

RICE UNIVERSITY

**Calibration of Flush Air Data Sensing Systems Using  
Surrogate Modeling Techniques**

by

**Ankur Srivastava**

A THESIS SUBMITTED  
IN PARTIAL FULFILLMENT OF THE  
REQUIREMENTS FOR THE DEGREE

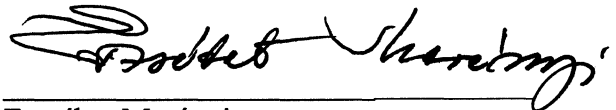
**Doctor of Philosophy**

APPROVED, THESIS COMMITTEE:



---

Andrew J. Meade, Jr., Committee Chair  
Professor of Mechanical Engineering &  
Materials Science



---

Erzsébet Merényi  
Research Professor of Electrical and Computer  
Engineering



---

Brent C. Houchens  
Assistant Professor of Mechanical Engineering &  
Materials Science

HOUSTON, TEXAS  
NOVEMBER, 2010

# ABSTRACT

Calibration of Flush Air Data Sensing Systems Using Surrogate Modeling Techniques

by

Ankur Srivastava

In this work the problem of calibrating Flush Air Data Sensing (FADS) has been addressed. The inverse problem of extracting freestream wind speed and angle of attack from pressure measurements has been solved. The aim of this work was to develop machine learning and statistical tools to optimize design and calibration of FADS systems. Experimental and Computational Fluid Dynamics (EFD and CFD) solve the forward problem of determining the pressure distribution given the wind velocity profile and bluff body geometry. In this work three ways are presented in which machine learning techniques can improve calibration of FADS systems.

First, a scattered data approximation scheme, called Sequential Function Approximation (SFA) that successfully solved the current inverse problem was developed. The proposed scheme is a greedy and self-adaptive technique that constructs reliable and robust estimates without any user-interaction. Wind speed and direction prediction algorithms were developed for two FADS problems. One where pressure sensors are installed on a surface vessel and the other where sensors are installed on the Runway Assisted Landing Site (RALS) control tower.

Second, a Tikhonov regularization based data-model fusion technique with SFA was developed to fuse low fidelity CFD solutions with noisy and sparse wind tunnel data. The purpose of this data model fusion approach was to obtain high fidelity, smooth and noiseless flow field solutions by using only a few discrete experimental measurements and a low fidelity numerical solution. This physics based regularization technique gave better flow field solutions compared to smoothness based solutions when wind tunnel data is sparse and incomplete.

Third, a sequential design strategy was developed with SFA using Active Learning techniques from the machine learning theory and Optimal Design of Experiments from statistics for regression and classification problems. Uncertainty Sampling was used with SFA to demonstrate the effectiveness of active learning versus passive learning on a cavity flow classification problem. A sequential G-optimal design procedure was also developed with SFA for regression problems. The effectiveness of this approach was demonstrated on a simulated problem and the above mentioned FADS problem.

## **Acknowledgements**

First and foremost I would like to offer my deepest regards and sincere gratitude towards my advisor Dr. Andrew J. Meade for guiding me throughout my graduate studies. His authority on this subject, kind nature and friendly supervision were of great assistance to me. Special regards go to Dr. Erzsébet Merényi for giving valuable suggestions towards making this work more complete and for serving in my thesis committee. I am profoundly thankful to Dr. Brent C. Houchens for serving as a member in my thesis committee and providing thoughtful insights during this work. I am also thankful to Kurt Long and Ali Mokhtarzadeh for providing the wind tunnel data and CAD model for this work.

In my daily work I have been blessed with a group of friendly and cheerful fellow students and friends. Their company has been lovely and they formed an integral part in making my graduate life pleasant. I would like to thank all the department staff members especially Aya Ossowski, Alicia Contreras and Cicely Ortolani for providing a nice environment to work and for all their help in finishing up the required paperwork. I would also like to thank Rice Fondren library for providing the necessary books and papers. My sincere thanks go to Rice Coffee House and Rice Recreation Center which kept me going throughout my entire stay at Rice. It would be unfair not to thank Google and Wikipedia for their immediate help in refreshing old concepts and learning new things. Finally, I would like to thank my parents and GOD for their blessings without which nothing would have been possible.

# Contents

|  |           |
|--|-----------|
| Abstract.....  | ii        |
| Acknowledgements.....  | iv        |
| List of Figures.....   | ix        |
| List of Tables.....  | xv        |
| Nomenclature.....  | xvi       |
| <br>   |           |
| <b>1. Introduction .....</b>   | <b>1</b>  |
| 1.1 Challenges in Flush Air Data Sensing .....                                   | 3         |
| 1.2 Intersection of Machine Learning and Fluid Dynamics .....                    | 5         |
| 1.2.1 Approximation of physical relations and simulation of flow fields .....    | 6         |
| 1.2.2 Flow optimization and control.....   | 7         |
| 1.2.3 Hybrid modeling with numerical simulations .....                           | 8         |
| 1.2.4 Hybrid design with numerical simulations and optimization techniques ..... | 9         |
| 1.3 Need for the Current Research in ANN.....                                    | 9         |
| 1.3.1 Formalization of the final network .....                                   | 9         |
| 1.3.2 Choice of training data.....   | 10        |
| 1.3.3 Assimilating physical knowledge of the system.....                         | 11        |
| 1.4 Proposed Ideas and Objectives .....  | 12        |
| <br>   |           |
| <b>2. Greedy Function Approximation .....</b>                                    | <b>15</b> |
| 2.1 Greedy Algorithms .....  | 17        |
| 2.1.1 Matching pursuit.....  | 18        |

|  |           |
|--|-----------|
| 2.1.2 Sparse approximation methods.....                | 21        |
| 2.1.3 Other related methods.....                       | 22        |
| 2.2 Sequential Function Approximation.....             | 23        |
| 2.3 Characteristics of SFA.....                        | 27        |
| <b>3. Flush Air Data Sensing Systems.....</b>          | <b>29</b> |
| 3.1 Wind Tunnel Data.....                              | 30        |
| 3.1.1 Surface vessel.....                              | 31        |
| 3.1.2 RALS tower.....                                  | 34        |
| 3.2 Forward and Inverse Problems.....                  | 37        |
| 3.2.1 Dynamic pressure.....                            | 37        |
| 3.2.2 Wind direction.....                              | 39        |
| 3.3 Results.....                                       | 42        |
| 3.3.1 RALS tower.....                                  | 42        |
| 3.3.2 Surface Vessel.....                              | 47        |
| 3.4 Pressure Sensor Sensitivity.....                   | 51        |
| 3.4.1. Graceful degradation.....                       | 54        |
| <b>4. Data-Model Fusion.....</b>                       | <b>56</b> |
| 4.1 Fusion of variable fidelity models.....            | 60        |
| 4.1.1 Relation to Data Assimilation.....               | 65        |
| 4.2 Tikhonov regularization and data-model fusion..... | 68        |
| 4.2.1 Data-model fusion for FADS.....                  | 70        |
| 4.2.2 Handling noise, sparsity and incompleteness..... | 72        |

|   |            |
|---|------------|
| 4.2.3 Fusion in the pressure sensor position $\theta$ .....         | 76         |
| 4.3 Wind speed and direction prediction .....                       | 79         |
| <b>5. Sequential Experiment Design .....</b>                        | <b>84</b>  |
| 5.1 Active Learning for regression problems .....                   | 86         |
| 5.2 Implementation issues and application .....                     | 90         |
| 5.2.1 Singularity of Fisher Information Matrix .....                | 91         |
| 5.2.2 Overparameterization.....                                     | 92         |
| 5.2.3 Initial number of training points and stopping criteria ..... | 94         |
| 5.2.4 RBF center selection.....                                     | 94         |
| 5.2.5 FIM parameter selection .....                                 | 96         |
| 5.2.6 The bias problem .....  | 97         |
| 5.2.7 Grid size of candidate sampling pool .....                    | 98         |
| 5.2.8 Examples.....   | 99         |
| 5.2.8.1 Peaks problem.....  | 99         |
| 5.3 Sequential Experiment Design for FADS.....                      | 105        |
| <b>6. Conclusions and Future Work .....</b>                         | <b>110</b> |
| 6.1. Future Work.....   | 112        |
| <b>References.....</b>  | <b>113</b> |
| <b>Appendix A .....</b>   | <b>125</b> |
| A.1 Inverse approach to predict yaw angle.....                      | 125        |
| <b>Appendix B.....</b>  | <b>135</b> |

|   |     |
|---|-----|
| B.1 Active Learning for Classification Problems.....                  | 135 |
| B.1.1 Optimization Approaches.....                                    | 138 |
| B.2 Uncertainty Sampling with SFA.....                                | 140 |
| B.3 Wind tunnel experiment design for cavity flow classification..... | 143 |



## List of Figures

Figure 1.1 Flush air data system installation at the nose of the F-18 Systems Research Aircraft.

Figure 1.2 Two-dimensional flow about a right circular cylinder.

Figure 1.3 Machine learning tools can provide a framework in which EFD and CFD can be integrated in a principled manner.

Figure 2.1 An optimum basis function is one that is the most aligned with  $\vec{r}_{n-1}$ .

Figure 3.1 Figure shows a) Surface vessel and b) RALS tower. Both geometries require pressure sensors to be installed all around the surface to account for cross-flow.

Figure 3.2 Model Deckhouse Pressure Port Locations (Top View).

Figure 3.3 Direction of bow, port, stern and starboard side winds on the schematic top view of the surface vessel.

Figure 3.4 Static pressure variation vs. the yaw angle for different wind speeds.

Figure 3.5 Coefficient of pressure variation for (a) bottom ring ports and (b) middle ring ports versus the yaw angle at 165 fps freestream wind speed.

Figure 3.6 Wind tunnel model of the RALS tower.

Figure 3.7 Static pressure variation vs. the yaw angle for different wind speeds.

Figure 3.8 Pressure coefficient variation vs. the yaw angle for different wind speeds.

Figure 3.9 Dynamic pressure hyper-cone represented as a function of a) two pressure sensors and b) one pressure sensor.

Figure 3.10 Non-uniqueness of the yaw angle prediction problem at a) 120 fps and b) 40fps (RALS tower data).

Figure 3.11 a) Wind speed prediction on RALS tower data using just one network for wind speed given by Eq. (3.1) and b) Example surrogate of the wind speed hyper-cone in 2 dimensions.

Figure 3.12 a) Logarithm of discrete inner product norm of residual error and b) Wind speed prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle.

Figure 3.13 a) Residual error convergence and b)  $C_p$  prediction by a model of each pressure sensor. Blue dots represent wind tunnel data and the red line represents approximation by SFA as a function of the yaw angle. Clockwise from top plots show  $C_p$  model of pressure sensor at 0, 90, -90 and 180 degrees.

Figure 3.14 a) Logarithm of objective function of Eq. (3.3) obtained after a line search conducted for a test point with a yaw angle of 0 degrees and b) Yaw angle prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle using the forward approach.

Figure 3.15 a) Logarithm of discrete inner product norm of residual error and b) Wind speed prediction errors on the surface vessel data when training was conducted on data at every 4 degree increments of yaw angle.

Figure 3.16 Wind speed prediction on the surface vessel data using one network for wind speed given by Eq. (3.1) with fictitious points at higher speeds added to the training set.

Figure 3.17 Yaw angle prediction errors for the surface vessel problem using the forward approach.

Figure 3.18 Average input sensitivities in the prediction of (a) wind speed and (b) ship bow yaw angle.

Figure 3.19 Logarithm of prediction error degradation with the number of pressure taps.

The bars show the standard deviation of the errors.

Figure 3.20 Logarithm of prediction error degradation with the number of training points.

The bars show the standard deviation of the prediction errors.

Figure 4.1 Concept of a hybrid wind tunnel.

Figure 4.2 The drag bucket for SC1095 at  $Re = 11.16 \times 10^6$ ,  $M = 0.90$ .

Figure 4.3  $C_l$  vs.  $\alpha$  for NACA 0012 interpolation test comparisons.  $Re = 3.72 \times 10^6$ ,  $M = 0.30$ .

Figure 4.4 Comparison of bow side a) smoothness and b) physics based solutions to handle sparse and noisy pressure data at 120 fps.

Fig. 4.5 Comparison of bow side a) smoothness and b) physics based solutions to handle sparse and noisy pressure data at 40 fps.

Figure 4.6 Comparison of a) starboard side smoothness, b) starboard side physics, c) stern side smoothness and d) stern side physics e) port side smoothness and f) port side physics based solutions to handle sparse and noisy pressure data at 40 fps.

Figure 4.7 Wind tunnel coefficient of pressure as a function of yaw angle and pressure sensor position.

Figure 4.8 CFD coefficient of pressure as a function of yaw angle and pressure sensor position.

Figure 4.9 CFD and fused coefficient of pressure as a function of yaw angle and pressure sensor position at a location next to a) Starboard b) Stern c) Port and d) Bow side sensor.

At these locations wind tunnel measurements were not taken.

Figure 4.10 Prediction accuracies of a) wind speed and b) yaw angle for noise-free data at a yaw angle resolution of 4 degrees.

Figure 4.11 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using smoothness based regularization technique to obtain training data.

Figure 4.12 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using physics based regularization technique to obtain training data.

Figure 4.13 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using physics based regularization technique to obtain training data. All 56 fused pressure signals were used to construct the training set.

Figure 5.1 a) An overparameterized 1-D regression problem and b) Singularity in the standardized variance of an overparameterized surrogate model.

Figure 5.2 a) Surrogate model by SFA using RBF center selection heuristic and b) Erroneous standardized variance of the model output.

Figure 5.3 a) Surrogate model by SFA when RBF centers were optimized, and Standardized variance of the output when RBF center parameters were b) included and c) not included in the construction of the FIM.

Figure 5.4 Matlab peaks function

Figure 5.5 Reduction of mean squared error with increasing number of training points. Bars show standard error computed over 10 repetitions. The red curve corresponds to passive LHS, the black curve corresponds to the G-optimal design and the blue curve

represents the best possible result if the true function is known a priori. The blue curve is plotted just to put the performance of variance only active learning scheme in perspective.

Fig. 5.6 Reduction of the maximum standardized variance with increasing number of training points. Bars show standard error computed over 10 repetitions.

Fig. 5.7 a) Training points chosen by the G-optimal design procedure and b) two dimensional view of the *peaks* function.

Fig. 5.8 a) Comparison of mean squared error, and b) Training points chosen by the G-optimal design procedure over the *peaks* problem with a larger domain.

Figure 5.9 Reduction of  $C_p$  mean squared error with increasing number of training points. Bars show standard error.

Figure 5.10  $C_p^2$  shown as a function pressure sensor position and yaw angle.

Figure 5.11 Reduction of  $C_p^2$  mean squared error with increasing number of training points. Bars show standard error.

Figure 5.12 Training points chosen by the G-optimal design procedure.

Figure A.1 Quadrants depicting different ranges of the yaw angle for (a) Surface vessel and (b) RALS tower.

Figure A.2 Pressure coefficient variation vs. the yaw angle for selected ranges of angle of attack.

Figure A.3 a) Logarithm of discrete inner product norm of residual error for each quadrant and b) Yaw angle prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle.

Figure A.4 a) Logarithm of discrete inner product norm of residual error for each quadrant and b) Yaw angle prediction errors on the surface vessel data when training was conducted on data at every 4 degree increments of yaw angle.

Figure B.1 a) Mean percentage classification accuracy as number of training points increase, errorbars show standard error and b) Crosses represent the training points chosen by the active learning scheme and solid line is the class discrimination boundary  $y = x^2$ .

Figure B.2 a) Open bomb bay of a F-22 Raptor and b) open missile bay of a F-35 Lightning.

Figure B.3 Occurrence of resonance superimposed with observed cavity flow conditions.

Figure B.4. Schematic representation of Active Learning.

Figure B.5. Mean cavity flow type prediction accuracy of active and passive learning at a)  $w/h = 1$ , b)  $w/h = 4$  and c)  $w/h = 8$ . Error bars show the standard error.

## **List of Tables**

Table 2.1: Specifications of the data sets used

Table A.1 Distribution of pressure taps for the surface vessel problem

Table B.1 Comparison of classification accuracies.

## Nomenclature

|       |   |   |
|-------|---|---|
| $a$   | = | parameters of the softmax function  |
| $A$   | = | arbitrary rectangular matrix  |
| $b$   | = | bias term   |
| $B$   | = | error covariance matrix of the background                                 |
| $c_n$ | = | $n^{\text{th}}$ linear coefficient in the approximation                   |
| $C_p$ | = | coefficient of pressure   |
| $C$   | = | arbitrary constant  |
| $d$   | = | number of input dimensions  |
| $e$   | = | difference between CFD and wind tunnel data                               |
| $E$   | = | expected error  |
| $f$   | = | arbitrary function  |
| $F$   | = | Fisher information matrix   |
| $g$   | = | product of the coefficient and the bias term at the $n^{\text{th}}$ stage |
| $G$   | = | inverse operator  |
| $h$   | = | Euclidean distance  |
| $H$   | = | Sobolev space   |
| $I$   | = | book-keeping matrix   |
| $j^*$ | = | component index of $\vec{r}_n$ with the maximum absolute magnitude        |
| $J_b$ | = | discrepancy of the analysis with the background                           |
| $J_o$ | = | discrepancy of the analysis with the observation                          |



|             |   |   |
|-------------|---|---|
| $k$         | = | partial derivative of the approximation with its parameters         |
| $K$         | = | function of error covariance matrices of background and observation |
| $l$         | = | number of parameters in the approximation                           |
| $m$         | = | number of output dimensions   |
| $M$         | = | Mach number   |
| $n$         | = | number of basis functions   |
| $N$         | = | number of classes   |
| $O$         | = | radius of influence   |
| $p$         | = | static pressure   |
| $P$         | = | probability operator  |
| $q$         | = | dynamic pressure  |
| $Q$         | = | environment probability   |
| $\vec{r}_n$ | = | $n^{\text{th}}$ stage function residual vector                      |
| $R$         | = | Real coordinate space   |
| $Re$        | = | Reynolds number   |
| $s$         | = | number of samples   |
| $T$         | = | error covariance matrix of the observation                          |
| $u$         | = | target function   |
| $v$         | = | arbitrary function  |
| $V$         | = | wind speed  |
| $w$         | = | weight  |

|                          |   |  |
|--------------------------|---|--|
| $x$                      | = | arbitrary variable   |
| $y$                      | = | arbitrary variable   |
| $z$                      | = | arbitrary variable   |
| $\langle f \rangle$      | = | inner product of $f$ with respect to one                             |
| $\langle f, v \rangle_D$ | = | $\sum_i^s (f_i v_i)$ = discrete inner product of $f$ and $v$         |
| $ f $                    | = | absolute value of $f$  |
| $\ f\ _2$                | = | $\left( \int_{\Omega} f^2 d\xi \right)^{1/2}$ = $L_2$ norm of $f$    |
| $\ f\ _{2,D}$            | = | $\left( \sum_i^s (f_i)^2 \right)^{1/2}$ = discrete $L_2$ norm of $f$ |

### Greek Symbols

|               |   |  |
|---------------|---|--|
| $\alpha$      | = | degree of smoothness                             |
| $\beta$       | = | yaw angle  |
| $\theta$      | = | flow incidence angle                             |
| $\rho$        | = | air density                                      |
| $\wp$         | = | set of kernel hyper-parameters                   |
| $\vartheta_n$ | = | angle between $\vec{\phi}_n$ and $\vec{r}_{n-1}$ |
| $\kappa$      | = | arbitrary constant                               |
| $\eta$        | = | model parameters                                 |
| $\Lambda$     | = | proxy function                                   |
| $\mu_n$       | = | basis shape changing parameter                   |

|               |   |  |
|---------------|---|--|
| $\Delta_n$    | = | average distance between basis centers                               |
| $\xi$         | = | d-dimensional input of the target function                           |
| $\xi_i$       | = | i <sup>th</sup> sample input of the target function                  |
| $\xi_n^*$     | = | sample input with component index $j^*$ at the n <sup>th</sup> stage |
| $\sigma$      | = | width parameter of Gaussian basis function                           |
| $\lambda$     | = | associated with the basis width                                      |
| $\psi$        | = | also associated with the basis width parameter                       |
| $\tau$        | = | user specified tolerance   |
| $\phi$        | = | basis function   |
| $\delta$      | = | input sensitivity  |
| $\Omega$      | = | regularizing functional constraint                                   |
| $\gamma$      | = | regularized objective function                                       |
| $\varepsilon$ | = | observational noise  |
| $\zeta$       | = | standardized variance  |
| $\omega$      | = | regularization parameter   |

### Subscripts

|       |   |   |
|-------|---|---|
| $i$   | = | dummy index   |
| $j^*$ | = | component index of $r_{(n-1)}$ with the maximum magnitude |
| $n$   | = | associated with the number of bases                       |
| $a$   | = | represents analysis                                       |

- $b$  = represents background
- $o$  = represents observation
- $s$  = associated with the number of samples

### **Superscripts**

- $d$  = number of input dimensions
- $a$  = approximation
- $\wedge$  = estimated quantity
- $-$  = mean of the vector
- $\infty$  = freestream condition
- $m$  = number of output dimensions

# Chapter 1

## Introduction

Intrusive aircraft anemometer booms have long been successfully used to measure air data parameters for subsonic and supersonic flight. Rapid development in aerospace and naval technology has continuously demanded smaller and more accurate air data systems. The performance of probe based air data systems suffer at high angles of attack or when dynamic maneuvering is required. These same conditions can lead to degraded flying handling qualities [1]. Errors due to vibration, poor alignment and physical damage during operation or maintenance can also pose significant limitations on the use of protruding booms. Probe based air data systems cannot be used with stealthy air or surface vessels because they increase the total radar cross section of the vessel which can potentially jeopardize the mission and risk human lives. Hypersonic flight regimes are another area where probe based air data systems cannot be used because the vehicle nose reaches extremely high temperatures that might melt the protruding boom. Also, such flow protruding booms cannot be used with research aircraft or surface vessels without disturbing the airflow or the boundary layer. Finally, air data booms are too heavy and costly to use with unmanned micro air vehicles (MAVs) [2]. Reference [2] mentions that an 80% decrease in instrumentation weight and a 97% decrease in instrumentation cost can be gained by eliminating probe based air data systems from MAVs.

In order to overcome the above mentioned drawbacks, Flush Air Data Sensing (FADS) systems were developed. A popular flush air data system is a series of pressure

taps mounted on a vehicle periphery that can be used to derive airflow parameters. FADS were first developed during the X-15 program. A hemispherical nose with pressure sensors was installed on this hypersonic aircraft to measure stagnation pressure and wind direction during re-entry [3]. Further research on FADS for hypersonic vehicles was conducted during the Space Shuttle program [4]. After enjoying success on hypersonic flights the compatibility of FADS were tested on supersonic and subsonic flight regimes. The authors of the reference [5] developed and flight tested a flush air data sensing system for the F-18 Systems Research Aircraft (Fig. 1.1). The authors concluded that the developed non-intrusive technique was clearly superior to the probe based air data systems. They showed that FADS were unaffected by dynamic flight maneuvers and they performed well in high angle of attack flights. FADS are also unaffected by vibration, icing effects and are less prone to damage during operation and maintenance. They are an ideal choice for stealth vessels because they are self-sustained and give no additional radar cross-sectional area to the vessel signature.

Other non-intrusive air data systems include flush mounted optical air data systems and flush mounted heat films. The optical air data systems transmit lasers or acoustic signals generated by electromechanical transducers or loud speakers through a fluid medium to one or multiple receivers and measure the travel time to infer air data information. Even though these systems are non-intrusive, they are not self-sustained and are expensive and heavy to use [6].

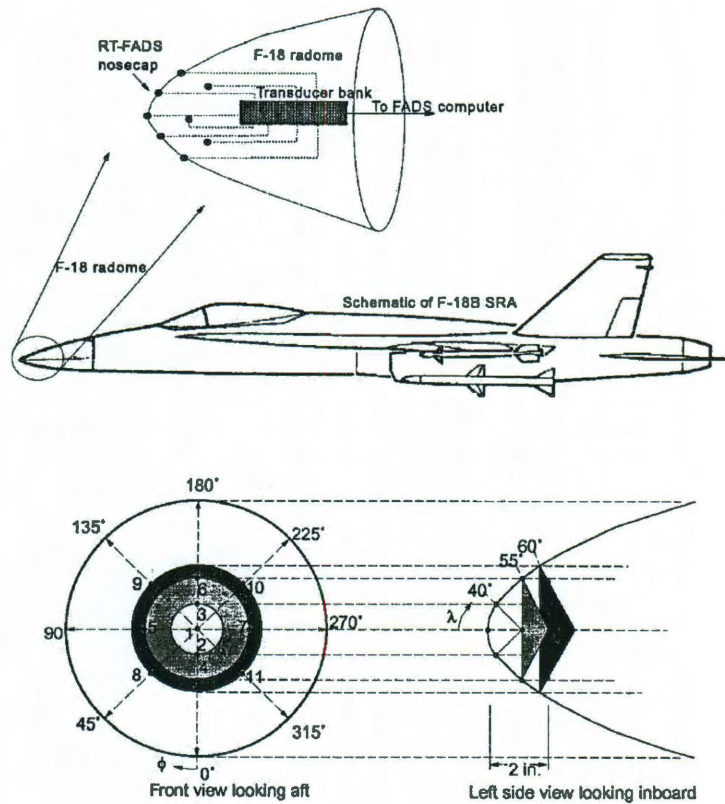


Figure 1.1 Flush air data system installation at the nose of the F-18 Systems Research Aircraft [1].

## 1.1 Challenges in Flush Air Data Sensing

As previously mentioned, FADS infer the air data parameters from pressure measurements taken with an array of ports that are flush to the surface of the aircraft. However, because the locations of the pressure measurements are on the outer surface of the aircraft on geometrically simple components (e.g., hemispherical or conical nose), properties of the local flow fields like compressibility and flow separation can drastically affect these devices. Unlike for aircraft, the FADS for bluff bodies like ships and buildings must be placed completely around the structure and can routinely experience

separated and unsteady cross-flow. Also, to compete with conventional air data systems, the FADS must be able to estimate wind speeds ( $V_\infty$ ) over a range of 40 to 120 fps,  $\pm 3.4$  fps (i.e.,  $\pm 2$  knots), and wind directions ( $\beta$ ) ranging from 0 to 360 degrees,  $\pm 2$  degrees. Finally, the accuracy of FADS should degrade gracefully with progressive sensor failures and allow the user to determine the sensitivity of the output to sensor placement.

Using the incompressible flow about a right circular cylinder as an example of a bluff body (Fig. 1.2), the problem of estimating relative wind speed and direction from the surface pressure at position  $\theta$  can be framed as either a forward or inverse mapping problem, i.e.,

$$p = F1(\rho_\infty, V_\infty, \theta, \beta) \} \text{ the forward problem}$$

$$\left. \begin{aligned} V_\infty &= G1(P, \rho_\infty, \theta, \beta) \\ \beta &= G2(P, \rho_\infty, V_\infty, \theta) \end{aligned} \right\} \text{ the inverse problem}$$

Here  $p$  is static pressure and  $\rho_\infty$  is freestream air density. For simple geometries, closed-form semi-empirical equations can be developed and used to solve the forward problem of estimating the static pressure coefficient,  $C_p$ , given the freestream wind speed and direction [7]. Panel methods [8] and other popular computational fluid dynamic (CFD) based approaches solve the forward problem on more complex geometries given the freestream wind speed and direction. However, this approach requires guessing the freestream velocity and solving for the pressure distribution until it matches the measured values. To solve the inverse problem, look-up tables from experimental fluid dynamics (EFD) can be constructed for the mapping function  $G$ . Unfortunately, look-up tables suffer from a number of drawbacks when their real-time use is considered including the



inability to handle high-dimensional inputs, noisy data, and nonlinear interpolation. It is proposed that machine learning methods, specifically Artificial Neural Networks (ANN), be explored to solve the inverse problem. In addition, this dissertation will demonstrate on how a particular form of ANN can also be used to determine the best locations for sensor placement, to maximize available FADS data by seamlessly combining CFD and EFD outputs, and to help guide physical experiments thereby minimizing the time needed in testing facilities.

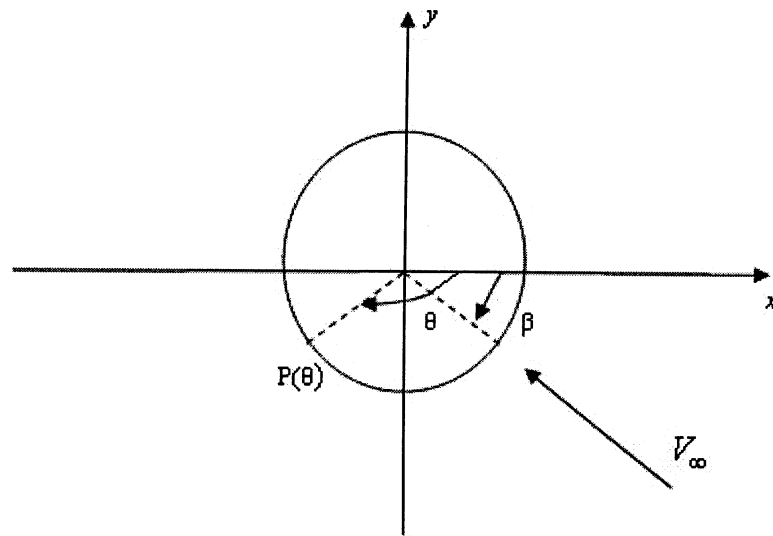


Figure 1.2 Two-dimensional flow about a right circular cylinder.

## 1.2 Intersection of Machine Learning and Fluid Dynamics

In general, mechanical and aerospace engineering systems present a number of challenging problems especially in the field of fluid dynamics. Understanding pressure and velocity distributions of the fluid flow is essential to the design and control of complex multi-component interacting systems. Experimental and computational methods in fluid

dynamics have long been successfully used to model such engineering systems. However, intensive research in both fields spanning over several decades have highlighted several shortcomings [9].

Over the last few decades, as complexity of engineering systems increased, it became a necessity rather than academic curiosity to adopt inter-disciplinary approaches to study a system. Intelligent data understanding tools like ANNs [10] and kernel methods from machine learning and statistics have enjoyed increased popularity in mechanical and aerospace engineering problems.

The problem of developing surrogates falls under the category of regression problems in machine learning research. Popular regression methods include splines [11], projection pursuit regression [12], radial basis function networks [13] and back-propagation networks [14]. These methods require the use of user-determined control parameters and/or kernel hyper-parameters. The user must find the optimum values of the control parameters for the entire data set either by cross-validation or grid search approach. In such approaches the data must be used to generate numerous randomly selected subsets for training and testing. The values of the control parameters must then be optimized on each of these testing subsets and the optimum control parameters averaged.

### **1.2.1 Approximation of physical relations and simulation of flow fields**

ANN based learning approaches have shown promise in approximating inverse physical relations where EFD or CFD methods are useful in solving only the forward

problem. These ANN learning approaches have also been successful at simulating turbulent flow fields with far less computational effort than CFD methods. For example, Faller and Schreck used Recursive Neural Networks (RNN) to predict time dependent unsteady boundary layer development, separation, dynamic stall and dynamic reattachment [15]. Neither experimental, nor computational methods have been successful at characterizing three-dimensional unsteady flow fields at parameter ranges corresponding to practical aerodynamic applications. These neural network models also form the foundation of adaptive control systems. Whitmore and Rohloff demonstrated the use of ANNs to predict angle of attack, sideslip and dynamic pressure from static pressure measurements to calibrate Flush Air Data Systems (FADS) for aircraft [1]. Experimental and computational methods could only solve the forward problem of obtaining surface pressures given a velocity and geometry configuration. Giralt *et al.* used a fuzzy neural network pattern recognition technique that could learn the nonlinear dynamics of a turbulent velocity field and predict the presence of coherent motions in the turbulent wake given an initial velocity condition [16].

### **1.2.2 Flow optimization and control**

Artificial neural networks have also been used to develop low order models predicting near wall dynamics in turbulent flows [17]. Such low order models are helpful for drag reduction and turbulent flow control. Near-wall stream-wise vortices increase skin friction drag and wall actuation in the form of blowing and suction can significantly reduce drag. Traditional control methods require velocity field information from the entire domain and are computationally time consuming, thereby are impractical in real-time situations. Artificial neural networks, on the other hand, can approximate a second order model of

near-wall velocity using only surface pressures and shear stresses as inputs. Babcock *et al.* demonstrated that a neural controller performed similarly to an analytical controller in a fully turbulent simulation [7,8]. The authors also discussed the practical and fundamental issues of using a neural controller for drag reduction.

### 1.2.3 Hybrid modeling with numerical simulations

Large Eddy Simulation (LES) is an attractive numerical simulation technique because it has good accuracy and relatively lower computational expense than DNS techniques. However, the computational effort devoted to resolving the viscous sublayer is still considerable. Hybrid use of ANNs has been proposed with LES techniques to resolve boundary layers of wall bounded flows. Sarghini *et al.* used neural networks as subgrid scale models with LES [20]. The authors demonstrated that turbulent viscosity coefficients can be accurately mapped with neural networks saving significant computational time. Successful use of such a hybrid method opens up the possibility of generating approximate boundary conditions to pair LES in the free turbulence region and RANS models in the near wall regions.

In a related work, Wollblad and Davidson [21] proposed to replace RANS with filtered DNS data to give resolved subgrid velocities and stresses. The authors formed orthonormal bases of the subgrid stresses using Proper Orthogonal Decomposition (POD) and computed the coefficients of the expansion using neural networks. They concluded that ANNs performed significantly better than a linear stochastic estimator, however, the integration of LES and RANS still needs significant ad-hoc adjustment.

### **1.2.4 Hybrid design with numerical simulations and optimization techniques**

Artificial neural networks have been widely used with CFD simulations and optimization techniques to conduct parametric design of aerodynamic structures. Such a design method can be seen as a variation of the traditional Response Surface Methodology (RSM), where the response surface is constructed by neural networks. Once a response surface has been obtained, an optimization procedure can be used to search a solution with optimal performance characteristics. ANNs provide enhanced flexibility than RSM due to their ability to handle multi-dimensional interpolation with unstructured data. Rai and Madhavan used neural networks with RANS and conjugate gradient optimization scheme to obtain an optimal design of a turbine airfoil with 15 design parameters. They also proposed a hybrid scheme, using neural networks and low-order polynomials that they believe can handle more parameters than a design scheme with ANNs alone [22].

## **1.3 Need for the Current Research in ANNs**

Although neural networks are enjoying increased popularity in the mechanical and aerospace engineering community a number of issues must be addressed.

### **1.3.1 Formalization of the final network**

A typical Radial Basis Function (RBF) ANN involves several parameters to be defined by the user including: a) number of neurons in the hidden layer, b) initial weight matrix, c) parameters of the basis functions, and d) values of the learning rates. Ad-hoc procedures exist for choosing each of these parameters. However, lack of a principled approach to network construction generally leads to an over parameterized network. To

optimize the structure of the network, exhaustive grid search and cross-validation approaches are often needed. To address this problem, this work focuses on greedy scattered data approximation approaches [23]. In these approaches the user needs to load only the training and test sets and the algorithm solves for the required parameters in a self-adaptive and greedy fashion. In other words, greedy scattered data approximation approaches provide a formal way to construct networks in an optimized manner.

### **1.3.2 Choice of training data**

It is often unclear how many training points are sufficient for acceptable generalization error and how these training data should be chosen. Traditionally, network training and testing is done only after all data is collected. This is called passive learning in the machine learning literature, where the learning algorithm does not take part in the data collection procedure. Collecting data without taking into account the input-output functional relationship like the traditional Latin Hypercube Design [24] methods can result in choice of data that add little to no information to training of the network and can even adversely affect the generalization ability of the training method. This is especially a problem when data collection is costly, like flight tests, or when the input domain is excessively large to sample. Active learning [25] methods are popular in the machine learning literature where the learning algorithm takes part in the data collection procedure and new input configurations are sampled such that they maximize an information gain criterion in a principled manner. These sampling procedures are essentially sequential in nature and not only help the engineer to accelerate through the test matrix, but also allow the learning algorithm to have better generalization capability.

### **1.3.3 Assimilating physical knowledge of the system**

ANN provide a powerful tool to interpolate experimental data. However, it is often a concern with any statistical learning technique that it does not incorporate the physics of the system in approximating physical quantities as a function of input design variables. Regularization approaches like the Generalized Tikhonov Regularization method provides us a framework where it is possible to include physical knowledge of the system, in the form of numerical simulations, as a priori information in the training of the network [26]. This type of data assimilation procedure can be seen as approximating noisy and scattered experimental data with physics based smoothness. It can also be interpreted as improving low fidelity numerical simulations with more accurate experimental data. This tool can result in acceptable flow field solutions by fusing sparse experimental data with low fidelity CFD data, thereby saving resources.

