has been recently applied to several LPT cases to discern the loss causes (e.g., Dellacasagrande et al., 2023; Russo et al., 2024).

The amount of data that are generated nowadays by LES or DNS easily exceeds tens of terabytes. Therefore, the use of high-performance computing services is becoming necessary also for the post-processing of such simulations. An open source high performance data analytic (HPDA) code, developed within the Euro-HPC ACROSS project, has been described in Biassoni et al. (2024). This open code performs POD on large datasets (tens of TB) and applies the loss breakdown described in Lengani et al. (2019). The present paper provides an extension of these previous work by computing the losses in the Spectral POD (Towne et al., 2018) modal decomposition framework. Among the different open versions of the SPOD (see for example He et al., 2021; Rogowski et al., 2024), the present one leverages on the distributed computation of the Dask framework of Python by providing an ease of programming and good efficiency on multi-node high performance computers.

Methodology

Fundamental equations

Losses of turbomachinery bladings can be evaluated by integrating the substantial derivative of entropy over the computational domain, as one of the most used metrics (e.g., Wheeler et al., 2016; Russo et al., 2024). In most cases, by considering an adiabatic transformation, this operation is equivalent to the integral volume of the substantial derivative of the total pressure with the opposite sign, according to the Gibbs equation (see for example Denton, 1993). Following a Reynolds decomposition approach, the time average of the substantial derivative of the total pressure can be written, adopting Einstein notation, as Przytarski et al. (2024b):

$$\overline{\frac{Dp_t}{Dt}} = \overline{u}_i \frac{\partial \overline{p}}{\partial x_i} + \frac{D}{Dt}\left(\frac{1}{2}\rho \overline{u}_i \overline{u}_i\right) + \frac{D}{Dt}\left(\frac{1}{2}\rho \overline{u'_i u'_i}\right) + \frac{\partial}{\partial x_j}\left[\overline{u}_i\left(\rho \overline{u'_i u'_j}\right)\right] + \frac{1}{2}\rho \frac{\partial \overline{u'_j(u'_i u'_i)}}{\partial x_i} + \frac{\overline{\partial u'_i p'}}{\partial x_i} \tag{1}$$

where the equation is written just for simplicity in the incompressible formulation but it may be easily extended to the compressible case by using Favre averaging operation. The overbar indicates the time average operation and the u_i components indicate the mean velocity, while the term u'_i indicates the fluctuating velocity. The first three terms on the right-hand side correspond to the substantial derivative of the time mean total pressure, whereas the last three terms may be considered in most cases negligible and exceptions to this are discussed in detail in Przytarski et al. (2024a,b). By further neglecting the substantial derivative of the turbulent kinetic energy $(\overline{u'_i u'_i})$, the total pressure variation can be approximated as:

$$\overline{u_i}\frac{\partial \overline{p_t}}{\partial x_i} \approx \overline{u}_i \frac{\partial \overline{p}}{\partial x_i} + \frac{D}{Dt}\left(\frac{1}{2}\rho \overline{u}_i \overline{u}_i\right) = \underbrace{-\rho\nu\left(\frac{\partial \overline{u_i}}{\partial x_j}\right)^2}_{\phi_M} - \underbrace{\left(-\rho \overline{u'_i u'_j}\frac{\partial \overline{u_i}}{\partial x_j}\right)}_{P_{TKE}} \underbrace{+\rho\nu \frac{\partial^2 K}{\partial x_j^2} - \frac{\partial\left(\overline{u_i}\overline{u'_i u'_j}\right)}{\partial x_j}}_{\text{Higher order terms}} \tag{2}$$

where K is the mean flow kinetic energy. The viscous dissipation due to the mean strain (ϕ_M) depends just on the time mean flow and the P_{TKE} depends on instantaneous fluctuating velocity. Namely, each instantaneous flow field realization (a snapshot) contributes to the overall time mean value. In particular, in Equation 2, the terms responsible for the largest contribution of loss generation are ϕ_M and P_{TKE}, as also discussed in Zhao and Sandberg (2020). The overall variation of total pressure (i.e., entropy rise) can be then calculated as the volume integral of ϕ_M and P_{TKE}:

$$\Delta p_t \propto \int_V \left(\phi_M + P_{TKE}\right)\, dV \tag{3}$$

As discussed in Lengani et al. (2019), it is possible to decompose the contribution to the Reynolds stress tensor $(\overline{u'_i u'_j})$ by projecting the instantaneous fluctuating velocity on a complete orthogonal basis. Namely, an orthonormal basis X may be used to decompose the fluctuating velocity field as $u_i'(x) = X\phi_{ui}(x)$. The modes ϕ_{ui} are carrying the square root of the energy and provide the spatial representation of the velocity u_i'. Considering that the orthonormal basis is in common between the different velocity components, the Reynolds stress tensor may be written as (see also Perrin et al., 2007):

$$\begin{aligned}\overline{u_i'(x)u_j'(x)} &= \frac{1}{N_t}u_j^T(x)u_i(x) = \frac{1}{N_t}\left(X\varphi_{uj}(x)\right)^H X\varphi_{ui} = \frac{1}{N_t}\varphi_{uj}^H(x)X^H X\varphi_{ui}(x) = \frac{1}{N_t}\varphi_{uj}^H(x)\varphi_{ui}(x)\\ &= \frac{1}{N_t}\sum_{k=1}^{N_t} conj(\varphi_{ui}^{(k)}(x))\varphi_{uj}^{(k)}(x)\end{aligned} \tag{4}$$

where the time mean operation is initially expressed as the scalar product between the velocity data array, N_t is the number of instantaneous flow field realizations, ^{H} indicates the Hermitian transpose and the product $X^H X$ is the identity matrix by definition of the orthonormal basis. The Reynolds stress may be then computed as a summation of the spatial modes (note that the complex conjugate *conj* applies when the modes are complex numbers). Finally, it is possible to compute the turbulent kinetic energy production as Lengani et al. (2019):

$$P_{TKE} = -\rho\sum_k conj(\varphi_i^{(k)})\varphi_j^{(k)}\frac{\partial\overline{u_i}}{\partial x_j} \tag{5}$$

