STOCHASTIC OPTIMISATION OF AMERICA'S CUP CLASS YACHTS

BY

ANDREW PHILLIP MASON

A THESIS SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR

THE DEGREE OF

DOCTOR OF PHILOSOPHY

AUSTRALIAN MARITIME COLLEGE

UNIVERSITY OF TASMANIA

FEBRUARY 2010

DECLARATION OF ORIGINALITY

This thesis contains no material which has been accepted for a degree or diploma by the University or any other institution, except by way of background information and duly acknowledged in this thesis, and to the best of my knowledge and belief, no material previously published or written by another person, except where due acknowledgement is made in the text of the thesis, nor does the thesis contain any material that infringes copyright.

Much of the experimental data reproduced within this thesis is the property of the Alinghi America's Cup team and is reproduced with permission.

[illegible] Andrew Mason, February 19, 2010

STATEMENT OF AUTHORITY OF ACCESS

[illegible] Andrew Mason, February 19, 2010

ABSTRACT

This thesis describes the design and implementation of an optimisation system for America's Cup Class (ACC) yachts. The system, named VESPA, uses a measure of merit that closely approximates the actual America's Cup race format; a round-robin match-racing tournament, held over many races between a population of candidate designs, using a stochastic wind model. VESPA was used by the Alinghi team to provide design recommendations for the 2007 America's Cup.

The optimisation of racing yachts is a problem that has been considered resistant to full analysis due to its complexity. Consequently, attempts at yacht design optimisation to date have been restricted to simplified subsets of the problem. While Velocity Prediction Programs (VPP) have been widely used to provide details of sailing performance for one or more yachts, the statistical models on which these programs are based have not been sufficiently accurate to allow optimisation of hull shapes. Other efforts to automate yacht design optimisation have used an objective function that evaluates the performance of each boat using Computational Fluid Dynamic (CFD) analysis of the hull. This approach suffers from long execution times, which may result in the adoption of a restricted measure of merit, such as hull resistance at a small number of forward speeds, heel and yaw angles.

In order to permit the use of the chosen measure of merit while retaining acceptable performance, a sparse sample of designs, derived from a parent hull using a novel parametric transformation method, had their hydrodynamic characteristics calculated by the SPLASH potential flow code. The output from SPLASH was subsequently used to train a set of neural-network based hydrodynamic metamodels for use by the VPP. The need to assess a population of designs for the tournament-based measure-of-merit makes the problem well suited to stochastic, population based optimisation methods. As a result, a Genetic Algorithm (GA) was chosen to perform the optimisation, using a parsimonious Race Modelling Program (RMP) to simulate a tournament of races based on performance data provided for each boat by the VPP.

Each component within the VESPA system was validated to ensure confidence in the optimisation results. Optimisation runs were performed over several months using multiple parent models to investigate the effect of changes to various design variables. Finally, a design optimised by VESPA was tank tested at 1/3 scale, confirming the improvements over its parent design predicted by SPLASH.

VESPA proved itself capable of making genuine design improvements to an existing parent model while retaining reasonable execution times. VESPA also revealed several unexpected insights into the nature of the solution space for the design of ACC yachts, including multiple optima and the potential for intransitivity in the solution when interactions between boats at rounding marks are considered.

ACKNOWLEDGEMENTS

I would like to express my heartfelt gratitude to my wife Mandy, who despite her opinion that Ph.D. stood for "Phile for Divorce", showed immense patience in allowing me the time necessary to complete this work.

My thanks to my mother, for her patience and gentle guidance over the years, and to my late father, for buying me a boat and teaching me to sail as a child, instilling in me an enduring love of the water.

I would also like to thank my supervisor, Dr. Giles Thomas, for the disciplined approach he brought to the task and for his quiet encouragement, without which this thesis may have foundered.

Thanks also to Barbara Backhouse for her meticulous proofreading skills.

Finally, I would like to thank the Alinghi design team, in particular Manuel Ruiz de Elvira, Grant Simmer, Rolf Vrolijk and Michael Richelsen, for the support and assistance they provided, without which this work would not have been possible.

FOREWORD

"Every point of sailing suggests an appropriate and different form of hull. The shape that is well adapted for one kind of weather is ill adapted for another sort; vessels that move as by magic in light airs may be of little use in a whole sail breeze; one that is by no means a flier in smooth water may be very hard to beat in a seaway.

In short, a vessel must be light enough to be driven easily by a moderate breeze, stiff enough to stand up to her canvas in a hard wind, shallow enough to be docked with ease and to run with speed. She must have depth enough to hold her up to windward, breadth enough to give her stability; she should be long enough to reach well, and short enough to turn well to windward; low in the water so as not to hold too much wind, with plenty of freeboard to keep the sea off her decks.

The satisfaction of any one requirement necessitates something antagonistic to some other requirement equally clamorous for satisfaction. Your vessel, to be perfect, must be light, of small displacement, and with the centre of gravity brought very low; she must also have large displacement, and the ballast must not be too low, in order that she may be easy in a seaway; she must be broad, narrow, long, short, deep, shallow, tender, stiff. She must be self-contradictory in every part. A sailing yacht is a bundle of compromises, and the cleverest constructor is he who, out of a mass of hostile parts, succeeds in creating the most harmonious whole.

It is not strange that designers pass sleepless nights, and that anything like finality and perfection of type is impossible to conceive. No wonder that yacht designing is a pursuit of absorbing interest."

Lord Dunraven, challenger for the 1893 and 1895 America's Cups.

CONTENTS

FIGURES

TABLES

NOMENCLATURE – NAVAL ARCHITECTURE

A_W	waterplane area
B_{WL}	maximum beam of waterline
C_F	coefficient of frictional resistance [ITTC-57 correlation line]
C_{IL}	longitudinal waterplane inertia coefficient
C_M	midship area coefficient
C_P	prismatic coefficient
C_R	coefficient of residuary resistance
C_T	coefficient of total resistance
C_W	coefficient of wave resistance
C_{WP}	waterplane area coefficient
Fr	Froude number, $Fr = V/\sqrt{gL}$
Fr_d	depth Froude number, $Fr_d = V/\sqrt{gd}$
FB	distance from the forward perpendicular to centre of buoyancy
g	acceleration due to gravity
L_{CB}	non-dimensional longitudinal centre of buoyancy, $L_{CB} = FB/L_{WL}$
L_{CF}	longitudinal centre of flotation
L_{CG}	longitudinal centre of gravity
L_{WL}	waterline length
$L/\nabla^{1/3}$	slenderness ratio
Re	Reynolds number [VL/ν]
R_{RU}	residuary resistance upright
$R_{R\varphi}$	residuary resistance heeled
R_V	viscous resistance
S	wetted surface area
T_C	draft of canoe body
V_B	boat velocity
∇	volume of displacement
ν	fluid kinematic viscosity
φ	heel angle
ρ	fluid density

NOMENCLATURE – YACHT DESIGN

β_{TW}	true wind angle
β_{AW}	apparent wind angle
V_{AW}	apparent wind velocity
V_{MC}	velocity made good relative to the direction of the course
V_{MG}	velocity made good relative to the direction of the wind
V_{TW}	true wind velocity
λ	leeway angle

NOMENCLATURE - STATISTICAL

SD or σ	Standard Deviation	$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2}$, where $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$
AAE	Average Absolute Error	$(\sum_{i=1}^{N} \lvert x_i - x_i' \rvert)/N$
ARE	Average Relative Error	$(\sum_{i=1}^{N} \frac{\lvert x_i - x_i' \rvert}{\lvert x_i \rvert})/N$

ACRONYMS – DESIGN & ANALYSIS

ACC	America's Cup Class
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
DOE	Design of Experiments
FFD	Free Form Deformation
NURBS	Non-Uniform Rational B-Splines
PDF	Probability Density Function
RANS	Reynolds Averaged Navier-Stokes
RAO	Response Amplitude Operator
RMP	Race Modelling Program
VESPA	Virtual Evolution based Sailing Performance Analysis
VPP	Velocity Prediction Program

ACRONYMS - OPTIMISATION METHODS

BFGS	Broyden Fletcher Gordfarb Shanno
DFP	Davidson Fletcher Powell
EA	Evolutionary Algorithm
ES	Evolution Strategies
GA	Genetic Algorithm
GP	Genetic Programming
LM	Levenberg Marquardt
MMA	Method of Moving Asymptotes
MOGA	Multi-Objective Genetic Algorithm
NLP	Non-Linear Programming
PSO	Particle Swarm Optimisation
QN	Quasi Newton
SA	Simulated Annealing
SQP	Sequential Quadratic Programming
SUMT	Sequential Unconstrained Minimisation Technique

ACRONYMS - APPROXIMATION METHODS

ANN	Artificial Neural Networks
MARS	Multivariate Adaptive Regression Splines
PR	Polynomial Regression
RBF	Radial Basis Functions
RSM	Response Surface Methods
SVM	Support Vector Machines

AMERICA'S CUP CLASS - RULE VARIABLES

ABC	aft beam correction
AG	aft girth
AGC	aft girth correction
AGS	aft girth station
ALM	aft length mark (positioned at the aft end of *LBG*)
DSP	ACC displacement in cubic metres
E1..E5	mainsail girths
FBC	forward beam correction
FG	forward girth
FGC	forward girth correction
FGS	forward girth station
FLM	forward length mark (positioned at the forward end of *LBG*)
FP	forward penalty
G	girth component of *LM*
I	height of foretriangle
J	base of foretriangle
L	rated length
LBG	length between girth stations
LM	measured length
MWL	measurement water line
P	mainsail luff length
S	rated sail area
SM	measured sail area
W	weight of yacht
Φ	mean of the angles port and starboard of the topsides measured at *AGS*, relative to the vertical, in degrees
Θ	mean of the angles port and starboard of the topsides measured at *FGS*, relative to the vertical, in degrees

"The America's Cup is a design race. At this level, against this calibre of sailors, if you have a slightly faster boat you are going to beat them. At every America's Cup I've done, the fastest boat has won."

Brad Butterworth, Alinghi skipper and three times America's Cup winner.

1 · OVERVIEW

The 2007 America's Cup involved 12 teams whose budgets totalled more than \$US1 billion. Of this figure, approximately 25% was expended on design, research and development aimed at winning the America's Cup final, a series of 7 yacht races held over a period of 2 weeks.

Despite this large investment of time and money, design optimisation of ACC yachts has remained primarily a manual process. This is not to say that the design of ACC yachts does not involve sophisticated technology; ACC design teams use some of the most powerful computational fluid dynamics and structural analysis software available, and utilise clusters of high performance computers to perform a volume of calculations that would have been unthinkable 20 years ago.

Rather, the problem is that there has been little work done on automating the exploration of the design space. Despite the emphasis on computational tools, the overall approach to design optimisation has changed little since computer based simulation tools were adopted.

Prior to the 1980s, the design development of America's Cup yachts primarily relied on tank testing, with only limited computing power available for numerical analysis. With computer costs plummeting and computing power increasing dramatically in the early eighties, it became possible to consider simulating, on a computer, the experiments previously performed in a towing tank.

Prior to the introduction of Computational Fluid Dynamics (CFD) software, yacht design optimisation had been based on a matrix of proposed designs that were built as scale models and tank tested. This allowed some parameter variation to take place and conclusions to be drawn from the result. However, the time and cost of tank testing prohibited the systematic optimisation of a hull design.

The quality of the finished hull design was very much dependent on the skill and experience of the designer, and the conclusions that could be drawn from a small quantity of data and a large amount of intuition. The insight of the yacht designer was paramount and it is no surprise that the post-war years of the America's Cup were dominated by the designs of one exceptional designer, Olin Stephens, who was involved in the design of all but one America's Cup winner from 1937 to 1980.

One of the first America's Cup yachts to gain significant advantage from numerical simulations was Australia II, winner of the 1983 America's Cup, with the intuition of her designer Ben Lexcen complemented by the computational work of two Dutch engineers, Peter van Oossanen and Joop Slooff, (van Oossanen 1985). Since then, cup teams have devoted ever-increasing amounts of effort to both hydrodynamic and aerodynamic simulations of the hull and rig designs.

Although much work has been done in the past 25 years with CFD analysis of yacht designs, surprisingly little has been done to automate the process to allow the computer to search for an optimal design for a given set of weather conditions. To some degree this has been due to the large amount of computing power required. It is also a consequence of the enormous complexity of the yacht design and analysis process. To solve the problem effectively there is a need to reduce this complexity, where appropriate, in order to perform the optimisation in a reasonable amount of time on available computer hardware.

If a workable design optimisation system could be developed, several benefits would result. Firstly, optimisation of new designs would occur more rapidly. It has taken more than fifteen years for America's Cup Class designs to evolve from wide, heavily flared yachts with soft bilges, to narrow, hard-bilged hulls with vertical topsides. It is possible that the optimality of this area of the design space could have been rapidly discovered by a well-designed optimisation system.

Once a design is close to optimal, it becomes more difficult to discover design variations that will result in improvements in performance. In this area of diminishing returns, an automated optimisation system may also have significant benefits, allowing design improvements to be achieved without the expenditure of an excessive number of man hours.

In addition, if the system is able to perform global optimisation, thoroughly exploring the available design space, novel design approaches may be uncovered that might not be obvious to a designer employing traditional manual methods.

1.1 PROBLEM DEFINITION

This thesis describes the research, design and development of a design optimisation system for America's Cup Class (ACC) yachts. The system, named VESPA (short for Virtual Evolution based Sailing Performance Analysis), has been developed in collaboration with the Alinghi America's Cup team. VESPA was used to make design recommendations for Alinghi's successful defence of the 2007 America's Cup.

VESPA was developed with the aim of assisting a yacht designer to determine the optimal design parameters required for a yacht to succeed in winning the America's Cup. In this regard, VESPA does not attempt *ab initio* design of the hull shape. Rather, VESPA attempts to refine a specific parent hull by varying a small number of key parameters, while retaining key design features that may have resulted from the expenditure of thousands of hours of manual refinement and research.

The first step in such an optimisation problem is determining how to ascertain whether one yacht is superior to another; that is, what measure of merit should be used? As pointed out by Harries (2001a):

> *"For an ACC yacht the design evaluation becomes a challenging task in itself since a probabilistic measure of merit ought to be considered... however, an extraordinary amount of computation will be required to determine this ultimate measure of merit and to optimise for it." (2001a, pp.11-12)*

Harries highlights the two key issues in creating a system such as VESPA; the correct choice of measure of merit and the problem of system performance. Using modern Reynolds Averaged Navier-Stokes (RANS) CFD codes directly in an optimisation loop could conceivably result in execution times that stretch into weeks or months for the fastest optimisation methods. Korpus (2004) expressed similar concerns regarding measures of merit and the performance of RANS codes:

> *"While faster computers have made RANS analyses possible, most applications to date fall short of being practical. If an America's Cup designer is to improve boat speed, he or she must analyze hundreds of design alternatives - not the few isolated samples usually associated with RANS. And even when a large number of flow analyses are available, the measures of merit required to rank designs are not obvious RANS outputs like flow detail or drive force." (2004, p.249)*

Unfortunately, the most appropriate measure of merit for this particular problem, the win/loss probabilities for an entire fleet of boats competing in a round-robin match-racing tournament, requires some of the slowest optimisation methods. This is due to it being necessary to maintain a population of designs, rather than attempting to optimise a single design. This combination of slow analysis with slow optimisation methods makes the option of direct CFD analysis unattractive, even when using solvers running in parallel on a cluster of computers.

1.2 BACKGROUND AND CONTEXT

In general, the design of ships and boats can be considered an optimisation problem in which a multitude of factors, including size, cost, speed, seaworthiness, stability, comfort, manoeuvrability, accommodation space, crew workload and even aesthetics are traded off against one another. The interaction between these objectives and constraints can be complex, with some complementary and many others conflicting.

Yacht design is a specialised discipline within the field of naval architecture due to the need to account for the presence of sails that apply both driving force and heeling moment to the yacht. These forces affect the stability, motions and appendage design of the vessel, the result being an extremely complex set of constraints and objectives that need to be satisfied. For this reason, yacht design has often been regarded as more of an art than a science, due to the number of variables and their resultant interactions being so numerous as to defy complete analysis.

Racing yacht design adds further complexity, with a mix of performance characteristics, both upwind and downwind, in a variety of sea conditions and wind velocities, needing to be considered.

The design process used by naval architects to satisfy multiple, conflicting objectives has traditionally proceeded through a series of iterations of what has been referred to as a design spiral (Evans 1959), illustrated in Figure 1.

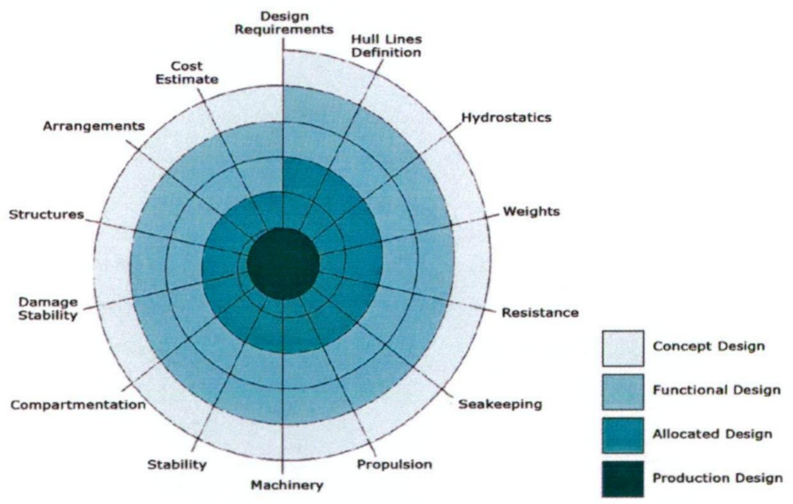


Figure 1. Design Spiral

Starting at the conceptual design stage, the naval architect defines a preliminary set of hull lines and performs various forms of analysis to determine whether the design will satisfy requirements for such objectives as stability, seakeeping, strength, performance and construction cost. Modifications are made to the hull lines, compartmentation, arrangements, equipment and structure, and the process is repeated, each revolution around the spiral resulting in a more refined design. This process continues through the various design phases in a sequential and iterative manner until all design criteria are adequately satisfied, concluding with the production design phase.

Although this process has served the naval architecture community well for many years, it is time consuming and not guaranteed to result in an optimal design. Balancing multiple, conflicting objectives is a difficult task requiring a great deal of skill and experience on the part of the designer, and time and cost limitations often result in less than optimal designs being constructed.

The design spiral approach can be viewed as an extension of the widely held idea that when seeking to improve a system that has multiple parameters, it is important to vary only one parameter at a time while keeping the others constant, so that the effect of any change can be easily quantified.

Within the field of optimisation, this approach is referred to as the alternating variable method (Fletcher 1987). It typically progresses by searching a single parameter direction until no further improvement is seen, followed by a switch to a different parameter, which is again searched until once again no further benefit is seen. At this point additional parameters may be searched or the process may repeat from the first parameter until no improvement can be made.

An example of the alternating variable method incorporating two parameters is illustrated in Figure 2. Parameter P1 is searched until a maximum is found, followed by a search using parameter P2 resulting in a solution that is very close to the optimum. This may be improved by repeating the cycle one or more times. On well-behaved functions that have a reasonably small number of independent parameters, the approach rapidly converges on the optimal solution.

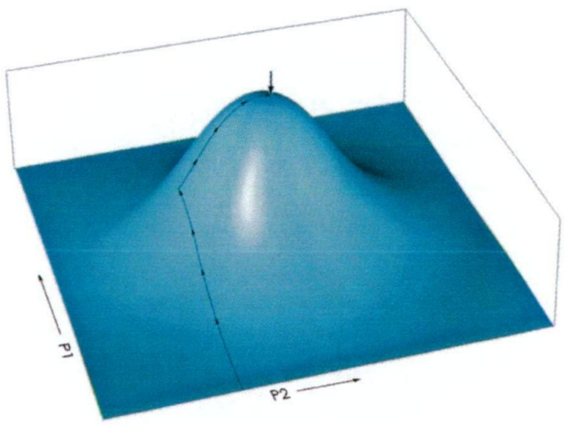


Figure 2. Alternating variable method

The problem with this method is that it ignores the possibility of correlation between the variables. This correlation effectively causes a ridge or valley to form diagonally through the solution space. When this occurs, it causes the search in the current search direction to destroy the property that the current point is the minimiser or maximiser in previously used directions, (Fletcher 1987).

This leads to oscillatory behaviour of the alternating variable algorithm, illustrated in Figure 3.

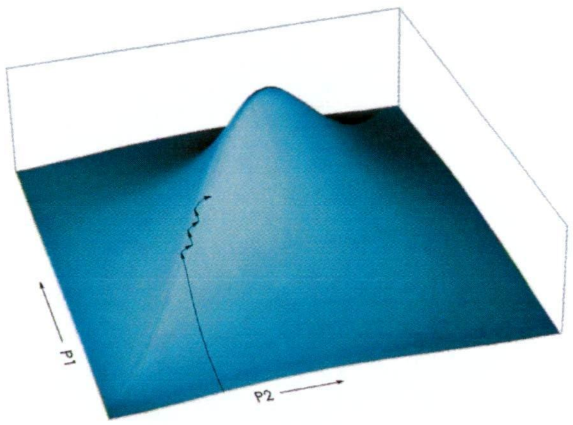


Figure 3. Alternating variable method with correlated variables

In this case, an initial search along P1 locates a ridge in the solution surface that runs diagonally relative to the variables being searched. A switch to a search of P2 does not go far before it also reaches a maximum. A further switch back to the P1 search direction makes another small gain, followed by small gains on each successive alternation of the search direction.

Interdependence of design variables may confound any search for an optimal design, as a change to one variable needs to be matched by simultaneous changes to the correlated variables. The result is an optimisation method that either progresses slowly or fails to locate an optimum at all.

The alternating variable method is seldom used as a computer based optimisation algorithm, and automated optimisation methods have been designed to avoid these pitfalls. The traditional naval-architecture design spiral is functionally equivalent to the alternating variable method and suffers from similar shortcomings when parameters are not independent. Unfortunately, in the design of ships and boats, interdependence of design parameters is the rule rather than the exception.

In contrast, many computer based optimisation algorithms, for example the conjugate gradient method, (Hestenes and Stiefel 1952), attempt to determine the best search direction for convergence. The potential for automated optimisation procedures to find solutions where manual methods fail suggests that there may be scope for an automated, computer based optimisation system to improve on the best designs achieved using the conventional design spiral.

1.3 OPTIMISATION – A DEFINITION

The term optimisation has been widely used within the field of marine design to describe many different processes. These include manual searches of design alternatives, the selection of the best design from a systematic series or matrix of designs, or an automated search using computer-based simulations.

To clarify the meaning of the term as used within this thesis, optimisation is defined as the process of finding a good (and possibly the best) feasible solution to a problem, based on the evaluation of an objective criterion.

An optimisation problem with a single objective can be stated as:

$$\textit{minimise} \;\; F(\mathbf{x}), \qquad \mathbf{x} \in \mathbb{R}^n$$

having the following components:

Measure of merit or objective function: criteria by which a solution is assessed:

$$F(\mathbf{x})$$

Free variables, decision variables, design variables: independent variables that can be modified directly and which uniquely describe the optimisation problem:

$$\mathbf{x} = (x_1, x_2, \dots, x_n)^T$$

Constraints: define the boundaries of the feasible regions of the design space. These can be categorised as:

- Bounds:

  $$x_{imin} \leq x_i \leq x_{imax} \text{ for i = 1,2,...,n}$$

- Equality constraints:

  $$h_j(\mathbf{x}) = 0 \text{ for j = 1,2,...,m}$$

- Inequality constraints:

  $$g_k(\mathbf{x}) \geq 0 \text{ for k = 1,2,...,p}$$

Dependent variables: are those not directly controlled, but dependent on the values of the free variables.

Parameters: additional values and conditions that are not under the direct control of the optimisation process.

Constants: values that do not change during the optimisation process.

1.4 DESIGN OPTIMISATION – STATE OF THE ART

During the past 30 years, significant research has been performed within the field of naval architecture on the subject of automated design optimisation of marine vessels. Although much work has taken place in such areas as optimisation of structure, compartmentation and propulsive systems, these domains are outside the scope of this thesis, whose focus is on those factors that specifically influence the hydrodynamic performance of the vessel.

In order to review the existing work in this field it is necessary to be aware of the key areas in which the various optimisation approaches examined may differ. These can be summarised as-

- **What are we trying to improve?** That is, what measure of merit should we choose?
- **How do we measure it?** What analysis method should we use to calculate the measure of merit?
- **How do we improve it?** What optimisation method do we select?
- **How do we vary design parameters in order to make improvements?** What hull geometry representation and variation method should we adopt?

Researchers have found many different answers to these questions during the past three decades, and research and development is ongoing. Despite dramatic advances in computer hardware and software performance, the goal of fast, flexible and accurate simulation and optimisation is yet to be achieved. Creating a system that has acceptable performance and provides useful results is a significant challenge.

Before reviewing the research on the optimisation of sailing yachts in general, and ACC yachts in particular, it is appropriate to examine the research that has been performed on the optimisation of other forms of marine vessels. Although yacht design is a highly specialised area within the field of naval architecture, developments in design and analysis methods for full-sized ships share many common features.

1.4.1 EARLY SHIP DESIGN OPTIMISATION

Prior to 1980 there are surprisingly few instances in the literature describing the use of optimisation methods within the field of naval architecture. Much of the early research into ship design optimisation was limited to the investigation of variations in principal dimensions at the initial design stage, rather than the detailed refinement of hull shapes.

The measure of merit typically used was based on ship economics, rather than focussing on specific objectives such as resistance or ship motions. A summary based on Nowacki (2001), showing some early optimisation studies in basic ship design, is shown in Table 1.

Table 1. Early Optimisation Studies in Basic Ship Design

Authors	Year	Ship Type	Criterion	Approach
Murphy et al.	1965	GCS	LCC	Systematic variation and interpolation
Mandel and Leopold	1966	T, GCS	LCC, RFR, NPV	Exponential random search, unconstrained
Gallin	1967	GCS	Several choices	Systematic variation
Kuniyasa	1968	T, BC	CRF	Parametric study
Gilfillan	1969	BC	RFR	Parametric study
Puchstein	1969	GCS & Diverse	Building cost, owner's criteria	Non-linear model, implicit constraints
Nowacki et al.	1970	T	RFR	NLP: SUMT & DS
Fisher	1972	BC	RFR	NMS, implicit constraints
Lorentz	1973	T	CRF, operational cost	NLP: SUMT & DS
Jagoda	1973	Diverse	NPV	NMS
Söding and Poulsen	1974	BC	AAC	NLP with slack variables
Nowacki and Lessenich	1976	T,BC, GCS	RFR	NLP: feasible directions & penalty functions
Kupras	1976	BC	Building cost	Parameter studies & adapted DS

Abbreviations

BC = bulk carrier	AAC = average annual cost	DS = direct search
GCS = general cargo ship	CRF = capital recovery factor	NLP = non-linear programming
T = tanker	LCC = life cycle cost	NMS = Nelder-Mead simplex
	NPV = net present value	SUMT = sequential unconstrained minimisation technique
	RFR = required freight rate	

Parsons (1975) was among the first to publish a comprehensive review of optimisation methods targeted at practising naval architects. Rather than focussing on solutions to specific problems, this work is primarily a description of optimisation techniques, with special focus on the Nelder-Mead Simplex method (Nelder and Mead 1965) and the Hooke-Jeeves direct search method (Hooke and Jeeves 1961).

1.4.2 ANALYSIS METHODS

Early researchers such as Nowacki et al. (1970), Jagoda (1973) and Kupras (1976), based their optimisation work on statistical models derived from either the performance of full-size ships, or measurements determined from tank testing. In the case of resistance, regression analysis methods such as Holtrop and Mennen

(1978), Oortmerssen (1971) and Savitsky (1964) have been widely used in ship design since their initial formulation, and subsequently in early optimisation studies. The seakeeping optimisation examined in Bales (1980) used a regression model that was based on measurements derived from a systematic series of hulls and analysed using strip-theory calculations.

As low-cost computers became available in the early 1980s, direct numerical analysis methods became more widely used in ship design optimisation. Ship resistance was typically calculated using relatively simple numerical methods such as thin ship theory (Hsiung 1981; Hsiung and Shenyan 1984; Hsiung and Shenyan 1985), with seakeeping being predicted using strip-theory analysis of geometry based on Lewis mapping of hull sections, (Walden et al. 1985).

Slender ship theory (Noblesse 1983) has also been used by several researchers. Due to its performance advantages, slender ship theory continues to be of value in circumstances where slender hulls such as catamarans and trimarans are optimised, (Yang et al. 2000; Hendrix et al. 2001; Percival et al. 2001; 2001).

COMPUTATIONAL FLUID DYNAMICS - PANEL METHODS

CFD is the process of simulating the fluid flow around a body such as a yacht hull by solving a set of equations, ideally the Navier-Stokes equations, which describe how the velocity, pressure, temperature and density of a moving fluid are related. Due to the difficulty of finding solutions to these equations, CFD methods initially used idealisations that simplified the task. These idealisations are -

- removal of viscosity terms from the Navier-Stokes equations yields the Euler equations;
- removal of vorticity from the Euler equations yields the full potential equations;
- linearisation of these yields the linearised potential equations.

These linearised potential equations were the first to be solved, with two-dimensional methods being developed in the 1930s. However, a practical three-dimensional method to solve the linearised potential equations was not developed until 1966, (Hess and Smith 1966). This method breaks the surface geometry into panels, resulting in this class of programs being referred to as panel methods.

Although panel methods have been used for some time for ship resistance calculation, the first appearance in the literature of an optimisation process based on panel methods is Maisonneuve (1993). Maisonneuve used the REVA CFD code (Maisonneuve 1989) for the calculation of resistance, together with the AQUA code (Delhommeau 1987) for seakeeping, in order to optimise a Small Waterplane Twin Hull (SWATH) vessel.

ADVANCED CFD METHODS IN OPTIMISATION

Some optimisation work has taken place using Reynolds Averaged Navier-Stokes (RANS) CFD methods. However, the high computational cost of RANS methods has resulted in several different approaches being taken to reduce this burden.

An example of the performance penalties incurred by modern RANS codes is provided by Parolini and Quarteroni (2007), who described analysis work performed on ACC appendages for the Alinghi team for the 2007 America's Cup. Running on a cluster of 32 AMD Opteron processors, each simulation required approximately 30 hours to reach convergence for a grid of up to 20 million points. These very large computational grids were necessary due to the requirements imposed by the transition and turbulence models that were used. Similarly, Korpus (2007) described a RANS based hydrodynamic analysis of ACC yachts that took 8 days to analyse 147 simulation cases using 16 Itanium-2 processors for a grid of 6-8 million points.

Clearly, a global optimisation method requiring hundreds or thousands of RANS iterations would be impractical. Although accelerating the performance by using computing hardware with hundreds or thousands of processors is theoretically possible, the cost for such a system is currently prohibitive.

One innovative approach to efficient optimisation using high quality solvers was taken by Janson and Larsson (1996) with the SHIPFLOW code. This program divides a ship hull into three zones, shown in Figure 4, applying a different computational approach to each zone:

- Zone 1 uses a free-surface potential flow method for the whole hull and part of the free surface.
- Zone 2 encompasses the thin layer close to the hull where a boundary layer method of the momentum integral type is used.
- Zone 3 covers the aft portion of the hull and extends about one half of the length of the ship downstream, as well as extending radially for about the same distance. This zone uses a time averaged Navier Stokes solution to calculate the effects of a thick, turbulent boundary layer where boundary layer methods would fail.

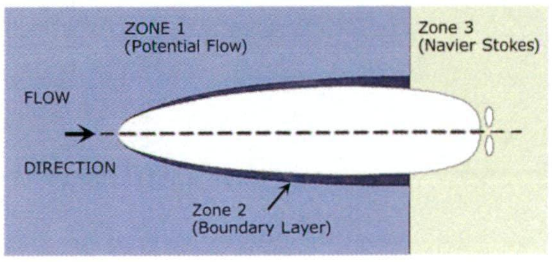


Figure 4. SHIPFLOW zones, from Janson and Larsson (1996)

The result is a program that retains sufficient accuracy for overall resistance calculations, yet avoids spending unnecessary time on calculations that are not required for particular portions of the hull.

Janson and Larsson used a gradient based method for their optimisation work, the Method of Moving Asymptotes (MMA) that worked well for that particular application. The method used to model and constrain the hull shape did not appear to fare so well. The approach taken moved surface points along a vector, typically the surface normal. However, no constraints were applied for flat of bottom, flat of side, or longitudinal and transverse fairness, all essential requirements in the design of a merchant ship hull.

Peri et al. (2001) investigated the use of variable complexity modelling, where models and solvers of different levels of complexity are used at different stages of the optimisation in order to improve performance. For example, during the early stages of the optimisation process a 2D strip theory potential flow solver is used with a simple geometrical model. As the optimisation proceeds, this model is defined in greater detail and analysed, firstly using a full 3D potential flow solver, followed by a RANS code utilising a multigrid, multiblock, finite-volume solver.

Campana et al. (2006) described several innovations designed to improve the performance of what he refers to as Simulation Based Design (SBD) and set up two different optimisation approaches to compare the alternatives. Campana makes special comment on the intrinsic parallelism of Genetic Algorithms and their suitability for implementation on multiple processors. This is seen as a way of significantly accelerating the performance of RANS codes used in conjunction with global optimisation methods.

In Campana's first optimisation approach (labelled SBD-A), a Genetic Algorithm having a very narrow available range for its independent variables is used. This narrow-band GA was run in parallel on a 64 processor cluster.

Campana's second optimisation approach (SBD-B) consisted of a variable fidelity solver that used a multigrid method to trade off result accuracy against execution time. Solutions in the early part of the optimisation process used a low resolution grid, with the grid resolution increased as the solution was judged by a heuristic procedure to be near to the optimal solution.

Note that Campana's variable fidelity method differs from Peri's variable complexity method; variable fidelity modelling uses the same solver but diffent grid resolutions, whereas variable complexity modelling also uses solvers having different levels of complexity in the fluid-flow equations they solve.

Validation was performed, not only comparing original and optimised designs using RANS analysis, but by also constructing tank models and measuring their resistance in a towing tank. Camapana obtained excellent correlation between the gains shown by the RANS optimised design over its parent model and a corresponding comparison of tank test model results. Seakeeping performance was also specified as an inequality constraint, with the result that both heave and pitch RAOs showed small improvements.

The two methods adopted by Campana achieved similar results in similar periods. Both methods took approximately four days to run. However, SBD-A used 64 processors, while SBD-B used a single processor. These results suggest that there is potential for combining the use of parallel genetic algorithms with variable fidelity modelling to improve performance.

METAMODELS

In cases where direct numerical analysis is prohibitively slow, a statistical approximation, or metamodel, may be created in order to provide rapid evaluation of an otherwise expensive objective function. Metamodels typically differ from older regression models such as those proposed by Holtrop (1978) Oortmerssen (1971) and Savitsky (1964) in several regards.

Metamodels are usually based on deterministic analytical methods and, as a result, do not have the high levels of random noise often encountered with tank test data.

Although metamodels may use traditional polynomial regression, they are more likely to use methods more suited to automated data analysis and fitting, such as Response Surface Methods (RSM), (Minami and Hinatsu 2002), Kriging, Neural Networks (NN) or Radial Basis Functions (RBF), (Peri and Campana 2005b; Peri and Campana 2005d). These methods are better able to handle non-linear datasets of high dimensionality and offer the possibility of creating a global approximation model that is valid for the whole design space.

Neural networks and radial basis function networks have attracted a great deal of interest for metamodelling. Within the domain of naval architecture, neural networks have been used to create metamodels for ship resistance (Duvigneau and Visonneau 2002), propeller design (Neocleous and Schizas 1995; Mesbahi and Atlar 2000), wetted surface area (Koushan 2003), hull form design (Mesbahi and Atlar 2000; Islam et al. 2001), catamaran resistance ((Couser et al. 2004; Mason et al. 2005)), ship stability (Alkan et al. 2004), roll stabilisation (Birmingham et al. 2002), manoeuvring (Hess et al. 2004; Seif and Jahanbakhsh 2004), ship motions identification and engine control (Mesbahi and Atlar 2000).

Several researchers have used metamodels to improve the performance of optimisation or analysis methods that suffer from long calculation times. Peri and Campana (2005c) used a variable fidelity approach in which a lower fidelity metamodel was used in the early stages of the optimisation, derived from a small number of points calculated by a RANS solver.

As the optimisation progressed, the number of points used to calculate the metamodel was increased, particularly in the regions of interest highlighted by the optimiser. In the final stages of the optimisation a switch was made to a local optimisation method that bypassed the metamodel and accessed the RANS code directly.

This hybrid variable-fidelity approach, combining the use of a metamodel in the early stage, with direct analysis in the final stage, provided an environment where both fast global search and accurate local refinement could take place in an efficient manner and shows great promise for future development.

1.4.3 OPTIMISATION METHODS

Prior to 1997, the most widely used optimisation methods for naval architecture were Sequential Unconstrained Minimisation Technique (SUMT), Sequential Quadratic Programming (SQP), Nelder-Mead Simplex and Hooke-Jeeves direct search, (Nowacki et al. 1970; Parsons 1975; Kupras 1976; Hsiung and Shenyan 1985; Walden et al. 1985; Keane et al. 1991).

SUMT and SQP are classified as gradient based methods, as they use derivatives of the solution surface to determine the direction of each step in the optimisation process. The Nelder-Mead and Hooke-Jeeves methods are considered pattern search or derivative free methods, as they use direct comparisons between multiple points to determine the direction taken for each successive step in the optimisation process.

Each of these methods can be considered a local optimisation method in that they may become trapped in false optima if the solution space is multi-modal. The widespread adoption of these methods by ship-design optimisation researchers implies that they considered their optimisation problems to be unimodal; that is, having only one potential optimum. However, later experience with global optimisation methods has shown that this is not a valid assumption.

During this early period, multiple objectives were typically handled using weighted sums of the individual objectives, leaving the outcome of the optimisation dependent on the weights chosen. Alternatively, a measure of merit could be used that was based on the sum of the squares of the differences between the preferred values for each objective and the value obtained for the current design in the optimisation process, as in Sarioz et al. (1992).

STOCHASTIC OPTIMISATION METHODS

During the early 1990s, significant experimentation with different optimisation methods took place in the field of aerodynamics. One research direction was the investigation of stochastic, global optimisation methods, such as Genetic Algorithms (GA), (Goldberg 1989) and Simulated Annealing (SA), (Kirkpatrick et al. 1983), as an alternative to the gradient based or pattern search based local optimisation methods that had previously been favoured. The success of these methods when applied to complex aerodynamic problems, (Quagliarella and DellaCioppa 1994; Yamamoto and Inoue 1995a; Galan et al. 1996), suggested that they might be equally applicable to optimisation problems in hydrodynamics. Investigations into their use in naval architecture soon followed.

Sahoo (1997) compared two optimisation methods, Sequential Unconstrained Minimisation Technique (SUMT) and Simulated Annealing (SA), using objective functions derived from two statistical regression methods for planing hulls, Savitsky (1964) and Keuning et al. (1993). Sahoo found that the SUMT method did not give good results, being sensitive to initial parameter values, whereas the SA method was found to be slow but robust. This was an early indication that stochastic methods might be superior to deterministic methods when applied to non-linear functions such as ship resistance.

The ability of stochastic global optimisation methods to solve problems with non-linear and multi-modal solution spaces makes them particularly appropriate to the optimisation of multihull designs. The interaction of the wave patterns between hulls contributes in a non-linear manner to the resistance of multihulls, making their resistance more difficult to predict.

Recognising this, Hearn and Wright (1997) used a GA to optimise a catamaran, apparently the first instance of a GA being used for the optimisation of marine hull shapes. Hull shape variation was achieved using a Lackenby transformation (Lackenby 1950) of an existing hull shape. Fifteen different objective functions were examined being various combinations of frictional resistance, wave making resistance, added resistance, relative bow motion and vertical bow acceleration. These values were calculated using the Salvesen, Tuck and Faltinsen (1970) strip theory method for sea-keeping analysis, a Michell (1898) thin-ship theory based wave-making resistance algorithm and a fine-form added resistance formulation.

The research performed by Day and Doctors (1997; 2000) shared many characteristics with the work of Hearn and Wright. The target of the optimisation was a catamaran, with both resistance and seakeeping measures included in the objective function. A GA was also used to perform the optimisation. Hull geometry was handled by a wireframe representation that modelled below-water hull shape only, controlled by fifteen geometric variables.

Both of these research projects resulted in workable optimisation systems using Genetic Algorithms, although Day and Doctors were concerned at the limited range of applicability of the analysis due to their reliance on thin-ship theory. Both projects were limited by the use of simplistic methods for representing and varying hull geometry, as well as by the use of simple weighted sums of multiple objectives.

Yasukawa (2000) compared a GA to the Hooke-Jeeves method for merchant ship hull shapes, finding that the GA gave superior results. Similarly, Hirata (2004) compared optimisations of merchant ships performed with a GA to optimisations performed with SQP. Results showed that the SQP method consistently converged to local optima rather than the global optimum, while the GA reliably found a design that was close to optimal.

Hirata's results are illustrated in Figure 5, showing the large number of points evaluated by the GA, together with the sub-optimal path followed by the SQP method.

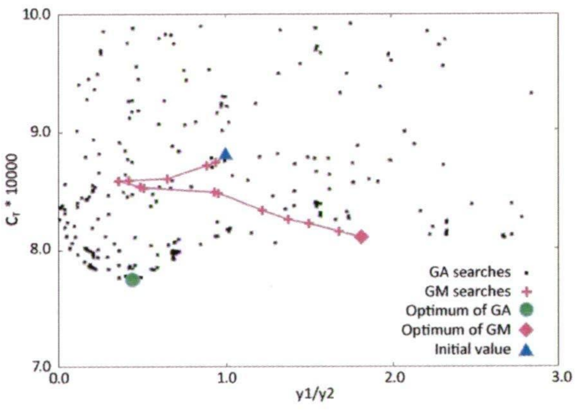


Figure 5. Superiority of GA versus SQP, from Hirata (2004)

In this case the variables y1 and y2 relate to transverse waterline beam at particular longitudinal locations, and are measures of the shape of the ship's waterplane and L_{CF}. It can be seen that the SQP method converges on a local optimum that has a higher coefficient of resistance relative to the best design found by the GA.

In recent years, other global optimisation methods have been investigated (Pinto et al. 2004; Pinto and Campana 2005; Pinto et al. 2007) with positive results. These methods included the Multistart Gradient Method, the Diagonal Rectangular Algorithm for Global Optimisation (DRAGO), Particle Swarm Optimisation (PSO) and Multi-Objective Deterministic Particle Swarm Optimization algorithm (MODPSO).

The Particle Swarm Algorithms (PSO and MODPSO) performed well, particularly for problems involving multiple objectives, with significantly improved results compared to the local optimisation methods examined (Nelder-Mead Simplex and Hooke-Jeeves pattern search), although at the cost of increased calculation times.

MULTI-OBJECTIVE OPTIMISATION

Although there have been many efforts to optimise multiple objectives using weighted sums of output values, true multi-objective optimisation of marine vessels was first described by Sen and Todd (1997) and Poloni and Pedirodav (1997). In both cases, the optimisation procedure was a Multi-Objective Genetic Algorithm, or MOGA, provided by the modeFRONTIER optimisation shell. Rather than the output from the optimisation being a single optimal design, the MOGA located a collection of designs that occupied a Pareto optimal front.

The Pareto front is made up of those designs that are better than all other designs in at least one objective measure; that is, they are non-dominated designs. For example, in a problem where larger values of functions y_1 and y_2 are better, as illustrated in Figure 6, from (Zitzler 2002), the non-dominated points on the Pareto front are those that are better than all other points in either y_1 or y_2.

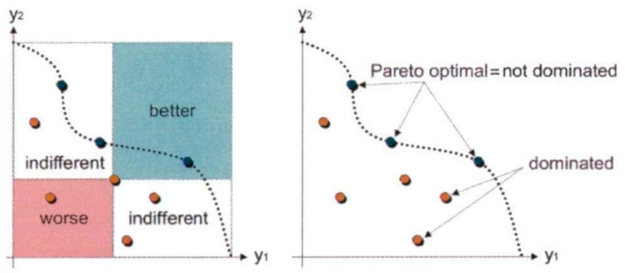


Figure 6. Illustration of Pareto optimality, from Zitzler (2002)

Multi-objective methods are necessary in situations where there it is not possible to frame the optimisation problem as having a single objective with multiple constraints. However, if it is possible to reduce the problem to having a single objective, this is to be preferred, as using multiple objectives means that a subjective choice remains for the designer at the conclusion of the optimisation process.

MULTIDISCIPLINARY OPTIMISATION

Multidisciplinary optimisation differs from multi-objective optimisation in that the key objectives do not share common features. For example, the optimisation of the effects of hull shape variations on resistance and seakeeping uses multiple objectives that are hydrodynamic in nature. These can potentially be calculated using a single piece of software. On the other hand, the simultaneous optimisation of the resistance, structural strength and cost of a vessel involves calculations drawn from three unrelated disciplines.

Neu et al. (2000a) proposed a multidisciplinary approach to the optimisation of commercial ships. The objective function for this optimisation was a measure of required freight rate, determined by combining conventional measures of performance with cargo capacity and ship economics.

Poloni et al.(1999) used a MOGA to perform a multidisciplinary optimisation the keel of a sailing yacht. This involved optimising lift and drag while constraining keel depth, mass, centre of gravity and structural strength. These factors present a complex trade off between a thinner foil section with smaller volume having less resistance, versus a thicker foil having greater strength and righting moment.

This particular problem is directly applicable to ACC yacht design as ACC keels have significant problems with lateral deflection due to their small section coupled with the 18 tonne lead bulb attached to the tip of the keel fin. The trade off between performance loss due to deflection of the keel under load, versus the increase in hydrodynamic drag due to increased keel thickness, is an optimisation problem that was addressed for ACC yachts by Campana et al. (2007).

ROBUST DESIGN OPTIMISATION

Whitfield (1998; 1999) focussed on the optimisation of catamaran configurations using a genetic algorithm. In this case, the measure of merit was solely concerned with ship motions. However, the major contribution of this work is the introduction of the concept of robustness to the field of marine design. Whitfield defines robustness as –

> *"A product's robustness is a measure of the variation in its utility experienced in a typical application. That is to say, the lower the sensitivity or variation in utility, the greater the robustness of the design." (Whitfield et al. 1998, p.373)*

The problem of robustness is often overlooked, with the result that the optimised designs work well only in a given environment and perform poorly when the conditions fluctuate slightly. Such a design is likely to be less desirable than one whose peak performance is lower, yet performs well in a variety of conditions.

This situation can be observed in many of the attempts to optimise ship hull lines for a single target Froude number (*Fr*), such as Hsiung (1981), Janson and Larrson (1996), Dejhalla et al. (2001) and Chen (2004). If fairness and convexity are not constrained, the result is usually a hull shape with longitudinal undulations, as shown in Figure 7.

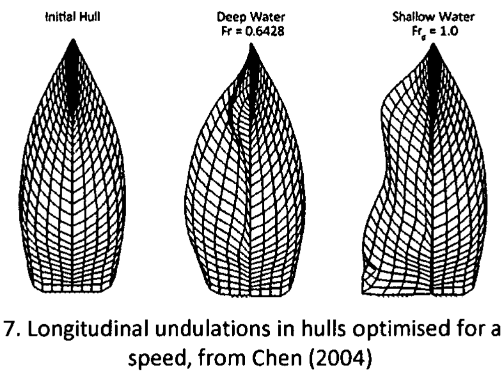


Figure 7. Longitudinal undulations in hulls optimised for a single speed, from Chen (2004)

Such a shape may be ideal for one particular speed, but performs poorly if the speed is changed only slightly. Percival et al. (2001) optimised the hull shape of a Wigley hull at three different Froude numbers, resulting in three very different hull shapes, as shown in Figure 8.

Each of these hulls had significant concavities in the longitudinal direction, as well as distinct bulbs at the bow and stern, which were quite extreme for the hull optimised for *Fr* = 0.408. However, when the parent hull was optimised for the three Froude numbers simultaneously, the resulting hull shape had long fine ends with no trace of bulbous bow or stern, or longitudinal undulations.

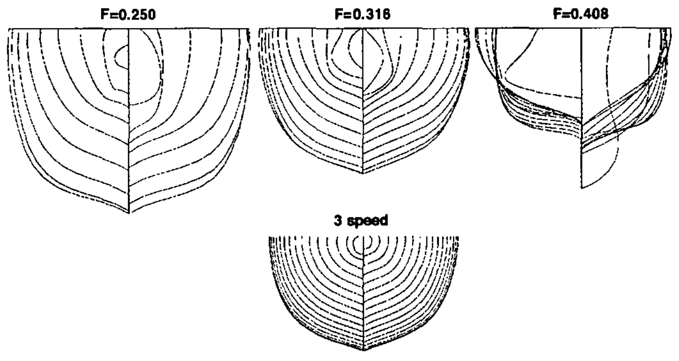


Figure 8. Optimisation of a Wigley hull at different Froude numbers, from Percival et al. (2001)

Figure 9 shows curves of total drag coefficient C_T for the parent Wigley hull, the three hulls optimised for single *Fr* values, and the hull optimised for all three *Fr* values. The graph shows how poorly the three hulls optimised for a single *Fr* perform in off-target conditions, with resistance increasing rapidly as *Fr* deviates from the target speed. Conversely, the hull optimised for a combination of three *Fr* values has resistance values only slightly greater than the three single-speed optimised hulls at each of their target speeds.

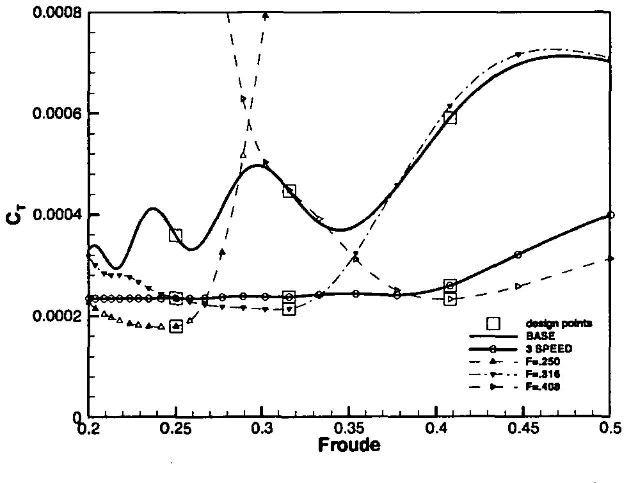


Figure 9. Resistance curves for hulls optimised by Percival et al. (2001)

The concept of robustness is directly applicable to sailing yacht design. A boat that performs exceptionally on one point of sailing, in one set of weather conditions, is generally not as desirable as a boat that performs well in a variety of wind velocities, sea conditions and points of sail. A measure of merit formulated for the optimisation of a racing yacht needs to take into account a wide variety of conditions, rather than focussing on a narrow range of heel, yaw and velocity values.

1.4.4 HULL SHAPE VARIATION METHODS

In order for a design optimisation process to function, a method for parametrically varying individual designs is required. This is not a trivial problem, as not only is it necessary to vary hull shape based on changes to parameters such as C_P and L_{CB}, it is also necessary to ensure that constraints, such as constant displacement and hull fairness, are not violated.

PARAMETRIC TRANSFORMATION

Initial attempts at hull design optimisation such as Hearn and Wright (1997) used the method described by Lackenby (1950) to vary hull shape. Lackenby's method performs a longitudinal shift of hull sections in order to modify parallel mid-body, C_P and L_{CB}.

Although Lackenby's method is good at retaining the fairness and features of the parent hull, the basic method allows displacement to vary in an uncontrolled manner. This must be compensated for by scaling the hull shape along its principal directions. A more significant problem is that Lackenby's method does not provide any mechanism for varying the cross sectional shape of the vessel.

Markov and Suzuki (2001) applied an analogous approach to a the control vertices of a B-spline surface representing commercial ship hull forms. Hull shapes were optimised using the Davidson-Fletcher-Powell (DFP) method with resistance calculated using a higher-order Rankine source panel method.

Although other methods have been proposed to permit more flexible variation of a parent hull to match desired longitudinal and cross sectional parameters, such as those by Söding and Rabien (1977) and McNaull (1980), these approaches do not appear to have been utilised in any optimisation research.

HULL BLENDING

One alternative shape variation method was suggested by Chen and Parent (1989), who explored the possibility of using a weighted average of two or more parent models. Although this research was aimed at general industrial design, the approach is equally applicable to naval architecture.

A similar approach was implemented by Neu (2000a; 2000b), who used a barycentric blend of a number of different hull shapes within an optimisation procedure. This can be represented as:

$$\text{Resultant Ship Hull} = \sum C_n \text{Basis Hull}_n$$

where:

$$C_n = \text{Blending Coefficient for Basis Hull}_n$$

$$\sum C_n = 1$$

and

$$0 \leq C_n \leq 1,$$

$$n = 1, 2, \ldots, \mathrm{N}$$

Figure 10 illustrates a simple midship section formed using this method by blending two basis hulls.

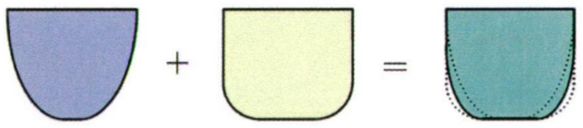


Figure 10. Barycentric blending of hull shapes, from Neu et al. (2000)

This approach is particularly easy to implement if the parent hull representations are NURBS surface models, made up of control point networks having the same number of control points. In this case, a weighted sum of each of the control point co-ordinates from each parent hull surface is used. This method has been used by ACC designers since 1990, (D. Peterson 1992, pers. comm.) for the creation of systematic series of hull models.

The disadvantage of the blending technique is that individual design parameters are not independent, as all parameters are fully correlated with the blending coefficient. This results in large portions of the design space being inaccessible to the optimisation procedure.

PARAMETRIC MODELLING

Parametric modelling, the generation of a ship hull from scratch based on a set of key parameter values, was pioneered by Nowacki (1970; 1977; 1983; 1993; 2005) with contributions from Walden et al. (1985), Peacock (1996), Birmingham and Smith (1998), and Islam (2001).

However, it was not until the work of Stefan Harries that parametric modelling of ship hulls became a viable alternative to the parametric transformation of parent models for marine hull design and optimisation.

Harries, in collaboration with Claus Abt and others, has been possibly the most prolific investigator on the subject of hydrodynamic optimisation of marine vessels, (Harries and Abt 1999a; Birk and Harries 2000; Abt et al. 2001a; Harries et al. 2001b; Valdenazzi et al. 2002; Abt et al. 2003; Harries et al. 2003b; Heimann and Harries 2003; Valdenazzi et al. 2003; Abt et al. 2004).

The common component in these works is the use of the FRIENDSHIP-Modeller, later known as the FRIENDSHIP-Framework, to parametrically define and vary hull geometry. Developed by Harries and Abt, the FRIENDSHIP tools provide a flexible environment for hull shape creation and manipulation, while retaining high quality, fair hull surfaces.

The work of Harries and Abt has covered a wide variety of vessel types, including semi-submersible offshore rigs, military vessels, fast ferries, and containerships. Optimisation methods used have included Hooke-Jeeves direct search, the tangent search method, Newton-Raphson solvers, Multi-Objective Genetic Algorithms (MOGA) and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method.

This extensive body of work demonstrates the utility of the parametric modelling approach for a wide range of hull types and analysis methods. However, the fact that this method does not use a known parent model as a starting point, nor is it able to exactly reproduce such a parent model if required, is a potential stumbling block in the field of ACC yacht design. ACC designers wish to have specific features and preferences incorporated into a hull shape and a system that is not able to cater to these requirements is at a significant disadvantage.

FREE-FORM DEFORMATIONS

Sederberg and Parry (1986) proposed a method for deforming geometric objects by embedding them within a flexible volume defined by trivariate splines. This enclosing volume could then be deformed by moving its defining control points, resulting in an equivalent deformation to the embedded object.

FFDs have been used successfully for aerodynamic optimisation (Desideri et al. 2006; Duvigneau 2007) as they permit smooth deformation of complex geometries using a small number of variables.

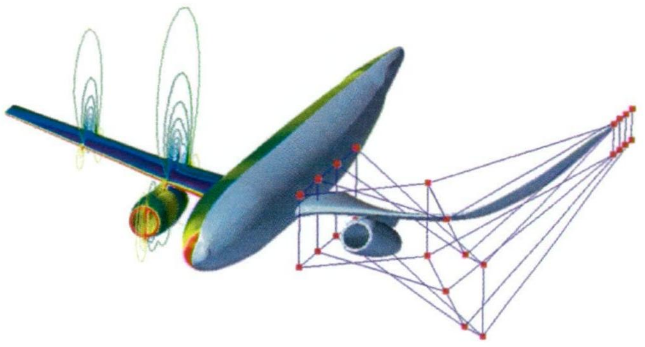


Figure 11. An FFD used for deformation of an aircraft wing (© INRIA Opale Project-Team 2007, reproduced with permission)

Peri and Campana (2005a) examined the use of the Free Form Deformation (FFD) in ship design optimisation. In this case, FFDs were used to enclose and deform specific blocks of the ship, namely the bulbous bow, forward waterlines and stern section, in an effort to reduce the resonant whipping behaviour of the ship. The FFD used to deform the bulbous bow of the ship is shown in Figure 12, with control points subject to movement shown with black arrows.

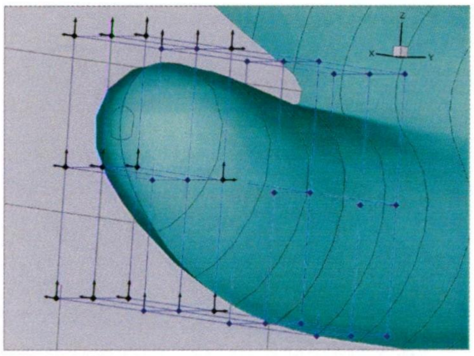


Figure 12. An FFD lattice enclosing the bulbous bow of a cruise ship, from Peri and Campana (2005a)

Duvigneau and Visonneau (2001; 2002) investigated the use of FFDs for the deformation of ship hull forms for optimisation. Duvigneau showed that FFDs provided a flexible way of varying design geometry and resulted in significant time savings when the FFD was used to directly deform the computational grid used for CFD analysis, rather than recreating the grid at each iteration.

The optimisation systems incorporating FFDs that were used by these researchers were implemented successfully. However, the FFD method is not trivial to implement and there appears to have been little further work utilising FFDs within the field of marine design optimisation.

FFDs AND EVOLUTION

One inspiration for the use of the FFD method in design optimisation is drawn from parallels with biological evolution. D'arcy Thompson's "On Growth and Form", (Thompson 1917), detailed the variations that occurred during the evolution of different species. Thompson showed that many of the high-level changes that occurred in evolution consist of both linear and non-linear transformations of previous forms (Figure 13).

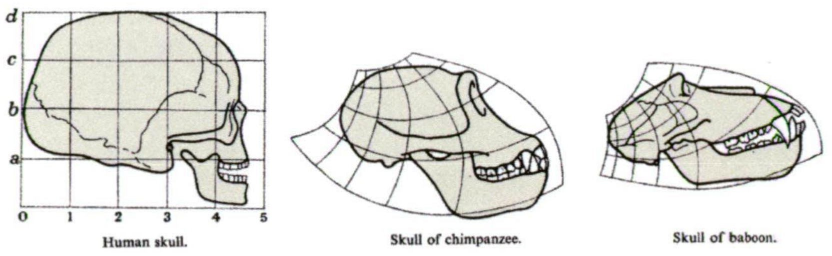


Figure 13. Non-linear deformation of primate skulls, from Thompson (1917)

Thompson's work also illustrated some of the different types of deformations that may take place. Figure 14 illustrates various linear, radial and non-linear deformations that Thompson suggested could describe some of the variation that occurs between different species of fish.

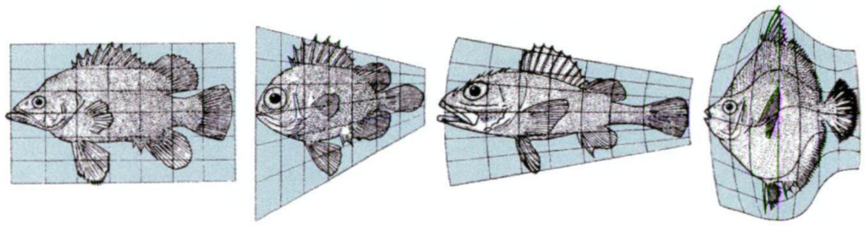


Figure 14. Variation of fish forms using linear and non-linear transformations, from Thompson (1917)

Each of these mappings is directly analogous to transformations required to deform a parent hull shape to match a set of desired hull parameters.

Both natural evolution and evolutionary optimisation systems benefit from methods that can achieve significant and effective variation, yet can be encoded in a concise manner. Biological evolution is able to vary small details as well as apply global variations, encoding each efficiently using a small number of parameters. Such a hierarchy of transformations may also be of value when applied to the variation of geometry during an optimisation task, reducing the number of dimensions in the search space and allowing the optimisation to proceed more efficiently.

1.4.5 SAILING-YACHT DESIGN OPTIMISATION

Although there has been a great deal of research performed on the calculation of yacht performance, there is a relatively small body of literature specifically on the subject of computer based yacht design optimisation.

An early use of genetic algorithms in sailing yacht design was described by Day (1993). In this work, the sail plan of a yacht was optimised using an objective function based on a non-linear vortex lattice model. Once the GA had converged on a solution, a further optimisation was performed using a search pattern based algorithm, Powell's direction set method. In all cases, the classical optimisation method was unable to improve on the solution found by the GA. This result shows that genetic algorithms are capable of converging on solutions of equivalent quality to traditional optimisation methods when faced with real-world problems. However, Day commented that performance of the GA was a significant issue.

Poloni et al. (1999) used a Multi-Objective Genetic Algorithm (MOGA) combined with a neural network metamodel to optimise the lift and drag of the keel of a sailing yacht while constraining mass, centre of gravity and structural parameters. Limitations of this work were the use of a fixed heel and yaw angle, rather than a simulation of a variety of sailing conditions and keel loadings.

Other than the yacht design optimisation work performed by Harries, Abt and Hochkirch (Harries and Abt 1999b; Abt et al. 2001b; Harries et al. 2001a; Hochkirch et al. 2002) there has been little other yacht design optimisation work performed outside of the America's Cup arena.

Optimisation work that has been performed specifically for the design and refinement of Americas Cup yacht design (Harries et al. 2001a; Philpott 2003; Philpott et al. 2004; Baik and Gonella 2005; Peri and Mandolesi 2005) is discussed in detail in Chapter 2.

1.5 DESIGN OPTIMISATION SUMMARY

A review of the literature concerning the automated optimisation of marine vessels has revealed several clear trends over the past thirty years, illustrated in Table 2.

Table 2. Summary of Developments in Yacht Design Optimisation

	Hull Variation Method	Hull Geometry	Measure of Merit	Optimisation Method	Analysis Method
↓ Time	Variation of parameter values	Parameter values only	Single objective	Gradient based, local optimisation	Polynomial regression of tank test data
	Lackenby	Wireframe	Multiple objectives	Derivative free pattern searches	Thin ship/ slender ship theory
	Parent Blending, Parametric Modelling	Polyhedral	Multi-disciplinary optimisation	Simulated Annealing	Panel methods, potential flow codes
	Free Form Deformation	NURBS	Robust design optimisation	Stochastic global optimisation	RANS codes

Each of these developments has involved greater algorithmic complexity and a larger number of calculations compared to its predecessors. This has been offset to some extent by hardware developments, both the inexorable rise in CPU performance in accordance with Moore's law, (Moore 1965), which predicts a doubling of CPU transistor count every 24 months, as well as the increasing use of multi-core chips and parallel clusters for computationally intensive applications.

Despite the exponential increase in computing power since the introduction of the transistor, this has not been as great as the rate of increase in the number of calculations demanded by modern analysis and optimisation methods. This imbalance between processing speeds and processing needs has resulted in the exploration of approaches that can streamline the optimisation process and reduce the total computational cost. One relatively recent addition to the field of optimisation is the adoption of a metamodel to act as an inexpensive surrogate for a computationally expensive objective function. Although not applicable to all situations, pre-calculated metamodels may dramatically improve the performance of some ship-design optimisation problems.

The results of these developments have been encouraging, with a genuine improvement in resistance and seakeeping performance for many vessels. Importantly, automated optimisation methods have also been responsible for the discovery of novel and non-intuitive hull shapes, such as those reported by Harries et al. (2006). Harries demonstrated that automated optimisation methods may occasionally be able to uncover superior combinations of design parameters that might have otherwise been overlooked or dismissed by a human designer.

1.6 OUTLINE OF THESIS

One of the key motivations for the work described in this thesis is the conflict between the need for accurate calculations and the desire to get results within an acceptably short time. In order for the results of an optimisation to have anything but academic interest, it must be able to make design recommendations that result in real improvements when verified with independent CFD analysis, tank testing and full sized trials. Yet these results must also be produced within the restricted budget and limited period of time available during the design cycle of an America's Cup campaign.

