AMSI VACATION RESEARCH SCHOLARSHIPS 2018-2019

Utilising Sparse Grids for Airfoil simulations & Uncertainty Quantification

James Martini
Supervised by Stephen Roberts
Australian National University

Vacation Research Scholarships are funded jointly by the Department of Education and Training and the Australian Mathematical Sciences Institute.

AMSI AUSTRALIAN MATHEMATICAL SCIENCES INSTITUTE

Australian Government
Department of Education and Training

Contents

1 Introduction **2**

2 Airfoil designs **2**

2.1 NACA 2

2.2 Airfoil Parameterisation 4

3 Simulations in SU2 **5**

4 Full Grids and Sparse Grids: Visualisation, construction and interpolation **6**

4.1 Full Grids 6

4.2 Sparse Grids 6

4.3 Construction of Sparse grids (by example) 8

4.4 Sparse grid interpolation (by example) 10

5 Example CFD simulations **11**

5.1 Shape optimisation 12

1 Introduction

Consider a function $u : [0,1]^d \to \mathbb{R}$ that takes values called shape design parameters (sdps), which define an airfoil design $(\xi_1, \ldots, \xi_d)$, we want to know how different choices of these sdps affect the result in the calculation of drag $u(\xi_1, \ldots, \xi_d)$. The way of making these choices of sdps involved applying a gridding procedure to the domain $[0,1]^d$. Different gridding procedures can involve varying degrees of accuracy in the result, this variety is achieved by specifying a different number of points, or utilising the underlying structure of the simulation function $u \mapsto u(\xi)$ to identify the highest weighing, most important points where we should perform the simulations, this is the topic of sparse grids and is what is briefly explored in this report.

2 Airfoil designs

In order to specify a sdp for each airfoil, an initial shape must be chosen which corresponds to $(\xi_1, \ldots, \xi_d) = (0, \ldots, 0)$. To meet this end, I explored aircraft design literature ([2], [3], [4]) for a simple standard to work with. It seemed that there were a few standards floating around, and the NACA 4-digit design standard was the one I focused on.

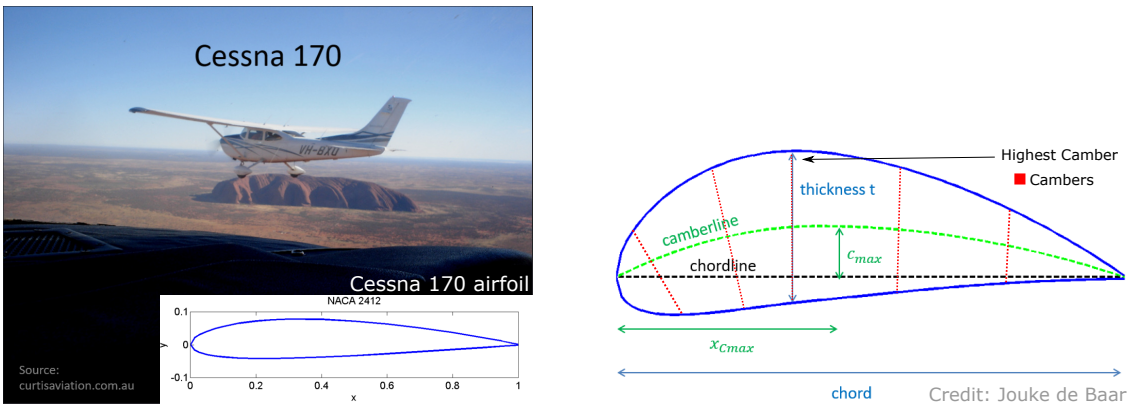
2.1 NACA

The NACA standard has all of its geometry specified by the four digits M, P and XX

- M is the length of the highest camber as a percentage of the chord
- P is the position of the highest camber as a tenth of a percent of the chord
- XX is the section thickness as a percentage of the chord

To put airfoils in context, Figure 1 includes a Cessna 170 airplane and how the NACA airfoil are a 2D slice of the much bigger 3D wing. The values for the Cessna airfoil are $M = 2$, $P = 4$, $XX = 12$. This means that the maximum camber is 2% of the chord length, and occurs at 4/10ths of the way along the chord (from the rounded leading edge).