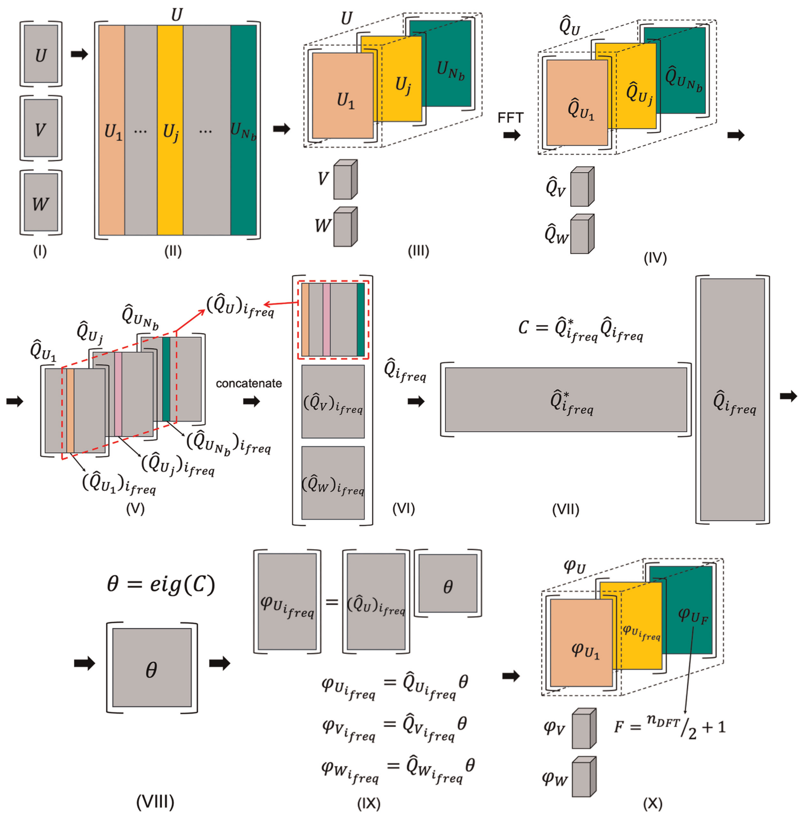
In this way, each mode k contributes to a quota of the turbulent kinetic energy production. By finding an optimal base X, it is possible to decouple the effect of the different dynamical features (represented by the modes $\varphi^{(k)}$) to the total pressure loss as computed from Equation 3. The application of the POD modal decomposition framework has been developed in several previous papers (e.g., Lengani et al., 2019; Dellacasagrande et al., 2023; Russo et al., 2024) and it is here employed as reference for comparison with the SPOD results. Note that, for this analysis to hold true, it is essential that the orthonormal basis X is common to the velocity components. Consequently, it is necessary to build the data matrix from at least the three velocity components. This operation may result in significant memory requirements for the computation, (e.g., for this limitation Fiore et al., 2023 computed SPOD from a single velocity component by adopting a serial code). The next section describes in detail the new SPOD-HPDA code, that leverages distributed memory among several compute nodes, and presents its scalability results. Two different algorithms for the computation of SPOD modes are used and described.

Python SPOD procedure

This new procedure has been developed in Python by leveraging the Dask (https://www.dask.org/) library for distributed computation. The advantage of Dask is that it is no longer necessary for the user to manually partition domain data to provide parallel computation, since data chunks are automatically managed by Dask. Another advantage of Dask is that you can start from a serial prototype based on NumPy and Pandas libraries and convert it very easily to a fairly large-scale distributed code. For a more complete description of how Dask works, see Viviani et al. (2022).

The procedure is visualized in Figure 1a and described in the algorithm depicted in Figure 1b. Please note that our beseline SPOD implementation follows the theoretical formulation of Towne et al. (2018). For the practical “serial” implementation the works of Schmidt and Colonius (2020), He et al. (2021) and Fiore et al. (2023) may be taken as a reference. The common practice of overlapping data in time, typically used in such implementations, is not employed here. This decision ensures a complete orthonormal decomposition in both frequency and SPOD modes. A parallel implementation of SPOD, which overcomes limitations of memory usage and running time, is presented in Rogowski et al. (2024) by utilizing MPI communication between computational nodes. In the present paper, distributed computation is directly managed by Dask, and the variables in the SPOD Kernel are kept separated as much as possible. In detail, the relevant operations managed by Dask are highlighted in the algorithm of Figure 1b with the suffix “da,” otherwise the explicit use of numpy library is indicated with “np.” The output files are read by Dask (see Biassoni et al., 2024) and stored in three data matrices. The three fluctuating velocity components (U_i in the algorithm) are stored in the three data matrices U, V, W (step I of Figure 1a). Each column vector of the matrix retains the spatial information. The rows got dimension N_t (i.e., the number of instantaneous realizations or snapshots), while the columns got dimension M_s, the total number of computational mesh points. The first step of SPOD is fragmenting the time sequence in a

(a)



(b)

```
Algorithm 1 HPDA algorithm
Inputs: Output files from CFD
% Read input files with Dask
% to obtain data matrices
% (see Biassoni et al. 2024)
$U_i.shape = [M_s, N_t]$
% Compute and remove the mean
$u_i\_m = da.mean(u_i)$
$U_i = U_i - u_i\_m$
% Compute SPOD modes
$U_i = U_i.reshape(M_s, N_b, n_{DFT})$
$\hat{Q}_{Ui}$ =da.fft($U_i$,axis= 2)
For $i_{freq}$ in range$(0, n_{DFT}/2+1)$
$(\hat{Q}_{Ui})_{ifreq}$ =da.squeeze($\hat{Q}_{Ui}(:,:,i_{freq})$)
$\hat{Q}_{ifreq}$=da.concatenate($(\hat{Q}_{Ui})_{ifreq}$)
$C_{ifreq}$ =da.dot($\hat{Q}^H_{ifreq}, \hat{Q}_{ifreq}$)
$\theta$ =np.eigh($C_{ifreq}$)
$\varphi_{Ui}$ =da.dot($(\hat{Q}_{Ui})_{ifreq}, \theta$)
End
% SPOD modes are normalized
% End of SPOD function
% Compute derivative of mean flow
$du_i\_dx_j = derivative(u_i, x_j)$
% Compute $P_{TKE}$ per mode
$P_{TKE} = conj(\varphi_{Ui}) * \varphi_{Uj} * du_i\_m\_dx_j)$
% Intergral of $\phi_M$ and $P_{TKE}$
$\phi_M = \sum((du_i\_m/dx_j)^2 * vol)$
$P_{TKE} = \sum(P_{TKE} * vol)$
Output: $\phi_M$, $P_{TKE}$
```

