Applied Mathematical Modelling 37 (2013) 8930–8945

Contents lists available at SciVerse ScienceDirect

Applied Mathematical Modelling

journal homepage: www.elsevier.com/locate/apm

Reduced order modelling for unsteady fluid flow using proper orthogonal decomposition and radial basis functions

S. Walton *, O. Hassan, K. Morgan

College of Engineering, Swansea University, Swansea SA2 8PP, Wales, UK

ARTICLE INFO

Article history:
Received 12 June 2012
Received in revised form 29 January 2013
Accepted 19 April 2013
Available online 2 May 2013

Keywords:
Reduced order modelling
Proper orthogonal decomposition
Radial basis functions
Unsteady fluid flow
Interpolation
Parameterised problems

ABSTRACT

A technique is presented for interpolating unsteady solutions to parameterised fluid flow problems, using a combination of proper orthogonal decomposition and radial basis functions. The technique is validated by considering simulations involving three dimensional unsteady compressible inviscid flow over an oscillating ONERA M6 wing. It is demonstrated that the approach can result in a large reduction in the cpu time required to find solutions at new parameter values, without a significant loss in accuracy.

1. Introduction

Fluid dynamics is a key technology in many practical engineering design and analysis problems and the designer frequently turns to methods of computational fluid dynamics to aid the design process. However, accurate unsteady simulations, involving a given geometry and a number of flow parameters [1], often require so much cpu time to sweep the parameter space that they have limited impact on the design cycle. To be an effective practical addition to the design process, computational simulation time must be shortened and reduced order modelling has been shown to be an effective method for attaining this goal [2,3].

The fundamental aim of reduced order modelling is to reduce the number of degrees of freedom (DoF) necessary to produce the required information to an acceptable level of accuracy [4]. The vast majority of reduced order models (ROMs) achieve this result by projecting the governing equations onto a series of spatial modes [5]. The number of unknowns then becomes equal to the number of these modes, with the modes being calculated by employing equation driven or data driven methods. In equation driven methods, such as nonlinear normal modes [6], the modes are constructed by applying mathematical reduction techniques directly to the equations themselves. Data driven methods, such as proper orthogonal decomposition (POD), use sample stored calculated solutions or experimental data to generate the spatial modes [1]. It is the data driven approach that is adopted here.

POD methods have been widely employed for the analysis of problems involving fluid flow [7–9]. The objective is to identify an optimal coordinate system to represent an ensemble of snapshots, i.e., a sample of solution fields, by removing redundant information [10]. In practice, this amounts to finding the smallest possible set of basis functions which can be employed to reconstruct each snapshot of the ensemble [3,11]. The set of basis functions forms a series of POD modes, each carrying a

* Corresponding author.
E-mail addresses: s.p.walton@swansea.ac.uk (S. Walton), o.hassan@swansea.ac.uk (O. Hassan), k.morgan@swansea.ac.uk (K. Morgan).

0307-904X © 2013 Elsevier Inc. Open access under CC BY license.
http://dx.doi.org/10.1016/j.apm.2013.04.025

percentage of the flow energy [7]. An important issue is that the basis, and hence the model, will only contain information that is present in the ensemble of snapshots [10]. Intelligent selection of the snapshots is, therefore, the key to constructing a successful POD basis. In practice, the POD modes are calculated using a set of snapshots made up of time dependent discretised flow field state vectors. Using these snapshots, the POD modes are obtained by performing a singular value decomposition (SVD) of the matrix whose columns are the snapshots [1,7]. Reliable algorithms have been developed to compute the SVD and these can be readily applied to POD applications [12].

Galerkin projection is the most frequently used method for obtaining a reduced order model from a POD for unsteady problems [3]. Essentially, the exact time–dependent governing equations are directly projected onto the POD basis using an inner product [2]. As the POD modes are mutually orthogonal, the unknowns then become a set of time dependent coefficients, with the number of coefficients being equal to the number of POD modes used. Projection techniques of this nature have been successfully applied to a number of applications, from turbomachinery [13,14] to complete aircraft configurations [15]. Computing the POD mode coefficients requires the evaluation of integrals over the entire computational domain and this can be a computationally expensive task for large problems [2]. This cost is often reduced by eliminating some of the higher order POD modes, which carry small energy contributions. Stability problems have been encountered when solving for the time dependent coefficients in this manner [4,3,16,17]. Further to this the snapshots may contain large concentrations of errors due to discontinuities in the geometry (e.g. at corners) which would be reproduced in the ROM, and the full order CFD model might contain nonphysical stabilisation terms which would need to be accounted for in the ROM [2]. This provides a key motivation for the development of the alternative technique that is presented here.

The approach introduced by My–Ha et al. [18] for steady parameterised fluid problems does not require the evaluation of integrals. The snapshots are sampled throughout a parameter space of interest and are used to calculate the POD basis. The POD mode coefficients are determined for each snapshot and interpolated, or extrapolated, to enable the construction of a new solution at different parameter values. Although a similar technique was successfully used by Qamar and Sanghi [19], the approach does not appear to have been widely adopted as an alternative to projection techniques. A possible limitation of the method is that it can rapidly become expensive as the number of parameters of interest increases. This could be a significant issue in many practical problems, such as shape optimisation, where the number of parameters involved can be very large [20]. In addition, the method, in its current form, is only applicable to the solution of steady state problems. In this paper, we attempt to address these two limitations of this type of approach. To overcome the issue of interpolation cost, we employ radial basis function (RBF) interpolation [21], where the interpolation expense does not increase significantly with the number of dimensions. RBF interpolation can be readily implemented, as it does not require a triangulation to define the connectivity between the data points [22]. It has been applied to many fields, including the solution of partial differential equations [23]. For application to unsteady flow problems, we apply a second level of model reduction to the time dependent coefficients themselves. This second level of model reduction has not been attempted elsewhere. The method is validated by demonstrating the results that can be produced for the simulation of a number of different problems involving three dimensional inviscid flow over an oscillating ONERA M6 wing.

2. Theory

2.1. The full order model

We are interested in solving three dimensional unsteady inviscid flow problems with moving boundary components. For such problems, a convenient starting point is an arbitrary Lagrangian–Eulerian (ALE) formulation for the compressible Euler equations. In such a formulation, the conservation statement for flow in a domain $\Omega(t)$, with surface $\partial\Omega(t)$, may be expressed, relative to a cartesian coordinate system (x_1, x_2, x_3), in the integral dimensionless form

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega(t)} \boldsymbol{U}\mathrm{d}\boldsymbol{x} + \int_{\partial\Omega(t)} (\boldsymbol{F}_j - v_j\boldsymbol{U})n_j\mathrm{d}\boldsymbol{x} = 0 \quad j = 1,2,3 \tag{1}$$

where the summation convention is employed. Here, the outward normal vector to $\partial\Omega(t)$ is denoted by $\boldsymbol{n} = (n_1, n_2, n_3)$ and the velocity of $\partial\Omega(t)$ is $\boldsymbol{v} = (v_1, v_2, v_3)$. The unknown vector, $\boldsymbol{U}$, and the inviscid flux vectors, $\boldsymbol{F}_j$, are written, in dimensionless form, as

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho u_1 \\ \rho u_2 \\ \rho u_3 \\ \rho\varepsilon \end{bmatrix} \quad \boldsymbol{F}_j = \begin{bmatrix} \rho u_j \\ \rho u_1 u_j + p\delta_{1j} \\ \rho u_2 u_j + p\delta_{2j} \\ \rho u_3 u_j + p\delta_{3j} \\ (\rho\varepsilon + p)u_j \end{bmatrix} \tag{2}$$

where ρ is the fluid density, $\boldsymbol{u} = (u_1, u_2, u_3)$ is the fluid velocity, p is the pressure and ε is the specific total energy. The system is closed by using the perfect gas equation of state.

For the full order model (FOM), the spatial domain $\Omega(t)$ is discretised using an unstructured tetrahedral mesh and Eq. (1) is approximated using a cell vertex finite volume method in space. Stabilisation and discontinuity capturing is achieved by the explicit addition of artificial viscosity. A fully implicit three level second order method is adopted for the time discretisation. At each time step, the implicit equation system is solved by explicit iteration, with multigrid acceleration [24,25]. For regions with discontinuities a first-order harmonic term is added, where a pressure switch is used to ensure the term is only significant in regions of strong pressure gradients [26]. The JST scheme [27] is applied for stabilization, more details on the solver can be found in [26]. For the examples included here, the explicit solution process continues within each time step until the residual has been reduced by five orders of magnitude.