In order to create a workable optimisation system, several key problems must be solved. These include -

- The selection of a measure of merit that is appropriate to the problem of determining the best yacht to compete in a match racing series, against an opponent whose design parameters are unknown. This question is addressed in Chapter 2, Measure of Merit.
- The creation of parametric transformation methods that allow ACC hulls to be varied to match specific parameters, yet satisfy the constraints imposed by the ACC rule (ACC 2003). These areas are covered in Chapter 3, Constraints, and Chapter 4, Data Approximation.
- The adoption of techniques to enable an otherwise expensive set of analysis functions, such as CFD analysis of the flow around a yacht hull, to be rapidly evaluated. This is necessary in order for the optimisation environment to have acceptable performance. This subject is discussed in Chapter 4, Data Approximation.
- The selection of CFD method for determining the lift and drag of a hull shape at various heel angles, yaw angles and velocities is discussed in Chapter 4, Data Approximation. The selection or creation of Velocity Performance Prediction (VPP) and Race Modelling Program (RMP) software for determining and comparing the sailing performance of different designs is covered in Chapter 5, Performance Evaluation.
- The selection of an optimisation method capable of locating an optimal design, in what is likely to be a non-linear, multi-modal solution space. The method should be suitable for use with the chosen measure of merit. The choice of optimisation method is discussed in Chapter 6, Optimisation Algorithm.
- Integration of all components into a robust system that produces reliable and repeatable results, see Chapter 7, Implementation, and Chapter 8, Verification and Validation.
- Finally, results obtained and their implications for future work are detailed in Chapter 9, Results and Discussion, and Chapter 10, Conclusions.

1.7 CONTRIBUTION

Although the research described by this thesis focuses on a narrow range of racing-yacht design parameters, the approach described is broadly applicable to a wide range of marine craft. It is envisaged that this research will contribute not just to the design of ACC yachts, but to the field of naval architecture in general.

A significant amount of recent research exists involving the optimisation of ships and boats. However, much of this is focussed on the use of direct CFD solvers having prohibitively long run times. The approach taken with this research, combining the use of a high-fidelity meta-model with a stochastic global optimisation method utilising a probabilistic measure of merit, has the potential to allow optimisations, which would otherwise be impractically slow, to be completed within a reasonable time on conventional personal computers.

This research has identified problems and created innovative solutions in the several areas. Contributions to the field include:

- A survey of the state of the art for the design optimisation of marine vessels.
- The design and implementation of a geometrical parametric transformation function based on a hierarchical free-form deformation procedure.
- The use of neural network metamodels for hydrodynamic data for ACC yachts;
- The identification of an appropriate measure of merit for an ACC design optimisation process, and the creation of an appropriate race and tournament modelling system to support this.
- The modelling of the random variation that occurs in the yacht-racing environment.
- Recognition of the dynamic fitness landscape that results from the adoption of a competitive fitness function, and the identification of its effects on the choice of optimisation method.
- Recognition of the potential for a co-evolutionary arms race to occur, due to the use of the competitive fitness function, and the implementation of steps to limit any adverse effects from such an arms race.
- The treatment of the ACC design optimisation problem as a stochastic, multiplayer game, and recognition that this system may have no Nash equilibrium, due to penalties incurred at each turning mark.

2 · MEASURE OF MERIT

The first step in setting up an optimisation process is to precisely determine the objective of the optimisation. This is often not a trivial exercise, with multiple candidates for the measure of merit presenting themselves. In many cases, what initially appear to be objectives can be treated as constraints; in other cases, multiple objectives remain, and these require the adoption of multi-objective optimisation methods.

Many attempts at yacht design optimisation have viewed the task as a multi-objective problem, or at least a single objective problem with a large number of constraints. Although this may be a reasonable assumption for cruising yacht design, where there are conflicting requirements for speed, comfort, stability, seaworthiness and cost, ACC optimisation is based solely around the wish to be the winner of a specific set of races, held at a particular location during a specific month and year. As a result, the design optimisation of ACC yachts can be regarded as a problem with a single objective, although the formulation of the objective function is a formidable task.

2.1 MEASURE OF MERIT OR OBJECTIVE FUNCTION?

Although the terms "measure of merit" and "objective function" are generally regarded as synonymous in optimisation research, this thesis distinguishes between the two terms.

The reason for making a distinction is simple. In most optimisation procedures, the measure of merit used to discriminate between designs is a deterministically calculated value; that is, given the same set of input parameters, the output value will be the same each time it is calculated and as such is truly an objective measure.

For some optimisation procedures, this is not the case. The worth of a particular design may be measured, not against some objective criteria, but relative to the performance of another design. Depending on the performance of the design being used for comparison, the current design point in the optimisation may be ranked comparatively high or low. Consequently, the measure of merit for a specific set of design parameters may differ during the optimisation process.

Within this thesis, the term "objective function" is taken to mean an algorithm used to calculate the performance characteristics for an individual vessel. These calculations are purely deterministic and therefore can be considered an objective measure of the performance of a yacht. The "measure of merit", on the other hand, is the final value, derived from the output of the objective function, used by the optimisation procedure to rank designs.

There are several potential candidates for adoption as a measure of merit for an ACC yacht design optimisation. The form taken by this measure of merit influences the choices made at every stage of the optimisation process. It is therefore important to closely examine the benefits and drawbacks of each possible alternative.

2.2 UPRIGHT RESISTANCE

Many researchers have attempted to minimise the resistance of an upright hull, with or without appendages, at one or more speeds. While this approach has been used primarily for the optimisation of ships, there have also been examples, such as Harries et al. (2001a) and Maisonneuve (2003), where this approach has been applied to ACC yachts.

In both cases, unappended hulls operating at zero heel and yaw for a single Froude number were examined. However, for yachts that operate over a wide range of velocities, heel and yaw angles, the upright case is not indicative of the overall performance of the yacht, and this approach is of little value.

2.3 HEELED LIFT AND DRAG

A more detailed form of analysis encompasses the optimisation of both lift and drag at multiple velocities, heel and yaw angles. This approach has mainly been used by authors working directly with CFD solvers such as Peri and Mandolesi (2005).

Although this approach is an improvement over the upright case, it suffers from not incorporating a measure of stability, and therefore sail carrying ability, in the objective function. For example, a narrower hull may have less resistance, but may also have less stability, giving it less sail carrying ability. There is no way of knowing, without more-detailed analysis, whether the yacht would be faster or slower overall.

Baik and Gonella (2005) investigated the performance of ACC yachts using an optimisation of the resistance of unappended canoe bodies at four different angles of heel. Hulls were analysed with zero yaw and no account was taken of righting moment. The results of these optimisation runs illustrate the pitfalls of using simplistic measures of merit (Figure 15).

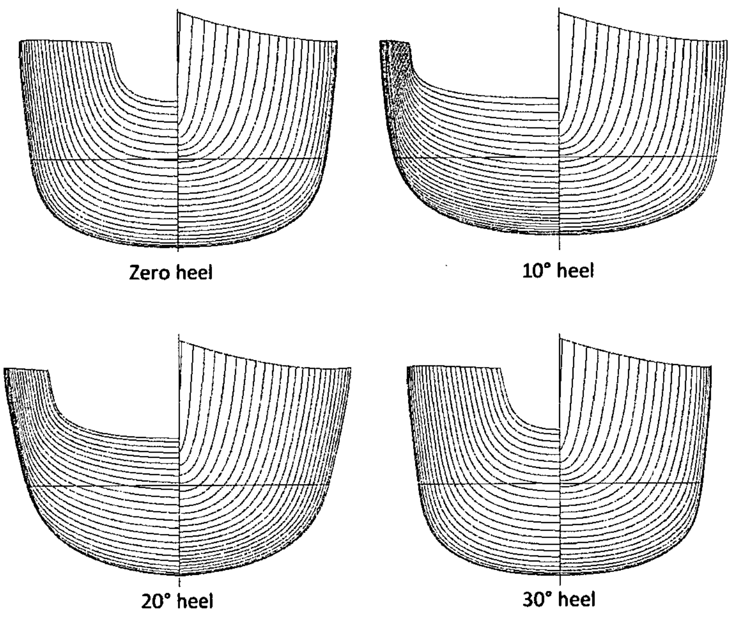


Figure 15. Optimal hull shapes for different heel angles, from Baik and Gonella (2005)

Each successive optimisation run used a design objective that differed by only 10° of heel angle from the previous run, yet the optimised hulls are quite dissimilar, with obvious wide variation in B_{WL}, C_M, T_C, flare and beam at transom.

Despite their optimisation runs producing many different hull shapes, Baik and Gonella did not provide any methodology for selecting which of the various designs should be recommended for a particular America's Cup match.

2.4 VELOCITY PREDICTION PROGRAMS

For stability to be properly incorporated, it is necessary to combine lift and drag data for the hull with information about the weight, centre of gravity and sail plan of the yacht, in order to calculate actual sailing performance in different wind velocities and different points of sail. Software that calculates yacht performance in this way is referred to as a Velocity Prediction Program, or VPP. Although a VPP can calculate a yacht's speed, heel angle and leeway for a given wind velocity and direction, overall performance is often presented graphically in the form of a set of polar performance curves, as shown in Figure 16.

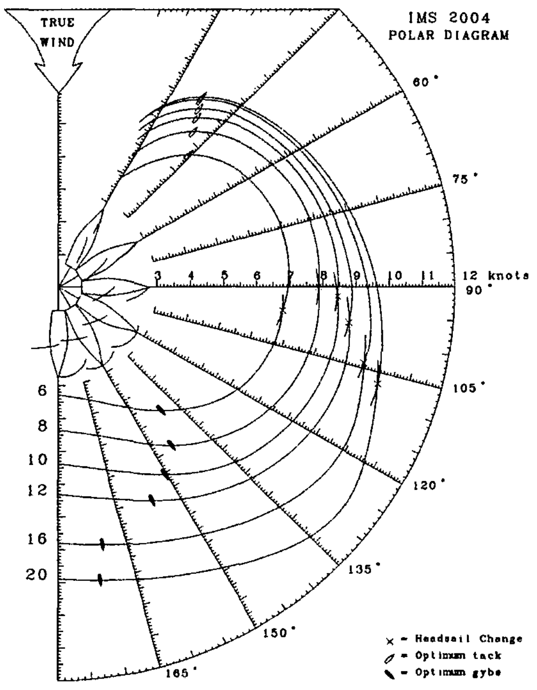


Figure 16. Polar performance curves, © ORC 2004, reproduced with permission

Much work has gone into the development of VPPs during the past thirty years, (Kerwin 1978; Oliver and Claughton 1995), with the result that they are now capable of producing reasonably accurate sets of polar performance data over a range of true wind directions and wind velocities for a wide variety of yacht types.

Although the use of yacht polar performance data is superior to the use of hull lift and drag as a measure of merit, it raises the difficulty of how one selects the exact conditions for which the yacht is to be optimised. In some cases, such as Fassardi and Hochkirch (2006), the best V_{MG} for a single wind velocity is used as a measure of merit. In other cases such as Jacquin et al. (2002), V_{MG} for multiple wind velocities are used for multi-objective optimisation.

Jacquin et al. (2002) used upwind and downwind V_{MG} values for 10 and 20 knots of wind as objectives in a multi-objective optimisation of ACC yacht designs. This analysis resulted in a group of designs being identified as Pareto optimal for the specified conditions, as shown in Figure 17, but no guidance was given as to how a selection should be made among these designs.

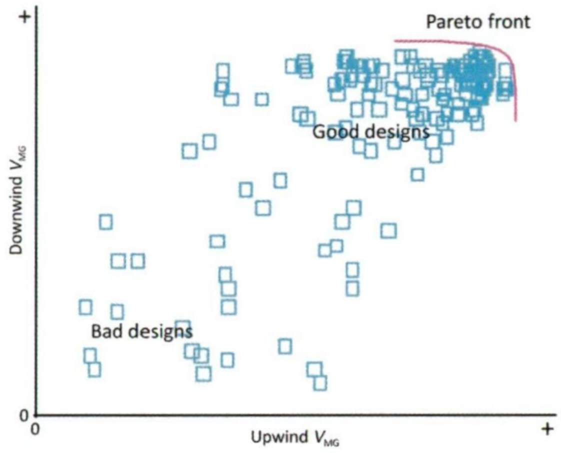


Figure 17. Pareto optimal ACC designs, from Jaquin et al. (2002)

Korpus (2007) described an optimising VPP that used RANS analysis of both hydrodynamic analysis of the hull and appendages, as well as aerodynamic analysis of the rig. Optimisation capabilities allowed up to four independent variables, which could be design parameters such as beam, C_P or mainsail camber, or operating parameters such as traveller position or tab angle. Independent variables could be restricted to the hydrodynamic or aerodynamic domains, or could be mixed.

The measure of merit chosen by Korpus was V_{MG} for multiple wind velocities. However, how these were chosen and weighted relative to one another was not described.

While these approaches are an improvement over simple lift and drag measures, VPP output is not sufficient to differentiate between two boats unless the weather in which the boats are to be sailed is taken into account. As stated by Oliver et al. (1987):

> *"There is an essential stochastic character in yacht racing, in that the relative performance of two yachts depends on the wind speed and sea conditions, which vary randomly from day to day, and more predictably from month to month. VPP results by themselves are therefore inconclusive and possibly misleading for determining the order of merit of two candidate yachts" (1987, p.240)*

2.5 COMPARISON PLOTS

In order to compare the performance of two boats it is necessary to convert the VPP derived polar data into comparison plots as shown in Figure 18.

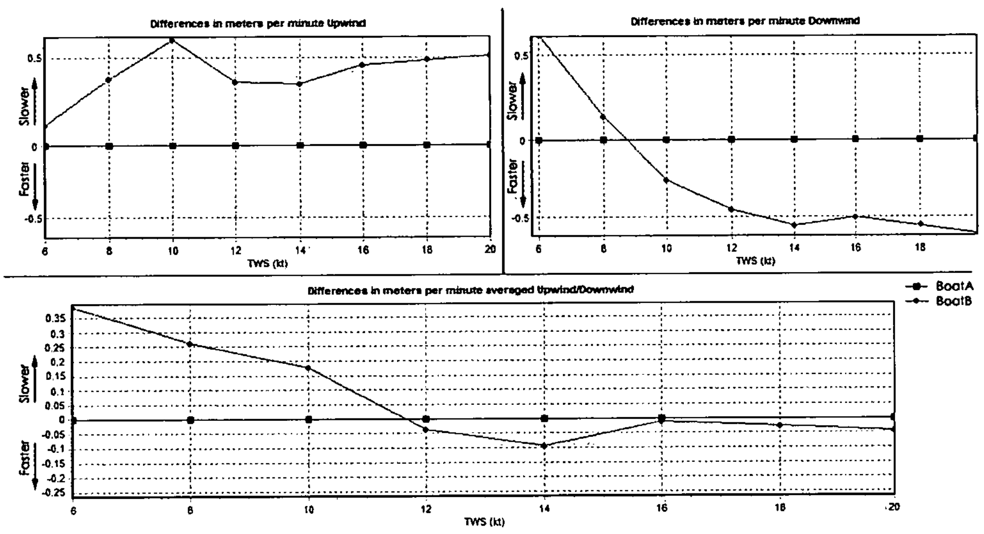


Figure 18. Performance comparison plot

Wind speed is plotted on the x-axis and deltas between the V_{MG} of the two boats on the y-axis, typically expressed in metres per minute or seconds per mile. The performance of the reference boat is plotted as a horizontal line and the test boat deltas plotted against it, with points below the abscissa being faster and points above, slower.

Comparison plots or time deltas are not appropriate as a measure of merit for an automated optimisation process, as the plots contain no information about the wind velocities or course directions in which the boats will be sailing. Hence, unless one boat is superior in all wind velocities and at all apparent wind angles, the possibility will exist that the outcome of the race will be dependent on the weather and course.

In spite of this limitation, comparison plots have been widely used as the basis for manual design optimisation for some time, (Chance 1987; Rosen et al. 2000; DeBord et al. 2002). Most importantly, they provide a rapid visual check of overall performance, and therefore play an important role in the validation of any designs created by an optimisation system.

2.6 RACE MODELLING PROGRAMS

A Race Modelling Program (RMP) uses performance characteristics derived from a VPP for two or more boats, in order to simulate a race between the boats. In its simplest form, the RMP may use a uniform wind velocity and direction and ignore interactions between boats. However, in order to model an entire yacht race more accurately other factors need to be taken into account.

On an actual racecourse the wind varies in both speed and direction, boats influence one another with backwind and wind shadow, there are multiple legs, and boats are forced to concede right of way on the course and at rounding marks. The result is a bias in favour of the boat that is faster upwind and can lead around the first windward mark. Statistics collated by the author for the America's Cup Acts between 2004 and 2006 show that for the top four boats, 80% of races were won by the boat that rounded the first weather mark in front. This corresponds with a figure of 80% quoted in several articles (Lloyd 1995; Clarey 2000) for America's Cup racing in general.

To handle these conditions and more accurately account for this bias it is necessary to create a more detailed simulation. This can be achieved by dividing the race into discrete periods and stochastically sampling wind conditions for each step from distributions derived from historical data.

Many RMP have been developed to date with varying levels of complexity, from simple probabilistic methods through to fully detailed simulations. Possibly the most sophisticated to date is the ACROBAT program, described by Philpott (2003; 2004). ACROBAT is a fixed interval time stepped simulation and was intended to be a highly accurate model of the racing performance of an ACC yacht. ACROBAT incorporated detailed calculations for many aspects of yacht racing including:

- A stochastic wind model generated using a Markov chain.
- Independent wind fields for each of the two yachts in the race, correlated according to their spatial separation.
- Modelling the dynamics of both tacking and mark rounding.
- Interactions between yachts, including backwind effect and wind shadows.
- Route optimisation, covering and collision avoidance penalties.

However, choosing the level of detail required to rank two boats accurately for the purposes of optimisation is not straightforward. Simulations that are more detailed have longer execution times but may not necessarily have a better ability to rank the performances of two boats in a match race.

2.7 MONTE CARLO RACE MODEL SIMULATIONS

Regardless of how sophisticated it is, a simulation of a single race does not provide sufficient information to permit full optimisation of a yacht design. A single race cannot sufficiently capture all of the random variation in conditions for which racing yachts need to be designed.

A more effective measure is based on a Monte Carlo simulation, where the RMP is run repeatedly for hundreds or thousands of races. Although Monte Carlo race simulations do not appear to have been used for an automated design optimisation procedure, there has been some excellent work done in this area for the comparison of specific yacht designs. This work includes that performed by PACT, the Partnership for America's Cup Technology, (Gretzky and Marshall 1993) and the ACROBAT program, (Philpott 2003; Philpott et al. 2004).

However, the seminal work on the use of Monte Carlo simulations for the design of America's Cup yachts was performed by the Sail America team for the 1987 Cup (Chance 1987; Letcher et al. 1987; Oliver et al. 1987).

Racing to select the challenger for the America's Cup final started in late October when the winds were moderate and continued through to the end of January when the sea breezes were very strong. The different rounds of the challenger selection series, the Louis Vuitton Cup, were also awarded progressively more points for successive rounds of competition, making early losses less important in terms of total points. Boats that performed poorly were eliminated after certain rounds in the competition.

The development of the winning yacht in the 1987 America's Cup, *Stars & Stripes 87,* was recounted in Letcher et al. (1987).

> *"All the technology described (CFD, VPP, RMP and Monte Carlo racing simulations) flowed together into a strategy for the design of Stars & Stripes 87. It owes its very existence to the velocity-prediction and race-model programs, because without their clear dictates it would surely not have been built.*
>
> *Our earlier 12-Metre designs, two yachts named Stars & Stripes but further designated as '85 and '86 according to the year of construction, had proved to be of unprecedented size, power, stability and speed in heavy winds. Stars & Stripes '85 had proved to be slightly faster under most conditions, and its crew had developed tremendous confidence in the boat.*
>
> *In late 1985 and early 1986, however, when the Sail America team was training and testing in Hawaii, the Australian defender candidates and many of the challengers were training in Perth and competing in the 12-Metre World Championship races. Careful observation, including photogrammetric analysis,*

showed that all these boats were substantially smaller than we had decided would be optimal for summer conditions. Results from the race-model program showed that although Stars & Stripes '85 had high probabilities of beating any known competitor in a four-of-seven series in January or February, it had only a marginal chance of surviving the round-robin eliminations among a fleet of 13 challengers that were mostly two or three feet shorter.

Faced with these predictions, Sail America had no choice but to build another boat small enough to compete effectively in the round robins. A new design could take advantage of the new hull shapes (developed with slender-ship theory and confirmed in tank tests) and of progress in computer optimised keel design. The length was chosen to be a little more than the rest of the fleet, as suggested by game theory. After the elimination rounds the span of the winglets could be increased, the boat re-ballasted for greater weight (a move that would produce a longer waterline) and the keel made more bulbous to lower the centre of gravity. Thus equipped for the stronger winds of summer, the boat would be hard to beat."
(1987, p.40)

The predictions of the Sail America design team proved correct. *Stars & Stripes 87* struggled to win races in the early rounds of the challenger selection series, despite significant changes to sail area and ballast aimed at "re-moding" her for light winds. As the summer progressed and the Fremantle sea-breezes strengthened, *Stars & Stripes 87* improved her standing relative to the other challengers, and this culminated in the boat winning the challenger selection series. *Stars & Stripes 87* proceeded to dominate the Australian defender in the America's Cup series, winning four races to nil.

The work performed by Sail America was ground breaking for several reasons. Not only did it result in an America's Cup winning design, it did so by adopting an apparently high-risk strategy that would not have been obvious without the analysis performed using Monte Carlo and game theory based methods. Other design teams faced with the same weather predictions concluded that a boat with good all-round performance was required. In contrast, Sail America's early analysis determined that losses in light winds during the early rounds of the challenger selection series were of little importance, compared to wins in the later, heavy weather rounds, as long as *Stars & Stripes 87* could avoid elimination.

The surprise for Sail America was that other teams did not come to the same conclusion and build boats tailored to stronger winds. The realisation by Sail America that its best yacht for the expected conditions was not ideal, because of the designs chosen by its competitors, was crucial. The game-theory based analysis that followed, which resulted in a compromise length being chosen which was optimal compared to the known dimensions of competing yachts, was instrumental in Sail America winning the 1987 America's Cup.

2.8 TOURNAMENT MODELLING

The research performed by the Sail America design team illustrates several key points:

- Variations in wind velocity, both intra-day and across the duration of the competition need to be accounted for.
- The structure of the tournament (i.e. number of competitors, points allocated per race and timing of competitor eliminations) can affect the outcome of the simulation.
- Most significantly, the ideal boat for a competition may not necessarily be a static optimum; rather, it may be dependent on the design of the opposing boats.

These points are as valid today as they were in 1987. The America's Cup is now a best of nine race series, while the Louis Vuitton Cup, the selection series for the America's Cup challenger, has significantly more races over a period of 8 weeks.

The successful challenger for the America's Cup has to first win the challenge series and then compete against the defending yacht. The total period for this is close to 3 months, over which the standard deviation of wind velocity will be significantly higher than the typical standard deviation for a typical sailing day. Due to seasonal variations, the mean wind velocity may also vary over the 3 month period.

In addition, the America's Cup is a tournament of individual matches between two yachts, and success is based on points accumulated for race wins, not accrued race times. Races won by one second carry as many points as races won by five minutes. This has a significant effect on the measure of merit. For example, compare a yacht that wins five races by five seconds each and loses one race by one minute, against a yacht that loses five races by five seconds and wins one by one minute. The second yacht may have a shorter accrued time for all the races taken together, so arguably could be termed the faster yacht. However, the first yacht will get the majority of the points, and in the case of the America's Cup final, would win the event.

To account for these effects it is important to evaluate the ability of a yacht to win races against a range of opponents, over a range of weather conditions, by simulating an entire tournament of races.

2.9 WEATHER MODEL

The weather model for VESPA was formulated to have the following properties:

- A mean wind strength chosen for each simulated race, by sampling a wind distribution. This distribution should ideally be derived from historical meteorological data for the specific times of day and period of the tournament. This distribution should also be adjusted for any drift in the mean wind velocity over the period of interest. These wind velocity distributions should include upper and lower cut-off values to handle the wind velocity limits above and below which races are not sailed.
- Each race should model temporal variation in wind velocity and direction based on an estimate of variance for wind velocity and direction. For example, the wind velocity standard deviation for a 3-month period may be 8 knots, while the standard deviation for a single race, taking only 90 minutes, may be 3 knots.

2.10 MEASURE OF MERIT - SUMMARY

These factors make a strong case for the use of a full tournament model as the measure of merit for an America's Cup yacht design optimisation. This choice of measure of merit has a significant impact on the design of the VPP and RMP, as well as affecting the selection of an appropriate optimisation method.

Based on the factors outlined in the previous sections, the measure of merit for VESPA was formulated to have the following properties:

- It should use a tournament of multiple boats, with the measure of merit being the win/loss ratio against all competitors.
- It should include multiple races against all competitors, each race having its own mean wind velocity.
- Wind velocity and direction for each race should vary about their mean, based on empirically derived variance for wind velocity and direction.

3 · CONSTRAINTS

The primary constraints that apply to any optimisation of an ACC yacht are those imposed by the ACC rule, which specifies significant limits on the design of both hull and rig. These constraints need to be taken into account for each design change that occurs during the optimisation process.

3.1 THE AMERICA'S CUP CLASS RULE

The ACC rule specifies a small number of measurements that are combined in a simple equation, the result of which cannot exceed a given value. Although earlier versions of the rule permitted a wide range of designs, the most recent rule, version 5, reduced this range considerably. Despite these recent amendments, there is still considerable scope for hull shape variation within the confines of the rule. However, this tends to be in areas other than the three key measurements of length, sail area and displacement.

The rule also imposes significant constraints on several areas of the hull design which must be complied with. In particular, minimum freeboard values are specified at three points along the hull, and maximum girths are specified at the fore and aft girth stations. In addition, the ACC rule requires that the hull shape be convex in all directions.

The ACC rule requires that the following equation be satisfied-

$$\frac{L + 1.25 \times - \sqrt{S} - 9.8 \times \sqrt[3]{DSP}}{0.686} \leq 24.000 \tag{3.1}$$

where:

- S is the rated sail area in m^2
- DSP is the displacement in m^3
- L is the rated length in m

These parameters are defined by the following equations:

$$S = SM \times (1 + 000.1 \times (SM - 320)^4) \tag{3.2}$$

where SM is the measured sail area in m^2

$$DSP = W/1025 \tag{3.3}$$

where W is the weight of yacht in kg.

$$L = LM \times (1 + 2000 \times (LM - 22.1)^4) + FP + WP \tag{3.4}$$

$$LM = LBG + G \tag{3.5}$$

$$G = FGC + AGC \tag{3.6}$$

FGC is the greater of -

$$0.3m \text{ OR } 1.25 \times (FG - 2.4 + FBC) \tag{3.7}$$

FBC is the greater of -

$$-0.116m \text{ OR } -1.8 \times [(1/cos\Theta) - 1] \tag{3.8}$$

AGC is the greater of -

$$1.6m \text{ OR } 1.75 \times (AG - 1.8 + ABC) \tag{3.9}$$

ABC is the greater of -

$$0 \text{ OR } 1.414 - 1/cos\Phi \tag{3.10}$$

$$WP = 4 \times [\sqrt[3]{W} - 28.845] \tag{3.11}$$

where:

- *LM* is the measured length in metres
- *LBG* is the length between girth stations in metres
- *G* is the girth component of *LM* in metres
- *FGC* and *AGC* are the forward and aft girth corrections in metres
- *FG* and *AG* are the forward and aft chain girths in metres
- *FBC* and *ABC* are the forward and aft beam corrections in metres
- Θ and Φ are the mean of the angles port and starboard of the topsides, measured at *FGS* and *AGS* relative to the vertical, in degrees
- *FP* is the freeboard penalty in metres

Figure 19 illustrates the locations at which these dimensions are measured. Note that *LBG* is measured in a plane 200mm above the measurement waterline *MWL*. This is an effective way of preventing designers from creating excessive overhangs fore and aft to gain unmeasured length. If L_{WL} was the only length measure used this would encourage designers to utilise extreme overhangs in their designs.

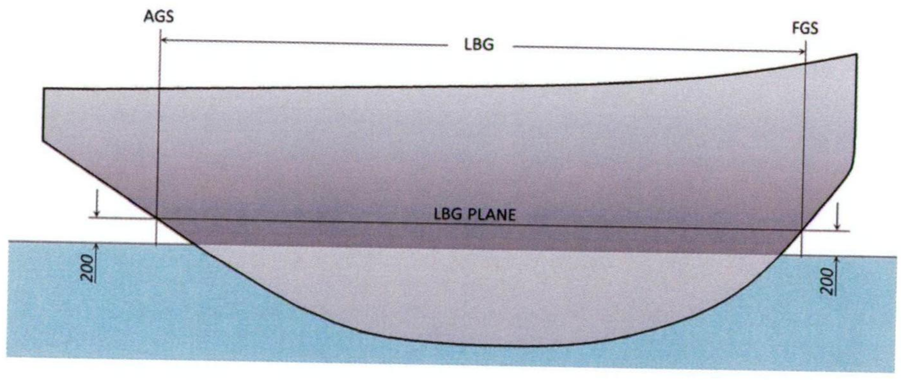


Figure 19. ACC Hull Measurements

The two girth correction values, *FGC* and *AGC*, are penalties that are invoked once *FG* and *AG* rise above certain values. In general, designers of ACC yachts aim to use the maximum possible unpenalised girth values of 0.3m for *FGC* and 1.6m for *AGC*, giving a total value for *G* of 1.9m.

A freeboard penalty, *FP*, is also applied for freeboards that are less than those specified by the rule, and a weight penalty, *WP*, is applied if the weight is greater than 24,000 kgs (see eq. 4.11). ACC designs usually avoid incurring these penalties by staying within the limits specified by the ACC rule.

Assuming zero values for *FP* and *WP*, equation (3.4) can be restated as –

$$L = (LBG + 1.9) \times \left(1 + 2000 \times \left((LBG + 1.9) - 22.1\right)^4\right) \tag{3.12}$$

This shows that if no weight, freeboard or girth penalties are incurred, *L* is solely dependent on *LBG* and is at a minimum at an *LBG* of 20.182m. Deviation from this *LBG* results in a rapidly increasing penalty, as shown in Figure 20.

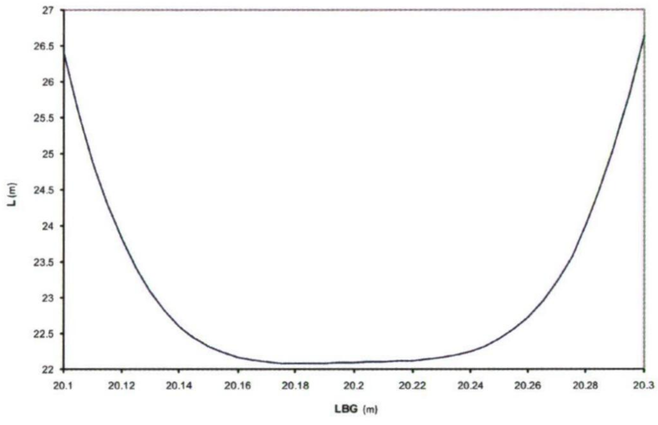


Figure 20. Rated Length (L) versus Length Between Girths (LBG)

As can be seen from equation (3.2), rated sail area *S* is dependent solely on the value of measured sail area, *SM*. This function is graphed in Figure 21.

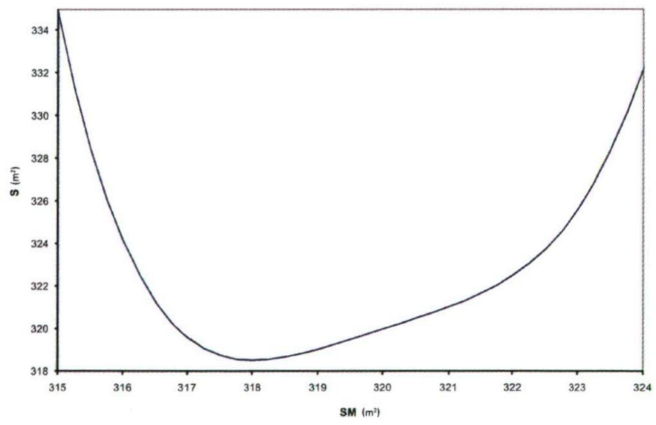


Figure 21. Rated Sail Area (S) versus Measured Sail Area (SM)

To view the three-dimensional parameter space of the ACC rule, it is necessary to restate equation (3.1) in terms of weight –

$$\mathrm{W} = \left[\frac{L + 1.25 \times \sqrt{S} - 24 \times 0.686}{9.8}\right]^3 \times 1025 \tag{3.13}$$

Given that *L* and *S* can be expressed in terms of *LBG* and *SM* (equations (3.12) and (3.2) respectively), *W* can be graphed as a function of *LBG* and *SM*, as illustrated in Figure 22.

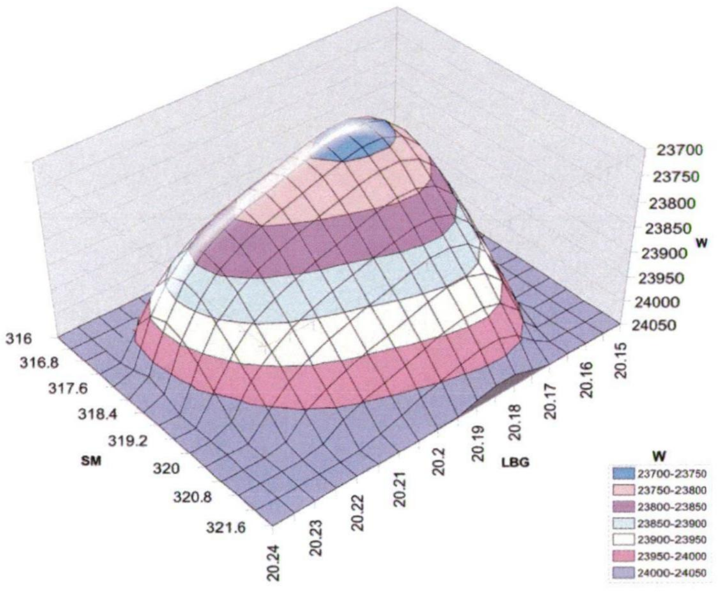


Figure 22. Weight (W) as a function of Length Between Girths (LBG) and Measured Sail Area (SM)

The weight penalty *WP* heavily penalises displacement as it deviates from a value of 24,000 kg, leaving most of the displacement bands shown in Figure 22 unusable. In reality, the available parameter range is even smaller, as only values in the upper right quadrant of the design space will maximise both *LBG* and *SM*.

As a result, allowable values for *LBG* and *SM* lie on the curve shown in Figure 23. The usable range for *LBG* is from 20.182m to 20.234m, a difference of 0.052m, while the range for *SM* is from 318m^2 to 321.4 m^2, a difference of only 3.4 m^2.

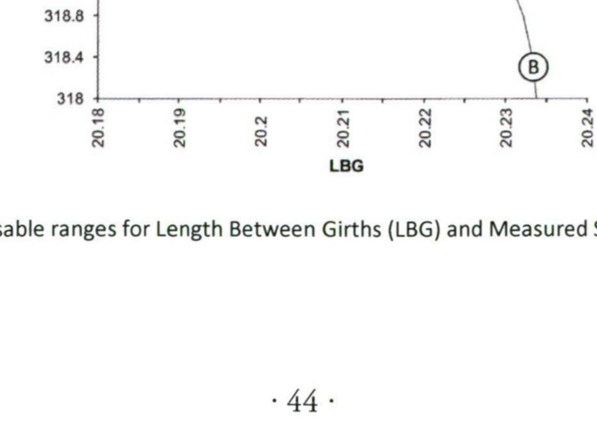


Figure 23. Usable ranges for Length Between Girths (LBG) and Measured Sail Area (SM)

Variation in expected weather conditions for an event may result in some optimisation of these variables taking place. For example, a light weather event might result in designs clustering around point A in Figure 23, (i.e. moderate *LBG* and maximum *SM*) while a heavy weather event may see designs favouring longer *LBG* and lower *SM* values, such as at point B. However, it is clear that the narrowness of the range for each parameter provides very little scope for significant variation of *SM* and *LBG* design variables. For example, boats with *LBG* and *SM* values shown by points A and B in Figure 23, would each have speed advantages over the other of less than 0.5 seconds per mile, in the conditions that suit each best.

CONSTRAINTS OR BOUNDS?

While the key dimensions of an ACC yacht are heavily constrained by the ACC rule, these constraints are most efficiently applied at the point in the optimisation process that design variables are modified. Rather than the optimisation process having conventional constraints, design variations are confined to the feasible range by adjusting dependent hull-design variables so that the ACC rule is not violated. For example, regardless of how other hull shape parameters are varied, *LBG* is scaled so that it always stays within the acceptable range, while *W* is kept at a value of 24,000 kg, primarily by scaling canoe body draft, T_C.

As a result of this preliminary check for feasibility, the optimisation process becomes an unconstrained problem, with simple upper and lower bounds on design variables. Although extreme values of these bounds may result in designs that were not geometrically achievable, in practice, given the relatively narrow ranges specified for the bounds for each design variable, no design variable combinations used by VESPA were found to be infeasible.

EFFECT OF THE ACC RULE ON DESIGN CHOICES

Like most yacht rating rules, the ACC rule is type forming, in that it encourages great similarity between designs due to the values chosen for trade-offs between length, sail area and displacement, and the nature and location of the measurement points. ACC yachts have evolved to be extremely slender, with long overhangs, deep draft and high ballast ratios. The hulls themselves have become boxy and slab sided with significant flat areas, joined by areas of high curvature.

A good example of these trends is illustrated by Figure 25, which shows ITA-94, the most recent yacht of the Italian Luna Rossa team. The flat bottom and vertical topsides are examples of the extremes to which hull shapes have developed. These extreme characteristics have occurred, not because they are the most efficient hydrodynamically, but because they are the best compromise within the constraints of the rule.

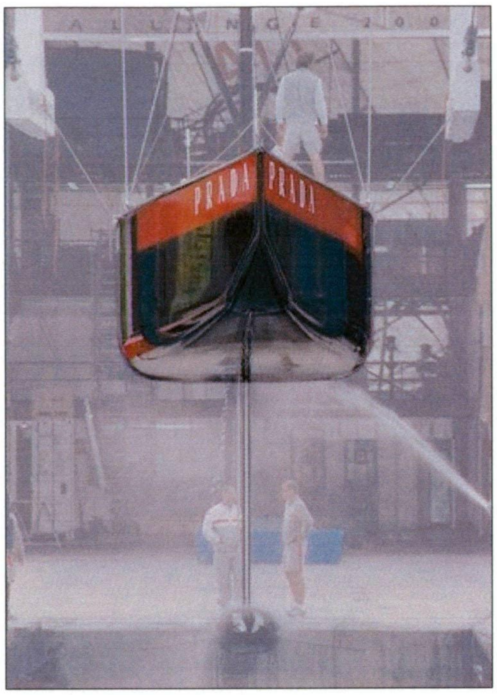


Figure 24. ITA-94, an example of ACC hull design extremes

There is also little scope available for the optimisation of rig parameters. The ACC rule strictly regulates the length of the mast, together with its weight and centre of gravity. The luff length of the mainsail *P* and the foretriangle height *I* also have set limits. As a result, all ACC yachts have rig proportions similar to those shown in Figure 25.

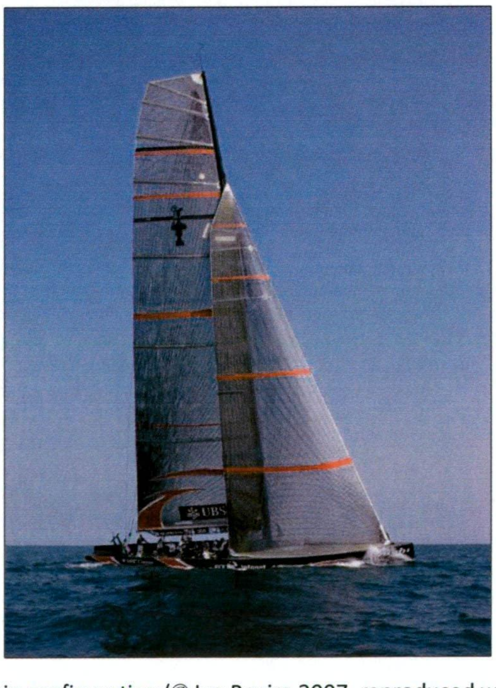


Figure 25. ACC rig configuration (© Ivo Rovira 2007, reproduced with permission)

As a result of *P* and *I* values being similar for all boats, optimisation of the sailplan is restricted to the foretriangle base value *J* and the horizontal mainsail girth values *E1* to *E5*. As most teams have adopted *J* values of approximately 8.5m, only the mainsail girths have been a subject of significant design variation for this Cup cycle.

For the purposes of the VESPA project, no attempt was made to optimise rig proportions. Rather, a set of rig dimensions was adopted that corresponded very closely to those used by Alinghi's rig designers, and these dimensions were used for all hull design variants. It was considered that within the range of hull shapes examined, the standard parameters used by VPPs to vary and optimise sail forces, i.e. reef, flat and twist, are adequate to handle the required variation in sail shapes.

Similarly, no attempt was made to optimise appendage design using VESPA, with a single bulb, fin and rudder used for all design variations. The geometric model of these appendages extended up into the canoe body so that decreases or increases in canoe body draft would reveal more or less fin and rudder area. Keels and rudders were moved fore and aft to compensate for different L_{CB} and aft waterline ending positions, with the L_{CG} of the yacht moved to correspond with the L_{CB} location and the rudder maintaining a constant distance from the aft waterline ending.

It was considered that the design of the appendages was sufficiently independent of the hull design variations examined, that the use of a single appendage package was an acceptable simplification of the optimisation problem.

With the key speed producing variables of length, sail area and displacement severely constrained by the ACC rule, there are only a small number of hull shape variables remaining to be explored by an optimisation process. Of these, the most obvious candidates for design optimisation are parameters controlling transverse sectional shape (B_{WL}, T_C, C_M and flare) and longitudinal area and volume distribution (L_{CB}, C_{WP}, C_{IL} and C_P).

While the restricted design space makes optimisation more difficult, as the gains to be made are likely to be small, the relatively small number of free parameters also simplifies the optimisation significantly, with the variation of hull shape restricted to as few as five key variables.

The narrowing of the design parameter space has also dramatically limited a team's ability to "re-mode" a boat between races to suit variations in weather. In 1987, the designers of *Stars & Stripes* changed the ballast and sail area carried by the yacht in order to tailor performance to the expected weather conditions for each round-robin of the elimination series. This was possible within the 12 Metre class yachts used for the America's Cup at the time, which allowed a wide range of waterline lengths, displacements and sail areas to be used. While earlier versions of the ACC rule also allowed some re-moding to occur, the narrowing of design parameter ranges for the version 5.0 ACC rule has prevented significant re-moding of the competing boats.

4 · DATA APPROXIMATION

In order to perform a useful optimisation of an America's Cup Class yacht design within a reasonable time using readily available computer hardware, it is necessary to find ways to reduce the number of calls made to expensive analysis functions.

Although CFD methods are constantly developing and improving, they are at the same time becoming more, rather than less, computationally expensive. Potential flow methods, although theoretically less accurate than RANS codes, are still widely used due to their relatively short execution times. However, potential-flow methods can still be prohibitively time consuming when a large number of cases must be calculated.

To make a significant reduction in the time taken to calculate the lift and drag of a yacht being analysed, it is necessary to look at the use of an approximation derived from a small number of sampled data points. This approximation model, known as a surrogate, or more commonly, a metamodel, is typically calculated in advance using samples calculated by the chosen CFD code. Alternatively, the metamodel may be calculated in real time as required, and may be progressively refined using additional samples as the optimisation proceeds.

Metamodels have been widely used in aerodynamic optimisation work for more than a decade, (Greenman 1998; Simpson et al. 1998; Pierret 1999; Jin et al. 2001;

Jin et al. 2002; Ong et al. 2003; Queipo et al. 2005; Zhou et al. 2007; Alonso et al. 2009). However, there are few examples of the use of metamodels in the hydrodynamic optimisation of ships and boats, such as Duvigneau (2002), Peri and Mandolesi (2005), Peri and Campana (2005b) and Mason et al. (2005).

In order to create a metamodel it is necessary to sample the design parameter space to provide a set of analysis results to which the hypersurfaces of the metamodel can be fitted. Although there are many possible sampling schemes, such as random sampling or full factorial arrays, the approach favoured in the metamodelling literature, summarised by Giunta and Wojtkiewicz (2003), is the use of a quasi-random sequence having low discrepancy between the sampled points. In this case, the discrepancy $D(N)$ for a sequence $\{s1, s2, s3, ...\}$ with respect to the interval $[a,\ b]$ may be defined as:

$$D(N) = \sup_{a \le c \le d \le b} \left| \frac{|\{s_1, \ldots, s_N\} \cap [c, d]|}{N} - \frac{d-c}{b-a} \right| . \tag{4.1}$$

A sequence is thus equidistributed if the discrepancy $D(N)$ tends to zero as N tends to infinity.

The selection of a set of design parameters is not sufficient to allow CFD analysis, as this also requires a geometric model of a complete hull surface in order to perform its calculations. For this to occur, it is necessary to have a method that can create from scratch a complete hull geometry meeting those parameters (parametric modelling), or alternatively, be able to deform an existing hull design to match the design parameters required (parametric transformation).

This chapter describes the factors that influence the choice of sampling method, metamodel type and hull-shape representation and variation method adopted for use by VESPA.

4.1 DATA CHARACTERISTICS

The choice of metamodel selected for a particular optimisation task is influenced by several factors, some of which are related to the characteristics of the source data on which the metamodel is to be based.

In the case covered by this thesis, the data in question comes from SPLASH, a potential flow program that has been widely used in the field of America's Cup yacht design since 1987. SPLASH is used by the Alinghi team for day-to-day CFD analysis and the program was used for all hydrodynamic calculations performed in the course of this research.

Like other CFD codes, SPLASH is a deterministic program, meaning that it will give identical results when given identical inputs. However, this does not mean that there is no noise in the system. SPLASH may contain low levels of systematic noise caused by round-off error, as well as discretisation artifacts caused by breaking the hull surface into a finite number of panels.

SPLASH may also fail to converge on a solution in some cases, particularly at combinations of high forward speed coupled with significant heel and/or yaw. These non-converged points may have significant error associated with them, in which case they need to be identified as outliers and eliminated. Alternatively, the error may be small enough that the data point can be included but be treated as if it has a small amount of random, but repeatable noise.

4.2 METAMODELS

Many alternative metamodel formulations exist which may be suitable for use in the proposed optimisation system, ranging from traditional statistical regression through to methods that are quite recent, such as radial basis function networks. Despite the wide range of alternatives, no method has shown to be overwhelmingly superior across a range of approximation tasks.

Criteria used in the selection of a metamodelling method include the quality of fit of the metamodel to the sampled data points and the smoothness of the resultant fitted hypersurfaces. However, the choice of metamodel type for a particular application is also dependent on several other factors:

- Quantity of data available;
- Dimensionality of the solution space;
- Complexity of the solution surface;
- Degree of noise associated with the data;
- Ease of use.

QUANTITY OF DATA

For a problem where only a small number of data points are available, the metamodelling strategy may differ from the case where a large volume of data has been provided. For example, a neural network metamodel may be unsuitable for small datasets, particularly as a significant proportion of the data needs to be set aside for validation and test sets and is therefore not available for network training. In such cases, a conventional regression model may be more appropriate.

SOLUTION COMPLEXITY.

The solution space of a CFD analysis may have a few or many dimensions; the solution surfaces may be relatively smooth or highly non-linear; and there may be a single optimum or multiple local optima.

Some metamodelling methods, such as polynomial regression, which work exceptionally well on smooth, unimodal solutions surfaces having few dimensions, may be completely inadequate when applied to high-dimensional, non-linear, multimodal solution spaces. It is essential that the characteristics of the metamodelling method chosen is appropriate to the type of data to be fitted.

In the CFD case examined in this work, the parameter space was 10 dimensional, the input variables for each analysis being *LBG*, B_{WL}, C_P, C_M, L_{CB}, heel angle, yaw angle, rudder angle, tab angle and hull velocity. The results of Sahoo (1997), Yasukawa (2000), Hirata (2004) and Pinto (2004) strongly suggest that solution surfaces in this case will be both non-linear and multimodal.

NOISE.

Physical experiments such as towing tank data usually include a component of random noise which causes problems for metamodelling approaches that fit data points exactly. For these data, it is preferable to use a least-squares regression method that allows a smooth surface to be fitted through the noisy data.

Similarly, although some deterministic computer experiments may be completely noise free, many CFD codes contain low levels of systematic noise caused by round-off errors and discretisation artifacts. This type of data also benefits from a least squares fitting approach.

Another issue is that calculations do not always converge perfectly, particularly in cases of high angles of heel and yaw. This is illustrated in Figure 26, reproduced from Keane and Nair (2005), which shows an objective function contaminated by discretisation noise and occasional failed calculations.

This form of noise differs from the noise expected from physical experiments, in that it cannot be assumed as normally distributed with a zero mean. This potential for non-random noise occurs with SPLASH, making purely interpolative methods inappropriate in this case.

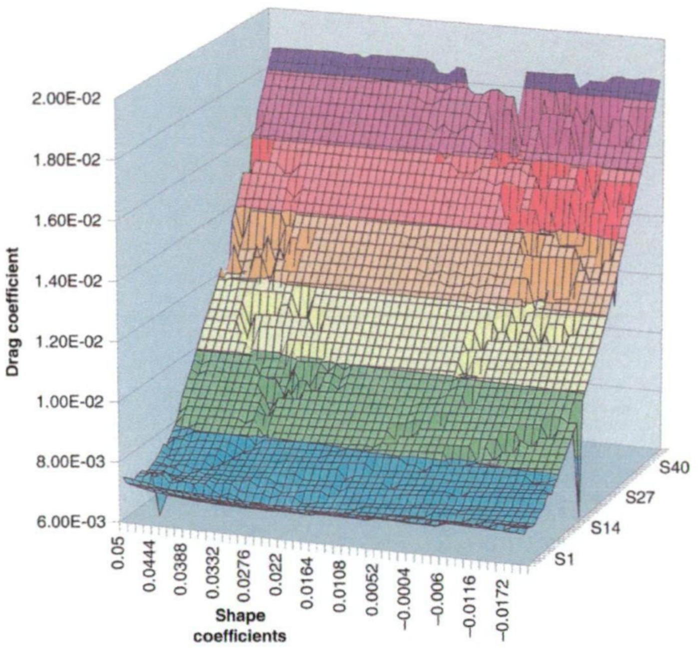


Figure 26. Objective function contaminated by noise, from Keane and Nair (2005)

EASE OF USE.

Ease of use may appear to be a minor consideration. However, it encompasses several issues that may directly affect the successful creation of the metamodel. These include:

- Ease of implementation. Is tested and validated software available for the metamodelling method, either source code or executable file, which allows it to be easily implemented within the optimisation environment?
- Is the metamodelling method automatic, or does it have multiple settings which require expert knowledge in order to tune and optimise the metamodel? If it does have multiple tuning parameters, are the parameters and tools for optimising the metamodel accessible and easy to use?
- Does the method provide information about the nature of the data being fitted, or is it a "black box" method that provides no information other than the outputs produced for a given set of input parameters?
- What measures are available to verify the quality and prediction accuracy of the metamodel?

4.2.1 METAMODEL CANDIDATES

Metamodelling methods supported by the optimisation literature include polynomial regression (PR), Response Surface Methods (RSM), (Box and Wilson 1951), Gaussian Processes, Kriging, Artificial Neural Networks (ANN), Radial Basis Function networks (RBF), Support Vector Machines (SVM), (Burges 1998), Least Interpolating Polynomials (De Boor and Ron 1990), and Multivariate Adaptive Regression Splines (MARS), (Friedman 1991).

Of these, the first six methods are widely used within the field of optimisation, and each of these has been shown to work well across a range of problems.

Several authors have performed comparative evaluations to determine the methods that have the best performance for a range of problem domains. Jin et al. (2003) compared polynomial regression, MARS, Kriging and RBF across a range of test problems. Polynomial regression outperformed the other methods on small-scale, low-order, non-linear problems. However, RBF had superior performance in all other domains.

Simpson et al. (2000) compared a range of sampling strategies combined with four metamodel methods: second order RSM, MARS, Kriging and RBF. The polynomial based RSM did well approximating low-order non-linear functions, but performed poorly with more complex problems. The least stable method was found to be MARS, while both RBF and Kriging performed well.

Simpson et al. (2001) surveyed prior literature to provide recommendations on metamodel selection. Simpson's conclusions are summarised in Table 3.

Table 3. Recommendations For Model Choice And Use, After Simpson (2001)

Model Choice	Characteristics/Appropriate Uses
Response Surfaces	• well established and easy to use • best suited for applications with random error • appropriate for applications with < 10 factors
Neural Networks	• good for highly non-linear or very large problems • best suited for deterministic applications • high computational expense • best for repeated application
Kriging	• extremely flexible but complex • well suited for deterministic applications • can handle applications with < 50 factors • limited support is currently available for implementation

Note that although Kriging and Gaussian Processes are both widely mentioned in the literature, they are essentially the same thing, with the term Kriging originating in the field of geostatistics, while Gaussian Processes is a term used by the statistics community. Kriging/Gaussian Processes are interpolative methods, making them ideal for deterministic computer experiments that do not have a noise component. However, this characteristic can cause difficulty when dealing with noisy data. When noisy data points are precisely interpolated, the result tends to be uncontrolled oscillations in the solution surfaces, raising the possibility of false optima being created by the metamodelling process.

The wave-making resistance of marine vessels is known to contain non-linearities. Hirata (2004) and Yasukawa (2000) showed that even for basic monohull merchant ships, the solution space of the hull resistance optimisation problem is multi-modal. It is also known that CFD codes such as SPLASH may contain low level noise and unconverged results.

As a result, it is expected that the data produced by a SPLASH analysis of a fleet of America's Cup Class yachts will be non-linear; with a potentially multi-modal solution space of moderate complexity and dimensionality; derived from a deterministic analysis but containing some element of systematic noise and non-Gaussian random noise. Consequently, the following methods may be eliminated as candidates, based on the recommendations contained in the references cited above:

- Polynomial regression, due to the complexity and non-linearity of the CFD data.
- Response Surface Methods, due to the complexity and non-linearity of the CFD data.
- Kriging is an interpolative method that fits the data exactly. Consequently, it is not ideal for use with data containing random or systematic noise.

This leaves artificial neural networks, together with their related method, radial basis function networks, as the preferred options for the metamodelling of high-dimensional, non-linear, multi-modal data containing systematic noise. Regarding neural networks, Chen et al. (2003) make the comment:

> *"Although ANNs are generally flexible enough to model anything, they are computationally intensive, and a significant quantity of representative data is required to both fit and validate the model.... However, given enough good data, ANNs can outperform all the other previously described statistical modeling methods." (2003, p.245)*

ARTIFICIAL NEURAL NETWORKS (ANN)

An artificial neural network (ANN) is a network composed of many simple processors referred to as units or neurons, linked by communication channels or connections that carry encoded numeric data. These units operate only on their local data and input they receive via the connections, effectively applying a multiple linear regression followed by a non-linear transformation, on the output value y. This transformation, or transfer function, is most commonly a sigmoidal function, as shown in Figure 27. However, other transfer functions such as tangent-sigmoid or linear may also be used.

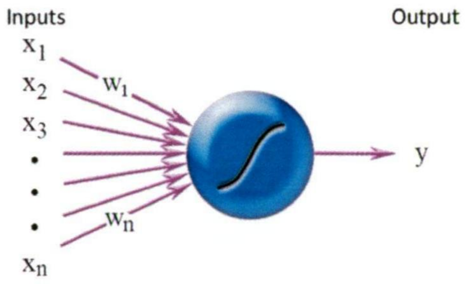


Figure 27. Neuron with sigmoidal activation function

If the inputs to each neuron are designated $\{x_1, x_2, \dots, x_n\}$, and the regression coefficients are designated by the weights, $\{w_1, w_2, \dots, w_n\}$, then the output, y, is given by:

$$y = \frac{1}{1 + e^{-\eta}} \tag{4.2}$$

where

$$\eta = \sum_{i=0}^{n} w_i x_i + \beta \tag{4.3}$$

and

β is the "bias value" of the neuron.

A neural network is created by assembling neurons into a network architecture. There are many different architectures and topologies for neural networks. The form that has found widest application in the area of metamodelling is the feed-forward network or multi-layer perceptron (MLP), shown in Figure 28.

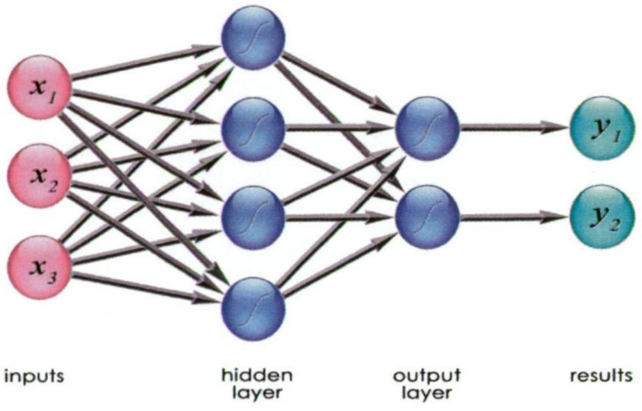


Figure 28. Multi-layer perceptron architecture

An MLP has an input layer with a series of inputs, one or more hidden layers and an output layer. The number of input elements is determined by the number of variables in the input dataset, while the number of outputs is determined by the number of result values required.

The number of hidden layers and elements in the network can vary, and finding the optimal network architecture for fitting a given dataset is a non-trivial problem.

Rather than being programmed, a neural network "learns" from examples presented to it based on some form of training rule. In many cases the backpropagation algorithm (Rumelhart et al. 1986) is used to train the neural network. However, neural networks are equally amenable to other gradient-based approaches such as the conjugate gradient method, the quasi-Newton method or the Levenberg-Marquardt algorithm.

According to Sarle (1994):

> *"MLPs are general purpose, flexible, non-linear models that, given enough hidden neurons and enough data, can approximate virtually any function to any desired degree of accuracy. In other words, MLPs are **universal approximators.**" (1994, p. 5)*

Although Sarle identifies many similarities between neural networks and traditional polynomial regression methods, neural networks also have several distinct advantages. Neural networks are global models, allowing them to model the entire range of interest rather than a smaller portion. The lack of appropriate global models has been a source of difficulty in the past for ship resistance approximation, with vessels having different speed/length or displacement/length ratios requiring different resistance regression models.

Neural networks are non-parametric models, and consequently there is no need to choose the functional form for each dimension, meaning that a potential source of error is avoided. However, a non-parametric approach brings with it the risk of over fitting to any noise or errors in the data. As a result, it is essential to use a rigorous approach to validation of the trained neural network.

As neural networks learn by example, they do not require the traditional statistical assumptions such as constant error variance and Gaussian distribution of errors. Instead, the neural network user gathers representative data and invokes training algorithms to learn the structure of the data. Although the user requires some knowledge of how to select and prepare the data, the level of statistical expertise required to create a useful neural network model is less than is required to create a conventional regression model of equivalent quality.

RADIAL BASIS FUNCTION NETWORKS (RBF)

Radial Basis Function networks (RBF) have been developed for scattered multivariate data approximation (Dyn et al. 1986). They are similar in structure to a multi-layer perceptron, being a feed-forward network with an input layer, hidden layer and output layer Figure 29.

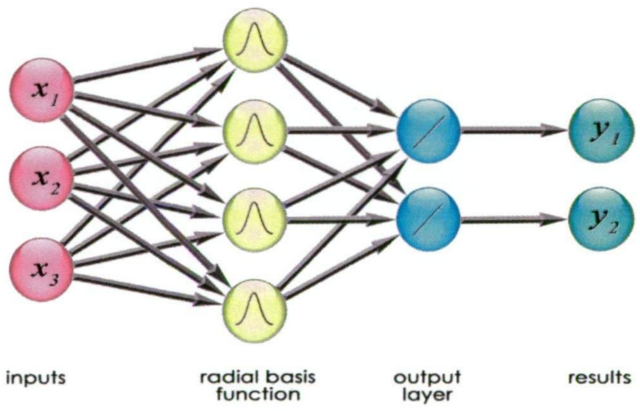


Figure 29. Radial Basis Function network architecture

Rather than use the sigmoidal activation function, RBFs use linear combinations of a radially symmetric basis function, using a Euclidean distance, to approximate response functions.

A radial basis function model can be expressed as:

$$y(\mathbf{x}) = \sum_{i=1}^{N} w_i \phi \|\mathbf{x} - \mathbf{c}_i\| \tag{4.4}$$

where $\mathbf{x}$ is a vector of inputs, w_i are weighting coefficients, c_i are RBF centres, $\|\cdot\|$ denotes the Euclidean norm, and ϕ is a nonlinear function, typically Gaussian:

$$\phi\|\mathbf{x}-\mathbf{c}_i\| = \exp[-\beta\|\mathbf{x}-\mathbf{c}_i\|^2] \tag{4.5}$$

RBF approximations have been shown to produce good fits to both deterministic and stochastic response functions (Powell 1987). They have been used successfully by several researchers in the marine optimisation field including Peri and Campana (2005c), and Fassardi and Hochkirch (2006).

4.2.2 METAMODELLING – PRACTICAL USAGE

Both neural networks and radial basis function networks can be expected to give good results for the lift and drag data that VESPA uses, which is derived from SPLASH. The SPLASH data forms a solution surface, which is likely to be relatively smooth, yet non-linear and possibly multi-modal, with the presence of some non-random noise.

Other significant differentiating factors in the choice of a metamodel are data management issues to do with ease of fitting, validation and visualisation. These are primarily related to the user interface of the software involved and favour methods that are sufficiently mature and widely used for there to be a variety of well-designed commercial software available.

The ability to optimise the quality of fit and validate results easily is paramount, as a small error can drive the optimisation process to give unrealistic results. Software with good data manipulation and visualisation capabilities is therefore of great value.

4.2.3 METAMODEL SELECTION

After careful evaluation of alternatives, neural networks were selected for use in VESPA for the creation of metamodels for lift and drag data created by SPLASH. In previous work by the author, (Mason et al. 2005), neural networks were shown to perform well in this type of application, using powerful commercial software available at reasonable cost. Although RBF networks showed great promise, the lack of available software with strong analysis and verification features prevented their adoption.

Although the primary use of a metamodel is the reduction of the evaluation time of an expensive function, there are additional benefits. For noisy functions, a metamodel can smooth the data making the optimisation process simpler. For functions that can suffer from some numerical instability, such as some CFD programs, the process of fitting the metamodel can also be used to filter out non-converged points for a net increase in data quality.

4.3 DESIGN OF EXPERIMENTS

The adoption of a metamodel as a surrogate for direct calculation of CFD data allows for a dramatic reduction in the number of CFD calculations but requires that the design space be sampled. As this sampling will be sparse, it is important that the spacing of the sample points be regular so that the metamodel hypersurfaces may be fitted without introducing excessive bias and variance.

In order to determine the most efficient way of sampling the design space, it is necessary to refer to the field of Design of Experiments (DOE), which commenced with the work of geneticist and statistician Sir Ronald A. Fisher, (1935). DOE is based on several concepts originated by Fisher, including orthogonality of variables, experiment randomisation, Analysis of Variance (ANOVA) and the use of factorial experiments rather than the traditional one-factor-at-a-time method.

One difficulty for full factorial designs was the "curse of dimensionality". As the number of factors in an experiment increased, the number of experiments required for a full factorial experimental design grows exponentially. For example, in order to fit a linear model to an experiment having ten factors or dimensions, 2^{10} samples are required. If the solution surface had a quadratic form, requiring 3 points in each dimension, a ten dimensional problem would require 3^{10} samples.

In place of full factorial designs, DOE practitioners investigated how to minimise the number of experiments performed while retaining the ability to accurately determine the influence of each factor in the experiment. This resulted in various fractional factorial designs being created, of which two of the better known designs, the Box-Benkhen and face-centred, central composite design are illustrated in Figure 30.

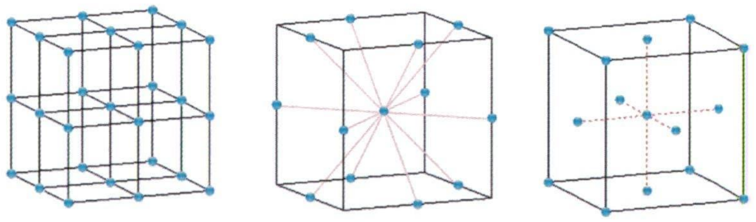


Figure 30. Full factorial, Box-Benkhen and central composite experimental designs

Early DOE work was based on the assumption that experimental results incorporated random noise and consequently, experiments were structured to minimise the effects of this noise. As a result, traditional DOE methods tend to have experiments at the extremes of the parameter space to minimise linear fitting errors, and utilise replication of experiments in order to estimate the variability of the experimental results. Results were assumed as linear or quadratic in nature, allowing polynomial response surfaces to be fitted to approximate the phenomenon under investigation.

With the advent of computer-based simulations, experimental design underwent several changes. The deterministic nature of computer experiments eliminated the need for replication of individual experiments, as replicated simulations would give identical results, rather than being subject to random error. More complex problems could also be investigated, with the result that quadratic approximations were no longer sufficient. As a result, fractional factorial designs were supplanted by space filling designs for computer based experiments, as these distributed data throughout the parameter space, rather than primarily about its perimeter.

