Seventh South African Conference on Computational and Applied Mechanics
SACAM10
Pretoria, 10−13 January 2010

REDUCED ORDER MODELLING TECHNIQUES FOR MESH MOVEMENT STRATEGIES AS APPLIED TO FLUID STRUCTURE INTERACTIONS

Alfred E.J. Bogaers [a], Schalk Kok[b] and Arnaud G. Malan[c]

[a]Department of Mechanical and Aeronautical Engineering
University of Pretoria, Pretoria 0002, South Africa,
[a]abogaers@csir.co.za
[b]Advanced Mathematical Modelling, CSIR Modelling and Digital Science,
P.O. Box 395, Pretoria 0001, South Africa
[b]skok@csir.co.za
[c]Aeronautic System Competency, CSIR Defence, Peace, Safety and Security,
P.O. Box 395, Pretoria 0001, South Africa
[c]amalan@csir.co.za.

Keywords: Mesh movement, Reduced Order Modelling, Proper Orthogonal Decomposition

Abstract

In this paper, we implement the method of Proper Orthogonal Decomposition (POD) to generate a reduced order model (ROM) of an optimization based mesh movement technique. In the study it is shown that POD can be used effectively to generate a ROM, that accurately reproduces the full order mesh movement algorithm, with a decrease in computational time of over 99%.

1 Introduction

The numerical simulation of flow across a boundary arises in many engineering related problems, e.g. flutter simulations of wings, blood flow through veins and arteries and parachute dynamics. These, and other fluid-structure interaction (FSI) problems involve flow induced moving boundaries, and in order to accurately complete these unsteady flow simulations it becomes necessary for the computational grid to conform to the new domain. To this end, several methods or algorithms have been developed that can cheaply adapt unstructured meshes to the new displaced boundary, including the spring analogy [4], solving a set of Laplacian or Bi-harmonic equations [7], radial basis function (RBF) interpolation [3, 15] or through mesh optimization [1, 6].

Despite the successes of these algorithms in reducing the frequency and necessity for remeshing, they still account for a significant percentage of CPU time for any FSI simulation. The aim of this paper is to attempt to generate effective reduced order models for these mesh

movement strategies in the hope of reducing their associated cost, while not sacrificing the quality of the resulting meshes. The reduced order modeling technique that we make use of is the method of Proper Orthogonal Decomposition (POD), also commonly referred to as Principal Component Analysis (PCA), Singular Value Decomposition (SVD) or Karhunnen-Loéve (KL) decomposition.

Proper Orthogonal Decomposition (POD) is a mathematical procedure aimed at finding low-dimensional approximate descriptions of high-dimensional systems. POD is in essence an empirical spectral method, similar to Fourier decomposition, where field variables are approximated using expansions of a set of projected basis functions or modes. POD obtains these basis functions from a set of observations, where these observations can be obtained either experimentally or through numerical simulations of a real system. What makes POD remarkable is that the selected modes are not only appropriate but make up the optimal linear basis for describing any given system.

POD has been applied in a wide range of disciplines including image processing [16], data compression, control in chemical engineering and oceanography. The first use of POD in the field of Fluid Mechanics was by Lumley [12] as a post processing step for determining coherent structures within turbulent flow. Since then POD has successfully been applied to many engineering problems including the characterization of dominant turbulent flow properties [8, 14], aircraft flutter prediction [11] and reduced order models for multidisciplinary optimization [5, 10].

2 Proper Orthogonal Decomposition

The governing theory, derivation and application of the POD method can readily be found in literature, and for a comprehensive discussion on the method refer to Holmes *et al.* [8], and Chatterjee [2] for an easy to understand introduction. In order to keep this paper as self contained as possible, we provide a brief overview of POD, and the method of snapshots.

Consider that we have a set of system observations, or snapshots, $\{\boldsymbol{x}^k\}$, where for mesh movement, these snapshots are the nodal co-ordinates of the mesh at various instances of boundary displacement. The method of POD extracts the predominant variances of these system observations in the form of *optimal linear* basis modes $\{\boldsymbol{\varphi}_j\}$, and allows for an approximation of $\{\boldsymbol{x}^k\}$ through the following linear combination:

$$\{\boldsymbol{x}^k\} = \sum_{j=1}^{M} \alpha_j^k \boldsymbol{\varphi}_j, \tag{1}$$

where $\{\boldsymbol{\alpha}_j\}$ is an appropriate set of expansion coefficients and M is the number of retained modes.