Parametric equations can be derived for the NACA airfoil [7], this allows us to specify (x, y) coordinates for our initial foil. In Figure 2, we can see that from $[0, P]$ and $[P, 1]$ (Assuming a chord length of $C = 1$ to simplify) it would be easier to specify a piecewise function to get the points in each



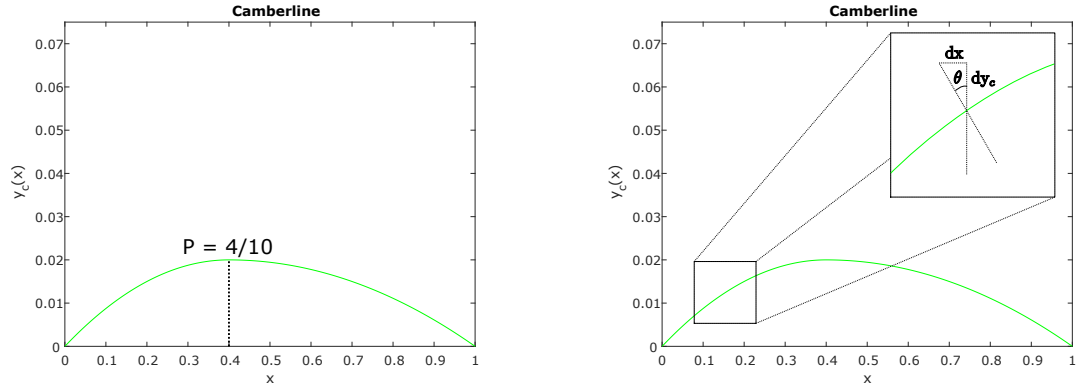
(a) Cessna 170 airplane with defining 2412 airfoil (b) NACA diagram

Figure 1: NACA airfoils

domain. If we take

$$y_c = \begin{cases} \frac{M}{P^2}\left(2Px - x^2\right), & 0 \le x < P \\ \frac{M}{(1-P)^2}\left(1 - 2P + 2Px - x^2\right), & P \le x \le 1 \end{cases}$$

we get the appropriate geometry.



(a) Camberline of the NACA2412 (b) Finding perpendicular

Figure 2: Camberline definition and thickness appropriation

The next step is to add thickness, the way that this can be done is by adding the required thickness amount in the direction perpendicular to the camberline, as the camber makes a non-zero angle (which is changing for each x) we must specify the angle θ, as in Figure 2 (b), then by basic trig, we have that

$$\tan\theta = \frac{dy_c}{dx} \implies \theta = \arctan\left(\frac{dy_c}{dx}\right)$$

we can easily take the derivative of y_c to obtain

$$\frac{dy_c}{dx} = \begin{cases} \frac{M}{P^2}\left(2P - 2x\right), & 0 \leq x < P \\ \frac{M}{(1-P)^2}\left(2P - 2x\right), & P \leq x \leq 1 \end{cases}$$

If we want to add the thickness

$$y_t = \frac{XX}{0.2}\left(a_0 x^{1/2} + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4\right)$$

along the perpendicular of the camberline, we just take the right trigonometric component, so the parametric equations are thus given by

Upper Surface	**Lower Surface**
$x_u = x - y_t \sin\theta,$	$x_l = x + y_t \sin\theta$
$y_u = y_c + y_t \cos\theta,$	$y_l = y_c - y_t \cos\theta$

The resultant airfoil is shown in Figure 3 and the colors are consistent with the defining equations above, where green represents the camberline, blue represents the upper surface and red represents the lower surface

