

# **Multiobjective Evolutionary Optimisation of Small Wind Turbine Blades**

Mark Hampsey, B.E.

Doctor of Philosophy  
Department of Mechanical Engineering  
University of Newcastle  
August 2002



## **Declaration**

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

(Signed) \_\_\_\_\_

*Mark Hampsey*



## Acknowledgements

I would like to thank my supervisor, Associate Professor David Wood, for guiding my progress through both the completion of this thesis and the research project it documents. He was an expert guide for the technical aspects of the research and is a patient, dedicated listener. These are rare qualities, particularly in the field of postgraduate supervision, and any research student lucky enough to find a supervisor who possesses both should, like me, feel privileged.

I'd also like to thank Dr. Phillip Clausen for jointly accepting the risk of taking me on as a student. His gracious assistance during the first year of my candidacy made everything a lot less painful. Jayantha Epaarachchi, Yuhua Guo, Paul Peterson and Daniel Riley deserve my gratitude for just being a friendly bunch of people. I wish them every future success. The Mechanical Engineering workshop staff were always available and I thank them for their help.

Since this is the only opportunity I will ever have to do it, I offer my sincerest gratitude to everyone who has ever donated their time to publishing free software, especially the GNU project ([www.gnu.org](http://www.gnu.org)), and the Linux operating system, with special thanks deserved by Red Hat Linux ([www.redhat.com](http://www.redhat.com)) and a host of thousands of other software developers too numerous to mention. This project would not exist, and likely *could never* exist, without it.

Any number of words are simply inadequate to encapsulate a lifetime of gratitude to my parents. How do I do it in one paragraph? Thank you Mum and Dad. You made me who I am. You deserve the blame. Both of you are best-of-generation individuals.

And the biggest thank you goes to Elizabeth. Just for being Elizabeth.



# Table of Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Literature Review</b>	<b>15</b>
2.1	Wind turbine performance analysis and prediction . . . . .	15
2.1.1	The small wind turbine starting problem . . . . .	15
2.1.2	Low-speed aerofoils for wind turbines . . . . .	17
2.1.3	Rotary wings . . . . .	18
2.1.4	Blade Element/Wake Momentum Method . . . . .	19
2.1.5	Panel Methods . . . . .	20
2.2	Aerodynamic optimisation . . . . .	23
2.2.1	Two-dimensional geometries . . . . .	24
2.2.2	HAWT optimisation . . . . .	26
2.3	Evolutionary Optimisation . . . . .	26
2.3.1	Genetic Algorithms . . . . .	28
2.3.2	Evolution Strategies . . . . .	30
2.3.3	Differential Evolution . . . . .	31
2.3.4	Multiobjective Optimisation . . . . .	32
2.3.5	Pareto Optimality . . . . .	33
<b>3</b>	<b>Geometric representation of aerofoils and blade surfaces</b>	<b>35</b>
3.1	Introduction . . . . .	35
3.2	B-spline curves . . . . .	37
3.2.1	Least-squares B-spline curve fitting to point data . . . . .	40
3.3	B-spline surfaces . . . . .	42
3.4	Aerofoil curve fitting using a DE optimiser . . . . .	44

3.4.1	Application: creation of transition region geometry . . . . .	54
<b>4</b>	<b>Panel Method</b>	<b>55</b>
4.1	Introduction . . . . .	55
4.2	Blade discretisation . . . . .	57
4.3	Wake model and wake discretisation . . . . .	63
4.4	Modifications to the simple wake model . . . . .	64
4.4.1	First wake panel . . . . .	64
4.4.2	Near wake . . . . .	66
4.4.3	Wake pitch relaxation . . . . .	67
4.5	The 3D constant-source/doublet panel method . . . . .	71
4.5.1	Numerical solution of Laplace's equation . . . . .	72
4.5.2	Panel geometry . . . . .	74
4.5.3	Doublet singularity . . . . .	76
4.5.4	Source singularity . . . . .	76
4.6	Calculation of quantities of interest . . . . .	77
4.7	Simple iterative Kutta condition . . . . .	79
4.8	Exploiting the symmetry of multi-bladed rotors . . . . .	82
4.9	Two-dimensional code validation . . . . .	83
4.10	Three-dimensional code validation . . . . .	98
4.10.1	A simple correction for viscosity . . . . .	98
4.10.2	Test case: large constant-speed wind turbine . . . . .	105
4.10.3	Test case: small variable-speed wind turbine . . . . .	120
4.10.4	Test case: untapered, twisted NREL Phase III turbine blade .	125
4.10.5	Test case: small, twisted, tapered wind turbine blade . . . . .	144
<b>5</b>	<b>A simplified starting model</b>	<b>149</b>
5.1	Introduction . . . . .	149
5.1.1	Resistive torque . . . . .	151
5.2	Second moment of inertia calculation . . . . .	152
5.2.1	Volume discretisation . . . . .	152
5.2.2	Connecting surface and through-thickness vertices into discrete hexahedral bricks . . . . .	153
5.2.3	Volume and Centroid calculation . . . . .	154

5.3	Assessing blade starting performance . . . . .	155
5.4	A linear-static finite element model . . . . .	157
<b>6</b>	<b>Multiobjective Differential Evolution wind turbine blade optimiser</b>	<b>163</b>
6.1	Introduction . . . . .	163
6.2	Single-objective Differential Evolution algorithm . . . . .	165
6.3	Blade parameterisation . . . . .	168
6.4	Pareto optimality . . . . .	170
6.4.1	Pareto selection . . . . .	171
6.4.2	Selecting for viability . . . . .	172
6.5	Geometric Constraints . . . . .	174
6.5.1	Hard envelope constraints . . . . .	174
6.5.2	Straight leading edge constraint . . . . .	176
6.5.3	The “sane geometry” constraint . . . . .	176
6.5.4	Minimum thickness constraint . . . . .	177
6.6	Experimental run parameters . . . . .	180
6.6.1	Task specification . . . . .	180
6.6.2	Objective functions . . . . .	181
6.6.3	Seed blade geometry . . . . .	181
6.7	Optimisation results: 5 kW blade . . . . .	186
6.7.1	Existing 5 kW blade geometry . . . . .	186
6.7.2	NACA 0012 seed blade . . . . .	194
6.7.3	Evolving a population of B-spline wind turbine blades . . . . .	202
6.7.4	A Pareto optimal set of wind turbine blades . . . . .	206
6.7.5	Pareto set: “Slender” blade family . . . . .	209
6.7.6	Pareto set: Slender-Ordinary hybrids . . . . .	210
6.7.7	Pareto set: “Ordinary” blade family . . . . .	211
6.7.8	Pareto set: Ordinary-Heavy hybrids . . . . .	212
6.7.9	Pareto set: “Heavy” blade family . . . . .	213
6.7.10	Pareto set: member [A] . . . . .	214
6.7.11	Pareto set: member [B] . . . . .	222
6.7.12	Pareto set: member [C] . . . . .	228
6.7.13	Pareto set: member [D] . . . . .	236

6.7.14	Pareto set: member [E] . . . . .	244
6.7.15	Pareto set: member [F] . . . . .	250
6.7.16	Pareto set: member [G] . . . . .	258
6.7.17	Pareto set: member [H] . . . . .	266
6.8	Discussion . . . . .	274
6.9	Blades and aerofoils: aerodynamic differences . . . . .	278
6.10	Obtaining a “rounder” leading edge . . . . .	292
6.11	Optimisation results: 600 Watt blade . . . . .	294
6.11.1	Existing 600 Watt blade . . . . .	297
6.11.2	600 Watt “seed” blade . . . . .	304
6.11.3	600 Watt [A] blade . . . . .	313
6.12	Discussion . . . . .	320
<b>7</b>	<b>Conclusion</b>	<b>323</b>
<b>References</b>		<b>329</b>
<b>A</b>	<b>B-spline curve fundamentals</b>	<b>339</b>
A.1	Curve derivatives . . . . .	339
A.2	Projection of a point onto a curve . . . . .	340
<b>B</b>	<b>Lanchester-Betz limit via actuator disc theory</b>	<b>341</b>
B.1	Finding the flow velocity at the rotor . . . . .	341
B.2	Betz limit . . . . .	343
<b>C</b>	<b>Code Outline (on supplemental CD)</b>	<b>345</b>
<b>D</b>	<b>Supplemental CD</b>	<b>465</b>

# List of Figures

1.1	Apparent flow velocity at radius $r$ . . . . .	4
3.1	A quadratic B-spline curve ( $p=2$ ) . . . . .	38
3.2	SD7062 cubic B-spline representation . . . . .	40
3.3	A spanwise assembly of SD7062 aerofoils . . . . .	42
3.4	chord and twist distributions for 5kW blade . . . . .	43
3.5	LM19.1 blade planform (Sørensen (1999)) . . . . .	45
3.6	LM19.1 unit-chord aerofoil profiles (Sørensen (1999)) . . . . .	45
3.7	B-spline aerofoils for the LM19.1 blade . . . . .	46
3.8	Skinned LM19.1 blade discretised into quadrilateral surface panels . . . . .	46
3.9	B-spline aerofoil and circular data point-set . . . . .	49
3.10	A set of evolving best-of-generation curves . . . . .	50
3.11	An evolved circular B-spline curve sharing the SD7062 knot vector . . . . .	51
3.12	Fitness history of initial optimisation run . . . . .	51
3.13	Fitness history of refinement optimisation run . . . . .	52
3.14	A spanwise sequence of circular and aerofoil B-spline curves . . . . .	52
3.15	A skinned and meshed parametric surface . . . . .	53
3.16	Blade without root transition . . . . .	54
3.17	Blade with root transition . . . . .	54
4.1	Two-bladed panel mesh showing global co-ordinate system . . . . .	57
4.2	Panelling direction . . . . .	58
4.3	Panel numbering scheme . . . . .	59
4.4	Chordwise cosine spacing . . . . .	60
4.5	HAWT blade mesh. N_PANELS=30; M_PANELS=20 . . . . .	60
4.6	Wake panel numbering scheme . . . . .	63

4.7	A wake mesh; M_PANELS=10, W_PANELS=20, TURNS=2 . . . . .	64
4.8	Finding a direction vector which bisects the blade trailing edge . . . . .	66
4.9	Denser near-wake mesh. M_PANELS=10 ; W_PANELS=10; TURNS=2	67
4.10	NREL combined experiment turbine tip vortex visualisation; and simple wake model by Wood and Boersma (2001) . . . . .	68
4.11	Wake pitch convergence for 5kW rotor (40x20mesh) $\lambda=10$ . . . . .	69
4.12	Strength of doublet on each streamwise wake strip . . . . .	72
4.13	Panel co-ordinates . . . . .	74
4.14	Calculation of source strength over one panel edge . . . . .	76
4.15	Scheme for central differences on panels . . . . .	78
4.17	Pressure equalisation at trailing edge of 5kW blade . . . . .	80
4.18	Van de Vooren Aerofoil (15% thick) . . . . .	84
4.19	Aerofoil mapping (finite trailing edge angle) (Katz and Plotkin (2001))	84
4.20	Analytical vs Panel method pressure distributions for the Van de Vooren aerofoil . . . . .	88
4.21	Dependence of panel method accuracy on panel density . . . . .	92
4.22	Van de Vooren lift coefficient vs $\alpha$ at various panel densities . . . . .	96
4.23	Van de Vooren circulation vs $\alpha$ at various panel densities . . . . .	97
4.24	Scheme for “viscous” boundary condition correction . . . . .	99
4.25	Computed and experimental lift coefficients: SD7062 and NACA 4412 aerofoils . . . . .	102
4.26	Computed and experimental lift and drag polars: SD7062 aerofoil . .	103
4.27	Computed and experimental lift and drag polars: NACA 4412 aerofoil	104
4.28	Comparison with results from Sørensen (1999) . . . . .	106
4.29	Comparison with Risø LM19.1 predictions: spanwise axial and tangential force distributions . . . . .	108
4.30	Panel method vs Risø LM19.1 pressure predictions at $U_0 = 7$ m/s . .	111
4.31	Panel method vs Risø LM19.1 pressure predictions at $U_0 = 10$ m/s .	114
4.32	Panel method vs Risø LM19.1 pressure predictions at $U_0 = 15$ m/s .	117
4.33	Computed power curve for the 5 kW rotor . . . . .	120
4.34	Computed $U_1$ vs $\lambda$ for the 5 kW rotor . . . . .	122
4.35	Axial force distributions for the 5 kW blade at $U_0=10$ m/s . . . . .	122
4.36	Tangential force distributions for the 5 kW blade at $U_0=10$ m/s . . . . .	123

4.37	Torque distributions for the 5 kW blade at $U_0=10\text{m/s}$	123
4.38	Circulation distributions for the 5 kW blade	124
4.39	NREL Phase III blade and rotor geometry	125
4.40	Comparison of predicted and experimental Power curves for unta-pered, twisted NREL Phase III rotor	127
4.41	Panel method $C_p$ predictions vs NREL Phase III experimental results	130
4.44	Panel method $C_{torque}$ and $C_{thrust}$ predictions vs NREL Phase III experimental results	141
4.47	Panel code (with simple viscous correction) and PROPID power predictions for three rotor configurations	145
4.48	Panel code (with simple viscous correction) and PROPID thrust predictions for three rotor configurations	146
4.49	Panel code (with simple viscous correction) and PROPID lift predictions. NB=3: 5 deg pitch and 72 rpm	147
5.1	The starting sequence of a 5 kW turbine (Mayer et al. (2001))	150
5.2	a small mass rotating about an axis	152
5.3	Blade modelled as a skin of fibreglass bricks surrounding a foam core	153
5.4	Scheme for brick connectivity: (a)vertices (b)faces (c)face vertex order	153
5.5	Torque and Acceleration curve comparisons for the existing 5 kW blade and a straight, untwisted “seed” geometry constructed from NACA 0012 aerofoils	155
5.6	Predicted starting torque distribution for existing 5 kW blade	156
5.7	Aerodynamic loads; two-bladed 5kW HAWT; $U_0=10\text{ m/s}$ $\lambda=10$	159
5.8	Displacement in x direction; two-bladed 5kW HAWT; $U_0=10\text{ m/s}$ $\lambda=10$	159
5.9	Principal stress; two-bladed 5kW HAWT; $U_0=10\text{ m/s}$ $\lambda=10$	161
5.10	Principal strain; two-bladed 5kW HAWT; $U_0=10\text{ m/s}$ $\lambda=10$	161
6.1	Crossover for $D=7$ , $n=2$ and $L=3$	167
6.2	Crossover for $D=7$ , $n=5$ and $L=4$	168
6.3	(a)common knot vector (b)one aerofoil vector (c)complete parameter vector	170
6.4	Blade twist envelope	175
6.5	Scheme for B-spline aerofoil thickness determination	178

6.6	Least-squares curve fit to SD7062 point data with 7 control points . . . . .	182
6.7	B-spline approximation to the SD7062 aerofoil with 7 control points . .	183
6.8	B-spline approximation to the NACA 0012 aerofoil with 7 control points	183
6.9	A simplified rectangular hub section . . . . .	184
6.10	Original 5 kW blade - Geometry and performance results . . . . .	186
6.15	NACA 0012 seed blade - Geometry and performance results . . . . .	194
6.20	Performance figures for the evolving population over 2000 generations	204
6.21	Pareto front after 2000 generations . . . . .	206
6.22	Span-wise cross-sections through the “Slender” blade family . . . . .	209
6.23	Span-wise cross-sections through the Slender-Ordinary hybrid series .	210
6.24	Span-wise cross-sections through the “Ordinary” blade family . . . . .	211
6.25	Span-wise cross-sections through the Ordinary-Heavy hybrid series .	212
6.26	Span-wise cross-sections through the “Heavy” blade family . . . . .	213
6.27	Pareto [A] blade - Geometry and performance results . . . . .	214
6.32	Pareto [B] blade - Geometry and performance results . . . . .	222
6.37	Pareto [C] blade - Geometry and performance results . . . . .	228
6.42	Pareto [D] blade - Geometry and performance results . . . . .	236
6.47	Pareto [E] blade - Geometry and performance results . . . . .	244
6.52	Pareto [F] blade - Geometry and performance results . . . . .	250
6.57	Pareto [G] blade - Geometry and performance results . . . . .	258
6.62	Pareto [H] blade - Geometry and performance results . . . . .	266
6.67	Original 5 kW blade (SD7062 section): 2D and 3D pressure comparisons	284
6.68	Pareto [D] blade: cross-section (1 of 7) . . . . .	285
6.69	Pareto [D] blade: cross-section (2 of 7) . . . . .	286
6.70	Pareto [D] blade: cross-section (3 of 7) . . . . .	287
6.71	Pareto [D] blade: cross-section (4 of 7) . . . . .	288
6.72	Pareto [D] blade: cross-section (5 of 7) . . . . .	289
6.73	Pareto [D] blade: cross-section (6 of 7) . . . . .	290
6.74	Pareto [D] blade: cross-section (7 of 7) . . . . .	291
6.75	Scheme for ensuring a “round” leading edge . . . . .	292
6.76	600 Watt original blade - Geometry and performance results . . . . .	297
6.81	600 Watt seed blade - Geometry and performance results . . . . .	304
6.86	Performance figures for the evolving population over 1600 generations	310

6.88	600 Watt Pareto [A] blade - Geometry and performance results . . . . .	313
A.1	Projecting a point onto a curve (Piegl and Tiller (1997)) . . . . .	340
B.1	Lanchester-Betz actuator disc .	341



# Nomenclature

$\alpha$	Angle of Attack
$\phi$	Velocity potential
$\Gamma$	Circulation around a turbine blade ( $\frac{m^2}{s}$ )
$\lambda$	Tip speed ratio = $\frac{R\Omega}{U_0}$
$\mu$	Viscosity of air at 1 atm. & 300K = $1.846 \times 10^{-5} (\frac{Ns}{m^2})$
$\nu$	Kinematic viscosity of air at 1 atm. & 300K = $1.589 \times 10^{-5} (\frac{m^2}{s})$
$\Omega$	Angular speed ( $\frac{rad}{s}$ )
$\dot{\Omega}$	Rotational acceleration ( $\frac{rad}{s^2}$ )
$\rho$	Density of air at 1 atm. & 300K = $1.204 (\frac{kg}{m^3})$
$\theta$	Angle of incidence
$\theta_p$	Pitch angle
$C_D$	Drag coefficient
$C_L$	Lift coefficient
$C_P$	Power coefficient
$C_p$	Pressure coefficient
$c$	Chord length of an aerofoil or turbine blade section (m)
$D$	Drag force: ( $N$ ) for wing; ( $\frac{N}{m}$ ) for aerofoil

<b>I</b>	Second moment of inertia ( $kgm^2$ )
<b>L</b>	Lift force ( $N$ ) for wing; ( $\frac{N}{m}$ ) for aerofoil
<b>n</b>	Unit-normal vector pointing outwards from solid blade surface
$p_\infty$	Wake vortex pitch (m)
<b>R</b>	Blade length (rotor radius) ( <i>metres</i> )
$r$	Distance along blade span from hub (m)
$Re$	Reynolds Number = $\frac{U_t c}{v}$
<b>S</b>	A body surface
<b>S</b>	Planform area of a blade element strip ( $m^2$ )
$t$	time (s)
<b>T</b>	Resultant torque on rotor (Nm)
<b>U<sub>F</sub></b>	Total fluid velocity in rotating frame of reference ( $\frac{m}{s}$ )
<b>U<sub>W</sub></b>	Unperturbed fluid velocity in rotating frame of reference ( $\frac{m}{s}$ )
$U_0$	Free-stream wind speed ( $\frac{m}{s}$ )
$U_1$	Component of free-stream wind speed at wind turbine( $\frac{m}{s}$ )
$U_\infty$	Velocity of far wake (in the direction of the free-stream)( $\frac{m}{s}$ )
$U_t$	Total apparent wind velocity seen at wind turbine( $\frac{m}{s}$ )
<b>X</b>	Local speed ratio = $\frac{r\Omega}{U_0}$

# Abstract

Wind turbines, large and small, usually rely on the wind to accelerate them from rest to speeds where power production becomes possible. This is less of a problem for large wind turbines than for small, since judicious site selection for commercial wind farms ensures that large wind turbines see relatively high winds for considerable portions of the year. Small wind turbines, on the other hand, are usually sited where the power is required, which means that long portions of their operational lives may be spent idling in sub-optimal wind conditions. It is vitally important, then, that small wind turbines accelerate from rest quickly whenever strong winds start to blow so as not to squander this infrequent resource.

The project documented in this thesis employed a multiobjective evolutionary algorithm (based on Differential Evolution) to “evolve” a Pareto optimal set of wind turbine blade designs, with each member offering a unique trade-off between the conflicting objectives of good starting performance and good peak power production. A first-order three-dimensional panel method with constant source and doublet distributions was employed to approximate the aerodynamic performance of candidate blade designs at a low tip speed ratio (for starting performance) and at a high operational tip speed ratio (for peak power production). A simple helical wake model with iterative wake pitch adjustment was included to model the circulation shed from each blade’s trailing edge. To allow geometry evolution to occur, a framework for compactly representing a huge range of possible three-dimensional blade geometries as bi-cubic B-spline surfaces was developed. Blades represented in such a way were subsequently discretised into quadrilateral surface panels for use by the panel solver.

Results show that blade designs evolved by this method accelerate more quickly than current blade designs without suffering a degradation in peak power output. The most significant avenue for improved acceleration performance was via a reduction in second moment of inertia of candidate blades, which seems to involve a move towards more slender taper distributions and thinner “aerofoil” sections, resulting in geometries which nevertheless behave well aerodynamically.



# Chapter 1

## Introduction

The natural alternative to sitting down and writing a thesis is to sit down and watch television. When there is nothing worth watching on television (a depressingly common phenomenon), the next-best alternative is to sit and stare out the window at a very large eucalypt. Occasionally, a family of kookaburras take up residence for a couple of days and entertain me with their cackles. Sometimes it's a pair of sulphur-crested cockatoos, scarcely less noisy than their cousins and just as entertaining, or groups of large, healthy magpies with their amazingly musical calls. And at other times, when the wind really blows, I can neither see nor hear the presence of bird-life perching or gliding or swooping in or around that huge expanse of wood and leaf. The feathery denizens of the wind know the ether in which they are immersed can wield ferocious power.

There is a lot of energy in wind. Imagine standing at the precipice of a cliff overlooking the sea. At sea level, the density of air is 1.204 kilograms per cubic metre. Imagine holding up an empty, square picture frame, one-metre to a side, and that the wind is blowing in from the sea, through the picture frame, at a speed of ten metres per second, or 36 kilometres per hour - a moderately stiff breeze. Every second, 12.04 kilograms of air passes through the picture frame. The kinetic energy of the 12.04 kilograms of air is given by

$$K.E. = \frac{1}{2} \text{mass} \times \text{velocity}^2 \quad (1.1)$$
$$\approx 600 \text{ Joules}$$

which means that, every second, 600 Watts of untapped, natural power is flowing

through the picture frame from the sea, power which the earth uses to rearrange the sand on the beach, the leaves on the trees and, during storms, the roofs of people's houses. Power which, properly captured and converted, can be used in the same way that fossil and nuclear power is used now.

600 Watts per square metre in a 10 m/s wind may not *sound* like much. An electric kettle uses three or four times that sort of power to boil a couple of cups of water in a minute. Nuclear and fossil energy lobbyists have been known to refer to wind (and solar) energy derisively as "piddle power" (<http://www.energyadvocate.com>) for just this reason: typical fossil or nuclear fueled commercial power plants deal in the tens of Megawatts to *Gigawatts* of non-renewable power. But consider a really, really large picture frame, a circular one, with a radius of 100 metres. The area through which a 10 metre per second wind blows is over 31,000 times greater than a 1 metre square frame, and carries with it almost 19 Megawatts of power.

Horizontal-axis wind turbines with 100 metre long blades are now being built in Denmark. Gaining a foothold in the early 1970's when the fossil-fuel energy crisis (very slightly) shook the western-world's faith in an ever-continuing bounty of oil, wind-generated electricity has since developed into a mature and thriving industry. The market for large, commercial-scale wind turbines (and conglomerations of turbines, or wind *farms*) is currently growing at a geometric rate, and the trend is towards the construction of larger and larger turbines to take advantage of the economies of scale offered by very large swept areas.

A horizontal-axis wind turbine, or HAWT, is the dominant design for capturing a portion of the power in the wind and converting it into electrical energy. It consists of one or more "blades", much like the blades of a fan or aeroplane propeller, attached to the shaft of an electric generator. The blade/generator system is typically situated atop a mast to give it clearance from the ground. The mast is at least 1.5 rotor diameters tall and usually much higher for obvious operational and safety reasons, but also to lessen the effect of wind shear. The other major design, the vertical-axis wind turbine, has not been as successful at capturing the imagination of technocrats as the HAWT due to a number of technical and economic reasons, and very, very few exist, even as demonstration systems. For this reason, "wind turbine" in this thesis means "horizontal-axis wind turbine".

Large commercial wind turbines are all very well and good for furthering the public

acceptance of renewable energy and, at the beginning of the twenty-first century, are showing all the signs of being one of the dominant technologies waiting to supplant nuclear and fossil fuels during the next few decades. Denmark in particular has taken to wind energy in a major way.

At the other end of the size scale lie small wind turbines. Their existence owes more to necessity than to any popular notion of “green energy” or “sustainability”. They make possible the provision of electricity to individual homes, communities or vehicles which do not have access to an electricity grid and who otherwise must rely on a diesel generator and a constant supply of fuel. The provision of electrical power to non-grid-connected dwellings or communities is by a Remote Area Power System (RAPS), and is typically built around a diesel genset. The addition of supplementary systems such as solar hot water, solar photo-voltaic (PV) and small-scale wind lessen the reliance on expensive and sometimes-scarce fuel, and so are valuable additions to a RAPS setup. PV and wind in RAPS usually charge a bank of DC batteries, with AC power drawn from these via an inverter as demand requires. Clausen and Wood (2000) classify small wind turbines as: “micro” - turbines that have a blade length of less than 1.5 metres, generate less than 1 kilowatt and power the likes of electric fences and the equipment on yachts; “mid-range” - turbines with a blade length of around 2.5 metres, generate a peak of around 5 kilowatts and supply the power requirements of remote houses; and “mini” - turbines with a blade length of 5 metres or greater, generate a peak of 20 kilowatts or more and which supply the requirements of mini-grids for small remote communities.

Wind turbine blades, like aeroplane wings and helicopter blades, are long (high aspect-ratio) structures with a cross-section that looks like an aerofoil. Consider the section of wind turbine blade shown in Figure 1.1. The wind turbine rotates about an axis parallel to the free stream wind velocity,  $U_0$  with angular speed  $\Omega$ . The apparent flow speed seen by the wind turbine at a radius  $r$  is <sup>1</sup>

$$U_t = \sqrt{U_1^2 + (r\Omega)^2} \quad (1.2)$$

where  $U_1$  is the component of  $U_0$  seen at the wind turbine (see appendix B.1). The angle  $U_t$  makes with the plane of rotation is called the *angle of incidence*, and is given

---

<sup>1</sup>Ignoring the usually small rotational inflow factor acting in the opposite direction to the  $r\Omega$  term.

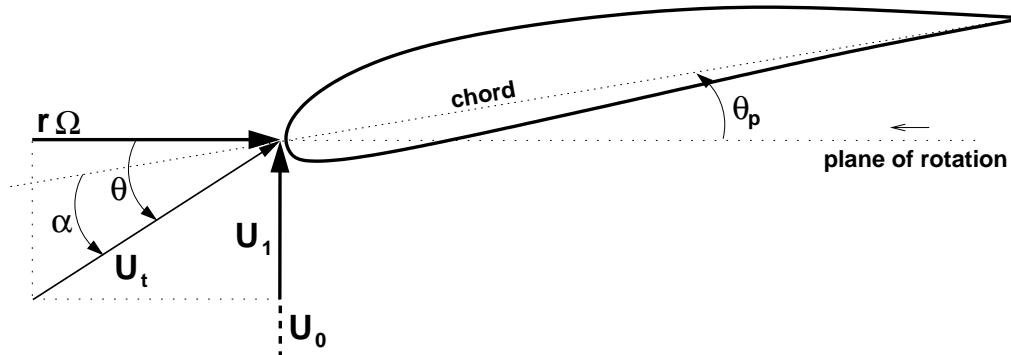


Figure 1.1: Apparent flow velocity at radius  $r$

by

$$\theta = \tan^{-1} \frac{U_1}{r\Omega} \quad (1.3)$$

The *local speed ratio*,  $X$ , is given by

$$X = \frac{r\Omega}{U_0} \quad (1.4)$$

with an important special case of the local speed ratio occurring at radius  $r = R$ , where  $R$  is the length of the blade. This is called the *tip speed ratio*

$$\lambda = \frac{R\Omega}{U_0} \quad (1.5)$$

The angle  $U_t$  makes with the blade section's chord line is called the *angle of attack*

$$\alpha = \theta - \theta_p \quad (1.6)$$

where  $\theta_p$  is an arbitrary *pitch angle* set by the blade's designer.

When using standard low-speed aerofoil sections, with empirically-gathered performance data, the design of wind turbine blades for maximum power extraction is a well-understood task, and predicting the power output and blade loads can be accomplished with a high-degree of accuracy via the “Blade Element Method” (BEM), a technique developed in the nineteenth-century for the design of large steam ship propellers (Gouriéres (1982) and Spera (1994) provide details of the BEM as applied to wind turbines). The theoretical maximum amount of energy a horizontal-axis wind turbine, large or small, can extract from the wind is about 59%, a universally-accepted performance limit calculated via the Lanchester-Betz limit derived from actuator disc

theory (appendix B.1). Current commercial blade designs achieve a respectable 50% power extraction at their operational peak.

It is important to note that optimal-power only occurs when aerofoils along the entire span of the blade (or a significant portion of it) are themselves performing optimally. Aerofoils are shapes that “perform” best at low-to-moderate angles of attack, after which the kinematic viscosity of the air,  $\nu$ , starts to exert its influence, resulting in *flow separation*, and other complicated aerodynamic behaviour. This may significantly degrade the desirable *lift* force experienced by the aerofoil, and also gives rise to a corresponding increase in the undesirable *drag* force, phenomena which lead to such things as a dangerous tendency to *stall* when a aircraft attempts to climb too steeply. Viscosity also tends to have more deleterious consequences for lower speeds and smaller aerofoils: an aerofoil moving through the air quickly at a certain angle of attack will, generally, tend to exhibit less flow separation than that same aerofoil at the same angle of attack at low speed. And a smaller aerofoil won’t fare as well as a larger one at the same angle of attack and speed. This is quantified by a non-dimensional velocity, the *Reynolds number*:

$$Re = \frac{U_t c}{\nu} \quad (1.7)$$

where  $c$  is the *chord* length of the aerofoil and  $\nu$  is the kinematic viscosity of the air. Just as aerofoils behave well at low-to-moderate angles of attack, they also perform more reliably and desirably at higher Reynolds numbers. The consequences for wind turbines are thus: optimal power production will occur when the blades experience “good” (high) Reynolds numbers at “good” (low) angles of attack. Small wind turbines thus need to spin much more quickly than large wind turbines - a fact illustrated by the long majestic sweeps of giant wind turbines operating at peak performance on the sides of hills and the rapid rotation of a micro turbine on the hull of a yacht. This is a fundamental rule for the design of wind turbines. Which poses a problem when *peak power* is not the only design goal.

Almost all small wind turbine blades rely on the wind to accelerate them from rest to speeds where power production becomes feasible. Imagine a remote homestead that relies on PV and mid-range wind to charge a bank of batteries so that the occupants can watch television (and not have to resort to their small diesel genset and their precious stash of fuel). The wind turbine in this RAPS has the task of providing electricity *where*

*it is needed*, and does not have the luxury that its giant cousins enjoy - to be sited on the sides of hills or in the middle of the ocean, where the wind blows constantly and hard for significant portions of the year. Obviously, the middle of a paddock is unlikely to experience the sort of wind regimes eagerly sought by governments and power utilities for their wind farms. It is important, then, that this small turbine catch as much of the available wind as possible - a task that can only be accomplished if it is ready to *start* spinning and *accelerate* to power-production speeds as soon as the wind starts to blow.

The *starting problem*, then, is illustrated by the flow vectors depicted in Figure 1.1. A stationary rotor ( $\Omega = 0$ ) with untwisted blade sections ( $\theta_p = 0$ ) can be seen to experience an angle of attack,  $\alpha$ , of  $90^\circ$ , with a  $U_t$  component nearly equal to the wind speed, giving a tip-Re of about 15,000 for a 2.5 metre blade with a 75 millimetre tip chord. At such extreme angles and tiny Reynolds numbers the aerofoil sections along the span of the blade start acting like bluff bodies, with lift and drag characteristics similar to those of a flat plate. It is very, very difficult to design aerodynamic shapes that perform well in these circumstances, a problem not helped by the fact that the libraries of aerofoil sections available to wind turbine blade designers have evolved from a century of aviation design, and rarely have data for angles of attack greater than twenty degrees, and Reynolds numbers lower than 100,000. Despite this, turbine blades *do* accelerate from rest to power-producing speeds. This fact is a fortunate by-product of their peak-power design. The ability to extract power optimally, however, does not imply optimal starting performance.

Most wind turbine designs, large and small, have decreasing pitch with increasing radius. This is intended to compensate for the relatively low local speed ratios (and high angles of incidence) near the blade hub to maximise power output by keeping the circulation constant along the blades during normal operation. For small wind turbines, large pitch angles near the hub of the blade partially offset the very high angles of attack during starting and offer the initial torque required for acceleration from rest. It should be emphasised that this is typically *not* a design feature intended to have a direct bearing on starting performance, but rather a beneficial (and necessary) coincidence. In fact very little research data exists, experimental or otherwise, which would allow a wind turbine engineer to reliably include starting performance as a design goal.

At a minimum, the aerodynamic torque experienced by a stationary wind turbine must overcome the resistance to rotation offered by its generator. As an example, the

experimental 5 kW wind turbine operated by the Wind Energy Group at the University of Newcastle has a generator requiring a minimum of 1 Nm to rotate. Which means that, if the cut-in wind speed goal is 3 m/s, the stationary 5 kW wind turbine blades must generate at least 1 Nm in a 3 m/s wind. And this rather tall order gets worse as wind turbine sizes get smaller. The magnitude of the wind turbine's shaft torque scales with  $R^2$ , the square of its blade length. So, imagine a wind turbine with blades identical in every respect to the 2.5 metre, 5 kW turbine blades, except they are scaled to a length of 1 metre, giving a wind turbine with a nominal 600 Watt output at  $U_0=10$  m/s. And also imagine that, for whatever reason, the 5 kW blades are capable of only producing a bare minimum of 1 Nm of starting torque in a 3 m/s wind. The equivalent starting torque on this micro wind turbine in a 3 m/s wind would be a tiny 0.16 Nm. Commercial permanent magnet generators typically used on 600 Watt turbines can have a resistive cogging torque of around 0.4 Nm. Which means that the cut-in wind speed of very small wind turbines is highly dependent on the resistive torque of their generators. Thus starting *torque* is an important blade parameter to optimise, and this importance rapidly increases as turbine sizes decrease.

Which brings us to the topic of this thesis: how can the starting performance of small wind turbines be improved, without degrading power production? The obvious solution would seem to be: find an aerodynamicist to conduct many experiments, physical and computational, to come up with a suite of aerofoil sections which perform well at the typical angles of attack and Reynolds numbers encountered by small wind turbines. To which the obvious response is: but experienced low-speed aerodynamicists are rare, and the required experiments would be awfully expensive and time-consuming. Why can't a computer just do it?

The project documented in this thesis is an attempt at "just getting a computer to do it". This sounds facile, but it is actually quite a powerful idea: removing the requirement of a human aerodynamicist's skill and knowledge; providing a computer with only the rudiments of what is required, setting it off in the right direction and then simply waiting for it to come up with an answer. There is a precedent for this sort of knowledge-free optimisation: life on Earth. Darwin's theory of *Evolution* holds that all life on Earth has evolved, via natural selection, from one or more basic seed life forms, themselves perhaps evolved at ancient under-sea hydrothermal vents from self-reproducing organic molecules resembling randomly-folded protein chains. The

survival of a particular design, or *species*, is simply a product of how well the individual members of the species survive by adapting to their environment. This process is *not* optimisation, at least not in the engineering sense of the word. There is no one “optimal” life form on Earth. Every species has evolved to fill a survival niche, with each possessing a set of sub-optimal qualities allowing them to co-exist with other species and, to greater or lesser degrees of success, providing the balance between fertility and extinction. Computer scientists and engineers adopted this idea, bringing into existence the field of “Evolutionary Computation”, a suite of related techniques for evolving approximate solutions to problems as diverse as that of finding a computer program that can play Grand-Master level chess, to the creation of a light-weight, stiff space-frame design for a spar on a geo-stationary satellite.

Evolutionary Optimisation of small wind turbine blades via a computer would thus seem to require two complementary parts:

- a computer-generated representation of a wind turbine blade that is flexible enough to represent a very wide range of potential blade shapes, and compact enough to allow the efficient storage and manipulation of a viable breeding *population* of candidate blades.
- a representation of the “environment” into which the members of this evolving population are born, and one that can accurately assess a candidate blade’s “fitness” for survival *compared to that of its peers*, and hence its suitability for reproduction.

The artificial environment chosen to test the fitness of candidate blade designs in the current project is a three-dimensional panel method - perhaps the simplest *general* three-dimensional aerodynamic solver possible. A constant-diameter, constant-pitch wake model was added to approximate the real helical surface of shed circulation trailed by an operating wind turbine. The assumptions made in adopting this aerodynamic solver were:

- Inviscid flow. Panel methods solve Laplace’s equation for the flow velocities at the surface of a body immersed in a moving flow. Laplace’s equation is a simplification of the full viscous Navier-Stokes equations, with all of the viscous terms removed. For aerofoil/wing/wind turbine blade flows at low angles of attack and

high Reynolds numbers - that is, attached flow - it supplies performance results very close to reality. Which poses an obvious problem: how to model starting, a flow regime where viscous effects are significant? The answer to that, at least in the context of the current project, is to solve the inviscid model and then add an empirical correction for angle of attack and Reynolds number. This would be an outrageous thing to do if the goal was to *accurately predict wind turbine blade loads on general three-dimensional geometries during starting* - a task perhaps best left to time-consuming, grid-based (ie: finite difference/finite volume/finite element) Navier-Stokes solvers and a supercomputer for each evaluation. The justification for using an inviscid method with a fairly unsophisticated viscous correction is that it is fast and it is useful in the sense that the dominant optimisation pathway is via comparisons *between* peer blade designs - each of which has been evaluated by the same limited aerodynamic objective function, rather than between a candidate blade design and a physical standard.

- Steady flow. Starting is an unsteady aerodynamic process, even in a steady wind, simply because an accelerating wind turbine blade has constantly changing  $Re$  and  $\alpha$  along its span as a consequence of its increasing rotational speed. A measure of starting performance, which should rightly include unsteady aerodynamic effects, was abandoned in the pursuit of fast evaluation times: Starting performance is represented by a discrete measure of instantaneous angular acceleration, calculated by simply dividing the steady aerodynamic torque on the wind turbine by its second moment of inertia at a low tip speed ratio. Unsteadiness is also a factor when designing wind turbine blades for optimal power production in wind regimes featuring constantly shifting wind speed and direction, which feeds into the design of wind turbine systems for optimal yaw behaviour - an important topic, since small wind turbines should quickly change their yaw direction to orient the axis of rotation with the dominant wind direction. In this project, wind turbine performance in the presence of yaw was specifically neglected.
- Constant pitch / constant diameter wake. Real wind turbine wakes are complex and self-interacting, and a comprehensive aerodynamic treatment of wind turbine performance will involve a process to discover the “real” trailing wake

geometry - a task which is computationally expensive, and results in something known as a “free wake model”. Free wake models typically feature “rolled up” wakes, especially near the outer circumference, and may have different wake pitches in the near and far wakes. Wake expansion was also neglected - the process of extracting energy from moving air reduces its kinetic energy, which by definition reduces the wind speed through the blades. This has the effect of causing the wake behind the blades to expand so that mass is conserved. This expansion was not modelled in the current implementation, an omission which slightly reduces the accuracy of the influence exerted by the wake on the wind turbine. The wake model used is considerably more accurate than a fixed-pitch wake model, however: together with the strength of the shed circulation carried by the wake, its pitch is the single most influential factor determining the wake’s influence on wind turbine performance. A wake pitch based on the maximum circulation on the rotating wind turbine blade is used and presented, and results in high- $\lambda$  wind turbine performance figures very close to reality.

- Similarity of flow conditions at low and high wind speeds. A very important measure of the performance of small wind turbines is the “cut-in” wind speed, that is, the lowest wind speed at which measurable power is produced. The cut-in wind speed of the “real” wind turbine used as a benchmark in this project (the 5 kW turbine operated by the University of Newcastle) is around 3 m/s, with a tip Reynolds number on a stationary rotor of around 15,000. This tiny Re means that the “true” aerodynamic conditions experienced by the blade at low wind speeds are extremely viscous, with the measured pressures and blade loads being considerably different to those calculated by the panel method (the predictive accuracy of which only matches reality for high-Re flows). So, a major assumption is made that, by optimising starting performance at relatively high Reynolds numbers (using a constant wind speed of  $U_0=10$  m/s and a “low” tip speed ratio of  $\lambda=3$ ), the enhancements in performance will be reflected at low wind speeds (and the cut-in wind speed) and low(er) tip speed ratios - again essentially ignoring viscous effects.

A three-dimensional aerodynamic solver was chosen because wind turbine blades are, in reality, complex three-dimensional shapes. Three-dimensional wings (both ro-

tary and otherwise) are only accurately approximated by two-dimensional aerofoils when they are very straight (untwisted and untapered), very high aspect ratio and have a constant cross section. The disadvantage of not having high- $\alpha$ , low-Re lift and drag data for a diverse range of two-dimensional aerofoils is thus (almost) a blessing in disguise, since it encourages the pursuit of three-dimensional aerodynamic solutions, which may be significantly different to the combined solutions of extruded two-dimensional sections assembled along the blade's span (which is the process at the heart of the ubiquitous blade-element method). In fact, cross-sections of real blades perform *better* than the aerofoils which they resemble, particularly at the high- $\alpha$  conditions encountered near the blade hub - a region where significant starting torque is generated.

As for the representation of a wide range of possible blade shapes, one geometric technique stands out immediately: B-spline curves and surfaces. As will be shown in the following chapters, a highly-accurate representation of a two-dimensional aerofoil can be constructed from a B-spline curve stored as just seven X-Y control points. This fact is remarkable when it is considered that aerofoil coordinate data is routinely recorded by 60 or more surface co-ordinates. A B-spline curve representation can thus be stored much more compactly than raw surface co-ordinates, a fact which has major ramifications on computer storage, evolution efficiency and aerofoil smoothness. The proper handling of these three factors is a crucial pre-requisite for the effective evolution of three-dimensional aerodynamic shapes, and B-spline curves offer a solution to all three. And as will be shown, creating a complex B-spline *surface* from just a few B-spline section curves is both simple and powerful, and makes possible extremely compact representations of entire wind turbine blade surfaces which, with a little manipulation, can be discretised and fed into our aerodynamic solver.

The result of all this is a computational technique for “discovering” wind turbine blade designs which perform well both during starting and at peak power production.

The outline of this thesis is as follows:

- **Chapter 1** is this introduction
- **Chapter 2** contains a review of the literature pertinent to the construction of the current computational optimisation method. This includes coverage of material on wind turbine performance, a general overview of aerodynamic optimisation

with special emphasis given to low-speed aerodynamic flows, and a selection of material dealing with the field of Evolutionary Computation and how it can be applied to engineering optimisation problems.

- **Chapter 3** introduces the use of bi-cubic B-spline surfaces as a convenient, flexible, accurate and compact way of representing the complex three-dimensional geometry of wind turbine blades. As an adjunct, a novel synthesis of an evolutionary optimisation technique and a two-dimensional B-spline curve representation is presented which results in a very useful aerofoil curve-fitting application.
- **Chapter 4** explains in detail a three-dimensional, first-order aerodynamic panel method solver which may be used to efficiently evaluate the performance of *general three-dimensional blade surface geometries*. A simple iterative wake model is also presented. Through validation of the method against performance measurements (and third-party computational performance calculations) of real wind turbine blades it is shown that, even with the use of many simplifying assumptions, this general aerodynamic solver offers good performance predictions, especially at moderate-to-high tip speed ratios.
- **Chapter 5** outlines the “starting problem”, and establishes the methodology by which a panel-method calculation of instantaneous aerodynamic torque at a low tip speed ratio, together with a finite-element calculation of blade second-moment of inertia, can be used to provide a relative measure of wind turbine starting performance.
- **Chapter 6** describes the Differential Evolution technique and its use as a multi-objective, real-valued object vector optimiser. A way to represent bi-cubic B-spline blade surfaces as a (compact) real-valued one-dimensional vector of geometry parameters is presented, as is a method for “evolving” an initially randomly-generated pool of these into a “Pareto optimal set” of *new* optimal blade designs using the DE optimiser. Results of a computational experiment employing this technique are offered which include a novel Pareto set of blade geometries, designs which span the gamut of exceptional starting performance/poor power production to poor starting acceleration/very good power production.

All of the computer programs coded as part of the current project were written by the author, except where noted. Appendix C contains an outline of the code.



# **Chapter 2**

## **Literature Review**

### **2.1 Wind turbine performance analysis and prediction**

#### **2.1.1 The small wind turbine starting problem**

The “starting” period of wind turbines - the time during which the wind accelerates the turbine rotor from rest to rotational speeds where power production can take place - can be a significant portion of the operational life of small wind turbines. Very little research has been undertaken, however, which would allow a wind turbine blade designer to understand how to maximise starting performance so as to reduce this non-power-producing period. Small wind turbine blades have traditionally been designed for optimal power extraction - an operational condition not directly influenced by the turbine dynamics prevalent during starting. Power production is completely independent of the inertial properties of the system’s rotating components, and operational torques are typically several orders of magnitude greater than the “resistive” torque of the generator. Starting, on the other hand, is a dynamic process dependent on both the second-moment of inertia of the rotating components of the system and the small resistive torques imposed by the stationary/slowly rotating generator. The combination of rotational inertia and generator resistive torque presents a hurdle which must be overcome by the small aerodynamic torques generated by stationary wind turbine blades before the rotor/generator system can begin accelerating.

The first documented study of the starting performance of small wind turbines was carried out at the University of Newcastle by Ebert and Wood (1997). The University’s

Wind Energy Group operated a 5kW (5 metre diameter) upwind horizontal axis wind turbine that served as the experimental platform for the study. For the starting analysis the rotor axis was assumed parallel to the wind direction (zero yaw error). The experimental rotor included a pitching mechanism - a device connecting the blades to the turbine's hub which pitched the blade into the wind whilst stationary, with the intent of reducing the large angles of attack and improving starting torque. As rotor speeds increased, the blades were gradually pitched back to their "operating" position by the mechanism. Analysis of their experimental data showed that the starting sequence consisted of a brief initial acceleration from rest, followed by a substantial period of time during which the rotor's acceleration slowed appreciably - a result of the flow impinging on the blades at unfavourably large angles of attack, even with the use of the pitching mechanism. These angles decreased slowly as the rotor's speed gradually increased. Once a critical tip speed ratio had been reached, angles of attack favourable to high sectional lift:drag ratios were achieved, which resulted in a sharp increase in the aerodynamic torque produced by the rotor. At this point the starting sequence ended, with the rotor suddenly accelerating to power-producing speeds. This starting behaviour was further studied by Mayer et al. (2001) using a different turbine platform with similar 2.5 metre blades. The wind speed and the rotor's angular velocity were logged for a series of experimental starts.  $U_0$  over the entire course of the experiments was recorded at around 8 m/s - a value considerably higher than the cut-in wind speed of the studied 5 kW platform (which was around 3 m/s). The one independent variable modified during these experiments was the applied, constant, pitch angle of the blades via an adjustment of the pitching mechanism connecting the blades to the rotor hub. For each experimental start the rotor was brought to a halt, the blades fixed to the required pitch angle and the rotor released and allowed to accelerate with no generator loading. The data was used to test a computer program capable of predicting starting performance. Here, the angular speed of the unyawed rotor was calculated by initialising a standard Blade Element/Wake Momentum rotor model at time  $t = 0$  and a stationary rotor ( $\Omega=0$ ), then applying the actual experimental wind speed data. The angular acceleration of the rotor provided by the code was taken as the aerodynamic torque divided by the second moment of inertia of the rotor/generator system (which is dominated by the torque contribution from the blades). This was numerically integrated over time using a fourth-order Runge-Kutta method.

Mayer et al. (2001) provides examples of starts for each pitch angle studied. The results demonstrate that the method is capable of reproducing the main features of the starting performance of a given HAWT design, provided that the pitch angle is larger than zero and that the aerofoil sections from which the modelled rotor is constructed have lift/drag data available for low Reynolds numbers and very high angles of attack. Since a stationary small wind turbine experiences angles of attack approaching ninety degrees (and for negatively-twisted tip sections, exceeding ninety degrees) and in a 3 m/s wind experiences a tip Re on the order of 15,000, this last requirement is a major problem for researchers hoping to use the Blade Element method for automated aerodynamic optimisation. Experimental aerofoil performance data are difficult and expensive to obtain and there is very little data available for low-speed aerofoils at high angles of attack. The lift and drag datasets that *are* available have been accumulated over the years for a wide range of aerofoils used by the aviation industry (for the most part), whose operating requirements specify large Reynolds numbers and comparatively mild angles of attack - usually within the bounds of  $\pm 20^\circ$ . Wood (2001) describes a method to approximate the lift and drag force on aerofoils at high incidence. The method was used to predict the cut-in wind speed for a small wind turbine using the blade element method. Comparison of prediction results with available empirical data suggested that aerofoils at the very high angles of attack experienced by stationary turbine blades behave in a *generic* way, and that simple formulas for lift and drag are potentially applicable to a wide range of aerofoils at high incidence. The method seems to significantly underpredict the aerodynamic force at high incidence, however, and the author concluded that more empirical aerofoil data at high incidence is required before the method can be improved upon.

### 2.1.2 Low-speed aerofoils for wind turbines

A most comprehensive source of low-speed aerofoil geometries and performance data is provided by Selig et al. (2001) at the University of Illinois at Urbana-Champaign (UIUC). Professor Selig has attracted a wide range of committed researchers, professionals and hobbyists who continue to help him accumulate excellent experimental lift and drag data for aerofoils used in powered and unpowered radio-controlled model aircraft and, more recently, small wind turbines. SoarTech Publications distribute com-

pendiums of aerofoil coordinates from the UIUC project, the most recent of which is Lyon et al. (1998). An example of the work performed by UIUC in tackling the problem of designing aerofoils for small HAWTs is provided by Giguère and Selig (1998), who report on the development of their SG604x set of aerofoils. These aerofoils feature enhanced lift-to-drag ratios for low Reynolds numbers, and were specifically designed for use on small, variable-speed HAWT blades. Lift and drag curves covering Reynolds numbers from 100,000 to 500,000 derived from experimental wind tunnel tests are presented, along with x-y coordinates for all four aerofoils. It should be noted however that the lower Re for this data is still substantially greater than the tiny tip Reynolds numbers experienced by stationary small wind turbines, and the tabulated angles of attack do not exceed  $\pm 25^\circ$ . Early catalogues of low-speed aerofoils for wind turbine applications are provided by Miley (1982), Althaus (1980) and Althaus (1996).

Although it is generally recognised that finite-thickness aerofoils perform better than thin, cambered plates for a wide range of angles of attack and Reynolds number, Laitone (1996) showed that a thin plate with 5% camber had *higher* lift:drag ratios and developed more lift across a wide range of angles of attack when the Reynolds number was reduced to below  $7 \times 10^4$ . Such small Reynolds numbers (and lower) are commonly encountered by small wind turbine blades during starting, which raises the possibility that the use of very thin shapes with camber may be warranted to improve starting torque, particularly at the blade tips. A contrasting design philosophy was prompted by Sato and Sunada (1995), who showed that adding a *blunt* trailing edge to finite-thickness aerofoils surprisingly lessened their low-Re drag burden by preventing the premature formation of large trailing-edge separation bubbles. This result is perhaps fortuitous, since blade cross-sections approaching the root are typically quite thick to allow attachment to the generator hub, and may lose their sharp trailing edge to accommodate the attachment-piece geometry, which is usually of rectangular or circular section.

### 2.1.3 Rotary wings

Most of the published experimental work regarding rotary wings concerns helicopter rotors, with the “hovering” maneuver being somewhat similar to the mode of operation of wind turbines, particularly because the wake shed from the rotor is reminiscent of

the coaxial helical vortices shed by wind turbine blades and subsequently convected downstream. Interesting experimental data of helicopter rotors in hover is provided by Caradonna and Tung (1981) and Branum and Tung (1997). The latter is particularly comprehensive and offers rotor geometry descriptions and detailed surface pressure data tables. These studies are valuable when validating rotary wing prediction codes because of the dearth of similar information available for wind turbines and the similarity in flow generated by hovering helicopter rotors. Possible validation data is also provided by Wolfe and Ochs (1997). They report on a study that compared the predictions of a commercially available CFD code with wind tunnel tests of two common aerofoil sections used in HAWT designs.  $C_p$  vs chord data are provided for S809 and NACA0012 aerofoils at various angles of attack and Reynolds number of  $1.5 \times 10^6$ . Other examples of techniques which solve the full Navier-Stokes equations are provided by Hsiao and Pauley (1999) for marine propeller flows and Xu and Sankar (2000) for flows about wind turbines.

A review of the development of HAWT technology from the 1970's up until the early 1990's is provided by Hansen and Butterfield (1993). Conlisk (1997) offers a recent review of the aerodynamics of helicopter rotors which also serves as a general introduction to rotary wing aerodynamics.

#### 2.1.4 Blade Element/Wake Momentum Method

The BEM is computational method which is widely used for performance predictions of rotating wings: marine/aircraft propellers, helicopter rotors and wind turbine blades. Its general premise is quite simple, and involves modelling the blade/propeller in question as a span-wise series of "extruded" two-dimensional aerofoil sections. Each of these quasi-three-dimensional is called a "blade element", with the aerodynamic properties of the blade element known *a priori* from tables of empirical lift and drag vs Re and  $\alpha$  for the aerofoil from which it is made. Better predictive accuracy is attained by approximating the "real" geometry by a more refined blade element model - a task which consists of decreasing the span-wise width of each element and increasing their number. The contribution of the trailing wake is handled by an iterative control-volume calculation which balances the momentum in the column of air passing through an "ideal" rotor with rotor thrust and torque (a calculation similar to that pre-

sented in appendix B.1). The method relies on the assumption that a good aerofoil is also a good blade element, and it has been repeatedly shown that the assumption holds when the computed flow encountered by blade element aerofoils is “optimal” - conditions requiring low angles of attack and high Reynolds numbers. These conditions allow the method to accurately predict the optimal power of wind turbine blade designs constructed from existing aerofoil sections. In fact, the technique is so successful that it is directly responsible for many of the commercial propeller/blade designs produced over the last century. Spera (1994) contains a comprehensive treatment of the BEM for the analysis of wind turbine performance. Buhl et al. (1997) offer a comparison of three HAWT aerodynamic prediction codes: Bladed (Bossanyi (1998)), WT\_Perf and YawDyn (Hansen and Cui (1989) and Hansen et al. (1990)). All are based on the Blade Element/Wake Momentum method, with each containing one or more additional models catering for hub/tip loss modeling, wind shear, tower shadow and dynamic stall. The three methods were used to predict the performance of a test HAWT rotor comprising two simple theoretical blades. The test blade had no twist or taper, used a single aerofoil with zero drag and lift with constant slope  $2\pi$  and ran at a constant 60 r.p.m. The study concluded that the main differences in the aerodynamic predictions given by the codes were due to the different tip-loss correction factors during the axial-induction factor calculations, as well as the dynamic stall models used.

Bladed is a commercial HAWT performance prediction code based on the Blade Element/Wake Momentum method which contains additional models for unsteady aerodynamics and dynamic stall. Built on top of the numerical aerodynamic code is a host of additional features, including structural dynamics, power train dynamics and closed loop control modules. Bossanyi (1998) gives a comprehensive review of the Blade Element/Wake Momentum method as well as detailed equations for the aerodynamic components of the code.

### **2.1.5 Panel Methods**

Hess (1986) provides a history of the development of singularity panel methods for inviscid non-lifting and lifting flows in two and three dimensions. Fundamental early work on three-dimensional non-lifting flows by Hess and Smith (1966) employed a piece-wise constant singularity (source) distribution over a body’s surface. The first

methods were concerned with finding the source density distribution,  $\sigma(p)$  by solving a Fredholm integral equation of the second kind over the boundary surface,  $S$

$$2\pi\sigma(p) - \iint_S \frac{\partial}{\partial n} \left( \frac{1}{r(p,q)} \right) \sigma(q) dS = -\vec{n}(p) \cdot \vec{U}_0 \quad (2.1)$$

where  $\vec{n}(p)$  is the unit normal at a point  $p$  on the surface and the kernel  $\frac{\partial}{\partial n} \left( \frac{1}{r(p,q)} \right)$  is the outward normal velocity at the point  $p$  due to a unit point source at the point  $q$ .

Potential flow formulations are inherently inviscid (or, more properly, model attached lifting flow at infinite Re via the trailing-edge pressure-equality (Kutta) condition), which poses a challenge for the source-only singularity methods if the goal is the accurate modelling of lifting flows. This is because, for closed bodies modelled in such a way, the calculated circulation around the body will be zero. Hess (1974) details the addition of a vorticity distribution into the source singularity panel method so that lifting flows could be accommodated. Hess (1986) provides the basic integral equation that has been used to represent the potential field for lifting flows

$$\phi = \iint_S \left[ \frac{1}{r} \sigma + \frac{\partial}{\partial n} \left( \frac{1}{r} \right) \mu \right] dS \quad (2.2)$$

The dipole distribution  $\mu$  accounts for the vorticity in the lifting model which is absent in the source-only formulation, Equation (2.1).

A very popular source/doublet panel method implementation for lifting flows around arbitrary geometries was formulated by Morino et al. (1974). This Green's Function approach is extended to the flow around HAWTs by Preuss et al. (1980), who give a convenient integral equation for the velocity potential on the surface of any general lifting configuration

$$2\pi\phi = - \iint_{S_R} \frac{\partial\phi}{\partial n} \frac{1}{r} dS_R + \iint_{S_R} \phi \frac{\partial}{\partial n} \left( \frac{1}{r} \right) dS_R + \iint_{S_W} \Delta\phi \frac{\partial}{\partial n_1} \left( \frac{1}{r} \right) dS_W \quad (2.3)$$

The first integral on the right hand side is modelled as a piece-wise constant distribution of source singularities on the lifting body's surface ( $S_R$ ). The following two integrals represent a similarly piece-wise constant doublet distribution over the body's surface and over the surface of the trailing wake ( $S_W$ ), respectively.  $n_1$  is the unit-normal on side 1 of the wake (the side adjacent to the blade suction surface). The Dirichlet boundary condition  $\left\{ \left( \frac{\partial\phi}{\partial n} \right)_k \right\}$ , represents the impervious solid boundary and is calculated as the component of the unperturbed impinging fluid velocity normal to

the body's surface at each collocation point,  $k$ . It is included in the discretised integral equations to give a linear system of equations

$$[\delta_{hk} - c_{hk}] \{\phi_k\} = [b_{hk}] \left\{ \left( \frac{\partial \phi}{\partial n} \right)_k \right\} + [f_{hm}] (\Delta \phi)_m \quad (2.4)$$

which is subsequently solved for the scalar value of  $\phi$  at each collocation point on the lifting body. The velocity potential distribution  $\{\phi_k\}$  is then numerically differentiated to give the flow velocity on the surface of the body and, thence, the pressure via the Bernoulli equation. A method for calculating the coefficients  $c_{hk}$  and  $b_{hk}$  (the doublet and source strengths respectively on the body panels) and  $f_{hm}$  (the doublet strength on the wake panels) is provided by Morino et al. (1974). Newman (1986) provides another, arguably more convenient treatment for source and doublet strength calculation over quadrilateral panels in three-dimensional geometries. Morino et al. (1974) also details a *very* convenient panel vertex coordinate system representation. This, together with other methods described by Morino et al. (1974), Preuss et al. (1980) and Newman (1986) form the basis of the potential flow numerics used in this project and are explained in detail in later chapters. As will be seen, their selection as the basis of the current aerodynamic solver results in an evaluation routine which is fast and accurate. The constant singularity expressions by Newman (1986), in particular, were chosen because the formulation allows the use of *non-flat* panels, an invaluable feature which considerably simplifies the discretisation of three-dimensional blade geometries. Ar-suffi et al. (1993) expand on a similar “Morino” method to allow the treatment of unsteady flow around yawed HAWT rotors. They provide favourable comparisons of their method with experimental results. Kinnas and Hsin (1992) employ a very similar method to calculate the unsteady potential flow around marine propellers. They also explain a very interesting “iterative Kutta condition” method to achieve pressure equality at the propeller trailing edge. Katz and Plotkin (2001) provide perhaps the best introduction to two- and three-dimensional panel methods available. They use the source and doublet singularity coefficient calculations of Hess and Smith for their panel calculations and provide detailed implementation details of a variety of flow solution methods. The geometries employed to explain their solution techniques are aerofoils and wings (and rotors) and they offer comprehensive implementation details. Particularly valuable are their analytic solutions to common symmetric aerofoil geometries. These have been employed as validation studies of the panel method solution routines

in this project.