The POD modes themselves may be computed using the "method of snapshots" developed by Sirovich [18]. The kernel of the POD modes is a finite dimensional autocorrelation matrix of the form

$$\mathcal{R} = \frac{1}{M}\boldsymbol{X}\boldsymbol{X}^T, \tag{2}$$

where $\boldsymbol{X}$ is an observation matrix of size $M \times N$, where each row vector of the matrix represents a snapshot. N is the total number of x and y nodal co-ordinates within the mesh and M is the number of snapshots. The eigenvectors $\boldsymbol{a}$ of $\mathcal{R}$ is computed as an intermediate step from

$$\mathcal{R}\boldsymbol{a} = \lambda\boldsymbol{a}, \tag{3}$$

where the POD basis modes can then be computed as the linear combination

$$\boldsymbol{\varphi}^k = \sum_{i=1}^{M} a_i^k \boldsymbol{x}^i \;\; \text{for}\, k = 1, 2, ..., M, \tag{4}$$

where a_i^k is the i^{th} element of eigenvector $\boldsymbol{a}^k$ corresponding to λ_k. For the mesh movement problem, the eigenvalues λ have no physical interpretation, save that their magnitudes provide an indication as to how much of the system information is captured by the associated POD mode. Ordering λ and the associated POD modes in order of descending magnitude, we may reproduce the approximation in (1) by retaining only first K most dominant modes.

It should also be noted, that if the coefficients in (1) are chosen as the eigenvectors $\boldsymbol{a}^k$, corresponding to the k^{th} POD mode, then we would generate an approximation of the k^{th} snapshot. If however we would like to compute an approximation of the mesh movement based on boundary displacements other than those used in the training snapshots, we would require the computation of an appropriate set of expansion coefficients $\{\boldsymbol{\alpha}\}$.

3 Mesh Movement Through Optimization

Of the various available mesh movement techniques, mesh optimization arguably produces the highest quality meshes at a high computational cost, making it a prime candidate for the concepts of reduced order modeling. For the purposes of our POD demonstrator and snapshot training, we therefore make use of the mesh movement method based on optimization.

Mesh movement through mesh quality optimization is not so much a mesh movement technique but rather a mesh smoothing operation. At each instance of domain boundary movement, the mesh is regularized or smoothed by allowing each of the mesh vertices to relocate, through the use of some optimization algorithm. The mesh is optimized according to an objective function which in some fashion describes the overall quality of the elements. Formally, the mathematical optimization problem is stated as follows:

$$\underset{\text{w.r.t}\, \boldsymbol{x}}{\text{minimize}}\, F(\boldsymbol{x}), \;\; \boldsymbol{x} = [x_1, x_2, ..., x_n]^T \tag{5}$$

where $F(\boldsymbol{x})$ is the objective function to be optimized and $\boldsymbol{x}$ is the x and y co-ordinates of the interior mesh vertices that are allowed to move.

For this particular study we limit ourselves to 2D unstructured triangular meshes, where the mesh quality is defined by the shape-size metric

$$F = \sum_{\text{elements}} \left(\frac{r_{\text{out}}}{r_{\text{in}}} \right) \left(\frac{1}{f_{\text{size}}} \right), \tag{6}$$

where r_{out} and r_{in} are the radii of the circumscribed and inscribed circles of a triangle respectively, and the quotient $\left({}^{r_{\text{out}}}/_{r_{\text{in}}}\right)$ is a measure of the element shape. Assuming isotropic physics, the perfect triangular element shape is an equilateral triangle, for which the quotient is 3, and tends to infinity as the element deteriorates. f_{size} is a size metric that relates the volume of an element in its deformed state, w_d, to the original volume in the starting mesh, w_o, defined by

$$f_{\text{size}} = \frac{2\tau}{1+\tau^2}, \tag{7}$$

where $\tau = {}^{w_d}/_{w_o}$. The range of f_{size} is $0 \leq f_{\text{size}} \leq 1$, where 1 implies an exact match, and the quotient $\left({}^{1}/_{f_{\text{size}}}\right)$ ranges between 1 and ∞ for perfect and degenerate elements respectively. The quality metric (6) is a combination of metrics proposed by Braess et al.[1] and Knupp [9].