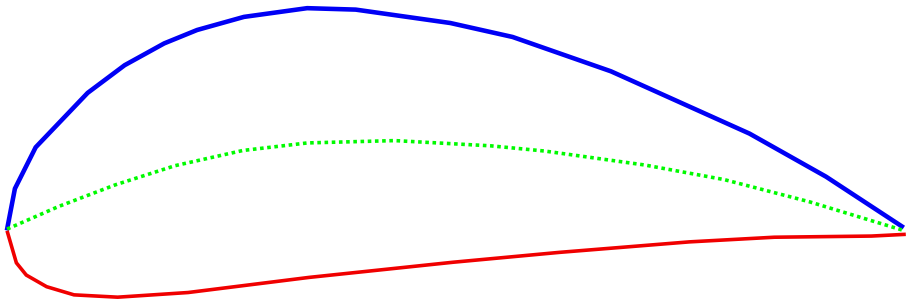


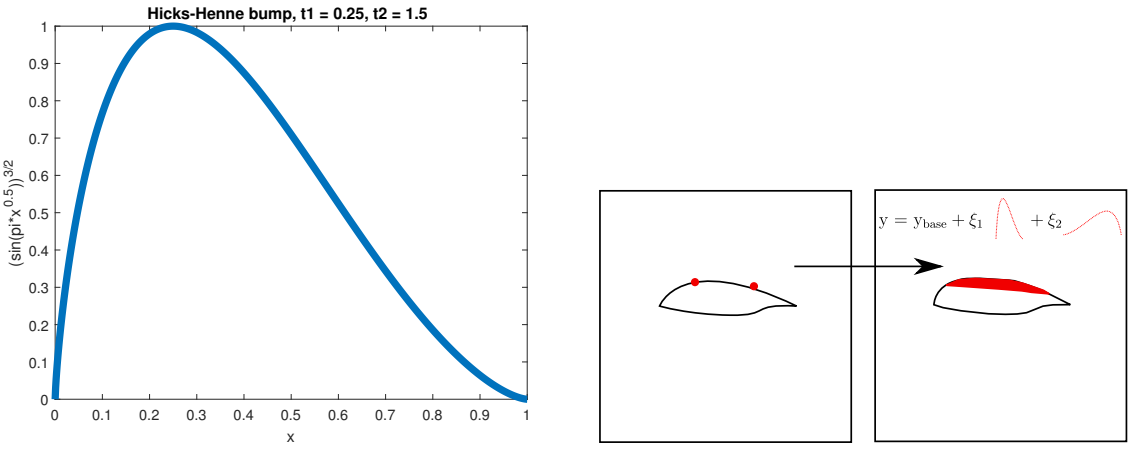
Figure 3: NACA airfoil demonstration

2.2 Airfoil Parameterisation

In order to perform multiple simulations, it is necessary to slightly modify the airfoil shape so subsequent calculations can be performed. This can be done by adding some error terms to the base airfoil shape y_{base} as

$$y = y_{\text{base}} + \sum_{i}^{d} \xi_i f_i(x)$$

The specific type of function I chose was based on the literature ([8], [9]), it is called the Hicks-Henne bump function, defined by $f(x) = \left[\sin\left(\pi x^{\ln(0.5)/\ln(t_1)}\right)\right]^{t_2}$.



(a) Hicks-Henne Bump example

(b) Shape transformation example

Figure 4

Figure 4 shows how a Hicks-Henne bump (a) can be specified by values of t_1 and t_2 to produce the shape transformation as visualised in (b). One may guess that these bump functions are utilised as they are smooth, sine like functions which do not introduce discontinuities when added onto the base shape.

3 Simulations in SU2

There are two main components of the SU2 CFD package [1] which are utilised in the performing of multiple simulations

- SU2_CFD solves PDEs to give quantities of interest such as drag, and pressure distributions
- SU2_DEF takes in an initial mesh configuration and parameters to produce a new mesh

In this case, the Euler equations mentioned are solved by SU2_CFD, I did not explore any of the numerical solvers in this method and essentially used it as a black box to output quantities of interest.

Figure 5 shows how a particular shape design in the parameter space can change the base shape y_{base}, then SU2_DEF can give us a mesh configuration which we can pass to SU2_CFD to solve the Euler equations on. This results in the collection of quantities such as drag and lift.