Figure 1. Spectral-POD procedure: SPOD computational sequence (a); complete HPDA algorithm (b).

discrete number of blocks N_b (step II), which is obtained by means of an array reshape operation (step III). Dask then manages the discrete Fourier transform (DFT) using the FFT algorithm along the temporal direction of size n_{DFT} (with $n_{DFT} * N_b = N_t$) to obtain the matrices $\hat{Q}_{Ui}$ for each velocity component (step IV). Each 3D array $\hat{Q}_{Ui}$ contains the frequency information along its last dimension (shape 2 according to Python indexing, which starts from 0). In the baseline implementation of SPOD, the data for each frequency are then collected within a *for* loop in order to store the matrices $(\hat{Q}_{Ui})_{ifreq}$ at the frequency loop index *ifreq* (step V), this operation is done for all variables at hand (step VI). As observed by Rogowski et al. (2024), the number of frequencies of interest is $n_{DFT}/2 + 1$, as specified in the algorithm, and this is because of the symmetry properties of the DFT. The matrices $(\hat{Q}_{Ui})_{ifreq}$ are then concatenated to form the $\hat{Q}_{ifreq}$ matrix (from step VI to VII) in order to compute the cross-correlation matrix C_{ifreq} within the *for* loop (step VII) and its eigenvectors θ_{ifreq} (step VIII). Note that, at present, Dask does not provide the equivalent of the eigenvector calculation of the numpy library and therefore the computation of step VIII is managed by numpy. Nevertheless, the C_{ifreq} matrix is very small with dimension $[N_b, N_b]$. The SPOD modes are computed separately for each variable as $(\varphi_{Ui})_{ifreq} = (\hat{Q}_{Ui})_{ifreq}\theta$ (step IX) and stored in a 3D array (step X) with the same shape of matrices in step III. Finally, to ease the computation of the P_{TKE}, the SPOD modes are reshaped to a matrix with dimension $[M_s, N_t]$. Namely, SPOD modes are sorted to have first the ordered modes at each frequency index, as it will be better discussed in the result section.

The SPOD computation, detailed in steps V to IX of Figure 1a, has been optimized by eliminating the use of *for* loops. The improved algorithm, outlined in Figure 2, takes advantage of the execution order in the eigenvector solver from the numpy library. The input data are rearranged using stacking and transposing operations, resulting in a three-dimensional array, C, with dimensions $[n_{DFT}/2 + 1, N_b, N_b]$. The first dimension corresponds to the frequency index, and the other two dimensions represent the cross-correlation matrix for each frequency, C_{ifreq}. The numpy function *eigh* is then applied, solving the eigenvector problem for each C_{ifreq}. This optimization ensures that the output SPOD modes are equivalent to those produced by the baseline implementation.

The procedure with the two SPOD implementations is openly available at the following link: https://git.mycloud-links.com/across-public/orchestrator/applications/aeronautics. The online code includes a test dataset and both POD and SPOD procedure for computation of the P_{TKE}.

Algorithm 2 Optimized SPOD algorithm

```
Inputs: Data matrix
% Compute SPOD modes
$U_i = U_i.reshape(M_s, N_b, n_{DFT})$
$\hat{Q}_{Ui}$ =da.fft($U_i$,axis= 2)
$\hat{Q}_{Ui} = \hat{Q}_{Ui}[:,:,0 : n_{DFT}/2+1]$
$\hat{Q}$ =da.stack($\hat{Q}_{Ui}$,axis= 1)
$\hat{Q} = \hat{Q}.reshape(M_s * 3, N_b, n_{DFT}/2+1)$
$\hat{Q} = \hat{Q}.transpose(2,0,1)$
$C$ =da.matmul($\hat{Q}^H, \hat{Q}$)
$\theta$ =np.eigh($C$)
$\varphi_{Ui}$ =da.matmul($(\hat{Q}_{Ui}).T, \theta$)
% with $\hat{Q}_{Ui}.T = \hat{Q}_{Ui}.transpose(2,0,1)$
$\varphi_{Ui} = \varphi_{Ui}.transpose(1,0,2))$
Output: $\phi_M$
```

Figure 2. Optimized Spectral-POD algorithm with no *for* loops.

Scalability tests

Scalability results, concerning the SPOD distributed computation only, are presented here on a relatively large LES dataset. The dataset consists of approximately 80 million computational points and 950 snapshots. Originally stored in CSV files, the snapshots were converted to binaries using the procedure outlined in Biassoni et al. (2024). The binary files (in Parquet format) occupy around 2TB of disk space, as the three velocity variables were stored.

Both strong and weak scalability tests for baseline and optimized SPOD implementations are shown in Figure 3. In the former, the LES database is analyzed with an increasing number of computational nodes to observe the speed-up obtained by increasing the CPUs. Conversely, in the latter, the number of snapshots is progressively increased alongside the number of computational nodes to assess the code efficiency. All tests described here were executed on the CINECA cluster LEONARDO (https://www.hpc.cineca.it/systems/hardware/leonardo/), utilizing the general-purpose CPU partition (DCGP). In brief, each computational node in DCGP is equipped with 2 × 56 Intel Sapphire Rapids cores running at 2.0 GHz and 512 GB of RAM. Strong scalability tests were conducted using a halved snapshot size ($N_t = 480$) to limit the use of resources. The tests have been conducted by setting $N_b = 10$ and consequently $n_{DFT} = 48$, and were repeated at least three times for statistical validity. In Figure 3a, the running time is depicted as black curves, while the computed efficiency is shown in blue, with its corresponding axis on the right for reference. The baseline SPOD implementations is identified by