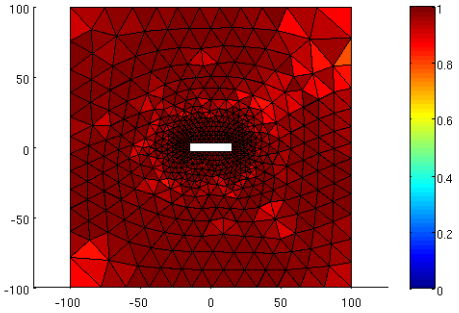
Two gradient based optimization algorithms were made use of, namely the Conjugate gradient method and Newton's method with line search. A line search is implemented in conjunction with Newton's method to reduce starting point dependency. The quality metric defined in (6) is continuous and differentiable everywhere, allowing for the first derivatives and the Hessian required by the optimization algorithms to be computed analytically. To take advantage of the sparsity of the Hessian matrix, only the non-zero entries are saved and a sparse matrix solver with LU factorization is implemented along with node re-numbering to produce LU factors with near minimal bandwidth.

4 Test Case

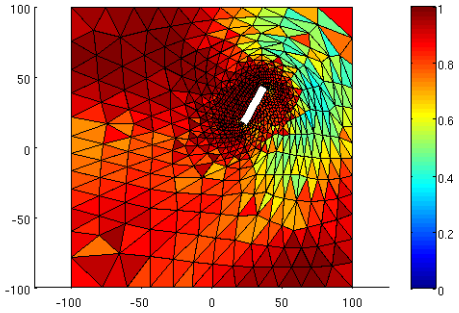
To test the applicability of POD applied to mesh movement we make use of a 2D unstructured triangular mesh with extreme rotation and translation. The test problem consists of a 200×200 unit square domain with a small inner rectangle that is rotated 60° counter-clockwise and translated by $\triangle x = 30$ and $\triangle y = 30$ units, with the initial mesh shown in Figure 1(a). The contour plots use an element shape-size quality indicator defined in [9], where $0 \leq f_{ss} \leq 1$, for degenerate and perfect elements respectively.

4.1 Snapshot Generation

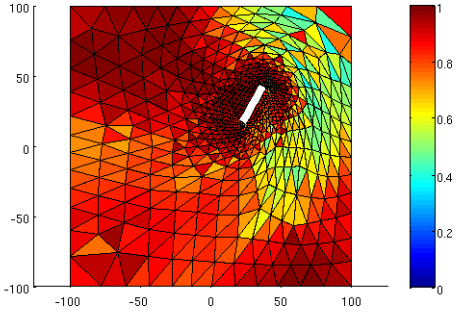
The first requirement for generating a POD model is to acquire a set of snapshots. A POD based model can usually only reproduce information within close proximity to the information contained in these snapshots. The boundary movements must therefore be chosen to



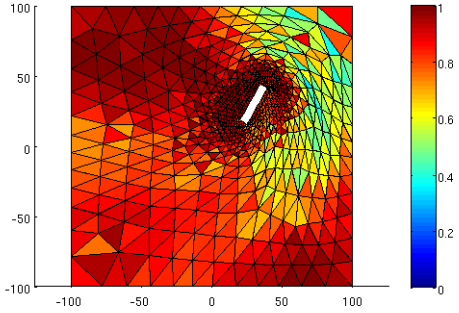
(a) Initial mesh, generated using NETGEN [17].



(b) Full order mesh optimization.



(c) ROM: Coefficient Interpolation.



(d) ROM: Coefficient Optimization.

Figure 1: Rotation and translation test case. Mesh quality contour plots using quality metric $0 \leq f_{ss} \leq 1$ [3].

closely mimic the expected movement in an actual simulation. This inherently means that a certain *a priori* knowledge is required as to how the actual system will behave. Fortunately, in most engineering problems, a substantial amount of information relating to a problem, such as the predominant physics at play, is usually known, or can be estimated prior to the simulation.

As a side note, it is certainly possible, for the current application of POD to mesh movement, to generate the snapshots adaptively rather than as a pre-processing step. In so doing, we make use of the POD model to move the mesh until the mesh quality deteriorates below some lower limit, at which point a full order mesh optimization step is performed. The results of the full order mesh movement can then be used as an additional training snapshot to update the ROM.

