Compression of aerodynamic databases using high-order singular value decomposition

L.S. Lorente [a], J.M. Vega [a], A. Velazquez [b]

[a] *Aerospace Propulsion and Fluid Mechanics Department, School of Aeronautics, Universidad Politécnica de Madrid, Plaza Cardenal Cisneros 3, 28040 Madrid, Spain*
[b] *Applied Mathematics Department, School of Aeronautics, Universidad Politécnica de Madrid, Plaza Cardenal Cisneros 3, 28040 Madrid, Spain*

ARTICLE INFO

ABSTRACT

A methodology based on high-order singular value decomposition is presented to compress multidimensional (with the various dimensions associated with both the spatial coordinates and parameter values) aerodynamic databases. The method is illustrated with a database containing computational fluid dynamics calculations of the outer flow around a wing, with two free parameters, the Mach number and the angle of attack. Comparison is made between the results of compressing just one flow snapshot (for fixed values of the parameters), compressing a one-parameter family of snapshots, and compressing the whole database. Several compressing strategies are also discussed that deal with (a) treating the flow variables separately or considering all flow variables at a time, (b) considering the whole flow domain simultaneously or dividing it into blocks, and (c) using various measures of errors. The main conclusion is that a large compression factor is generally obtained. Furthermore, the compression factor increases exponentially as the dimension of the database increases for any fixed error, namely the compression factor increases by an order of magnitude with each new database dimension for an error level of 1%.

1. Introduction

One of the current trends of engineering design in the aeronautic industry is the growing importance of numerical simulation activities. The reason is that computer simulation is both very flexible and cost-competitive compared to wind tunnel testing, and is becoming more and more reliable. This, of course, does not mean that experimental testing is to be disposed of. On the contrary, wind tunnel tests are used both to calibrate numerical simulation methods and to study complex configurations.

On the practical side, the daily use of computational fluid dynamics (CFD) for design purposes means that a vast amount of information is generated continuously. This fact has the obvious drawback of storage. Even though the price of storage devices has dropped sharply in the last few years, the exponentially growing number and size of stored files remains a source of trouble for industrial companies. However, the main difficulty associated with the massive use of CFD could be related to the fact that working sites within a given aeronautic company tend to be spread out all over a large geographic area. Also, design teams are seldom confined to a single location and it is commonplace that a great amount of information continuously travels back and forth between the different sites. Keeping this flow of information alive has a significant cost, which leads to an obvious question: is all the data being transferred really required to convey the needed information or is it a case of over kill? The answer to this question cannot be made in general terms. On the contrary, it is likely that it may vary depending on the technical discipline involved. For instance, regarding CFD-generated aerodynamic flow field calculations, the question is whether the values of all flow variables are needed at all mesh points or the database itself could be described accurately using a substantially smaller amount of information.

In this context, Leng [9] reported back in the late nineties a method to compress aerodynamic databases based on multivariable Chebyshev polynomials. These polynomials were used to generate function approximations of the tabular aerodynamic database. In particular, the author presented an application test case related to the F16 fighter aircraft. Curiously enough, there is little else in the literature in connection to the aeronautic field. This is not the case, for example, in pharmaceutical applications, where Miled et al. [11] have used compression techniques to map out drug databases. Among the many different mathematical techniques that could be used for database compression purposes, *singular value decomposition* (SVD) approaches [6] have been reported by various authors in different fields. For example, del-Castillo-Negrete et al. [4] considered various SVD based algorithms to compress three-dimensional magneto-hydrodynamic databases; in particular, they used a generalised low rank approximation [17], which requires an iterative process and is suitable for databases in which one of the dimensions is dominant. Also, Kanth et al. [12]

have developed a novel SVD technique that allows for image reconstruction with an error caused by the approximate computation smaller than 10%. Another SVD based methodology for face image recognition has been reported by Sakalli et al. [13], who developed a step-wise approach to the problem that involved the *Karhunen–Loeve transform* of clustered image blocks. More recently, a variant of SVD called *partial singular value decomposition* has been applied by Tougas and Spiteri [16] to latent semantic indexing, which is an information retrieval method. In particular, the authors focused on achieving substantial savings in computational time with small losses of accuracy. Finally, although not directly related to SVD, it is worth citing the work by Lee et al. [8], who pointed out the relation between data mining and database compression. What the authors do in this work is to eliminate redundant information in databases, prior to compression, by finding out association rules. Then, once the redundant information is discarded, compression can proceed smoothly.

The purpose of this paper is to explore the feasibility of using SVD based techniques to compress aerodynamic related information of aeronautic interest. In fact, an extensive use will be made of *high-order singular value decomposition* (HOSVD), an extension to tensors of SVD, which only applies to matrices. The simplest aerodynamic database is an ASCII file with as many rows as points in the CFD computational mesh, and eight columns containing the spatial coordinates, x, y and z, the three velocity components, the pressure, and the temperature. Since the number of points in the 3-D computational domain is of the order of 10^6, the size of the ASCII file resulting from each computation runs easily into the hundreds of megabytes or even into the gigabyte range. Those calculations must be made for many combined values of the various relevant parameters, like the angle of attack, yaw angle, deflection angles of various control surfaces (ailerons, rudders, etc.), and the Reynolds and Mach numbers. On the other hand, it is to be recognised that this large amount of information is described by five equations (the continuity, compressible Navier–Stokes, and energy equations) and the associated boundary conditions containing in particular the shape of the body whose aerodynamic behaviour is being analysed. Typically, in practical industrial applications, the size of body surface definition files is of the order of 1% of the whole database file. Then, the question arises as to whether it is feasible to find out an intermediate description of the problem that is reasonably accurate in between of the two limits already stated above: the full database (N terabytes) and the equations and boundary conditions (impractical). In this context, a description of the HOSVD method, including the specific variants needed to address aerodynamic problems of industrial interest, are presented first, in Section 2. Then, a test case is described in Section 3, results are presented and discussed in Section 4, and conclusions are stated in Section 5.

2. High-order singular value decomposition

HOSVD [2] is an extension to (third- or higher-order) tensors of standard SVD, which only applies to matrices and is recalled first for the sake of clarity. The SVD of an $m \times n$ matrix A is of the form [6] $A = U \cdot D \cdot V^T$, where superscript T stands for transpose, U and V are orthogonal matrices (namely, $U \cdot U^T = U^T \cdot U = I$ and $V \cdot V^T = V^T \cdot V = I$), and D is a diagonal, positive definite matrix with r nonzero elements, called the *singular values* of A; here, r is the *rank* of A, defined as usually. The SVD of A is written in terms of the elements of the matrix as

$$A_{ij} = \sum_{l=1}^{r} \delta_l u_{il} v_{jl} \tag{1}$$

where $\delta_1, \ldots, \delta_r$ are the singular values of A, and for each value of the subscript l, u_{il} and v_{il} are the elements of the first r columns of U and V, respectively, which are known as the left and right SVD modes, respectively. The singular values of A are calculated as the square root of the strictly positive eigenvalues of the positive definite, symmetric matrix $A \cdot A^T$, which coincide with the strictly positive eigenvalues of the matrix $A^T \cdot A$. The left and right SVD modes are the (orthonormal) eigenvectors of the matrices $A \cdot A^T$ and $A^T \cdot A$ (associated with strictly positive eigenvalues), respectively.