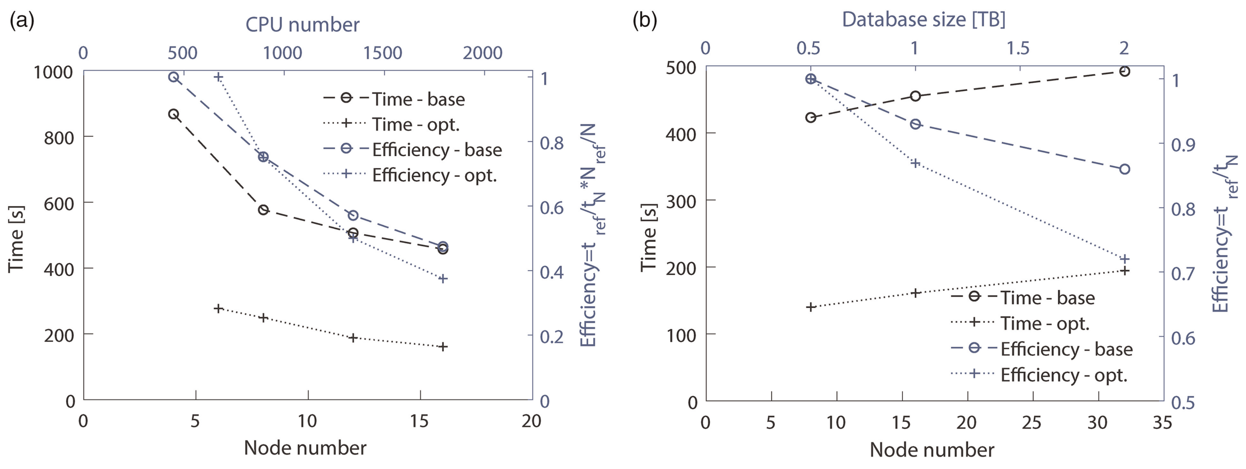


Figure 3. Running time and efficiency of scalability tests: Strong scalability (a); weak scalability (b).

circles, and the optimized one by crosses. The optimized implementation is consistently faster (i.e., lower running time) than the baseline by a factor of about 3. However, while the speed-up is consistent, it is not ideal, as using 16 nodes results in an efficiency of about 40% for both implementations. Note that the baseline implementation could be executed with fewer nodes. This is because the operation in the optimized implementation are performed all at once by the Dask scheduler, which leads to higher memory usage compared to the baseline case. In terms of strong scalability, the present method is less efficient than that of Rogowski et al. (2024) since they obtained a 76% efficiency while increasing the computational resources by a factor of 10.

Weak scalability tests have been performed by increasing the number of snapshots while maintaining the parameter n_{DFT} constant, following the approach outlined in Rogowski et al. (2024). The results, depicted in Figure 3b, illustrate the running time (in black) and efficiency (in blue). Notably, the running time remains almost constant for both implementations. It is below 10 minutes for the baseline code, and about 3 minutes for the optimized one even as the database size reaches 2TB. The efficiency consistently remains high, above 70% for both implementations, despite the dataset size quadrupling. In this scenario, the Dask implementation results seem to surpass those reported in Rogowski et al. (2024), since they achieved a 64% efficiency while tripling the dataset size.

Results and discussion

To demonstrate the potential of the modal decomposition framework for loss analysis, the HPDA code is applied to LES of a typically aft-loaded LPT blade. The simulation was conducted using the commercial software STAR-CCM+ and involving a segregated flow model with bounded-central differencing. A 2nd order implicit backward differentiation scheme has been adopted for the time derivative, while the WALE model was selected for the subgrid-scale model. Given the low Mach number (<0.05) of the selected condition, simulations have been run considering an incompressible flow regime, while Reynolds number has been set to 1.1×10^5. The LPT blades were situated within a stage-like environment, composed of a 2D repeating stage with a degree of reaction equal to 0.5 and the Reynolds number based on the axial chord and exit velocity is $1.1{*}10^5$. The computational grid used for the calculations is a 2D-extruded polyhedral mesh, extending for approximately a 25% of the axial chord in the span-wise direction and comprising around 30 million spatial points, with $y^+ < 1$ and $x^+ = z^+ \approx 20$ all over the blade profile. More details about the LES numerical setup can be found in Russo et al. (2024). Instantaneous data are collected for a total of 4 blade passing periods, corresponding to a total of 560 snapshots. The HPDA application focuses on the vane blade domain. The first step in the procedure, as described in Biassoni et al. (2024), includes data interpolation from the unstructured mesh to a structured one, which is made as uniform as possible to exclude the volumes in the mode computations. Please also note that this dataset is smaller and different than the one used for scalability tests.

Figure 4 displays the contour plot of the fluctuating axial velocity u' for a representative snapshot of the instantaneous flow field (on the left). The wake generated by the upstream blade is visible in the snapshot, marked as blue patterns at the domain's inlet. Each snapshot contributes to the time mean P_{TKE} defined in Equation 2. It is therefore possible to define the quota of P_{TKE} for the kth snapshot at time (t_k) as $P_{TKE}(t_k) = -\rho u_i(t_k)' u_j(t_k)' \frac{\partial \overline{u_i}}{\partial x_j}$. This instantaneous P_{TKE} is shown as a contour plot with superimposed isosurfaces at low and high production on the right of the picture. Intuitively, it is possible to associate the production of TKE with the inlet wake in the middle of the blade passage. Similarly, positive P_{TKE} occurs in the boundary layer, where back-scatter (conversion of turbulence to mean kinetic energy) is also observed with negative value of P_{TKE}. However, it is difficult to decouple the dynamical features by this approach. Namely, the volume integral of the instantaneous P_{TKE} may only allow the identification of where losses occur.