The examples that will be considered involve the periodic oscillation of a wing, with the FOM solver used to simulate ten complete oscillation cycles. The results of the final cycle are used to form a snapshot for one prescribed set, $\boldsymbol{\alpha}$, of the flow parameters. Different values of the parameter sets lead to different snapshots. These snapshots will be used to create a ROM that can then be employed to efficiently produce the flow solution for different parameter values. For illustration here, it is only the pressure values of the final cycle that are used to form the snapshots. Any ROM built in this way will only be valid inside the flow regime described by the snapshots and the examples have been selected so as to ensure that the flow regime does not change within the sampled parameter space.

2.2. The reduced order model using POD

Suppose that, for a certain problem, M snapshots are computed, using the FOM for M different values, $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_M$, of the flow parameter set on a mesh with D nodes. If each snapshot involves N time steps, the output from the FOM is a total of $N \times M$ solution vectors, $\boldsymbol{p}^n(\boldsymbol{\alpha}_k)$, where $k = 1, \ldots, M$ and $n = 1, \ldots, N$. These solution vectors can be used to form the columns of a snapshot matrix

$$\boldsymbol{P} = \begin{bmatrix} p_1^1(\boldsymbol{\alpha}_1) & p_1^2(\boldsymbol{\alpha}_1) & \cdots & p_1^N(\boldsymbol{\alpha}_1) & p_1^1(\boldsymbol{\alpha}_2) & \cdots & p_1^N(\boldsymbol{\alpha}_M) \\ \vdots & \vdots & \ldots & \vdots & \vdots & \ldots & \vdots \\ p_D^1(\boldsymbol{\alpha}_1) & p_D^2(\boldsymbol{\alpha}_1) & \cdots & p_D^N(\boldsymbol{\alpha}_1) & p_D^1(\boldsymbol{\alpha}_2) & \cdots & p_D^N(\boldsymbol{\alpha}_M) \end{bmatrix} \tag{3}$$

where $p_I^n(\boldsymbol{\alpha}_k)$ denotes component I of the vector $\boldsymbol{p}^n(\boldsymbol{\alpha}_k)$. This matrix can be factorised, using SVD, as

$$\boldsymbol{P} = \boldsymbol{\Phi}\boldsymbol{\Sigma}\boldsymbol{V}^* \tag{4}$$

where the columns of $\boldsymbol{\Phi}$, denoted by $\boldsymbol{\Phi}_j$ for $j = 1, \ldots, N \times M$, are the left singular vectors of $\boldsymbol{P}$ and these will be used as the POD modes. In addition, $\boldsymbol{\Sigma}$ is the diagonal matrix containing the singular values of $\boldsymbol{P}$, which indicate the relative importance of each mode, and the columns of $\boldsymbol{V}^*$ are the right singular vectors of $\boldsymbol{P}$. The vectors $\boldsymbol{\Phi}_j$ are mutually orthogonal and can be regarded as a basis for an $N \times M$ dimensional space in which each element represents a different solution vector for the fluid problem under consideration. It follows that each column of $\boldsymbol{P}$ can be reconstructed, using the POD modes, as

$$\boldsymbol{p}^n(\boldsymbol{\alpha}_k) = \sum_{j=1}^{N\times M} T_j^n(\boldsymbol{\alpha}_k)\boldsymbol{\Phi}_j \tag{5}$$

For each set of parameter coordinates, the vector $\mathbf{T}^n(\boldsymbol{\alpha}_k)$ of coefficients in this reconstruction can be viewed as a path through the coordinate system $\boldsymbol{\Phi}$. The goal is to predict a path that the system takes through $\boldsymbol{\Phi}$ for a set of parameter values not included in the original sampling. This is achieved by considering each member of $\boldsymbol{\Phi}$ in turn and creating new $N \times M$ snapshot matrices, $\boldsymbol{S}_j$, for $1 \leqslant j \leqslant N \times M$, where

$$\boldsymbol{S}_j = \begin{bmatrix} T_j^1(\boldsymbol{\alpha}_1) & \cdots & T_j^1(\boldsymbol{\alpha}_M) \\ \vdots & \ldots & \vdots \\ T_j^N(\boldsymbol{\alpha}_1) & \cdots & T_j^N(\boldsymbol{\alpha}_M) \end{bmatrix} \tag{6}$$

Each column in $\boldsymbol{S}_j$ consists of the values of the coefficients, at different times, for a given POD mode, $\boldsymbol{\Phi}_j$, for a given snapshot. A SVD of $\boldsymbol{S}_j$ produces the matrix $\boldsymbol{\Psi}^j$ of left singular vectors, whose columns $\boldsymbol{\Psi}_i^j$, where $i = 1, \ldots, M$, will be used as POD modes. Again, each matrix $\boldsymbol{\Psi}^j$ can be viewed as a coordinate system, where a point represents the path taken by a given set of parameters along a given axis of $\boldsymbol{\Phi}$. These paths can be reconstructed using

$$\boldsymbol{T}_j(\boldsymbol{\alpha}_k) = \sum_{i=1}^{M} Q_i^j(\boldsymbol{\alpha}_k)\boldsymbol{\Psi}_i^j \tag{7}$$

where $\boldsymbol{T}_j$ is the vector whose nth component is T_j^n. Values of $\boldsymbol{Q}^j(\boldsymbol{\alpha}_k)$ can be readily calculated, when the POD modes are all generated.

When this process is completed, the solution vector $\boldsymbol{p}^n(\boldsymbol{\alpha})$, at a new set of parameter coordinates $\boldsymbol{\alpha}$, can then be determined. Interpolation is required to find the values of $\boldsymbol{Q}^j(\boldsymbol{\alpha})$ and these interpolated values may be used to obtain $\boldsymbol{T}_j(\boldsymbol{\alpha})$, as

$$\boldsymbol{T}_j(\boldsymbol{\alpha}) = \sum_{i=1}^{M} Q_i^j(\boldsymbol{\alpha})\boldsymbol{\Psi}_i^j \tag{8}$$

The ROM approximation to the solution vector follows as

$$\boldsymbol{p}^n(\boldsymbol{\alpha}) = \sum_{j=1}^{N\times M} T_j^n(\boldsymbol{\alpha})\boldsymbol{\Phi}_j \tag{9}$$

An initial example is included at this stage to demonstrate the basic validity of the approach that has been described.

2.3. Initial validation

Example 1 involves inviscid flow over an oscillating ONERA M6 wing at a free stream Mach number of 0.2. In the dimensionless coordinates employed here, the wing chord is 10 units and the wing span is 14 units. The wing tip follows a prescribed sinusoidal pitch oscillation, with a maximum amplitude of ± 3 degrees, and the root of the wing is held fixed. The amplitude of the pitch oscillation is assumed to vary linearly between the fixed root and the tip. For this example, the only flow parameter that is varied is the frequency, *f*, of the oscillation. An initial mesh with 44 573 nodes was generated and, to allow for the complete movement of the wing, a further 31 meshes were obtained from this mesh by mesh movement. This was accomplished using the Delaunay graph concept [28], resulting in the same nodal connectivities on each mesh. The simulations were performed using 32 time steps per cycle, with each mesh representing the geometrical configuration at one time level. With this approach, oscillations of different frequencies could be simulated by simply altering the size of the physical time interval between successive meshes. Five FOM simulations were performed, with the dimensionless frequency values employed, designated *f*1 to *f*5, and the corresponding time step sizes, shown in Table 1. To illustrate the effectiveness of the proposed process, the solutions obtained with the frequencies $f1, f2, f4$ and *f*5 were used to create snapshots and the ROM was then utilised to predict the solution at the frequency *f*3. The lift polar for each of these snapshots is shown in Fig. 2.

Table 1
Frequencies and time step sizes used in Example 1.

Identifier	Frequency	Time step size
*f*1	0.1563	0.2
*f*2	7.813×10^{-2}	0.4
*f*3	5.208×10^{-2}	0.6
*f*4	3.906×10^{-2}	0.8
*f*5	3.125×10^{-2}	1.0

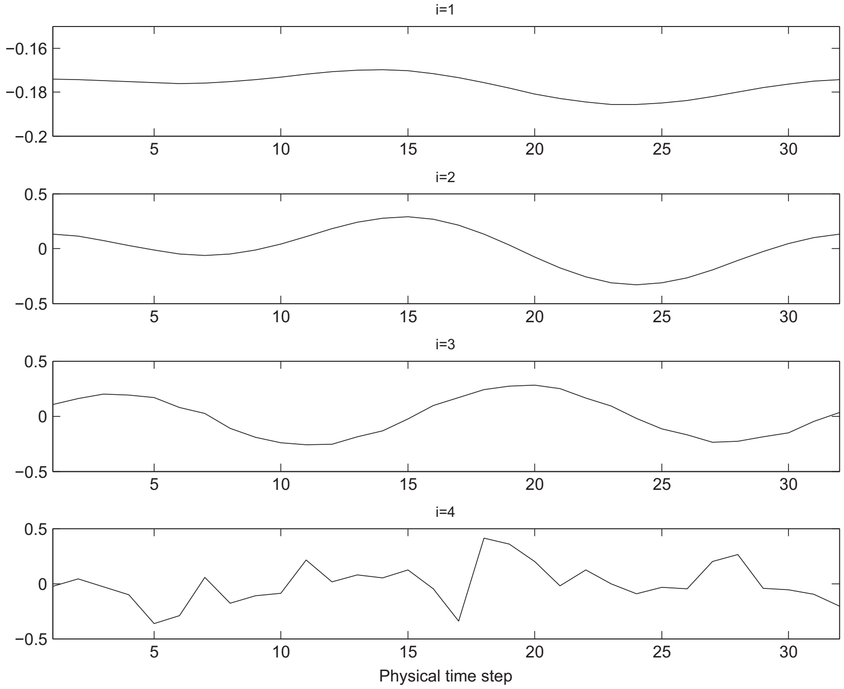

Fig. 1. Temporal modes $\boldsymbol{\Psi}_i^1$, for $i = 1, 2, 3, 4$, for Example 1.