A simple space filling design can be created by selecting a suitable number of sample points at random throughout the design space, known as pseudo-Monte Carlo sampling. However, an examination of such a random sample shows clustering of points and large areas that are poorly sampled. Random sampling displays high discrepancy, that is, a significant deviation from a uniform distribution.

4.3.1 QUASI-RANDOM METHODS

One approach to achieving low discrepancy sampling is the Latin hypercube (McKay et al. 1979), which ensures that the parameter space is sampled uniformly by dividing the domain [0,1] along each dimension into n sub-intervals, each of which contains one sample point. In its simplest form in two dimensions, a Latin hypercube distributes the required number of samples equally along the diagonal of the parameter space (Figure 31d). This diagonal distribution is highly correlated and does not sample the space in a uniform manner.

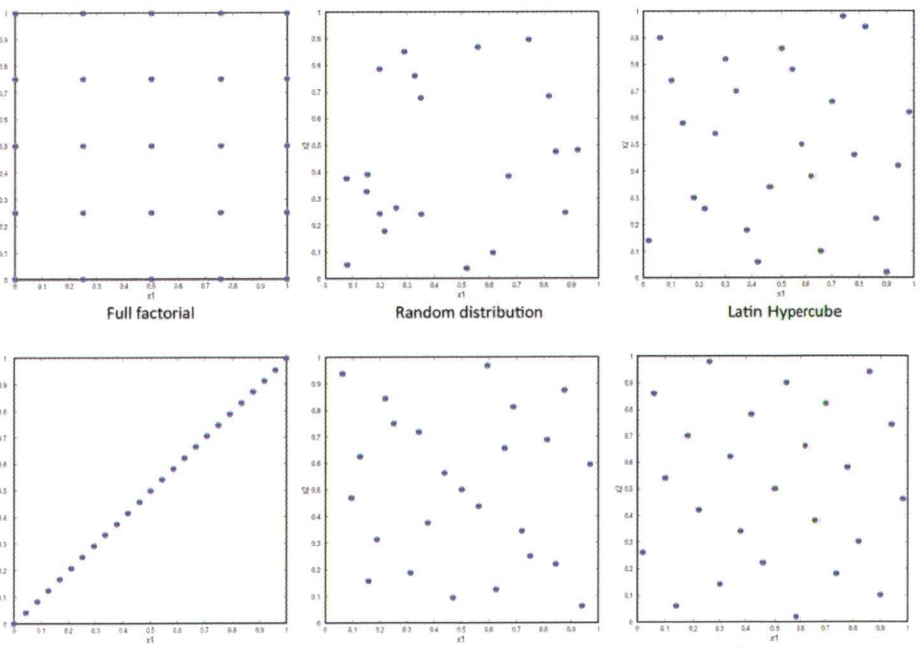


Figure 31. Quasi-random sampling methods

Up to $n!$ valid two-dimensional Latin hypercubes can be generated by permuting the n columns of the diagonal distribution. Determining which of these has the lowest discrepancy becomes increasingly difficult as the number of samples increases, particularly when the parameter space is extended into many dimensions.

Various methods have been proposed by many different authors for generating Latin hypercube samplings having low discrepancy. These include; Maximin Latin hypercubes (Johnson et al. 1990; Morris and Mitchell 1995), Minimal Integrated Least Square Error designs (Sacks et al. 1989), Orthogonal Array based Latin hypercube designs (Tang 1993), Integrated Mean Square Error optimal Latin hypercubes (Park 1994) and the Uniform Design (Fang et al. 2000).

One alternative to variations on the Latin hypercube is the low-discrepancy sequence, which allows an unlimited number of points to be added incrementally to an n dimensional space such that discrepancy is kept low. Such sequences have been proposed by Hammersley, Halton, Sobol, Faure, and Niederreiter, and are detailed in Niederreiter (1992). Figure 31 illustrates several space filling experimental designs, each having 25 sample points, including a 5 x 5 full factorial design, a random sequence, a Latin hypercube, a Sobol low-discrepancy sequence and a Uniform Design. It can be seen that although all sequences span the space effectively, the Uniform Design exhibits a more regular spacing than all other methods.

Hurrion (1999) demonstrated that neural network metamodels based on randomised sampling outperformed metamodels based on conventional regression using full factorial sampling. Similarly, Giunta and Wojtkiewicz (2003) argued strongly in favour of the adoption of quasi-random sampling methods for deterministic, computer based experiments.

As a result, a quasi-random, low-discrepancy DOE method was selected for the sampling of design parameters used to calculate the hull performance metamodel within VESPA. The method selected, the Uniform Design, (Fang 2004), was chosen on the basis of recommendations contained in Notz (2003) and Simpson et al. (2000).

Although low-discrepancy sequences such as those by Sobol (1967) were considered, the regular spacing of the Uniform Design was preferred, as the intervals could be chosen to correspond with portions of the test matrix previously used both for CFD analysis and for tank testing. This made comparison of neural network predictions with existing data easier, as results from multiple sources could be compared on the same graph. If parameter values had fallen on irregular intervals such as would occur with a Sobol sequence, direct comparison between raw CFD data, tank data and metamodel output would not have been possible.

4.4 HULL SHAPE REPRESENTATION

A yacht-design optimisation system must be able to represent different design configurations in a numerical format, in order for these configurations to be varied, evaluated and ranked as part of the search process. There have been many formats used for the numerical representation of hull shapes and these can be broadly categorised as follows:

- A wireframe representation consisting of three-dimensional curves, each approximated by a polygon. Although this method has been widely used for capturing hull shape data from lines drawings or actual vessels, this approach is not currently utilised by any of the CAD systems used specifically for the design of marine hull shapes.
- A wireframe representation consisting of three-dimensional curves defined by a spline function such as a cubic spline, Bezier curve, B-spline or NURBS curve. This approach was used by early ship design systems such as Autokon, (Reenskaug 2003), and Steerbear, (McNaull 1980), which faired hand-drawn hull lines. The method has now been superseded in systems used for ***ab initio*** hull-design by surface based methods.
- Hybrid wireframe/surface models as used NAPA Design (Lengyel 2003). NAPA uses a wireframe definition of sections, waterlines and buttocks as the basis for the automatic construction of a network of surface patches.
- Recent hybrid systems such as the FRIENDSHIP Modeller, (Harries et al. 2003a), and Paramarine, (Bole and Lee 2006), use a network of parametrically defined feature lines, such as those defining stem profile, sheerline, flat of side and flat of bottom, to automatically generate hull surfaces.
- A three-dimensional surface representation as used by programs such as Maxsurf, (Formation Design Systems 2006). These programs typically use a small number of large surfaces, typically subdivision, Bezier, B-spline, or NURBS surfaces to define a hull shape.

Of the above methods, the one most widely used by commercial marine-design software is the NURBS surface model, adopted by interactive design programs such as Maxsurf, Autoship, Fastship, Rhino, Prolines and ProSurf.

INTERACTIVE NURBS SURFACE MODELLING

The use of B-spline surfaces for defining ship hull surfaces was first described by Rogers (1977). Non-uniform rational B-splines (NURBS), a more general form of B-splines, came to prominence with their inclusion in the Initial Graphics Exchange Specification (IGES), first published in 1980. NURBS were included in a commercially available marine hull design system for the first time in 1985, being added to Maxsurf, a program written by the author of this thesis.

A NURBS surface of degree (p, q) is defined by Piegl and Tiller (1997) as:

$$\mathbf{S}(u,v) = \frac{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)\, N_{j,q}(v)\, w_{i,j}\mathbf{P}_{i,j}}{\sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(u)\, N_{j,q}(v)\, w_{i,j}} \qquad 0 \le u, v \le 1 \qquad (4.6)$$

where $N_{i,p}$ and $N_{j,q}$ are the B-spline basis functions, $\mathbf{P}_{i,j}$ are control points, and the weight $w_{i,j}$ is the last ordinate of the homogeneous point $\mathbf{P}_{i,j}$.

The i-th B-spline basis function of degree p, written as $N_{i,p}(u)$, is defined recursively by the Cox-de Boor algorithm as follows:

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \qquad (4.7)$$

A NURBS surface modeller typically presents the user with a network of control points, as shown in Figure 32, which are moved in x, y and z directions to produce the desired hull shape.

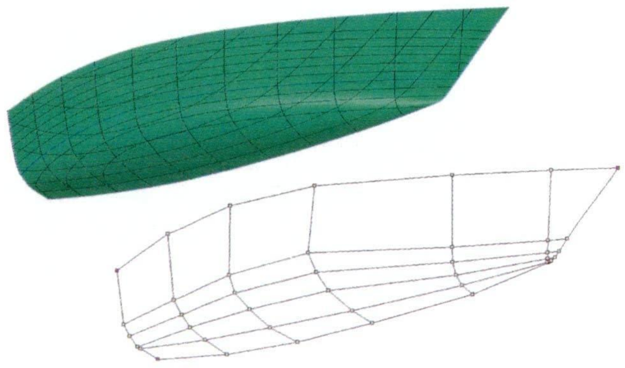


Figure 32. NURBS surface and corresponding control point net

Although this method allows a designer to create almost any shape that can be imagined, the design process is manual and the achievement of a suitable hull shape is highly dependent on the skill of the user.

ALTERNATIVE APPROACHES TO HULL-FORM DEFINITION

Surface based programs have now almost completely superseded wireframe-based programs for the *ab initio* design of marine hull shapes. However, the differing requirements of the designers of large versus small vessels have resulted in the adoption of different solutions to the question of the best process for hull design.

Typically, the designers of large commercial ships commence the design process with a set of hull form parameters. These are derived using an initial design procedure, or estimated from a database of existing vessels. These parameters are used to either automatically generate a suitable hull shape, or to deform an existing design to match the parameter values. In other words, the designers of large ships use the hull form parameters to derive the hull geometry.

Designers of small craft typically work in the opposite direction, creating a unique geometry and deriving the hull form parameters from it. This approach is often required as there is a great deal more variability in the design requirements for small craft, making it more difficult to modify a parent hull to suit. Once a hull shape has been created, its hydrostatic properties may need further adjustment, and changing the hull surface manually to achieve this is a trial-and-error process.

To address this problem, procedures for the parametric transformation of hull surfaces have been created to permit a designer to automatically adjust the hydrostatic parameters of a hull shape to meet requirements.

As a result of the development of parametric transformation functions for surface based hull modellers, the design process for small vessels now more closely matches the approach taken for large ships. The major remaining difference is that while the ship designer may have the luxury of sourcing a parent design from a library of similar existing vessels, the small craft designer will need, in most cases, to first create a suitable parent hull shape.

Interactive hull design software now includes the ability to apply parametric transformations to hull shapes represented as NURBS surfaces. However, during the past ten years an alternative approach has also been developed. This approach is typified by programs such as Paramarine, (Bole and Lee 2006), and the FRIENDSHIP Modeller, (Harries et al. 2003a), which allow for the automatic creation of a hull shape from a set of key parameter values and feature curves.

The development of these two different methods for creating and varying hull shapes has had significant implications for the automated optimisation of marine vessels. In order to choose which is most appropriate for the optimisation of ACC yachts, it is necessary to examine possible hull variation procedures in more detail.

4.5 HULL SHAPE VARIATION

An optimisation algorithm progresses by making small adjustments to parameter values in order to search the available solution space. Each design variation is evaluated and ranked against previous solutions to determine the direction in which the next step in the solution space should be made.

There are several ways in which variation to the hull design of a marine vessel might be achieved within an optimisation algorithm:

- direct manipulation of NURBS surface control points;
- blending of two or more parent hull shapes;
- creation of a hull shape entirely from numerical parameter values;
- deformation of a parent design using a combination of linear and non-linear transformations.

Methods that operate on a parent hull form have an inherent advantage for ACC yacht design optimisation, as they can preserve subtle design features that may be difficult or impossible to capture using an approach that creates a hull shape from scratch. Although this is not a significant problem for commercial ships, the design of racing yachts can involve subtle adjustments that may be crucial to the final performance of the yacht. The precise capturing of a designer's intent at this level of detail may be beyond the capability of a parametric modelling system, or may require so many interrelated parameters that optimisation becomes unwieldy. Conversely, a parent hull captures this design intent precisely, and this can be retained so long as the procedure used for hull variation is well designed.

4.5.1 DIRECT CONTROL POINT MANIPULATION

Although direct control point manipulation is widely used by interactive CAD systems for *ab initio* hull design, it is not ideal as a strategy for shape control in an automated optimisation procedure, due to the large number of variables required. For example, Hendrix et al. (2001) performed an optimisation of the resistance of a Wigley hull using a 5×9 control point mesh, each with x, y, z co-ordinates. By constraining the movement of many of the control points, the available degrees of freedom were reduced from 135 (i.e. $5 \times 9 \times 3$) to 61. No constraints for fairness and convexity were enforced.

While the optimisation of a simple form like a Wigley hull may be feasible using this approach, an ACC hull is more difficult. A typical ACC hull might be modelled with a control point mesh of 10 columns and 8 rows, giving 240 degrees of freedom, with a significant number of constraints required. In particular, fairness and convexity constraints require that movement of adjacent control points be correlated to some extent, as the movement of a single control point would be likely to introduce a bump or hollow in the hull surface.

The large number of parameters and constraints, and the requirement that movements to adjacent control points be correlated, makes successful optimisation of ACC hulls using this approach difficult if not impractical.

4.5.2 BLENDING

Where multiple parent models exist having characteristics that may be appropriate to the final design, it may be possible to perform a weighted blend of two or more parent hulls to create a new design.

Neu (2000a; 2000b) used a barycentric blend of the form:

$$\text{Resultant Ship Hull} = \sum C_n \text{Basis Hull}_n \tag{4.8}$$

where,

$$C_n = \text{Blending Coefficient for Basis Hull}_n \tag{4.9}$$

$$\sum C_n = 1 \tag{4.10}$$

and,

$$0 \leq C_n \leq 1, \tag{4.11}$$

$$n = 1, 2, \ldots, \text{N} \tag{4.12}$$

To perform a barycentric blend, the parent hull shapes must use the same numerical representation, which requires a method for mapping equivalent parameterisations for each hull surface. This limitation does not occur if the hull surfaces being blended are NURBS surfaces, having control point nets of equivalent degree and dimensions. In this case the blending occurs, not between points on the surface, but between vertices in the control point net.

This is illustrated in Figure 33, which shows the effect of blending two bow sections from different ACC hulls. Parent hull A has a form referred to as a Davidson bow, as used by Team New Zealand to win the 2000 America's Cup. This combines a knuckle below the waterline with a steeply upswept forward overhang, designed to maximise sailing length for a given *LBG*. Parent B uses a "destroyer" style bow with shorter overhang and no knuckle.

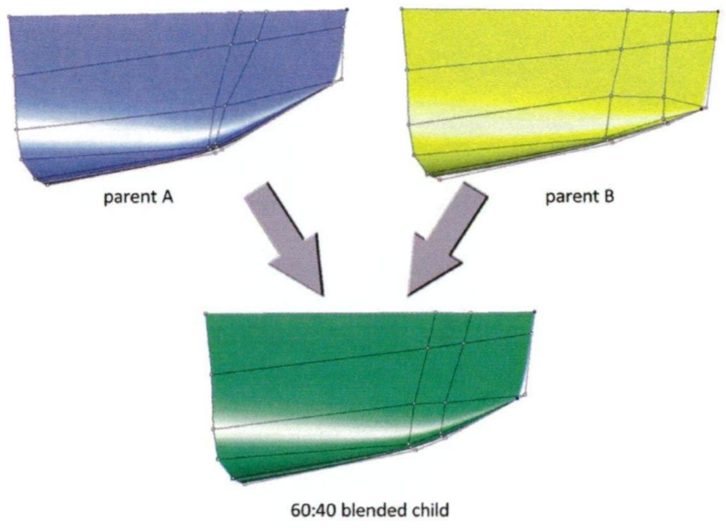


Figure 33. Blending of ACC bow sections

These two shapes are blended together in a 60:40 ratio to obtain the child hull shape. In this case, each control point for the child hull surface is calculated as;

$$\text{Child CP}_{i,j} = 0.6 \text{ ParentA CP}_{i,j} + 0.4 \text{ ParentB CP}_{i,j} \tag{4.13}$$

The blending approach allows existing hulls with known qualities to be used as parents. If the parent hulls are fair and the mapping of equivalent points on each hull is good, derived hull shapes should also be fair. However, the barycentric blending method suffers from the problem that variations to hull parameters are not independent. The blending coefficient C_n applies to each hull in its entirety, and as a result hydrostatic parameters of the blended hull are fully correlated to one another.

As an example, assume two parents, one with both high C_P and high C_M, and one with low C_P and low C_M. In this case, there is no way to blend the two parents to achieve a child hull that combines a high C_P with a low C_M, nor is it possible to achieve a hull form that has a low C_P with a high C_M.

This is a significant limitation for the use of the blending approach as a variation method within an optimisation system, as it prevents large portions of the design parameter space from being explored.

Although the usefulness of blending as the sole variation method is limited, the approach does have benefits when combined with other methods such as parametric transformations. This is particularly true when used within an optimisation system based on an evolutionary algorithm, where the blending method may be used as a form of recombination operator (see Section 6.4.2, Recombination), allowing variations on geometric features that are independent of the hull form's dimensional and hydrostatic parameters, to be explored effectively.

4.5.3 PARAMETRIC MODELLING

Parametric modelling, as elaborated in Harries and Abt (1999b) and Harries et al. (2001a), defines hull shapes using a set of key parameters and constraints. These are satisfied simultaneously by a search algorithm, which gives as its output a hull shape that matches all the required values.

The parameters specified may be overall dimensions, such as L_{WL}, B_{WL}, T_c; they may be hydrostatic values, such as L_{WL}, B_{WL}, T_c, V, L_{CB}, C_M, C_P and C_B; or they may be positions, slopes and curvatures of individual defining curves, such as a stem profile or sheerline. An example of the parameters that might be specified for the design of an ACC hull is shown in Table 4. In this case, an ACC hull was created within the FRIENDSHIP parametric modeller using 44 parameters.

Table 4. Parameters Used to Describe an ACC Hull, From Harries et al. (2001)

```
name                       Kiwihunter

// parameters of layer 2 (optional)
MeterClass                  24.0          // measurement value
S                          280.0          // sail area
DSP                         19.5          // displacement
lcBuoy                      10.50         // longitudinal center of boyancy
lcFlot                      11.00         // center of flotation
lcLatArea                   10.00         // center of lateral area canoe body

// parameters of layer 1 (required)
fairFlag                    0             // fair skinning switch
nosec                       11            // number of sections
noVerticesPerSection        8             // vertices per section
useFlatInterpol             0             // intial velocity of sections switch
atCurveParameter            0.2           // parameter of flat interpolation

// keel contour
design_elevation            0.2           // IACC - measurement level
design_length              20.0           // lenght at measurement level
design_draft                0.76          // maximum draft of canoe body
design_draft_x              9.0           // position of max. draft

incline_bow                10.0           // stem contour modifier
incline_stern               5.0           // stern contour modifier

overhang_bow                1.5           // overhang at bow
overhang_stern              2.0           // overhang at stern

design_freeboard            1.2           // constant freeboard (for simplicity)

// deck
beam_bow                    0.3           // beam at bow
maxbeam                     4.8           // maximum beam
maxbeam_x                  11.9           // position of maximum beam
beam_stern                  2.8           // beam at stern
angle_bow                  14.0           // angle of waterlines at deck
angle_stern                13.0           // angle of waterlines at stern

// flare at deck
deck_flare_bow              8.0           // at stem
flare_change_bow            0.0           // gradient at stem
deck_flare_max_beam         0.0           // at maximum beam
flare_change_stern         30.0           // gradient at stern
deck_flare_stern           20.0           // at stern

// deadrise
deadrise_bow                0.0           // ...
deadrise_change_bow         0.0           // ...
deadrise_max_draft          0.0           // ...
deadrise_change_stern       0.0           // ...
deadrise_stern              0.0           // ...

// flat of side
flat_bow                    0.25          // value at stem
flat_change_bow             3.0           // gradient at stem
flat_max_beam               0.5           // value at max. beam
flat_change_stern          -5.0           // gradient at stern
flat_stern                  0.28          // value at stern

// actual waterline
dwl_max_beam                3.8           // maximum beam
dwl_max_beam_x             10.6           // position of max. beam
dwl_tangent_bow_dist        0.1           // waterline entry modifier
```

Parametric modelling is an ideal approach to use on simple geometric shapes. In the case of the optimisation of semi-submersible oil rigs, Birk (2005) was able to define the geometry of the semi-submersible pontoons and struts with a small number of parameters. For commercial ships that have flat-of-bottom and flat-of-side joined by a bilge radius, the number of parameters increases. However, the ship can still be defined with a relatively small number of variables.

ACC hull shapes, on the other hand, have specific geometric features, which are consequences of the locations at which the hull is measured. The resulting hull shapes have bumps and flats in unusual places to gain the greatest benefit under the ACC rule, and the specific form of these features is subject to the preferences of the individual designer.

These geometric peculiarities make it difficult for a parametric modeller to form hull shapes that meet the requirements of the designers involved in ACC design. As an example, an ACC hull shape resulting from an optimisation study performed by Harries ((2001a)) using the FRIENDSHIP Modeller is shown in Figure 34.

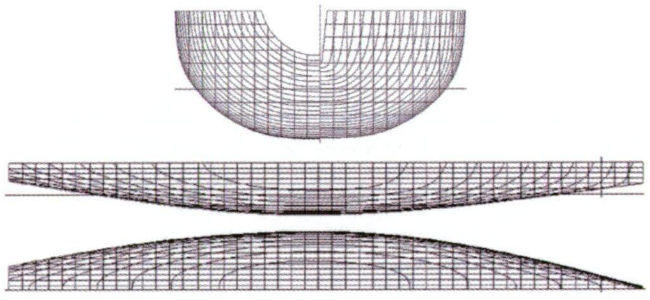


Figure 34. An ACC canoe body optimised using FRIENDSHIP, from Harries et al. (2001)

Although the parametrically modelled hull is fair and, accordingly, is likely to be hydrodynamically efficient, the hull bears little resemblance to actual ACC hulls. The plan and profile view of Harries' hull shape is compared with a typical 2007 ACC hull in Figure 35. Note that the hulls in this illustration have been compressed along the longitudinal axis by 50% in order to emphasise shape differences. It is clear from the angular nature of the modern ACC hull that there are many design features that have not been captured by the parametric modeller.

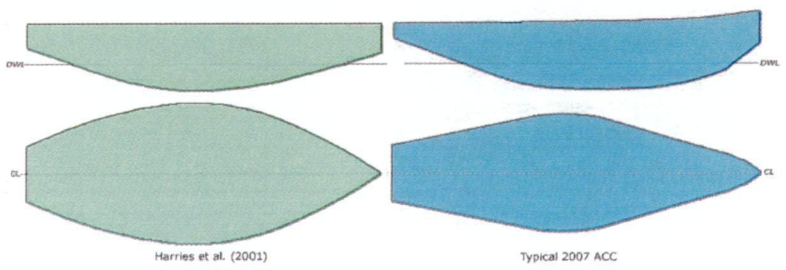


Figure 35. Comparison of parametrically modelled ACC hull with typical 2007 ACC design

Baik and Gonella (2005) also used the FRIENDSHIP modeller to optimise ACC hulls, choosing to specify the hull shape using 51 parameters. Once again, hulls produced by the FRIENDSHIP parametric modeller, although showing more realistic cross-sectional shapes than achieved by Harries, failed to capture significant design features.

In addition, Baik and Gonella failed to properly constrain for hull surface convexity, a requirement under the ACC rule, as well as neglecting to constrain the hull dimensions to satisfy the fore and aft girth requirements. This resulted in unrealistic hull forms, as shown in Figure 36.

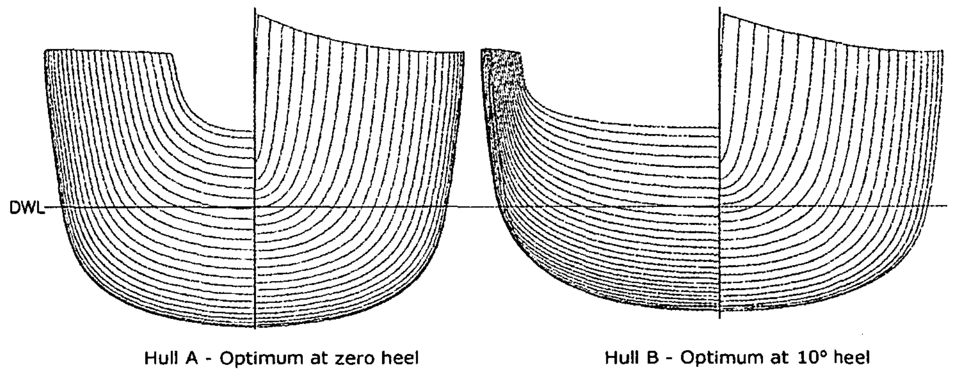


Figure 36. Cross sectional shapes of optimal hulls, from Baik and Gonella (2005)

Hull A, the optimal shape found using SHIPFLOW for the zero heel case, has a very narrow beam and a particularly narrow transom, indicating that its aft girth measurement would be less than the maximum permitted. Hull B, the optimum found using SHIPFLOW for the 10° heel case, has an extremely wide transom and large aft girth measurement, and would clearly not measure as a legal ACC hull.

It is likely that the parametric modelling of a realistic ACC hull shape, although not out of the question, would take substantially more than 50 parameters in order for the shapes produced to be acceptable to current yacht designers involved with the class. The use of such a large number of parameters and constraints increases the complexity of the optimisation process and result in slower analysis and convergence.

Although parametric modelling is a promising technology, particularly in the realm of commercial ship design, the complexity of the method and the difficulty of achieving hull shapes that are acceptable at the highest levels of ACC yacht design resulted in it being an unattractive option for this research work.

4.5.4 PARAMETRIC TRANSFORMATION

Simple linear transformations, such as the scaling of length, beam and depth can be applied to a hull shape. However, this is seldom sufficient to achieve the variation required for optimisation. To achieve variation of hydrostatic parameters such as L_{CB}, C_B, C_P and C_M, it is necessary to apply non-linear transformations to the hull geometry.

Several schemes have been proposed to perform these non-linear transformations. One of the first descriptions of a technique to parametrically modify the L_{CB} and C_P of an existing lines plan was provided by Lackenby (1950). Lackenby's method consisted of a technique for moving sections of the hull forward and aft according to a quadratic function to match prescribed parallel mid-body length and location as well as L_{CB} and C_P values. However, other parameters such as ∇, water-plane area and L_{CF} could vary in an uncontrolled manner.

Lackenby's method did not attempt to modify the sectional shape of the vessel, with C_M remaining constant and changes to C_B limited to what could be achieved through longitudinal changes to the volume. Although variations on Lackenby's method have been proposed by various authors, (Volker 1954; Söding and Rabien 1977), these refinements did not address the lack of control over transverse hydrostatic properties, limiting the usefulness of the approach for ship hull optimisation procedures.

The introduction of NURBS surface modelling systems for the definition of hull shapes introduced an additional difficulty. Lackenby's method operates directly on hull sections, with intermediate sections needing to be interpolated and faired. This was a straightforward process when the hull definition was stored as a lines plan or as a three-dimensional wireframe. However, a NURBS surface is defined by the locations of its control points and its hull sections are calculated as required from this surface definition.

For NURBS surfaces it is not possible to move hull sections directly; rather, it is the control points for the surface that need to be adjusted. In order for a transformation such as Lackenby's to be applied to NURBS surface models, it is necessary for the surface control points to be repositioned by the transformation.

Markov and Suzuki (2001) addressed this issue, moving columns of control points based on the output of a Davidson-Fletcher-Powell optimisation algorithm. This approach is not necessarily fairness preserving, whereas the suggestion of Halley (1987) that the transformation be based on a smooth piecewise cubic polynomial in the form of a cubic spline results in fair deformations of the parent hull.

FREE FORM DEFORMATION

One potential solution to the problems inherent in Lackenby's method is the use of a parametric transformation procedure based on the Free Form Deformation (FFD) of Sederberg and Parry (1986).

FFDs have been widely used by the computer graphics and animation industry to permit objects to be deformed in a fluid manner, but have only recently been adapted for use in design and optimisation. Peri and Campana (2005a) successfully used FFD transformations of a cruise-ship hull form in order to optimise its sea-keeping qualities. Menzel et al. (2005) investigated the use of FFDs in the evolutionary optimisation of turbine blade aerofoils, finding that they resulted in a less complex genome and improved optimisation performance.

FFD DETAILS

An FFD consists of a trivariate Bernstein polynomial, such as a Bezier, B-spline or NURBS volume, in which the shape that is to be deformed is embedded. Deforming this volume by moving one or more control points results in a corresponding change to the embedded shape.

To create an FFD as defined by Sederberg and Parry, a local co-ordinate system is imposed on a local parallelepiped region, known as a lattice space.

A point X within the lattice space has (s, t, u) coordinates in this system such that:

$$\mathrm{X} = \mathrm{X}_0 + s\mathrm{S} + t\mathrm{T} + u\mathrm{U} \tag{4.14}$$

where

$$0 < s < 1, \qquad 0 < t < 1, \qquad 0 < u < 1$$

The (s, t, u) coordinates of X can be found using linear algebra. A vector solution is:

$$s = \frac{\mathrm{T} \times \mathrm{U} \cdot (\mathrm{X} - \mathrm{X}_0)}{\mathrm{T} \times \mathrm{U} \cdot \mathrm{S}}, t = \frac{\mathrm{S} \times \mathrm{U} \cdot (\mathrm{X} - \mathrm{X}_0)}{\mathrm{S} \times \mathrm{U} \cdot \mathrm{T}}, u = \frac{\mathrm{S} \times \mathrm{T} \cdot (\mathrm{X} - \mathrm{X}_0)}{\mathrm{S} \times \mathrm{T} \cdot \mathrm{U}} \tag{4.15}$$

A grid of control points $\mathbf{P}_{i,j,k}$ is imposed on the parallelepiped, forming $l + 1$ planes in the S direction, $m + 1$ planes in the T direction, and $n + 1$ planes in the U direction.

$$\mathbf{P}_{ijk} = \mathrm{X}_0 + \frac{i}{l}S + \frac{j}{m}T + \frac{k}{n}U \tag{4.16}$$

where

$$i = 0, \dots, l; \quad j = 0, \dots, m; \quad k = 0, \dots, n;$$

An example of a tessellated sphere enclosed by a lattice space defined by $\mathrm{S}, \mathrm{T}, \mathrm{U}$ directions, with its origin at X_0, and having an $l = 2, m = 2, n = 2$ control-point grid, is shown in Figure 37.

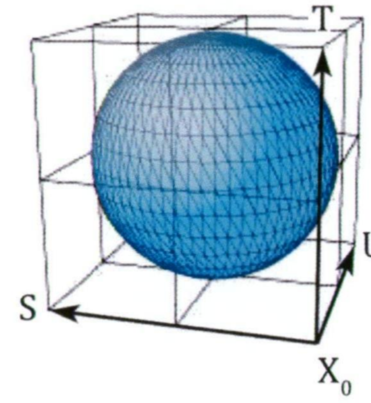


Figure 37. FFD lattice enclosing a tessellated sphere

A deformation to the embedded shape is performed by moving a lattice control point $\mathbf{P}_{ijk}$ from its undisplaced position in the lattice. In this case the deformation function is defined by a trivariate tensor-product Bezier volume, (Bezier 1974). An arbitrary point $\mathbf{X}$ within the lattice space is found by first calculating its (s, t, u) coordinates using equation 4.15, with the deformed position $\mathbf{X}'$ being calculated using:

$$\mathbf{X}' = \sum_{i=0}^{l} B_i^l\,(s)\left[\sum_{j=0}^{m} B_j^m\,(t)\left[\sum_{k=0}^{n} B_k^n\,(\mathrm{u})\mathbf{P}_{ijk}\right]\right] \tag{4.17}$$

where $\mathbf{X}'$ is a vector containing the Cartesian coordinates of the displaced point, $\mathbf{P}_{ijk}$ is a vector containing the Cartesian coordinates of the control point, and B is a Bernstein polynomial of the form:

$$B_k^n = \binom{n}{k} u^k (1-u)^{n-k} \tag{4.18}$$

and

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!} \tag{4.19}$$

Such a deformation is shown in Figure 38, which shows the displacement of control point $\mathbf{P}_{i=1,j=2,k=1}$ from the FFD lattice, with the resulting deformation of a sphere embedded in the lattice space.

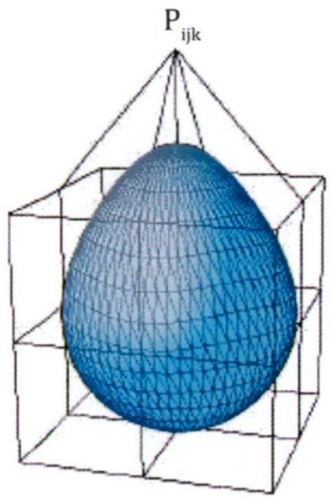


Figure 38. Deformation of an FFD lattice

Sederberg's definition of an FFD utilised a tensor product volume based on the Bezier formulation. However, an FFD can also be created using B-splines or NURBS. Using NURBS provides an additional element of control over the Bezier and B-spline formulations, as it is possible to deform the embedded object by changing the weight w of each control point, rather than modifying the control point's x, y, z coordinates.

In this case, the deformed position $\mathbf{X}'$ for an arbitrary point $\mathbf{X}$ within the lattice space, given its (s, t, u) coordinates, can be found using:

$$\mathbf{X}' = \frac{\sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{\mathrm{k}=0}^{\mathrm{n}} N_{i,p}(s)\, N_{j,q}(t)\, N_{k,r}(u)\, w_{i,j,k}\mathbf{P}_{i,j,k}}{\sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{\mathrm{k}=0}^{\mathrm{n}} N_{i,p}(s)\, N_{j,q}(t)\, N_{k,r}(u)\, w_{i,j,k}} \tag{4.20}$$

where $\mathbf{P}_{i,j,k}$ are control points, the weight $w_{i,j,k}$ is the last ordinate of the homogeneous point $\mathbf{P}_{i,j,k}$, and where $N_{i,p}(s)$, $N_{j,q}(t)$ and $N_{k,r}(u)$ are the B-spline basis functions of degree p, q and r.

Modifying the control point weight values has several advantages in the context of the parametric transformation of marine hull surfaces, as these typically require the preservation of fairness and tangency rather than the introduction of geometric features that would compromise the hydrodynamic properties of the vessel.

A simplified example is shown in Figure 39, where a hull section, in this case a circular arc, is shown mapped into the unit square. A conventional FFD defined by a NURBS bicubic patch is shown top right, with its control points adjusted to bring about a deformation of the patch along with its embedded curve. This FFD, while providing quite powerful shape modification capabilities, has 32 degrees of freedom (i.e. x and y coordinates for 16 control points).

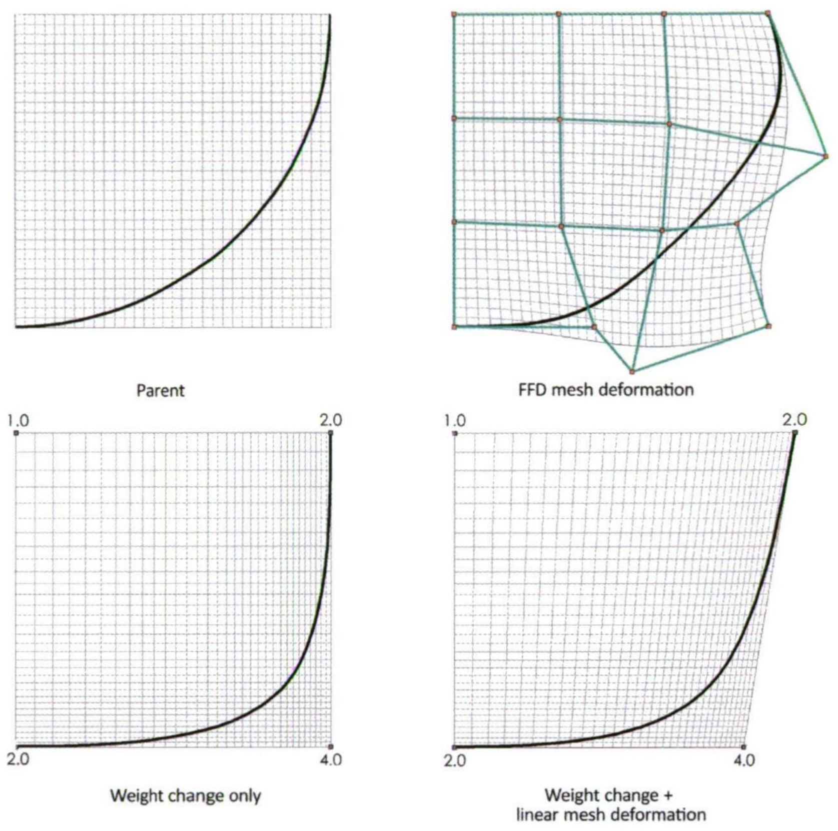


Figure 39. Free Form Deformation of an embedded curve

An alternative to the conventional FFD is shown in Figure 39 bottom left. In this case, a bilinear NURBS patch is defined as the FFD surface and 3 of its 4 control points have their weight values increased. As the control-point weight values along the bottom and right edge of the surface are increased, the parametric spacing of the surface compresses in the direction of the increased weights, and the embedded curve distorts in a fair manner.

If the embedded curve in this example is regarded as representing the midship section of a yacht, the combination of weight change with deformation of a linear FFD achieves effective variation of B_{WL}, T, C_M and topside flare, while automatically constraining flat of floor, convexity, fairness and tumblehome.

The effect of the increase in weight values in Figure 39 bottom left is to increase the C_M of the section without changing the tangent direction of the section at the hull centreline and sheerline.

By moving one FFD control point in the x direction, as shown in Figure 39 bottom right, a measure of topside flare can also be introduced. These examples show that it is possible to make functional changes to a hull section by changing only 4 parameter values, while maintaining fairness, convexity, tumblehome and flat of floor constraints.

The use of weight changes and small control point movements within an FFD can provide excellent control over the longitudinal form parameters of a hull design. Several examples are illustrated in Figure 40, which shows a half-plan view of a yacht's sheerline and waterline undergoing various deformations by an FFD.

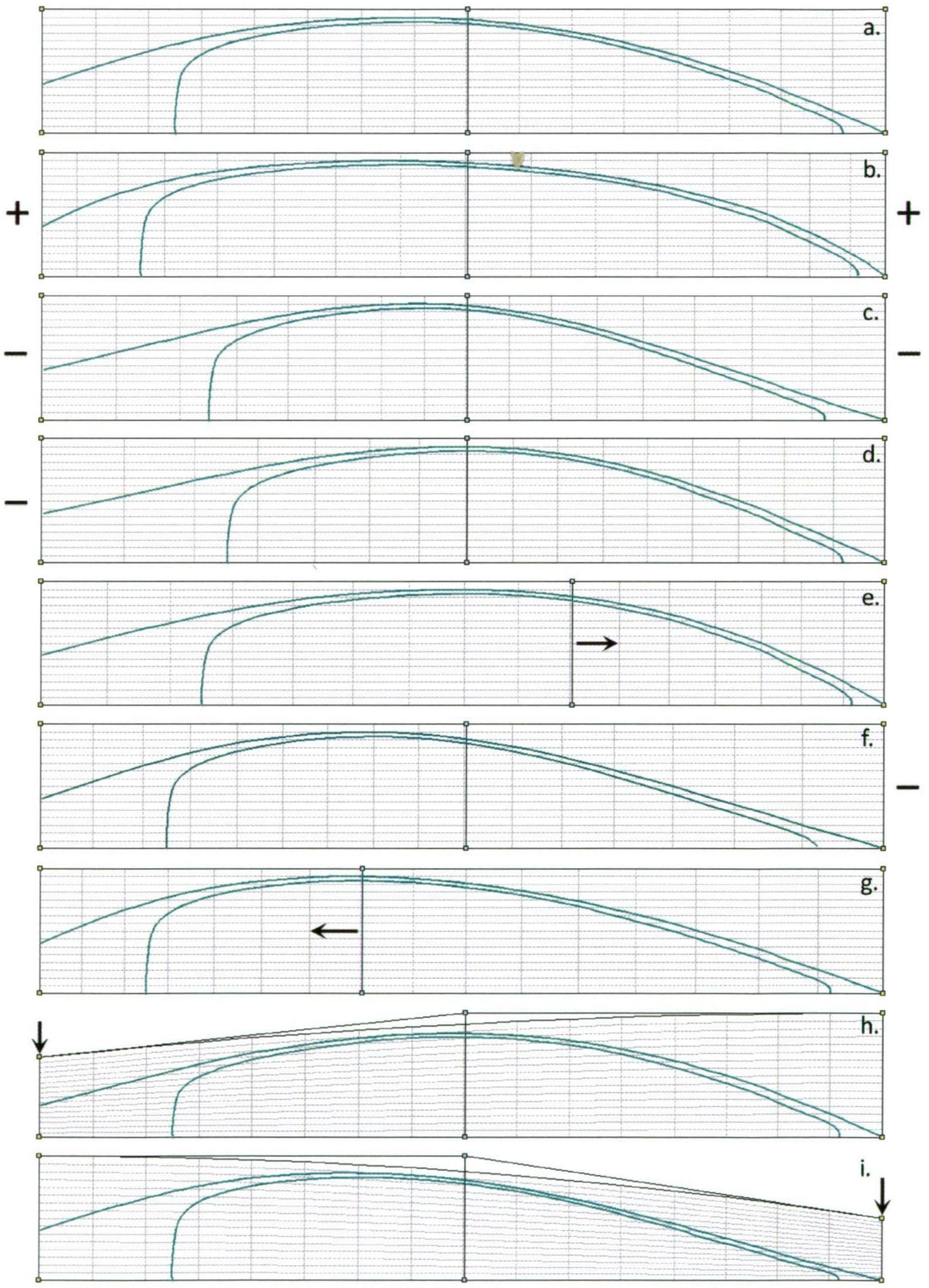


Figure 40. Effects of FFD weight and control point changes on longitudinal volume distribution

Figure 40 (a) shows the original hull lines overlaid onto an undistorted FFD. Figure 40 (b) and (c) show how increasing or decreasing the weight values at each end of the FFD changes the volume distribution, effectively increasing or decreasing the C_P, C_{WP} and C_{IL} values for the hull.

Reducing weight values at the aft end of the hull (d) results in a shift of L_{CB} and L_{CF} forward, as does forward movement of the centre column of FFD lattice control points (e). Analogous shifts aft of L_{CB} and L_{CF} are shown in (f) and (g). Note that the weight changes in (d) and (f) have very similar effects on the hull lines when compared to the control point shifts of (e) and (g) respectively.

The effect of small transverse movements of FFD control points is shown in (h) and (i). This direct control over the fineness of the ends of the hull is directly applicable to the design of ACC yachts, due to the restrictions in the ACC rule regarding maximum girth limits at the fore and aft girth stations.

4.5.5 VESPA HULL SHAPE TRANSFORMATION

Based on the strengths and weaknesses of the methods investigated, a hierarchical FFD based parametric transformation method was chosen to perform the hull variation required. One of the most important advantages of such a system is that a known parent hull, encapsulating the designer's knowledge and preferences, is used as the basis for any design variation.

An existing NURBS surface representation of an ACC design contains an enormous amount of information, sometimes the result of years of research and millions of dollars in development costs, and it is important that design features be preserved wherever possible. If an approach that involved creating each hull solely from parameter values were to be adopted, it would not necessarily be easy to reproduce the parent hull design's features to the precision required.

It is important that transformed hulls do not sacrifice fairness, convexity or flatness where required. As the applied transformations are curvature continuous, they are fairness preserving. Constraints are also applied to the transformations to ensure that the result also preserves convexity or planarity requirements. These are determined by examining the sign of the principal curvatures at multiple points on the NURBS surface.

Hull shapes resulting from the transformation must also meet the requirements of the ACC rule. These include the measurement of length in a plane 200mm above the water plane, a total displacement of 24,000 kg, a limit on girth values at the FGS and AGS stations, and a requirement that the hull be convex in all directions.

Finally, it is important that the transformation method used be able to function using a small number of design variables. Such parsimony in the description of transformations to a hull shape is of great importance to a procedure used for design optimisation, as the number of parameters used to describe variations in hull design, directly affect the speed and efficiency of the optimisation process.

HIERARCHICAL FFDS

Although a global 3D FFD lattice gives good control of longitudinal hull form parameters such as L_{CB}, L_{CF}, C_P and C_{WP}, it does not necessarily give precise control of the transverse sections of the hull. One problem is illustrated in Figure 41, where a section shape is distorted by changing the weights of the global FFD. Rather than the section being tightly bounded by the limits of the FFD, as was the case in the examples shown in Figure 39, the curve in this example is significantly smaller than the bounding FFD. In this case, although the sectional shape is deformed in the correct way to increase the C_M of the hull, the vertical location of the section is affected, as is the vertical and horizontal scale.

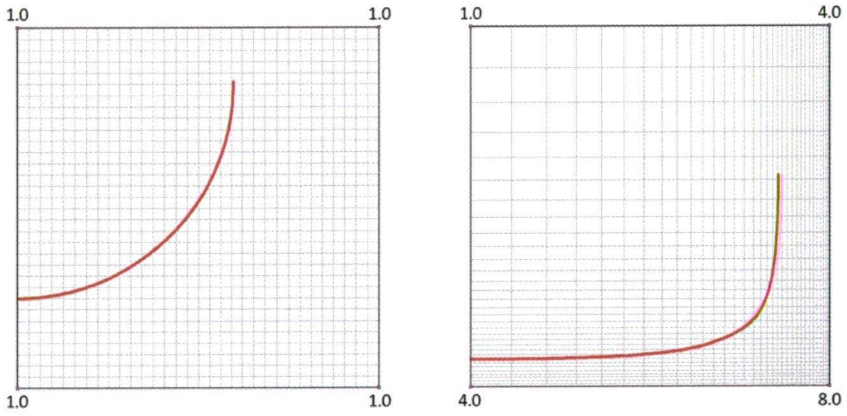


Figure 41. Deformation of hull section using a global FFD

These unwanted side effects have a significant effect on the ability of the FFD to perform meaningful changes to the hull shape, as changes to one parameter, such as C_M, would have an effect on the draft and beam of the section, as well as the shape and location of the hull centreline profile and sheerline.

In order to avoid this problem, a hierarchical FFD has been created in order to provide precise control over hull sectional shape. In this method, the global 3D FFD lattice encloses a series of local planar FFD lattices that lie in the planes of the transverse columns of the hull surface control-points. Each column of control points is projected into its own local FFD. Changes to the shape of these control point columns are dependent on the deformations applied to the local FFDs by the global FFD.

As each column of control points is mapped onto the unit square, changes to the column are constrained to this space. This means that the vertical and horizontal dimensions of the embedded control point column do not change during a local FFD transformation based on variation of the local FFD control point weights.

An ACC parent hull surface, with its enclosing global FFD and a series of planar FFDs located at the position of each hull surface control-point column is shown in Figure 42, together with a transformed version of the parent design. Changes have been made to C_P, L_{CB}, B_{WL}, C_M and flare, while constraining AG and FG.

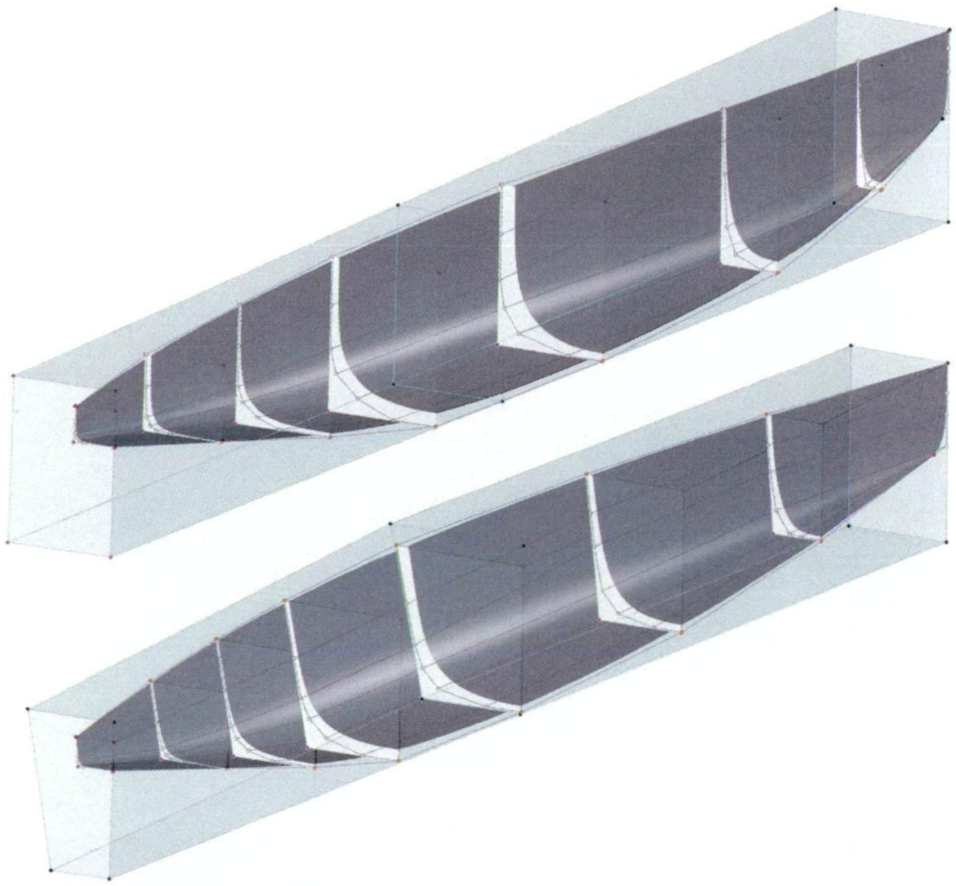


Figure 42. Transformation of an ACC hull using a hierarchical FFD

Variation of hull parameters can occur in several ways using this approach:

- Changes to the length, beam and depth of the hull are applied via scaling of the global 3D FFD lattice.
- Changes to longitudinal volume distribution are achieved by adjusting the weights of lattice control points in unison, to smoothly deform the FFD and its embedded surfaces. This can be seen in Figure 42, which shows a hull shape before and after parametric transformation using an FFD. Changes to the

longitudinal volume distribution in this case include a reduction in C_P and an aftward shift of L_{CB} and L_{CF}.

- Changes to the fore and aft taper of the hull can be adjusted by moving the outer control points at each end of the global FFD in a transverse direction. This feature is useful for ACC design due to the constraint on girth length at the fore and aft girth stations. To achieve maximum girths it is necessary to be able to adjust the beam of the hull independently at the ends relative to the middle.

  This feature also allows some control of L_{CF} independently of L_{CB} position. If the aft end of the global FFD is made wider while the control point weights at the aft end of the global FFD are reduced by a suitable amount, the L_{CF} can be moved aft, while the L_{CB} is held stationary. If these variations are changed appropriately, L_{CF} can be held stationary while L_{CB} is moved forward.

- Changes to the fullness of each section are achieved by modifying the control-point weight values of each local 2D FFD as illustrated in Figure 39. The original hull in Figure 42 has a high C_M, which is reduced somewhat in the transformed hull by reducing the weight values on the outer and lower edges of each planar sectional FFD.

- Changes to hull flare are achieved by vertical tapering the global FFD so that the top of the lattice is wider than the bottom in the areas where flare is to be introduced. This in turn modifies the shape of each of the embedded 2D FFDs, tapering each one by different degrees based on its longitudinal location in the global FFD. This is also illustrated in Figure 42, where the vertical topsides of the upper hull are flared in the transformed hull.

  Note that the degree of flare throughout the hull is controlled by the movements of only one or two control points in the global FFD. Flare can be introduced amidships and tapered out fore and aft, or it can have its maximum at the transom and reduce forward, but in each case the changes are smooth and occur automatically.

4.6 SUMMARY

Following a review of the alternatives, the methods adopted for use in VESPA included:

- sampling based on the Uniform Design;
- hull geometry representation using NURBS surfaces;
- a parametric transformation of hull shapes based on a hierarchical FFD;
- a neural network based metamodel.

5 · PERFORMANCE EVALUATION

Chapter 4 described the way in which hydrodynamic data are sampled and analysed for a variety of hull shapes, with the results encapsulated for rapid retrieval in a neural network based metamodel for use by VESPA.

When combined with an aerodynamic model for an ACC sail plan, this hydrodynamic data may be used to calculate actual sailing performance for any yacht design generated during the exploration of the design space. This creation of sailing performance data is the role of the Velocity Prediction Program or VPP.

Once VPP data can be produced for multiple yachts, a Race Modelling Program, or RMP, may be used to perform a simulation of one or more boats sailing a specific race course, having a set number and location of turning marks, in a stochastically derived wind field. Boats may be raced against one another multiple times using the RMP, with the results tallied to give a probability of each boat winning against a fleet of others.

The design of the RMP is crucial to the success of the overall optimisation system. The RMP must include an appropriate level of detail in its simulation and should not expend valuable time simulating features that are not relevant to the determination of yacht performance.

5.1 VPP

A Velocity Prediction Program (VPP) calculates the performance of a yacht when sailing. Modern VPPs are primarily based on the work performed in the Pratt project at MIT reported by Kerwin (1978). VPPs have undergone widespread development due primarily to their incorporation in the International Measurement System rule (IMS 2004) for racing yachts. Improvements have been proposed by various authors (Letcher 1974; Milgram 1993; Schlageter and Teeters 1993; van Oossanen 1993; Oliver and Claughton 1995; Claughton and Oliver 2003).

Given true wind direction β_{TW} and wind velocity $V_{TW,}$ a VPP balances aerodynamic forces: lift, drag and heeling moment, against the hydrodynamic forces, lift (from the keel), resistance and righting moment, in order to determine the equilibrium velocity for the yacht. Some VPPs also balance aerodynamic yaw moment against the corrective moment generated by the lift produced by the yacht's rudder.

Forces acting on a yacht sailing upwind are shown in Figure 43. Note that the aerodynamic thrust vector T_A and the hydrodynamic resistance vector R_H are equal and opposite, indication that the hull velocity is at equilibrium. Similarly, the transverse component of the aerodynamic sideforce vector SF_A and the hydrodynamic lift vector L_H are in balance.

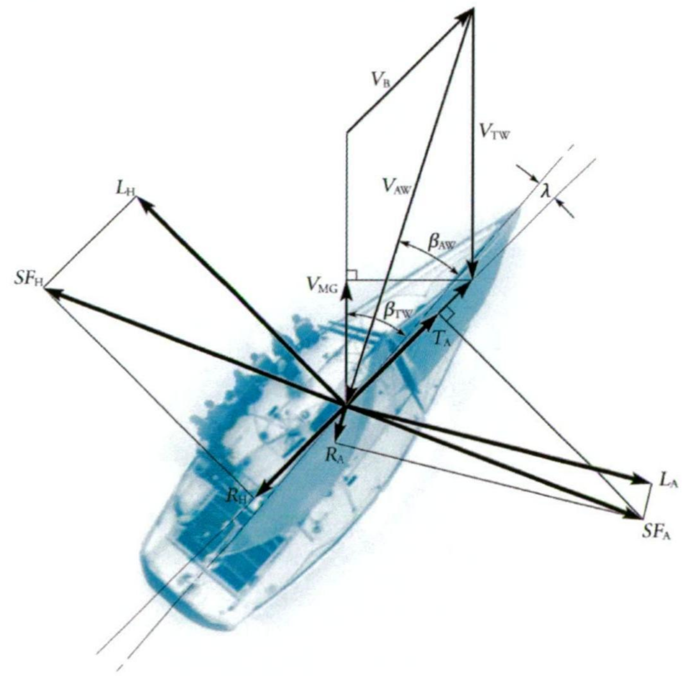


Figure 43. Forces acting on a sailing yacht

In addition, heeling moment from both the aerodynamic and hydrodynamic lift vectors must be balanced by yacht's righting moment. This righting moment is the displacement of the vessel multiplied by the lever arm between the vessel's heeled transverse centre of gravity and its transverse centre of buoyancy.

Inputs to a typical VPP include either hull lines, which are processed to determine key dimensions, or direct input of those dimensions in tabular form. Additionally, a VPP requires information regarding hull stability, mast and sail dimensions, as well as details of appendage sizes and shapes. Outputs from a VPP for a given β_{TW} and V_{TW} typically includes hull velocity V_B, apparent wind angle β_{AW}, apparent wind velocity V_{AW}, heel angle, heeling moment and righting moment. A VPP may also provide a value for leeway angle λ.

A VPP may also provide information about the adjustments made to the sail force model in order to balance the aerodynamic and hydrodynamic forces. The three values commonly used by VPPs for adjusting sail configuration in order to optimise boat speed for a given point of sail:

- **Flat,** a value that varies from 0.0 to 1.0 and is analogous to the degree of reduction in the camber of the sails and therefore the amount of lift produced. Flat tends to be applied at higher wind velocities for $\beta_{TW} < 60°$.
- **Reef,** a value that varies from 0.0 to 1.0 and is equivalent to a reduction in luff length of the mainsail. The reef parameter is used to lower the centre of effort of the sails in order to reduce heeling moment. Reef tends to be applied at higher wind velocities for $\beta_{TW} > 60°$. For ACC yachts, the reef parameter is not used to change mainsail parameters, as mainsails are always used at full hoist, although it may be used to reduce headsail area by reducing foot length.
- **Twist,** analogous to allowing the head of a sail to twist to reduce power. Permits a reduction in heeling moment by changing the spanwise lift distribution of the sails. This is of value when a rig needs to be depowered without lowering the aspect ratio of the rig or reducing the lift coefficient of the lower portion of the sails.

In the case of VESPA, the VPP used was provided by the Alinghi design team. Named PAP and written by Manolo Ruiz de Elvira of Nautatec, this VPP has been developed over a period of 10 years and was first used by the Bravo España Challenge for America's Cup 2000. PAP operates with two degrees of freedom, finding an equilibrium solution by balancing forward thrust against drag, and heeling moment against righting moment. No yaw balance is performed by PAP.

Although it is possible to incorporate a calculation of added resistance in waves within the VPP, this has not been included for use in PAP at this time, as the small amplitude and high frequency of the seas off Valencia does not create sufficient variation in added resistance to justify the additional computational effort.

In order for PAP to be integrated into the VESPA optimisation system, modifications were required to allow it to use neural-network metamodels for hydrodynamic data. In addition, PAP was modified to allow it to be called as a dynamically linked library (DLL) component within the VESPA system.

5.2 VESPA RACE MODELLING PROGRAM

Prior to reviewing the choices made in the design of the VESPA Race Modelling Program, it is worthwhile describing some of the many possible interactions between boats, each of which may influence the outcome of a yacht race.

5.2.1 MATCH RACING TACTICS

Interactions between yachts may take the form of right-of-way situations, where one yacht must keep clear of another, losing ground in the process. A yacht upwind of another also has the ability to disturb the flow of air encountered by another yacht's sails, reducing the performance of the leeward yacht. Alternatively, the threat of disturbed air may result in an affected yacht choosing a less favourable course.

In order to simulate a match race with sufficient accuracy it is necessary to understand these interactions and the significance of their effects on the race outcome. In some cases, the effects are minor and are not relevant to the determination of the faster design. In other cases, the effects are important but may be approximated in a less complex manner than performing a detailed simulation.

The following list of interactions between boats is by no means exhaustive, and does not include the myriad tactical manoeuvres that make up a match-racing helmsman's arsenal. However, the situations described are some of the most significant in their effect on the outcome of typical match races.

WIND SHADOW AND BACKWIND

When yachts sail in close proximity, each boat disturbs the wind field that it passes through, potentially affecting the wind encountered by other yachts. This is illustrated in Figure 44.

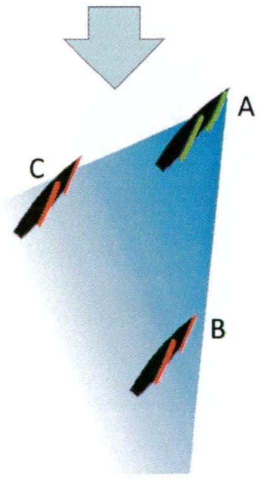


Figure 44. Wind shadow effect of leading boat, A

Boat A creates a wind shadow that extends to leeward in the direction of the apparent wind, as well as behind and slightly to windward of the yacht's centreline.

This wind shadow is made up of two components; turbulence, which extends to leeward of the yacht for a distance that is dependent on the height of the rig; and upwash, the local distortion of the wind field flowing over the yacht's sails.

Yacht B is directly affected by the wind shadow extending downwind from yacht A. Yacht C, while not affected by turbulence created by the rig of yacht A, is affected by its upwash, seen as a small veer in the wind direction. This effect, commonly referred to as backwind or lee-bow effect, results in yacht C not being able to point as high as yacht A, resulting in a loss of ground.

Safe leeward

Although disturbed wind can affect a yacht behind and to leeward, two yachts may be able to sail close and parallel windward courses where their wind shadows do not affect one another. This is referred to as the safe leeward position, as shown in Figure 45. Each yacht is vulnerable to the other moving slightly ahead, as this will result in the slower yacht encountering disturbed air. However, if the two yachts are of similar speed, they may be able to maintain such a close parallel course for some time without disadvantage to either.

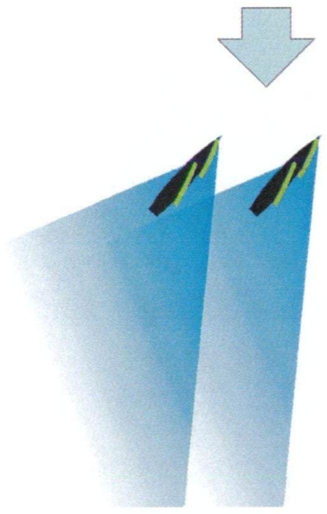


Figure 45. Safe leeward position

Covering

The leading boat may use its wind shadow to adversely affect the performance of a trailing boat. In Figure 46, yacht A crosses in front of yacht B and tacks so that her wind shadow falls directly onto B's sails. In practice, this is such a damaging position for B that she will invariably tack immediately to obtain clear air.

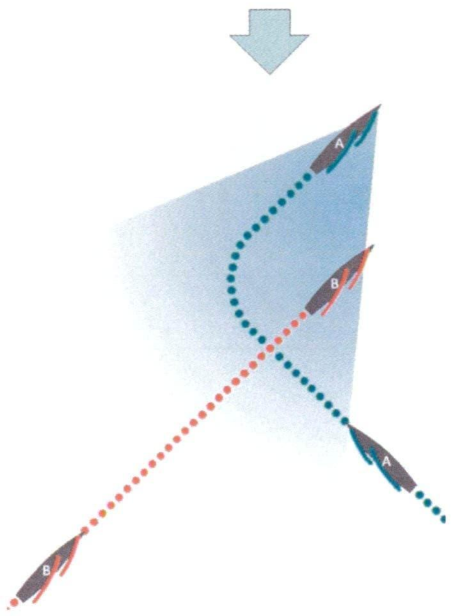


Figure 46. A tacking into a tight cover on B

THE LEE-BOW

When two boats sailing upwind on opposite tacks converge, the boat on starboard tack has right of way. Despite this, the port tack boat may have tactical control of the situation if it is level with or slightly ahead of the starboard tack yacht. Figure 47 illustrates this situation. At position 1, yacht A is on starboard tack and has right of way. Yacht B must choose to tack or cross behind A. However, if yacht B tacks into a lee-bow position, as shown in position 2, yacht A will be backwinded and will lose distance. In this situation the best option for yacht A is to tack immediately to gain clear air, as shown in position 3.

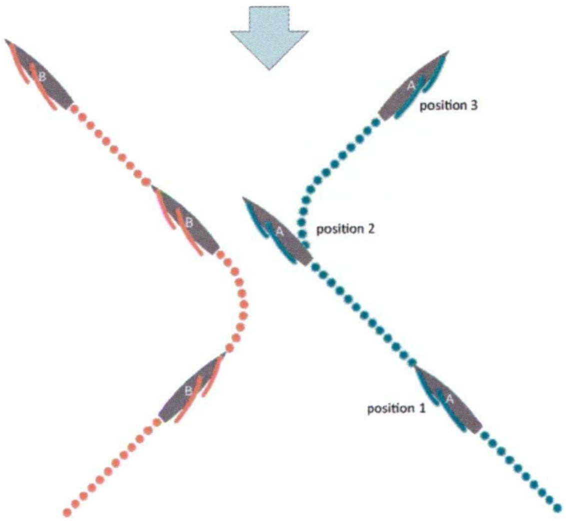


Figure 47. Tactical control of A by port tack yacht, B

Laylines

Once a trailing yacht reaches the layline, a choice must be made as to whether to tack on the layline behind the leading boat, or to continue one to two boat-lengths before tacking, in order to ensure clear air. These options are illustrated in Figure 48.