Maskew (1982) compared a low-order method with higher order solution methods and found that, for comparable panel densities, their low order method gave good agreement with standard test problems. The significantly reduced computing costs associated with piece-wise constant source and doublet distributions makes these methods attractive solution techniques for inviscid, subsonic problems, especially when their accuracy is on par with more expensive methods.

## 2.2 Aerodynamic optimisation

In general, the pressure at the surface of an aerodynamic object is very sensitive to small changes in the surface's shape: the domain of aerodynamic optimisation provides great scope for designing shapes which perform in a particular, desirable way, whether they be aeroplane wings with high lift:drag ratios, helicopter rotors which make a minimum of noise or buildings that can survive cyclonic winds. There are two broad aerodynamic design optimisation methodologies: direct shape optimisation and inverse shape optimisation. Direct shape optimisation, as the name implies, is concerned with directly specifying the shape of physical surfaces with the expectation that this change will provide the solution to some design challenge. By their nature, direct methods are experimental and iterative: take an existing shape; measure its performance experimentally; vary one or more parameters that describes the shape; repeat until the design performance goal is reached. The experimental measurement of performance often involves testing a physical model of the shape in a controlled fluid flow, such as a wind tunnel, but increasingly also involves numerical simulation experiments using CFD codes. The iterative nature of direct methods and their dependence on expensive physical and/or computational evaluation techniques makes them slow and cumbersome and, therefore, unattractive when resources such as time, personnel, money or hardware are scarce. Their attraction lies in the fact that if a direct optimisation run converges to a meaningful result then that result will provide a definitive answer to the design problem. Inverse methods take the other side of the argument: "let's find the aerodynamic characteristics necessary, such as a desirable pressure or velocity distribution over the span of our aeroplane wing and *then* look for a geometry that will deliver it.". Inverse methods have the distinct advantage of requiring far fewer

CFD code iterations than direct methods. An inverse design practitioner, though, needs to address two major issues: how is it possible to tell what an ideal pressure distribution really is?; and what happens if the inverse geometry specification algorithm can't find a geometry to give that distribution? Similarly, what if the geometry it *does* come up with is unrealistic?

### 2.2.1 Two-dimensional geometries

Selig and Maughmer (1992) apply a Conformal Mapping approach and the Eppler code (Eppler and Somers (1980)) to the problem of multipoint inverse aerofoil design. Harvey (1999) used Simulated Annealing to find optimal pressure distributions over two-dimensional turbomachinery blades. The object vectors consisted of parameters describing B-spline representations of compressor blade aerofoils. A quasi three-dimensional Navier Stokes CFD solver was used as the evaluation routine, and fitness was assigned by an objective function that rewarded blades with minimal thickening of the boundary layer, and thus minimal energy loss. Successful optimisation runs which produced optimal blades for a single angle of incidence prompted a subsequent “multipoint optimisation” study which evaluated each object blade over a range of angles of incidence. The method was capable of producing blade designs whose performance was consistent over a range of flow directions, but this robustness came at a cost of decreased peak performance.

Obayashi and Takanashi (1996) describe an interesting application which employs optimal target pressure distributions for an inverse aerofoil design problem. The authors point out that the task of actually specifying an optimal pressure distribution is typically left to the skills of an experienced aerodynamicist. The systematic optimisation of pressure distributions via computational means has suffered due to both the non-robust nature of gradient based optimisation algorithms and the non-linear and sometimes highly non-convex nature of aerodynamic object functions. Evolutionary algorithms, in general, are inherently robust optimisation methods. Obayashi and Takanashi (1996) chose to parameterise a two-dimensional pressure distribution as a closed B-spline polygon whose parameters form the Genetic Algorithm (GA) object vector. The problem was posed as a constrained minimisation problem, with the goal to minimise drag for a given lift and pitching moment subject to various con-

straints such as limitations on the extent and shape of the pressure curve and minimum aerofoil thickness. They present experimental case studies for two aerofoil and one wing design tasks. Vicini and Quagliarella (1997) apply a genetic algorithm to the problem of transonic aerofoil design. They explore various configurations of GA and study the applicability of this evolutionary approach for both inverse and direct design. They emphasise that an evolutionary approach is very amenable to multipoint optimisation. Jones et al. (2000) apply a multiobjective GA to an interesting direct aerofoil design problem. Their work highlights the flexibility of an evolutionary optimisation approach: a closed B-spline aerofoil shape was represented by twenty two-dimensional control points. A *random* binary representation of the coordinates of these control points became the initial parameter vector for a population of aerofoil shapes, which presented the evaluation routines (XFOIL for aerodynamic analysis and WOPWOP (Brentner (1986)) for aeroacoustic analysis) with some very unconventional initial foil geometries. A Pareto optimal set of aerofoils was generated that represented a trade-off between aerodynamic and aeroacoustic performance. The authors point out that the flexibility inherent in evolutionary optimisation techniques is often a double-edged sword: the same powerful searching mechanisms used to explore parameter space will indiscriminately exploit weaknesses in the evaluation method and shortcomings in the problem formulation. For this reason, they state that their (admittedly weird) optimised aerofoil shapes should be viewed as interesting starting points for further design refinements, rather than as final aerofoil designs.

Drela (1989) and Drela (2001) describe XFOIL, a very popular, freely available and widely used two-dimensional aerofoil design code. At its heart it is an inviscid 2D vorticity/source panel method. Additional modules model viscous flows (with and without separation) and compressibility. A fundamental feature is its full-inverse and mixed-inverse design capabilities - the user can specify a surface speed or  $C_p$  distribution along a full or partial aerofoil contour and the code will endeavour to find a matching surface geometry. Whilst XFOIL is an excellent interactive aerofoil design tool, it requires a degree of operator skill to produce meaningful aerofoil performance predictions: a typical interactive design requires the operator to “tweak” their aerofoil over a range of parameter iterations to avoid meaningless non-convergent designs. The successful convergence of viscous flow solutions, in particular, is a delicate balancing act that requires well-specified geometries, low angles of attack and reliable seed

solution inputs to the iterative solver. These reasons preclude the use of XFOIL for the automated design of small HAWT blades for optimal starting performance, which typically include flows at very high angles of attack and very low Reynolds numbers.

### **2.2.2 HAWT optimisation**

Wind turbine optimisation efforts have to date focused on optimal power extraction (not starting performance), with work mostly devoted to the generation and selection of appropriate low-speed, high-lift, two-dimensional aerofoil designs. Giguère and Selig (1997) use a binary-coded Genetic Algorithm coupled with an inverse design code to design small stall-regulated wind turbine rotors for optimal power capture and adequate high speed stall. The inverse design code selects an appropriate two-dimensional aerofoil section for each radial span position and finds lift and drag data for each blade shape by interpolating the tabulated lift and drag data for each blade shape. The GA generates root/tip chord and twist distributions which are evaluated by the inverse design code.

Fuglsang and Madsen (1999) optimise a 1.5MW HAWT rotor using a combination of gradient-based optimisation methods. To achieve their goal of designing a rotor with minimum total lifetime cost of energy, they employed a Blade Element/Wake Momentum aerodynamic code as their aerodynamic objective function, along with an aeroelastic solver for fatigue loading and noise production. The object vector consisted blade shape parameters (chord and twist distributions), aerofoil lift and drag distributions and regulation variables (rotational speed and tip pitch angle). No aerofoil sections were specified in the method. Rather, ideal lift and drag distributions were sought which could later be fed into an inverse design code to achieve the required aerofoil shapes.

## **2.3 Evolutionary Optimisation**

Why choose an Evolutionary approach to optimisation? Fogel (1999) gives an easy introduction to Evolutionary Computation (EC), with special emphasis on the use of EC techniques as optimisation methods for the solution of engineering problems. His “Advantages of Evolutionary Computation” highlight why an EC algorithm may be

the best choice of optimiser for a broad range of problems:

- **Computational simplicity.** Unlike many purpose-specific heuristic optimisation algorithms, EC techniques require very little information about a candidate solution to give it a relative “fitness” ranking.
- **Broad applicability.** Virtually any problem that can be characterised by one or more objective functions which return a single (scalar) or multiple (vector) fitness score is amenable to EC optimisation.
- **Outperform classic methods on many problems.** Classic (gradient-based) optimisation techniques are usually designed to efficiently solve strongly convex test problems. Engineering problems, however, are characterised by non-linear constraints, objective functions that are multimodal and/or not concerned with least-square error, involve non-stationary conditions and incorporate noisy observations. The application of gradient based methods results to such problems, in many cases, results in rapid convergence to local optima.
- **Potential to use knowledge and hybridize with other methods.** The simplicity of EC algorithms also makes them very flexible. Problem-specific solution techniques can be incorporated into virtually any EC algorithm. If a particular problem contains an objective function that has been shown to be efficiently solved by an inexpensive and specialised classic optimisation algorithm, then that technique can be incorporated into a customised optimisation component (a mathematical operator, say) that can improve the performance of the EC algorithm.
- **Parallelism.** Evolution, either biological or computational, is a highly parallel process. A typical EC run involves the evaluation of thousands of independent individual solutions. A computational platform that has  $N$  computational nodes will (ideally) be able to process almost  $N$  times more individual solutions in a given time than a platform with only one computational node. This makes the design of distributed EC algorithms an almost trivial process and makes possible the solution of large, computationally expensive problems by utilising relatively inexpensive commodity computer hardware. Tomassini (1999) gives a general review of parallel and Distributed Evolutionary Algorithms.

- **Robust to dynamic changes.** EC optimisation algorithms typically maintain a population of individuals, each of which may exhibit a range of survival strategies for maintaining their hold in a competitive fitness landscape. As in biological evolution, this variability between the characteristics of individual members in a particular “species” is beneficial if the fitness landscape suddenly changes, for example at the onset of a coming ice-age or if, say, in the middle of a month-long optimisation run it is discovered that one of the defining assumptions used to model the “fitness” of the evolving population is wrong, necessitating a change in the objective function used. Classical optimisation methods have trouble with such situations since they, by their nature, search for the “true optimum” for any given problem, and usually need to be reinitialised to begin a run for the new objective function(s) since the parameters which describe an optimal solution for one problem may be very far away from those which describe another.
- **Capability for self-optimisation.** Most optimisation techniques, whether classic, EC, or otherwise, rely to a greater or lesser extent on user-specified settings to control how the optimisation algorithm behaves. These settings are usually found by repeated experimentation. EC techniques have the added advantage of being able to treat their run settings like any other object variable and evolve them alongside their populations of potential problem solutions.
- **Able to solve problems that have no known solution.** EC techniques have the potential to be able to “solve the problem of how to solve problems”. The flexibility and robustness of EC algorithms make it possible for them to explore areas of problem space that are inaccessible to the sometimes brilliant (yet rigid and fragile) creations of the human mind. “Knowledge discovery” takes place when we are able to remove the requirement of problem-specific information from our objective functions, and this allows the solution of intractable problems that are resistant to current optimisation methods.

### 2.3.1 Genetic Algorithms

Holland (1975) is generally credited with the creation of the Genetic Algorithm, and Holland (1992), gives an entertaining introduction to genetic algorithms and how they

relate to processes of biological evolution. He points out that the success of evolutionary computation over other search and optimisation techniques, such as simple hill climbing, is due to EC's ability to make use of *partial solutions* to problems at each iteration. A solution's chromosome (object vector) may contain many strategies to solve a particular problem. As long as the relative fitness of an individual allows it to survive the "selection" step during a particular iteration, its genes will form the building blocks for the next generation, even though the strategies they encode may, at this point, be sub-optimal. Wayner (1991) gives a similar popular introduction. Both focus on basic Genetic Algorithms with bit-strings as their object vector encoding method. Cerrolaza and Annicchiarico (1999) describe an application of Genetic Algorithms to a problem of three-dimensional truss shape optimisation. The truss shape is modelled as B-spline curves and Finite Element and Boundary Element models are built from these to be evaluated by FE and BE objective functions. Diveux et al. (2001) used a genetic algorithm to design a horizontal axis wind turbine system for Mediterranean sites (in contrast to most other wind turbine blade designs, which the authors state are "well designed for northern European sites"). The turbine system was encoded as an object vector containing eight "Principal Parameters": number of blades; rotor diameter; hub height; (constant) rotational speed of the rotor; nominal power; design wind speed; a choice of stall or pitch regulation; and a choice of constant or variable speed asynchronous generators. No mention is made of the object vector encoding method, which suggests that some sort of real-valued parameter to bit-string "gene" encoding/decoding process is used, and that a "classic" single-objective Genetic Algorithm was the optimisation engine. The optimisation objective was to find a HAWT system which minimised the "annual cost of electricity" for a typical Mediterranean site. A single-equation analytical function was used to estimate the aerodynamic performance of the turbine rotor, with  $C_L$  and  $C_D$  data for a single aerofoil section (a NACA 63·4) being the only aerodynamic data considered in the calculation. The authors conclude that this optimisation method is capable of successfully designing a wind turbine system suitable for the particular needs of a Mediterranean site, that such a Mediterranean turbine is "smaller" with "larger power parameters" than turbines designed for northern European conditions, and that energy "is cheap" when produced by a turbine of their design.

The defining feature of classic Genetic Algorithms is their reliance on bit-strings

as their object vector encoding method. Many practical optimisation problems, however, have as their goal the optimisation of real-valued parameters, with the resolution of each real-valued parameter dependent on number of bits devoted to it. If a 16-bit number is required to represent a parameter, and if, say, 10 parameters define a practical optimisation problem, the the object vector will be a bit string of dimension 160. This seems unnecessary, especially since object vector length has a direct and deleterious impact on the performance of Evolutionary Algorithms - increasing object vector length necessitates an increase in population size so that genomic diversity is maintained. Which raises the question: why can't real-valued problems just be represented by real-valued object vectors?

### 2.3.2 Evolution Strategies

Schwefel (1995) and Bäck (1996) describe Evolution Strategies, an Evolutionary Algorithm that works with real-valued object vectors. The first evolution strategy was developed by Schwefel and Rechenberg as a purely practical optimisation method and was employed during experimental pipe and nozzle optimisation projects. The optimisation parameters in these cases were discrete system variables: eg. diameter and length of pipe segments in their experimental set up. This gave rise to the (1+1) ES, a method that utilises only two population members (a parent and a child) per generation. *Mutation* in the (1+1) ES (and in all later variants) is the main evolutionary operator, and involves the addition of small Gaussian-random perturbations (with mean zero and standard deviation  $\sigma$  - the *step size* ) to the parent's parameters to form a new child object vector. The child vector replaces the parent vector if the child's fitness exceeds that of the parent. Later variants introduced an evolving *population* of individuals: The  $(\mu, \lambda)$  ES completely replaces the old generation of  $\mu$  parents with  $\lambda$  offspring (with  $\lambda \geq \mu$ ), whilst the  $(\mu + \lambda)$  ES replaces only those parents in the current population whose fitness is inferior to the best offspring. The magnitude of  $\sigma$  is an important issue: mutations should be large enough at the beginning of a run so that the parameter space is effectively searched, but smaller at the end so that important optima in the fitness landscape are not passed over. Bäck and Schwefel (1995) addressed this by introducing step size *self-adaptation* into the ES by expanding  $\sigma$  into a vector with the same dimension as the parameter vector. This step size vector was then evolved alongside

the population. Other techniques that improve on the convergence rate of the step size self-adaptation process have been developed.

One such improvement, the *Covariance Matrix Adaptation* of Hansen and Ostermeier (1996) was employed by Hampsey and Wood (1999) in their preliminary small turbine blade optimisation work. Here, the BEM code used in Mayer et al. (2001) to predict the starting performance of small wind turbines was used as the basis of an ES objective function. The wind turbine blade being optimised was modelled by ten blade elements. One aerofoil section was used for the blade (SD7062, a low-speed aerofoil from the Selig (2000) library). Chord and twist values for each blade element made up the optimiser's object vector. The initial population for the optimisation run started off with constrained *random* chord and twist distributions. The optimiser typically converged to a solution after about fifty generations. The twist distribution for the evolved blade were remarkably similar to current 5kW blade designs.

As well as reporting on the effectiveness of an Evolution Strategy for the solution of two standard multimodal test problems, Bäck et al. (1999) also tackle the problem of inverse aerofoil design. Here, they start with a Bezier spline description of a NACA 0012 aerofoil. The parameter vector to be optimised consisted of the position of the 12 Bezier points that define the curve. The objective function makes use of a Finite Element Navier Stokes solver and a target pressure distribution for a NACA 4412 aerofoil. The problem is defined as a minimisation problem, with the fitness of each individual being the difference between the computed pressure and the target pressure at discrete points along the pressure curve.

### 2.3.3 Differential Evolution

Storn and Price (1995) and Storn (1996) introduce the method of Differential Evolution (DE). Although no convergence proof for DE yet exists, it has proven to be a very effective parameter optimisation technique for a range of standard test problems (Storn and Price (1995)), as well as real-world optimisation problems. The three main advantages of DE over other EC techniques are:

- **Real-valued object vector encoding.** Discrete bit-strings, as used in classic Genetic Algorithms and other EC approaches, were created in an attempt to mimic the functioning of biological DNA. Most engineering problems are described by

continuous real variables, however. EC Algorithms that use real-encoded object vectors dispense with the cumbersome steps of translating between number systems.

- **Conceptual simplicity.** A “Weighted Difference” approach is used: two randomly selected object vectors are selected from the population, the vector difference of a randomly selected subset of their parameters is multiplied by a scaling factor (drawn from a Gaussian random distribution) and the resultant perturbation vector is added to a third randomly selected vector to form a new individual. This approach results in a minimum of control parameters that need to be specified (and tuned) by the operator.
- **Self organising.** Unlike the Evolution Strategies of Bäck and Schwefel (1995) Hansen et al. (1995) and Hansen and Ostermeier (1996), the “weighted difference” scheme used in DE does not require the algorithm to keep track of a separate mutation distribution. This further simplifies the algorithm whilst maintaining an efficient means of searching parameter space. It also avoids the associated memory and computational overhead, which can be a significant percentage of the overall run requirements for large populations and/or low-overhead objective functions.

### 2.3.4 Multiobjective Optimisation

The concept of “fitness” must be addressed when a problem is defined by more than one objective function. Veldhuizen and Lamont (2000) and Fonseca and Fleming (1998a) review methods for tackling Multiobjective problems using Evolutionary Algorithms. Perhaps the simplest approach is to use a weighted sums method. Here, multiple fitness scores are first normalised by a pre-defined standard weighting constant, and then summed to form a final aggregate fitness, which can then be used with single-objective Evolutionary Algorithms. The value of each weighting constant needs careful consideration in such a method to avoid incorrectly biasing the optimum solution in favour of one or more objective functions over another.

Marco et al. (1999) made use of a multiobjective genetic algorithm to design a range of aerofoils. Using a two-dimensional Euler solver they constructed two objective functions: one for a subsonic, high angle of incidence flow and one for a transonic,

low angle of incidence flow. Starting with a NACA 0012 aerofoil, they evolved a Pareto Optimal set of aerofoil shapes that span from high-lift shapes to low-drag shapes.

Wang and Sheu (2000) applied a weighted minimax Multiobjective DE to the problem of finding suitable model parameters for the accurate modelling of fermentation processes during the fed-batch production of ethanol.

Coello (2001) maintains a comprehensive Evolutionary multiobjective optimisation bibliography.

### 2.3.5 Pareto Optimality

The problems associated with scalarising multiple fitness scores has led to the concept of *Pareto Optimality* being favoured. Unlike the weighted sums scalarisation method and other *a priori* (Veldhuizen and Lamont (2000)) methods that attempt to find one “optimal” solution, algorithms that employ Pareto Optimality seek an *N-dimensional trade-off surface* from *N* objective functions. This is known as an *a posteriori* multi-objective decision method: at the end of an optimisation run the user is presented with a range of “non-inferior”, or uniquely-good solutions that offer a trade-off between the different (and often competing) objective functions. A major challenge encountered when implementing Pareto-based methods is how to get a *uniform and representative* Pareto surface. Many so-called niching, and fitness-sharing add-ons to the basic Pareto concepts have been proposed to try to deal with the severe depletion in genetic diversity that goes hand-in-hand with the strategy of selecting only the best individuals and culling the rest. Such strategies lead to clumping of individuals in discrete areas of the fitness landscape, which runs counter to the requirement that population fitness scores should be dispersed evenly over the Pareto surface so that the trade-off relationships between various objectives are smooth and clearly defined. Descriptions of fitness sharing and other approaches are outlined in Veldhuizen and Lamont (2000) and Fonseca and Fleming (1998a) and examples of their use are provided by Takahashi et al. (1998) and Chang et al. (1999). Pareto Optimality concepts are covered in more detail in Chapter 6.

Deb (1999) described the use of EAs for Multi-Criterion Optimization in engineering design. He points out the distinction between single-objective optimisation problems and multi-objective problems. Many interesting engineering problems have

multiple *conflicting* objectives that must be optimised during a run. This gives rise to the idea of a population of *non-dominated* optimal solutions rather than just a single optima. This population of solutions is known as the “Pareto Optimal Set”, and its members “Pareto Optimal solutions”. He describes the concepts of Pareto dominance and the Pareto Optimal front and also gives an outline of various EAs that make use of Pareto ideas. A selection of multi-criterion algorithms have been applied to a range of engineering problems.

Takahashi et al. (1998) used a real-valued multiobjective GA to design aircraft wings. The objectives to be achieved by the optimisation method span across numerous aircraft design principles, with the generated Pareto surface displaying a trade-off between: minimum aerodynamic drag; minimum wing weight; and maximum fuel weight, subject to a specified constrained minimum take-off lift and adequate structural strength. The aerodynamic evaluation routine computed two-dimensional lift and induced drag via a potential flow solver. They emphasise that fitness sharing is essential if a pareto surface representative of the problem space is desired.

Fonseca and Fleming (1998b) applied a Pareto-based multiobjective Genetic Algorithm to the problem of optimising a gas turbine engine controller. Eight performance indicators where used as the eight objective functions.

Chang et al. (1999) formulated a method of introducing Pareto Optimality ideas into Differential evolution and apply this new Multiobjective DE to optimising an automatic train controller. The method presents the operator with a range of optimal trade-off parameters based on the competing objective functions of safety, punctuality, ride comfort and energy consumption. These ideas are applied to the optimal design of small wind turbine blades in later chapters.

Cvetković (2000) reviews a number of Multiobjective Optimisation techniques for engineering optimisation.

# **Chapter 3**

## **Geometric representation of aerofoils and blade surfaces**

### **3.1 Introduction**

Modern wind turbine blades are complex three dimensional shapes. If a blade is sliced perpendicular to its span the cross-section resembles an aerofoil, an aerodynamic shape whose purpose is to be directed into a moving air flow so that its top (suction) side experiences a lower pressure than its bottom (pressure) side, thus providing lift - the force that keeps aircraft aloft.

The complexity of the blade shape comes from three separate parameters: the shape of the aerofoil; the way that the *twist* of the blade's chord line changes along its span; and the *taper* of the blade, or the variation of the blade's chord length along its span.

If the Blade Element Method were to be used to evaluate the aerodynamic loads experienced by an operating wind turbine blade then the blade would be represented as a sequence of constant-chord, constant-twist sections, with the spanwise width of each section dependent on the total number of blade elements. In this case the geometric representation of the blade is completely defined by its spanwise chord and twist distributions, since the aerodynamic performance embodied in the shape of the spanwise aerofoil sections are available, *a priori*, as tabulated lists of lift and drag data at specific angle of attack and Reynolds number.

The attractions of this evaluation approach, namely its ease of use and computational efficiency, belie its one big weakness: it is geometrically inflexible. What if a

*new* aerofoil section is to be studied to see if it improves the performance of the blade? Well, this new aerofoil shape will first need to be evaluated, either in a wind tunnel or on a computer, to obtain the lift and drag data needed by the blade element method. Needless to say, this severely hampers the advantages of speed and convenience provided by the BEM.

Another approach is to use an aerodynamic solution technique that uses no *a priori* knowledge about aerodynamic performance, and instead calculates blade loads directly from the blade geometry. A *two-dimensional* (2D) panel method, for example, could be used to calculate the aerodynamic performance of general aerofoil shapes, and this data then used with the BEM in much the same way as tabulated data. But why do that when, with scarcely more effort, a three-dimensional (3D) panel solver can be coded which has all the advantages of the 2D code, as well as providing potentially important 3D aerodynamic effects? After all, wind turbine blades *are* three-dimensional shapes, they're *not* stacks of blade elements, which need to be very thin indeed (read: high blade element density) to be able to capture quick geometry changes, such as the large twist/taper gradients commonly seen at blade roots. This is related to another pertinent issue: the BEM can be used very successfully to predict the performance of common wind turbine blade geometries at moderate-to-high tip speed ratios, operating points where sectional angles of attack are low and the blade sections are approximated well by aerofoil sections. This is not the case, however, at the low tip speed ratios commonly encountered during starting, and results in geometries, particularly near the blade root, which *behave differently* to the aerofoils from which they are constructed (see, for example, Wood (1991)). If optimal blade sections *do not behave as aerofoils*, then there is some doubt as to whether *any* two-dimensional aerodynamic evaluation technique is suitable for use in a three-dimensional geometry optimisation task. The 3D panel solver described in the next chapter provides, if not an entirely physically accurate solution to this problem, then at least an *adequate* solution methodology which neatly avoids the geometric limitations of the blade element method.

This raises the question: how do we represent complex three-dimensional blade geometries in a *compact* way, in a form that can represent a very wide range of blade geometries whilst still being easy to manipulate? A very neat answer to this question comes in the form of a geometry representation technique known as *B-spline curves and surfaces*. The B-spline formulation used to represent complex three-dimensional

wind turbine blade shapes in the current project is presented in the following sections.

## 3.2 B-spline curves

Piegl and Tiller (1997) define a *p*th-degree B-spline curve

$$\mathbf{C}(u) = \sum_{i=0}^n N_{i,p}(u) \mathbf{P}_i \quad a \leq u \leq b \quad (3.1)$$

where the  $\{\mathbf{P}_i\}$  are the curve's *control points*, and the  $N_{i,p}(u)$  are *p*th-degree B-spline basis functions, defined recursively as

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

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \quad (3.3)$$

defined on the non-decreasing, non-periodic and non-uniform knot vector

$$U = \underbrace{\{a, \dots, a\}}_{p+1}, u_{p+1}, \dots, u_{m-p-1}, \underbrace{\{b, \dots, b\}}_{p+1} \quad (3.4)$$

where  $a=0$  and  $b=1$ . Following the array-numbering convention used by the C programming language,  $m$  is the high-index of the knot vector and  $n$  assigned as the high-index of the control-point vector. Each piece-wise *p*th-degree polynomial B-spline curve,  $\mathbf{C}$ , thus contains  $(m+1)$  knots and  $(n+1)$  control points. These defining constants are related by

$$m = n + p + 1 \quad (3.5)$$

Descriptions of the Basis functions (together with mathematical proofs and comprehensive C code) are provided by Piegl and Tiller (1997). The reader is directed to this excellent text for theoretical and implementation details not covered here.

Figure 3.1 shows a degree 2 B-spline curve with seven control points defined on the knot vector  $\{0, 0, 0, \frac{1}{5}, \frac{2}{5}, \frac{3}{5}, \frac{4}{5}, 1, 1, 1\}$ .

A few very useful features of B-spline curves are illustrated by Figure 3.1

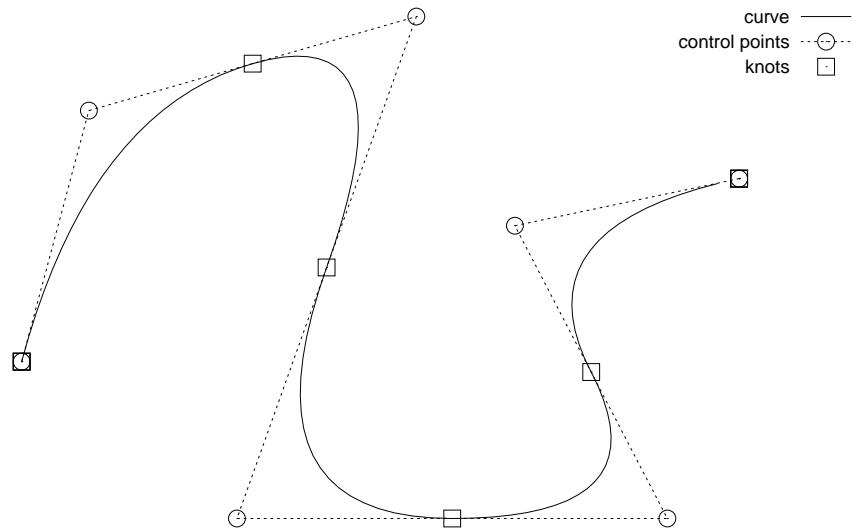


Figure 3.1: A quadratic B-spline curve ( $p=2$ )

- The curve is functionally equivalent to five quadratic curves “tied” together at the knots and is continuous down to the  $(p-1)$ th derivative at unique knots. This inherent smoothness is important for aerodynamic surface modelling. If desired, irregularities such as sharp corners may be introduced by specifying a single knot coordinate for one or more knots. Each time a knot is repeated it removes one degree of continuity.  $p$  repeated knots results in the zero-th derivative (the curve itself) becoming discontinuous.  $p+1$  repeated knots results in a break in the curve. These are used at the end-points.
- Complex shapes can be represented by just a few parameters. This is especially useful for shape optimisation problems where the positions of the control points (and/or the knots) are the object variables. Object vector length has a direct bearing on the number of optimiser iterations necessary for convergence. Very long object vectors (as would be encountered with most non-parametric geometry representations) can very well lead to unrealistic computation times and intractable problems.
- Control points influences are localised to discrete curve segments. Moving  $\mathbf{P}_i$

only modifies that section of  $\mathbf{C}(u)$  on the interval  $[u_i, u_{i+p+1}]$ . This affords great flexibility to an evolutionary shape optimiser and allows iterative and *partial* solutions to the problem to accumulate. This is a very important feature, since it makes possible true *geometry evolution* - a beneficial *segment* of a aerofoil may offer the whole aerofoil some survival value and need not be sacrificed when other adjoining segments are modified.

- The first and last control points interpolate the curve's end-points, which makes the specification of a sharp trailing edge (for closed-curve aerofoil-like shapes) a trivial matter.
- *Affine invariance* makes applying affine transforms (such as translation, scaling and rotation) to the B-spline curve a matter of applying them to just the control points. This comes is very useful when modelling aerodynamic shapes (such as wind turbine blades) as assemblies of two-dimensional B-spline aerofoil curves and makes possible the specification of chord, twist and span parameters without fear of modifying the underlying curve geometry.
- The strong *convex hull* property of B-spline curves makes constraining the geometry a simple matter. The polygon formed by the control points will *always* contain the curve, so, for example, setting an upper or lower limit on aerofoil thickness becomes a matter of specifying the limiting dimensions of a control polygon. A related application also solves a rather annoying problem: when evolving aerofoil shapes it is essential that the closed curves representing aerofoils *look* like aerofoils, with the minimal requirement that they be simply connected, have a “sharp” trailing edge and a rounded leading edge and, most importantly, that they do not self-intersect. Such “reasonably” shaped closed curves will always have a better chance of succeeding as aerodynamic shapes, so weeding out transgressors before they enter the computationally expensive aerodynamic evaluation process saves valuable computation time. The convex hull property means that checking for simple closed curve geometries is a matter of checking for control polygon edge intersections - a comparatively simple task.

### 3.2.1 Least-squares B-spline curve fitting to point data

Piegl and Tiller (1997) supply the equations and algorithms required to assemble a very useful global curve approximation routine. In the current implementation the user supplies point data lying in two-dimensional Cartesian space, the degree of the curve ( $p$ ) and the maximum permissible error between generated curve and point data ( $E$ ).  $E$  is chosen as an arbitrarily small real number, with smaller error tolerances resulting in B-spline curves which, in general, approximate the point data more closely and are defined by more sub-curves, which necessitates the storage and manipulation of a greater number of control point co-ordinates and knot parameters. The routine starts by generating an initial curve geometry with knots and control points equal to the data points in both number and position. This results in a first-degree curve which interpolates the data points exactly. An iterative loop then:

- removes as many knots as possible whilst maintaining a curve that deviates from the data points at that segment by no more than  $E$
- elevates the degree of the curve by adding knots
- finds the control point positions for this elevated knot vector that results in minimal least-square error between the generated curve and the data points
- continues until  $\text{degree}=p$  or until the least-squares control point positioning routine fails to meet the required error expectations

Figure 3.2 shows the first goal of the current project: a simple, flexible B-spline representation of an aerofoil curve.

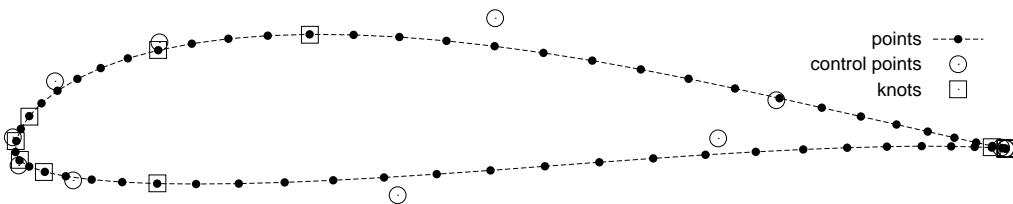


Figure 3.2: SD7062 cubic B-spline representation

The SD7062 is one of many low-speed aerofoils developed at the University of Illinois at Urbana-Champaign for model aviation applications (Selig (2000)). It has

been adopted by the University of Newcastle's Wind Energy Group for their range of small (0.6kW-20kW) wind turbine blades. Its geometry is specified in Lyon et al. (1998) as a list of x-y ordered pairs giving the coordinates of sixty-one unevenly-spaced points on its perimeter.

The least-squares curve fitting technique was applied to this point data, resulting in the curve shown in Figure 3.2. The curve is cubic ( $p=3$ ) and is defined by twelve control points ( $n=11$ ), and sixteen knots ( $m=15$ , since  $m=n+p+1$ ). The error specified for this particular run of the curve fitting routine was  $E=\frac{5}{1000}$ , resulting in a curve which deviates from each of the sixty-one data points by no more than 0.5% of the chord length. Its control point and knot vectors are:

Control Points	Knots
1.00000 0.00000	0.00000
0.76843 0.04586	0.00000
0.48452 0.12380	0.00000
0.14558 0.10085	0.00000
0.04011 0.06391	0.34597
-0.00243 0.01072	0.42068
0.00338 -0.01600	0.49190
0.05839 -0.03024	0.50513
0.38605 -0.04425	0.51447
0.70962 0.00976	0.52809
0.99526 0.00102	0.58349
1.00000 0.00000	0.99315
	1.00000
	1.00000
	1.00000
	1.00000

The curve starts at the aerofoil's trailing edge, loops around the leading edge via the upper (suction) side then over the lower (pressure) side and back to the trailing edge. The high curvature at the leading edge is accommodated by a clustering of knots and control points in that region.

### 3.3 B-spline surfaces

The foregoing cubic B-spline representation of the SD7062 aerofoil is the cornerstone of the geometry representation employed in the current project. By stringing a handful of scaled and rotated curves such as this along the the blade's span and then “skinning” a similar cubic B-spline surface over them in the radial direction, an accurate, compact and flexible wind turbine blade model is created that can be put to a variety of interesting uses. The skinning process is described below.

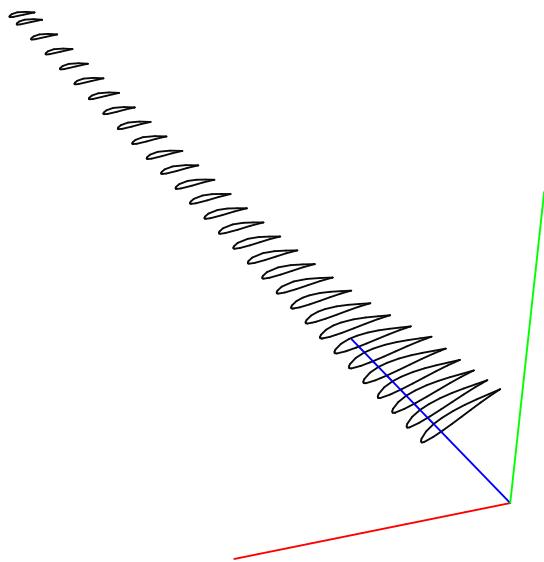


Figure 3.3: A spanwise assembly of SD7062 aerofoils

Figure 3.3 shows a typical distribution of thirty B-spline aerofoils used to model a 5 kW turbine blade. The geometry of the blade is based on the chord and twist distributions for the current-generation commercial 5kW HAWT blade developed by the Wind Energy Group at the University of Newcastle. The chord and twist distributions of this blade are shown in Figure 3.4.

The section curves in the blade are stored as as the curve array  $\mathbf{C}[](\mathbf{u})$ . The affine transformations performed on the base B-spline curve to generate an approximation of the Newcastle 5kW HAWT (scaling and rotation) only affect the control points, resulting in a *common knot vector*,  $\{\mathbf{u}\}$  shared by all aerofoils along the span. This is *essential* when creating surfaces from these section curves. A novel method to obtain a shared knot vector across a range of dissimilar B-spline section curves is presented in Section 3.4.

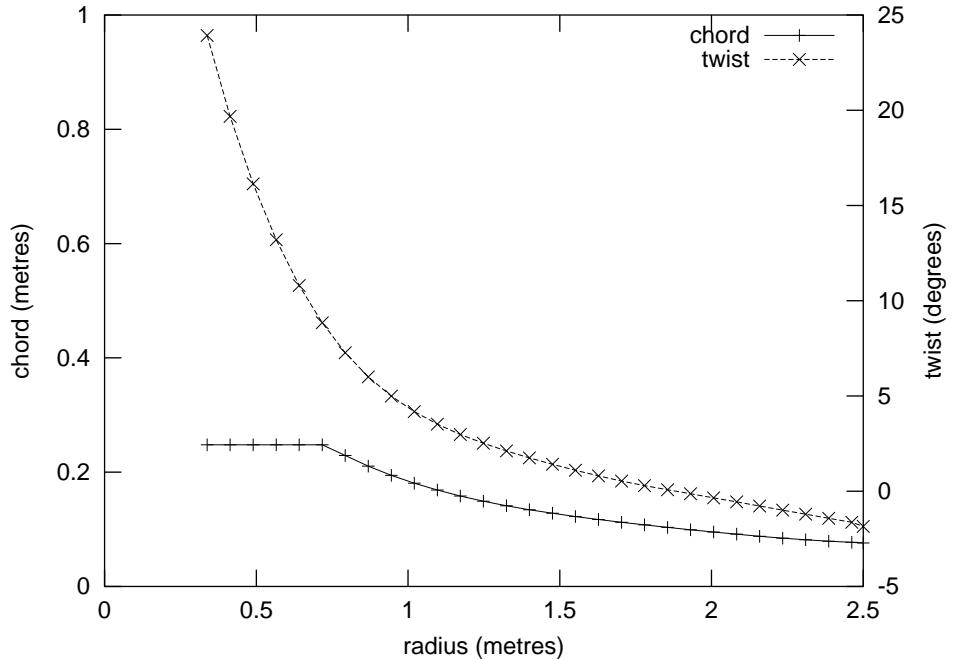


Figure 3.4: chord and twist distributions for 5kW blade

The blade is now described as a two dimensional parametric surface:

$$\mathbf{S}(u, v) = \sum_{i=0}^n \sum_{j=0}^m N_{i,p}(u) N_{j,q}(v) \mathbf{P}_{i,j} \quad (3.6)$$

The standard nomenclature is U, for the chordwise dimension and V for the spanwise dimension (with upper case referring to the parametric direction, and lower case u and v usually referring to individual parameters lying on the curve/surface in these directions). The process of generating surface geometry in the V direction is known as *Skinning*, and simply involves finding the spanwise knot vector  $\{\bar{v}_k\}$ , that accommodates the existing aerofoil control points along the span of the blade. For the current implementation, a cubic surface is specified ( $q=3$  in the V direction), together with  $K+1$  control points (ie: K is the high-index of the control-point vector in the V direction, so  $K+1=\text{number of B-spline aerofoils along span}$ ). The “skinning” method of Piegl and Tiller (1997) is then employed in a two-step process: calculate the curve parameters  $\{\bar{v}_k\}$  in the V direction:

$$\begin{aligned} \bar{v}_0 &= 0 & \bar{v}_K &= 1 \\ \bar{v}_k &= \bar{v}_{k-1} + \frac{1}{n+1} \sum_{i=0}^n \frac{|\mathbf{P}_{i,k} - \mathbf{P}_{i,k-1}|}{d_i} & k &= 1, \dots, K-1 \end{aligned} \quad (3.7)$$

where  $d_i$  is the length of the curve. Knots are then computed by averaging

$$\begin{aligned} v_0 = \cdots = v_q &= 0 & v_{m-q} = \cdots = v_m &= 1 \\ v_{j+q} &= \frac{1}{q} \sum_{i=j}^{j+q-1} \bar{u}_i & j &= 1, \dots, K-q \end{aligned} \quad (3.8)$$

The only other thing necessary for this flexible blade representation to be useful is a method for converting the location of a point  $(u,v)$  on the surface from two-dimensional parametric space into a point  $(x,y,z)$  in three-dimensional Cartesian space. Piegl and Tiller (1997) do this in three steps:

1. search for the knot spans on which  $u$  and  $v$  lie
2. compute the non-zero basis functions in both directions (Equation 3.3)
3. multiply the values of the non-zero basis functions and the corresponding control points:

$$\mathbf{S}(u, v) = [N_{k,p}(u)]^T [\mathbf{P}_{k,l}] [N_{l,q}(v)] \quad i-p \leq k \leq i, j-q \leq l \leq j \quad (3.9)$$

Routines for computing Basis function values and the position of points on the surface of the B-spline blade representation (adapted from Piegl and Tiller (1997)) are included in the source code which accompanies this thesis, with an outline given in the appendices (page 407).

### **3.4 Aerofoil curve fitting using a DE optimiser**

At this point a digression is in order. At some time it may be desirable to create a bi-cubic B-spline blade surface by assembling a range of pre-specified aerofoil sections. For example, Sørensen (1999) present the geometry of the 20.5 metre-long LM19.1 wind turbine blade (figures 3.5 and 3.6). It may be interesting to generate a B-spline representation of this blade geometry which can then be meshed with quadrilateral panels for use by a panel solver routine. This has been accomplished by digitising the published aerofoil profiles into sequences of x-y coordinates, the twist angle of each aerofoil calculated from this data, and the chord length of each section measured on the planform diagram. The method described in the preceding section was used to fit

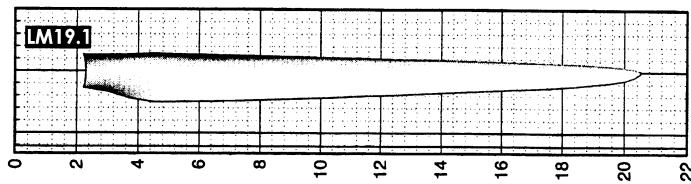


Figure 3.5: LM19.1 blade planform (Sørensen (1999))

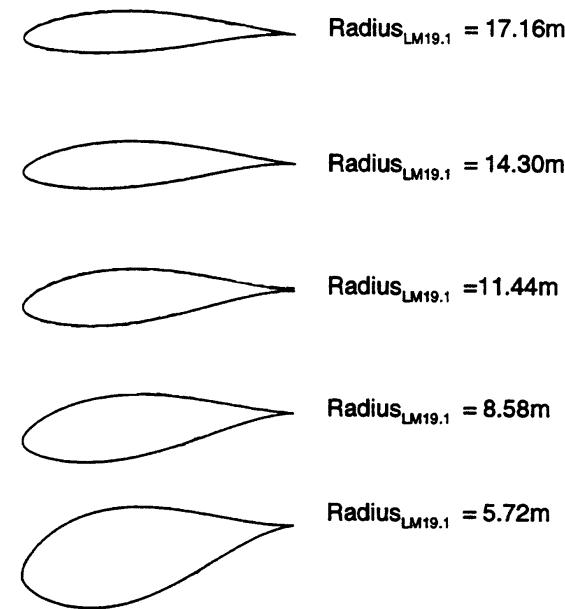


Figure 3.6: LM19.1 unit-chord aerofoil profiles (Sørensen (1999))

a least-squares B-spline curve to each set of aerofoil data. A set of curves generated in this manner are depicted in figure 3.7. A spanwise assembly of these aerofoils were subsequently skinned to form a bi-cubic representation of the LM19.1 blade, which was then discretised into a mesh of quadrilateral surface panels (figure 3.8).

When creating a surface from differing section curves, each must have the same number of control points, a common knot vector and the same degree. The least-squares curve approximation technique used in the previous section will easily produce individual B-spline aerofoil representations with identical numbers of control points and degree, but will also invariably create a *unique* knot vector for each. Piegl and Tiller (1997) use the method of *Knot Refinement* to tackle this problem, which involves merging each curve's knot vector into one large knot vector (that is, splitting up each curve into as many polynomial pieces as necessary so that each curve has an

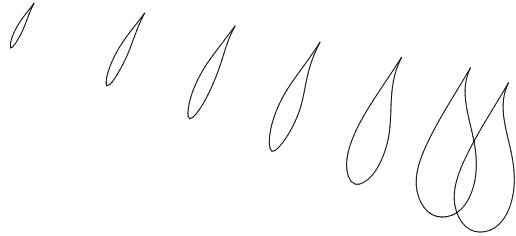


Figure 3.7: B-spline aerofoils for the LM19.1 blade

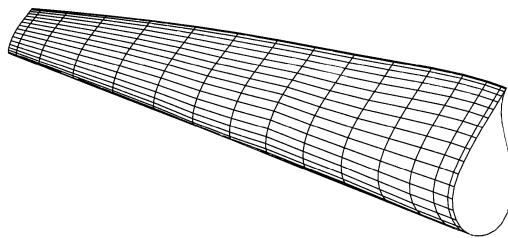


Figure 3.8: Skinned LM19.1 blade discretised into quadrilateral surface panels

equal number of pieces) and then computing a *new* set of control points for each and every curve. The associated blowout in control point numbers runs counter to the requirement that these be kept to an absolute minimum (recall that object vector sizes in evolutionary optimisation have a direct and deleterious impact on solution convergence times).

The problem, then, is to find some way to generate compact B-spline aerofoil approximations on one *common* knot vector and then compute only the positions of their control points,  $\{\mathbf{P}\}_{(n+1)}$  to give minimum deviation from the data points,  $\{\mathbf{Q}\}_N$ . This, algorithmically speaking, is easier said than done. One possible solution is to employ an Evolutionary Computation approach and *evolve the positions of the control points* by phrasing the problem as a minimal-error optimisation problem: Let

$$\mathbf{x}_\epsilon = \{e_i\} = \{e_0, e_1, \dots, e_{N-1}\}^T \quad \forall i = 0, \dots, N-1 \quad (3.10)$$

be a real-valued vector of error measurements representing distance between the curve  $\mathbf{C}(u)$  and  $N$  two-dimensional Cartesian data points,  $\{\mathbf{Q}\}_N$ . The parameter values  $\{u_i\}$  are calculated (appendix A.1) by finding the closest projection of  $\mathbf{Q}_i$  onto  $\mathbf{C}(u)$ . The error vector,  $\mathbf{e}_\epsilon = \{e_i\}$ , between all  $\mathbf{Q}_i$  and  $\mathbf{C}(u)$  can then be defined as

$$\{e_i\} = |\mathbf{C}(u_i) - \mathbf{Q}_i| \quad \forall i = 0, \dots, N-1 \quad (3.11)$$

A simple scalar *fitness* function can now be defined

$$\text{fitness} = f(\mathbf{e}_\epsilon) = \sum_{i=0}^{N-1} e_i \quad (3.12)$$

which states that a “fitter” candidate curve has a lower accumulated error score than its peers and thus better approximates the given set of foil data points. The problem is then stated as

$$\text{minimise } f(\mathbf{e}_\epsilon) \quad (3.13)$$

An additional requirement that has proven to be useful in practice is that the curve *be of minimal length* which, equivalently, requires that the sum of the length of edges of the control polygon be minimal

$$\begin{aligned} & \text{minimise } g(\mathbf{P}_{(n+1)}) \quad \text{where} \\ & g(\mathbf{P}_{(n+1)}) = \sum_{j=1}^n |\mathbf{P}_j - \mathbf{P}_{j-1}| \end{aligned} \quad (3.14)$$

Without this requirement the optimiser will reward curves that come closest to interpolating the data points, without regard given to the route taken to achieve that goal. The best interpolating curve may very well be one that takes wild excursions away from the intended curve, usually in areas of high-curvature with inadequate data point density. The simplest way of incorporating these two complementary objectives is to multiply them together to give a composite optimisation problem

$$\text{minimise } f(\mathbf{e}_\epsilon)g(\mathbf{P}_{(n+1)}) \quad (3.15)$$

Two constraints are imposed on the optimiser

1. Curves must not self-intersect. The evaluation is halted and a large fitness value returned if any control polygon edge intersects another. A description of a self-intersection checking routine used in the current project appears in the Code Outline appendix on page 417.
2. The angle formed by the upper and lower surfaces at the trailing edge (TE) must be finite. In practice, aerofoils with a TE angle of less than some tolerance (say, 5°) are penalised by multiplying their fitness by an arbitrary constant.

Algorithm 1: Minimise  $f(\mathbf{e}_\varepsilon)g(\mathbf{P}_{(n+1)})$  using DE

```

let  $\mathbf{p} = \{\mathbf{P}_0^x, \mathbf{P}_0^y, \mathbf{P}_1^x, \mathbf{P}_1^y, \dots, \mathbf{P}_n^x, \mathbf{P}_n^y\}$  be a real-valued “seed” object vector
let  $\mathbf{r} = \{r_0, r_1, \dots, r_{2n}\}$  be a real-valued vector of Gaussian-random deviates
pop =  $\{\mathbf{x}_i\} = \{\mathbf{p} + \mathbf{r}_i\} \quad \forall i \in \{0, N-1\}$ ; a population of perturbed object vectors
evaluate  $\{fitness_{(i)}\} = f(\{\mathbf{x}_i\})g(\{\mathbf{x}_i\}) \quad \forall i \in \{0, N-1\}$ 
for  $gen = 1$  to  $MAXGENS$  do
    for  $i = 0$  to  $(N - 1)$  do
        create  $\mathbf{t}_{(i,gen)} = \mathbf{x}_{(rnd_1,gen-1)} + F(\mathbf{x}_{(rnd_2,gen-1)} - \mathbf{x}_{(rnd_3,gen-1)})$  (equation 6.6)
         $\mathbf{z}_{(i,gen)} = crossover(\mathbf{t}_{(i,gen)})$  (equation 6.7)
        evaluate  $fitness_{(i,gen)} = f(\mathbf{z}_{(i,gen)})g(\mathbf{z}_{(i,gen)})$ 
        if  $fitness_{(i,gen)} > \{fitness_{(i)}\}$  then
             $\{\mathbf{x}_i\} = \mathbf{z}_{(i,gen)}$ 
             $\{fitness_{(i)}\} = fitness_{(i,gen)}$ 
        end if
    end for
end for

```

---

Differential Evolution (Storn and Price (1995) and Storn (1996)) has been chosen to implement Equation (3.15). Details of the DE evolutionary optimisation method are provided in Chapter 6. The algorithm is summarised in Algorithm 1.

As stated previously, it is important for assemblies of B-spline aerofoils along the span of a wind turbine blade to share a common knot vector, even if the control points which define the shape of each curve are very different, resulting in very different curve shapes. The major practical reason for this requirement is to ensure regular blade panelling when the B-spline surface representation is subsequently discretised for use by the panel solver. So, let the circle shown in figure 3.9 represent the initial

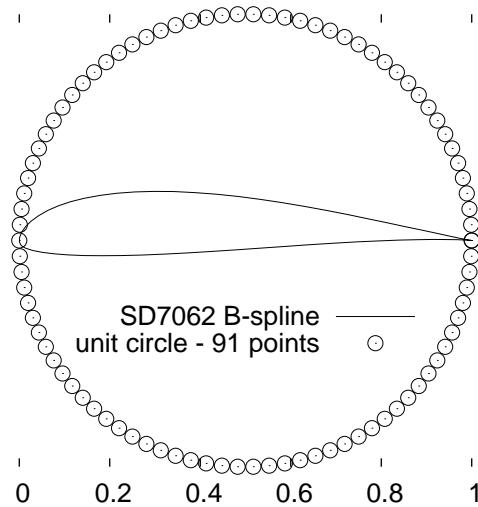


Figure 3.9: B-spline aerofoil and circular data point-set

conditions of an example B-spline curve evolution problem. This might be the case, for example, if an interesting surface geometry defined by the SD7062 aerofoil and a circle was described in a journal article, and it was thought worthwhile to construct a bi-cubic B-spline surface representation based on this description. The goal, then, may be to create a circular B-spline curve defined on the knot vector of an existing seed SD7062 aerofoil B-spline curve.

The solution to this problem was implemented as follows: the control points of the SD7062 B-spline aerofoil became the initial object vector in Algorithm 1; a population of fifty randomly perturbed object vectors was generated; and the algorithm was allowed to progress for one-thousand generations.

Figure 3.10 shows the best curve taken from selection of generations during a typ-

ical DE optimiser run. The best curve of generation 1000 then formed the seed for

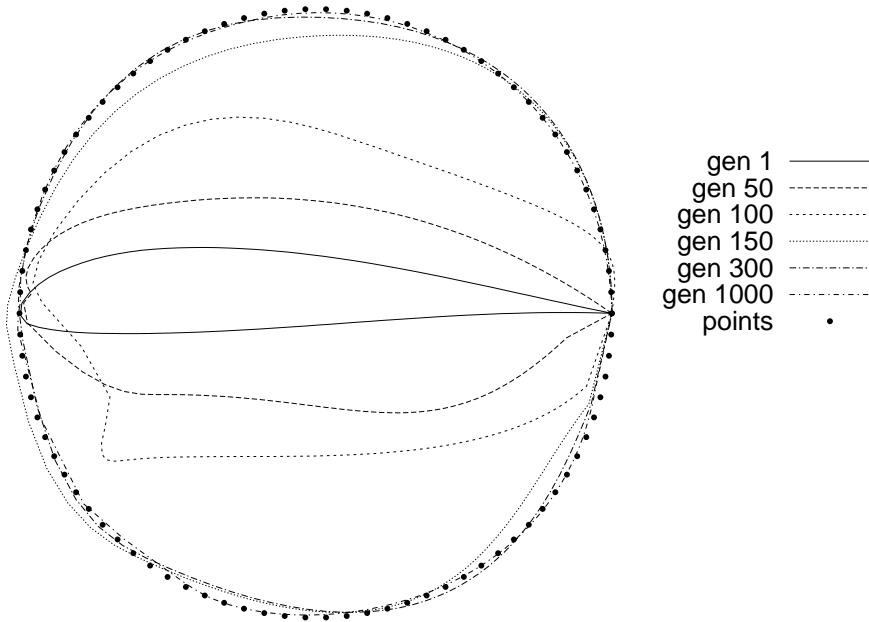


Figure 3.10: A set of evolving best-of-generation curves

a subsequent “refinement” run of 2000 generations, resulting in the curve shown in figure 3.11. Figures 3.12 and 3.13 show the history of best and average fitness scores for the evolving populations. Figure 3.14 presents a sequence of the evolved circular and aerofoil B-spline section curves featured in this example. A parametric surface has been skinned over these and appears as a collection of discretised quadrilateral surface panels in figure 3.15.

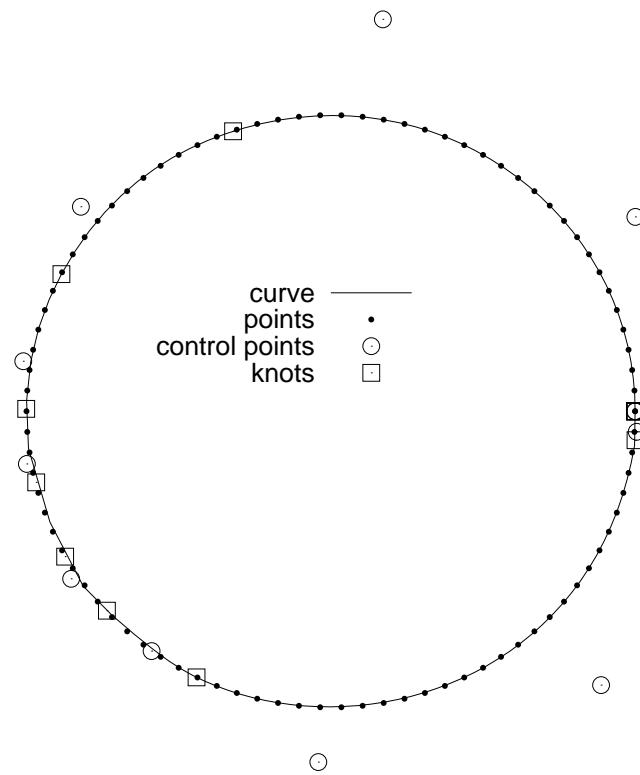


Figure 3.11: An evolved circular B-spline curve sharing the SD7062 knot vector

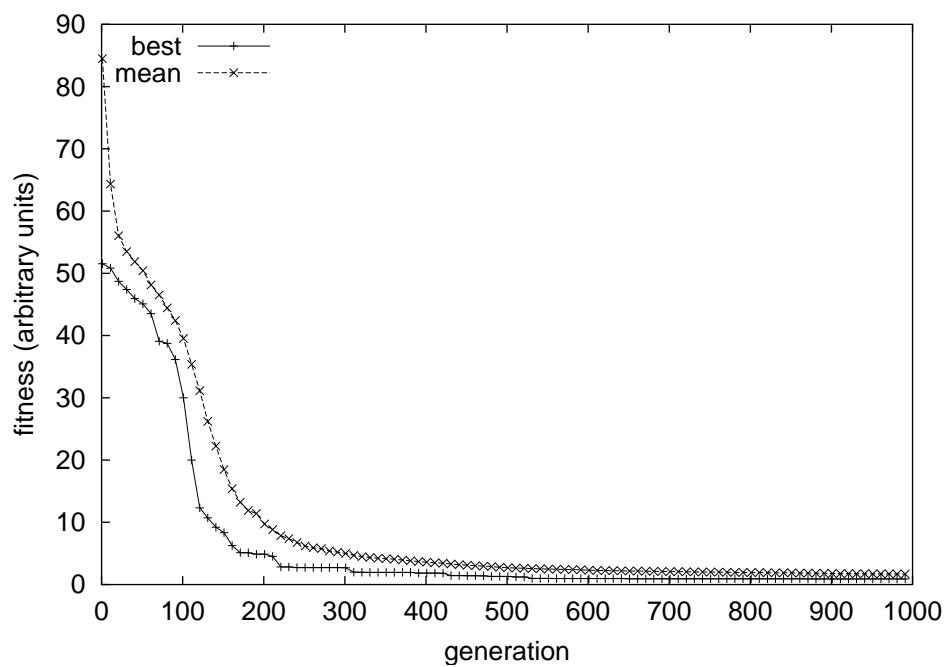


Figure 3.12: Fitness history of initial optimisation run

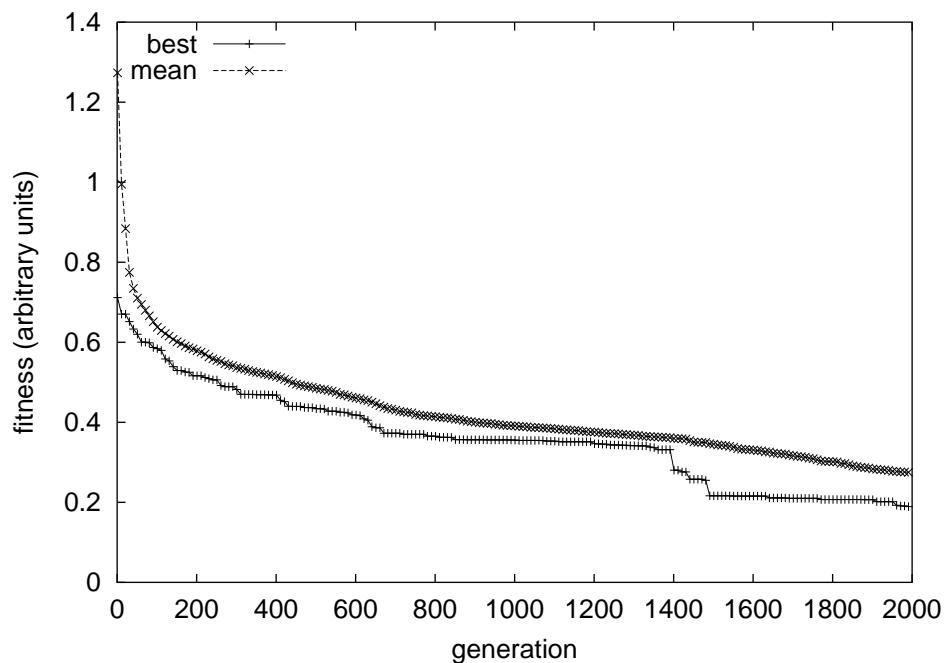


Figure 3.13: Fitness history of refinement optimisation run

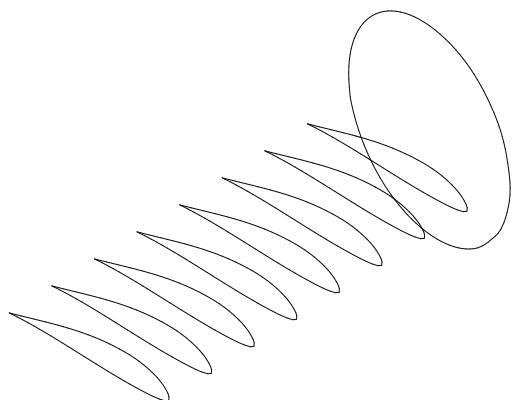


Figure 3.14: A spanwise sequence of circular and aerofoil B-spline curves

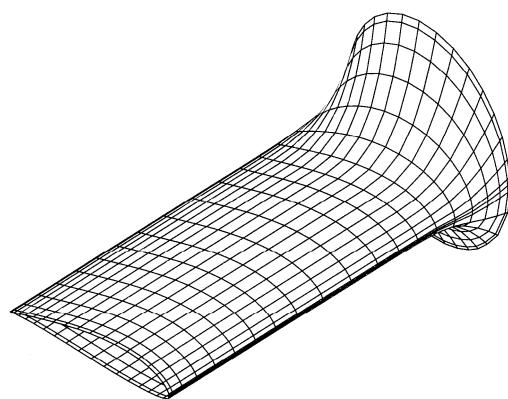


Figure 3.15: A skinned and meshed parametric surface

### **3.4.1 Application: creation of transition region geometry**

A useful practical application of this modelling technique is the creation of “transition section” geometry linking the “aerodynamic” part of the blade with the “structural”, load-bearing part near the blade root for use in the current model. Figure 3.17 shows



Figure 3.16: Blade without root transition



Figure 3.17: Blade with root transition

the result of replacing the first aerofoil section at the root of Figure 3.16 with a curve more representative of the rectangular butt section used on the current 5 kW Newcastle blade. The butt curve has rounded corners, a sharp trailing edge (to accommodate a “smooth” shed wake) and is formed on the same knot vector as the SD7062 B-spline aerofoil used in the rest of the blade. The three-dimensional transition geometry created in this manner is a smooth blend from the relatively thin aerofoil section to the rectangular hub connector piece. The quadrilateral surface panel and hexahedral brick mesh discretisations of this geometry have the major advantage of being smooth and regular: essential features for use in aerodynamic and structural solver routines.