For the current test problem, we generate 60 snapshots as a pre-processing step, where the magnitudes of rotation and translation of the inner rectangle is chosen via the method of Latin hypercube sampling (LHS). LHS is often used in uncertainty analysis, and was developed primarily to generate a distribution of plausible collections of parameter values from a multidimensional distribution [19, 13]. In essence, LHS allows for random selections of the

displacement and rotation magnitudes, with uniform distribution.

There is no way of knowing prior to an actual analysis how many snapshots are required to adequately describe the system, and the choice of 60 snapshots is simply an educated guess. The mesh movement for each of the LHS specified locations is performed using the mesh optimization method, with a minor modification. For the optimization method, the largest incremental boundary displacement is limited to the smallest elements sizes located along the moving boundary. Deformations larger than these elements result in element inversion, and the subsequent failure of the optimization algorithm to find an appropriate minimum. As a consequence, large boundary movements have to be broken into smaller increments. Each of these increments involves the solution of a full order optimization problem, which is highly expensive. To minimize the associated cost we make use of the method of radial basis function interpolation (RBF), to provide an initial guess, followed by optimization.

RBF interpolation is currently one of the more popular mesh movement methods available, as it results in comparably good quality meshes, at relatively cheap computational costs. For the sake of brevity, we will not discuss the details regarding the RBF method, but for more information refer to de Boer *et al.* [3] or Rendall *et al.* [15]. RBF interpolation moves the internal mesh nodal co-ordinates according to an interpolation function based on the motion of the boundary nodes. If used on its own, RBF interpolation requires several displacement increments in order to attain high quality final meshes. For our particular application, we are not interested in the quality of the mesh provided by RBF, save that the method deforms the mesh from its original position to the LHS specified point without any element inversion. To this end, a single RBF increment is used before the the mesh is smoothed using full order optimization.

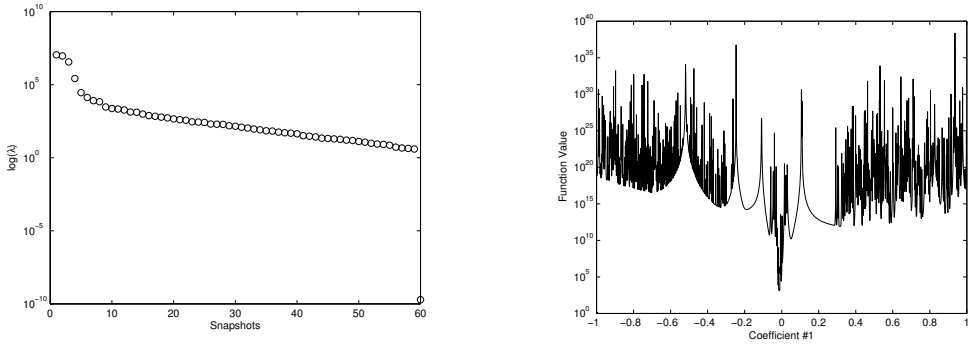
4.2 POD Based Reduced Order Model of the Optimization Mesh Movement Method

With the snapshots generated, we are able to compute the POD basis modes, where Figure 2(a) shows the ordered eigenvalues associated with the POD modes. From the magnitudes of the eigenvalues it may be noted that the first 8 POD modes contain over 99% of the system information. For our POD model we choose to retain the first 10 modes, essentially reducing the degree of freedom of the problem from $2N$, the total number of grid points, to just 10.

By substituting the approximation for $\boldsymbol{x}$ in (1) into (5), we can now solve our original optimization problem, but now only in terms of our expansion coefficients α, and the 10 POD modes,

$$\underset{\text{w.r.t. } \alpha}{\text{minimize}}\, F(\boldsymbol{x}(\alpha)),\ \boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_n]^T . \tag{8}$$

The objective function (8) based on α is populated with a large set of local minima. To illustrate, an example of the function profile is shown in Figure 2(b), generated by altering α_1. Because of these local minima, there is no gradient based optimization routine available that will guarantee convergence of the solution to the global minimum, unless the starting



(a) **(b)**

Figure 2: (a) Ordered eigenvalues, provides an indication to the percentage of system information captured by associated POD modes. (b) Example function profile for objective function as a function of expansion coefficients $\{\alpha\}$.

points are chosen to be close to the exact solution, or more appropriately, within the valley of the global minimum.