If the singular values are sorted in a decreasing order and the decomposition (1) is truncated to s ($< r$) terms, the associated truncation error (in terms of the *Frobenius norm*, $\|A\|^2 = \sum_{i,j}(A_{ij})^2$) is equal to

$$\left[\sum_{l=s+1}^{r} (\delta_l)^2\right]^{1/2}$$

When this error is small enough, then a good approximation of the $m \times n$ elements of the matrix A can be stored by means of $s + s \times m + s \times n$ numbers. This means that the compression factor scales with $\min(m/s, n/s)$ when s is much smaller than both m and n. The latter occurs when the elements of A exhibit some correlation (due to, e.g., a physical law). In that case, a quite effective storing compression results.

Let us now consider higher-order tensors. Even though the description that follows applies to tensors of any order, for the sake of clarity a third-order, $m_1 \times m_2 \times m_3$ tensor will be considered. The immediate extension of Eq. (1) to the tensor A would be a decomposition of A in terms of *rank-one* tensors, as

$$A_{ijk} = \sum_{l=1}^{r} \delta_l u_{il} v_{jl} w_{kl} \tag{2}$$

where the minimal number r of rank-one tensors is called the *rank* of A. However, both determination of the rank of a higher-than-two-order tensor and construction of effective algorithms to calculate the associated minimal decomposition (2) are open problems nowadays [3]. Furthermore, calculation of minimal decompositions is an ill-posed problem [1]. Thus instead of (2), other less restrictive decompositions have been tried [7]. Among these, HOSVD is of the form [2,7]

$$A_{ijk} = \sum_{\eta=1}^{r_1} \sum_{\mu=1}^{r_2} \sum_{\varsigma=1}^{r_3} \sigma_{\eta\mu\varsigma} u_{i\eta} v_{j\mu} w_{k\varsigma} \tag{3}$$

where $\sigma_{\eta\mu\varsigma}$ are the components of another third-order tensor, sometimes called the *core tensor*, and for each set of values of the indexes η, μ, and ς, $u_{i\eta}$, $v_{j\mu}$, and $w_{k\varsigma}$ are the elements of three vectors that are known as the *HOSVD modes*. This makes an essential difference with the decomposition (2), where the singular values δ_l depend only on one index. A second important difference is that calculation of both, the core tensor and the *HOSVD modes* involve well posed mathematical problems, which are explained now. For each set of values of the indexes η, μ, and ς, the HOSVD modes are the (orthonormal) eigenvectors associated with the strictly positive eigenvalues of the positive definite, symmetric matrices B^1, B^2, and B^3, defined as

$$B^1_{il} = \sum_{j,k} A_{ijk} A_{ljk}, \qquad B^2_{jl} = \sum_{i,k} A_{ijk} A_{ilk}$$
$$B^3_{kl} = \sum_{i,j} A_{ijk} A_{ijl} \tag{4}$$

respectively. Namely, the HOSVD modes are given by

$$\sum_{l=1}^{m_1} B^1_{il} u_{l\eta} = (\alpha_\eta)^2 u_{i\eta}, \quad \eta = 1, \ldots, r_1$$

$$\sum_{l=1}^{m_2} B^2_{jl} v_{l\mu} = (\beta_\mu)^2 v_{j\mu}, \quad \mu = 1, \ldots, r_2$$
$$\sum_{l=1}^{m_3} B^3_{kl} u_{l\varsigma} = (\gamma_\varsigma)^2 w_{k\varsigma}, \quad \varsigma = 1, \ldots, r_3 \tag{5}$$

where (for $i = 1, 2, 3$) r_i ($\leqslant m_i$) is the rank of the matrix B^i; the positive scalars α_η, β_μ, and γ_ς will be referred to as the *high-order singular values* (HOSVs) of the decomposition. Once the HOSVD modes have been calculated, the core tensor $\sigma_{\eta\mu\varsigma}$ is readily obtained multiplying Eq. (3) by the (i, j, k) component of the HOSVD modes, adding in the indexes i, j, and k, and recalling that these are orthonormal. It follows that

$$\sigma_{\eta\mu\varsigma} = \sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{k=1}^{m_3} A_{ijk} u_{i\eta} v_{j\mu} w_{k\varsigma} \tag{6}$$

Note that calculating each element of the core tensor, which is the most expensive process of HOSVD in terms of CPU time, requires a number of operations that is proportional to the size of the original tensor. Therefore, the number of operations to compute the whole core tensor scales with $(m_1 \times m_2 \times m_3) \times (r_1 \times r_2 \times r_3)$, which makes it crucial to reduce the number of retained elements in the core tensor. Thus, if the three sets of eigenvalues are sorted in a decreasing order and the decomposition (3) is truncated to $s_1 \leqslant r_1$, $s_2 \leqslant r_2$, and $s_3 \leqslant r_3$ (with at least one of these inequalities being a strict inequality) terms, as

$$A_{ijk} \cong \sum_{\eta=1}^{s_1}\sum_{\mu=1}^{s_2}\sum_{\varsigma=1}^{s_3} \sigma_{\eta\mu\varsigma} u_{i\eta} v_{j\mu} w_{k\varsigma} \tag{7}$$

then the error (in terms of the *Frobenius norm* $\|A\|^2 = \sum_{i,j,k}(A_{ijk})^2$) of the truncated reconstruction of A is bounded by (see [2])

$$\|\text{error}\| \leqslant \sqrt{\sum_{\eta=s_1+1}^{r_1} (\alpha_\eta)^2 + \sum_{\mu=s_2+1}^{r_2} (\beta_\mu)^2 + \sum_{\varsigma=s_3+1}^{r_3} (\gamma_\varsigma)^2} \tag{8}$$

which provides an a priori error bound that allows to estimate the error before calculating the decomposition. As it happened with SVD, truncated HOSVD allows to store an approximation of the $m_1 \times m_2 \times m_3$ elements of the tensor A by means of only the $s_1 \times s_2 \times s_3 + s_1 \times m_1 + s_2 \times m_2 + s_3 \times m_3$ numbers involved in the right hand side of Eq. (7). This yields a strong memory saving when the compression factor,

$$\text{COMPRESSION FACTOR } (C_F) = \frac{m_1 \times m_2 \times m_3}{s_1 \times s_2 \times s_3 + s_1 \times m_1 + s_2 \times m_2 + s_3 \times m_3} \tag{9}$$

is large; the saved memory (in %) is $(1 - 1/C_F) \cdot 100$. Now, the compression factor scales with $\min(m_1 \times m_2 \times m_3/s_1 \times s_2 \times s_3, m_2 \times m_3/s_1, m_1 \times m_3/s_2, m_1 \times m_2/s_3)$ when s_1, s_2, and s_3 are much smaller than m_1, m_2, and m_3, respectively. As above, the latter occurs in particular when redundancies are present among the elements of the tensor A, which can be due to, e.g., physical laws like the Navier–Stokes equations.

Furthermore, higher-dimensional databases behave even better. If redundancies affect all the tensor dimensions simultaneously (as it happens frequently in physical problems such as aerodynamics problems), the ratios s_i/m_i remain small and almost constant as the order of the tensor is increased, which means that the compression factor increases exponentially as tensors of increasingly higher order are considered. This will be illustrated in Section 4 for an aerodynamic test problem and is quite promising keeping in mind applications to aerodynamic databases for full aircraft configurations, in which the number of relevant parameters can be really high, as mentioned above.

All these make a difference with compression methods based on SVD, which do not take full advantage of redundancies in all dimensions of the database. Direct application of SVD (either applying SVD to two-dimensional slices of the third-order tensor or considering two of the indexes as a joint index) gives a compression factor that is an order of magnitude smaller than that resulting from applying HOSVD, as will be illustrated in Section 4. Other, more sophisticated (iterative) applications of SVD, like that based on generalised low rank approximations [4,17], provide compression factors that are somewhere in between of those provided by HOSVD and direct application of SVD; but such improvement requires that the tensor be appropriate (one of the tensor dimensions must be dominant) and the extension to higher-(than three)-order tensors is not obvious.