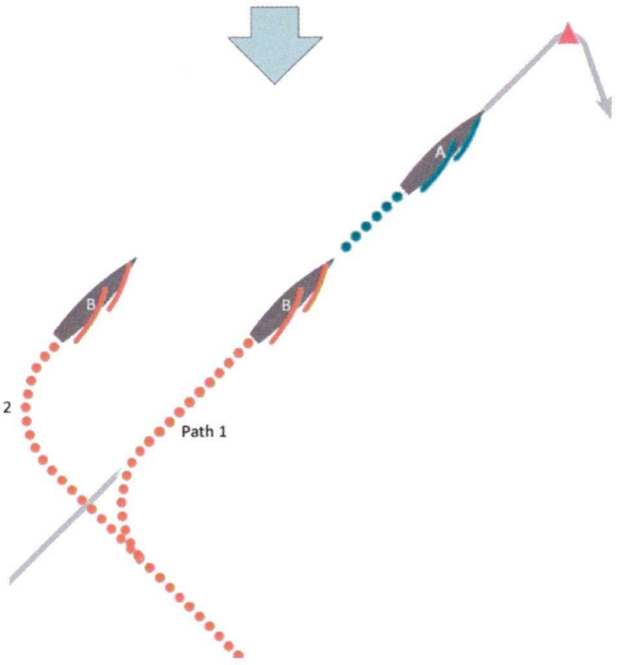


Figure 48. Trailing boat options at the layline

If the boats are close and yacht B tacks line astern of yacht A, as illustrated by path 1 in Figure 48, the effect of the backwind from yacht A will be significant and yacht B will rapidly lose one to two boat-lengths. If yacht B continues on starboard tack slightly further prior to tacking, path 2, clear air will be guaranteed, however the extra distance sailed is distance lost to the leader.

Which of these two options results in the smallest loss is determined by the proximity of the yachts to the mark. When the leading yacht is close to the mark, the period of time during which the trailing yacht is backwinded is brief, limiting the loss. If the yachts have a significant distance to sail before the mark is reached, sacrificing some distance initially to gain clear air will result in a smaller overall loss.

This option is important in the consideration of the disadvantage suffered by trailing boats within a race simulation. Although the trailing boat is disadvantaged once the layline is reached, the magnitude of that disadvantage can always be limited to the extra distance that must be sailed beyond the layline in order to obtain clear air.

MARK ROUNDING

The effect of the trailing boat disadvantage at each windward layline is that it is extremely difficult for a trailing boat to be overlapped with the leading boat as the windward mark is rounded.

This is not the case at leeward marks, where yachts have the option of choosing to round either end of a gate, allowing them round level with another boat without loss. The gate also allows a trailing boat to gain clear air and separation from the leading boat, as shown in Figure 49.

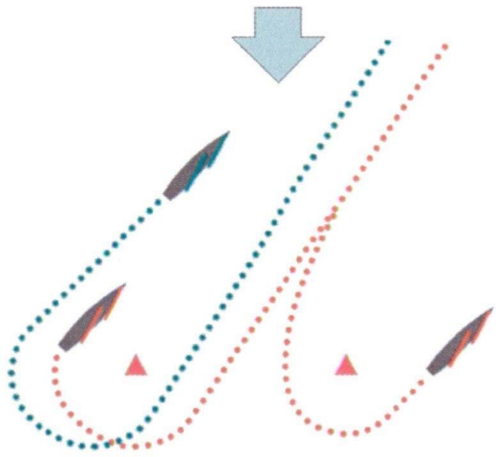


Figure 49. Leeward gate options for trailing boat

START LINE ADVANTAGE

A yacht that is able to obtain the starboard end of the start line at the commencement of a race has an inherent advantage due to the right-of-way status this confers. Any tournament simulation needs to ensure that competitors are allocated the favourable starting position in an equal number of races.

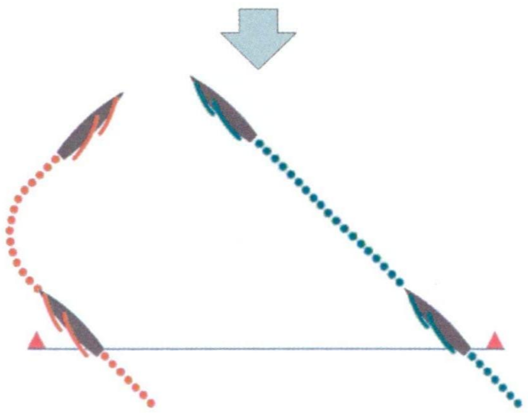


Figure 50. Right-of-way advantage at the start

5.2.2 RMP BACKGROUND

Race Modelling Programs came to prominence with their use by the Sail America Team for the design of *Stars & Stripes 87*, the winning yacht in the 1987 America's Cup (Oliver et al. 1987). Other RMPs have been developed since that time, most notably the ACROBAT RMP, described by Philpott et al. (2003; 2004). These RMPs have ranged from simple probabilistic models through to full time-domain simulations, with execution times varying widely according to the complexity of the simulation performed.

SAIL AMERICA

Sail America used two different RMP, a simple probabilistic model, RMP1, and a time-domain race simulation, RMP2.

RMP1

RMP1 focussed on evaluating the probability of a particular yacht winning a race against another yacht. It was based on work described by Letcher (1974) and assumed a wind velocity that would vary throughout the race according to a probability density function $p(V_{TW})$.

Where the relative performance of two yachts, A and B, can be displayed as a comparison plot in which the time difference curves cross only once, the probability that yacht B beats yacht A can be expressed as $P(V_{TWc})$, where V_{TWc} is the crossover velocity.

$P(V_{TW})$ is the cumulative probability density distribution:

$$P(V_{TW}) = \int_0^{V_{TW}} p(V_{TW}) d\, V_{TW} \tag{5.1}$$

Letcher recognised the influence of random variations in conditions on the outcome of races and attempted to model this using an "uncertainty function" $f(\Delta T; V_{TW})$ which gives the probability $P(\text{win})$ that B wins over A when her computed time difference is greater than ΔT:

$$P(\text{win}) = f(\Delta T; V_{TW}) = \begin{cases} 0 & if\ \ \Delta T < -\tau \\ 0.5 + 0.5(\Delta T/\tau) & if\ -\tau \leq \Delta T \leq \ \tau \\ 1 & if\ \ \Delta T > \tau \end{cases} \tag{5.2}$$

where $\tau = 800/_{V_{TW}}$, a subjective measure of the margin required to ensure a win. This value was derived from the opinions of sailors as to what constituted an unbeatable lead in a race.

The probability that B wins a single race is:

$$P(\text{win}) = \int_0^\infty p(V_{TW}) f[\Delta T; V_{TW}] dV_{TW} \tag{5.3}$$

RMP1 integrated (5.3) using a $p(V_{TW})$ derived from historical wind velocity distributions from the regatta site of interest. While RMP1 had the advantage of extremely short run times due to its simple structure, the use of the subjective parameter τ to quantify uncertainty in the race results was a questionable approach.

RMP2

The second RMP developed for Sail America, RMP2, was a time domain simulation that raced two boats around a full America's Cup course, using wind distributions derived from wind data for the previous 12 years for Fremantle, the site of the 1987 America's Cup. RMP2 implemented interactions between boats, as well as penalties for various disadvantageous tactical situations, including mark roundings. An uncertainty function was applied to finishing times to determine a win/loss probability for each race. Race simulations were repeated for each day of the months of October through January for the previous 12 years, totalling almost 1500 runs.

RMP2 was used for the majority of the race modelling performed by Sail America. The number of runs required was relatively small, as independent variation of individual design parameters was not attempted. Rather, the comparison was limited to allometric variation of length, constrained by the 12-Metre rule and scantling requirements. The output from this set of tests, expressed as winning probabilities, showed surprising complexity, as illustrated in Figure 51.

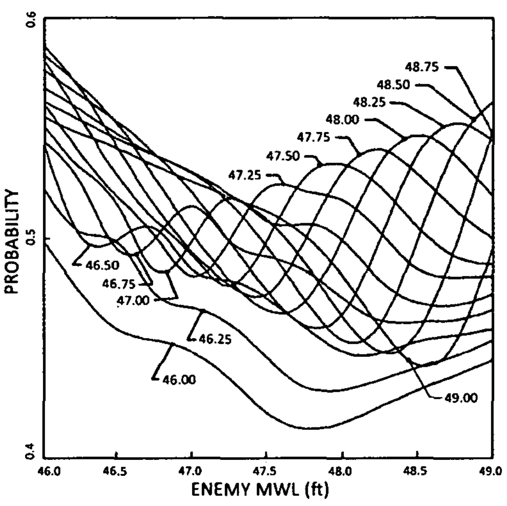


Figure 51. Win probabilities for various *Stars* & *Stripes* Measurement Waterline Lengths (MWL) against opponents of various lengths (January Perth conditions), from Oliver et al. (1987)

ACROBAT

The detail and complexity of Sail America's RMP2 was extended by the ACROBAT program, developed in conjunction with Team New Zealand for the defence of the 2000 America's Cup.

ACROBAT attempts to create an accurate simulation of a match race by modelling the performance of the two yachts, the interactions between them, the course on which they sail, and the wind conditions they encounter.

ACROBAT incorporated the following features:

- fixed-time-increment simulation, with integration of accelerations between time steps;
- wind velocity and direction modelled as Markov chains, utilising the Taylor hypothesis for movement of the wind field. This treats wind turbulence, although random, as fixed, with eddies travelling downwind at a mean wind speed V, so an anemometer a distance d upwind of a mark, will give exactly the same reading as an anemometer at the mark (d/V) seconds later.
- variation of wind fields between boats, correlated to their separation;
- a highly detailed tactical decision model for determining the course taken by the yachts, including:
  - simulation of course and velocity changes during tacking;
  - mark rounding simulation;
  - wind shadow and backwind effects between yachts.

ACROBAT TACTICAL DECISION MODEL

ACROBAT included a complex set of penalties in order to encourage each yacht to sail an optimal route. Penalties were applied for:

- sailing in the disturbed air from a leading boat;
- sailing on the headed tack;
- lateral separation from a trailing boat;
- straying too far from the centre of the course;
- sailing past a layline on the wrong tack;
- tacking when below equilibrium speed;
- approaching the mark from the incorrect side;
- collision courses with right-of-way yachts;
- passing situations.

Penalties within ACROBAT are weighted and summed to give a total penalty P^c for the current tack:

$$P^c = \sum_i P_i^c w_i \tag{5.4}$$

where w_i are weights for each penalty.

The total penalty P^o for the opposite tack is given by:

$$P^o = \sum_i P_i^o w_i \tag{5.5}$$

The tack with the smallest total penalty is chosen to be the best tack at that point in the simulation. However, the tactical model as described by Philpott does not appear to look ahead and, as a result, may be prone to sailing into tactical situations less favourable than currently encountered.

Although the use of different tactical models for each boat is described in some detail by Tierney (1998), later descriptions of ACROBAT downplay this. Philpott (2004) states:

> *"In order to compare yacht designs with different design tradeoffs there must be no bias introduced with regard to different tactical abilities of the two helmsmen. Consequently each yacht uses an identical tactical decision model" (Philpott et al. 2004, p. 11)*

5.2.3 PROPOSED VESPA RMP METHODOLOGY

The design of the RMP for use within VESPA involves several key decisions that determine the architecture and overall performance of the complete system. An RMP may be a simple probabilistic model, such as that used for the Sail America RMP1, taking a yacht performance profile derived from a VPP and combining it with an expected wind distribution for the race, to produce a combined probability density function (PDF). This can be integrated and compared to that of a different yacht in order to determine the probability of winning a particular race.

The computational performance of this approach is expected to be good. However, care must be taken to ensure that this method encapsulates sufficient detail to accurately rank two yachts in a simulated race.

At the other end of the scale of complexity, it is possible to perform a full simulation of one or more yachts racing around a course, involving tacking, gybing and mark rounding manoeuvres in varying wind and sea conditions. This option involves a great deal of computation, particularly if accelerations between time steps are taken into account, and may be prohibitively slow when incorporated into an optimisation procedure. This is the approach taken by the ACROBAT RMP.

The primary motivation for the level of detail incorporated in ACROBAT is the belief that it enables the program to better estimate the probability of one yacht winning against another:

> *"Previous RMPs do not consider the tactical advantages that a faster yacht has over a slower yacht... and therefore may underestimate the probability that a faster yacht wins... In a simulation of a match-race, the tactical advantages that a leader has can be modelled, thereby improving the estimate of the win/loss probability when comparing yacht designs." (Philpott et al. 2004, p.2)*

However, the benefit claimed by Philpott is unlikely to have been as great as was expected, due to an over-emphasis on factors involving interaction between boats, such as wind shadow and backwind. While the leading boat in a match race has a wind shadow that may affect the trailing boat and result in an increased lead, such interactions are not as common as might be expected.

Rather than a trailing boat experiencing significant disturbed air on a windward leg, it is the threat of disturbed air that is used by the leading boat to shepherd the trailing boat towards a layline. Once pushed to the layline, the trailing boat has limited tactical and strategic options and will be forced to sail in disturbed air, or to sail extra distance to avoid it.

The losses due to interactions between yachts once both are at the layline are more easily quantified, as the relative positions of the yachts are restricted to a small number of alternatives:

- line astern;
- opposite tacks;
- trailing boat forced above layline to obtain clear air.

Although a detailed simulation such as that used for ACROBAT may be useful for evaluating the effects of design variations that influence manoeuvring, or for real-time modelling of tactical situations for crew training purposes, the additional detail included adds little to the evaluation of yacht performance. It may result in a less accurate model that is more difficult to verify and validate. In this regard Sánchez (2006) gives the following advice:

> *"Many modelers make the mistake of equating detail with accuracy. They start with a grand vision of a highly detailed model which mirrors every aspect of the real world system... The sheer magnitude of such programs makes verification and validation nearly impossible. The behavior of the program is determined by dozens to hundreds of...inputs whose correspondence to reality is tenuous at best." (Sánchez 2006, p.4)*

For a race model intended to rank the performance of two boats, the features of ACROBAT described in Section 5.2.2 were considered unnecessary for the following reasons:

Integration of accelerations between time steps. Prior to the Version 5 ACC rule, a significant range of waterline lengths, displacements and sail areas could be adopted for an ACC design. As a result, the differences in rates of acceleration between two boats when manoeuvring or sailing in varying wind conditions may have been significant. Under Version 5 of the ACC rule, all boats can be considered to have the same length, displacement and sail area, and differences in their rates of acceleration are insignificant.

This is particularly true on upwind legs due to the relatively narrow upwind speed range speed of ACC yachts. The V_{MG} for a typical ACC yacht varies by little more than one knot for wind velocities from 9 to 20 knots. Changes in hull velocity as wind velocity changes are small and the benefits of integrating accelerations between time steps upwind are negligible. Downwind, boat speed variation across the wind range is greater, with a 5 knot difference in downwind V_{MG} between 9 and 20 knots of wind. However, the rate of acceleration of the boat is also greater due to the large amount of additional sail area carried.

Rather than using a fixed-time-increment simulation, the VESPA RMP divides the distance between each pair of marks into 100 bands (approximately 50 metres per band), stochastically sampling wind velocity and direction from the specified distributions for each band. Each band is regarded as a steady-state simulation, with yacht performance derived from the VPP. The time taken for each yacht to sail through each fixed band of wind is calculated and added to the elapsed time for the yacht for the leg.

Wind velocity and direction modelled as Markov chains. The VESPA RMP randomly samples the specified wind distribution. However, as the simulation uses discrete steps without integration of accelerations, the order of the samples is not relevant and Markov chains for wind velocity and direction are not required.

Variation of wind fields between boats correlated to the separation between them. ACROBAT incorporates a complex wind model that varies both spatially and temporally. The effect is that two boats sailing close together will experience similar wind conditions, but as the boats separate, the wind fields they encounter will differ. This introduces unnecessary variance into the race model, blurring the outcome. Philpott confirms this interpretation:

> *"In the case where the yachts see no correlation in the weather conditions it appears that the advantage of the faster yacht has been reduced due to the random nature of the weather observed on each yacht, which swamps the speed difference between the yachts" (Philpott et al. 2004, p.15)*

As a consequence, the VESPA RMP uses identical wind fields for each boat in order to discriminate performance differences between them without the introduction of unquantified variance.

Highly detailed tactical decision model for determining the course taken by the yachts. Although this is a centrepiece of ACROBAT's design, the complexity and uncertainty involved in a detailed tactical decision model was considered counter-productive for the VESPA RMP. Consequently, the VESPA RMP ignores alternative courses on each leg and the tactical considerations that may determine them. Each leg is treated as a one-tack "drag-race", with no need for tactical route planning and no interactions between boats until the mark is reached.

Tacking simulation. Subsequent to the introduction of the version 5.0 ACC rule, no appreciable difference exists between the length, displacement and sail area of different ACC yachts. While this may not have been the case in previous years, the difference in tacking dynamics between boats is now small. Differences that do exist are primarily due to factors that may be difficult to quantify in a simulation, such as the effects of keel and rudder area and section, or overall balance of the yacht.

In general, boats engaged in a match race will minimise the tacks performed and, on average, will tend to execute a similar number of tacks during a race. As an example, the two competitors in the seven races of the 2007 America's Cup completed the tacks listed in Table 5:

Table 5. Tacks Performed During Races 1-7, America's Cup 2007

	Leg 1		Leg 3	
	Leading boat	**Trailing boat**	**Leading boat**	**Trailing boat**
Race 1	15	15	17	16
Race 2	7	9	4	5
Race 3	6	5	6	8
Race 4	1	1	12	13
Race 5	2	3	8	9
Race 6	3	5	6	7
Race 7	9	9	10	9
Total	**44**	**46**	**64**	**66**

For the fourteen legs listed, three had the same number of tacks for each boat, eight legs were won by the boat having the fewest tacks and three legs were won by the boat executing the greater number of tacks. ETNZ, winner of two of the seven races, performed 112 tacks, while Alinghi performed 108 tacks.

The VESPA RMP disregards both the dynamics of tacking and the number of tacks required on the basis that they have little effect on the ranking of version 5.0 ACC yachts. Although there is a small bias in favour of the boat that executes fewer tacks, this may be handled statistically and applied as a penalty function for the trailing boat at the completion of each leg.

Mark rounding simulation. Each boat in a match race is required to round the same number of windward marks in the same direction. As all ACC yachts designed to version 5.0 of the ACC rule have effectively identical length, displacement and sail area, these mark roundings can be considered identical, with no benefit accruing to either boat as long as the boats are not overlapped. An example can be seen in Figure 52, which shows almost identical paths followed by both competitors during an actual mark rounding, involving a change of course of greater than 210 degrees.

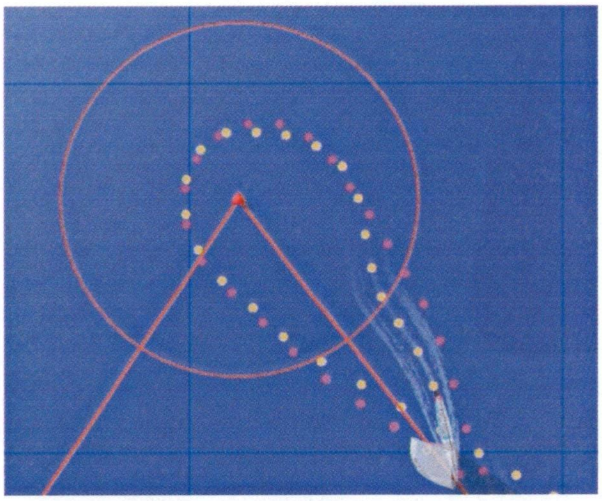


Figure 52. Windward mark rounding, America's Cup 2007, race 5, leg 1

Examination of actual America's Cup match races shows that overlaps at windward marks are uncommon, due to the disadvantage suffered by the trailing boat once the layline is reached. For example, analysis performed by the author of windward mark deltas during the 2007 America's Cup, showed deltas ranging between 7 and 25 seconds. At no time were the yachts overlapped at any mark roundings other than start and finish lines during the seven races sailed.

Consequently, VESPA does not model the dynamics of either windward or leeward mark roundings, assuming instead that marks are rounded with no advantage accruing to either boat.

Wind shadow and backwind effects between yachts. ACROBAT explicitly models wind shadow and backwind interactions between boats, particularly on the windward legs. This appears to be based on the assumption that, as the leading boat casts a wind shadow that is capable of significantly disturbing the wind encountered by the trailing boat, it is therefore essential that this effect be modelled.

This is a naive assumption based on a misinterpretation of yacht racing tactics and strategy. In match racing, the effect on the trailing boat of sailing in disturbed air is so great that this situation will be avoided wherever possible. Racing sailors learn very early in their careers that clear air is paramount and will take whatever steps are necessary to obtain it.

This can be seen clearly in traces taken from actual America's Cup races, as shown in Figure 53. During these windward legs, the trailing boat generally avoids disturbed air, either by splitting tacks with the leader, as shown at position A on leg 1 of race 1, or by establishing sufficient lateral separation on the course to be in undisturbed airflow. This situation usually continues until both boats are at the layline, typically 100-200 metres from the windward mark.

While a trailing yacht is not always able to establish clear air, it is the exception rather than the rule for a trailing yacht to be forced into this position for extended periods. Situations where a leading boat may have been in a position to affect the wind encountered by the trailing boat are those portions of the legs shown where the tracks are close and parallel. Other than when both boats are on a layline for the windward mark, where a trailing boat has no alternative but to accept any bad air it receives, this situation only occurs for the races illustrated during the latter portion of leg 2, race 2 and for a brief period in the middle of leg 3, race 3.

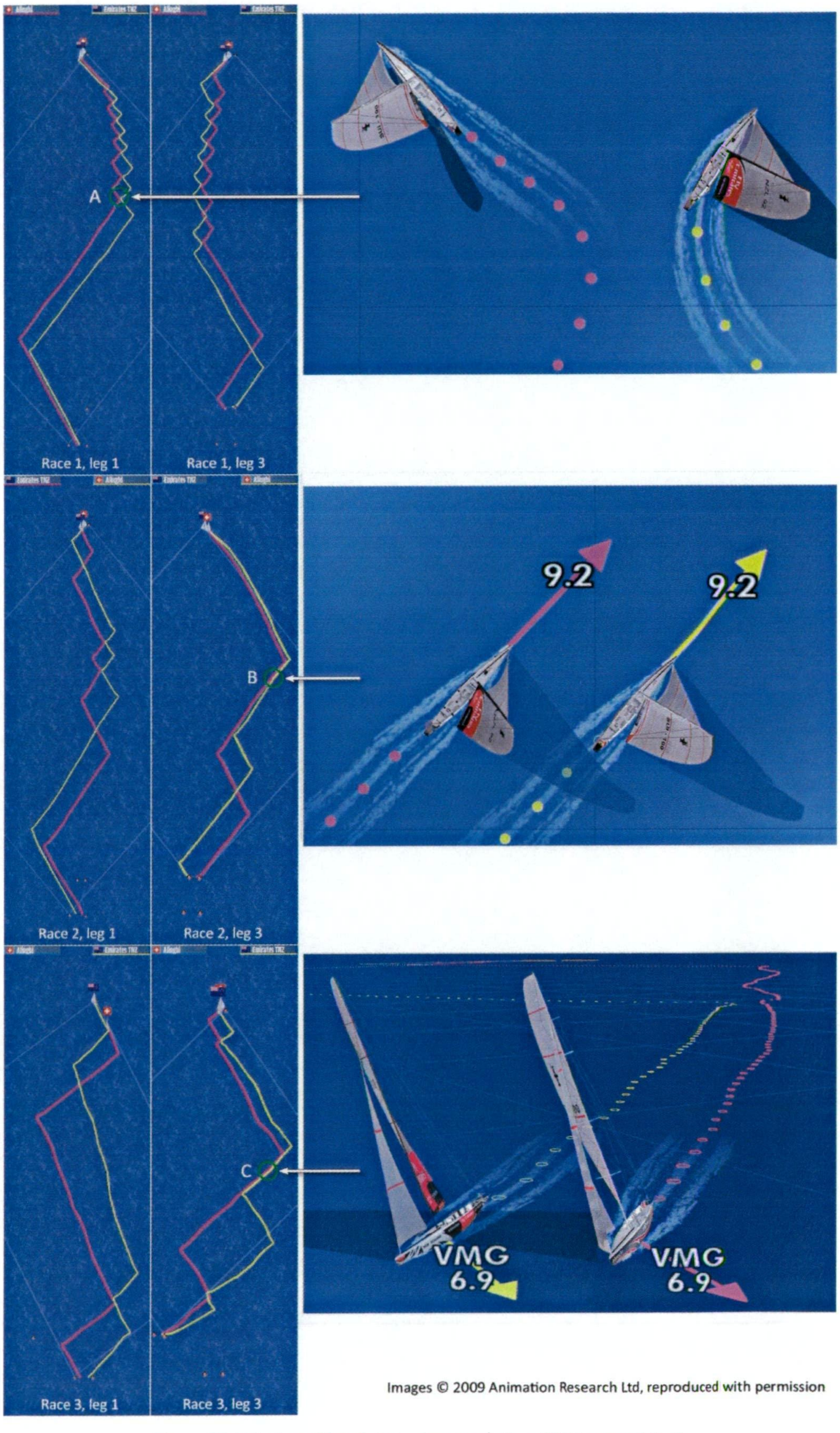
Remarkably, rather than the trailing boat being adversely affected by bad air from the leading boat on these two legs, the trailing boat actually made significant gains and was able to pass the leading boat. This can be seen in the race time data shown in Table 6, which shows that the trailing boat passed the leading boat on both legs.

Table 6. Times for Races 2 and 3, America's Cup 2007

	ALINGHI			ETNZ	
Race 2	Gain	Lead	Delta	Lead	Gain
Leg 1	0:22	•	0:19		
Leg 2		•	0:13		0:06
Leg 3			0:15	•	0:28
Leg 4			0:28	•	0:13

	ALINGHI			ETNZ	
Race 3	Gain	Lead	Delta	Lead	Gain
Leg 1			1:23	•	1:31
Leg 2	0:21		1:02	•	
Leg 3	1:17	•	0:15		
Leg 4			0:25	•	0:40

Despite the two boats sailing close parallel courses for portions of these legs, there is no evidence of the trailing boat suffering from disturbed air. In fact, during the periods of close, parallel courses, both boats established positions giving them mutual clear air, which they were able to maintain until the last few hundred metres of the leg. This is shown in the images on the right of Figure 53, which illustrate the yachts at points B and C. In both positions, each yacht has clear air despite their close proximity.



Images © 2009 Animation Research Ltd, reproduced with permission

Figure 53. Windward leg details, America's Cup 2007, races 1 to 3

These examples suggest that when simulating match races for the purposes of ranking yachts, the interactions between the boats are generally not an issue until the boats are on the same layline for the windward mark. Once in the vicinity of the mark, the penalty for the trailing boat resulting from the wind shadow and backwind of the leading boat can be handled statistically rather than requiring a physical simulation. Importantly, the loss suffered by the trailing boat in this situation is always limited to the extra distance that it must sail beyond the layline in order to obtain clear air prior to tacking.

5.2.4 ACCOUNTING FOR UNCERTAINTY

The outcome of a single match-race may be influenced by a considerable degree of random variation in the environment encountered by the yachts. Wind velocity and direction vary both temporally as well as spatially, waves of different size and wavelength impede the motion of the yacht, and sailors make errors in boat and sail handling.

In order to estimate the relative performance of two yachts it is necessary to incorporate these variations within the RMP.

WIND DISTRIBUTIONS

A wind velocity distribution for the event is defined. In its simplest form, this consists of a mean wind velocity and standard deviation for the duration of the event.

A series of races are defined, each having its own set of wind conditions. The wind velocity distribution for the entire event is sampled to derive a mean wind velocity for the race. This value is used to create a wind velocity distribution using an individual standard deviation (SD) for that race. This SD covers only the period of the race (2-3 hours) so is significantly smaller than the SD for the entire event. Importantly, this SD may vary based on other factors. For example, the variance in the wind velocity may be correlated with both the wind velocity and direction. Conversely, two races having the same mean wind velocity may encounter different variance in the wind conditions.

This wind velocity distribution used by VESPA for each race is clipped to upper and lower limits. These are required due to race regulations for the Americas Cup event, which stipulate a minimum and maximum wind velocity in which racing may take place.

RANDOM VARIATION IN PERFORMANCE

VESPA uses a stochastic wind model, incorporating Monte Carlo sampling of wind distributions for wind velocity and direction. However, once a wind strength and direction is chosen for each step of the simulation, each race is modelled in a deterministic manner.

When the win/loss ratio for a number of such races is used as the measure of merit for an optimisation process, the amount by which a boat wins a race does not contribute to its overall fitness. A race that is won by one second ranks equally as a race that is won by ten minutes. This does not produce a robust solution, as only a small error in the estimation of a yacht's performance may result in a significant change in the overall win/loss ratio.

In real-world racing conditions, the greater the boat speed advantage a yacht has against its competitors, the better. In a tournament where random variation occurs in the racing conditions, a large boat speed advantage will translate into a higher win/loss ratio.

As an example of why this is the case, consider two boats, one of which is only one second faster around the course than the other. In a strictly deterministic race model simulation, the faster boat will win 100% of its races. Yet in the real world, the faster boat will not win 100% of its races, nor will it win a large majority of its races. Rather, it will win only slightly more than 50% of its races. The randomness inherent in real-world sailing conditions will outweigh the small boat-speed advantage in almost every race.

Conversely, if one of the yachts has a 30-second boat-speed advantage around the course for the specific set of conditions, when the two boats race one another in real-world weather conditions multiple times, the slower boat may still win a small percentage of the races.

The slower boat's win/loss ratio will not be zero because a random component exists due to variations in wind velocity and direction experienced by each boat, variations in the waves they encounter, as well as the inevitable errors in boat handling and crew work. This random variation will occasionally favour the slower boat sufficiently for it to overcome the boat speed advantage of the faster boat. However, the faster yacht will win a large majority of the races, as its boat speed advantage will outweigh the random variations in sailing conditions in the majority of races.

In order to provide a more robust measure of the probability that one boat will beat another, this variability may be modelled using Monte Carlo methods. Races simulations are repeated multiple times with identical environments, but with a small, random time variation added to the elapsed time for each yacht for each leg of the course.

Note that if a race consisted of only one leg, it would be sufficient to treat the race times for each yacht as distributions. In this case, the probability that one yacht defeats another may be determined by comparing the mean and standard deviation of their race time distributions. However, the multiple-leg format of America's Cup races prevents this approach being taken as each turning mark effectively applies a small penalty to the trailing boat. The total elapsed time for each yacht is therefore dependent on interactions between the yachts at these marks and cannot be determined in a statistical fashion.

As the random time variation for a particular leg of a racecourse is a summation of a large number of randomly occurring events, it is expected to have a normal distribution. The variance of this distribution is difficult to determine through simulation, but may be estimated from actual performance data of America's Cup Class yachts, such as that logged during two-boat testing.

In order to estimate the magnitude of the random noise inherent in real world sailing conditions, data was obtained from the Alinghi design team for a series of two boat tests, taken over a period of 15 days during May and June 2006. The purpose of these tests is to examine the effects of small adjustments to sail trim, rig setup and appendage configuration. Consequently, the yachts used for testing are set up to be as similar in performance as possible.

Twenty-five test runs, which encountered wind velocities in the range of 7–15 knots, were selected for analysis. This selection ensured that the wind velocity distribution was similar to that expected for the America's Cup match in June 2007 and that the mean and standard deviation of the wind velocity were close to that used by VESPA's race modelling program.

The results of this testing are shown in Table 7, where the test-run date and time, standard deviation of the true wind direction (β_{TW} SD), V_{TW}, V_{MG} and boat lead at the end of each 10 minute test run are tabulated. Results of this investigation found that the SD of the time difference between the boats in seconds for a 10-minute test was 4.47 seconds.

Table 7. Results of Two-Boat Testing, May-June 2006

Date (yyyymmdd)	Start time (hhmmss)	End time (hhmmss)	β_{TW} SD (degrees)	V_{TW} (knots)	V_{MG} (knots)	Lead (seconds)
20060525	155856	160858	1.56	7.52	7.10	3.39
20060525	161234	162239	1.61	7.54	7.26	8.47
20060529	140702	141707	1.44	12.97	8.59	1.33
20060529	142654	143658	2.40	11.75	8.36	1.62
20060529	144024	145030	1.68	12.02	8.32	9.14
20060529	160658	161710	3.15	11.52	8.28	9.41
20060529	162609	163624	1.67	12.70	8.44	4.24
20060529	164010	165015	1.33	12.88	8.55	3.36
20060530	120025	121033	2.15	11.95	8.24	13.43
20060530	124147	125152	2.43	12.66	8.35	-2.41
20060530	125810	130822	2.50	13.42	8.38	-3.99
20060530	140524	141549	2.14	14.77	8.64	7.73
20060530	142410	143423	2.07	14.35	8.53	8.47
20060602	160230	161232	1.80	13.79	8.69	3.70
20060607	145500	150505	3.62	10.07	8.02	10.82
20060607	154440	155445	1.94	12.64	8.60	5.10
20060607	160657	161704	3.30	13.90	8.59	-0.82
20060607	162027	163030	0.98	14.55	8.60	2.49
20060608	113059	114447	2.59	11.30	8.31	0.28
20060608	114816	115822	1.93	11.57	8.58	8.59
20060608	142235	143242	2.10	12.73	8.42	7.43
20060608	144426	145430	1.51	13.09	8.53	-0.32
20060609	135843	140846	3.87	11.16	8.39	4.94
20060609	141210	142713	4.75	11.93	8.53	6.17
20060609	144145	145147	2.41	12.69	8.68	10.30
			mean	**12.22**	**8.36**	**4.92**
			std. deviation	**1.80**	**0.39**	**4.47**

The windward leg of an America's Cup race for the 2007 event is specified as 3.0 nautical miles. Using the average V_{MG} taken from the two boat testing results in a windward leg that takes approximately 20 minutes to complete when sailing in average wind conditions.

As the test period used was typically 10 minutes, this SD value needs to be scaled to suit the longer length of legs used for actual racing. To do this, the SD can be scaled using the equation for generalised volatility σ_T for time horizon T, which is expressed as:

$$\sigma_T = \sigma\sqrt{T} \qquad (5.6)$$

If the SD of the time difference between two boats for a 10-minute test period is 4.47 seconds and the SD for a 20-minute period is required, this is calculated as:

$$\sigma_T = 4.47\sqrt{\frac{20}{10}} = 6.32 \text{ seconds} \tag{5.7}$$

Subsequent to this analysis, VESPA's RMP incorporated multiple repetitions of each race with noise in the form of a random time penalty added to one boat for each leg of the race, in order to produce a more robust estimate of a yacht's probability of winning races. The magnitude of this penalty is derived from a normal distribution with a standard deviation of 6 seconds. This standard deviation, derived from the two boat testing data, was considered conservative, yet sufficiently large to have an effect on the outcome of the optimisation process.

RANDOM VARIATION IN WIND DIRECTION

Wind varies not only in its strength but also in its direction. Large changes in direction occur from day to day, but smaller changes also occur on shorter timescales, even in apparently steady wind conditions. Some wind conditions, such as thermal sea-breezes, have a stable mean direction, but oscillate regularly by 5-10 degrees either side of the mean.

It is possible that in these conditions the strategies employed to sail the fastest course may result in a change to the optimal design for an ACC yacht. To understand why this may be the case it is necessary to consider a strategy that, although widely used by sailors for many years, who referred to it as "footing to the header", was first formalised in 1987 by Ockam Instruments (2006) and given the name "wallying".

Wallying was first used by the *Stars & Stripes* crew competing in the 1987 America's Cup. Course recommendations were calculated by the yacht's instrumentation and navigation system, based on the polar performance curves for the yacht.

To avoid the onboard TV cameras and microphones revealing a new technique that the *Stars & Stripes* crew considered a key competitive advantage, the crew referred to a fictitious crewmember, Wally, when course adjustments were relayed to the helmsman, Dennis Conner. Messages from the Navigator such as "Wally suggests two tenths faster than target, Dennis" were clear to those onboard, but baffling to those unaware of the technique being used (Teeters 2004).

If a yacht is sailing towards a mark that is directly to windward, the best performance that can be obtained is referred to as Velocity Made Good or V_{MG}. This can be illustrated using a polar performance curve, as shown in Figure 54, where best V_{MG} is a found by a line perpendicular to the wind direction and tangent to the polar curve.

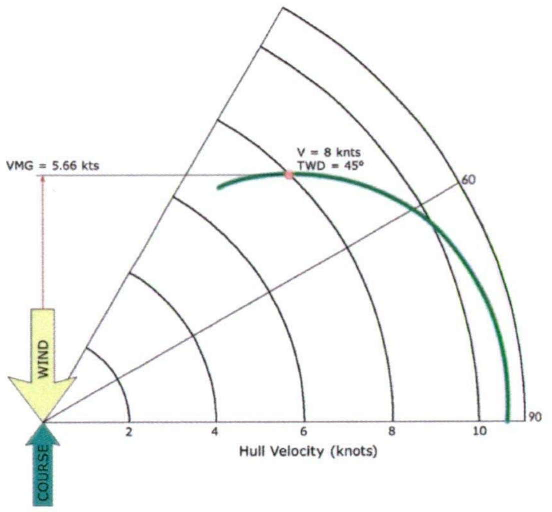


Figure 54. Polar performance curve with V_{MG}

If a permanent wind shift occurs, and the magnitude of the shift is not sufficient to allow the yacht to lay the mark on a single tack, the optimal course remains the angle of best V_{MG}.

However, when the wind direction shifts, but is expected to shift back prior to the yacht reaching the lay line, it is possible to improve on the speed made good towards the next mark by using a technique that involves sailing at a angle lower than V_{MG}.

Wallying is the practice of deviating from the optimal velocity made good to windward (V_{MG}) when sailing in a wind direction that is shifted from the mean wind direction, such that the velocity made good in the direction of the next mark (V_{MC}) is maximized (Figure 55).

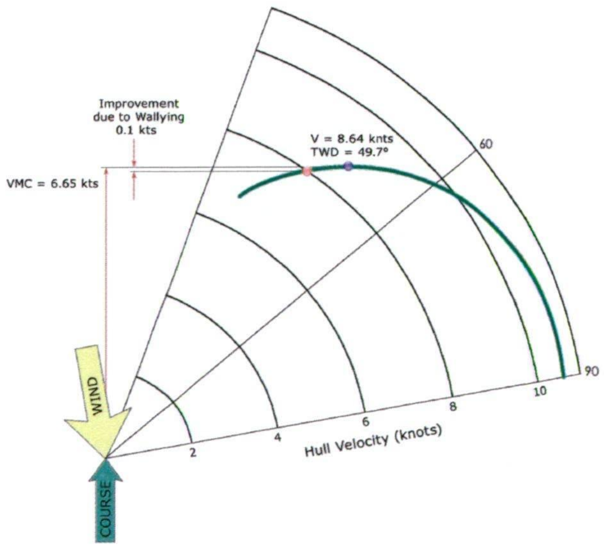


Figure 55. Polar performance curve for a shifted wind direction, showing advantage of wallying

Benefits from wallying only eventuate when the wind shifts back towards its original direction or beyond. If the shift in wind direction is permanent, i.e. a persistent shift, wallying will result in a longer elapsed time to reach the windward mark. Consequently, the successful use of wallying is dependent on the probability that the wind, once shifted in direction, will shift back prior to the yacht reaching the layline. For this to occur the wind should be oscillating in its direction, and there should be sufficient time for one full oscillation before the layline is reached.

Wallying is best used when the expectation of a mean reverting oscillation is high, for example, near the start of a windward leg, when the wind is oscillating predictably, with a period shorter than the length of the leg. Conversely, wallying is unlikely to be used when the time to the layline is short, the wind is shifting persistently in one direction, or when the oscillation period is long.

Wallying was included in the VESPA race model in order to investigate whether wallying in winds of variable direction could have a measurable effect on the design of the yacht. Because wallying requires the helmsman to sail at a slightly lower heading and at a higher boat speed upwind, it is conceivable that a boat designed for a regularly oscillating wind might have slightly different optimal design parameters compared to a boat designed for the same wind velocity but without directional variation. For example, a boat designed for winds that oscillated significantly in direction would on average sail lower and faster, so righting moment may need to be greater, meaning wider B_{WL} and/or higher C_M, as well as higher C_P and a L_{CB} that is further aft.

TRAILING BOAT PENALTIES

The purpose of trailing boat penalties is to quantify the benefit of being the first boat to reach a windward mark in a match race. This benefit is important as it determines to what extent a boat should have its performance biased to upwind work relative to downwind work.

These considerations resulted in the inclusion of two different forms of trailing boat penalties within the VESPA race model. The first is a simple port tack penalty – if overlapped, the port tack boat is forced to cross behind the starboard tack boat.

The second component is a penalty for disturbed air. If a boat is behind at the first mark, it is penalized by the addition of a few seconds to its elapsed time for the leg. This penalty is at a maximum of 6 seconds when the boats are almost overlapped, tapering linearly to zero for a lead of 20 seconds.

Initially it was considered that a separate trailing boat penalty for leeward mark roundings would be necessary in addition to the windward mark trailing boat penalty. However, the course for the 2007 America's Cup was modified from that used for the 2003 America's Cup by the provision of a leeward mark gate, made up of two marks approximately 150 metres apart.

The effect of this was to remove any penalty for the boat trailing at the end of the downwind leg. Instead of being forced to follow the leading boat around the mark, making it easy for the leader to gain tactical control, the trailing boat could opt for a different mark than that rounded by the leading boat. As a result of this change to the design of the course, it was concluded that leeward mark roundings did not require a trailing boat penalty to be applied by the VESPA RMP.

5.2.5 SUMMARY

Rather than attempting to create a perfect simulation of the physics of a match race, the VESPA RMP focuses on modelling the simplest possible race that will correctly rank two boats. Factors that help to clarify the primary performance differences between yachts have been given priority, while second order effects, such as the impact of hull shape changes on manoeuvring, are ignored.

Some of this simplification is possible because of recent changes to the ACC rule. In previous versions of the ACC rule a range of waterline lengths, displacements and sail areas were legal. The version 5.0 amendments to the ACC rule reduced these ranges to the point that all boats can be considered to have the same length, displacement and sail area. Consequently, there now are negligible differences between yachts accelerating and decelerating in gusts and lulls, or during tacking or mark rounding manoeuvres.

As short execution times are considered essential, the VESPA RMP dispenses with direct physical modelling of many of the complexities of yacht racing where it is considered that no net benefit for either boat occurs, or when the effect can be approximated in a probabilistic manner. The VESPA RMP has been designed to have exceptionally low execution times, as the optimisation approach adopted may require that millions, rather than hundreds, of race simulations be performed.

6 · OPTIMISATION ALGORITHM

Chapter 4 described the use of a neural network metamodel, which approximates hydrodynamic data derived from the SPLASH potential flow code for variations to a parent ACC hull design.

In order to find the best design within the parameter space, it is necessary to explore this space using some form of optimisation algorithm. This chapter examines the various optimisation methods available and describes the factors determining the selection of a particular optimisation methodology for VESPA.

6.1 OPTIMISATION METHODS

Many different approaches to non-linear parameter optimisation exist. These can be broadly divided into three categories:

- Gradient based approaches. These methods calculate first, and in some cases, second derivatives of the function being optimised, in order to approximate the function to determine the next step in the optimisation process. Examples include the Newton-Raphson, quasi-Newton, Levenberg-Marquardt and conjugate gradient methods.
- Pattern search methods do not rely on derivative information, relying instead on comparison between multiple evaluations of the function performed in a

fixed pattern. The best known of these methods are the Nelder-Mead downhill simplex and the Hooke-Jeeves pattern search.

- Stochastic optimisation methods are those that introduce some element of randomness in their search for an optimal solution, rather than proceeding in a totally deterministic way. These methods include simulated annealing, particle swarm optimisation and evolutionary algorithms.

Given this degree of choice, the selection of the most suitable optimisation method is typically determined by the availability and reliability of the derivative information; the presence of noise in the function; and the potential for the function to be multimodal.

In the case of ACC yacht design optimisation, several other factors restrict the choice of optimisation method. The use of a competitive fitness function, i.e. one that is based on a comparison of the performance of one yacht against a variety of others, requires an entire population of boats to be modelled and improved. When the fitness of a design is based on its performance relative to its competitors, the fitness landscape is dynamic, changing with each step in the optimisation. The optimisation method adopted should be capable of handling such a dynamic fitness landscape.

In the case of VESPA, the optimisation algorithm needs to meet the following requirements:

- capable of handling non-linear and multi-modal solutions;
- able to evaluate competitive fitness functions;
- capable of finding solutions to problems having dynamic fitness landscapes.

6.2 GRADIENT AND PATTERN SEARCH METHODS

Although the need for a population of solutions and the presence of a dynamic fitness landscape bring with them some challenges, they do constrain the choice of optimisation method. The dynamic nature of the fitness landscape makes it unlikely that a gradient based method (e.g. Newton-Raphson or Quasi-Newton) or search pattern based method (e.g. Nelder-Mead simplex or Hooke-Jeeves direct search) would behave in a stable fashion. The fact that a population of boats needs to be modelled and evaluated makes stochastic, population-based approaches, such as Genetic Algorithms and Particle Swarm methods a natural choice.

Consequently, traditional gradient based or search pattern based methods can be eliminated from the search for a suitable optimisation method for this research. This is not to say that these methods are not applicable if the structure of the objective function differed. For example, if a Volvo 70 yacht design intended for an around the world yacht race was being optimised, the measure of merit would be concerned with elapsed time on the course rather than performance against a single opponent. In this case, the fitness landscape will be static, and a gradient-based optimisation method may be faster and give more accurate results.

6.3 STOCHASTIC METHODS

There are three widely used approaches to stochastic optimisation, as illustrated by Figure 56. Within the category of Evolutionary Algorithms the taxonomy branches further to include three subgroups.

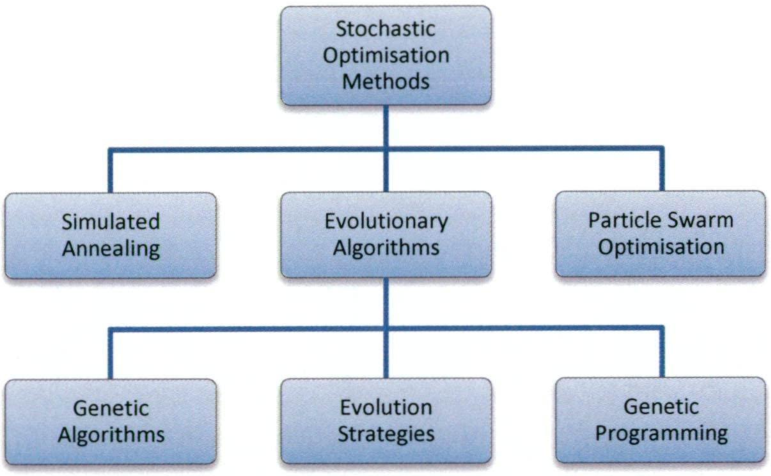


Figure 56. A taxonomy of stochastic optimisation methods

6.3.1 SIMULATED ANNEALING

Simulated Annealing (SA) is a global optimisation method analogous to annealing in metallurgy, the controlled cooling of a material to increase the size of its crystals. At higher temperatures, atoms are free to move from their initial energy states and randomly move through states of higher energy; slow cooling increases the probability of finding configurations with lower internal energy than the initial one.

The SA algorithm proceeds by replacing the current solution with a nearby one, chosen at random with a probability that depends on a global temperature value T. At the commencement of the optimisation process, T is large, allowing greater movements in the parameter space and permitting solutions that may be worse than the current one. As the optimisation proceeds, T is reduced resulting in smaller steps and a smaller allowance for worse solutions. The allowance for solutions that may be worse than the current one helps to prevent the method becoming stuck in local optima, and it is this feature that distinguishes SA from a simple search method.

SA was found independently by Kirkpatrick et al. (1983), and by Černý (1985) and is an adaptation of the Metropolis-Hastings algorithm, described by Metropolis et al. (1953). Chen (1996) utilised a simulated annealing algorithm to optimise the principal dimensions of a ship, noting that the method was able to avoid a local extreme point and find the global optimal point, regardless of the choice of initial point. Morishita and Akagi (2005) found significant reductions in the calm water resistance of both monohull and catamaran fast-ferry hulls, using a simulated annealing method.

Although SA is by definition a stochastic rather than deterministic optimisation method, it optimises a single design rather than being population based, making it unsuitable for the population-based measure of merit required for VESPA.

6.3.2 PARTICLE SWARM OPTIMISATION

The Particle Swarm Optimisation (PSO) method was created by Eberhart and Kennedy (1995). A stochastic, population-based method, PSO was inspired by the behaviour of a flock of birds flocking or a school of fish. Potential solutions to the problem under investigation, referred to as particles, are initially placed randomly throughout a multi-dimensional parameter space. Each particle has three pieces of knowledge: the best solution that it has found along its trajectory through the parameter space; the best solution encountered in the particle's neighbourhood; and the global best solution found by any other particle.

These particles are set in motion and the direction and velocity of the motion of the particles is influenced by the particles which have achieved the best solutions to the problem being evaluated according to the following rules:

$$v_{id} \leftarrow w_i v_{id} + c_1 r_1 (p_{id} - x_{id}) + c_2 r_2 (p_{gd} - x_{id}) \tag{6.1}$$

$$x_{id} \leftarrow x_{id} + v_{id} \tag{6.2}$$

Where v is velocity, x is position, w is the inertia weight, d is the dimension, c_1 and c_2 are positive constants, r_1 and r_2 are random functions, p_i is personal best and p_g is global best.

The algorithm proceeds through a series of time steps, updating the location and evaluating solutions for each particle. The size of the swarm and its initial distribution determines how effectively the global space is explored, while the communication between members of the swarm will result in eventual convergence on the best solution found. PSO methods are relatively simple to implement as there are no mutation and recombination operators involved as there are with evolutionary algorithms.

Despite apparent advantages, PSO methods have not shown themselves superior to well implemented genetic algorithms for single-objective parameter optimisation problems. Tan (2003) found that GA outperformed PSO methods on single objective aerodynamic shape optimisation problems, although for multi-objective problems the PSO method was better able to locate the Pareto front. Pinto et al. (2005; 2007) successfully used PSO methods for multi-objective optimisation of merchant ships and found that PSO found a greater number of Pareto optimal solutions.

6.3.3 EVOLUTIONARY ALGORITHMS

Evolutionary algorithms are search methods inspired by the process of evolution that occurs in biological species. Originating in research by Holland into cellular automata, (Holland 1975), evolutionary algorithms quickly differentiated into three disciplines, Genetic Algorithms (GA), Evolution Strategies (ES) and Genetic Programming (GP).

All three methods share the key features of an evolutionary algorithm –

- **Population based.** The methods use a population of candidate designs, each represented as a vector of real valued or binary numbers, or as a tree structure.
- **Ranking.** A procedure that evaluates and ranks individuals from the population using an objective function.
- **Selection.** A selection procedure that chooses individuals, based on their ranking or fitness, to have their genes represented in the following generation.
- **Recombination.** The recombination of characteristics from selected individuals to create new individuals for the following generation.
- **Mutation.** Small variations made to the genome on a random basis in order to introduce diversity into the population.

Although there are many similarities between the methods, the differences between them are also significant. Genetic programming, based on the evolutionary programming of Fogel et al. (1966) and refined by Cramer (1985) and Koza (1992), is primarily concerned with the evolution of computer programs.

Rather than using a simple vector of numbers, genomes for genetic programming tend to be based on tree structures where each node is a small segment of code. Although a very powerful approach for particular problems, GP is not applicable to the problem under investigation and can be eliminated as a candidate for an optimisation method.

There are two major philosophical differences between the fields of GA and ES; the format used to encode parameters in the genome and the emphasis placed on the recombination and mutation operators.

ENCODINGS

Evolution Strategies, (Rechenberg 1973; Schwefel 1974), were designed from the outset to use a floating-point representation. Conversely, genetic algorithms initially used a binary representation, as the building block hypothesis and its associated schema theorem, which had been formulated as theoretical underpinnings for genetic algorithms by Holland (1975), implied an incompatibility with floating-point representations.

Although the arguments by Holland and Goldberg (1989) in favour of binary representations dominated genetic algorithm research for many years, more recent work has cast doubt on the theoretical advantages of binary representations. Janikow and Michalewicz (1991) and Wright (1991) demonstrated that real coded genetic algorithms outperformed binary in a majority of test problems. Real coded GAs have become widely used, particularly for the solution of parameter optimisation problems where a binary representation would be difficult due to concerns related to magnitude or precision.

MUTATION AND RECOMBINATION

Although canonical Evolution Strategies emphasise the use of mutation as the primary search operator, recently developed ES have introduced more powerful recombination operators. These developments in ES, when combined with the increasing use of floating-point representations for GA, have served to blur the traditional distinctions between the two approaches.

The different emphasis placed on mutation and recombination remains the primary difference between real-valued GA and ES. ES typically use high rates of mutation, with highly specialised mutation operators, together with low levels of crossover. In contrast, a typical GA will have higher rates of recombination coupled with low mutation rates.

OPTIMISATION ALGORITHM SELECTION

In a comparison of the performance of GA, ES, GP and SA methods for two deceptive test problems, Keane (1996) demonstrated that basic implementations of GP, ES and SA were all outperformed by a GA.

Previous research into the optimisation of marine vessels has indicated that the solution space is often non-linear and multimodal. Several researchers, including

Sahoo (1997), Yasukawa (2000), Hirata (2004) and Pinto (2004) found that optimisation algorithms based on gradient information or local pattern searches achieved inferior results to global optimisation methods, for problems related to the hydrodynamics of marine vessels.

Based on the factors described, a floating-point based genetic algorithm was chosen as the optimisation method for use by VESPA. Some mutation and crossover methods from the field of ES have also been adopted. The selection of the various parameters, operators and settings used by the VESPA GA are described in the following section.

6.4 PROPOSED GA METHODOLOGY

The steps involved in a genetic algorithm are as follows:

```
1: Initialise the population
2: Evaluate initial population
3: REPEAT
4:    Perform competitive selection
5:    Apply genetic operators to generate new solutions
6:    Evaluate solutions in the population
7: UNTIL some convergence criteria is satisfied
```

Each iteration of a GA involves a competitive selection that penalises poor solutions. Solutions with high fitness are recombined with others to produce members of the next generation, each member inheriting properties from its parents. Solutions are also mutated by making small, random changes to one or more free parameters.

The effect of recombination and mutation is to generate new solutions, which are biased towards regions of the solution space from which good solutions have already been seen.

6.4.1 GA PARAMETERS

A GA based search can be viewed as a trade off between the maintenance of sufficient diversity in the population to permit the coverage of the entire solution space (exploration), and the need to converge on the optimal solution within an acceptable time (exploitation), (Eiben and Schippers 1998).

There are several variables or algorithms that can be adjusted to improve the performance of the GA, or to manipulate the degree of exploration or exploitation that the GA exhibits, (Goldberg 1989).

These include:

- recombination
- mutation
- selection
- population size
- fitness sharing
- elitism

6.4.2 RECOMBINATION

The theory of recombination in biological organisms was first detailed by Morgan (1916), who described both single point and dual point crossover in simple chromosomes, as shown in Figure 57.

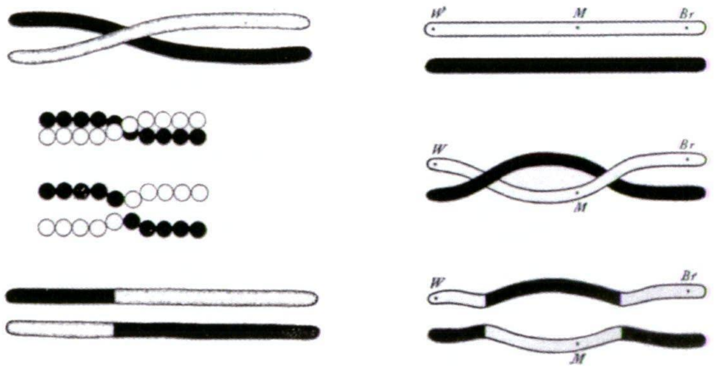


Figure 57. Single-point and dual-point crossover, from Morgan (1916)

Recombination within a binary genome of a GA is a similar procedure. A crossover locus is selected at random and the bits on one side of this locus are exchanged between the two parent genomes (Figure 58). This crossover is directly analogous to the recombination that occurs in sexual reproduction within single chromosome (haploid) organisms.

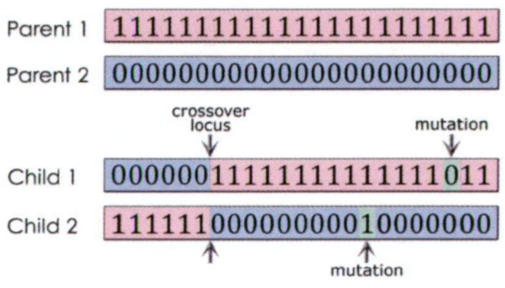


Figure 58. Binary recombination and mutation

Crossover for floating-point representations presents a more difficult problem. Although crossover can be performed by exchanging portions of two floating-point vectors, this does not change individual values for the exchanged vector elements. A binary representation, on the other hand, may perform crossover at any point in the binary string. If the genome is made up of a vector of 8-bit integers, crossover can occur at the integer boundaries, or within the integer itself, resulting in the changes to the values of numbers represented as bit strings. For vectors of floating-point numbers, crossover can only occur at the level of the numbers themselves, not within their binary representations.

Many schemes have been proposed for recombination operators for real-valued GAs. These include operators that exchange values between chromosomes:

- **Single-point crossover**: a crossover location $i \in \{1, 2, \dots, n-1\}$ is randomly chosen. Two new offspring chromosomes H^1 and H^2 are created from parent chromosomes C^1 and C^2:

$$\begin{aligned} H^1 &= (c_1^1, c_2^1, \dots, c_i^1, c_{i+1}^2, \dots, c_n^2) \\ H^2 &= (c_1^2, c_2^2, \dots, c_i^2, c_{i+1}^1, \dots, c_n^1) \end{aligned} \quad (6.3)$$

- **Dual-point crossover**: two crossover locations $i, j \in \{1, 2, \dots, n-1\}$ with $i < j$ are randomly chosen. Two new chromosomes are created:

$$\begin{aligned} H^2 &= (c_1^2, c_2^2, \dots, c_i^2, c_{i+1}^1, \dots, c_j^1, c_{j+1}^2, \dots, c_n^2) \\ H^1 &= (c_1^1, c_2^1, \dots, c_i^1, c_{i+1}^2, \dots, c_j^2, c_{j+1}^1, \dots, c_n^1) \end{aligned} \quad (6.4)$$

- **Discrete crossover**: h_i is a randomly selected value from the set $\{c_i^1, c_i^2\}$

A recombination method for a real-valued GA may also perform a weighted blend between two or more chromosomes:

- **Line crossover**: an offspring $H = (h_1, \dots, h_i, \dots, h_n)$ is generated, where h_i is a randomly chosen value on the interval $[c_i^1, c_i^2]$.
- **Extended line crossover**, (Muhlenbein and Schlierkamp-Voosen 1993):

$$h_i = c_i^1 + \alpha(c_i^2 - c_i^1) \quad (6.5)$$

  where α is a randomly selected value on the interval $[-0.25, 1.25]$

- **Extended intermediate crossover**:

$$h_i = c_i^1 + \alpha_i(c_i^2 - c_i^1) \quad (6.6)$$

  where α_i is a randomly selected value on the interval $[-0.25, 1.25]$

- **Directional crossover**, (Yamamoto and Inoue 1995b):

  $$H = c_i^1 + S \cdot sign(F_i^1 - F_i^2) \cdot (c_i^1 - c_i^2) + T \cdot sign(F_i^1 - F_i^3) \cdot (c_i^1 - c_i^3) \tag{6.7}$$

  where S and T are random values on the interval [0.0,1.0] and F is the value of the fitness function for the corresponding vector of variables c.

  Yamamoto and Inoue define c^1 as belonging to generation j, while c^2 and c^3 are its parents from generation $j - 1$. However, c^2 and c^3 may also be derived from generation j.

  Directional crossover creates a bias for the offspring in the direction of the parents with the greatest fitness. This bias, combined with the use of three parent points forming a simplex in the search space, introduces elements of pattern search methods, with evident similarities to the Nelder-Mead Simplex method. While this may speed convergence, it may also sacrifice some of the exploration capabilities of the GA.

The recombination methods selected for use by the VESPA GA are dual-point crossover and directional crossover. These are applied in the proportion 80:20 to avoid premature convergence resulting from high levels of directional crossover.

6.4.3 MUTATION

Mutation is straightforward when using binary representation, achieved by random flipping of individual bits within the bit string that forms the genome. Recommendations for the probability of mutation for each individual bit have been determined empirically by several researchers, including de Jong (1975) and Schaffer et al. (1989), and are typically in the range 0.005 – 0.01.

The implementation of mutation operators for real-coded GAs is not as simple as for binary representations. Mutation of a floating-point vector genome may be performed by randomly perturbing individual elements within the vector. As with the binary representation, vector elements are selected using a mutation probability. This value is typically greater for the floating-point number than would be used to mutate the individual bits that would be used to represent that number in a binary genome.

Once a vector element has been selected, its value has a small random perturbation added to it. This perturbation is generated from a normal distribution using a standard deviation specific to the vector element.

Levels of mutation that are too high can seriously disrupt the evolution process, while low mutation levels can slow convergence considerably, particularly in algorithms such as ES that do not emphasise recombination operators. Mutation for real coded parameters has been extensively researched by the ES community with schemes for adaptive mutation and covariance matrix based mutation widely used.

The VESPA GA uses random mutation with probability of 0.1 per design variable. Perturbations are sampled randomly from a normal distribution having a standard deviation equal to 0.05 times the parameter range (i.e. upper limit – lower limit) for that design variable.

6.4.4 SELECTION

Much research has been done into different selection methods with two of the most widely used methods being roulette wheel selection and tournament selection.

Roulette wheel selection assigns a likelihood of representation in the following generation proportional to the magnitude of an individual's fitness score. The difficulty inherent in the use of roulette wheel selection is that the likelihood of representation in the following generation is dependent on the scaling factor used to scale the results of the objective function to create fitness scores. Changing this scaling factor can dramatically change the results of the selection process, and it is difficult to determine the correct scaling factor ahead of time.

Tournament selection works by choosing two individuals from the population and comparing their fitness scores. The individual with the highest fitness score is selected to be a parent for an individual in the subsequent generations.

As tournament selection is a simple comparison of fitness values, it is independent of scaling factors. Tournament selection is both easier to implement and more robust in use than roulette wheel selection.

Consequently, tournament selection has been adopted for use by the VESPA GA due to its inherent simplicity and robustness. The absence of scaling issues is seen as a distinct advantage in an environment where the range of objective values may vary dramatically due to the dynamic fitness function used by VESPA.

6.4.5 POPULATION SIZE

For a GA, the smaller the population, the faster each generation will run. On the other hand, it is vital that the population size is large enough to maintain sufficient diversity for evolution to progress.

Despite the widespread use of Genetic Algorithms there seems to be little agreement on the ideal population size. A review of the literature reveals population sizes between 10 and 300 in use, with few researchers providing justification for their choice of population size. Grefenstette et al. (1986) studied optimal values for control parameters, reporting that a population size of 30 gave the best results. Similarly, Schäffer et al. (1989) obtained the best results using a population size of 20 to 30 individuals.

In previous work by the author (2005) a sensitivity analysis was performed for population sizes ranging between 10 and 150, for a multimodal, non-linear problem that involved finding the minimum wave resistance of a catamaran. Negligible differences in the rate of convergence or the fitness level of the best solution were observed for populations greater than 20 individuals.

6.4.6 FITNESS SHARING

Fitness sharing encourages diversity in the population by scaling the fitness of individuals based on how similar they are to all other individuals in the population. Individuals that are very similar to others in the population are given a slight penalty, while individuals that show novel features are rewarded by increasing their fitness relative to the population. This is intended to ensure that the population maintains sufficient diversity to avoid premature convergence on false optima.

Fitness sharing was shown by Goldberg and Richardson (1987) to be effective in maintaining population diversity, which helps to prevent premature convergence on local rather than global optima. While this may be of use during the exploration phase, this diversity may inhibit convergence to an optimal solution during the exploitation phase.

Although the roulette wheel selection method appears to benefit from fitness sharing, Oei et al. (1991) found that fitness sharing can display chaotic interactions when combined with tournament selection.

While fitness sharing has some attractive properties, the potential for chaotic interactions with the dynamic fitness function and the tournament selection method used by VESPA outweigh the benefits. Fitness sharing has not been adopted for use with the VESPA GA at this time.

6.4.7 ELITISM

Elitism ensures that the next generation's best individual will be at least as good as any from the previous generation by automatically including the best individual from the previous population.

De Jong (1975) found that elitism improved the performance of a genetic algorithm for unimodal functions, while performance for multimodal functions was degraded. This suggests that while elitism improves local search and may be appropriate during the later exploitation phase of an optimisation, it may also inhibit global exploration.

Rudolph (1994) showed that while genetic algorithms without elitism are not guaranteed to converge to a global optimum, a GA with suitable mutation operators and elitism is guaranteed to reach a global optimum if given sufficient time.

VESPA applies simple elitism to all generations of the genetic algorithm. In addition, VESPA uses more complex elitism methods, necessitated by the competitive fitness function adopted.

COMPETITIVE FITNESS

An objective function that is based on a comparison of an individual in competition with a variety of other individuals from the population, is termed a competitive fitness function. Competitive fitness indicates that the problem has a dynamic fitness landscape that may change from one generation of an evolutionary algorithm to the next.

Competitive fitness commonly takes one of two forms:

- one-population competitive fitness;
- N-population competitive fitness, also referred to as competitive co-evolution. This form encompasses competition between multiple species, including parasite-host and predator-prey co-evolution.

Competitive fitness has advantages and disadvantages. On the positive side, competitive fitness can act as a self-adapting mechanism to increase problem difficulty, as individuals in the population adapt themselves to the co-evolutionary environment. A co-evolutionary arms race can also act to speed up the evolution process significantly.