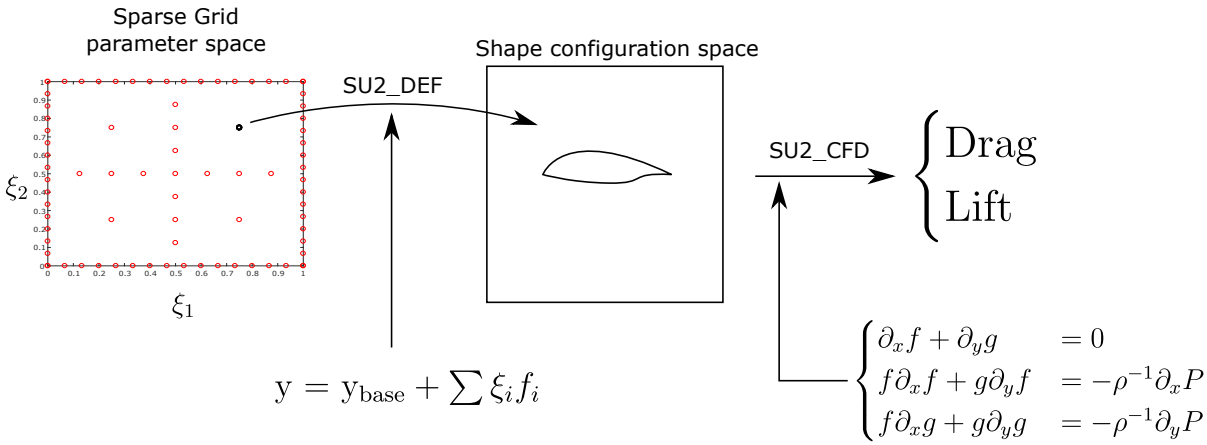


Figure 5: Simulation in SU2

4 Full Grids and Sparse Grids: Visualisation, construction and interpolation

It is important to only perform simulations when they are absolutely necessary, and to avoid ones which have a negligible contribution to the overall result [10]. This is because multi-dimensional problems can be exceedingly expensive to solve on a computer, particularly if each point represents a solving of multiple PDEs. When there is necessity to perform multiple simulations over a variety of parameters, it becomes useful to analyse different ways of selecting points, this is what this section is about.

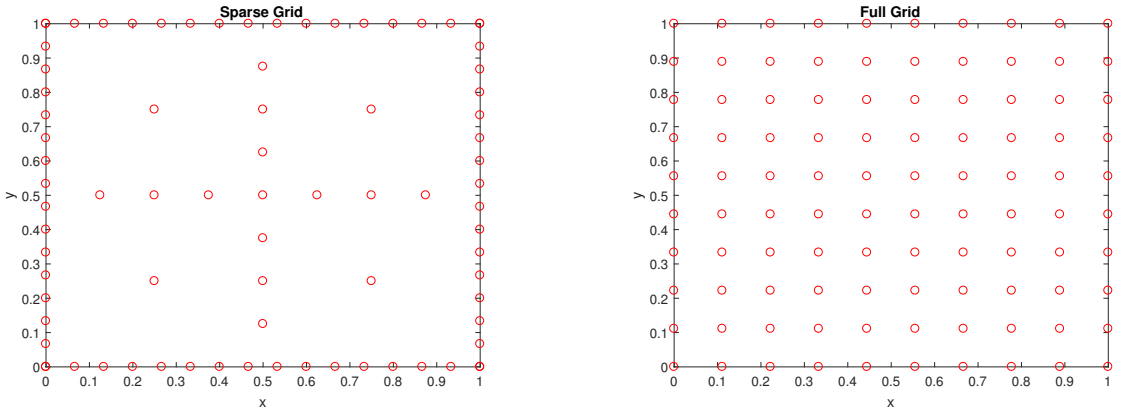
4.1 Full Grids

The key feature of full grids is that they contain an equal number of N points in each dimension, which is tensored to form a lattice of N^d points, as shown in Figure 6 (b). Applying this method generally gives the most accurate result for a set of simulations, given that it treats each point of simulation identically. However, we run into issues with computational expense at higher dimensions. We have a guaranteed accuracy up to N^{-2} given that $\left|\partial_{x_i}\partial_{x_j}u\right|$ is bounded for all $i, j = 1, \ldots, d$ [6].

4.2 Sparse Grids

We can choose points in a specific way in order to preserve accuracy. Given the stronger smoothness condition that $\left|\frac{\partial^{2d}u}{\partial x_1^2 \cdots \partial x_d^2}\right|$ is bounded, we have accuracy up to at least $N^{-2}\ln(N)^{d-1}$ in the sparse grid [6]. A sparse grid structure is shown in Figure 6 (a). As we can see, the error is dependent on the dimension of the problem, but due to the nature of the natural log, this term only becomes an issue

at much larger dimensions, partially eliminating the 'curse of dimensionality'.