Now, denoting the element-by-element error of the reconstructed tensor (7) as error_{ijk}, averaged errors will be measured below in two ways. The *root mean square error* (RMSE) in % is defined as usually, namely

$$\text{RMSE} \equiv \sqrt{\frac{1}{N}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{k=1}^{m_3}(\text{error}_{ijk})^2} \times \frac{100}{(A_{\max} - A_{\min})} \leqslant \text{APEB} \tag{10}$$

where $N = m_1 \times m_2 \times m_3$ is the number of elements of the tensor A, $A_{\max}$ and $A_{\min}$ are the maximum and minimum values of the tensor elements A_{ijk} and the stated inequality results from Eq. (8). Here, APEB (a priori error bound) denotes the a priori RMSE bound in %. Invoking Eq. (8), APEB is seen to be given by

$$\text{APEB} \equiv \sqrt{\frac{1}{N}\left[\sum_{\eta=s_1+1}^{r_1} (\alpha_\eta)^2 + \sum_{\mu=s_2+1}^{r_2} (\beta_\mu)^2 + \sum_{\varsigma=s_3+1}^{r_3} (\gamma_\varsigma)^2\right]} \times \frac{100}{(A_{\max} - A_{\min})} \tag{11}$$

In addition, we shall use the *mean error* (ME), again in %, defined as

$$\text{ME} \equiv \frac{1}{N}\sum_{i=1}^{m_1}\sum_{j=1}^{m_2}\sum_{k=1}^{m_3}|\text{error}_{ijk}| \times \frac{100}{(A_{\max} - A_{\min})} \leqslant \text{RMSE} \tag{12}$$

where the stated inequality readily follows invoking the standard Cauchy–Schwartz inequality and shows that bounding RMSE is more stringent than bounding ME. Now, when using HOSVD, the numbers of modes in each dimension (s_1, s_2, and s_3) are selected as the minimum number of modes ($s_1 + s_2 + s_3$) that meet the requirement that the truncation error be smaller than some small quantity ε, which can be done in two ways. Namely, either

(i) Requiring that the APEB defined in Eq. (11) be smaller than ε, which involves a less computational effort but provides a non-optimal selection of s_1, s_2, and s_3.
(ii) Iterating the procedure (i) as follows. For each value of the APEB, we calculate the numbers of HOSVD modes s_1, s_2, and s_3, and also calculate the ME and the RMSE, using the exact expressions in (10) and (12), which needs to calculate error_{ijk}. Then, we iterate on the varying value of the APEB, to obtain the numbers of modes that meet the requirements that either $\text{ME} \leqslant \varepsilon$ or $\text{RMSE} \leqslant \varepsilon$.

The strategy (i) is more convenient when CPU time is limited and the compressed database is to be used only a few times, while

alternative (ii) is better when CPU time is not an issue and the compressed database is to be used many times. In order to get some insight into the practical implementation of the method, results on using ME and RMSE to bound the error, and estimating the number of modes by means of the strategies (i) and (ii) will be discussed in Section 4.

Summarising, a third-order tensor A can be written in the form (3), with the HOSVD modes calculated as the eigenvectors (5), whose associated eigenvalues provide the a priori error bound given in Eq. (11). Then, the expansion (3) is truncated invoking (10)–(12), obtaining the compressed form (7), where the core tensor $\sigma_{\eta\mu\varsigma}$ is calculated using (6). Fourth- and higher-order tensors are treated similarly.

In some cases, the whole database can be divided into blocks in such a way that each of these is also given by a third-order tensor. This could lead to a better compression when information in different blocks is somewhat independent. In fact, if the blocks were completely independent, the number of HOSVD modes in the complete database would be the product of the numbers of modes in the various blocks; if instead the various blocks are highly correlated, then the compression ratio will slightly worsen because redundancies between the blocks will not be fully accounted for. When dividing into blocks, the HOSVD method described above must be applied to each block, labelled with the superscript (k); the root mean square error $\mathrm{RMSE}^{(k)}$ and the mean error $\mathrm{ME}^{(k)}$ are calculated as explained above. The resulting overall a priori error bound, the compression factor, and the overall errors are (cf. Eqs. (10)–(12))

$$\mathrm{APEB}^{(k)} \equiv \sqrt{\frac{1}{N^{(k)}}\left[\sum_{\eta=s_1^{(k)}+1}^{r_1^{(k)}}\left(\alpha_\eta^{(k)}\right)^2+\sum_{\mu=s_2^{(k)}+1}^{r_2^{(k)}}\left(\beta_\mu^{(k)}\right)^2+\sum_{\varsigma=s_3^{(k)}+1}^{r_3^{(k)}}\left(\gamma_\varsigma^{(k)}\right)^2\right]}\times\frac{100}{A_{\max}-A_{\min}} \tag{13}$$

$$\mathrm{APEB}=\sqrt{\frac{1}{N}\sum_{k=1}^{N_{\mathrm{blocks}}}\left[\left(\mathrm{APEB}^{(k)}\right)^2\cdot N^{(k)}\right]} \tag{14}$$

$$\mathrm{COMPRESSION\ FACTOR}=\frac{m_1\times m_2\times m_3}{\sum_{k=1}^{N_{\mathrm{blocks}}}[s_1^{(k)}\times s_2^{(k)}\times s_3^{(k)}+s_1^{(k)}\times m_1^{(k)}+s_2^{(k)}\times m_2^{(k)}+s_3^{(k)}\times m_3^{(k)}]} \tag{15}$$

$$\mathrm{RMSE}=\sqrt{\frac{1}{N}\sum_{k=1}^{N_{\mathrm{blocks}}}\left(\mathrm{RMSE}^{(k)}\right)^2\cdot N^{(k)}}$$

$$\mathrm{ME}=\frac{1}{N}\sum_{k=1}^{N_{\mathrm{blocks}}}\mathrm{ME}^{(k)}\cdot N^{(k)}\leqslant\mathrm{RMSE} \tag{16}$$

where $N^{(k)}=m_1^{(k)}\times m_2^{(k)}\times m_3^{(k)}$ and $N=m_1\times m_2\times m_3$ denote the number of mesh points in the k-th block and in the whole mesh, respectively, $s_1^{(k)}$, $s_2^{(k)}$, and $s_3^{(k)}$ are the numbers of HOSVD modes in the k-th block, $r_1^{(k)}$, $r_2^{(k)}$, and $r_3^{(k)}$ are the ranks of the matrices $B^{1(k)}$, $B^{2(k)}$, and $B^{3(k)}$, defined in Eq. (4) for the k-th block, and $\alpha_\eta^{(k)}$, $\beta_\mu^{(k)}$, and $\gamma_\varsigma^{(k)}$ are the HOSVs associated with the HOSVD in the k-th block, which are calculated from (5). Also, dividing into blocks leads to a more homogeneous distribution of errors if, as will be done below, all blocks are required to show the same APEB. This is because local errors exhibit a smaller effect in the APEB of the whole wing than in the APEB of the particular block where the errors are localised.