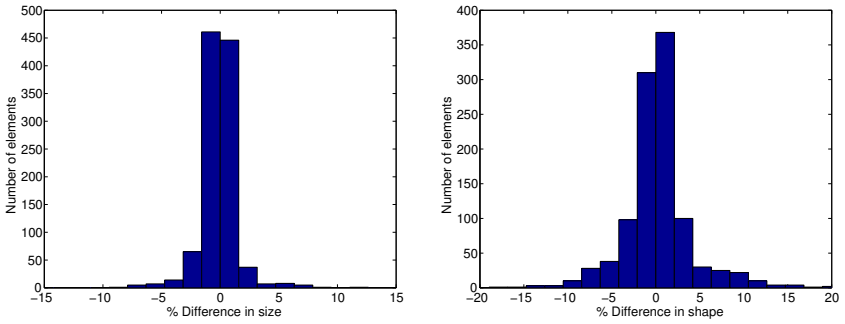
The presence of the local minima is attributed to element inversion. A similar problem is present within the full mesh optimization, but never arises as a full computational issue. The optimization algorithm individually moves each of the nodal co-ordinates, and allow none to be moved to locations that result in element inversion. When defining the objective cost function in terms of the coefficients α, these local minima are an acute problem, because even a minor change in one of the expansion coefficients has an effect on all the nodal co-ordinates.

We propose the use of an interpolation method to obtain an approximation to the coefficients as an initial guess. The interpolation method used is through RBF interpolation, primarily as it is adept at handling scattered data of highly non-linear relationships, and an already working RBF code is available from the mesh movement work. Interpolation is used to equate a function relating the expansion coefficients to the moving boundary nodes based on the generated set of snapshots.

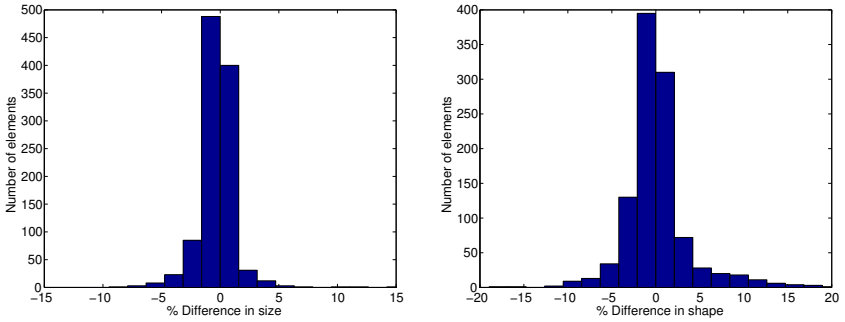
4.3 Results

Figure 1 depicts the final meshes produced, for the mesh rotation and translation, by the full order mesh movement scheme and ROMs, based on coefficient interpolation, and coefficient optimization. The results themselves appear to be promising, with little noticeable difference between the ROMs and the full order mesh movement; furthermore there appears to be little difference between the two ROMs.

To physically quantify the comparison between the ROMs and the full order mesh movement, histogram plots in Figure 3 are shown comparing the percentage difference in terms of size and shape between each of the mesh elements compared to the full order solution. For the ROM based solely on coefficient interpolation, 97.8% of the elements differ by less than 5% in terms of size and 86.1% of the elements shapes differ by less than 5%. By further



(a) POD ROM with coefficient interpolation only.



(b) POD ROM with coefficient optimization.

Figure 3: Histogram plot of the percentage difference in shape and size for each element in the mesh of the ROMs compared to the full order mesh optimization. For $\triangle x = 30$, $\triangle y = 30$, $\phi = 60°$CCW.

optimizing the coefficients, a minor improvement of 98.7% and 88.3% of the elements differ by less than 5% in terms of size and shape respectively.

Of more importance than the physical differences between the meshes is rather whether they are appropriate for the use in an actual simulation. In Table 1, we compare the minimum and mean element qualities for each of the movement models, based on the quality metric $0 \leq f_{ss} \leq 1$ [9]. While the ROMs are by no means an exact replication, they are certainly no less suitable than the full order optimization. Neither of the ROMs result in degenerate elements, and according to the metric f_{ss}, the ROM based only on coefficient interpolation produces a better final mesh than the full optimization.

	Initial Mesh	Full Optimization	Coefficient Interpolation	Coefficient Optimization
$\min(f_{ss})$	0.7917	0.3812	0.3770	0.3563
mean (f_{ss})	0.9743	0.8306	0.8326	0.8299
$F(\boldsymbol{x})$	-	3.9540×10^5	5.0750×10^5	5.0527×10^5

Table 1: Minimum and mean element qualities in terms of $0 \leq f_{ss} \leq 1$, as well as the minimized objective function, for the mesh movement depicted in Figure 1.