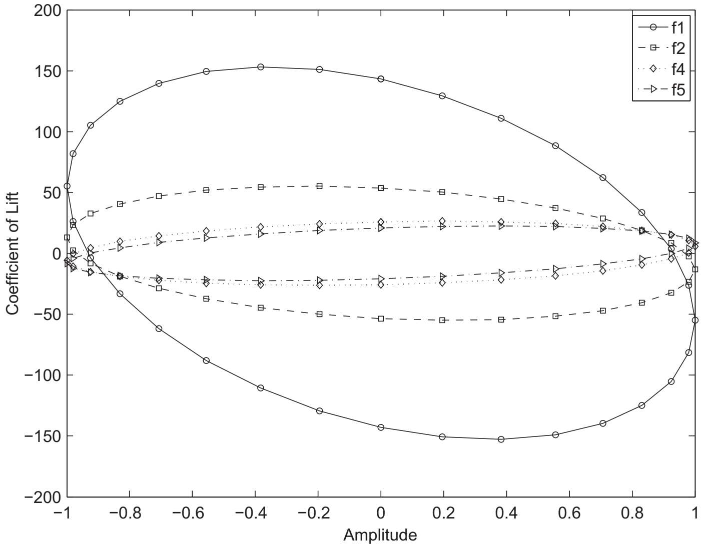

Fig. 2. The lift polar for each snapshot employed in Example 1.

The snapshot pressure fields were used to create the snapshot matrix $\boldsymbol{P}$ of Eq. (3) and the POD modes, $\boldsymbol{\Phi}_j$, where $j = 1, \ldots, 32 \times 4$, were generated from this matrix. The first four modes on the initial mesh, obtained in this fashion, are plotted on the upper surface of the wing in Fig. 3. A set of coefficients was calculated, for each mode for each snapshot, and Fig. 4 shows the variation in time of the coefficient of the first mode, $\boldsymbol{\Phi}_1$, for each snapshot. The time variation of the first spatial modes are used to generate the snapshot matrix $\boldsymbol{S}_1$, as shown in Eq. (6). The POD modes $\boldsymbol{\Psi}_i^1$, where $i = 1, 2, 3, 4$ are obtained

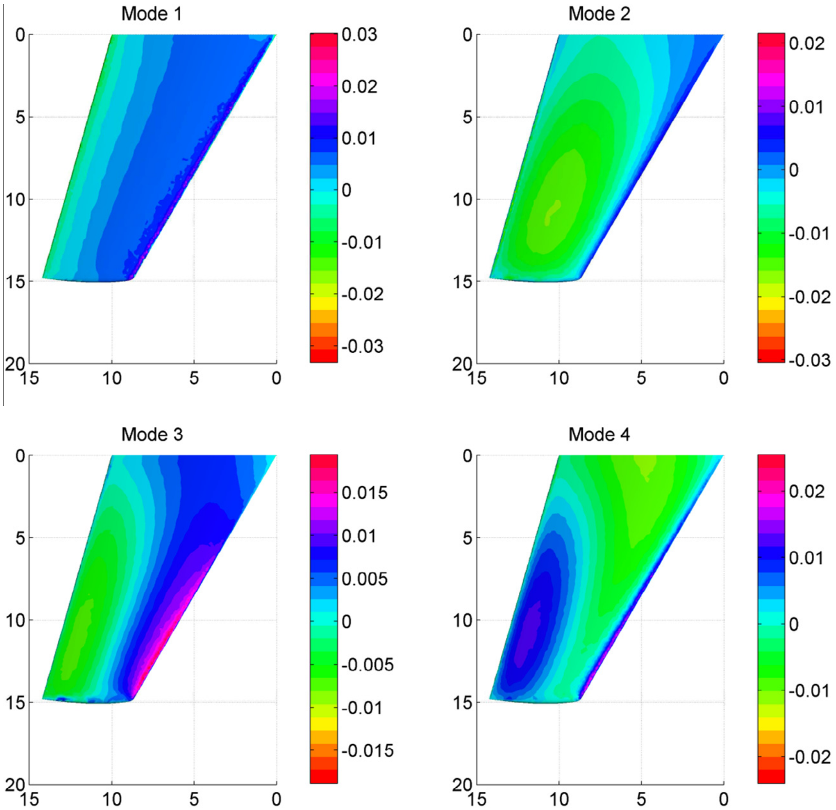

Fig. 3. The first four $\boldsymbol{\Phi}_j$ modes on the upper surface of the wing for Example 1.

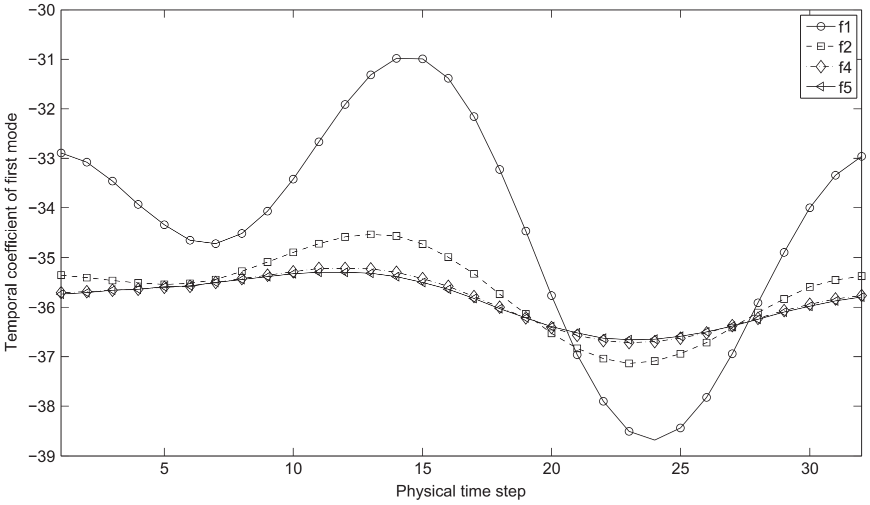

Fig. 4. Time variation of the coefficient of the first mode for each snapshot for Example 1.

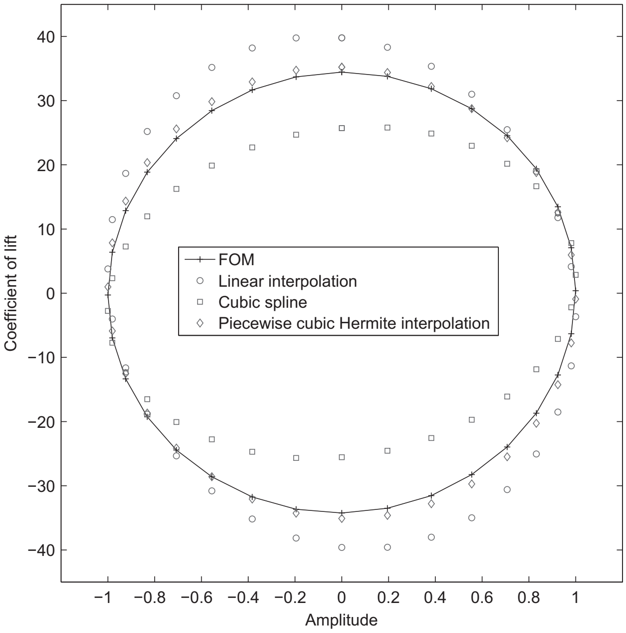

Fig. 5. Comparison between the lift polars obtained using the ROMs constructed for Example 1 with that obtained from the FOM.

by applying SVD to this matrix. These temporal modes are plotted in Fig. 1. A coefficient is calculated for each of these modes, such that a linear combination of the modes reconstructs one of the snapshots in Fig. 4, as shown in Eq. (7). These coefficients are a function of the flow parameter, which in this case is the frequency of the oscillation.