3. Test problem and specific application of HOSVD

We consider the free stream flow around a 3-D wing whose span length is 1.5 times the root chord, see Fig. 1a. The Reynolds number is maintained fixed and equal to $20\cdot 10^6$, but two additional parameters, the Mach number M and the angle of attack α, are allowed to vary in the ranges from 0.4 to 0.8 and from -3° to 3°, respectively. The database to be compressed below has been generated in a standard way, namely using a finite volume numerical scheme [15], based on the usual Navier–Stokes equations plus the Edwards-corrected [5] Spalart–Allmaras turbulence model [14], and some small higher-order terms added to avoid numerical instability; further details on the numerical code will not be necessary below. The piece of the structured computational mesh associated with the database corresponds to a vicinity of the wing (see Fig. 1a), which contains 225 mesh points along the chords, 101 mesh points in the spanwise direction, and 37 mesh points along the wall-normal direction, which gives a total number of 840,825 mesh points. This structured mesh is divided into 16 blocks in the CFD code. Each block corresponds to fixed numbers of chordwise and spanwise mesh points, as illustrated in Fig. 1b and contains all mesh points in the wall-normal direction; thus a structured submesh is defined in each block. Blocks are selected in a way that they exhibit a more or less uniform amount of flow information. Therefore, that division into blocks appears as reasonable also to apply HOSVD block by block, which will be done below.

For each pair of values of the parameters M and α, the snapshot of the fluid flow involves the values at these points of the three velocity components, u, v, and w, the density ρ, the temperature T, and the pressure p. CFD computations have been performed and stored in the database for the $13\times 9=117$ combinations of the following values of parameters M and α (see Fig. 2):

- α (13 values): -3°, -2.5°, -2°, -1.5°, -1°, -0.5°, 0°, 0.5°, 1°, 1.5°, 2.0°, 2.5°, and 3°.
- M (9 values): 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, and 0.8.

Now, the equispaced snapshots above have been selected after some calibration to ensure that they are representative of the flow fields in the whole parameter space (namely, RMSE errors of the order of 1%) using HOSVD plus interpolation, as explained in Ref. [10]. A smaller number of snapshots would involve no information enough about the database, while a larger number would lead to spurious redundancies, resulting from just taking too many snapshots.

Now, the HOSVD method described in Section 2 will be applied below in three cases:

1. HOSVD will be applied in Section 4.1 below to a particular snapshot, for fixed values of the parameters M and α. In this case, the distribution of each flow variable can be seen as a third-order tensor. For instance, the pressure distribution can be written as P_{ijk}, where the indexes label the mesh points in the three mesh directions, namely, along the chord, the span, and the wall-normal directions. In addition, in order to check the various possibilities of treating the database, the method will be applied both in the whole computational mesh and block by block, and also both to each flow variable and to all flow variables simultaneously, averaging errors according to Eq. (17) below. Based on this comparison, the remaining calculations in the paper will be made block by block to all flow variables simultaneously, which turns out to be the best strategy for the test problem.
2. HOSVD will be applied in Section 4.2 below to a one-parameter family of snapshots associated with, e.g., varying one of the parameters while maintaining the other one fixed.

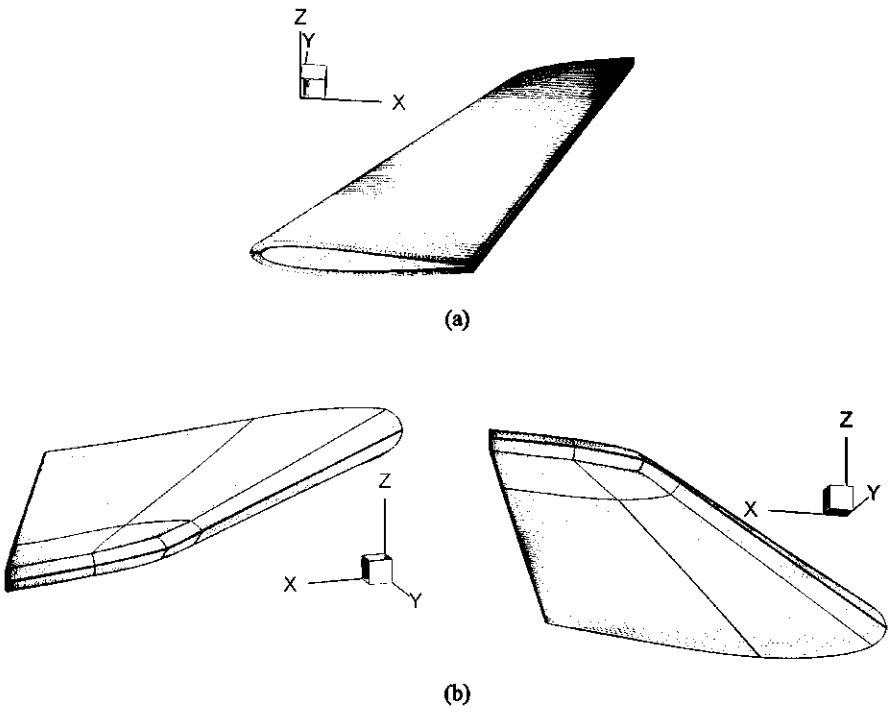

Fig. 1. (a) Wing surface and mesh overviews; (b) illustration of block division on the surface of the wing in the suction and pressure sides.

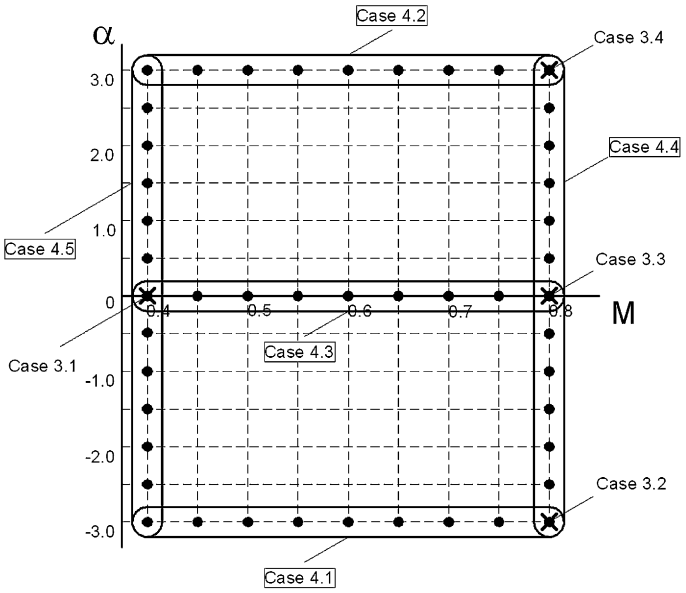

Fig. 2. Parameter space. CFD calculations are made at the intersection points of the plotted (with dashed lines) parameter mesh. Particular values of the parameters to illustrate third-order-tensor compression are indicated with crosses. The various parameter combinations to illustrate fourth-order-tensor compression are plotted with filled circles. The fifth-order case consists of all the CFD calculations together.

If, e.g., only the Mach number is varied, then the pressure distribution can be seen as a fourth-order tensor P_{ijkl}, where the first three indexes label the mesh points, as above, and the fourth index labels the discrete values of the Mach number M.

3. HOSVD will be applied in Section 4.3 to the whole two-parameter database. In this case, the pressure distribution can be seen as a fifth-order tensor P_{ijklm}, where the first three indexes label again the mesh points and the latter two indexes label the discrete values of the parameters M and α.