This concludes the overview of both the geometry representation and optimisation methods used throughout this thesis. Further chapters will describe how they form the basis of a powerful surface geometry evolution technique that has been applied to a problem of wind turbine blade shape optimisation.

# Chapter 4

## Panel Method

### 4.1 Introduction

The aim of the current project is to develop a method which “evolves” wind turbine blade shapes. This immediately poses two important questions: how can the aerodynamic performance of general, three dimensional blade geometries be *evaluated*? And, taking into consideration the fact that *thousands* of potential blade designs will be evaluated during every optimisation run, how can these evaluations be made to run *quickly* on existing (modest) commodity computer hardware?

The standard way of evaluating wind turbine blade performance is via the Blade Element Method (BEM). This is a *very* quick computational technique which uses tabulated experimental two-dimensional lift and drag data to approximate three-dimensional blade loads. The utility of the BEM is lost, however, when blade geometries stray from the rather limited domain of designs that only use *existing aerofoil sections* at low-to-moderate angles of attack, and this problem is exacerbated by the fact that small wind turbines often encounter low-Re, high- $\alpha$  flows. These are regimes which most existing aviation aerofoils (that is, most aerofoils) were simply never designed to handle, resulting in a dearth of low-Re, high- $\alpha$  lift and drag data. Thus the BEM fails the “generality” requirement.

A possible way to remedy the situation would involve the use of a 2D viscous solver to evaluate general 2D “aerofoil” shapes, with the results fed into a BEM evaluator. The stand-out candidate for this task is XFOIL (Drela (2001)), which uses an iterative viscous-inviscid interaction method to solve for moderate- $\alpha$ , moderate-Re

flows. The disadvantages of this idea are that the iterative nature of the viscous/inviscid calculation negates most of the time-saving advantages of the BEM, especially since worthwhile, high-resolution BEM evaluations require many thin span-wise blade elements; and the risk of mis-convergence is quite high for the very-low-Re, very-high- $\alpha$  flows encountered by small wind turbines, and a pathological viscous result can be assumed to be considerably less valuable as an arbiter of starting performance than a merely inaccurate inviscid result.

At the other end of the generality scale lie the so-called “grid-based” aerodynamic solution methodologies: finite difference/element/volume methods that partition the flow domain into many discrete elements, and a numerical technique applied to solve the full viscous Navier-Stokes equations over the discretised domain. These are the mainstay of the Computational Fluid Dynamics field and can be classified broadly as “very accurate, but very, very computationally expensive”. The practicalities of running such a Navier-Stokes solver on any *one* general three-dimensional wind turbine blade geometry necessitate lots of RAM, a fast processor (or processors) and a considerable wait. Running thousands of such evaluations for a single optimisation run is, simply, impractical.

Between these two extremes lies the workhorse of the aerodynamic evaluation industry: the Boundary Element Method. Instead of discretising the entire external flow domain (for external aerodynamic problems), Boundary Element Methods typically solve for particular flow quantities, such as velocity, velocity potential or pressure, *where they are most often needed* - on the surface of a solid body submerged in a flow. This reduces the number of domain discretisations that need to be constructed, since only the solid boundary is partitioned, resulting in a corresponding decrease in the number of discrete equations to be stored and solved - typically by a few orders of magnitude - which results in a substantial decrease in solution times and required computer memory. Discrete boundary units are known as “surface panels”, which is why the techniques are commonly called “Panel Methods”. The price paid for this convenience is a high one: the removal of all viscous flow information from the calculations, bar the first-order viscous correction supplied by the Kutta condition. These methods solve the Laplace equation at each surface panel, which is essentially a major simplification of the Navier-Stokes equations via a removal of all viscous terms. The utility of panel methods for a large subset of external aerodynamic flows comes from

one important fact: for high-Re, low- $\alpha$  aerofoil/wing flows, where viscous effects like separation and stall are absent, the assumption of attached flow is very accurate.

A three-dimensional panel method using distributions of constant source and doublet singularities over the discretised blade surface was chosen as the aerodynamic solver for the current project. A similar panel discretisation was constructed for the trailing wake, with a distribution of constant doublet singularities imposed to model the circulation shed at the trailing edge of an operating wind turbine. This choice of solution technique was made simple by the fact that the method is capable of storing and evaluating many individual blade designs on modest commodity hardware, with acceptable accuracy and within an acceptable time.

## 4.2 Blade discretisation

Once a bi-cubic B-spline model of some general blade geometry,  $S(u,v)$ , is available it can be partitioned into quadrilateral surface panels for use in an aerodynamic panel solver. Figure 4.1 shows an example of a wind turbine rotor consisting of two B-spline blade surfaces, each of which has been discretised into a mesh of quadrilateral surface panels. The right-handed global co-ordinate system consists of an X-axis aligned with

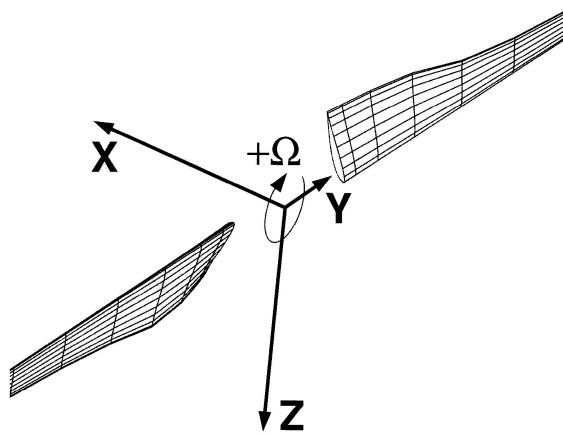


Figure 4.1: Two-bladed panel mesh showing global co-ordinate system

the rotor axis which points in the direction of the undisturbed wind flow,  $U_0$ . The global Y-axis runs parallel to the leading-edge of blade zero (the key blade) and passes through quarter-chord point of the first hub-wise aerofoil. The global Z-axis is orthog-

onal to the X-Y plane. The rotor spins in the Y-Z plane in a clockwise direction from positive Y-axis to positive Z-axis with angular speed  $\Omega$ . Figure 4.2 shows the global

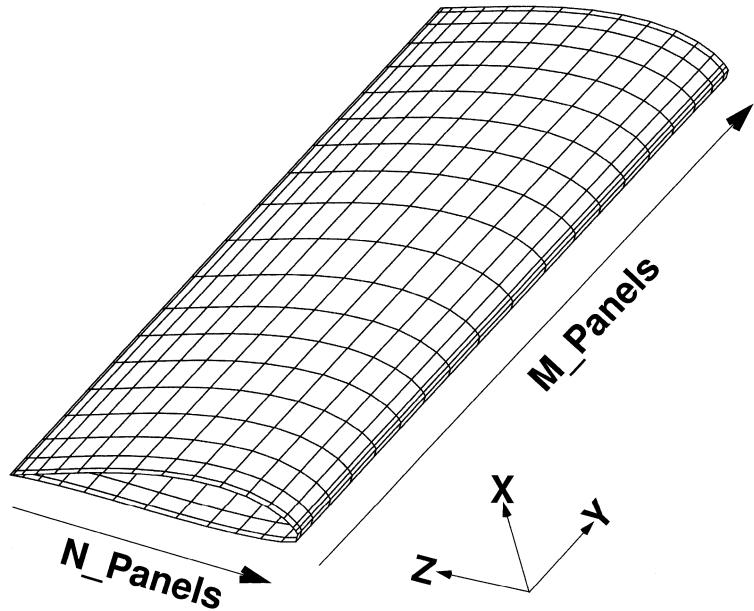


Figure 4.2: Panelling direction

panelling scheme. The surface is divided into a mesh of quadrilateral surface panels, with each panel's vertices lying on the surface. There are  $N\_PANELS$  divisions in the chordwise (u) direction and  $M\_PANELS$  divisions in the spanwise (v) direction. Each chord-wise “strip” of  $N\_PANELS$  can be thought of as equivalent to a “blade element” - only with more geometric flexibility, since the aerofoil sections at either end are not required to be identical. Blade panels are stored in the computational blade mesh object as a one-dimensional array of panel objects (see appendices on pages 435 and 439). The panel array numbering is shown in figure 4.3. Panel[0] is the first trailing-edge upper (suction) blade hub panel (that is, nearest the global co-ordinate-system origin). Panel numbering for the first panel strip continues across the upper side of the blade, around the leading edge, across the lower side and back to the trailing edge. Numbering resumes at the next upper-side trailing-edge spanwise panel and continues like this until the last lower-side trailing-edge panel on the blade tip panel is reached. A helpful fact is that the first panel on each blade panel strip is always a multiple of  $N\_PANELS$ .

“Cosine spacing” (figure 4.4) is used to increase the panel density in regions of computational interest in both directions: at the blade tip and hub in the V direc-

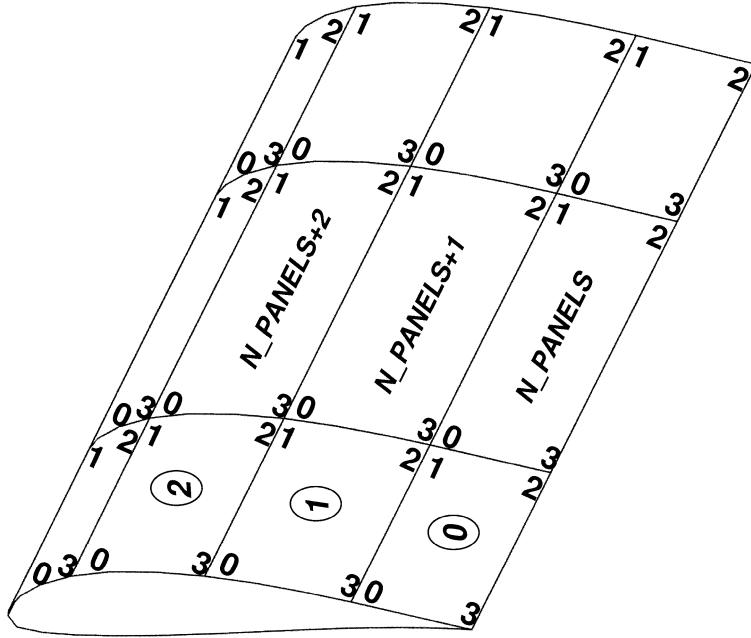


Figure 4.3: Panel numbering scheme

tion; and at the trailing and leading edges in the U direction. Panel vertices in global Cartesian co-ordinates are calculated by first finding appropriate  $\{u\}$  and  $\{v\}$  surface parameters and then evaluating the point  $S(u,v)$  using the routine outlined on page 413. The  $\{v\}$  vertex parameters,  $v[]$ , are calculated via the following pseudo-code:

$$v[i] = 1 - \frac{1 + \cos(i\delta\theta)}{2} \quad \forall i = \{0, \dots, M\_PANELS\}; \delta\theta = \frac{\pi}{M\_PANELS} \quad (4.1)$$

A slightly more involved method is needed for the  $\{u\}$  vertex parameters. This is because, for maximum flexibility, each B-spline aerofoil has a unique leading edge  $\{u\}$  parameter, and so has a unique ratio of upper and lower surface lengths. The obvious choice of simply putting the upper-surface on  $\{u\}=\{0.0,0.5\}$  and the lower on  $\{u\}=[0.5,1.0]$  will miss the correct position of the leading edge for anything but a symmetrical aerofoil. The leading edge of each panel strip should coincide with all others along the span of the meshed blade, so the procedure shown in Algorithm 2 is adopted. Once the panel vertex parameters  $\{u\}$  and  $\{v\}$  have been calculated, finding the global co-ordinates of the panel vertices is a simple matter of evaluating  $S(u,v)$  for strategic vertices, then copying over these co-ordinates for any coincident vertices (algorithm 4).

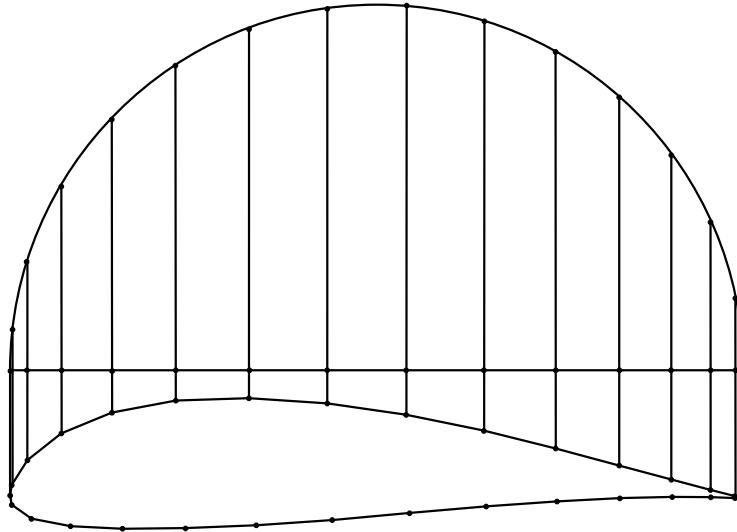


Figure 4.4: Chordwise cosine spacing

Figure 4.5 shows a typical HAWT blade panel mesh (constructed from a model skinned over the aerofoils in figure 3.3).

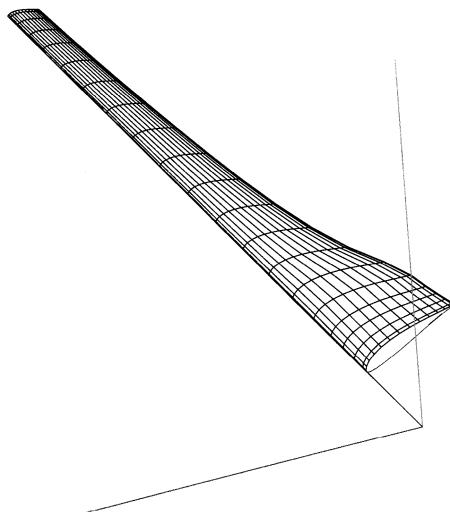


Figure 4.5: HAWT blade mesh. N\_PANELS=30; M\_PANELS=20

Rotors with more than one blade are catered for by simply multiplying the storage array sizes by the number of blades, NB, and then making copies of the blade mesh data structures. The vertices of the key blade are rotated about the origin in the global Y-Z plane by an angle  $\theta = \frac{2\pi}{NB}$  (algorithm 3). The number of panels in the entire rotor mesh is then  $NB \times N\_PANELS \times M\_PANELS$ .

Algorithm 2: Calculating  $u[]$  panel vertex parameters for a particular  $v$  parameter

find  $u_{LE}^i \forall i = 0, \dots, N_{spanwise\_foils}$  (Appendices, page 412)

find the two aerofoils surrounding  $v$  ( $f_j$  and  $f_{j-1}$ )

find the leading edge U parameter for this  $v$  by interpolating linearly:

- $a = (f_{j-1}^y - f_{hub}^y) / (R\_blade - f_{hub}^y)$
- $b = (f_j^y - f_{hub}^y) / (R\_blade - f_{hub}^y)$
- $ratio = (v - a) / (b - a)$
- $u_{LE}^* = u_{LE}^{j-1} + ratio \times (u_{LE}^j - u_{LE}^{j-1})$

 $\delta\theta = \frac{2\pi}{N\_PANELS}$ 

**for**  $k = 0$  to  $N\_PANELS$  **do**

$$u[k] = u_{LE}^*(1 - \cos(k\delta\theta)) / 2 ; \quad 0 \leq k\delta\theta \leq \pi$$

$$u[k] = u_{LE}^* + (1 - u_{LE}^*) \times (1 - \cos(k\delta\theta - \pi)) / 2 ; \quad \pi < k\delta\theta \leq 2\pi$$

**end for**

Algorithm 3: Copying the key blade mesh for multi-bladed rotors

$B = N\_PANELS \times M\_PANELS$

$\theta = \frac{2\pi}{NB}$

**for**  $blade = 1$  to  $NB - 1$  **do**

$costh = \cos(blade \times \theta)$

$sinth = \sin(blade \times \theta)$

**for**  $i = 0$  to  $B - 1$  **do**

**for**  $j = 0$  to  $3$  **do**

$y = panel[i].vertex[j].y$

$z = panel[i].vertex[j].z$

$panel[i + (blade * B)].vertex[j].x = panel[i].vertex[j].x$

$panel[i + (blade * B)].vertex[j].y = y * costh - z * sinth$

$panel[i + (blade * B)].vertex[j].z = y * sinth + z * costh$

**end for**

**end for**

**end for**

---

Algorithm 4: Calculating panel vertex co-ordinates

```

k = 0;
for  $i = 0$  to  $M\_PANELS$  do
    for  $j = 0$  to  $N\_PANELS$  do
        if  $j == 0$  then
             $panel[k].vertex[3] = \mathbf{S}(u[j]_{v[i]}, v[i])$  {appendix, page 413}
             $panel[k].vertex[2] = \mathbf{S}(u[j]_{v[i+1]}, v[i+1])$ 
        else
             $panel[k].vertex[3] = panel[k - 1].vertex[0]$ 
             $panel[k].vertex[2] = panel[k - 1].vertex[1]$ 
        end if
        if  $j < N\_PANELS - 1$  then
             $panel[k].vertex[1] = \mathbf{S}(u[j + 1]_{v[i+1]}, v[i + 1])$ 
             $panel[k].vertex[0] = \mathbf{S}(u[j + 1]_{v[i]}, v[i])$ 
        else
             $panel[k].vertex[1] = panel[k - N\_PANELS + 1].vertex[2]$ 
             $panel[k].vertex[0] = panel[k - N\_PANELS + 1].vertex[3]$ 
        end if
         $k ++;$ 
    end for
end for

```

---

### 4.3 Wake model and wake discretisation

The wake shed by each blade is similarly represented by a quadrilateral panel mesh, whose numbering scheme is shown in figure 4.6.

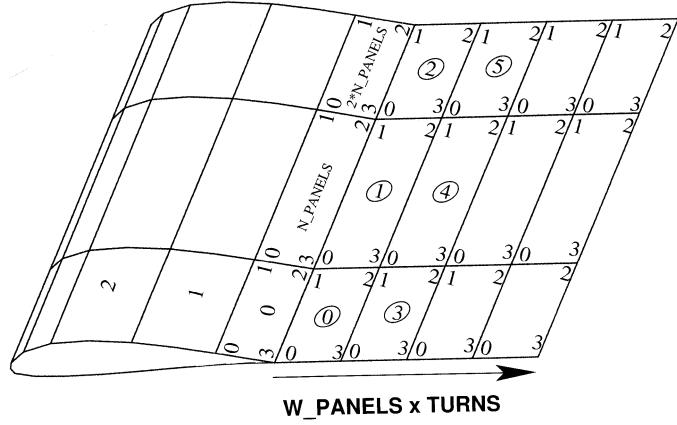


Figure 4.6: Wake panel numbering scheme

A helical rotor wake mesh has  $W\_PANEL$  panels per wake revolution and  $TURNS$  wake revolutions (figure 4.7). A wake panel “strip” is shed from each blade strip trailing edge, giving  $M\_PANELS \times W\_PANELS \times TURNS$  wake panels for each blade.

A basic helical wake model is provided by Preuss et al. (1980). They partition each wake revolution into equal segments spanning an angle  $\theta = \frac{2\pi}{W\_PANELS}$  with the rotor axis. Each spanwise row of  $M\_PANELS$  wake panels (parallel to the blade trailing edge) is thus represented by an angle  $\alpha_i$  (equation (4.2)).

$$\alpha_i = i \cdot \theta \quad i = \{0, \dots, (W\_PANELS \times TURNS) - 1\} \quad (4.2)$$

The wake is thus defined as a sequence of streamwise rows of wake panel vertices “shed” by the blade trailing edge (TE) vertices and “convected” downstream at the undisturbed flow velocity (Equations (4.3) to (4.6)).

$$\alpha_{TE} = \tan^{-1} \left( \frac{z_{TE}}{y_{TE}} \right) \quad (4.3)$$

$$x = x_{TE} + \frac{\alpha_i U_0}{\Omega} \quad (4.4)$$

$$y = \sqrt{y_{TE}^2 + z_{TE}^2} \cos(\alpha_{TE} - \alpha_i) \quad (4.5)$$

$$z = \sqrt{y_{TE}^2 + z_{TE}^2} \sin(\alpha_{TE} - \alpha_i) \quad (4.6)$$

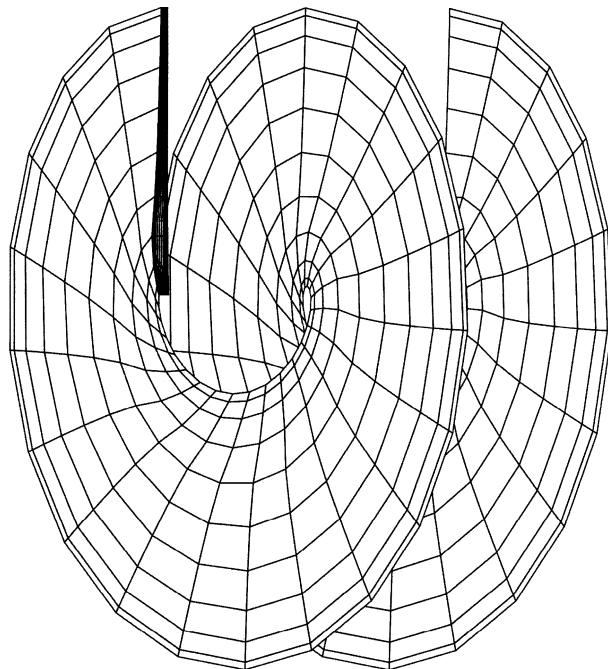


Figure 4.7: A wake mesh; M\_PANELS=10, W\_PANELS=20, TURNS=2

Quadrilateral wake panels are finally assembled from these vertices (algorithm 5).

## 4.4 Modifications to the simple wake model

The wake model of the preceding section can be improved in a number of ways whilst keeping its computational burden manageable.

### 4.4.1 First wake panel

The potential flow assumption in the current numerical method requires that the rotor wake be shed *smoothly* from the blade's trailing edge. In practice, this entails having the first streamwise row of wake panels bisect the angle subtended at the blade trailing edge. At high tip speed ratios,  $\lambda$ , this requirement is satisfied by the basic wake geometry. However at low  $\lambda$ , especially near the rotor hub, the simple wake model describes wake panels that make unacceptably large angles with the trailing edge, resulting in wildly incorrect trailing edge potential calculations. The scheme shown in figure 4.8 is used to find a unit vector which bisects the the blade trailing edge. Point **a** is taken as

---

Algorithm 5: Assembling quadrilateral wake panels

```

k = 0 ; alpha = 0.0
θ =  $\frac{2\pi}{W\_PANELS}$ 
for i = 0 to ( $W\_PANELS \times TURN_S$ ) - 1 do
    for j = 0 to  $M\_PANELS - 1$  do
        if i == 0 then
            panel[k].vertex[0] = TE[k].vertex[3]
            panel[k].vertex[1] = TE[k].vertex[2]
            panel[k].vertex[2] = wake_point(panel[k].vertex[1], θ) (equation (4.3))
            panel[k].vertex[3] = wake_point(panel[k].vertex[0], θ)
        else
            panel[k].vertex[0] = panel[k - M_PANELS].vertex[3]
            panel[k].vertex[1] = panel[k - M_PANELS].vertex[2]
            panel[k].vertex[2] = point(panel[j].vertex[2], α) (equation (4.3))
            panel[k].vertex[3] = point(panel[j].vertex[3], α)
        end if
        k ++
    end for
    α = α + θ
end for

```

---

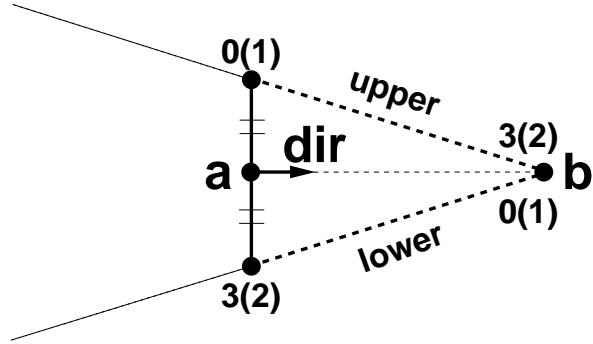


Figure 4.8: Finding a direction vector which bisects the blade trailing edge

the midpoint between the vertex[0] and vertex[3] on the upper and lower trailing edge panels, respectively, for the hubwise wake vertex strip and vertex[1] and vertex[2] on the upper and lower trailing edge panels, respectively, for the tipwise wake vertex strip. Point **b** is taken as the accompanying TE point. The bisecting unit vector is then

$$\mathbf{dir} = \frac{\mathbf{b} - \mathbf{a}}{\|\mathbf{b} - \mathbf{a}\|} \quad (4.7)$$

This gives the location of the first row of wake vertices as

$$\mathbf{wake\_vertex}_0^{new} = \mathbf{TE} + \|\mathbf{wake\_vertex}_0^{current} - \mathbf{TE}\| \cdot \mathbf{dir} \quad (4.8)$$

which produces a panel with the same dimensions as that given by the original model but which also bisects the blade's trailing edge to the blade trailing edge. The panel length can be tailored by replacing  $\|\mathbf{wake\_vertex}_0^{current} - \mathbf{TE}\|$  with some arbitrary constant length (or some small multiplier such as a fraction of the local chord length).

#### 4.4.2 Near wake

Reducing the panel count whilst maintaining solution accuracy is important when creating panel meshes for use by an evolutionary optimisation objective function. The wake's influence on the blades is strongest in the near-wake, so it is important to maintain a high panel density in this region. The influence of individual wake panels becomes less critical as their downstream distance increases, thus making it possible to decrease the downstream panel density to free-up memory and computational resources. This is achieved in the current wake model by compressing the first wake

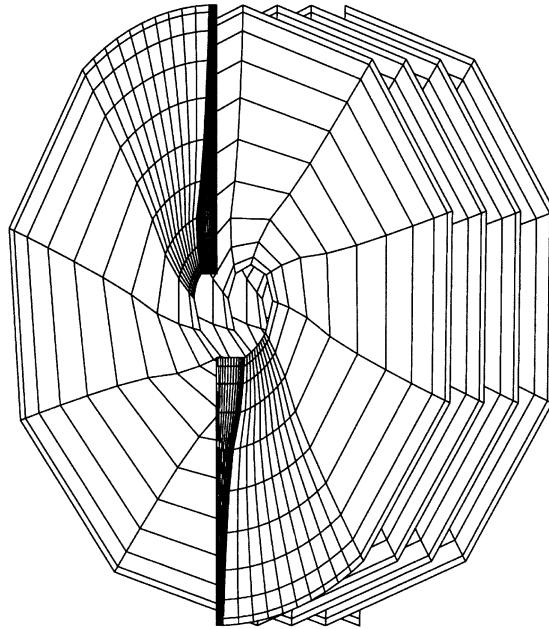


Figure 4.9: Denser near-wake mesh. M\_PANELS=10 ; W\_PANELS=10; TURNS=2

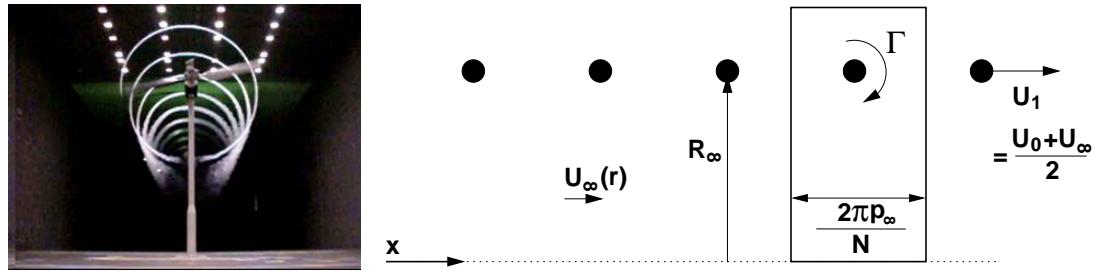
revolution into the space of one “normal” wake column:

$$\alpha_{w\_row[0]} = \frac{\theta}{W\_PANELS} = \frac{2\pi}{(W\_PANELS)^2} \quad (4.9)$$

#### 4.4.3 Wake pitch relaxation

Equation (4.3) will always calculate a wake pitch greater than the actual pitch, since it uses the free-stream velocity,  $U_0$  instead of the flow velocity through the rotor,  $U_1$  to convect the wake helices downstream. This reduces the overall influence of the wake on the blades and results in an over-prediction of the power produced. The difference between  $U_0$  and  $U_1$  increases as the rotor thrust (and power) increases. So the simple model will give good wake pitch predictions at low  $\lambda$ , and progressively worse predictions as the rotor’s operating  $\lambda$  is approached. This is a result of more kinetic energy being extracted from the flow and the accompanying reduction in the velocity of the column of air passing through the rotor disc.

Wood and Boersma (2001) provide a relation between that maximum circulation on the turbine blade,  $\Gamma$ , and the non-dimensional pitch,  $p_\infty = \frac{pitch}{2\pi}$ , of a helical tip vortex (an example of which appears in figure 4.10.1). As an application of their derivation of the self-induced velocity of a tip vortex, they presented the scheme shown in fig-



4.10.1:

4.10.2:

Figure 4.10: NREL combined experiment turbine tip vortex visualisation; and simple wake model by Wood and Boersma (2001)

ure 4.10.2 (all lengths are normalised by  $R$ , the wind turbine radius, and all velocities normalised by the free-stream velocity,  $U_0$ ). Consider a rectangular contour with radial extent  $r$  and side length  $\frac{2\pi p_\infty}{N}$ , which is aligned with the wind turbine axis. When  $r > R_\infty$ , and the contour encloses  $N$  tip vortices without intersecting them, the area integral of the vorticity within the contour is given by

$$\int \Omega dA = -N\Gamma \quad (4.10)$$

and the contribution of the circulation from the two radial legs cancel, giving a circulation of

$$\Gamma = U_1 \frac{2\pi p_\infty}{N} \quad (4.11)$$

A basic control volume analysis (appendix B.1) shows that the velocity seen at the rotor disc can be expressed as the average of the upstream and downstream velocities

$$U_1 = \frac{1 + U_\infty}{2} \quad (4.12)$$

Using this result, and equating the vorticity integral and the circulation gives

$$U_\infty = 1 - \frac{N\Gamma}{2\pi p_\infty} \quad (4.13)$$

The (non-dimensional) pitch of a helical tip vortex convected downstream with velocity  $U_1$  is given by

$$p_\infty = \frac{1 + U_\infty}{2\lambda} \quad (4.14)$$

which, when inserted into (4.13) and rearranging for  $U_\infty$  gives

$$\frac{U_\infty}{U_0} = \sqrt{1 - \frac{N\Gamma\lambda}{\pi}} \quad (4.15)$$

This may be used with (4.12) to express  $U_1$  in terms of  $\Gamma$  and  $\lambda$ . The expression for the streamwise component of each wake vertex co-ordinate, (4.4), may then be modified slightly to give a wake model whose pitch is dependent on the maximum blade circulation (equation (4.16)).

$$x = x_{TE} + \frac{\alpha_i U_1}{\Omega} \quad (4.16)$$

The circulation at any spanwise blade cross-section is easily calculated by the panel method (equation 4.25). So the creation of a wake geometry tuned to a particular blade design becomes a matter of creating an initial wake pitch by guessing a value of  $U_1$ , using the panel method to predict power, thrust and blade maximum circulation (and by (4.15) and (4.12) a new  $U_1$ ) and then iterating until  $U_1$  converges. This is a simple and effective method that avoids the inaccuracies of the basic fixed-pitch wake model, as well as being far less computationally expensive than free-wake models. Figure

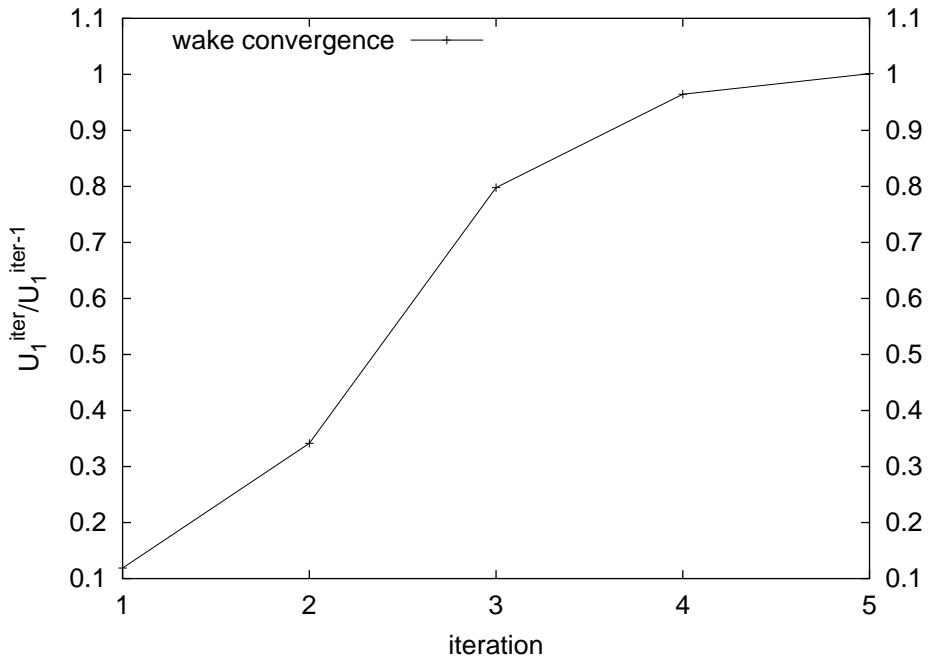


Figure 4.11: Wake pitch convergence for 5kW rotor (40x20mesh)  $\lambda=10$

4.11 shows a typical wake pitch relaxation history for this iterative wake model. The

flow velocity at the rotor,  $U_1$ , for the zeroth iteration is taken as the free stream velocity,  $U_0$ . The linear system of equations described in the next section is solved for the value of the velocity potential on each panel,  $\{\phi\}_k$ , from which the circulation around each panel strip,  $\Gamma = \Delta\phi_{TE}$ , is calculated. A corresponding  $U_1$  for this iteration is calculated and checked against the prediction of the previous iteration for convergence within some small tolerance. If convergence has not been achieved a new  $U_1$  prediction is formed as the average of the old prediction and the current calculation, and the wake iteration continues with a new wake geometry with pitch determined by the newly predicted  $U_1$ . In the current implementation convergence is assumed within ten iteration steps to conserve computing time. It has been observed in many numerical experiments that, in most cases, the wake model arrives at a converged  $U_1$  value within five to six iteration steps.

Computer code for all wake geometry routines is integrated with the `blade_mesh` object code, and is outlined in the appendices on page 439.

## 4.5 The 3D constant-source/doublet panel method

Following Preuss et al. (1980), consider the rotating system shown in figure 4.1. The global Cartesian co-ordinate system rotates about the x-axis with constant angular velocity  $\Omega$  and is submerged in a uniform air flow with speed  $U_0 = \mathbf{U}_0 \cdot \mathbf{i}$  coaxial with the global x-axis.

The fluid velocity at any point in this rotating frame of reference is given by

$$\mathbf{U}_F = \mathbf{U}_W + \nabla\phi + \nabla \times \boldsymbol{\pi} \quad (4.17)$$

where the perturbation potential,  $\boldsymbol{\pi}$  is negligible if the flow is assumed inviscid, so

$$\mathbf{U}_F = \mathbf{U}_W + \nabla\phi \quad (4.18)$$

and  $\mathbf{U}_W$  is the velocity of the unperturbed fluid flow at any point in the flow,  $\mathbf{P}$

$$\mathbf{U}_W = \mathbf{U}_0 - \Omega \mathbf{i} \times \mathbf{P} \quad (4.19)$$

An observer fixed to a point  $\mathbf{P}$  on the surface of a rotating blade experiences a fluid velocity of  $\mathbf{U}_F$ . A boundary condition is imposed which states that the fluid does not penetrate the surface of the solid blade

$$\mathbf{U}_F \cdot \mathbf{n} = 0 \quad (4.20)$$

where  $\mathbf{n}$  is the unit-vector normal to the blade surface at  $\mathbf{P}$ . Combining (4.18) and (4.20), the boundary condition in terms of the perturbation potential becomes

$$\frac{\partial\phi}{\partial n} = -\mathbf{U}_W \cdot \mathbf{n} \quad (4.21)$$

Laplace's equation for the velocity perturbation potential gives

$$\nabla^2\phi = 0 \quad (4.22)$$

with  $\phi = 0$  for  $\mathbf{P}$  at infinity. The solution of (4.22) gives the velocity potential at any point in the rotating frame of reference. Solving for  $\phi$  on the surface of the blades allows the calculation of the local perturbation velocities,  $\nabla\phi$  and, via the Bernoulli equation in a rotating frame of reference, the steady-state induced pressures

$$C_p = \frac{p - p_0}{\frac{1}{2}\rho U_W^2} = \frac{1}{|\mathbf{U}_W|^2}(-|\nabla\phi|^2 + 2\nabla\phi \cdot (-\mathbf{U}_W)) \quad (4.23)$$

### 4.5.1 Numerical solution of Laplace's equation

Using a Green's function approach and following Morino et al. (1974) and Preuss et al. (1980), equation (4.22) is transformed into a Fredholm integral equation of the second kind

$$2\pi\phi(\mathbf{P}_0) = - \oint_{S_R} \frac{\partial\phi}{\partial n} \frac{1}{r} dS_R + \oint_{S_R} \phi \frac{\partial}{\partial n} \left( \frac{1}{r} \right) dS_R + \oint_{S_W} \Delta\phi \frac{\partial}{\partial n} \left( \frac{1}{r} \right) dS_W \quad (4.24)$$

where  $S_R$  is the surface of the rotor and  $S_W$  is the surface of the wake. The strength of the doublet singularity on every panel element on each streamwise wake strip is taken as the circulation around the adjacent blade strip and is given by

$$\Delta\phi = \phi_{upper} - \phi_{lower} \quad (4.25)$$

with  $\phi_{upper}$  and  $\phi_{lower}$  referring respectively to the upper and lower blade trailing edge panels which shed that particular streamwise wake strip (figure 4.12).

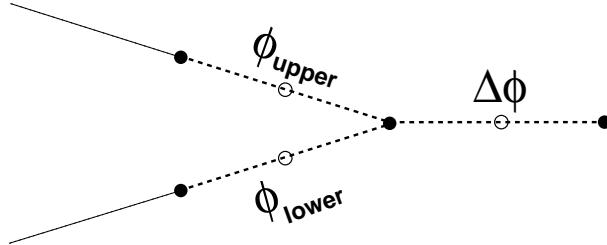


Figure 4.12: Strength of doublet on each streamwise wake strip

The continuous blade and wake surfaces are discretised into quadrilateral surface panels as described in sections 4.2 and 4.3. Equation (4.24) is then transformed into the following linear system of equations

$$[\delta_{hk} - c_{hk}] \{\phi_k\} + [t_{hm}] \{(\Delta\phi)_m\} = [b_{hk}] \left\{ \left( \frac{\partial\phi}{\partial n} \right)_k \right\} \quad (4.26)$$

with the elements  $c_{hk}$  being the influence of a constant doublet singularity located at the *collocation point* of blade panel  $k$  at the collocation point of blade panel  $h$  ( $h$  and  $k \in \{0, \dots, (NB \times N\_PANELS \times M\_PANELS) - 1\}$ ).  $b_{hk}$  is the influence of a constant source singularity located on panel  $k$  on blade panel  $h$ .  $t_{hm}$  is the influence of a constant doublet singularity located on each wake panel  $m$  at the collocation point of blade panel  $h$ , with  $m \in \{0, \dots, (NB \times M\_PANELS \times W\_PANELS \times TURNS) - 1\}$ .  $\left( \frac{\partial\phi}{\partial n} \right)_k$  is given

by equation (4.21) as the no-flow boundary condition at the collocation point of blade panel  $k$ . From the definition of the Kronecker delta,  $\delta_{hk}$ ,

$$\delta_{hk} - c_{hk} = \begin{cases} 0.5 & h = k \\ c_{hk} & \text{otherwise} \end{cases} \quad (4.27)$$

This accounts for the influence of a constant doublet singularity on itself.

The  $\mathbf{C}_{hk}$  and  $\mathbf{B}_{hk}$  matrices, both square with dimension  $NB \times N\_PANELS \times M\_PANELS$  are assembled by finding unit doublet and source influences of each blade panel on each other. The  $\mathbf{T}_{hm}$  matrix is also constructed as a square  $NB \times N\_PANELS \times M\_PANELS$  matrix, with non-zero elements representing equation (4.25) as follows:

Algorithm 6: Assembling the wake influence matrix

```

B = NB × N_PANELS × M_PANELS
W = NB × M_PANELS × TURNS × W_PANELS
zero(Thm)
for m = 0 to W do
    for h = 0 to B do
        thm = doublet(wake_panel[m], blade_panel[h].P[0])
        k = wake_panel[m].upper_TE_panel
        Thm[h][k] = Thm[h][k] + thm
        k = wake_panel[m].lower_TE_panel
        Thm[h][k] = Thm[h][k] - thm
    end for
end for
```

---

The equations can now be expressed as

$$(\mathbf{C}_{hk} + \mathbf{T}_{hm})\{\phi_k\} = \mathbf{B}_{hk} \left\{ \left( \frac{\partial \phi}{\partial n} \right)_k \right\} \quad (4.28)$$

The right-hand-side of equation (4.28) is easily calculated from the boundary condition, resulting in a linear system of equations that can be solved for the blade panel potentials,  $\{\phi_k\}$ . The numerical solver chosen for this project is the simple and widely used LU decomposition technique provided by Press et al. (1992). It is unlikely that another solution method would be significantly faster than LU decomposition for use in the current application.

### 4.5.2 Panel geometry

The source and doublet singularity expressions from Newman (1986) were adopted for use in the current method. The expressions are valid for *non-flat* quadrilateral panels, a fact which has major implications for the accuracy of the current panel solver, and makes possible the use of a blade/wake mesh representation constructed from panels with vertices lying on the surface of the blade/wake, rather than from planar panels. For the calculation of  $[c_{hk}]$ ,  $[b_{hk}]$  and  $[t_{hm}]$  using these doublet and source singularity expressions, it is necessary to convert the vertex positions of panel  $k$  (and  $m$ ) and the field point (the collocation point of panel  $h$ ) from the global Cartesian frame into a panel-local frame. Morino et al. (1974) provides a very convenient panel co-ordinate system representation, which has been modified for the current implementation (figure 4.13). Each panel has an orthogonal, three-dimensional co-ordinate system with major

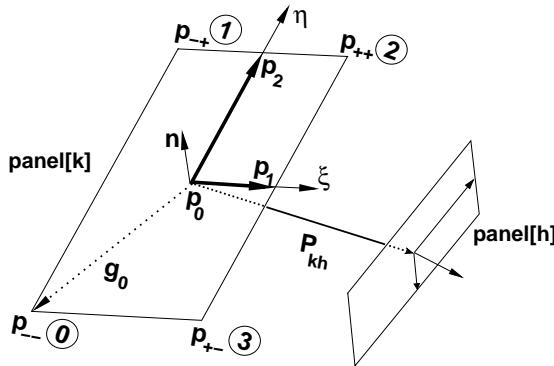


Figure 4.13: Panel co-ordinates

axis,  $\xi$  and minor axis,  $\eta$ , centred at the panel's collocation point,  $\mathbf{p}_0$ . Morino et al. (1974) model the quadrilateral panel as a segment of a hyperbolic paraboloid, with corner nodes coinciding with the panel vertices already calculated. The position of any point,  $\mathbf{P}$  on this hyperboloidal panel is

$$\mathbf{P} = \mathbf{p}_0 + \xi \mathbf{p}_1 + \eta \mathbf{p}_2 + \xi \eta \mathbf{p}_3 \quad (4.29)$$

where  $\mathbf{p}_i$  are obtained from the existing panel vertices

$$\left\{ \begin{array}{l} \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}_2 \\ \mathbf{p}_3 \end{array} \right\} = \frac{1}{4} \left[ \begin{array}{cccc} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \end{array} \right] \left\{ \begin{array}{l} \mathbf{p}_{++2} \\ \mathbf{p}_{+-3} \\ \mathbf{p}_{-+1} \\ \mathbf{p}_{--0} \end{array} \right\} \quad (4.30)$$

The vector  $\mathbf{R}(\xi, \eta)$  points from the collocation point of panel  $h$  to any point  $\mathbf{P}(\xi, \eta)$  on panel  $k$  (or  $m$ ), and is given by

$$\mathbf{R}(\xi, \eta) = \mathbf{P} - \mathbf{P}_h = \mathbf{p}_0 + \xi \mathbf{p}_1 + \eta \mathbf{p}_2 + \xi \eta \mathbf{p}_3 - \mathbf{P}_h \quad (4.31)$$

Covariant basis vectors,  $\mathbf{a}_1$  and  $\mathbf{a}_2$  and associated normal  $\mathbf{n}$  are defined as

$$\mathbf{a}_1(\xi, \eta) = \frac{\partial \mathbf{R}}{\partial \xi} = \mathbf{p}_1 + \eta \mathbf{p}_3 \quad (4.32)$$

$$\mathbf{a}_2(\xi, \eta) = \frac{\partial \mathbf{R}}{\partial \eta} = \mathbf{p}_2 + \xi \mathbf{p}_3 \quad (4.33)$$

$$\mathbf{n}(\xi, \eta) = \frac{\mathbf{a}_1 \times \mathbf{a}_2}{|\mathbf{a}_1 \times \mathbf{a}_2|} \quad (4.34)$$

In the current context the normal vector and two unit-direction vectors along the local major and minor panel axes are evaluated at the panel collocation point ( $\mathbf{P}(0, 0) = \mathbf{p}_0$ )

$$\mathbf{n} = \frac{\mathbf{a}_1(0, 0) \times \mathbf{a}_2(0, 0)}{|\mathbf{a}_1(0, 0) \times \mathbf{a}_2(0, 0)|} = \frac{\mathbf{p}_1 \times \mathbf{p}_2}{|\mathbf{p}_1 \times \mathbf{p}_2|} \quad (4.35)$$

$$\hat{\xi} = \frac{\mathbf{a}_1(0, 0)}{|\mathbf{a}_1(0, 0)|} = \frac{\mathbf{p}_1}{|\mathbf{p}_1|} \quad (4.36)$$

$$\hat{\eta} = \frac{\mathbf{a}_2(0, 0)}{|\mathbf{a}_2(0, 0)|} = \frac{\mathbf{p}_2}{|\mathbf{p}_2|} \quad (4.37)$$

This forms the panel's orthogonal co-ordinate system, with the collocation point at the origin. It is now possible to find the locations of the panel vertices in panel co-ordinates:

$$\mathbf{g}_i = \mathbf{p}_{\textcircled{i}} - \mathbf{p}_0 \quad (i \in \{0, \dots, 3\}) \quad (4.38)$$

$$\mathbf{p}_{\textcircled{i}}^{x\_local} = \mathbf{g}_i \cdot \hat{\xi} \quad (4.39)$$

$$\mathbf{p}_{\textcircled{i}}^{y\_local} = \mathbf{g}_i \cdot \hat{\eta} \quad (4.40)$$

$$\mathbf{p}_{\textcircled{i}}^{z\_local} = \mathbf{g}_i \cdot \mathbf{n} \quad (4.41)$$

as well as the position of the collocation point on panel  $h$  (the field point):

$$\mathbf{P}_{kh} = \mathbf{p}_0^h - \mathbf{p}_0 \quad (4.42)$$

$$\mathbf{P}_{kh}^{x\_local} = \mathbf{P}_{kh} \cdot \hat{\xi} \quad (4.43)$$

$$\mathbf{P}_{kh}^{y\_local} = \mathbf{P}_{kh} \cdot \hat{\eta} \quad (4.44)$$

$$\mathbf{P}_{kh}^{z\_local} = \mathbf{P}_{kh} \cdot \mathbf{n} \quad (4.45)$$

Code for this panel co-ordinate system implementation is integrated into the aerodynamic solver routines, and is outlined in the appendices on page 457.

### 4.5.3 Doublet singularity

Newman (1986) provides expressions (and their derivation) for the influence of doublet and source distributions over quadrilateral panels. Consider a quadrilateral panel with vertices at the points  $\mathbf{p}_{\textcircled{n}}^{\text{local}} = (\xi_n, \eta_n, 0)$  with  $n \in \{0, \dots, 3\}$ . The potential at an arbitrary field point  $\mathbf{p}_0^h(x, y, z)$  due to a constant doublet distribution over the panel with density  $-4\pi$  is given by

$$\Phi = \sum_{n=0}^3 \left\{ \tan^{-1} \frac{\Delta\eta_n [(x - \xi_n)^2 + z^2] - \Delta\xi_n [(x - \xi_n)(y - \eta_n)]}{R_n z \Delta\xi_n} \right. \\ \left. - \tan^{-1} \frac{\Delta\eta_n [(x - \xi_{n+1})^2 + z^2] - \Delta\xi_n [(x - \xi_{n+1})(y - \eta_{n+1})]}{R_{n+1} z \Delta\xi_n} \right\} \quad (4.46)$$

where  $\Delta\eta_n = (\eta_{n+1} - \eta_n)$ ,  $R_n = |\mathbf{p}_0^h - \mathbf{p}_{\textcircled{n}}^{\text{local}}|$  and the cyclic convention holds for vertex numbering.

### 4.5.4 Source singularity

Newman (1986) similarly provides an expression for the potential of a source distribution of constant strength  $-4\pi$  over a quadrilateral panel at a field point  $\mathbf{p}_0^h(x, y, z)$

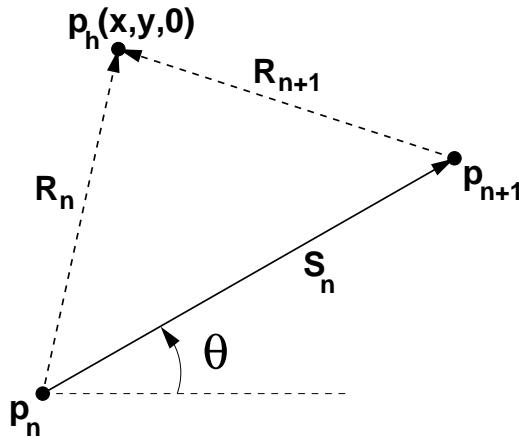


Figure 4.14: Calculation of source strength over one panel edge

$$\Psi = \sum_{n=0}^3 [(x - \xi_n) \sin \theta_n - (y - \eta_n) \cos \theta_n] \log \frac{R_n + R_{n+1} + S_n}{R_n + R_{n+1} - S_n} - z \Phi \quad (4.47)$$

where  $\Phi$  is the result of equation (4.46),  $S_n$  the length of the panel edge  $\mathbf{p}_{n+1}^{local} - \mathbf{p}_n^{local}$  and  $\theta_n$  the angle this edge makes with the positive local  $\xi$  axis

$$\theta_n = \tan^{-1} \frac{\Delta\eta_n}{\Delta\xi_n} \quad (4.48)$$

Care must be taken when evaluating (4.48) to ensure that the  $\Delta\xi_n$  term never goes to zero, and that  $-\pi \leq \theta_n \leq \pi$  to reflect the correct geometric quadrant. Code for these singularity evaluations is integrated with the aerodynamic solver routines, and their usage is outlined in the appendices on page 457.

## 4.6 Calculation of quantities of interest

Once equation (4.28) has been solved for the  $\{\phi_k\}$ , the steady pressure and pressure coefficient on each panel is found via the Bernoulli equation in a rotating frame of reference (equation (4.23))

$$C_p = \frac{p - p_0}{\frac{1}{2}\rho U_W^2} = \frac{1}{U_W^2}(-|\nabla\phi|^2 + 2\nabla\phi \cdot (-\mathbf{U}_W^{local}))$$

where  $\mathbf{U}_W^{local}$  is the velocity of the unperturbed flow resolved into the panel-local coordinate frame

$$\begin{aligned} U_W^\xi &= \mathbf{U}_W \cdot \hat{\xi} \\ U_W^\eta &= \mathbf{U}_W \cdot \hat{\eta} \\ U_W^n &= \mathbf{U}_W \cdot \mathbf{n} \end{aligned} \quad (4.49)$$

and the local components of  $\nabla\phi$  are found by central differences (figure 4.15 and equation (4.50)).

$$\begin{aligned} \left(\frac{\partial\phi}{\partial\xi}\right)_k &= \frac{\phi_{k+1} - \phi_{k-1}}{2|\mathbf{p}_1^k| + |\mathbf{p}_1^{k+1}| + |\mathbf{p}_1^{k-1}|} \\ \left(\frac{\partial\phi}{\partial\eta}\right)_k &= \frac{\phi_{k+N\_PANELS} - \phi_{k-N\_PANELS}}{2|\mathbf{p}_2^k| + |\mathbf{p}_2^{k+N\_PANELS}| + |\mathbf{p}_2^{k-N\_PANELS}|} \end{aligned} \quad (4.50)$$

and  $(\frac{\partial\phi}{\partial n})_k$  is given by the boundary condition (equation (4.21)).

The aerodynamic force acting normal to each panel is then

$$\mathbf{F}_k = \frac{-P_k \cdot \mathbf{n}_k}{4|\mathbf{p}_1^k||\mathbf{p}_2^k|} \quad (4.51)$$

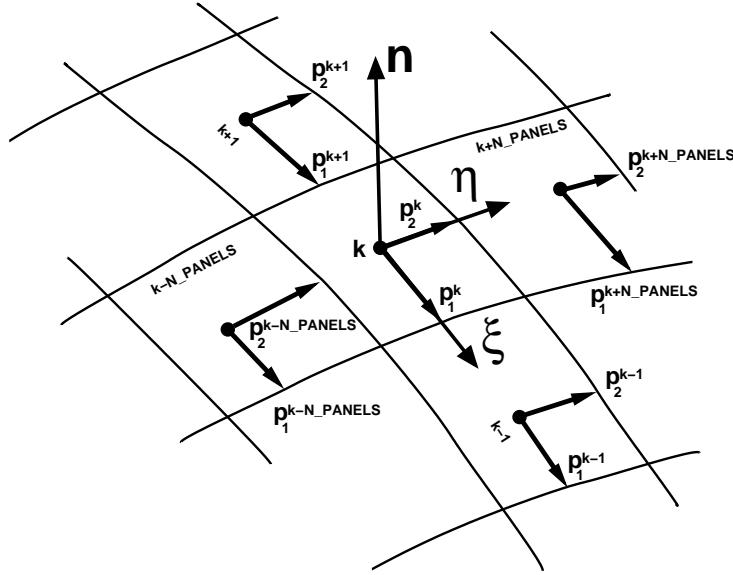


Figure 4.15: Scheme for central differences on panels

where  $P_k$  is the steady-state pressure on panel  $k$  given by equation (4.49).

The total torque provided by the rotor about the rotor's axis (in this case the global X-axis) is then calculated as the sum of all the panel torque contributions

$$Q_{rotor} = \sum Q_k = \sum_{k=0}^{panels-1} \mathbf{F}_k \cdot (\mathbf{i} \times \mathbf{p}_0^k) \quad (4.52)$$

with  $panels = NB \times N\_PANELS \times M\_PANELS$ . The total axial force on the rotor (the thrust) is, similarly, the sum of the elemental thrusts

$$F_x = \sum_{k=0}^{panels-1} \mathbf{F}_k \cdot \mathbf{i} \quad (4.53)$$

For steady flow, the power provided by the rotor is then

$$P_{rotor} = Q_{rotor} \Omega \quad (4.54)$$

and the power coefficient

$$C_P = \frac{P_{rotor}}{\frac{1}{2} \rho \pi R^2 U_0^3} \quad (4.55)$$

For comparison with the results of other studies, it is also convenient to express the aerodynamic force on each span-wise panel strip (or blade element) in terms of lift and pressure-induced drag (figure 4.16).

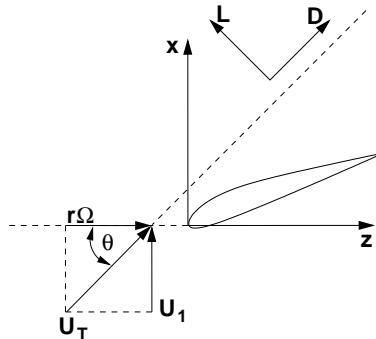


Figure 4.16: Lift and drag directions

$$L = -F_z \sin(\theta) + F_x \cos(\theta)$$

$$D = F_z \cos(\theta) + F_x \sin(\theta) \quad (4.56)$$

with the lift and drag coefficients given by

$$\begin{aligned} C_L &= \frac{L}{\frac{1}{2}\rho U_T^2 S} \\ C_D &= \frac{D}{\frac{1}{2}\rho U_T^2 S} \end{aligned} \quad (4.57)$$

where  $S$  is the planform area of the blade element strip under consideration, and is the product of the element's chord and span lengths.

## 4.7 Simple iterative Kutta condition

The strategy of creating a short first wake panel which bisects the trailing edge angle in most cases results in adequate pressure equalisation at the blade trailing edge but by no means ensures that it occurs. Figure 4.17 gives a  $C_p$  versus chord plot showing a typical case of trailing edge pressure inequality. Trailing edge pressure deficits are particularly common near the blade root where comparatively large angles of attack are seen, especially at low tip speed ratios.