One dimensional linear, cubic spline and Hermite interpolation were used to determine the values of the coefficients in the model appropriate for the frequency $f3$. This was accomplished using the MATLAB [29] function interp1. The resulting ROMs were then used to predict the lift polar for the frequency $f3$ and the predicted lift polars are compared to that obtained from the FOM solution for this frequency in Fig. 5. The pressure fields on the upper surface of the wing, at the 15th time step, predicted by these ROMs are compared to the distribution obtained from the FOM at this time in Fig. 6.

The root mean square error in the pressure, Er_p, and in the lift coefficient, Er_{C_ℓ} were determined and expressed as a percentage of the range of p and C_ℓ respectively. The results enabled a quantitative comparison to be made between the performance of the different methods, with the use of piecewise cubic Hermite interpolation producing the lowest values of $Er_p = 0.2242\%$ and $Er_{C_\ell} = 1.5505\%$.

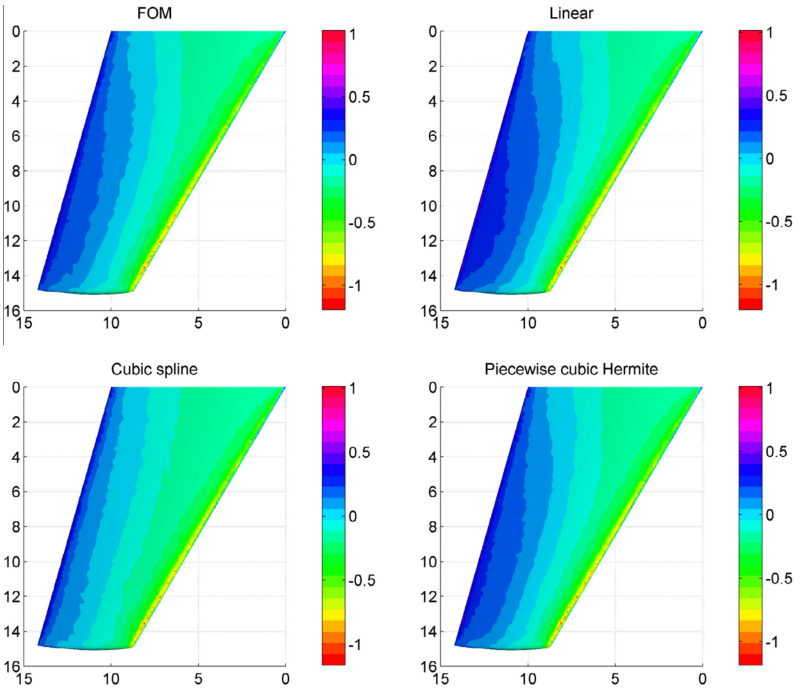

Fig. 6. The pressure fields on the upper surface of the wing, at a prescribed time, computed for Example 1 with the FOM and predicted by the different ROMs.

All ROM calculations reported here were performed on a standard desktop machine, with an Intel Core i5-2500 running at 3.30 GHz, while the FOM calculations used a cluster of 4 AMD Opteron 240 processors. The typical cpu time for a single FOM simulation was 13 500 s. On average, the total cpu time used to complete all the tasks in the ROM for this example was 7 s, which includes generating the modes and calculating the coefficients. Subsequent predictions of the solution at different values of the frequency have negligible cpu cost, since the modes and coefficients are stored. To put this into perspective, plotting the three dimensional solutions involves a larger cpu cost than generating the solutions using the ROM.

3. A note on POD Interpolation type techniques

The vast majority of POD based ROMs rely on first generating the POD modes using sample snapshots from a FOM then projecting these modes onto the governing equations using Galerkin projection. As well as the previously discussed problems with doing this it is also known that a particular set of snapshots may result in POD modes which are only appropriate in a limited parameter space. For example Lieu et al. [15] applied Galerkin projection to a complete F-16 aircraft model which depended only on the angle of attack and free-stream Mach number M_∞. Snapshots were taken from an unsteady finite element full order CFD model by inducing vibrations in the system at a particular M_∞. This resulted in a POD basis which can only be applied to a ROM of a flow at that specific value of M_∞. When considering applying these methods to optimisation/design type problems where the parameter space is to be explored this poses a serious problem. Lieu et al. [15] approached the problem by first performing a subspace angle interpolation of the POD modes then projecting these on the governing equations. To avoid this problem and others introduced by projection methods the strategy adopted in this work is to treat POD as a basis for interpolation.

POD coefficient interpolation type techniques, as introduced by Ly and Tran [11], have not been widely adopted as an alternative to projection type techniques. This family of techniques is discussed elsewhere [30,31] where it is stated that the applicability of POD interpolation techniques to unsteady problems is unclear. The results presented here show that they are clearly applicable. When all POD modes are retained, these techniques are exactly equivalent to element wise interpolation of the solution field. To see this, consider the simple case of two snapshots in a one dimensional parameter space at coordinates α_1 and α_2 where

$$\boldsymbol{T}_l(\alpha_1) = \sum_{i=1}^{M} Q_i^l(\alpha_1)\boldsymbol{\Psi}_i^l \tag{10}$$

and

$$\boldsymbol{T}_l(\alpha_2) = \sum_{i=1}^{M} Q_i^l(\alpha_2)\boldsymbol{\Psi}_i^l \tag{11}$$

Using linear interpolation $Q_i^l(\alpha)$ can be obtained for any α [31]

$$Q_i^l(\alpha) = Q_i^l(\alpha_1) + \frac{Q_i^l(\alpha_1) - Q_i^l(\alpha_2)}{\alpha_1 - \alpha_2}(\alpha - \alpha_1) \tag{12}$$

for $i = 1 \ldots 2$. Since

$$\boldsymbol{T}_l(\alpha) = \sum_{i=1}^{2} Q_i^l(\alpha)\boldsymbol{\Psi}_i^l \tag{13}$$

then substituting from Eqs. (10)–(12), and applying simple summation rules Eq. (13) becomes

$$\boldsymbol{T}_l(\alpha) = \boldsymbol{T}_l(\alpha_1) + \frac{\alpha - \alpha_1}{\alpha_1 - \alpha_2}(\boldsymbol{T}_l(\alpha_2) - \boldsymbol{T}_l(\alpha_1)) \tag{14}$$

This is simply element by element interpolation. The same finding was discussed by Bouhoubeiny and Druault [32], who applied a similar technique to interpolate experimental data in time. This proof can be generalised to higher numbers of dimensions and other interpolation techniques. It equally applies to the unsteady version of the technique presented here and to the steady versions presented and discussed elsewhere. The implications of this result will be discussed in the conclusion.

4. Computational effectiveness of the ROM

The ROM has been developed with the objective of reducing the cost associated with using a computational simulation method as part of a design or optimisation process. The results presented for Example 1 illustrate the potential of the technique, but it is worth considering how this technique might be employed within the design process and how the resulting cpu costs might be estimated. Since the technique is based upon interpolation, it would be naive to think that satisfactory accuracy could be achieved with the same number of snapshots for every example. The accuracy that can be achieved will depend upon the number of snapshots and also upon the location of the snapshots in parameter space. This means that validation will be necessary for each new example. Fortunately, however, this is actually quite a simple process. As part of the normal procedure, the parameter space of interest will be sampled using the FOM and these samples are then available for use as snapshots for producing a ROM. A series of ROMs can be created by simply omitting each snapshot, in turn, during the ROM generation process and measuring the errors that are obtained when using the ROM to predict the omitted solution. If the error calculated in this manner is too large, more FOM solutions can be calculated and more snapshots generated. These additional FOM solutions can be targeted towards those areas of the parameter space in which the errors were originally found to be the largest.