(a) Example of a Sparse Grid in $2D$

(b) Example of a Full Grid in $2D$

Figure 6

In contrast to the standard N^d points present in a full grid, a sparse grid contains $\mathcal{O}\left(N\ln(N)^{d-1}\right)$ points. In order to see the power of sparse grids consider the example in the red circle in Figure 7. In this case, $d = 5$, $N = 5$

- $\mathcal{O}(N\ln(N)^{d-1}) = 61$ sparse grid pts, $N^d = 3125$ full grid pts
- $(5^{-2}\ln(5)^4 \approx 0.27)$ 27% $\implies$ interpolation error
- using only $(61/3125 \approx 0.02)$ 2% of points
- Compared to the standard 4% full grid error
- **Take home message: Accuracy of at least** 73% **by using only** 2% **points** (Worst case scenario)

Although in applications an accuracy of 73% is hardly ever a good accuracy to attain, it provides insight into how using the right type of evaluations, we can significantly reduce the number of points needed. Time permitted, it would be interesting to explore how further adjustments can be made, by using derivative or supplementary conditions, to further increase the accuracy. A quick result to note however is that the derivative condition stated above for the sparse grid case can be strengthened to obtain better bounds [6]

$$\left\|\partial_{x_1}^r \cdots \partial_{x_d}^r u\right\| < \infty \implies \|u_{SG}^r - u\| = \mathcal{O}\left(N^{-r}\ln^{d-1}(N)\right)$$

d	N	Sparse grid	Full grid
2	5	13	25
	9	29	81
5	5	61	3125
	9	241	59,049
10	5	221	9,765,625
	9	1581	$> 3 \times 10^9$
50	5	5101	$> 8 \times 10^{34}$
	9	171,901	$> 5 \times 10^{47}$
100	5	20,201	$> 7 \times 10^{69}$
	9	1,353,801	$> 2 \times 10^{95}$

Figure 7: Comparison of SPGD and FGD points [5]

4.3 Construction of Sparse grids (by example)

In the context of interpolating functions, a method called 'hierarchical' construction can be used to produce the sparse grid structure. Recall that any function can be approximated as $f(x) \approx \sum_i w_i \phi_i(x)$, where ϕ_i are basis functions and w_i the appropriate weights. The sparse grid method requires that the weights w_i are strictly decreasing in value, this is where the nomenclature of 'hierarchical' comes in. It works by essentially discarding relatively low weighing basis elements to reduce computational expense.

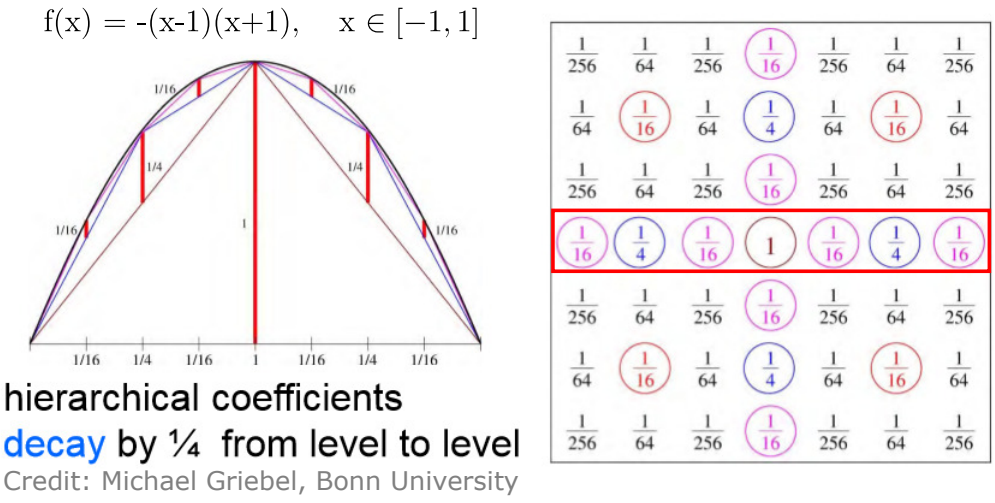
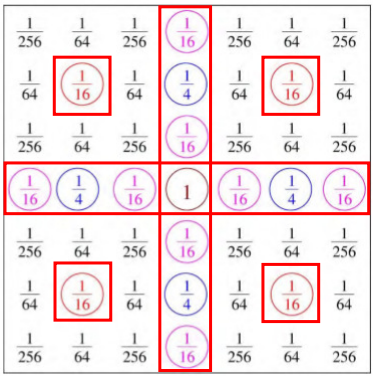