The fact that the coefficient interpolation ROM yielded better results does not mean that it

produced a better approximation of the full order mesh movement. The metric f_{ss} is not the same mesh metric as was used in the cost function. By comparing the cost function, we find that the best solution is obtained by the full optimization with the closest approximation attained by optimizing the coefficients. We make use of f_{ss} to evaluate the appropriateness of the mesh qualities because it is well bounded between 0 and 1 and allows for nice intuitive comparisons; the cost function metric on the other hand ranges from 0 to ∞, where the actual values have little meaning, save that they should be as low as possible. As a side note, f_{ss} is a poor choice for an optimization cost function because the extremes are not harsh enough. If used in a gradient based optimization routine, it leads to several elements becoming near degenerate for an overall improvement in mesh quality.

In short, both ROMs produce acceptable quality meshes. Unless a higher degree of accuracy is desired for the replication, using only an interpolation method to compute the expansion coefficients is more than sufficient for the purposes of generating an effective mesh movement ROM.

4.4 Computational Efficiency

To determine the computational cost associated with the POD method, a ROM is implemented and tested on the same square domain with small inner rectangle for a range of mesh sizes. The results of the CPU and memory scaling is shown in Figure 4.

The Conjugate gradient method is known to scale well with problem size, and for large problems is expected to outperform second order methods. For the full-order mesh movement problem, we find that Newton in fact performs better for an increase in problem size. The reason for this is based on the manner in which the two methods obtain their respective search directions. The Conjugate gradient method utilizes only first order, local gradient information. As a result, elements far away from one another have no inter-relating information.

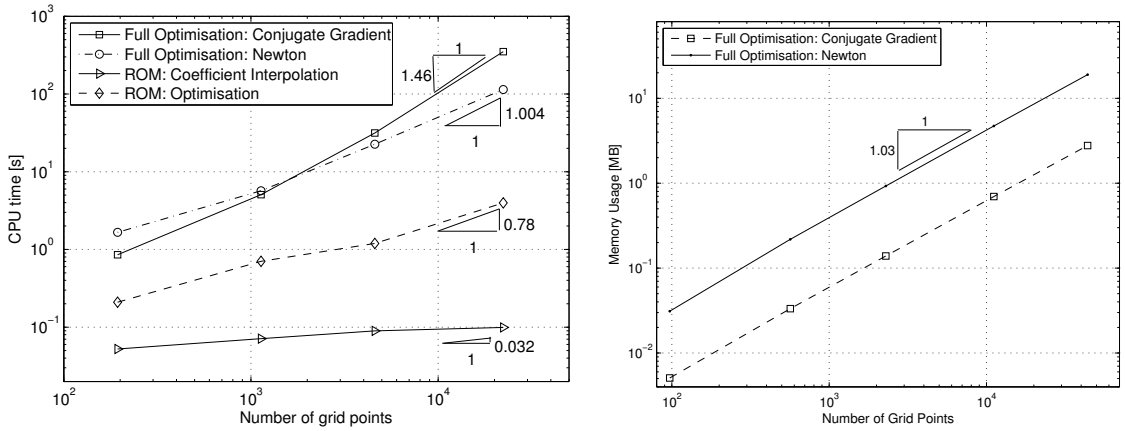
For example, if the elements along the inner rectangle boundary are distorted, the elements further away at the edge of the domain have no gradient information regarding this distortion. It takes several iterations for this information to propagate through the mesh. On the other hand, Newton obtains search directions based on a Hessian, which incorporates information for the whole mesh domain; while being more expensive per iteration, Newton requires substantially fewer iterations to reach convergence.

A further benefit is the sparsity of the Hessian. For the mesh in Figure 1, the size of the Hessian matrix is $[1132 \times 1132]$, while containing only 13200 non-zero entries, accounting for just a little over 1% of the total number of entries in the full matrix. The effect of using a sparse solver on such a sparse matrix, is that Newton scales at a near one to one ratio as a function of problem size, in terms of both CPU and memory usage.

If we now compare the computational cost of the ROMs to the full order Newton, we find that using coefficient interpolation as the starting point, and then further optimizing, that we have a total CPU savings ranging from 87.35% to 96.51% for the smallest to largest mesh respectively. While these cost reductions are significant, an even greater saving of up to

99.99% is found by using interpolation alone, and more importantly a cost scaling factor of 0.032.