To investigate the cpu costs involved in validating the ROM in this way, suppose that the number of solutions required in a particular parameter space, for a particular design cycle, is M_T and suppose that the cost of a single FOM evaluation is C_{FOM}, expressed as seconds of wall clock time. It follows that the total cpu cost that would be incurred by using only the FOM in the design cycle would be

$$C_{TotalFOM} = M_T \times C_{FOM}. \tag{15}$$

The total cpu cost when the ROM is used to perform the same task can be expressed as

$$C_{TotalROM} = M_T \times C_{ROM} + C_{Val} + C_{Build} + M \times C_{FOM} \tag{16}$$

where C_{Build} is the cost of building the ROM, but not including evaluation of the snapshots, C_{Val} is the cost of validating the ROM, C_{ROM} is the cost of evaluating a solution of the ROM and $M \times C_{FOM}$ represents the cost of calculating the M snapshots. The cost of validating the ROM, using the technique that has been described, can be expressed as

$$C_{Val} = M \times (C_{Build} + C_{ROM}). \tag{17}$$

It follows that the requirement for the process to be worthwhile, in terms of cpu saving, is that

$$C_{TotalFOM} > C_{TotalROM} \tag{18}$$

and, using the above expressions, this requires that

$$M_T > \frac{M \times (C_{Build} + C_{ROM} + C_{FOM}) + C_{Build}}{C_{FOM} - C_{ROM}}. \tag{19}$$

Since it is reasonable to assume that

$$C_{Build} \ll M \times (C_{Build} + C_{ROM} + C_{FOM}), \tag{20}$$

the requirement of Eq. (19) may be expressed in the form

$$\frac{M_T - M}{M} > \beta - 1, \tag{21}$$

where

$$\beta = \frac{C_{FOM} + C_{Build} + C_{ROM}}{C_{FOM} - C_{ROM}}. \tag{22}$$

The minimum number of new solutions which would need to be calculated using the ROM, so as to result in a cpu saving, can then be determined, in terms of β, as

$$(M_T - M)_{min} = M(\beta - 1). \tag{23}$$

When $C_{ROM} + C_{Build}$ is much smaller than C_{FOM}, the value of β will be very close to unity. In this case, it would always be worth using this technique provided a ROM of satisfactory accuracy can be achieved for $M < M_T$. In the remainder of this paper, the value of $(M_T - M)_{min}$ will be used as an indication of the cpu saving that can be achieved by using the ROM.

As discussed previously POD based interpolation is exactly the same as node and time independent interpolation. The process results in less interpolation operations than nodalwise interpolation. The number of interpolation operations undertaken without using POD would be

$$D \times N.$$

Using POD the number of interpolation operations is

$$M_T \times N.$$

The use of

$$M_T \times N$$

$$M_T < D,$$

will result in a computational saving using POD based interpolation. This result is independent of the interpolation method implemented.

It may first seem that this condition will almost always be fulfilled since, in industrial applications, D is very large. However, since the interpolation is of each node independently, the nodes used to generate the POD modes need only be in the area of interest in the design or optimisation process. For example, the nodes on boundary surfaces may be the only of interest. In these cases, D may be smaller, which may lead to nodalwise interpolation being more efficient.

5. Interpolation using radial basis functions

As the dimension of the parameter space increases, the process of interpolation for the required coefficients becomes considerably more complicated. In this case, an attractive alternative is to employ interpolation using RBF. To illustrate this process, suppose that the entries in the $L \times 1$ vector $\boldsymbol{f}$ are the values, $f_\ell, \ell = 1, \ldots, L$, of an unknown function $F(\boldsymbol{x})$ at a set of data points, $\boldsymbol{x}_\ell, \ell = 1, \ldots, L$ in $\in \mathbb{R}^d$. The RBF approximation to F is the function

$$F_{RBF}(\boldsymbol{x}) = \sum_{\ell=1}^{L} w_\ell \Theta(|\, \boldsymbol{x} - \boldsymbol{x}_\ell \,|), \tag{24}$$

where Θ is defined, employing a multiquadric form, as

$$\Theta(|\, \boldsymbol{x} - \boldsymbol{x}_\ell \,|) = (|\, \boldsymbol{x} - \boldsymbol{x}_\ell|^2 + c^2)^{1/2}. \tag{25}$$

Here, c is a scalar parameter that effects the radius of influence of the data points. Suppose that the entries in the $L \times 1$ vector $\boldsymbol{w}$ are the values of the unknown coefficients, $w_\ell, \ell = 1, \ldots, L$ and let

$$W_{m\ell} = \Theta(|\, \boldsymbol{x}_m - \boldsymbol{x}_\ell \,|) \quad 1 \leqslant \ell, \quad m \leqslant L \tag{26}$$

It follows, from Eq. (24), that the unknown coefficients can be obtained by solving the matrix equation

$$\boldsymbol{W}\boldsymbol{w} = \boldsymbol{f}, \tag{27}$$

where $\boldsymbol{W}_{m\ell} = W_{m\ell}$. When the value of c is specified, the unknown coefficients can be obtained, provided that $\boldsymbol{W}$ is non-singular. It is known that the accuracy of the RBF approximation depends upon the value that is adopted for the parameter c [22]. Rippa [23] introduced an algorithm to find the optimum value for a particular data set but, for all the cases considered

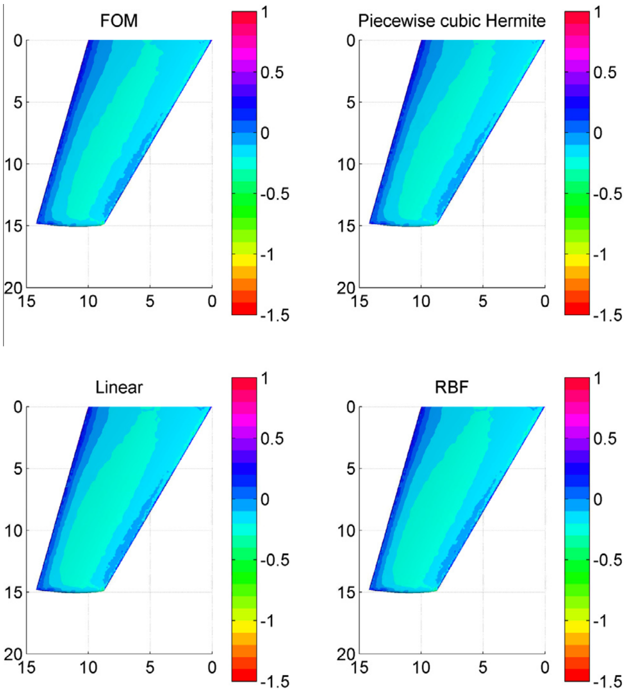

Fig. 7. The pressure field on the upper surface of the wing computed for Example 1 by the FOM and the ROM based upon RBF, Hermite and Linear interpolation at the time corresponding to the maximum deflection of the tip.

here, the additional expense incurred in performing an analysis of this type did not prove to be justified. Instead, the value of c is simply calculated to be the mean distance between the data points.

The primary advantage of using RBFs is its meshless nature. Other interpolation techniques require gradient information at each data point, which results in the need for a triangulation, or possibly, restrictions on the structure of the points. Using RBFs removes the need to consider the number of dimensions and allows easy refinement by the addition of data points. It is almost certainly the case that in the examples considered here, the advantages of RBFs are diminished due to the small number of parameters. However, when considering practical applications such as shape optimisation where the parameter space may contain many more than three dimensions, the advantages become more apparent. In addition to this in a practical setting, the number of parameters may change through the development/design process; this can be done seamlessly with an RBF implementation.

When compared with 31 competing interpolation techniques on a range of functions it was found that RBFs performed best in terms of accuracy for a variety of functions [33]. Interpolation techniques based on triangulations have the disadvantages of a large auxiliary storage required for the triangulation [33]. It was also found that the accuracy of techniques which require derivative information at the data points, such as Hermite interpolation, depended highly on the accuracy of the derivative estimates. Franke [33] states that extending triangulation and derivative based techniques to problems with more than two dimensions can be extremely difficult and in some cases impossible. The primary disadvantage to RBF interpolation is as the number of data points exceeds 100 the cost of calculating the weights becomes significant [33].

To validate this interpolation approach, Example 1 was reconsidered, and a ROM was again based upon FOM snapshots calculated with oscillation frequencies of $f1$, $f2$, $f4$ and $f5$. The pressure distribution on the upper surface of the wing predicted by the RBF based ROM is compared with corresponding results from the FOM in Fig. 7. When RBF interpolation of the coefficients is employed, the magnitude of the errors produced when the ROM was used to predict the solution with the oscillation frequency $f3$ were $Er_p = 0.4345\%$ and $Er_{C_\ell} = 6.4293\%$. Although these are less accurate than the results obtained previously using Hermite interpolation, it is appropriate to perform the analysis using RBFs which will be the most desirable interpolation technique to use in real applications.

6. Results

The numerical performance of the approach is now demonstrated for three further examples, involving additional complexity. For each of these examples, the RBF method is used to interpolate for the values of the coefficients.