The non-linear relationship between pressure and the velocity potential on each panel makes it necessary to introduce an explicit Kutta condition to ensure trailing edge pressure equality. This is achieved by iteratively adjusting the circulation around each blade element. Consider a vector containing the differences between the pressure

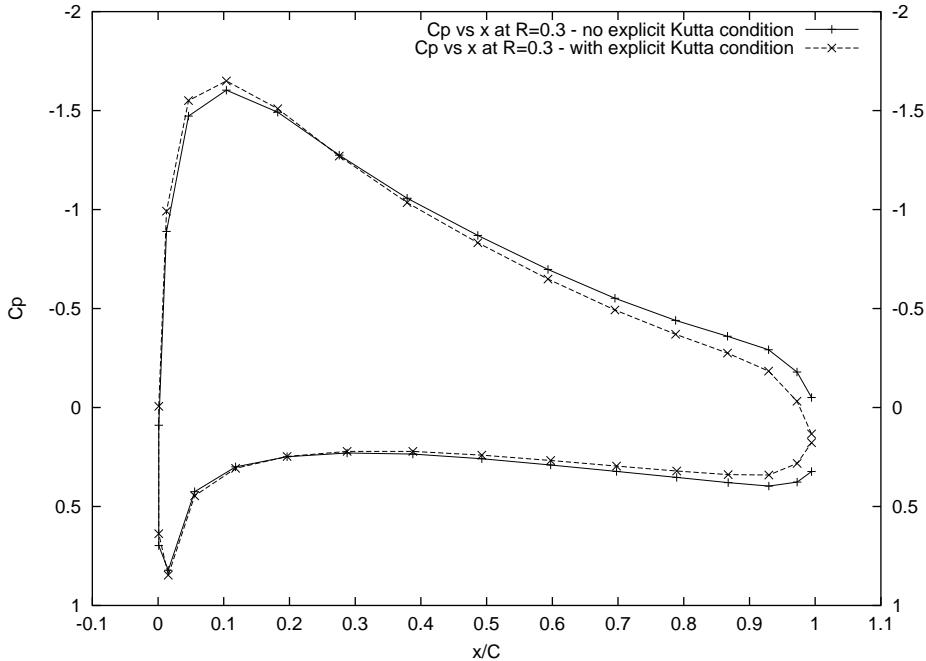


Figure 4.17: Pressure equalisation at trailing edge of 5kW blade

coefficient on the upper and lower blade trailing edge panels

$$\{\Delta C_p\}_m = C_p^{upper} - C_p^{lower} \quad m \in 0, \dots, M\_PANELS \quad (4.58)$$

$$upper = m \times N\_PANELS$$

$$lower = upper + N\_PANELS - 1$$

Next consider the wake influence matrix  $\mathbf{T}_{hm}$  whose coefficients are assembled using algorithm 6. Algorithm 7 provides a simple and effective method for perturbing the columns of trailing edge wake influences in  $\mathbf{T}_{hm}$  to give blade element circulations better suited to minimal trailing edge pressure deficits. At each iteration step the slightly adjusted wake influence coefficients contained in the modified  $\mathbf{T}_{hm}$  matrix are added to the blade influence matrix ( $\mathbf{C}_{hk}$ ) and the linear system solved for  $\{\phi\}$ . This continues until the the pressure deficit at each panel strip's trailing edge has diminished to less than some small tolerance. The current implementation of this explicit Kutta condition has been limited to four iterations to reduce computing times and gives results similar to the converged solution shown in figure 4.17.

---

Algorithm 7: Iterative wake adjustment for the explicit Kutta condition

```

 $tol = 0.01$ 
 $B = NB * N\_PANELS * M\_PANELS$ 
for  $m = 0$  to  $m < M\_PANELS$  do
    if  $|\{\Delta C_p\}_m| > 0.5$  then
         $mult = 1. + 0.03/iteration$ 
    else
         $mult = 1. + 0.01/iteration$ 
    end if
    if  $\{\Delta C_p\}_m < -tol$  then
         $\beta = 1./mult$ 
    else if  $\{\Delta C_p\}_m > tol$  then
         $\beta = mult$ 
    else
         $\beta = 1.0$ 
    end if
    for  $j = 0$  to  $j < B$  do
         $Thm[j][upper]* = \beta$ 
         $Thm[j][lower]* = \beta$ 
    end for
end for

```

---

## 4.8 Exploiting the symmetry of multi-bladed rotors

Recall the linear set of equations that are solved for the  $\{\phi_k\}$

$$[\mathbf{C}_{hk}]\{\phi_k\} + [\mathbf{T}_{hm}]\{\Delta\phi_m\} = [\mathbf{B}_{hk}] \left\{ \left( \frac{\partial\phi}{\partial n} \right)_k \right\} \quad (4.59)$$

The matrices  $\mathbf{C}_{hk}$ ,  $\mathbf{T}_{hm}$  and  $\mathbf{B}_{hk}$  are square and, for a one-bladed rotor, have dimension  $B = N\_PANELS \times M\_PANELS$ . Similarly,  $\{\phi_k\}$ ,  $\{\Delta\phi_m\}$  and  $\left\{ \left( \frac{\partial\phi}{\partial n} \right)_k \right\}$  are arrays of length  $B$ . The  $\mathbf{C}_{hk}$  matrix and  $\{\phi_k\}$  array for this one-bladed rotor geometry take the form

$$\mathbf{C}_{hk} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1B} \\ c_{21} & c_{22} & \cdots & c_{2B} \\ \dots & \dots & \dots & \dots \\ c_{B1} & c_{B2} & \cdots & c_{BB} \end{pmatrix} \quad \phi_k = \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_B \end{Bmatrix} \quad (4.60)$$

with identical structures for the other matrices and arrays.

A simple yet inefficient method for representing an  $N$ -bladed rotor geometry involves expanding the dimension of each matrix and array to  $N \times B$ , so that the influence of each blade and wake panel over every blade panel in every blade is explicitly stored, for example

$$\bar{\mathbf{C}}_{hk} = \begin{pmatrix} \mathbf{C}_{hk}^{11} & \mathbf{C}_{hk}^{12} & \cdots & \mathbf{C}_{hk}^{1N} \\ \mathbf{C}_{hk}^{21} & \mathbf{C}_{hk}^{22} & \cdots & \mathbf{C}_{hk}^{2N} \\ \dots & \dots & \dots & \dots \\ \mathbf{C}_{hk}^{N1} & \mathbf{C}_{hk}^{N2} & \cdots & \mathbf{C}_{hk}^{NN} \end{pmatrix} \quad \bar{\phi}_k = \begin{Bmatrix} \phi_k^1 \\ \phi_k^2 \\ \vdots \\ \phi_k^N \end{Bmatrix} \quad (4.61)$$

where the sub-matrix  $\mathbf{C}_{hk}^{11}$  is the single-bladed influence matrix shown in (4.60), the sub-matrix  $\mathbf{C}_{hk}^{12}$  stores coefficients for the influence of the doublets on blade two over blade one, and the sub-array  $\phi_k^1$  is the single-bladed array in (4.60) and stores the values of the velocity potential on blade one. The  $N$ -bladed matrix  $\bar{\mathbf{C}}_{hk}$  is thus  $N^2$  times larger than the corresponding one-bladed influence matrix.

Considerable savings in computer memory and processor time can be achieved by taking advantage of the physical symmetry of multi-bladed wind turbine rotors and solving for the  $\{\phi_k\}$  on *one* blade only. These are then copied for the other blades, which is a physically reasonable simplification. Implementing this is straight-forward,

and involves rearranging the sub-matrices in matrix equation (4.59) to take advantage of the repeated  $\{\phi_k\}$  and  $\left\{\left(\frac{\partial \phi}{\partial n}\right)_k\right\}$  vectors. The reduced N-bladed matrices in the linear system then take the form

$$\bar{\mathbf{C}}_{hk} \cdot \bar{\phi}_k \equiv (\mathbf{C}_{hk}^{11} + \mathbf{C}_{hk}^{12} + \cdots + \mathbf{C}_{hk}^{1N}) \cdot \phi_k^1 \quad (4.62)$$

which reduces the matrix storage and solution requirements of any N-bladed problem to that of a one-bladed problem, and the influence coefficient calculation overhead to N times that of the one-bladed problem. The advantages of this cannot be over-emphasised: a factor of  $N^2$  saving in computer memory *and* a correspondingly large saving in solution time can mean the difference between a problem being reasonably accommodated on available computing hardware or else being insoluble.

Equation (4.62) is currently implemented by declaring only  $B \times B$  matrices and length-B vectors and summing the corresponding coefficients directly in the coefficient matrix assembly routines. Their usage is outlined in the appendices on pages 457.

## 4.9 Two-dimensional code validation

Katz and Plotkin (2001) provide exact analytical solutions for a number of symmetric aerofoils. The Van de Vooren aerofoil with 15% thickness and a finite-angle trailing edge has been chosen as the two-dimensional test case for this project. Its geometry is shown in figure 4.18. Points on the surface of this two dimensional aerofoil in the x-z plane are described by a mapping from a circle centred at the origin the h-g plane with radius  $a$  (figure 4.19).

The co-ordinate mappings are as follows:

$$x = \frac{r_1^k}{r_2^{k-1}} [\cos(k\theta_1) \cos((k-1)\theta_2) + \sin(k\theta_1) \sin((k-1)\theta_2)] \quad (4.63)$$

$$z = \frac{r_1^k}{r_2^{k-1}} [\sin(k\theta_1) \cos((k-1)\theta_2) - \cos(k\theta_1) \sin((k-1)\theta_2)]$$

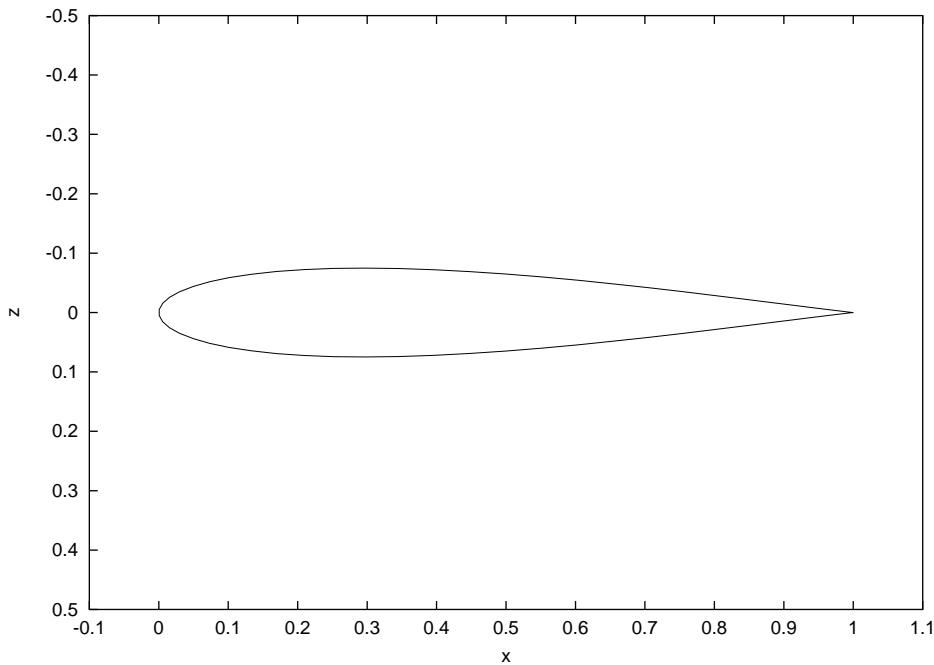


Figure 4.18: Van de Vooren Aerofoil (15% thick)

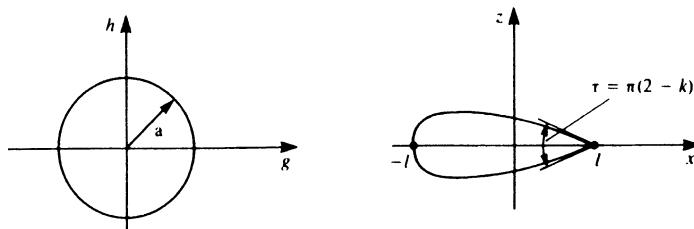


Figure 4.19: Aerofoil mapping (finite trailing edge angle) (Katz and Plotkin (2001))

where

$$\begin{aligned}
 r_1 &= \sqrt{(a\cos(\theta) - a)^2 + a^2 \sin^2(\theta)} \\
 r_2 &= \sqrt{(a\cos(\theta) - \varepsilon a)^2 + a^2 \sin^2(\theta)} \\
 \theta_1 &= \arctan\left(\frac{a \sin(\theta)}{a \cos(\theta) - a}\right) + \pi \\
 \theta_2 &= \arctan\left(\frac{a \sin(\theta)}{a \cos(\theta) - \varepsilon a}\right) + n_1 \pi
 \end{aligned}$$

where  $\varepsilon$  is a thickness parameter and  $n_1$  depends on the quadrant where  $\theta_2$  is being evaluated (0 first quadrant, 1 in the second and third quadrants and 2 in the fourth quadrant).

The velocity distribution around this aerofoil is then

$$u = 2U_0 \frac{r_2^k}{r_1^{k-1}} \frac{\sin(\alpha) - \sin(\alpha - \theta)}{D_1^2 + D_2^2} (D_1 \sin(\theta) + D_2 \cos(\theta))$$

$$w = 2U_0 \frac{r_2^k}{r_1^{k-1}} \frac{\sin(\alpha) - \sin(\alpha - \theta)}{D_1^2 + D_2^2} (D_1 \cos(\theta) - D_2 \sin(\theta))$$

where  $\alpha$  is the angle of attack and

$$A = \cos((k-1)\theta_1) \cos(k\theta_2) + \sin((k-1)\theta_1) \sin(k\theta_2)$$

$$B = \sin((k-1)\theta_1) \cos(k\theta_2) - \cos((k-1)\theta_1) \sin(k\theta_2)$$

$$D_0 = a(1 - k + k\epsilon)$$

$$D_1 = A(a \cos(\theta) - D_0) - B a \sin(\theta)$$

$$D_2 = A(a \sin(\theta)) + B(a \cos(\theta) - D_0)$$

The pressure coefficient is then calculated from the Bernoulli equation

$$C_p = 1 - \frac{u^2 + w^2}{U_0^2} \quad (4.64)$$

Setting  $\epsilon = 0.0655$  gives an approximate aerofoil thickness of 15%. For the current study, an arbitrary trailing-edge angle of  $15^\circ$  was chosen, giving the trailing-edge angle parameter,  $k$ , as:

$$k = 2 - \frac{(15\pi/180)}{\pi}$$

$$\approx 1.9167 \quad (4.65)$$

The mapping for the chord parameter  $a$ , from the h-k plane to the x-z plane is then:

$$a = chord(1 + \epsilon)^{k-1} 2^{-k} \quad (4.66)$$

which gives  $a \approx 0.280726$  for the unit-chord, 15% thick,  $15^\circ$  trailing-edge angle Van de Vooren aerofoil.

The procedure for testing the current panel method against the analytical solution was as follows: a set of sixty ordered-pairs of co-ordinates of a unit-chord, 15% thick Van de Vooren aerofoil were generated using equation (4.63); A B-spline curve fitted to these co-ordinates was generated using the least-squares curve-fitting technique; The

Differential Evolution function optimiser was applied to the B-spline aerofoil representation's control points to enhance its fit to the data points; An untapered (chord=1) very high aspect ratio ( $R=1000$ ) "wing" was created by skinning a bi-cubic B-spline surface over a spanwise distribution of the enhanced B-spline curves; the surface was discretised into a quadrilateral panel mesh, with  $N\_PANELS=120$  panels in the chordwise direction, and  $M\_PANELS=5$  span-wise blade elements; And a planar wake mesh emanating from the blade's trailing edge and parallel to the global X axis was generated (wake length = 100,  $W\_PANELS=100$ ).

Blade panel density is most important in the chordwise direction, with the highest concentration of panels required at the leading edge to capture the curvature in as much detail as possible. Spanwise panel density (that is, number of blade elements), is not critical for this sort of quasi-2D evaluation, and a few blade elements ( $> 1$ ) will do the same job as many, just as long as there are an odd number of them and that the required 2D pressure values are collected from the mid-span collocation points. Wake panel density is similarly not critical, but the length of the wake should approximate "infinity" - that is, the wake should be very long with respect to the chord length.

The pressure at the mid-span of this wing ( $\frac{r}{R}=0.5$ ) will approach the two-dimensional solution as the aspect ratio and the wake length increase. The panel method was applied to this geometry over a range of  $\alpha$ , with the results shown in figures 4.20.1 to 4.20.8. The predicted pressure distributions match the analytical solutions quite accurately at a panel density of  $N\_PANELS=120$ . Small pressure deviations (most notably at the leading edge) are seen in regions where the geometrical error between the actual curve and the compact B-spline representation is highest. More effort devoted to optimising B-spline leading-edge smoothness and fit would result in better pressure approximations, but this was avoided for two reasons: the chordwise panel densities employed for the optimisation work were much lower - 30 to 40 panels - resulting in larger leading edge panels which bridge over small surface irregularities, and which also negate the effort put into accurately modelling very high curvature; and the accurate modelling of high-curvature is easily accommodated by B-spline curves, but at the cost of curve compactness - more sub-curves are needed. Since compact aerofoil representations are a necessary component of the current optimisation method, small pressure inaccuracies are a small price to pay.

Panel method predictions at an angle of attack of  $\alpha=5^\circ$  at various panel densities

appear in Figures 4.21.1 to 4.21.6. Varying chord-wise panel density has the expected effects: as the number of panels along the surface of the aerofoil (N\_PANELS) is reduced the areas of large pressure gradient are smoothed, with, for example, the suction-side leading edge pressure spike progressively being less well predicted. Increasing the number of N\_PANELS increases the accuracy but only up to the limit of an individual B-spline representation's goodness of fit to point data and smoothness. The results presented for the B-spline aerofoil with N\_PANELS=120 are realistically close to the best possible for this particular geometry. An application requiring greater accuracy can be accommodated by simply improving the fit and smoothness of the B-spline aerofoil representation.

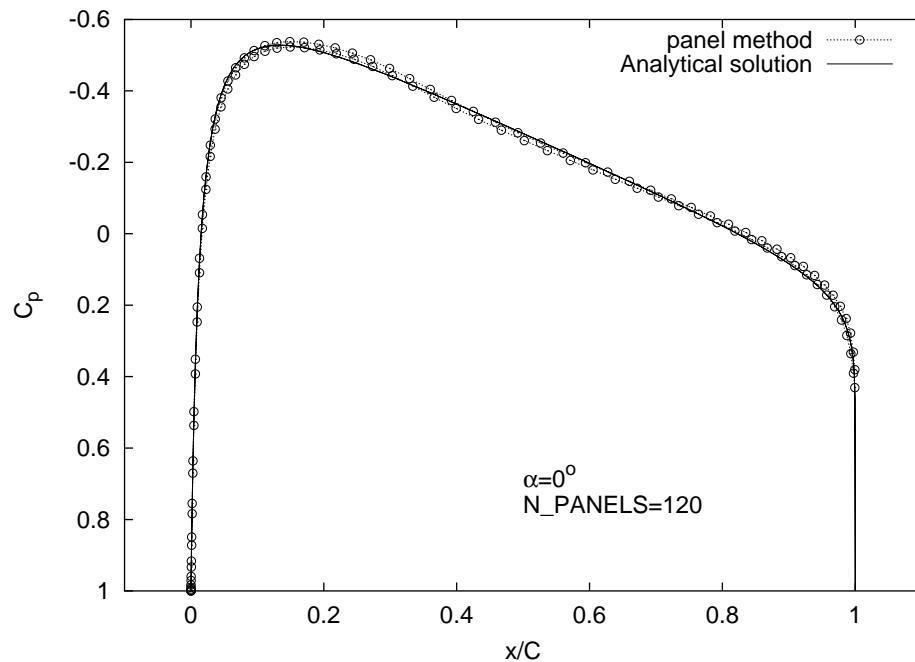
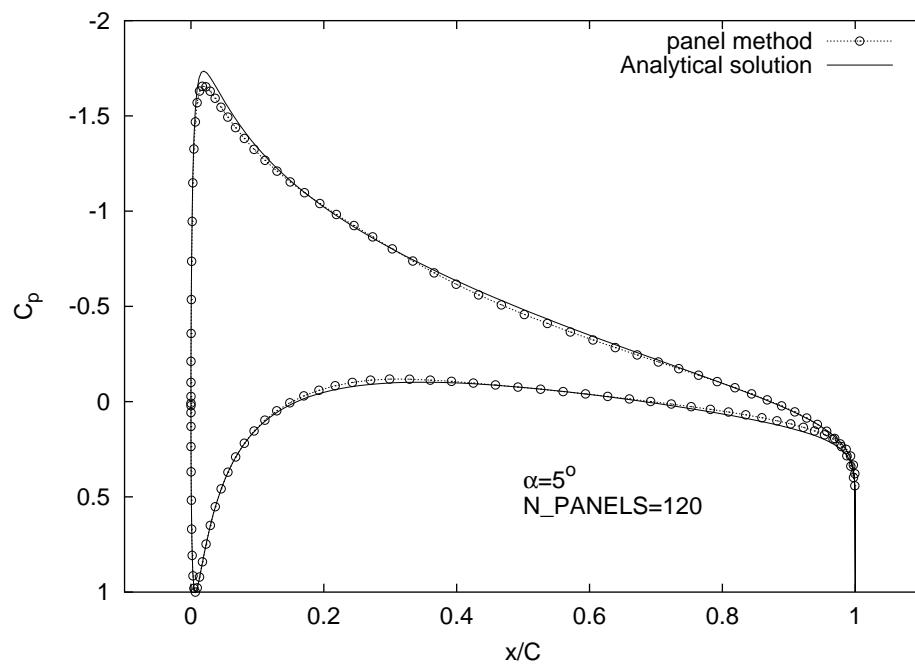
4.20.1:  $\alpha = 0$  degrees4.20.2:  $\alpha = 5$  degrees

Figure 4.20: Analytical vs Panel method pressure distributions for the Van de Vooren aerofoil

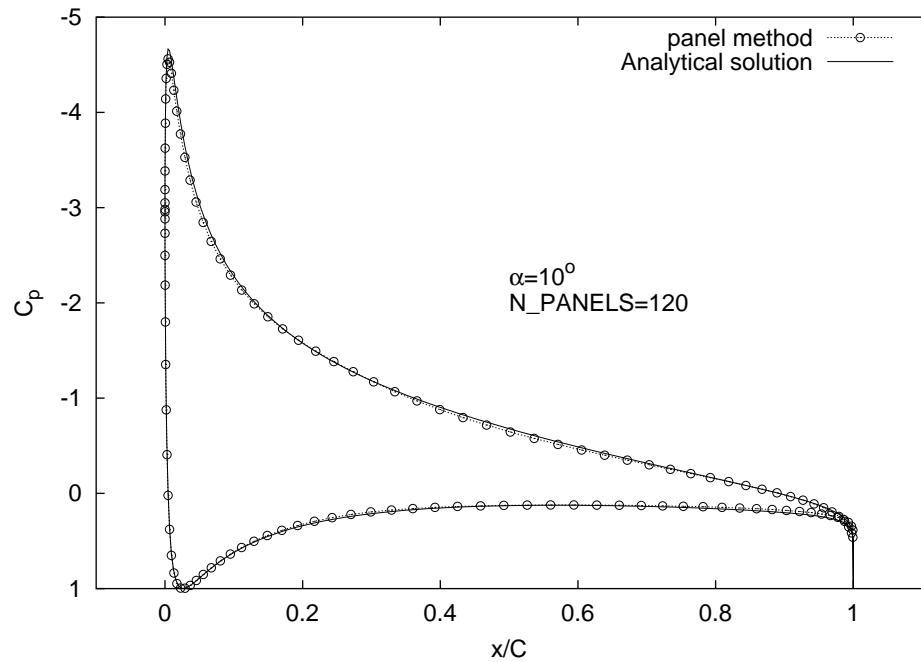
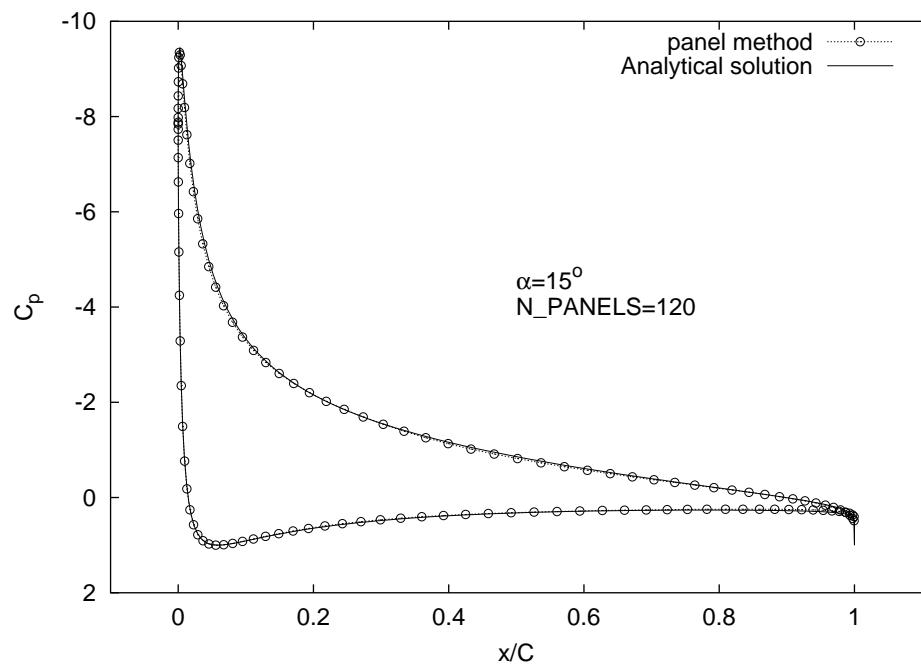
4.20.3:  $\alpha = 10$  degrees4.20.4:  $\alpha = 15$  degrees

Figure 4.20: Van de Vooren aerofoil pressure comparisons (2 of 4)

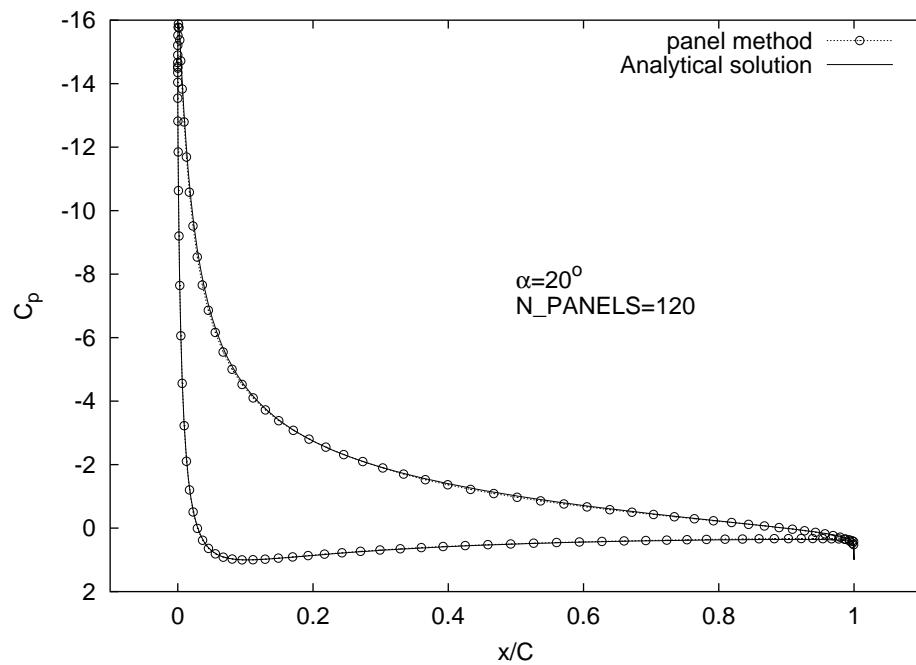
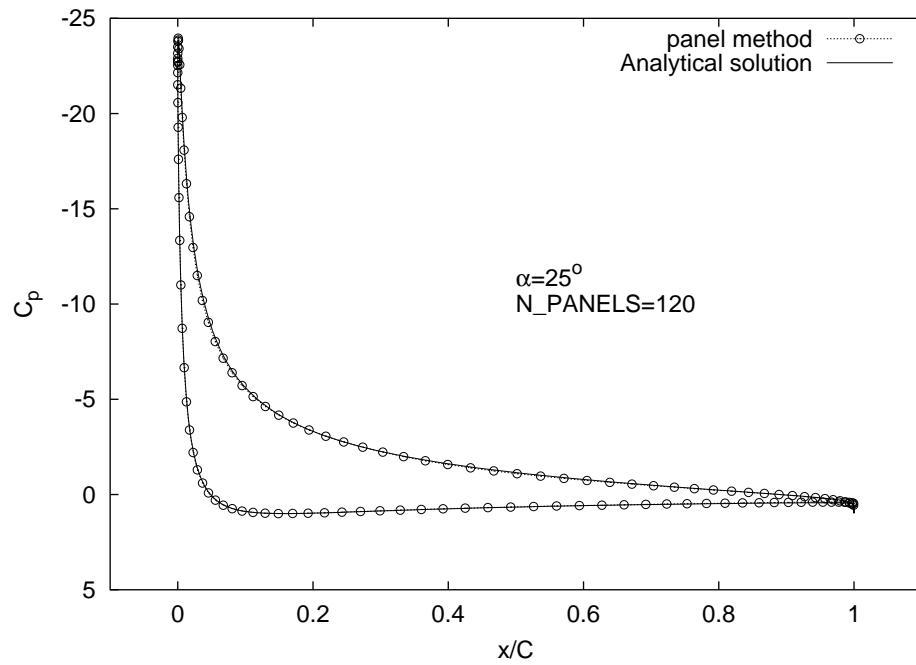
4.20.5:  $\alpha = 20$  degrees4.20.6:  $\alpha = 25$  degrees

Figure 4.20: Van de Vooren aerofoil pressure comparisons (3 of 4)

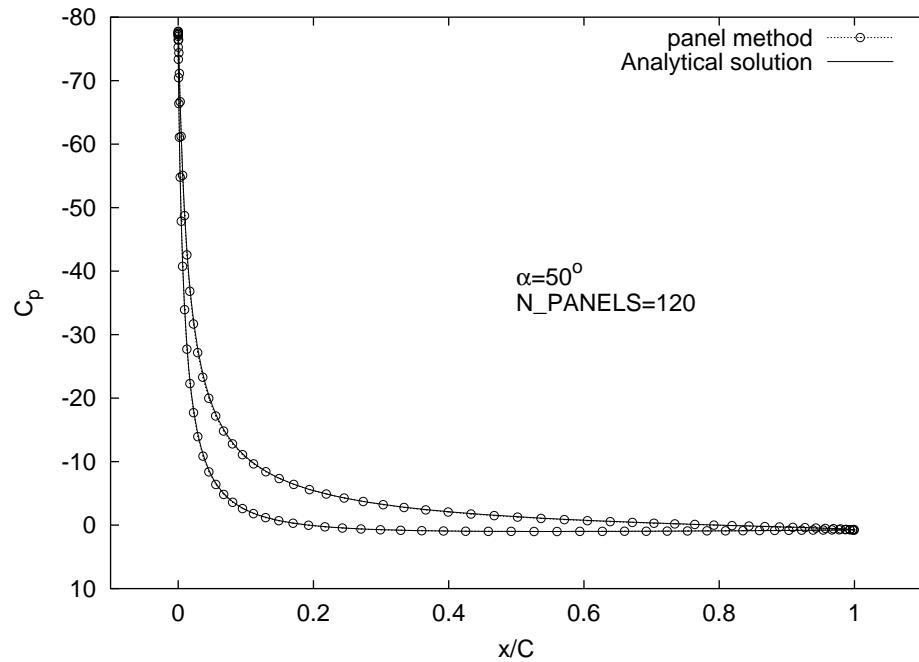
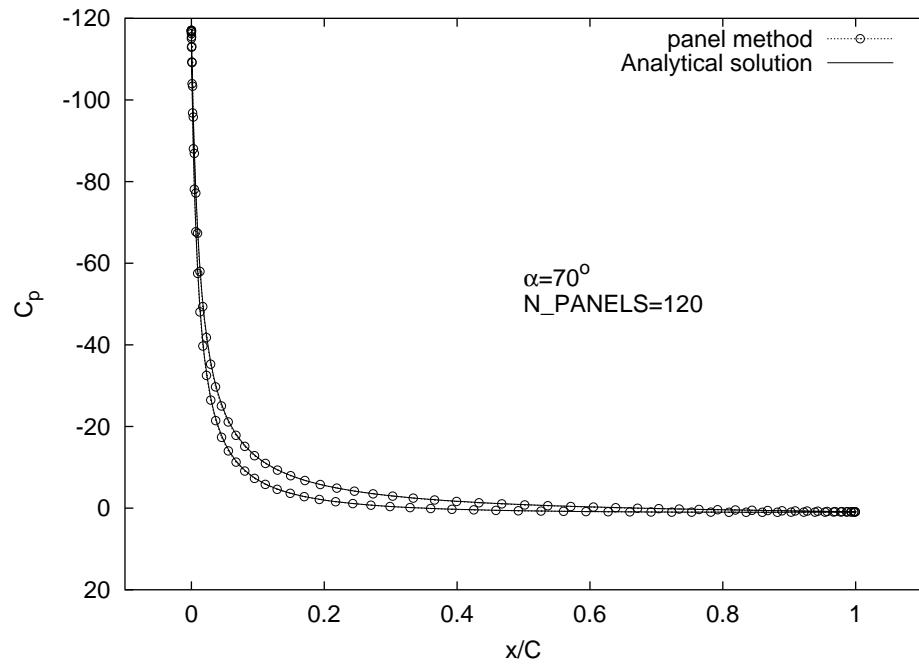
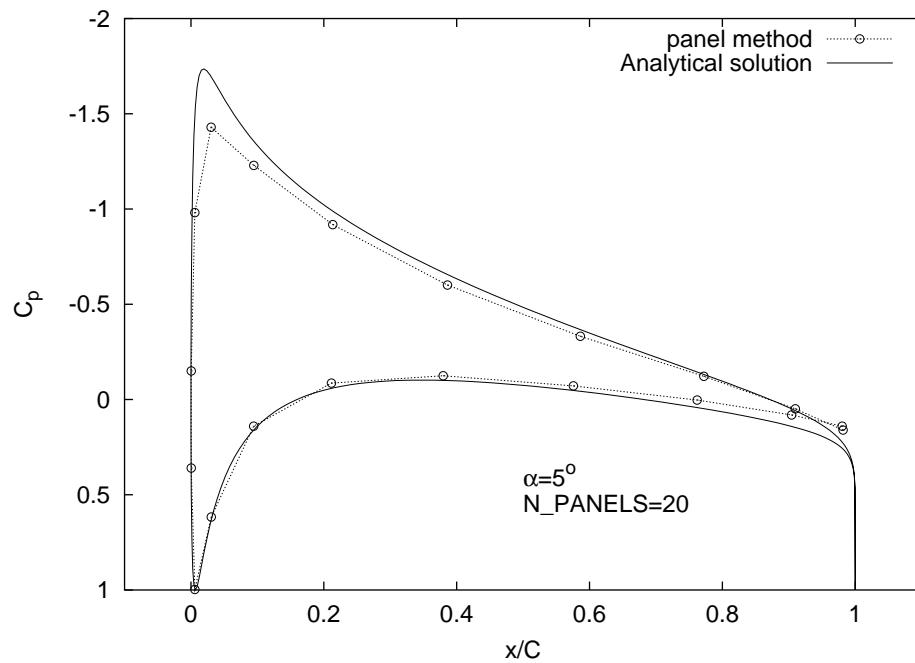
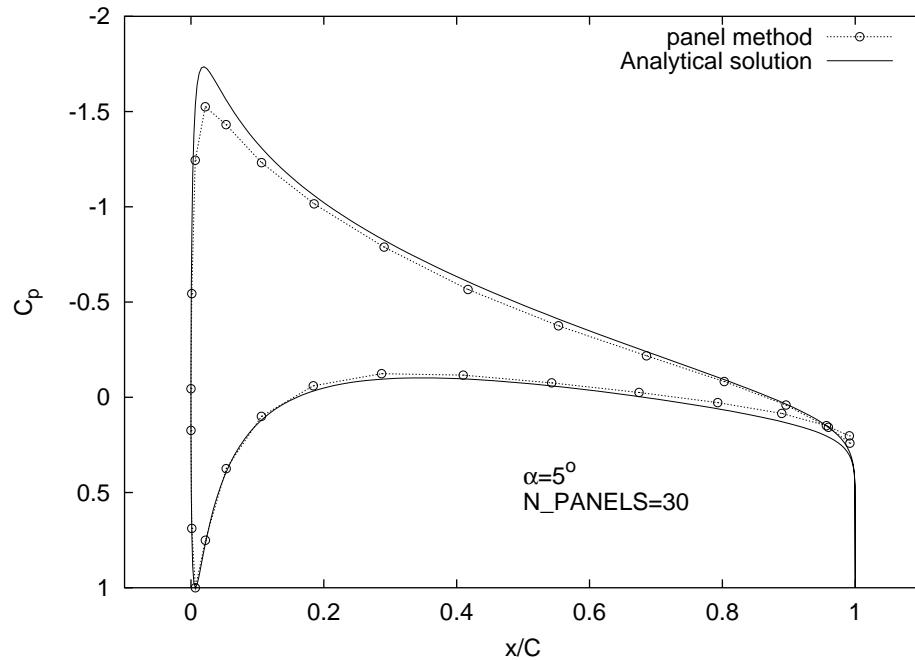
4.20.7:  $\alpha = 50$  degrees4.20.8:  $\alpha = 70$  degrees

Figure 4.20: Van de Vooren aerofoil pressure comparisons (4 of 4)



4.21.1: N\_PANELS = 20



4.21.2: N\_PANELS = 30

Figure 4.21: Dependence of panel method accuracy on panel density

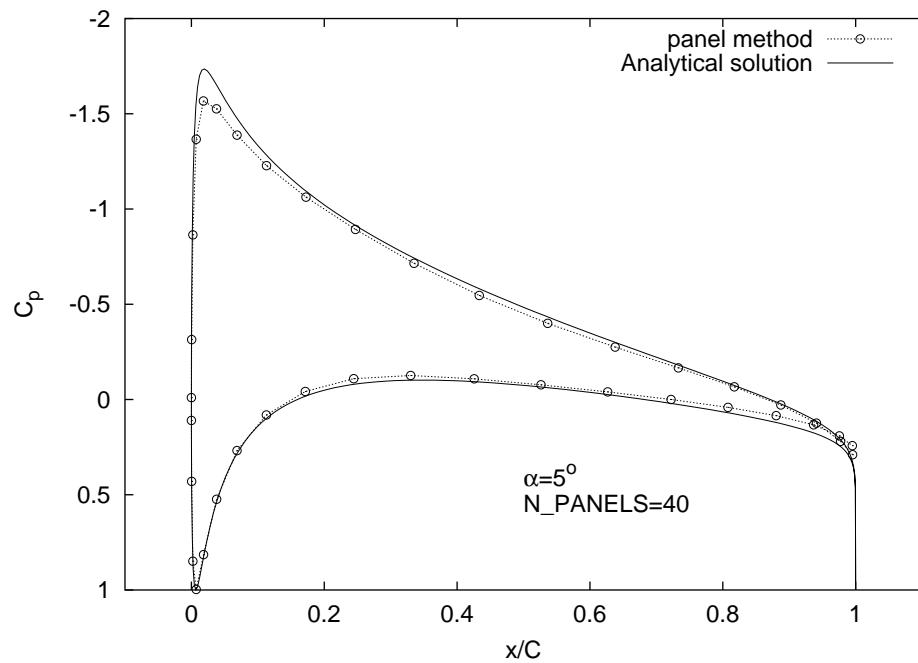
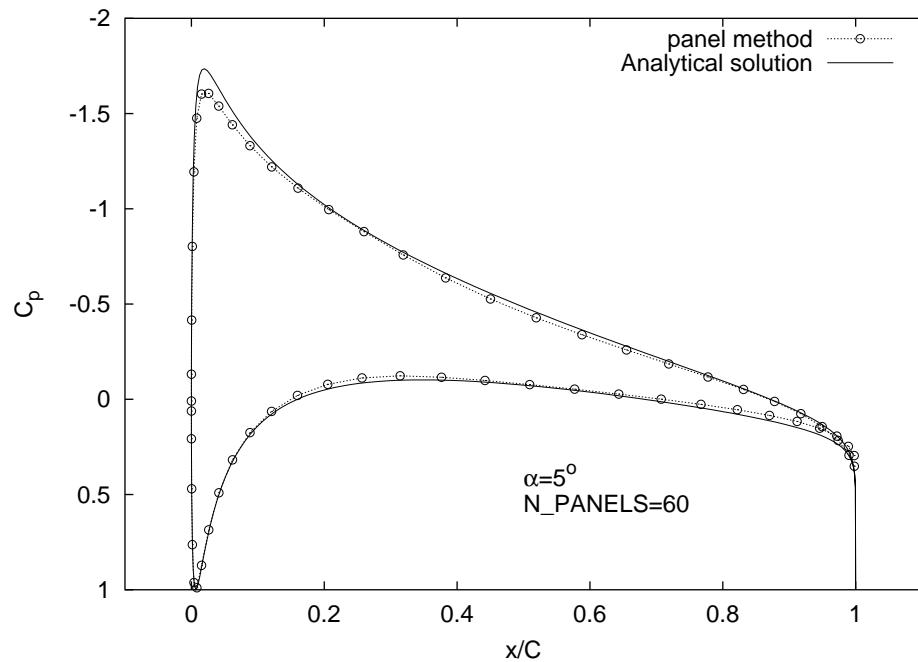
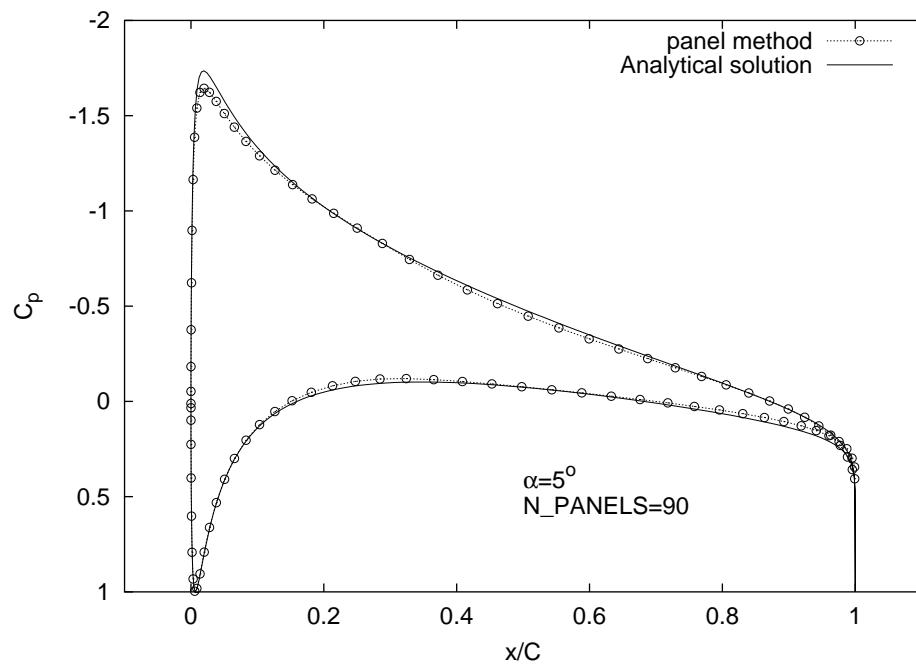
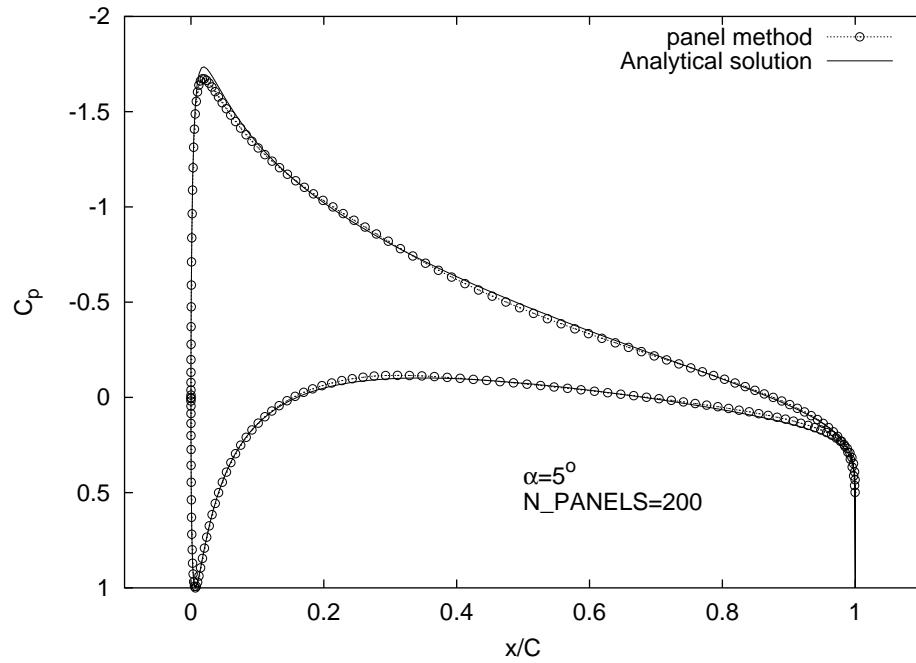
4.21.3:  $N_{\text{PANELS}} = 40$ 4.21.4:  $N_{\text{PANELS}} = 60$ 

Figure 4.21: Dependence of panel method accuracy on panel density (2 of 3)



4.21.5: N\_PANELS = 90



4.21.6: N\_PANELS = 200

Figure 4.21: Dependence of panel method accuracy on panel density (3 of 3)

The circulation around a Joukowski aerofoil is given analytically by Katz and Plotkin (2001) as:

$$\Gamma = 4\pi a U_0 \sin(\alpha + \beta) \quad (4.67)$$

where, for the 15% thick, 15° trailing-edge angle symmetric Van de Vooren aerofoil

$$\begin{aligned} a &= chord(1 + \varepsilon)^{k-1} 2^{-k} \\ &\approx 0.280726 \\ \beta &= 0 \end{aligned} \quad (4.68)$$

By the Kutta-Joukowski theorem, the force acting normal to the free-stream (the lift) experienced by an aerofoil is dependent on the circulation around the aerofoil, and is given by:

$$L = \rho U_0 \Gamma \quad (4.69)$$

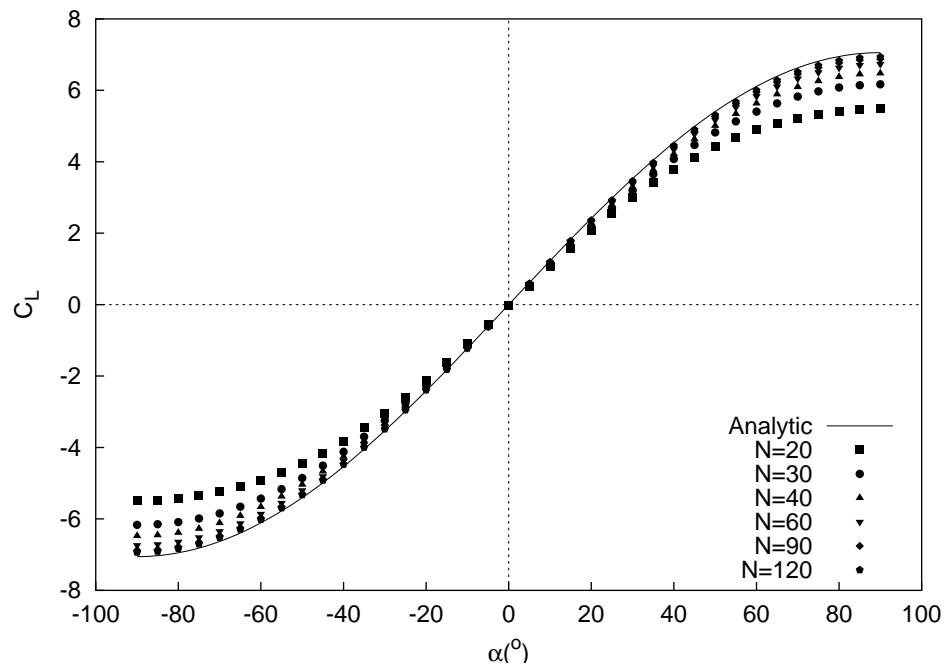
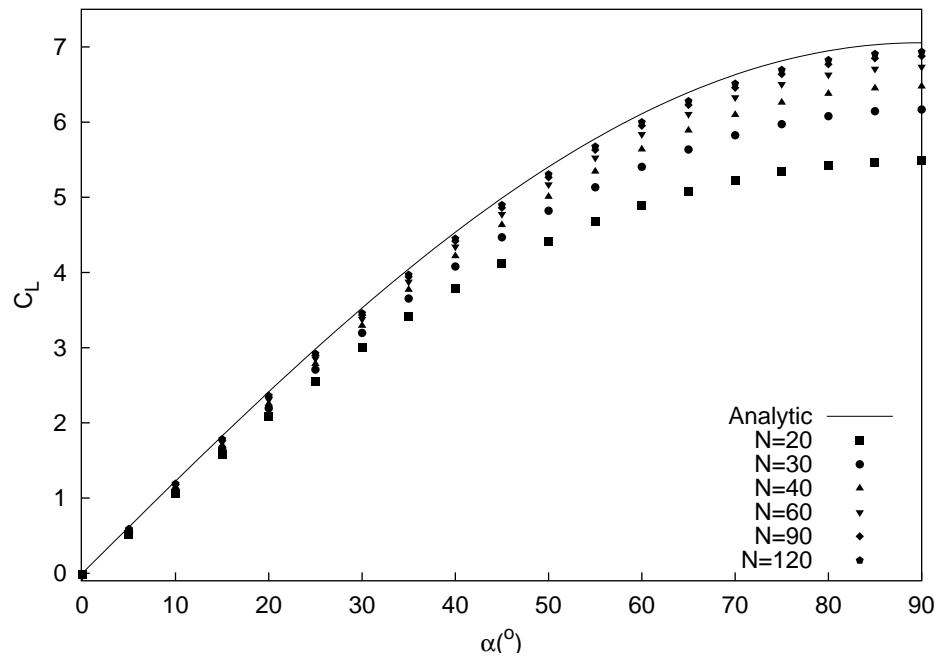
The analytical lift coefficient is then

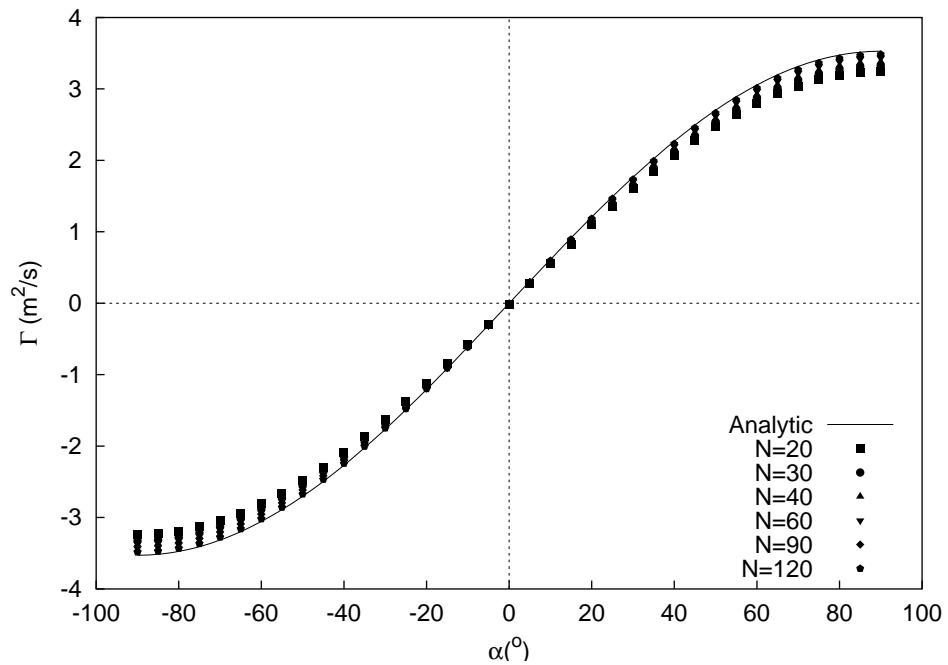
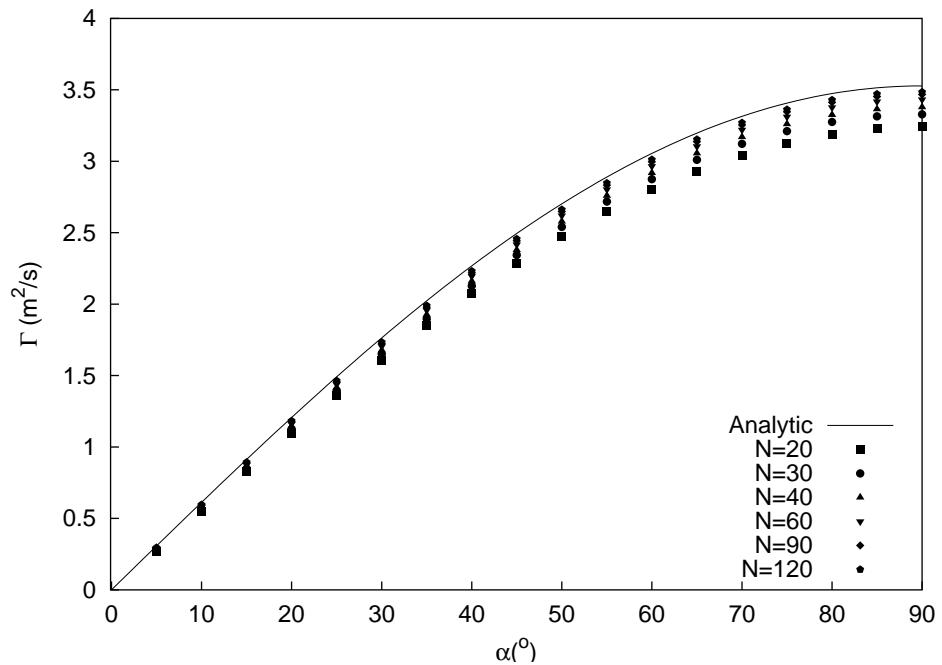
$$C_L = \frac{L}{\frac{1}{2} \rho U_0^2 chord} \quad (4.70)$$

$$= 8\pi \frac{a}{chord} \sin(\alpha) \quad (4.71)$$

Figures 4.22.1 and 4.22.2 compare this analytic lift coefficient result with panel method predictions at various panelling densities, across a range of angles of attack. As expected, the predicted lift accuracy increases with increasing panel density, with the highest panel density (N\_PANELS=120) providing very good agreement.

Even closer agreement is provided by the circulation predictions across the studied  $\alpha$  range. These results are shown in Figures 4.23.1 and 4.23.2. Given that the accuracy of the circulation calculation is vitally important for the iterative helical wake model used in the three-dimensional wind turbine performance predictions, the fact that predicted  $\Gamma$  values are close to the analytical 2-D solution is an important validation of the prediction code.

4.22.1:  $C_L$  vs  $\alpha$ :  $-90 \leq \alpha \leq 90$ 4.22.2:  $C_L$  vs  $\alpha$ :  $0 \leq \alpha \leq 90$ Figure 4.22: Van de Vooren lift coefficient vs  $\alpha$  at various panel densities

4.23.1:  $\Gamma$  vs  $\alpha$ :  $-90 \leq \alpha \leq 90$ 4.23.2:  $\Gamma$  vs  $\alpha$ :  $0 \leq \alpha \leq 90$ Figure 4.23: Van de Vooren circulation vs  $\alpha$  at various panel densities

## 4.10 Three-dimensional code validation

### 4.10.1 A simple correction for viscosity

The ultimate measure of the panel method's performance will always be its prediction of the aerodynamic loads and power output of real wind turbines. Which raises the obvious and necessary question: how can a fundamentally *inviscid* solution technique ever hope to model the actual fluid mechanics of the high angle of attack, low Reynolds number flows (and the accompanying phenomena of separated boundary layers and stall) experienced by even large, constant-speed wind turbines? That is, how do we deal with viscosity?

The short answer is that a physically accurate treatment of viscous flows is beyond the capabilities of the current panel method solver, with only a very simple and computationally inexpensive empirical correction able to be considered. Inviscid aerodynamic solvers are capable of calculating accurate lift distributions for high Re and moderate  $\alpha$  for a wide range of aerofoil sections, but these predictions increasingly become less accurate as Re falls and sectional  $\alpha$ -s rise. This is a major problem for the low (and very low) Reynolds numbers encountered during the operation of small wind turbines, especially during starting. Viscous-inviscid interaction methods hold great promise for extending the prediction accuracy of boundary element methods whilst maintaining their economy, but these fall outside the scope of this thesis. A reader with a will to improve the viscous performance of the current prediction method is encouraged to look to Drela and Giles (1987), Cebeci (1998), Cebeci and Cousteix (1999), Besnard et al. (1998) and Cebeci and Besnard (1998) for practical implementations of Viscous-Inviscid Interaction methods.

Having thus deferred the inclusion of a physically-rigorous viscous model, a simple empirical correction to the boundary condition of the panel method has been applied which, as will be shown in the following test cases, improves the method's predictive capabilities. The correction is a simple (and simplistic) attempt to model the effect of viscosity on each spanwise “blade element” as a function of the sectional angle of attack and Reynolds number.

The correction takes as inspiration the *transpiration velocity* concept which, together with an unimplemented boundary layer thickness measure, forms the basis of the aforementioned viscous-inviscid interaction methods. The transpiration, or blow-

ing, velocity is a correction to the Neumann boundary condition (equation 4.21)

$$\frac{\partial \phi}{\partial n} = -\mathbf{U}_W \cdot \mathbf{n}$$

which, with the empirical correction applied becomes

$$\left( \frac{\partial \phi}{\partial n} \right)^* = (-\mathbf{U}_W \cdot \mathbf{n})(1 - K) \quad (4.73)$$

in which the blowing velocity,  $V_b$ , is applied as a fraction,  $K$ , of the normal velocity at the boundary necessary to impose the inviscid boundary condition (figure 4.24).

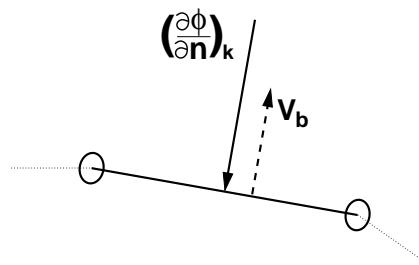


Figure 4.24: Scheme for “viscous” boundary condition correction

For each spanwise blade element,  $K$  is calculated as a basic function of the sectional angle of attack and Reynolds number:

$$f(\alpha) = 0.00075|\alpha|^2$$

$$f(Re) = 0.2e^{-5 \times 10^{-6}Re}$$

$$K = \begin{cases} f(\alpha) + f(Re) & 0. < K < 0.95 \\ 0 & K < 0. \\ 0.95 & K \geq 0.95 \end{cases} \quad (4.74)$$

and applied to the Neumann boundary condition at the centroid of each of the element’s panels. The numerical constants used in the correction were determined by trial-and-error to give a “reasonable” fit to lift and drag polars for two real aerofoil sections: the SD7062 aerofoil (Lyon et al. (1998)) and the NACA 4412 aerofoil (Miley (1982)). The procedure is implemented in function 4.1.