Figure 8: Choosing coefficients

In the red strip in the table of coefficients in Figure 8 represent the approximation of the function. This structure can be generalised to 2D, as seen in Figure 9 (a), as we can see, there are coefficients of

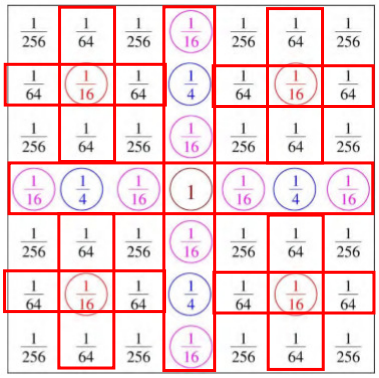
$\frac{1}{16}$ which are present in a more complicated capacity than the 1D example. Note that in this image, the 2D function has been left out as the representation gets too messy.

In order to provide a finer structure to the sparse grid, a notion of level can be introduced, by specifying coefficients of lower value. Figure 9 (b) shows this procedure of increasing the level.



Credit: Michael Griebel, Bonn University

(a) Level 1



Credit: Michael Griebel, Bonn University

(b) Level 2

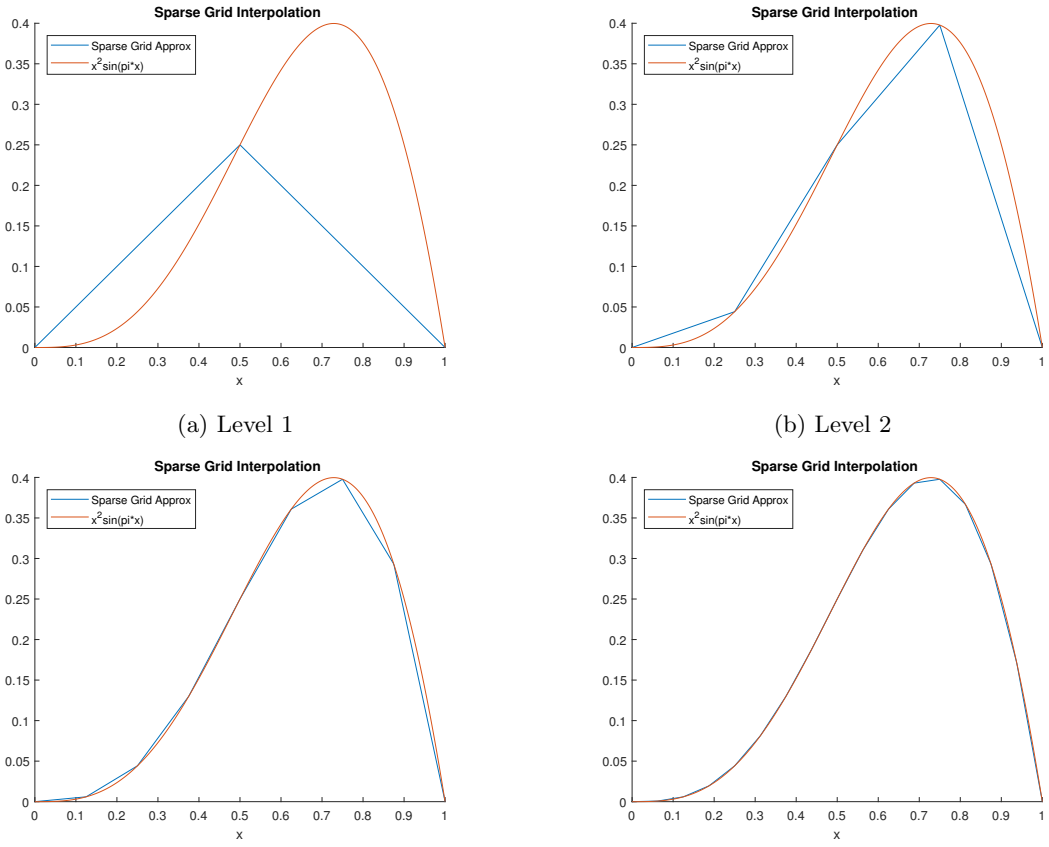
Figure 9

4.4 Sparse grid interpolation (by example)

Consider the problem of interpolating a function $f(x) = x^2 \sin(\pi x)$ on the interval $[0, 1]$. Basis functions can be specified by contractions, dilations or translations of

$$\phi(x) = \begin{cases} 1 - |x|, & \text{if } -1 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

As we increase the level of the approximation to n, we are increasing the number of basis functions to $2n + 1$, Figure 10 shows how accurate the approximation is after just 4 increases in level.



(a) Level 1

(b) Level 2

(c) Level 3

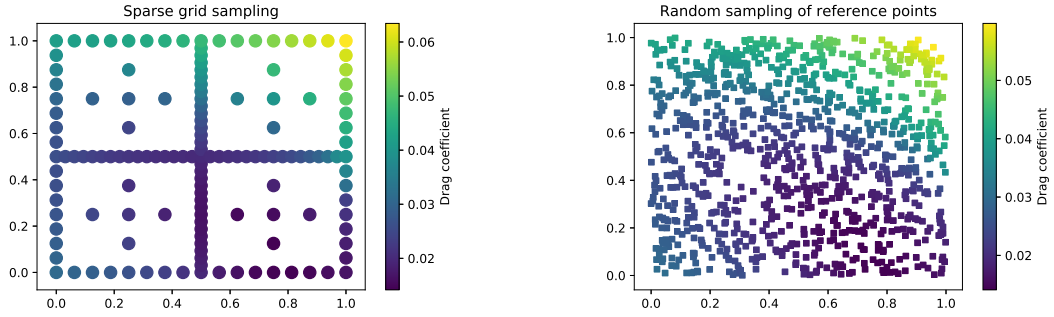
(d) Level 4

Figure 10

5 Example CFD simulations