It is believed that integration of machine learning methods with fluid dynamics research is the key to optimizing information extraction (Fig. 1.3). Furthermore, the issues mentioned in Sections 1.3.1 through 1.3.3 should be addressed from an engineering point of view. This thesis will demonstrate the effectiveness of the machine learning methods in the calibration and design of a Flush Air Data System (FADS).

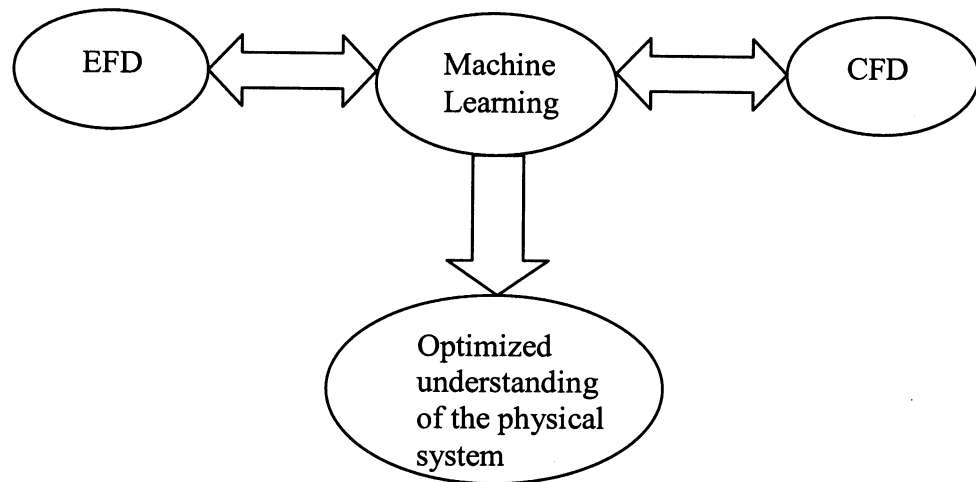


Figure 1.3 Machine learning tools can provide a framework in which EFD and CFD can be integrated in a principled manner.

## 1.4 Proposed Ideas and Objectives

In this work, a learning tool has been developed that can solve the inverse problem under consideration, and is labeled Sequential Function Approximation (SFA). The first objective in this thesis is to develop a computational surrogate that can predict wind speed and yaw angle from static surface pressure measurements. The surrogate should not have wind speed prediction errors greater than  $\pm 3.4$  fps and yaw angle prediction errors greater than  $\pm 2$  degrees. The second objective is to investigate how data model fusion can improve wind speed and direction prediction accuracies. The third objective is to develop a general intelligent sampling strategy with active learning and SFA on regression problems.



Chapter 2 presents an introduction to greedy function approximation and its merits. Section 2.1 discusses existing greedy sparse approximation methods related to SFA. Section 2.2 discusses the Sequential Function Approximation algorithm and its implementation issues. Section 2.3 presents comparison of SFA with competing state of the art sparse approximation methods and justifies the need and effectiveness of SFA. Section 2.4 presents the areas of the algorithm which need improvement and presents ways to enhance the performance of SFA. Section 2.5 presents results of comparisons between the two versions of SFA on artificial and real world classification and regression problems.

Chapter 3 discusses estimation of air data parameters from static pressure measurements in more detail. Section 3.1 presents the wind tunnel pressure measurements for FADS calibration on two problems, namely a surface vessel problem and Runway Assisted Landing Site (RALS) control tower. Section 3.2 presents several freestream wind speed and direction estimation techniques with a thorough discussion of their advantages and drawbacks. Section 3.3 presents the prediction accuracies of the estimation techniques on both the surface vessel and the RALS tower problem. Section 3.4 presents pressure sensor sensitivity analysis with respect to air data parameter prediction. The chapter ends with learning curves illustrating the graceful degradation of prediction accuracy with decreasing number of pressure sensors and increasing number of training points.

Chapter 4 begins by discussing the importance of fusing flow solutions of variable fidelity in order to enhance the understanding of the system and accelerate its design. The concept of data-model fusion is introduced and the techniques to do so are discussed in Section 4.1. Section 4.2 discusses the role of regularization techniques in accommodating the smoothness of physics-based flow solutions in the learning problem at hand. Section 4.2.1 and 4.2.2 discuss the role and objectives of data-model fusion in the FADS problem and presents how it can handle noise, sparsity and incompleteness in the wind tunnel data. Finally, Section 4.3 presents improvements in the wind speed and direction prediction accuracies due to a better training set construction by data-model fusion.

Chapter 5 begins by introducing the fields of active learning and optimum experiment design. Section 5.1 discusses optimum experiment design for regression problems and presents a fairly detailed literature review of the topic. Section 5.2 discusses a G-optimal design procedure with SFA and presents its implementation issues application on a simulated regression problem. Finally, Section 5.3 presents the application of this technique on the FADS problem. Chapter 6 presents conclusions and future avenues of research. Appendix A discusses an inverse approach to predict yaw angle for the RALS tower and the surface vessel. Appendix B discusses active learning for classification tasks. Section B.1 presents a brief literature search and Section B.2 discusses the use of uncertainty sampling with SFA for binary and multi-class classification problems. Finally, Section B.3 presents the application of this technique to develop a sequential wind tunnel experiment design strategy for a cavity flow problem.

## Chapter 2

### Greedy Function Approximation

In a general supervised machine learning problem of function approximation, given a  $d$  dimensional input data sample  $x \in R^d$  and outputs  $y \in R^m$ , where  $d$  and  $m$  are input and output dimensions, the task of the learner is to approximate the function mapping the input to the output. In the current problem of interest the input is the pressure data and the outputs are the freestream wind speed and wind direction. In general for a regression problem the mapping function is known to the learner only at a discrete set of points in the input space. Since no other a priori information is available the learner assumes that the mapping function is smooth and continuous [27]. Using this assumption the learner can now use a linear sum of  $n$  smooth basis functions from function approximation theory to approximate the mapping function, as shown in Eq. (2.1).

$$y_n^a(x) = \sum_{i=1}^n c_i (\phi(x, \phi_i) + b_i) \quad (2.1)$$

Here  $y_n^a$  is the approximated output,  $c_i, b_i$  are the coefficients of linear expansion,  $\phi$  is the basis function,  $\phi_i$  is the set of kernel parameters for the  $i^{\text{th}}$  basis function and  $n$  is the number of basis functions. For classification problems the output is discrete. Either  $y = [-1, +1]$ , as in a binary classification problem, or it can take multiple integer values as in a multi-class classification problem or all classes could also be handled simultaneously. In a classification problem the mapping function is again assumed to be smooth and continuous and the output is manifested by the application of a proxy function  $\Lambda$  on the mapping function as shown in Eq. (2.2).

$$y_n^a(x) = \Lambda \left( \sum_{i=1}^n c_i (\phi(x, \theta_i) + b_i) \right) \quad (2.2)$$

For binary classification problems a popular proxy function is the sign function. A popular approach to attempt multi-class classification problems is by combining several binary classifiers using a one vs. all approach. Handling all classes simultaneously has also shown promise.

Gaussian Radial Basis Functions (RBF) have been successfully used in numerous machine learning problems and their approximation properties have been rigorously studied in function approximation theory [13]. Gaussian radial basis functions have good generalization power and are free from the curse of dimensionality [13]. In this work Gaussian radial basis functions with uniform spread in all dimensions is used, however the proposed approach, can be applied to a number of basis functions like hyperbolic tangents, polynomials,  $B$ -splines, and trigonometric functions. Gaussian RBF's, given by Eq. (2.3) have two kernel parameters associated with each basis function, the basis center  $x^*$  and the basis width  $\sigma$ .

$$\phi(x, \theta_i) = \exp \left[ -\frac{(x - x_i^*) \cdot (x - x_i^*)}{\sigma_i^2} \right] \quad (2.3)$$

To construct an approximation as a linear sum of  $n$  Gaussian RBFs as shown in Eqs. (2.1) and (2.2) the values to a total of  $3n+1$  unknown parameters would have to be determined. Solving an optimization problem for all the  $3n+1$  parameters would naturally be excessive computational burden. Greedy function approximation techniques simplify the above problem by adding one basis function at a time to the approximation. This results in a series of smaller optimization problems involving only three unknowns.

## 2.1 Greedy Algorithms

In this work a greedy approach has been used to construct approximations as a linear sum of Gaussian Radial Basis Functions. Having chosen an appropriate basis function it is imperative to optimally determine the coefficients and the kernel parameters so that the residual error decreases optimally with the addition of basis functions. In a greedy function approximation setting, the initial target or output vector is stored as the initial residual error. This residual error is used to pick the next basis function kernel parameters and the linear coefficients that reduce the error as much as possible. This makes the greedy algorithms a highly nonlinear constructive approximation technique. Typical, greedy function approximation algorithms solve for the coefficients  $c_i$  from the principles of linear optimization by minimizing the discrete inner product norm of the residual error.

Determination of the basis centers is the more challenging issue since they involve a  $d$ -dimensional optimization problem for each basis function. One way of simplifying this problem is to constrain the choice of the basis centers from one of the  $s$  available training points. This leaves us with the optimization of the basis width for each basis function. The popular sparse kernel modeling techniques, like Support Vector Machines (SVM) [28], and greedy function approximation algorithms, like the Matching Pursuit (MP) [29] and others, fix the width of the basis functions to a value estimated by a cross-validation or a similar parameter estimation technique. Keeping the width of the basis functions fixed will result in several limitations. First, to carry out a cross-validation procedure, or grid search, a chunk of training points have to be kept aside. A grid search would demand

repeated training and testing to be carried out consuming necessary computational time and expense. Using the same width for all basis functions increases the dependence on any initial choice of the width parameter. If locally optimal solutions are used for each basis then it decreases the dependence of the generalization power of the algorithm on the basis width. It also reduces user interaction and results in a self-adaptive network. Finally a larger portion of the residual error is dissipated if the basis function has its width optimized, as compared to a basis function with a fixed width. Section 2.1.1 presents sparse kernel techniques that bear some similarities with the proposed approach. However, none of the techniques mentioned in this section study the dependence of the predictive power of the algorithm on the basis width.

### **2.1.1 Matching pursuit**

Matching Pursuit [29] is a popular type of greedy algorithm in the signal processing community. The aim of Matching Pursuit is to approximate a target signal as a linear sum of elementary signals or atoms or basis. Usually a large, linearly dependent collection of signals, known as a dictionary, is available. If the dictionary is orthonormal, then it is possible to reconstruct the target signal by using only a few atoms. This can be done by choosing atoms that are most strongly correlated with the residual error. This procedure of approximating a high-dimensional signal into a sum of several low dimensional signals is popular in compression of images, audio and video signals. Developing sparse approximations to target signals can be justified on the basis of economy and simplicity. A simple representation of a signal provides redundancy and robustness against external noise. It is also justified by Occam's razor which states that "Causes must not be

multiplied beyond necessity.” Vincent and Bengio [30] showed how Matching Pursuit could be used to build a kernel based solution to a machine learning problem and called it Kernel Matching Pursuit (KMP). It is a greedy algorithm for building an approximation of a discriminant function as a linear combination of basis functions chosen from a kernel-induced dictionary.

Vincent and Bengio [30] compared the performance of KMP with SVM and RBF networks on several real world data sets. They concluded that KMP gave comparable results to SVMs with sparser approximations. However, Suykens et al. [31] showed that sparser solutions can be obtained by SVMs via a pruning strategy. SFA bears some similarities with the basic version of KMP but the primary differences are the following. In KMP the basis function at the  $n^{\text{th}}$  stage is chosen such that it maximizes

$$abs\left(\frac{\langle \vec{r}_{n-1}, \vec{\phi}_n \rangle}{\langle \vec{\phi}_n, \vec{\phi}_n \rangle}\right).$$

Since a constant width is chosen and the basis functions are centered on the training points, choosing a new basis function amounts to choosing a basis center from the available training data based on the above mentioned criterion. From a geometrical perspective such an approach will choose the basis function  $\vec{\phi}_n$  that is most aligned with the residual vector  $\vec{r}_{n-1}$  in  $R^s$ . An improvement is possible if the width of the basis functions is not fixed. Allowing the width to vary would increase the size of the dictionary generated by the training points. A larger dictionary allows us to choose a better basis function that maximizes the above mentioned criterion.

In SFA the spread of each basis function is optimized and the basis center corresponding to the maximum absolute value of the residual at that stage is chosen. This can be interpreted as choosing basis functions from a larger dictionary that are aligned more parallel to  $\vec{r}_{n-1}$  in  $R^s$  as shown in Fig. 2.1. Choosing a new basis function in this manner will reduce more of the residual error and result in sparser approximations, as compared to KMP.

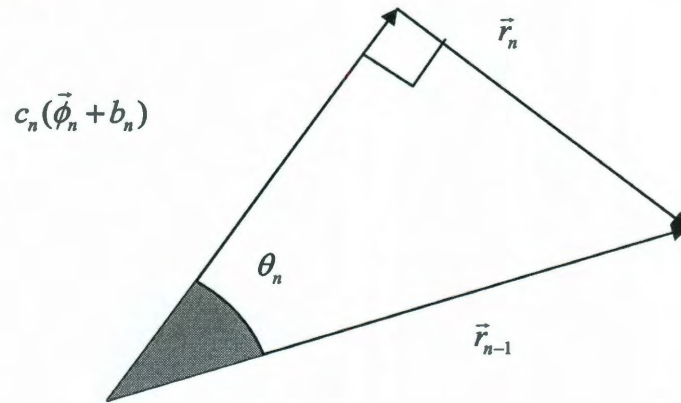


Figure 2.1 An optimum basis function is one that is the most aligned with  $\vec{r}_{n-1}$

A significant limitation of the basic version of KMP is that if the residual vector is orthogonal to all the available basis functions, then there will be no decay in the residual error and the algorithm will continue choosing the same basis center. The basic version of KMP did not perform to satisfaction in the experiments done by Vincent and Bengio [30]. To address those limitations the authors proposed a back-fitting and a pre-fitting version of KMP [30]. Properties of orthogonal projections were used in the back-fitting and the pre-fitting versions of KMP to update all previously chosen basis function and coefficients of linear sum. The aim of the back-fitting and pre-fitting versions of KMP was to select a basis function that is the most aligned to the residual vector and is the



most perpendicular to the previously chosen basis function. Keeping the basis centers fixed at the training points and the width of the RBF fixed does not always result in a good choice of basis function from the available dictionary. Because an additional degree of freedom is available, allowing the width of the basis function to vary will result in a better choice of the basis function. SFA was compared against the basic version of KMP and the pre-fitting version of KMP on artificial and real world classification problems [32]. The results showed that SFA obtained sparser classifiers with similar classification accuracies on almost all data sets. Moreover only one pass through the dictionary was required by SFA to choose the basis function as opposed to the pre-fitting version of KMP that required two passes.

### **2.1.2 Sparse approximation methods**

Sparse approximation methods broadly fall in the following categories: a) sequential forward greedy algorithms, b) backward greedy algorithms and c) mathematical programming approaches [12]. KMP and SFA are examples of sequential forward greedy algorithms. Other popular examples are Natarajan's Order Recursive Matching Pursuit (ORMP) [33] and Orthogonal Least Squares RBF (OLS-RBF) [34]. Vincent and Bengio [30] described the pre-fitting version of KMP, with squared error loss and Gaussian kernel, to be identical to the OLS-RBF algorithm. They compared OLS-RBF against Gaussian Support Vector Machines (SVM) to conclude that OLS-RBF performed as well as Gaussian SVM with sparser approximations. In the reference [32] authors compared the performance of Least Square Support vector Machines (LS-SVM) [31] against KMP algorithms and SFA. The primary difference between ORMP and SFA

is that ORMP normalizes and reorients all unselected basis functions and avoids the recycling problem by choosing from previously unselected basis functions. SFA avoids the recycling problem by assigning a unique width to each basis function. Other greedy algorithms in the spirit of Natarajan's algorithm have also been published [23]. However, in these works the question of dependence of classification accuracy on the width of the Gaussian radial basis functions has not been addressed.

The backward greedy algorithms [35] start with a  $s \times s$  Gram matrix, where  $s$  is the number of observations in the training set and iteratively eliminates columns. Even though this helps in achieving guaranteed convergence properties it is more computationally expensive than the forward algorithms. The mathematical programming approaches like Basis Pursuit [36] differ from the sequential forward greedy algorithms in the sense that they use regularization in the cost function to determine a sparse solution. Basis pursuit also employs quadratic programming like support vector machines to minimize the associated cost function. Girosi [37] proposed a modified version of Basis Pursuit De-noising [36] and showed that SVMs are equivalent to them as they solve the same quadratic programming problem. However sequential forward greedy algorithms like SFA and KMP enforce regularization via sparsity.

### **2.1.3 Other related methods**

In the statistics community the most popular versions of greedy learning are Boosting [38] and Projection Pursuit [12]. KMP in its basic form can be seen to be similar to Boosting if the weak learners were interpreted as the kernel functions centered

on the training points. The Projection Pursuit method constructs the approximation of the target vector as a linear sum of ridge functions in a forward stagewise manner. The input training points are projected on a unit direction vector which is optimized at each stage. A ridge function which results in the maximum reduction of the residual error is selected. The addition of ridge functions continue until the residual error is smaller than a user-defined threshold. This method bears clear similarities to the Matching Pursuit methods of the signal processing community.

In the neural network community constructing the neural architecture in a sequential forward manner goes by the name of Cascade-correlation [39]. A new hidden unit is added if the previously added neurons do not reduce the residual error of the network below a specified threshold. The algorithm tries to maximize the magnitude of correlation between the new unit's output and the residual error signal. With these algorithms the users do not need to worry about the size and topology of the network. Other constructive algorithms include Dynamic node creation [40] and the Resource-allocating network [41].

## **2.2 Sequential Function Approximation**

Approximation of the unknown target function  $y$  is constructed by noting that a continuous  $d$ -dimensional function can be arbitrarily well-approximated by a linear combination of radial basis functions  $\phi$ . Sequential Function Approximation (SFA) was developed from mesh-free finite element research but shares similarities with the

Boosting [38] and Matching Pursuit [29] algorithms. Approximation of the target vector  $y$  is constructed by utilizing the Gaussian radial basis function  $\phi$ .

$$y_n^a = \sum_{i=1}^n c_i (\phi(x, \varrho_i) + b_i) \text{ with } \phi(x, \varrho_i) = \exp[\log[\lambda_i](x - x_i^*) \bullet (x - x_i^*)] \text{ for } 0 < \lambda_i < 1 \quad (2.4)$$

Traditionally, Gaussian radial basis functions are written as:

$$\phi(x, \varrho_i) = \exp \left[ -\frac{(x - x_i^*) \bullet (x - x_i^*)}{\sigma_i^2} \right]$$

Radial basis function is written as Eq. (2.4) in order to setup the optimization problem for  $\lambda_i$  as a bounded nonlinear line search instead of an unconstrained minimization problem for  $\sigma_i$ . The basic principles of our greedy algorithm are motivated by the similarities between the iterative optimization procedures of Jones [30,32] and Barron [44] and the Method of Weighted Residuals (MWR), specifically the Galerkin method [45]. The function residual vector  $\vec{r}_n$  at the  $n^{\text{th}}$  stage of approximation can be written as in Eq. (2.5):

$$\begin{aligned} \vec{r}_n &= y(x) - y_n^a(x) = y(x) - y_{n-1}^a(x) - c_n (\phi(x, \varrho_n) + b_n) \\ &= \vec{r}_{n-1} - c_n (\phi(x, \varrho_n) + b_n) = \vec{r}_{n-1} - c_n (\vec{\phi}_n + b_n) \end{aligned} \quad (2.5)$$

Using the Petrov-Galerkin approach, a coefficient  $c_n$  is selected that will force the function residual to be orthogonal to the basis function and  $b_n$  using the discrete inner product  $\langle \cdot, \cdot \rangle_D$  given by Eq. (2.6)

$$\vec{r}_n \bullet (\vec{\phi}_n + b_n) = \sum_{j=1}^s r_n(x_j) (\phi_n(x_j, \varrho_n) + b_n) = \langle \vec{r}_n, (\vec{\phi}_n + b_n) \rangle_D = - \left\langle \vec{r}_n, \frac{\partial \vec{r}_n}{\partial c_n} \right\rangle_D = - \frac{1}{2} \frac{\partial \langle \vec{r}_n, \vec{r}_n \rangle_D}{\partial c_n} = 0 \quad (2.6)$$

which is equivalent to selecting a value of  $c_n$  that will minimize  $\langle \vec{r}_n, \vec{r}_n \rangle_D$ . Writing Eq.

(2.5) as Eq. (2.7) where  $g_n = c_n \cdot b_n$

$$\vec{r}_n = \vec{r}_{n-1} - c_n \vec{\phi}_n - g_n \quad (2.7)$$

Expanding  $\langle \vec{r}_n, \vec{r}_n \rangle_D$  and taking the derivative with  $c_n$ , we get Eq. (2.8)

$$\langle \vec{r}_n, \vec{r}_n \rangle_D = \langle \vec{r}_{n-1}, \vec{r}_{n-1} \rangle_D + c_n^2 \langle \vec{\phi}_n, \vec{\phi}_n \rangle_D + s g_n^2 - 2c_n \langle \vec{r}_{n-1}, \vec{\phi}_n \rangle_D - 2g_n \langle \vec{r}_{n-1} \rangle_D + 2c_n g_n \langle \vec{\phi}_n \rangle_D$$

$$\frac{\partial \langle \vec{r}_n, \vec{r}_n \rangle_D}{\partial c_n} = 2c_n \langle \vec{\phi}_n, \vec{\phi}_n \rangle_D - 2 \langle \vec{r}_{n-1}, \vec{\phi}_n \rangle_D + 2g_n \langle \vec{\phi}_n \rangle_D = 0$$

$$c_n = \frac{\langle \vec{r}_{n-1}, \vec{\phi}_n \rangle_D - g_n \langle \vec{\phi}_n \rangle_D}{\langle \vec{\phi}_n, \vec{\phi}_n \rangle_D} \quad (2.8)$$

Taking the derivative of  $\langle \vec{r}_n, \vec{r}_n \rangle_D$  with  $g_n$ , relates  $c_n$  and  $g_n$  in Eq. (2.9)

$$\frac{\partial \langle \vec{r}_n, \vec{r}_n \rangle_D}{\partial g_n} = 2s g_n - 2 \langle \vec{r}_{n-1} \rangle_D + 2c_n \langle \vec{\phi}_n \rangle_D = 0$$

$$g_n = \frac{\langle \vec{r}_{n-1} \rangle_D - c_n \langle \vec{\phi}_n \rangle_D}{s} \quad (2.9)$$

Plugging Eq. (2.9) this into Eq. (2.8) and solving for  $c_n$  in Eq. (2.10)

$$c_n = \frac{\left( s \langle \vec{\phi}_n, \vec{r}_{n-1} \rangle_D - \langle \vec{r}_{n-1} \rangle_D \langle \vec{\phi}_n \rangle_D \right)}{\left( s \langle \vec{\phi}_n, \vec{\phi}_n \rangle_D - \langle \vec{\phi}_n \rangle_D \langle \vec{\phi}_n \rangle_D \right)} \quad (2.10)$$

Since  $b_n = \frac{g_n}{c_n}$ , Eq. (2.11) gives  $b_n$

$$b_n = \frac{\left( \langle \vec{r}_{n-1} \rangle_D \langle \vec{\phi}_n, \vec{\phi}_n \rangle_D - \langle \vec{\phi}_n \rangle_D \langle \vec{\phi}_n, \vec{r}_{n-1} \rangle_D \right)}{\left( s \langle \vec{\phi}_n, \vec{r}_{n-1} \rangle_D - \langle \vec{r}_{n-1} \rangle_D \langle \vec{\phi}_n \rangle_D \right)} \quad (2.11)$$

The discrete inner product  $\langle \vec{r}_n, \vec{r}_n \rangle_D$ , which is equivalent to the square of the discrete  $L_2$  norm, can be re-written, with the substitution of Eq. (2.10) and (2.11), as

$$\sum_{j=1}^s r_n(x_j)r_n(x_j) = \|\vec{r}_n\|_{2,D}^2 = \langle \vec{r}_n, \vec{r}_n \rangle_D$$

$$\langle \vec{r}_n, \vec{r}_n \rangle_D = \left\langle \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s}, \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s} \right\rangle_D \left( 1 - \frac{\left\langle \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s}, \vec{\phi}_n - \frac{\langle \vec{\phi}_n \rangle_D}{s} \right\rangle_D^2}{\left\langle \vec{\phi}_n - \frac{\langle \vec{\phi}_n \rangle_D}{s}, \vec{\phi}_n - \frac{\langle \vec{\phi}_n \rangle_D}{s} \right\rangle_D \left\langle \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s}, \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s} \right\rangle_D} \right) \quad (2.12)$$

Recalling the definition of the cosine given by Eq. (2.13), using arbitrary functions  $f$  and  $v$  and the discrete inner product,

$$\cos(\vartheta) = \frac{\langle f, v \rangle_D}{\langle f, f \rangle_D^{1/2} \langle v, v \rangle_D^{1/2}} \quad (2.13)$$

Eq. (2.12) can be written as

$$\|\vec{r}_n\|_{2,D}^D = \left\langle \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s}, \vec{r}_{n-1} - \frac{\langle \vec{r}_{n-1} \rangle_D}{s} \right\rangle_D \sin^2(\vartheta_n) = \left( \|\vec{r}_{n-1}\|_{2,D}^D - \frac{\langle \vec{r}_{n-1} \rangle_D^2}{s} \left[ 2 - \frac{1}{s} \right] \right) \sin^2(\vartheta_n) \quad (2.14)$$

where  $\vartheta_n$  is the angle between  $\vec{\phi}_n$  and  $\vec{r}_{n-1}$  since  $\frac{\langle \vec{r}_{n-1} \rangle_D}{s}$  and  $\frac{\langle \vec{\phi}_n \rangle_D}{s}$  are scalars. With Eq.

(2.14) one can see that  $\|\vec{r}_n\|_{2,D}^2 < \|\vec{r}_{n-1}\|_{2,D}^2$  as long as  $\vartheta_n \neq \pi/2$ , which is a very robust

condition for convergence. By inspection, the minimum of Eq. (2.14) is  $\vartheta_n = 0$ , implying

Eq. (2.15)

$$c_n(\phi(x_i, \vartheta_n) + b_n) = r_{n-1}(x_i) \quad \text{for } i=1, \dots, s \quad (2.15)$$

Therefore, to force  $\|\vec{r}_n\|_{2,D}^2 \rightarrow 0$  with as few stages  $n$  as possible, a low dimensional

function approximation problem must be solved at each stage. This involves a bounded

nonlinear minimization of Eq. (2.12) to determine the two variables  $\lambda_j$  ( $0 < \lambda_j < 1$ ) and index  $j^*$  ( $x_n^* = x_j \in \mathbb{R}^d$ ) for the basis function center taken from the training set. The dimensionality of the nonlinear optimization problem is kept low since only one basis function needs to be solved at a time.

### 2.3 Characteristics of SFA

The concept of shift invariant subspaces is applied to the study of SFA approximation error using multi-dimensional bell shaped basis functions. The authors studied the convergence rate of the approximation error when a) bases are added altering their shape and keeping their spacing constant and b) adding bases altering their shape and decreasing their spacing in a coupled fashion. Reference [46] concluded that the first method of controlling network approximation error resulted in only a linear convergence rate while altering the shape and decreasing the spacing in a coupled fashion yielded exponential convergence rate. An adaptation of the results given by Meade and Zeldin is given in Eq. (2.16)

$$\|u - u_n^a\|_{H^\nu} \leq C_1 (\mu_n)^{\frac{(\alpha-\nu)}{2}} + C_2 \left(\frac{\mu_n}{\Delta_n}\right)^\nu \exp\left[\frac{-C_3}{\Delta_n}\right] \quad (2.16)$$

where  $\mu_n$  is the basis shape changing parameter and is also a measure of the width of the basis functions ( $\mu_n = \sqrt{-\ln(\lambda_n)}$ ) and  $\Delta_n$  is the average distance between their centers. The term on the left hand side of the inequality is a measure of approximation error after addition of  $n$  basis functions in Sobolev space.  $C_3$  is an arbitrary positive constant,  $u(x) \in H^\alpha(\mathbb{R}^d)$ ,  $\alpha > \nu$ , and  $\nu$  is a positive constant. Since SFA constructs its

approximation sequentially, Eq. (2.16) is inspected at each iteration. Consequently, the degree of smoothness of the approximation  $\alpha$ , the shape parameter  $\beta$  and the average distance between basis function centers  $\Delta$  will change with each additional basis function and their corresponding values at each iteration are represented with subscripts of  $n$ . After the addition of  $n^{\text{th}}$  basis function, and for the  $L_2$  norm  $\|r_n\|_{H^v} = \|r_n\|_{H^0} = \|r_n\|_2$ , Eq. (2.16) can be rewritten as:

$$\|r_n\|_2 \leq C_1 (-\ln(\lambda_n))^{\frac{\alpha_n}{2}} + C_2 \exp\left[\frac{-C_3}{\Delta_n}\right] \quad (2.17)$$

For a finite problem domain  $\Delta_n \rightarrow 0$  as  $n \rightarrow \infty$  and since  $-\ln(\lambda_n)$  is the only optimization parameter in Eq. (2.17), then to obtain optimal exponential convergence of a linear technique with infinitely smooth basis one must have  $(-\ln(\lambda_n))^{\frac{\alpha_n}{2}} \propto \exp\left[\frac{-C_3}{\Delta_n}\right]$ .

Therefore, Eq. (2.17) can be written as:

$$\|r_n\|_2 = C_4 \exp\left[\frac{-C_3}{\Delta_n}\right] = \exp\left[\ln[C_4] - \frac{C_3}{\Delta_n}\right] \quad (2.18)$$

assuming a constant  $C_4$  exists such that the equality is valid.

Reference [49] showed that for bell-shaped bases  $\frac{1}{\Delta_n} \propto n$ , so assuming  $\frac{1}{\Delta_n} = n$  since  $C_3$  is an arbitrary positive constant, one can write

$$\|r_n\|_2^2 \equiv \langle r_n, r_n \rangle_D = \exp\left[2\ln[C_4] - 2C_3n\right] = \exp\left[\kappa_1 - \kappa_2n\right] \quad (2.19)$$

for positive constants  $\kappa_1$  and  $\kappa_2$ . Equation (2.19) shows that for optimal convergence, the logarithm of the inner product of the residual is a linear function of the number of bases ( $n$ ).



## Chapter 3

### Flush Air Data Sensing Systems

Chapter 1 discussed the need for Flush Air Data Sensing (FADS) systems and the challenges associated with it. FADS systems present a challenging inverse problem which is to estimate freestream wind speed and direction from static pressure measurements. Sparsity, noise and incompleteness in the available pressure data make it an ill-posed inverse problem encouraging use of ANN and other machine learning techniques which have been proven to give useful estimates based on limited information. As also mentioned in Chapter 1, FADS systems have been successfully calibrated for subsonic, supersonic and hypersonic aircrafts. Aircraft FADS systems need to install pressure sensors only on the nose of the vehicle because of the lack of significant cross-flow. However, for general bluff bodies with arbitrary geometry and cross-flow pressure sensors have to be installed about the perimeter. For example, the FADS system for the surface vessel shown in Fig. 3.1 (a) requires sensors to be installed about the upper hull and similarly FADS system for the Runway Assisted Landing Site (RALS) control tower shown in Fig. 3.1 (b) requires at least one sensor on each face of the tower. In these problems, closed form solutions for pressure distribution do not exist as they did for the aircraft problem as was shown in Fig. 1.1. In this chapter, techniques using SFA are presented for wind speed and yaw angle estimation. Section 3.1 will present the available wind tunnel data for two problems: a) Surface vessel and b) RALS tower. Section 3.2 will present the training and testing approach for a general FADS

problem. Section 3.3 will present the results of the wind speed and direction estimation techniques on both the problems.



(a)



(b)

Figure 3.1 Figure shows a) Surface vessel and b) RALS tower. Both geometries require pressure sensors to be installed all around the surface to account for cross-flow.

### 3.1 Wind Tunnel Data

Pressure data for flush air data systems were obtained for two problems a) surface vessel and b) RALS tower. Detailed wind tunnel tests on both geometries were conducted at the Fluid Mechanics Laboratory at NASA Ames Research Center.

### 3.1.1 Surface vessel

Wind tunnel tests were conducted to test the feasibility of a flush air data measurement system on a 1/180 scale model of a naval surface vessel [47]. Fifty-seven pressure sensors were mounted flush with the deckhouse periphery as shown in Fig. 3.2. The pressure sensors were aligned in three rings namely the bottom ring (taps 1-28, 57) and the middle ring (taps 29-50) and top ring (taps 51-56). The top ring sensors were excluded because of faulty measurements. A four-hole Cobra probe was used to measure the freestream wind speed and direction. The pressure sensor locations were chosen that were least disturbed by wake and model blockage effects. This network of pressure sensors was tested on a variety of wind speeds ranging from 40 fps to 175 fps and the wind direction varied from 0 to 360 degrees. The bow yaw angle  $\beta$  is measured with respect to the centerline of the vessel. For all port side winds  $\beta$  is negative and for all starboard side winds it is positive. Figure 3.3 shows the convention used for bow, port, starboard and stern side winds on the schematic of the surface vessel in this work. Variation of the wind attitude in three dimensions will be a part of the future work.

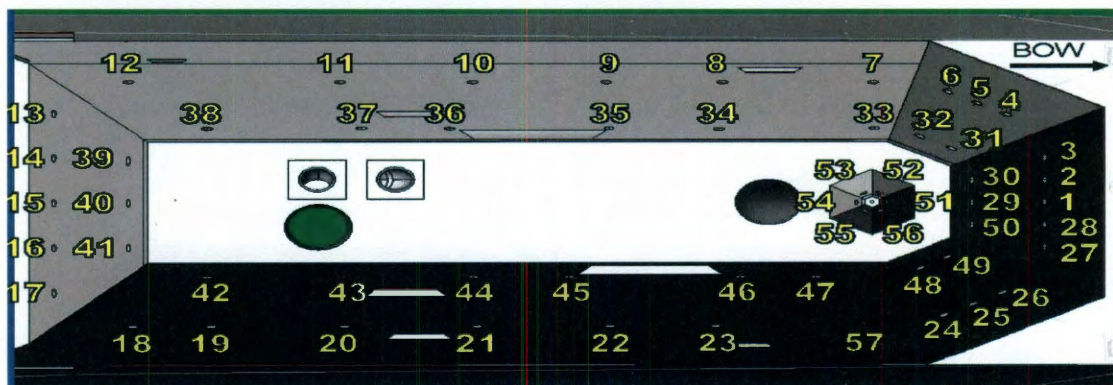


Figure 3.2 Model Deckhouse Pressure Port Locations (Top View)[47].

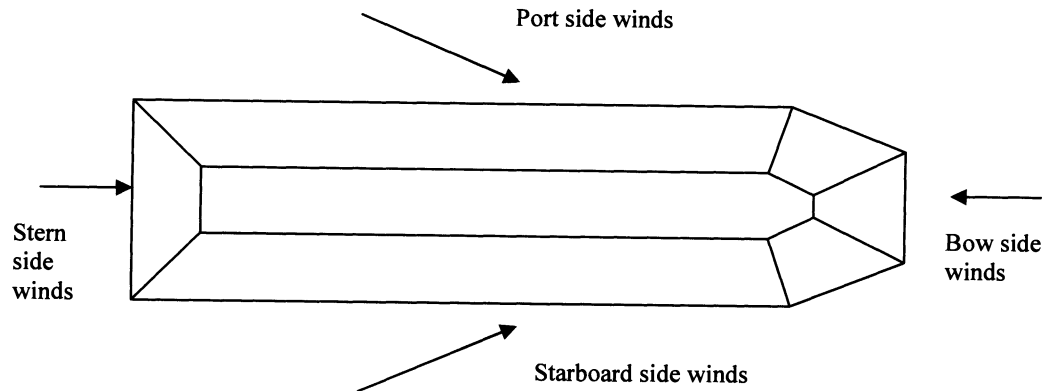


Figure 3.3 Direction of bow, port, stern and starboard side winds on the schematic top view of the surface vessel.