In each of these cases, the process to obtain the compressed tensor (either in the whole spatial domain simultaneously, or block by block) for a previously specified value of the APEB, the RMSE, or the ME, is as follows:

1. First, the singular values and their respective eigenvectors are computed, using Eq. (5).
2. Secondly, the number of retained modes in each dimension is defined invoking Eq. (11) (or Eq. (13) if compression is made block by block) in the third-order case, and similar expressions for fourth- and fifth-order tensors. Now, when treating the whole database simultaneously, the numbers of modes in the various dimensions of the tensor are selected according to the associated values of the HOSVs defined in (5); namely,

each new mode is selected as that associated with the largest HOSV. In this manner, the requirement on the APEB error (see Eq. (11)) is fulfilled with a minimum number of modes. When applying the method block by block, the numbers of modes in each block will be selected as explained above, requiring that all blocks exhibit the same APEB; this allows an independent treatment of the various blocks, which would be quite convenient to parallelise the method.

3. If the HOSVD modes are to be selected requiring an APEB level, new modes are added until the required APEB level is reached and the procedure ends after calculating the compression factor, using either (9) or (15).
4. If instead HOSVD modes are to be selected according to either ME or RMSE, the tensor is reconstructed with the retained modes, using Eq. (7), either in the whole spatial domain or block by block. For each value of the APEB, the reconstructed tensor is compared, element by element, with the CFD solution obtaining the error tensor error_{ijk}, and the RMSE and the ME are computed using either (10) and (12) or (16). The procedure continues until the required level of either ME or RMSE is reached. Again, the compression factor is calculated using either (9) or (15).

In most cases (except in Section 4.1), the six databases corresponding to the six flow variables mentioned above will be compressed simultaneously. Errors of the combined compression process are defined as follows

$$E = \frac{\sum_{i=1}^{N_{\text{variables}}} E(\text{variable}_i)}{N_{\text{variables}}} \tag{17}$$

where E stands for either the RMSE or the ME and $N_{\text{variables}} = 6$ is the number of variables.

4. Results

The three cases mentioned in Section 3 are now subsequently considered.

4.1. Third-order tensor to compress one snapshot

As indicated in Section 3, the flow variables for each snapshot (namely, for each set of parameter values) are written as a third-order tensor, where the three indexes label discrete values of the three curvilinear coordinates in the spatial domain, namely those along the chord, the span, and the wall-normal directions. Four representative snapshots are selected among the 117 available ones, one at low Mach number at zero angle of attack and three at high Mach number, for three representative values of angle of attack, namely (see Fig. 2):

Case 3.1: $M = 0.4$ and $\alpha = 0°$.
Case 3.2: $M = 0.8$ and $\alpha = -3°$.
Case 3.3: $M = 0.8$ and $\alpha = 0°$.
Case 3.4: $M = 0.8$ and $\alpha = 3°$.

Those third-order tensors will be compressed and results on the APEB, RMSE, and ME will be calculated as explained at the end of Section 3, and plotted vs. the compression factor. But let us first illustrate the advantages of HOSVD as compared to standard SVD. To this end, we consider the snapshots 3.1–3.4 and plot in Fig. 3 the ME vs. the compression factor, compressing the tensor in three different manners: (i) using HOSVD (thick solid lines) and applying standard SVD either to (ii) each section perpendicular to the spanwise direction (thin dashed lines) or (iii) to the matrix that results when each wall-normal/spanwise section is treated as a vector (thin solid lines). As seen in these plots, HOSVD provides

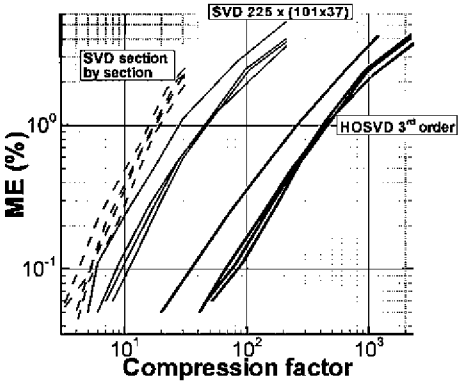

Fig. 3. ME versus the compression factor for the snapshots 3.1–3.4 resulting from compressing the whole wing at a time applying: HOSVD (thick solid line), standard SVD to each of the 101 chordwise/wall-normal sections (thin dashed line), and standard SVD, treating the spanwise and wall-normal directions together (thin solid line).

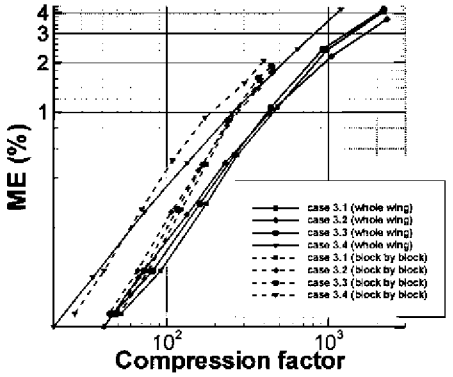

Fig. 4. Mean error vs. the compression factor applying HOSVD block by block (dashed lines) and considering the whole spatial domain at a time (solid lines), for the four cases considered in Section 4.1, as indicated.

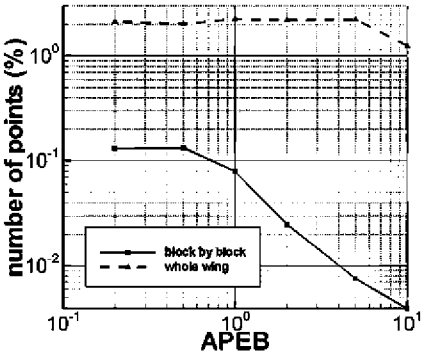

Fig. 5. Number of points (in %) in which the local error is 5 times larger than the required APEB for the flow variable u in test case 3.4, both compressing block by block and treating the whole wing at a time.

compression factors that are roughly ten times larger than those resulting from application of standard SVD.

In order to elucidate whether dividing into blocks leads to better results than treating the whole wing at a time, the ME is plotted in Fig. 4 vs. the compression factor, for the four cases indicated above. This plot shows that dividing into blocks worsens the compression factor, which (as explained by the end of Section 2) means that the various blocks are highly correlated. This must be due to the (approximately) parabolic nature of the aerodynamic flow near the wing along each streamline direction, which makes downstream bocks dependent on upstream blocks. On the other