---

 Function 4.1: viscous\_boundary.cc
 

---

```

void viscous_boundary_correction(blade_mesh& X){
    //a function which applies an empirical viscous boundary correction
    int i,j,k,blade,element;
    double f_Re, f_alpha, func;
    //adjust the boundary condition based on the local
    //Reynolds number and angle of attack
    k=0;
    for(blade=0; blade<NB; blade++){
        for(element=0; element<M_PANELS; element++){
            f_alpha=0.00075*pow(fabs(X.element[element].alpha),2.);
            f_Re=0.2*exp(-5.0*X.element[element].Re/1.E6);
            func= f_alpha+f_Re;
            if (func<0.0)func=0.0; if(func>0.95)func=0.95;
            for(j=0; j<N_PANELS; j++){
                X.blade_panel[k].dphi_dn *= (1.-func);
                k++;
            }
        } //end of 'element' loop
    } //end of 'blade' loop
    return;
}
  
```

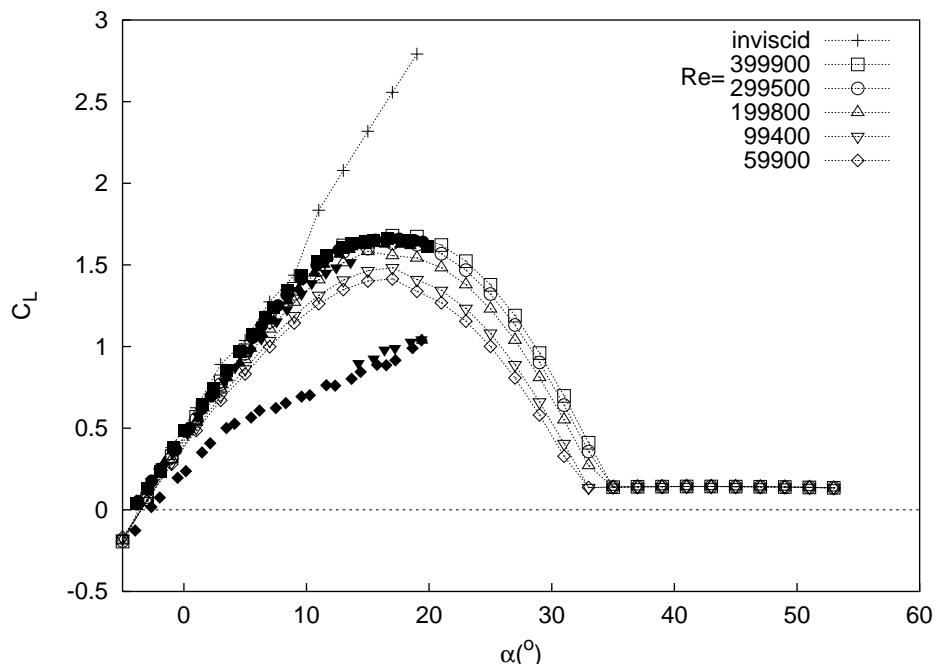
---

Function 4.1 is called immediately after the calculation of the inviscid boundary condition. Its effect on sectional lift and drag is illustrated in figures 4.25.1 and 4.26.1 for the SD7062 aerofoil, and figures 4.25.2 and 4.27.1 for the NACA 4412 aerofoil. No specific attempt was made to model drag correctly, this being a task that a future Viscous/Inviscid Interaction correction may go some way to solving, but also one which is exceedingly complex and expensive, even for state-of-the-art CFD codes such as grid-based Navier-Stokes solvers running on high-end computing hardware. Nevertheless, the correction serves the goal of modelling some of the *qualitative* features of the effects of viscosity on the forces experienced by spanwise blade elements. Figures 4.25.1 and 4.25.2 show predictions of lift coefficient over a range of angles of attack and Reynolds number at the mid-span of a straight, untwisted “wing” of very high aspect ratio (on the order of 1000). It can be seen that lift predictions for the inviscid solver are acceptable for moderate  $\alpha$  (<10 degrees) for a wide range of low-to-moderate Reynolds numbers. This accuracy degrades rapidly with rising  $\alpha$  and falling Re. The empirical correction provides a model for this degradation in lift, at least qualitatively,

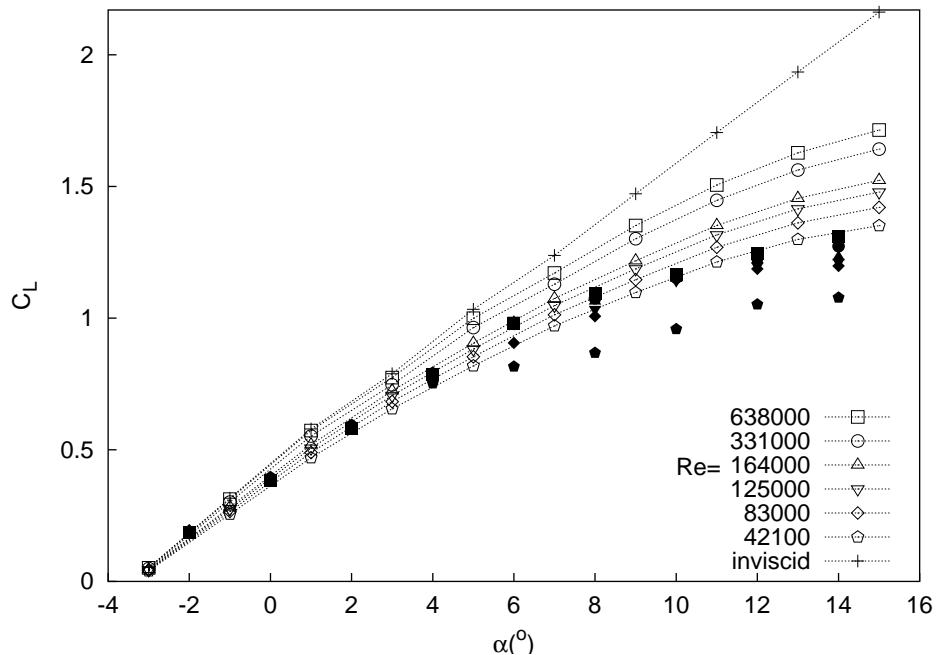
up to an angle of attack of approximately  $35^\circ$ . The lift coefficient at values of  $\alpha$  greater than this can be seen to remain constant at a small, non-zero figure. This result is due to the upper-limit imposed on the magnitude of the “blowing” velocity (which was set arbitrarily at 95% of the boundary condition).

Figures 4.26.1 and 4.27.1 show that drag is being grossly underestimated, compared with the experimental lift and drag polars presented in (figures 4.26.2 and 4.27.2). This result perhaps isn’t surprising given that the underlying method is, at its heart, inviscid and the correction so unsophisticated. The correction was “tuned” by choosing the three numerical constants in (4.74) to give reasonable lift and drag performance over the available ranges of experimental Reynolds number and angle of attack data.

The correction, in and of itself, is not terribly useful for the prediction of actual viscous aerodynamic flows. Its utility lies entirely within the scope of the current application: it subjects the evolving blade population to additional *selection pressure* for Reynolds number and angle of attack performance in a reasonable (if not physically accurate) way, so that the optimisation algorithm has a method for distinguishing between competing blade designs on the basis of local viscous effects. And it does so in computationally inexpensive way - a crucial requirement for the iterative optimisation process described in subsequent chapters.

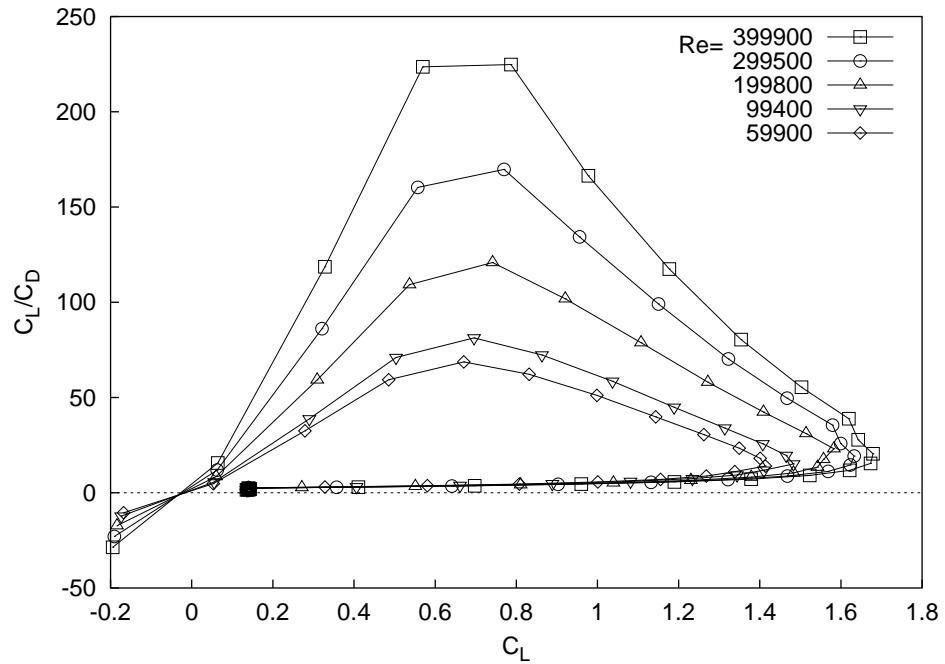


4.25.1:  $C_L$  predictions via panel method with simple viscous correction (open symbols) vs experimental results (closed symbols) for the SD7062 aerofoil (Lyon et al. (1998))

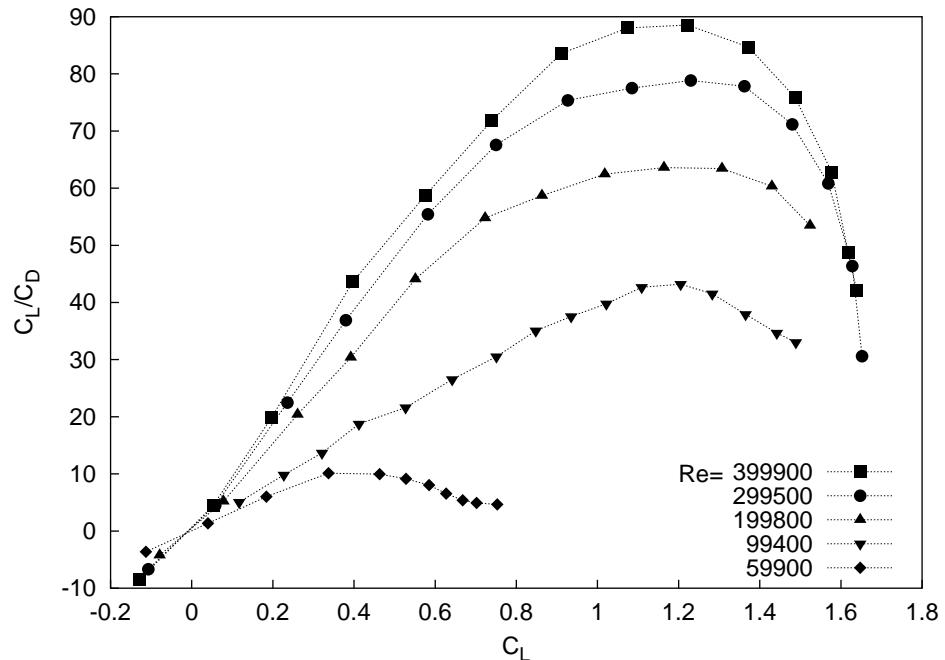


4.25.2:  $C_L$  predictions via panel method with simple viscous correction (open symbols) vs experimental results (closed symbols) for the NACA 4412 aerofoil (Miley (1982))

Figure 4.25: Computed and experimental lift coefficients: SD7062 and NACA 4412 aerofoils

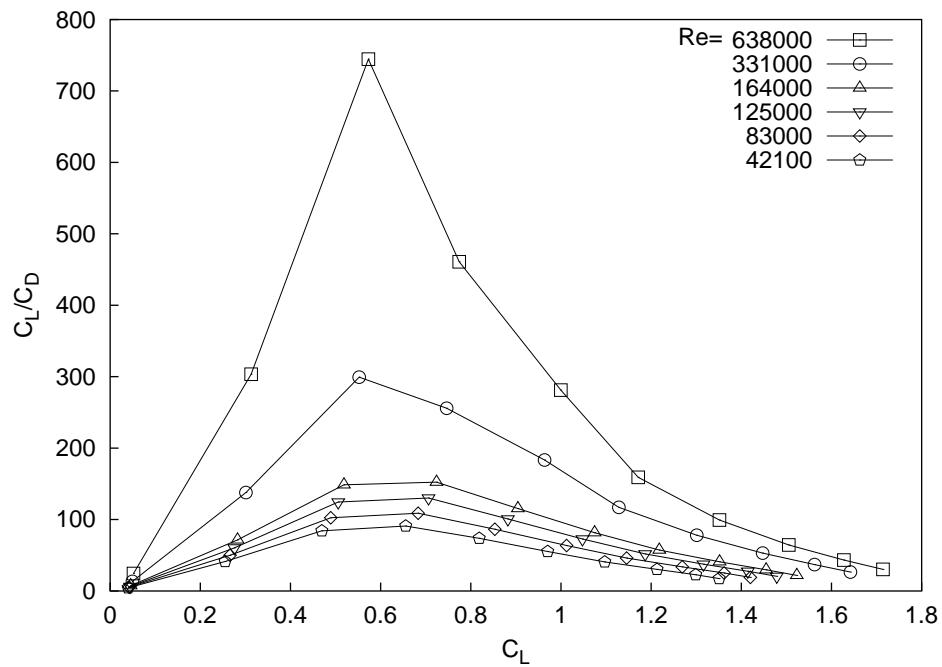


4.26.1: Lift and drag polar via the panel method with simple viscous correction for the SD7062 aerofoil

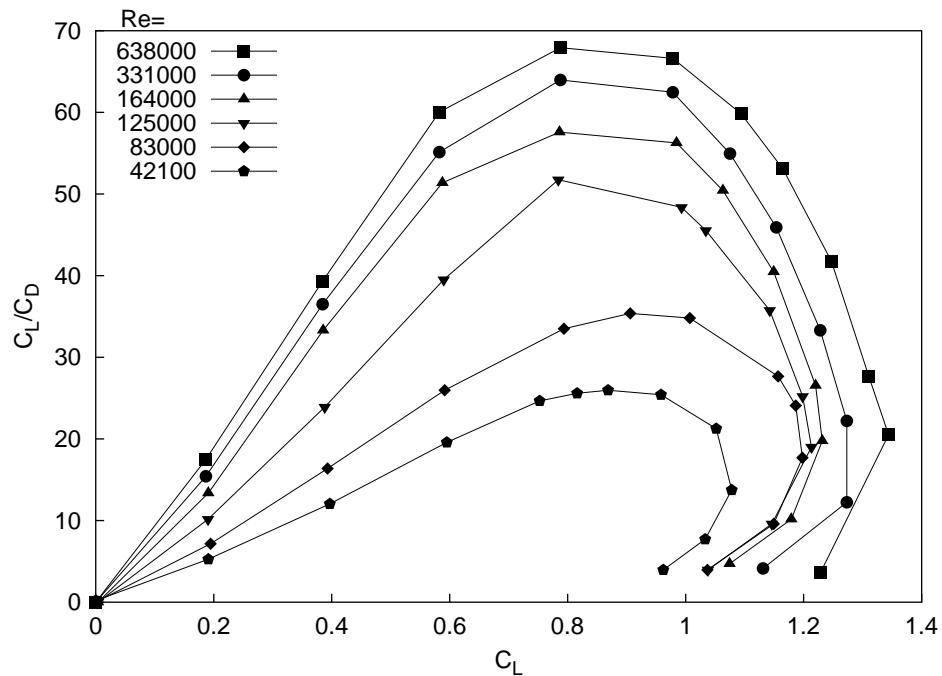


4.26.2: Experimental lift and drag polar for the SD7062 aerofoil (Lyon et al. (1998))

Figure 4.26: Computed and experimental lift and drag polars: SD7062 aerofoil



4.27.1: Lift and drag polar via the panel method with simple viscous correction for the NACA 4412 aerofoil



4.27.2: Experimental lift and drag polar for the NACA 4412 aerofoil (Miley (1982))

Figure 4.27: Computed and experimental lift and drag polars: NACA 4412 aerofoil

### 4.10.2 Test case: large constant-speed wind turbine

The geometry of the LM19.1 wind turbine blade was chosen to test the panel method in three-dimensions. The blade is used on the commercial three-bladed, 41-metre diameter Nordtank 500 kW wind turbine. Sørensen (1999) details the results of computational performance studies undertaken by three European wind energy research partners (Flygtekniska Försökanstalten (FFA) from Sweden, Deutsche Forschungsanstalt für Luft- und Raumfahrt - Braunschweig (DLR-B) from Germany and Risø National Laboratory from Denmark) which provide convenient comparison data for the current validation. Sørensen (1999) also provide diagrams of five aerofoil sections and a planform of the blade (figures 3.5 and 3.6) from which a (slightly crude) three-dimensional bi-cubic B-spline approximation was constructed using the methods described in the previous chapter. This surface was subsequently discretised into a mesh of quadrilateral surface panels ( $NB=3$ ;  $N\_PANELS=40$ ;  $M\_PANELS=20$ ). The current computational results were compared with those of the FFA, DLR-B and Risø research groups. To this end, the constant-speed Nordtank rotor ( $\Omega=27.1$  RPM  $\approx 2.838$  rad/s) was studied over a range of wind speeds ( $U_0 \in \{5, 6, \dots, 15\}$  m/s), with a new wake mesh generated for each wind speed ( $TURNS=5$ ;  $W\_PANELS=10$ ). The methods of iterating for the wake pitch and trailing edge pressure equalisation were employed.

Figure 4.28 compares the predicted power curve of the current panel method with the experimental performance data and computational predictions presented in Sørensen (1999). Each of three European wind energy partners employed different viscous CFD codes to solve the full Navier Stokes equations for the flow around one blade, with periodic boundary conditions to exploit the symmetry of the three-bladed rotor. All calculations were assumed to be fully turbulent, with each partner using a different turbulence model. Each computational domain for the CFD flow solvers extended approximately three rotor diameters up and downstream of the rotor disc, with a diameter of three rotor diameters. The solver used by FFA contained 300,000 computational cells, DLR-B used 1,400,000 cells and Risø 900,000 cells. The authors do not report on the computational environments used to perform these simulations, but describe the 3D computations as “very time consuming”. It can be assumed that the computational effort required by these solution techniques, with so many simultaneous equations to solve, is orders of magnitude greater than that required by a boundary-

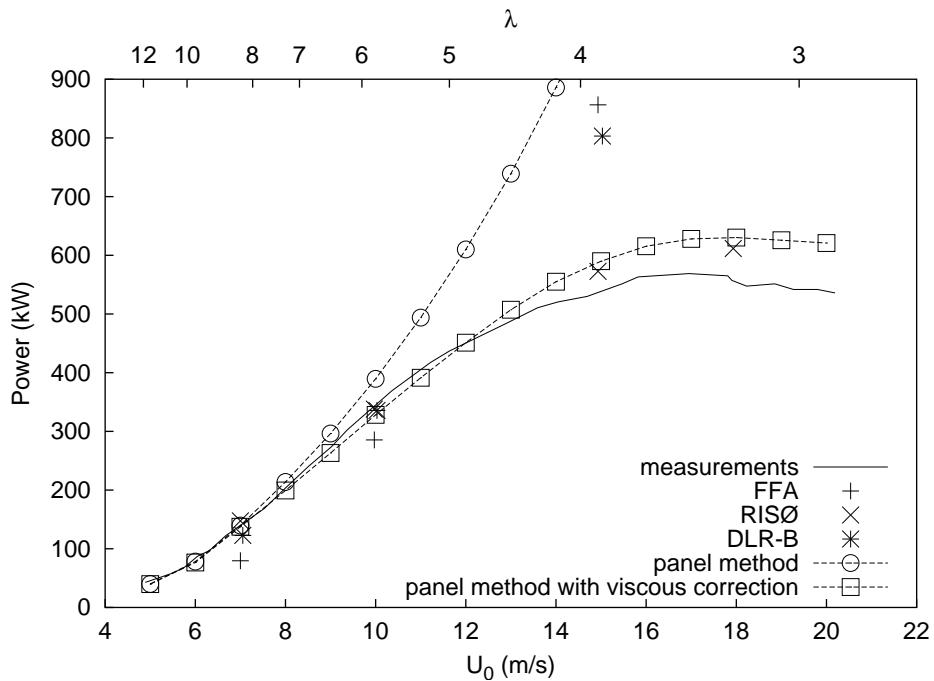


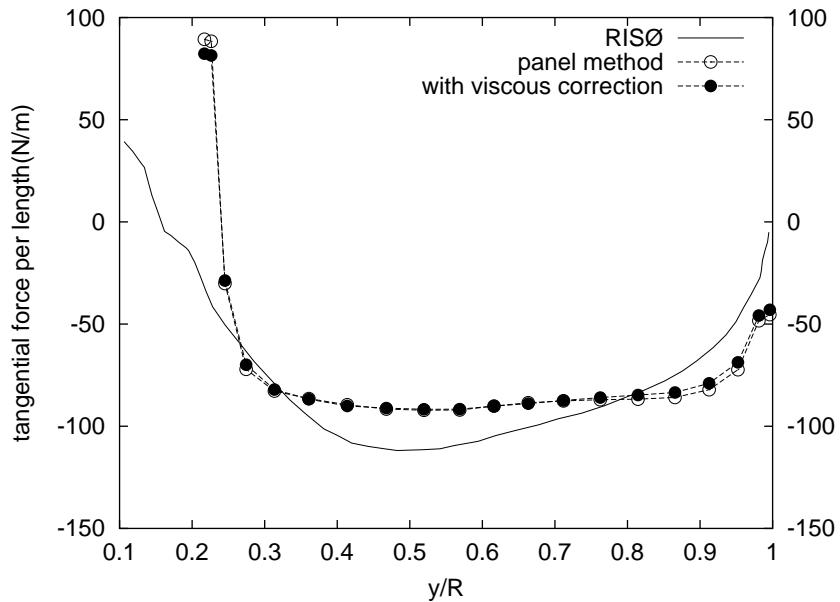
Figure 4.28: Comparison with results from Sørensen (1999)

element solver.

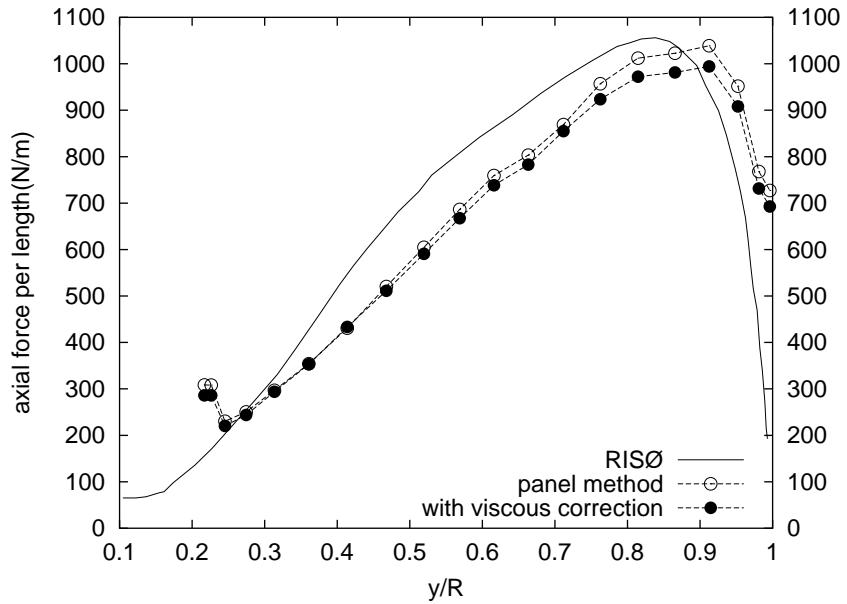
The inviscid panel method can be seen to adequately compete with the viscous flow solvers for the high  $\lambda$  values produced at low wind speeds for this turbine. High  $\lambda$  flows result in low  $\alpha$  angles experienced by the blade and thus strongly attached flow. This is adequately handled by the inviscid panel method. Panel method accuracy degrades as  $\lambda$  decreases and  $\alpha$  angles increase (particularly near the blade root) and the viscous effects of flow separation and stall are exhibited. These are neglected by the panel method, which assumes attached flow. It should be noted that successful prediction of viscous effects is by no means guaranteed by the turbulence models employed by viscous flow solvers, as is demonstrated by the power predictions of the FFA solver and, to a lesser extent, the DLR-B and Risø solvers. The simple empirical correction greatly improves the panel method's power predictions as the tip speed ratio decreases and the sectional angles of attack increase.

Sørensen (1999) provides tangential and axial force distributions for the more accurate Risø simulations at three wind speeds: 7, 10 and 15 m/s. Comparisons with this data appear in figures 4.29.1 through 4.29.6. Good qualitative agreement in the

tangential and axial force distribution comparisons for the lowest wind speed (highest  $\lambda$ ) is mirrored by the associated accurate power prediction by the panel method. The inviscid panel method increasingly over-predicts tangential (and axial) force as the tip speed ratio declines, resulting in a related over-prediction for the produced power. Figures 4.30.1 through 4.32.5 illustrate comparisons between the Risø pressure distribution calculation at discrete span positions for each of the studied wind speeds, as reported in Sørensen (1999). Comparison of the panel code results against this data sheds some light on the discrepancies apparent in the axial and tangential force comparisons. It is evident that, at least for the  $U_0 = 7$  m/s case and to a slightly lesser extent the  $U_0 = 10$  m/s case, the inviscid panel code predicts the correct pressure distributions near the blade tip. This is to be expected, since the local  $\alpha$  values decrease as the local speed ratios increase, with an accompanying greater likelihood of attached flow. The predicted pressure distributions degrade as the blade root is approached, with the combination of very thick aerofoil section, high  $\alpha$  and low local speed ratio resulting in local flow separation. The over-prediction in suction-side pressure evident in all of the comparisons where flow separation is present (notably, the entire series of  $U_0 = 15$  m/s comparisons) is typical of the results provided by an inviscid solver operating in a viscous flow regime. These over-predictions are accounted for, at least in a qualitative sense, by the empirical viscous correction and point to the likely success of a future viscous-inviscid interaction enhancement to the code.

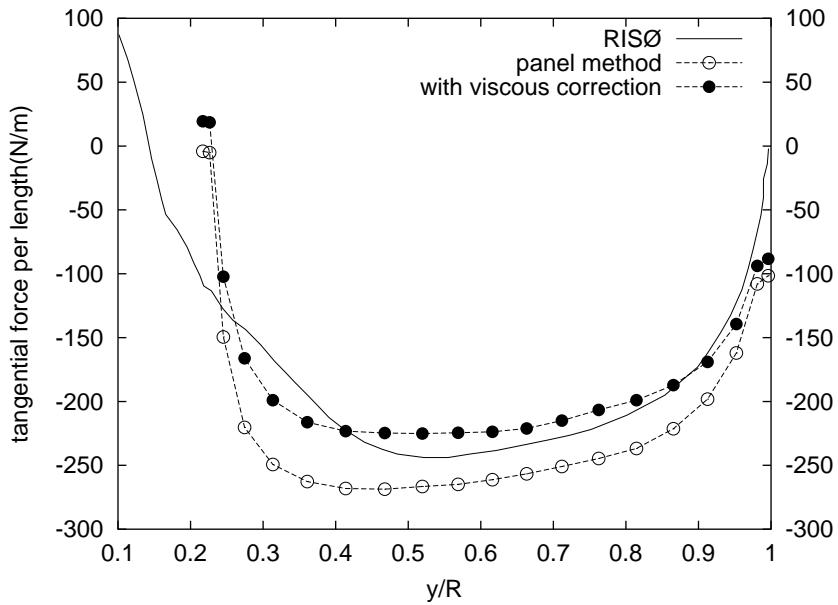


4.29.1: Tangential force distribution: 7 m/s

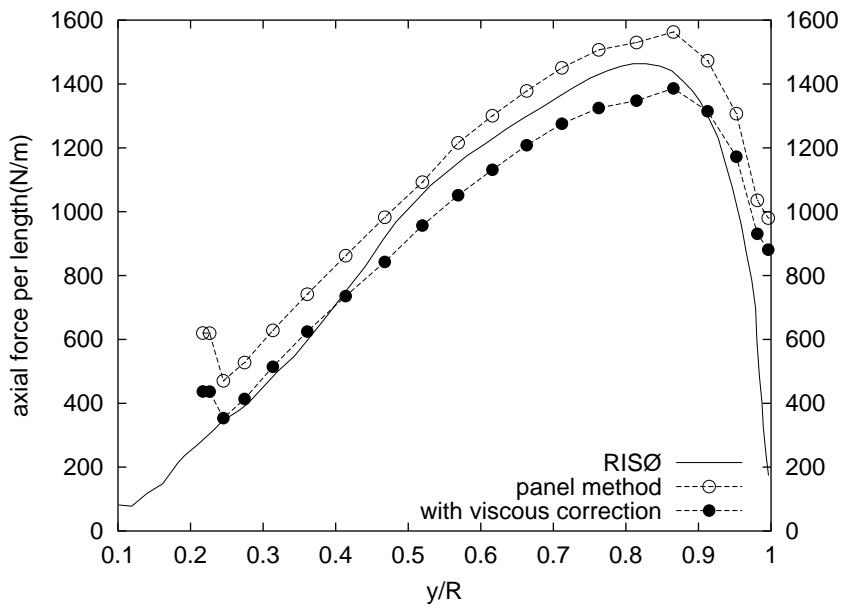


4.29.2: Axial force distribution: 7 m/s

Figure 4.29: Comparison with Risø LM19.1 predictions: spanwise axial and tangential force distributions

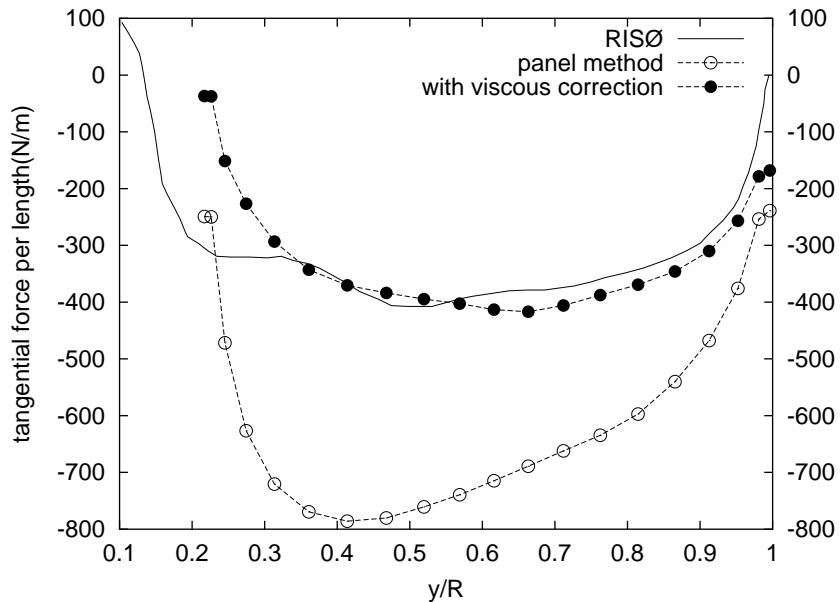


4.29.3: Tangential force distribution: 10 m/s

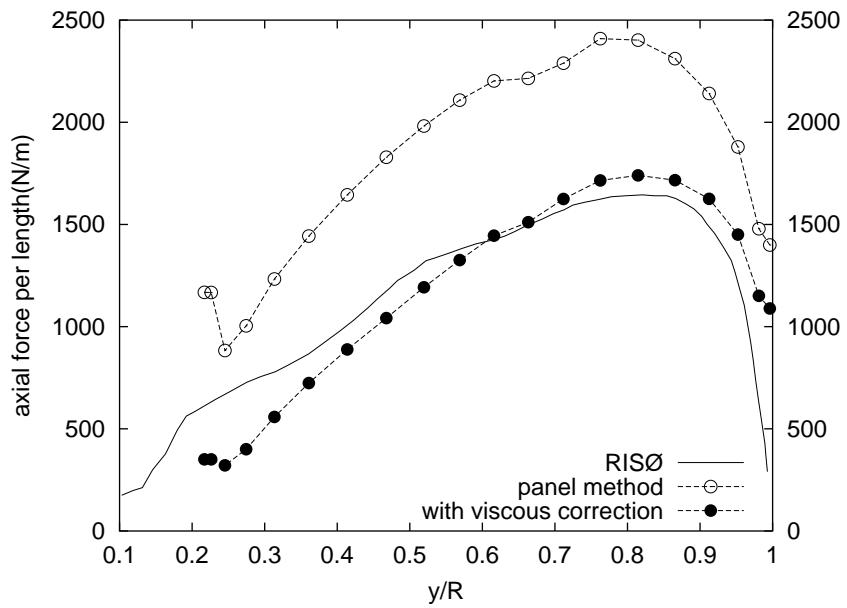


4.29.4: Axial force distribution: 10 m/s

Figure 4.29: LM19.1 axial and tangential force distributions (contd.)

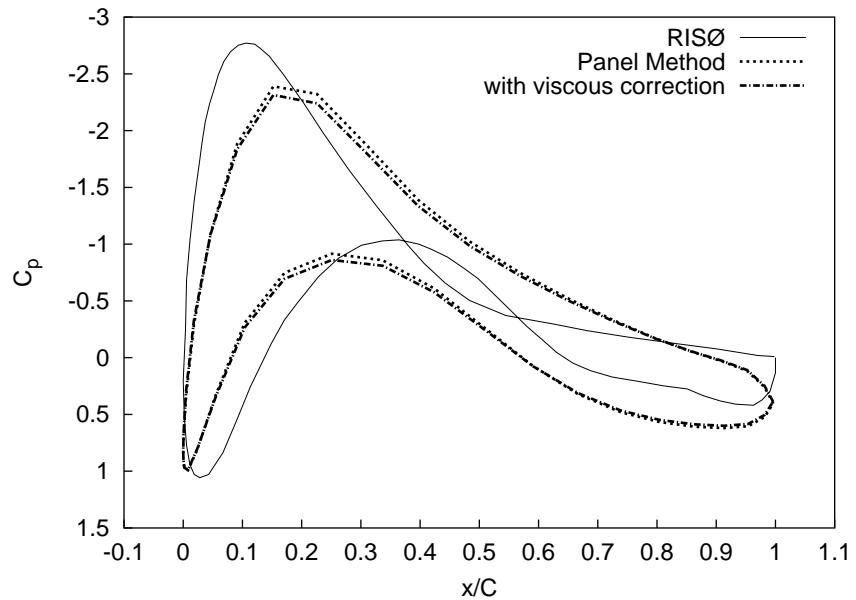
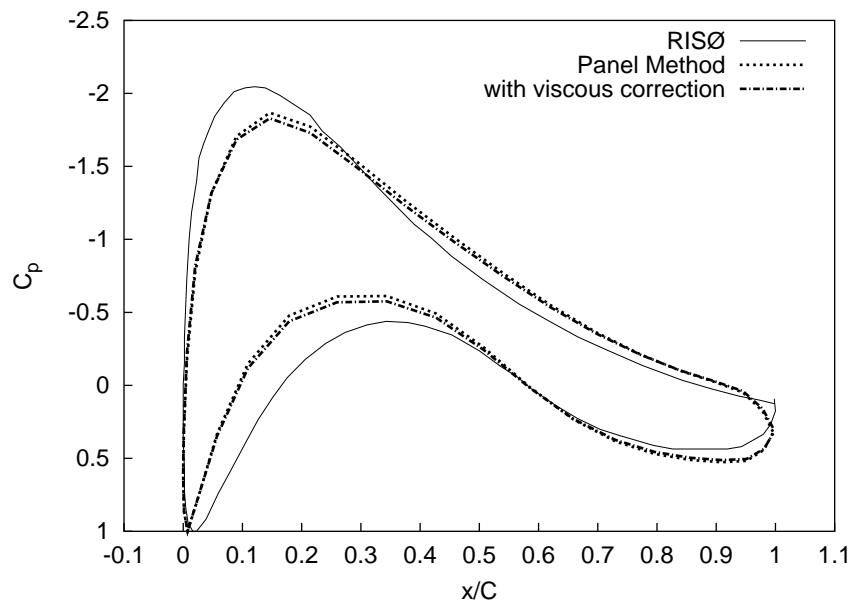


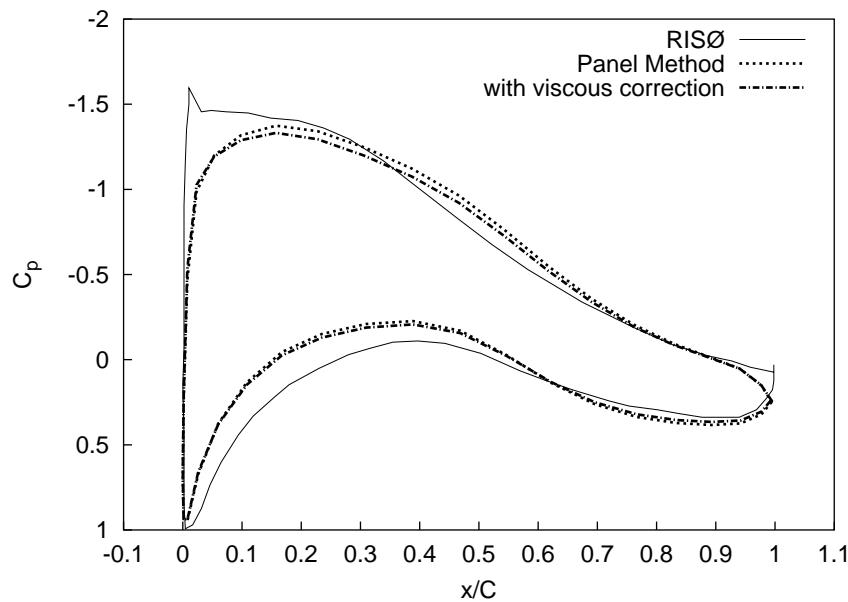
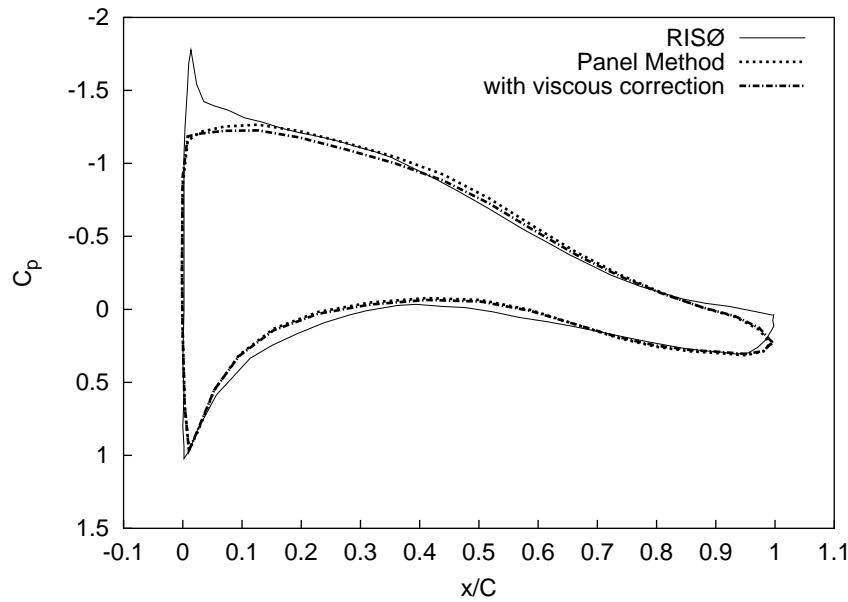
4.29.5: Tangential force distribution: 15 m/s



4.29.6: Axial force distribution: 15 m/s

Figure 4.29: LM19.1 axial and tangential force distributions (contd.)

4.30.1: Pressure distribution:  $U_0=7$  m/s;  $R=6$  m4.30.2: Pressure distribution:  $U_0=7$  m/s;  $R=9$  mFigure 4.30: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 7$  m/s

4.30.3: Pressure distribution:  $U_0=7$  m/s;  $R=12$  m4.30.4: Pressure distribution:  $U_0=7$  m/s;  $R=15$  mFigure 4.30: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 7$  m/s (contd.)

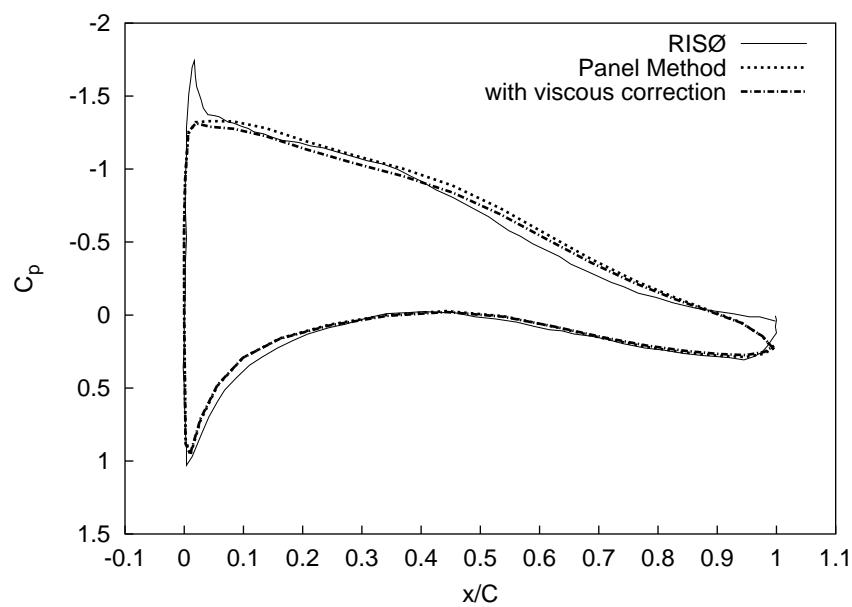
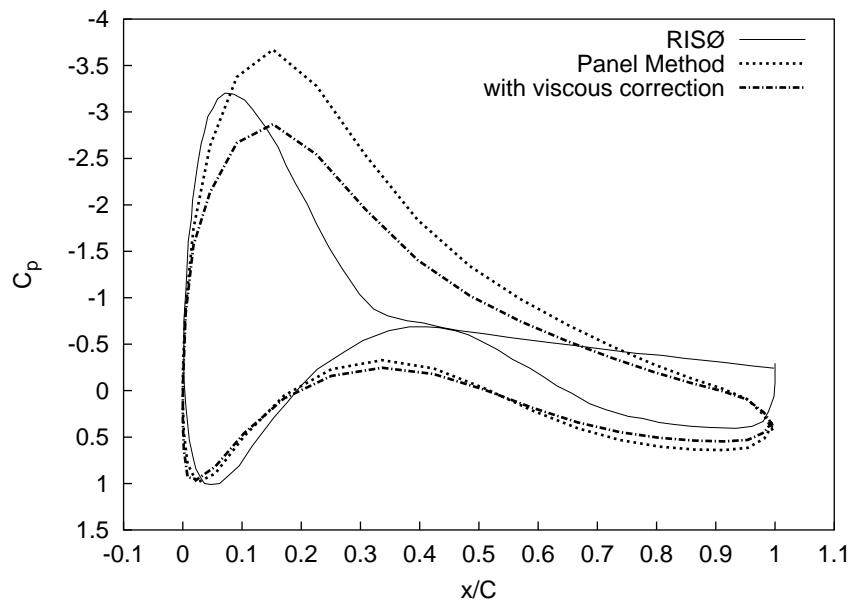
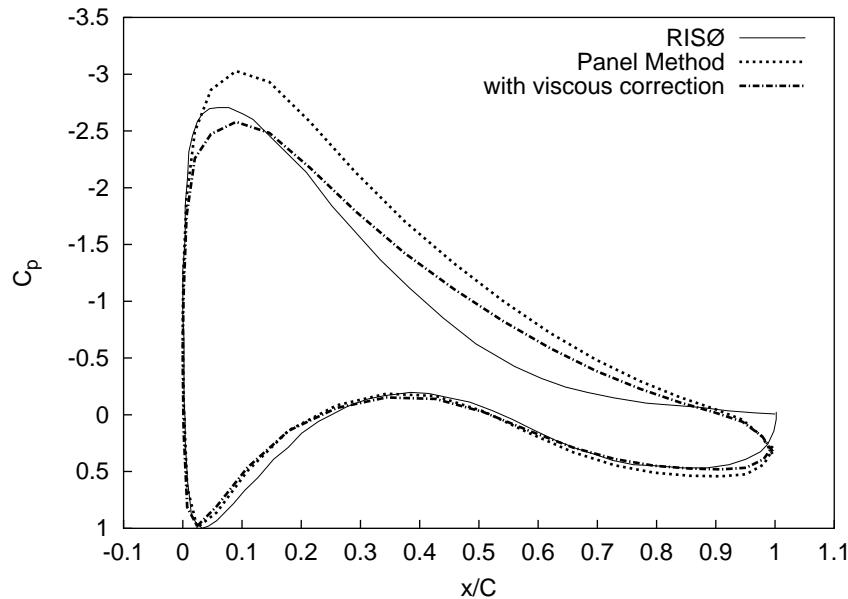
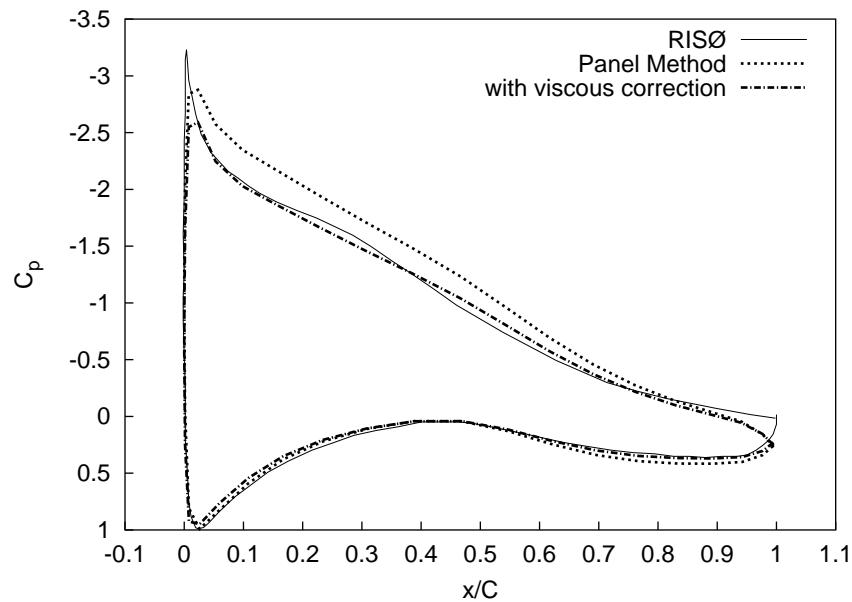
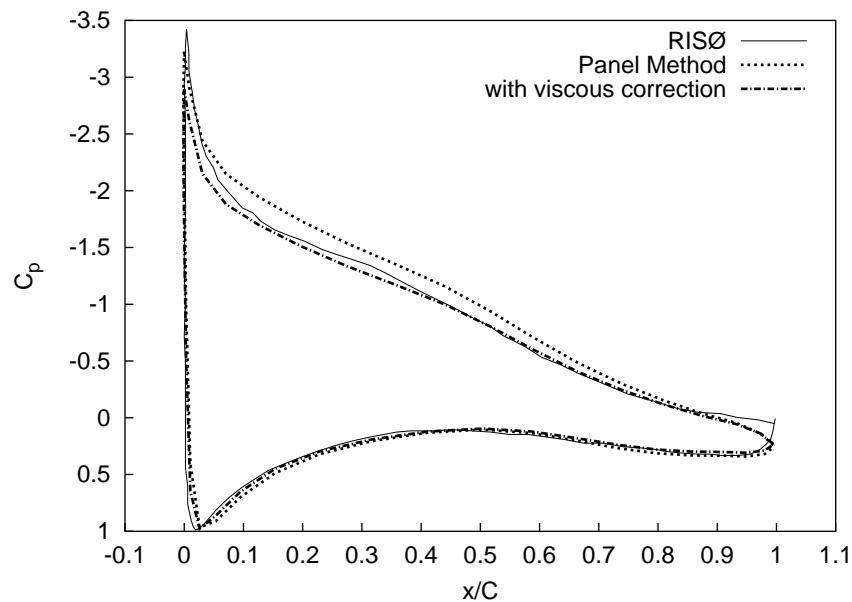
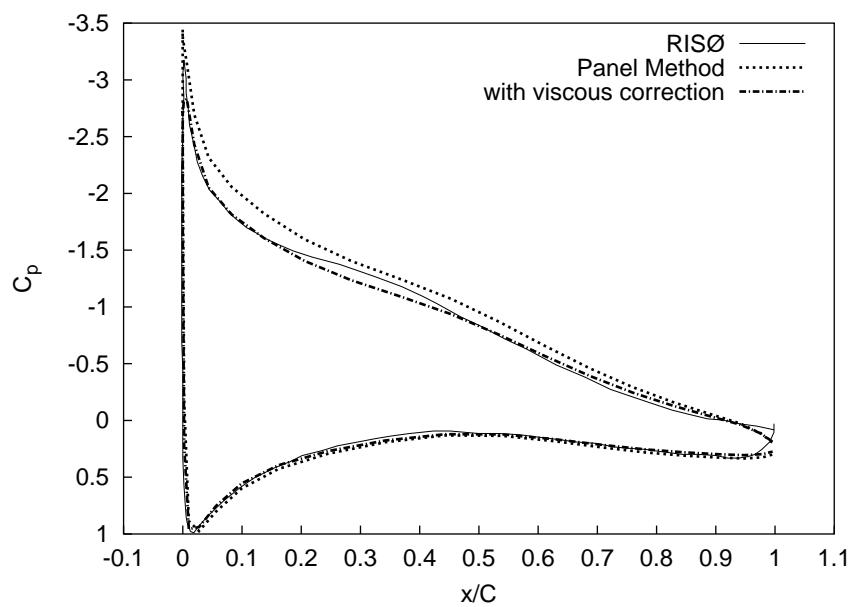
4.30.5: Pressure distribution:  $U_0=7$  m/s;  $R=18$  m

Figure 4.30: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 7$  m/s (contd.)

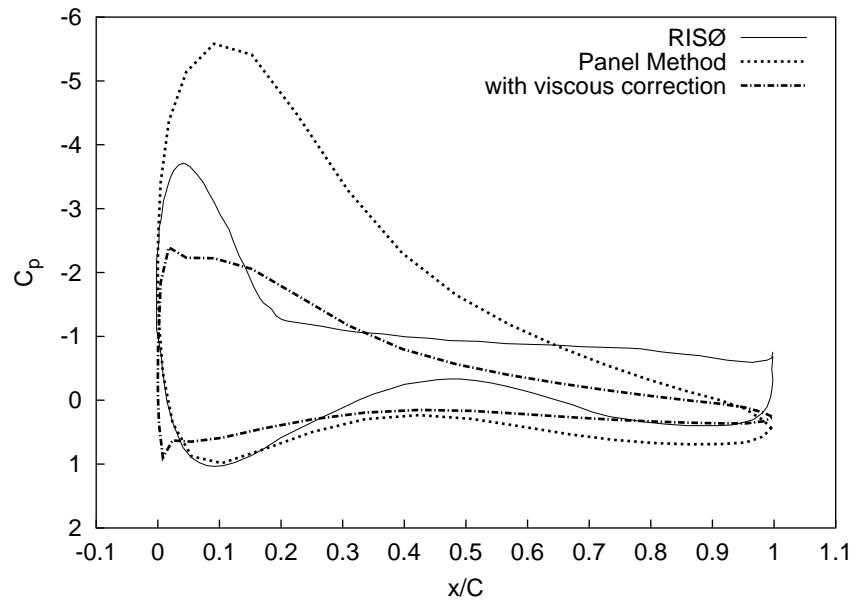
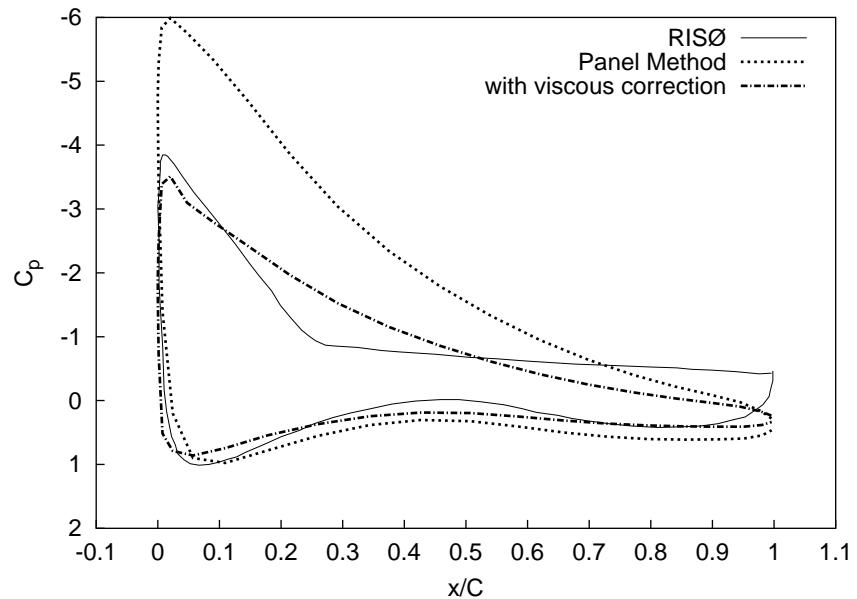
4.31.1: Pressure distribution:  $U_0=10 \text{ m/s}$ ;  $R=6 \text{ m}$ 4.31.2: Pressure distribution:  $U_0=10 \text{ m/s}$ ;  $R=9 \text{ m}$ Figure 4.31: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 10 \text{ m/s}$

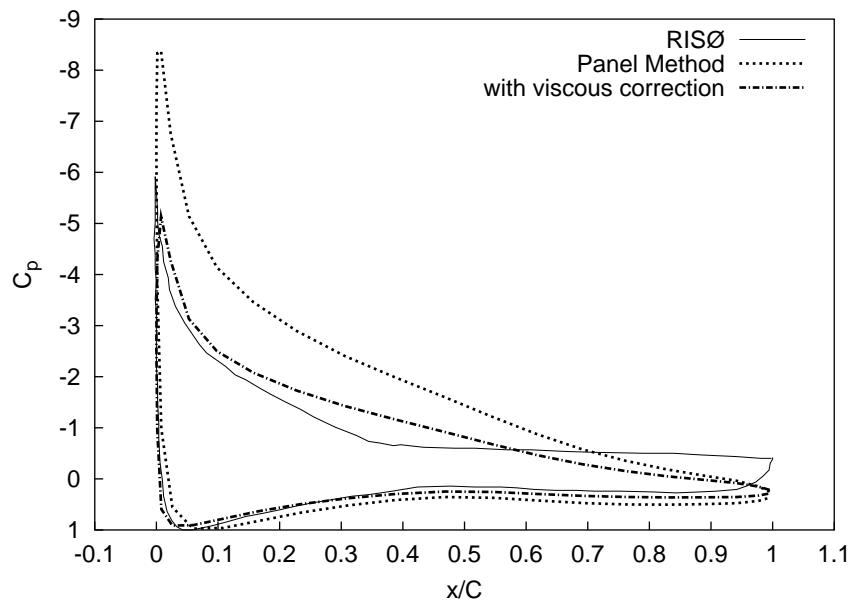
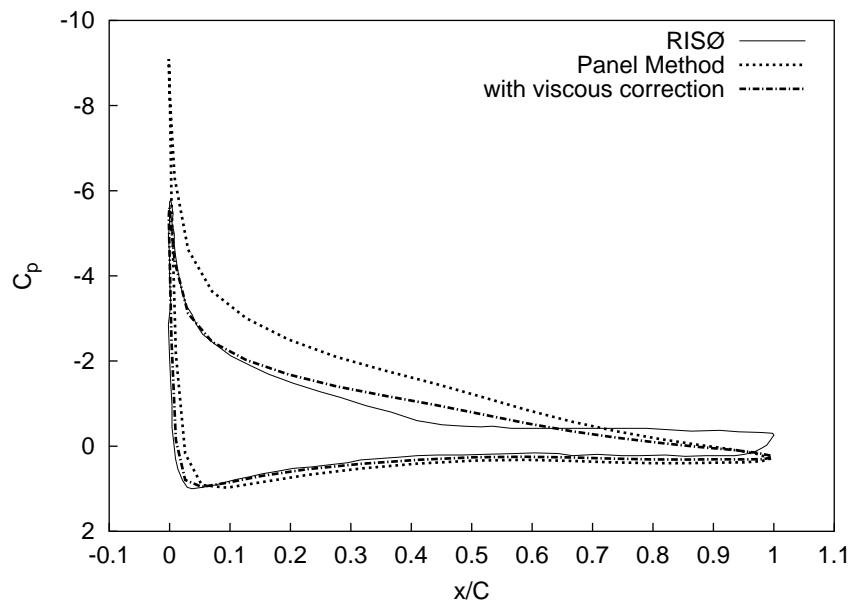
4.31.3: Pressure distribution:  $U_0=10$  m/s;  $R=12$  m4.31.4: Pressure distribution:  $U_0=10$  m/s;  $R=15$  mFigure 4.31: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 10$  m/s (contd.)



4.31.5: Pressure distribution:  $U_0=10$  m/s;  $R=18$  m

Figure 4.31: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 10$  m/s (contd.)

4.32.1: Pressure distribution:  $U_0=15$  m/s;  $R=6$  m4.32.2: Pressure distribution:  $U_0=15$  m/s;  $R=9$  mFigure 4.32: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 15$  m/s

4.32.3: Pressure distribution:  $U_0=15$  m/s;  $R=12$  m4.32.4: Pressure distribution:  $U_0=15$  m/s;  $R=15$  mFigure 4.32: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 15$  m/s (contd.)

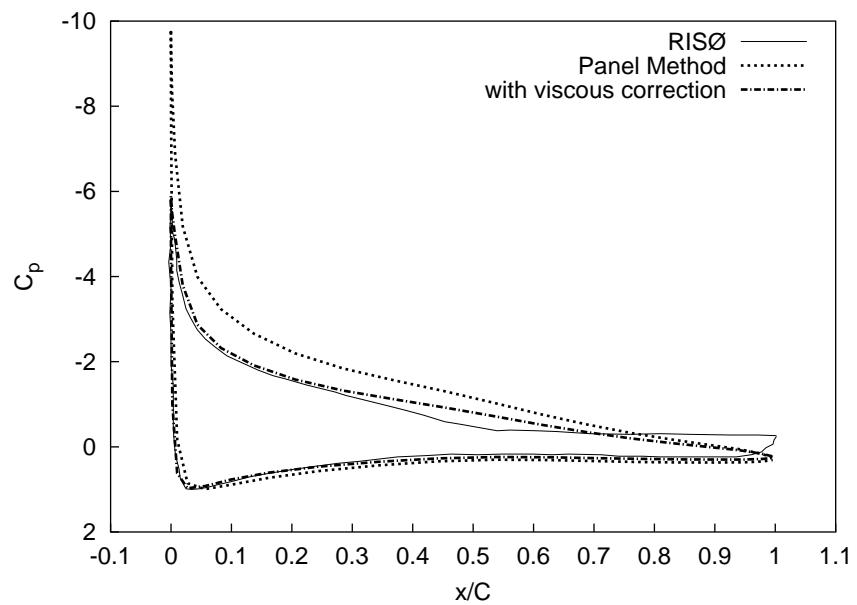
4.32.5: Pressure distribution:  $U_0=15$  m/s;  $R=18$  m

Figure 4.32: Panel method vs Risø LM19.1 pressure predictions at  $U_0 = 15$  m/s (contd.)

#### 4.10.3 Test case: small variable-speed wind turbine

A bi-cubic B-spline model was constructed of the blade used in the variable-speed 5 kW wind turbine. This blade was designed by the Wind Energy Group at the University of Newcastle to extract optimal power at a tip speed ratio of  $\lambda=10$ , with a nominal 5 kW being generated at this  $\lambda$  when  $U_0 = 10$  m/s. The panel method was applied to this model, with both the iterative wake model and iterative and iterative Kutta condition employed to predict the wind turbine's performance at a constant coaxial wind speed of  $U_0 = 10$  m/s over a range of tip speed ratios. A mesh of  $40 \times 20$  panels for each blade and a wake mesh comprised of 5 turns with 20 streamwise panels per turn were employed. The predicted power curve shown in figure 4.33 is compared

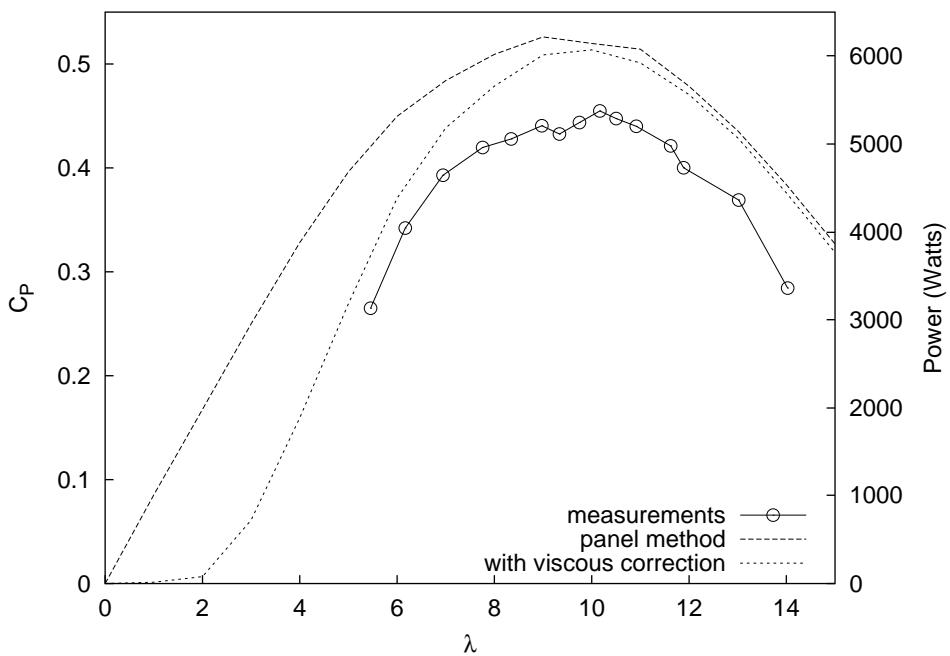


Figure 4.33: Computed power curve for the 5 kW rotor

with experimental data taken from Anderson et al. (1982). The small wind turbine presented in that paper had similar chord and twist distributions to the Newcastle 5 kW blade, with a NACA 4412 section used instead of the Newcastle blade's SD7062. Using the latter, the more modern blade should exhibit improved performance. The panel method predictions suggest that this is the case, with the predicted peak power coefficient agreeing well with field test data for the Newcastle blade. The predictions

also agree qualitatively with the older blade's experimental data to the extent that the peak power coefficient occurs at the designed optimal tip speed ratio, together with a similar curve shape, from low to high  $\lambda$ . The quantitative differences can be attributed largely to the different aerofoil section, as well as to the panel method's tendency to consistently slightly over-predict blade loads, a problem which decreases with increasing mesh density.