On the other hand, competitive fitness may suffer from problems that serve to inhibit convergence of the genetic algorithm. These problems include:

- **disengagement** occurs when the gradient of the fitness values for the population flatten, leading to stalling or drifting of the evolutionary process. This may be accompanied by a loss of diversity in the population caused by premature convergence on a solution, which may be mistaken for a global optimum.
- **cycling** is a repeated pattern of visitation to the same areas of the solution space. Cycling is most likely to occur if the solution contains intransitivities i.e. B defeats A, C defeats B, but then A defeats C. However, de Jong (2004) demonstrated that cycling can also occur for purely transitive relationships.

These pathologies may be inhibited by the use of strategies that encourage diversity in the population such as fitness sharing. One approach that has been favoured is the use of a "hall-of-fame" (Rosin and Belew 1996), which saves good individuals from previous generations.

The performance of an evolutionary algorithm based on competitive fitness is also dependent on the topology of the competition adopted. Panait and Luke (2002) identified several different structures for one-population competitive fitness:

- round robin
- random pairing
- K-random pairing
- single elimination tournament

The choice of competition topology determines the type of fitness assessment that is used. Single elimination tournaments result in a *duel methodology*: A is better than B if and only if A usually beats B in a match. Round robin tournaments result in the *Renaissance man methodology*: A is better than B if A beats more competitors than B, even if B usually beats A in a match.

Some sporting contests use a combination of the two methodologies. For the America's Cup, challengers compete in a round robin competition to qualify for a single elimination tournament for a small number of boats. The winner of this tournament then competes in an elimination match with the defender of the America's Cup.

The implication for ACC design optimisation is that, as the tournament structure is asymmetric, in that the challenger and defender experience different paths to the final elimination match, the optimal design for a challenger may not be the same as for the defender.

EXEMPLARS AND THE "HALL-OF-FAME"

In order to prevent the occurrence of disengagement and cycling, two related measures, based on the "hall-of-fame" approach of Rosin and Belew, were adopted for use by the VESPA GA.

A simple hall-of-fame approach involves the inclusion in the population of the best individual from each of the previous generations, effectively the extension of elitism to encompass all generations. This approach is not guaranteed to be successful, as the hall-of-fame members may not be sufficiently numerous or diverse to prevent cycling or disengagement.

An alternative approach is the inclusion in the population of an additional elite individual that has been selected based on an objective measure other than the competitive fitness function.

For VESPA, one candidate for this objective measure is the average elapsed course time for a yacht in the absence of competitors. While the boat with the lowest average elapsed time is not likely to be the boat that wins the most races, due to the bias introduced by trailing boat penalties, this boat is always likely to be a strong competitor, and its presence should assist to prevent cycling of the solution.

A second strategy was also adopted in an attempt to avoid cycling of the solution. This strategy involved the creation of individuals with specific design variables, termed exemplars, which are included in the round-robin tournament used by the GA. For a design to be considered optimal, it must be capable of outperforming these exemplars.

The use of exemplars is effectively a manual "hall-of-fame" method. By specifying the exemplars directly, the method has the advantage that the design parameters of specific competitors can be added to the tournament, allowing comparison and evolution of proposed designs against known opponents.

7 · IMPLEMENTATION

In order to create VESPA, a workable optimisation system for the design of ACC yachts, it was necessary to create and integrate several key functions:

- a powerful parametric transformation function to allow VESPA to vary hull shapes based on a small number of key parameters, while still producing hull shapes that are both acceptable to the design team and legal under the ACC Rule, (ACC 2003);
- sampling of the design space using a modern Design of Experiments (DOE) method. The parametric transformation function is used to create hulls, derived from a specified parent model, matching these sampled parameter values;
- analysis of lift, drag and hydrodynamic heeling moment for the hulls in the systematic series, using the SPLASH potential flow code;
- the use of neural networks to fit the SPLASH data for all hulls in the systematic series to produce a global regression model, or metamodel, for lift, drag and other parameters;
- customisation of a proven Velocity Prediction Program (VPP) to read the neural network based metamodel and to allow it to be called as a Dynamically Linked Library (DLL) from another application;

- a stochastic Race Modelling Program (RMP) that creates a tournament of races for the population of boats. This RMP uses performance data derived from the VPP to race each boat against every other multiple times, in wind conditions sampled from a specified distribution of wind velocity;
- a Genetic Algorithm (GA) based optimiser that takes the parent ACC hull design, which is used to generate the lift and drag models, and evolves a population of variations using the parametric transformation function.

7.1 INTEGRATION OF COMPONENTS

The components of VESPA are integrated in the manner illustrated in Figure 59. There are two main parts to the system, the creation of the metamodel, which occurs in advance of the optimisation process being performed, and the optimisation loop itself.

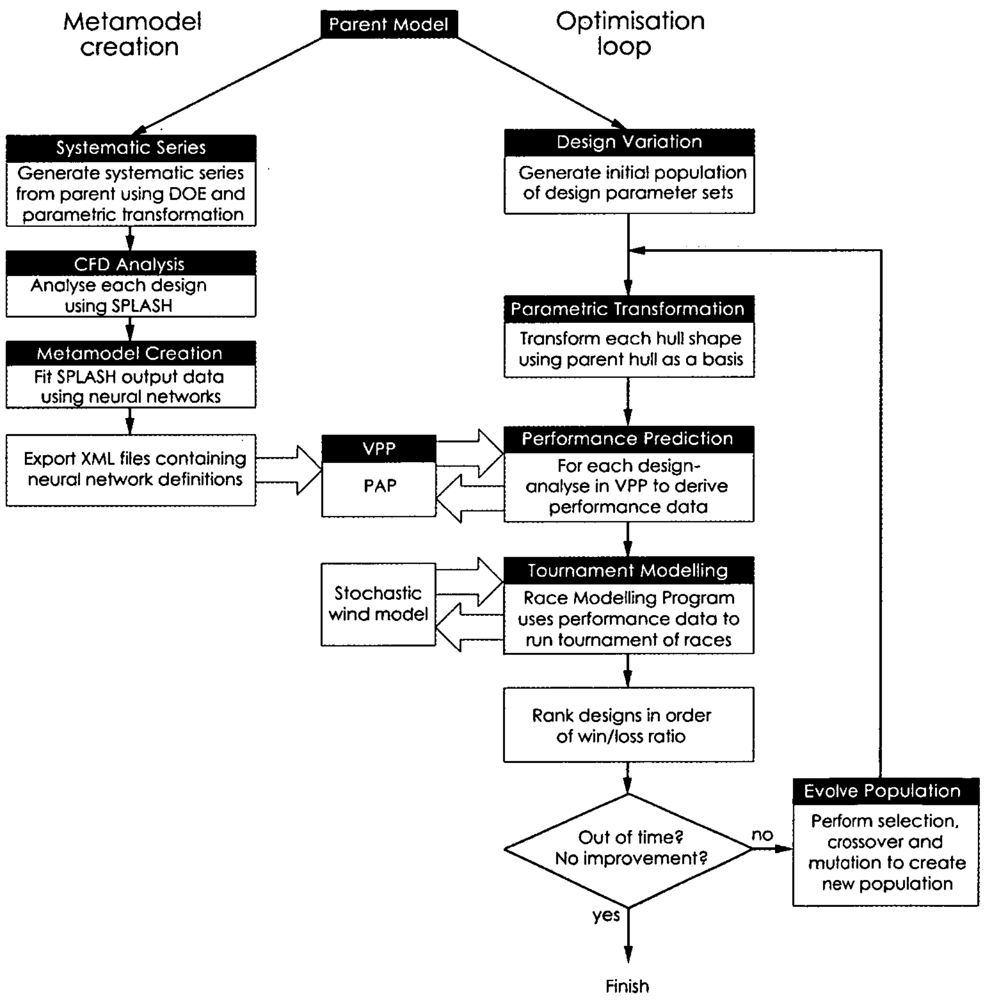

Metamodel creation
Parent Model
Optimisation loop
Systematic Series
Generate systematic series from parent using DOE and parametric transformation
CFD Analysis
Analyse each design using SPLASH
Metamodel Creation
Fit SPLASH output data using neural networks
Export XML files containing neural network definitions
VPP
PAP
Design Variation
Generate initial population of design parameter sets
Parametric Transformation
Transform each hull shape using parent hull as a basis
Performance Prediction
For each design-analyse in VPP to derive performance data
Stochastic wind model
Tournament Modelling
Race Modelling Program uses performance data to run tournament of races
Rank designs in order of win/loss ratio
Out of time? No improvement?
no
yes
Evolve Population
Perform selection, crossover and mutation to create new population
Finish

Figure 59. VESPA logic flow

7.2 PARAMETRIC TRANSFORMATION FUNCTION

Analysis of the restrictions imposed by the version 5.0 ACC rule, combined with discussions with the key members from the Alinghi design team, determined that the design variables with the most scope for optimisation by VESPA were topside flare, B_{WL}, T_c, C_M, L_{CB} and C_P.

The parametric transformation function was developed in two stages:

- **Stage one** allowed generic parametric transformation of marine hull shapes. It performed linear scaling of L_{WL}, B_{WL}, T_c and ∇, as well as non-linear variations of L_{CB} and C_P. As both L_{CB} and C_P are coefficients (L_{CB} is specified as a fraction of waterline length) these are invariant under the subsequent linear transformations required to achieve the desired L_{WL}, B_{WL}, T_c and ∇ values.

  Note that although any three of the four linear scaling parameters (L_{WL}, B_{WL}, T_c and ∇) may be specified, it is not possible to simultaneously specify all four, as this leads to an over-constrained problem. If less than three of the parameters are specified, these are varied such that the ratio between the free variables is kept constant. For example, if L_{WL} and ∇ are set to specific values while B_{WL} and T_c are left as free variables, B_{WL} and T_c will be scaled accordingly while keeping $\frac{B_{WL}}{T_c}$ constant.

- **Stage two** implemented transformations specific to the additional design variables required by VESPA for ACC yachts. These variables were topside flare and C_M, as well as three constraints specific to the ACC rule, *LBG, FG* and *AG.* Hull shape convexity, another requirement of the ACC rule, was also required as a constraint.

  Although ACC yachts are restricted in length, this is measured in the *LBG* plane, placed 200 mm above the datum waterline, rather than along the datum waterline. This has a significant effect on the design of the parametric transformation function, as ∇ is no longer a linear scaling of L_{WL}, B_{WL} and T_c.

  The effect of measuring length in the *LBG* plane is that length ceases to be independent of changes to L_{CB} or C_P. For example, while keeping the length measure constant, a change to C_P that resulted in a small change to ∇ could now not be compensated for by making a single, proportional change to B_{WL} or T_c; instead, an iterative search must be conducted to find the correct value.

The non-linear form variations, such as L_{CB} and C_P, require a search procedure to find the hull shape that satisfies the design parameters specified. This was initially handled using a Nelder-Mead pattern search. However, the additional demands imposed by the non-linear transformations required to satisfy flare, *LBG*, C_M, *FG* and *AG* values made it necessary to implement a Newton-Raphson solver in its place.

The resulting parametric transformation, when combined with a parent model of sufficient quality, provides the ability to create a wide range of realistic, class legal hull shapes based on a very small number of meaningful parameters. In the current implementation, the variables chosen were B_{WL}, C_P, C_M, flare and L_{CB}, with *FG*, *AG*, *LBG* constrained to within 1 mm of their maximum values, and displacement constrained to within 1 kg of 24,000 kg.

7.2.1 HIERARCHICAL FREE-FORM DEFORMATION

To perform a parametric transformation using a hierarchical FFD, a hull design represented by a single NURBS surface is required. Such a surface has a control-point network **C** having $p+1$ rows and $q+1$ columns. The bounding box of the projection of each control point column into the (y,z) plane is mapped into the unit square, as shown in Figure 60:

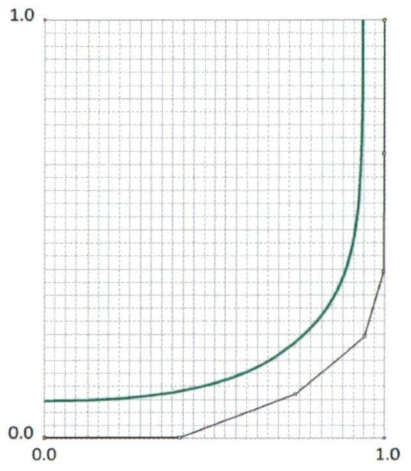


Figure 60. Control points mapped into the unit square

Each control point in the column has its parametric co-ordinates (v,w) in the unit square calculated from its (y,z) co-ordinates using:

$$\mathbf{v}_{i,j} = \frac{y_{i,j} - min\,(y_i)}{max\,(y_i) - min\,(y_i)}$$

$$\mathbf{w}_{i,j} = \frac{z_{i,j} - min\,(z_i)}{max\,(z_i) - min\,(z_i)} \tag{7.1}$$

For each control-point column, a planar bi-linear NURBS surface is created whose corner control point (y,z) co-ordinates match those of the corners of the bounding box for that column, and whose x co-ordinates are equal to the mean value of the x co-ordinates of the control points in that column. Such planar NURBS surfaces are illustrated in Figure 61.

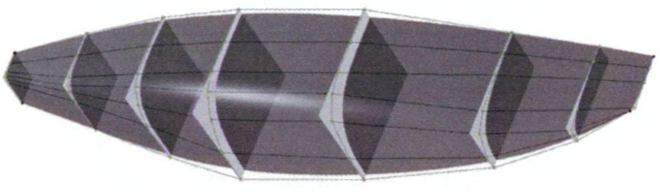


Figure 61. Bounding boxes displayed for each control point column in **C**

This array of planar NURBS surfaces **Q** is enclosed by a tri-variate NURBS volume, as shown in Figure 62. This NURBS volume is defined by a lattice of control points $\mathbf{P}_{i,j,k}$, having $l+1$ control points in the s direction, $m+1$ control points in the t direction, and $n+1$ control points in the u direction. In this case, $l=2,\ m=1,$ and $n=1$, with degree $p=2,\ q=1,\ r=1.$

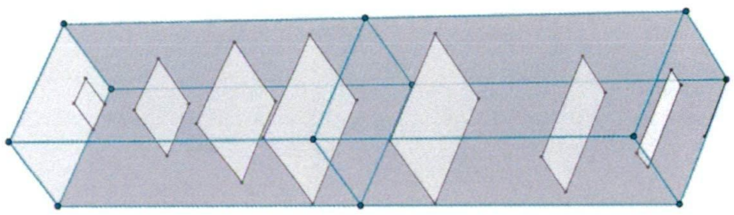


Figure 62. FFD planes enclosed by FFD lattice **P**

The (s,t,u) parametric values for the corner points of each planar NURBS surface are calculated using:

$$s = \frac{x - x_{min}}{x_{max} - x_{min}}$$

$$t = \frac{y - y_{min}}{y_{max} - y_{min}}$$

$$u = \frac{z - z_{min}}{z_{max} - z_{min}} \tag{7.2}$$

Changes to the (x,y,z,w) values of the FFD control points result in an amended FFD lattice co-ordinates $\mathbf{P}'_{i,j,k}$ and amended lattice weights $w'_{i,j,k}$. These changes result in deformation of the embedded planar NURBS surfaces, as shown in Figure 63, which can be calculated using:

$$\mathbf{Q}' = \frac{\sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} N_{i,p}(s)\, N_{j,q}(t)\, N_{k,r}(u) w'_{i,j,k}\, \mathbf{P}'_{i,j,k}}{\sum_{i=0}^{l}\sum_{j=0}^{m}\sum_{k=0}^{n} N_{i,p}(s)\, N_{j,q}(t)\, N_{k,r}(u) w'_{i,j,k}} \tag{7.3}$$

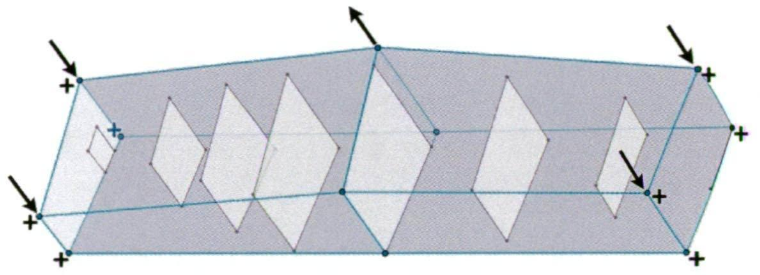


Figure 63. Deformed FFD lattice **P'** with updated weights w', resulting in deformations to the embedded planar NURBS surfaces **Q'**

The updated control point values for each modified planar NURBS surfaces $\mathbf{Q}'$, together with modifications to the planar NURBS weight values $w'_{i,j}$ are used to create an updated hull-surface control point network $\mathbf{C}'$.

$$\mathbf{C}'_{(y,z)} = \frac{\sum_{i=0}^{i=1} \sum_{j=0}^{j=1} N_{i,2}(\mathbf{V})\, N_{j,2}(\mathbf{W})\, w'_{i,j}\mathbf{Q}'_{i,j}}{\sum_{i=0}^{i=1} \sum_{j=0}^{j=1} N_{i,2}(\mathbf{V})\, N_{j,2}(\mathbf{W}) w'_{i,j}}$$

$$\mathbf{C}'_{(x)} = \mathbf{Q}_{(x)} \tag{7.4}$$

The result is a hull surface that has been smoothly deformed to vary B_{WL}, C_{P}, C_{M}, flare, L_{CB}, FG and AG in a controlled manner. An example of such a transformation is shown in Figure 64.

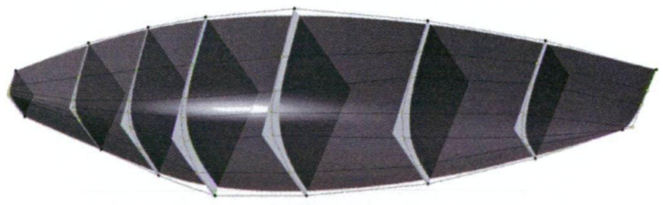


Figure 64. Deformed hull surface defined by updated control point network **C'**

This transformation was achieved by changing only ten values:

- the y co-ordinates of five control points in the FFD lattice;
- two FFD weight values, one applied simultaneously to the four forward FFD lattice points and the other applied simultaneously to the four aft FFD lattice points;
- three weight values applied simultaneously to each embedded planar NURBS surface. These local weight changes are used to change the sectional shape of the hull.

7.3 SYSTEMATIC SERIES CREATION

To create a suitable metamodel on which an optimisation process can be based, it is necessary to sample the design space to obtain a set of values which span the ranges of the design variables required. These sampled values define a systematic series of hull forms such as those shown in Figure 65.

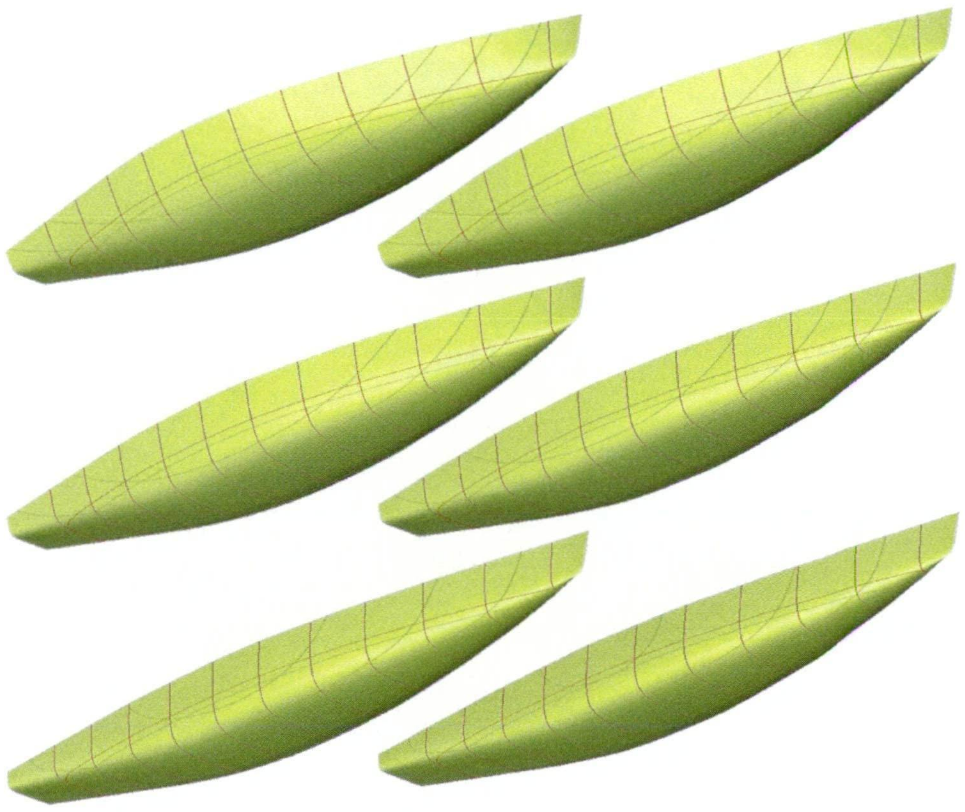
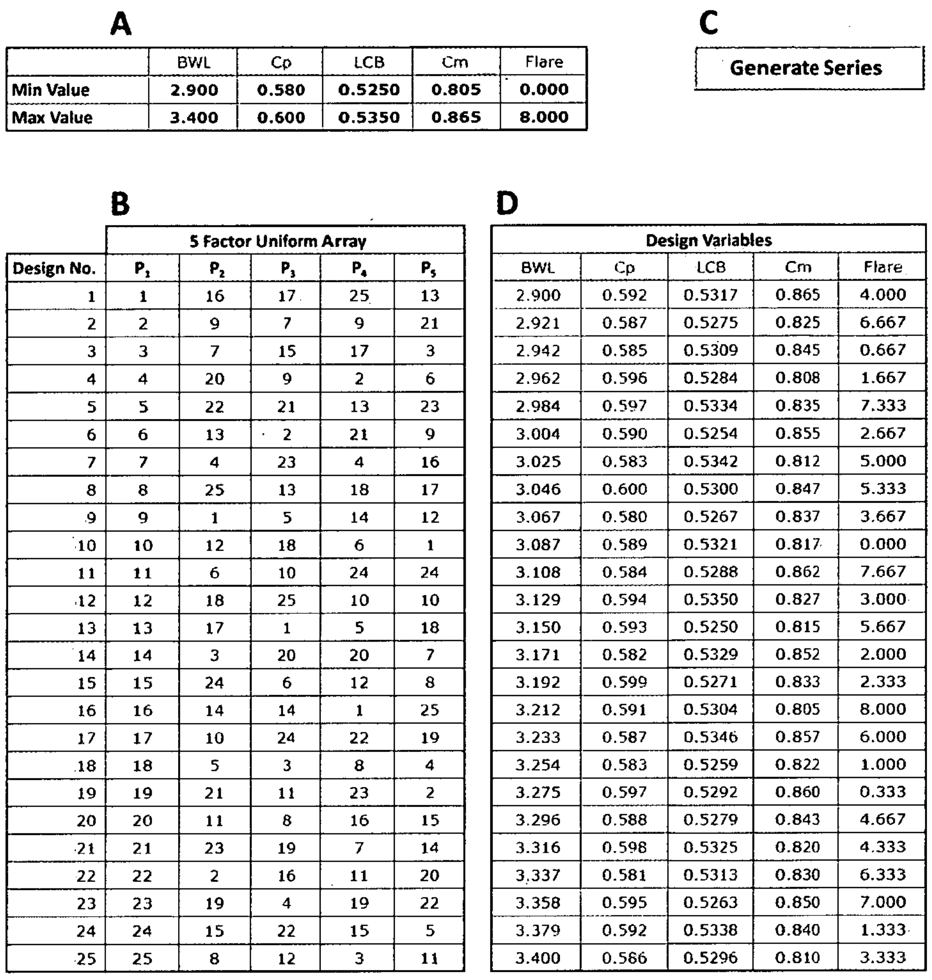


Figure 65. A simple systematic series of hulls, varying in C_P, B_{WL}, C_M , T and flare

Sampling based on either regular grids or random samples display significant problems when used for the fitting of metamodels. It is therefore necessary to use a form of sampling that has superior performance to both a regular grid and a truly random sample. These methods are known as quasi-random samples, or in some cases, as low discrepancy sequences, and display distributions that are highly uniform in all dimensions.

In order to create the hull forms required from these sampled parameters, the parametric transformation described in Section 7.2 is used to transform a parent hull to match the required values.

An Excel, spreadsheet shown in Figure 66, was created to generate families of hull shapes by, deforming a parent hull form to match the parameters required.



A

	BWL	Cp	LCB	Cm	Flare
Min Value	2.900	0.580	0.5250	0.805	0.000
Max Value	3.400	0.600	0.5350	0.865	8.000

C

Generate Series

B / D

	5 Factor Uniform Array					Design Variables				
Design No.	P1	P2	P3	P4	P5	BWL	Cp	LCB	Cm	Flare
1	1	16	17	25	13	2.900	0.592	0.5317	0.865	4.000
2	2	9	7	9	21	2.921	0.587	0.5275	0.825	6.667
3	3	7	15	17	3	2.942	0.585	0.5309	0.845	0.667
4	4	20	9	2	6	2.962	0.596	0.5284	0.808	1.667
5	5	22	21	13	23	2.984	0.597	0.5334	0.835	7.333
6	6	13	2	21	9	3.004	0.590	0.5254	0.855	2.667
7	7	4	23	4	16	3.025	0.583	0.5342	0.812	5.000
8	8	25	13	18	17	3.046	0.600	0.5300	0.847	5.333
9	9	1	5	14	12	3.067	0.580	0.5267	0.837	3.667
10	10	12	18	6	1	3.087	0.589	0.5321	0.817	0.000
11	11	6	10	24	24	3.108	0.584	0.5288	0.862	7.667
12	12	18	25	10	10	3.129	0.594	0.5350	0.827	3.000
13	13	17	1	5	18	3.150	0.593	0.5250	0.815	5.667
14	14	3	20	20	7	3.171	0.582	0.5329	0.852	2.000
15	15	24	6	12	8	3.192	0.599	0.5271	0.833	2.333
16	16	14	14	1	25	3.212	0.591	0.5304	0.805	8.000
17	17	10	24	22	19	3.233	0.587	0.5346	0.857	6.000
18	18	5	3	8	4	3.254	0.583	0.5259	0.822	1.000
19	19	21	11	23	2	3.275	0.597	0.5292	0.860	0.333
20	20	11	8	16	15	3.296	0.588	0.5279	0.843	4.667
21	21	23	19	7	14	3.316	0.598	0.5325	0.820	4.333
22	22	2	16	11	20	3.337	0.581	0.5313	0.830	6.333
23	23	19	4	19	22	3.358	0.595	0.5263	0.850	7.000
24	24	15	22	15	5	3.379	0.592	0.5338	0.840	1.333
25	25	8	12	3	11	3.400	0.566	0.5296	0.810	3.333

Figure 66. Hull-series generation spreadsheet

The spreadsheet consists of four parts:

- Section A allows upper and lower limits for each design variable to be specified. The independent variables used for the creation of the systematic series are B_{WL}, C_M, C_P, L_{CB}, topside flare.
- Section B lists the Uniform Design matrix for 25 runs (i.e. designs) and 5 factors (i.e. design variables). This matrix was derived from uniform design tables provided online (Fang 2004), and forms a five dimensional Latin hypercube, with points uniformly distributed in each dimension.
- Button C runs an Excel macro to populate the systematic series shown in Section D and creates hull geometry using the parametric transformation function. This macro calls the VESPA parametric transformation function via a Microsoft COM interface.

- Section D lists the design variable values created by combining the values in sections A and B using:

$$x_{i,j} = min(x_i) + (max(x_i) - min(x_{i,}))\frac{P_{i,j} - 1}{max(P_i) - 1} \tag{7.5}$$

  where x is an array of design variable values, P is an array of values derived from the Uniform Design, i is an index to the variable type, j is an index to the design number, while $minx$ and $maxx$ are the upper and lower limits for each design variable type.

Once the parametric transformation has been performed, the resulting hull definitions for the systematic series are exported as IGES NURBS surface files. These are subsequently used as input to SPLASH, a potential-flow CFD program.

In addition to the initial series of 25 hull shapes, two additional groups of hulls were created. The first group of 7 hulls was created using design variables chosen at random within the defined design space at the same time as the initial series, and analysed using SPLASH. The SPLASH results for these hull shapes were not used for neural network training, being reserved solely for use as an independent test set for validation of the prediction accuracy of the neural network metamodel.

The second group of designs, also 7 hulls, was defined once a significant amount of optimisation had been performed using the initial neural-network metamodels. This group of designs was created using a test matrix based on the Uniform Design, but using ranges for each design variable that were restricted to approximately 20% of the initial range, centred on the values found to be optimal by VESPA.

The intention for this second set of designs was to allow some local refinement of the neural network metamodels in the vicinity of the design optimum, so that the optimal design may be estimated more accurately. In this regard, the additional designs function as a rudimentary variable-fidelity model.

7.4 SPLASH ANALYSIS

SPLASH is the potential flow code selected by the Alinghi design team for its CFD analysis work. Execution times are reasonable, with a 200 point matrix of parameter variations taking approximately 12 hours on a single CPU. This performance is approximately two orders of magnitude faster than an equivalent RANS code.

SPLASH executes a matrix of test runs covering various combinations of hull velocity, heel, yaw, rudder angle and trim tab angle. This matrix of test points is also defined using the Uniform-Design quasi-random DOE method, in order to provide uniform spacing of result data for the metamodel fitting process.

Hydrodynamic data were calculated by SPLASH for each design, using the following independent variables:

- $V_{B,}$
- heel,
- yaw,
- rudder angle – set proportional to heel angle,
- trim tab angle – fixed angle for upwind, zero degrees downwind.

Dependent variables were:

- lift,
- residuary resistance,
- hydrodynamic heeling moment,
- wetted surface area of the canoe body,
- waterline length.

Note that although SPLASH is capable of calculating total resistance as well as residuary resistance, in this case the Alinghi designers preferred to use the residuary resistance value and to calculate the frictional resistance independently, using the wetted surface area and waterline length provided by SPLASH.

7.5 NEURAL NETWORK METAMODEL CREATION

Neural network metamodels were created with the commercial neural network package NeuroIntelligence, (Alyuda 2005), using the output data created by SPLASH. Each of these metamodels used the following as independent variables: B_{WL}, C_M, C_P, L_{CB}, topside flare, $V_{B,}$, heel, yaw.

Metamodels were created for each family of designs, for each of the following values:

- lift,
- upright residuary resistance, R_{RU},
- heeled residuary resistance delta, $(R_{R\varphi} - R_{RU})$,
- hydrodynamic heeling moment,
- wetted surface area S of the canoe body in m^2,
- waterline length, L_{WL} in m.

Metamodels were output from Neurointelligence as XML network definition files.

7.6 VPP

VESPA was designed to work with any VPP that could be compiled as a DLL and operated through a simple function call interface.

During its initial prototyping phase, VESPA was integrated with the SPAN VPP, originally written by the author as part of the Maxsurf suite of design software. However, SPAN was not specifically adapted to suit ACC yachts, and as a result was not able to perform accurate ACC performance prediction.

For ACC optimisation, VESPA was interfaced to PAP, a VPP developed by a member of the Alinghi design team, Manuel Ruiz de Elvira. PAP was originally developed for the 2000 America's Cup and was subsequently used for the design and analysis of SUI-64, the winner of the 2003 America's Cup. PAP is a reliable, well-validated program used by the design team of the current holder of the America's Cup, and was seen as an ideal VPP on which to base an optimisation system.

VPPs typically use a regression model for hull lift and drag based on a series of tank tests known as the Delft series (Gerritsma et al. 1981; Keuning and Sonnenberg 1999). Some VPPs, such as PAP, also have the capability to use lift and drag data for a specific hull, derived from either tank test results or CFD calculations. This allows more accurate performance prediction for a particular yacht. In order to work with VESPA, this capability was extended to incorporate neural-network based metamodels so that lift, drag and hydrodynamic heeling moment could be estimated for variations to a parent ACC hull.

NeuroIntelligence, the neural network software used for the creation of metamodels for VESPA, allows an XML file describing the architecture and weights of a trained neural network to be written as output. PAP was modified to read these metamodel definition XML files. Once read by PAP, the data contained in these files are used to reconstruct the neural networks to allow hydrodynamic data to be derived for any variation of the parent hull design required by VESPA. An example of an XML neural-network definition is listed in Appendix 1.

In order to utilise this file format, it was necessary to write software that could read the file and re-create the neural network from the definition contained in the XML data. This neural network software module, named XML_NN, was written in such a way that it could be compiled and linked directly into a C++ program, or alternatively, the code could be compiled as a Microsoft Windows Dynamically Linked Library (DLL).

Compiling as a DLL has several advantages. Firstly, it allows the software to be used with a variety of other programs, even if these programs were written in different languages. For example, the DLL was able to be called from within an Excel spreadsheet, permitting extensive validation to take place utilising standard spreadsheet functions and charting facilities. The DLL was also able to be linked with a Visual BASIC application and function as an integral part of that application. This is the approach that was taken with the PAP VPP, which was modified to link to the XML_NN DLL.

The XML_NN module was designed to be able to reproduce any neural network architecture that could be created within NeuroIntelligence. XML_NN permits up to five hidden layers to be specified, with up to 1000 neurons in each layer. Linear, sigmoidal or hyperbolic tan activation functions may be selected on a layer by layer basis. A listing of the header file for the XML_NN module, showing the class structures, is provided in Appendix 2.

7.6.1 DLL LINK FROM VPP TO OPTIMISER

In addition to the PAP VPP calling the XML_NN module as a DLL, it was also necessary for the VESPA optimiser to call the PAP VPP as a DLL. There are several reasons for VESPA to use dynamically linked modules that are assembled at runtime.

VPPs are highly specialised programs, which have typically been developed over a long period by particular design groups and America's Cup teams. The internal workings of these programs are proprietary and confidential, making it unlikely that the author of such a VPP would be willing to provide source code or libraries to enable a VPP to be compiled by an outside party.

By specifying an interface to a DLL, rather than requiring code to be compiled and statically linked together, it is possible for each group providing software components to keep their source code confidential.

This also facilitates updating or changing components, such as the VPP used, without the need to recompile the whole system. DLLs may also be written in any one of a number of languages. As long as the new VPP is compatible with the calling interface of the DLL, it may be dynamically linked at runtime with no other changes required.

PAP was modified to allow it to be compiled as a DLL and called from VESPA. A listing of the header file for calling the PAP VPP module is provided in Appendix 3.

7.7 RACE MODELLING PROGRAM

The VESPA RMP was coded in C++ and is tightly integrated with VESPA's genetic algorithm code. The code covering the calculation of leg and race times from V_{MG} values for each discrete step is shown in Appendix 4. C++ code for the simulation of races between two boats, including random time variation and trailing boat penalties, is listed in Appendix 5.

The VESPA race model consists of six legs, together totalling 20 nautical miles. Legs are divided into 60 steps of approximately 100 metres, each step having its own wind velocity and direction determined by sampling the wind distribution specified for the event. V_{MG} (or V_{MC} if wallying is applied) is calculated for each step and the total time for the leg accumulated.

One-hundred different sets of race conditions are created, each containing wind velocity and direction for every step of the 6 race legs. Every boat in the population, which typically consists of 25 individuals, has its leg times calculated for each of the races. Once this is done, each boat is raced against every other boat (e.g. 25×24 races) for each of the 100 races, and each race is repeated 100 times with random time variation included.

Each race consists of looking up the pre-calculated times for each boat for each leg in turn. Times at the end of the leg are adjusted for random variation, port tack penalty and trailing boat penalty. The adjusted times for each boat are then used as the starting point for the following leg. This process is repeated until all legs are completed, at which point the race is concluded and the winning boat has its winning tally incremented.

The random time penalty applied at the end of each leg is normally distributed with a standard deviation of six seconds. While it is possible that the amount of variation added to the system should vary based on the wind velocities encountered during each leg, with lighter winds having less variation and higher winds having more, the analysis required to quantify these differences was beyond the scope of this research. A sensitivity analysis, described in Section 8.8, showed that adding only a small amount of random variation to the system had similar results to adding much greater amounts of noise. It was therefore considered that using a random variation appropriate to the mean wind velocity should not adversely affect results at the extremes of the scale of wind velocities.

Once the random time variation has been incorporated, an analysis of starboard tack advantage for windward marks is performed, with a penalty applied to port tack boats that are neither clear ahead nor clear astern. This penalty eliminates any overlap between the yachts, equivalent to a port tack yacht being forced to cross behind a starboard tack yacht on an upwind leg.

This is followed by the application of a penalty for the trailing boat, the magnitude of which is a function dependent on the time difference between the boats. This penalty reduces in a linear fashion, with a maximum penalty of six seconds when the two yachts are nearly overlapped, reducing to zero when the leading yacht is ahead by twenty seconds.

The 25×24 round robin implies that yachts A and B race each other twice (A vs. B and B vs. A). However, these races are not the same, as the advantage of the starboard entry to the starting line is allocated to the first boat in each pairing. For two boats that are closely matched, this starboard end advantage may be the deciding factor in the outcome of the race, and it is essential that this advantage is equalised.

7.7.1 WIND MODEL

VESPA uses an approximation of the wind velocity distribution defined by mean wind velocity, standard deviation, and upper and lower wind limits. To verify that this approach was a reasonable approximation to actual conditions, an analysis of historical wind records for Valencia for the months of June and July was performed.

Figure 67 shows a histogram of wind velocities measured during June 2005, between the hours of 13:00 and 16:00, together with a normal distribution curve having a mean of 10 knots and a standard deviation of 2.5 knots. The historical wind distribution is well approximated by the normal distribution in this case, despite the short period during which data was collected.

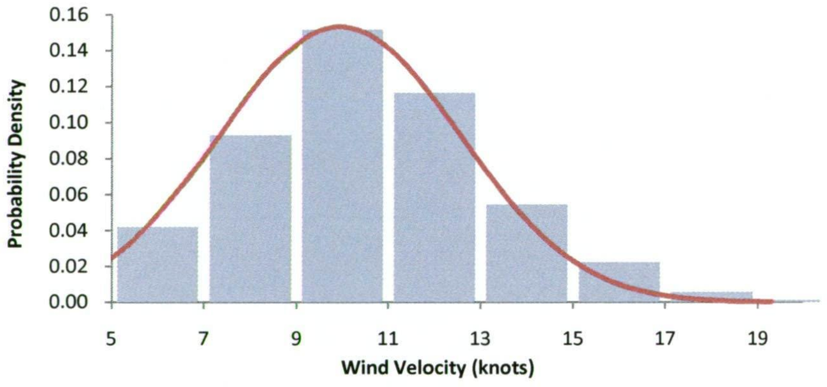


Figure 67. Comparison of wind velocity distributions

An analysis of meteorological data spanning a ten-year period resulted in the use by VESPA of a mean wind-strength of 11 knots, a standard deviation for the America's Cup event of 4 knots and a standard deviation for each race of 1.4 knots. A lower limit of 6 knots and an upper limit of 20 knots were adopted to correspond with wind strength limits used for actual racing.

7.7.2 WALLYING

To apply wallying within VESPA, a standard deviation for the wind direction is required, along with the extent to which wallying will be applied: 0%, 50% or 100%. The 100% value requires that the boat sails at the maximum wally value at all times. This is not a realistic assumption, as it implies complete certainty as to the future changes in wind direction.

The 50% wallying setting can be thought of as using 100% wallying at the start of the upwind leg, tapering linearly to 0% by the time the top mark is reached. This case is more realistic, as it suggests that, at the start of the leg, the sailors will be confident that a lift will revert to the mean before the top mark is reached, but that this confidence in a shift reverting to the mean will progressively reduce as they get closer to the layline for the mark.

7.8 GENETIC ALGORITHM

A floating-point genetic algorithm was written in C++ using template-based classes. A listing of the Evolver.h header file, showing the class structures for the genetic algorithm code, is provided in Appendix 6.

The use of C++ templates allowed the GA code to be extremely flexible, separating the representation of the genome from the mechanics of the evolution process. The representation of the genome is defined by the Genome virtual base class and this may be overridden to use a binary representation, a floating point representation or even a mixture of the two. Similarly, genes are not restricted to being a binary or floating point value, they may be entire data structures containing additional information. For example, each gene could be set up store its own mutation rate, and this could be evolved in parallel with the rest of the genome.

Mutation and recombination operators are also defined within the genome class, making customisation of the GA straightforward and self-contained.

GA OPERATORS

The following genetic operators were adopted, based on recommendations derived from previous research by the author, (Mason et al. 2005):

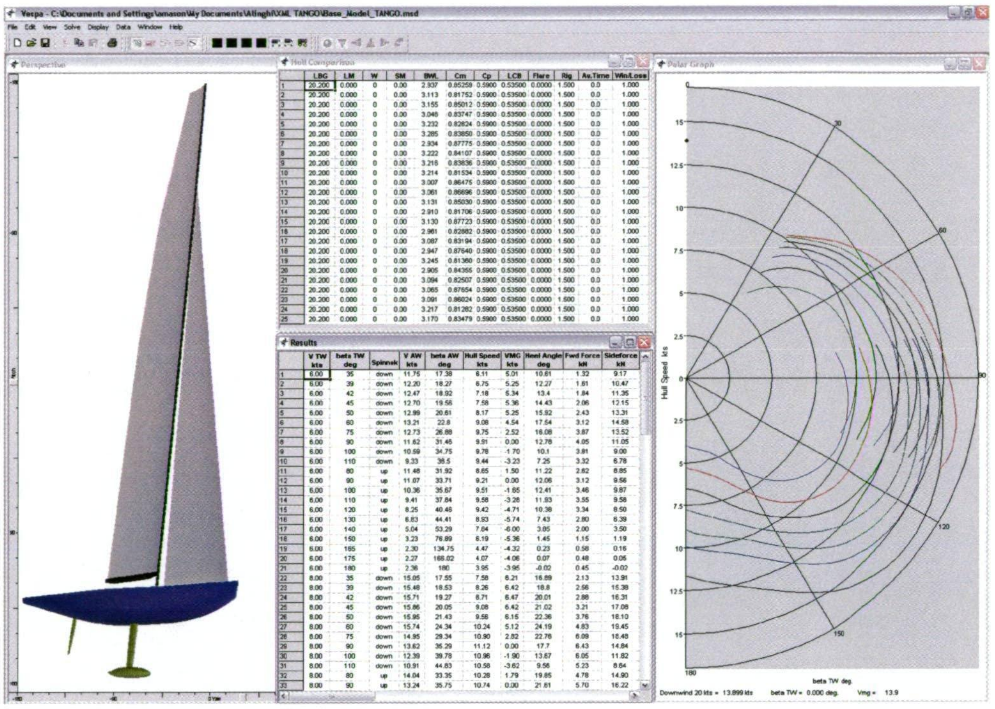
- dual-point crossover;
- directional crossover;
- Gaussian distributed mutation;
- elitism;
- tournament selection.

Input variables to the GA were bounded to lie within the ranges specified for the systematic series used for the fitting of the lift and drag metamodels. Values outside these limits are unlikely to be predicted accurately.

Applying strict bounds to the input variables resulted in no infeasible designs being encountered. However, future work that explores greater variable ranges will need to consider the possibility that infeasible designs may be generated.

7.9 USER INTERFACE

VESPA was implemented as a Microsoft Windows program with multiple viewing windows for hull and rig geometry, VPP results, polar performance graphs and optimisation results. A typical layout showing these four main windows is shown in Figure 68.


Vespa - C:\Documents and Settings\amason\My Documents\Alinghi\XML TANGO\Base_Model_TANGO.msd
File Edit View Solve Display Data Window Help
Perspective
Hull Comparison
Results
Polar Graph
Hull Speed kts
beta TW deg
Downwind 20 kts = 13.899 kts
beta TW = 0.000 deg.
Vmg = 13.9

	LBG	LM	W	SM	BWL	Cm	Cp	LCB	Flare	Rig	Av.Time	Win/Loss
1	20.200	0.000	0	0.00	2.937	0.85259	0.5900	0.53500	0.0000	1.500	0.0	1.000
2	20.200	0.000	0	0.00	3.113	0.81752	0.5900	0.53500	0.0000	1.500	0.0	1.000
3	20.200	0.000	0	0.00	3.155	0.85012	0.5900	0.53500	0.0000	1.500	0.0	1.000
4	20.200	0.000	0	0.00	3.046	0.83747	0.5900	0.53500	0.0000	1.500	0.0	1.000
5	20.200	0.000	0	0.00	3.232	0.82824	0.5900	0.53500	0.0000	1.500	0.0	1.000
6	20.200	0.000	0	0.00	3.285	0.83850	0.5900	0.53500	0.0000	1.500	0.0	1.000
7	20.200	0.000	0	0.00	2.934	0.87775	0.5900	0.53500	0.0000	1.500	0.0	1.000
8	20.200	0.000	0	0.00	3.222	0.84107	0.5900	0.53500	0.0000	1.500	0.0	1.000
9	20.200	0.000	0	0.00	3.216	0.83836	0.5900	0.53500	0.0000	1.500	0.0	1.000
10	20.200	0.000	0	0.00	3.214	0.81534	0.5900	0.53500	0.0000	1.500	0.0	1.000
11	20.200	0.000	0	0.00	3.007	0.86475	0.5900	0.53500	0.0000	1.500	0.0	1.000
12	20.200	0.000	0	0.00	3.061	0.86696	0.5900	0.53500	0.0000	1.500	0.0	1.000
13	20.200	0.000	0	0.00	3.131	0.85030	0.5900	0.53500	0.0000	1.500	0.0	1.000
14	20.200	0.000	0	0.00	2.910	0.81706	0.5900	0.53500	0.0000	1.500	0.0	1.000
15	20.200	0.000	0	0.00	3.130	0.87723	0.5900	0.53500	0.0000	1.500	0.0	1.000
16	20.200	0.000	0	0.00	2.961	0.82882	0.5900	0.53500	0.0000	1.500	0.0	1.000
17	20.200	0.000	0	0.00	3.087	0.83194	0.5900	0.53500	0.0000	1.500	0.0	1.000
18	20.200	0.000	0	0.00	2.947	0.87640	0.5900	0.53500	0.0000	1.500	0.0	1.000
19	20.200	0.000	0	0.00	3.245	0.81360	0.5900	0.53500	0.0000	1.500	0.0	1.000
20	20.200	0.000	0	0.00	2.905	0.84355	0.5900	0.53500	0.0000	1.500	0.0	1.000
21	20.200	0.000	0	0.00	3.094	0.82507	0.5900	0.53500	0.0000	1.500	0.0	1.000
22	20.200	0.000	0	0.00	3.065	0.87654	0.5900	0.53500	0.0000	1.500	0.0	1.000
23	20.200	0.000	0	0.00	3.091	0.86024	0.5900	0.53500	0.0000	1.500	0.0	1.000
24	20.200	0.000	0	0.00	3.217	0.81282	0.5900	0.53500	0.0000	1.500	0.0	1.000
25	20.200	0.000	0	0.00	3.170	0.83479	0.5900	0.53500	0.0000	1.500	0.0	1.000

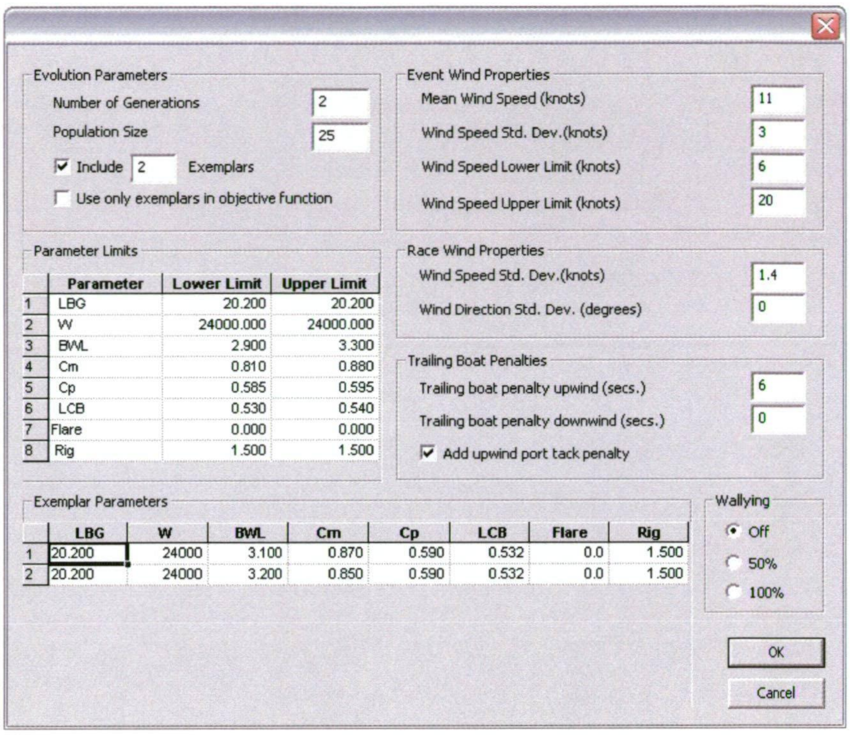
	V TW kts	beta TW deg	Spinnak	V AW kts	beta AW deg	Hull Speed kts	VMG kts	Heel Angle deg	Fwd Force kN	Sideforce kN
1	6.00	35	down	11.75	17.38	6.11	5.01	10.61	1.32	9.17
2	6.00	39	down	12.20	18.27	6.75	5.25	12.27	1.61	10.47
3	6.00	42	down	12.47	18.92	7.18	5.34	13.4	1.84	11.35
4	6.00	45	down	12.70	19.56	7.58	5.36	14.43	2.06	12.15
5	6.00	50	down	12.99	20.61	8.17	5.25	15.92	2.43	13.31
6	6.00	60	down	13.21	22.8	9.08	4.54	17.54	3.12	14.58
7	6.00	75	down	12.73	26.88	9.75	2.52	16.08	3.87	13.52
8	6.00	90	down	11.62	31.46	9.91	0.00	12.78	4.05	11.05
9	6.00	100	down	10.59	34.75	9.78	-1.70	10.1	3.81	9.00
10	6.00	110	down	9.33	38.5	9.44	-3.23	7.25	3.32	6.78
11	6.00	80	up	11.46	31.92	8.65	1.50	11.22	2.62	8.85
12	6.00	90	up	11.07	33.71	9.21	0.00	12.06	3.12	9.56
13	6.00	100	up	10.36	35.67	9.51	-1.65	12.41	3.46	9.87
14	6.00	110	up	9.41	37.84	9.58	-3.28	11.93	3.55	9.58
15	6.00	120	up	8.25	40.46	9.42	-4.71	10.38	3.34	8.50
16	6.00	130	up	6.83	44.41	8.93	-5.74	7.43	2.80	6.39
17	6.00	140	up	5.04	53.29	7.84	-6.00	3.85	2.00	3.50
18	6.00	150	up	3.23	76.89	6.19	-5.36	1.45	1.15	1.19
19	6.00	165	up	2.30	134.75	4.47	-4.32	0.23	0.58	0.16
20	6.00	175	up	2.27	166.02	4.07	-4.06	0.07	0.48	0.05
21	6.00	180	up	2.36	180	3.95	-3.95	-0.02	0.45	-0.02
22	8.00	35	down	15.05	17.55	7.58	6.21	16.89	2.13	13.91
23	8.00	39	down	15.48	18.53	8.26	6.42	18.8	2.56	15.38
24	8.00	42	down	15.71	19.27	8.71	6.47	20.01	2.88	16.31
25	8.00	45	down	15.86	20.05	9.08	6.42	21.02	3.21	17.08
26	8.00	50	down	15.95	21.43	9.56	6.15	22.36	3.76	18.10
27	8.00	60	down	15.74	24.34	10.24	5.12	24.19	4.83	19.45
28	8.00	75	down	14.95	29.34	10.90	2.82	22.76	6.09	18.48
29	8.00	90	down	13.82	35.29	11.12	0.00	17.7	6.43	14.84
30	8.00	100	down	12.39	39.78	10.96	-1.90	13.67	6.05	11.82
31	8.00	110	down	10.91	44.83	10.58	-3.62	9.56	5.23	8.64
32	8.00	80	up	14.04	33.35	10.28	1.79	19.85	4.78	14.90
33	8.00	90	up	13.24	35.75	10.74	0.00	21.81	5.70	16.22

Figure 68. VESPA application

Controls were provided for VESPA's optimisation parameters in a single dialog used to commence the optimisation process. These controls include:

- genetic algorithm parameters, including the number of generations and the size of the population;
- controls for including and defining exemplars where required;
- upper and lower limits for each design variable;
- design variable values for exemplars;
- parameters controlling the wind distribution used by the RMP;
- variables related to trailing boat penalties;
- Wallying options.

The dialog displaying these controls is shown in Figure 69.



Evolution Parameters

- Number of Generations: 2
- Population Size: 25
- [✓ Include] 2 Exemplars
- [ ] Use only exemplars in objective function

Event Wind Properties

- Mean Wind Speed (knots): 11
- Wind Speed Std. Dev.(knots): 3
- Wind Speed Lower Limit (knots): 6
- Wind Speed Upper Limit (knots): 20

Parameter Limits

	Parameter	Lower Limit	Upper Limit
1	LBG	20.200	20.200
2	W	24000.000	24000.000
3	BWL	2.900	3.300
4	Cm	0.810	0.880
5	Cp	0.585	0.595
6	LCB	0.530	0.540
7	Flare	0.000	0.000
8	Rig	1.500	1.500

Race Wind Properties

- Wind Speed Std. Dev.(knots): 1.4
- Wind Direction Std. Dev. (degrees): 0

Trailing Boat Penalties

- Trailing boat penalty upwind (secs.): 6
- Trailing boat penalty downwind (secs.): 0
- [✓ Add upwind port tack penalty]

Exemplar Parameters

	LBG	W	BWL	Cm	Cp	LCB	Flare	Rig
1	20.200	24000	3.100	0.870	0.590	0.532	0.0	1.500
2	20.200	24000	3.200	0.850	0.590	0.532	0.0	1.500

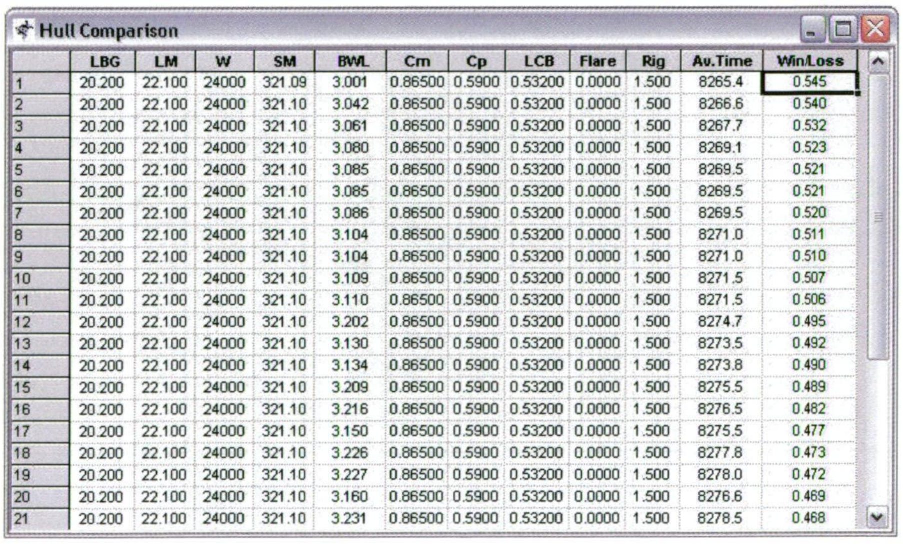
Wallying

[✓ Off] 50% 100%

Figure 69. VESPA optimisation controls

Once these options have been specified, clicking the OK button commences an optimisation run. Evaluation of a population of 25 yachts takes approximately 15-30 minutes per generation on standard PC hardware.

Once completed, results of the optimisation run are presented in VESPA's results window, with members of the population sorted in descending order of fitness, as shown in Figure 70.



Hull Comparison

	LBG	LM	W	SM	BWL	Cm	Cp	LCB	Flare	Rig	Av.Time	Win/Loss
1	20.200	22.100	24000	321.09	3.001	0.86500	0.5900	0.53200	0.0000	1.500	8265.4	0.545
2	20.200	22.100	24000	321.10	3.042	0.86500	0.5900	0.53200	0.0000	1.500	8266.6	0.540
3	20.200	22.100	24000	321.10	3.061	0.86500	0.5900	0.53200	0.0000	1.500	8267.7	0.532
4	20.200	22.100	24000	321.10	3.080	0.86500	0.5900	0.53200	0.0000	1.500	8269.1	0.523
5	20.200	22.100	24000	321.10	3.085	0.86500	0.5900	0.53200	0.0000	1.500	8269.5	0.521
6	20.200	22.100	24000	321.10	3.085	0.86500	0.5900	0.53200	0.0000	1.500	8269.5	0.521
7	20.200	22.100	24000	321.10	3.086	0.86500	0.5900	0.53200	0.0000	1.500	8269.5	0.520
8	20.200	22.100	24000	321.10	3.104	0.86500	0.5900	0.53200	0.0000	1.500	8271.0	0.511
9	20.200	22.100	24000	321.10	3.104	0.86500	0.5900	0.53200	0.0000	1.500	8271.0	0.510
10	20.200	22.100	24000	321.10	3.109	0.86500	0.5900	0.53200	0.0000	1.500	8271.5	0.507
11	20.200	22.100	24000	321.10	3.110	0.86500	0.5900	0.53200	0.0000	1.500	8271.5	0.506
12	20.200	22.100	24000	321.10	3.202	0.86500	0.5900	0.53200	0.0000	1.500	8274.7	0.495
13	20.200	22.100	24000	321.10	3.130	0.86500	0.5900	0.53200	0.0000	1.500	8273.5	0.492
14	20.200	22.100	24000	321.10	3.134	0.86500	0.5900	0.53200	0.0000	1.500	8273.8	0.490
15	20.200	22.100	24000	321.10	3.209	0.86500	0.5900	0.53200	0.0000	1.500	8275.5	0.489
16	20.200	22.100	24000	321.10	3.216	0.86500	0.5900	0.53200	0.0000	1.500	8276.5	0.482
17	20.200	22.100	24000	321.10	3.150	0.86500	0.5900	0.53200	0.0000	1.500	8275.5	0.477
18	20.200	22.100	24000	321.10	3.226	0.86500	0.5900	0.53200	0.0000	1.500	8277.8	0.473
19	20.200	22.100	24000	321.10	3.227	0.86500	0.5900	0.53200	0.0000	1.500	8278.0	0.472
20	20.200	22.100	24000	321.10	3.160	0.86500	0.5900	0.53200	0.0000	1.500	8276.6	0.469
21	20.200	22.100	24000	321.10	3.231	0.86500	0.5900	0.53200	0.0000	1.500	8278.5	0.468

Figure 70. VESPA results window

8 · VERIFICATION & VALIDATION

The nature of optimisation is that it will ruthlessly exploit weak data. Rather than minimising the function of interest, an optimisation procedure may simply locate the point of maximum disparity between simulation and reality. As stated by Karl L. Kirkman, one of the pioneers of racing yacht research:

> *"the first result, from my experience, of using optimization tools is to detect the flaws in the algorithms." (Oliver et al. 1987, p.261)*

The accuracy of the results of an optimisation system such as VESPA is dependent on the quality of both the source data and algorithms used. It is essential that the performance of these components be verified and validated, both individually, as well as when integrated into a complete system.

In the case of VESPA, this consisted of seven areas that required validation:

- parametric transformation;
- SPLASH results;
- neural network metamodel;
- accurate representation of the metamodel by the neural network DLL code;
- GA convergence and performance;
- VPP;
- RMP.

8.1 PARAMETRIC TRANSFORMATION VALIDATION

The parametric transformation function was first checked to verify that the hulls created matched all the design parameters specified. This was done by creating a spreadsheet macro that automatically created several hundred design variations of a parent model using the parametric transformation. Each of these was subsequently analysed to verify that its hydrostatic and dimensional parameters matched the values that had been used as inputs to the parametric transformation.

This testing resulted in some work being performed to improve the speed and convergence of the parametric transformation algorithm. Additionally, the method was refined to prevent introduction of hollows in the aft run of the hull that appeared when C_P was reduced or L_{CB} moved forward.

8.2 SPLASH VALIDATION

SPLASH is a potential flow code that has been widely used by America's Cup teams for more than twenty years (Rosen et al. 1993; Rosen et al. 2000; DeBord et al. 2002). During this period, SPLASH has been extensively compared and validated against tank test data and results from full scale, and as a result its capabilities and limitations are well understood (Reichel et al. 1994; DeBord et al. 2002).

Consequently, no further effort was expended to validate SPLASH output for ACC designs. Rather, results were examined critically and areas in which SPLASH was considered to be giving potentially unreliable results were restricted in their range of applicability. One example of this was the favouring by VESPA of designs with high C_P and aft L_{CB} values. This trend was noted early during VESPA's testing period, and as the designs recommended as optimal had consistently higher C_P and L_{CB} values than the best models from tank testing, the decision was made to restrict C_P and L_{CB} ranges during optimisation.

This is not to say that the SPLASH data was in error, as it may have been that the narrow B_{WL}, high C_M designs favoured by VESPA genuinely benefitted from higher C_P and L_{CB} values. Rather, the limited time and tank testing resources available made it impractical to attempt to validate this design direction. As a result, C_P and L_{CB} values were restricted to known good values.

Final validation of the SPLASH predictions used by VESPA consisted of the comparison of SPLASH data and tank data for the parent and optimised hull. In the case where tank testing was performed, the deltas between the two hulls as predicted by SPLASH and the tank corresponded well, as described in Section 9.5.3. However, difficulties were noted in the ability of SPLASH to accurately predict the impact of extreme values of some design parameters, such as combinations of high heel and yaw angles.

8.3 METAMODEL TRAINING & VALIDATION

To simplify the neural network fitting task, output data from SPLASH were modified to reduce the degree of non-linearity in the data. Lift, drag and hydrodynamic heeling moment values were divided by $0.5\rho V^2$. While this did not make the hydrodynamic heeling moment truly non-dimensional, removing the V^2 relationship from the heeling moment data did assist with the approximation task.

The dataset was subsequently partitioned into three categories:

- **Training Set**. The discrepancies between the output values contained in the training set and the values predicted by the neural network are used by the training algorithm to determine changes to the network weights during neural network training.
- **Validation Set**. The validation set is used as an independent measure of error. Points in the validation set are not used for training the neural network and as a result, may be used to indicate whether the network is generalising (i.e. successfully predicting unknown points) or over-fitting (i.e. having improved prediction accuracy for training-set data, but worse prediction accuracy for validation set data). Over-fitting is generally indicative of an excess of neurons in the hidden layer of the neural network.
- **Test Set**. The test set is comprised of data that has been put aside for the sole purpose of providing an unbiased estimator of the prediction accuracy of the trained neural networks. In this regard, the test set is not used for any purpose that may result in it introducing bias into the training process.

  Such bias may be an issue with the validation set, as it may be used to select the best network at the completion of training. Consequently, the neural network may be biased towards the contents of the validation set, even though none of the points in the validation set were specifically used for training the network.

  For the VESPA metamodels, the independence of the test set was also encouraged by the use of different test matrices for the test set hulls. Rather than using the Uniform Design based sampling adopted for the training and validation data sets, the test set design variables were generated randomly so that their design variable values did not match any regular spacing.

  Similarly, the test set hulls did not use a Uniform Design based matrix for their array of hull velocities, heel and yaw angles. Instead, the sampling used the Alinghi tank test matrix values, a full factorial matrix focussed around likely combinations of hull velocity, heel and yaw.

Partitioning of the training set, validation set and test set approximated the ratio 50% : 25% : 25%.

TRAINING ALGORITHMS

NeuroIntelligence contains several different training algorithms, including Back Propagation, Quickprop, Conjugate Gradient, Quasi-Newton (QN), and Levenberg-Marquardt (LM) methods. These algorithms differ significantly in their performance characteristics and convergence properties, as shown in Table 8, which lists the times taken to train a neural network to fit upright drag data to an RMS error of 1.0×10^{-6}.

Table 8. Comparison of Neural Network Training Algorithms

Method	Number of iterations	Iterations / second	Elapsed time (s)
Backpropagation (Batch)	unconverged after 100,000	n/a	n/a
Backpropagation (Online)	unconverged after 100,000	n/a	n/a
Conjugate Gradient	6,827	8.34	818.6
Quickprop	10,573	65.11	162.4
Quasi-Newton	1,190	10.15	117.2
Levenberg-Marquardt	99	2.81	35.2

Despite the LM method significantly outperforming all other training algorithms in this testing, it did not prove to be a suitable choice as the sole neural network training method, as it appeared to stop prematurely before it had fully converged.

In a test where the LM method was used to train a neural network to the lowest error achievable for heeled drag data, with training repeated 10 times to ensure the best possible outcome, the results in row 1 of Table 9 were recorded.

When the QN method was used to train the network to the same error value achieved by the LM method (Table 9, row 2), the elapsed time was more than thirty times greater than that taken for a single run of the LM method.

Table 9. Convergence Properties of LM and QN Methods

Run No.	Method	No. of iterations	Iterations /second	Elapsed time (s)	AAE
1	LM (best of 5)	53	0.92	57.5	2.3560
2	QN	6,075	3.93	1546.1	2.3556
3	LM (run 1) + QN (average of 3 runs)	5,000	3.90	1307.1	**1.7838**
4	QN (run 2) + QN (average of 3 runs)	5,000	3.88	1805.6	1.8959
5	QN	11,075	3.73	2975.6	1.9747

The results of runs 1 and 2 were both able to be improved significantly with further training using the QN method. When each run was followed by an additional 5,000 iterations of the QN method, the LM + QN sequence outperformed the QN + QN sequence, as shown in rows 3 and 4. Finally, an unbroken run of 11,075 iterations of the QN method gave slightly worse results than the QN + QN sequence, showing that stopping and restarting the QN algorithm did not dramatically affect the outcome and may have been beneficial.