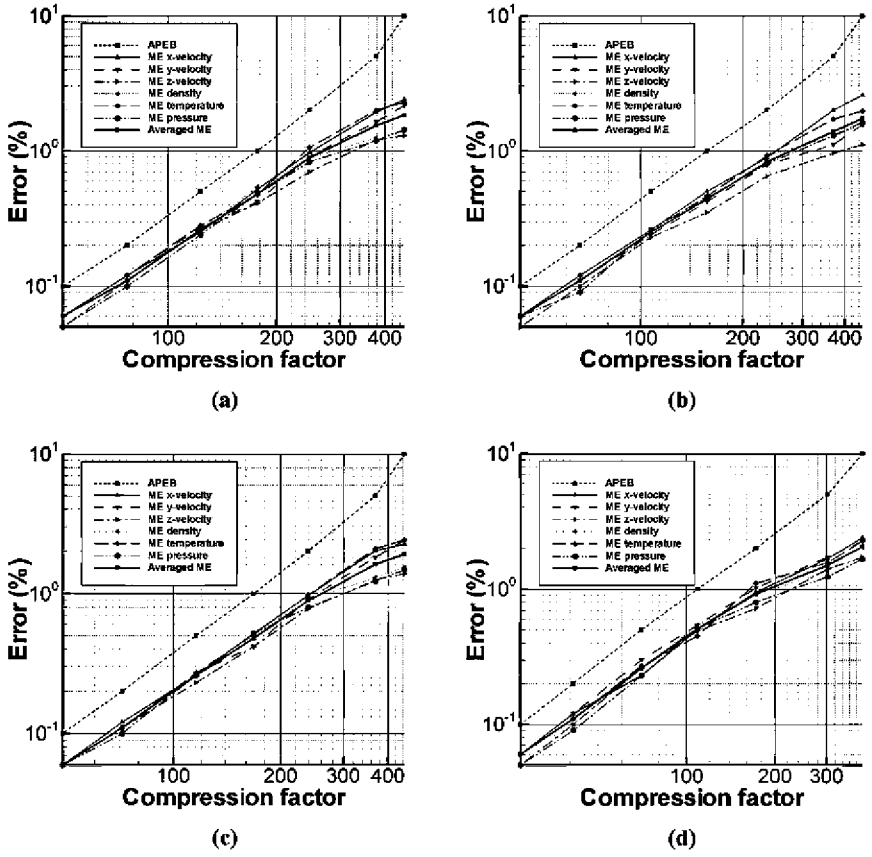

Fig. 6. APEB and ME vs. the compression factor in test cases 3.1 (a), 3.2 (b), 3.3 (c), and 3.4 (d); ME is calculated using Eq. (17) when treating all flow variables simultaneously and using (16) for each flow variable when treating the variables independently.

hand, as also explained at the end of Section 2, the block-by-block approach provides a more homogeneous spatial error distribution. This is illustrated in Fig. 5, where the number of mesh points (in %) in test case 3.4 that exhibit a local error in the variable u larger than 5 times the APEB is plotted vs. the APEB, when compression is made both block by block and treating the whole wing at a time; the remaining test cases and flow variables behave similarly. Note that now the advantage of the block-by-block approach is clear, and compensates the smaller compression factor. Thus, the block-by-block treatment will be followed in the sequel, which is a conservative strategy.

Now, we compare the six flow variables in connection with the compression factor. This is done in Fig. 6, where both the ME for each flow variable (as defined in Eq. (16)) and the averaged ME (as defined in Eq. (17)) are plotted vs. the compression factor in the four cases defined above; the APEB (which is imposed to be the same for all flow variables) is also plotted for reference. Note that all flow variables exhibit a similar behaviour. Thus, for the sake of brevity only the averaged errors for all flow variables will be plotted below.

Comparison between the various errors defined in Section 2 is made in Fig. 7, where the ME (solid line), the RMSE (dashed line), and the APEB (dash-dotted line) are plotted vs. the compression factor, for the four test cases defined above. As can be seen in this figure at, e.g., the ME levels of 0.4% and 1% (which are quite reasonable in aeronautic design applications), the compression factor of the HOSVD is in the ranges 100–200 and 200–300, respectively. In fact, it has been checked that the ME level of 0.4% provides quite good spatial error distributions. As anticipated above, such high compression factors are due to strong (spatial) correlation between the aerodynamic flow at different locations. Also note that using APEB to select the number of modes is a somewhat pessimistic criterion, as suggested by the inequality in Eq. (10), and that using the RMSE to select the numbers of modes is more stringent than using the ME, as anticipated in Eq. (16). For example, plot (b) shows that at the level of 1% errors, APEB, RMSE, and ME provide compression factors of 160, 200, and 280, respectively; note however that the latter two better results are at the expense of a longer computation process. Still, truncating with the RMSE provides a more homogeneous spatial error distribution, as illustrated in Fig. 8, where the number of mesh points (in %) that exhibit a local truncation error larger than 5 times the ME (solid line) and the RMSE (dashed line) is plotted vs. the associated value of the ME and the RMSE, respectively.

4.2. Fourth-order tensor to compress a one parameter family of snapshots

Now, we add a fourth index to label the discrete values of one of the parameters, either the Mach number or the angle of attack, maintaining constant the remaining parameter. Five representative cases will be considered among the 22 possible cases (9 values of M plus 13 values of α), namely (see Fig. 2):

Case 4.1 (9 snapshots): $\alpha = -3^\circ$ and $M = 0.4, 0.45, \ldots,$ and 0.8.
Case 4.2 (9 snapshots): $\alpha = 3^\circ$ and $M = 0.4, 0.45, \ldots,$ and 0.8.
Case 4.3 (9 snapshots): $\alpha = 0^\circ$ and $M = 0.4, 0.45, \ldots,$ and 0.8.
Case 4.4 (13 snapshots): $M = 0.8$ and $\alpha = -3^\circ, -2.5^\circ, \ldots,$ and 3°.
Case 4.5 (13 snapshots): $M = 0.4$ and $\alpha = -3^\circ, -2.5^\circ, \ldots,$ and 3°.

Note that in the first three cases (4.1–4.3), the HOSVD is applied to groups of the 9 snapshots associated with each value of α, while in the last two cases (4.4 and 4.5) the groups contain the 13 snapshots associated with each value of M.

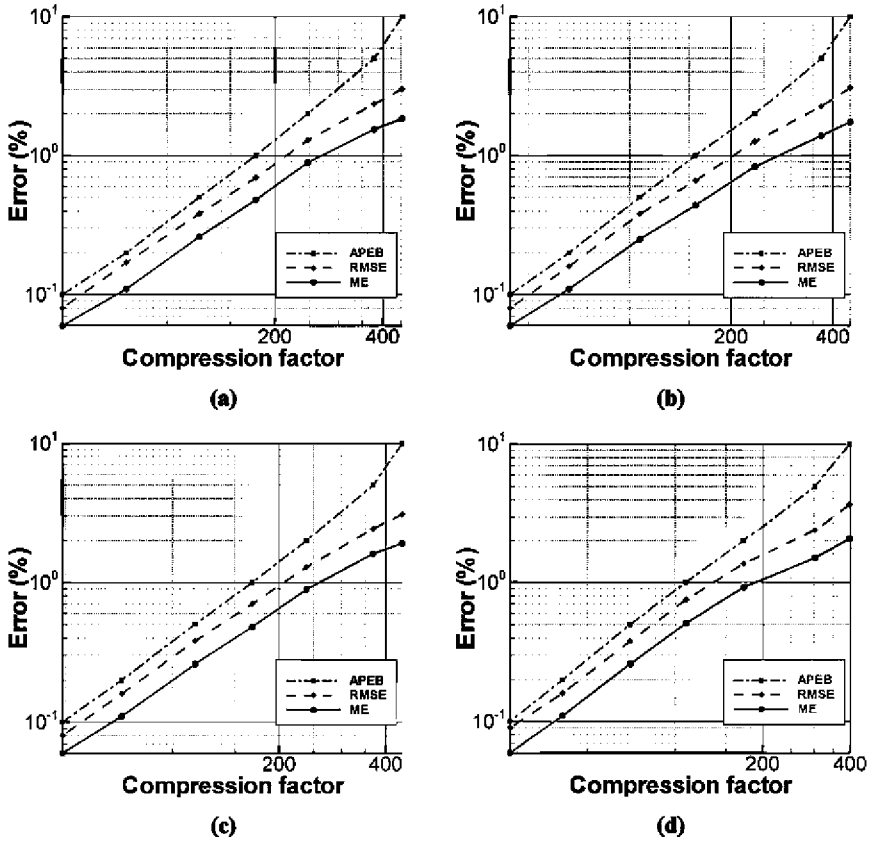

Fig. 7. Averaged ME (solid), RMSE (dashed), and APEB (dash-dotted) vs. the compression factor in the test cases 3.1 (a), 3.2 (b), 3.3 (c), and 3.4 (d).