Modal decomposition

The modal decomposition framework aims to overcome the limitation observed in the analysis of the instantaneous flow field. To this purpose POD and SPOD have been already applied to better understand the flow field in LPT (see for example Lengani et al., 2022; Fiore et al., 2023, respectively). However, at least to the best of the authors' knowledge, the two decomposition techniques have not been directly compared in the turbomachinery research field. Figure 5 aims at summarizing their different properties. Figure 5a shows the relationship between the mode number and the frequency. For SPOD this relationship is about linear since the SPOD modes are obtained at a fixed frequency index as discussed in the section Python SPOD procedure. In this case, we considered $N_b = 5$ and ordered the SPOD modes by frequency index. Consequently, the 5 SPOD modes at the 0 frequency index range from 1 to 5, modes 6 to 10 are at the frequency index 1, and so forth. In contrast, POD modes are orthogonal in time and do not have a fixed or ordered frequency index. The DFT of the POD

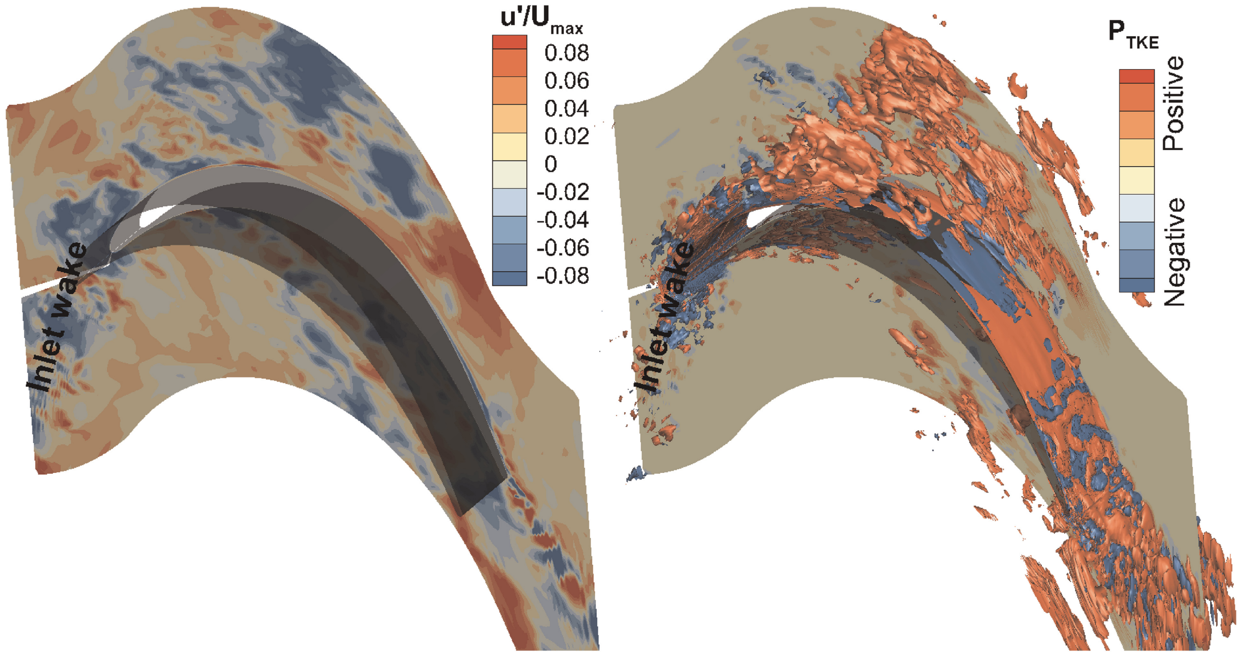


Figure 4. Exemplary time snapshot of the instantaneous flow field (left) and instantaneous P_{TKE} (right).

temporal coefficients may reveal the main frequency of the mode, represented by the black dots in Figure 5a. However, almost all POD modes exhibit more than one active frequency (these active frequencies are depicted by the gray contour). Please note that the inclination of the frequency fit of POD temporal coefficients (dotted blue line) is half of that of SPOD since the SPOD modes are complex-valued and symmetric above the Nyquist frequency. As indicated by Equation 5, each mode carries a portion of the P_{TKE}. The volume integral of P_{TKE} per mode is depicted for the first 15 modes in Figure 5b. Since only $n_{DFT}/2+1$ SPOD modes were computed (as shown in Figure 1), their magnitudes were doubled to account for their symmetric part (above the Nyquist frequency). Both modal decomposition approaches highlight modes that alone capture more than 1% of the total integral value. Particularly, POD modes 1 and 2, being the most energetic, together capture more than 5%. Similarly, SPOD mode 6, representing the most energetic mode at frequency index 1, captures about 5% of the P_{TKE}. According to Figure 5a, POD modes 1 and 2 and SPOD mode 6 have almost the same frequency, or at least, given the lower frequency resolution of SPOD, they fall within the same frequency bin. Furthermore, POD modes are solely ordered by energy, and, likewise, the P_{TKE} per mode decreases as the mode number increases. However, they may represent considerably different dynamics. For instance, the frequency of POD