Figure 4.34 gives the dependence of  $U_1$  on  $\lambda$ , as predicted by the iterative wake model. The important feature to note in a  $U_1$  history is that the Lanchester-Betz limit (appendix B.1) gives  $U_1 = \frac{2}{3}$  at the point corresponding to optimal power extraction. The expectation is that this  $U_1$  figure would appear at  $\lambda=10$ , but, as shown in figure 4.34, this seems to occur consistently at  $\lambda \approx 8$ . This discrepancy is probably due to a number of assumptions made about the wake, most notably that it is finite: no far-wake approximation is included in the current method, with the wake extending for 5 turns downstream; constant diameter: no wake expansion is included in the model - a gross simplification that may be worthwhile remedying; and constructed on a wake pitch based on a single, rather arbitrary, maximal  $\Gamma$  measure along the span of the blade (see section 4.4.3). All of these simplifications undoubtedly contribute to the discrepancy. Nevertheless, the wake model provides worthwhile predictions of blade loads as it stands.

Figure 4.35 shows how the predicted axial force distributions on each blade change with increasing tip speed ratio. Figures 4.36 and 4.37 give similar histories of tangential force distributions and associated torque distributions, respectively. Figure 4.38 shows how the circulation distribution around each blade,  $\Gamma$ , changes with tip speed ratio. Note that  $\Gamma$  is nearly constant with radius near optimum power ( $\lambda=10$ ), showing that most of the bound vorticity goes directly into the tip vortices.

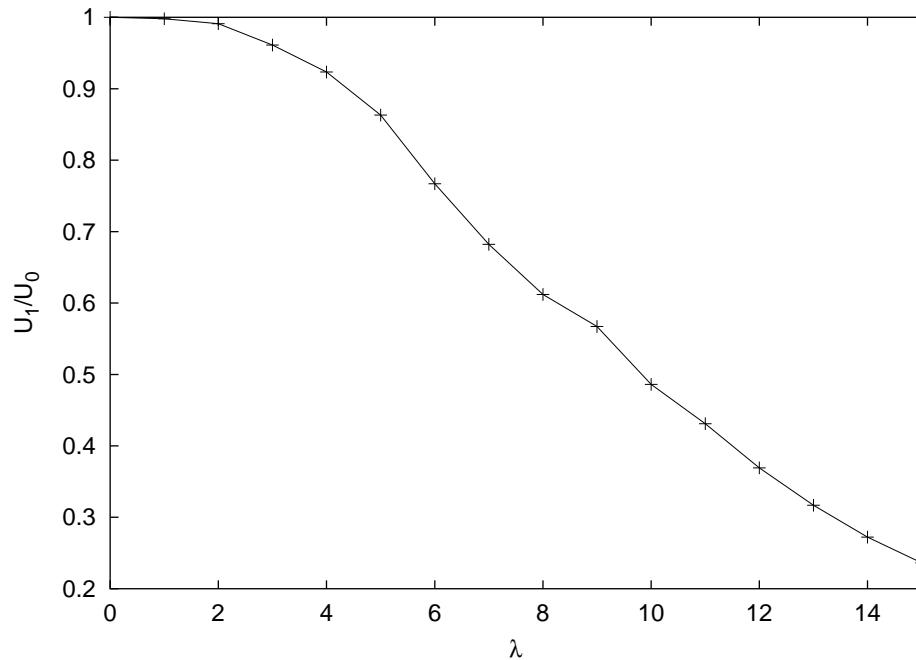


Figure 4.34: Computed  $U_1$  vs  $\lambda$  for the 5 kW rotor

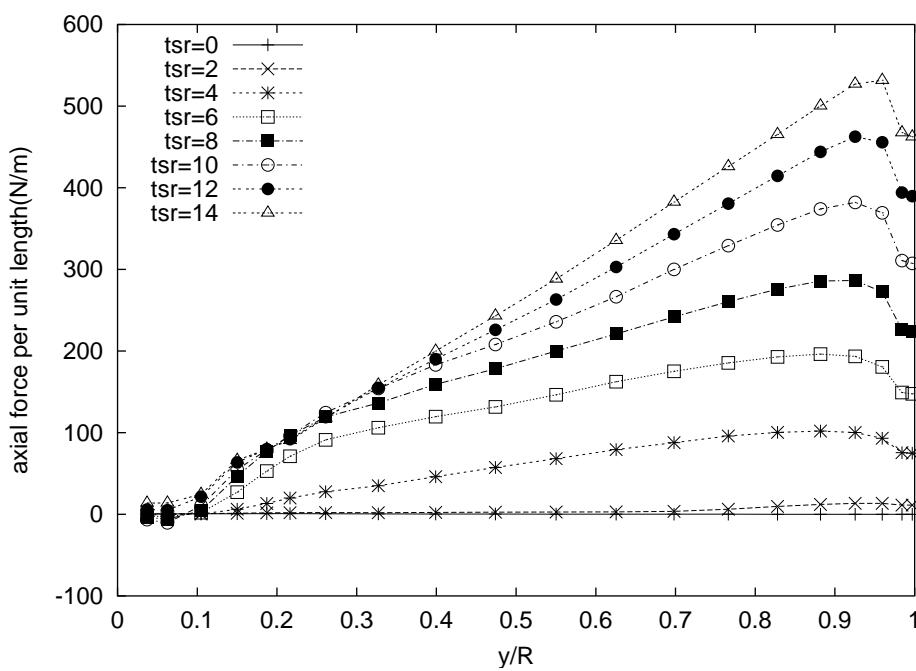
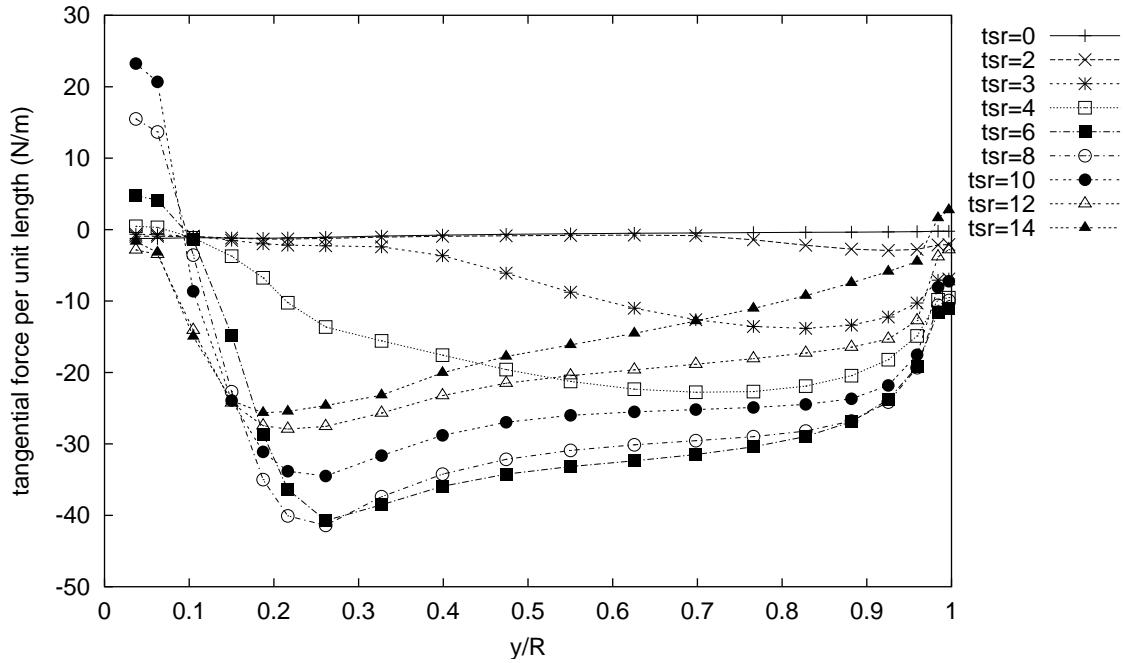
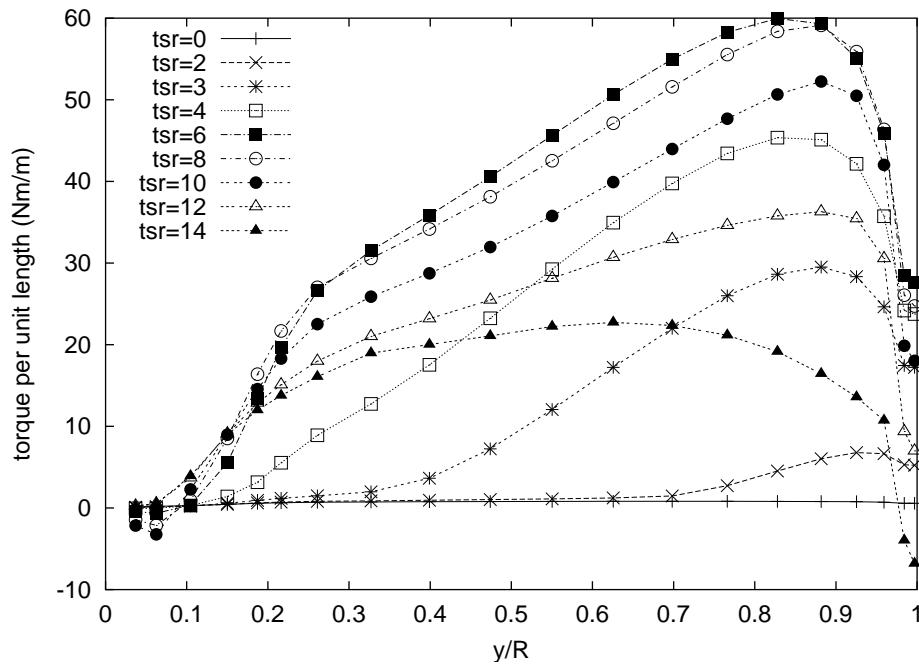


Figure 4.35: Axial force distributions for the 5 kW blade at  $U_0=10\text{m/s}$

Figure 4.36: Tangential force distributions for the 5 kW blade at  $U_0=10\text{m/s}$ Figure 4.37: Torque distributions for the 5 kW blade at  $U_0=10\text{m/s}$

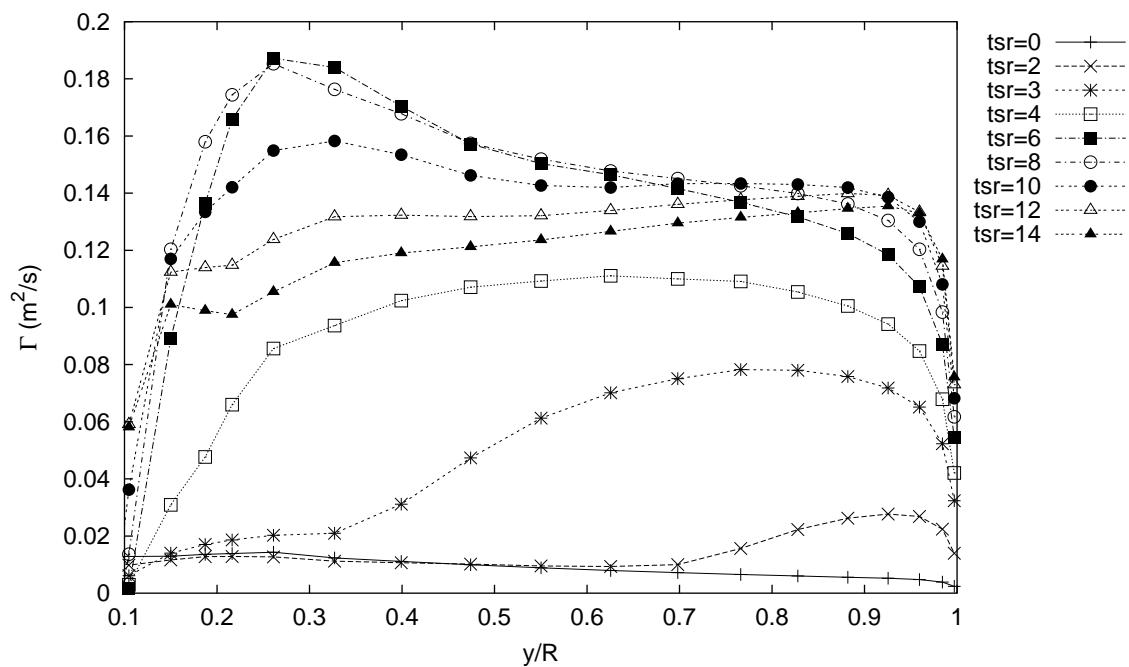
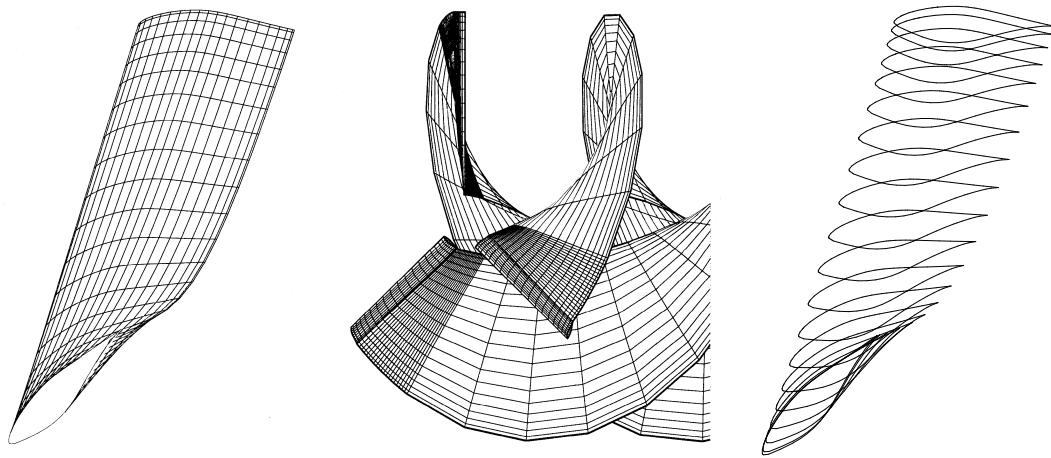


Figure 4.38: Circulation distributions for the 5 kW blade

#### 4.10.4 Test case: untapered, twisted NREL Phase III turbine blade

Simms et al. (1999) present three complete experimental campaigns covering the field testing of two related wind turbine blade geometries. The experiments offer an extraordinary amount of data on many different aspects of wind turbine operation, and form a component of the coordinated wind turbine aerodynamics study program presented in Schepers et al. (1997). A comprehensive database of results can be obtained from the web site included with that reference.



4.39.1: Discretised bi-cubic B-spline NREL Phase III blade

4.39.2: Model of three-bladed NREL Phase III rotor and wake

4.39.3: Blade cross-sections

Figure 4.39: NREL Phase III blade and rotor geometry

The untapered, twisted turbine blade geometry studied in the Phase III NREL campaign was selected for use as a validation geometry in the current study. The blade used an S809 aerofoil throughout, a chord length of 0.4572 metres, was 5.023 metres long and was twisted from root to tip, with a root twist of  $44.67^\circ$  and tip twist of  $0^\circ$ . The experimental wind turbine rotor had three blades which were pitched to  $3^\circ$ , and rotated at a constant 71.62 r.p.m.

Four experimental datasets exist for the un-yawed Phase III NREL rotor. Datasets were acquired at  $U_0 \approx 7$  m/s; 10 m/s; 13 m/s; 16 m/s. Each data acquisition experiment consisted of sixty seconds of turbine operation at its fixed rotational speed, with data gathered from one instrumented blade. Measurements were taken for a wide variety

of pressures, loads, angles and velocities at intervals of 0.0192 seconds, giving 3126 discrete measurement periods per experiment. The tabulated data for each of the four campaigns includes a preface of the mean, standard deviation, minimum and maximum value of each measurement over this period. This reduced data was used to validate the steady-state aerodynamic prediction performance of the current panel method.

Blade design parameters supplied by Simms et al. (1999) include a spanwise twist distribution, together with unit-chord surface co-ordinates for the S809 aerofoil. A bi-cubic B-spline representation of the Phase III NREL blade was created by first fitting a cubic B-spline curve to this S809 aerofoil co-ordinate data via the DE curve fitting method described in chapter 3. This unit-chord aerofoil was then used as a template for the entire blade, with one cubic B-spline aerofoil created and rotated for each supplied twist angle, then scaled to the required 0.4572 metre chord length. The bi-cubic B-spline blade representation was then discretised into a quadrilateral surface panel mesh with N\_PANELS=40 and M\_PANELS=20. A wake model constructed from quadrilateral panels was added to the trailing edge of each of the three blades in the rotor model, with W\_PANELS=20 and TURNS=5. Each meshed blade in the computational rotor model was pitched to 3°. Figures 4.39.1 and 4.39.2 depict this blade and rotor panelling scheme, with Figure 4.39.3 showing chordwise cross-sections through the meshed blade.

The panel method was applied to the meshed rotor/wake model at a range of wind speeds, from  $U_0 = 5$  m/s to  $U_0 = 18$  m/s, with the rotor speed held constant at 71.62 r.p.m. Power predictions at each wind speed were recorded for comparison with the NREL experimental power results. Panel method predictions of blade surface pressures and spanwise thrust and tangential forces were also stored at three wind speeds ( $U_0=7$  m/s; 10 m/s; 13 m/s) for comparison with results from the NREL instrumented-blade experiments.

Figure 4.40 shows the power output of the experimental rotor together with panel method power predictions over a range of wind speeds. The experimental power figures refer to electrical power, a quantity which is less than the “aerodynamic” power

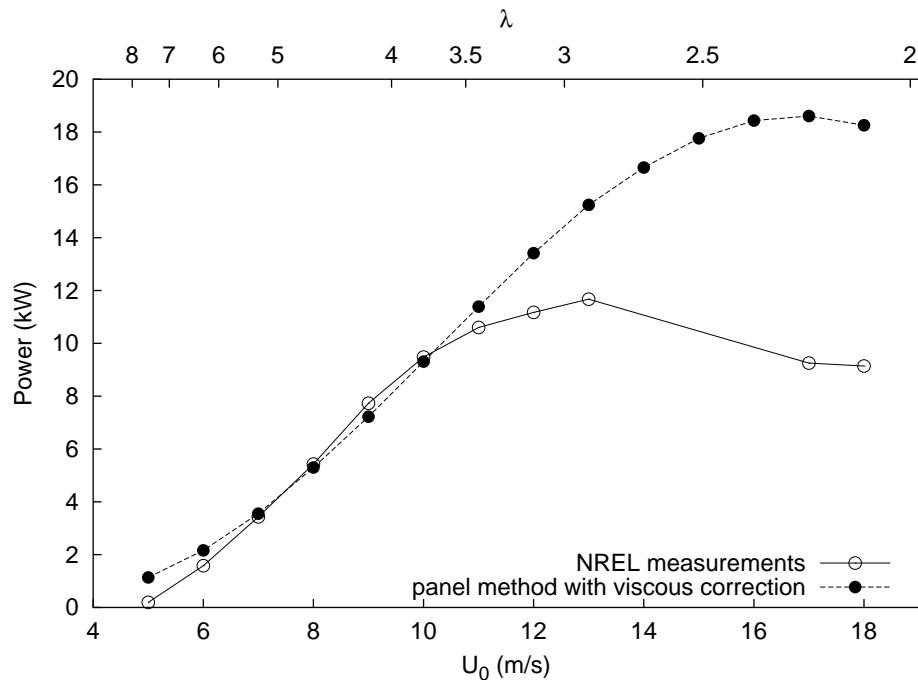


Figure 4.40: Comparison of predicted and experimental Power curves for untapered, twisted NREL Phase III rotor

supplied by the rotating blades due to conversion inefficiencies in the power-train:

$$\eta_{generator} = 0.78$$

$$\eta_{gearbox} = 0.97$$

$$\eta_{windage} = 0.98$$

$$P_{electrical} = \eta_{generator} \eta_{gearbox} \eta_{windage} P_{aerodynamic}$$

$$P_{electrical} \approx 0.7415 P_{aerodynamic} \quad (4.75)$$

These conversion efficiencies are provided by Simms et al. (1999) for the gearbox/generator used in the NREL Phase II-IV campaigns. The resulting 74.15% conversion efficiency was applied to the aerodynamic torques predicted by the panel method, giving the results shown in Figure 4.40.

The defining feature of the power predictions seems to be: very good “high”- $\lambda$  performance which rapidly degrades as wind speed increases - that is, as the  $\lambda$  experienced by a constant rotational speed wind turbine decreases. The divergence from experimental power output at the lowest recorded wind speeds (highest  $\lambda$ ) can largely

be explained by the accompanying relatively very small aerodynamic torques, values which are increasingly overwhelmed by the resistance to motion of the generator. The “maximum brake torque” of the experimental generator was reported in Simms et al. (1999) as 115.24 Nm. This static torque was not modelled in the current implementation.

Whilst good “high”- $\lambda$  prediction performance is an expected prerequisite for a prediction code intended to evaluate small variable-speed wind turbine blade designs, it should be recognised that reasonable power calculations can be seen to occur even at  $\lambda \approx 3.5$  for the current geometry. This is a very low tip speed ratio, at least in the realm of the small wind turbines at the focus of the current project. It is to be expected that the large local angles of attack and relatively low Reynolds numbers at such operating conditions would give rise to the viscous effects which reduce lift - a deliberate *design feature* of this turbine known as “stall regulation”. The example further reinforces the lower tip speed ratio at which we can expect panel method to perform reasonably, with a  $\lambda$  of 3 being assumed in the following chapter to be the best possible “low”- $\lambda$  condition at which competing variable-speed small wind turbine blade designs can be reasonably evaluated by the current panel method for “starting” performance.

The instrumented NREL Phase III blade was equipped with 22 pressure taps on the surface of the blade at 30%, 47% and 80% span positions, and 23 pressure taps at 63% and 95% span positions. For comparison with the steady-state predictions of the current panel method, the average dynamic pressures (and their standard deviation) measured via the pressure taps over the course of each 60 second data acquisition experiment were divided by the mean stagnation pressure to give the pressure coefficient at each chordwise surface location

$$C_p = \frac{P}{P_{stagnation}} \quad (4.76)$$

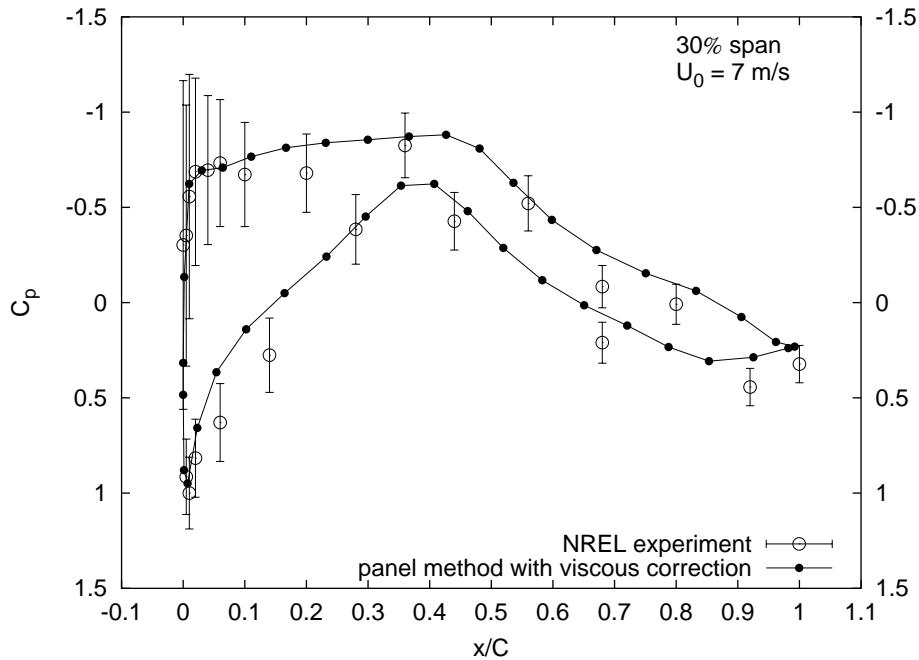
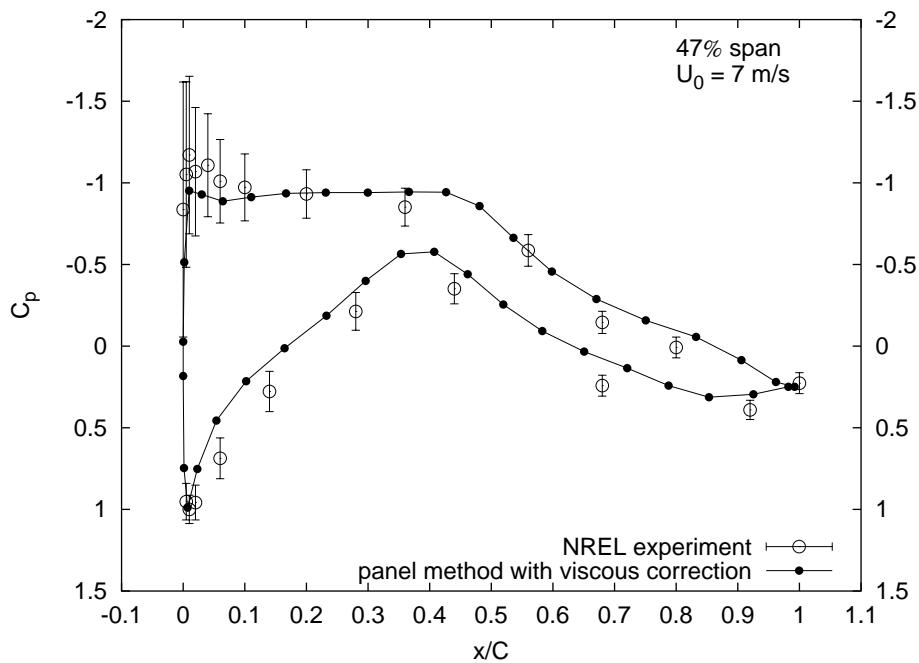
where the stagnation pressure was taken as the maximum positive mean dynamic pressure at each spanwise position (which is always, by definition, the pressure-side spike on the  $C_p$  vs chord plot).

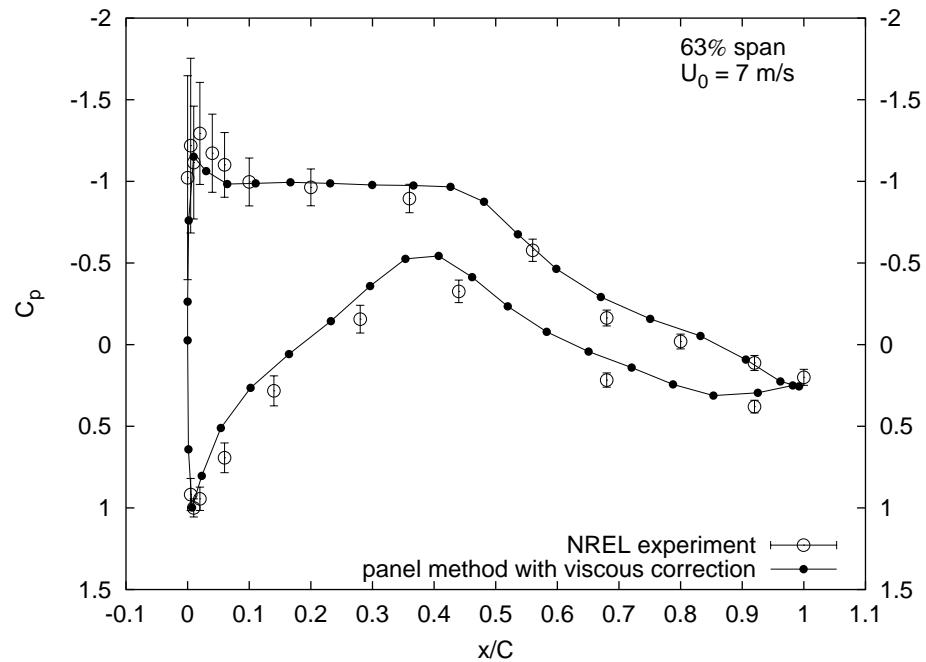
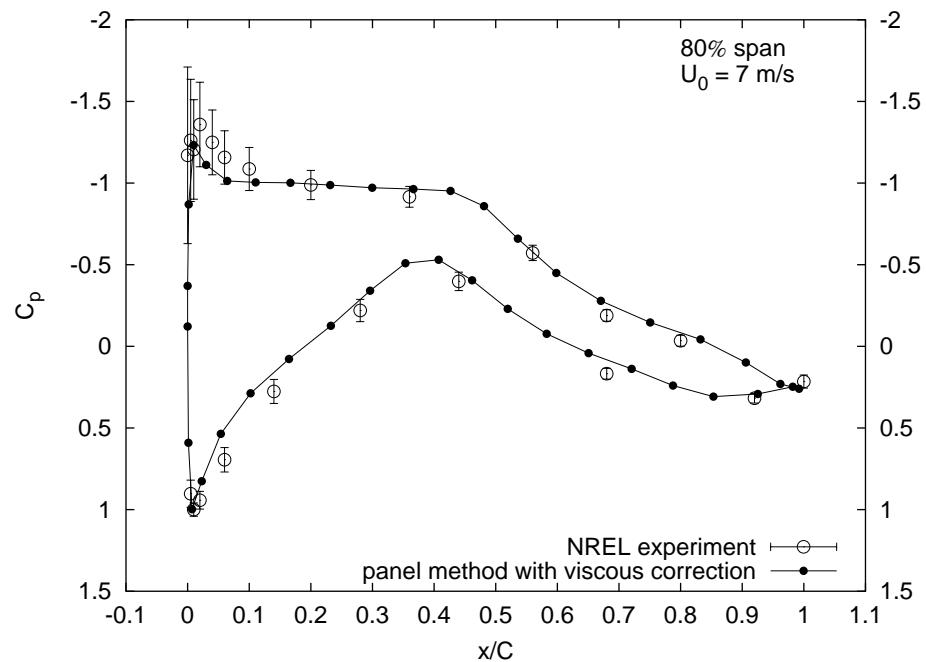
Figures 4.41, 4.42 and 4.43 show comparisons between mean experimental NREL Phase III rotor surface pressure coefficient measurements and panel method  $C_p$  calculations at  $U_0 = 7$  m/s, 10 m/s and 13 m/s, respectively.

The range spanned by upper and lower limits of the error bars on the experimental

measurements indicates one standard deviation from the mean, measured over each sixty-second unsteady data acquisition experiment. The larger variation in experimental measurements closer to the rotor axis probably indicates the greater influence of viscous and unsteady aerodynamic effects in that region.

The comparisons again seem to show that panel method provides a reasonably good way to predict blade surface pressures, with better prediction accuracy observed at out-board span positions than those calculation closer to the hub. As expected, better accuracy is also seen at higher tip speed ratios. In particular, a rather severe leading-edge suction-side pressure spike over-prediction can be seen to occur at the 13 m/s wind speed and, to a lesser extent, at the 10 m/s wind speed.

4.41.1: Pressure coefficient distribution:  $U_0=7 \text{ m/s}; R=30\%$ 4.41.2: Pressure coefficient distribution:  $U_0=7 \text{ m/s}; R=47\%$ Figure 4.41: Panel method  $C_p$  predictions vs NREL Phase III experimental results

4.41.3: Pressure coefficient distribution:  $U_0=7 \text{ m/s}$ ;  $R=63\%$ 4.41.4: Pressure coefficient distribution:  $U_0=7 \text{ m/s}$ ;  $R=80\%$ Figure 4.41: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 7 \text{ m/s}$  (contd.)

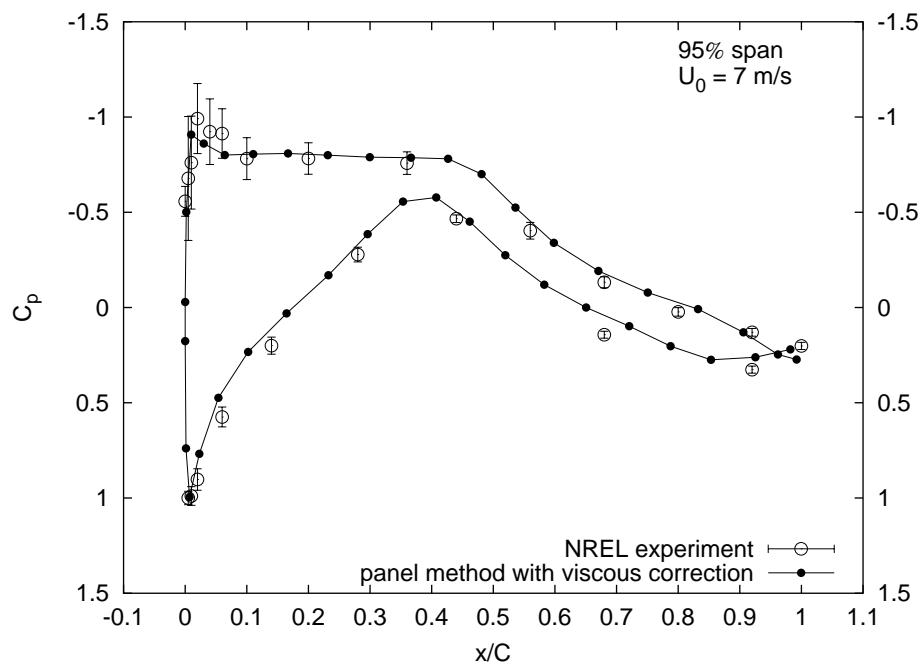
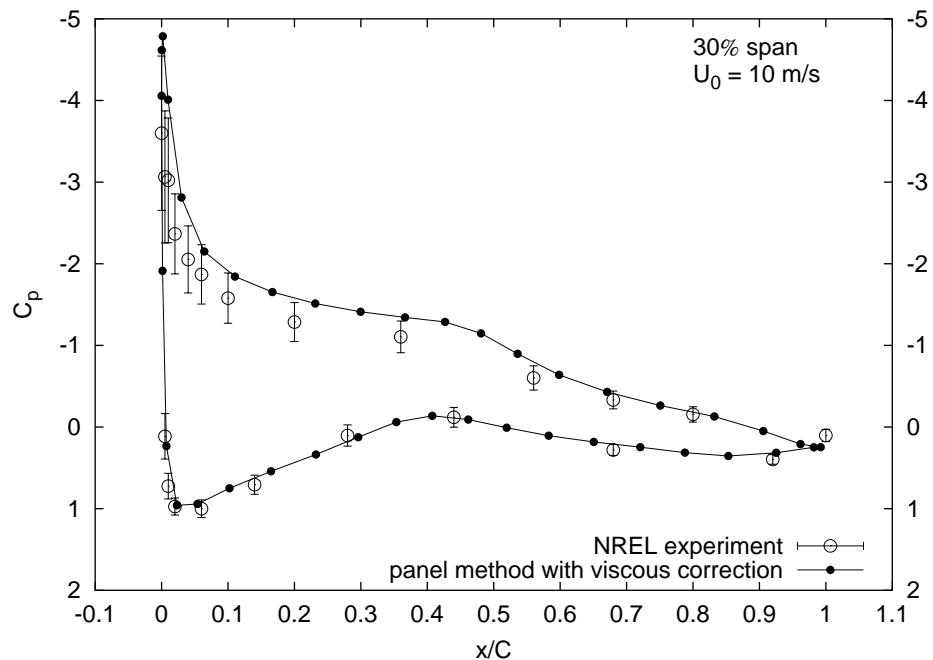
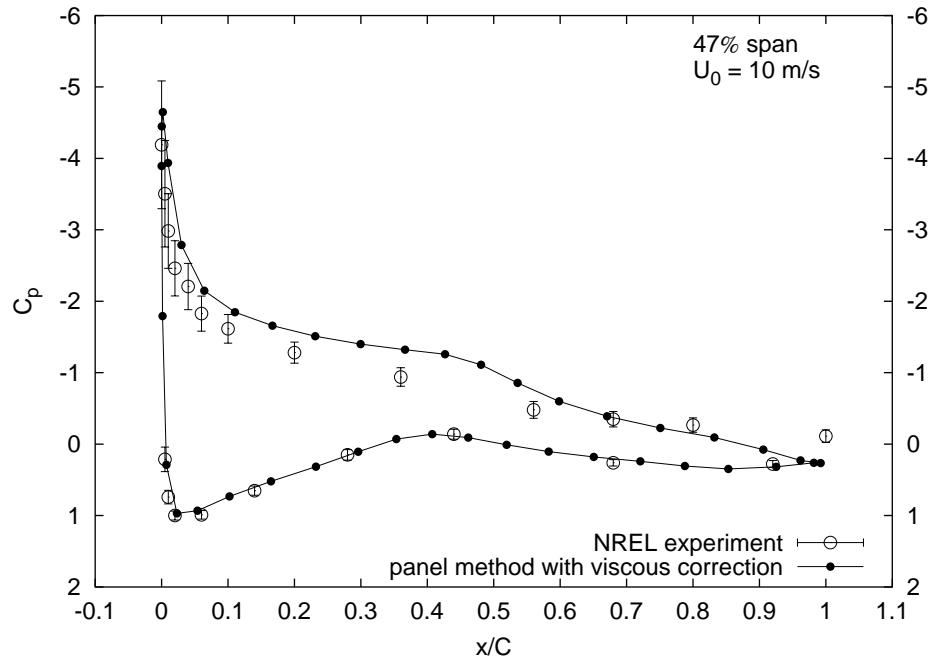
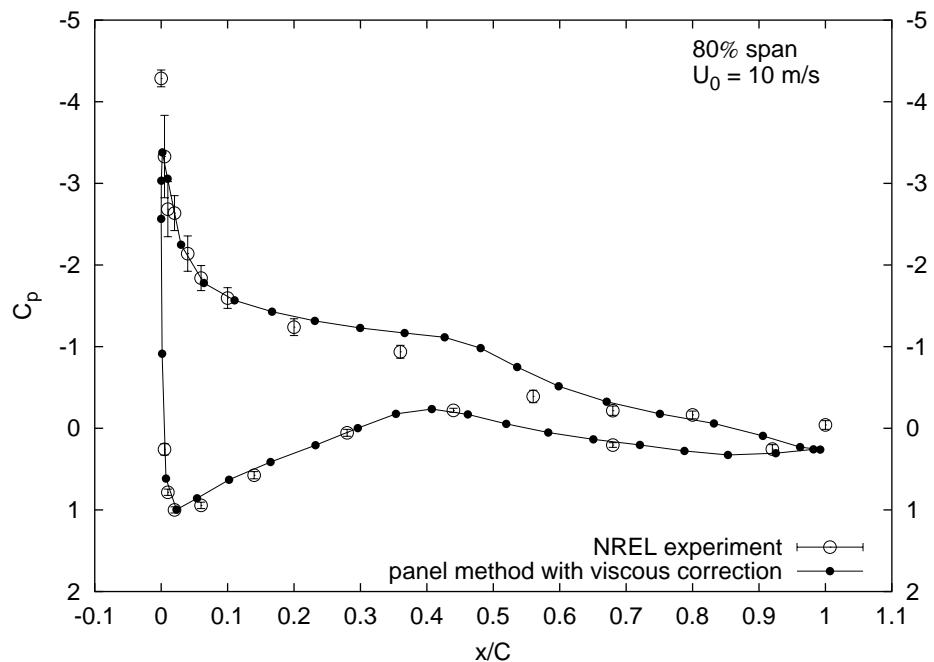
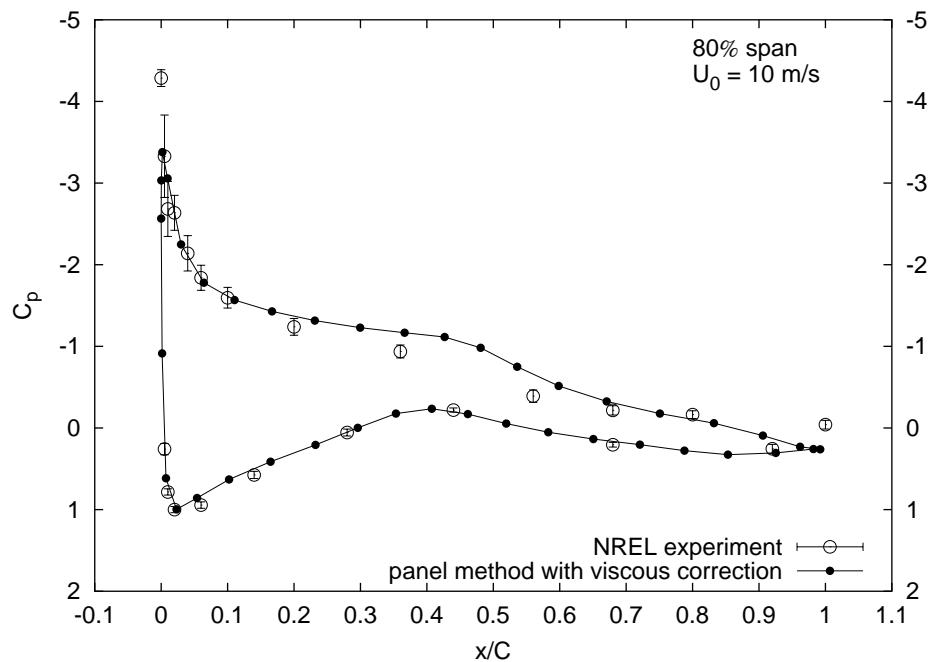
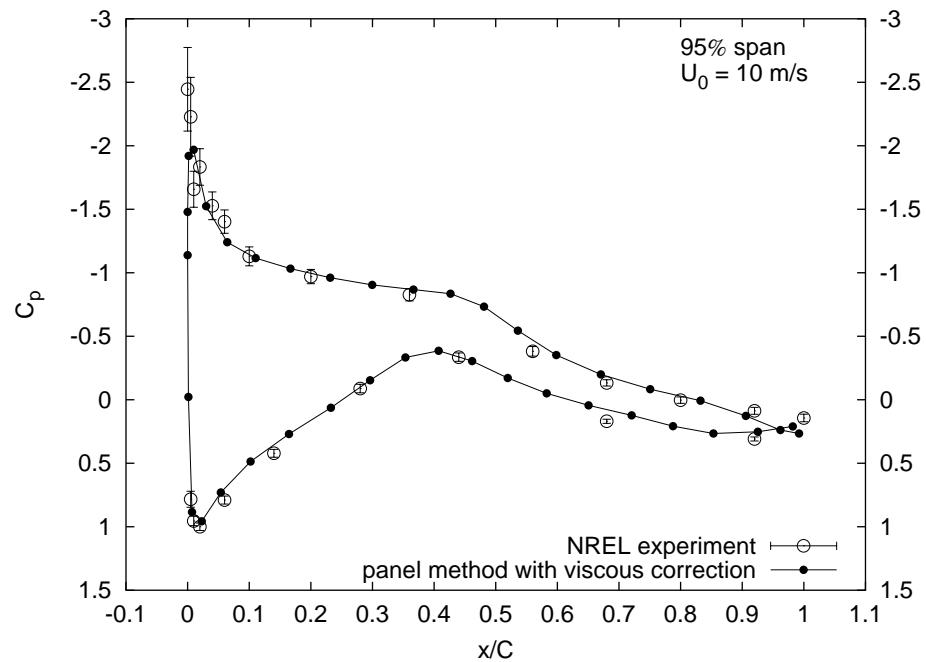
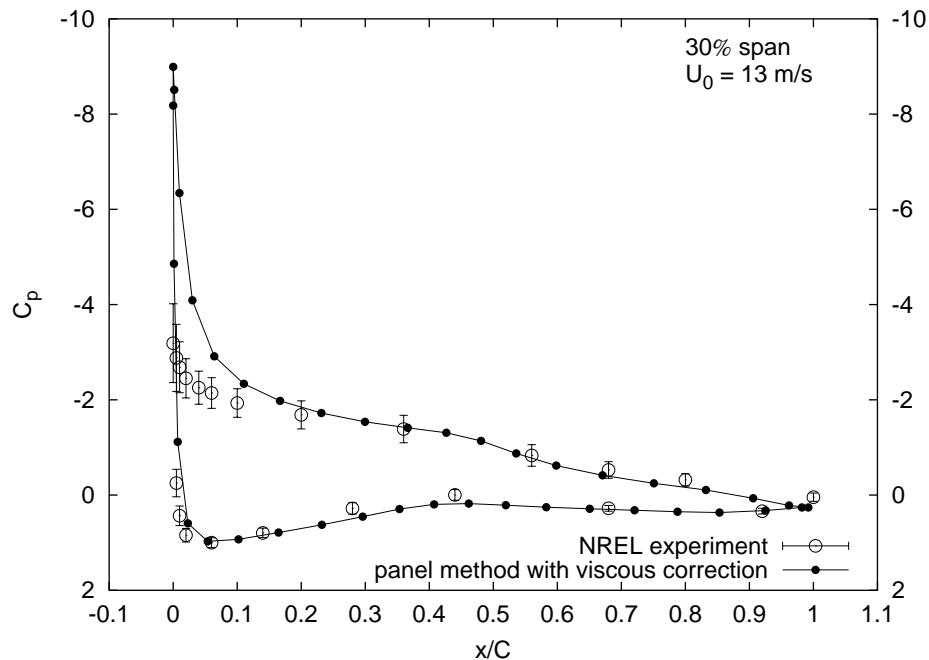
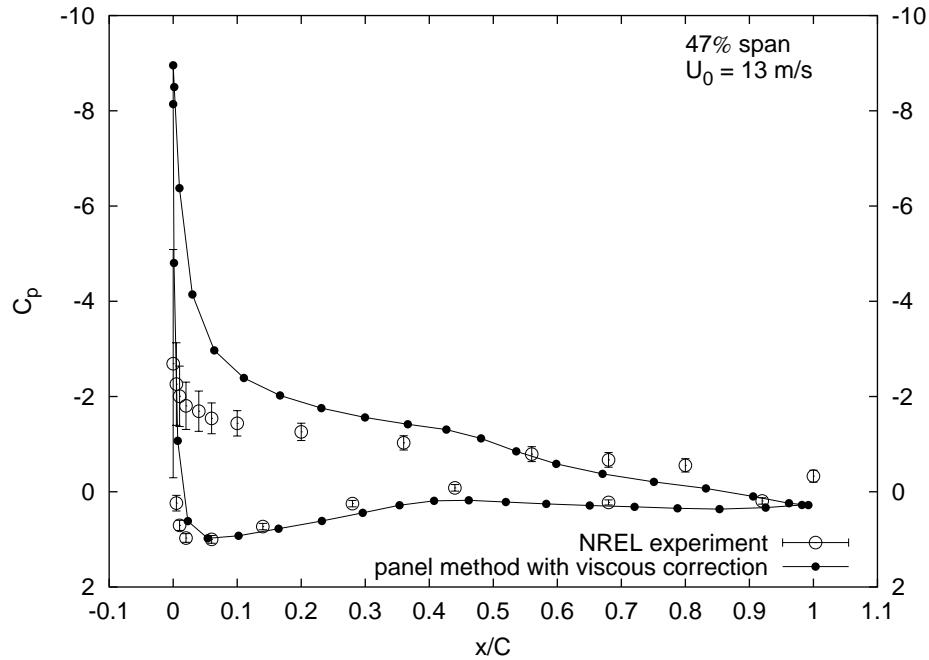
4.41.5: Pressure coefficient distribution:  $U_0=7 \text{ m/s}$ ;  $R=95\%$ 

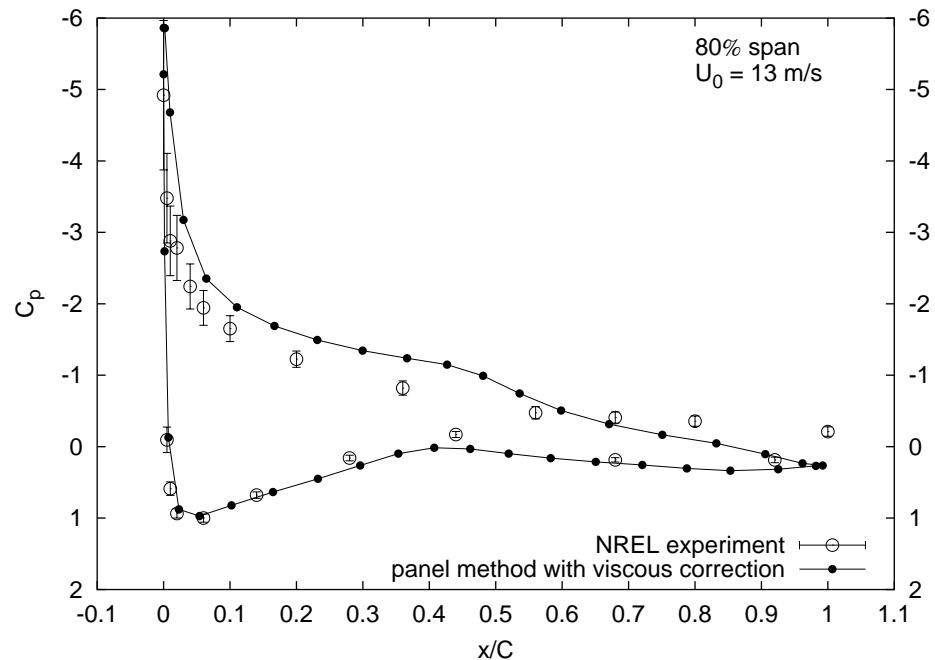
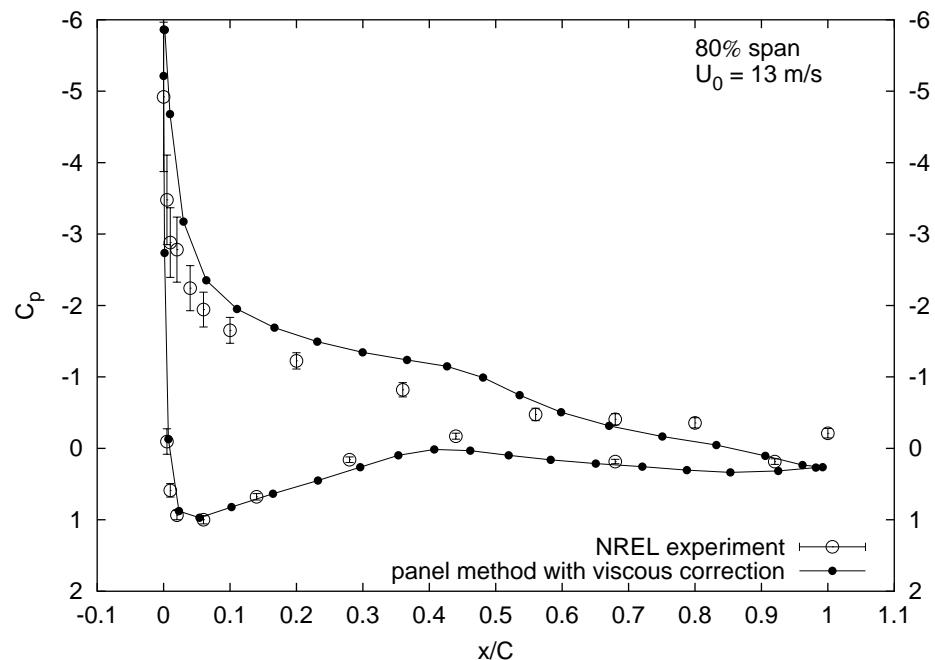
Figure 4.41: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 7 \text{ m/s}$  (contd.)

4.42.1: Pressure coefficient distribution:  $U_0=10 \text{ m/s}$ ;  $R=30\%$ 4.42.2: Pressure coefficient distribution:  $U_0=10 \text{ m/s}$ ;  $R=47\%$ Figure 4.42: Panel method  $C_p$  predictions vs NREL Phase III experimental results:  $U_0 = 10 \text{ m/s}$

4.42.3: Pressure coefficient distribution:  $U_0=10 \text{ m/s}$ ;  $R=63\%$ 4.42.4: Pressure coefficient distribution:  $U_0=10 \text{ m/s}$ ;  $R=80\%$ Figure 4.42: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 10 \text{ m/s}$  (contd.)

4.42.5: Pressure coefficient distribution:  $U_0=10 \text{ m/s}$ ;  $R=95\%$ Figure 4.42: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 10 \text{ m/s}$  (contd.)

4.43.1: Pressure coefficient distribution:  $U_0=13 \text{ m/s}$ ;  $R=30\%$ 4.43.2: Pressure coefficient distribution:  $U_0=13 \text{ m/s}$ ;  $R=47\%$ Figure 4.43: Panel method  $C_p$  predictions vs NREL Phase III experimental results:  $U_0 = 13 \text{ m/s}$

4.43.3: Pressure coefficient distribution:  $U_0=13 \text{ m/s}$ ;  $R=63\%$ 4.43.4: Pressure coefficient distribution:  $U_0=13 \text{ m/s}$ ;  $R=80\%$ Figure 4.43: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 13 \text{ m/s}$  (contd.)

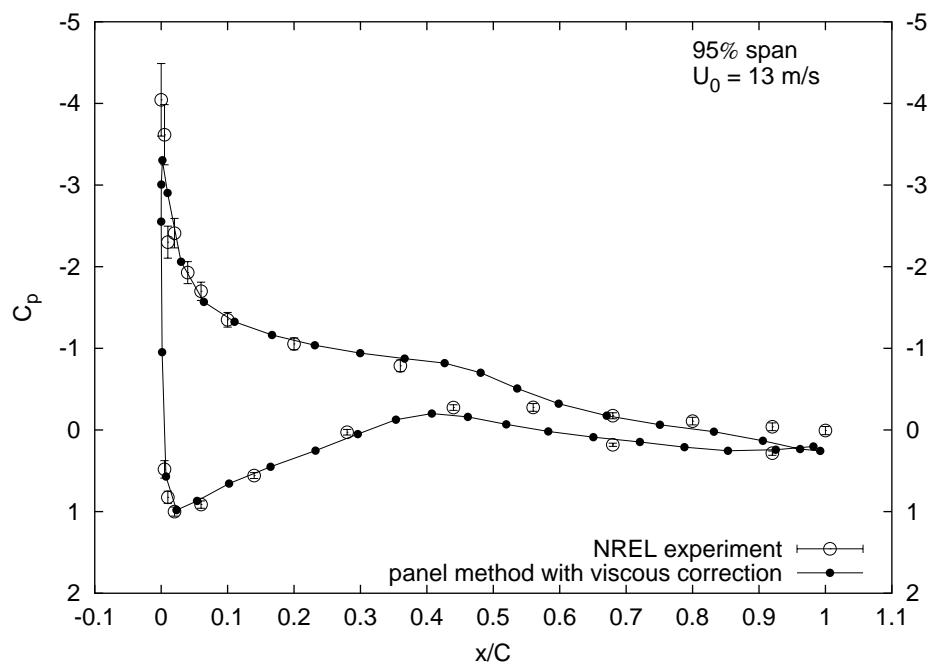
4.43.5: Pressure coefficient distribution:  $U_0=13 \text{ m/s}$ ;  $R=95\%$ 

Figure 4.43: Panel method  $C_p$  predictions vs NREL Phase III experimental results at  $U_0 = 13 \text{ m/s}$  (contd.)

Simms et al. (1999) calculated blade load coefficients at the five discrete spanwise positions by integrating the measured pressure coefficient distribution, with one normal and one tangential force component calculated for each tap:

$$C_N = \sum_{i=1}^{\# \text{ of taps}} \left( \frac{C_{p_i} + C_{p_{i+1}}}{2} \right) (x_{i+1} - x_i) \quad (4.77)$$

$$C_T = \sum_{i=1}^{\# \text{ of taps}} \left( \frac{C_{p_i} + C_{p_{i+1}}}{2} \right) (y_{i+1} - y_i) \quad (4.78)$$

where  $x_i$  is the normalised distance along the chord line from leading edge to the  $i^{th}$  pressure tap, and  $y_i$  is the normalised distance of the orthogonal projection from the  $i^{th}$  pressure tap to the chord line.

The experimental data-set records these force coefficients in the above “normal” and “tangential” forms as well as in two other forms: a “torque” and “thrust” form; and a “lift” and “(pressure) drag” form. These were resolved from the normal and tangential force coefficients as follows:

$$\begin{aligned} C_{\text{torque}} &= C_N \sin(\phi + \beta) + C_T \cos(\phi + \beta) \\ C_{\text{thrust}} &= C_N \cos(\phi + \beta) - C_T \sin(\phi + \beta) \end{aligned} \quad (4.79)$$

$$\begin{aligned} C_L &= C_N \cos(\alpha) + C_T \sin(\alpha) \\ C_D &= -(C_N \sin(\alpha) - C_T \cos(\alpha)) \end{aligned} \quad (4.80)$$

where  $\phi$  was the local twist angle,  $\beta$  the blade pitch angle and  $\alpha$  the local angle of attack.

It should be noted that  $C_{\text{torque}}$  refers to the *force* acting normal to the blade span in the plane of rotation - the force that gives rise to the blade torque, not the blade torque itself.  $C_{\text{thrust}}$  acts in the direction of the rotor axis (and in the unyawed case, the wind direction). To compare these experimental force coefficients with panel method predictions, the coefficient of the resultant force acting normal to each chordwise panel strip “blade element” was first defined:

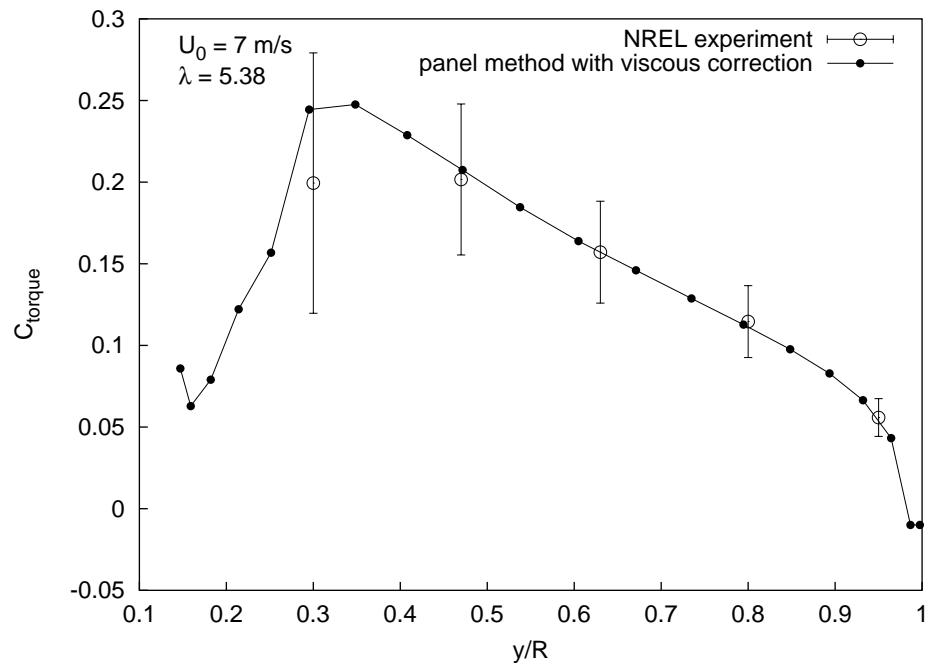
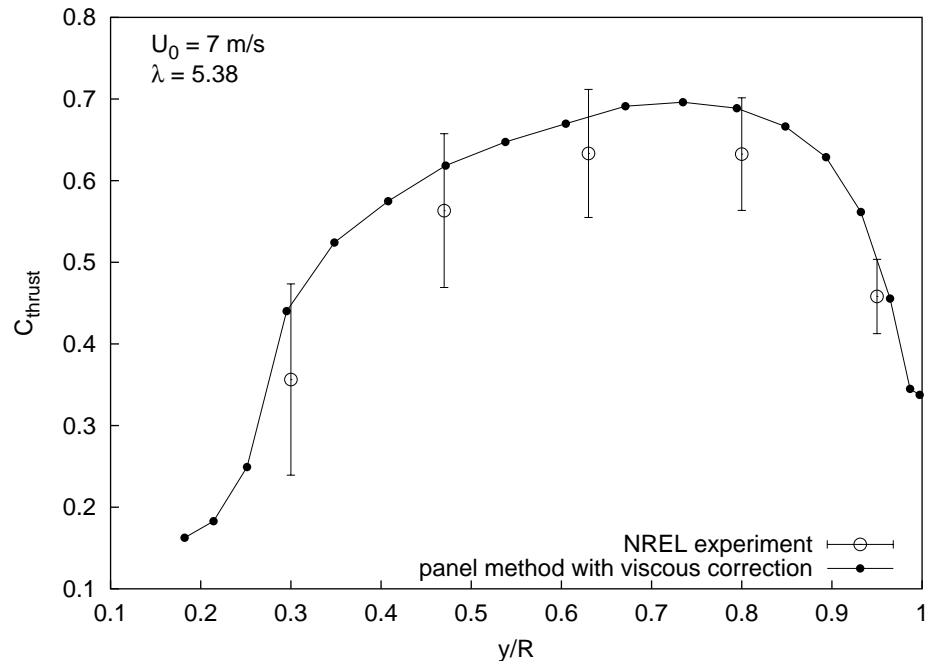
$$\vec{F} = \sum_{i=1}^{N\_PANELS} \vec{F}_i \quad (4.81)$$

$$C_{\vec{F}} = \frac{\vec{F}}{\frac{1}{2} \rho U_T^2 chord \times length} \quad (4.82)$$

where  $chord$  is the chord length at the centre of the current blade element, and  $length$  is the spanwise length of the current blade element.  $C_{torque}$  is then the component of  $\vec{C}_F$  along the global Z axis (in the global co-ordinate system scheme used by the current panel method), and  $C_{thrust}$  is the component along the global X axis.