In Fig. 3.4 sample experimental static pressure distributions are shown for taps 1, 9, 40 and 22 placed on the bow, port, stern and the starboard sides, respectively. Figures 3.5 (a) and (b) show the variation of the coefficient of pressure for four bottom ring and four middle ring pressure ports, respectively, with bow yaw angle at a wind speed of 165 fps. It can be seen from Fig. 3.5 that for both lower and middle ring pressure ports there is a substantial variation in the coefficient of pressure. A substantial variation in the coefficient of pressure encourages the use of an intelligent algorithm that can extract information from pressure measurements to predict air data parameters.

Another important property that the pressure port plots show is that for any wind velocity configuration, the pressure ports lying on the downwind side show less variation in the pressure values than those lying in the upwind side. This a priori information can be used to accelerate the training and testing procedure which is explained in the next section.

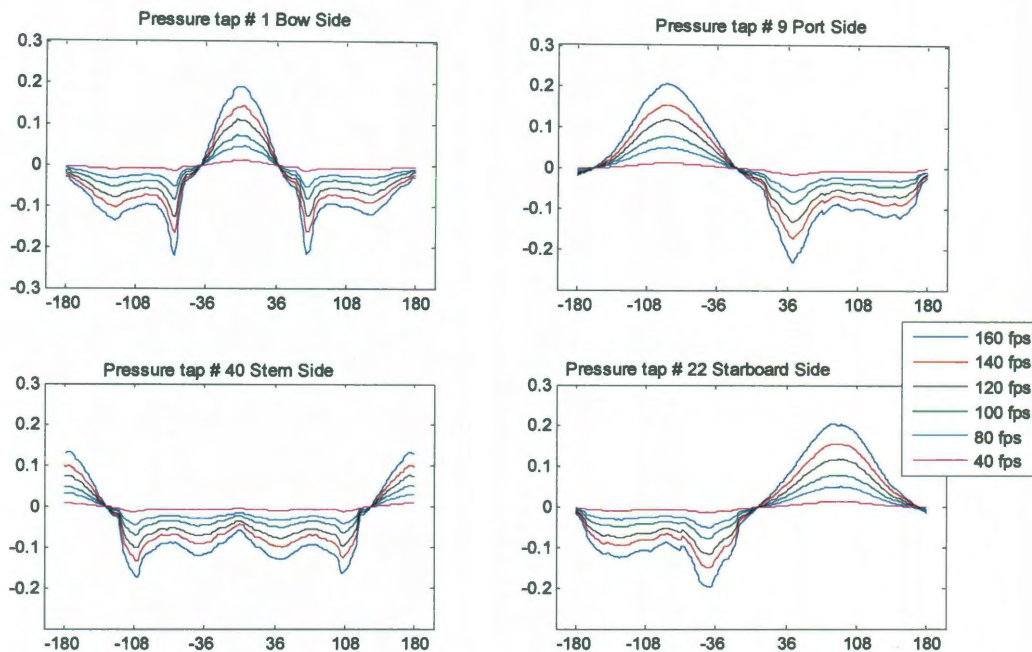


Figure 3.4 Static pressure variation vs. the yaw angle for different wind speeds.

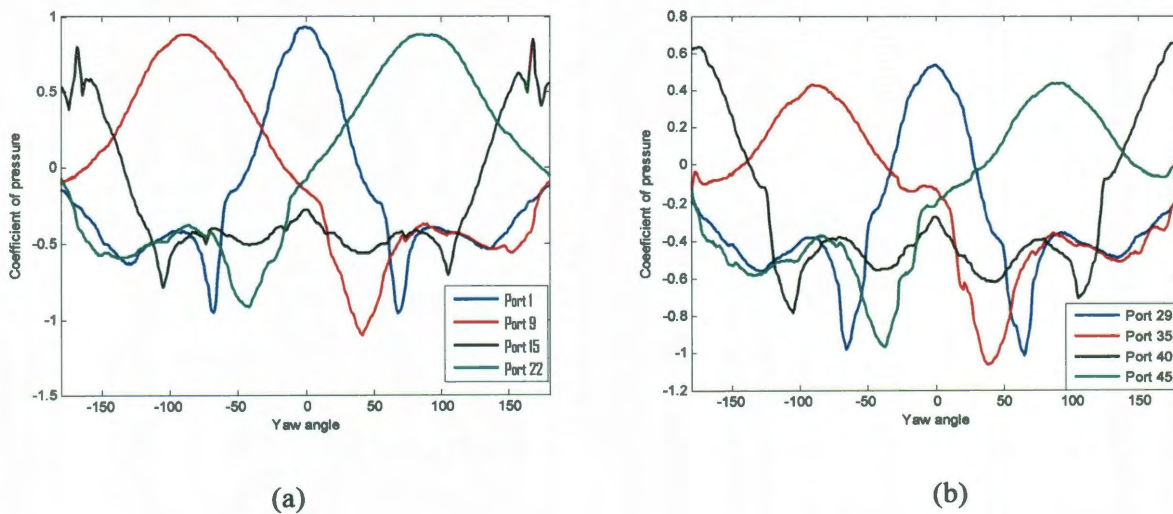


Figure 3.5 Coefficient of pressure variation for (a) bottom ring ports and (b) middle ring ports versus the yaw angle at 165 fps freestream wind speed.

### 3.1.2 RALS tower

The Runway Arrested Landing Site (RALS) control tower is located at the Naval Air Warfare Center Lakehurst, New Jersey. Wind tunnel tests were conducted on a 1/72 scale model of the RALS control tower [48]. One pressure sensor was mounted on each face of the control tower as shown in Fig. 3.6. A four-hole Cobra probe was used to measure the freestream wind speed and direction. The FADS system was tested on a variety of wind speeds ranging from 40 fps to 120 fps approaching the model from all 360 degrees of yaw. Variation of the wind incidence in three dimensions will be a part of the future work.

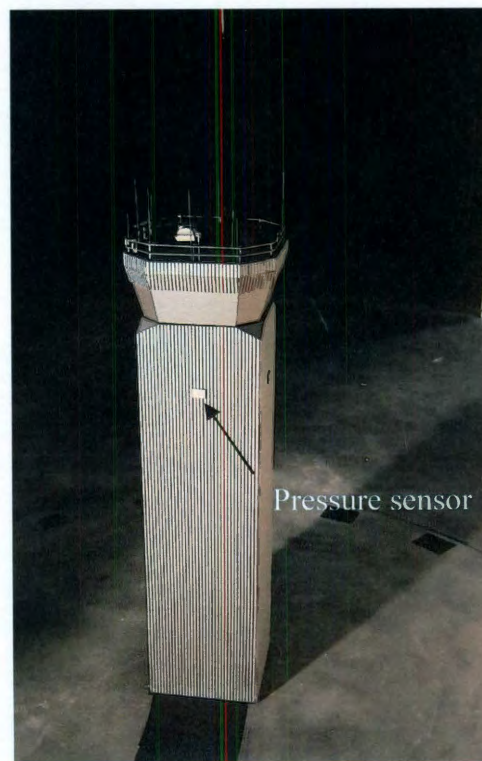


Figure 3.6 Wind tunnel model of the RALS tower [48]

Borrowing from flight mechanics literature and restricting our problem to two-dimensions,  $\beta$  is defined as the yaw angle and is measured in the anticlockwise sense with respect to the centerline of the model. So winds approaching the south port have  $\beta = 0$  degrees, for the east port  $\beta = 90$  degrees, for north port  $\beta = 180$  degrees and for the west port  $\beta = 270$  degrees. In Fig. 3.7 shown below static pressure is plotted for each pressure port. The pressure port on the south face bears positive pressure values when wind is incident head on at the pressure port and bears negative values for winds hitting the north face. Similarly the pressure port on the west face bears positive pressure values when the yaw angle is positive and bears negative values for negative yaw values. Also evident from Fig. 3.7 is that the pressure ports do not show much variation when they are on the leeward side.

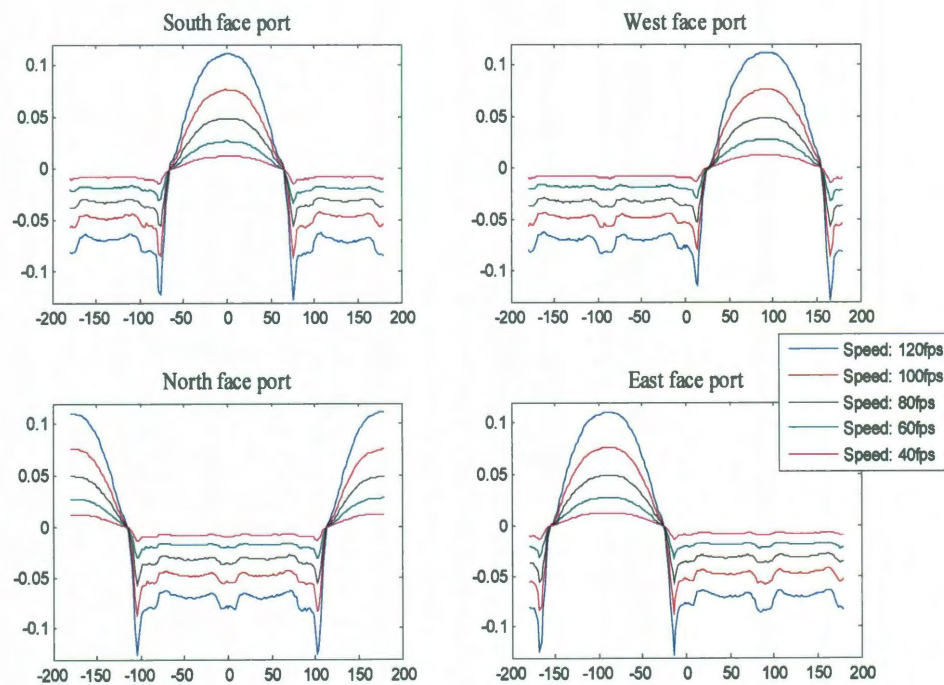


Figure 3.7 Static pressure variation vs. the yaw angle for different wind speeds.

Figure 3.8 shows the variation of coefficient of pressure for each pressure port. The pressure coefficient is calculated by normalizing the static differential pressure with the corresponding dynamic pressure. This normalization eliminates the effect of wind speed and results in a coefficient that is only a function of wind direction. Figure 3.8 shows all pressure coefficient graphs collapsed into one graph for each port. It also shows a substantial variation in the coefficient of pressure which encourages the use of an intelligent algorithm that can extract information from pressure measurements to predict air data parameters.

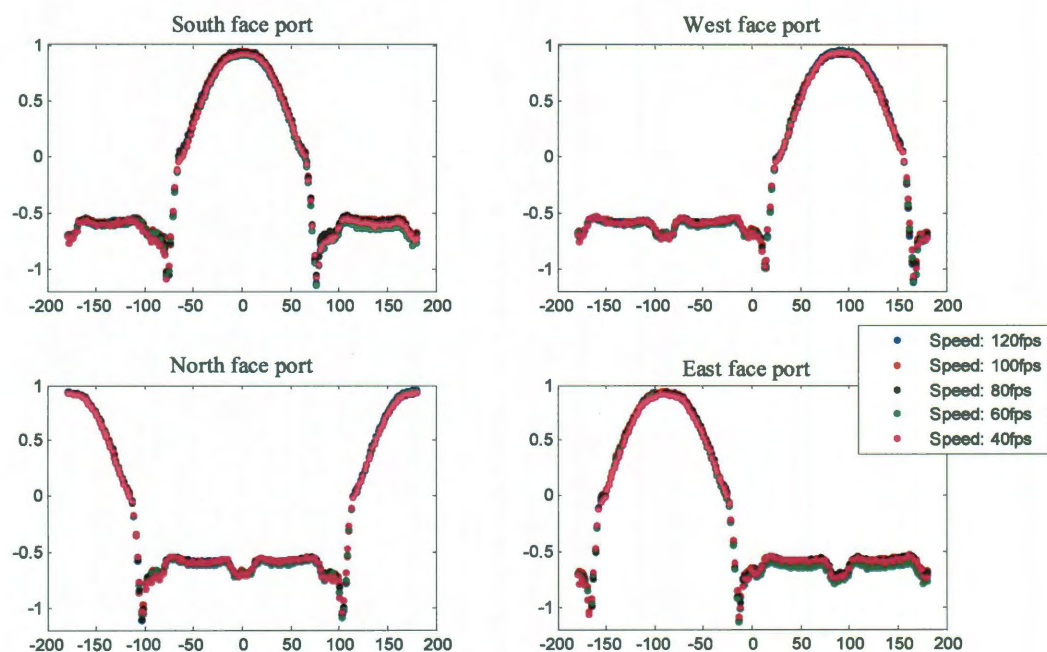


Figure 3.8 Pressure coefficient variation vs. the yaw angle for different wind speeds.



## 3.2 Forward and Inverse Problems

Computational algorithms like neural networks that learn from data can be very effective in solving inverse problems and bear potential advantages in the construction of the FADS mapping function  $G$  (Section 1.1) compared to look-up tables. Examples of neural networks advantages include high-dimensional mapping, smoother nonlinear control, intelligent empirical learning and fewer memory requirements [49-52]. For speed of evaluation, reduced complexity, and the potential for graceful performance degradation with the failure of pressure sensors, the inverse problem is solved using only two sets of neural networks with pressures read from all surface sensors,

$$V_{\infty} = G1(p_1, \dots, p_d) \quad \text{and} \quad \beta = G2(p_1, \dots, p_d, V_{\infty}).$$

### 3.2.1 Dynamic pressure

Even though our problem involves turbulence and cross-flow about nontrivial geometries, the existence of a functional relationship between  $P$  and  $V_{\infty}$  and between  $C_p$  and  $\beta$  can be safely assumed. The wind speed was predicted using only the surface pressure data while the wind direction was predicted using the pressure coefficient data derived from measured pressure and predicted wind speed values. It is noted that this coupling of networks put the burden of high accuracy on the wind speed predictor. Freestream air density was kept constant in the current problem which makes prediction of freestream wind speed equivalent to dynamic pressure. Sequential function approximation was used to construct one RBF network to model  $G1$  and the corresponding wind speed surrogate is represented by Eq. (3.1)

$$\hat{q}_{pre} = \sum_{i=1}^n c_i \exp(\ln[\lambda_i] * (\bar{p}_{test} - \bar{p}_i^*) * (\bar{p}_{test} - \bar{p}_i^*)) + b_i \quad (3.1)$$

where  $\hat{q}_{pre}$  is the predicted dynamic pressure. The hyper-surface for the current dynamic pressure prediction problem can be imagined as a hyper-cone with static pressure as input dimensions and dynamic pressure as output. Dynamic pressure increases linearly with static pressure given a yaw angle  $\beta$  for all pressure sensor positions,  $\theta$ . So each wind tunnel sweep at a constant speed would yield data points lying at the contours of the hyper-cone. Also, the slope of the hyper-cone corresponds to the coefficient of pressure at a given value of the yaw angle  $\beta$ .

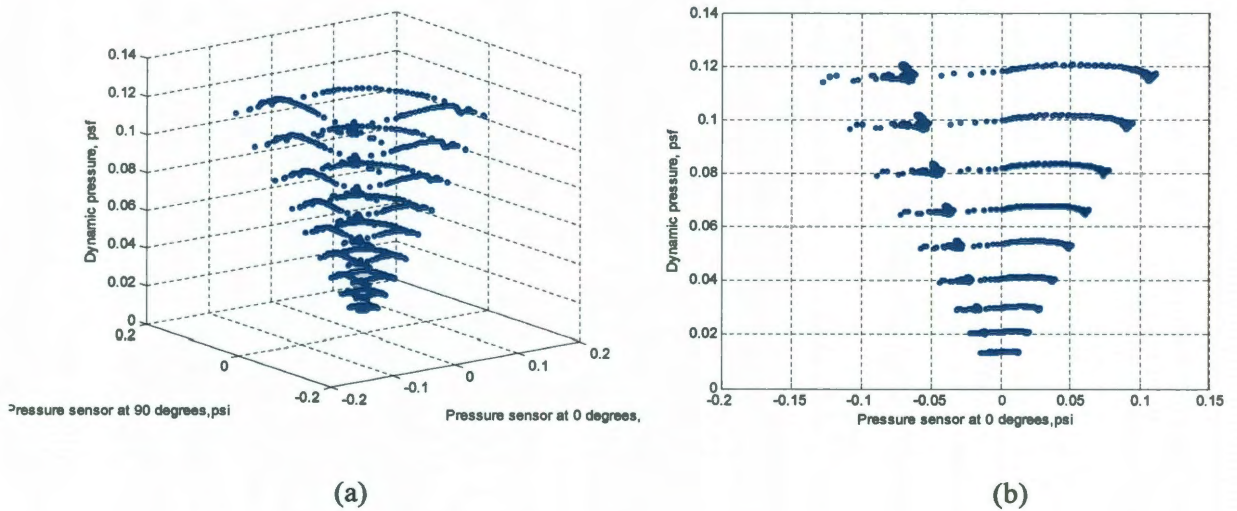


Figure 3.9 Dynamic pressure hyper-cone represented as a function of a) two pressure sensors and b) one pressure sensor.

Figure 3.9 (a) and b show the aforementioned hyper-cone in two and one dimensions respectively. Looking at Fig. 3.9 (b), it is evident that the span of the cone includes all pressure measurements of the sensor at  $\theta = 0$  degrees at all wind speeds and yaw angles. The right edge of the cone corresponds to the case when the sensor faces directly into the

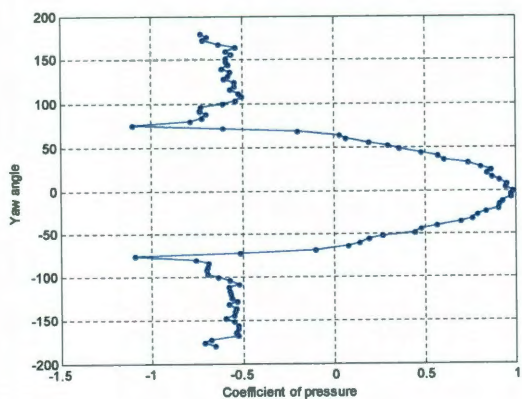
wind or  $\beta = \theta$ . As one moves from the right to the left edge of the cone the separation at the pressure sensor increases. A RBF network constructed by SFA can suitably approximate this hyper-cone, however it is possible that a better approach might exist for this approximation. This will constitute a part of the future work.

### 3.2.2 Wind direction

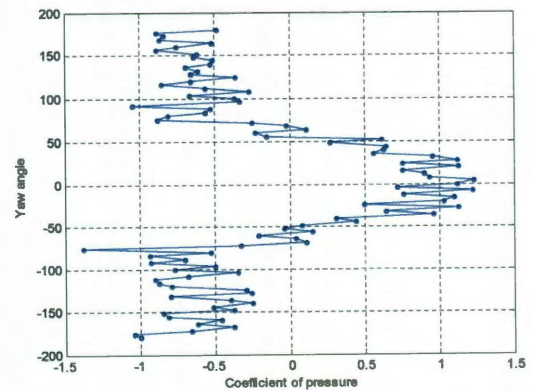
Once the wind speed predictor was available, the test static pressure values were divided by the corresponding predicted dynamic pressure to obtain the test coefficient of pressure values. Several ways exist to predict the yaw angle  $\beta$ . One straightforward way is to construct one RBF network for  $G2$  given by Eq. (3.2), where  $\hat{\beta}_{pre}$  is the predicted yaw angle

$$\hat{\beta}_{pre} = \sum_{i=1}^n c_i \exp(\ln[\lambda_i] * (\bar{c}_{p,test} - \bar{c}_{p,i}^*) * (\bar{c}_{p,test} - \bar{c}_{p,i}^*)) + b_i \quad (3.2)$$

However, unlike dynamic pressure, yaw angle prediction is a strongly non-unique problem.



(a)



(b)

Figure 3.10 Non-uniqueness of the yaw angle prediction problem at a) 120 fps and b) 40fps (RALS tower data).

Figure 3.10 (a) and 3.10 (b) show the variation of the yaw angle with bow side coefficient of pressure at 120 and 40 fps respectively when random noise of magnitude 0.005 was added to the RALS tower pressure data. It is evident from Fig. 3.10 (a) that at least two solutions for  $\beta$  exist for most values of  $C_p$ , and except when the pressure sensor faces separated flow. This problem worsens when noise is present in the pressure measurements. Information from different pressure sensors helps in this non-uniqueness problem but only to a limited extent. This becomes a serious problem when the number of sensors are limited to four, one on each side of the RALS tower, or at lower speeds where the signal to noise ratio drops significantly due to lower pressure magnitudes (Fig. 3.10 (b) ). First an inverse approach was developed to predict the yaw angle, however, it did not perform up to expectations. After much experimentation, a simple forward problem approach was developed to estimate the yaw angle which is discussed next and the inverse problem approach is presented in Appendix A.

In this approach, the forward problem of approximating  $C_p$  as a function of the yaw angle  $\beta$  is addressed. It is well known from fluid mechanics that in external flow  $C_p$  is a function of both  $\beta$  and the surface pressure sensor position,  $\theta$ . Constructing a surrogate model of  $C_p$  as a function of both  $\beta$  and  $\theta$  is equivalent to approximating  $C_p$  for each pressure sensor as a function of just the yaw angle  $\beta$ . Even though this increases the memory requirements, it yields faster and more accurate models for  $C_p$ . Once a network

has been constructed to represent each sensor, the objective function in Eq. (3.3) can be minimized to predict the yaw angle for a vector of test pressure coefficients:

$$\hat{\beta}_{pre} = \min_{\beta} \sum_{i=1}^{NS} \left( \frac{p_{test}(i)}{\hat{q}_{pre}} - \hat{c}_{p,i}(\beta) \right)^2 \quad (3.3)$$

Here  $NS$  is the number of pressure sensors,  $\hat{c}_{p,i}$  represents the approximated  $C_p$  for the  $i^{th}$  pressure sensor,  $p_{test}$  is the test vector of pressure values and  $\hat{q}_{pre}$  is the predicted dynamic pressure for  $p_{test}$ . A simple line search for  $\beta$  would suffice for the above minimization problem. However, searching for  $\beta$  over the entire range of  $[-180,180]$  degrees would be wasteful, so  $\beta$  could be searched over  $[\chi - \iota, \chi + \iota]$  where  $\chi$  is the pressure sensor position that bears the maximum static pressure value and  $\iota$  is a value of the yaw angle chosen by the user so that range for the line search is sufficiently large. Because L-2 norm is the most resilient to random noise it was chosen to express the differences between the test  $C_p$  and predicted  $C_p$ . Also, Eq. (3.3) allows the user to put weights on the contribution of each pressure sensor. Different weights could be helpful in situations where there is a combination of high and low fidelity sensors, or in the case of sensor damage or local flow field disturbance. Besides ease of learning, better redundancy to noise this wind direction estimation approach uses all the available sensors for prediction instead of just those in a quadrant.

### 3.3 Results

In this section wind speed and direction estimation accuracies are presented. Results for both the surface vessel and RALS tower are presented using the estimation approaches discussed in Section 3.2. The SFA algorithm was implemented using the MATLAB programming environment on a Windows-configured PC with a Pentium 4 2.66 GHz processor and 1.0 GB of RAM.

#### 3.3.1 RALS tower

Wind tunnel data for the RALS tower was available at speeds ranging from 40 fps to 120 fps as wind direction was varied from -180 to +180 degrees at increments of 2 degrees. All available data points were used in the test set, while a training set was created by taking data present at increments of 4 degrees at each wind speed. Once the training and test sets were constructed one network was constructed to predict dynamic pressure according to Eq. (3.1). The tolerance parameter was kept close to zero so as to minimize any errors in the approximation of  $C_p$ . For dynamic pressure prediction, the RALS tower problem is at a disadvantage compared to the surface vessel problem because it has only four pressure sensors. Figure 3.11 (a) shows the prediction errors if SFA was used to construct one RBF network for wind speed prediction as a function of static pressure. However, there are several errors larger than the tolerance of 3.4 fps at 120 fps and on a closer look one realizes that most of the errors occur at yaw angles in the vicinity of 0, 90, 180 and -90 degrees. One possible explanation for the wind speed errors displaying a bias at the highest wind speed is shown in Fig. 3.11 (b). In this problem, a hyper-cone is approximated with Radial Basis Functions (RBF). RBFs model

the smooth slopes of the hyper-cone, however they do not approximate the flat top of the hyper-cone that exists due to the wind speeds of the highest magnitude.

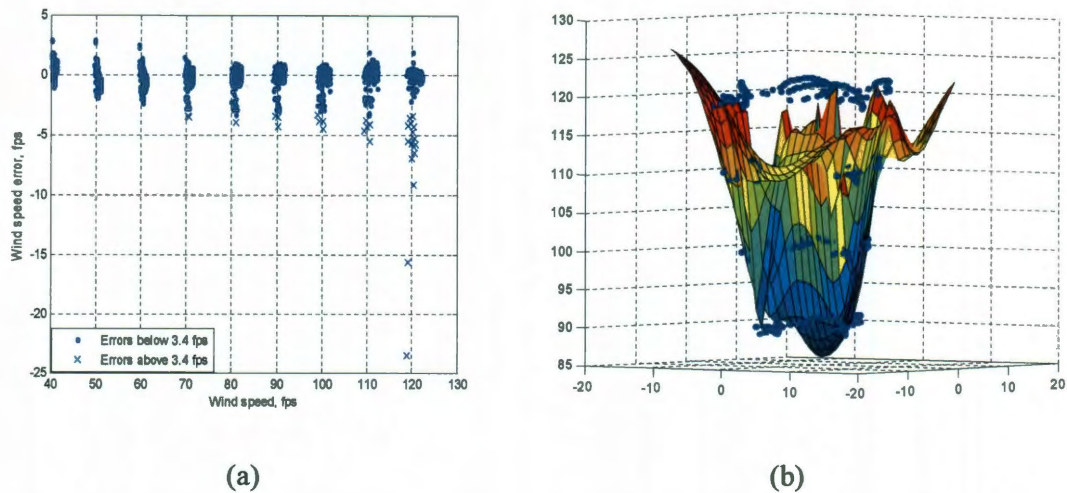


Figure 3.11 a) Wind speed prediction on RALS tower data using just one network for wind speed given by Eq. (3.1) and b) Example surrogate of the wind speed hyper-cone in 2 dimensions.

One simple solution to avoid this problem is to add fictitious points at higher speeds to the training set. One could calculate the coefficient of pressures given the available training data and use the coefficients to estimate the static pressures at higher speeds of 130 and 140 fps. One could train on this augmented training set and test on the available RALS tower data and reduce prediction errors at the highest wind speed.

There is another heuristic which could be used to reduce the errors shown in Fig. 3.12 (a). As mentioned before most of the errors occur in the vicinity of the 0, 90, 180, -90 degrees. For the following ranges of the yaw angle  $|\beta| \leq 25^\circ$ ,  $|\beta - 90^\circ| \leq 25^\circ$ ,  $|\beta| \geq 155^\circ$ , and  $|\beta + 90^\circ| \leq 25^\circ$ , the corresponding pressure

sensors face a head-on freestream wind and bear a positive pressure value and the remaining sensors bear a negative pressure value. This a priori information can be used in conjunction with the dynamic pressure model to improve prediction in the following manner. Training points lying in the above mentioned range could be set aside before starting to construct Eq. (3.1). During testing, if a test point is such that only one sensor bears positive pressure value, the dynamic pressure of that test point could be predicted using a simple nearest neighbor search from the training points that were set aside. Prediction of dynamic pressure is coupled with coefficient of pressure or the yaw angle. If  $\beta$  of a test point is known a priori, linear dependence of dynamic pressure with static pressure could be easily used to predict the dynamic pressure of a test point. With the help of this heuristic one can approximately deduce  $\beta$  and use linear dependence of dynamic and static pressure to predict dynamic pressure and thereby wind speed of a test point.

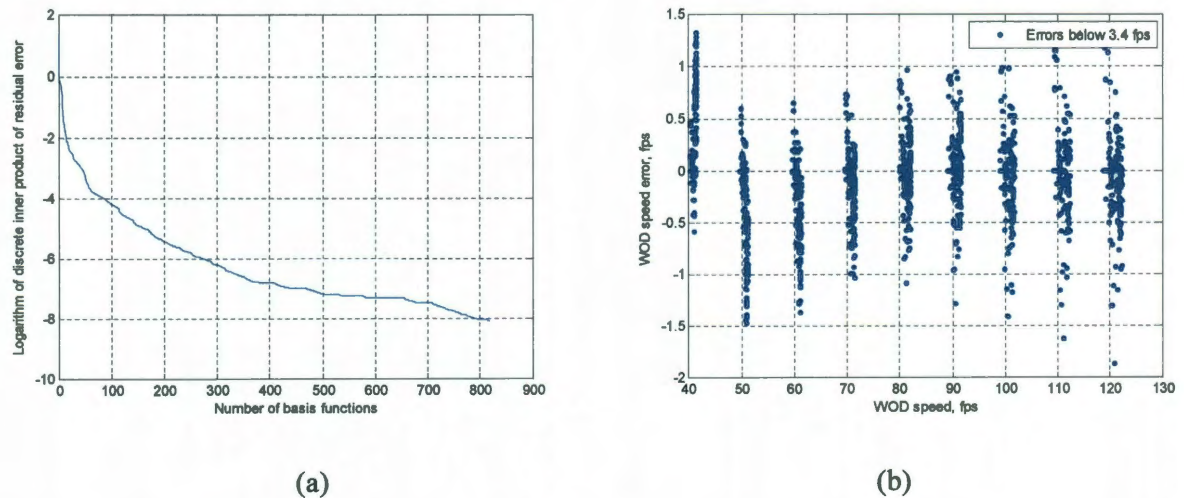
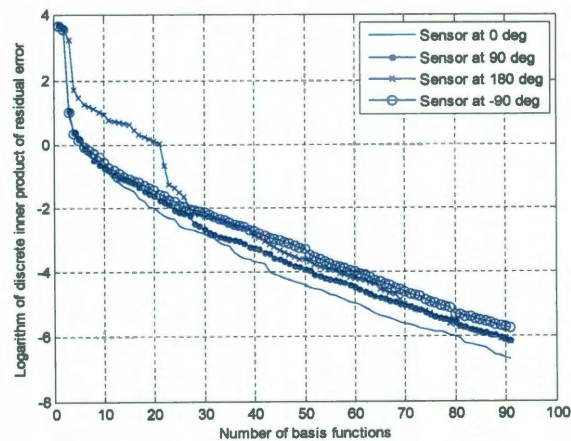


Figure 3.12 a) Logarithm of discrete inner product norm of residual error and b) Wind speed prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle.

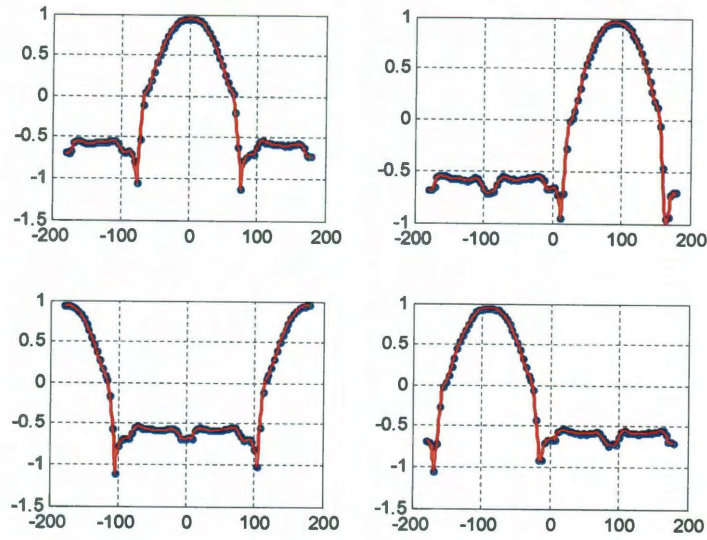


Figure 3.12 (a) and 3.12 (b) show convergence of residual error and wind speed prediction errors on the test set, respectively. Blue dots show errors below the acceptable error tolerance of 3.4 fps. The maximum error of 25 fps shown in Fig. 3.11 (a) was reduced to less than 2 fps with the help of the nearest neighbor heuristic.

Once the wind speed predictor was in place, the predicted dynamic pressure values were used to obtain test coefficient of pressure values. The resulting test  $C_p$  values were then input to the forward approach discussed in Section 3.2.2. To implement the forward problem approach first four models of pressure coefficient was constructed as a function of the yaw angle. Figures 3.13 (a) and 3.13 (b) show the residual error convergence and approximation of  $C_p$  for each pressure sensor respectively. Since sufficient noise-free training data was available, tolerance was kept low and so the  $C_p$  models interpolate the wind tunnel data accurately.



(a)



(b)

Figure 3.13 a) Residual error convergence and b)  $C_p$  prediction by a model of each pressure sensor. Blue dots represent wind tunnel data and the red line represents approximation by SFA as a function of the yaw angle. Clockwise from top plots show  $C_p$  model of pressure sensor at 0, 90, -90 and 180 degrees.

Ability of SFA to handle noise and sparsity in training data will be covered in the next chapter. Once the  $C_p$  models have been constructed the objective function shown in Eq. (3.3) was minimized by a line search to predict the yaw angle for each test point. Figure 3.14 (a) shows an example of the logarithm of the objective function for a test point with a yaw angle of zero degrees.

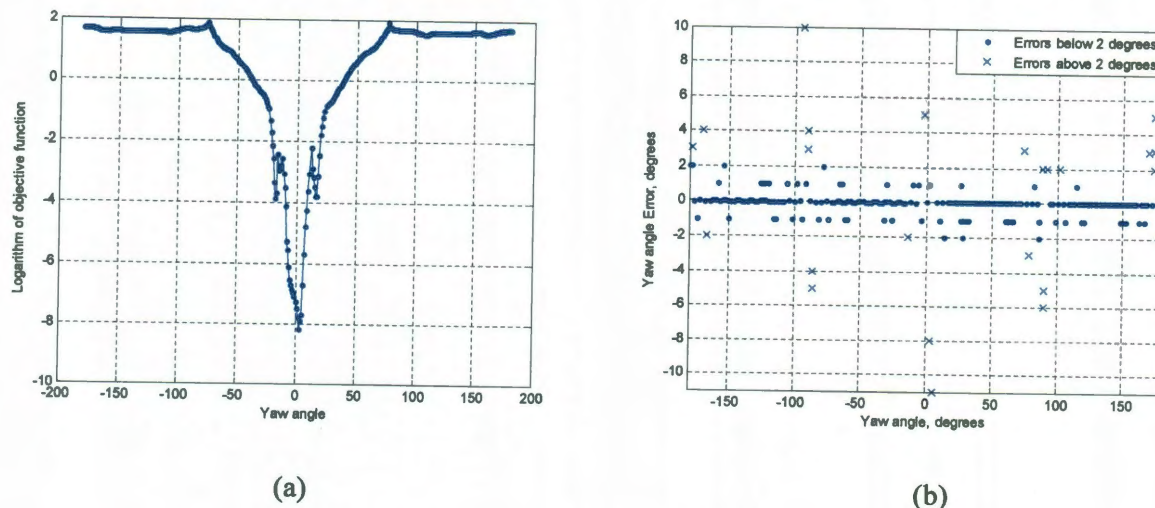


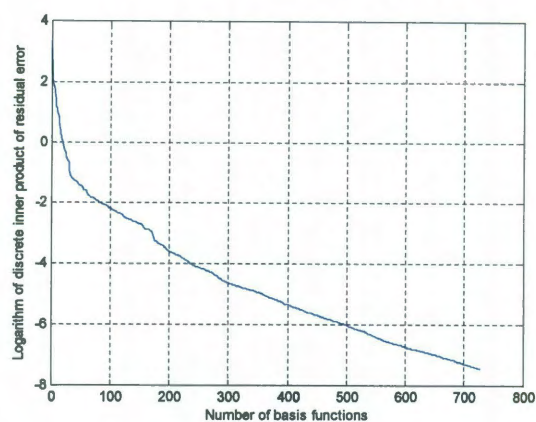
Figure 3.14 a) Logarithm of objective function of Eq. (3.3) obtained after a line search conducted for a test point with a yaw angle of 0 degrees and b) Yaw angle prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle using the forward approach.

Figure 3.14 (b) shows yaw angle prediction errors on the RALS tower data using the forward problem approach. Improvement over the inverse problem approach (Fig A.3) is clearly demonstrated in the results of both approaches. The maximum yaw angle prediction error is reduced from 45 degrees to 11 degrees.