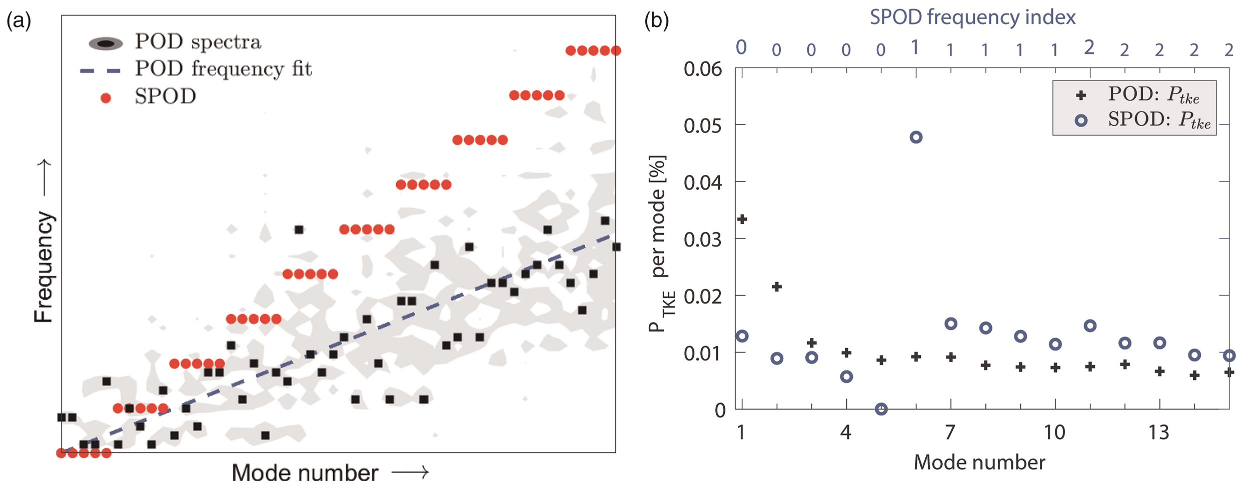


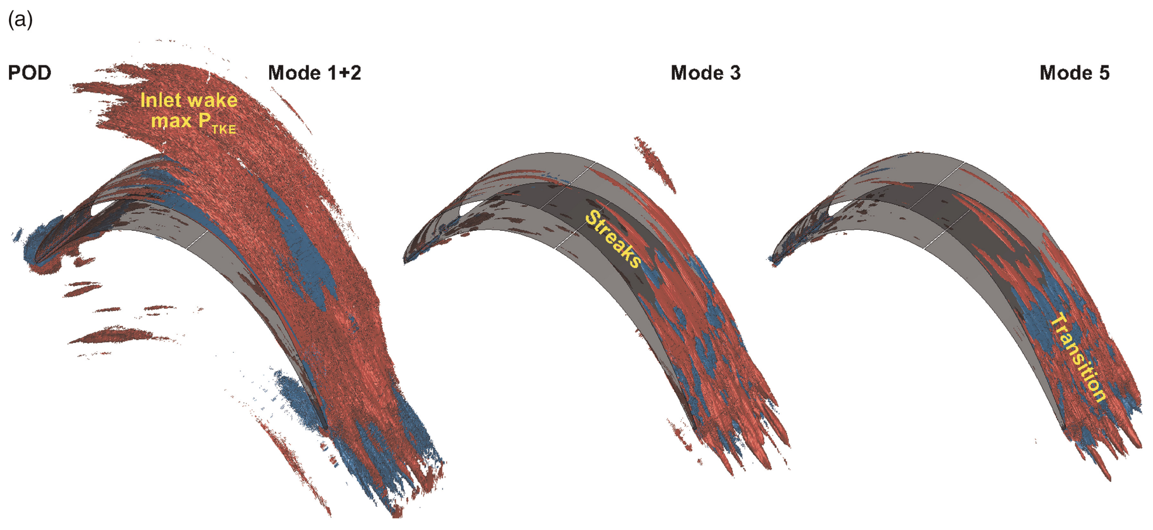
Figure 5. Modal decomposition results and comparison between POD and SPOD. The SPOD modes are arranged in descending order of energy, i.e. the first mode at a given frequency is the most energetic. (a) Frequency-mode relationship and (b) Integral value of P_{TKE} per mode.

modes 3 and 4 is much lower than that of mode 5. In contrast, the SPOD modes clearly highlight the frequency band of interest that groups them (as also indicated by the abscissa at the top of Figure 5b).

Figure 6a and 6b show the local P_{TKE} for selected modes of POD and SPOD, respectively. To be consistent with the above discussion, POD modes 1 and 2, which represent the periodic deterministic disturbance of the inlet wake (as discussed in Lengani et al., 2019), are summed up. This is consistent with the SPOD representation of mode 6 (the most energetic at frequency index 1), since the SPOD mode is complex and its symmetric DFT part is considered by doubling the real-valued P_{TKE}. The region of maximum P_{TKE} is located within the blade passage, where the highest interaction between the deterministic velocity fluctuations of the inlet wake and the strain within the blade passage occurs. This region is highlighted in yellow in the plots of POD modes 1+2 and SPOD mode 6. Although both representations are quite similar, POD is energetically optimal as it captures larger iso-surfaces. This may be also related to an earlier convergence of POD with respect to SPOD.

POD modes 3 and 5 capture structures close to the SPOD frequency index 0 and 2 (according to Figure 5a). Namely, the representation of SPOD mode 1 at 0 frequency index is close to POD mode 3, since both track low-frequency streaks (marked in yellow) in the fore part of the blade. This is not the case for SPOD mode 11 at frequency index 2. POD mode 5, in fact, highlights structures related to transition (as marked over the plot) since the iso-surfaces of P_{TKE} appear small and their production (red contour) is limited to the rear part of the suction side (see also Lengani et al., 2022). On the other hand, SPOD mode number 11 at frequency index 2 still highlights a streamwise elongated area of high P_{TKE}, being representative of streaky structures related to the inlet wake perturbations.

The observations above are confirmed and quantified by the bar charts of Figure 7, where the contribution of the modes under examination to the total P_{TKE} is reported as a fraction for each subregion of interest within the



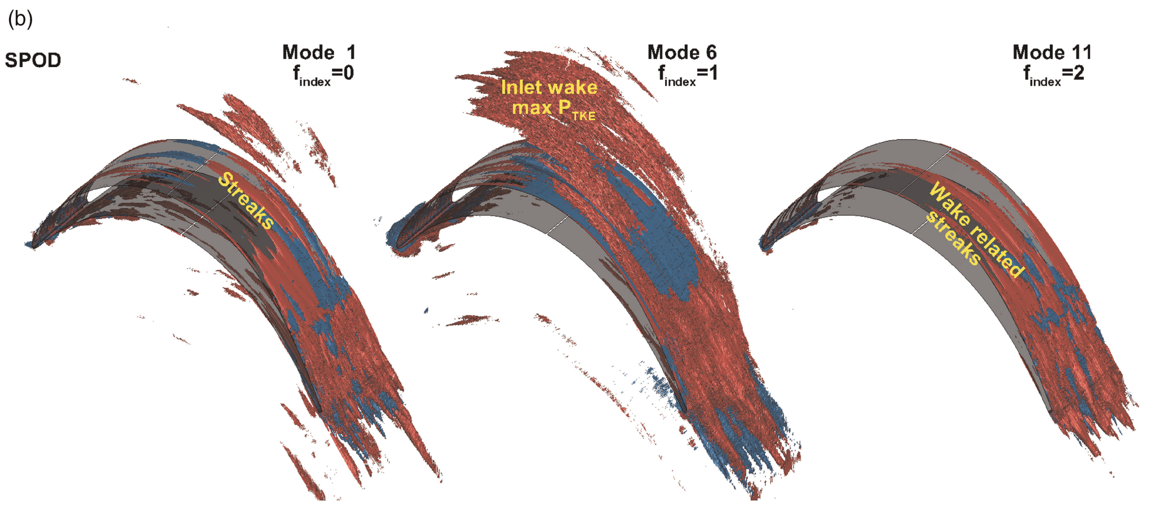


Figure 6. POD (a) and SPOD (b) mode decomposition, P_{TKE} per mode, red and blue iso-surfaces identify regions of production and backscatter, respectively. The most energetic SPOD modes for each frequency index are shown.