Figures 4.44.1 and 4.44.2 compare the coefficients of the mean experimental blade loads experienced by the NREL Phase III turbine blade at  $U_0 = 7$  m/s with panel method predictions. Similar comparisons at  $U_0 = 10$  m/s and 13 m/s appear in Figures 4.45.1 to 4.46.2. As with the pressure coefficient comparisons, the extent shown by the error bars indicates one standard deviation from the mean of the data measured over the sixty-second experimental data acquisition period.

Load predictions can be seen to agree well at the highest tip speed ratio (at  $U_0 = 7$  m/s), particularly the “torque” load calculations. This accuracy degrades with decreasing tip speed ratio. It can be seen that the large over-prediction in the leading-edge pressure spike observed along the blade at  $U_0 = 13$  m/s translates into a corresponding over-prediction in blade loads, a discrepancy which is more pronounced towards the blade root than towards the tip. Again, this is to be expected, and is due to viscous effects at the high local angles of attack created by very low tip speed ratios, conditions which the panel method (and simple viscous correction) cannot hope to model with any degree of accuracy.

4.44.1:  $C_{torque}$  distribution :  $U_0 = 7$  m/s4.44.2:  $C_{thrust}$  distribution :  $U_0 = 7$  m/sFigure 4.44: Panel method  $C_{torque}$  and  $C_{thrust}$  predictions vs NREL Phase III experimental results

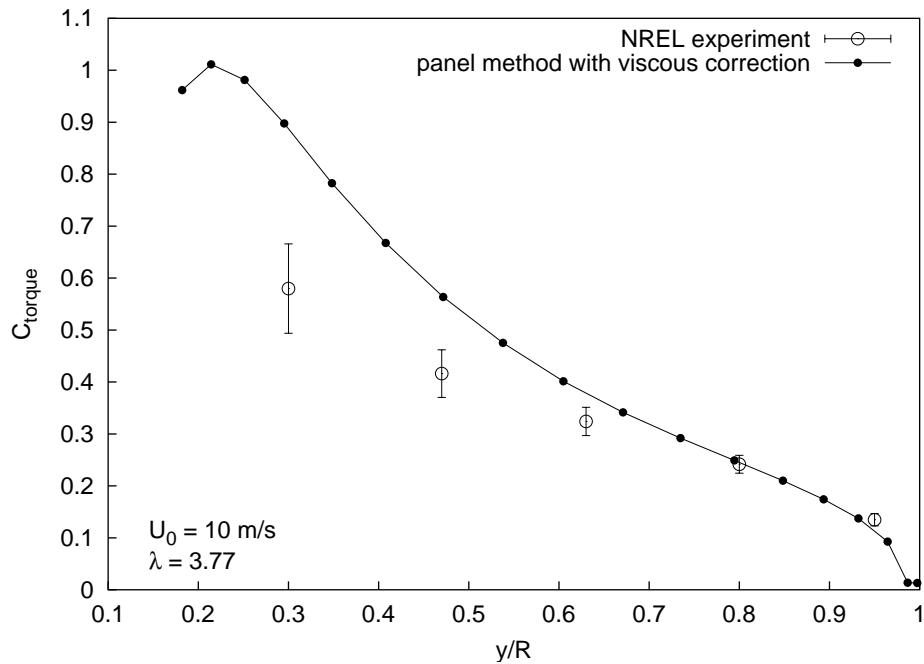
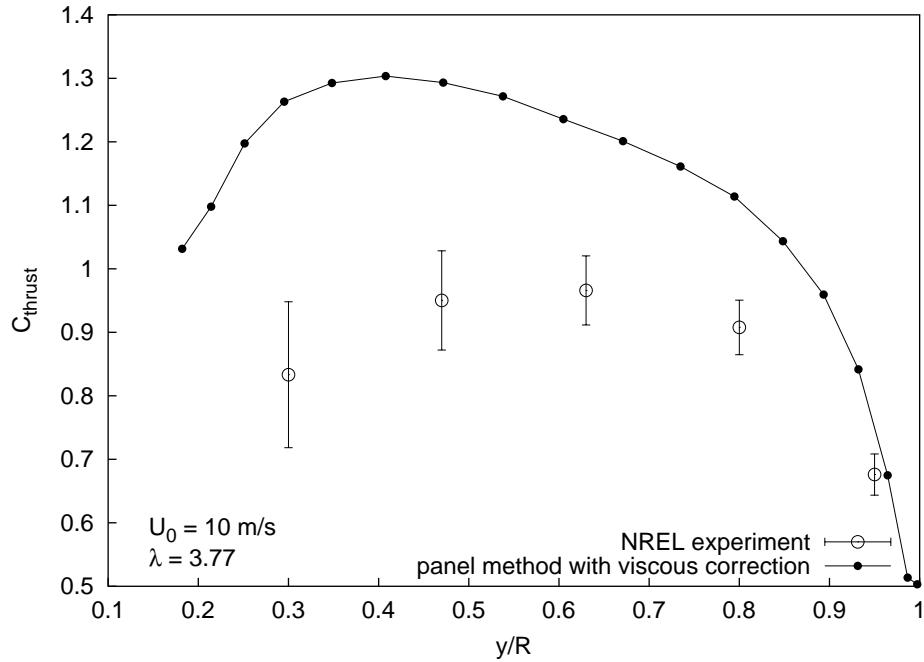
4.45.1:  $C_{torque}$  distribution :  $U_0 = 10 \text{ m/s}$ 4.45.2:  $C_{thrust}$  distribution :  $U_0 = 10 \text{ m/s}$ 

Figure 4.45: Comparison with experimental NREL Phase III rotor: spanwise  $C_{torque}$  and  $C_{thrust}$  distributions at  $U_0 = 10 \text{ m/s}$

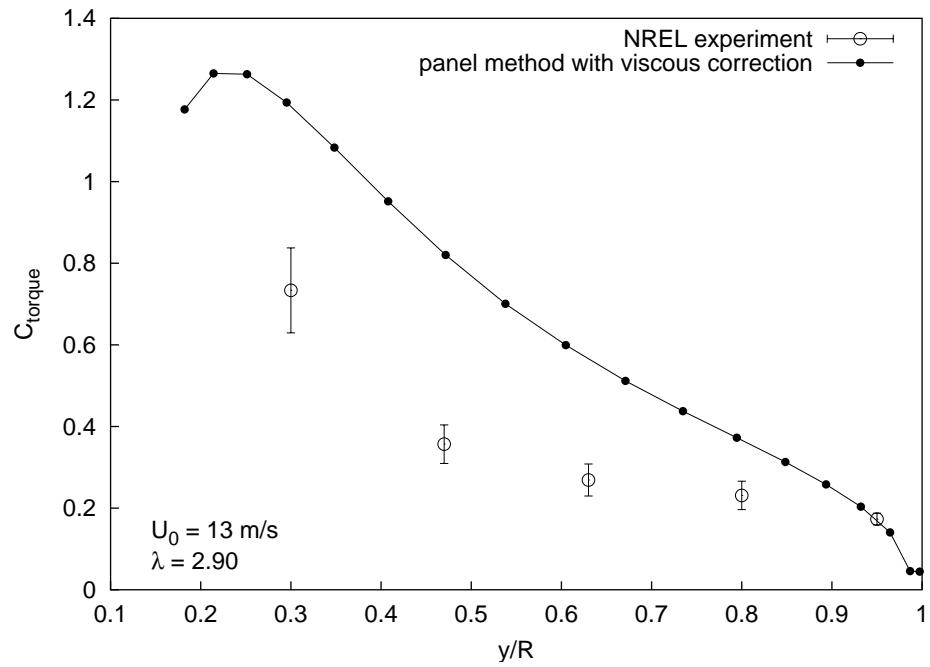
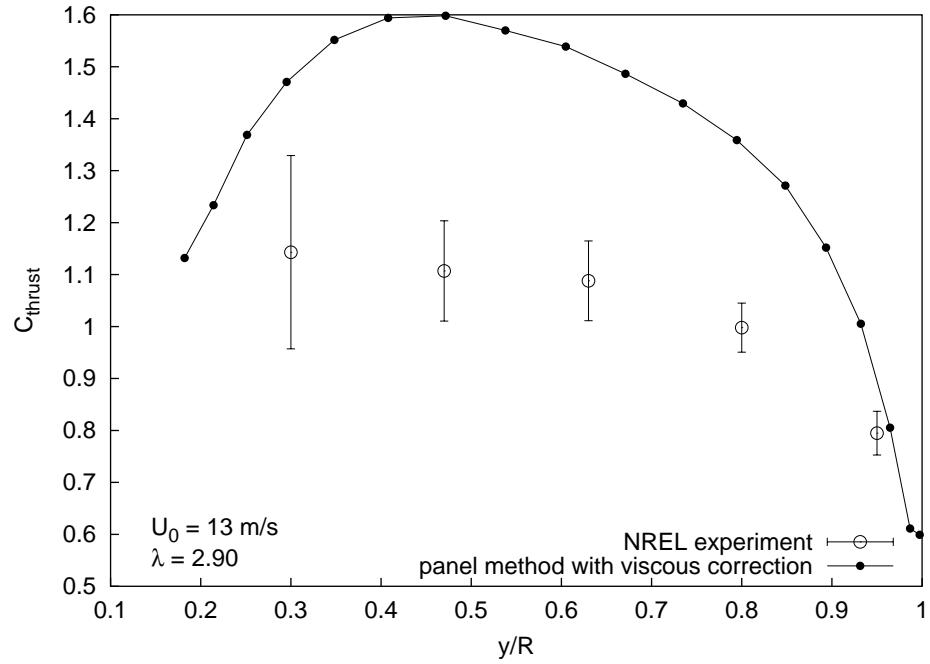
4.46.1:  $C_{torque}$  distribution :  $U_0 = 13 \text{ m/s}$ 4.46.2:  $C_{thrust}$  distribution :  $U_0 = 13 \text{ m/s}$ 

Figure 4.46: Comparison with experimental NREL Phase III rotor: spanwise  $C_{torque}$  and  $C_{thrust}$  distributions at  $U_0 = 13 \text{ m/s}$

#### **4.10.5 Test case: small, twisted, tapered wind turbine blade**

Giguère and Selig (1999) present a 5.03 metre blade geometry for use in the NREL Combined Experiment Rotor trials. The blade employs a single S809 profile along its span and is tapered and twisted. The report presents a variety of very interesting performance calculations for both two-bladed and three-bladed turbine implementations, with their blade element code “PROPID” employed to provide performance predictions.

To model this rotor, a B-spline representation of the S809 aerofoil was created by first fitting a least-squares curve to the aerofoil’s co-ordinates. Chord, twist and span distributions provided by Giguère and Selig (1999) were then used to create a skinned bi-cubic B-spline surface representation, which was subsequently discretised into a  $40 \times 20$  quadrilateral surface panel mesh. Comparisons of the panel code’s predictions and those of Giguère and Selig’s Blade Element/Inverse Design code PROPID (Giguère and Selig (1997)) appear in the following plots. Figure 4.47 again demonstrates the good power prediction capabilities of the panel code, especially at the higher tip speed ratios. The simple viscous boundary condition correction can be seen to provide good qualitative agreement for the shape of the power curve at the lower tip speed ratios. This would seem to add further weight to the proposition that a viscous/inviscid interaction scheme would dramatically enhance the panel method’s performance. Figure 4.48 demonstrates one way in which the current simple attempt at a viscous model fails. It does, however, provide a qualitative improvement over the inviscid thrust prediction. Figure 4.49 shows that the source of this discrepancy is due to large over-predictions in lift close to the blade hub. These get worse as  $U_0$  increases, with an accompanying decrease in the tip speed ratio. This is predictable: comparatively large local angles of attack and low local speed ratios over the inboard region of the blade create viscous flow conditions which are very difficult to model, even for computationally expensive grid-based Navier-Stoke solvers. Perhaps a more elaborate (and/or better-considered) empirical correction will provide a useful stepping stone between the current simple model and a viscous-inviscid approach.

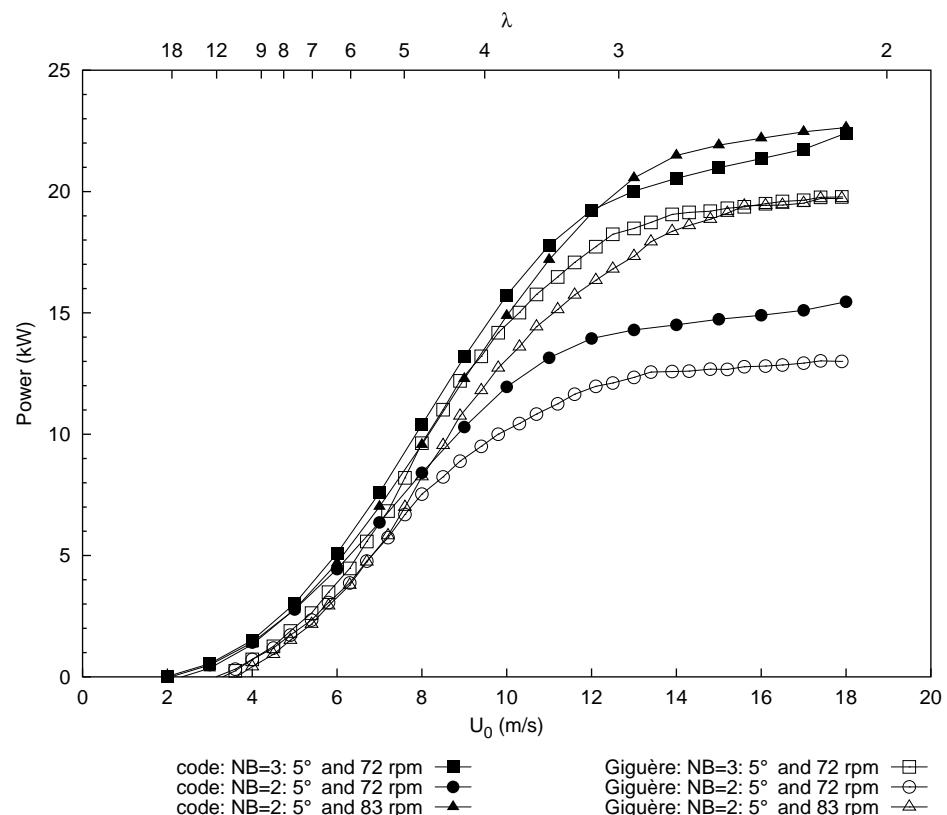


Figure 4.47: Panel code (with simple viscous correction) and PROPID power predictions for three rotor configurations

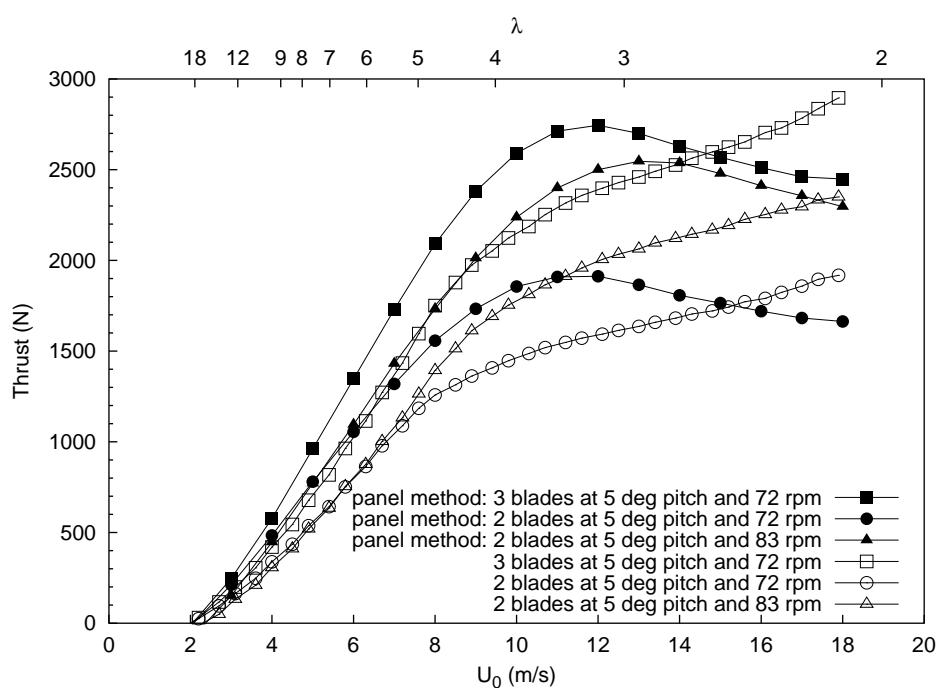


Figure 4.48: Panel code (with simple viscous correction) and PROPID thrust predictions for three rotor configurations

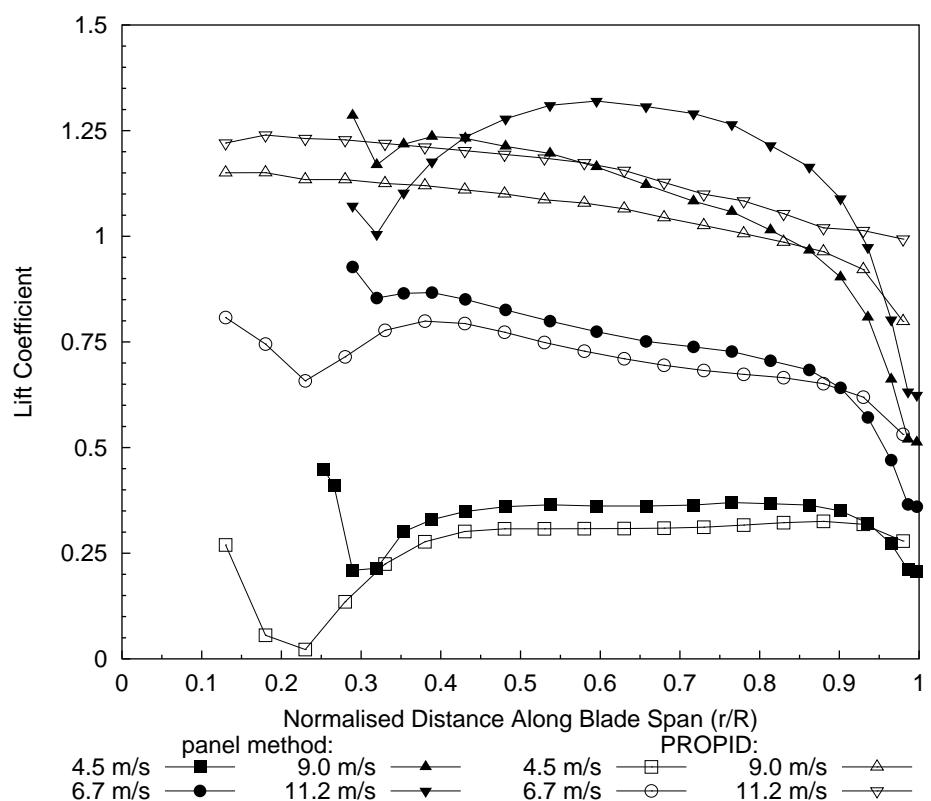


Figure 4.49: Panel code (with simple viscous correction) and PROPID lift predictions.  
NB=3: 5 deg pitch and 72 rpm



# Chapter 5

## A simplified starting model

### 5.1 Introduction

Ebert and Wood (1997) showed that the starting sequence of small wind turbines begins with a slow acceleration from rest (or “idling period”). This may sometimes be preceded by an initial brief “burst” of acceleration from rest which quickly diminishes. Once a  $\lambda$  is reached, the idling period ends with rapid acceleration to power-producing speeds, a phenomenon we will call “take-off”. The time taken from rest to take-off can be substantial, even in moderately fast winds. Figure 5.1, for instance, shows that a typical starting sequence of a 5 kW wind turbine in an 8 m/s wind took approximately 50 seconds. The reason for this is clear: small wind turbine blade geometries are optimised for an operating  $\lambda$  of between 6 and 10. This results in very large angles of attack along the entire span during starting, and thus very low aerodynamic tangential force components. The painstaking work of the aerodynamicist in creating the high-lift, low Reynolds number aerofoils counts for little at low rotor speeds, since the oncoming wind sees a large, flat expanse of the pressure-side of the aerofoil, and little else. The first design challenge, then, is to create aerodynamic shapes which give better torque performance at very high angles of attack whilst maintaining the optimal torque performance at rated tip speed ratio of current state-of-the-art aerofoil designs. This task is made a little easier by the fact that aerodynamic drag does not need to be considered for a stationary blade.

The rotational acceleration experienced by any system rotating about an axis with

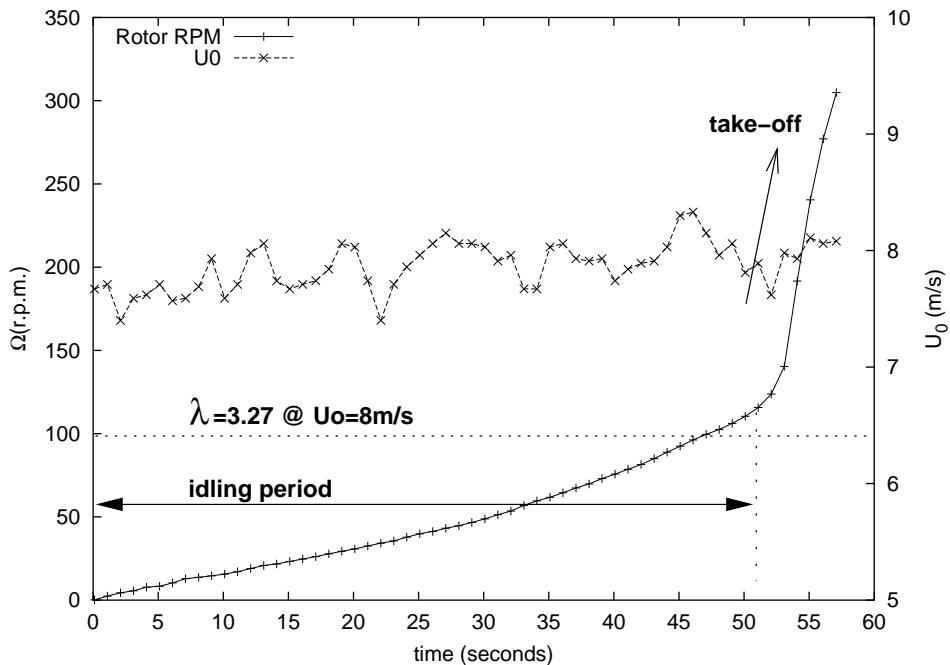


Figure 5.1: The starting sequence of a 5 kW turbine (Mayer et al. (2001))

second moment of inertia,  $I$ , acted up on by a torque  $T$  is:

$$\dot{\Omega} = \frac{T}{I} \quad (5.1)$$

A seemingly ideal way to improve starting acceleration would thus involve an increase in the chord length along a portion of the span of the blade to simply scale-up the existing very small starting torques. The problem here lies with the large increase in volume and second moment of inertia (or rotational mass) which accompanies larger blades. Volume scales with the cube of chord, and inertia scales with the square of volume (assuming constant blade density), whilst the aerodynamic force normal to the blade surface scales linearly with chord. So it becomes evident that even marginally more slender blades (with identical section) will have significantly lower values of  $I$ , and hence better acceleration for only slightly lower torques. The second design challenge, then, is to find a successful trade-off between chord length and aerodynamic shape so that torque is maintained over the whole operational tip speed ratio range whilst reducing the second moment of inertia of the blades about the axis of rotation as far as possible. Note that  $I$  is only relevant when the blades are *accelerating* or

*decelerating*, and so is irrelevant for power production.

A third supplementary design challenge is raised by structural concerns: blades should not be so slender (or cross-section so thin) that the stresses induced in the blade become excessive. This is a particular problem near the blade root: very thick aerofoil sections have traditionally been selected for inboard positions to cope with the large root bending moments caused by aerodynamic thrust (axial force component). This is problematic for starting, since it has been shown that thick aerofoil sections perform badly at low  $Re$  (Sato and Sunada (1995) and Laitone (1996)). Giguère and Selig (1998) provides an example of the substantial effort devoted to producing structurally *and* aerodynamically suitable root aerofoil sections.

### 5.1.1 Resistive torque

The resultant torque on a wind turbine rotor will be the difference between the aerodynamic torque and the resistive torque of the machine to which the rotor is attached

$$T = T_{aero} - T_{machine} \quad (5.2)$$

The actual  $T$  will vary greatly from machine to machine and will be a function of many variables, ranging from mechanical design (gearbox resistive torque, lubrication, oil seals, bearings) to electrical machine parameters, all of which will be affected to a lesser or greater degree by environmental factors (temperature, humidity), operating speed and machine wear-and-tear.

$T_{machine}$  was assumed in all subsequent calculations to be a small constant torque acting in the opposite direction to the accelerating torque. It was set to 4 Nm, a value considerably higher than the measured static resistive torque offered by the generator on the Fort Scratchley 5 kW turbine (about 1 Nm). This was intended to promote “good” low- $\lambda$  torque performance - a strategy which will be shown in the Results section to be beneficial for the survival of designs in the evolving blade population.

It should be kept in mind that the resistive torque of any real wind machine will have an effect on the “cut-in”, or minimum, wind speed at which electrical power is first produced as the wind speed increases. Manufacturer’s and/or experimental data detailing resistive torque (perhaps over a range of shaft speeds) would thus be of great assistance when selecting a generator for a real wind machine.

## 5.2 Second moment of inertia calculation

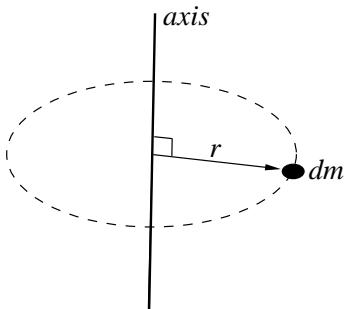


Figure 5.2: a small mass rotating about an axis

The mass moment of inertia of any arbitrary body about an axis is calculated as the “second moment” integral of all the differential-sized elements with mass  $dm$  of which it is comprised:

$$I = \int_m r^2 dm = \int_V \rho r^2 dV \quad (5.3)$$

where  $dV$  is the elemental volume,  $\rho$  the density of the material from which the element is made and  $r$  the distance between the element centroid and the axis of rotation.

### 5.2.1 Volume discretisation

The integral in equation 5.3 can be approximated by a summation over  $N$  finite “bricks”

$$I \approx \sum_{i=1}^N r_i^2 \rho_i v_i \quad (5.4)$$

with the mass of each brick approximated by an equivalent point mass located at its centroid.

In the current implementation, the blade’s volume is discretised into six-sided, eight-node bricks, or hexahedrons, using the vertices of the existing surface panels detailed in previous chapters. The volume of the blade is divided into M\_PANELS bricks in the spanwise direction, N\_PANELS/2 bricks in the chordwise direction and three bricks through the blade thickness. Figure 5.3 shows the simple model adopted here, with a “skin” of fibreglass bricks surrounding a foam brick core. N\_PANELS/2 bricks are initially created (one brick thick through the blade thickness), with external edges provided by the blade panelling and internal edges formed by constructing

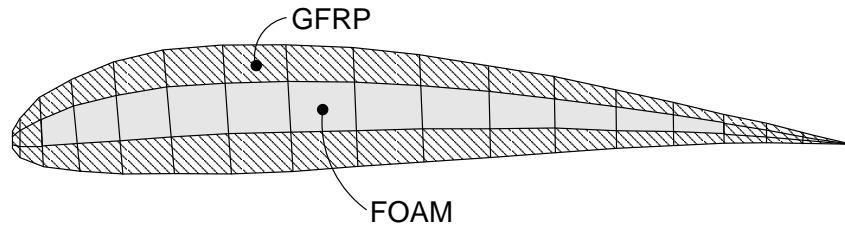


Figure 5.3: Blade modelled as a skin of fibreglass bricks surrounding a foam core

a vector pointing from each upper-side vertex to the closest lower-side vertex. These internal edges are subsequently divided to create the desired number of bricks through the blade depth. The thickness of the glass-fibre reinforced plastic (GFRP) skin is specified by the user as a fraction of the length of the local internal edge. An arbitrary GFRP wall thickness of 30% adequately models the construction of the physical blade and gives good numerical results. As with the physical blade, the foam core is bounded by solid GFRP at the leading and trailing edges for stability and strength. In the current model these solid fibreglass areas extend three bricks from the leading edge and five bricks from the trailing edge.

The sum in Equation (5.4) will approximate the integral in Equation (5.3) more accurately as the mesh density increases. The spanwise/chordwise divisions provided by the surface panelling algorithms (typically,  $40 \times 20$ ) has proven to be adequate in the current implementation, with mass and inertia calculations for the 5 kW Newcastle blade being within 10% of their measured values.

### 5.2.2 Connecting surface and through-thickness vertices into discrete hexahedral bricks

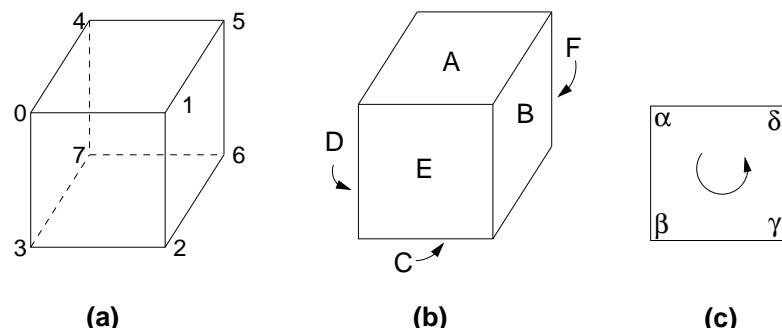


Figure 5.4: Scheme for brick connectivity: (a)vertices (b)faces (c)face vertex order

Figure 5.4 shows the brick connectivity scheme. Each brick is thus defined by its eight vertices, and is implemented as a two dimensional array (in C) as:

$$\begin{aligned} \text{int } facet[6][4] = & \{\{4, 0, 1, 5\}, \{5, 1, 2, 6\}, \{6, 2, 3, 7\} \\ & \{7, 3, 0, 4\}, \{0, 3, 2, 1\}, \{7, 4, 5, 6\}\} \end{aligned} \quad (5.5)$$

### 5.2.3 Volume and Centroid calculation

Any general polyhedron can be subdivided into tetrahedra, each sharing one surface with a surface of the polyhedron, with remaining surfaces formed by rays extended from the surface's vertices to any other polyhedron vertex. The volume and centroid of any general polyhedron with  $n_f$  facets, with  $n_v$  vertices per facet is calculated by summing over the volumes and centroids of the tetrahedra which comprise it:

$$v_{ij} = (\vec{r}_{i,0} - \vec{r}_{ref}) \times (\vec{r}_{i,j+1} - \vec{r}_{ref}) \cdot (\vec{r}_{i,j+2} - \vec{r}_{ref}) / 6 \quad (5.6)$$

$$\vec{c}_{ij} = mean(\vec{r}_{ref}, \vec{r}_{i,0}, \vec{r}_{i,j+1}, \vec{r}_{i,j+2}) \quad (5.7)$$

$$v = \sum_{i=0}^{n_f-1} \sum_{j=0}^{n_v-3} v_{ij} \quad (5.8)$$

$$\vec{c} = \frac{1}{v} \sum_{i=0}^{n_f-1} \sum_{j=0}^{n_v-3} \vec{c}_{ij} \cdot v_{ij} \quad (5.9)$$

where  $v$  is the sum of tetrahedral volumes and  $\vec{c}$  is the sum of volume-weighted tetrahedral centroids. For a hexahedron with quadrilateral facets,  $n_f=6$  and  $n_v=4$ . The  $\vec{r}$  refer to the brick vertices, with  $\vec{r}_{ref}$  being any fixed point on the hexahedron (vertex[0] is used in the current implementation).

These elemental volumes are used with Equation (5.4) to determine an approximation to the blade second moment of inertia. Total blade volume is determined by simply summing the elemental volumes. The blade mass is similarly the sum of the elemental masses ( $\rho v$ ), with  $\rho_{GFRP} = 1500 \text{ kg/m}^3$  and  $\rho_{Foam} = 150 \text{ kg/m}^3$ . The blade centroid is determined by summing and averaging the volume-weighted elemental centroids (similar to the elemental centroid determination). Predictions of inertial properties of the Newcastle 5 kW blade using this scheme (2.5 metres long, root transition modelled, 30% GFRP wall thickness,  $40 \times 20 \times 3$  volume discretisation) give acceptable results: mass  $\approx 6.5 \text{ kg}$ ,  $I \approx 6.41 \text{ kgm}^2$  and centroid at approximately one-third-chord and one-

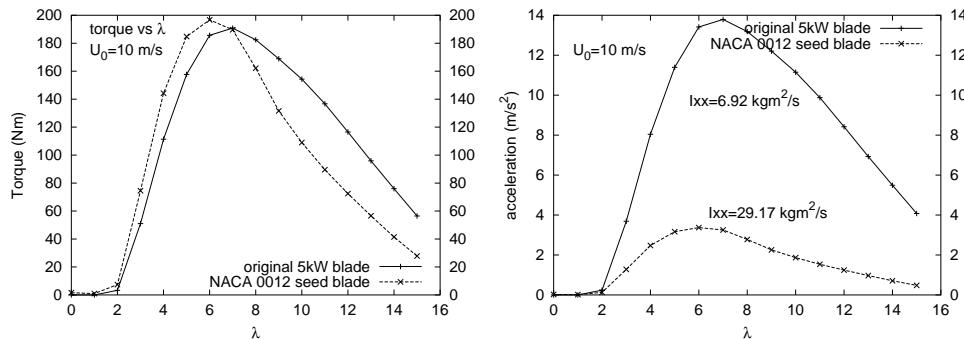
third-span positions (measured from the blade leading edge and hub, respectively). As stated earlier, these are within 10% of their measured values.

Data structures, vertex connectivity and volume mesh creation routines as used in the current project are outlined in the appendices on pages 444 and 461.

### 5.3 Assessing blade starting performance

The two components for assessing candidate blade designs are now available: an aerodynamic solver for calculating the induced torque experienced by the turbine rotor; and a method for calculating the second-moment of inertia of each blade. The best procedure for obtaining a starting acceleration measure will always involve the numerical integration of equation (5.1) over time, with a fourth-order Runge-Kutta routine for example. Needless to say, the large number of panel code evaluations this would entail would be very computationally expensive, even for a single blade evaluation over a short span of time. For this reason, a simplification was adopted.

Figures 5.5.1 and 5.5.2 show panel code predictions of torque and acceleration for the existing 5 kW blade geometry and an untapered, untwisted “seed” blade geometry (which is introduced in the following chapter).



5.5.1:

5.5.2:

Figure 5.5: Torque and Acceleration curve comparisons for the existing 5 kW blade and a straight, untwisted “seed” geometry constructed from NACA 0012 aerofoils

Immediately obvious is these plots is the dramatic increase in torque (and accompanying instantaneous, no-load acceleration) as the blade accelerates past a tip speed

ratio of  $\lambda=2$ . It would seem that a method for accurately determining the aerodynamic torque at a tip speed ratio less than 2 is required to optimise true “starting” performance. Figure 5.6 shows how the panel solver predicts comparatively tiny spanwise torques along the blade during starting, which steadily progress to significant (and perhaps much more accurate) values as the tip speed ratio increases. This suggests that a

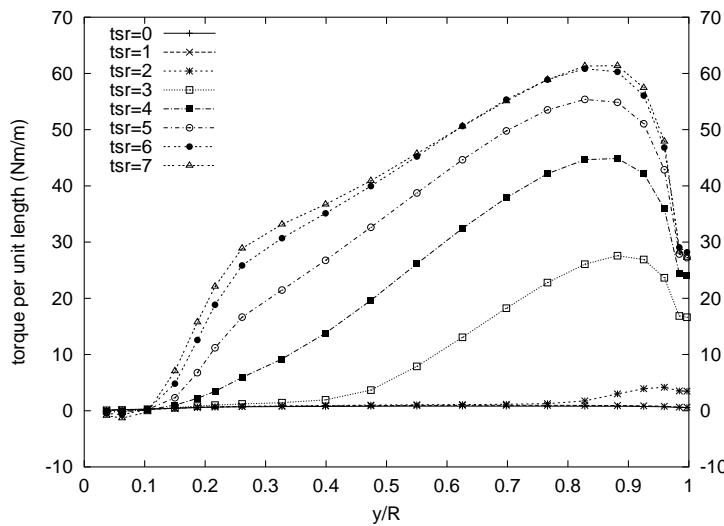


Figure 5.6: Predicted starting torque distribution for existing 5 kW blade

$\lambda$  greater than 2 may be the lowest tip speed ratio that can realistically be considered as a reliable measure of torque in the proximity of starting, since to evaluate a potential blade design for starting *from rest* would require a better wake model, as well as a viscous aerodynamic solver capable of dealing with separated flow with considerably better accuracy at low-Re and high- $\alpha$  than the current panel solver can deliver. In fact, current state-of-the-art viscous/inviscid interaction methods do not necessarily work for separated flows. Applying a panel solver with anything less than an accurate (read: computationally intensive) viscous/inviscid interaction method at tip speed ratios less than 3 would thus risk obtaining meaningless results.

The starting sequence shown in figure 5.1 suggests that a tip speed ratio of 3 is at the upper-end of the idling period and so may nevertheless be low enough to offer some measure of “starting” performance close to stationary rotor conditions, even with the current panel method. Here the assumption is made that optimising for better  $\lambda=3$  acceleration will also improve stationary performance. Whether this assumption is valid for *general* blade geometries will remain a question for future studies.

## 5.4 A linear-static finite element model

The eight-node bricks used for the inertial calculations are convenient for use in a finite element solver for structural performance calculations. Only a crude first approximation to the stresses in an operating wind turbine blade is sought in the current study, a fortunate requirement due to the coarseness of the mesh described in the previous section ( $40 \times 20 \times 3$ ). The focus of an evolutionary optimisation routine will be to compare the maximum stresses and strains *between evolving blade designs*, rather than between a model and experimental results. Suitable blade designs produced in this manner may be subsequently evaluated individually with finer, more accurate meshes. This scheme reduces the computational burden of the evolving population by several orders of magnitude whilst still providing interesting and useful geometries as the basis of the complex fabric lay-ups used in modern small wind turbine blades.

The linear-static finite element code used in this project is SLFFEA Le (2000), a simple, free, finite element analysis system with source code. As a first approximation, the blade was modelled as the assembly of eight-node bricks described in the previous section, with bulk Young's moduli for the GFRP and foam bricks taken as 70 GPa and 1 GPa, respectively. Brick vertices lying on facets on the hub-wise face were constrained to give zero degrees of freedom to approximate the way the real blade is bolted to the generator hub. To keep the model as simple as possible, the aerodynamic force acting normal to the centre of each surface panel was relocated to vertex[0]. This reduces the number of nodes required in the volume mesh whilst maintaining reasonable accuracy. Loads arising from the blades' rotation are addressed by calculating the centripetal acceleration about the axis of rotation of each brick and then multiplying by the brick mass

$$\vec{f}_i = (m\vec{a})_i = -m_i \Omega^2 \vec{r}_i \quad (5.10)$$

where  $\vec{r}_i$  is normal to the axis of rotation, and is the projection of the vector pointing from the axis of rotation to the brick centroid

$$\vec{r}_i = \vec{c}_i - (\vec{c}_i \cdot \vec{i}) \vec{i} \quad (5.11)$$

where  $\vec{i}$  is the unit vector along the axis of rotation (taken here as the global X axis). For simplicity this load is applied to the mesh at vertex[3].

This additional structural solver routine is currently implemented as follows: the panel code is executed on a rotor configuration to determine aerodynamic loads. A hexahedral brick mesh is then created from the panel discretisation with centripetal loads and inertial properties calculated from this mesh. Vertices, brick connectivity, loads and constraints are then saved to an output file. The external command-line SLFFEA finite element solver is subsequently triggered and reads the data file as input. A results file produced by SLFFEA, containing stresses, strains and displacements at each blade node, is thus produced which can be used to judge the blade’s “structural fitness”. A graphical display of stress and strain distributions through the blade is also possible via a post-processor.

The current optimisation code used the FEA solver as a component of the objective function. A blade’s “structural fitness” is measured by finding the node in each blade mesh exhibiting the highest *von-Mises equivalent stress*, when acted upon by aerodynamic and centripetal loads at the upper tip speed ratio operating point ( $\lambda=10$ ):

$$\bar{\sigma}_{VM} = \frac{1}{\sqrt{2}} \sqrt{(\sigma_{xx} - \sigma_{yy})^2 + (\sigma_{yy} - \sigma_{zz})^2 + (\sigma_{zz} - \sigma_{xx})^2 + 6(\tau_{xy}^2 + \tau_{yz}^2 + \tau_{zx}^2)} \quad (5.12)$$

with fitter blades having a lower maximum von-Mises equivalent stress than their peers.

A graphical post-processor was written as part of the current project. Sample screen shots of a typical set of results for a 5kW turbine blade appear in the following pages.

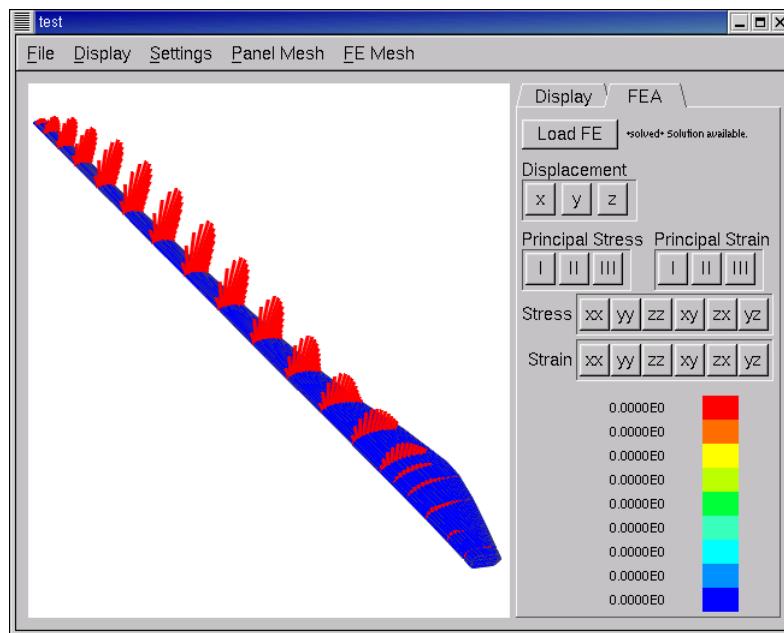


Figure 5.7: Aerodynamic loads; two-bladed 5kW HAWT;  $U_0=10$  m/s  $\lambda=10$

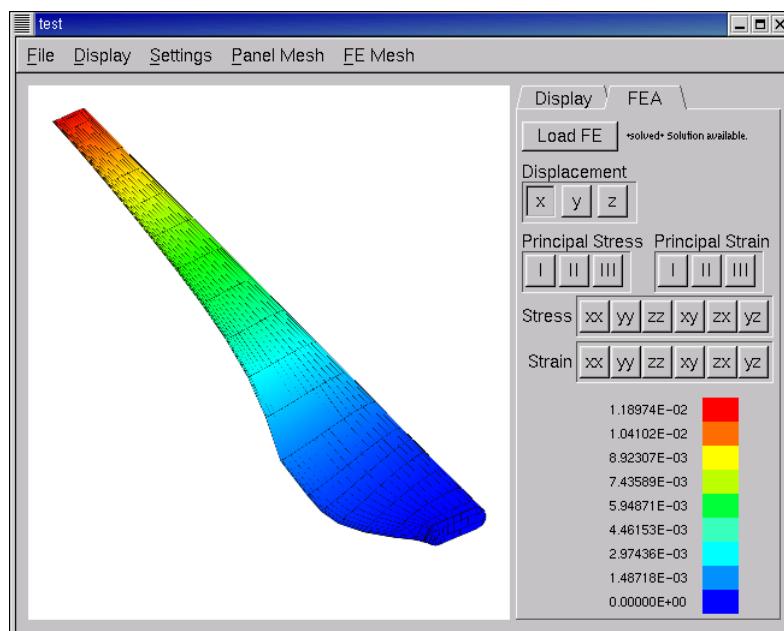


Figure 5.8: Displacement in x direction; two-bladed 5kW HAWT;  $U_0=10$  m/s  $\lambda=10$



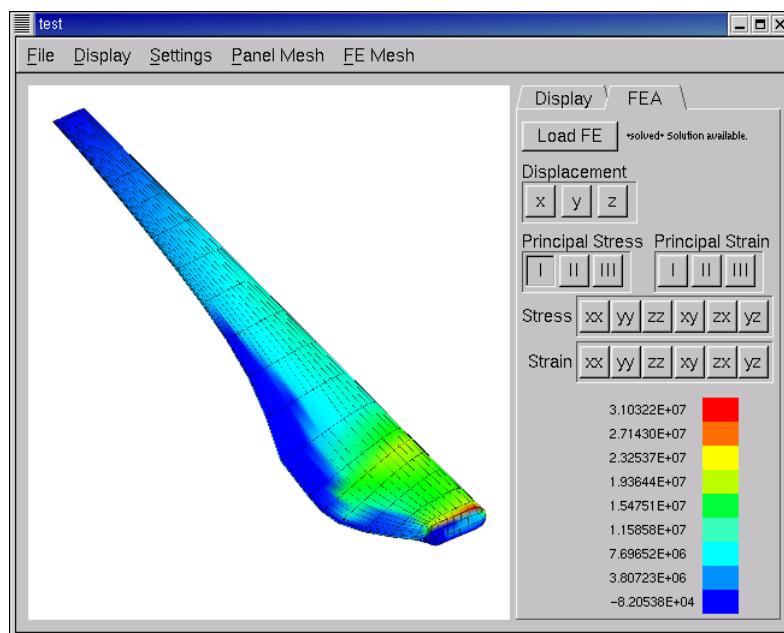


Figure 5.9: Principal stress; two-bladed 5kW HAWT;  $U_0=10$  m/s  $\lambda=10$

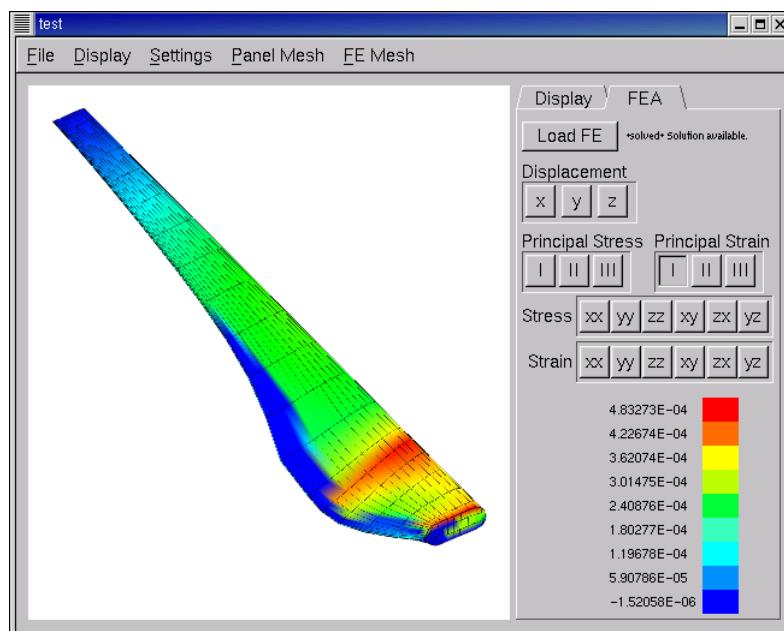


Figure 5.10: Principal strain; two-bladed 5kW HAWT;  $U_0=10$  m/s  $\lambda=10$



# **Chapter 6**

## **Multiobjective Differential Evolution wind turbine blade optimiser**

### **6.1 Introduction**

The aerodynamic and structural solvers described in previous sections provide the framework for a number of *objective functions*, or evaluation routines. These form the core of the multiobjective optimisation technique described in this Chapter. Drawing inspiration from biological evolution, an objective function serves as the *environment* into which a candidate solution is placed and where it is forced to compete for survival. The *fitness* of an individual candidate solution in a particular environment is evaluated by an objective function, which typically returns a scalar *fitness score*. The fitness of any individual is a measure of how well it copes with the stressors imposed by its environment. Fitness is the sole input to a *selection* operator, the arbiter of whether any particular candidate solution is fit enough to survive competition with its peers and thus have the opportunity to pass its encoded information on to the next generation.

A single-objective optimisation routine will judge a *population* of candidate solutions using one objective function, returning one fitness score per individual per *generation*, or optimisation iteration. Equivalently, a single-objective optimisation routine can be built around more than one objective function if each of these individual fitness scores are somehow converted into just one, combined score. It has been shown that this is indeed a worthwhile strategy if the objectives do not compete: if an improvement in one has no bearing on the outcome of any of the others. If, for example, a

horticulturist wishes to breed a large, red variety of apple, and if the biological process that control apple size have no bearing on apple colour (and vice versa), then the fitness scores returned by the two objective functions judging a candidate apple's size and colour could be successfully combined to yield one defining fitness score (by weighted sums, the product of the two or by any number of ways).

However, when conflict arises between multiple objectives, as is very often the case, another strategy must be adopted. In real physical systems there is usually competing interaction between performance objectives, and a balance must be reached by the system designer: a trade off. An optimal design in such cases may thus be composed of sub-optimal configurations of the individual objectives that comprise it. If the exact relationships between the competing objectives are known and can be formulated as a scalar function (increasing "redness" by 20% decreases size by 30%) then the task can again be successfully phrased as a single-objective optimisation problem. The interaction between competing objectives in physical systems is, however, usually poorly understood, or if well understood then very difficult to formulate in terms of an interaction function. The construction of an aggregate scalar fitness score for most multiobjective problems is thus extremely difficult, owing to the ambiguity as to *how* to successfully combine the individual fitness scores so that no one objective function is unwittingly favoured at another's expense. The case in point in the current project concerns the effect of blade shape on starting performance, peak power production and (to a lesser extent) blade stresses and strains. There is possibly a strong (negative) correlation between starting performance and peak power production, since they both depend on the aerodynamics of the system, which is entirely dependent on the three-dimensional blade shape. But what is it? Nobody has studied starting with the depth required for the formulation of definitive rules which would indicate how altering blade shape affects starting performance, let alone how these changes would subsequently affect power production.

One possible strategy is to simply ignore any explicit interaction between objective functions and treat the problem as a number of *mutually evolving* optimisation problems. This is the basis of the concept of *Pareto optimality* as used in Evolutionary Optimisation (see Fonseca and Fleming (1998a)). In the Pareto concept, "fitness" must change from the one-dimensional view (solutions are ranked according to one scalar fitness value; there is one "fittest" individual in the population per generation) to one

where there is a selection of candidates, each of whom provides a better solution to one or more objectives than any of its peers. (A definition of Pareto optimality appears later in this Chapter). Here, if we have  $N$  competing objectives, then we also have an  *$N$ -dimensional object space* that contains every possible solution to our problem, as well as an  *$N$ -dimensional trade-off surface*, on which the best, or fittest, or *non-dominated* candidate solutions lie. The designer's job is thus transformed from one of running an optimisation code and plucking the single "best" individual from the population, to a task of selecting from a range of individuals, each of which provides a uniquely "good" solution.

The multiobjective optimisation routine implemented in the current project is built on the foundation of a single-objective evolutionary optimisation technique known as Differential Evolution.

## 6.2 Single-objective Differential Evolution algorithm

The Differential Evolution algorithm (Storn and Price (1995) and Storn (1996)) "evolves" populations of real-valued object parameter vectors. Each parameter vector encodes the salient features of one candidate member in that population, serving much the same function as a biological organism's DNA. In practice, a real-valued representation of each candidate solution is simply stored as a one-dimensional, real-valued parameter array, or vector (denoted as  $\vec{x}$ ), of length  $D$ .

$$\vec{x} = \{x_0, x_1, \dots, x_{D-1}\} \quad (6.1)$$

An objective function,  $g$ , evaluates each object vector and returns a real scalar fitness value,  $f$

$$f = g(\vec{x}) \quad (6.2)$$

A *population*,  $P$ , containing  $NP$  different object vectors is created and maintained over a number of generations,  $G$

$$\vec{P}_G = \{\vec{x}_{0,G}, \vec{x}_{1,G}, \dots, \vec{x}_{NP-1,G}\}, \quad 0 \leq G \leq G_{max} \quad (6.3)$$

where  $G_{max}$  is the maximum number of algorithm iterations and is chosen arbitrarily by the user. The individual vector parameters in the initial population are usually

taken from a pool of uniform-random numbers, constrained between the extremes of allowable parameter values.

$$\vec{x}_{*,0} = \{R_0, R_1, \dots, R_{D-1}\} R_i = \text{uniform\_rand}_i[0, 1] \cdot (x_i^U - x_i^L) + x_i^L, \quad 0 \leq i < D \quad (6.4)$$

Another popular initialisation method is to start with a parameterisation of an existing solution,  $\vec{x}'$ , and then generate a population of object vectors from this “seed” by adding arbitrarily small perturbations drawn from a Gaussian random distribution (mean=0, variance=1) to each parameter.

$$\vec{x}_{*,0} = \{x'_0 + R_0, x'_1 + R_1, \dots, x'_{D-1} + R_{D-1}\} R_i = \text{gaussian\_rand}_i[0, 1] \cdot \delta, \quad 0 \leq i < D \quad (6.5)$$

where the perturbed parameters are rejected and re-calculated if they fall outside acceptable constraints. (Outlines of the code for the random number generators used in this project appear in the appendices on page 414). In both initialisation methods it is important to have as much diversity in the initial population as possible - the variance in the random numbers (or random perturbations) used should be large enough so that an even distribution of parameter vectors over the available parameter space results. To this end, a perturbation multiplier ( $\delta$  in Equation (6.5)) can be used to scale the Gaussian deviate added to each the seed variable, and a different  $\delta$  used for each variable, allowing the inclusion of knowledge about the expected ranges of each variable into the initial population.

The preceding establishes the foundation on which the DE optimiser is built, and is common to all Evolutionary Computation methods which operate on real-valued parameter vectors. Where DE differs substantially to other EC techniques, such as Genetic Algorithms and Evolution Strategies, is in how “evolution” takes place, that is, how the traditional evolutionary operators of Crossover, Mutation and Selection are implemented.

The major conceptual difference is the *trial vector*,  $\vec{v}$ , which is generated for each existing parameter vector as the first step in each iteration of the algorithm. A range of possible crossover strategies have been proposed to create the trial vector, the most popular of which is the DE/rand/1 scheme

$$\vec{v} = \vec{x}_{r_1,G} + F(\vec{x}_{r_2,G} - \vec{x}_{r_3,G}) \quad (6.6)$$

where  $r_1$ ,  $r_2$ , and  $r_3$  are random integers drawn uniformly from the range [0,NP-1], and form indices which are mutually different to each other and the index of the current parameter vector. F is an arbitrary weighting factor chosen by the user and is greater than zero and usually less than 1.

The comparison vector  $\vec{u}$  is then formed

$$\vec{u} = (u_1, u_2, \dots, u_D) \quad (6.7)$$

$$\text{with } u_j = \begin{cases} v_j & \text{for } j = \langle n \rangle_D, \langle n+1 \rangle_D, \dots, \langle n+L-1 \rangle_D \\ (x_{i,G})_j & \text{otherwise} \end{cases} \quad (6.8)$$

where  $\langle \cdot \rangle_D$  denotes the modulo function with modulus D. This is implemented by first generating a random starting index, n, then looping to find an offset index, L, via

```
L=0;
for(i=0;i<D;i++){
    L++;
    minimal_rand2(x); // a uniform random number on [0,1]
    if(x>CR || L>=D)break;
}
```

where CR is the crossover probability which, along with the F multiplier, constitutes the only other user-supplied algorithm control variable. CR lies on the range [0,1], as does the (real) random number rand() to which it is compared. This results in the comparison vector depicted in Figure 6.1.

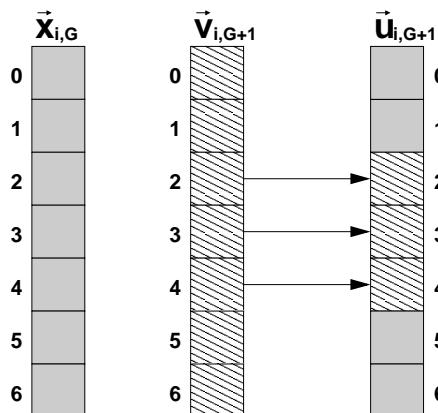


Figure 6.1: Crossover for D=7, n=2 and L=3

The Storn and Price algorithm simply ignores offset indices which are greater than D. This is modified slightly in the current implementation by “wrapping” large offset indices over to the beginning of the vector (figure 6.2).

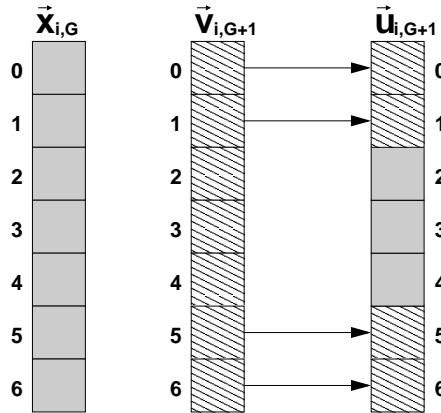


Figure 6.2: Crossover for  $D=7$ ,  $n=5$  and  $L=4$

In this manner a comparison vector,  $\vec{u}_{i,G+1}$ , is generated for each object vector in the population at each generation ( $\vec{x}_{i,G}$ ). The single-objective *selection* operator, then, simply involves obtaining a fitness score for this trial vector, comparing it with the fitness of the current object vector (obtained at a previous generation) and then either discarding the trial vector if its fitness is worse than that of the current object vector or replacing the current object vector with the trial vector if the trial vector’s fitness is superior. If the trial vector is indeed superior, an identical comparison takes place between the fitness of the trial vector and that of the current “best” vector, and the index of the current best vector stored for future use.

### 6.3 Blade parameterisation

The first challenge in assembling an evolutionary optimisation technique, then, is to find a way to represent a wide range of blade geometries as real-valued, one-dimensional object vectors. Associated challenges are

- To keep the object vector as compact as possible. A rule of thumb for successful DE optimisation is that the population size should be ten times that of the object vector dimension. Larger populations experience greater genomic diversity

which naturally leads to more effective parameter space searches. This, not surprisingly, raises a problem: Smaller viable populations result in fewer expensive objective function evaluations, and thus shorter optimisation run times. Whilst the rule of thumb is impractical in the current context (it would result in populations of about 700 individuals), it remains an important feature of the current implementation that population sizes be as large as possible, even if this means that a reduction in panel density (and thus evaluation accuracy) is required to keep code run times within manageable limits.

- To ensure that the parameter space covered by the encoding method is capable of representing the widest practical variety of possible blade shapes. This conflicts with the “compact object vector” requirement, since longer object vectors result in higher resolution searches which are naturally able to explore smaller niches of parameter space.