Since the sparse grid structure applies only to functions which have sufficient smoothness conditions, it is important to have a plausibility argument for why sparse grids would work in the context of airfoil simulations. When performing a CFD simulation on a particular shape profile, the drag can be obtained. Small perturbations of the shape can be applied and a new value of drag can be calculated. In this context, we would assume that the drag of the latter profile is sufficiently similar to the drag of the former. That is, it should be plausible to assume that small changes in the aircraft design correspond to small changes in the value of the drag coefficient. We would also need to argue that SU2 sufficiently approximates the real life fluid dynamics to say the same about our simulations. Although it is definitely a point of consideration, our situation of a 2D airfoil is simple enough situation that we should be able to assume consistency. Thus, we can assume that the underlying simulation function $u : [0,1]^d \to \mathbb{R}$ is a (smooth enough) function such that a sparse grid approximation $\tilde{u}^{\mathrm{SG}} : [0,1]^d \to \mathbb{R}$ would operate as a reasonable surrogate.

For the simulation example, we consider a 2 dimensional level 5 sparse grid. Drag coefficients were calculated at each point in the grid using SU2 to produce the results in Figure 11 (a), which has 145 points of evaluation. Meanwhile, drag coefficients were calculated for 1000 randomly sampled points which are used as a reference point to determine how well the sparse grid surrogate has performed.