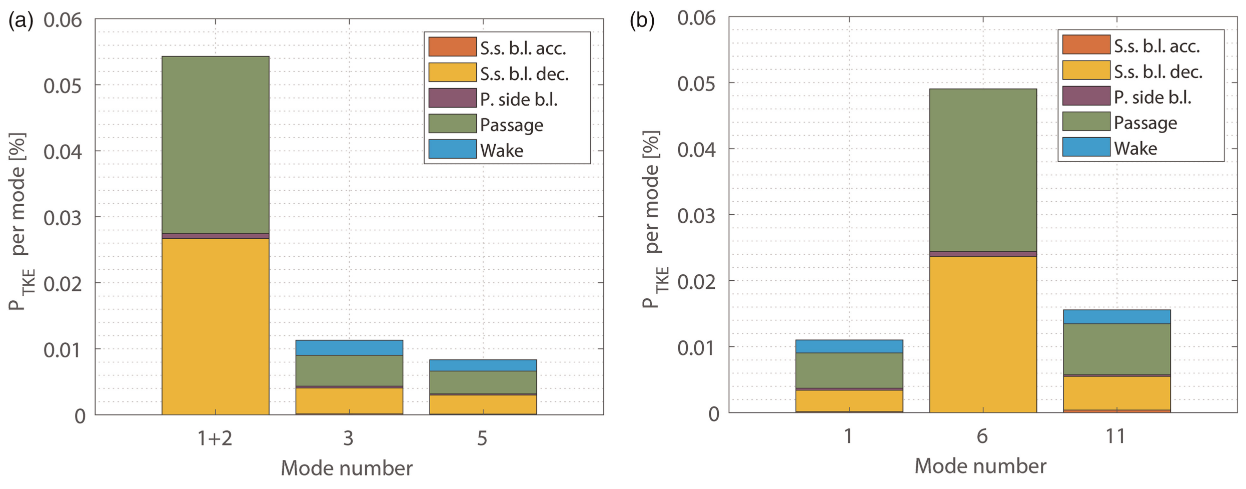


Figure 7. Bar charts of mode contribution to P_{TKE} per domain zone: POD results (a), SPOD (b).

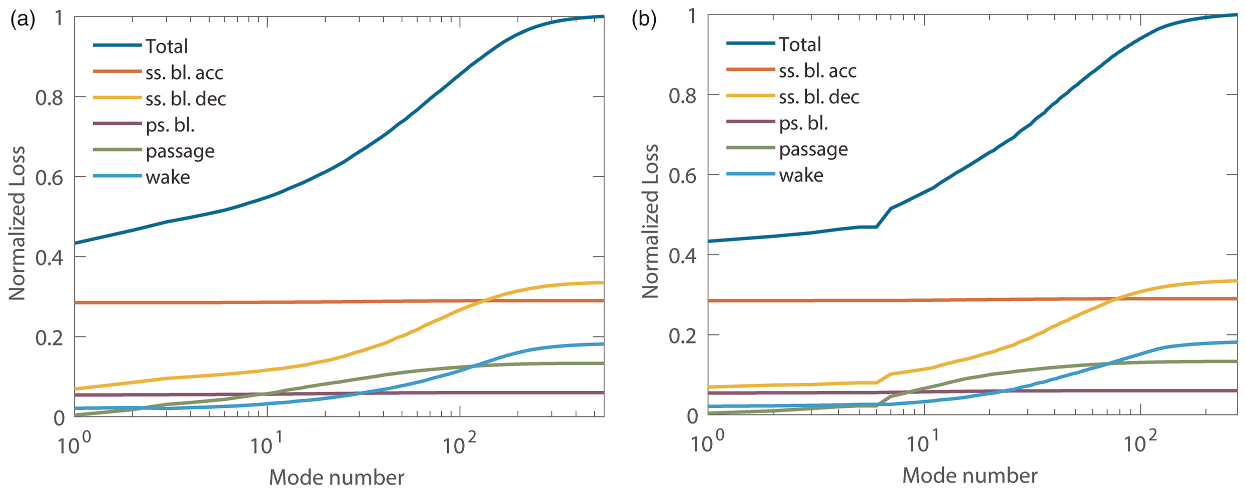


Figure 8. Cumulative contribution of modes to the overall losses: POD results (a), SPOD (b).

computational domain. This is done by leveraging the properties of the volume integral approach, by which 5 different integration regions are defined: the accelerating part of the suction side in red, its decelerating part in yellow, the pressure side in purple, the wake region (downstream of the trailing edge) in light blue and the passage region in green.

Overall results, as previously discussed in Lengani et al. (2019) and Dellacasagrande et al. (2023), are summarized with the contribution per mode of the volume integral of P_{TKE} (see Equation 5). Figure 8 shows the cumulative loss contribution per mode of POD and SPOD (left and right, respectively). The total loss is depicted in blue. Both modal decompositions lead to the same final integral value and the accuracy of the proposed methodology is within 3% of the inlet to outlet loss computation (see Biassoni et al., 2024). Note that the viscous dissipation is a reference value taken as mode 0. The main difference between POD (Figure 8a) and SPOD (Figure 8b) lies in the distribution of loss with respect to mode number. SPOD makes clear that low frequency content (first 5 modes) has a minor contribution only in the passage region. Otherwise, loss generation in the decelerating part of the suction side and in the wake region is triggered by higher frequency content. It is interesting to note that SPOD representation is almost as optimal as POD while in this latter case the frequency contents are somewhat mixed.

Conclusion

The paper provides an open-access HPDA code for the analysis of the transport equation of total pressure in high-fidelity simulations of turbomachinery. The algorithm of the procedure and its implementation, utilizing

Python and the Dask library, are discussed in detail. Strong and weak scalability tests are provided and compared with similar codes in the literature. The current procedure exhibits less efficiency in terms of strong scalability. However, it demonstrates highly efficient weak scalability, making it well-suited for handling very large datasets with low programming effort.