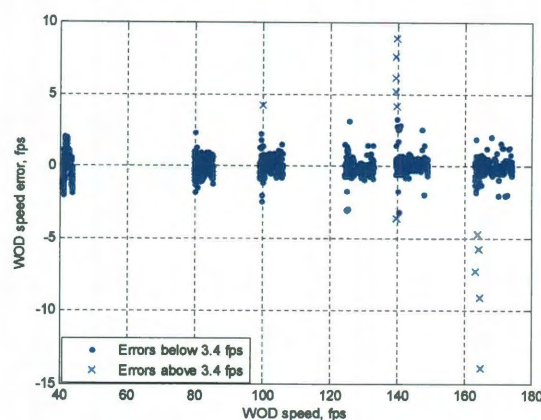
### 3.3.2 Surface Vessel

Wind tunnel data for the RALS tower was available at speeds ranging from 40 fps to 175 fps when wind direction was varied from -180 to +180 degrees at increments of 2 degrees. All available data points were used as test set, while a training set was created by taking data present at increments of 4 degrees at each wind speed. Once the training and test sets were constructed one network was constructed to predict dynamic pressure

according to Eq. (3.1). The tolerance parameter was kept close to zero so as to minimize any errors in the approximation of  $C_p$ .



(a)



(b)

Figure 3.15 a) Logarithm of discrete inner product norm of residual error and b) Wind speed prediction errors on the surface vessel data when training was conducted on data at every 4 degree increments of yaw angle.

Figure 3.15 (a) shows the convergence of the residual error and 3.15b shows the prediction errors when one RBF network is constructed by SFA to learn freestream wind speeds. Even though fifty pressure sensors are evenly distributed around the geometry of the surface vessel, prediction errors still show a bias as wind speeds increase. Again most of the errors occur in the vicinity of 0, 90, 180 and -90 degrees. Both the heuristics, namely, augmentation of training set with fictitious points at higher speeds and a nearest neighbor search at select yaw angles, can be used in the current problem. However, developing a nearest neighbor heuristic for this problem would be overly complicated because of the large number of pressure sensors. Since there were only 4 sensors present in the RALS tower problem it was easier to inspect the pressure values to determine the

yaw angle and devise a nearest neighbor search. Such a heuristic is possible for this problem but would be harder to assemble than the nearest neighbor heuristic. The other solution of adding fictitious points to the training set provides a more logical way to handle the bias in prediction errors and simpler to implement. Figure 3.16 shows the improvement in prediction errors when fictitious data points at 190 and 200fps were added to the training set. The maximum errors of 15fps shown in Fig. 3.15 (b) are reduced to about 5.5 fps with the help of this heuristic.

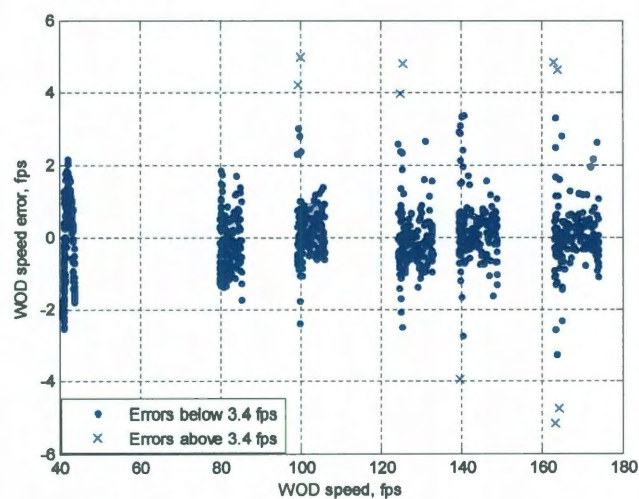


Figure 3.16 Wind speed prediction on the surface vessel data using one network for wind speed given by Eq. (3.1) with fictitious points at higher speeds added to the training set.

Once the wind speed predictor was in place, the predicted dynamic pressure values were used to obtain test coefficient of pressure values. The resulting test  $C_p$  values were then input to the forward approach discussed in Sections 3.2.2. Figure 3.17 shows yaw angle prediction errors on the surface vessel data using the forward problem approach. Improvement over the inverse problem approach (Fig. A.4) is clearly demonstrated in the

results of both approaches. The maximum yaw angle prediction error is reduced from 55 degrees to 4 degrees. The errors appear to be in increments of 1 degree because the line search for the yaw angle was done between -180 and +180 degrees at an interval of a degree. The minimum difference between the predicted and true yaw angle would correspond to grid size of the line search. In fact, an error of 4 degrees using this approach would mean that the actual error is between 3 and 4 degrees.

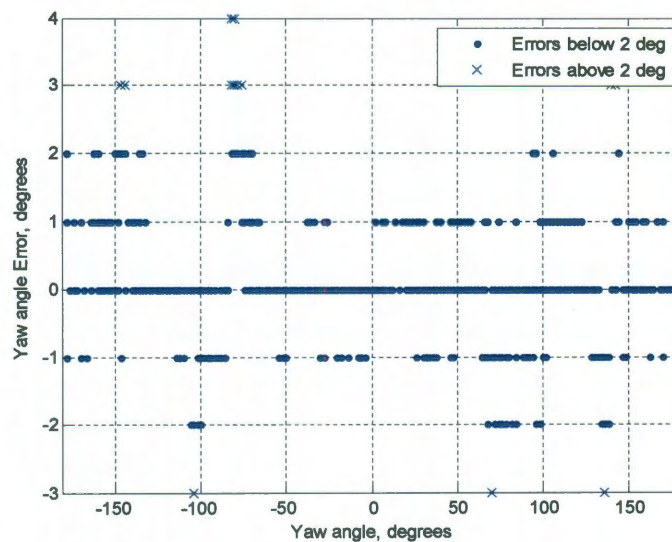


Figure 3.17 Yaw angle prediction errors for the surface vessel problem using the forward approach.

The wind speed and direction estimation techniques discussed above are general in nature and could be formulated with any learning technique other than SFA. However, motivation to use SFA is due to several reasons. First, the residual error convergence curves shown in Figs. 3.12 (a), 3.13 (a), 3.15 (a) demonstrate exponential convergence of residual error after the addition of the first few basis functions. The error decreases super-

exponentially in most cases for the first few basis functions. Improvement is possible if the basis centers are also optimized instead of heuristically placing them over data points corresponding to the maximum absolute value of the residual. Second, SFA does not need any grid search or cross-validation studies to be performed for control or kernel hyper-parameters. This makes the approach easy to use, since the user needs to load only the training and test sets and self-adaptive which makes SFA independent of any user chosen values of parameters. Another advantage of SFA is that the resulting surrogate model can be used to determine input sensitivities, which is discussed in the next section.

### 3.4 Pressure Sensor Sensitivity

One element of the surface vessel problem was to determine which pressure sensor displayed low sensitivity. Reducing the number of necessary pressure sensors will lower the cost of integrating the pressure measurement system with the surface vessel. Besides lowering the cost it will allow the engineers to explore new locations to mount pressure taps, make the calibration of the pressure measurement system faster, and help the engineers accelerate through the test matrix. In the field of machine learning the problem of finding the most sensitive input variables is known as feature selection. In the current work, a partial derivative method was chosen since the authors view the problem of regression as a function approximation problem. Partial derivative of the approximating function, given by Eq. (2.1), was calculated with respect to each pressure sensor. Rewriting Eq. (2.1) as:

$$y_n^a(x) = \sum_{i=1}^n c_i (\phi(x, \phi_i) + b_i) = \sum_{i=1}^n c_i \left( \exp \left[ \ln(\lambda_i) \sum_{j=1}^d (x_j - x_{i,j}^*)^2 \right] + b_i \right)$$

and taking the partial derivative with respect to  $k$ th measurement of the  $j$ th pressure sensor as shown in Eq. (3.4)

$$\frac{\partial y_n^a}{\partial x_{k,j}} = \sum_{i=1}^n 2|x_{k,j} - x_{i,j}^*| c_i \phi_{i,k} \ln(\lambda_i) \quad (3.4)$$

where  $\phi_{i,k} = \exp\left[\ln(\lambda_i) \sum_{j=1}^d (x_{k,j} - x_{i,j}^*)^2\right]$ ,  $i = 1, 2, 3, \dots, n$ ,  $j = 1, 2, 3, \dots, d$  and  $k =$

$1, 2, 3, \dots, s$ . The input sensitivities ( $\delta_j$ ) were determined by summing the squares of the derivatives over the number of training points as shown in Eq. (3.5).

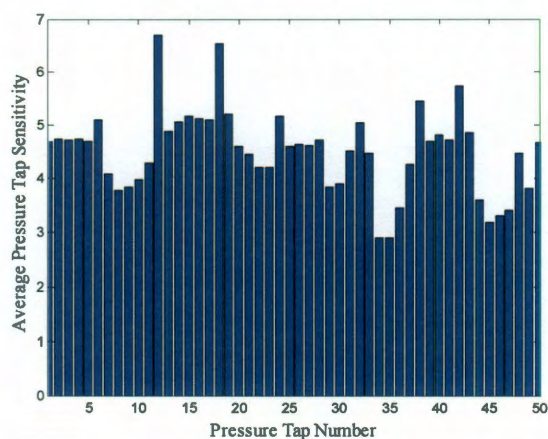
$$\delta_j = \frac{1}{s} \sum_{k=1}^s \left( \frac{\partial y_n^a}{\partial x_{k,j}} \right)^2 \quad \text{where } j = 1, 2, 3, \dots, d \quad (3.5)$$

The parameter  $x$  in Eqs. (3.4) and (3.5) represent the static pressure used for both dynamic pressure and coefficient of pressure predictions. For computing input pressure sensitivity, the inverse problem approach (Appendix) was selected as it is a straightforward approach to computing input pressure sensitivity with respect to wind speed and direction prediction. Pressure sensor sensitivities were computed only for the surface vessel problem since the RALS tower had only 4 non-redundant sensors. The pressure sensor sensitivities were determined by summing the squares of the derivatives over the bases used to construct the approximating function. This was done once for each of the four networks and an average taken to find the final input sensitivities. The input sensitivity values obtained depend on the training and test set chosen. Consequently, this calculation was performed on 20 randomly chosen training data sets and an average was

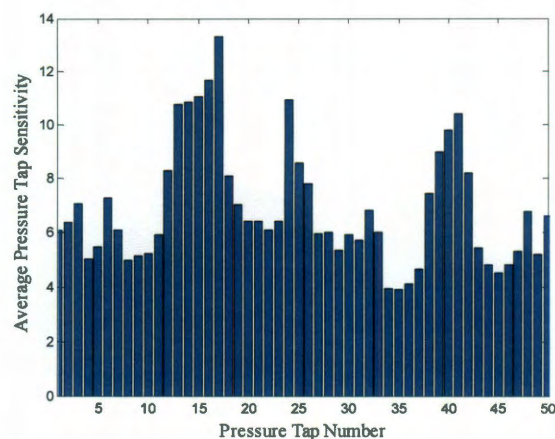


computed. Figures 3.18 (a) and (b) display these values for wind speed and yaw angle, respectively. It can be seen from Fig. 3.18 that the average pressure sensor sensitivity for yaw angle prediction varies more compared to wind speed prediction.

For yaw angle prediction the pressure sensors mounted on and close to the stern side have greater sensitivity for port, stern and starboard side winds and less sensitivity values for bow side winds. Consequently their average sensitivity values are greater than the sensors mounted on the bow side.



(a)



(b)

Figure 3.18 Average input sensitivities in the prediction of (a) wind speed and (b) ship bow yaw angle.

### 3.4.1. Graceful degradation

Figure 3.19 shows the degradation in wind speed and yaw angle percentage prediction accuracy as the least important pressure taps are removed. Yaw angle prediction accuracy decreases more rapidly than the wind speed prediction accuracy. In general, the wind speed prediction is insensitive to the location of pressure sensors on the surface vessel, while sensors mounted on and close to the stern side are more important for yaw angle prediction. Figure 3.20 shows the convergence rate of the wind speed and direction percentage prediction errors as the number of training points increase. This plot of the convergence rate demonstrates the ability of SFA to learn the functional relationship between the pressure measurements and air data parameters. With the exponential convergence rate displayed in Fig. 3.20, SFA has the potential of predicting how many more data sets are required in an experiment to achieve a desired level of modeling error.

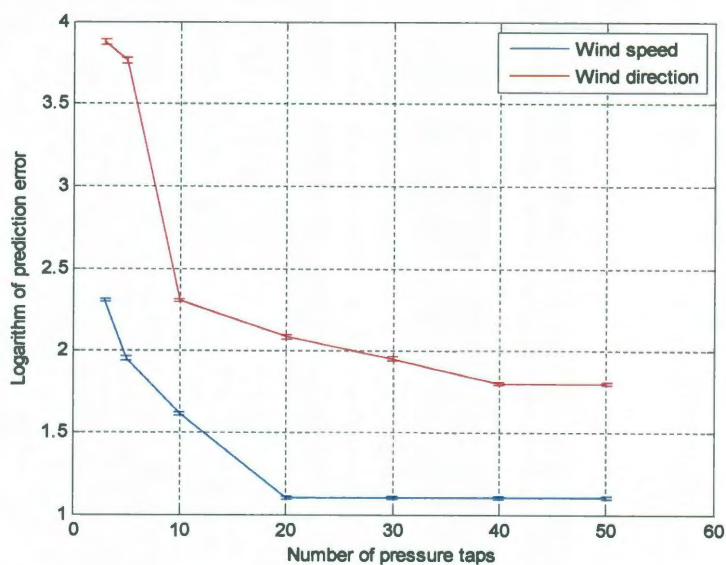


Figure 3.19 Logarithm of prediction error degradation with the number of pressure taps. The bars show the standard deviation of the errors.

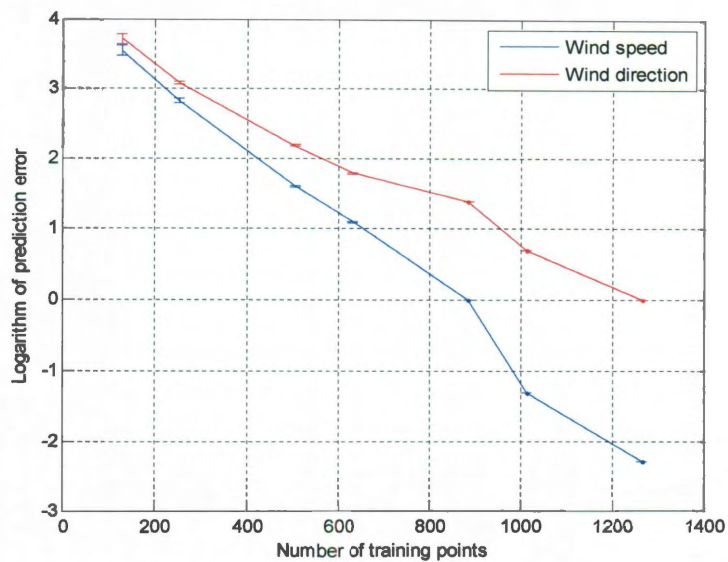


Figure 3.20 Logarithm of prediction error degradation with the number of training points. The bars show the standard deviation of the prediction errors.

## Chapter 4

### Data-Model Fusion

In this chapter we explore the use of data-model fusion in improving the performance of the FADS system discussed in Chapter 3. As mentioned in Chapter 1 both EFD and CFD methods have shortcomings when tested on complex, multi-component mechanical and aerospace engineering systems. Despite a century for EFD and three decades of research in CFD, neither tool is yet self-sufficient in analyzing an engineering system. We are at a stage in fluid dynamics where integration of experimental and computational methods are not merely used for verification and validation, but necessary for analysis and design. The concept of EFD and CFD integration has been exclusively studied at the Japan Aerospace Exploration Agency (JAXA) in the name of the Hybrid Wind Tunnel (HWT) project [53].

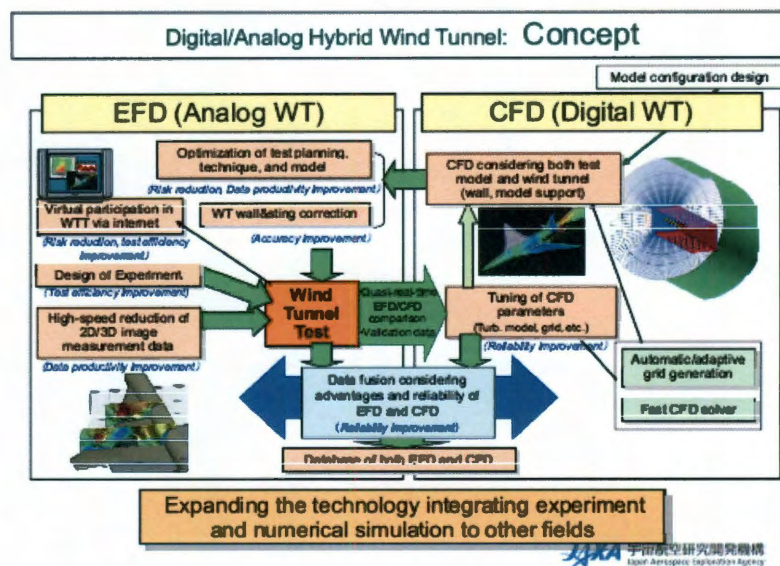


Figure 4.1 Concept of a hybrid wind tunnel [53].

The Hybrid Wind Tunnel is the result of synergy between Analog Wind Tunnel (EFD) and Digital Wind Tunnel (CFD). Few functionalities of the HWT include a priori use of CFD to prepare the EFD test matrix, use of CFD to correct model and wall support interference effects, CFD data refinement based on EFD data and also data fusion between CFD and wind tunnel data. Key technical challenges in successful use of a HWT include availability of fast automatic grid generation and solvers and high speed data reduction of imaging techniques. EFD and CFD integration techniques include rapid qualitative and quantitative comparison resulting in the evaluation of the wind tunnel data. Integration of EFD and CFD has also been independently studied by a number of researchers in fluid dynamics.

Nisugi et al. [54] have extensively worked towards developing a systematic approach for real time integration of experimental and numerical results in the hope of resulting in a more accurate solution that also facilitates the analysis of the flow. In emulating a real fluid flow problem, numerical simulations are conducted with boundary conditions that comprise of the feedback error between the actual and the simulation output and a feed-forward signal is used to adjust the upstream velocity boundary condition. This concept of the hybrid wind tunnel was evaluated on a fundamental problem of Karman vortex sheet in the wake of a square cylinder. The proposed system could predict the flow oscillations exactly like the experiment whereas the ordinary numerical simulation is never as accurate as the wind tunnel experiments. Also, the hybrid wind tunnel system can give more detailed information about flow domain as compared to the wind tunnel experiments.

The authors in the reference [55] addressed the differences in the CFD and wind tunnel data due to Reynolds number effects, and optimized the sting blade support to minimize the effect of support corrections and the associated uncertainty. The authors used a CFD model of flow past the model sting to record the changes in the freestream Mach number and the incident angle of attack and used the results to obtain equivalent upstream conditions to model flow past model with sting. Rufolo et al. [56] and Pettersson [57] address the problem of integrating wind tunnel and CFD data to obtain free flight data. Wind tunnel data often have discrepancies from free flight conditions because of wall/model support effects and differences due to lower Reynolds number in the wind tunnel. This discrepancy from the real flight data has been proposed to be filled by deriving analytical scaling laws based on CFD data. For example, the authors used a polynomial-log functional form to relate Reynolds number to its pressure correction effects. These analytical scaling laws were calibrated from CFD data and added to the base wind tunnel data to achieve real flight conditions. Pretest CFD planning is especially helpful in predicting the effects of nonlinear aerodynamics when a novel aircraft is being tested. Accurate prediction of boundary layer transition is very desirable and scaling wind tunnel data to free flight conditions can induce uncertainties in the flow characteristics during boundary layer transition. In such cases LES, DNS or hybrid semi-empirical methods can be used to reduce the uncertainties in boundary layer transition at Reynolds number higher than the wind tunnel can handle [58].

Planquart et al. [59] demonstrated three examples where an integrated EFD/CFD approach was used for design and analysis. The first project dealt with the design and

performance evaluation of a solar blade protection system to be installed on a building in presence of wind loads. Wind tunnel tests were conducted to choose the fluid structure interaction coupling and CFD analysis was conducted to optimize the design of the blade. The performance of the design of the blades under heavy wind loads and their vibration structure was studied experimentally. The second project dealt with the design of a new polar station. The conceptual design phase of the building was handled purely by wind tunnel experiments. The detailed design phase of the building included calculation of the aerodynamic loads on the building corresponding to different configurations in a CFD model. The final project dealt with the aerodynamic design of an ultra-streamlined land vehicle. CFD simulations were used to narrow down the final design candidates of the car and the design was finalized by conducting wind tunnel simulations. Once a design was finalized, parametric tests were conducted to create a data set of aerodynamic coefficients.

Jouhaud et al. [60] addresses problems in verification and validation which include limited availability of high resolution/detailed reference data and uncertainty in the definition of validation cases. The authors propose to use the kriging approach to develop a response surface where the inputs are spanned by uncertainties in the numerical model, the turbulence model and parameters like the Mach number and the angle of attack. The surrogate model prepared by kriging requires CFD solutions only on a few input combinations. Once a response surface has been established, the optimal corrections for wind tunnel data are derived by searching the response surface for its global optimum.

The nonintrusive nature of the proposed approach and the ability of kriging to work with discontinuities in the response surface make it a strong approach.

#### **4.1 Fusion of variable fidelity models**

A primary technique for EFD/CFD integration is fusion of solutions of variable fidelity. High fidelity solutions may be costly to obtain and so might be discrete or fewer in number whereas low fidelity solutions, however inaccurate, might be continuous. Such aerodynamic data fusion has been made popular by Unger [61] where high fidelity solutions were used to provide absolute values while the low fidelity solutions were used to provide trends. The high and low fidelity solutions can be EFD and CFD simulations, CFD simulations of variable fidelity or experimental data from different sources. Even though surrogate models can be constructed without knowledge of the governing equations, the idea here is to study how a metamodel can be constructed using the domain specific knowledge in the hope of increasing the accuracy of the model. In engineering problems, there can be a number of situations where several solutions of variable fidelity are available. Some examples include, the high fidelity solution can be one with a finer mesh compared to a lower fidelity coarser solution. The high fidelity solution can be a result of better physical models, for example Navier-Stokes versus Euler equations. The high fidelity solution can be a fully convergent solution compared to a partially converged low fidelity solution. And the low fidelity solution could be a result of semi-empirical approximation, as is common in concept design studies [62].



Haftka [63] introduced zero and first order scaling approaches, called global local approximation strategy, where accurate local solutions were used to correct lesser accurate global solutions. Multiplicative and additive scaling factors were introduced which were used for constant correction in the zero order scaling, whereas, in the first order scaling approach the first order Taylor expansion of the scaling factor was used to correct the global solution. The accuracy of zero order scaling approach degrades significantly with distance. Even though the first order approach is more accurate than the zero order scaling approach, it can only be applied when the first order derivatives of the local and global solutions are available. Tang et al. [64] also address the problem of data-fusion where data from different sources and variable fidelity are combined into a single package. This data fusion approach allows the user to choose the flow solvers of variable sophistication depending on the requirement of accuracy in the flow field. To fuse low and high fidelity solutions scaling laws have to be determined which computes either differences or ratios of high and low fidelity solutions. Response surface modeling techniques or algorithms similar to kriging can be used to construct a model of the discrepancy between the two solutions and correspondingly a fused solution could be achieved by fusing few high fidelity solutions with many low fidelity solutions.

Eldred et al. [65] address the problem of convergence rates of model hierarchy surrogate based optimization where consistency between the surrogate models and the truth is enforced via corrections. The authors show that the first order additive and multiplicative scaling achieves consistency only about a single point and second order correction methods outperform the former. Again, even though the second order scaling

approach would be better than the first order approach, it could only be applied when the Hessian of the low and high fidelity solutions are available. The authors propose the use of quasi-Newton and Gauss Newton approximation of the derivatives that make use of the first order sensitivity information. The authors also proposed a weighted combination of additive and multiplicative scaling.

The zeroth, first and second order scaling methods primarily improve the local accuracy of the low fidelity solution. An effective way to improve the global accuracy of low fidelity solutions is to use nonlinear regression techniques like kriging, neural networks, radial basis functions or sparse kernel methods to construct a hyper-surface of the scaling factor instead of the high fidelity solution. Naverrete et al. [66] investigated the usefulness of such global scaling using surrogate models in estimating airfoil characteristics using incomplete airfoil tables and low fidelity CFD solutions. Naverrete tested the validity of the proposed approach on two airfoils NACA 0012 and SC1095. The NACA 0012 airfoil is very popular and has one of the most complete airfoil tables. This problem was used to calibrate the parameters of the fusing approach namely the appropriate scaling of the multi-dimensional data, the appropriate training tolerance value and the optimization network parameter. For the SC1095 airfoil test case the coefficients of lift, drag and moment were approximated as a function of independent variables Reynolds number, Mach number and angle of attack. Figure 4.2 shows the effectiveness of the fusing approach where the high fidelity wind tunnel data is available only for a limited range of angle of attack while the CFD solution is continuous over the whole domain but of lower fidelity.

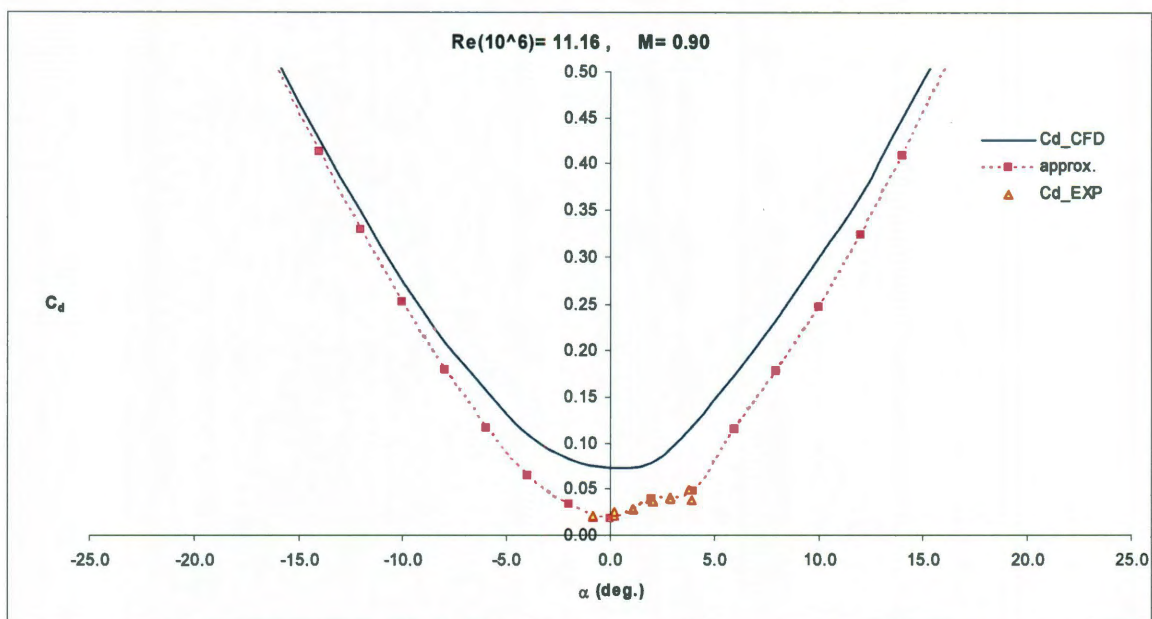


Figure 4.2 The drag bucket for SC1095 at  $Re = 11.16 \times 10^6$ ,  $M = 0.90$  [66].

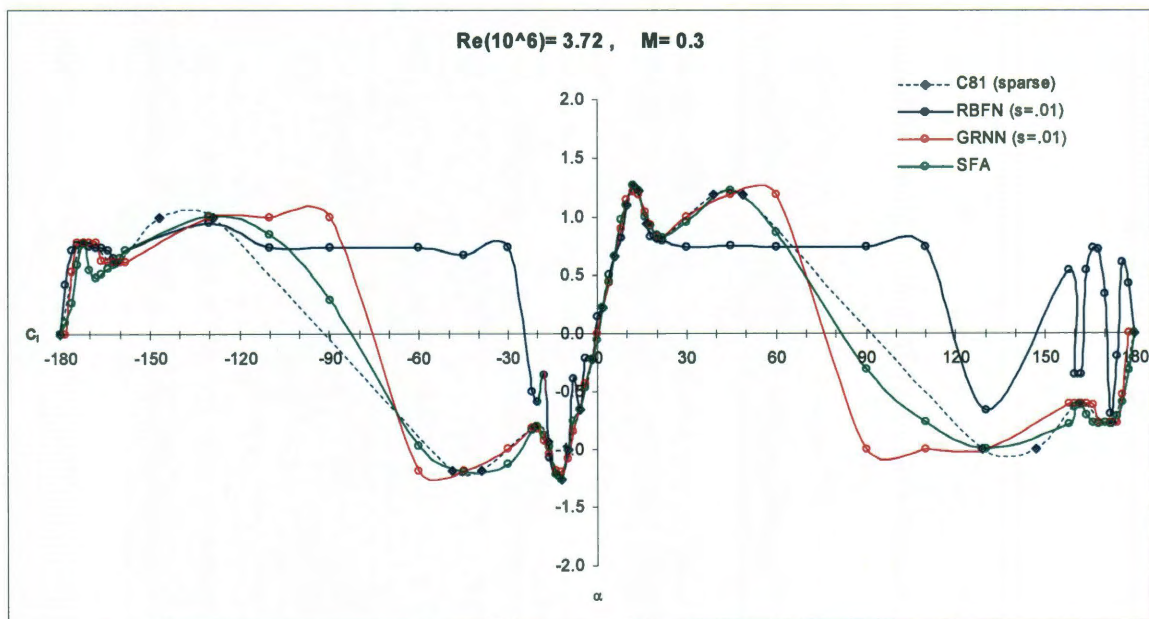


Figure 4.3  $C_l$  vs.  $\alpha$  for NACA 0012 interpolation test comparisons.  $Re = 3.72 \times 10^6$ ,  $M = 0.30$  [66].

Naverrete also compares the ability of several approximation tools namely Radial Basis Function Networks (RBFN), Generalized Regression Neural Network (GRNN) and SFA to reproduce the airfoil coefficients of the NACA 0012 airfoil given sparse high fidelity data chosen from the C81 tables which are exhaustive coefficient tables for the airfoil. Figure 4.3 demonstrates the vulnerability of the approximation tools when approximating sparse high fidelity solution. As shown above, the approximation of sparse high fidelity solution can result in a model which is worse in some regions than the low fidelity solution. However, following trends of the low fidelity solution through the absolute values of the high fidelity solution gives a better result with reduced uncertainty.

Kennedy and O'Hagan [67] investigate how several codes of variable fidelity can be combined to estimate the output of a complex and sophisticated code of high fidelity. The authors propose a Bayesian approach of combining several surrogate models and are also capable of conducting uncertainty analysis of the resulting model. The primary assumption the authors make is that the output of various codes are correlated. In emulating a high fidelity code, the training process is augmented with output from low fidelity codes. The authors used a Gaussian process model to conduct the training, the uncertainty analysis and showed on a reservoir simulation problem that surrogates of comparable accuracy could be achieved using just a quarter of high fidelity solutions.

Keane [68] proposes how an empirical drag prediction tool, kriging response surface method and design of experiments and data fusion methods can be used in synergy with three dimensional CFD codes for wing design optimization. The author constructed a

response surface of the differences between an empirical drag prediction tool and a CFD tool. The primary advantage of using kriging to construct a hypersurface is its ability to predict the expected error of the model which could be used to decide where to add more points in the surrogate. The author concluded that the proposed fusion based approach is better than the direct search or a simple response surface optimization using only data from a three dimensional CFD code. Forrester et al. [69] show how global partially converged solutions could be combined with information about expected improvement updates could result in a faster construction of a more accurate surrogate. The author addresses the important question of at what input design point a fully convergent solution must be obtained that results in a better objective function. Criteria were suggested for determining the quality of an initial surrogate and the number of design of experiment based data points that should be used to make an appropriate decision where the next fully convergent simulation should be conducted.

#### **4.1.1 Relation to Data Assimilation**

The concept of fusion of high and low fidelity solution in aerodynamics is similar to the analysis part of data assimilation which is popular in weather forecasting. In the analysis section of the data assimilation procedure, the true state has to be estimated from observations. The output of this step feeds into as input to the next step, for example an estimate of the atmosphere could be used as initial condition to numerical weather forecast. If the number of observations is more than the number of unknown variables, it results in an over-determined system and can be solved by interpolation techniques. However, generally systems are under-determined and some background or prior

information is needed to estimate the true state. This background information is provided by data from physical models. So the problem at hand becomes fusion of observed data and data from physical models in order to estimate a better understanding of the true state. Cressman analysis [70] expresses the fused solution or the analysis as a weighted sum of the observed values and the background solution. The analysis is weighted such that it would coincide with the discrete observed values and the solution would decay to background as distance from the observed value increases. The weights given to each observation point depend on the radius of influence set by the user as shown in Eq. (4.1).

$$x_a(j) = x_b(j) + \frac{\sum_{i=1}^n w(i, j) \{y(i) - x_b(j)\}}{\sum_{i=1}^n w(i, j)} \quad (4.1)$$

$$w(i, j) = \max\left(0, \frac{O^2 - h_{i,j}^2}{O^2 + h_{i,j}^2}\right)$$

Here  $x_a$ ,  $x_b$  and  $y$  are the analysis, the background solution and observed data respectively. The weights  $w$  are a function of the radius of influence the radius of influence  $O$ , and the distance of the  $i^{th}$  observed value to the  $j^{th}$  value of the background solution  $h_{i,j}$ . Cressman analysis and its variants are simple to apply, however, its performance degrades if there is noise present in the observed data, there is no optimal way to determine the weights, the resulting analysis is not guaranteed to be smooth and the method does not take into account the distribution of the observed values relative to each other.

A more principled approach to estimate the analysis in the data assimilation literature is of statistical interpolation with least squares estimator [70]. The optimal least squares estimator or the best linear unbiased estimator can be determined by solving the following optimization problem in Eq. (4.2).

$$\begin{aligned} x_a &= \min_x J_b(x) + J_o(x) \\ &= (x - x_b)^T B^{-1} (x - x_b) + (y - I[x])^T T^{-1} (y - I[x]) \end{aligned} \quad (4.2)$$

Here  $J_b(x)$ ,  $J_o(x)$  is the discrepancy of the analysis with the background and observation, the matrices  $T$  and  $B$  represent the covariance matrices of observation and background errors respectively, and  $I$  is an observation operator that facilitates the computation of differences between the observed and the background values. The resulting linear optimum unbiased estimator is a weighted sum of the background and observed values as a function of the optimal gain written as a function of  $I$ ,  $T$  and  $B$  shown in Eq. (4.3).

$$\begin{aligned} x_a &= x_b + K(y - I[x_b]) \\ K &= BI^T (IBI^T + T)^{-1} \end{aligned} \quad (4.3)$$

These concepts from data assimilation bear similarities with the current context of fusing solutions from different sources and variable fidelity. The approach of global scaling via surrogate modeling described in the previous section will be the approach adopted in the rest of this chapter. This approach is also directly related to the static data assimilation methods using Tikhonov regularization.

## 4.2 Tikhonov regularization and data-model fusion

When discussing the problem of learning from data it is important to focus on the role of regularization techniques used to arrive at the final solution. The concept of regularization arises from the solution of ill-posed inverse problems [71]. An inverse problem is ill-posed if any one of the three conditions of existence, uniqueness and stability is violated. Regularization techniques are designed to formulate the inverse problem in such a way that with some compromise of accuracy the resulting solution is uniquely solvable and is stable to noise in the measured data. Classical methods include 1) regularization by singular value truncation, 2) Tikhonov regularization and truncated iterative methods. The most popular and widely used method is the Tikhonov direct regularization method. For a linear system  $A\bar{x} = \bar{y}$ , the Tikhonov regularized solution  $\bar{x}_\omega^*$  for the regularization parameter  $\omega$  is found by solving Eq. (4.4):

$$\bar{x}_\lambda^* = \min_{\bar{x}} \|A\bar{x} - \bar{y}\|^2 + \omega \|\bar{x}\|^2 \quad (4.4)$$

Ulbrich [72] investigated the extension of Tikhonov regularization for nonlinear ill-posed problems and called it Generalized Tikhonov Regularization. In the nonlinear case also, the objective function is composed of two parts, one which measures the interpolation error of the solution and the second which constrains some linear or nonlinear functional of the solution. Learning problems where the objective is to determine an estimator from a finite number of training samples is directly related to ill-posed inverse problems. Several authors including Poggio and Girosi [27] have formulated the learning problem as regularized least squares which arises from the study of generalized splines. The final approximation  $f_\omega^*$  is determined by minimizing Eq. (4.5):



$$f_{\omega}^* = \min_f \frac{1}{s} \sum_{i=1}^s (f(x_i) - y_i)^2 + \omega \Omega(f) \quad (4.5)$$

where  $\omega$  and  $\Omega$  are the regularization parameter and a constraining functional on  $f$ . Poggio and Girosi [27] establish the connection between regularization approaches and network based approximation techniques like Generalized Radial Basis Function technique. Since nothing is known about the learning problem to begin with, a priori assumptions about smoothness have to be made. Tikhonov regularization techniques exploit the smoothness constraints by the formulating the objective function as a variational problem in Eq. (4.5). The regularization parameter  $\lambda$  controls the interpolation error of the approximators and its degree of smoothness. Starting the approximation as a linear sum of RBF's where the number of RBF units ( $n$ ) and the associated number of parameters ( $3n+1$ ) are much less than the number of training samples ( $s$ ), the problem becomes over-determined and thereby needing regularization. From regularization theory, Poggio and Girosi [27] derive that the regularized solution could be written as a linear of basis functions of the radial type.