Example 2 involves transonic inviscid flow over an oscillating ONERA M6 wing at a free stream Mach number of 0.84. The wing follows a prescribed sinusoidal pitch oscillation together with an oscillatory vertical heaving motion. The

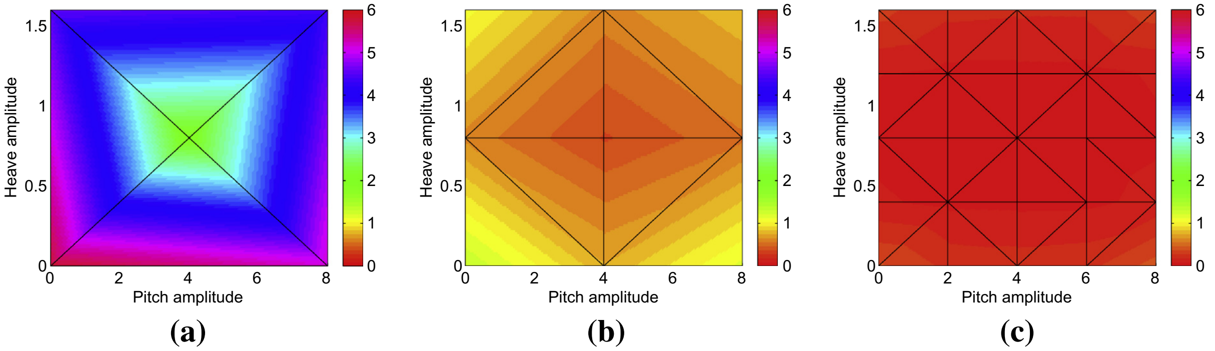

Fig. 8. Variation of Er_p over the parameter space for Example 2 for (a) the 5 snapshot set; (b) the 9 snapshot set; (c) the 25 snapshot set. For each plot, the horizontal axis denotes the pitch amplitude and the vertical axis the heave amplitude.

dimensionless frequency of the heave oscillation is prescribed to be $f = 2.778 \times 10^{-2}$, while the frequency of the pitch oscillation is taken to be equal to $2f$. The wing is held fixed at the root and the values of both the pitch and the heave at any instant are assumed to vary linearly between their values at the tip and at the root. The parameters of interest are the amplitude, a_p, of the pitch at the tip and the amplitude, a_h, of the heave at the tip, for the range of values $0 \leqslant a_p \leqslant 8^\circ$ and $0 \leqslant a_h \leqslant 1.6$.

An initial mesh was generated and 47 other meshes were produced, by mesh movement, to describe the wing geometry during each stage of the heave oscillation. Three ROMs were constructed, employing 5, 9 and 25 snapshot sets. All snapshot sets included the case where both a_p and a_h were equal to zero, which represents a steady state solution. The performance of each of the three ROMs was tested in the proposed manner, with each snapshot, in turn, being omitted during the construction of the ROMs. This enabled the ROM error Er_p to be estimated at each snapshot location in turn. The distribution of Er_p is plotted, as a function of location $\boldsymbol{\alpha} = (a_p, a_h)$ in the parameter space, in Fig. 8 for each of the three ROMs. Each vertex of the triangulation shown in these figures indicates the location of one of the snapshots. The triangulation was only used to interpolate, for plotting, between the computed values of Er_p at the vertices. It is interesting to note that the ROMs constructed using the 9 and the 25 snapshot sets are able to successfully predict the steady solution, using only unsteady snapshots. With

Table 2
Maximum percentage errors calculated during the validation procedure for Example 2 using ROMs with different numbers of snapshots.

Snapshots	Er_p	Er_{C_ℓ}
5	5.1563	24.807
9	1.1613	1.7277
25	0.4253	0.1413

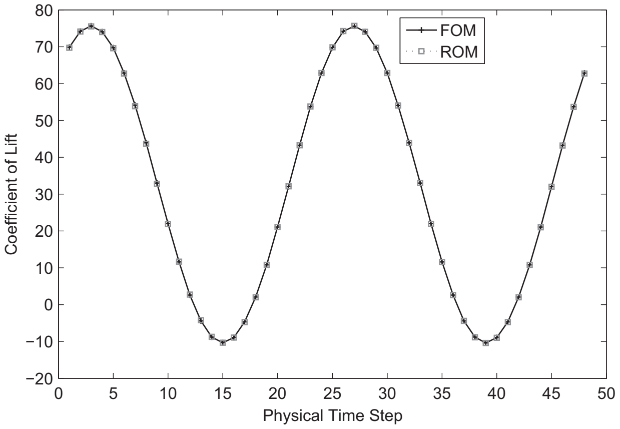

Fig. 9. Comparison of the variation in the lift coefficient computed with the FOM with that obtained from the ROM using 25 snapshots for Example 2, at the point $\boldsymbol{\alpha} = (8^\circ, 0)$ in the parameter space corresponding to the highest error. The difference is barely visible.

Table 3
cpu costs for Example 2.

Snapshots	C_{Build}	C_{ROM}	$(M_T - M)_{min}$
5	1.7644	4.6279	0.0001
9	4.8053	8.6116	0.0004
25	115.6546	36.8045	0.0105

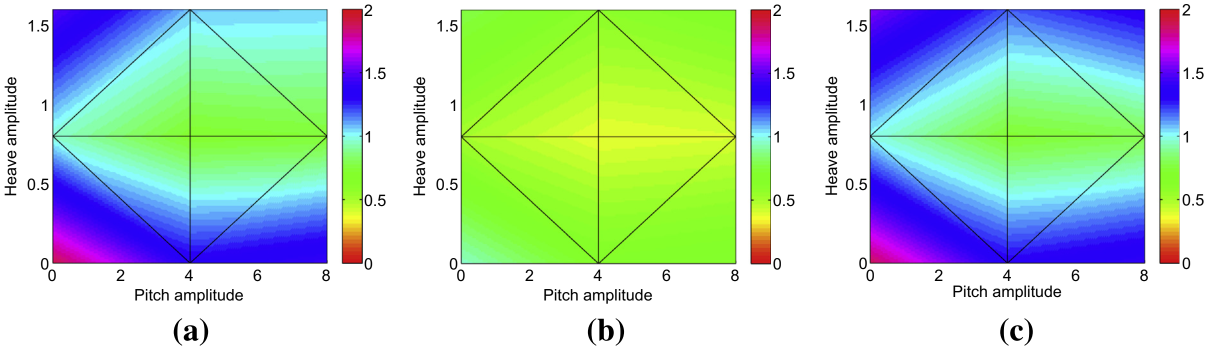

Fig. 10. Variation of Er_p over the parameter space for the ROM used in Example 3 for a frequencies (a) 4.167 × 10^{-2}; (b) 2.778 × 10^{-2}; (c) 2.083 × 10^{-2}. For each plot, the horizontal axis denotes the pitch amplitude and the vertical axis the heave amplitude.

the ROM based upon the 25 snapshot set, the steady state value of the lift coefficient was predicted to within 0.5268% of the actual value.

Table 2 lists the maximum percentage error measurements for the unsteady solutions, calculated using the validation procedure discussed in Section 4. The maximum error for the 25 snapshot set occurred for the case $\boldsymbol{\alpha} = (8^\circ, 0)$. This point is located on the boundary of the parameter space, where a high error could be expected. Fig. 9 compares, at this point in the parameter space, the variation in C_ℓ computed by the FOM with the variation obtained from the ROM.

During the validation process, the values of C_{Build} and C_{ROM} were measured and this enabled the calculation of $(\beta - 1)$ for each set of snapshots, using mean values of the cpu cost. In the evaluation of β, the value of C_{FOM} was taken as 442 800 s, which was a typical time for the FOM calculations. The resulting values are shown in Table 3. It can be observed that, for the largest snapshot set, the value of $(M_T - M)_{min}$ is 0.0105, which means that there would almost always be a net saving in cpu time.

The specification for Example 3 was exactly the same as that for Example 2, apart from the fact that the frequency f was included in the list of flow parameters of interest, so that $\boldsymbol{\alpha} = (a_p, a_h, f)$. The frequency range was defined to be $2.083 \times 10^{-2} \leqslant f \leqslant 4.167 \times 10^{-2}$. Snapshots at different frequencies were computed, on the same set of 48 meshes, by changing, as in Example 1, the size of the physical time step. For the purpose of illustration, the frequency values $2.083 \times 10^{-2}, 2.778 \times 10^{-2}$ and 4.167×10^{-2} were selected for the snapshots, together with the nine values of (a_p, a_h) used in Example 2. This leads to a set of 27 snapshots. The resulting ROM was again tested, by omitting one snapshot in turn, and the resulting distribution of Er_p is plotted in Fig. 10. The highest error in lift coefficient was 2.5045%, which occurred at the point $\boldsymbol{\alpha} = (0^\circ, 1.6, 2.083 \times 10^{-2})$. The time variation of the lift coefficient obtained from the ROM at this point is compared with the results of the FOM in Fig. 11. For this case, $(M_T - M)_{min} = 0.0135$, which represents a significant potential saving in cpu time using the method.

The specification for Example 4 was exactly the same as that for Example 2, apart from the fact that the values of the amplitudes of both the pitch and the heave at any instant are assumed to vary quadratically between their values at the tip and at the root. The quadratic variation adopted is such that the values at the semi-span are a quarter of the values at the tip. The solution procedure follows the approach adopted for Example 2, and the computed values of Er_p, obtained with the 25 snapshot set, are plotted, as a function of the amplitude of the pitch at the tip and the amplitude of the heave at the tip, in Fig. 12.