These results are typical of those experienced during training of SPLASH data using NeuroIntelligence. The NeuroIntelligence implementation of the LM method was able to converge rapidly on a good solution, but stopped prematurely. The QN method, on the other hand, was slow to find solutions of equivalent quality when starting from scratch, but given the best result provided by the LM method, was able to refine it significantly. This testing indicated that the best approach to neural network training within NeuroIntelligence was to use repeated runs of the LM method for initial training, followed by refinement using the QN method.

During initial metamodel fitting work, training was terminated if the validation error was stable or worsening for several thousand iterations, as it was considered that no further improvement was likely to be seen. While this appears to have been a valid assumption for the standard backpropagation and Quickprop algorithms, the quasi-Newton algorithm repeatedly showed that it was able to make significant improvements after long periods of apparent stability in the validation set error values.

An example of the benefits of extended training times is shown in Figure 71. In this case, validation set error values worsen between 10,000 and 30,000 iterations. At this point, typical early stopping criteria would halt the training process. If training was allowed to continue, a very long period of no improvement follows until 240,000 iterations, at which point the validation error undergoes a dramatic reduction of approximately 20%.

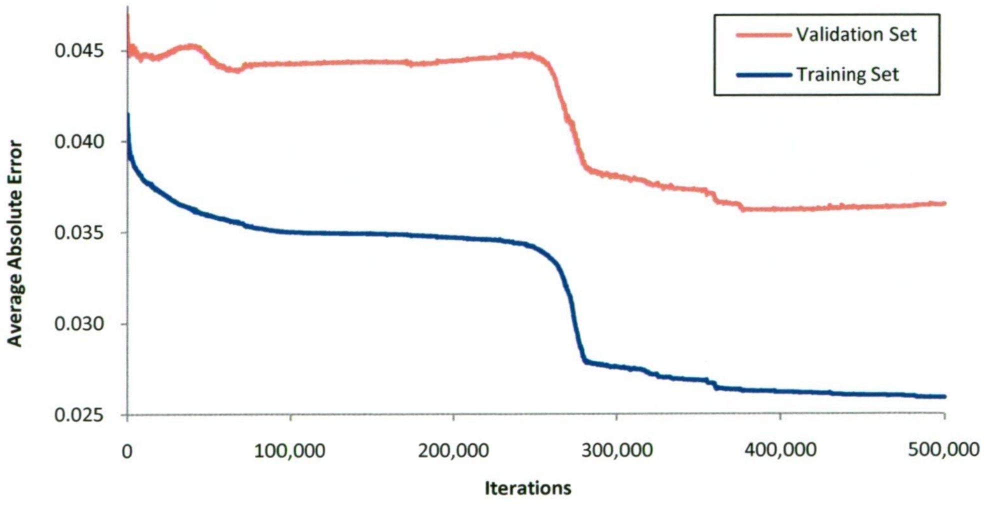


Figure 71. Extended neural network training

As a result of examples such as this, a policy of extended "training to exhaustion" was adopted for VESPA metamodels, using the quasi-Newton algorithm for at least 250,000 iterations. In many cases this showed no benefit, with the training error converging on a minimum after several thousand iterations. On other occasions, extended training proved to be advantageous, with significant reductions in error values occurring late in the training process.

OUTLIERS

Initial neural network training of each data set focussed on outlier identification. Outliers were common, due to both errors in SPLASH input parameters, as well as the occurrence of non-converged points in SPLASH. Although NeuroIntelligence has automatic outlier identification functions, it became apparent that this often missed true outliers, while incorrectly classifying known good points.

It was found that the fastest way to filter outliers was to perform a preliminary neural network fit using the Levenberg-Marquardt method. This preliminary fit could be performed rapidly due to the exceptional performance of the LM method.

Initial fitting was followed by examination of the correlation plot for points that were predicted poorly. An example of such a plot is illustrated in Figure 72. This shows three small groups of outliers, each of which could be identified and removed from the training data set.

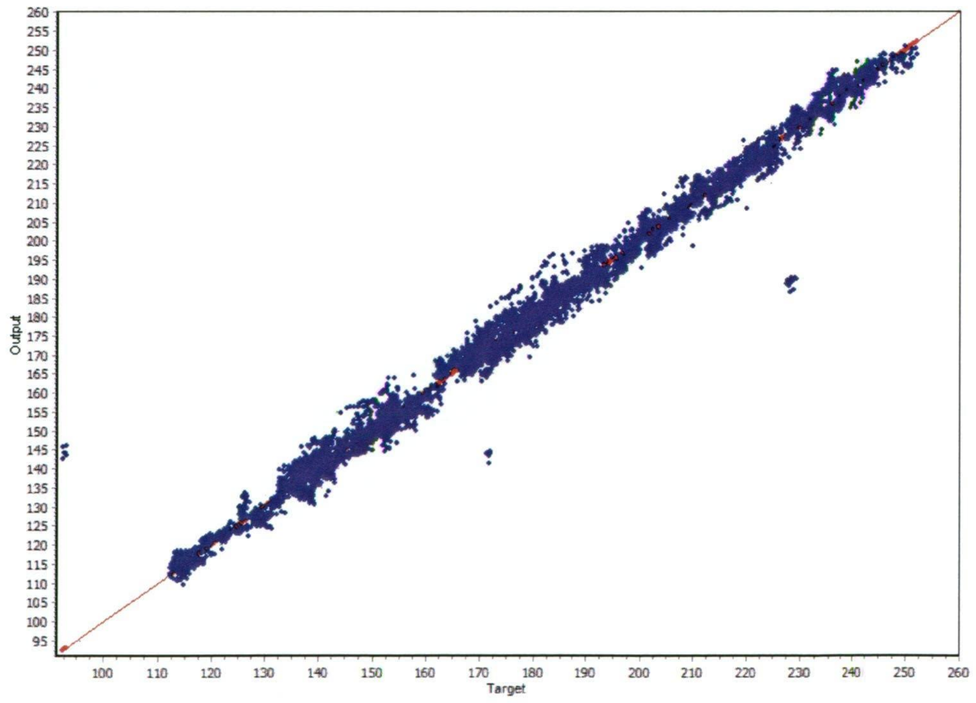


Figure 72. Correlation plot of neural net prediction versus actual, showing outliers

Outlier identification and removal is an iterative process. As outliers are removed, the quality of each retrained neural network improves, allowing more outliers to be identified.

Some outliers may be clearly associated with specific errors in input to SPLASH. In these cases, the errors in the input values may be corrected and SPLASH re-run. For example, in the case shown in Figure 72, the three groups of obvious outliers were due to the incorrect specification of trim-tab angle in the model used for SPLASH, resulting in anomalous lift and drag results. These points were successfully recalculated using the correct trim-tab values and the correct results substituted into the training data set.

METAMODEL TRAINING PROCESS

Once outlier removal had taken place, the neural net training process within NeuroIntelligence consisted of four steps:

- **Architecture search.** NeuroIntelligence is capable of performing an automated search for the network architecture that results in the best performance. Upper and lower limits are placed on the number of neurons to be incorporated in the hidden layer of the network. The system will step through from least to most, repeating neural network training a set number of times at each level until complete, at which point the architecture with the best measure of merit is selected.

  This measure of merit may be training set error (not advisable due to the bias it introduces to the architecture design), test set error (not advisable as it makes the test set no longer truly independent), validation set error (best, but introduces some bias), or a weighted sum of two or more of the three sets.

  The adoption of the architecture with the smallest error is not recommended as it may result in a network with too many neurons in the hidden layer and a high risk of over-fitting. In this regard, Akaike's criterion (Akaike 1974), which weighs error values against network complexity in order to recommend a compromise solution, was found to be a reliable predictor of the optimal network architecture.
- **Initial fitting** was performed using the Levenberg-Marquardt algorithm with between 5 and 10 retrains.
- **Extended training** was performed using the limited memory quasi-Newton method for up to 250,000 iterations.
- **Export** of trained networks to an XML file for input to the VPP.

This process was repeated for each of the metamodels required by PAP, resulting in the creation of six XML files for each family of designs derived from a particular parent model.

8.4 NEURAL NETWORK VALIDATION

Prior to fitting neural network metamodels, SPLASH output data was partitioned into three separate groups. Of the data for the 24 hulls in the systematic series, 67% were allocated to a training set and 33% to a validation set. An additional six hulls, whose parameters were selected randomly, were analysed for use solely as an independent test set. This allowed the prediction capability of the neural networks to be tested to a high degree of confidence.

After extensive neural-network architecture search, followed by extensive training of several alternative configurations, selection of the best neural network was performed based on minimum test-set error.

Table 10 lists the average absolute error (AAE) and average relative error (ARE) values for the neural network metamodels used for Parent Model 3, the final design evaluated by VESPA. Analysis of training, validation and test sets showed that ARE was less than 1% for all values.

Table 10. Neural Network Prediction Errors – Parent Model 3

	Drag Upright (N)	Drag Heeled (N)	Lift (N)	Heeling Moment (N.m)	L_{WL} (m)	Wetted Surface (m^2)
Average Absolute Error	2.676	9.144	79.583	82.616	0.020	0.040
Average Relative Error	0.29%	0.75%	0.77%	0.82%	0.09%	0.07%

Once optimisation runs had been performed and a likely area of the parameter space isolated as a location for an optimal design, a further six to eight designs were created and analysed in the vicinity of this design. In some cases, these additional designs were used to extend the range of a design variable where the optimal design had been found close to or at the upper or lower limit of the existing range for that variable.

Once these additional designs were analysed using SPLASH, their hydrodynamic data were incorporated into the original dataset. Neural networks were then retrained, with a resulting improvement in the prediction accuracy in the area of the optimal design.

8.5 NEURAL NETWORK DLL VALIDATION

The XML_NN module was tested extensively to ensure that it gave identical results to the same neural network running within NeuroIntelligence. Testing verified that results were identical in all cases, within the limits of double-precision, floating-point rounding error.

8.6 GA VALIDATION

The GA was validated using a variety of simple problems to ensure correct functioning. The GA was also tested within the VESPA system using synthetic VPP performance data to test the ability of VESPA to correctly find a known optimal design. The synthetic VPP data was defined as:

$$V_{MG} = \begin{cases} 1.5 \times V_T \times \sin(\beta_{TW}) \times \cos(\beta_{TW}) & if \;\; 0 \leq \beta_{TW} \leq 90 \\ 2.0 \times V_T \times \sin(180 - \beta_{TW}) \times \cos(\beta_{TW}) & if \;\; 90 < \beta_{TW} \leq 180 \end{cases}$$

$$\begin{aligned} V_{MG} = V_{MG} &\times (1.0 - |3.2 - B_{WL}| * 0.1) \times (1.0 - |0.82 - C_M| \\ &\times (1.0 - |5.0 - \text{Flare}| * 0.05) \times (1.0 - |0.58 - C_P| \\ &\times (1.0 - |0.54 - L_{CB}|) \end{aligned} \tag{8.1}$$

This function resulted in simple polar curves having best upwind V_{MG} at 45°and best downwind V_{MG} at 135°. Optimal performance occurred at design parameter values of B_{WL} = 3.2m, C_M = 0.82, flare = 5°, C_P = 0.58 and L_{CB} = 0.54.

Test cases allowed either five design variables (B_{WL}, C_M, C_P, L_{CB} and Flare) or three design variables (B_{WL}, C_M, and flare). Population size for this series of tests was set at 25.

The performance of the GA for this test function is shown in Table 11. This shows that a close approximation to the correct optimum of the test function was reliably found within 20 generations of the GA.

Table 11. Genetic Algorithm Performance for Synthetic VPP Data

No. of Free Variables	No. of Generations	B_{WL} (m)	C_M	C_P	L_{CB}	Flare (deg.)	ARE
5	5	3.247	0.8153	0.5825	0.5397	5.0329	0.636%
	10	3.216	0.8199	0.5821	0.5380	5.0217	0.305%
	15	3.226	0.8208	0.5805	0.5398	5.0062	0.235%
	20	3.199	0.8199	0.5802	0.5392	5.0023	0.053%
	25	3.204	0.8200	0.5800	0.5392	4.9984	0.063%
3	5	3.191	0.8235	n/a	n/a	4.8938	0.946%
	10	3.204	0.8219	n/a	n/a	5.0247	0.283%
	15	3.207	0.8190	n/a	n/a	5.0051	0.148%
	20	3.200	0.8200	n/a	n/a	5.0007	0.005%
	25	3.201	0.8200	n/a	n/a	4.9993	0.015%

Average Relative Error (ARE) values for the GA optimisation, using the test function with three and five design variables, are graphed in Figure 73.

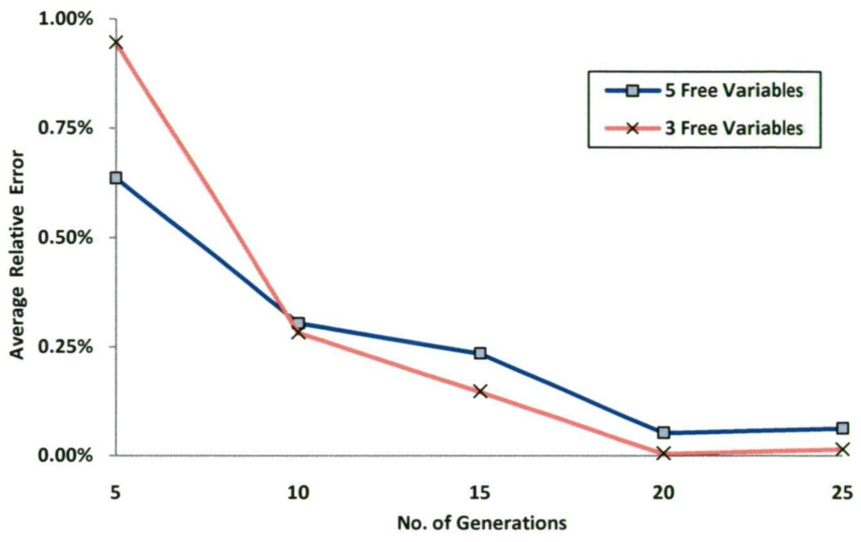


Figure 73. Error values for different numbers of GA generations

Testing was also performed to determine the best population size for use by the GA. If population size is increased while generation count is kept constant, a bias will be created in favour of the larger populations, as the total number of evaluations performed will be greater. To avoid such bias, this test maintained a constant number of objective function evaluations, choosing combinations of population size and number of generations that resulted in 192 objective function evaluations in all cases. Results of the study are shown in Table 12 and Figure 74.

This testing showed that population sizes of between 12 and 32 gave similar results, a figure that broadly agrees with the research cited in Section 6.4.5, which found population sizes between 20 and 30 to be optimal. Smaller populations appear to lack the diversity required for recombination operators to be effective, while larger populations that use the same number of objective function evaluations have insufficient generations for appreciable evolution to occur.

Table 12. Population Size Study Outcomes

Population Size	No. of Generations	B_{WL} (m)	C_M	C_P	L_{CB}	Flare (deg.)	ARE
6	32	3.116	0.8195	0.4749	0.5399	5.0351	4.307%
8	24	3.189	0.8513	0.5709	0.5384	5.0093	1.241%
12	16	3.252	0.8212	0.5801	0.5403	5.0076	0.398%
16	12	3.193	0.8206	0.5820	0.5381	4.9673	0.328%
24	8	3.171	0.8187	0.5802	0.5408	4.9766	0.341%
32	6	3.196	0.8198	0.5815	0.5359	5.0392	0.389%
48	4	3.279	0.8165	0.5799	0.5419	5.0730	0.945%
64	3	3.165	0.8208	0.5794	0.5429	5.1989	1.166%
96	2	3.223	0.8257	0.5876	0.5337	4.7939	1.604%

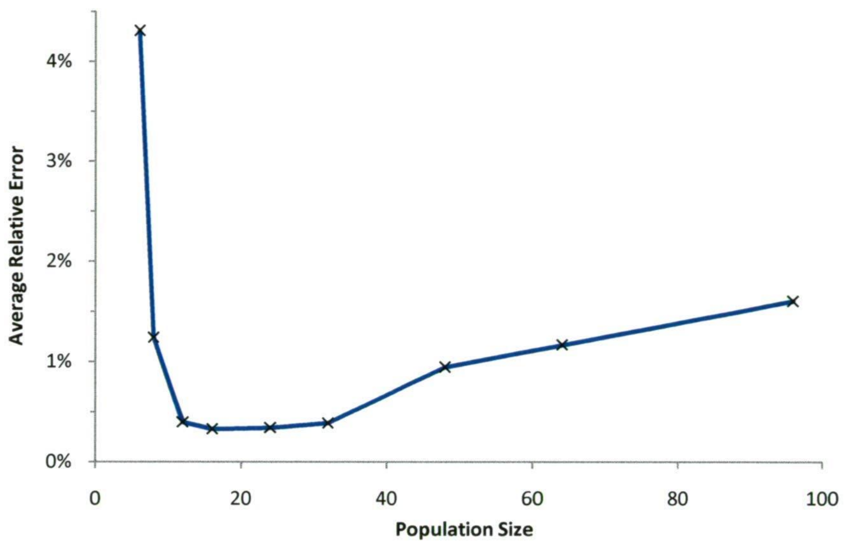


Figure 74. Average Relative Error values for variations in population size

As the test function employed was comparatively simple, it was considered that populations smaller than 20 may be insufficient when more complex solution spaces were encountered. As a result, a population size of 25 was adopted as a conservative compromise between the conflicting needs of genetic diversity and low computational cost.

8.7 VPP VALIDATION

Specific validation of the VPP was not performed, as PAP has been extensively tested in the context of ACC yacht design during the past ten years. PAP has been regularly compared to performance data from full scale testing and has been found to predict ACC yacht performance reliably. PAP was used by the Alinghi team for the design of SUI-64, winner of the 2003 America's Cup.

The principal change made to PAP was the incorporation of the neural network XML file definitions as the source for hydrodynamic lift and drag data. Aerodynamic models and the algorithms used for balancing forces and moments within the VPP were left unchanged. These changes required significant testing, which revealed some convergence problems for PAP when the neural network metamodels were not sufficiently smooth.

This problem was addressed in two ways. PAP's solver was improved to make it less sensitive to noise in the metamodels. More importantly, procedures for training neural network metamodels were amended to avoid over-fitting of the source data. The use of large test and validation sets, the adoption of extended training periods and the selection of the smallest possible number of neurons in the hidden layers were key components of this strategy.

8.8 RMP VALIDATION

The VESPA RMP was coded and tested using standard software development and testing methodologies. These included unit testing at the subroutine level, using a set of test data with software stubs to uncoded subroutines, followed by system level testing using synthetic data.

The VESPA RMP was tested using the same synthetic VPP data previously used to validate the GA optimiser. Statistical analysis of actual race data was used to quantify some of the values used for the RMP, such as trailing boat penalties applied at rounding marks.

RANDOM TIME VARIATION

Section 5.2.4 describes the analysis of real-world test data to determine the random time variation used by the VESPA RMP. This approach carries with it the risk of bias due to the small size of the available sample.

To determine the impact of such a bias on optimisation outcomes, a sensitivity analysis was performed using the race modelling capabilities of VESPA. For this analysis, each race was repeated 100 times with a small random time penalty added to each boat's time for each leg. The SD of this variation was incremented from zero to twelve seconds in three-second steps, with the results shown in Table 13.

Table 13. Change in Optimal B_{WL} with the Incorporation of Random Noise

	Optimal B_{WL} (m)	
Random Variation (s)	**Wind velocity SD 1.0 knots**	**Wind velocity SD 1.5 knots**
0	3.116	3.170
3	2.999	2.997
6	2.997	2.997
9	2.999	n/a
12	2.997	n/a

Results of this analysis showed that the introduction of very small random variations into the tournament model resulted in an immediate change in the best parameter values found by VESPA, but that further increases to the magnitude of the random variation had little additional effect on the results.

This was an encouraging result as it shows that the outcome was not highly sensitive to the amount of random variation provided some variation occurred. This allowed a conservative value for the SD of the variation to be chosen in the knowledge that it was unlikely to have a detrimental effect on the outcome.

8.9 VESPA SYSTEM TESTING

8.9.1 OBJECTIVES AND METHODOLOGY

To evaluate the functioning of the entire VESPA system, a series of validation tests were performed using an actual ACC design. The measure of success for these tests was an evaluation of the recommended optimal designs using SPLASH and PAP, followed by comparison with similar data for the parent model. For VESPA to be successful in optimising the design, significant improvements in performance over the parent model should be evident in these comparisons.

Using a parent model supplied by Alinghi, twenty-five variations were generated using a uniform design sequence within the parameter limits shown in Table 14, to give the hull shapes shown in Figure 75. These limits were chosen based on ranges considered by Alinghi design team members as likely to contain optima.

Table 14. Parameter Variation Limits – Validation Model

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Minimum	2.9	0	0.805	0.555	0.5275
Maximum	3.4	8	0.855	0.575	0.5475

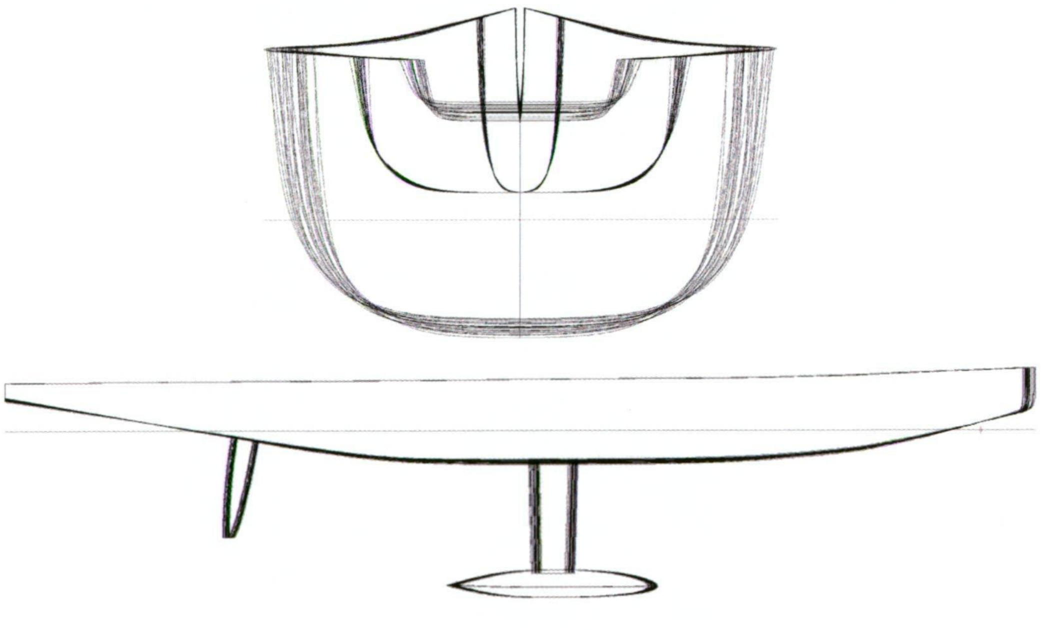


Figure 75. Family of 25 hull design variations

In addition, five hull designs were created using random values of the above parameters within the ranges shown. These additional hulls were created solely as an independent test set for the neural network metamodels.

Appendages for the transformed hulls were identical to those of the parent model, although keel and bulb were moved longitudinally to maintain the alignment of the L_{CG} of the yacht with its L_{CB}. In addition, the longitudinal location of the rudder was moved to maintain a constant distance between its trailing edge and the aft end of each hull's waterline.

These thirty hull designs were exported as IGES NURBS surface files and analysed using the SPLASH potential flow code. Once SPLASH analysis was completed, results for all hulls were collated into a single file and used as input to the NeuroIntelligence neural network software. Of the data for the initial twenty-five hulls, 67% were allocated at random to the training set, with the remaining 33% allocated to the validation set. Data for the five randomly generated hulls were allocated to an independent test set.

Neural network metamodels were trained for each of the following variables –

- lift;
- drag upright – residuary resistance for the zero heel, zero yaw case;
- drag delta – additional drag due to heel and yaw;
- hydrodynamic heeling moment;
- waterline length;
- wetted surface area.

Lift, drag and hydrodynamic heeling moment values were partially linearised by dividing each by $\frac{1}{2}\rho V^2$ to remove the V^2 component. For drag, this gives the drag area f, the area of a flat plate, held normal to the direction of flow with a $C_d = 1$, having equivalent resistance.

Converting the lift, drag and hydrodynamic heeling moment data in this way reduces the degree of non-linearity in the data and allowed more accurate fitting of the neural network metamodels. If this linearisation of the data is not performed, least-squares fitting of the data is dominated by the errors for higher hull velocities, with the fitting of low speed data suffering as a result.

8.9.2 OPTIMISATION RUNS

Three optimisation runs were performed with VESPA using two different wind velocities and wind velocity standard deviations. The purpose of these runs was to investigate the effect of different wind velocity parameters on the hull design. While a correlation between B_{WL} and wind velocity may be a reasonable expectation, the effect of changes to the variance of the wind velocity are not as easily predicted.

Two different mean wind velocities were chosen, along with two different standard deviation values for wind strength, in order to establish trends for the two variables.

Two different wind velocity standard deviation (SD) values are used; one SD for the event, a total of 100 races, and one SD for each race. This is necessary because of the different timescales involved. An event may take place over a period of several weeks, while a race will take place over a period of 2-3 hours. The SD of the wind velocity distribution for each of these two periods will differ significantly.

8.9.3 RESULTS AND VALIDATION

Results of the three optimisation runs are summarised in Table 15.

Table 15. Results Summary – Validation Model

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	B_{WL} (m)	Flare (deg.)	C_M	Number of Generations
Run 1	16	4	1	3.376	0.911	0.812	10
Run 2	11	4	1	3.189	0.076	0.805	15
Run 3	11	1	1	3.023	0.105	0.805	20

For Run 1, VESPA searched for optimum values for B_{WL}, Flare, C_M, L_{CB} and C_P. The GA was run for 10 generations. However, it was clear that in this case the GA did not fully converge, as there was significant variation of parameter values in the population.

B_{WL} for Run 1 was higher than expected. Flare was low, but with significant variation within the population. This was initially interpreted as incomplete convergence on an optimal value. However, later runs also showed anomalous scatter.

Optimisation Run 2 used a mean wind of 11 knots and an event SD of 4 knots. For this and subsequent runs the L_{CB} was fixed at 53.5% and the C_P fixed at 0.57. The purpose of fixing these values was to simplify the search process by reducing the number of dimensions of the parameter space. This allowed the optimal values for other parameters to be found in fewer iterations. This optimisation was run for 15 generations of the GA.

Optimal B_{WL} reduced for this run, as did C_M. Flare approached zero for the most successful boats, although once again, variance for this parameter was high. C_M was at the lower limit of the test matrix range.

Optimisation Run 3 used a mean wind of 11 knots and an event wind strength SD of 1 knot. Although this is not a realistic value for an event wind strength SD, it was used to establish the trend in B_{WL} and C_M parameters as SD is reduced.

This run was performed with fixed L_{CB} and C_P, and was run for 20 generations of the GA to encourage convergence. Once again, the optimal B_{WL} reduced and the C_M was at the lower limit. Variance for B_{WL} and flare was small, indicating good convergence. However, it is likely that lower C_M values would have resulted had VESPA been free to use them.

The trend towards lower B_{WL} at lower wind velocities was not unexpected, as lower viscous and residuary resistance resulting from lower beam, even at the cost of reduced stability, is a reasonable trade-off for light weather. The unexpected outcome of these initial test runs was the low C_M values predicted. Based on the previous experience of Alinghi's design team, this appeared to be an anomalous result.

On the basis of the trends seen in the three optimisation runs, and in order to explore the low B_{WL}, low C_M parameter range, four variations of the parent hull were created using two B_{WL} values and two C_M values, as shown in Table 16.

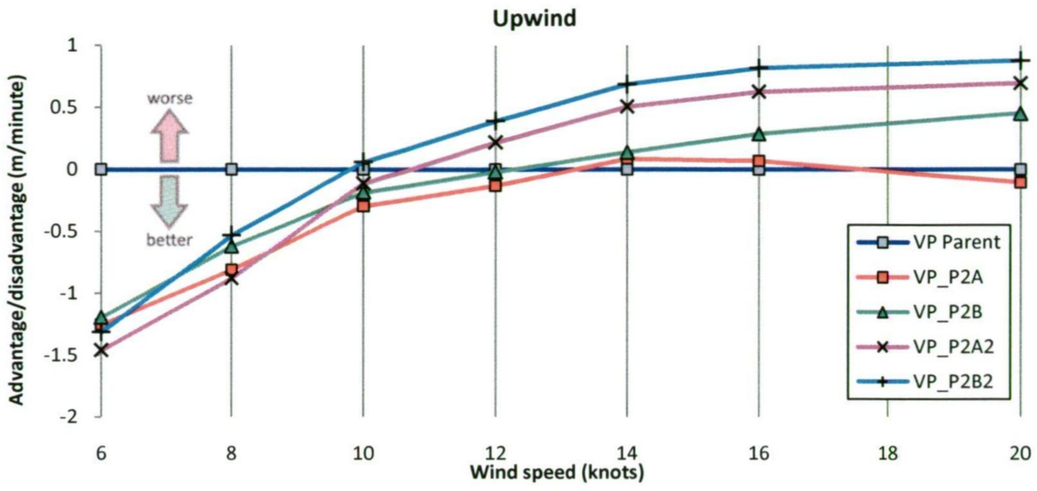
Table 16. Test Design Parameters – Validation Model

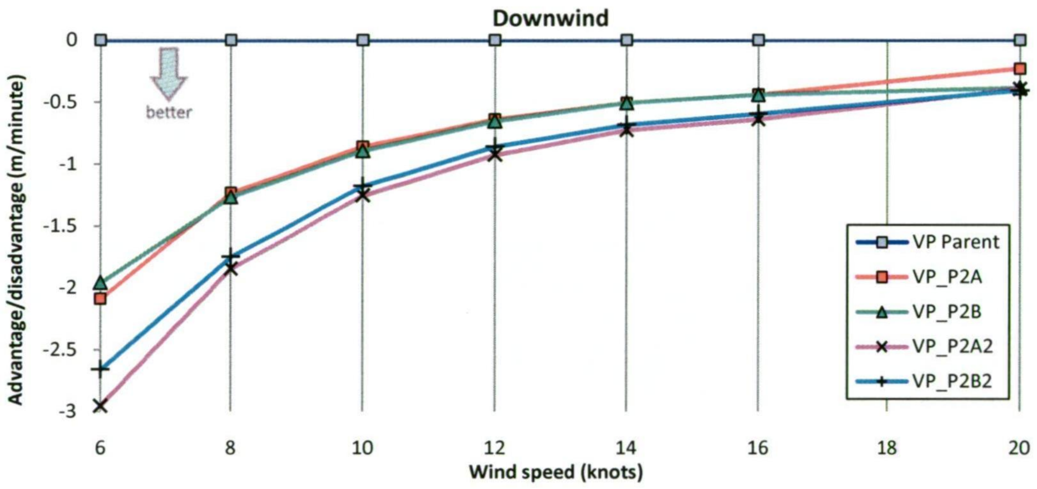
	B_{WL} (m)	C_M	Flare (deg.)
VP_P2A	3.075	0.815	0
VP_P2B	3.075	0.805	0
VP_P2A2	2.975	0.815	0
VP_P2B2	2.975	0.805	0

The four test designs were analysed directly using SPLASH and their performances compared with their parent model, using the performance comparison charts shown in Figure 76.

These charts indicate relative performance of a yacht or yachts against a baseline boat, whose performance is shown as a horizontal line along the abscissa. A yacht having better performance than the baseline boat for a particular wind velocity, measured in metres per minute, will have a value plotted below the abscissa. Conversely, performance inferior to the baseline boat will show as points plotted above the abscissa.

The four hulls tested showed good performance downwind compared with the parent design. Upwind performance, although superior in less than 10 knots of wind, was generally worse than the parent design above 12 knots of wind. Of the four designs, the two narrow boats were fastest downwind and slowest upwind. For equivalent beam, the boats with the higher C_M value were superior upwind and equal to, or better, downwind. The boat with possibly the best overall performance, VP_P2A, had the highest B_{WL} and C_M values.





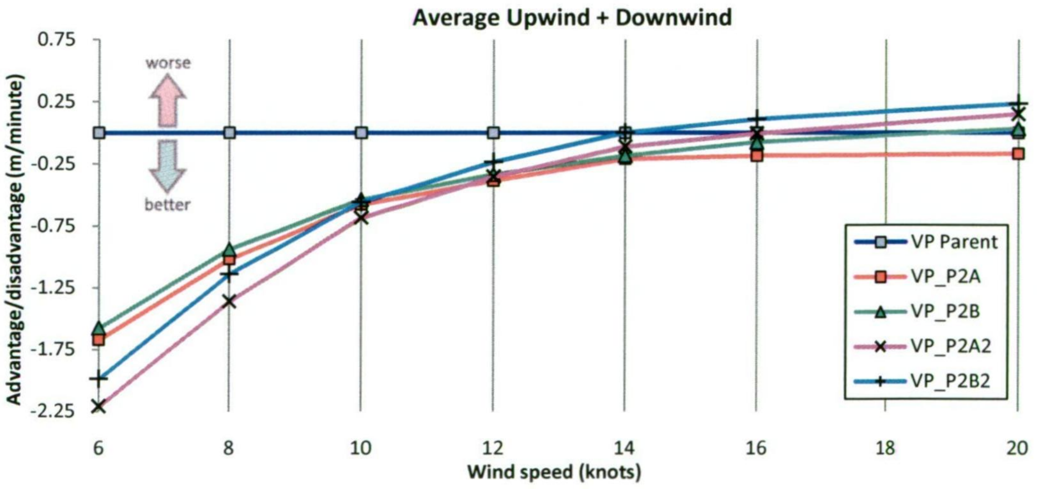


Figure 76. Performance comparison plots of test models versus validation model

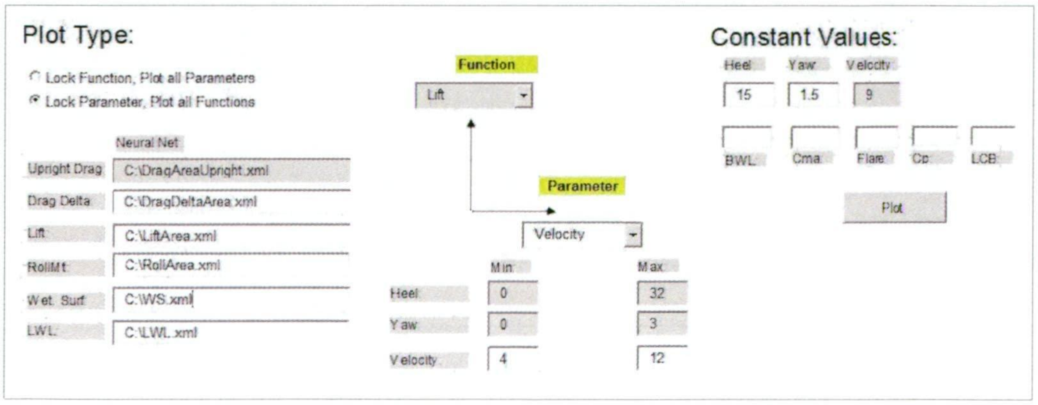
Although these results demonstrated that the four test designs improved on the performance of the parent model, they did not confirm the trend to low C_M suggested by VESPA. It was clear that VESPA had failed to correctly determine the optimal set of values for the design variables under examination.

The failure of VESPA to perform as expected was a cause for concern and significant time was spent determining the source of the problem. The initial focus of these investigations was the numerical stability of PAP, the VPP used by VESPA.

PAP was originally an interactive program that calculated a small number of sailing performance values, which were then used to fit polar curves. When such a program is used as a "black box" solver for an optimisation loop, reliable operation is essential. Instead of a small number of data points being generated, a very large number of cases, potentially in the millions, must be accurately calculated. Numerical instability may result in a small percentage of cases failing, resulting in erratic behaviour by the optimisation procedure, or convergence to a false optimum.

Analysis of PAP's error logs revealed that for a small percentage of cases, PAP failed to converge, resulting in the calculation of an incorrect hull velocity. Although measures were put in place to trap errors and assist PAP's solver to converge on a correct answer, convergence problems remained. After further investigation, it became apparent that these were caused by small regions of the hydrodynamic data that were poorly fitted by the neural network metamodels.

Examination of the neural network metamodels was performed by comparing metamodel predictions for the test designs with actual SPLASH outputs. These comparisons showed significant discrepancies between predicted and actual results for drag at high heel angles. To assist in the visualisation of these errors an Excel spreadsheet was created that allowed slices to be defined through the neural network hypersurfaces for any parameter combination (Figure 77). These slices were displayed as charts as shown in Figure 78, which compares neural network predictions with SPLASH results for 15 degrees of heel and 1.5 degrees of yaw.



Plot Type:

- ○ Lock Function, Plot all Parameters
- ● Lock Parameter, Plot all Functions

	Neural Net
Upright Drag	C:\DragAreaUpright.xml
Drag Delta	C:\DragDeltaArea.xml
Lift	C:\LiftArea.xml
RollMt	C:\RollArea.xml
Wet. Surf	C:\WS.xml
LWL	C:\LWL.xml

Function: Lift

Parameter: Velocity

	Min	Max
Heel	0	32
Yaw	0	3
Velocity	4	12

Constant Values:

Heel	Yaw	Velocity
15	1.5	9

BWL	Cma	Flare	Cp	LCB

Figure 77. Validation spreadsheet controls

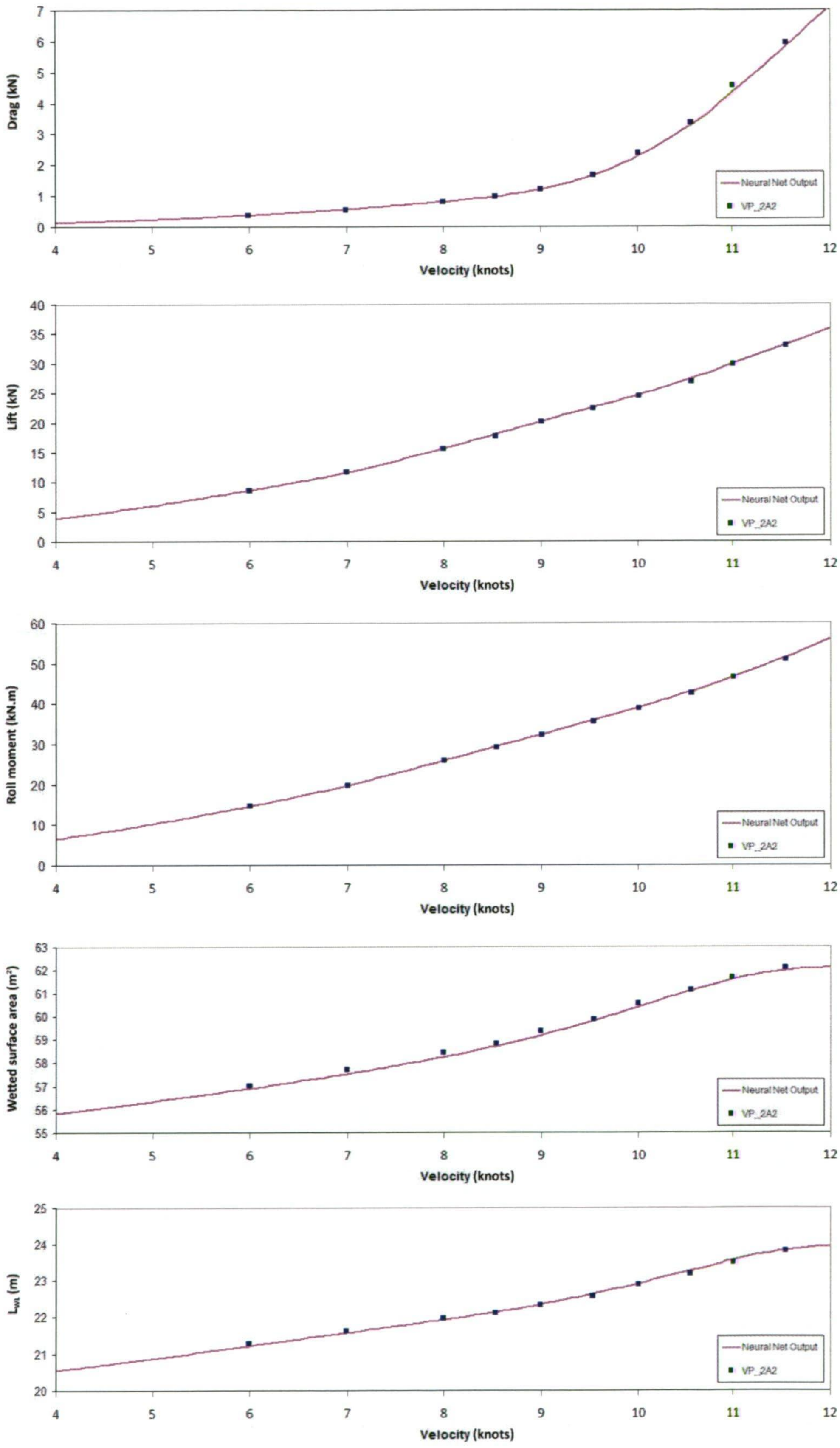


Figure 78. Metamodel validation graphs

Examination of these graphs revealed that an error existed in the metamodel for heeled drag, which resulted in resistance decreasing significantly at high heel angles (Figure 79). This had the effect of rewarding yachts with lower stability, for example those with low B_{WL} and C_M values, resulting in these hull forms dominating the optimisation process.

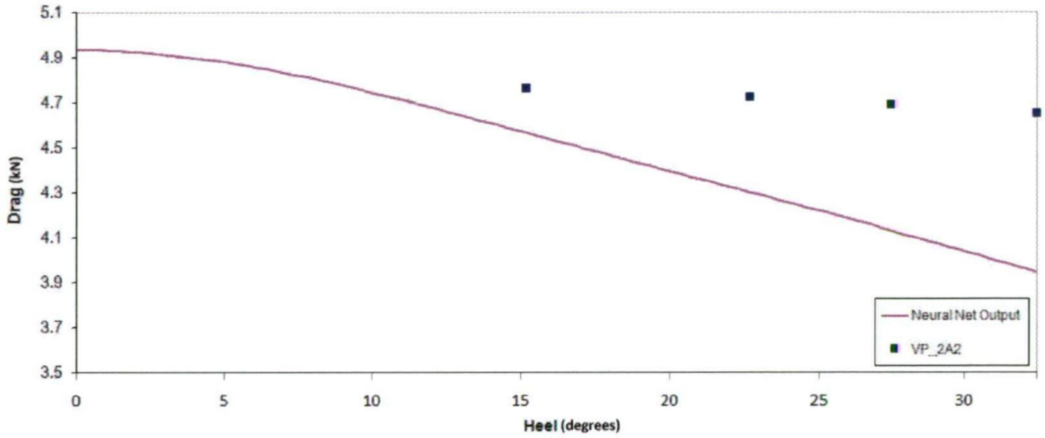


Figure 79. Drag errors for 11 knots velocity, 1.5 degrees of yaw

8.9.4 VALIDATION MODEL CONCLUSIONS

No firm conclusions regarding design optima could be drawn from VESPA testing of the validation model, due to metamodel fitting errors. As a result of these difficulties, a significant effort was put into determining the most effective training process for the neural networks metamodels. Systematic testing of different training algorithms, network architectures and training periods was performed, together with the development of procedures for effectively testing and validating the neural network metamodels.

The SPLASH analysis performed on the four test designs did provide support for the following observations:

- Reduced B_{WL} improved downwind performance in all wind velocities, with the greatest gains occurring in light winds.
- Reduced B_{WL} resulted in improved upwind performance in light winds, but was responsible for poorer upwind performance in greater than 12 knots of wind.
- Lower C_M values did not appear beneficial in any wind velocity, upwind or downwind.

9 · RESULTS & DISCUSSION

Optimisation runs using VESPA took place between June and August 2006, using three different parent hull models supplied by the Alinghi design team.

During this period, Alinghi also continued with its conventional design development process. This consisted of hull shape experimentation by lead yacht designer Rolf Vrolijk, CFD analysis using SPLASH and other CFD codes, simulation of sailing performance using VPP software, followed by one-third scale tank testing of the most promising designs.

Over the period during which VESPA was first run and progressively refined, the Alinghi design team independently explored many promising design directions. Some of these proved beneficial and were incorporated in the final design of SUI-100, the yacht that went on to win the 2007 America's Cup, while others proved to be dead ends and were rejected.

Design avenues that were not successful had often shown benefits during CFD analysis, which were subsequently not confirmed by tank testing. This is one of the realities of modern yacht design and analysis; the numerical tools used are imperfect and can easily result in false leads being followed, unless results are rigorously validated.

As a result of this ongoing design development process, the parent model used by VESPA for optimisation changed several times. In some cases, the parent models used were later found to be retrograde steps; at other times the parent, when tested in the towing tank, was shown to be the current best of breed.

As the development process continued and the time remaining before delivery of final hull lines for SUI-100 reduced, the time available for each cycle of VESPA testing shortened. While parent model 1 was used for VESPA optimisation work for two months, the time available for VESPA analysis of parent model 3 was less than 10 days. During this period, the process for generating the systematic series, performing SPLASH analysis and creating fully trained neural networks became more streamlined and automated, with significantly less manual intervention required.

As testing progressed and refinements such as trailing boat penalties were added to the VESPA RMP, it became apparent that the optimisation problem was significantly more complex than first thought, with implications for both the design of the optimisation algorithm as well as for the interpretation and application of its results. These issues and their implications are discussed in Section 9.1.

9.1 CONSEQUENCES OF TRAILING BOAT PENALTIES

The VESPA RMP models interactions between yachts at turning marks via the application of penalty functions applied to the trailing boat, as described in Section 5.2.4. These penalty functions were still being developed at the time testing of parent model 1 commenced, and consequently, no trailing boat penalties were applied for the initial runs of VESPA using parent model 1. For later runs using parent model 1, trailing boat penalties were utilised without the port tack penalty, which was first implemented for use with parent model 2.

As penalties are applied at windward marks but not at leeward marks, their effect is to favour boats that perform well upwind, at the expense of boats whose performance profile is biased towards downwind work.

When no trailing boat penalties are used, the VESPA solution space often appears bimodal. Yachts whose performance is biased to upwind sailing in preference to downwind sailing may have similar elapsed times for an upwind/downwind racecourse as yachts whose performance is biased towards downwind performance.

Once trailing boat penalties are applied this bimodality disappears, as yachts that are fast downwind are unable to recover their upwind losses. Unfortunately, the removal of this bimodality does not imply a simpler solution, with the effect of trailing boat penalties on the ideal design parameters adding significant complexity to the problem.

During testing with trailing boat penalties, it became apparent that VESPA could produce different results for two similar runs. Optimal hulls produced would be similar in most regards, but could vary in B_{WL} over a range of approximately 150-200mm. It was clear that the optimum was not stationary; rather, it shifted based on the makeup and diversity of the population at the time. If the population was not diverse (i.e. design variables for most of the boats were similar), the optimum could move around, whereas when the population was made up of a variety of different designs, the optimum would tend to stabilise at high B_{WL} values.

Testing was performed to investigate this problem. In the first set of tests, VESPA was run for 25 generations, with a population size of 20, using parent model 1. Results are plotted in Figure 80.

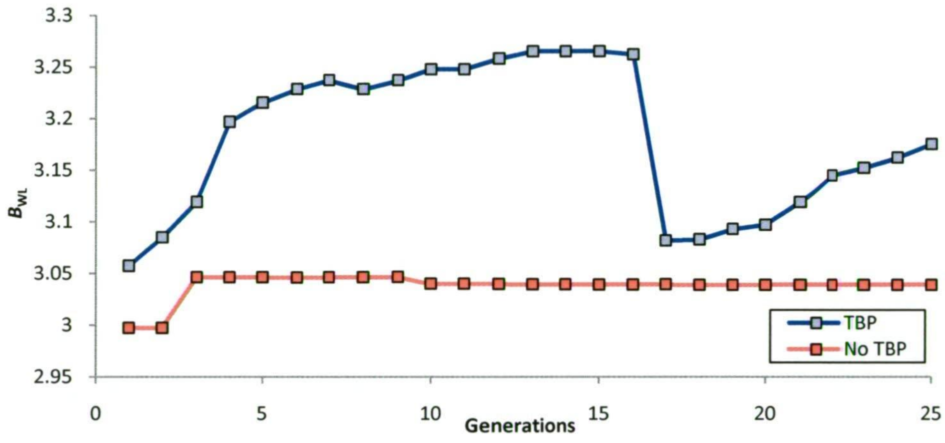


Figure 80. Investigation into the effects of Trailing Boat Penalties (TBP)

Without trailing boat penalties, the optimal B_{WL} value stabilizes after only a few generations at approximately 3.05m. When trailing boat penalties are introduced, the B_{WL} increases to more than 3.25m and appear to stabilise, before dropping suddenly to less than 3.1m. This instability in the solution did not always occur, and on many occasions, the design with the highest fitness simply stabilised at a high B_{WL} value. However, with trailing boat penalties in use VESPA could not be relied on to give reproducible results.

NASH EQUILIBRIA AND MIXED STRATEGY GAMES

To explain this behaviour, it is necessary to introduce the game-theory concept of a pure-strategy Nash equilibrium. Nash (1950) proposed that in a two-player game, a pure strategy existed if one player could adopt a strategy that did not need to be changed, regardless of how an opponent's strategy changed. In yacht design, this is equivalent to finding the design parameters for the fastest boat for an upwind leg for a fixed wind velocity. Regardless of how your opponent changes their design, it will not be faster than yours. If a pure-strategy Nash equilibrium exists, it is possible to locate a single optimum that cannot be improved on for a specific set of conditions.

The introduction of a penalty for the trailing boat at the first mark, say 1.5-boat lengths, changes the situation. As an example, assume a boat **A**, which has the best aggregate of upwind and downwind performance when no trailing boat penalty exists. A boat that has a slightly greater B_{WL} (boat **B**) may be 2 boat lengths faster upwind and 3 boat lengths slower downwind, and therefore slower by 1 length for an upwind/downwind course. However, if boat **A** incurs a 1.5 length penalty at the first mark, boat **B** will win the race by ½ a length. If both competitors in a match race choose boat **B** for their design, neither will gain an advantage.

If a third boat, **C**, is slightly wider again, to the extent that it is 2 lengths faster than **B** upwind and 3 lengths slower downwind, **B** will incur the 1.5 length penalty at the first mark. In this case, boat **C** will win by ½ a length.

B beats **A** and **C** beats **B**. Is **C** therefore the best design to choose? Unfortunately, when **C** races **A**, it loses. Boat **C** is 4 lengths faster than A upwind and picks up the 1.5 length upwind bonus, giving it a 5.5 length upwind advantage. However, it is 6 lengths slower downwind, so it loses overall by half a boat length. In this case, if each competitor could choose only from the designs of boats **A**, **B** or **C**, there is no strategy that guarantees success. In a round robin tournament between the three, each boat would come away with one win and one loss.

This type of problem is considered to have a mixed-strategy equilibrium – there is no single strategy that guarantees success, and there are multiple strategies that have an equal chance of success. This is illustrated by the following diagrams, which use idealised performance values for hypothetical boats. Figure 81 shows the change in upwind and downwind performance as B_{WL} is varied. As B_{WL} increases, the upwind performance improves while the downwind performance deteriorates. The fastest boat overall is the one with best average of upwind and downwind performance, shown by the peak in the upwind + downwind average curve.

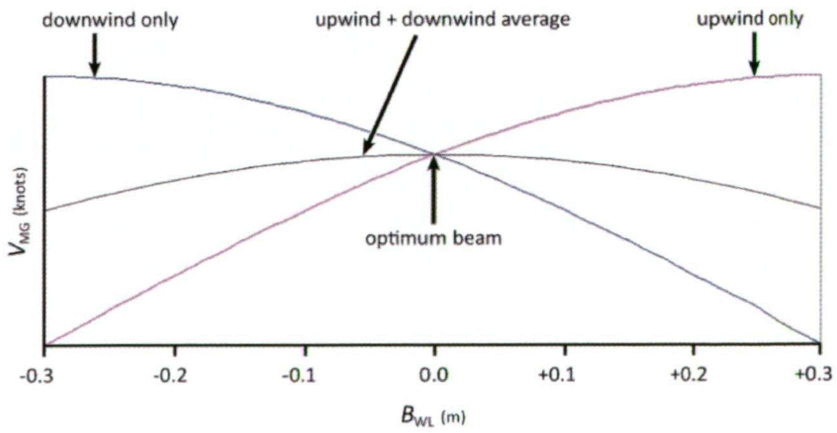


Figure 81. Optimal beam for a simple yacht performance model

The situation changes when a trailing boat penalty is introduced. In this case, the wider boat in any match will have the performance shown by the upwind + downwind average curve. However, the narrower boat of the pair will be penalized on the upwind leg and its aggregate performance will be that of the curve running parallel to and below the unpenalised performance curve. In this situation, a mixed-strategy outcome occurs, as illustrated by Figure 82.

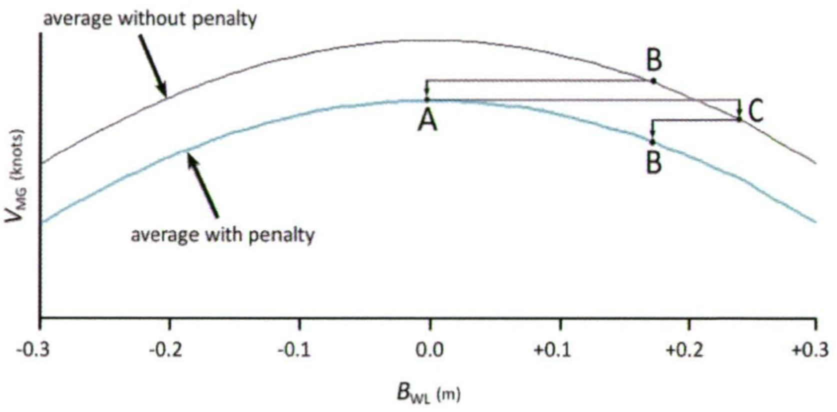


Figure 82. Effect of trailing boat penalties on race performance

In the first match, **B** versus **A**, boat **A** incurs a trailing boat penalty and **B** wins. In the second match, **B** incurs the penalty and **C** wins. In the third match **C** is the faster boat upwind, but **A**'s aggregate performance, even with the penalty included, is superior to that of **C**. No strategy is superior, and all boats will score the same in a round robin tournament.

Several mixed strategy games are similar to this problem. If we rename boat **A** "Rock", boat **B** "Paper" and boat **C** "Scissors" the similarities become obvious.

- Paper beats Rock.
- Scissors beat Paper.
- Rock beats Scissors.

In a game of rock, paper, scissors, no single strategy is superior because the hands are revealed simultaneously. However, if a short delay occurs between the revealing of the hands, the later player will always be able to choose a winning hand. In the America's Cup, the situation is similar when two teams are choosing boats for the final round of the event. If both boats must be nominated simultaneously, neither team can guarantee a winning strategy. If one team is forced to nominate their design in advance of the other, and the later team is able to determine the design parameters and performance profile of the first team's boat, a boat that is slightly superior upwind performance should be chosen.

Note that it does not necessarily pay to be significantly faster upwind – a boat that was halfway between boat **A** and boat **B**, so that it was only 1 boat length faster upwind and 1.5 boat lengths slower downwind, would win the race by 1 length, whereas boat **A** beat boat **B** by only ½ a length. Therefore, the strategy for an America's Cup defender would appear to be to choose a boat that is similar to the challenger's boat, but slightly biased towards upwind work.

For the challenger the situation is slightly different. Rather than having to compete against one boat, challengers are each required to compete against ten other teams for the right to challenge for the cup. This makes the problem of boat selection a different one to the problem faced by the defender. Each challenger needs to select a boat that will have the highest win/loss ratio when matched against a diverse range of competing boats. In this case, there is an optimum, although it is dependent on the makeup of the fleet.

It is possible to perform a simple Monte Carlo simulation to show how the probability of winning varies with beam against a fleet of opponents in a round robin format. Figure 83 shows the outcome of an analysis using a simplified yacht performance model, in which 50 boats of different beams were matched against one another. It can be seen that the curve of average win/loss ratio peaks to the right of the B_{WL} of the fastest boat, but drops dramatically once past the peak. It is clear that the effect of applying a trailing boat penalty at the windward mark is to bias the optimal design in the direction of greater upwind performance via increased B_{WL}.

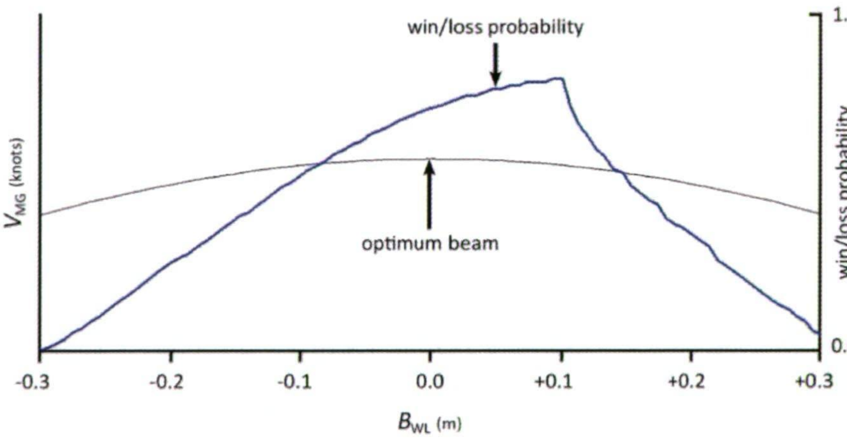


Figure 83. Win/loss ratio for a range of B_{WL} values, utilising trailing boat penalties

The above examples are for a simplified and idealized performance profile for one wind velocity. As the VESPA RMP uses a wind velocity distribution rather than a single wind velocity, it was expected that the clear peak and sharp decline shown in the curve above would be blurred substantially, but that a shift of the optimal beam to the right and a sharp decline once past the optimum should be apparent.

To test this hypothesis, a tournament model using the VESPA RMP was run for a test design whose beam was varied over a range of 0.35 m. One tournament was performed with trailing boat penalties turned on and a second with trailing boat penalties turned off. The results are shown in Figure 84.

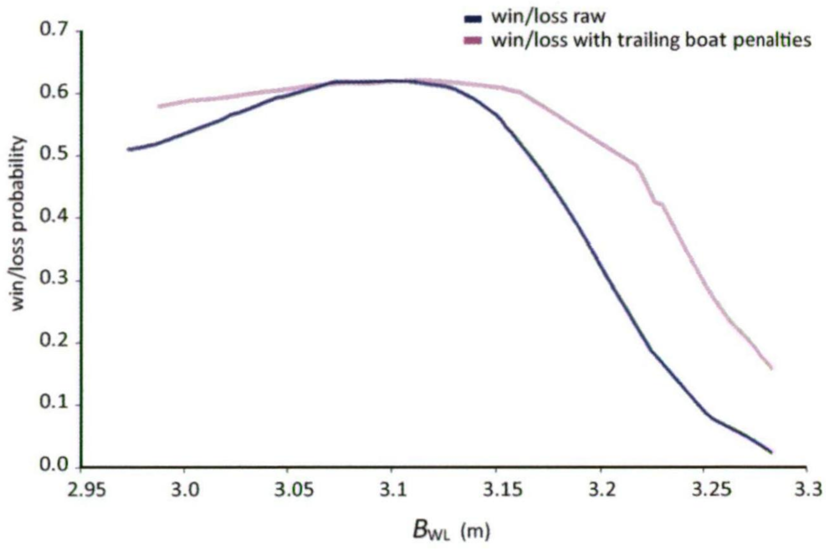


Figure 84. Effect of trailing boat penalties

Without trailing boat penalties, the curve of win/loss ratio is almost flat from about 3.070m to 3.110m, but declines quickly at higher beams. With trailing boat penalties included, the range of optimal beam is broader and is shifted to the right. It also starts to decline rapidly at beams above 3.160m. This area of rapid decline is where the rock, paper, scissors effect applies, and sets the limit on how wide a boat can be made before it becomes rapidly uncompetitive.

The introduction of trailing boat penalties resulted in the optimal solution changing from having one or two stationary optima, to one defined by a limit cycle with no stationary optimum. This required that the GA be modified to limit any cycling of the solution to the minimum range, as it was possible for a co-evolutionary arms race to result in design variables such as B_{WL} drifting away from realistic values.

9.1.1 GENETIC ALGORITHM MODIFICATION

One effect of the use of trailing boat penalties was the occurrence of co-evolutionary arms races in the population, resulting in a solution that increased monotonically in B_{WL}, or cycled over a range of B_{WL} values, rather than settling on a single equilibrium solution. In order to prevent this occurring, two related measures were implemented within VESPA, both based on the "hall-of-fame" approach of Rosin and Belew (1996).

In addition to using simple elitism, which retains the best individual from the previous generation, a modified elitism method was implemented within VESPA. Using this method, the individual retained from the previous generation is the one that had the fastest average time when sailing all of the courses alone. Although this individual may not be equivalent to the boat that wins the most races, it will be of sufficient quality to provide an objective performance benchmark for other members of the population, and may prevent cycling or drifting of best fitness values due to disengagement.

Exemplars were also evaluated as a method for avoiding cycling of the solution and allowing the GA to settle on a single optimal design. Exemplars have the advantage that the design parameters for a known or expected competitor can be specified, allowing VESPA to determine the optimal design to use against that competitor.

Test runs were performed using the same parent model and parameters as used for the testing shown in Figure 80, but with modified elitism and a single exemplar. The single exemplar used B_{WL} = 3.05m. Results are shown in Figure 85.

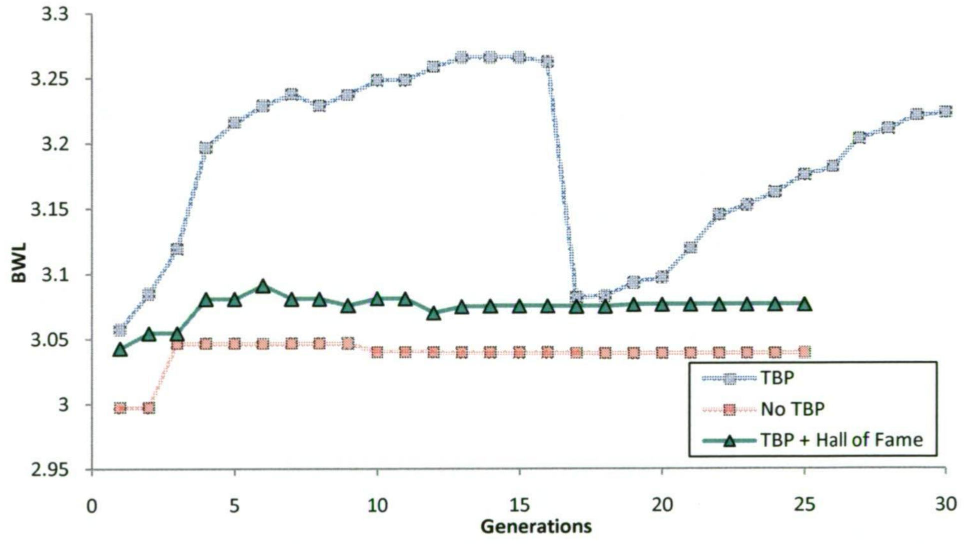


Figure 85. Investigation into the effects of exemplars and/or Hall of Fame

Although these measures may assist VESPA to converge on a single optimal solution, this design is not necessarily an optimal solution against all opponents. The implication is that, rather than a single optimal design existing for a given set of weather conditions, the optimal design must instead be considered within a game-theoretic framework and needs to be selected based on the knowledge or expectation of the design parameters of the opposing yacht or yachts.

9.2 PARENT MODEL 1, MAY 28, 2006

Design optimisation of high quality design candidates was commenced during May 2006. A family of twenty-five models was generated from the parent using the parameter ranges show in Table 17.

Table 17. Parameter Variation Limits – Parent Model 1

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Minimum	2.9	0	0.80	0.575	0.53
Maximum	3.4	8	0.85	0.585	0.54

In addition, seven extra hulls were generated with random parameters within the same ranges. These seven hulls were used solely for testing of the results of the neural network training process.

The thirty-two hulls, along with the original parent design, were analysed by SPLASH and the results collated into a single file. This data file was then used to train neural networks for upright residuary resistance, heeled drag delta, lift, hydrodynamic heeling moment, L_{WL} and wetted surface area.

The results of the training were tested and validated by comparing with the original training and testing data, both in tabular form and using the validation spreadsheet. The results of this validation were excellent, with average relative error values less than 1% in all cases. Fitting of the data was visually checked using the validation spreadsheet, in order to ensure that there were no oscillations or areas of poor fit in the neural network metamodels.

Once neural networks metamodels had been trained, several runs were performed to determine optimal design parameters for different wind velocities and wind velocity variance, shown as runs 1 to 7 in Table 18. C_P and L_{CB} were fixed for these runs to values in the centre of the ranges analysed by SPLASH.