Meade and Zeldin [26] investigate how Tikhonov regularization could be used to reformulate the learning problem where the smoothness constraints on the solution also include the smoothness of the low fidelity solutions. The authors provide a mathematical framework to fuse high fidelity experimental data with smooth, low fidelity CFD data. The fusion approach proposed by Meade and Zeldin can be interpreted as correcting low fidelity solutions with high fidelity data, or filling the gaps in the discrete experimental data with smooth physics based solutions. This idea of this approach is similar to the approaches discussed in Section 4.1. Reference [26] proposed to incorporate

mathematical models in the determination of the fused solution by using the quadratic energy form of the differential equations as the smoothing constraint in the regularized objective function. The regularization parameter was determined such that the approximation smoothed out the noise in the experimental data. Wang [73] extended this approach to fuse experimental data with inviscid-viscous solution of flow past the NACA 0012 and RAE 2822 airfoils. Reference [73] proposed the following objective functional for the learning problem in Eq. (4.6).

$$\gamma^\omega [f_a, f_{\text{exp}}, f_{\text{CFD}}] = \frac{1}{2} \omega \sum_i (f_{\text{exp}}(\bar{x}_i) - f_a(\bar{x}_i))^2 + \frac{1}{2} \int \sum_{r=0}^1 \left( \frac{d^r (f_a(x) - f_{\text{CFD}}(x))}{dx^r} \right)^2 dx \quad (4.6)$$

where  $\bar{x}_i = (x_{1,i}, x_{2,i}, \dots, x_{d,i})$  and  $d$  denotes the dimensionality of the problem. From this equation, it can be seen that for  $\omega \rightarrow \infty$ ,  $f_a(x) \rightarrow f_{\text{exp}}(x)$ , and the CFD data becomes less relevant than the experimental data. For  $\omega \rightarrow 0$ , the experimental data become less relevant to the solution. Considerable computation effort had to be invested in determining the optimal value of the regularization parameter. In order to avoid the computational expense, Navarrete and Meade [66] proposed a fusing approach where SFA was used to approximate the differences between the high and the low fidelity solution.

#### 4.2.1 Data-model fusion for FADS

Estimating freestream wind speed and direction from static pressure measurements is an ill-posed problem. The ill-posedness is induced due to the non-uniqueness of the problem which is worsened in the presence of noise in the pressure data. The inverse

problem addressed in this dissertation is to reconstruct a complete pressure signal given sparse, noisy and incomplete experimental data. This work develops an alternate method of fusing experimental and computational data to approximate pressure signals to estimate freestream wind speed and direction. The approach uses a neural network method as the inverse modeling tool and applies a simplified Tikhonov-related regularization scheme to correct for the original data error. The purpose of this section is to introduce a fusing approach using the SFA neural network that maximizes the use of experimental data with the help of CFD data in approximating a smooth, continuous and accurate pressure distribution. The fusing approach first involves calculating the error function of the CFD and experimental data defined by the following Eq. (4.7)

$$e(\vec{x}_i) = u^{CFD}(\vec{x}_i) - u^{EXP}(\vec{x}_i), \quad (4.7)$$

for  $i = 1, \dots, s$ , where  $s$  is the number of training data samples. The error vector,  $e$ , is then used to train the SFA network to a predetermined tolerance,  $\tau$ . The resulting error surface,  $e(\vec{x}_i)$ , will naturally involve some scatter directly related to the experimental data noise. Training the network to the given tolerance allows the SFA to regulate the noisy experimental data with a priori CFD information. Assuming the  $u^{CFD}$  surface is known, then the error surface approximation,  $e^{SFA}$ , can be subtracted from the  $u^{CFD}(\vec{x}_i)$  data to give the approximation in Eq. (4.8),

$$u^{SFA} = u^{CFD} - e^{SFA} \quad (4.8)$$

The  $\tau$  value can be regarded as the regularization parameter and controls how well the approximations fit the experimental or CFD data. On the one hand, a very high tolerance value allows the training process to end prematurely with very few network units. As a result, the network “under-learns” the training data and the majority of the

approximations reach a value of zero. For data points with an error value of zero, Eq. (4.8) shows that the approximation value will reproduce the CFD data. On the other hand, a very small tolerance value will force the network to use too many network units to reach the smallest possible tolerance. In this case, the network “over-learns” the training data and will fit even the experimental noise in the error surface. As a result, the approximations will reproduce the experimental data. The user must carefully choose the tolerance value to best fit the experimental data using the CFD information.

#### **4.2.2 Handling noise, sparsity and incompleteness**

As mentioned before, one of the objectives of this work is to improve the quality of the training data set by fusing numerical solutions with experimental data. In this section smoothness based regularized solutions were compared versus the physics based regularized solutions in their ability to handle noise, sparsity and incompleteness of data. If only experimental data points were used to construct the wind speed and direction surrogates, it is called smoothness based regularized solutions because the RBF network uses just the mathematical smoothness of Gaussian radial basis functions to construct the hyper-surface. However, if numerical solutions were used as a priori information to construct the surrogates they are referred to as physics based regularized solutions. In the following graphs, the smoothness based solutions are indicated by ‘SFA’ and the physics based solutions as ‘Fused’. The available experimental data for the RALS tower had 9 sets of wind tunnel runs from 40 fps to 120 fps at increments of 10 fps. Each set of wind tunnel run had pressure measurements in the range  $-180^\circ \leq \beta \leq 180^\circ$  at increments of 2 degrees. To simulate a noisy and sparse data set, a uniform random noise of magnitude

0.005 psi (approximately 4.5% of the maximum pressure magnitude) was added to each pressure measurement. From this noisy data set, pressure measurements at every 20 degrees were selected to simulate sparsity in the training set. The ability of smoothness based and physics based techniques to recover the original pressure signal is tested on this new degraded subset of the data set. The numerical simulations were conducted using the commercially available Star-ccm software. Steady state three-dimensional flow around RALS tower was solved using Reynolds-Averaged Navier Stokes equations.

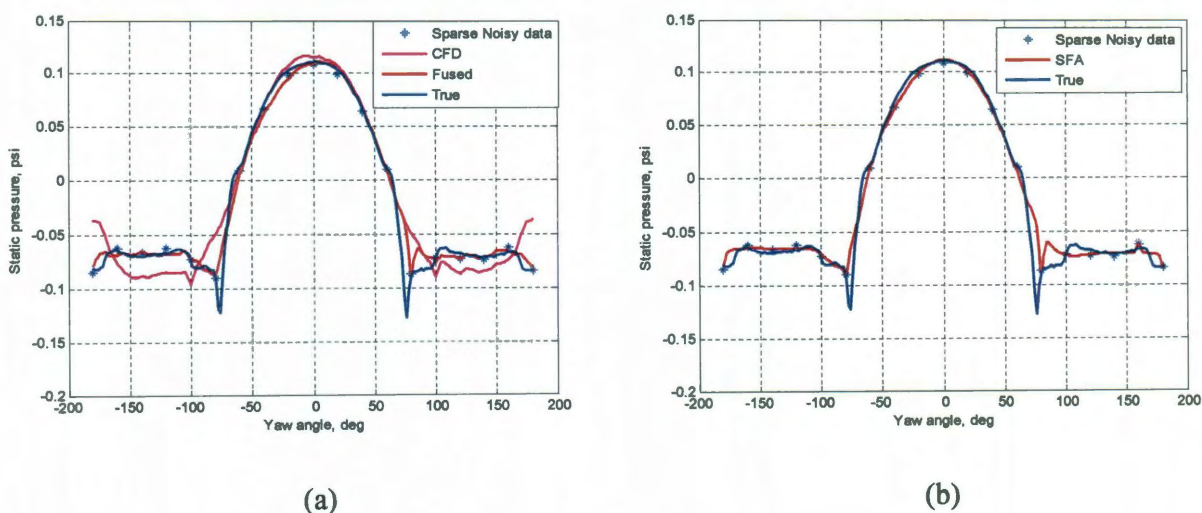


Figure 4.4 Comparison of RALS tower bow side a) physics and b) smoothness based solutions to handle sparse and noisy pressure data at 120 fps.

Figures 4.4 (a) and (b) compare the ability of smoothness and physics based regularization techniques to estimate a clean pressure signal or distribution at the bow side sensor. The regularized solutions shown by 'Fused' and 'SFA' are smooth and continuous compared to the sparse, noisy experimental data. However, in this problem both regularization techniques display similar results. Physics based regularization techniques will have significant advantages over smoothness based techniques in regions

where experimental data is not measured or is known to be inaccurate, for example, due to the presence of inlet and exhaust valves. The current data-model fusion technique could be used to obtain fused solutions with different tolerances in different ranges of the yaw angle. For example, if the chosen numerical model cannot properly capture pressure distribution in regions with separated flow, it is possible to define which technique, experimental or computational, is more important in which regions of the yaw angle. This would yield a more accurate fused pressure distribution developed optimally from the available wind tunnel data and numerical solutions.

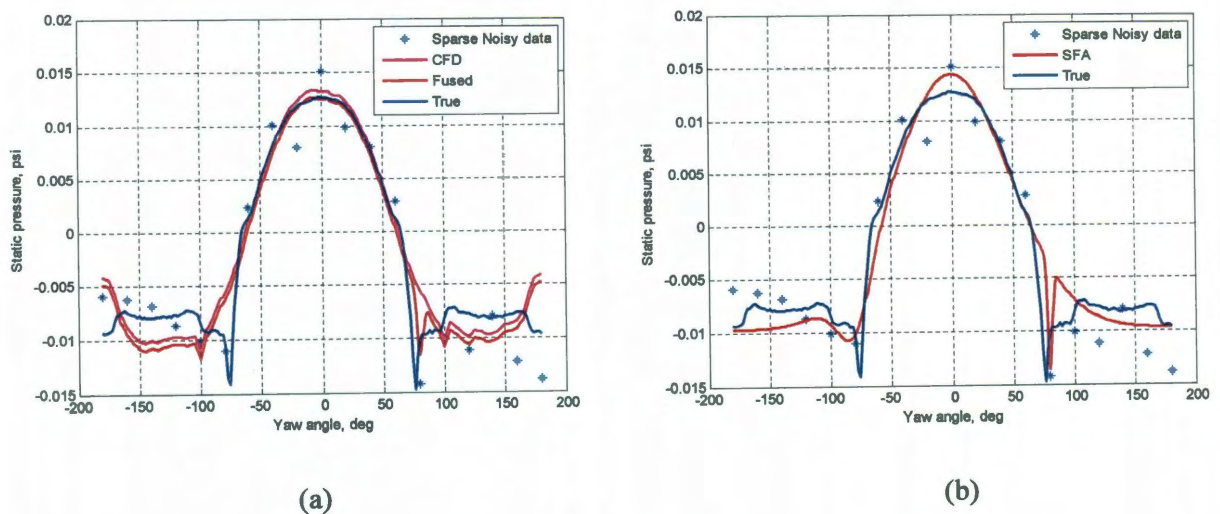
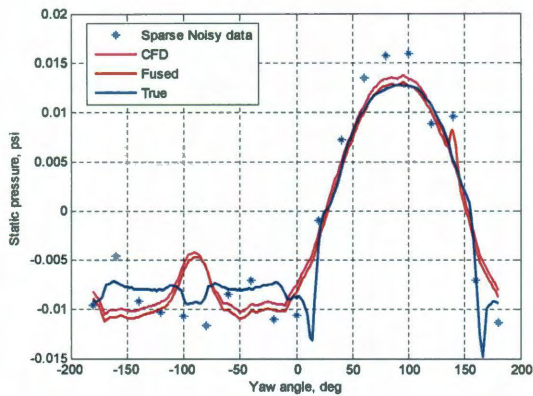


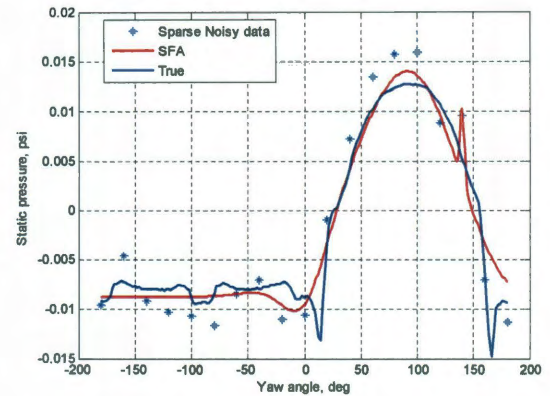
Fig. 4.5 Comparison of RALS tower bow side a) physics and b) smoothness based solutions to handle sparse and noisy pressure data at 40 fps.

Figure 4.6 compares the pressure distributions at port, stern and starboard side sensors obtained using the two regularization techniques at 40 fps. Again, a random noise of magnitude 0.005 psi was added to the pressure measurements. Physics based 'Fused' solutions look better than the smoothness based 'SFA' solutions for the stern and the port side pressure sensors. Another quality that makes this data-model fusion generic is that it

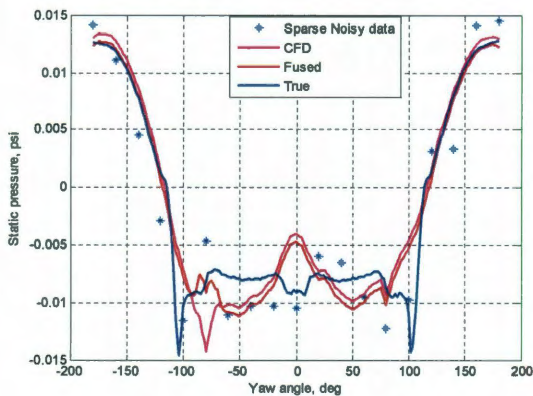
could be used for fusion of information/data from any two sources, two CFD codes, wind tunnel data from different experiments to name a few. A machine learning or neural network technique can be used to learn the differences between the two solutions. Appropriate tolerance criterion can be user-defined depending on the relative accuracy and importance of the two solutions and by adding the predicted differential hyperfunction to the less accurate solution.



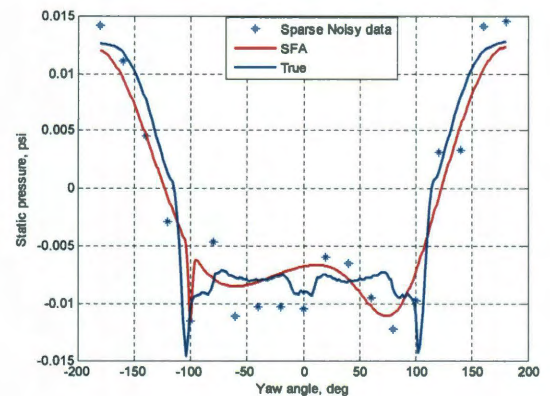
(a)



(b)



(c)



(d)

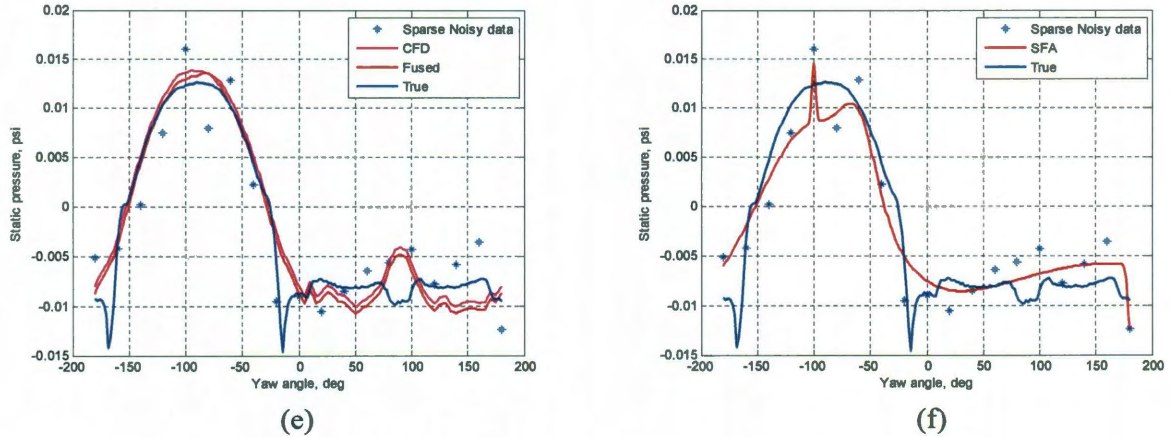


Figure 4.6 Comparison of RALS tower a) starboard side physics, b) starboard side smoothness, c) stern side physics and d) stern side smoothness e) port side physics and f) port side smoothness based solutions to handle sparse and noisy pressure data at 40 fps.

#### 4.2.3 Fusion in the pressure sensor position $\theta$

In external flow past bluff bodies, coefficient of pressure is a function of the yaw angle  $\beta$  and local pressure sensor position  $\theta$ . In the RALS tower problem wind tunnel data is available at only four values of  $\theta = [-90, 0, 90, 180]$  degrees shown in Fig. 4.7. A complete representation of  $C_p$  as a function of  $\beta$  and  $\theta$  is necessary because it can possibly inform the experimentalist what locations should be chosen for pressure sensor installation to improve wind direction prediction accuracy.



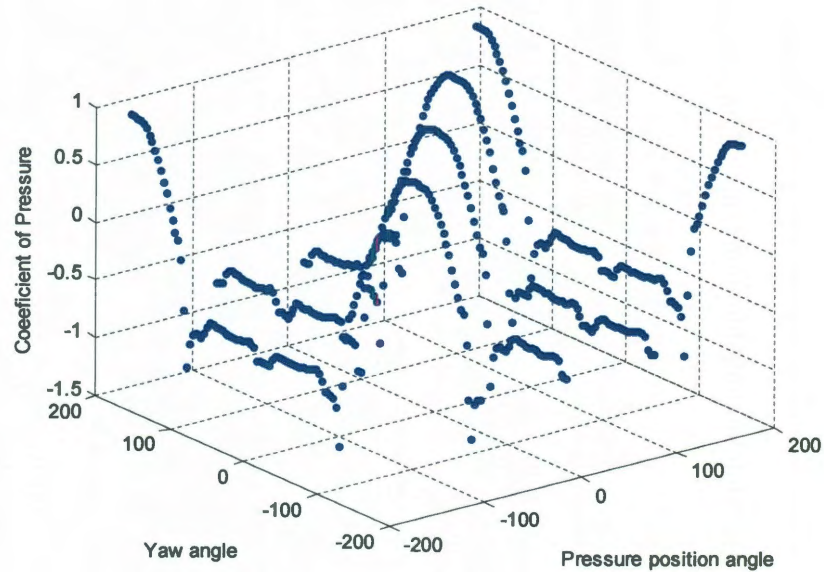


Figure 4.7 Wind tunnel coefficient of pressure as a function of yaw angle and pressure sensor position.

Numerical solutions, however, provide full coefficient of pressure distribution as a function of yaw angle and pressure sensor position. The data-model fusion technique discussed in Section 4.2.1 could similarly be applied to correct low fidelity  $C_p$  solutions even at values of  $\theta$  where no wind tunnel data is available. This could be done simply by calculating the differences between the wind tunnel and CFD solutions at  $\theta = [-90, 0, 90, 180]$  degrees. Once the differences have been calculated, it can be treated as a function approximation problem by SFA with  $\beta$  and  $\theta$  as inputs and  $C_p$  as output. Once a surrogate to the hyper-surface of differences has been created, it could be added to the CFD  $C_p$  surface shown in Fig. 4.8 to obtain a  $C_p$  distribution more complete than wind

tunnel  $C_p$  distribution and more accurate than CFD  $C_p$  distribution. The numerical solutions were extracted at 56 values of  $\theta$  for each yaw angle.

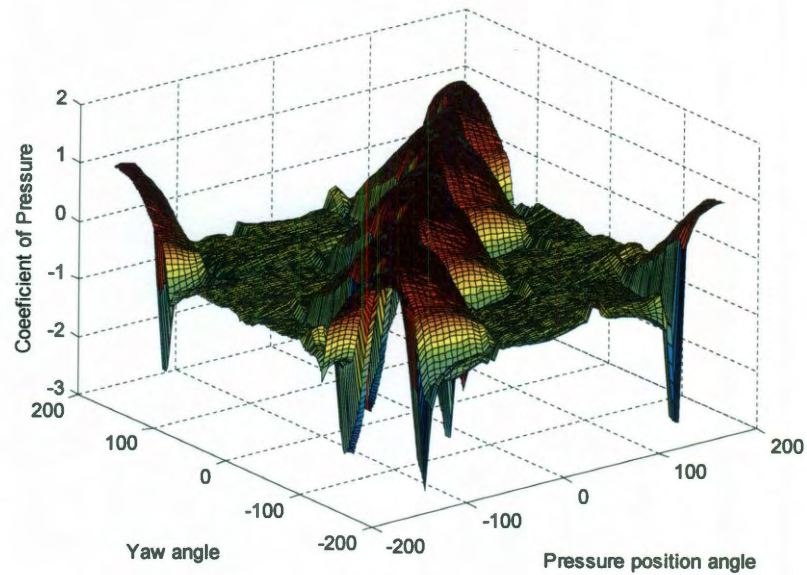
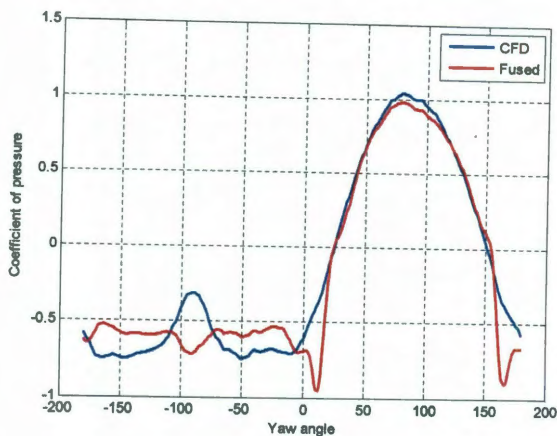
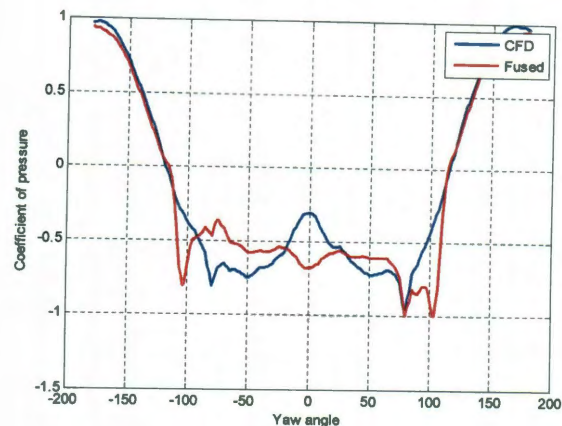


Figure 4.8 CFD coefficient of pressure as a function of yaw angle and pressure sensor position.

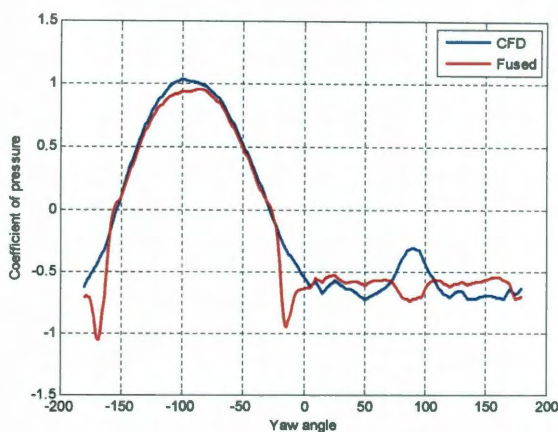
The fused  $C_p$  distribution was obtained in the manner as mentioned above and is shown in Figs. 4.9 and 4.10 for individual values of  $\theta$  to clearly visualize the validity of the fused distributions. Figures 4.9 (a), (b), (c) and (d) show the  $C_p$  distribution for sensors located just next to the starboard, stern, port and bow side respectively.



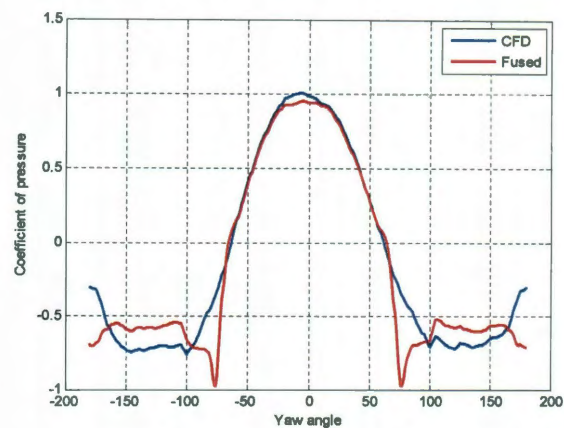
(a)



(b)



(c)



(d)

Figure 4.9 CFD and fused coefficient of pressure as a function of yaw angle and pressure sensor position at a location next to a) Starboard b) Stern c) Port and d) Bow side sensor. At these locations wind tunnel measurements were not taken.

### 4.3 Wind speed and direction prediction

In this section the freestream wind speed and direction prediction accuracies are presented. Wind speed and direction estimation techniques discussed in Chapter 3 were used to compute these results and, as discussed before, they are susceptible to noise and

sparsity in the training data. Section 4.2 discussed the smoothness and physics based regularization techniques that can generate a smooth pressure signal given sparse and noisy wind tunnel data. As presented before, a noisy and sparse pressure data was simulated with a noise magnitude of 0.005 psi and a yaw angle resolution of 20 degrees. This degraded data set was input to the smoothness and physics based regularization techniques to result in cleaner and smoother pressure signals which were input to the wind speed and direction estimation routines to predict wind speed and direction. The airdata estimation techniques were tested against the original clean wind tunnel data set shown in Chapter 3. Figure 4.10 shows the performance of the airdata estimation techniques when noise-free data at a yaw angle resolution of 4 degrees was taken as input. The error tolerance was  $\pm 3.4$  fps and  $\pm 2$  degrees for wind speed and direction respectively.

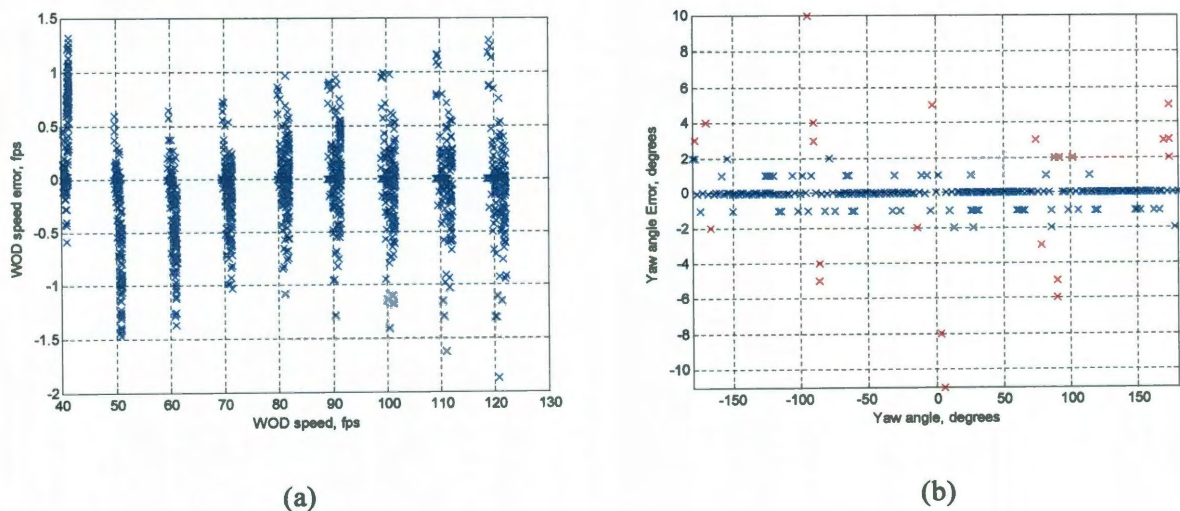


Figure 4.10 Prediction accuracies of a) wind speed and b) yaw angle for noise-free data at a yaw angle resolution of 4 degrees.

It can be seen from Fig. 4.10 (a) that wind speed errors are well below the tolerance level of 3.4 fps, however, wind direction errors exceed the desired tolerance level. This is because there are only 4 pressure sensors installed, one on each face of the RALS tower. Since the sensors are installed in the middle of each face, they are unable to capture any significantly different pressure signal in the range of  $-25^\circ \leq \beta \leq 25^\circ$ ,  $65^\circ \leq \beta \leq 115^\circ$ ,  $|\beta| > 165^\circ$ , and  $-115^\circ \leq \beta \leq -65^\circ$  which is where all the errors shown in Fig. 4.10 (b) occur. In these ranges of yaw angle, three pressure sensors face separated flow and the predicted  $C_p$  values also do not vary significantly to give any useful information to the direction estimation technique. Figure 4.11 shows the wind speed and direction prediction performance when smoothness based regularization techniques were used to obtain pressure signals from data with a noise magnitude of 0.005 psi and a yaw angle resolution of 20 deg.

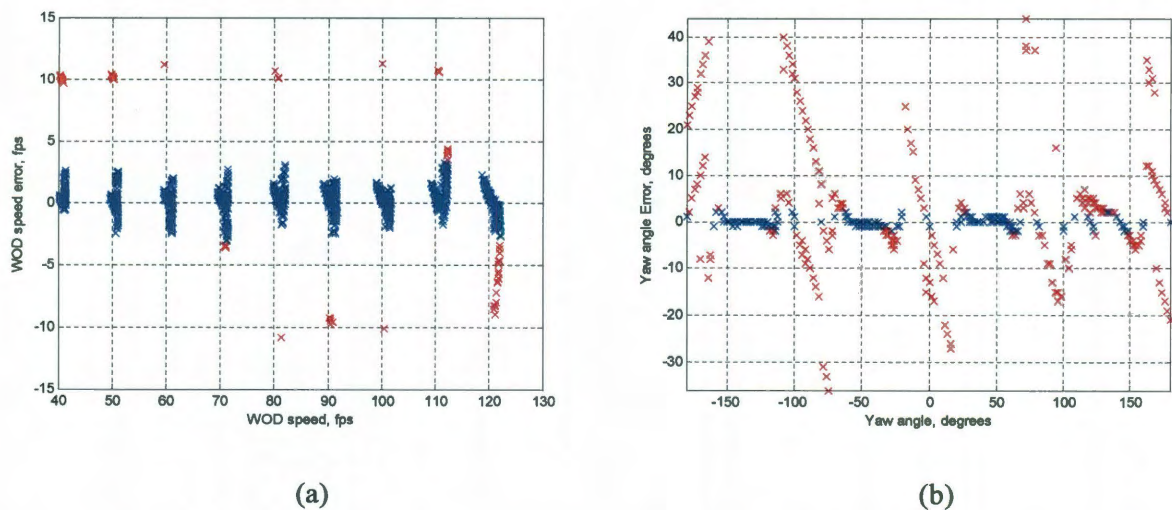


Figure 4.11 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using smoothness based regularization technique to obtain training data.

Figure 4.12 shows the wind speed and direction prediction accuracies for the same set of data when physics based regularization techniques were used to obtain cleaner pressure signals for the bow, starboard, stern and port side pressure sensors. Since the pressure signals obtained by the physics based regularization looked only slightly better than the signals obtained by the smoothness based regularization techniques, the estimated speed and direction accuracies also look slightly better. It should be emphasized here that the physics based regularization would have yielded better results if more accurate numerical solutions were used. Also, the importance of physics based regularization would become evident if the wind tunnel pressure data were incomplete either in  $\beta$  or  $\theta$  dimension.

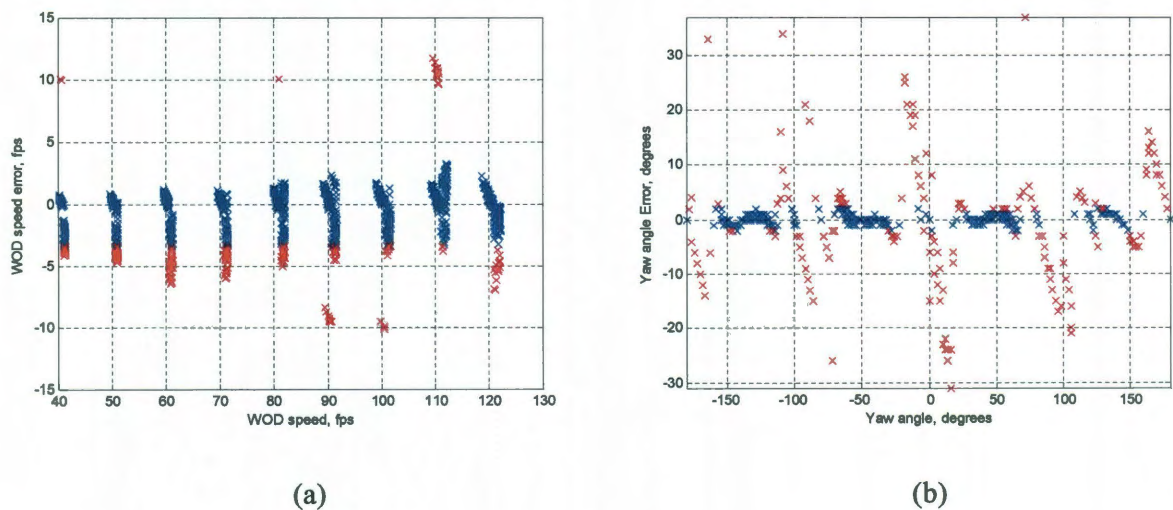


Figure 4.12 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using physics based regularization technique to obtain training data.

Figure 4.13 shows wind speed and direction prediction accuracies on the same set of data when fused pressure signals at all 56 pressure sensors used as training data. The training data generation procedure for this case was discussed in Section 4.2.3.

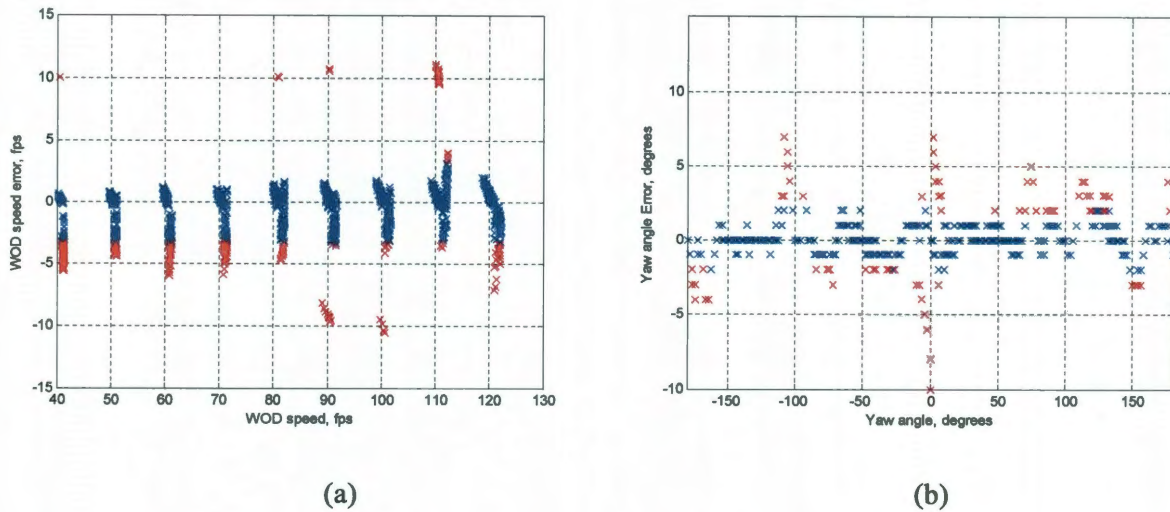


Figure 4.13 Prediction accuracies of a) wind speed and b) yaw angle for a noise magnitude of 0.005 psi at a yaw angle resolution of 4 degrees using physics based regularization technique to obtain training data. All 56 fused pressure signals were used to construct the training set.