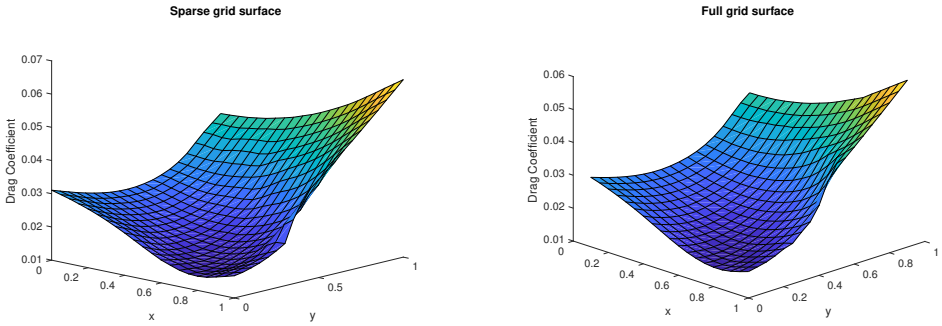


(a) Sparse Evaluations

(b) Reference Evaluations

Figure 11: Simulation results

The results in Figure 11 can be interpolated to produce the surfaces in Figure 12. As we can see, the surfaces are very similiar, which indicates that the sparse grid method has performed quite well for this example problem.



(a) Sparse Grid Surrogate (b) Full Interpolation

Figure 12: Interpolated surfaces

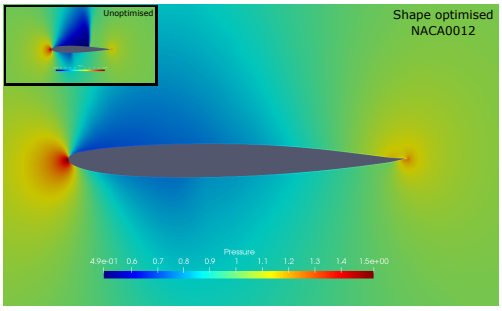


Figure 13: Shape optimisation

5.1 Shape optimisation

Shape optimisation is the method of solving the minimisation

$$\min_{\xi \in [0,1]^2} u(\xi)$$

However, with the calculation of a sparse grid surrogate $\tilde{u}^{\mathrm{SG}}$ the optimisation problem can be approximated by

$$\min_{\xi \in [0,1]^2} u(\xi) \approx \min_{\xi \in [0,1]^2} \tilde{u}^{\mathrm{SG}}(\xi)$$

Using this formulation, it is possible to retrieve the shape parameters ξ, allowing for the reconstruction of the optimal shape. Unfortunately, I didnt have enough time to implement this. Fortunately, howeer, SU2 has built in shape optimisation which essentially does it over a full grid as in 11 (b). Using the built in optimisation, I considered the unoptimised NACA0012 airfoil in the top left hand of the image, as seen in 13. Before optimising, there is a huge pressure shock as indicated in blue, the colormap in the subsequent optimisation image indicates that the pressure shock has been removed.

References

[1] SU2, Computational fluid dynamics package
https://su2code.github.io/

[2] NASA: Armstrong Express, 100 years of aircraft design,
https://www.nasa.gov/sites/default/files/files/NACA_100_X-Press.pdf

[3] Aerodynamics, Aircraft Design, University of Liege,
http://www.ltas-cm3.ulg.ac.be/AERO0023-1/ConceptionAeroAerodynamisme2015.pdf

[4] Wing design - selection of wing parameters
https://nptel.ac.in/courses/101106035/026_Chapter%205%20_L19_(03-10-2013).pdf

[5] Uncertainty Quantification: Theory, Implementation, and Applications, Ralph C Smith
http://bookstore.siam.org/cs12/

[6] Sparse Grid Methods for Uncertainty Quantification, Michael Griebel
https://archive.siam.org/meetings/uq16/griebel.pdf

[7] Explained: NACA 4-Digit Airfoil [Airplanes], JoshTheEngineer
https://www.youtube.com/watch?v=6pt8Uolfj0M

[8] Choice of design variables
http://www.ims.nus.edu.sg/Programs/wbfst/files/siva3.pdf

[9] Dominic A. Masters, Nigel J. Taylor, T. Rendall, Christian B. Allen, and Daniel J. Poole. "Review of Aerofoil Parameterisation Methods for Aerodynamic Shape Optimisation", 53rd AIAA Aerospace Sciences Meeting, AIAA SciTech Forum, (AIAA 2015-0761)

[10] Sparse Grids in a Nutshell, Jocken Garcke,
https://ins.uni-bonn.de/media/public/publication-media/sparse_grids_nutshell.pdf?pk=639