The maximum percentage errors in the unsteady solutions constructed using 5, 9 and 25 snapshots are shown in Table 4, where the maximum Er_{C_ℓ} value is 0.3301% and the maximum Er_p value is 0.2954%. Despite the increased complexity of the motion, these error values are seen to be of a similar order to those occurring in Example 2. The maximum value of Er_p again occurred for the case $\boldsymbol{\alpha} = (8^\circ, 0)$. For the case $\boldsymbol{\alpha} = (8^\circ, 1.6)$, the time variation of the lift coefficient obtained from the 25 snapshot ROM is compared to the results produced with the FOM in Fig. 13. The surface pressure distribution computed with this ROM is compared, at two different times, with the results obtained with the FOM in Fig. 14. The quadratic deformation of the wing surface is clearly apparent in the figure. For this ROM, it is found that $(M_T - M)_{min} = 0.0366$.

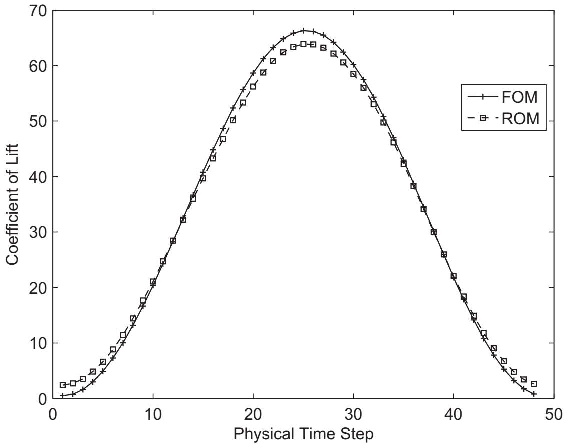

Fig. 11. Comparison of the variation in the lift coefficient computed with the FOM with that obtained from the ROM for Example 3 at the point $\boldsymbol{\alpha} = (0^\circ, 1.6, 2.083 \times 10^{-2})$ in the parameter space corresponding to the highest error.

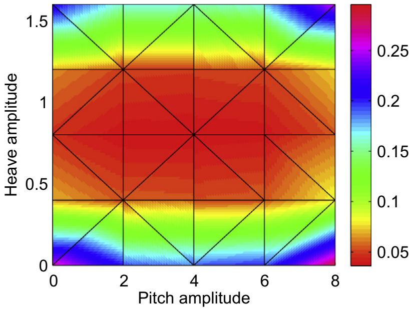

Fig. 12. Variation of Er_p over the parameter space for the ROM used in Example 4. The horizontal axis denotes the pitch amplitude and the vertical axis the heave amplitude.

Table 4
Maximum error measurements calculated during validation procedure for Example 4.

Snapshots	Er_p	Er_{C_ℓ}
5	3.9271	30.259
9	0.8944	3.6152
25	0.2954	0.3301

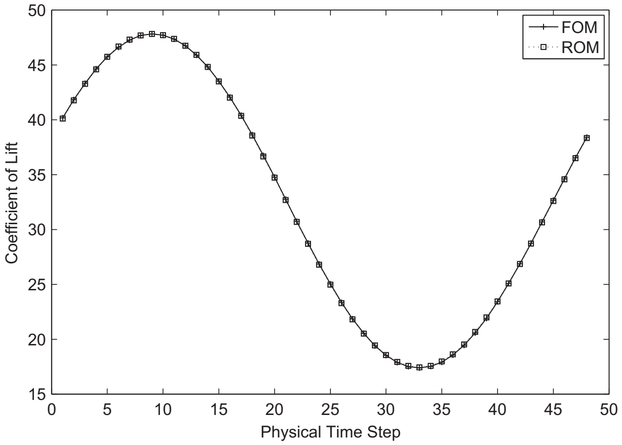

Fig. 13. Comparison of the variation in the lift coefficient computed with the FOM with that obtained from the 25 snapshot ROM for Example 4 at the point $\boldsymbol{\alpha} = (8^\circ, 1.6)$ in the parameter space.

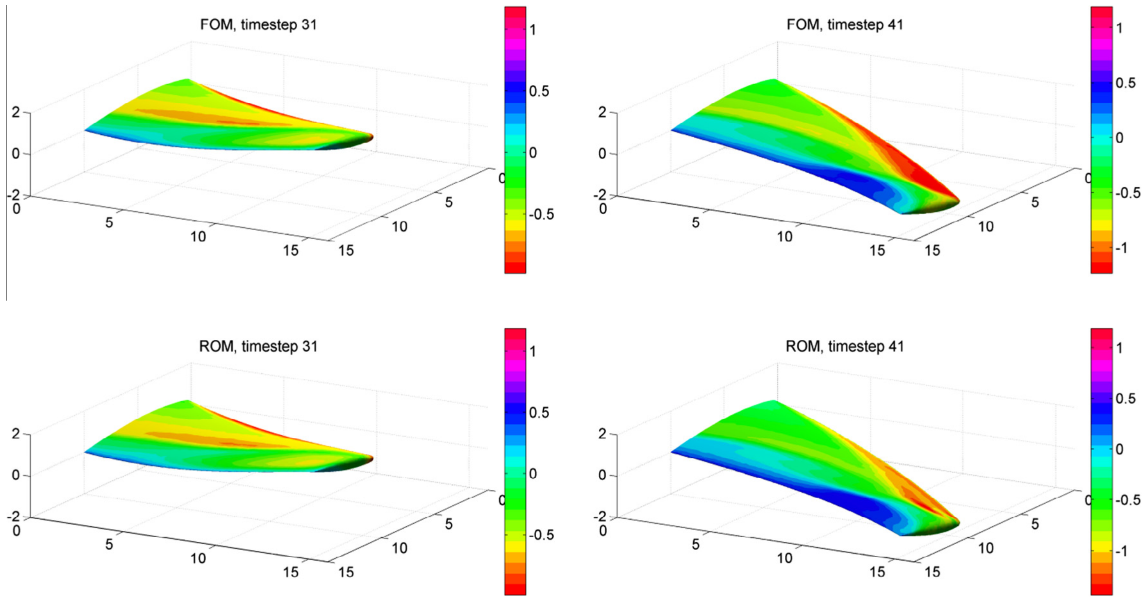

Fig. 14. The pressure field on the upper surface of the wing for Example 4 computed at the point $\boldsymbol{\alpha} = (8^\circ, 1.6)$ in the parameter space with the FOM and the 25 snapshot ROM at two different times.

7. Conclusion

The aim of this work was to attempt to address the problem of the large costs that are normally encountered when computational methods for unsteady flow are employed as part of a design process. The ROM technique that has been proposed acts on the computational domain and requires no knowledge of the system or of the solver used to generate the snapshots. The method is completely general and can be applied, without modification, to general time dependent problems. For the examples presented, the cpu cost of the ROM is several orders of magnitude lower than that for the FOM. Convergence of the error, as the number of snapshots increases, has been demonstrated and this increases confidence in the usefulness of the technique. The nature of the method means that the cpu cost of the ROM is almost always going to be much lower than that for the FOM, but we cannot assume that the loss of solution accuracy will be small in every case. However, generating the snapshots necessary for building the ROM would already be part of a normal design process. This method seamlessly fits into a parameter sampling cycle, starting with a coarse sampling of snapshots and iteratively validating the resulting ROM each time the sampling is refined. If an acceptable error in the ROM is reached, significant saving can be achieved by employing the ROM for the remaining parameter sweep.

One question remains, if these POD interpolation techniques turn out to be equivalent to a simple element by element interpolation as discussed in Section 3 is it worthwhile pursuing them further? Firstly, we need to consider how direct interpolation of the solution field compares to projection based ROM techniques. It is difficult to find comparisons of this nature, however in an early POD/ROM paper Pettit and Beran [34] *speculate* that using a ROM should be better than interpolation, but stated that this needed to be tested. Degroote et al. [30] provide a comparison between Kriging interpolation of the solution field and ROMs projected onto basis functions interpolated with respect to the parameter space. They found that direct interpolation of the solution field gives errors of similar order to that achieved by the projection ROMs. Results were also recently reported by Wang et al. [31] who compared POD interpolation techniques to POD projection techniques. In the first example presented, with a four dimensional parameter space, they found interpolation errors which were orders of magnitude lower than projection techniques. In the second example using a six dimensional parameter space the opposite result was shown, they argue that this shows POD interpolation to be ineffective, but it should be noted that only linear interpolation was used in their examples. When considering the stability problems [17,34,35] and boundary condition violations [17] reported when applying POD projection methods to simple two dimensional flow problems, our findings suggest interpolation may be an attractive alternative. The success of a direct interpolation technique relies on a smooth variation of the unknowns with respect to the parameters. In applications, such as electromagnetic scattering [36], where a small variation in parameters can result a sudden change in the unknowns, the sampling resolution required to capture parametric variation may be so high to make direct interpolation fruitless. In these cases, POD projection techniques may perform better.