It can be seen from Fig. 4.13 that both wind direction accuracies are significantly reduced as expected. In fact, the direction errors are less than when noise free data was used for training at a yaw angle resolution of 4 degrees as shown in Fig. 4.10 (b). However, wind speed prediction accuracies have not reduced significantly because estimation of wind speed does not depend strongly on the location of pressure sensors. As long as sufficient resolution is present in the dynamic pressure and yaw angle wind speed prediction will not change significantly.

## Chapter 5

### Sequential Experiment Design

In Chapters 2, 3 and 4, passive learning techniques were discussed for the FADS problem, where a learning algorithm like SFA, acts only after the training data set has been collected. The learning algorithm does not take part in the data collection procedure. Traditionally, popular space filling experiment design strategies like Latin Hypercube Sampling (LHS) are used to plan the data collection procedure. Use of LHS is popular with Monte Carlo techniques to estimate uncertainty in computer models. It has been proven that fewer samples are required with LHS compared to random sampling when estimating the output with a given variance [24]. Latin hypercube sampling is a simple and effective space filling sampling technique which does not need or assume the input-output functional relationship. Even though LHS is better than random sampling, a significant improvement is possible if the functional form of the input-output relationship is assumed and data are sampled from those regions where the output variance of the assumed input-output function is maximum. It is thought that advances in experiment design strategies will not only benefit the training of FADS but the fields of EFD and CFD in general.

Sub-optimal designs could be a serious problem if the input domain is excessively large or output determination is expensive. Careful selection of training points is also important to construct surrogates with low generalization error. Input sampling strategies where the learning algorithm actively takes part in the data collection procedure are



studied in the field of active learning or query learning as suggested by its name. In statistics they are studied in the field of Optimal Experiment Design (OED) [74].

For general nonlinear problems the input sampling strategies are sequential in nature. One starts with a few training points, construct a surrogate model using a learning algorithm, and then identify regions of the input domain where the surrogate model is most uncertain. Such a strategy is considered an exploitation based method where the expected mean squared error of the surrogate is minimized. Exploration based methods like adaptive sampling, on the other hand, try to improve input domain coverage by sampling from uncharted regions of the domain. Hybrid exploration-exploitation approaches are also popular [75], however, in this work only the exploitation based methods will be discussed. One shortcoming of an exploitation based sequential approach is that it requires the user to construct a new surrogate model each time a batch of input points are added to the training set. A learning algorithm with several control and kernel parameters that require grid search or cross-validation would require excessive user interaction and might be computationally prohibitive in an active learning scenario. Use of self-adaptive algorithms like SFA is favorable in such problems. Another significant limitation for nonlinear problems is that the designs are only as good as the surrogate models constructed by the learning algorithm. If the surrogate has significant biases then the resulting input designs could be far from optimal.

Section 5.1 will introduce OED approaches for general regression problems. Section 5.2 will present the development of a G-optimal design procedure with SFA and its

application to a simulated problem of approximating the *peaks* function in MATLAB. Finally, Section 5.3 will present the application of the G-optimal design procedure on the FADS problem. Active learning schemes were also developed for binary and multi-class classification tasks and they are discussed in Appendix B. Uncertainty Sampling [76] was developed with SFA and its application was shown on a simulated binary classification problem and a multi-class cavity flow classification problem.

## 5.1 Active Learning for regression problems

For continuous valued problems, significant research has been done in the field of statistics under the name of Optimal Experiment Design [74]. An acceptable active learning method for regression problems should be able to minimize mean squared error more than the passive learning methods using fewer training points. Concepts similar to Uncertainty Sampling cannot be used with regression problems because the magnitude of the output predicted by a learning algorithm now does not reveal any information regarding the uncertainty or significance of the particular sample point. Consider a general regression problem in Eq. (5.1):

$$y(x) = f(x, \eta) + \varepsilon \quad (5.1)$$

with  $d$  dimensional inputs  $x \in R^d$ , output  $y \in R$  and random errors  $\varepsilon$  which we assume to be independent and uncorrelated. Given a small, finite number of training samples  $s$ , the parameters of this regression problem can be estimated that minimize the mean squared error given by Eq. (5.2):

$$E_{MSE} = \int \|f(x, \hat{\eta}) - f(x)\|^2 dQ(x) \quad (5.2)$$

where  $Q$  is the environmental probability. However, since the environment is generally realized by the user only via the  $s$  number of training points, the following estimated mean squared error in Eq. (5.3) is minimized to estimate the parameters  $\eta$ :

$$S(\eta) = \min_{\hat{\eta}} \frac{1}{s} \sum_{i=1}^s (f(x_i, \hat{\eta}) - y(x_i))^2 \quad (5.3)$$

Given a finite number of training samples and a learning algorithm to estimate the parameters  $\hat{\eta}$ , an optimum active learning scheme would compute a sample point where the expected mean squared error over the whole input domain is maximum. Estimation of the future generalization error could be a very computationally demanding task for both pool based and population based active learning methods where information about the conditional probability distribution is either known or assumed. Paas and Kindermaann [77] use a Bayesian theoretic framework to develop an active learning method where they use Markov Chain Monte Carlo methods to approximate the expected loss. Roy et al. [78] estimate the expected error by adding all possible labels for each unlabeled samples to the training set. The proposed brute force approach is infeasible for regression problems and classification problems with many classes. The authors propose several ways to make this procedure efficient, for example, use of Naïve Bayes and SVMs where addition of a new training point does not need re-training, or estimate EMSE only on the neighborhood of the candidate points which compromises the generality of the approach.

Since estimation of expected future loss is infeasible, one can rely on the bias-variance decomposition of EMSE to estimate a sample where the expected error would be maximum. For a given sample  $x$ , let  $y = f(x, \eta)$ ,  $\hat{y} = f(x, \hat{\eta})$ , and  $\bar{y} = E[\hat{y}]$  be the true

output, estimated output and the expected value of the estimated output respectively. The expected value can be computed by averaging over different permutations of randomly chosen training sets of size  $s$ . The expected mean squared error can be decomposed into bias and variance as shown in Eq. (5.4):

$$\begin{aligned}
 E_{MSE} &= E\left[(\hat{y} - y)^2\right] \\
 &= E\left[\hat{y}^2\right] + y^2 - 2y\bar{y} \\
 &= E\left[\hat{y}^2\right] + y^2 - 2y\bar{y} + 2\bar{y}^2 - 2\bar{y}^2 \\
 &= (y - \bar{y})^2 + E\left[\hat{y}^2\right] + \bar{y}^2 - 2\bar{y}E[\hat{y}] \\
 &= \underbrace{(y - \bar{y})^2}_{\text{Bias}} + \underbrace{E\left[(\hat{y} - \bar{y})^2\right]}_{\text{Variance}}
 \end{aligned} \tag{5.4}$$

where the first term represents bias and the second term represents variance of the model. To elaborate, bias of a model can be understood as, for example, a quadratic function is being estimated by a linear model. The errors of the model are independent of the accuracy of the estimated parameters. The variance on the other hand is the error due to the errors between the estimated and the true parameters. For a simple regression problem shown below in Eq. (5.5)

$$y(x, \eta) = \eta_0 + \eta_1 x + \eta_2 x^2 + \dots + \eta_{l-1} x^{l-1}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ y_s \end{bmatrix} = \begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^{l-1} \\ 1 & x_2 & x_2^2 & \dots & x_2^{l-1} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & x_s & x_s^2 & \dots & x_s^{l-1} \end{bmatrix} \begin{bmatrix} \eta_0 \\ \eta_1 \\ \cdot \\ \cdot \\ \cdot \\ \eta_{l-1} \end{bmatrix} \tag{5.5}$$

$$\text{or } y = F\eta$$

Simple linear least squares regression dictates the estimated solution to be  $\hat{\eta} = (F^T F)^{-1} F^T y$ . The covariance matrix of the least squares estimator is given by  $\text{var}(\hat{\eta}) = \sigma^2 (F^T F)^{-1}$ . The  $l \times l$  matrix  $F^T F$  is called the Fisher Information Matrix (FIM) of  $\eta$ , the determinant of which is inversely proportional to the volume of the confidence ellipsoid of the parameters. The set of training points that maximize the determinant of FIM are called D-optimum. Given the variance of the parameter estimates, the standard variance of the predicted value of response at a given point  $x$  can also be calculated by Eq. (5.6):

$$\zeta(x) = \frac{\text{var}(\hat{y}(x))}{\sigma^2} = f^T(x) (F^T F)^{-1} f(x) \quad (5.6)$$

where  $f^T(x)$  is a row of  $F$  evaluated at  $x$ . The standardized variance given in the above equation gives a way to calculate the estimated variance of a linear model whose parameters are computed given a finite training set. This means that for a model with insignificant bias, an optimum sample can be computed that maximizes Eq. (5.6). At this sample the expected mean squared error will be maximum making it an optimum choice for a training point. The set of training points that minimize the maximum standardized variance are called G-optimum. The previous analysis was derived for linear regression problems, an extension to non-linear problems is possible using Taylor's expansion. Consider a non-linear model with  $l$  parameters  $y(x) = f(x, \vec{\eta})$  where  $\vec{\eta} = [\eta_1, \eta_2, \dots, \eta_l]$ . Taylor expansion of the model about  $\vec{\eta}^0 = [n_1^0, n_2^0, \dots, n_l^0]$  ignoring higher order derivatives would yield Eq. (5.7):

$$y(x) = f(x, \vec{\eta}^0) + \sum_{i=1}^l (\eta_i - \eta_i^0) \left. \frac{\partial f}{\partial \eta_i} \right|_{\eta_i = \eta_i^0} \quad (5.7)$$

Let  $z(x) = y(x) - f(x, \vec{\eta}^0)$ ,  $\gamma_i = \eta_i - \eta_i^0$ , and  $k_i = \left. \frac{\partial f}{\partial \eta_i} \right|_{\eta_i = \eta_i^0}$ , which gives us the following linearized form of Eq. (5.8):

$$z(x) = \sum_{i=1}^l \gamma_i g_i \quad (5.8)$$

The variance of this new linearized form can be computed in a similar manner as described above for a linear regression problem with  $l$  parameters. The standard variance of  $z(x)$  would be the same as  $y(x)$  and can be computed using Eq. (5.6) where

$$f^T(x) = \left[ \left. \frac{\partial f}{\partial \eta_1}, \frac{\partial f}{\partial \eta_2}, \dots, \frac{\partial f}{\partial \eta_l} \right] \right|_{\eta = \eta^0} \quad (5.9)$$

## 5.2 Implementation issues and application

There are several implementation issues that need to be carefully considered when developing a G-optimal design procedure using SFA with RBFs. In this section, those issues will be discussed in detail with examples from a 1-D simulated regression problem. Finally the steps of the design procedure will be laid down and its application will be presented on 2-D simulated regression problems. Let us consider a simple 1-D regression example where the target function is given by Eq. (5.10):

$$u(x) = 2 \exp[\log(0.2)(x+1)^2] + 1 \quad -3 \leq x \leq 3 \quad (5.10)$$

The objective here is to check whether the G-optimal design procedure with SFA is able to locate input samples that coincide or lie close to samples that bear the maximum squared error for the constructed surrogate. The initial training set was created by choosing data points according to Latin Hypercube Sampling method and SFA was used to fit a Gaussian RBF resulting in the following model and its parameter derivatives as shown in Eq. (5.11):

$$\begin{aligned}
 y(x) &= \sum_{i=1}^n c_i \phi(x, \psi_i, x_i^*) + b \text{ where } \psi_i = \log(\lambda_i), 0 < \lambda_i < 1, \text{ and } n = 1 \\
 \text{and } \phi(x, \psi_i, x_i^*) &= \exp(\psi_i (x - x_i^*)^2) = \phi_i \\
 \frac{\partial y}{\partial c_i} &= \phi_i \\
 \frac{\partial y}{\partial \psi_i} &= c_i \phi_i (x - x_i^*)^2 \\
 \frac{\partial y}{\partial x_i^*} &= 2c_i \phi_i \psi_i |x - x_i^*| \\
 \frac{\partial y}{\partial b} &= 1 \\
 f^T(x) &= \left[ \frac{\partial y}{\partial c_1}, \frac{\partial y}{\partial \psi_1}, \frac{\partial y}{\partial x_1^*}, \frac{\partial y}{\partial c_2}, \frac{\partial y}{\partial \psi_2}, \frac{\partial y}{\partial x_2^*}, \dots, \frac{\partial y}{\partial b} \right] \in \mathbb{R}^{3n+1}
 \end{aligned} \tag{5.11}$$

The following are the important implementation issues:

### 5.2.1 Singularity of Fisher Information Matrix

The singularity of Fisher Information Matrix (FIM) poses a significant problem in the implementation of a G-optimal design procedure especially with RBFs. The FIM can become singular due to several reasons. If  $f^T(x)$  represents a row vector of the derivatives of the surrogate model with respect to its parameters evaluated at point  $x$ . As

shown in Eq. (5.12), FIM is constructed by computing  $f(x_i) * f^T(x_i)$  which is a  $l \times l$  matrix and summing it over the available training points. It is important to realize that  $f(x_i) * f^T(x_i)$  is a matrix of rank 1, and that FIM gains rank as it is summed over increasing number of training points.

$$\text{FIM} = \sum_{i=1}^s f(x_i) * f^T(x_i) \quad (5.12)$$

This could serve as a guideline to select the minimum number of initial training points necessary to start the incremental G-optimal design procedure. The initial number of training points ( $s$ ) should be greater than or equal to the number of parameters ( $l$ ). For greedy algorithms like SFA, this could also decide the stopping criterion to add basis functions.

### 5.2.2 Overparameterization

Fukumizu [79] has shown that FIM can also become singular if any redundant basis functions are added to the approximation. FIM will become singular in the presence of singular basis functions even if it is constructed by summing over a large number of training points. This is also true for SFA in case a redundant basis function is added to the approximation when only noise is left in the residual error signal. A redundant RBF in a surrogate model constructed by SFA can easily be identified because its coefficient value will be very close to zero and the width of the RBF will also be very low, making the RBF appear as a spike. If this basis function is included in the surrogate model, the parameter derivatives and the FIM will bear this singularity resulting in spiking of the



standardized variance values at the redundant basis function center. In order to avoid this problem, if the coefficient or the width of a RBF is unusually low then it should be eliminated from the surrogate model and the addition of basis functions should be stopped in order to preserve the smoothness of the surrogate model and the standard variance.

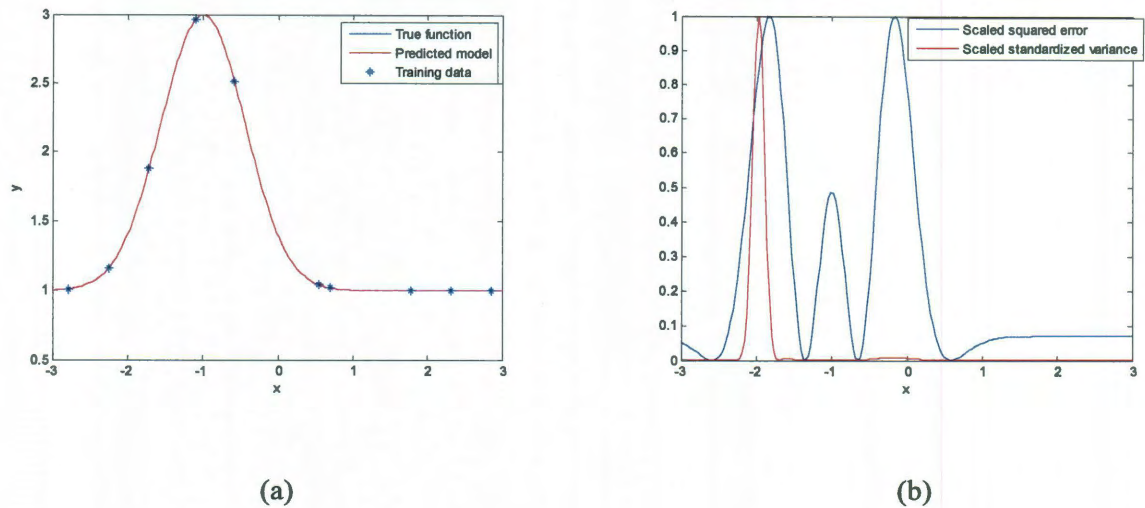


Figure 5.1 a) An overparameterized 1-D regression problem and b) Singularity in the standardized variance of an overparameterized surrogate model.

Figure 5.1 shows an example scenario where 2 Gaussian RBFs are chosen by SFA to model the problem given in Eq. (5.10) using ten training data points chosen by LHS. As seen in Fig. 5.1 (a) the predicted surrogate overlaps the true function. However, as previously discussed, the width parameter of the second basis function is of the order of  $1e-17$  and it is centered at  $x = -1.969$ . This singularity is evident in the standardized variance shown in Fig. 5.1 (b).

### 5.2.3 Initial number of training points and stopping criteria

The initial number of training points can be chosen at random as long as it is large enough to avoid construction of highly inaccurate surrogate models. As mentioned before, the number of training points ( $s$ ) should be larger than the number of parameters ( $l$ ). A simple heuristic such as  $s = 1.5l$  can be used to set the stopping criterion given the number of training points. However, this stopping criterion will continue adding an increasing number of basis functions as more training points are added, which will lead to overparameterization. This calls for an additional check that will stop the addition of basis functions if the coefficient, or the width of, the latest RBF falls below a user specified tolerance.

### 5.2.4 RBF center selection

As discussed in Chapter 2, the heuristic of placing RBFs at locations that correspond to the maximum absolute value of the residual works well if sufficient number of training data are present. This heuristic is unfavorable in the current optimal experiment design scenario for two reasons. First, at the beginning of the design procedure when there are only a few training points present, the RBF center selection heuristic might choose a center that is far from optimal. This could lead to a very inaccurate surrogate model that might result in inferior standard variance predictions. The second reason is that the FIM is constructed as a linear sum of  $s$  rank 1 matrices as shown in Eq. (5.12). If RBFs are placed at the available training points, then the contribution to the FIM due to each training point will be negligible because the derivatives of the surrogate model with respect to the RBF width and center will be zero. These two reasons

make it necessary to optimize the RBF centers minimizing the discrete inner product norm of the residual error. This does increase the computational burden, but is necessary for accurate prediction of the output variance.

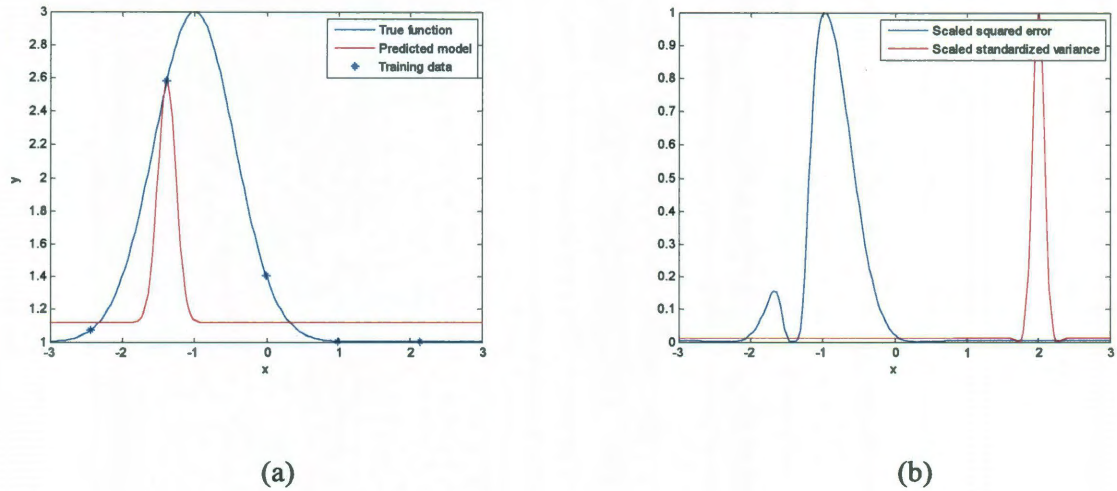
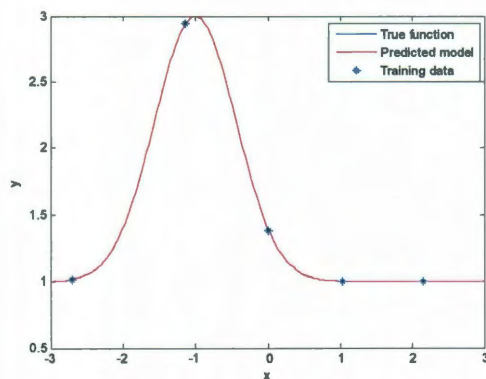


Figure 5.2 a) Surrogate model by SFA using RBF center selection heuristic and b) Erroneous standardized variance of the model output.

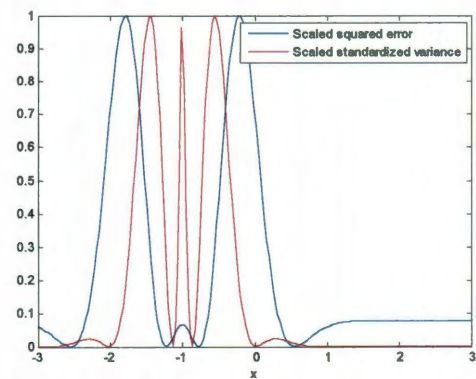
Figure 5.2 shows the surrogate model and its standardized variance when RBF centers are placed at locations corresponding to the maximum magnitude of the residual error. Figure 5.2 (a) shows how the use of the heuristic can lead to misleading surrogate models with high prediction errors. And Fig. 5.2 (b) shows the standardized variance of the surrogate model which is far from the squared residual error and might lead to choice of suboptimal training points.

### 5.2.5 FIM parameter selection

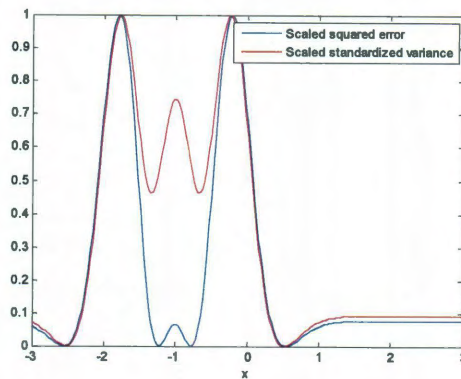
Another important implementation issue in the construction of the FIM is to decide whether to include derivatives of the surrogate model with respect to RBF center coordinates. A possible explanation for the inaccuracy in the prediction of standardized variance when RBF center derivatives are included is that it makes the FIM more singular. Figure 5.3 (a) shows a scenario where 5 points were used to construct a surrogate model by SFA, where RBF centers were optimized. Figures 5.3 (b) shows the standardized variance of the output when the RBF center parameters were included in the FIM construction, while Fig. 5.3 (c) shows the same when the center parameters were not included in the FIM construction. Standardized variance is more accurate when the center parameters are not included in FIM construction because of reduced singularity at the RBF center and its vicinity.



(a)



(b)



(c)

Figure 5.3 a) Surrogate model by SFA when RBF centers were optimized, and Standardized variance of the output when RBF center parameters were b) included and c) not included in the construction of the FIM.

### 5.2.6 The bias problem

The strategy of maximizing the standardized variance will result in an optimal design sample only if the model bias is insignificant. Previous work done by Kanamori and Sugiyama [80-82] have focused on using weighted maximum log-likelihood approaches to result in robust active learning schemes in case the interpolation model has been misspecified. Sugiyama [81] and Hering et al. [83] has also addressed the issue of using the conditional expectation of the generalization error instead of minimizing the expected loss in an asymptotic sense. However, these efforts are only possible for population based active learning methods where information about the probability distribution of the parameters is either available a priori or is assumed. In this dissertation, attention is focused only on pool based

active learning methods where no a priori information about the parameter probability distributions is available and sufficient initial data is not present to make good estimations about global parameter probability distribution functions. In this work, the strong approximation capabilities of universal approximators like Radial Basis Functions is relied on to attenuate the bias problem. A possible strategy is to start the active learning procedure with an over-parameterized model and remove any redundant basis functions before constructing the FIM. As number of training points increase the number of redundant basis functions should diminish. Another constraint on the number of basis functions is that the resulting number of parameters ( $2n+1$ ) should be less than the number of training points to prevent the FIM to being singular.

#### **5.2.7 Grid size of candidate sampling pool**

Once the FIM is constructed, the standard variance has to be maximized to determine the optimum sample point. However, it is computationally cheaper if a batch of optimum queries are generated at each iteration instead of just one. For this reason, a grid search for an optimum sample point will be suitable for low dimensional problems. However, even for low dimensional problems, one has to be careful in selecting the grid size for the search. This is because if a very fine grid size is selected then the chosen optimum points might lie next to each other compromising the diversity in the chosen batch of samples. A simple way to enforce diversity in the chosen batch of candidates is to enforce a minimum Euclidean distance between any two points in the chosen batch.

The final steps to implement the G-optimal design procedure with SFA are as follows:

1. Use LHS method to pick a small number of input points and evaluate the output on them to construct the initial training set.
2. Construct a surrogate model using SFA. The number of basis functions can be determined by using the heuristic  $s = 1.5l = 1.5 * (2n + 1)$ , where  $n$  is the number of basis functions.
3. Evaluate derivatives with respect to width and coefficient parameters of each basis function and the bias parameter and construct the FIM.
4. Do a line search to determine a sample input location that maximizes the standardized variance of the surrogate model output.
5. Add the chosen points to the training set and repeat the steps.

## 5.2.8 Examples

In this section, application of the G-optimal design procedure with SFA will be tested against passive LHS techniques to learn the peaks function which is constructed by scaling and transforming Gaussian RBFs and so SFA will have some bias in constructing a surrogate.

### 5.2.8.1 Peaks problem

In this section the active G-optimal design procedure with SFA would be compared against the passive Latin Hypercube Sampling (LHS) approach. The peaks function from Matlab is given by Eq. (5.13):

$$f(x, y) = 3(1-x)^2 \exp[-x^2 - (y+1)^2] - 10\left(\frac{x}{5} - x^3 - y^5\right) \exp[-x^2 - y^2] - \frac{1}{3} \exp[-(x+1)^2 - y^2]$$

$$-3 \leq x, y \leq 3 \quad (5.13)$$

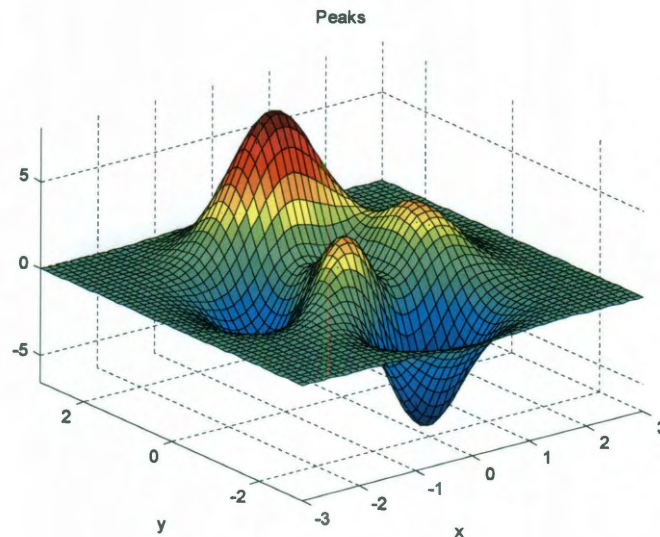


Figure 5.4 Peaks function

A set of 2500 uniformly spaced points were chosen as the test set for both the active and the passive learning methods. Fifty initial points were chosen by LHS via the Matlab's *lhsdesign* function for both active and passive learning methods. For passive learning *lhsdesign* was used to construct training sets in increments of 5 points upto a total of 200 points. To avoid any bias due to the training points, training and testing was repeated 10 times for both active and passive learning. The maximum number of basis functions was limited to 10 for both active and passive learning methods. Both the RBF center and width for each basis function was optimized according to the full objective function representing the discrete inner product norm of the residual error. The pattern search method from Matlab optimization toolbox was used to conduct this optimization because



of its simplicity and effectiveness. Addition of basis functions would also be stopped if the latest RBF coefficient magnitude exceeds 10 times the maximum absolute value of the current residual. This additional stopping criterion also prevented the addition of any redundant basis functions. The lower bound on the RBF width parameter was constrained to  $1e-4$ .

For active learning a pool of 400 uniformly spaced candidate points were initially chosen. Before selecting points from this candidate pool any points overlapping with the already chosen training points were removed. The FIM was constructed by taking derivatives of the surrogate model with respect to each RBF coefficient and width and the FIM was averaged over uniformly chosen points over the whole input domain. In each iteration, one point was chosen that corresponded to the maximum standardized variance. To put the performance of the active learning procedure in perspective, its performance was also compared to the scenario when it is assumed that the true model is known and five points are chosen in each iteration which correspond to the maximum squared error of the current surrogate model. Since the objective of the G-optimal design procedure is to sample points from locations that correspond to maximum variance, or lie in the vicinity of the maximum generalization error location, if little bias is present in the surrogate model.

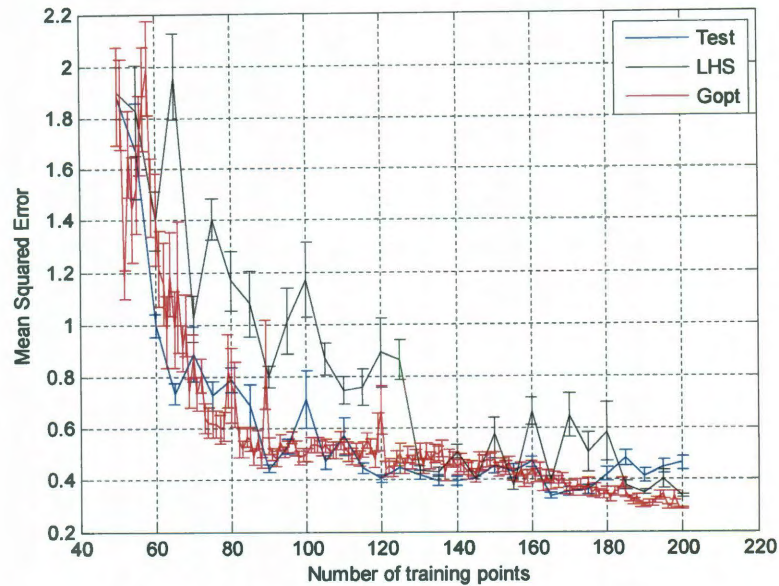


Figure 5.5 Reduction of mean squared error with increasing number of training points for Eq. (5.13). Bars show standard error computed over 10 repetitions. The black curve corresponds to passive LHS, the red curve corresponds to the G-optimal design and the blue curve represents the best possible result if the true function is known a priori. The blue curve is plotted to put the performance of variance only active learning scheme in perspective.

Figure 5.5 shows the superior performance of the G-optimal design procedure compared to the passive LHS technique on the peaks function which has some guaranteed bias when approximated by a linear sum of RBFs. Another way to evaluate the effectiveness of the G-optimal design procedure is to confirm that as new points are added to the training set the standardized variance of the model output decreases. Figure 5.6 justifies the validity of the design procedure.

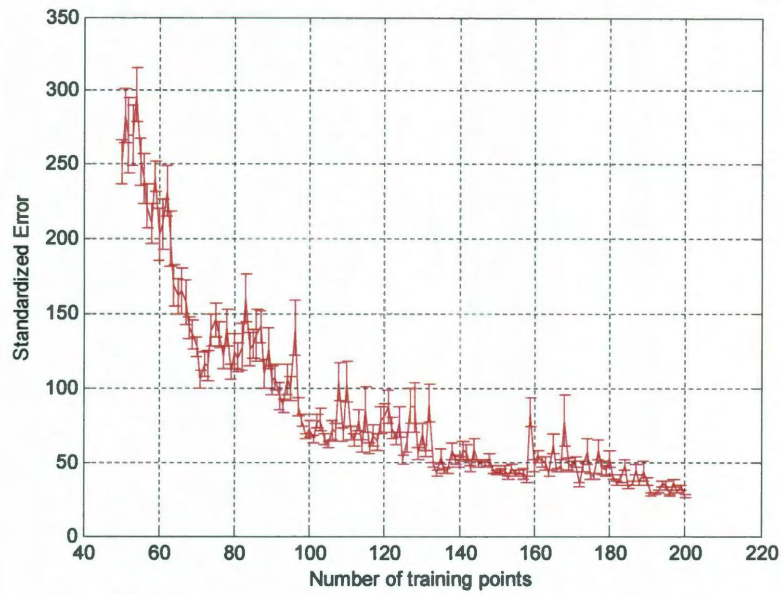


Fig. 5.6 Reduction of the maximum standardized variance with increasing number of training points. Bars show standard error computed over 10 repetitions.

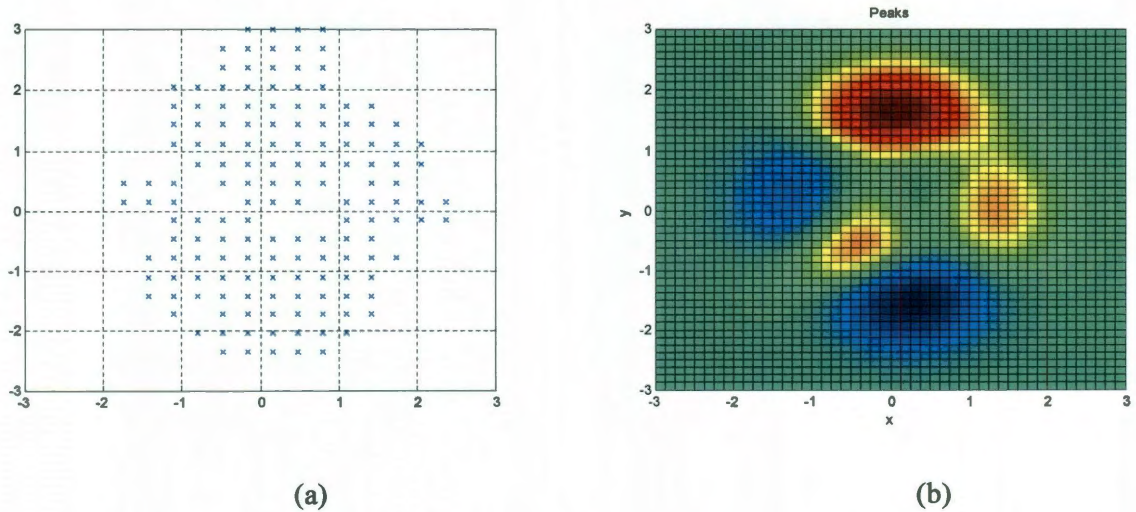


Fig. 5.7 a) Training points chosen by the G-optimal design procedure and b) two dimensional view of the *peaks* function.

Figure 5.7 (a) shows the positions of 150 training points chosen by the G-optimal design procedure with SFA based on 50 initial points chosen by LHS. This result shows the effectiveness of an input sampling scheme where the input-output functional relationship is used to sample data compared to a space filling experiment design approach. The improvement in design will be greater with an increase in number of input dimensions or a reduction in the problem domain. The experiment design technique was also tested on the peaks function when the input domain is expanded from  $[-3, 3]$  to  $[-10, 10]$ . There is no functional variation outside the  $[-3, 3]$  domain and so this problem tests if the algorithms can find the regions of functional activity. Figure 5.8 (a) shows the drastic performance improvement in the active learning scheme compared to the LHS technique. Figure 5.8 (b) shows the training points chosen by the active learning scheme.

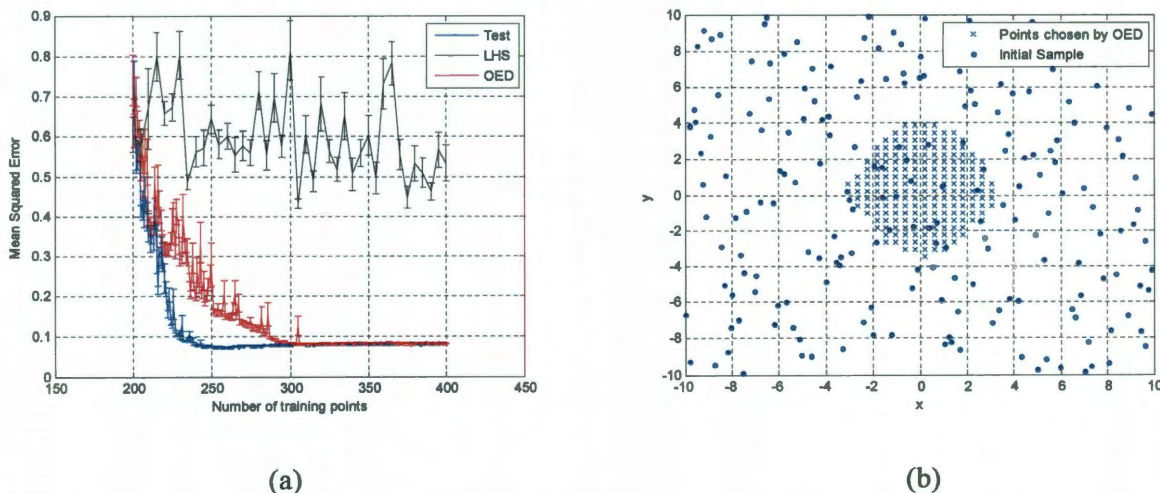


Fig. 5.8 a) Comparison of mean squared error, and b) Training points chosen by the G-optimal design procedure over the *peaks* problem with a larger domain.