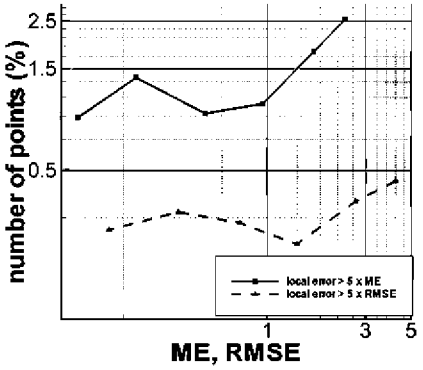

Fig. 8. Number of points (in %) in which the local error is 5 times larger than the ME and the RMSE vs. the associated value of the ME and the RMSE, respectively, for the flow variable u in test case 3.4.
vspace10pt

Results for these five cases are presented in Fig. 9. As in the preceding subsection, the ME, RMSE, and APEB are plotted vs. the compression factor. As can be seen, the compression factor is consistently much higher than in the one snapshot case considered in Section 4.1. For instance, at the ME levels of 0.4% and 1% the compression factor is in the ranges 500–1200 and 1000–3000, respectively. As in Section 4.1, the ME level of 0.4% yields quite good local error distributions. Noting that the amount of information contained in the fourth-order tensors (either 9 or 13 snapshots) is about one order of magnitude larger than in the third-order tensor (only one snapshot), it can be concluded that it is possible to store a set of about ten snapshots using only a slightly larger amount of numbers than that needed to store just one snapshot. The latter result confirms that the aerodynamic information in the database is highly correlated, both in the spatial dimensions in each snapshot and among the various snapshots, as the parameters are varied. Our HOSVD based method just takes advantage of this redundancy to obtain high compression levels.

4.3. Fifth-order tensor to compress the whole database

As indicated in Section 3, here we proceed as in Section 4.1, but adding two additional indexes that label the discrete values of the parameters, which builds a fifth-order tensor. Now, only one case is considered (test case 5.1) that includes at a time the 117 snapshots indicated in Fig. 2. The resulting database contains an amount of information that is one order of magnitude larger than in the fourth-order tensor case and two orders of magnitude larger than in the third-order tensor case. Again, the ME, the RMSE, and APEB results are plotted vs. the compression factor in Fig. 10. Note that the dependence of the errors on the compression factor is similar to that in Fig. 9, except that now the compression factor is an order of magnitude larger (ME levels of 0.4% and 1% lead to compression factors of 3000 and 17,000, respectively).

Now, the comment above in conjunction with our related conclusion in Section 4.2, confirms that the compression factor increases exponentially as the number of (correlated) dimensions of the database increases. This is further illustrated in Fig. 11, where a summary of the results above on the dependence of the ME on the compression factor for all the test cases is presented. As indicated above, the compression factor is multiplied by a factor each time a new parameter is added. Note however that all curves collapse at the ME level of 0.1%, which is due to the fact that this level is of the same order of magnitude as that of the (somewhat random) CFD errors, meaning that flow data is no longer correlated at this level. Therefore, CFD errors must be taken into account when applying this methodology.

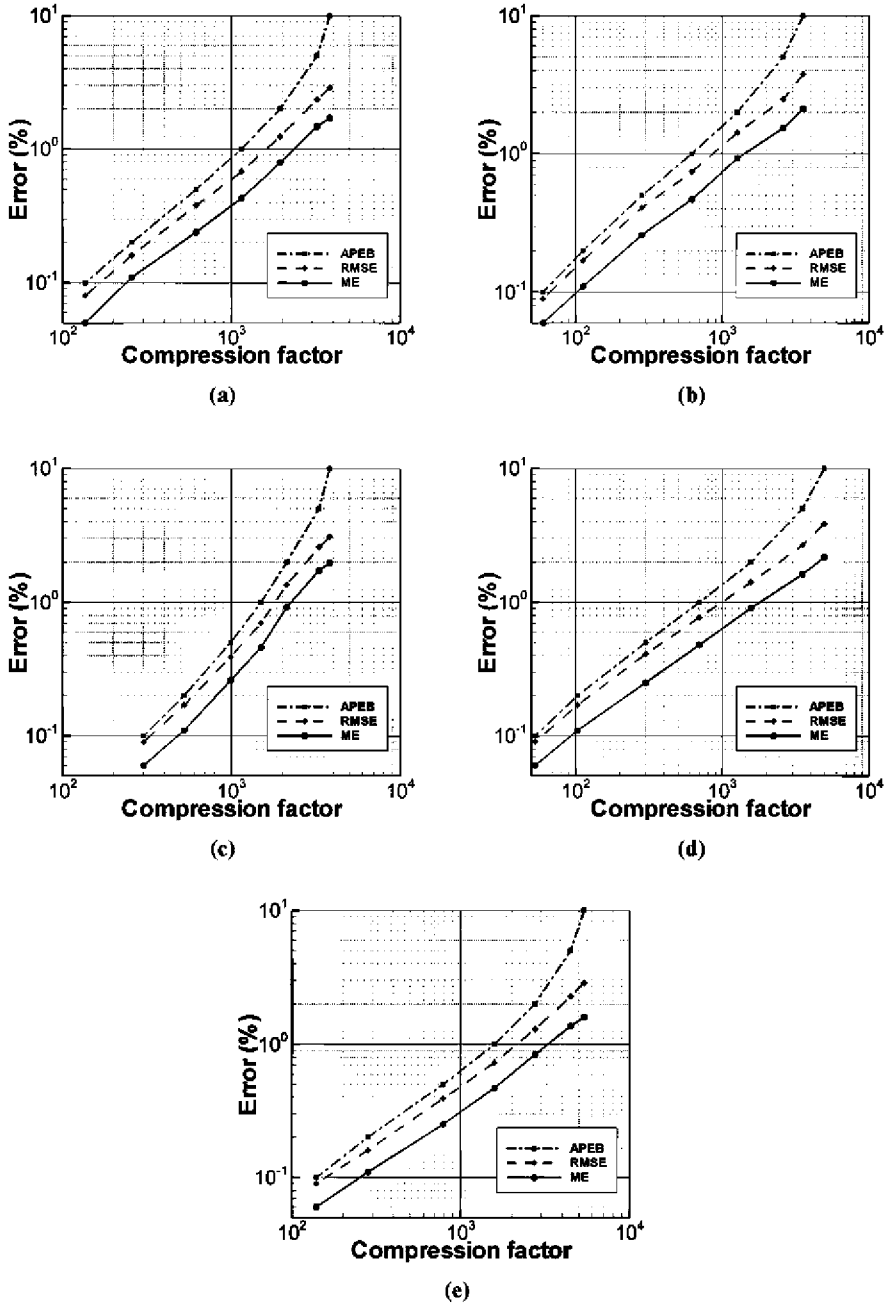

Fig. 9. As in Fig. 7, but for the test cases 4.1 (a), 4.2 (b), 4.3 (c), 4.4 (d), and 4.5 (e).

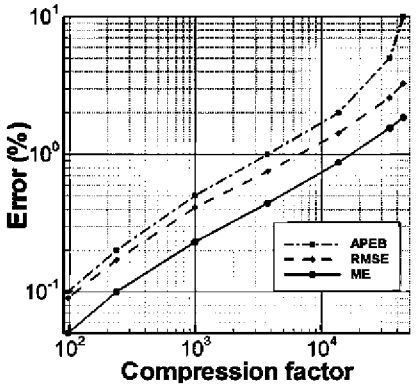

Fig. 10. As in Fig. 7, but for test case 5.1.