The HPDA code is applied to LES of a typical aft-loaded LPT blade. POD optimally captures (from a mathematical perspective) flow structures and ranks them by energy and, therefore, by P_{TKE}. Conversely, SPOD ranks them by frequency, facilitating the grouping of dynamic features based on frequency bands. Considering the rapid execution time of the current code on large databases, it holds promise for analyzing the growing volume of high-fidelity simulations within the turbomachinery domain. This could lead to achieving an optimal identification of loss sources by pinpointing the dynamic features responsible for these losses.

Acknowledgments

We acknowledge the CINECA award under the ISCRA initiative, for the availability of high-performance computing resources and support.

Competing interests

Daniele Biassoni declares that he has no conflict of interest. Matteo Russo declares that he has no conflict of interest. Davide Lengani declares that he has no conflict of interest.

Funding sources

European HPC Joint Undertaking (JU) under grant agreement No. 955648 provided funding within the ACROSS project for this work.

References

Biassoni D., Russo M., Viviani P., Vitali G., and Lengani D. (2024). A high-performance code for analyzing loss transport equations in high-fidelity simulations. In: ASME Turbo EXPO 2024. paper No. GT2024-127953. London, United Kindom: American Society of Mechanical Engineers (ASME).

Dellacasagrande M., Lengani D., Simoni D., Ubaldi M., Granata A. et al. (2023). Analysis of unsteady loss sensitivity to incidence angle variation in low pressure turbine. In: ASME Turbo Expo 2023. paper no. GT2023-103770. Boston, Massachusetts, USA: American Society of Mechanical Engineers (ASME).

Denton J. D. (1993). Loss mechanisms in turbomachines. *Journal of Turbomachinery October 1993*. 115 (4): 621–656.

Fiore M., Gojon R., Sáez-Mischlich G., and Gressier J. (2023). LES of the T106 low-pressure turbine: spectral proper orthogonal decomposition of the flow based on a fluctuating energy norm. *Computers & Fluids*. 252: 105761. https://doi.org/10.1016/j.compfluid.2022.105761.

He X., Fang Z., Rigas G., and Vahdati M. (2021). Spectral proper orthogonal decomposition of compressor tip leakage flow. *Physics of Fluids*. 33 (10): 105105.

Leggett J., Richardson E., Priebe S., Shabbir A., Michelassi V., and Sandberg R. (2020). Loss analysis of unsteady turbomachinery flows based on the mechanical work potential. *Journal of Turbomachinery*. 142 (11): 111009. https://doi.org/10.1115/1.4048162.

Lengani D., Simoni D., Pichler R., Sandberg R., Michelassi V., and Bertini F. (2019). On the identification and decomposition of the unsteady losses in a turbine cascade. *Journal of Turbomachinery*. 141 (3): 031005. https://doi.org/10.1115/1.4042164.

Lengani D., Simoni D., De Vincentiis L., Đurović K., Pralits J. et al. (2022). Investigation on strain and stress principal axes in unsteady DNS turbine data. In: ASME Turbo Expo 2022. paper no. GT2022-83197. Rotterdam, Netherlands: American Society of Mechanical Engineers (ASME).

Miki K. and Ameri A. (2022). Numerical investigation of the effect of trailing edge thickness of simulated CMC blades on loss profiles. In: ASME Turbo Expo 2022. paper no. GT2022-82335. Rotterdam, Netherlands: American Society of Mechanical Engineers (ASME).

Perrin R., Braza M., Cid E., Cazin S., Barthet A. et al. (2007). Obtaining phase averaged turbulence properties in the near wake of a circular cylinder at high reynolds number using POD. *Experiments in Fluids*. 43: 341–355. https://doi.org/10.1007/s00348-007-0347-6.

Przytarski P. J. and Wheeler A. P. (2020). The effect of gapping on compressor performance. *Journal of Turbomachinery*. 142 (12): 121006. https://doi.org/10.1115/1.4047933.

Przytarski P., Lengani D., and Sandberg R. (2024a). The impact of inflow unsteadiness on loss prediction. *ASME. J. Turbomach*. 147 (4): 041014.

Przytarski P. J., Lengani D., Simoni D., and Wheeler A. P. (2024b). The role of turbulence transport in mechanical energy budgets. *Journal of Turbomachinery*. 146 (10): 101008. https://doi.org/10.1115/1.4065287.

Rogowski M., Yeung B. C., Schmidt O. T., Maulik R., Dalcin L. et al. (2024). Unlocking massively parallel spectral proper orthogonal decompositions in the pyspod package. *Computer Physics Communications*. 302: 109246. https://doi.org/10.1016/j.cpc.2024.109246.

Russo M., Carlucci A., Dellacasagrande M., Petronio D., Lengani D. et al. (2024). Optimization of low-pressure turbine blade by means of fine inspection of loss production mechanisms. *Journal of Turbomachinery*. 147 (4): 041005. https://doi.org/10.1115/1.4066818.

Sandberg R. and Michelassi V. (2022). Fluid dynamics of axial turbomachinery: blade-and stage-level simulations and models. *Annual Review of Fluid Mechanics*. 54: 255–285. https://doi.org/10.1146/fluid.2022.54.issue-1.

Schmidt O. T. and Colonius T. (2020). Guide to spectral proper orthogonal decomposition. *AIAA Journal*. 58 (3): 1023–1033. https://doi.org/10.2514/1.J058809.

Towne A., Schmidt O. T., and Colonius T. (2018). Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *Journal of Fluid Mechanics*. 847: 821–867. https://doi.org/10.1017/jfm.2018.283.

Viviani P., Vitali G., Lengani D., Scionti A., Vercellino C., and Terzo O. (2022). Taming multi-node accelerated analytics: An experience in porting Matlab to scale with python. In: Complex, Intelligent and Software Intensive Systems, edited by Barolli, L. CISIS 2022. Lecture Notes in Networks and Systems, vol 497. Springer, Cham.

Wheeler A. P., Sandberg R. D., Sandham N. D., Pichler R., Michelassi V., and Laskowski G. (2016). Direct numerical simulations of a high-pressure turbine vane. *Journal of Turbomachinery*. 138 (7): 071003. https://doi.org/10.1115/1.4032435.

Zhao Y. and Sandberg R. (2020). Using a new entropy loss analysis to assess the accuracy of RANS predictions of an high-pressure turbine vane. *Journal of Turbomachinery*. 142 (8): 081008. https://doi.org/10.1115/1.4046531.