Another question which needs answering is; are there any benefits to POD coefficient interpolation over element wise interpolation? A number of benefits can be identified which all result from the reduced number of interpolation operations. We have measured that, even when including an SVD of the snapshot matrix, POD coefficient interpolation is orders of magnitudes faster than element wise interpolation when using RBFs. Fewer values requiring interpolation also means it becomes feasible to inspect the smoothness of these values with respect to parameter variation and to apply adaptive interpolation schemes [37]. It may even be possible, with future investigation, to identify and capture discontinuities in non smooth parameter spaces.

Work has started on the development of an adaptive snapshot refinement technique, based on the error estimation process outlined in this paper. The objective will be to automate the process of selecting the new $\boldsymbol{\alpha}$ values that would be used for generating additional snapshots.

Acknowledgements

The authors thank the UK Engineering and Physical Sciences Research Council for the financial support provided under a DTA grant at Swansea University.

References

[1] J. Burkardt, M. Gunzburger, H. Lee, POD and CVT-based reduced-order modeling of Navier–Stokes flows, Computer Methods in Applied Mechanics and Engineering 196 (2006) 337–355.
[2] D. Alonso, A. Velazquez, J.M. Vega, A method to generate computationally efficient reduced order models, Computer Methods in Applied Mechanics and Engineering 198 (2009) 2682–2691.
[3] M.F. Barone, I. Kalashnikova, D.J. Segalman, H.K. Thornquist, Stable Galerkin reduced order models for linearized compressible flow, Journal of Computational Physics 228 (2009) 1932–1946.
[4] F. Fang, C.C. Pain, I.M. Navon, G.J. Gorman, M.D. Piggott, P.A. Allison, P.E. Farrell, A.J.H. Goddard, A POD reduced order unstructured mesh ocean modelling method for moderate Reynolds number flows, Ocean Modelling 28 (2009) 127–136.
[5] P.G.A. Cizmas, B.R. Richardson, T.A. Brenner, T.J. O'Brien, R.W. Breault, Acceleration techniques for reduced-order models based on proper orthogonal decomposition, Journal of Computational Physics 227 (2008) 7791–7812.
[6] M. Amabili, C. Touzé, Reduced-order models for nonlinear vibrations of fluid-filled circular cylindrical shells: comparison of POD and asymptotic nonlinear normal modes methods, Journal of Fluids and Structures 23 (2007) 885–903.
[7] Y. Utturkar, B. Zhang, W. Shyy, Reduced-order description of fluid flow with moving boundaries by proper orthogonal decomposition, International Journal of Heat and Fluid Flow 26 (2005) 276–288.
[8] M.V. Tabib, J.B. Joshi, Analysis of dominant flow structures and their flow dynamics in chemical process equipment using snapshot proper orthogonal decomposition technique, Chemical Engineering Science 63 (2008) 3695–3715.
[9] X. Gilliam, J.P. Dunyak, D.A. Smith, W. Fuqiang, Using projection pursuit and proper orthogonal decomposition to identify independent flow mechanisms, Journal of Wind Engineering and Industrial Aerodynamics 92 (2004) 53–69.
[10] M. Bergmann, L. Cordier, Optimal control of the cylinder wake in the laminar regime by trust-region methods and POD reduced-order models, Journal of Computational Physics 227 (2008) 7813–7840.
[11] H.V. Ly, H.T. Tran, Modeling and control of physical processes using proper orthogonal decomposition, Mathematical and Computer Modelling 33 (2001) 223–236.
[12] K. Gaetan, J–C. Golinval, A.F. Vakakis, L.A. Bergman, The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview, Nonlinear Dynamics 41 (2005) 147–169.
[13] K.E. Willcox, J.D. Paduano, J. Peraire, K.C. Hall, Low order aerodynamic models for aeroelastic control of turbomachines, AIAA Paper, pp. 99–1261, 1999.
[14] K. Willcox, J. Peraire, J. White, An Arnoldi approach for generation of reduced-order models for turbomachinery, Computers & Fluids 31 (2002) 369–389.
[15] T. Lieu, C. Farhat, M. Lesoinne, Reduced-order fluid/structure modeling of a complete aircraft configuration, Computer Methods in Applied Mechanics and Engineering 195 (2006) 5730–5742.
[16] C.W. Rowley, T. Colonius, R.M. Murray, Model reduction for compressible flows using POD and Galerkin projection, Physica D 189 (2004) 115–129.
[17] D.J. Lucia, P.S. Beran, Projection methods for reduced order models of compressible flows, Journal of Computational Physics 188 (2003) 252–280.
[18] D. My-Ha, K.M. Lim, B.C. Khoo, K. Willcox, Real-time optimization using proper orthogonal decomposition: free surface shape prediction due to underwater bubble dynamics, Computers & Fluids 36 (2007) 499–512.
[19] A. Qamar, S. Sanghi, Steady supersonic flow-field predictions using proper orthogonal decomposition technique, Computers & Fluids 38 (2009) 1218–1231.
[20] J. Reuther, A. Jameson, J. Farmer, L. Martinelli, D. Saunders, Aerodynamic shape optimization of complex aircraft configurations via an adjoint formulation, AIAA Paper 96–0094, 1996.
[21] R.L. Hardy, Multiquadric equations of topography and other irregular surfaces, Journal of Geophysical Research 76 (1971) 1905–1915.
[22] P. Mokhasi, D. Rempfer, Nonlinear system identification using radial basis functions, International Journal for Numerical Methods in Fluids 63 (2010) 121–162.
[23] S. Rippa, An algorithm for selecting a good value for the parameter *c* in radial basis function interpolation, Advances in Computational Mathematics 11 (1999) 193–210.
[24] O. Hassan, K. Morgan, N. Weatherill, Unstructured mesh methods for the solution of the unsteady compressible flow equations with moving boundary components, Philosophical Transactions of the Royal Society A 365 (2007) 2531–2552.
[25] K.A. Sørensen, O. Hassan, K. Morgan, N.P. Weatherill, A multigrid accelerated time-accurate inviscid compressible fluid flow solution algorithm employing mesh movement and local remeshing, International Journal for Numerical Methods in Fluids 43 (2003) 1207–1229.
[26] O. Hassan, K.A. Sørensen, K. Morgan, N.P. Weatherill, A method for time accurate turbulent compressible fluid flow simulation with moving boundary components employing local remeshing, International Journal for Numerical Methods in Fluids 53 (2007) 1243–1266.
[27] A. Jameson, W. Schmidt, E. Turkel, Numerical simulation of the Euler equations by finite volume methods using Runge–Kutta timestepping schemes, AIAA Paper, pp. 81–1259, 1981.
[28] X. Liu, N. Qin, H. Xia, Fast dynamic grid deformation based on Delaunay graph mapping, Journal of Computational Physics 211 (2006) 405–423.
[29] MATLAB version 7.8.0. Natick, Massachusetts: The MathWorks Inc., 2009.
[30] J. Degroote, J. Vierendeels, K. Willcox, Interpolation among reduced-order matrices to obtain parameterized models for design, optimization and probabilistic analysis, International Journal for Numerical Methods in Fluids 63 (2010) 207–230.
[31] Y. Wang, B. Yu, Z. Cao, W. Zou, G. Yu, A comparative study of POD interpolation and POD projection methods for fast and accurate prediction of heat transfer problems, International Journal of Heat and Mass Transfer 55 (17-18) (2012) 4827–4836. <http://www.sciencedirect.com/science/article/pii/S0017931012003043>.
[32] E. Bouhoubeiny, P. Druault, Note on the POD-based time interpolation from successive PIV images, Comptes Rendus Mécanique 337 (2009) 776–780.
[33] R. Franke, Scattered data interpolation: tests of some methods, Mathematics of Computation 38 (1982). <http://www.ams.org/journals/mcom/1982-38-157/S0025-5718-1982-0637296-4>.

[34] C.L. Pettit, P.S. Beran, Application of proper orthogonal decomposition to the discrete Euler equations, International Journal for Numerical Methods in Engineering 55 (2002) 479–497.

[35] J.S.R. Anttonen, P.I. King, P.S. Beran, Applications of multi-POD to a pitching and plunging airfoil, Mathematical and Computer Modelling 42 (2005) 245–259.

[36] P.D. Ledger, J. Paraire, K. Morgan, O. Hassan, N.P. Weatherill, Parameterised electromagnetic scattering solutions for a range of incident wave angles, Computer Methods in Applied Mechanics and Engineering 193 (2004) 3587–3605.

[37] T.J. Mackman, C.B. Allen, Investigation of an adaptive sampling method for data interpolation using radial basis functions, International Journal for Numerical Methods in Engineering 83 (2010) 915–938.