Table 18. Parent Model 1 Results

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	B_{WL} (m)	Flare (deg.)	C_M	Trailing boat penalty (s)	Wally
Run 1	9	4	1.4	3.040	0	0.85	0	no
Run 2	11	4	1.4	3.050	<0.1	0.85	0	no
Run 3	11	4	1.4	3.043	0	0.85	0	no
Run 4	13	4	1.4	3.040	<0.1	0.85	0	no
Run 5	15	4	1.4	3.045	<0.1	0.85	0	no
Run 6	11	8	1.4	3.063	0	0.85	0	no
Run 7	11	8	2	3.029	7.81	0.85	0	no

VESPA runs were performed for mean wind velocities of 9, 11, 13 and 15 knots to determine how the optimal midship section parameters varied. A second run for 11 knots was performed using a different starting population, in order to test the reproducibility of the system. This repeated run achieved virtually identical results to the original.

Surprisingly, there was only a small variation in beam for these changes in wind velocity, with the optimal B_{WL} varying by only 0.010m across the range of mean wind velocities.

For the conditions similar to those expected in Valencia at the time of the 2007 America's Cup (mean wind 11 knots, S.D. 4 knots for the month of June, 1.4 knots S.D. per race) VESPA predicted a boat with a B_{WL} of between 3.040 and 3.050m, zero flare and a C_M of 0.85.

Two runs (numbers 6 and 7) with increased wind velocity variance were performed with unexpected results. The wind velocity SD for the event was increased from 4 to 8 knots and this resulted in a small increase in B_{WL} of between 0.013 – 0.020m. However, when the SD for each race was also increased from 1.4 knots to 2 knots, the beam of the optimal boat reduced significantly to 3.029 m while flare increased to 7.8 degrees, very close to the upper limit tested for this parent model.

This result again raised the possibility of a bimodal solution, with designs having moderate beam and zero flare dominant in steady conditions, while boats having narrower beam and significant amounts of flare showed promising results in more variable conditions.

One possible explanation for this is that a boat that is required to perform in a wide range of wind conditions may use a narrow B_{WL} to reduce drag, giving an advantage in light wind and on downwind legs, while regaining the stability needed for upwind sailing and heavier conditions by the addition of flare to the topsides.

As a result of these anomalous results, two test designs with different beam and flare characteristics were generated for SPLASH analysis and comparison with the original parent model. The design parameters and hull shapes for these comparison models are shown in Table 19 and Figure 86.

Table 19. Test Model Parameters – Parent Model 1

	B_{WL} (m)	C_M	Flare (deg.)
R1	3.050	0.85	0
R2	3.030	0.85	7.8
R1W	3.168	0.85	0

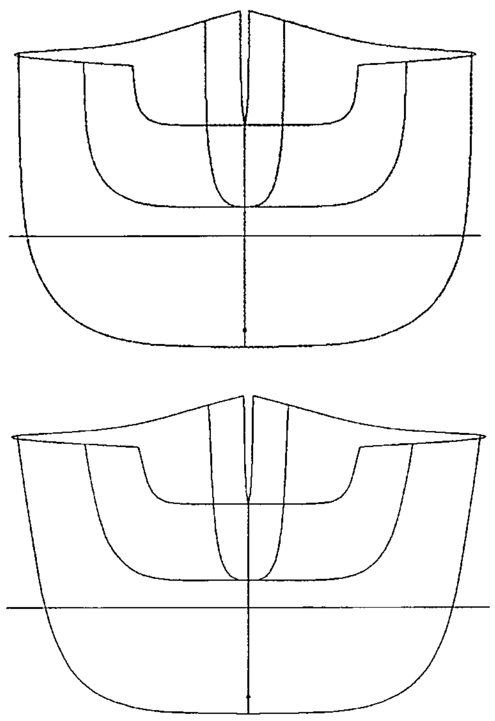
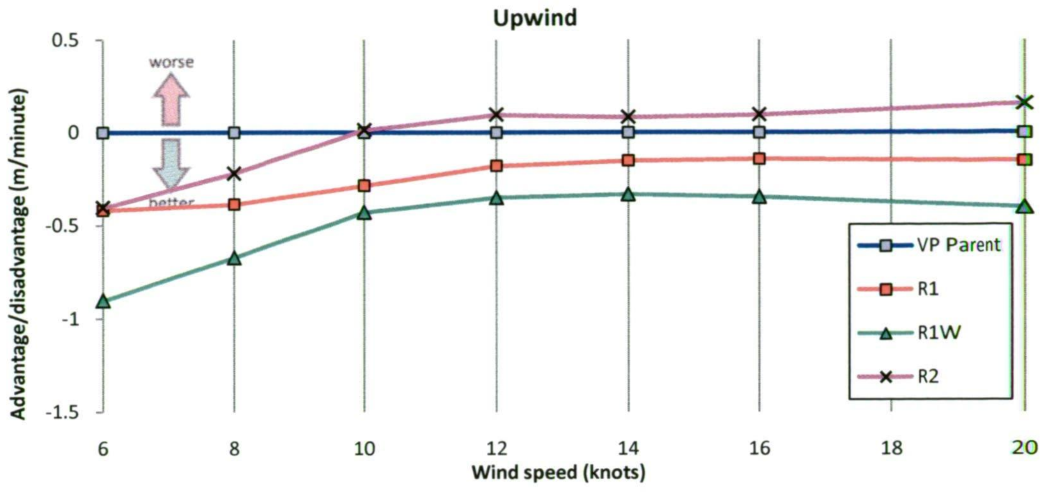


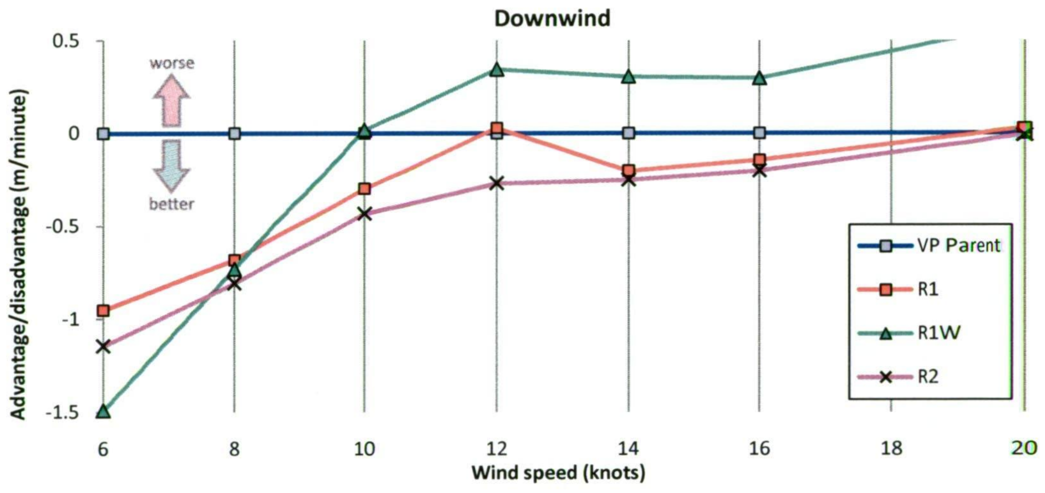
Figure 86. Test hulls R1 (top) and R2 (bottom)

These boats were run through SPLASH and compared to the parent design (Figure 87). Comparisons showed that boat R1 had a speed advantage over the base boat in almost all wind velocities, both upwind and downwind.

This testing also demonstrated that a bimodal solution was possible. Boat R2 had a similar aggregate upwind/downwind performance to R1. However, R2 achieved this by combining comparatively poor upwind performance with superior downwind speed. While the R2 design may have been at a significant disadvantage in actual racing conditions, which are likely to benefit boats having an upwind speed advantage, it was clear from these results that determining a single optimal design for a specific set of conditions was not as simple as first thought.

Subsequent to the positive outcomes shown by test design R1, members of the Alinghi design team expressed concern that R1 would be at a disadvantage in actual racing compared to a boat with greater beam. As a result, a wider version of R1 was generated and tested in SPLASH. This hull is designated R1W in Figure 87. Although R1W was clearly superior upwind, it was at a disadvantage downwind against all other hulls, and its averaged upwind/downwind results were generally inferior to R1.





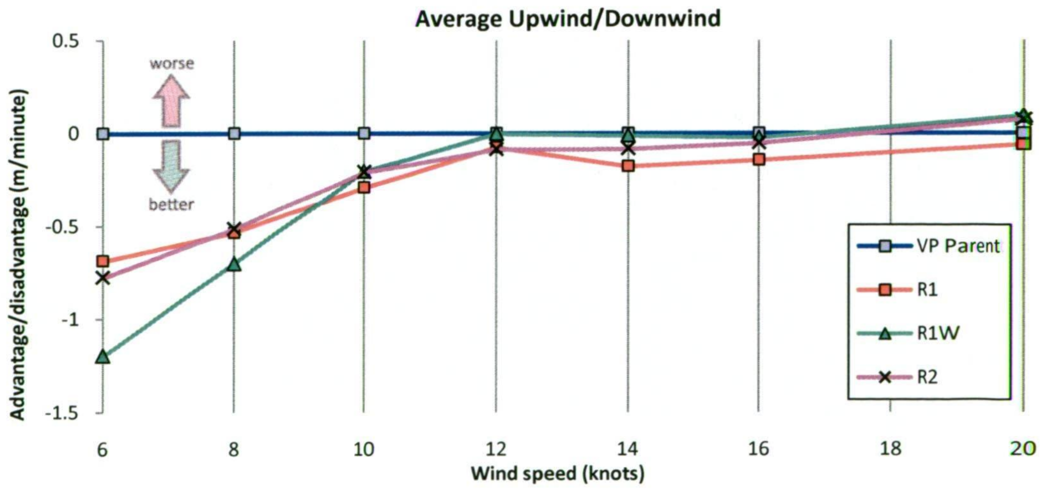


Figure 87. Performance comparison plots of test models versus parent model 1

9.2.1 TRAILING BOAT PENALTIES

The optimisation runs detailed above were performed with no trailing boat penalties or attempts to bias upwind versus downwind performance. To investigate the effects of trailing boat penalties, several runs were performed with a simple trailing boat penalty, with the results listed as runs 8, 9 and 10 of Table 20.

Table 20. Results With Trailing Boat Penalties Included – Parent Model 1

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	B_{WL} (m)	Flare (deg.)	C_M	Trailing boat penalty	Wally
Run 8	9	4	1.4	3.091	<0.1	0.85	6	no
Run 9	11	4	1.4	3.100	<0.1	0.85	6	no
Run 10	13	4	1.4	3.105	<0.1	0.85	6	no
Run 11	11	4	1.4	3.100	<0.1	0.87	6	no
Run 12	11	4	1.4	3.111	<0.1	0.87	6	yes

Compared to the tests listed in Table 18, trailing boat penalties increased the optimal beam by 50-60mm and resulted in the emergence of a correlation between B_{WL} and wind velocity.

C_M PARAMETER RANGE EXTENSION

One concern regarding these results was that the C_M values of the predicted boats were at the upper limit (0.85) of the range tested. In order to check for the possibility that the optimal design lay at C_M values higher than 0.85, a further seven hulls were generated with C_M values varying between 0.845 and 0.875. Three of these hulls are shown in Figure 88. It can be seen that the bilges of these boats are firm, but not to an extreme degree.

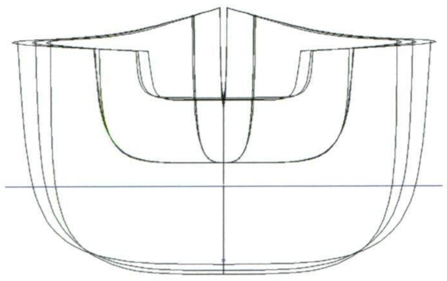


Figure 88. High C_M designs based on parent model 1

These seven hulls were analysed using SPLASH, their results combined with the previous training set, and neural network training performed. Only two optimisation runs could be performed in the small amount of time remaining prior to commencing work on parent model 2, with the results listed as runs 11 and 12 in Table 20. Although the recommended B_{WL} for run 11 was the same as for run 9, the recommended C_M for both runs increased to 0.87.

WALLYING TESTS

As the option for wallying in variable wind direction is included in VESPA's race model, a brief investigation was performed into the effect of wallying on the optimal design parameters.

One optimisation run was performed with wallying turned on, run number 12 in Table 20. Wind velocity parameters were the same as for run 11. However, the wind direction oscillated with a SD of 7 degrees. Although only one run was performed due to time constraints, it resulted in a small increase of 11mm in the optimal B_{WL}. This result was interesting but not conclusive, and other tests involving wallying were scheduled for later optimisation runs.

CONCLUSIONS – PARENT MODEL 1

VESPA located two optimal designs having lower B_{WL} and higher C_M values than parent model 1. These optimised designs were successfully validated using SPLASH analysis to provide comparisons with the SPLASH results of their parent models

The results of this testing raised the possibility that, in the absence of trailing boat penalties, the solution space is bi-modal i.e. having two distinct optimal designs. This situation may occur due to the format of the racecourse used, which has equal amounts of upwind and downwind sailing.

Design features that result in a fast boat upwind usually conflict with the factors that result in a fast boat downwind. In an event where an equal amount of time is spent on upwind and downwind legs, this may result in a situation where a boat that is fast upwind but slow downwind have similar elapsed time around the course when compared to a boat that is slow upwind but fast downwind.

This appears to be the case with designs R1 and R2. Despite having similar performance for an upwind/downwind course, these boats show different biases for upwind and downwind work. R1 has greater beam with zero flare and does well upwind while performing acceptably downwind. R2 is a narrower boat with 7.8 degrees of flare, and does very well downwind yet comparatively poorly upwind.

Factors that appear to favour one optimum over the other are the wind velocity variance and the degree of benefit conferred on the boat that leads around the first windward mark. The bimodality disappeared when an upwind bias in the form of a trailing boat penalty was applied, resulting in a single optimal design, which performed comparatively better upwind than down.

While flare appears to have some benefit in conditions that are highly variable, the expected variance in the wind velocity in Valencia during the summer months did not appear to be sufficient to justify the addition of flare to the design.

9.3 PARENT MODEL 2, JULY 30, 2006

Analysis of parent model 2 commenced in early August 2006. Parameter value ranges for this series are shown in Table 21. The choice of C_M range for parent model 2 was made before any optimisation could be performed using the extended parent model 1 dataset, and as a result the C_M range chosen did not extend higher than 0.87.

Table 21. Parameter Variation Limits – Parent Model 2

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Minimum	2.9	0	0.80	0.58	0.525
Maximum	3.4	8	0.87	0.60	0.535

For these runs, C_P and L_{CB} were fixed to values in the centre of the ranges analysed by SPLASH. The results of these runs, shown in Table 22, showed that, contrary to expectations, the optimal B_{WL} increased as mean wind velocity became stronger. It also became clear that VESPA preferred models with C_M values at the upper limit of the parameter range available.

Table 22. Parent Model 2 Results

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	B_{WL} (m)	Flare (deg.)	C_M	Trailing boat penalty	Wally
Run 1	9	4	1.4	2.900	0.0	0.863	0	no
Run 2	10	4	1.4	3.079	0.0	0.87	0	no
Run 3	11	4	1.4	3.112	0.0	0.87	0	no
Run 4	9	4	1.4	3.038	0.0	0.87	6	no
Run 5	11	4	1.4	3.125	0.0	0.87	6	no
Run 6	13	4	1.4	3.148	0.0	0.87	6	no

As the initial runs for parent model 2 showed little or no variation in optimal values for either C_M or flare, and as parent model 3 was due to be available for testing soon afterwards, the opportunity to perform a sensitivity analysis for the B_{WL} parameter was taken. This was achieved using VESPA as a tournament-modelling program, rather than as an optimiser, simply by limiting the number of generations to one.

For this analysis, a twenty-eight boat population was created which had identical design parameters aside from B_{WL}, which varied between 2.9 m and 3.4 m. C_M was fixed at 0.87 and flare set at zero degrees. Mean event wind velocity was set to 11 knots, event SD to 4 knots and race SD to 1.4 knots.

This population of yachts was analysed by VESPA, which calculated average race times as well win/loss ratios for each boat competing in a multi-race tournament against every other boat in the population. Additionally, each boat had its win/loss ratio calculated for a tournament against single opponents, termed exemplars. Two different exemplars were used, one with a B_{WL} of 3.05m and one with a B_{WL} of 3.2m.

Results of this analysis are displayed in Figure 89, showing win/loss ratios and average boat speed, normalised to the maximum for that optimisation run, for a range of B_{WL} values close to the optimal values.

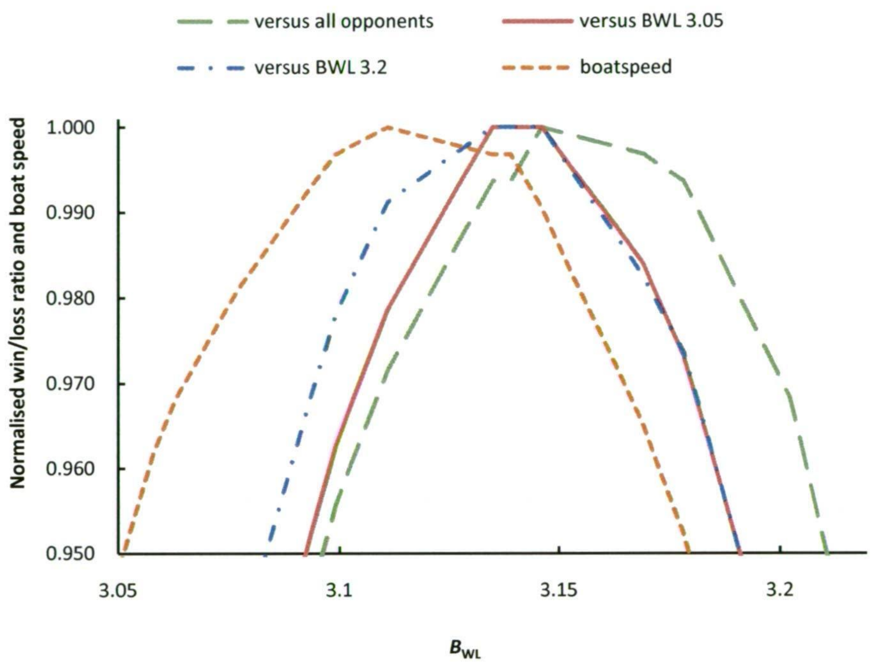


Figure 89. B_{WL} sensitivity analysis

The optimal B_{WL} value based on boat speed alone was 3.111m. However, when the win/loss ratio for each competitive scenario was considered, the optimal B_{WL} increased to between 3.135m and 3.146m.

A second sensitivity analysis was subsequently performed for C_M, which was varied between 0.82 and 0.89. B_{WL} was fixed at 3.05m. Results of this sensitivity analysis are shown in Figure 90. The win/loss ratio, determined using a tournament of match races against the entire population, is correlated with increasing C_M and reaches a maximum at 0.87, the highest C_M value available for this series of designs.

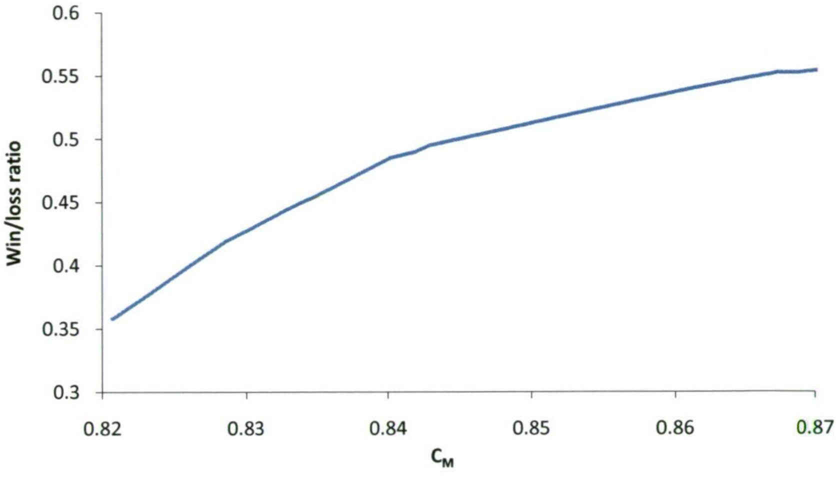


Figure 90. C_M sensitivity analysis

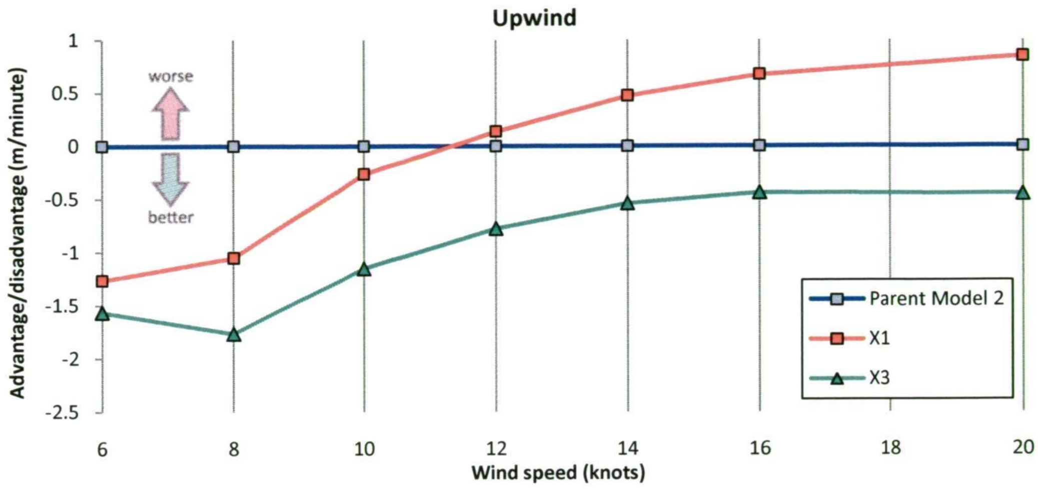
Based on these results, two test hulls were generated for comparison with the parent model using the parameters shown in Table 23. For comparison purposes, a narrow B_{WL} model with flare (X1) was generated to investigate the performance characteristics of this style of hull, as there was concern that this may have been a viable design option.

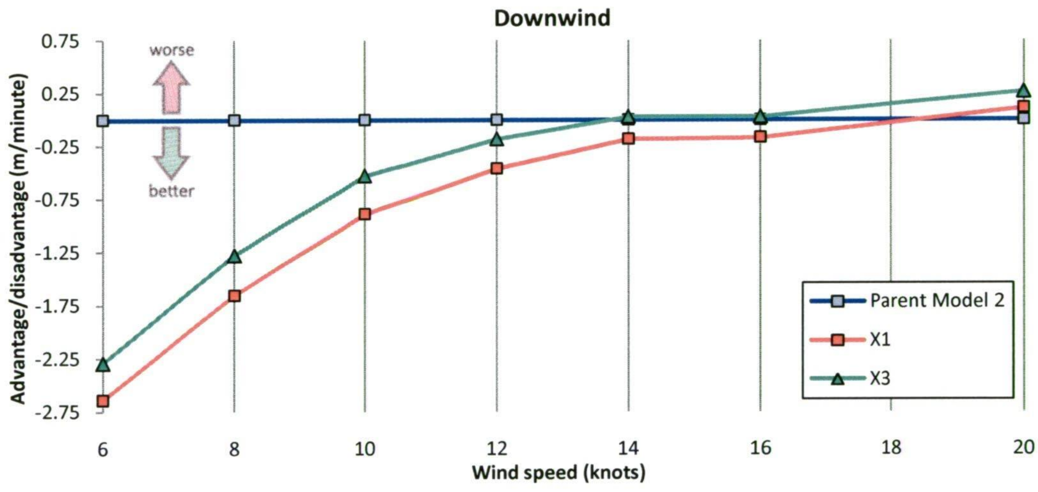
Table 23. Test Hull Parameter Values – Parent Model 2

	B_{WL} (m)	C_M	Flare (deg.)
X1	2.9	0.87	7
X3	3.100	0.87	0

Results of SPLASH testing of these two models are shown in Figure 91. Model X1 showed an improvement in performance relative to the parent model in wind velocities less than 12 knots, but performed poorly upwind in stronger winds. Model X3, on the other hand, showed superior performance to its parent model in all conditions upwind and in all but the strongest conditions downwind.

The performance of X3 was a pleasing confirmation that VESPA was successfully locating designs that were superior to those arrived at by manual design methods, within the accuracy constraints of the CFD code used. This is no guarantee that the design would actually prove superior at full scale, in actual sailing conditions, but locating a superior design within the limits of the design tools available was nonetheless a significant step forward.





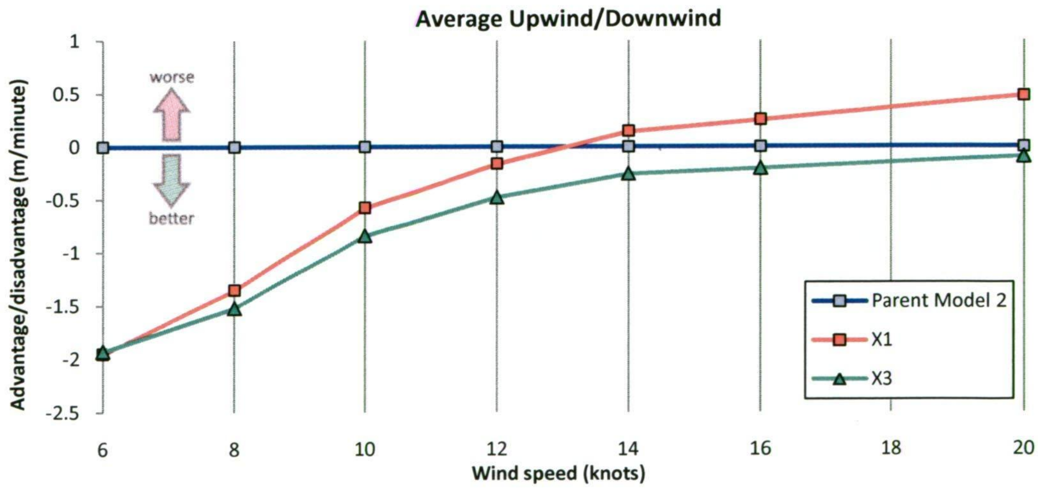


Figure 91. Comparison plots of X1 and X3 versus parent model 2

9.4 PARENT MODEL 3, AUGUST 18, 2006

The final parent model, known within Alinghi as MS, was optimised during a 9 day period in August 2006. Thirty-two design variations (25 training models and 7 test models) based on the parent were generated, and subsequently analysed using SPLASH. Parameter ranges for the training models are shown in Table 24.

Table 24. Parameter Variation Limits – Parent Model 3

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Minimum	2.9	0	0.80	0.575	0.520
Maximum	3.4	4	0.89	0.595	0.535

Neural network metamodels were created using the SPLASH output and optimisation runs performed for a variety of conditions using VESPA. Results for these runs are shown in Table 25. Rather than locking C_P and L_{CB} to specific values for these runs, a small range for each variable was allowed with upper limits of C_P = 0.592 and L_{CB}= 0.532

Table 25. Parent Model 3 Results

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	B_{WL} (m)	C_M	C_P	L_{CB}	Flare (deg.)	Trailing boat penalty (s)
Run 1	9	4	1.4	3.026	0.862	0.589	0.532	0	6
Run 2	10	4	1.4	3.042	0.863	0.589	0.532	0	6
Run 3	11	4	1.4	3.068	0.867	0.589	0.531	0	6
Run 4	13	4	1.4	3.110	0.877	0.589	0.532	0	6
Run 5	15	4	1.4	3.175	0.859	0.590	0.529	0	6

An evaluation of the optimal design using an exemplar, based on a probable competitor for the America's Cup final, was performed. Additionally, an evaluation was performed against a range of exemplars, which were based on a range of likely designs from challenging teams.

Table 26. Parent Model 3 Tests Versus Exemplars, Mean Wind Velocity = 11 knots, Event SD = 4 knots and Race SD = 1.4 knots

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Versus exemplar B_{WL} 3.05 C_M 0.885	3.075	0	0.88738	0.592	0.532
Versus all opponents	3.09	0	0.88877	0.592	0.532

These two experiments resulted in small increases in both B_{WL} and C_M, consistent with the hypothesis that VESPA would seek to gain a small advantage upwind over any specific exemplars.

An evaluation of Wallying was also performed for this parent design across a range of wind strengths to determine whether a significant effect was apparent. To simplify the analysis these tests were performed using fixed values for some design variables, with flare = 0°, C_P = 0.59 and L_{CB} = 0.532. Tests were performed with 100% wallying to exaggerate any differences. Test results are summarised in Table 27.

Table 27. Effects of Wallying on design optima – Parent Model 3

	Mean wind velocity (knts)	Event SD (knts)	Race SD (knts)	Wind direction SD (degr.)	B_{WL} (m)	C_M	Trailing boat penalty (s)
Run 6	9	4	1.4	10	3.025	0.868	6
Run 7	11	4	1.4	10	3.075	0.871	6
Run 8	13	4	1.4	10	3.119	0.867	6

These tests show that Wallying results in a small increase, approximately 5-10 mm, in the optimal B_{WL}. This difference is not sufficient to warrant the consideration of wallying as a factor in the design of an ACC yacht. This is particularly true for VESPA, as accounting for wallying requires the calculation, for a given wind velocity, of a portion of the yacht's performance polar-curve over a range of angles, rather than the calculation of a single V_{MG} value. This significantly reduces the performance of VESPA with little benefit in return.

9.5 ANALYSIS OF RESULTS

9.5.1 SENSITIVITY OF DESIGN FACTORS

The following are parameter sweeps produced by VESPA, keeping all parameters constant except for one free parameter, which was varied through a range of values. Baseline design parameters were B_{WL} = 3.1m, C_M = 0.87, flare = 0°, C_P = 0.589 and L_{CB} = 0.532. These sweeps were generated using a mean wind velocity of 11 knots, an event SD of 4 knots and a race SD of 1.4 knots.

Values on the Y axis are seconds per mile of V_{MG} relative to the baseline model, with lower values better than higher values. Other than the win/loss ratio curve shown in Figure 94, the vertical scales for all graphs show seconds per mile speed penalty relative to the optimal design. Vertical scales are equalised in order to simplify comparisons between the effects of different parameter variations.

Note that results are averages, taken from a stochastic model having a large number of samples. As a result, the curves produced are not perfectly smooth as might be expected if a deterministic model was used. Figure 92 displays the effect of variation in C_M on performance. This indicates a clear preference for higher C_M values, although there appears to be negligible benefit above a C_M of 0.87.

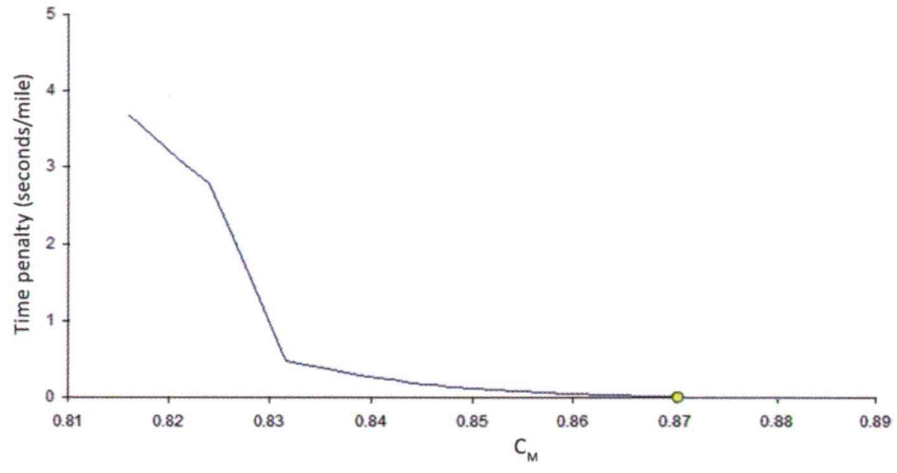


Figure 92. C_M sensitivity analysis

Figure 93 displays the effect of flare variation, showing a disadvantage for anything but the zero flare state. One anomaly apparent in this graph is the dip in the curve in the region of 5 to 6 degrees of flare. This may be interpreted as a remnant of the bimodality previously described. It is likely that if the wind velocity variance used for sensitivity analysis had been higher, the dip in the curve may have extended further towards the abscissa.

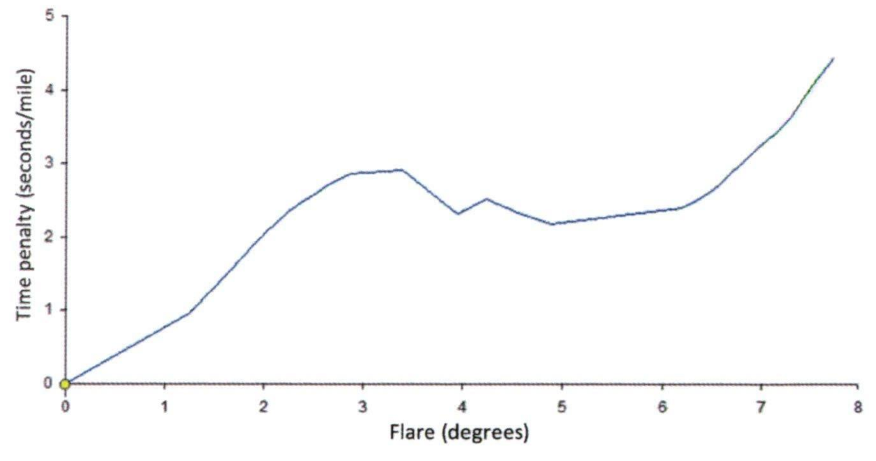


Figure 93. Flare sensitivity analysis

The final sweep, shown in Figure 94, is for variation in B_{WL}. This shows a preference for a beam, based on raw times, of approximately 3.025 m. However, when win/loss ratio is examined, the optimal beam increases to approximately 3.070 m.

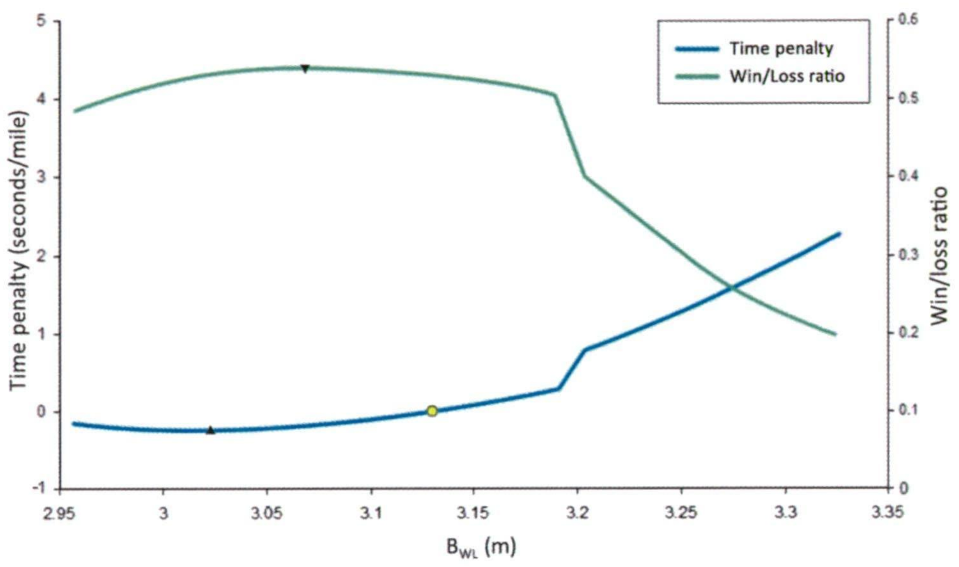


Figure 94. B_{WL} sensitivity analysis

Note that while the win/loss ratio curve appears to be similar to an inverted curve of raw times, there is a subtle difference in the shape of the curves. The peak of the win/loss curve is shifted to the right by approximately 50mm, as well as having a greater decline in value once a B_{WL} of 3.19 is exceeded.

9.5.2 CHOICE OF PARAMETERS FOR TANK TESTING

The result of this analysis was that two optimised designs were delivered to the Alinghi design team as candidates for a tank test model. The design parameters for these two boats are listed in Table 28.

Table 28. Tank Model Candidate Parameter Values – Parent Model 3

	B_{WL} (m)	Flare (deg.)	C_M	C_P	L_{CB}
Model A	3.100	0	0.87	0.589	0.532
Model B	3.140	0	0.87	0.589	0.532

While VESPA indicated that the narrower boat, Model A, was the faster of the two, Model B was considered a more versatile design in the event that wind conditions were significantly different from those expected during the final of the America's Cup. Model B was on the right shoulder of the optimal B_{WL} curve and was considered a more robust solution, particularly upwind. As a result, model B was selected as the basis for the construction a 1/3 scale tank-test model, shown in Figure 95.

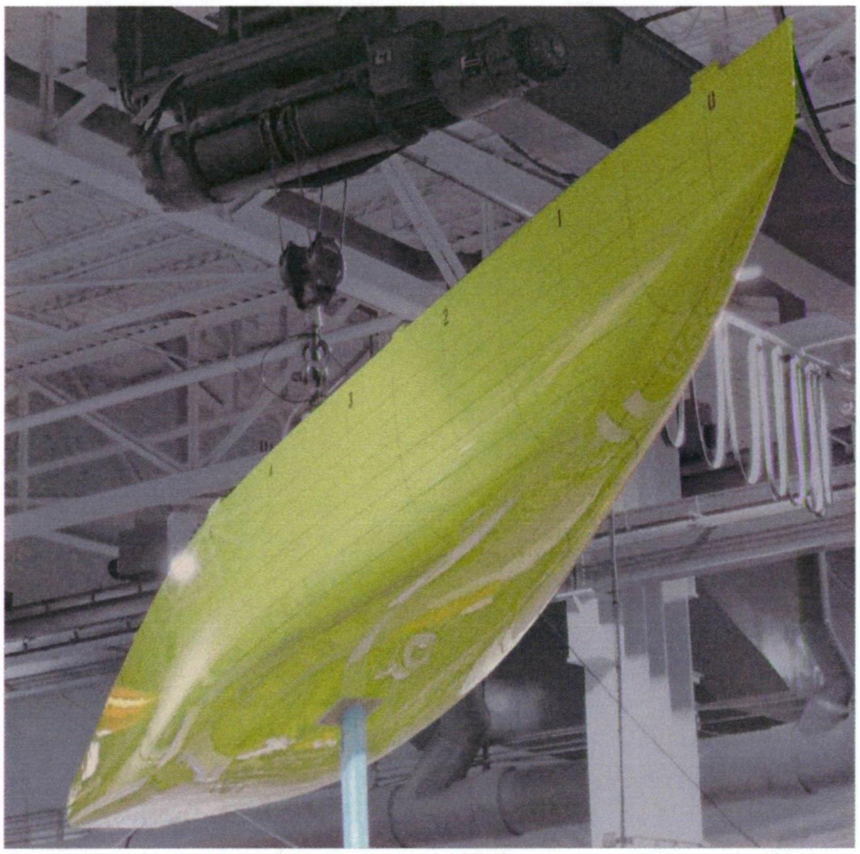


Figure 95. Tank model MT

9.5.3 VALIDATION AGAINST TANK TEST DATA

Tank testing of both the optimised model (MT) and parent model (MS) was performed during September 2006. Results are displayed in Figure 96 as comparison plots. These show performance comparisons calculated using SPLASH, as well as performance based on tank test data. The comparison baseline for each set of results is data from the equivalent source for the MS model, shown as a horizontal line.

The key feature of these results is not the specific performance of the MT model, which showed itself to be faster upwind in all wind velocities and slower downwind in light weather than MS. Rather, it is the close correspondence of the performance differences for the tank results and the SPLASH results that are significant. There is good agreement between SPLASH and tank data for the speed differences that MT would exhibit over its parent model.

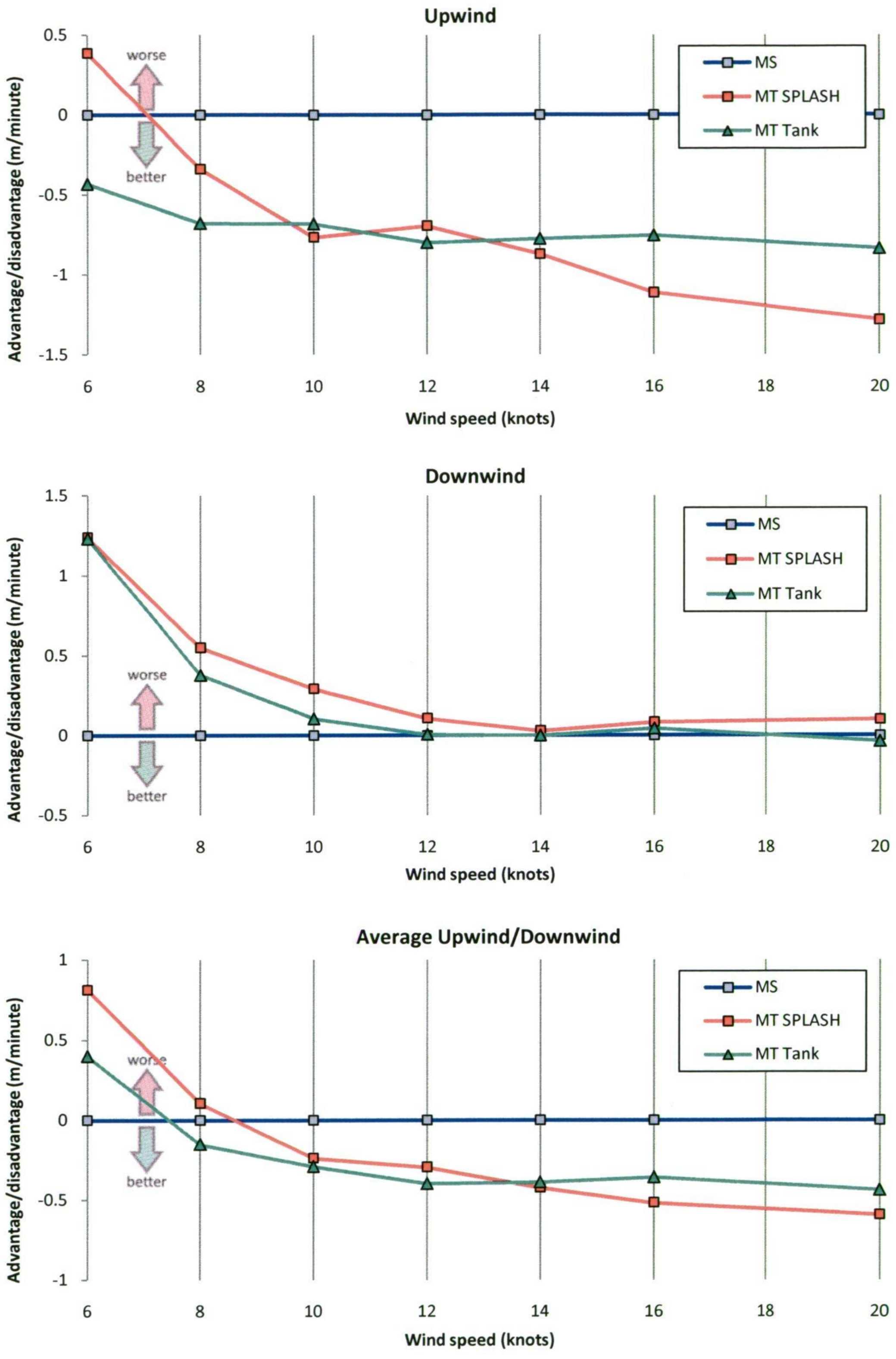


Figure 96. MS parent model versus MT, SPLASH and tank results comparison

9.5.4 OUTCOME

The MT tank model showed significant improvement over its parent model, MS. Unfortunately, when the MS model was tank tested, it was not found to be faster than the previous best design, MO. As a result, MT also did not show itself to be superior to MO.

This left the Alinghi design team in a difficult position. The recomendations made by VESPA across three parent models were remarkably consistent, with optimal values for each being close to 3.1m B_{WL} and 0.87 C_M. These parameters, known within the Alinghi design team as the X3 transformation, after the first design produced by VESPA that showed a significant advantage over its parent design, were considered by many to be applicable to any of the Alinghi design candidates.

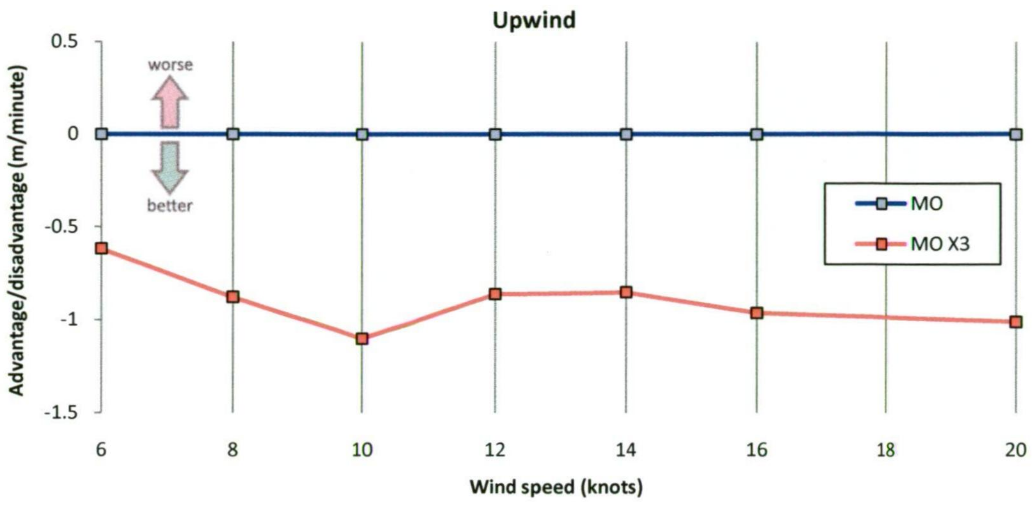
As a result, a design was generated taking MO as the parent model and applying the X3 transformation. The resulting design, MOX3, was analysed in SPLASH and PAP with encouraging results (Figure 97). Upwind performance in all conditions was excellent, while MO was slightly faster downwind in most conditions. Aggregate performance upwind and downwind was superior to MO in all wind strengths.

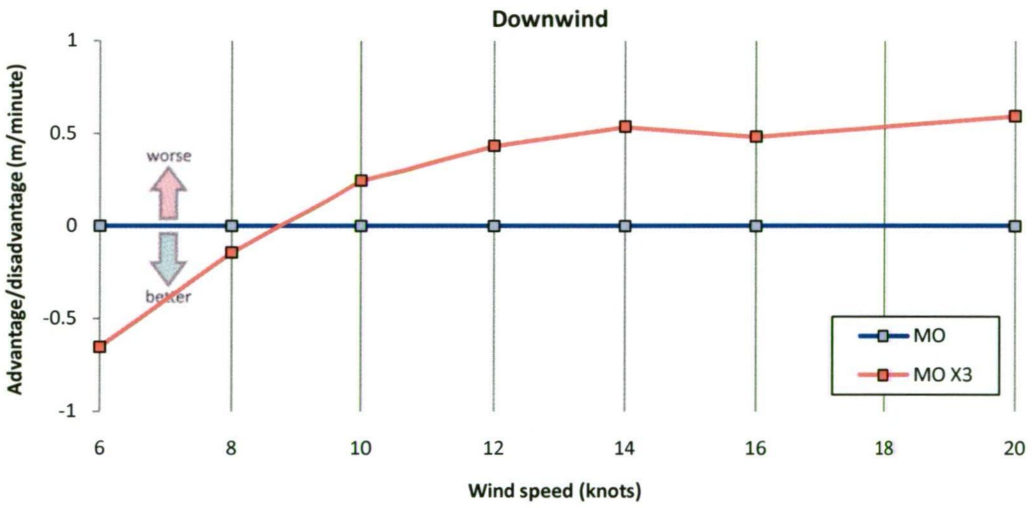
Examination of the performance comparison plots for designs optimised by VESPA using parent models 1 and 2 indicate that the optima found were biased towards lighter winds than their parents. This corresponded with the opinion of the Alinghi designers, who recognised that the performance profile of their second boat, SUI-100, should be focussed on lighter conditions than their first boat, SUI-91.

For parent model 3, this situation was reversed, with VESPA improving the performance in moderate to strong winds at the cost of some reduction in performance in lighter winds. This indicated that parent model 3 had been pushed too far in the direction of light wind performance. It is significant that the upwind performance curve of MOX3 is almost parallel to that of MO, indicating that VESPA concurred with the Alinghi designers as to the ideal performance profile for the final design.

Conversely, VESPA clearly preferred to trade off downwind performance in return for upwind performance for MOX3 relative to MO. Whether such a trade off would have had a positive influence on the outcome of the America's Cup final remains a matter for conjecture.

Regardless of any potential advantage shown by SPLASH testing, MOX3 had not been tank tested, and no time was available for further tank testing or design development. No matter how promising the design may have been based on CFD testing, it was universally agreed that the risks inherent in building a boat without confirmation from tank testing were unacceptable. As a result, MO was chosen as the design for SUI-100, the boat that subsequently won the 2007 America's Cup.





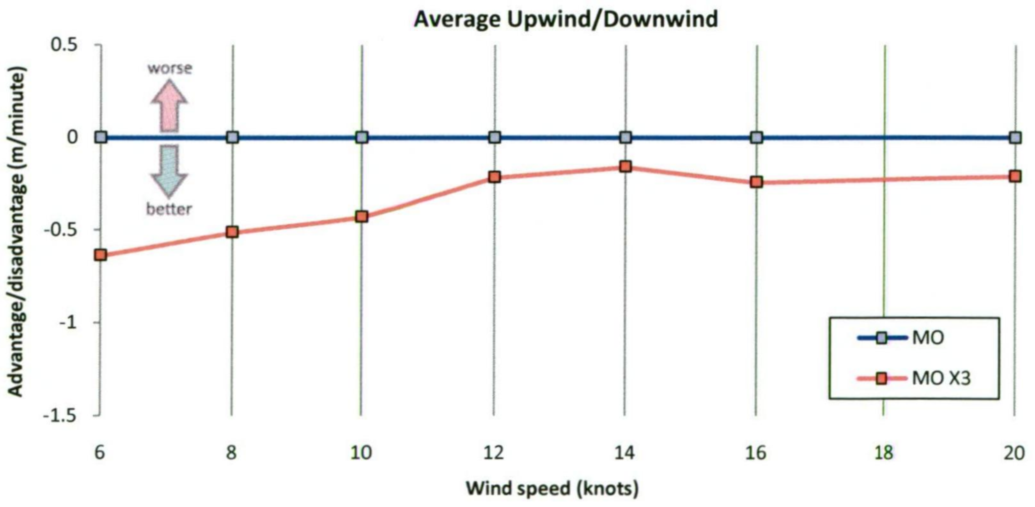


Figure 97. MO versus transformed design MOX3

9.6 SUMMARY

VESPA was used to make design recommendations based on three different parent designs. During this period, complications resulting from the competitive fitness function used by VESPA were identified, and steps taken to limit their adverse effects. These issues included:

- Recognition of the implications of game theory in relation to the choice of yacht design parameters for match racing. This suggests that the interactions between boats when approaching rounding marks results in a bias in favour of the leading boat, and that this bias may prevent a Nash equilibrium occurring for match races between specific yacht designs.
- Implementation of both a simple Hall-of-Fame method and optional use of exemplar designs by the GA.

The importance of high-quality metamodels was also recognised, with procedures put in place to ensure that metamodels had sufficiently low prediction error while avoiding problems with over-fitting.

Various investigations into factors that may have affected the choice of design parameters were performed, including:

- evaluation of the effects of different wind distributions, involving changes to both the mean wind velocity and its standard deviation, on the optimal design configuration;
- investigation into the effect of trailing boat penalties on the optimal design configuration;
- evaluation of the effect of wallying on the optimal design configuration.

Designs considered optimal by VESPA were re-analysed using SPLASH, and in the case of parent model 3, by tank testing of a model based on the optimised hull shape. Results from these analyses were compared to equivalent data derived from the parent models to verify that VESPA had been able to make genuine design improvements.

Despite significant variations in parent models used, the results provided by VESPA were remarkably consistent. The design parameter values recommended for each parent model are similar, and where differences do occur they are primarily because of variations in the design characteristics of the parent model.

10 · CONCLUSIONS

The VESPA project was intended to produce an automated optimisation system that could locate the optimal design parameters for an ACC yacht for a given set of weather conditions, while executing in a reasonable time on commonly available personal computer hardware.

In order to achieve these objectives, many disparate components had to be integrated into a functional and reliable system. These components comprised:

- **Measure of merit.** Rather than restrict the measure of merit to what was achievable using a particular optimisation approach, the aim for VESPA was to determine the ideal measure of merit and to design the system around it. The choices made in the design of VESPA are a consequence of the characteristics of this measure of merit,

  Rather than using a measure of merit based on the performance of a single boat, or on the results of a race between two boats, VESPA implements a tournament model where a boat's fitness is based on its performance against multiple boats, across multiple races, in a range of weather conditions.

- **Parametric transformation.** The parametric transformation function was required to satisfy a broad range of requirements. Transformed hulls must match the sometimes-conflicting set of design parameter values, while retaining the fairness and convexity characteristics of the parent hull, and simultaneously satisfying the constraints of the ACC rule. To satisfy these demands, an innovative parametric transformation method was devised. This combined powerful capabilities for the deformation of hull geometry with a concise numerical representation, ideally suited for use by an optimisation algorithm.

- **Sampling method.** A modern Design of Experiments method, the Uniform Design, was selected from a range of candidate DOE methods, based on the high degree of uniformity exhibited by its sampling.
- **ACC rule constraints.** The optimisation process undertaken by VESPA had the luxury of being unconstrained and having a single objective, due both to the nature of the measure of merit used and the form of the ACC rule. The ACC rule acts as a set of penalty constraints on the design variables, ensuring that only rule-feasible combinations of design variables are evaluated. All designs evaluated by VESPA are checked by a code module that verifies and, in conjunction with the parametric transformation function, enforces ACC rule compliance.
- **CFD Analysis.** A proven potential-flow CFD program, SPLASH, was used to provide hydrodynamic analysis of the various hull designs provided by the parametric transformation function.
- **Metamodel selection and fitting.** A review of recent research into the use of data approximation methods revealed that many candidates for metamodel representations exist. The determination of which method is best suited to a particular application must be based on a variety of factors, not least of which are the validation and verification features of the software used to create the metamodels. In the case of VESPA, an evaluation of the requirements revealed that artificial neural networks were best suited to the specific problem by virtue of their approximation properties, their ease of use and their availability in powerful commercially available software packages.
- **Metamodel re-creation.** To allow the neural-network metamodels to be used within VESPA, as well as by a variety of spreadsheet macros used for testing and validation of the system, a software module was written that could re-create a particular neural network from a definition contained in an XML file
- **Performance prediction.** VESPA was able to utilise an established performance prediction program supplied by the Alinghi team. This program, PAP, was modified to allow it to use neural-network metamodels as a source of hydrodynamic data, and to allow it to be called from the VESPA race-modelling program as a dynamically linked library.
- **Race modelling.** The VESPA race-modelling program was designed to be an accurate yet efficient representation of the simplest race that would correctly rank a pair of yachts. This was a departure from the trend toward increasingly detailed simulations of physical systems, necessitated by the requirement for computational efficiency, as well as by the pitfalls of attempting to model

highly complex systems in detail. While it is not feasible to accurately quantify the random variation in each of the multiple components of a detailed simulation, it is possible to make a reasonable estimate of the overall variance, and this approach has been adopted for the VESPA race model.

- **Tournament modelling.** VESPA uses a measure of merit based on a tournament of races featuring multiple boats sailing in multiple races, across a range of weather conditions. This approach provides a superior estimate of the probability of a yacht winning a series of races, particularly when the population can be seeded with individuals based on the design parameters of known or expected competitors.
- **Optimisation algorithm.** A review of optimisation methods indicated that a genetic algorithm was best suited to the problem posed by VESPA, due to the ability of the GA to handle multi-modal, non-linear solutions, and the need to optimise a population of candidate designs.
- **Testing and validation of components.** Each component created for VESPA was extensively tested for correct functioning, and validated to ensure accuracy of results
- **Testing and validation of results.** No simulation of a complex, real-world system can be one-hundred percent accurate, and for the optimisation of non-linear systems, a small inaccuracy can result in a large error. All design recommendations produced by VESPA were subsequently checked using SPLASH and PAP to verify their performance. Importantly, this verification was performed by members of the Alinghi design team in Spain, making it an independent test of the system's efficacy.

Once assembled and tested, VESPA worked remarkably well, showing itself capable of making genuine improvements to highly developed parent models, using modest computer resources. Optimised designs were shown to be superior to their parent designs when directly re-analysed using SPLASH, and in one case, when tank tested at one-third scale. Execution time for an optimisation run was reasonable, taking approximately 12 hours on a 2.4 GHz Pentium based system.

10.1 CRITICAL ASSESSMENT & CONTRIBUTION

The development of VESPA required the integration of many complex components. Failure of any one of these components to perform as required would have rendered the system unworkable. Significant verification and validation of each step in the process was implemented to ensure a viable optimisation system. This degree of rigour was considered essential, as the goal of the research was a system capable of producing design recommendations of value to the designers of the America's Cup defender.

This research endeavoured to achieve objectives that had not previously been attempted. Although other researchers had devised yacht-design optimisation systems that solved simplified subsets of the problem, the scope of work involved in the creation of VESPA was substantial, and the goals of the system ambitious.

The design of ACC yachts has progressed over a period of 15 years, to the point where current designs are highly optimised and the scope for further improvement in is small. Consequently, the accuracy required of VESPA's analysis and simulation tools was high, as the smallest of errors might result in erroneous outcomes from the optimisation process.

Despite undertaking a complex real-world optimisation problem, involving designs that had already undergone thousands of hours of design development, numerical analysis and tank testing, VESPA was able to make a meaningful contribution to the design development process of the Alinghi team. While limited to some extent by the accuracy of the CFD tools available, VESPA was able to achieve verifiable improvements to the highly refined parent models supplied by Alinghi.

The research described by this thesis makes significant contributions to the field of racing-yacht design and to the field of naval architecture in general. The use of metamodels for complex non-linear hydrodynamic data has been shown to be a viable approach, as has the sparse, quasi-random sampling of the design space to provide a concise basis for those metamodels. A novel parametric transformation method has been presented that combines a parsimonious set of control parameters with a powerful set of geometric controls, permitting variation of hull volumes longitudinally, transversely and vertically without violating fairness and convexity conditions. Lastly, the combination of an efficient race simulation with a tournament-modelling algorithm permitted the evaluation of a racing yacht in the most appropriate manner - via a comparison with a diverse assembly of its peers.

10.2 ADVANTAGES AND DISADVANTAGES OF VESPA

Currently the weak link in the methodology is the accuracy of the results produced by the CFD analysis, as small errors in this area can result in misleading outcomes from the optimisation process. However, CFD is a field undergoing constant development and it is likely that accuracy will improve in the future.

Alinghi also has access to two RANS codes, FLUENT and CFX. It was considered that the execution times of these programs were too long given the number of data points needing to be calculated, while the improvement in accuracy would be small. Although these programs can be run on a cluster of computers, the performance differential compared to SPLASH running on a single CPU is still significant.

It is possible, due to improvements in both hardware and software, that RANS codes may be sufficiently fast for use with VESPA at some time in the near future. If this was to occur, the change could be easily accommodated within VESPA, with the data from the RANS code being indistinguishable from data derived from SPLASH. Little or no change to the data preparation sequence would be required.

The use of metamodels by VESPA is an approach whose value increases as execution times for the CFD analysis grow larger. Consequently, it is anticipated that the use of metamodels will be of even greater benefit when used with RANS codes whose execution times are substantially higher than potential flow codes such as SPLASH.

Although VESPA has been specifically targeted at ACC design, it is equally appropriate to other forms of yacht design, including long distance ocean racing and fleet racing. In these cases, the objective function may need to be changed from being tournament based to being based on aggregate course times, and a different optimisation method may be more appropriate than the GA currently used by VESPA. However, use of non-linear metamodels to approximate the results of an expensive CFD analysis, while using a stochastic RMP to model the tournament structure of the actual racing, would remain as principal features of the system.

10.3 FUTURE WORK AND OUTLOOK

Although the scope of work undertaken in the creation of the VESPA system was broad, not all aspects of the problem could be addressed without raising the level of complexity above what was practicable within the time and resources available. As a result, there are many refinements that may be made to VESPA in the future. These include:

- adaptation of the method to vessels other than ACC yachts;
- incorporation of potentially superior CFD methods, such as Reynolds averaged Navier-Stokes (RANS) codes;
- incorporation of other metamodelling techniques such as radial basis function networks (RBF);
- investigation of automated, variable-fidelity models;
- parallelisation of the genetic algorithm to allow it to run on multiple CPU cores simultaneously, or alternatively, on low-cost, massively-parallel multicore GPU systems;
- blending of multiple parents as a complementary design variation method;
- extension of VESPA to encompass other design aspects, such as sail and rig optimisation, or multidisciplinary problems such as keel, bulb and rudder design. Currently the VESPA system is focussed on hull shape optimisation

only, however there is no reason why other components of a racing yacht should not be part of the optimisation process;

- incorporation of additional hydrodynamic analysis methods, such as added resistance in waves, into the simulation model. These refinements would, in turn, require the variation of additional design parameters, such as the independent movement of L_{CF} relative to the L_{CB} location. Additional hydrodynamic effects such as added resistance could be satisfied using direct analysis methods, or alternatively, using additional pre-calculated metamodels.

The America's Cup is currently in hiatus, due to legal challenges that have resulted in a competition between only two teams, to be sailed in large multihulls during February 2010. Once this event is concluded, it is likely that the America's Cup will return to ballasted monohulls, but in a new class that will replace the current America's Cup Class yachts.

A new class will provide a significant opportunity for the further development and use of VESPA. Rather than coaxing minute improvements from designs that are already close to optimal due to being governed by a mature rating rule, all designers will be working from a clean sheet of paper. In this situation, VESPA may be able to dramatically shorten design development time by rapidly isolating the most promising regions of the design space and exploiting them in an efficient manner.

PUBLICATIONS

The following are publications by the author, relevant to the subject matter contained in this thesis:

Couser, P., Mason, A. P., Von Konsky, B., Smith, C. & Mason, G. 2004, 'Artificial neural networks for hull resistance prediction', *COMPIT 04*, Siguenza, Spain.

Mason, A. P., Couser, P., Von Konsky, B., Smith, C. & Mason, G. 2005, 'Optimisation of vessel resistance using genetic algorithms and artificial neural networks', *COMPIT 05*, Hamburg, Germany.

Mason, A. P. & Thomas, G. 2007, 'Stochastic optimisation of IACC yachts', *COMPIT 07*, Cortona, Italy.

Mason, A. P. 2008, 'Stochastic optimisation for America's Cup Class yachts based on a genetic algorithm', *Schiffstechnik*, vol. 55, pp. 60-77.

REFERENCES

Abt, C., Bade, S. D., Birk, L. & Harries, S. 2001a, 'Parametric hull form design - a step towards one week ship design', *PRADS 2001*, Shanghai.

Abt, C., Harries, S., Heimann, J. & Winter, H. 2003, 'From redesign to optimal hull lines by means of parametric modeling', *COMPIT 03*, Hamburg.

Abt, C., Harries, S. & Hochkirch, K. 2001b, 'Efficient geometric modeling in modern yacht design', *Symposium für Yachtbau und Yachtkonstruktion*, Hamburg.

Abt, C., Harries, S. & Hochkirch, K. 2004, 'Constraint management for marine design applications', *9th Symposium on Practical Design of Ships and Other Floating Structures*, Luebeck-Travemuende, Germany.

ACC 2003, America's Cup Class rule version 5.0.

Akaike, H. 1974, 'A new look at the statistical model identification', *IEEE Transactions Auto. Control.*, vol. 19, pp. 716-723.

Alkan, A. D., Gulez, K. & Yilmaz, H. 2004, 'Design of a robust neural network structure for determining initial stability particulars of fishing vessels', *Ocean Engineering*, vol. 31, no. 5-6, pp. 761-777.

Alonso, J. J., LeGresley, P. & Pereyra, V. 2009, 'Aircraft design optimization', *Applied Mathematics and Computers in Simulation*, vol. 79, no. 6, pp. 1948-1958.

Alyuda 2005, *Neurointelligence Manual.*

Baik, J. & Gonella, P. 2005, *Automatic optimisation of IACC yacht hulls*, M.Sc. thesis, Chalmers University of Technology, Göteborg, Sweden.

Bales, N. K. 1980, 'Optimizing the seakeeping performance of destroyer-type hulls', *13th ONR Symposium on Naval Hydrodynamics*, Tokyo, Japan.

Bezier, P. 1974, 'Mathematical and practical possibilities of UNISURF', in *Computer Aided Geometric Design*, eds. R. E. Barnhill & R. F. Riesenfeld, Academic Press, New York.

Birk, L. 2005, 'Parametric modelling and optimization of offshore structure', *ICCAS 2005*, Busan.

Birk, L. & Harries, S. 2000, 'Automated optimization - a complementing technique for the hydrodynamic design of ships and offshore structures', *COMPIT 2000*, Potsdam/Berlin, Germany.

Birmingham, R. & Smith, T. A. G. 1998, 'Automatic hull form generation: a practical tool for design and research', *Practical Design of Ships and Mobile Units*, The Hague.