Computing the ROM coefficients through interpolation does not require the solution of any equations, but only equating a set of interpolation functions. The cost of these interpolation functions is based only on the number of grid points along the moving boundary and the number of retained POD modes. The number of nodes along the inner boundary is orders less than the total DOF of the mesh, and the number of modes and coefficients is kept constant at 10. As a result, we now have a mesh movement solution methodology, with an associated cost that essentially remains constant as the mesh problem size increases.



(a) CPU scaling.

(b) Memory usage scaling.

Figure 4: Comparison of the CPU and memory scaling.

5 Conclusion

In this paper we demonstrated the ability of the method of POD to generate effective reduced order models of mesh movement algorithms. We implemented and tested the method to an optimization mesh movement scheme, though the method itself is just as easily applicable to any movement method. The POD model was trained as a pre-processing step to a simple combined rotation and translation test case, and it was found that computing the POD expansion coefficients using interpolation alone resulted in comparably good quality final meshes, with CPU cost reductions in excess of 99.99%.

References

[1] H. Braess and P. Wriggers. Arbitrary Lagrangian Eulerian finite element analysis of free surface flow. *Computer Methods in Applied Mechanics and Engineering*, 190(1-2):95–109, 2000.

[2] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, 78(7):808–817, 2000.

[3] A. de Boer, M.S. can der Schoot, and H. Bijl. Mesh deformation based on radial basis function interpolation. *Compters and Structures*, 85:784–795, 2007.

[4] C. Farhat, C. Degend, B. Koobus, and M. Lesoinne. Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Computer Methods in Applied Mechanics and Engineering*, 163:231–245, 1998.

[5] R. Filomeno Coelho, P. Breitkopf, and C. Knopf-Lenoir. Model reduction for multidisciplinary optimization-application to a 2D wing. *Structural and Multidisciplinary Optimization*, 37(1):29–48, 2008.

[6] L. Freitag Diachin, P. Knupp, T. Munson, and S. Shontz. A comparison of two optimization methods for mesh quality improvement. *Engineering with Computers*, 22(2):61–74, 2006.

[7] B. T. Helenbrook. Mesh deformation using the biharmonic operator. *International Journal for Numerical Methods in Engineering*, 56:1007–1021, 2003.

[8] P. Holmes, J.L. Lumley, and G. Berkooz. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Univ Pr, 1998.

[9] P.M. Knupp. Algebraic mesh quality metrics for unstructured initial meshes* 1. *Finite Elements in Analysis and Design*, 39(3):217–241, 2003.

[10] P.A. LeGresley and J.J. Alonso. Airfoil design optimization using reduced order models based on proper orthogonal decomposition. In *Fluids 2000 Conference and Exhibit, Denver, CO*, 2000.

[11] T. Lieu, C. Farhat, and M. Lesoinne. Reduced-order fluid/structure modeling of a complete aircraft configuration. *Computer methods in applied mechanics and engineering*, 195(41-43):5730–5742, 2006.

[12] JL Lumley. The structure of inhomogeneous turbulent flows. *Atmospheric turbulence and radio wave propagation*, pages 166–178, 1967.

[13] M.D. McKay, RJ Beckman, and WJ Conover. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61, 2000.

[14] P. Moin and R.D. Moser. Characteristic-eddy decomposition of turbulence in a channel. *Journal of Fluid Mechanics*, 200:471–509, 1989.

[15] T.C.S Rendall and C.B. Allen. Unified fluid-structure interpolation and mesh motion using radial basis functions. *International Journal for Numerical Methods in Engineering*, 74:1519–1559, 2008.

[16] A. Rosenfeld and A.C. Kak. *Digital picture processing*. Academic Press, Inc. Orlando, FL, USA, 1982.

[17] J. Schöberl. NETGEN An advancing front 2D/3D-mesh generator based on abstract rules. *Computing and visualization in science*, 1(1):41–52, 1997.

[18] L. Sirovich. Turbulence and the dynamics of coherent structures. I-Coherent structures. II-Symmetries and transformations. III-Dynamics and scaling. *Quarterly of applied mathematics*, 45:561–571, 1987.

[19] M. Stein. Large sample properties of simulations using Latin hypercube sampling. *Technometrics*, 29(2):143–151, 1987.