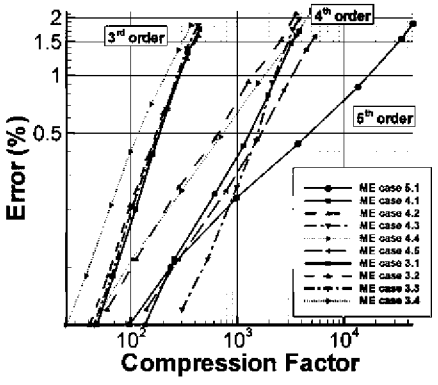

Fig. 11. ME versus compression factor for all test cases.

5. Conclusions

A tool based on HOSVD has been presented that allows for aerodynamic databases compression; application of the tool has been made on a database containing the CFD calculated aerodynamic flow around a wing. The method stores an approximation of the database, taking advantage of the redundancies that are present in these databases as a result of the physical laws (mass, momentum, and energy conservation, and equation of state) that the flow variables obey point wise. HOSVD allows taking into account redundancies in all database dimensions at a time, which makes a difference with more or less direct application of SVD to treat higher-dimensional databases, as explained in Sections 1 and 4 and illustrated in Fig. 3.

The approximation error is measured either as a root mean square error (RMSE) or as a mean error (ME), which can be calculated a posteriori, comparing the original database and its stored version; RMSE is more stringent than ME, as anticipated in Eq. (16) and illustrated in Figs. 7, 9 and 10, but provides a better local error distribution. A (somewhat pessimistic, see Figs. 7, 9 and 10) a priori upper bound of the RMSE, called APEB, is provided by the HOSVD itself. Results have been generally given plotting the RMSE and the ME vs. the compression ratio, defined as the ratio of the sizes of the compressed and the original databases.

Errors can be either estimated a priori or calculated a posteriori. Namely, there are two possible strategies:

- Truncate the HOSVD expansion according to the APEB, which leads to both a smaller compression factor and an also smaller computational effort.
- Iteratively increase the number of HOSVD modes, reconstruct, and compare with the original database until the required error precision is reached. Now both the compression factor and the computational effort are larger.

Physical laws (and thus, redundancies) relate the six flow variables (velocity components, density, temperature, and pressure) among themselves, which means that all variables behave similarly, as illustrated in Fig. 6. The method can be applied either:

- Considering the whole spatial domain simultaneously.
- Dividing the spatial domain into blocks, which has been made in the considered wing example using precisely the same blocks provided by the CFD tool.

The first alternative is to be preferred when the spatial domain is somewhat small. Dividing into blocks is to be preferred when the size of the spatial domain is larger. The second alternative in this case leads to a more homogeneous spatial error distribution (see Fig. 5), and also to a larger compression factor if the various blocks behave in an independent way, because in this case the required number of HOSVD modes is smaller when blocks are treated separately. This latter effect was not present in the wing example considered in this paper (see Fig. 1) because the aerodynamic flow in this case shows a strong spatial correlation. However, such correlation is not always present in aerodynamic flows. For instance, fuselage, wings, horizontal tail planes, and vertical tail plane are expected to behave somewhat independently in full aircraft configurations. Division into blocks in the parameter space could also be advisable when independent aerodynamic behaviours are to be expected at different regions of the parameter space.

Compression factors are generally quite large, provided that the required error bound of the stored approximation is larger than the (somewhat random) errors in the original database. In addition, the compression ratio behaves exponentially with the number of parameters (e.g., in the wing example, the compression factor is multiplied by a factor each time a new parameter is added), provided that correlations occur between the dependence of the aerodynamic flow on the various parameters. However, exponential increase of the compression factor requires both (a) that the errors in the database be not larger than the required compression error and (b) that the database contains a sufficiently large number of snapshots, such that the redundancies associated with physical laws are present in the numerical database. As explained in Section 3, the second requirement can be met after some calibration, ensuring that the number of snapshots be such that HOSVD + interpolation provides results with an accuracy comparable to the accuracy required in the compressed database. The exponential behaviour of the compression factor is quite promising envisaging applications to databases resulting from, e.g.:

- Full aircraft configurations, in which the number of relevant parameters (Mach and Reynolds numbers, angle of attack, yaw angle, and angle defection of, say, five control surfaces) leads to huge databases.
- Shape design of, e.g., an aircraft wing, in which the number of scalar parameters associated with shape deformation can be really large (say, of the order of 100).

Acknowledgements

This research has been partially funded by the Spanish Ministry of Education, under Grants DPI2009-07591 and TRA2007-65699. The authors are indebted to two anonymous referees for many useful comments on an earlier version of the paper that helped to improve presentation of the results.

References

[1] V. da Silva, L.H. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem, SIAM J. Matrix Anal. Appl. 30 (2008) 1084–1127.
[2] L. de Lathauwer, B. de Moor, J. Vandewalle, A multilinear singular value decomposition, SIAM J. Matrix Anal. Appl. 21 (2000) 1253–1278.
[3] L. de Lathauwer, B. de Moor, J. Vandewalle, On the best rank-one and rank-$(R_1, R_2, \ldots, R_N)$ approximation of higher order tensors, between the six flow variables, SIAM J. Matrix Anal. Appl. 21 (2000) 1324–1342.
[4] D. del-Castillo-Negrete, S.P. Hirshman, D.A. Spong, E.F. D'Azevedo, Compression of magnetohydrodynamic simulation data using singular value decomposition, J. Comp. Phys. 222 (2007) 265–286.
[5] J. Edwards, S. Chandra, Comparison of eddy viscosity-transport turbulence models for three-dimensional, shock-separated flowfields, AIAA J. 34 (1996) 756–763.
[6] G.H. Golub, G.T. Van Loan, Matrix Computations, John Hopkins University Press, 1996.
[7] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, SIAM Rev. 51 (2009) 455–500.
[8] C.F. Lee, S.W. Changchien, A data mining approach to database compression, Inform. Syst. Front. 8 (2006) 147–161.
[9] G. Leng, Compression of aircraft aerodynamic database using multivariable Chebyshev polynomials, Adv. Eng. Software 28 (1997) 133–141.
[10] L.S. Lorente, J.M. Vega, A. Velazquez, Generation of aerodynamic databases using high-order singular value decomposition, J. Aircraft 45 (2008) 1779–1788.
[11] Z.B. Miled, L. Huian, O. Bukhres, M. Bem, R. Jones, R. Oppelt, Data compression in a pharmaceutical drug candidate database, Informatica 27 (2003) 213–223.
[12] K.V. Ravi Kanth, D. Agrawal, A. El Abbadi, A. Singh, Dimensionality reduction for similarity searching in dynamic databases, Comput. Vision Image Underst. 75 (1999) 59–72.
[13] M. Sakalli, H. Yan, A. Fu, A region-based scheme using RKLT and predictive classified vector quantization, Comput. Vision Image Underst. 75 (1999) 269–280.
[14] P.R. Spalart, S.R. Allmaras, A one-equation turbulence model for aerodynamic flows, AIAA Paper 92-0439.
[15] J.C. Tannehill, D.A. Anderson, R.H. Pletcher, Computational Fluid Mechanics and Heat Transfer, Taylor & Francis, 1997.
[16] J.E. Tougas, R.J. Spiteri, Two uses for updating the partial singular value decomposition in latent semantic indexing, Appl. Numer. Math. 58 (2008) 499–510.
[17] J. Ye, Generalized low rank approximation of matrices, Mach. Learn. 61 (2005) 167–191.