Birmingham, R., Webster, B., Roskilly, T. & Jones, E. 2002, 'The application of artificial intelligence to roll stabilisation for a range of loading and operating conditions', *HISWA 2002, The International Symposium on Yacht Design and Yacht Construction.*

Bole, M. & Lee, B. S. 2006, 'Integrating parametric hull generation into early stage design', *Schiffstechnik*, vol. 53, pp. 115-137.

Box, G. E. P. & Wilson, K. B. 1951, 'On the experimental attainment of optimum conditions (with discussion)', *Journal of the Royal Statistical Society, Series B (Methodological)* vol. 13, no. 1, pp. 1-45.

Burges, C. J. C. 1998, 'A tutorial on support vector machines for pattern recognition', *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121-167.

Campana, E. F., Fasano, G., Peri, D. & Pinto, A. 2007, 'Nonlinear programming approaches in the multidisciplinary design optimization of a sailing yacht keel fin', *9th International Conference on Numerical Ship Hydrodynamics*, Ann Arbor, Michigan.

Campana, E. F., Peri, D., Tahara, Y. & Stern, F. 2006, 'Shape optimization in ship hydrodynamics using computational fluid dynamics', *Journal of Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 1-3, pp. 634-651.

Černý, V. 1985, 'A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm', *Journal of Optimization Theory and Applications*, vol. 45, no. 1, pp. 41-51.

Chance, B., Jr. 1987, 'The design and performance of Twelve Meter yachts', *Proceedings, American Philosophical Society*, vol. 131 no. 4

Chen, B., Zhao, C. & Qiu, X. 1996, 'Global optimization of ship main dimensions by simulated annealing algorithm', *Shipbuilding of China*, vol. 2, no. 133, pp. 16-26.

Chen, E. & Parent, R. 1989, 'Shape averaging and its applications to industrial design ', *IEEE Computer Graphics and Applications* vol. 9, no. 1, pp. 47-54.

Chen, P.-F. 2004, *The inverse design studies in estimating the optimal hull form,* Ph.D. thesis, National Cheng Kung University, Tainan, Taiwan.

Chen, V. C. P., Tsui, K.-L., Barton, R. R. & Allen, J. K. 2003, 'A review of design and modeling in computer experiments', *Handbook of Statistics,* no. 22, pp. 231-261

Clarey, C. 2000, 'Stars and Stripes loses on land and on the sea ', *NY Times,* January 10, 2000.

Claughton, A. R. & Oliver, J. C. 2003, 'Development in hydrodynamic force models for velocity prediction programs', *R.I.N.A. International Conference on the Modern Yacht.*

Couser, P., Mason, A. P., Von Konsky, B., Smith, C. & Mason, G. 2004, 'Artificial neural networks for hull resistance prediction', *COMPIT 04,* Siguenza, Spain.

Cramer, N. L. 1985, 'A representation for the adaptive generation of simple sequential programs', *International Conference on Genetic Algorithms and Their Applications,* Pittsburgh, PA.

Day, A. H. 1993, 'Steps towards an optimal sailplan', *RINA Spring Meeting 1993.*

Day, A. H. & Doctors, L. J. 1997, 'Resistance optimization of displacement vessels on the basis of principal parameters', *Journal of Ship Research,* vol. 41, no. 4, pp. 249-259.

Day, A. H. & Doctors, L. J. 2000, 'The survival of the fittest - evolutionary tools for hydrodynamic design of ship hull form', *RINA Transactions,*

De Boor, C. & Ron, A. 1990, 'On multivariate polynomial interpolation', *Constructive Approximation,* vol. 6, no. 3, pp. 287-302.

de Jong, E. D. 2004, 'Intransitivity in coevolution', *Parallel Problem Solving from Nature - PPSN VIII,* Birmingham, UK.

de Jong, K. A. 1975, *An analysis of a behaviour of a class of genetic adaptive systems,* Ph.D. thesis, University of Michigan, Ann Arbor.

DeBord, F., Reichel, J., Rosen, B. & Fassardi, C. 2002, 'Design optimization for the International America's Cup Class', *SNAME Transactions,* vol. 110

Dejhalla, R., Mrša, Z. & Vukovic, S. 2001, 'Application of genetic algorithm for ship hull form optimization', *International Shipbuilding Progress,* vol. 48, no. 2, pp. 117-133.

Delhommeau, G. 1987, *Les problèmes de diffraction-radiation et de résistance de vagues: Etude théorique et résolution numérique par la méthode des singularités,* Thèse de doctorat d'état, Université de Nantes, Nantes.

Desideri, J.-A., Duvigneau, R., Abou El Majd , B. & Tang, Z. 2006, 'Algorithms for efficient shape optimization in aerodynamics and coupled disciplines', *42nd AAAF Congress on Applied Aerodynamics*, Sophia-Antipolis, France.

Duvigneau, R. 2007, *Aerodynamic shape optimization with uncertain operating conditions using metamodels*, No.6143, INRIA.

Duvigneau, R. & Visonneau, M. 2001, 'Towards a practical design optimization tool for incompressible and turbulent flows', *4th Numerical Towing Tank Symposium,* Hamburg, Germany.

Duvigneau, R. & Visonneau, M. 2002, 'Hybrid genetic algorithms and neural networks for fast CFD-based design'.

Duvigneau, R., Visonneau, M. & Deng, G. B. 2002, 'On the role played by turbulence closures in hull shape optimization at model and full scale', *24th. ONR Symposium on Naval Hydrodynamics.*

Dyn, N., Levin, D. & Rippa, S. 1986, 'Numerical procedures for surface fitting of scattered data by radial basis functions', *SIAM Journal of Scientific and Statistical Computing,* vol. 7, no. 2, pp. 639-659.

Eberhart, R. C. & Kennedy, J. 1995, 'A new optimizer using particle swarm theory', *Sixth International Symposium on Micromachine and Human Science,* Nagoya, Japan.

Eiben, A. E. & Schippers, C. A. 1998, 'On evolutionary exploration and exploitation', *Fundamenta Informaticae,* vol. 35

Evans, J. H. 1959, 'Basic design concepts', *Naval Engineers Journal,* vol. 71, no. 4, pp. 671-678.

Fang, K.-T. 2004, *Uniform experimental design tables,* http://www.math.hkbu.edu.hk/UniformDesign/main.html#ud_tables, accessed September 2005.

Fang, K.-T. 2006, 'Recent development in the uniform experimental design', *Int. Conf. on Design of Experiments and its Applications,* Tianjin

Fang, K.-T., Lin, D. K. J., Winker, P. & Zhang, Y. 2000, 'Uniform design: theory and application', *Technometrics,* vol. 42, no. 3 pp. 237-248.

Fassardi, C. & Hochkirch, K. 2006, 'Sailboat design by response surface optimization', *2nd High Performance Yacht Design Conference*, Auckland, New Zealand.

Fisher, R. A. 1935, *The design of experiments*, Oliver and Boyd, Edinburgh.

Fletcher, R. 1987, *Practical methods of optimisation*, Wiley.

Fogel, L. J., Owens, A. J. & Walsh, M. J. 1966, *Artificial intelligence through simulated evolution*, Wiley.

Formation Design Systems 2006, *Maxsurf User Manual, version 12*, Formation Design Systems.

Friedman, J. H. 1991, 'Multivariate adaptive regression splines (with discussion)', *Annals of Statistics*, vol. 19, no. 1, pp. 1-141.

Galan, M., Winter, G., Montero, G., Greiner, D., Periaux, J. & Mantel, B. 1996, 'A transonic flow shape optimisation using genetic algorithms', *ECCOMAS conference on numerical methods in engineering No.2*, Paris, France.

Gerritsma, J., Onnink, R. & Versluis, A. 1981, 'Geometry, resistance and stability of the Delft yacht hull series', *International Shipbuilding Progress*, vol. 28, pp. 276-297.

Giunta, A. A. & Wojtkiewicz, S. F. J. 2003, *Overview of modern design of experiments methods for computational simulations*, AIAA 2003-0649, Sandia Laboratories.

Goldberg, D. & Richardson, J. 1987, 'Genetic algorithms with sharing for multimodal function optimization', *2nd International Conference on Genetic Algorithms*.

Goldberg, D. E. 1989, *Genetic Algorithms in search, optimization and machine learning*, Addison-Wesley, 412 pages.

Greenman, R. M. 1998, *Two-dimensional high-lift aerodynamic optimization using neural networks*, NASA/TM-1998-112233, NASA Ames Research Center, Moffett Field, California.

Grefenstette, J. J. 1986, 'Optimization of control parameters for genetic algorithms', *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, no. 1, pp. 122-128.

Gretzky, J. A. & Marshall, J. K. 1993, 'The Partnership for America's Cup Technology: an overview', *11th Chesapeake Sailing Symposium*, Annapolis.

Halley, D. 1987, *On the systematic variation of hull representations for computers*, Royal Institution of Naval architects.

Harries, S. & Abt, C. 1999a, 'Formal hydrodynamic optimization of a fast monohull on the basis of parametric hull design', *5th International Conference on Fast Sea Transportation (FAST 1999)*, Seattle.

Harries, S. & Abt, C. 1999b, 'Parametric design and optimization of sailing yachts', *14th Chesapeake Sailing Symposium*, Annapolis.

Harries, S., Abt, C., Heimann, J. & Hochkirch, K. 2006, 'Advanced hydrodynamic design of container carriers for improved transport efficiency', *Design & Operation of Container Ships*, London.

Harries, S., Abt, C. & Hochkirch, K. 2001a, 'Hydrodynamic modeling of sailing yachts', *15th Chesapeake Sailing Yacht Symposium*, Annapolis.

Harries, S., Birk, L. & Abt, C. 2003a, 'Parametric hull design - the FRIENDSHIP-modeler', *NAV 2003*. Palermo, pp.

Harries, S., Janson, C. E., Leer-Anderson, M., Marzi, J., Maisonneuve, J. J., Raven, H. & Valdenazzi, F. 2003b, 'The FANTASTIC Ro-Ro: CFD optimisation of the forebody and experimental verification', *NAV 2003*, Palermo.

Harries, S., Valdenazzi, F., Abt, C. & Viviani, U. 2001b, 'Investigation on optimization strategies for the hydrodynamic design of fast ferries', *6th International Conference on Fast Sea Transportation (FAST 2001)*, Southampton.

Hearn, G. E. & Wright, P. N. H. 1997, 'Seakeeping for design: optimisation of motion responses and wave making resistance of catamarans via the application of a genetic algorithm', *4th International Conference on Fast Sea Transportation (FAST 1997)*, Sydney.

Heimann, J. & Harries, S. 2003, 'Optimization of the wave-making characteristics of fast ferries', *7th International Conference on Fast Sea Transportation (FAST 2003)*, Ischia, Italy.

Hendrix, D., Percival, S. & Noblesse, F. 2001, 'Practical hydrodynamic optimization of a monohull', *SNAME Transactions*, vol. 109, pp. 173-183.

Hess, D. E., Faller, W. E., Ammen, E. S. & Fu, T. C. 2004, 'Neural networks for naval applications', *COMPIT 04*, Siguenza, Spain.

Hess, J. L. & Smith, A. M. O. 1966, 'Calculation of potential flow about arbitrary bodies', *Progress in Aeronautical Science*, pp. 1-138.

Hestenes, M. R. & Stiefel, E. 1952, 'Methods of conjugate gradients for solving linear systems', *Journal of Research of the National Bureau of Standards,* vol. 49, no. 6, pp. 409-436.

Hirata, N. 2004, 'Comparison of genetic algorithm and gradient-based method to ship shape optimization', *Journal of Kansai Society of Naval Architects,* no. 241, pp. 59-65.

Hochkirch, K., Röder, K., Abt, C. & Harries, S. 2002, 'Advanced parametric yacht design', *High Performance Yacht Design Conference,* Auckland.

Holland, J. 1975, *Adaptation in natural and artificial systems,* University of Michigan Press, Ann Arbor.

Holtrop, J. & Mennen, G. G. J. 1978, 'A statistical power prediction method', *International Shipbuilding Progress,* vol. 25 no. 290, pp. 253-256.

Hooke, R. & Jeeves, T. A. 1961, '"Direct search" solution of numerical and statistical problems', *Journal of the ACM,* vol. 8, no. 2, pp. 212-229.

Hsiung, C.-C. 1981, 'Optimal ship forms for minimum wave resistance', *Journal of Ship Research,* vol. 25, no. 2, pp. 95-116.

Hsiung, C.-C. & Shenyan, D. 1984, 'Optimal ship forms for minimum total resistance', *Journal of Ship Research,* vol. 28, no. 3, pp. 163-172.

Hsiung, C. & Shenyan, D. 1985, 'Applying resistance theory and quadratic programming method to determine optimal ship forms', *Workshop on Developments in Hull Form Design,* Wageningen, Netherlands.

Hurrion, R. D. & Birgil, S. 1999, 'A comparison of factorial and random experimental design methods for the development of regression and neural network simulation metamodels', *The Journal of the Operational Research Society,* vol. 50 no. 10 pp. 1018-1033.

IMS 2004, IMS - International Measurement System, a handicapping system for cruising/racing yachts.

Islam, M. M., Khondoker, M. R. H. & Rahman, C. M. 2001, 'Application of artificial intelligence techniques in automatic hull form generation', *Ocean Engineering,* vol. 28, no. 12, pp. 1531-1544.

Jacquin, E., Alessandrini, B., Bellevre, D. & Cordier, S. 2002, 'Yacht optimisation based on genetic algorithm and RANSE solver', *High Performance Yacht Design Conf.,* Auckland.

Jagoda, J. 1973, 'Computer-aided multi-level optimization method applied to economic ship design', *International Conference on Computer Applications in the Automation of Shipyard Operation and Ship Design,* Tokyo, Japan.

Janikow, C. Z. & Michalewicz, Z. 1991, 'An experimental comparison of binary and floating point representations in genetic algorithms', *Fourth International Conference on Genetic Algorithms.*

Janson, C. & Larrson, L. 1996, 'A method for the optimization of ship hulls from a resistance point of view', *21st Symposium on Naval Hydrodynamics,* Trondheim, Norway.

Jin, R., Chen, W. & Simpson, T. W. 2003, 'Comparative studies of metamodeling techniques under multiple modeling criteria', *AIAA 2000.*

Jin, Y., Olhofer, M. & Sendhoff, B. 2001, 'Managing approximate models in evolutionary aerodynamic design optimization', *IEEE Congress on Evolutionary Computation,* Seoul, South Korea.

Jin, Y., Olhofer, M. & Sendhoff, B. 2002, 'A framework for evolutionary optimization with approximate fitness functions', *IEEE Transactions on Evolutionary Computation,* vol. 6, no. 5, pp. 481-494.

Johnson, M., Moore, L. & Ylvisaker, D. 1990, 'Minimax and maximin distance design', *Journal of Statistical Planning and Inference,* vol. 26, pp. 131-148.

Keane, A. J. 1996, *A brief comparison of some evolutionary optimization methods,* Department of Engineering Science, University of Oxford, Oxford.

Keane, A. J. & Nair, P. 2005, 'Elements of numerical optimization', in *Computational Approaches for Aerospace Design: The Pursuit of Excellence,* John Wiley & Sons, p. 602.

Keane, A. J., Price, W. G. & Schachter, R. D. 1991, 'Optimisation techniques in ship concept design', *RINA Transactions, Pt A,* vol. 133, pp. 123-144.

Kerwin, J. E. 1978, *A velocity prediction program for ocean racing yachts, revised to June 1978,* Report No.78-11, Massachusetts Institute of Technology.

Keuning, J. A., Gerritsma, J. & Van Terwigsa, P. F. 1993, 'Resistance test of a series of planing hull forms with a 30 degree deadrise angle, and a calculation model based on this and similar systematic series', *International Shipbuilding Progress,* vol. 40, no. 424, pp. 333-385.

Keuning, J. A. & Sonnenberg, U. B. 1999, 'Approximation of the calm water resistance on a sailing yacht based on the Delft systematic yacht hull series', *14th Chesapeake Sailing Yacht Symposium*, Annapolis.

Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. 1983, 'Optimization by simulated annealing', *Science*, vol. 220, no. 4598, pp. 671-680.

Korpus, R. 2004, 'Reynolds-Averaged Navier-Stokes in an integrated design environment.', *MDY 04*.

Korpus, R. 2007, 'Performance prediction without empiricism: a RANS-based VPP and design optimization capability', *18th Chesapeake Sailing Yacht Symposium*, Annapolis.

Koushan, K. 2003, *Artificial neural networks for prediction of ship resistance and wetted surface area*, Norway.

Koza, J. R. 1992, *Genetic programming: on the programming of computers by means of natural selection*, MIT Press, Cambridge, MA.

Kupras, L. K. 1976, 'Optimization method and parametric study in precontracted ship design', *International Shipbuilding Progress*, vol. 23, pp. 138-155.

Lackenby, H. 1950, 'On the systematic geometrical variation of ship forms', *Transactions of The Institute of Naval Architects*, vol. 92, pp. 289-315.

Lengyel, A. 2003, 'Ship design and product modelling using the NAPA System', *SCHIP en WERF de ZEE*, pp. 22-24.

Letcher, J. S. 1974, 'Handicapping rules and performance of sailing yachts', *1st Chesapeake Sailing Yacht Symposium*, Annapolis, MA.

Letcher, J. S., Marshall, J. K., Oliver, J. C. & Salvesen, N. 1987, 'Stars & Stripes', *Scientific American*, vol. 257, no. 2, pp. 34-40.

Lloyd, B. 1995, 'New Zealand races away from Conner for 3-0 lead in the Cup', *NY Times*, May 10, 1995.

Maisonneuve, J. J. 1989, *Résolution du problème de la résistance de vagues des navires par une méthode de singularité de Rankine*, Ph.D thesis, Ecole nationale supérieure de mécanique de l'université de Nantes, Nantes.

Maisonneuve, J. J. 1993, 'Optimization tools for ship resistance and seakeeping problems', *Fast '93, 2nd Intl Conf on Fast Sea Transportation*, Yokohama, Japan.

Maisonneuve, J. J. 2003, 'Application examples from industry: America's Cup hull shape optimization', *Optimistic*, Berlin.

Markov, N. E. & Suzuki, K. 2001, 'Hull form optimization by shift and deformation of ship sections', *Journal of Ship Research,* vol. 45, no. 3, pp. 197-204.

Mason, A. P., Couser, P., Von Konsky, B., Smith, C. & Mason, G. 2005, 'Optimisation of vessel resistance using genetic algorithms and artificial neural networks', *COMPIT 05*, Hamburg, Germany.

McKay, M. D., Conover, W. J. & Beckman, R. J. 1979, 'A comparison of three methods for selecting values of input variables in the analysis of output from a computer code', *Technometrics* vol. 21, no. 2, pp. 239-245.

McNaull, R. 1980, 'Generating new ship lines from a parent hull', *REAPS Technical Symposium, National Shipbuilding Research Program*, Philadelphia, Pennsylvania.

Menzel, S., Olhofer, M. & Sendhoff, B. 2005, 'Application of free form deformation techniques in evolutionary design optimisation', *6th World Congress on Structural and Multidisciplinary Optimization*, Rio de Janeiro.

Mesbahi, E. & Atlar, M. 2000, 'Artificial neural networks: applications in marine design and modelling', *COMPIT 2000*, Potsdam.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. 1953, 'Equation of state calculations by fast computing machines', *Journal of Chemical Physics* vol. 21, no. 6, pp. 1087-1092.

Michell, J. H. 1898, 'The wave resistance of a ship', *Philosophical Magazine of the Royal Society Ser.5,* vol. 45, pp. 106-123.

Milgram, J. H. 1993, 'Naval architecture technology used in winning the 1992 America's Cup match', *SNAME Transactions,* vol. 101, pp. 399-436.

Minami, Y. & Hinatsu, M., , 2002, 'Multi-objective optimization of ship hull form design by response surface methodology', *24th Symposium on Naval Hydrodynamics*, Fukuoka, Japan.

Moore, G. E. 1965, 'Cramming more components onto integrated circuits', *Electronics,* vol. 38, no. 8

Morgan, T. H. 1916, *A critique of the theory of evolution*, Princeton University Press.

Morishita, M. & Akagi, S. 2005, 'Hull form optimization of fast ships using simulated annealing (SA) method', *COMPIT 05*, Hamburg.

Morris, T. D. & Mitchell, T. J. 1995, 'Exploratory designs for computational experiments', *Journal of Statistical Planning and Inference,* vol. 43, no. 3, pp. 381-402.

Muhlenbein, H. & Schlierkamp-Voosen, D. 1993, 'Predictive models for the breeder genetic algorithm', *Evolutionary Computation,* vol. 1, no. 1

Nash, J. 1950, *Non-cooperative Games,* Ph.D. thesis, Princeton University.

Nelder, J. A. & Mead, R. 1965, 'A simplex algorithm for function minimization', *Computer Journal,*

Neocleous, C. C. & Schizas, C. N. 1995, 'Artificial neural networks in marine propeller design', *ICNN, 1995*

Neu, W. L., Hughes, O., Mason, W. H., Ni, S., Chen, Y., Ganesan, V., Lin, Z. & Tumma, S. 2000a, 'A prototype tool for multidisciplinary design optimization of ships', *Ninth Congress of the International Maritime Association of the Mediterranean,* Naples, Italy.

Neu, W. L., Mason, W. H., Ni, S., Lin, Z., Dasgupta, A. & Chen, Y. 2000b, 'A multidisciplinary design optimisation scheme for container ships', *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimisation,* Long Beach, CA.

Niederreiter, H. 1992, 'Monte Carlo and quasi-Monte Carlo methods in scientific computing', *CBMS-NSF,* vol. 63

Noblesse, F. 1983, 'A slender-ship theory of wave resistance', *Journal of Ship Research,* vol. 23, no. 1, pp. 13-33.

Notz, W. 2003, 'Experimental designs for computer experiments', *Experimental Design Workshop, Academia Sinica,* Taipei, Taiwan.

Nowacki, H., Brusis, F. & Swift, P. M. 1970, 'Tanker preliminary design-an optimization problem with constraints', *SNAME Annual Meeting.*

Nowacki, H., Creutz, G. & Munchmeyer, F. 1977, 'Ship lines creation by computer - objectives, methods and results', *First International Symposium on Computer-Aided Hull Surface Definition (SCAHD77),* Annapolis, MD.

Nowacki, H. & Kim, H. C. 2005, 'Form parameter based design of hull shapes as volume and surface object', *ICCAS 2005,* Busan, Korea.

Nowacki, H. A. 1983, 'Design and fairing of ship surfaces', in *Surfaces in Computer Aided Geometric Design,* ed. R. a. B. W. Eds. Barnhill, North Holland, pp. 121-134.

Nowacki, H. A. 1993, 'Hull form variation and evaluation', *Journal of the Kansai Society of Naval Architects*, no. 219 pp. 173-184

Ockam 2006, *OckamU manual*, Ockam Instruments.

Oei, C. K., Goldberg, D. E. & Chang, S. J. 1991, *Tournament selection, niching, and the preservation of diversity*, 91011, University of Illinois at Urbana-Champaign, Urbana, Illinois.

Oliver, J. C. & Claughton, A. R. 1995, 'Development of multi-functional velocity prediction program for sailing', *Int. Conf. on Computer Aided Design and Production for Small Craft, CADAP '95*, Southampton.

Oliver, J. C., Letcher, J. S. & Salvesen, N. 1987, 'Performance prediction for Stars & Stripes', *SNAME Transactions*, vol. 95, pp. 239-261.

Ong, Y. S., Nair, P. B. & Keane, A. J. 2003, 'Evolutionary optimization of computationally expensive problems via surrogate modeling', *AIAA Journal*, vol. 41, no. 4, pp. 687-696.

Oortmerssen, G. 1971, 'A power prediction method and its application to small ships', *International Shipbuilding Progress*, vol. 18, no. 207

Panait, L. & Liuke, S. 2002, 'A comparison of two competitive fitness functions', *GECCO 2002*.

Park, J.-S. 1994, 'Optimal Latin-hypercube designs for computer experiments', *Journal of Statistical Planning and Inference*, vol. 39, no. 1, pp. 95-111.

Parolini, N. & Quarteroni, A. 2007, *Modelling and numerical simulation for yacht engineering*, MOX-Report No.10/2007, Department of Mathematics, Politechnico di Milano, Milan.

Parsons, M. G. 1975, 'Optimization methods for use in computer-aided ship design', *First Ship Technology and Research (STAR) Symposium*, Washington D.C.

Peacock, D., Smith, W. F. & Pal, P. K. 1996, *Hull-form generation using multi-objective optimisation techniques*, AME CRC C 96/11, Australian Maritime Engineering CRC.

Percival, S., Hendrix, D. & Noblesse, F. 2001, 'Hydrodynamic optimization of ship hull forms', *Applied Ocean Research*, vol. 23, no. 6, pp. 337-355.

Peri, D. & Campana, E. F. 2005a, 'Global optimization for safety and comfort', *COMPIT 05*, Hamburg.

Peri, D. & Campana, E. F. 2005b, 'High-fidelity models and multiobjective global optimisation algorithms in simulation-based design', *Journal of Ship Research,* vol. 49, no. 3, pp. 159-175.

Peri, D. & Campana, E. F. 2005c, 'High fidelity models in global optimization', *COCOS 2003*, Berlin / Heidelberg.

Peri, D. & Campana, E. F. 2005d, 'High fidelity models in simulation-based design',

Peri, D., Campana, E. F. & Di Mascio, A. 2001, 'Development of CFD-based design optimization architecture', *First MIT Conference on Computational Fluid and Solid Mechanics.*

Peri, D. & Mandolesi, F. 2005, 'Multiobjective design optimization of an IACC sailing yacht by means of CFD high fidelity solvers', *17th Chesapeake Sailing Yacht Symposium*, Annapolis.

Philpott, A. B. 2003, 'Stochastic optimization in yacht racing', in *Applications of Stochastic Programming,* eds. W. Ziemba & S. Wallace, Springer-Verlag.

Philpott, A. B., Henderson, S. G. & Teirney, D. P. 2004, 'A simulation model for predicting yacht match race outcomes', *Operations Research,* vol. 52, no. 1, pp. 1-16.

Piegl, L. & Tiller, W. 1997, *The NURBS book*, Springer-Verlag, New York.

Pierret, S. 1999, 'Turbomachinery blade design using a Navier-Stokes solver and artificial neural network', *ASME Journal of Turbomachinery,* vol. 121, no. 3, pp. 326-332.

Pinto, A. & Campana, E. F. 2005, 'A multi-swarm algorithm for multi-objective ship design problems', *COMPIT 05*, Hamburg.

Pinto, A., Peri, D. & Campana, E. F. 2004, 'Global optimization algorithms in naval hydrodynamics', *COMPIT 04*, Siguenza, Spain.

Pinto, A., Peri, D. & Campana, E. F. 2007, 'Multiobjective optimization of a containership using deterministic particle swarm optimization', *Journal of Ship Research,* vol. 51, no. 3, pp. 217-228.

Poloni, C., Giurgevich, A., Onesti, L. & Pediroda, V. 1999, 'Hybridisation of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics', Poloni, C. & Pediroda, V. 1997, 'GA coupled with computationally expensive simulations: tools to improve efficiency', in *Genetic algorithms in Engineering and Computer Science,* eds. J. Periaux, D. Quagliarella, C. Poloni & G. Winter, John Wiley & Sons.

Powell, M. J. D. 1987, 'Radial basis functions for multivariable interpolation: a review', in *Algorithms for Approximation,* eds. J. C. Mason & M. G. Cox, Oxford University Press, London.

Quagliarella, D. & DellaCioppa, A. 1994, 'Genetic algorithms applied to the aerodynamic design of transonic airfoils', *12th AIAA Applied Aerodynamics Conference,* Colorado Springs, CO.

Queipo, N., Haftka, R., Shyy, W., Goel, T., Vaidyanathan, R. & Tucker, K. 2005, 'Surrogate-based analysis and optimization', *Progress in Aerospace Sciences,* vol. 41, no. 1, pp. 1-28.

Rechenberg, I. 1973, *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution,* Frommann Holzberg Verlag, Stuttgart.

Reenskaug, T. M. H. 2003, 'Applications and technologies for maritime and offshore industries', *History of Nordic Computing Conference,* Trondheim.

Reichel, J., Pugh, J., Rosen, B. S. & Debord, F. 1994, 'Grand prix yacht design with the aid of computational and experimental techniques', *Yacht Vision '94 Symposium,* Auckland, New Zealand.

Rogers, D., F. 1977, 'B-spline curves and surfaces for ship hull design', *First International Symposium on Computer-Aided Hull Surface Definition (SCAHD77),* Annapolis, Maryland.

Rosen, B. S., Laiosa, J. P. & Davis, W. 2000, 'CFD studies for America's Cup 2000', *AIAA-2000-4339,*

Rosen, B. S., Laiosa, J. P., Davis, W. H., Jr. & Stavetski, D. 1993, 'SPLASH free-surface flow code methodology for hydrodynamic design and analysis of IACC yachts', *11th Chesapeake Sailing Yacht Symposium,* Annapolis.

Rosin, C. & Belew, R. 1996, 'New methods for competitive coevolution', *Evolutionary Computation,* vol. 5, no. 1, pp. 1-29.

Rudolph, G. 1994, 'Convergence analysis of canonical genetic algorithms', *IEEE Transactions on Neural Networks,* vol. 5, pp. 96-101.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. 1986, 'Learning internal representations by error propagation', in *Parallel Distributed Processing: Explorations in the Microstructures of Cognition,* MIT Press, Cambridge, MA, pp. 318-362.

Sacks, J., Welch, W. J., Mitchell, T. J. & Wynn, H. P. 1989, 'Design and analysis of computer experiments ', *Statistical Science,* vol. 4, no. 4, pp. 409-423.

Sahoo, P. 1997, 'Determination of hull form by use of non-linear optimization methods', *IV International Symposium on High-Speed Marine Vehicles*, Naples, Italy.

Salvesen, N., Tuck, E. O. & Faltinsen, O. 1970, 'Ship motions and sea loads', *Trans. SNAME,* vol. 78, pp. 250-287.

Sánchez, P. J. 2006, 'As simple as possible, but no simpler: A gentle introduction to simulation modeling', *Winter Simulation Conference 2006,* Monterey, California.

Sarioz, K., Hearn, G. E. & Hills, B. 1992, 'Practical seakeeping for design: an optimised approach', *PRADS '92, 5th Intl. Symp. on the Practical Design of Ships and Mobile Units,* Newcastle upon Tyne, U.K.

Sarle, W. S. 1994, 'Neural networks and statistical models', *Nineteenth Annual SAS Users Group International Conference,* Cary, NC.

Savitsky, D. 1964, 'Hydrodynamic design of planing hulls', *Marine Technology,*

Schaffer, J. D., Caruana, R. A., Eshelman, L. J. & Das, R. 1989, 'A study of control parameters affecting online performance of genetic algorithms for function optimization', *Third International Conference on Genetic Algorithms,* George Mason University, United States.

Schlageter, E. C. & Teeters, J. R. 1993, 'Performance prediction software for IACC yachts', *11th Chesapeake Sailing Yacht Symposium,* Annapolis, MD.

Schwefel, H.-P. 1974, *Adaptive mechanismen in der biologischen evolution und ihr einfluß auf die evolutionsgeschwindigkeit,* Abschluflbericht zum DFG-Vorhaben Re 215/2, Technical University of Berlin.

Sederberg, T. W. & Parry, S. R. 1986, 'Free-form deformation of solid geometric models', *ACM Siggraph Computer Graphics,* vol. 20, no. 4, pp. 151-160.

Seif, M. S. & Jahanbakhsh, E. 2004, 'Neural networks model for ship maneouver', *COMPIT 04,* Siguenza, Spain.

Sen, P. & Todd, D. S. 1997, 'Multiple criteria genetic algorithms: a catamaran design study', *7th International Conference on Genetic Algorithms*, East Lansing, MI.

Simpson, T. W., Korte, J. J., Mauery, T. M. & Mistree, F. 1998, *Comparison of response surface and kriging models for multidisciplinary design optimization*, 4755, AIAA.

Simpson, T. W., Lin, D. K. J. & Chen, W. 2000, 'Sampling strategies for computer experiments: design and analysis', *International Journal of Reliability and Applications*,

Simpson, T. W., Peplinski, J. D., Koch, P. N. & Allen, J. K. 2001, 'Metamodels for computer-based engineering design: survey and recommendations', in *Engineering with Computers*, Springer-Verlag, pp. 129-150

Sobol, I. M. 1967, 'On the distribution of points in a cube and the approximate evaluation of integrals', *U.S.S.R. Computational Mathematics and Mathematics Physics*, vol. 7, pp. 86-112.

Söding, H. & Rabien, U. 1977, 'Hull surface design by modifying an existing hull', *First International Symposium on Computer-Aided Hull Surface Definition (SCAHD77)*, Annapolis, MD.

Tan, C., Ray, T. & Tsai, H. 2003, 'A comparative study of evolutionary algorithm and swarm algorithm for airfoil shape optimization problems', *41st Aerospace Sciences Meeting and Exhibit*, Reno, Nevada.

Tang, B. 1993, 'Orthogonal array based Latin hypercube sampling', *Journal of American Statistical Association*, pp. 1392-1397.

Teeters, J. 2004, *IMS performance package*, U.S. Sailing - Offshore Racing Council.

Teirney, D. P. 1998, 'Yacht match race simulation', *33rd Annual Conference of the ORSNZ*, University of Canterbury

Thompson, D. 1917, *On Growth and Form*, Cambridge University Press.

Valdenazzi, F., Harries, S., Viviani, U. & Abt, C. 2002, 'Seakeeping optimisation of fast vessels by means of parametric modelling', *HSMV 02*, Napoli, Italy.

Valdenazzi, F., Pittaluga, C., Harries, S., Abt, C. & Avellino, G. 2003, 'Hydrodynamic design of the aftbody shape of a roro vessel', *NAV 2003*, Palermo.

van Oossanen, P. 1985, 'The development of the 12 Meter class yacht "Australia II"', *7th Chesapeake Sailing Yacht Symposium*.

van Oossanen, P. 1993, 'Predicting the speed of sailing yachts', *SNAME Transactions,* vol. 101, pp. 337-397.

Volker, H. 1954, 'Systematic variation of usual ship form', *International Shipbuilding Progress,* vol. 1, no. 1

Walden, D. A., Kopp, P. J. & Grundmann, P. 1985, 'Optimization of hull form for seakeeping and resistance', *Workshop on Developments in Hull Form Design,* Wageningen, Netherlands.

Whitfield, R. I., Hills, B. & Coates, G. 1999, 'The application of multi-objective robust design methods in ship design', *10th International Conference on Computer Applications in Ship Building, (ICCAS'99),* Cambridge, USA.

Whitfield, R. I., Wright, P. N. H., Coates, G. & Hills, W. 1998, 'A robust design methodology suitable for application to one-off products', *Journal of Engineering Design,* vol. 9, no. 4, pp. 373-387.

Wright, A. H. 1991, 'Genetic algorithms for real parameter optimization', in *Foundations of Genetic Algorithms,* Morgan Kaufmann, pp. 205-218.

Yamamoto, K. & Inoue, O. 1995a, 'Applications of genetic algorithm to aerodynamic shape optimization', *12th AIAA Computational Fluid Dynamics Conference,* San Diego.

Yamamoto, K. & Inoue, O. 1995b, 'New evolutionary direction operator for genetic algorithms', *AIAA Journal,* vol. 33, no. 10, pp. 1990-1992.

Yang, C., Noblesse, F. & Lohner, R. 2001, 'Practical hydrodynamic optimization of a trimaran', *SNAME Annual Meeting,* Orlando, FL.

Yang, C., Noblesse, F., Lohner, R. & Hendrix, D. 2000, 'Practical CFD applications to design of a wave cancellation multihull ship', *23rd Symposium on Naval Hydrodynamics,* Val De Reuil, France.

Yasukawa, H. 2000, 'Ship form improvement using genetic algorithm', *Ship Technology Research,* vol. 47, no. 1, pp. 35-44.

Zhou, Z., Ong, Y. S., Nair, P. B., Keane, A. J. & Lum, K. Y. 2007, 'Combining global and local surrogate models to accelerate evolutionary optimization', *IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews,* vol. 37, no. 1, pp. 66-76.

Zitzler, E. 2002, 'Evolutionary algorithms for multiobjective optimization', *Evolutionary Methods for Design, Optimisation and Control,* Barcelona, Spain

APPENDIX 1 – XML METAMODEL FILE STRUCTURE

```xml
<?xml version="1.0"  encoding="UTF-8" standalone="yes" ?>
<neuro_intelligence_network>
  <version><![CDATA[2.2.1]]></version>
  <network_configuration>
    <class>MLP</class>
    <inputs_count>9</inputs_count>
    <outputs_count>1</outputs_count>
    <layers_count>2</layers_count>
    <layer id="0">
      <neurons_count>19</neurons_count>
      <weights node="0">
        2.27389 | -0.0168789 | 0.0142284 | 0.0331169 | -0.0484834 | 0.0104526 | 0.906408 | 0.219396 | 3.28865 | -2.51538
      </weights>
      <weights node="1">
        12.0604 | -0.269754 | -0.0305208 | 0.0311745 | -0.0949391 | 0.00752494 | -0.501986 | -14.4025 | -1.22255 | 1.49662
      </weights>
      <weights node="2">
        -1.93846 | 0.07759 | -0.0100953 | -0.0964149 | -0.0259891 | 0.00120998 | 4.70502 | 0.257896 | -0.217008 | 0.715359
      </weights>
      <weights node="3">
        -3.84573 | 0.114015 | -0.00703897 | -0.0788651 | -0.00435289 | 0.00973844 | 3.15229 | 0.541173 | -0.27552 | -1.10279
      </weights>
      <weights node="4">
        -2.07257 | 0.0718469 | -0.0173137 | -0.115161 | -0.0335178 | -0.00383545 | 5.61589 | 0.266996 | -0.22866 | 0.646215
      </weights>
      <weights node="5">
        -0.0566205 | -0.105767 | 0.0235986 | 0.112012 | -0.016449 | -0.00774378 | -4.46584 | -0.270161 | 1.13415 | 1.88941
      </weights>
      <weights node="6">
        -1.82694 | 0.0290092 | -0.00976123 | -0.0137207 | 0.0475483 | -0.00959223 | -0.994516 | 0.113821 | -2.02477 | 0.05254
      </weights>
      <weights node="7">
        1.41216 | 0.00134032 | -0.274229 | 0.190854 | 0.586464 | -0.0157984 | -8.14977 | -0.153494 | 12.405 | -1.56189
      </weights>
      <weights node="8">
        0.298625 | -0.102296 | 0.0207433 | 0.035168 | -0.06466 | 0.0112783 | 0.75938 | -0.284146 | 3.57872 | -1.69784
      </weights>
      <weights node="9">
        -3.55835 | -0.0552117 | 0.0990875 | 0.0835501 | -0.209655 | -0.032842 | 0.454899 | 0.473587 | -18.4522 | -4.27917
      </weights>
      <weights node="10">
        2.25877 | 0.000409491 | 0.0108662 | 0.0318707 | -0.0427584 | 0.00963604 | 0.893062 | 0.320039 | 3.3255 | -2.33401
      </weights>
      <weights node="11">
        -1.00109 | 0.00513186 | -0.00322394 | 0.00950723 | 0.0142456 | 0.000582969 | -0.79681 | 0.660352 | -0.0496002 | -2.33824
      </weights>
      <weights node="12">
        12.6335 | -0.293333 | -0.0309483 | 0.0333139 | -0.103731 | 0.00706916 | -0.637784 | -15.2246 | -1.33688 | 1.50721
      </weights>
      <weights node="13">
        -4.1489 | -0.0192307 | 0.114977 | -0.139513 | -0.283196 | 0.011824 | 3.77891 | 0.10306 | -7.485 | 0.53306
      </weights>
      <weights node="14">
        0.307281 | 0.0847232 | -0.0203851 | -0.0934331 | 0.0150231 | 0.00612299 | 3.78207 | 0.228549 | -1.08468 | -1.61713
      </weights>
      <weights node="15">
        2.28838 | -0.022736 | 0.00679325 | 0.0118182 | -0.035702 | 0.00748383 | 0.724671 | -0.238144 | 1.47745 | -0.927311
      </weights>
      <weights node="16">
        -3.62682 | -0.0498745 | 0.093509 | 0.0750634 | -0.204819 | -0.0315754 | 0.413529 | 0.432028 | -18.4232 | -4.28831
      </weights>
      <weights node="17">
        10.6739 | -0.0998994 | 0.0147318 | 0.120914 | 0.00490192 | -0.0089116 | -4.45702 | -0.291712 | 1.12601 | -7.82569
      </weights>
      <weights node="18">
        -3.99828 | -0.0196679 | 0.111488 | -0.133346 | -0.275168 | 0.0106297 | 3.61801 | 0.10245 | -7.35598 | 0.493428
      </weights>
      <activation_function>Logistic</activation_function>

    </layer>
    <layer id="1">
      <neurons_count>1</neurons_count>
      <weights node="0">
        21.4273 | -3.38622 | 6.8474 | -4.03755 | -3.65813 | 13.3049 | -11.5235 | -0.191425 | -3.37995 | -3.64918 | -16.0135 |
9.99481 | 3.07343 | 8.81703 | 16.9704 | -16.2387 | 3.72837 | 4.79823 | -9.11998 | -5.50762
      </weights>
      <activation_function>Logistic</activation_function>
    </layer>
    <input_range>
      <min>-1</min>
      <max>1</max>
    </input_range>
    <output_range>
      <min>0</min>
      <max>1</max>
    </output_range>
  </network_configuration>

  <dataset_configuration>
    <columns>
      <count>20</count>
      <column id="0">
        <header><![CDATA[Upwind]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Input</type>
        <scaling_params>
          <min>-0.25</min>
          <max>1.25</max>
        </scaling_params>
      </column>

      <column id="2">
        <header><![CDATA[BWL]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Input</type>
        <scaling_params>
          <min>2.77481</min>
          <max>3.52508</max>
        </scaling_params>
      </column>

      <column id="3">
        <header><![CDATA[Cp]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Input</type>
        <scaling_params>
          <min>0.569679</min>
          <max>0.599817</max>
        </scaling_params>
      </column>
```

```
<column id="4">
  <header><![CDATA[LCB]]></header>
  <enabled>1</enabled>
  <format>Numeric</format>
  <type>Input</type>
  <scaling_params>
    <min>0.516012</min>
    <max>0.540029</max>
  </scaling_params>
</column>

<column id="5">
  <header><![CDATA[Cm]]></header>
  <enabled>1</enabled>
  <format>Numeric</format>
  <type>Input</type>
  <scaling_params>
    <min>0.777338</min>
    <max>0.912831</max>
  </scaling_params>
</column>

<column id="6">
  <header><![CDATA[Flare]]></header>
  <enabled>1</enabled>
  <format>Numeric</format>
  <type>Input</type>
  <scaling_params>
    <min>-1</min>
    <max>5</max>
  </scaling_params>
</column>

<column id="7">
  <header><![CDATA[HEEL]]></header>
  <enabled>1</enabled>
  <format>Numeric</format>
  <type>Input</type>
  <scaling_params>
    <min>-8.75</min>
    <max>43.75</max>
  </scaling_params>
</column>
```

```
      <column id="8">
        <header><![CDATA[YAW]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Input</type>
        <scaling_params>
          <min>-0.8125</min>
          <max>4.0625</max>
        </scaling_params>
      </column>

      <column id="9">
        <header><![CDATA[VEL]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Input</type>
        <scaling_params>
          <min>1.25</min>
          <max>17.75</max>
        </scaling_params>
      </column>

      <column id="16">
        <header><![CDATA[LiftArea]]></header>
        <enabled>1</enabled>
        <format>Numeric</format>
        <type>Output</type>
        <scaling_params>
          <min>-63.5154</min>
          <max>317.577</max>
        </scaling_params>
      </column>
    </columns>
  </dataset_configuration>
</neuro_intelligence_network>
```

APPENDIX 2 – NEURAL NETWORK CLASS STRUCTURE

```
#include <vector>
/**************************************************************/
//**  This code allows the construction and initialisation of
//**  neural nets of different sizes. Nets are fully connected
//**  with a variable number of hidden layers.
/**************************************************************/
const   intMaxHiddenLayers = 5;
const   intMaxNeuronsInLayer = 1000;
extern  int RandomSeed;
typedef std::vector<double> TDoubleArray0;
typedef enum {
             BinarySigmoidal,
             BipolarSigmoidal,
             HyperbolicTan,
             Linear,
             } ActivationKind;
/**************************************************************/
//** Base class for all Neurons in the network
/**************************************************************/
class Neuron
  {
  public:
    Neuron  (double Bias=0.0, double Output=0.0, double ErrorInformationTerm=0.0,
            double BiasCorrectionTerm=0.0, double SumOfWeightedInputs=0.0);
    void    UpdateWeightsAndBiases();
    void    CalculateOutput(ActivationKind ActivationFunction);
    double  CalculateOutputDerivative(ActivationKind ActivationFunction);
    void    CalculateWeightAndBiasCorrectionTerms(double LearningRate);
    void    EstablishArrayOfInputs(int PreviousLayerNeuronCount);
    void    EstablishWeightVectors();

    int     NumberOfInputNeurons;
    double  Bias;
    double  Output;
    double  ErrorInformationTerm;
    double  BiasCorrectionTerm;
    double  SumOfWeightedInputs;
    TDoubleArray0 NeuronInput;
    TDoubleArray0 WeightOfInputs;
  private:
    TDoubleArray0 WeightCorrectionTerm;
  };
/*************************************************************/
//**  Declare a derived class "HiddenNeurons" for neurons in
//**  the hidden layers of the network
/*************************************************************/
class HiddenNeurons : public Neuron
  {
  public:
    void    CalculateHiddenErrorInformationTerm(ActivationKind ActivationFunction);
  };



/*************************************************************/
//**  Declare a derived class "OutputNeurons" for neurons in
//**  the output layer of network
/*************************************************************/
class OutputNeurons : public Neuron
  {
  public:
    void    CalculateOutputErrorInformationTerm(double TargetValue, ActivationKind ActivationFunction);
    double  AbsoluteErrorDifference;
    double  ErrorDifferenceSquared;
  };



typedef std::vector<HiddenNeurons> HiddenNeuronsArray;

/*************************************************************/
//**  There may be more than one hidden layer, so create an array
//**  and a count
/*************************************************************/
class HiddenLayer
  {
  public:
    HiddenLayer();
    HiddenNeuronsArray  HLNeurons;
    ActivationKind  ActivationFunctionForHiddenLayer;
  };
```

```
typedef std::vector<HiddenLayer> HiddenLayerArray;
typedef std::vector<OutputNeurons> OutputNeuronsArray;

/*************************************************************/
//**  The following class represents an artificial neural network containing
//**  the topology, weights, training performance and testing performance.
//**  This network does not have a training mode associated with it, it is simply
//**  a generic structure that can use backpropagation or a GA for training
/*************************************************************/
class NeuralNetworkTopology
  {
  public:
    NeuralNetworkTopology();
    ~NeuralNetworkTopology(){};
    void    EstablishHiddenLayerActivationFunctions(int WhichLayer,ActivationKind HiddenAct);
    void    EstablishOutputActivationFunctions(ActivationKind OutputAct);
    void    ConstructAndInitializeNetwork(void);
    void    SetNumberOfInputLayerNeurons(int ILcount);
    void    SetNumberOfHiddenLayers(int HLcount);
    void    SetNumberOfHiddenLayerNeurons(int TheHiddenLayer,int HLcount);
    void    SetNumberOfOutputLayerNeurons(int OLcount);
    double  TransformInput(int WhichInputNeuron,double TheInputValue);
    double  TransformOutput(int WhichOutputNeuron,double TheOutputValue);

/*************************************************************/
//**  The following three calls are the core of the system
//**  Call SetInput to set each of the input values
//**  Call RunNeuralNetwork to calculate all network values
//**  Call GetOutput to retrieve the output value
/*************************************************************/
    void    SetInput(int WhichInputNeuron,double TheInputValue);
    void    RunNeuralNetwork();
    double  GetOutput(int WhichOutputNeuron);

    void    SetInputRangeMax (double inMax){ mMaxInput = inMax;}
    void    SetInputRangeMin (double inMin){ mMinInput = inMin;}
    void    SetOutputRangeMax(double outMax){ mMaxOutput = outMax;}
    void    SetOutputRangeMin(double outMin){ mMinOutput = outMin;}

    void    SetMaxInputParameterVal(int WhichInputNeuron,double inMax){ mMaxInputVal[WhichInputNeuron] = inMax;}
    void    SetMinInputParameterVal(int WhichInputNeuron,double inMin){ mMinInputVal[WhichInputNeuron] = inMin;}
    void    SetMaxOutputParameterVal(int WhichOutPutNeuron,double outMax){ mMaxOutputVal[WhichOutPutNeuron] = outMax;}
    void    SetMinOutputParameterVal(int WhichOutPutNeuron,double outMin){ mMinOutputVal[WhichOutPutNeuron] = outMin;}
    double  GetInputRangeMax (void) const { return mMaxInput;}
    double  GetInputRangeMin (void) const { return mMinInput;}
    double  GetOutputRangeMax(void) const { return mMaxOutput;}
    double  GetOutputRangeMin(void) const { return mMinOutput;}

    int  GetNumberOfHiddenLayers() const;
    int  GetHiddenLayerSize(int WhichLayer) const;
    int  GetILNeuronCount() const;
    int  GetOLNeuronCount() const;

    HiddenLayerArray HiddenLayers;
    OutputNeuronsArray OutputLayer;

  private:
    TDoubleArray0 mMaxInputVal;
    TDoubleArray0 mMinInputVal;
    TDoubleArray0 mMaxOutputVal;
    TDoubleArray0 mMinOutputVal;

    ActivationKind ActivationFunctionForOutputLayer;
    TDoubleArray0 InputValues;
    double mMaxInput, mMinInput, mMaxOutput, mMinOutput; // for scaling data
  };

double NNRandom(int& Seed);
```

APPENDIX 3 – VPP DLL INTERFACE

File BridgetoDLL.h

```
#ifndef _BRIDGETODLL_
#define _BRIDGETODLL_
double VPPSetInputData(double *values, int len) ;
double GetUpwindVMG(double Windspeed);
double GetDownwindVMG(double Windspeed);
double GetUpwindTWA(double Windspeed,double TrueWindAngle);
double GetDownwindTWA(double Windspeed,double TrueWindAngle);
#endif
```

File BridgetoDLL.cpp

```
#ifndef _BRIDGETODLL_
#include "BridgeToDLL.h"
#endif

#include <stdio.h>
#using <mscorlib.dll>
#using <PAP.dll>
#include "stdafx.h"
using namespace System;
using namespace PAPns;
#pragma managed

double VPPSetInputData(double *values, int len)
  {
  int  count;
  Array __gc *DesignParameters;
  PAPClass __gc *DotNetObject;

  DesignParameters = Array::CreateInstance(__typeof(System::Double),len);
  count = DesignParameters->GetLength(0);

  for (long i=0;i < count;i++)
    DesignParameters->SetValue(__box(values[i]), i);
  DotNetObject = new PAPClass;
  return DotNetObject->PAPSetInputData(DesignParameters);
  }
```

```
double GetUpwindVMG(double Windspeed)
  {
  PAPClass __gc *DotNetObject;

  DotNetObject = new PAPClass;
  return DotNetObject->PAPGetUpwindVMG(Windspeed);
  }

double GetDownwindVMG(double Windspeed)
  {
  PAPClass __gc *DotNetObject;

  DotNetObject = new PAPClass;
  return DotNetObject->PAPGetDownwindVMG(Windspeed);
  }

double GetUpwindTWA(double Windspeed,double TrueWindAngle)
  {
  PAPClass __gc *DotNetObject;
  double TempVelocity;
  DotNetObject = new PAPClass;
  TempVelocity = DotNetObject->PAPGetUpwindTWA(Windspeed,TrueWindAngle);
  return TempVelocity;
  }

double GetDownwindTWA(double Windspeed,double TrueWindAngle)
  {
  PAPClass __gc *DotNetObject;

  DotNetObject = new PAPClass;
  return DotNetObject->PAPGetDownwindTWA(Windspeed,TrueWindAngle);
  }

#pragma unmanaged
```

APPENDIX 4 – RACE TIME CALCULATION

```
TotalRaceTime = 0;
for(long TheRaceIndex=1;TheRaceIndex<=RaceCount;TheRaceIndex++)
  {
  TheRacecourseTime[TheRaceIndex] = 0;
  for(long TheLegIndex=1;TheLegIndex<=LegCount;TheLegIndex++)
    {
    TheLegTime[TheRaceIndex][TheLegIndex]=0;
    IsUpwindLeg = odd(TheLegIndex);
    for(long TheStepIndex=1;TheStepIndex<=LegStepCount;TheStepIndex++)
      {
      TheStepWindSpeed = TheRacecourses[TheRaceIndex].TheLegSteps[TheLegIndex][TheStepIndex].StepWindSpeed;
      TheStepWindDirection = TheRacecourses[TheRaceIndex].TheLegSteps[TheLegIndex][TheStepIndex].StepWindDirection;
      TheStepWallyingPerc = TheRacecourses[TheRaceIndex].TheLegSteps[TheLegIndex][TheStepIndex].StepWallyingPerc;
      TheStepLength = TheRacecourses[TheRaceIndex].TheLegSteps[TheLegIndex][TheStepIndex].StepLength;
      if (IsUpwindLeg)
        TheVMG = UpwindVMG[TheStepWindSpeed].GetVMC(vpp,TheStepWindDirection, TheStepWallyingPerc);
      else
        TheVMG = DownwindVMG[TheStepWindSpeed].GetVMC(vpp,TheStepWindDirection, TheStepWallyingPerc);

      StepTime = TheStepLength/TheVMG*3600; // convert to seconds
      TheLegTime[TheRaceIndex][TheLegIndex] = TheLegTime[TheRaceIndex][TheLegIndex] + StepTime;
      }
    TheRacecourseTime[TheRaceIndex] = TheRacecourseTime[TheRaceIndex] + TheLegTime[TheRaceIndex][TheLegIndex];
    }
  TotalRaceTime = TotalRaceTime + TheRacecourseTime[TheRaceIndex];
  }
```

APPENDIX 5 – TOURNAMENT MODELLING

```
void TVespaApp::OnSolveGAOptimise()
  {
  if (OptimisationDialog())
    {
    TrailingBoatPenalties.InitMarkPenalties();
    TheVPPEvolver.SetGenerationCount(VPPGenCount);
    TheVPPEvolver.SetPopulationSize(VPPpopulation);
    TheVPPEvolver.SetExemplarElitism(true);
    TheVPPEvolver.ClearPopulation();
    TheACRule.SetRuleVersion(5);

    if (DoRaceSetup)
      {
      TheRacecourseProfiles.InitialiseWindSpeeds();
      TheRacecourseProfiles.InitialiseRacecourseSteps();
      for(long LuckIndex=1;LuckIndex<=LuckIndexLimit;LuckIndex++)
        for(long TheLegIndex=1;TheLegIndex<=LegCount;TheLegIndex++)
          if (LuckIndex==1)
            LuckPerterbation[LuckIndex][TheLegIndex]= 0;  // no random fluctuation in race 1, primarily for debugging purposes
          else
            LuckPerterbation[LuckIndex][TheLegIndex]= Random.GuassianRandomDouble (0,6);  // random fluctuation with SD of 6s
      }
    TheVPPEvolver.EvolvePopulation();
    }
  }
```

```
void TVPPEvolver::EvaluatePopulationCallback()
  {
  double WinCount;
  double TotalRaceCount;
  double RaceTimeA,RaceTimeB,Boatlength=7;
  double TheMarkPenalty = 0,PenaltyWeighting=0;
  double LegTimeA,LegTimeB;
  Boolean RacerAIsStarboardTacker,IsUpwindLeg;
  Boolean iIsExemplar,jIsExemplar;

  for(long i=1;i<= mPopulationSize;i++)
    {
    WinCount=0;
    TotalRaceCount = 0;
    iIsExemplar = i > mPopulationSize - ExemplarCount;
    for(long j=1;j<= mPopulationSize;j++)
      if(i!=j)
        {
        jIsExemplar = j > mPopulationSize - ExemplarCount;
        if (!DoMaintainExemplars || jIsExemplar || (!jIsExemplar && !OnlyExemplarsInObjFunc))
          for(long LuckIndex=1;LuckIndex<=LuckIndexLimit;LuckIndex++)
            for(long TheRaceIndex=1;TheRaceIndex<=RaceCount;TheRaceIndex++)
              {
              RaceTimeA = 0;
              RaceTimeB = 0;
              TotalRaceCount++;
              for(long TheLegIndex=1;TheLegIndex<=LegCount;TheLegIndex++)
                {
                LegTimeA = mPopulation[i].TheLegTime[TheRaceIndex][TheLegIndex];
                LegTimeB = mPopulation[j].TheLegTime[TheRaceIndex][TheLegIndex];
                if (LuckPerterbation[LuckIndex][TheLegIndex]<0)
                  {
                  RaceTimeA = RaceTimeA + LegTimeA + fabs(LuckPerterbation[LuckIndex][TheLegIndex]);
                  RaceTimeB = RaceTimeB + LegTimeB;
                  }
                else
                  {
                  RaceTimeA = RaceTimeA + LegTimeA;
                  RaceTimeB = RaceTimeB + LegTimeB + fabs(LuckPerterbation[LuckIndex][TheLegIndex]);
                  }

                IsUpwindLeg = odd(TheLegIndex);
                RacerAIsStarboardTacker = odd(LuckIndex); // alternate port and starboard benefit
```

```
                    if (DoStarboardTackAdvantage && IsUpwindLeg)//only apply starboard tack advantage on upwind legs
                      {
                      if (RacerAIsStarboardTacker)
                        {
                        if ((RaceTimeB > RaceTimeA - Boatlength)&&(RaceTimeB < RaceTimeA + Boatlength)) // BoatB is overlapped
                            RaceTimeB = RaceTimeA + Boatlength; // BoatB is forced to cross behind BoatA
                        }
                      else // BoatB is Starboard tacker
                        if ((RaceTimeA > RaceTimeB - Boatlength)&&(RaceTimeA < RaceTimeB + Boatlength)) // BoatA is overlapped
                            RaceTimeA = RaceTimeB + Boatlength; // BoatA is forced to cross behind BoatB
                      }
                    double ThePenalty=TrailingBoatPenalties.TheMarkPenalties[TheLegIndex];
                    double ThePenaltyRange = 20;
                    if (ThePenalty != 0)
                      {
                      PenaltyWeighting = ThePenaltyRange - fabs(RaceTimeA - RaceTimeB);
                      PenaltyWeighting = min(max(PenaltyWeighting,0.0),ThePenaltyRange)/ThePenaltyRange;
                      TheMarkPenalty = fabs(PenaltyWeighting * ThePenalty);
                      if (RaceTimeA > RaceTimeB)
                        RaceTimeA = RaceTimeA + TheMarkPenalty;
                      else
                        if (RaceTimeB > RaceTimeA)
                          RaceTimeB = RaceTimeB + TheMarkPenalty;
                      }
                    }
                  if (RaceTimeA == RaceTimeB) // in the unlikely event of a draw
                    WinCount=WinCount+0.5;
                  else
                    if (RaceTimeA < RaceTimeB)
                      WinCount++;
                  }
            }
      if (TotalRaceCount > 0)
        mPopulation[i].SetEvaluationValues(1,1.0-WinCount/TotalRaceCount); // the GA is set to minimize so do 1.0 - win ratio
      else
        mPopulation[i].SetEvaluationValues(1,1.0); // this hull not included in obj function, so win/loss ratio = 0, obj funct = 1.0
      }

}
```

APPENDIX 6 – GENETIC ALGORITHM CLASSES

```
class FitnessData
   {
   public:
      FitnessData():mPartialSum(0),mIndex(0){};
      double mPartialSum;
      long mIndex;
   };

class PastFitness
   {
   public:
      PastFitness():Hi(0),Lo(0),Average(0){};
      double Hi;
      double Lo;
      double Average;
   };

//**  A TCArray is a te,plate based variable length array with a definable starting index i.e. 0 or 1
typedef TCArray<double,1> EvalValues;
typedef TCArray<EvalValues,1> DistanceArray;
typedef TCArray<double,1> GeneArray;
typedef TCArray<double,1> FitnessArray;
typedef TCArray<FitnessData,1> FitnessDataArray;
typedef TCArray<PastFitness,1> PastFitnessArray;
typedef TCArray<long,0> LongArray0;


//**************************************************************
//**  Virtual Base class for Genome
//**  To run a Genetic Algorithm, create your own genome using this
//**  class as the parent. The Evaluate Function must be over-ridden.
//**  Create a genetic algorithm class using TEvolver as the parent,
//**  passing your genome in as the template parameter.
//**************************************************************
```

```
class Genome
  {
  public:
    Genome();

    virtual void InitGenome();
    virtual void ClearGenomeValues();
    virtual Boolean CheckFeasibility(){return true;};

    virtual void RandomMutateOneGene(double MutationRate,double Divisor);
    virtual void RandomMutate(double MutationRate,double Divisor);
    virtual void CrossoverWith(double CrossoverRate, Genome& TestGenome1,Genome& TestGenome2);
    virtual double Distance(const Genome& TestGenome); // for fitness sharing, can be genotype or phenotype distance

    void SetEvaluationListRatio(long WhichRatio, double TheWeight);
    void SetEvaluationListSize(long TheListSize);
    long GetEvaluationListSize() {return mEvaluationListSize;}
    double GetEvaluationValues(long WhichObjective);
    void   SetEvaluationValues(long WhichObjective, double TheValue);
    double GetEvaluationWeights(long WhichObjective);

    virtual long GetParamCount(){return mTheParameterCount;};
    virtual double GetParamValue(long WhichParam);
    virtual double GetBoundValue(long WhichBound, Boolean Upper);
    virtual void SetParamValue(long WhichParam, double TheParamValue);
    virtual void SetBoundValue(long WhichBound, Boolean Upper, double TheBoundValue);
    virtual Boolean Evaluate() {return true;};//should return false if cancelled, true otherwise

  protected:

    GeneArray Gene;
    GeneArray LowerBounds;
    GeneArray UpperBounds;

    EvalValues mEvaluationValues;//initialize after every mutation and crossover
    Boolean mEvaluated; //initialize after every mutation and crossover

    EvalValues mEvaluationWeights;
    long mEvaluationListSize;
    long mTheParameterCount;

  private:
    Boolean ValidIndex(long IndexVal) {return (IndexVal >0 && IndexVal <= mTheParameterCount);};
    Boolean ValidObjective(long IndexVal) {return (IndexVal >0 && IndexVal <= mEvaluationListSize);};
  };
```

```
//*************************************************************
// Genetic Algorithm Template
//*************************************************************
template <class T> class TEvolver
  {
  public:
    TEvolver();

    virtual void ClearPopulation();
    virtual void SetPopulationSize(long NewPopulationSize);
    virtual void SetGenerationCount(long NewGenerationCountLimit);
    virtual void SetTimeLimit(long NewTimeLimit);
    virtual void SetMutationRate(double NewMutationRate);
    virtual void SetCrossoverRate(double NewCrossoverRate);
    virtual void SetElitism(Boolean ElitismOn);
    virtual void SetExemplarElitism(Boolean ExemplarElitismOn);
    virtual void SetFitnessSharing(Boolean FitnessSharingOn);
    virtual long GetPopulationSize(){return mPopulationSize;}
    virtual long GetGenerationCount(){return mGenerationCountLimit;}
    virtual long GetTimeLimit(){return mTimeLimit;}

    virtual void EvolvePopulation();
    TCArray<T,1> mPopulation; //1..mPopulationSize
    FitnessArray mFitness;    //1..mPopulationSize
    TAverage Values;
    TGenome Exemplar;


  protected:
    virtual Boolean Evolve();
    virtual double GetFitness(T& TheGenome);
    virtual void ScaleFitnessScores();
    virtual void ShareFitness();
    virtual void UpdatingCallback(); //gets called from Evolve - override to do graph updating etc.
    virtual void SetExemplarCallback(); // if you need to preserve any individual other than the best one,
                                        // call this to set the index of the exemplar individual
    virtual long FindAlternativeFittestCallback();
    virtual void EvaluatePopulationCallback();
    virtual void ClearPopulationCallback();
    virtual long RouletteWheel();
    virtual void RouletteWheelSelector(long& Parent1,long& Parent2);
    virtual void TournamentSelector(long& Parent1,long& Parent2);
    TCArray <T,1> mNewPopulation;  //1..mPopulationSize
    EvalValues mBestObjectiveValues;  //1..mPopulationSize
    EvalValues mCurrentObjectiveValues;  //1..mPopulationSize
    EvalValues mWorstObjectiveValues;  //1..mPopulationSize

    FitnessDataArray mFitnessData;  //1..mPopulationSize
    PastFitnessArray mFitnessHistory;  //1..mWhichGeneration


    Long mPopulationSize;// minimum 4
    long mGenerationCountLimit;
    long mTimeLimit;
    long mWhichGeneration;
    long mWhichMember;

    long mStartTime;
    long mCurrentGenerationCount;
    long mRestartIndividual;
    long mRestartGeneration;
    double mMutationRate;
    double mCrossoverRate;
    Boolean mElitism;
    Boolean mExemplarElitism;
    Boolean mFitnessSharing;
    Boolean mEvolutionCancelled;
    Boolean mUseRouletteWheelSelection;
};
```