### 5.3 Sequential Experiment Design for FADS

In this section, the G-optimal design procedure developed with SFA is tested on the FADS problem for the RALS tower. Wind tunnel tests for the FADS problem, measure static pressure on the surface at fixed sensor locations ( $\theta$ ). Test runs consist of sweeps where the Mach number or the freestream wind speed is fixed and the bluff body model is rotated resulting in pressure measurements at different incident yaw angles ( $\beta$ ). The time and effort of wind tunnel testing includes changing pressure sensor locations ( $\theta$ ) and freestream velocity ( $V_\infty$ ). However, CFD simulations give pressure distribution for all  $\theta$  for a given  $V_\infty$  and yaw angle  $\beta$ . Here the time consuming part is repeating the simulations for different  $\beta$ . The time and effort required would increase drastically if 3-D flow is considered and static pressure is measured as a function of  $V_\infty$ , yaw angle  $\beta$  and pitch  $\gamma$ . For 2-D flow, if  $\beta$  is measured at increments of 2 degrees, pressure measurements have to be made 180 times at each  $V_\infty$ . For 3-D flow, if  $\beta$  and  $\gamma$  are measured at increments of 2 degrees,  $180 \times 180$  pressure measurements have to be made at each  $V_\infty$ . An experiment design strategy for the hyper-surface  $C_p = f(\theta, \beta)$  could be helpful in reducing the time and cost of FADS. Optimal training data would also be able to give faster surrogates for  $C_p$  and would accelerate the forward problem approach discussed in Section 3.2. For demonstration purposes, it is assumed that one can freely change  $\theta$  and  $\beta$  to arbitrary continuous values within the interval  $[-180, 180]$ . However, in reality wind tunnel tests can only freely change  $\beta$  while CFD runs can only change  $\theta$  freely.

Figure 4.8 showed the hyper-surface of  $C_p$  as a function  $\theta$  and  $\beta$  which will be used to create the test data set for the design procedure. The G-optimal design procedure will be compared against the traditional LHS design procedure. An initial sample of 100 training points was chosen by LHS to start the design procedure. Surrogates were constructed by SFA using increasing number of LHS training points with increment of 5 points till a training set size of 500 was reached. For each training set size the training-testing procedure was repeated 10 times to eliminate any bias due to selection of training points and the mean and standard error of prediction accuracies were computed. The test set was constructed by the data-model fusion procedure. CFD solutions were constructed with a  $\beta$  increment of 1 degree and a  $\theta$  increment of approximately 6 degrees. The wind tunnel experiment used  $\beta$  increments of 4 degrees and four pressure sensors. The resulting test set had 20577 ( $= 361 \times 57$ ) points which captured the  $C_p$  variation in detail.

To implement the G-optimal design procedure, the stopping criteria discussed in Section 5.2 was used. This ensured that the FIM would not be singular. Also as the number of training points is increased the number of basis functions will also increase and address the bias problem. The lower bound on the RBF width parameter was constrained to  $1e-64$  and the center selection heuristic corresponding to the maximum absolute value of the residual was chosen. The basis centers were not optimized to save computational expense. One point was added to the training set at each iteration till 500 training points were collected. Again, to put the performance of the active learning procedure in perspective, its performance was also compared to the scenario when it is

assumed that the true model is known and one point is chosen at each iteration. This corresponds to the maximum squared error of the current surrogate model.

Figure 5.9 shows the reduction mean squared error in approximating  $C_p$  as training points are added. This seemed to indicate that the active learning procedure was not able to perform better than the LHS strategy. On closer inspection, it was determined that SFA could not properly construct surrogates of the coefficient of pressure. To remedy this problem, approximation of  $C_p^2$  was considered instead of  $C_p$  (Fig. 5.10).

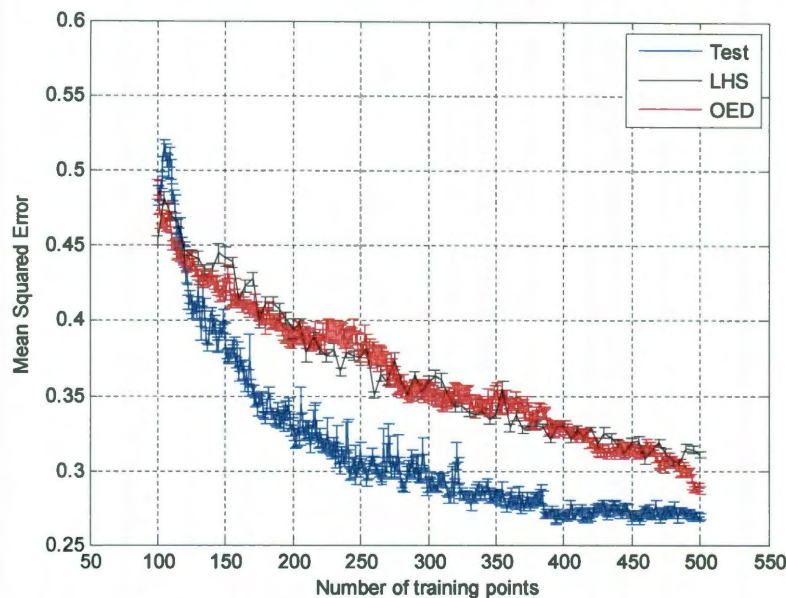


Figure 5.9 Reduction of  $C_p$  mean squared error with increasing number of training points. Bars show standard error.

It was also realized that additional information could also be incorporated in the training set for this problem.  $C_p^2$  bears peaks that correspond to those combinations of  $\theta$  and  $\beta$  that result in maximum flow separation. Due to the geometry of the RALS tower bluff

body, sensors on each wall measure the lowest pressures when the freestream wind is directed against the adjacent wall.

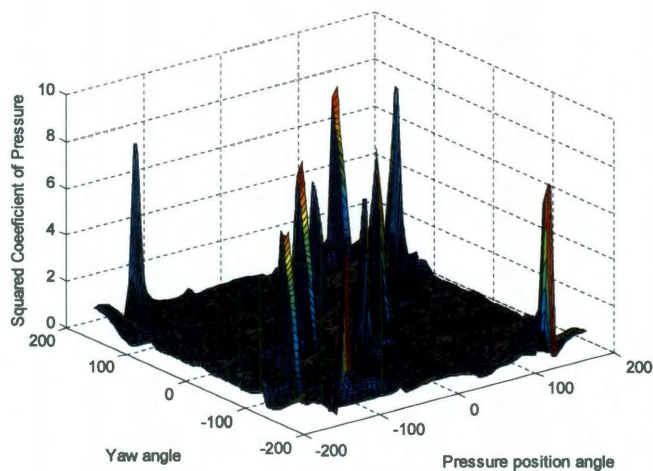


Figure 5.10  $C_p^2$  shown as a function pressure sensor position and yaw angle.

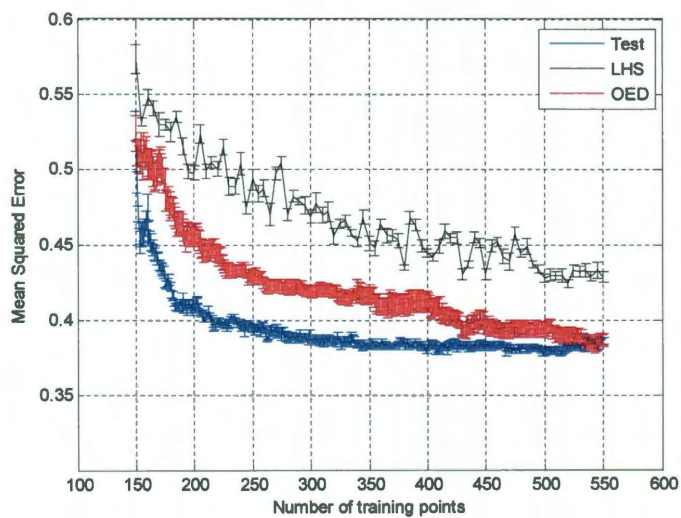


Figure 5.11 Reduction of  $C_p^2$  mean squared error with increasing number of training points. Bars show standard error.



This physical phenomena was used to enhance the information contained in the training set by adding points where  $\beta = [-180, -90, 0, 90, 180]$  degrees. Result for this case, shown in Fig. 5.11, display that active learning improves upon the LHS data collection method.

Figure 5.12 shows the  $C_p^2$  training points chosen by the G-optimal design procedure. The regions of the input domain corresponding to significant functional variation are efficiently covered by the design algorithm corroborating the claims of optimal experiment design.

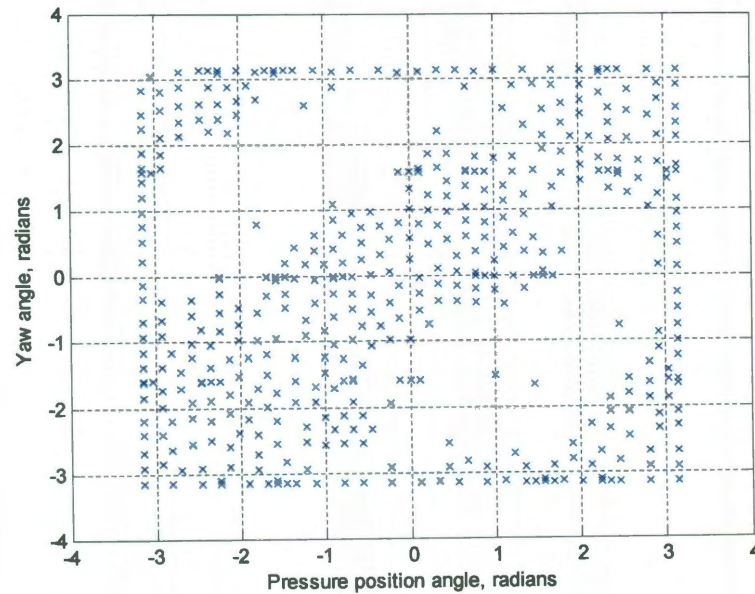


Figure 5.12 Training points chosen by the G-optimal design procedure.

## Chapter 6

### Conclusions and Future Work

This work was devoted to studying the compatibility of flush air data sensing systems with bluff bodies of arbitrary geometry. Non-trivial geometries and requirements for efficient real time usage resulted in a challenging inverse problem of determining air data parameters from surface pressure measurements that could be best solved by intelligent data driven algorithms. A data driven algorithm was developed based on the principles of scattered data approximation. It was proposed that mathematical models can be utilized as a surrogate that solves the inverse problem of determining wind speed and direction configuration from pressure values. This surrogate can be used in conjunction with an array of flush mounted pressure sensors to act as an air data measurement system. The proposed scattered data approximation algorithm, called Sequential Function Approximation (SFA), is a greedy self-adaptive function approximation tool that results in reliable and robust estimates without any user dependent control or kernel hyperparameter selection. In particular, a nonparametric and adaptive scattered data approximation tool accurately and efficiently mapped wind speed and yaw angle to pressure measurements taken from all four sides of the bluff body.

Another key contribution of this work was uncertainty reduction in FADS testing by fusing wind tunnel data with physical models. Wind speed and direction prediction is sensitive to noise, gaps and incompleteness in the wind tunnel data. It was shown that SFA could be used to fill the gaps and smooth out the noise in the wind tunnel data by

assimilating knowledge from low fidelity physical solutions. It was demonstrated that with data-model fusion wind speed and direction prediction accuracies improve in the presence of noise and gaps in the data. It could also be interpreted as improving low fidelity physical solutions with discrete wind tunnel data. Finally, it was shown that with data-model fusion a detailed representation of  $C_p$  could be obtained as a function of the yaw angle and flow incidence angle.

The final objective of this work was to develop a sequential experiment design strategy to accelerate through the test matrices of FADS. A general G-optimal design strategy was developed with SFA that sequentially adds data points to the training set that correspond to the maximum expected output variance. The proposed strategy was compared to traditional space filling design strategy of Latin Hypercube Sampling (LHS) and on both simulated regression and real world FADS problems. It was demonstrated that sequentially adding points achieved lower generalization error with fewer number of training points than the competing LHS method. The effectiveness of the proposed sequential design strategy was also demonstrated by comparing it to another active learning strategy when it is assumed that the true surface is known. In both the simulated and the FADS problems, the sequential experiment design strategy was successful in avoiding sampling from those regions of the input domain where the output did not have significant variation.

This work is also an example of how a function approximation tool like SFA can lie at the heart of statistical inference, uncertainty reduction and experiment design. The

developed mathematical techniques are general in nature and can very well be applied to complex mechanical and aerospace engineering systems.

### **6.1. Future Work**

Future avenues of research include accelerating SFA by developing heuristics to select the RBF center and width. The heuristics should decouple the center and width by exploring convergence rate of approximation by RBFs and thereby accelerate the computation of the parameters of each basis function. This would be particularly useful in high-dimensional problems. Another assignment for the future would be to test the data-model fusion algorithm in handling FADS when pitch is also introduced with yaw. Future work should also include the study of constrained optimum experiment design and its fidelity on the FADS problem.

## References

- [1] T. J. Rohloff, S. A. Whitmore, and I. Catton, "Fault-Tolerant Neural Network Algorithm for Flush Air Data Sensing," *Journal of Aircraft*, vol. 36, no. 3, pp. 541-549, 1999.
- [2] I. Samy, I. Postlethwaite, and D. Gu, "Subsonic Tests of a Flush Air Data Sensing System Applied to a Fixed-Wing Micro Air Vehicle," *Journal of Intelligent and Robotic Systems*, vol. 54, no. 1, pp. 275-295, 2008.
- [3] T. J. Rohloff, "Development And Evaluation Of Neural Network Flush Air Data Sensing Systems," Dept. of Mechanical Engineering, Univ. of California Los Angeles, 1998.
- [4] T. J. Larson, T. R. Moes, and P. M. Siemers III, *Wind-Tunnel Investigation of a Flush Airdata System at Mach Numbers From 0.7 to 1.4*. NASA Dryden Research Center, 1990.
- [5] S. A. Whitmore, R. J. Davis, and J. M. Fife, *In-Flight Demonstration of a Real-Time Flush Airdata Sensing (RT-FADS) System*. 1995.
- [6] G. Johnson, "Wind Energy Systems."
- [7] L. Milne-Thompson, *Theoretical Hydrodynamics*. The Macmillan Company, 1955.
- [8] J. Katz and A. Plotkin, *Low-Speed Aerodynamics*, 2nd ed. Cambridge University Press, 2001.
- [9] S. S. Desai, "Relative roles of computational fluid dynamics and wind tunnel testing in the development of aircraft.," *Current Science*, vol. 84, no. 1, pp. 49-64, 2003.
- [10] W. E. Faller and S. J. Schreck, "Neural networks: applications and opportunities in

- aeronautics,” *Progress in Aerospace Sciences*, vol. 32, no. 5, pp. 433–456, 1996.
- [11] C. de Boor, *A Practical Guide to Splines*. New York: Springer-Verlag, 1978.
- [12] J. Friedman and W. Stuetzle, “Projection Pursuit Regression,” *Journal of the American Statistical Association*, vol. 79, pp. 599-608, 1984.
- [13] M. Buhmann, *Radial Basis Functions*. New York: Cambridge University Press, 2003.
- [14] Y. Le Cun et al., “Handwritten Digit Recognition with a Back-propagation Network,” *Advances in Neural Network Information Processing Systems*, vol. 2, pp. 396-404, 1990.
- [15] W. E. Faller and S. J. Schreck, “Unsteady Fluid Mechanics Applications of Neural Networks,” *Journal of Aircraft*, vol. 34, no. 1, pp. 48-55, 1997.
- [16] F. Giralt, A. Arenas, J. Ferre-Giné, R. Rallo, and G. A. Kopp, “The simulation and interpretation of free turbulence with a cognitive neural system,” *Physics of Fluids*, vol. 12, no. 7, pp. 1826-1836, 2000.
- [17] S. D. Müller, M. Milano, and P. Koumoutsakos, *Application of machine learning algorithms to flow modeling and optimization*. Center of Turbulence Research, Stanford, 1999, pp. 169–178.
- [18] D. Babcock, C. Lee, B. Gupta, J. Kim, and R. Goodman, “Active Drag Reduction using Neural Networks,” *International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing (NICROSP '96)*, pp. 279-287, 1996.
- [19] C. Lee, J. Kim, D. Babcock, and R. Goodman, “Application of neural networks to turbulence control for drag reduction,” *Physics of Fluids*, vol. 9, pp. 1740-1748,

- 1997.
- [20] F. Sarghini, G. De Felice, and S. Santini, "Neural networks based subgrid scale modeling in large eddy simulations," *Computers & Fluids*, vol. 32, no. 1, pp. 97–108, 2003.
- [21] C. Wollblad and L. Davidson, "POD Based Reconstruction of Subgrid Stresses for Wall Bounded Flows Using Neural Networks," *Flow, Turbulence and Combustion*, vol. 81, no. 1, pp. 77-96, 2008.
- [22] M. M. Rai and N. K. Madavan, "Aerodynamic design using neural networks," *AIAA journal*, vol. 38, no. 1, pp. 173–188, 2000.
- [23] P. B. Nair, A. Choudhury, and A. J. Keane, "Some greedy learning algorithms for sparse regression and classification with mercer kernels," *The Journal of Machine Learning Research*, vol. 3, pp. 781-801, 2002.
- [24] M. D. McKay, "Latin hypercube sampling as a tool in uncertainty analysis of computer models," in *Proceedings of the 24th conference on Winter simulation*, pp. 557 - 564, 1992.
- [25] M. Hasenjaeger, "Active Data Selection in Supervised and Unsupervised Learning," zur Erlangung des akademischen Grades, 2000.
- [26] A. J. Meade and B. A. Zeldin, "Integrating Experimental Data and Mathematical Models in the Simulation of Physical Systems," *AIAA journal*, vol. 35, no. 11, pp. 1787-1790, 1997.
- [27] T. Poggio, F. Girosi, and C. MIT, "Networks for approximation and learning," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1481–1497, 1990.

- [28] B. Scholkopf and A. Smola, *Learning with Kernels*. Cambridge, Massachusetts,: The MIT Press, 2002.
- [29] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on signal processing*, vol. 41, no. 12, pp. 3397–3415, 1993.
- [30] P. Vincent and Y. Bengio, "Kernel Matching Pursuit," *Machine Learning*, vol. 48, pp. 165-187, 2002.
- [31] J. Suykens and J. Vandewalle, "Least Squares Support Vector Machine Classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293-300, Jun. 1999.
- [32] A. Srivastava and A. J. Meade, "A sparse greedy self-adaptive algorithm for classification of data," *Advances in Adaptive Data Analysis*, vol. 2, no. 1, pp. 97-114, 2010.
- [33] B. K. Natarajan, "Sparse Approximate Solutions to Linear Systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227-234, Apr. 1995.
- [34] S. Chen, C. Cowan, and P. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Transactions on Neural Networks*, vol. 2, no. 2, pp. 302-309, 1991.
- [35] C. Couvreur and Y. Bresler, "On the Optimality of the Backward Greedy Algorithm for the Subset Selection Problem," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 3, pp. 797-808, 2000.
- [36] S. Chen, "Basis Pursuit," Stanford University, 1995.
- [37] F. Girosi, "An equivalence between sparse approximation and support vector machines," *Neural Computation*, vol. 10, pp. 1455-1480, 1998.
- [38] Y. Freund, "Boosting a weak learning algorithm by majority," *Inf. Comput.*, vol.



- 121, no. 2, pp. 256-285, 1995.
- [39] S. E. Fahlman and C. Lebiere, "The cascade-correlation learning architecture," in *Advances in Neural Information Processing Systems*, vol. 2, pp. 524--532, 1990.
- [40] Z. Wang, C. Di Massimo, M. T. Tham, and A. Julian Morris, "A procedure for determining the topology of multilayer feedforward neural networks," *Neural Networks*, vol. 7, no. 2, pp. 291-300, 1994.
- [41] L. Yingwei, N. Sundararajan, and P. Saratchandran, "A sequential learning scheme for function approximation using minimal radial basis function neural networks," *Neural Computation*, vol. 9, no. 2, pp. 461--478, 1997.
- [42] L. K. Jones, "Constructive approximations for neural networks by sigmoidal functions," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1586--1589, 1990.
- [43] L. K. Jones, "A Simple Lemma on Greedy Approximation in Hilbert Space and Convergence Rates for Projection Pursuit Regression and Neural Network Training," *The Annals of Statistics*, vol. 20, no. 1, pp. 608-613, Mar. 1992.
- [44] A. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 930-945, 1993.
- [45] C. Fletcher, *Computational Galerkin Methods*. New York: Springer-Verlag, 1984.
- [46] A. J. Meade and B. A. Zeldin, "Approximation properties of local bases assembled from neural network transfer functions," *Mathematical and Computer Modelling*, vol. 28, no. 9, pp. 43-62, Nov. 1998.
- [47] K. Long and B. Storms, *Wind Tunnel Investigation of Wind Speed and Direction at Potential Anemometer Locations Aboard USS Zumwalt (DDG 1000) Class Ships*.

- Patuxent River, Maryland: Naval Air Warfare Center Aircraft Division, 2006.
- [48] K. Long, "Wind Tunnel Surface Pressure Measurements on a 1/72 Scale Model of the Runway Arrested Landing Site (RALS) Control Tower," Naval Air Warfare Center Aircraft Division, Patuxent River, Maryland, 2008.
- [49] D. Lowe and C. Zapart, "Point-wise confidence interval estimation by neural networks: A comparative study based on automotive engine calibration," *Neural Computing & Applications*, vol. 8, no. 1, pp. 77–85, 1999.
- [50] J. Hull, D. Ward, and R. R. Zakrzewski, "Verification and validation of neural networks for safety-critical applications," in *Proceedings of the American Control Conference*, vol. 6, pp. 4789–4794, 2002.
- [51] P. J. Roebber,, M. R. Butt, S. J. Reinke, and T. J. Grafenauer, "Real-time forecasting of snowfall using a neural network," *Weather and Forecasting*, vol. 22, pp. 676-684, 2007.
- [52] P. J. Shayler, M. Goodman, and T. Ma, "The exploitation of neural networks in automotive engine management systems," *Engineering Applications of Artificial Intelligence*, vol. 13, no. 2, pp. 147–157, 2000.
- [53] S. Watanabe, "A System Concept for EFD/CFD Integration: Digital/Analog Hybrid Wind Tunnel," Jan-2010.
- [54] K. Nisugi, T. Hayase, and A. Shirai, "Fundamental Study of Hybrid Wind Tunnel Integrating Numerical Simulation and Experiment in Analysis of Flow Field," *JSME International Journal Series B Fluids and Thermal Engineering*, vol. 47, no. 3, pp. 593-604, 2004.
- [55] M. Stojanowski and E. Germain, "The FALCON 7X: from ETW to flight," in *46th*

*AIAA Aerospace Sciences Meeting and Exhibit*, 2008.

- [56] G. Rufolo, P. Roncioni, M. Marini, R. Votta, and R. Palazzo, “Experimental and Numerical Aerodynamic Data Integration and Aerodatabase Development for the PRORA-USV-FTB\_1 Reusable Vehicle,” presented at the 14th AIAA/AHI Space Planes and Hypersonic Systems and Technologies Conference, Canberra, Australia, 2006.
- [57] K. Pettersson, “Scaling Techniques Using CFD and Wind Tunnel Measurements for use in Aircraft Design,” Department of Aeronautical and Vehicle Engineering, Division of Aerodynamics, 2006.
- [58] K. Pettersson and A. Rizzi, “Aerodynamic scaling to free flight conditions: Past and present,” *Progress in Aerospace Sciences*, vol. 44, no. 4, pp. 295-313, May. 2008.
- [59] P. Planquart, “Integration of CFD and Experimental Results at VKI in Low-Speed Aerodynamic Design,” presented at the 3rd International Symposium on Integrating CFD and Experiments in Aerodynamics, Colorado, USA, 2007.
- [60] J. Jouhaud, P. Sagaut, and B. Labeyrie, “A Kriging Approach for CFD/Wind-Tunnel Data Comparison,” *Journal of Fluids Engineering*, vol. 128, no. 4, pp. 847-855, Jul. 2006.
- [61] R. Unger,, M. Hutchinson,, M. Rais-Rohani, R. Haftka,, and B. Grossman,, “Variable-Complexity Design of a Transport Wing,” *Intl. J. Systems Automation: Res. and Appl. (SARA)*, vol. 2, pp. 87-113, 1992.
- [62] A. J. Keane and P. B. Nair, *Computational Approaches for Aerospace Design: The*

*Pursuit of Excellence*. Wiley, 2005.

- [63] R. Haftka,, “Combining Global and Local Approximations,” *AIAA journal*, vol. 21, no. 9, pp. 1523-1525, 1991.
- [64] C. Tang, K. Gee, and S. Lawrence, “Generation of Aerodynamic Data using a Design of Experiment and Data Fusion Approach,” presented at the 43rd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, 2005.
- [65] M. Eldred, A. Giunta, and S. Collis, “Second-Order Corrections for Surrogate-Based Optimization with Model Hierarchies,” *American Institute of Aeronautics and Astronautics*, no. 4457, 2004.
- [66] J. Navarrete and A. J. Meade, “Fusion of experimental data and mathematical models in the simulation of aerodynamic coefficients,” presented at the 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, 2004.
- [67] M. C. Kennedy and A. O'Hagan, “Predicting the Output from a Complex Computer Code When Fast Approximations Are Available,” *Biometrika*, vol. 87, no. 1, pp. 1-13, Mar. 2000.
- [68] A. J. Keane, “Wing optimization using design of experiment, response surface, and data fusion methods.,” *Journal of Aircraft*, vol. 40, no. 4, pp. 741-750, 2003.
- [69] A. Forrester, N. Bressloff, and A. J. Keane, “Optimization using surrogate models and partially converged computational fluid dynamics simulations,” presented at the Proceedings of the Royal Socceity, pp. 2177-2204, 2006.
- [70] J. Lewis, S. Lakshmivarahan, and S. Dhall, *Dynamic Data Assimilation: A Least Squares Approach*. 2006.
- [71] J. Kaipio and E. Somersalo, *Statistical and Computational Inverse Problems*, vol.

160. Springer, 2005.
- [72] M. Ulbrich, *A Generalized Tikhonov Regularization for Nonlinear Inverse Ill-Posed Problems*. Munchen, Germany: , 1998.
- [73] W. Wang, “Exploration of Tikhonov regularization for the fusion of experimental data and computational fluid dynamics,” M.S. thesis, Rice University, 1999.
- [74] A. Atkinson, A. Donev, and R. Tobias, *Optimum Experimental Designs, with SAS*. Oxford University Press, 2007.
- [75] K. Crombecq, D. Gorissen, L. De Tommasi, and T. Dhaene, “A novel sequential design strategy for global surrogate modeling,” in *Proceedings of the 41st Conference on Winter Simulation*, pp. 731-742, 2009.
- [76] D. Lewis and W. Gale, “A sequential algorithm for training text classifiers,” in *SIGIR '94: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 3-12, 1994.
- [77] J. Kindermann, G. Paass, and F. Weber, “Query construction for neural networks using the bootstrap,” in *In Proc. Int. Conf. Artificial Neural Networks*, vol. 95, pp. 135-140, 1995.
- [78] N. Roy and A. Mccallum, “Toward Optimal Active Learning through Sampling Estimation of Error Reduction,” in *18th International Conf. on Machine Learning*, pp. 441-448, 2001.
- [79] K. Fukumizu, “A Regularity Condition of the Information Matrix of a Multilayer Perceptron Network.,” *Neural Networks*, vol. 9, no. 5, pp. 871–879, 1996.
- [80] T. Kanamori and H. Shimodaira, “Active learning algorithm using the maximum weighted log-likelihood estimator,” *Journal of Statistical Planning and Inference*,

- vol. 116, no. 1, pp. 149-162, Sep. 2003.
- [81] M. Sugiyama, "Active learning in approximately linear regression based on conditional expectation of generalization error," *The Journal of Machine Learning Research*, vol. 7, pp. 141-166, 2006.
- [82] M. Sugiyama and S. Nakajima, "Pool-based active learning in approximate linear regression," *Machine learning*, vol. 75, no. 3, pp. 249–274, 2009.
- [83] P. Hering and M. Šimandl, "Sequential optimal experiment design for neural networks using multiple linearization," *Neurocomputing*, vol. 73, pp. 3284-3290, 2010.
- [84] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines," in *In Proc. 17th International Conf. on Machine Learning (2000)*, pp. 839-846., pp. 839–846, 2000.
- [85] S. Zomer, M. Sanchez, R. G. Brereton, and J. Pavon, "Active learning support vector machines for optimal sample selection in classification," *Journal of Chemometrics*, vol. 18, no. 6, pp. 294-305, Jun. 2004.
- [86] T. Luo, K. Kramer, B. Dmitry, and L. Hall, "Active learning to recognize multiple types of plankton," *Journal of Machine Learning Research*, vol. 6, pp. 589-613, 2005.
- [87] B. Cui, H. Lin, and Z. Yang, "Uncertainty sampling-based active learning for protein-protein interaction extraction from biomedical literature," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10344-10350, Sep. 2009.
- [88] M. Saar-Tsechansky and F. Provost, "Active Sampling for Class Probability Estimation and Ranking," *Machine Learning*, vol. 54, no. 2, pp. 153-178, Feb.

2004.

- [89] T. Waterhouse, "Optimal Experiment Design for Nonlinear and Generalized Linear Models," Ph.D. Thesis, University of Queensland, 2005.
- [90] B. Settles, "Active Learning Literature Survey," *Science*, vol. 10, no. 3, pp. 237–304, 1995.
- [91] H. S. Seung, M. Opper, and H. Sompolinsky, "Query by committee," in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 287-294, 1992.
- [92] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the Query by Committee algorithm," *Machine Learning*, vol. 28, pp. 133-168, 1997.
- [93] C. Körner and S. Wrobel, "Multi-class Ensemble-Based Active Learning," in *Machine Learning: ECML 2006*, 2006, pp. 687-694.
- [94] J. Platt, "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," in *Advances in Large Margin Classifiers*, pp. 61-74, 1999.
- [95] K. Duan, S. Keerthi, W. Chu, S. Shevade, and A. Poo, "Multi-category classification by soft-max combination of binary classifiers," In *4th International Workshop on Multiple Classifier Systems*, pp. 125-134, 2003.
- [96] M. B. Tracy, *Cavity unsteady-pressure measurements at subsonic and transonic speeds [microform] / Maureen B. Tracy and E.B. Plentovich*. Hampton, Va. : Springfield, VA :: National Aeronautics and Space Administration, Langley Research Center, 1997.
- [97] "F-22." [Online]. Available: <http://futureweaponsandmore.blogspot.com/>.

[Accessed: 04-Jan-2010].

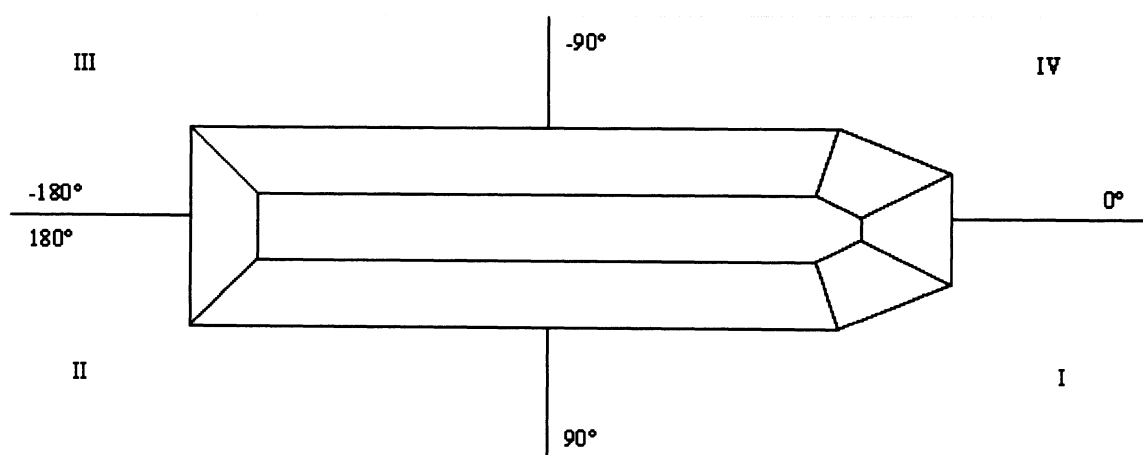
- [98] “F-35,” 04-Jan-2010. [Online]. Available:  
<http://wararchive.blogspot.com/2009/11/lockheed-f-35-lightning-ii-specs-and.html>.
- [99] A. Srivastava, J. Kugler, and A. J. Meade, “An Adaptive Algorithm for Classifying Weapons Bay Cavity Flow Types,” *Journal of aerospace computing, information, and communication*, Submitted. .



## Appendix A

### A.1 Inverse approach to predict yaw angle

The non-uniqueness problem motivated the use of an array of networks instead of just one to predict wind direction. Dividing the training pressure data into ranges of  $\beta$  would improve the learning ability of an individual network, however, assigning a test point to the correct network becomes more difficult with increasing number of networks. A logical compromise is to use an array of networks for the  $[0^\circ, 90^\circ]$ ,  $[90^\circ, 180^\circ]$ ,  $[0^\circ, -90^\circ]$  and  $[-90^\circ, 180^\circ]$  ranges of  $\beta$  as shown in Fig A.1. Again, the idea is to construct four separate training networks and devise a way to identify which quadrant a test point belongs to before predicting the actual value. In other words, the strategy is to first determine which quadrant the wind is approaching the body and then the corresponding set of training points is selected to predict the actual freestream wind direction.



(a)

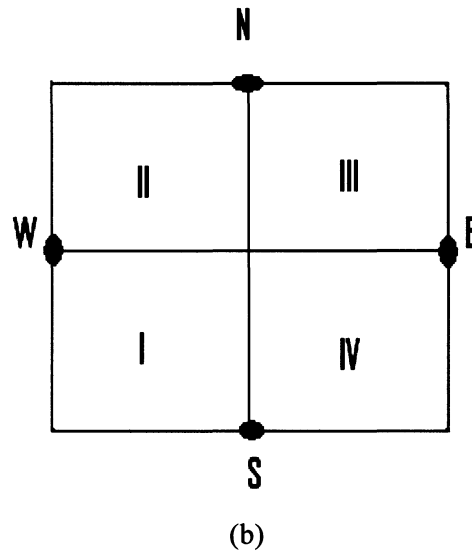


Figure A.1 Quadrants depicting different ranges of the yaw angle for (a) Surface vessel and (b) RALS tower.

This realization helped in formulating the following training and testing strategy for the surface vessel problem:

1. Select training and test points and train using SFA to predict wind speed with a low tolerance.
2. Calculate training and test pressure coefficient data.
3. Divide the available training data into four sets according to Fig. A.1 and train four separate network pairs,

$$V_{\infty}^a = \sum_{i=1}^n c1_i(\phi(P_1, \dots, P_d) + b1_i) \quad \text{and} \quad \beta^a = \sum_{i=1}^n c2_i(\phi(C_{p1}, \dots, C_{pd}) + b2_i)$$

4. For a test data point  $[P_1, \dots, P_d]$ , calculate  $V_{\infty}^a$  and the mean pressure coefficient ( $C_{p,mean}$ ) for each quadrant, as per Table A.1, and pick the quadrant with the maximum mean pressure coefficient. Calculate the approximate wind direction  $\beta^a$  for the quadrant.

5. Repeat this for other test sets.

| Quadrant | Pressure tap number |
|----------|---------------------|
| I        | 1, 22-29, 45-50     |
| II       | 15-22, 40-45        |
| III      | 10-15, 36-40        |
| IV       | 1-9, 29-35          |

Table A.1 Distribution of pressure taps for the surface vessel problem

The above strategy requires a sufficient number of pressure sensors to be present in each quadrant. Because the RALS tower problem has only two sensors per quadrant the shortage of pressure sensors posed a limitation on the ability of the wind direction estimation technique to place the test point in the right quadrant. This would especially be a problem at yaw angles in the vicinity of 0, 90, 180 and -90 degrees. To elaborate, Fig. A.2 shows the variation of pressure coefficient with yaw angle about the RALS tower in those regions where identifying the correct training network of a test point is difficult. For example, all points in the range  $20^\circ < \beta < 70^\circ$ , the south port and the west port bears positive pressure values while the other two bear negative values. However, for points in the range  $-20^\circ < \beta < 20^\circ$  only the south port bears positive values and in the range  $70^\circ < \beta < 110^\circ$  only the west port bears positive pressure values. Therefore, the correct training network for the points in the range  $-20^\circ < \beta < 20^\circ$  cannot be decided just by looking at the sign of the pressure values of the west and the east port. However, looking

at the magnitudes of the pressure values can identify the correct region to which the test point belongs. One would assume that for points in the range  $0^\circ < \beta < 20^\circ$  the west pressure port value would be greater than the east pressure port value as is true for points where  $\beta > 20^\circ$ . But Fig. A.2 shows that for southward winds the west pressure port value is actually less than or equal to the east pressure port value for points in the range  $0^\circ < \beta < 20^\circ$ .

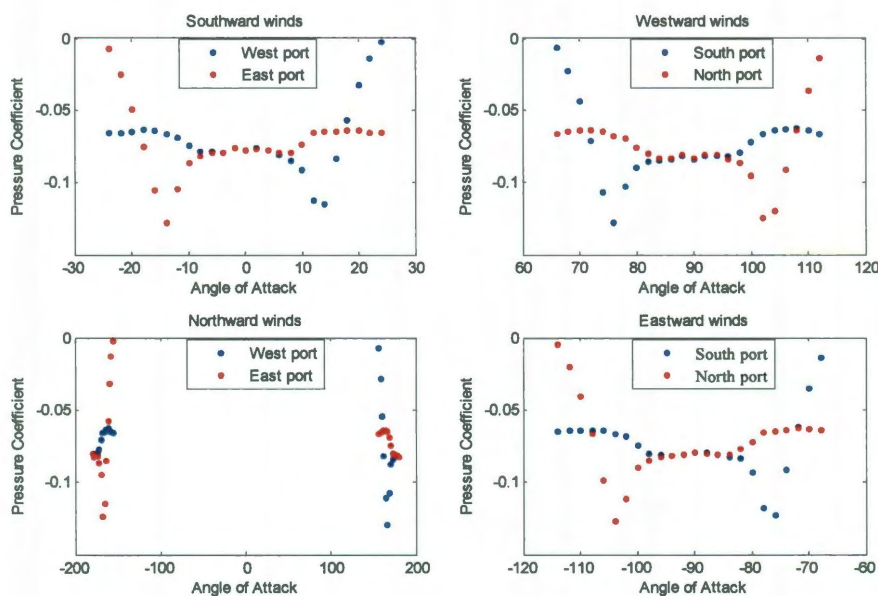


Figure A.2 Pressure coefficient variation vs. the yaw angle for selected ranges of angle of attack.

This transition takes place when both the west and the east pressure coefficient values are approximately equal to -0.055. On a closer look, one can notice that the same trend is true for all the four networks. For example, for the westward winds, when  $70^\circ < \beta < 90^\circ$  the north port pressure values are greater than the south port and the transition occurs when the adjacent north and south pressure coefficient values are approximately equal to -

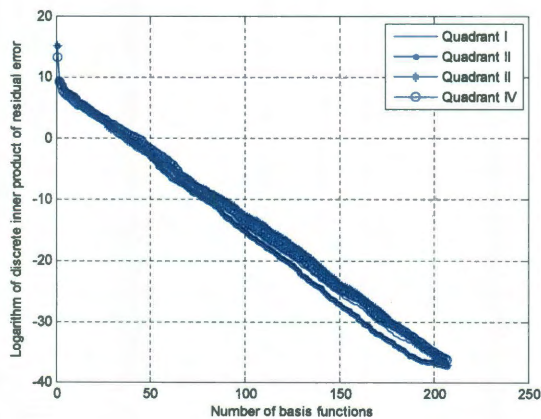
0.055. This heuristic helped us finalize the following training and testing strategy for the RALS tower problem.

1. Select training and test points and train using SFA to predict wind speed.
2. Calculate training and test pressure coefficient data.
3. Divide the available training data into 4 sets according to Fig. A.1 (b) and train 4 networks  $T_{SW}$ ,  $T_{NW}$ ,  $T_{NE}$  and  $T_{SE}$ .
4. For a test point  $X_{test} = [cp_S, cp_W, cp_N, cp_E]$  pick  $T_{SW}$  if  $cp_S > 0$  and  $cp_W > 0$ . Similarly pick  $T_{NW}$  if  $cp_N > 0$  and  $cp_W > 0$  and so on.
5. If  $cp_S > 0$  and  $cp_N, cp_W, cp_E < 0$ 
  - if  $\max(cp_W, cp_E) > -0.055$ 
    - if  $cp_W > cp_E$  then pick  $T_{SW}$ .
    - if  $cp_W \leq cp_E$  then pick  $T_{NW}$ .
  - if  $\max(cp_W, cp_E) \leq -0.055$ 
    - if  $cp_W > cp_E$  then pick  $T_{NW}$ .
    - if  $cp_W \leq cp_E$  then pick  $T_{SW}$ .
6. Repeat this for other test points with similar conditions for other networks.

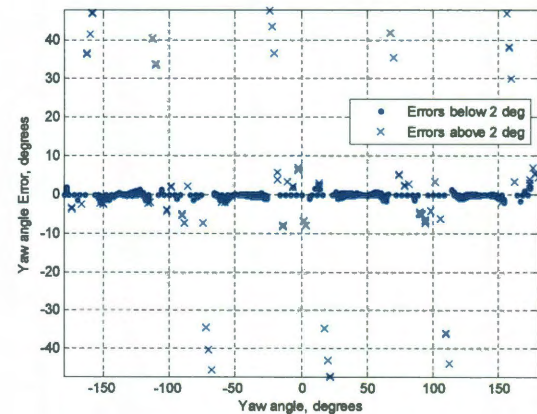
This strategy is also shown in a flowchart form in Fig. A.5. This realization would certainly help the estimation technique to locate the correct quadrant for each pressure sensor and thereby improve wind direction prediction accuracy. However, it would be adversely affected by the presence of noise in the pressure data. This yaw angle

estimation technique runs into problems when the test data has noise. Average pressure in each quadrant is a simple way to determine which quadrant the test point belongs. Random noise in the test static pressure and  $C_p$  distorts the average in each quadrant since only a finite number of sensors are present in each quadrant and yaw angle prediction accuracy decreases quickly with noise.

Once the wind speed predictor was in place, the predicted dynamic pressure values were used to obtain test coefficient of pressure values. The resulting test  $C_p$  values were then input to the inverse quadrant approach. As discussed before, separate networks were constructed for each quadrant. The tolerance for each network was again kept low because sparsity is not a primary concern here.



(a)



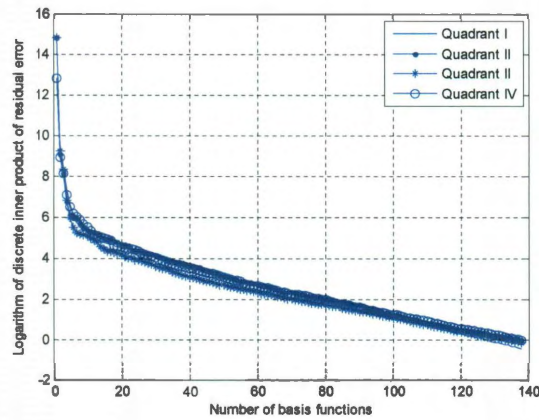
(b)

Figure A.3 a) Logarithm of discrete inner product norm of residual error for each quadrant and b) Yaw angle prediction errors on the RALS tower data when training was conducted on data at every 4 degree increments of yaw angle.

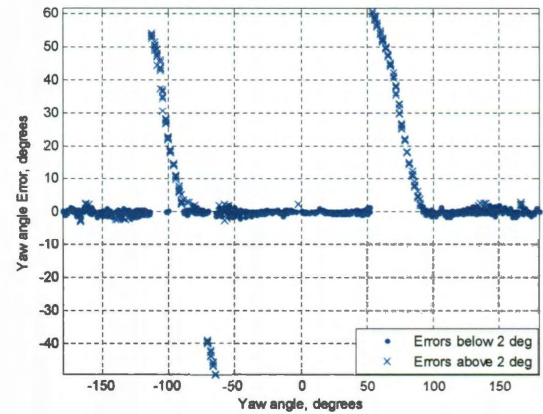
Once the networks have been constructed, the previously discussed testing strategy for the RALS tower was used to predict wind direction. Figure A.3 (a) and A.3 (b) show residual error convergence and yaw angle prediction errors on the RALS tower test set. Even with the RALS tower testing heuristic, few errors still exist in the vicinity of 0, 90, 180 and -180 degrees. Most of the errors with a magnitude greater than 10 degrees is due to an error in the quadrant selection. However, if the quadrant is well defined then the networks are able to predict yaw angle with errors less than 2 degrees.

For the surface vessel problem, once the wind speed predictor was in place, the predicted dynamic pressure values were used to obtain test coefficient of pressure values. The resulting test  $C_p$  values were then input to the inverse quadrant approach. As discussed before four separate networks were constructed, one for each quadrant, with pressure sensors distributed according to Table A.1. The tolerance for each network was again kept low because sparsity is not a primary concern here. Once the networks were constructed the previously mentioned testing strategy for the surface vessel was used to predict wind direction. Figures A.4 (a) and A.4 (b) show residual error convergence and yaw angle prediction errors on the surface vessel test set. Yaw angle prediction errors show strong bias on the port and the starboard side while the errors on the bow and starboard side are relatively low. This is because the average pressure value of each quadrant is not sufficient to discriminate between quadrants I and II and III and IV at +90 and -90 degrees respectively. A heuristic involving pressure sensors on both the port and starboard side could be developed, as shown in the Appendix, for the RALS tower problem. However, the forward approach to predict wind direction holds considerable

promise to show lower prediction errors as enough sensors are present around the geometry of the surface vessel.



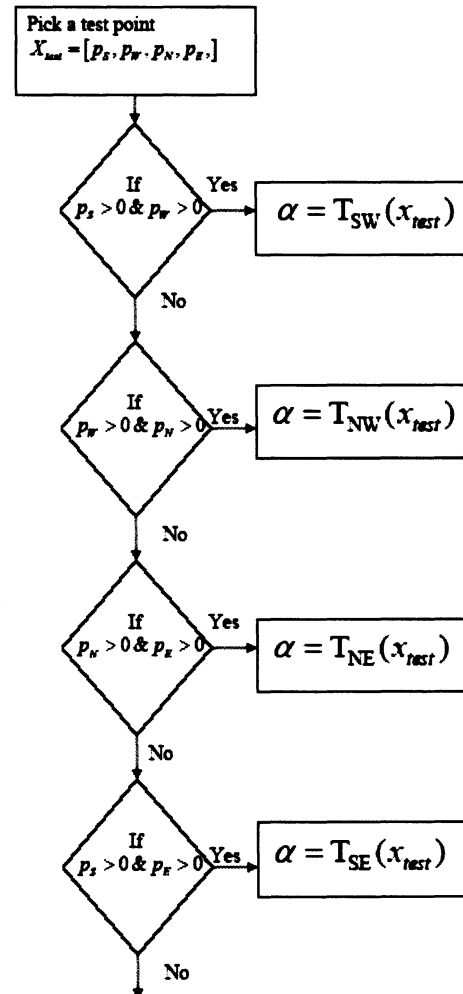
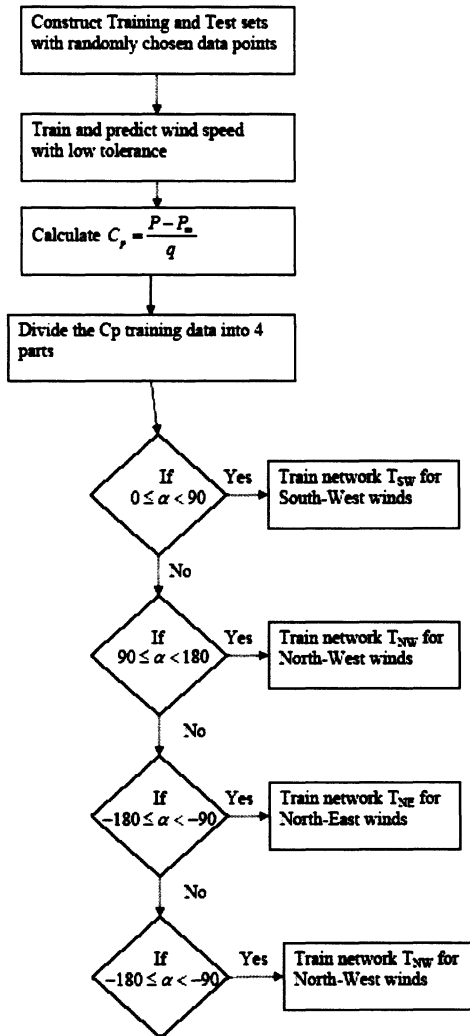
(a)



(b)

Figure A.4 a) Logarithm of discrete inner product norm of residual error for each quadrant and b) Yaw angle prediction errors on the surface vessel data when training was conducted on data at every 4 degree increments of yaw angle.







## Appendix B

### B.1 Active Learning for Classification Problems

In a classification problem, the task of a learning tool or a classifier is to map the  $d$ -dimensional input vectors to their respective classes or labels. A multi-class classification problem is attempted by breaking it down to several binary classification problems where the outputs are either -1 or +1. In the setting of a classification problem an informative training sample is one which cannot be classified with certainty. Addition of such a sample would have higher probability of changing the parameters of the surrogate model than any sample which could be predicted by more certainty. The task of an active learning tool would be to choose such an informative sample or query and ask the oracle to provide its output. Informative samples can either be chosen from a pool of unlabeled samples as is common in text classification problems, or they can be determined by solving an optimization problem maximizing some information gain criteria. Optimization approaches could also be used on a pool or random unlabeled samples generated according to the input distribution. However, line search approaches could be overwhelmed by the computational burden if the number of input dimensions is large. Also optimization approaches where a query is constructed might lead to odd training instances which the oracle cannot or has difficulty in labeling. In such cases, besides pool based active learning, stream based active learning or selective sampling methods are also popular where the learning algorithm goes through the unlabeled pool sequentially and decides whether or not to select them instead of ranking all unlabeled samples.

Say we are given a set of  $d$ -dimensional input data points  $\xi$  and outputs  $Y \in [1, 2, \dots, N]$ , where  $N$  is the number of classes. In binary classification tasks, the output can only take two possible values (-1 or +1). A straightforward way to compute the expected error from the addition of a point  $\xi$  to the training set is to calculate:

$$E_{\xi} = P(Y = 1 | \xi) \cdot E_{(\xi, Y=1)} + P(Y = -1 | \xi) \cdot E_{(\xi, Y=-1)} \quad (\text{B.1})$$

Equation (B.1) calculates the overall expected error on a test set when a data point  $\xi$  is added to the training set with output +1 or -1.  $P(Y = 1 | \xi)$  and  $P(Y = -1 | \xi)$  are the posterior probabilities that the label of the training point  $\xi$  is +1 and -1 respectively. Even though this approach is straightforward, it is computationally expensive [85]. In binary classification this computation can be avoided by selecting a data point that lies in the vicinity of the classification boundary. A point lying close to the class discrimination boundary is guaranteed to have an effect on the approximation of the discriminant function. Selecting unlabeled samples that lie in the vicinity of the discriminant boundary falls under the category of Uncertainty Sampling (US). Even though US does not optimize an information gain criterion, it has been proven to be effective in many practical applications [86-88]. The authors in the reference [85] have used US with Support Vector Machines (SVMs) for text classification problems. Since SVMs are discriminant classifiers computing the distance of data points from the class boundary is relatively straightforward. The authors tested this approach on two document classification problems. They concluded that only with a small number of actively chosen samples high classification accuracy could be obtained. These results were equivalent to

classification accuracy when all samples were used. And as the number of actively chosen samples was increased the classification accuracy would degrade and converge to random sample selection results. The authors in the reference [86] proposed a similar active learning method for SVMs. They proposed AL-SVM that would find an optimal solution by actively choosing samples from the input space according to US till no unlabeled samples lie within the margin of the support vectors. The authors tested the proposed algorithm on simulated problems and a mass spectroscopy problem of detecting hydrocarbons in the soil. They concluded that AL-SVM had good performance when the classes were well separated. Their performance suffered as the overlap between the classes increased. On the mass spectroscopy problem AL-SVM with only 14% of training points had performance similar to passive learning with the full training data set.

Authors in reference [89] proposed an active learning strategy that uses bootstrapping to estimate the class probability estimate of unlabeled samples and chooses a sample whose class probability estimates are tied. The authors compared their proposed active learning procedure with bootstrapping against learning with randomly selected training points on 20 real world data sets. They obtained an accuracy improvement on 15 out of 20 data sets when their proposed procedure was used instead of random sampling. The authors also compared their proposed algorithm against Uncertainty Sampling [77] and concluded that their proposed algorithm had superior performance on most data sets.

An alternate approach could be to search for new data points orthogonal to the space spanned by the current training set. This would be helpful when the problem has

high number of input dimensions. Other possible heuristics could be to choose places where there is no data, where there is poor performance, where the confidence is low, where one expects it to change our model or where data has resulted in learning [85].

### **B.1.1 Optimization Approaches**

Active learning approaches that fall under this category generate a new sample or choose an informative sample from a pool of unlabeled samples according to some optimization criteria. A popular optimization criterion is to maximize the expected information gain associated with a candidate training sample. Two popular approaches to calculate the expected information gain are either:

- i. To compute the reduction in the expected output variance of the model
- ii. To compute the expected reduction in the version space of the model.

Use of tools from Optimal Experiment Design [90], [75] has been popular to formulate a principled approach to estimate the output variance of the model. These techniques can be applied to both classification and regression problems and will be discussed in Section 5.1 where active learning for regression problems are discussed. A thorough literature survey is also available in the reference [25], [91]. As mentioned earlier a popular optimization criterion for active learning is to search for samples that reduce uncertainty of the model parameters. Query by Committee [92], [93] based methods are a popular approach that use a committee of models to estimate the expected information gain of an unlabeled sample. The concept of version space is often used to understand the meaning of information gain. Version space is a subset of all possible values of model parameters

or weight space that lead to zero training error [25]. So version space is the ideal set of model parameters that result in no training error on the problem and any unlabeled sample that eliminates uncertainty from the weight space would be an informative sample. These methods estimate the expected information gain of a sample as reduction in the version space of the problem. The reduction in the version space is measured by estimating the disagreement between the committee members on an unlabeled sample. The author in reference [25] have shown that the probability distribution of the model parameters and hence the associated information gain due to an unlabeled sample can be written in terms of the volume of the version space. For a binary classification problem an unlabeled sample divides the version space into two regions, each representing models that would classify the unlabeled point to each class. The expected information gain is maximized when the two volumes are equal. Therefore, an unlabeled point that bears the maximum disagreement among the committee members is the most informative sample.

Primary advantages of these approaches are that they have their foundation in the principles of information theory and are conceptually simple to implement since they do not depend on the formulation of the learner [25]. Shortcomings of Query by Committee include unrealistic assumptions that data is noise-free, existence of a perfect deterministic classifier, and that it is possible to draw classifiers from the version space. Uncertainty sampling approaches could be seen as a single classifier version of reducing the size of the version space and is described in the next section [77].

## B.2 Uncertainty Sampling with SFA

For algorithms like SFA that construct their approximations in the form of Eq. (2.1), points that lie in the vicinity of the classification boundary can be found by looking at the minimum absolute value of the argument of the proxy function. To implement uncertainty sampling with SFA the following steps need to be followed:

1. Use a Design of Experiment method like LHS technique to pick a small number of initial training points to begin the active data collection procedure.
2. Conduct training and evaluate the classifier on a pool of unlabeled samples.
3. Using  $u_n^a$ , determine the test point that bears the minimum absolute value of the argument of the proxy function. Multiple points can also be selected in a similar manner.
4. Add the chosen unlabeled samples to the training set and remove them from the pool of the unlabeled samples.
5. Repeat steps 2 through 4 until the pool of labeled samples is exhausted.

The application of this heuristic with SFA is demonstrated on the following simulated classification problem given in Eq. (B.2).

$$u(x, y) = \text{sign}[y - x^2] \quad \text{where } 0 \leq x, y \leq 1 \quad (\text{B.2})$$

The classifier was trained on one half of the available points and tested on the remainder.

Percentage accuracy was calculated by Eq. (B.3) shown below:

$$\text{Percentage accuracy} = \left( \frac{N_{test} - m_{test}}{N_{test}} \right) \times 100\% \quad (\text{B.3})$$

where  $N_{test}$  = number of points in the test set and  $m_{test}$  = number of misclassifications.

Results shown in Fig. B.1 were obtained by using SFA in conjunction with uncertainty



sampling to select new samples for labeling. The red and the blue curves in Fig. B.1 (a) show the increasing percentage prediction accuracy on a fixed test set (100 points) as training points were incremented sequentially using active learning and random sampling respectively. Initial approximations for both sampling strategies were constructed using ten points chosen by LHS design and two points were chosen at a time for labeling from a pool of 400 regularly spaced grid points. This process was repeated 50 times to eliminate any bias due to the choice of the initial set of training points. Fig. B.1 (a) shows mean percentage prediction accuracy computed over 50 permutations. Error bars show the standard error. Figure B.1 (b) shows the location of the data points chosen by the active learning scheme in one of the permutations. The active learning scheme clearly performs better than random sampling. Since there is an unlimited supply of unlabeled data points, the learning curves for active and passive learning do not converge.

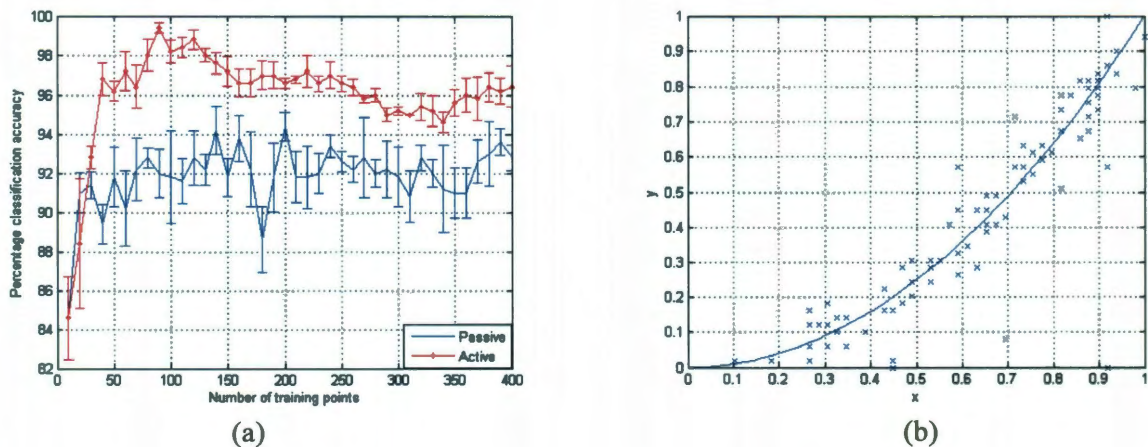


Figure B.1 a) Mean percentage classification accuracy as number of training points increase, errorbars show standard error and b) Crosses represent the training points chosen by the active learning scheme and solid line is the class discrimination boundary  $y = x^2$ .

As previously mentioned, multiclass classification problems are commonly attempted by combining several binary classifiers with a *one vs. all* or *one vs. one* approach. Selecting data points lying close to the classification boundaries in these problems would not be an optimal approach because one point could be informative for two classes but uninformative for the rest. A simple way to avoid this problem is to use the posterior probability estimates of the binary classifiers to pick a sample for labeling. According to uncertainty sampling an informative sample would be one that has the lowest classification uncertainty. Several authors have suggested picking a sample for labeling that bears the minimum difference between highest and the second highest posterior probability estimates [94], [87]. The authors in reference [87] proposed an active learning approach with SVMs for a multi-class image classification task of recognizing the types of plankton. The authors constructed several binary classifiers according to the *one vs. one* procedure and assigned probabilities to each classifier according to a parametric model. Informative images were chosen that had the smallest difference between the class probabilities. This approach was compared against the uncertainty sampling approach of reference [77] on plankton classification task. The authors concluded that the proposed approach had superior classification accuracy and often needed fewer images to do the job. In this work this method is used to select the next sample for labeling due to its simplicity and effectiveness.

SFA is a deterministic classifier that attempts to directly estimate the discriminant function of a binary classifier. Like Support Vector Machines (SVMs), SFA does not output posterior probabilities of a test point belonging to a particular class. Platt [95]

introduced a method to directly train the parameters of a sigmoid function to map the deterministic SVM outputs into posterior probabilities. Several authors extended this notion to a softmax function for multiclass classification problems [96] given by Eq. (B.4).

$$P(Y = k | \xi_i) = P_k^i = \frac{\exp[a_k u_n^{ak}(\xi_i) + b_{0,k}]}{\sum_{j=1}^M \exp[a_j u_n^{aj}(\xi_i) + b_{0,j}]} \quad (\text{B.4})$$

Here  $u_n^{ak}$  is the real valued output of the  $k^{\text{th}}$  binary classifier at  $\xi_i$ , and  $a_k, b_{0,k}$  are parameters of the softmax function. The parameters of the softmax function are determined by maximizing the following log-likelihood function given in Eq. (B.5).

$$\max_{a_k, b_{0,k}} \sum_{i=1}^s \log [P_k^i] \quad (\text{B.5})$$

### B.3 Wind tunnel experiment design for cavity flow classification

Internal carriage of stores in aircraft has many aerodynamic advantages especially in military applications. These include enhanced maneuverability, reduced drag and increased stealth of the aircraft. However, flow over the cavities might generate steady and unsteady disturbances that can affect safe discharge of stores [97]. Fig. B.2 (a) shows the open bomb bay of the F-22 Raptor which can generate self-sustaining oscillations that might lead to cavity resonance risking the structural integrity of the vehicle [98]. Fig. B.2 (b) shows the open missile bay of the F-35 Lightning II that can provide a large nose-up pitching moment to the stores on discharge [99]. These flow disturbances demand

extensive computational and experimental studies to be conducted at all operational speeds.



(a)



(b)

Figure B.2 a) Open bomb bay of a F-22 Raptor [98] and b) open missile bay of a F-35 Lightning II [99].

It is widely acknowledged that wind tunnel testing is essential in characterizing flow across a cavity. A number of parameters can influence the flow including freestream Mach number, geometric dimensions of the cavity, ratio of the boundary layer height to cavity depth, and location of stores within the cavity [97]. The number of parameter combinations and requisite data reduction can render wind tunnel testing tedious, expensive and time consuming. Any innovative mathematical technique that can reduce the time and expense of wind tunnel experiments is welcome.

In a related work, the authors demonstrated how machine learning tools could be used with Design of Experiments (DOE) to steer the experiment by investigating input parameter sensitivities to the classification of the cavity flow type [100]. The authors used SFA to predict the cavity flow type with or without acoustic resonance as a function

of length-to-depth ratio ( $l/h$ ), width-to-depth ratio ( $w/h$ ), and the freestream Mach number ( $M_\infty$ ). The authors treated this problem as a multi-class classification problem and justified the selection of SFA by comparison against state of the art classification tools.

However, the work presented in Reference [100] could only steer the cavity experiment by input parameter selection with respect to cavity flow type classification. It could not take part in the data collection procedure given a set of input parameters. This Section shows how Uncertainty Sampling with SFA could be used to construct a training set that leads to lower generalization errors compared to passive construction of training set by LHS methods. A total of 267 wind tunnel runs were conducted by Tracy and Plentovich [97] with the resulting data plotted in Fig. B.3. Percentage errors were computed in the same manner as that presented in the previous section. The misclassification error rate of SFA with training points chosen by US was compared against training points chosen by Latin Hypercube Sampling tested on all the available 267 points. The purpose of this comparison was to demonstrate that the active machine learning algorithm achieves better generalization ability, i.e. better cavity flow type prediction accuracy with fewer training points. The current problem is treated as a 3-class classification problem and focus on generating optimal  $M_\infty$  and  $l/h$  combinations for each  $w/h$ . Focus is not given on determining optimal  $w/h$  ratios because of its insufficient resolution in the available data.

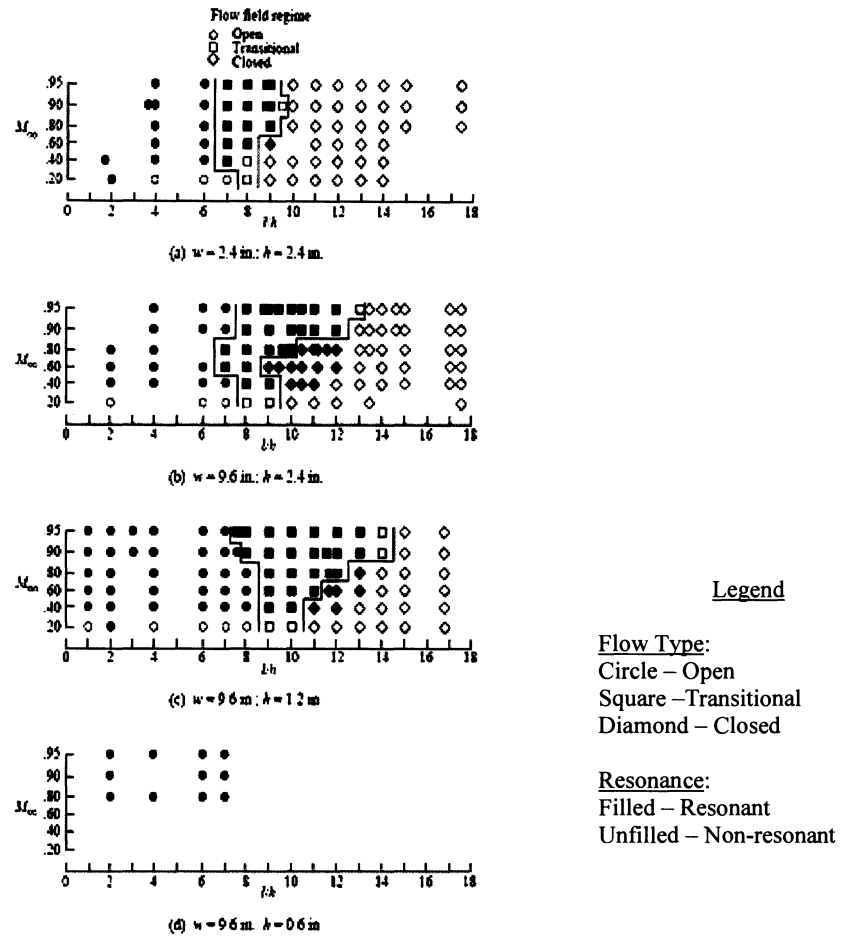


Figure B.3 Occurrence of resonance superimposed with observed cavity flow conditions [97].

The active learning scheme applied to the cavity problem can be schematically represented by Fig. B.4. Once a final training set has been constructed, the final form of the discriminant function can be constructed to predict labels in real time. The active machine learning algorithm initially needs a few training points to construct a hypersurface to determine the most uncertain data points. There is no principled approach to determine the optimum number of initial points, so 20 initial points were used to start the active learning procedure. These points were chosen by the LHS method on two dimensions  $M_\infty$  and  $l/h$ . To eliminate any biases due to the randomness in the LHS designs, the training and testing procedure was repeated 50 times. Two points were added

to the training set each time and the active data selection process continued until all of the available points were used. This approach allows the user to choose a single point or a batch of points per iteration. Two points were added per iteration to save computational time spent to complete 50 random permutations used to eliminate bias in the results.

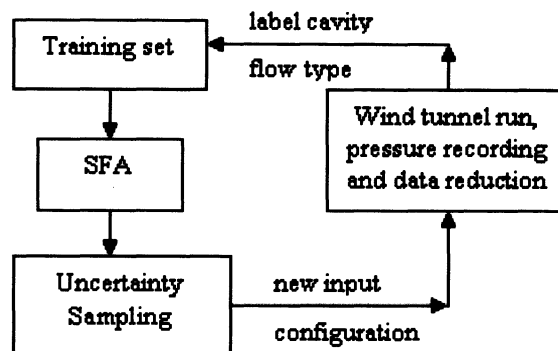
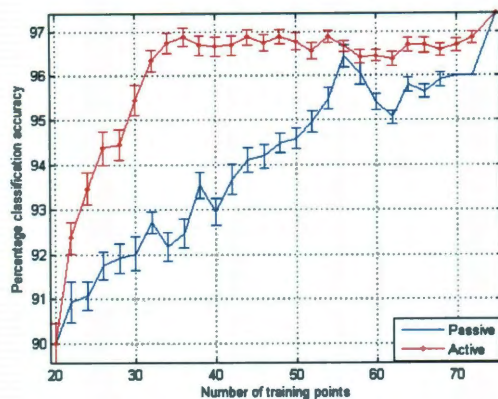
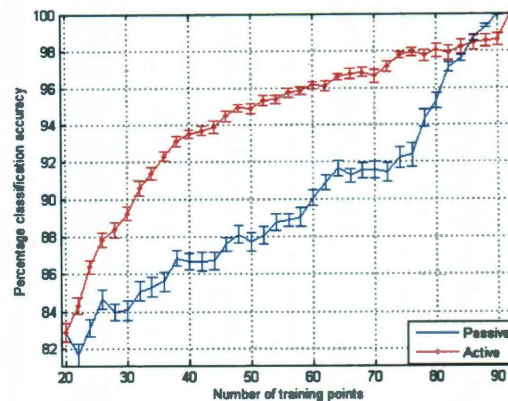


Figure B.4. Schematic representation of Active Learning.

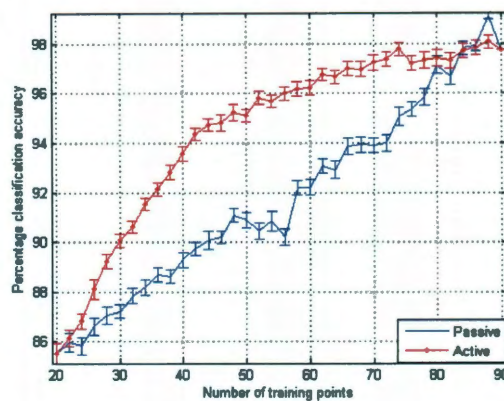
Figure B.5 shows that active learning clearly outperforms the passive LHS technique at all three  $w/h$  ratios. Active learning accuracy increases sharply and then gradually converges with LHS classification accuracy as unlabeled samples are exhausted. With 40 training data points at  $w/h = 1$ , active learning has a classification accuracy of  $97 \pm 0.2\%$  while LHS has  $93 \pm 0.3\%$ . This means that if the user decides to conduct 40 wind tunnel runs at  $w/h = 1$ , then actively sampling input configurations by the US technique would result in data that has more information than data sampled passively by the LHS technique. Training data sampled from critical regions of the input-output hyperspace give more generalization ability to SFA than the training data sampled just from the input space.



(a)



(b)



(c)

Figure B.5. Mean cavity flow type prediction accuracy of active and passive learning at a)  $w/h = 1$ , b)  $w/h = 4$  and c)  $w/h = 8$ . Error bars show the standard error.

The only disadvantage US based active learning has compared to passive LHS technique is its greater computational expense. However, the relatively high cost of wind tunnel testing more than justifies the increased computational cost. The active learning curve for  $w/h = 1$  flattens out at about 35 training points because the US algorithm chooses almost all of the points lying close to the classification boundaries in the first 8



iterations. Table B.1 shows cavity flow type classification accuracy of SFA, using half of the data points for training, sampled by the US and LHS techniques.

| $w/h$ | Number of<br>training points | LHS<br>(%)   | US<br>(%)    |
|-------|------------------------------|--------------|--------------|
| 1     | 38                           | $94 \pm 0.3$ | $97 \pm 0.2$ |
| 4     | 46                           | $88 \pm 0.4$ | $95 \pm 0.3$ |
| 8     | 44                           | $90 \pm 0.4$ | $95 \pm 0.3$ |

Table B.1 Comparison of classification accuracies

The use of SFA with the Uncertainty Sampling technique was demonstrated on a multi-class cavity flow type prediction problem. It is believed that active machine learning tools have the potential to help engineers to accelerate through wind tunnel testing by steering through the test matrix in an incremental and optimal manner.