The flexibility in geometric modelling afforded by B-spline curves and surfaces goes a long way to offering a trade-off between these two conflicting requirements. Smooth, complicated B-spline surfaces can be parameterised with just few real-valued control point and knot coordinates, and their shape subtly or greatly changed by modifying just one of these parameters, making them very suitable for the sort of manipulations encountered during an evolutionary optimisation process.

The blade parameterisation adopted in the current project consists of a knot vector in the parametric U-direction common to all B-spline aerofoil curves along the blade span and a sequence of B-spline aerofoil parameter vectors. Each two-dimensional B-spline curve along the blade span is fully described by a chord length, a twist angle, its position along the span of the blade (ie: an ordinate along the Y axis), and a sequence of control point co-ordinates in the X-Z plane. This parameterisation scheme is shown in Figure 6.3. To reduce the parameter vector length, only *unique* knots and control points are stored. Specifically,  $p+1$  knots at the start and end of the common knot vector can be removed and restored when the blade is reconstituted - by definition they are  $\{0,0,0,0\}$  and  $\{1,1,1,1\}$  for a curve of degree 3. Similarly, the control points at the start and end of each aerofoil curve are omitted because they are constrained to  $(1,0)$  in the X-Z plane to give a sharp trailing edge. It should also be noted that each set of control point co-ordinates encode the shape of one unit-chord, untwisted aerofoil

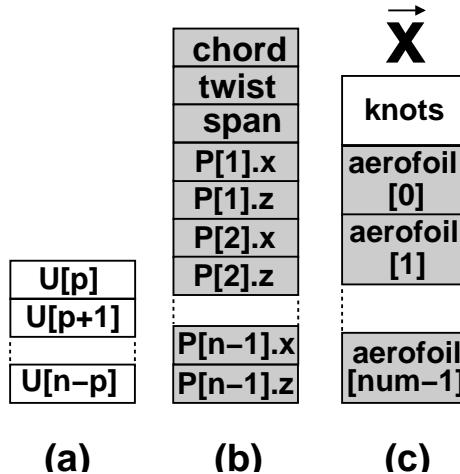


Figure 6.3: (a)common knot vector (b)one aerofoil vector (c)complete parameter vector

curve.

Decoding the object vector to give a skinned B-spline blade surface is, for the most part, a reversal of the encoding process, plus an extra step to skin a surface in the parametric V-direction using the technique given in Chapter 3. The C-code implementation of this encoding/decoding process are contained in the object\_vector class. This is outlined on page 374.

## 6.4 Pareto optimality

Consider the optimisation problem of finding a real-valued object vector  $\vec{x}$  by minimising  $k$  objective functions  $f_1(\vec{x}), f_1(\vec{x}), \dots, f_k(\vec{x})$  operating on  $\vec{x}$  (with any objective function solving for a maximum fitness value being converted into a minimisation measure by simple inversion). A vector  $\vec{x}_a$  is said to be *partially less* than another vector  $\vec{x}_b$  (both taken from the population  $P = \{\vec{x}_0, \vec{x}_1, \dots, \vec{x}_{num-1}\}$ ) if all of its objective function scores are at least as low as the latter's, and if at least one objective function score is strictly lower

$$\begin{aligned} \forall f_i(\vec{x}_a) &\leq f_i(\vec{x}_b) \quad (i = 1, \dots, k), \text{ and} \\ \exists f_i(\vec{x}_a) &< f_i(\vec{x}_b) \end{aligned} \tag{6.9}$$

In this case, vector  $\vec{x}_a$  *dominates*  $\vec{x}_b$ .

$\vec{x}_a$  is *Pareto optimal*, or non-dominated, if no other vector in a particular population

(or set) dominates it. The *Pareto optimal set*, then, consists of all the non-dominated solutions possible in the N-dimensional parameter space being considered, and form an N-dimensional trade-off surface. In multiobjective evolutionary optimisation it is convenient to keep track of all the non-dominated vectors generated during an optimisation run in another, distinct, population, the Pareto optimal set, which exists alongside the evolving population but does not take part in any “evolutionary” activity. After each comparison vector is evaluated and compared to its peer in the evolving population, a similar comparison is made between it and *with every vector in the Pareto optimal set*. If it dominates at least one other member in the set then it is allowed into the set. Conversely, any member of the Pareto optimal set which is dominated by the comparison vector is removed. In this way the Pareto optimal set improves as the optimisation proceeds, and in the limit converges on the “true” optimal trade-off surface, or *Pareto optimal front*.

The single-objective DE selection operator is thus simply transformed from one in which a better fitness score gives a comparison vector the right to enter the evolving population (and replace an existing population member) to one where a comparison vector which is partially less than its peer is favoured. This and the addition of a separate population containing a Pareto optimal set of object vectors are the only differences between the single-objective and multi-objective DE algorithms, and are easily implemented.

#### 6.4.1 Pareto selection

The Pareto selection operator described previously involves both a mixed inequality/equality comparison as well as a strict inequality comparison. A method for comparing floating-point fitness values for equality is thus required. The current implementation uses the following function:

```
bool equal(double a, double b){
    double tol=1.E-4;

    if (fabs(a-b) <= tol*fabs(a)) return TRUE;
    else return FALSE;
}
```

A function implementing the Pareto selection operator, which judges the relative fitness of the three objective fitness scores defining the current optimisation problem is as follows:

---

#### Function 6.1: pareto\_selection.cc

---

```
bool Pareto_selection(solver_results & results1 , solver_results & results2){
    bool inequalities , equalities ;

    inequalities =FALSE;
    equalities =FALSE;

    //1: All objectives should be at least equal.
    //Take "equality" to at least three significant figures
    if ((results1.tsr3_acceleration > results2.tsr3_acceleration ||
        equal(results1.tsr3_acceleration , results2.tsr3_acceleration ))
        && (results1.tsr10_Cp > results2.tsr10_Cp ||
        equal(results1.tsr10_Cp, results2.tsr10_Cp))
        && (results1.tsr10_max_stress < results2.tsr10_max_stress ||
        equal(results1.tsr10_max_stress, results2.tsr10_max_stress)))
    ) equalities =TRUE;

    //2:There should be at least one strict inequality
    if (results1.tsr3_acceleration > results2.tsr3_acceleration
        || results1.tsr10_Cp > results2.tsr10_Cp
        || results1.tsr10_max_stress < results2.tsr10_max_stress
    ) inequalities =TRUE;

    if ( inequalities && equalities )return TRUE;
    else return FALSE;
}
```

---

#### 6.4.2 Selecting for viability

The current optimisation makes a distinction between *viable* and *non-viable* blade designs, and treats them differently with regards to the selection operator. Viability is judged by the behaviour of the pressure coefficient ( $C_p$ ) vs chord data along each spanwise blade element in a blade design. A viable design satisfies both of the following requirements

- Trailing edge pressure equalisation is within a reasonable tolerance (the Kutta

condition).

- Trailing edge pressure coefficient magnitudes are “sane”, reflecting proper solver convergence.

Failure to satisfy either of these conditions, at any blade element strip, renders an “unviable” flag being registered for that design. Since two runs of the panel code are required to judge a blade, each blade design carries two flags: a low TSR flag and a high TSR flag. Excursions from the tolerance are also punished, with torque or  $C_p$  magnitudes suffering proportionally to the excursion. A “fully” viable blade design is thus one which delivers well-behaved pressure results at both low rotor speeds and high rotor speeds, and is rewarded by not having its performance figures downgraded.

A “viability score” then allows comparison between blade designs:

- Score=0: blade is fully unviable
- Score=1: blade is viable for either low TSR solutions or high TSR solutions
- Score=2: blade is fully viable

The viability score also offers a convenient way to stigmatise blade designs that transgress other performance constraints. An “unviable” flag is set if:

- $I_{xx}$  (the moment of inertia) less than 0.0 (or not finite)
- low- $\lambda$  torque less than 0.0 Nm or greater than 500.0 Nm (or not finite)
- high- $\lambda$   $C_p$  less than 0.01 or greater than 0.65 (or not finite)

The first part of the selection operator thus involves selecting for viability, with more viable blade designs being favoured over those less viable. A newly evaluated blade design is first compared with its peer in the evolving population, taking its place if its viability score is greater. If its viability score is lower it fails to enter the evolving population. If it and its peer’s viability score are equal then Pareto selection is used. Only fully viable blade designs are considered for entry into the Pareto set, with Pareto selection being the final arbiter of whether a candidate object vector enters the Pareto optimal set.

This method ensures that the important genomic information stored in unviable blade designs (generated during the initialisation stage) is not lost via culling, whilst ensuring that desirable viable designs have a survival advantage.

## 6.5 Geometric Constraints

Constraint handling is an integral part of any general parameter optimisation method (for a review see Michalewicz and Shoenauer (1996)). Lampinen (2001), for example, provides an updated selection operator for the single-objective DE algorithm that allows it to handle complicated, perhaps non-linear object parameter constraint functions. Constraint functions allow knowledge about the relationships between various parameter variables to be built into the problem and, (as with Lampinen's method), allows a measure of how close an object vector fulfills those constraints to be used by a selection operator in much the same way as objective functions are used in a Pareto-based multiobjective optimisation method.

A specific assumption at the core of the current optimisation project is that no correlation exists between the object parameters encoding the shape of a wind turbine blade. That is, the knot parameters, control point co-ordinates and aerofoil chord, twist and span parameters which comprise an object vector are not related, and no constraint function can be formulated which would usefully direct the evolutionary search of problem space into more fruitful areas. As such, only very simple “hard” boundary constraints have been placed on a few key geometric parameters:

### 6.5.1 Hard envelope constraints

Parameter	Constraint
tip radius	2.5 metres
hub radius	0.03*tip radius
attachment section chord	0.06*tip radius
attachment section twist	0.0 degrees
minimum chord	0.02*tip radius
twist envelope	(see below)

These are enforced by simply initialising them to their constrained values during the decoding step (when the object vector is parsed to give a B-spline blade surface). The inclusion of the tip radius constraint clamps the blade to the required length. The hub aerofoil constraints exist so that a physically reasonable shape for the blade's connection to the generator hub can be specified (for example, a rectangular block of determinate dimension). The 0.06\*tip radius dimension gives the size of the attachment

section on the current 5 kW Newcastle wind turbine blade (150 mm), and was chosen simply for convenience.

All aerofoil pitch angles are specified on the range  $[-20^\circ, 90^\circ]$ . The maximum negative twist of  $-20^\circ$  was chosen arbitrarily after consideration of existing experimental lift/drag data for a number of aerofoil sections, and seemed appropriate as the largest possible negative twist (and thereby the largest angle of attack during starting). The limitation on twist angle was made necessary by the viscous correction applied to the blade model (section 4.10.1), which takes as an argument sectional angles of attack, the magnitudes of which have a dramatic effect on the aerodynamic performance of each blade element. The twist envelope is shown in figure 6.4, with the constraint implemented in Function 6.2.

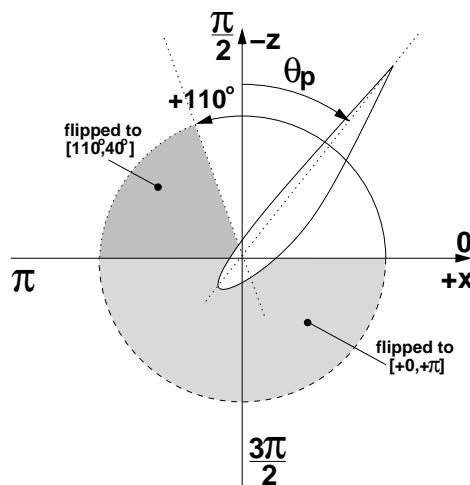


Figure 6.4: Blade twist envelope

---

Function 6.2: constrain\_twist.cc

---

```
void constrain_twist (double &theta){
    int fac;
    double max_twist;

    theta=abs(theta);
    fac = int(theta/PI); //whole-number divisor
    if (theta>PI){//transform all angles to [+0,+PI]
        if (fac%2==0) theta = theta - fac*PI; //upper two quadrants
        else theta = PI - theta + fac*PI; //lower two quadrants
    }
    //constrain maximum twist
}
```

```

max_twist = radians(110.);
if (theta>max_twist)theta = max_twist – (theta – max_twist);
return;
}

```

---

### 6.5.2 Straight leading edge constraint

A manufacturing requirement for the existing 5 kW blade is that all aerofoil sections along its span must be aligned at the leading edge. This is achieved by offsetting each B-spline aerofoil in the blade model in the +Z direction (from leading edge to trailing edge) by 0.03\*tip radius, which has the effect of moving the Y axis to the centre of the hub aerofoil. Like the other constraints, this is enforced during the object vector to B-spline blade decoding step.

### 6.5.3 The “sane geometry” constraint

The only other two constraints relate specifically to the creation of “realistic” B-spline surface geometries, and whether these can be regarded as smooth and simply-connected:

- The control polygon that defines each span-wise B-spline aerofoil must not be self-intersecting. It must form a simple loop which encloses a single “solid” area. (see intersect.cc on page 417)
- The knot vectors in both U and V directions must be monotonically increasing (function 6.3 below).

Function 6.3: bad\_knots.cc

---

```

bool bad_knots(bspline_blade& X){
    int i,n,p;

    //U-direction
    n = X.n; //common high-index of control-point vector in U direction
    p = X.p; //degree of curve in U direction

    for(i=p+1;i<n+1;i++)//unique knots (neglect head and tail knots)
        if(X.foil [0].U[i]<X.foil [0].U[i-1])return TRUE; //degenerate knot vector

```

```

//now do V direction (along span)
n = X.num-1; //high-index of control-point vector in V direction
p = X.K;      //degree of curve in V direction

for(i=p+1;i<n+1;i++)//unique knots (neglect head and tail knots)
    if(X.V[i]<X.V[i-1])return TRUE; //degenerate knot vector

    //also check to see that all spanwise curves are increasing in the Y
    //direction
    for(i=1; i<X.num; i++)
        if (X.foil [i].y < X.foil [i-1].y)return TRUE;

    return FALSE;//all knot vectors are increasing.
}

```

---

Function 6.3 also ensures that the Y-span positions of the constituent B-spline aerofoils increase monotonically, from root to tip.

The mechanism for enforcing these constraints is as follows: construct a potential candidate object vector (using the DE evolutionary operators described above); use this as the blueprint for a B-spline blade surface; check B-spline knots and control points for any irregularities; and either accept the object vector if no irregularities are found or discard the object vector and repeat the process until a suitable candidate vector is found.

The flexibility of this constraint scheme allows the generation of a huge number of physically reasonable blade shapes whilst quickly and efficiently discarding shapes that would otherwise waste precious object function evaluation time.

#### 6.5.4 Minimum thickness constraint

Optimising for minimal stress (via an FEA-based objective function) produces blade designs with substantial cross-sectional thicknesses. When the FEA objective function is removed from the evaluation (as was the case in a number of experimental runs of the optimiser, a strategy which substantially reduced the computing times required per evaluation), the optimiser invariably favoured blade shapes with very low second-moments of inertia, with correspondingly very thin cross-sectional “aerofoil” sections. The reason is clear: flat, cambered plates *will* produce substantial lift and, given the right chord and twist distributions, will offer valid wind turbine blade designs, albeit

simple ones with large stress concentrations over significant areas of the blade. The issue is one of materials and manufacturability, and whether thin, cambered aerofoil geometries made from a low-density material can withstand the sustained rigours of actual service in a wind turbine blade and still deliver low second-moments of inertia. A secondary concern is that aerofoil sections with finite thickness have the potential to perform much better aerodynamically than cambered plates, at least at high Reynolds numbers. The danger exists that “good” high- $\lambda$  aerodynamic performance and exceptionally low second-moments of inertia will dominate and supplant finite-thickness designs, thus removing their influence from the evolving population, with an inevitable reduction in population diversity and convergence to a local optimum.

The solution adopted in the current project was to set a lower limit for the possible thicknesses of each blade’s constituent B-spline aerofoil curves. The minimum thickness was specified as a nominal 10%: thinner than the NACA 0012 and SD7062 aerofoil sections used in the “seed” blade geometries, but thick enough to promote curves with rounded leading edges and sharp trailing edges, rather than very thin shapes with camber.

The following implementation partitions the aerofoil’s chord into num\_segments equal segments and finds nearest points on the upper and lower surfaces corresponding to the 1/4 and 3/4 chord points. The distance between the the 1/4 chord points gives the overall blade thickness, and the distance between the 3/4 chord points provides a convenient check for finite trailing-edge angles. Each B-spline aerofoil section in a blade design is checked, and the blade “failed” if at least one of these aerofoils is below the minimum thickness.

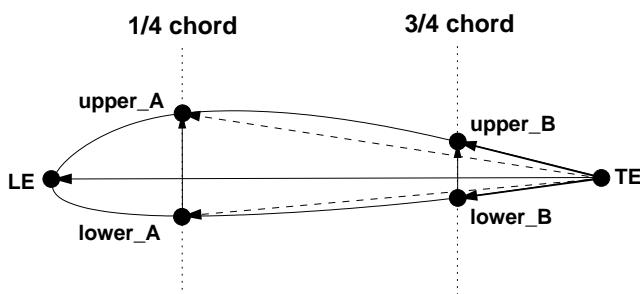


Figure 6.5: Scheme for B-spline aerofoil thickness determination

Function 6.4: `too_thin.cc`

---

```

bool too_thin(bspline_blade& X){
    double min_thickness = 0.1; //minimum 10% thick
    double thickness;
    int i;
    for(i=0;i<X.num;i++){
        thickness = get_thickness(X.foil [i]);
        //indicate failure if at least one aerofoil is too thin
        if (thickness < min_thickness)return 1;
    }
    return 0;//all aerofoils are above minimum thickness.
}

double get_thickness(const aerofoil &X){
    //a routine to find the thickness of a B-spline aerofoil
    int num_segments = 100;
    double u_upper, u_lower, delta_u_upper, delta_u_lower;
    point2D P_LE, P_TE, P_upper_A, P_upper_B, P_lower_A, P_lower_B;
    point2D P_unit_chord, point;
    double thickness;
    int i, n, p;

    n = X.n;
    p = X.p;

    C = new curve_data[n+p+2];
    copy_curve(C, X);

    delta_u_upper = X.U_LE/num_segments; //upper knot span increment
    delta_u_lower = (1.-X.U_TE)/num_segments; //lower " " "

    P_TE = CurvePoint(n, p, C, 0.0)/X.S; //normalised TE point
    P_LE = CurvePoint(n, p, C, X.U_LE)/X.S; //normalised LE point

    P_unit_chord = P_LE - P_TE;//unit vector from TE to LE

    //start at the TE, progress along chord and find the position
    //the 1/4 and 3/4 points on the upper and lower surfaces
    u_upper = 0.0;
    u_lower = 1.0;
    for(i=1;i<num_segments;i++){
        u_upper += delta_u_upper;
        u_lower -= delta_u_lower;
    }
}

```

```

//find the upper 1/4 and 3/4 chord points
point = CurvePoint(n, p, C, u_upper)/X.S;
if (fabs((point-P_TE)*P_unit_chord) <= 0.75)P_upper_A = point;
if (fabs((point-P_TE)*P_unit_chord) <= 0.25)P_upper_B = point;

//find the lower 1/4 and 3/4 chord points
point = CurvePoint(n, p, C, u_lower)/X.S;
if (fabs((point-P_TE)*P_unit_chord) <= 0.75)P_lower_A = point;
if (fabs((point-P_TE)*P_unit_chord) <= 0.25)P_lower_B = point;
}

thickness = NORM(P_upper_A - P_lower_A);
//ensure that the aerofoil has a finite trailing edge angle
if (NORM(P_upper_B - P_lower_B) < thickness/4.)
    thickness = NORM(P_upper_B - P_lower_B);

delete [] C;
return thickness;
}

```

---

The thickness check is performed alongside the “sane geometry” checks immediately after a new B-spline blade has been created from a new trial vector. Failed blades are rejected, prompting a new trial vector/B-spline blade to be created and pre-processed.

## 6.6 Experimental run parameters

### 6.6.1 Task specification

Find a range of bi-cubic B-spline blade surfaces that are 2.5 metres long and that have

- good starting performance
- good peak power production
- reasonable induced stresses during normal operation

at  $U_0=10$  m/s - the rated wind speed of the 5 kW Newcastle wind turbine. To this end the optimisation method selected to accomplish this goal is the Differential Evolution algorithm with multiobjective Pareto selection.

### 6.6.2 Objective functions

The three objectives in the task specification give rise to three competing objective functions:

- **Starting performance:** Panel method torque prediction at  $\lambda = 3$ ;  $U_0 = 10\text{m/s}$  coupled with a finite-element second moment of inertia calculation, to give a measure of instantaneous rotational acceleration.
- **Peak power production:** Panel method  $C_P$  prediction at  $\lambda = 10$ ;  $U_0 = 10\text{m/s}$
- **Induced stresses** SLFFEA structural finite element solver returning a prediction of maximum von Mises equivalent stress at a node in the FE mesh, employing the aerodynamic and centripetal loads calculated at  $\lambda = 10$ ;  $U_0 = 10\text{m/s}$ . Aerodynamic and centripetal loads during starting are small in comparison with operational loads ( $\lambda=10$ ), and the stresses they induce were ignored.

### 6.6.3 Seed blade geometry

Each blade in the initial population (the primordial soup, so to speak), is based on an arbitrary *seed blade* - in this case untapered and untwisted - constructed from just four symmetrical NACA 0012 aerofoils and one “hub” section. This rather extreme seed geometry was chosen for two reasons: the desire for the optimiser to “discover” novel blade geometries rather than improve an existing blade design; and the necessity of having to severely reduce the number of aerofoil sections along the span of the blade, and the number of control points/knots in each aerofoil, to keep the object vector dimension to an absolute minimum. The untwisted, untapered blade can be represented accurately by very few span-wise B-spline aerofoils (two, in fact, plus one for the hub section), making possible future comparisons with geometries employing fewer or greater B-spline aerofoils.

The bi-cubic ( $p=3$ ) B-spline surface formed from five B-spline aerofoil sections has  $n+1=5$  sets of control points in the V (span-wise) direction and  $n+p+2=9$  V knots, eight of which are constrained to the end points, by definition. Although this leaves only one free knot along the span, the flexibility of the modelling scheme allows for a huge array of possible blade shapes, as well as providing the likelihood that the taper and twist distributions will be smooth.

The first step towards a seed blade geometry with a minimum number of defining parameters, then, is to approximate the NACA 0012 aerofoil with just seven control points - the smallest number of control points that allows both the pressure-side and suction-side of an aerofoil to each be represented by two cubic sub-curves curves (one devoted to the representing the high-curvature near the leading edge and one for the rest of the side) with the first and last of the control points clamped at the trailing edge. The least-squares curve fitting technique described in Chapter 3 was first used to fit a B-spline curve to SD7062 point data, with fitting accuracy low enough so that only five free control points were generated (Figure 6.6 - axes not to scale).

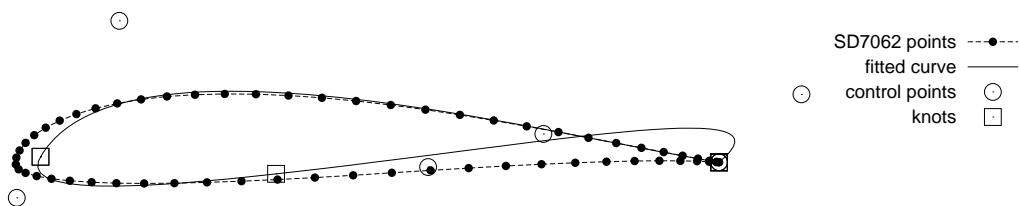


Figure 6.6: Least-squares curve fit to SD7062 point data with 7 control points

Control Point	knot
-	0.0
-	0.0
(1.000000, 0.000000)	0.0
(0.750231, 0.044893)	0.0
(0.147466, 0.224416)	0.514471
(0.001534, -0.056513)	0.514471
(0.586399, -0.007698)	0.669838
(1.117168, 0.107737)	1.0
(1.000000, 0.000000)	1.0
-	1.0
-	1.0

This curve has five free control points and three free knots and served as a prototype for a DE curve optimiser to give a better approximation to the SD7062 aerofoil coordinate data. Since the free knot parameters are arbitrary, they were altered to give a more regular curve: one at the leading edge (0.5), and two equally spaced on either side (0.25 and 0.75). The DE curve optimiser was applied to the control points of this proto-curve, giving the geometry shown in figure 6.7 (axes not to scale).

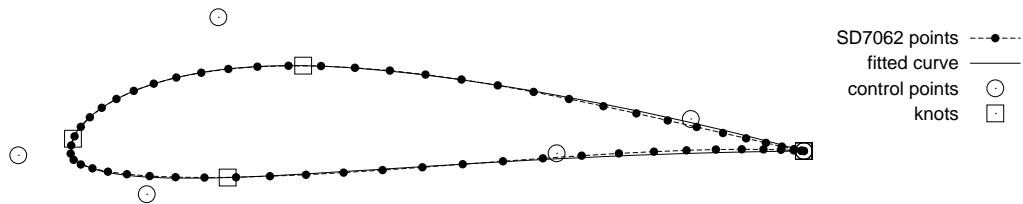


Figure 6.7: B-spline approximation to the SD7062 aerofoil with 7 control points

Control Point	knot
-	0.0
-	0.0
(1.000000, 0.000000)	0.0
(0.845757, 0.040721)	0.0
(0.201615, 0.170112)	0.25
(-0.071203, -0.005459)	0.5
(0.103872, -0.054884)	0.75
(0.662741, -0.002882)	1.0
(1.000000, 0.000000)	1.0
-	1.0
-	1.0

Apart from a little thickening near the trailing edge, the DE curve fit is quite good and, more importantly, is smooth. This minimal SD7062 B-spline aerofoil is used in the “Original” 5 kW comparison blade, whose performance figures form the baseline against which all subsequently evolved blades are judged.

The minimal SD7062 B-spline aerofoil was then treated as a prototype curve, and the same DE optimiser used to fit a minimal B-spline curve to NACA 0012 point data, giving the geometry shown in figure 6.8.

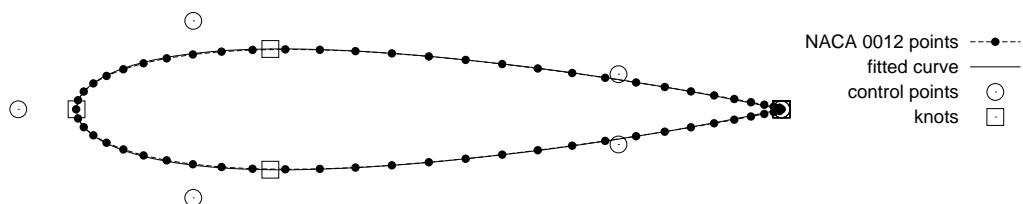


Figure 6.8: B-spline approximation to the NACA 0012 aerofoil with 7 control points

Control Point	knot
-	0.0
-	0.0
(1.000, 0.000)	0.0
(0.769, 0.0351)	0.0
(0.166, 0.0885)	0.25
(-0.082, 0.000)	0.5
(0.166, -0.0885)	0.75
(0.769, -0.0351)	1.0
(1.000, 0.000)	1.0
-	1.0
-	1.0

A “hub” aerofoil section was then created to blend the rest of the blade with the rectangular attachment section. This was defined on the same knot vector as the seed aerofoil and thus has the same number of control points, but does not include the sharp corners of the attachment section because of the difficulties this causes the aerodynamic solver (seven control points and three free knots are inadequate for modelling four sharp corners, anyway). Instead, a connector-piece shape with a “sharp” trailing edge and a rounded leading edge was adopted to promote a smooth transition (figure 6.9).

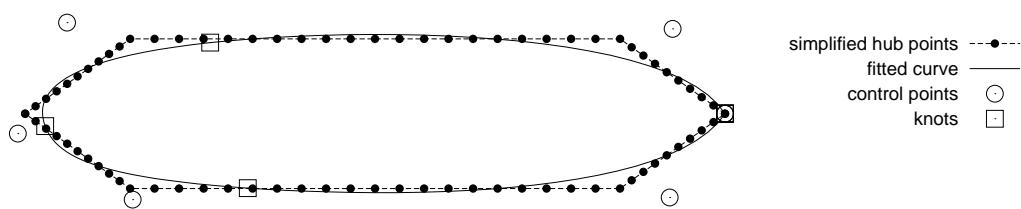


Figure 6.9: A simplified rectangular hub section

Control Point	knot
-	0.0
-	0.0
(1.000000, 0.000000)	0.0
(0.925096, 0.170246)	0.0
(0.059840, 0.182758)	0.25
(-0.010433, -0.039911)	0.5
(0.153738, -0.172208)	0.75
(0.920869, -0.167981)	1.0
(1.000000, 0.000000)	1.0
-	1.0
-	1.0

The seed blade geometry was finally constructed from an untwisted, untapered assembly of four minimal NACA 0012 B-spline aerofoil sections and one rounded hub section:

B-spline aerofoil	chord(m)	twist( $^{\circ}$ )	Y span(m)
hub	0.15	0.0	0.0750
NACA 0012	0.25	0.0	0.6250
NACA 0012	0.25	0.0	1.250
NACA 0012	0.25	0.0	1.875
NACA 0012	0.25	0.0	2.500

Performance results for this seed blade geometry appear in the following pages.

## 6.7 Optimisation results: 5 kW blade

### 6.7.1 Existing 5 kW blade geometry

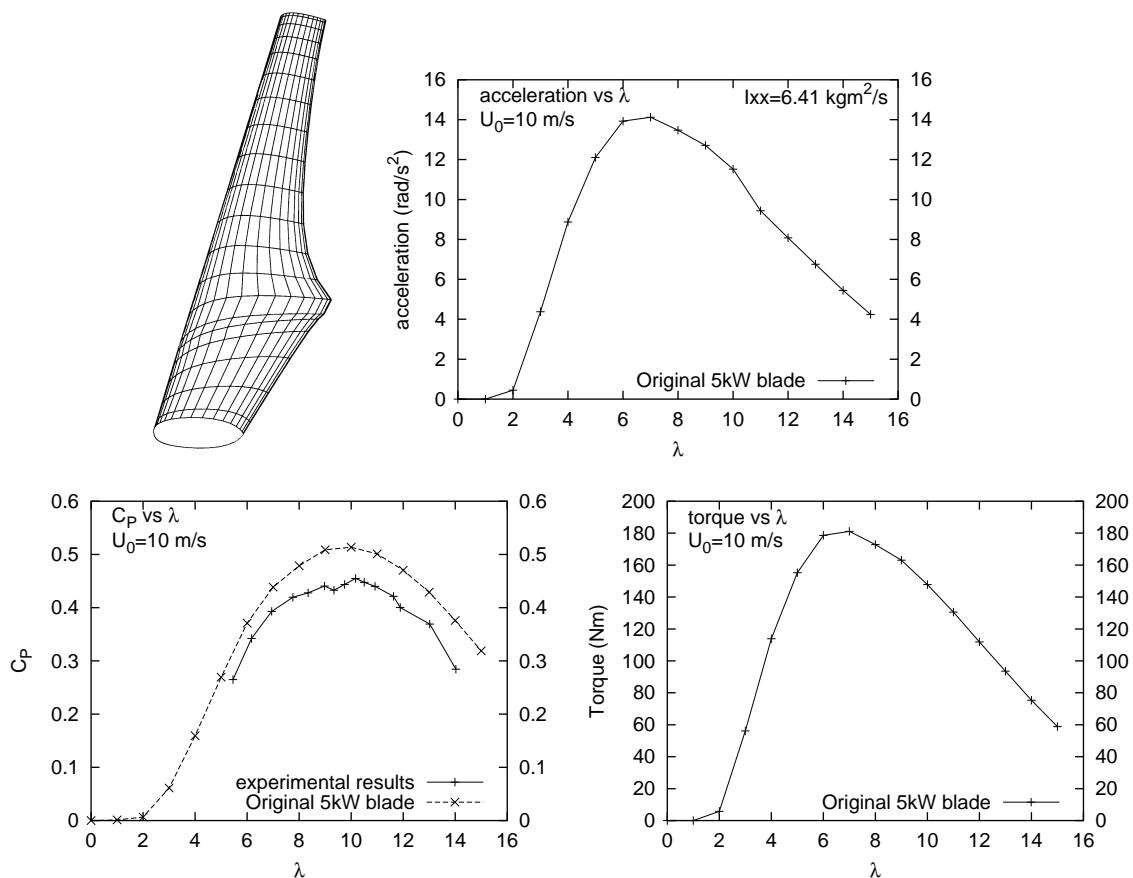


Figure 6.10: Original 5 kW blade - Geometry and performance results

A B-spline representation of the existing 5 kW Newcastle wind turbine blade was used as the benchmark geometry for the current optimisation study. The blade representation consisted of a bi-cubic B-spline surface skinned over 29 SD7062 B-spline aerofoil curves arranged along the blade span, each scaled and rotated to reflect the chord and twist distributions of the actual blade geometry. The standard “hub” aerofoil section described in the previous section was added at the blade root. The blade surface was discretised into a  $40 \times 20$  mesh of quadrilateral surface panels. A wake model consisting of 5 turns was added, with each turn divided into 20 radial quadrilateral surface panels. A depiction of the meshed blade surface appears in 6.10.

The panel method was then applied across the range of tip speed ratios spanning  $\lambda=0$  to  $\lambda = 10$ , with various performance measures recorded after each calculation. These measures form the baseline against which all other blade geometries in the following optimisation study are compared. Figure 6.10 shows acceleration, power, and torque performance results, with other relevant measures appearing in the following pages.



B-spline curves	degree	control points	U-knots	V-knots
30	3	7	11	34

6.11.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.11.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
6.4135	0.5136	6071	1048	60.12	4.344	7.40

6.11.3: RESULTS

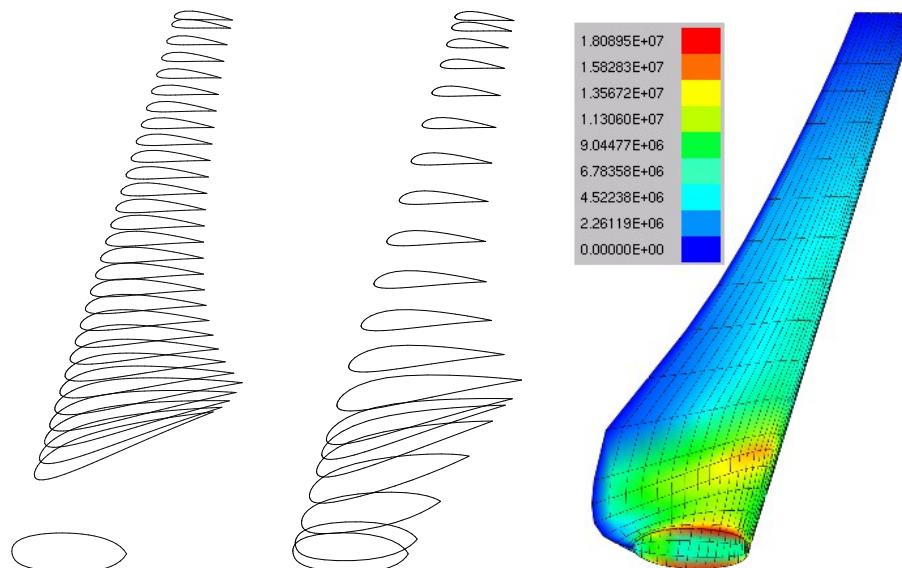
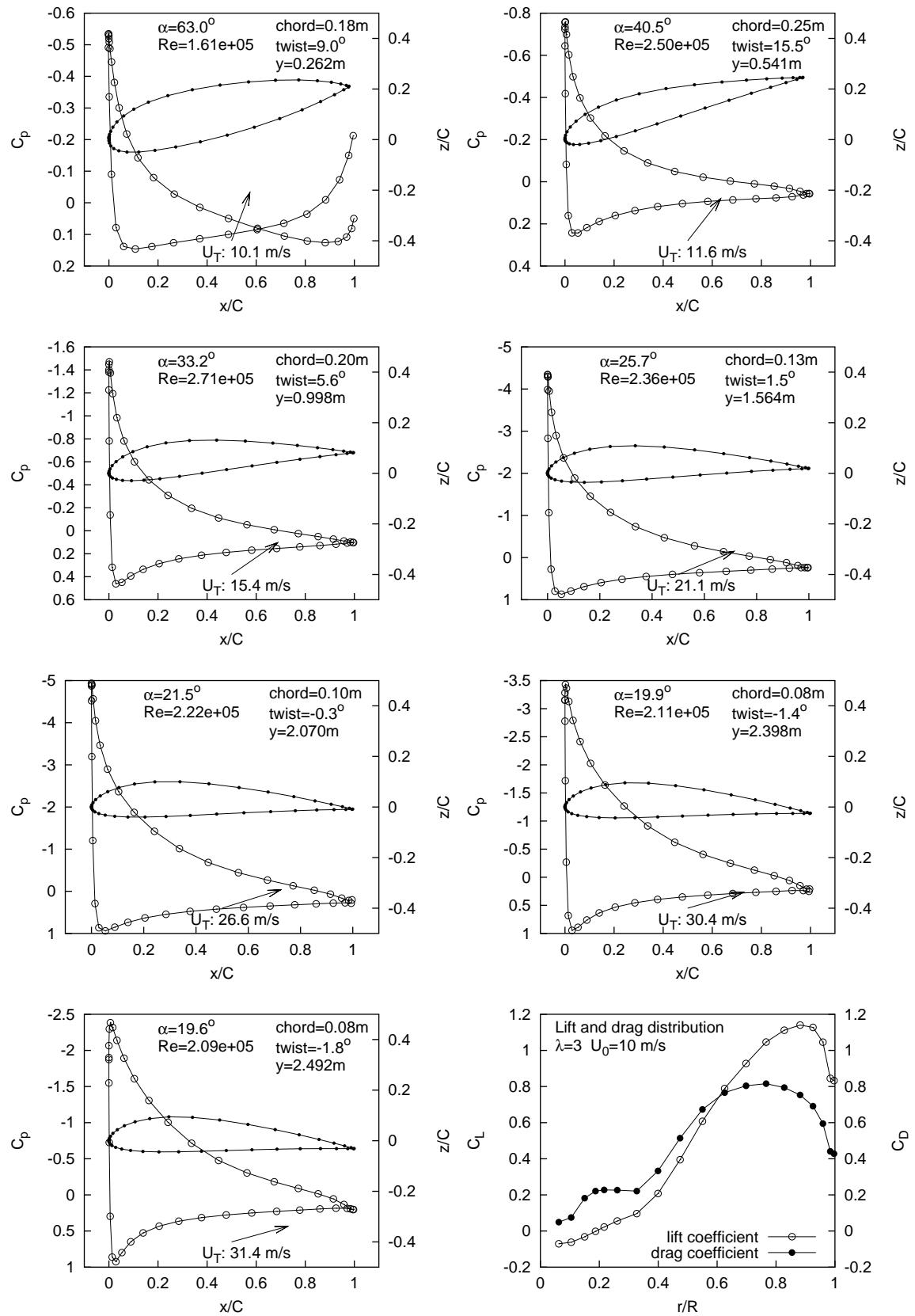


Figure 6.11: Original 5 kW blade - Geometry and performance results (continued)



Figure 6.12: Pressure and force distributions along Original 5 kW blade @  $\lambda=3$

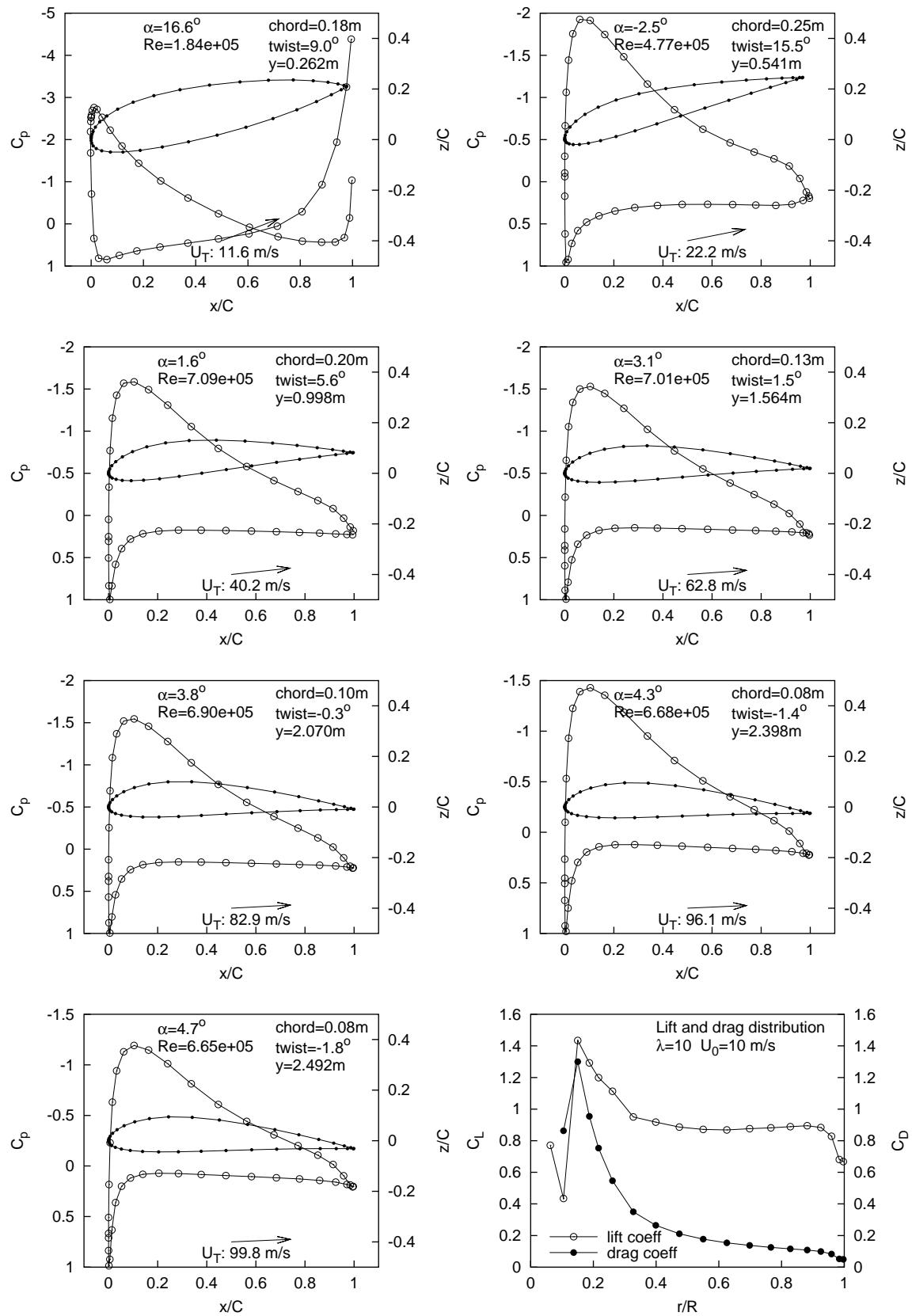


Figure 6.13: Pressure and force distributions along Original 5 kW blade @  $\lambda=10$

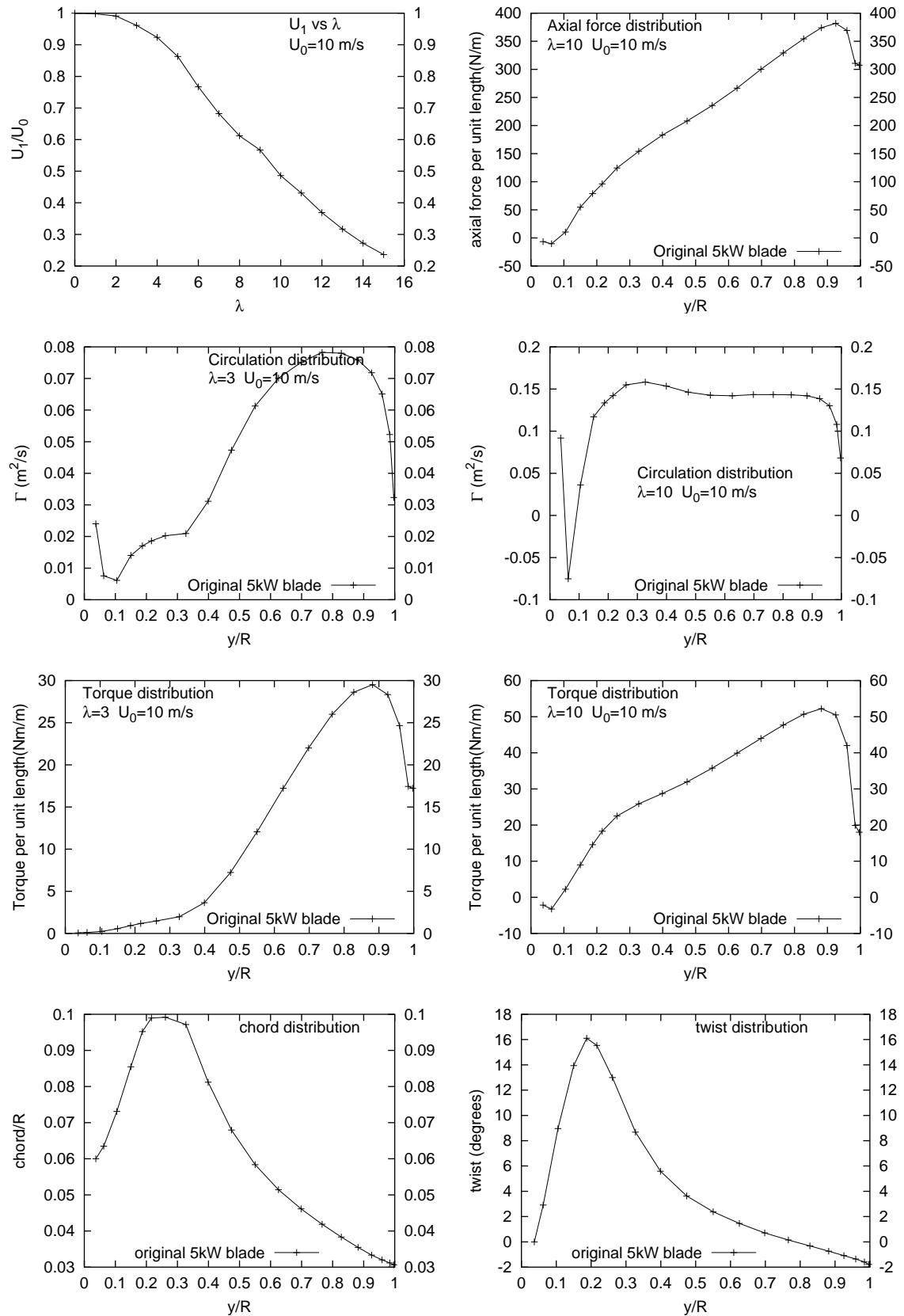


Figure 6.14: Performance results for Original 5 kW blade

### 6.7.2 NACA 0012 seed blade

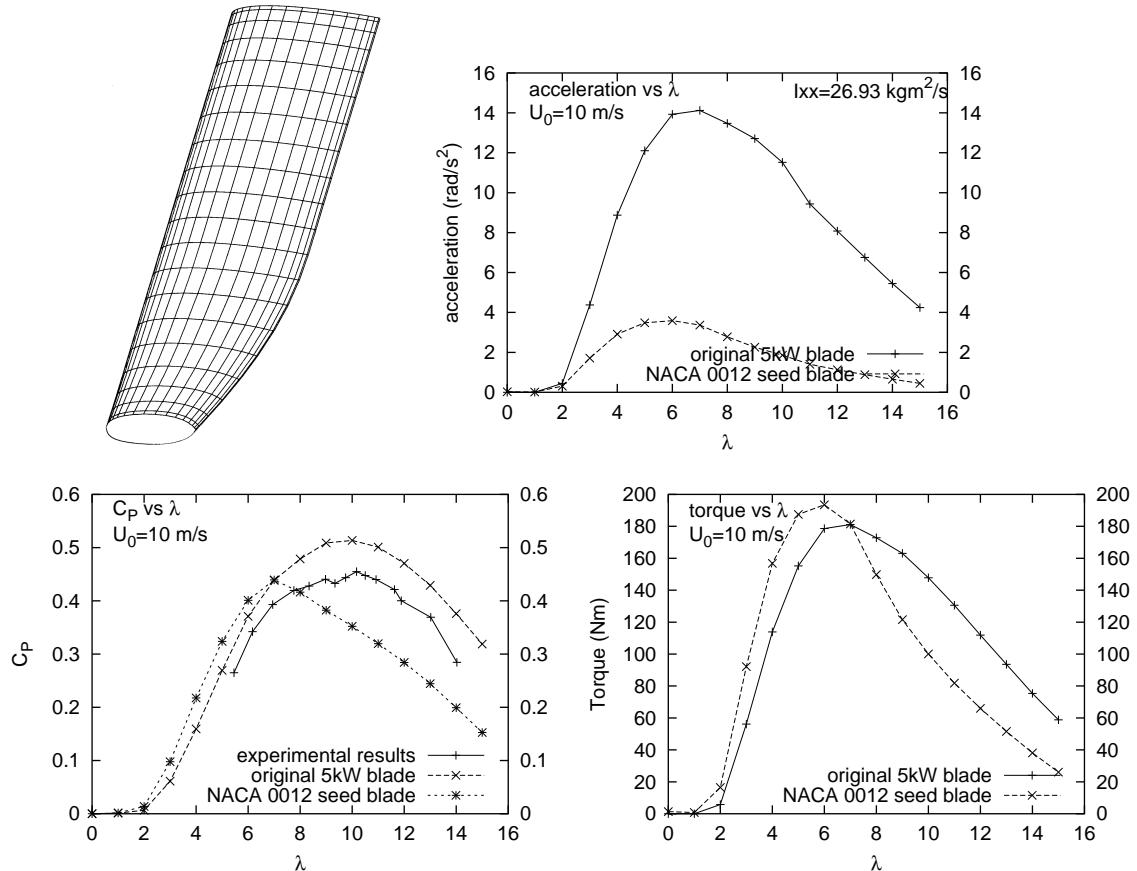


Figure 6.15: NACA 0012 seed blade - Geometry and performance results

The “seed” blade geometry described in the previous section was constructed by skinning a bi-cubic B-spline surface over four constant-chord, zero-twist NACA 0012 B-spline aerofoil sections arranged along the span of the blade. The standard “hub” aerofoil section was added at the blade root. As was the case with the 5 kW Newcastle comparison blade geometry, the seed blade surface was discretised into a  $40 \times 20$  mesh of quadrilateral surface panels. A wake model consisting of 5 turns was added, with each turn divided into 20 radial quadrilateral surface panels. A depiction of the meshed blade surface appears in Figure 6.15.

The panel method was then applied across the range of tip speed ratios spanning  $\lambda=0$  to  $\lambda = 10$ , with various performance measures recorded after each calculation. The most important of these performance measures (acceleration, power, and torque

over the range of studied tip speed ratios) are compared with those of the Newcastle 5 kW blade in Figure 6.15, with other relevant performance comparisons appearing in the following pages.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

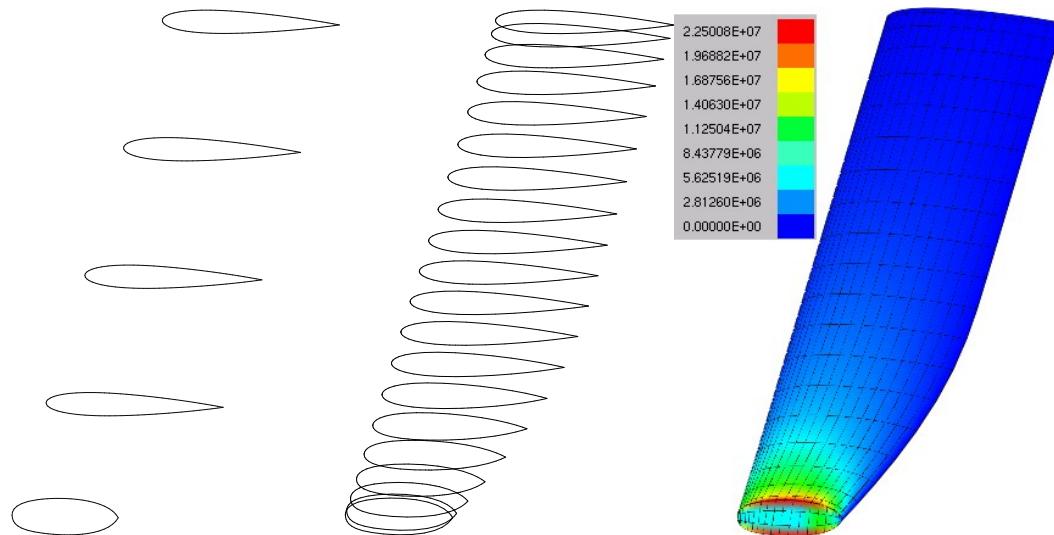
6.16.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.16.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
26.9333	0.3522	4163	1140	96.16	1.648	13.95

6.16.3: RESULTS



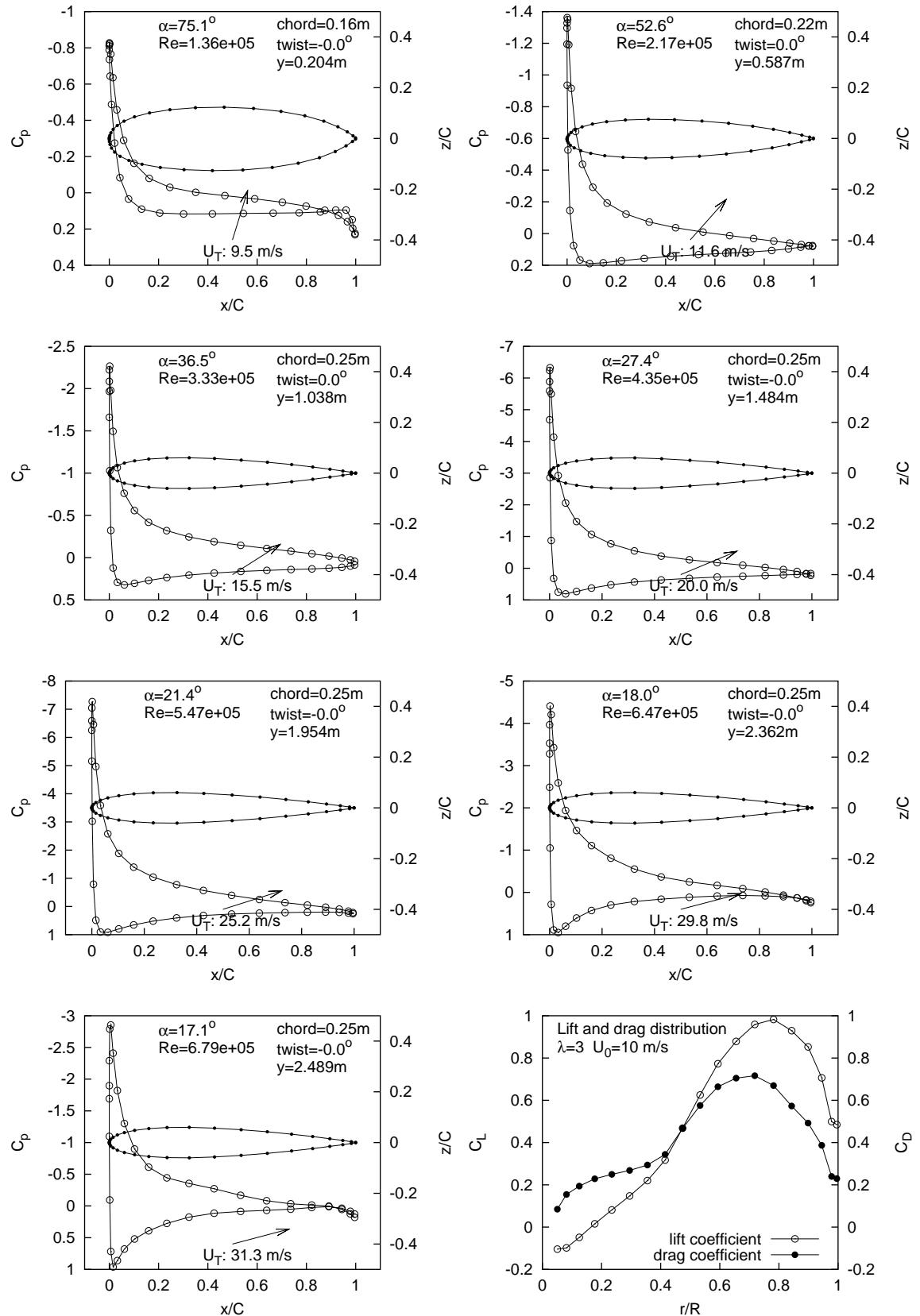
6.16.4: B-spline aerofoils

6.16.5: x-sections

6.16.6: von-Mises  
equiv.  
stress

Figure 6.16: NACA 0012 seed blade - Geometry and performance results (continued)



Figure 6.17: Pressure and force distributions along NACA 0012 seed blade @  $\lambda=3$

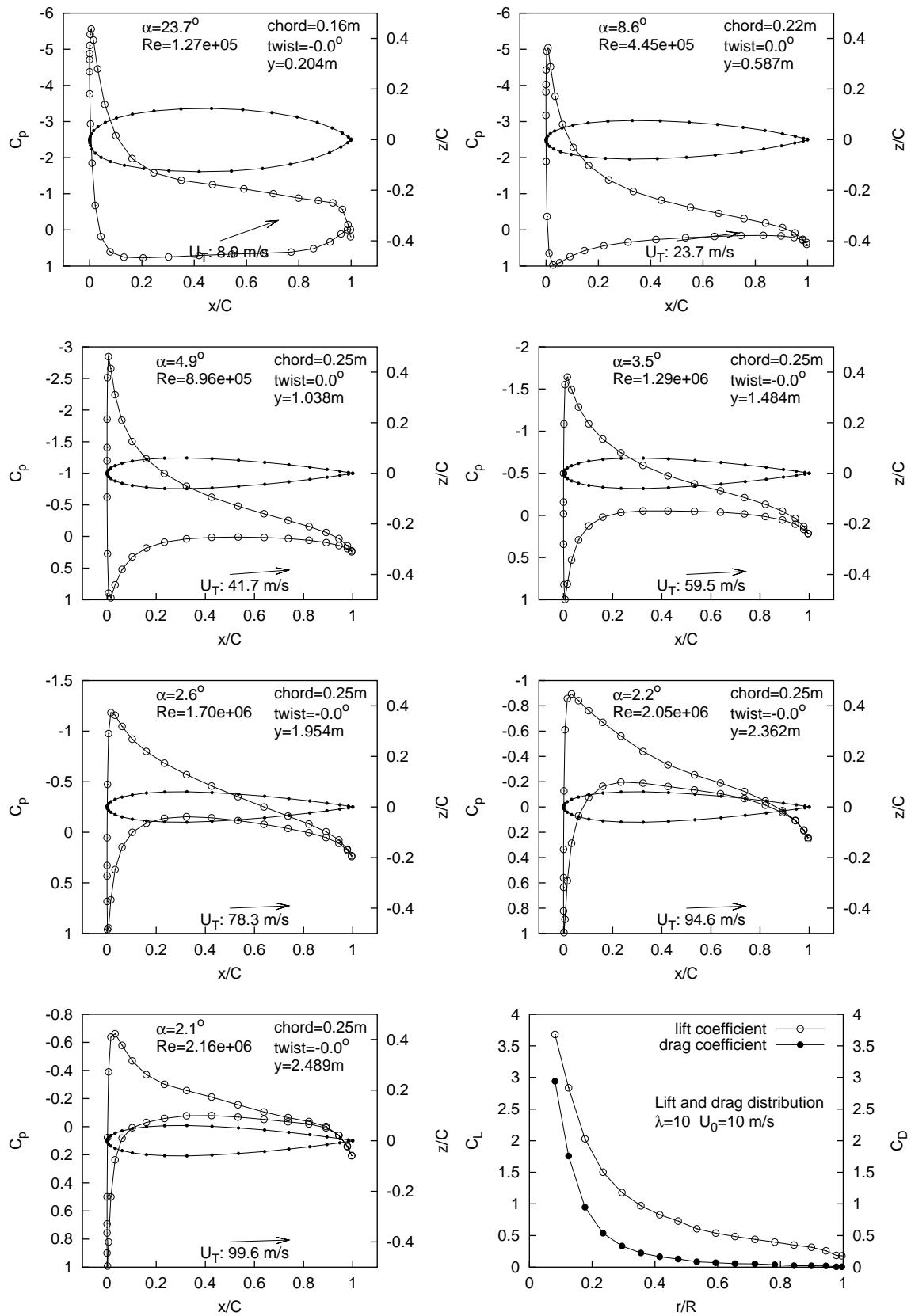


Figure 6.18: Pressure and force distributions along NACA 0012 seed blade @  $\lambda=10$

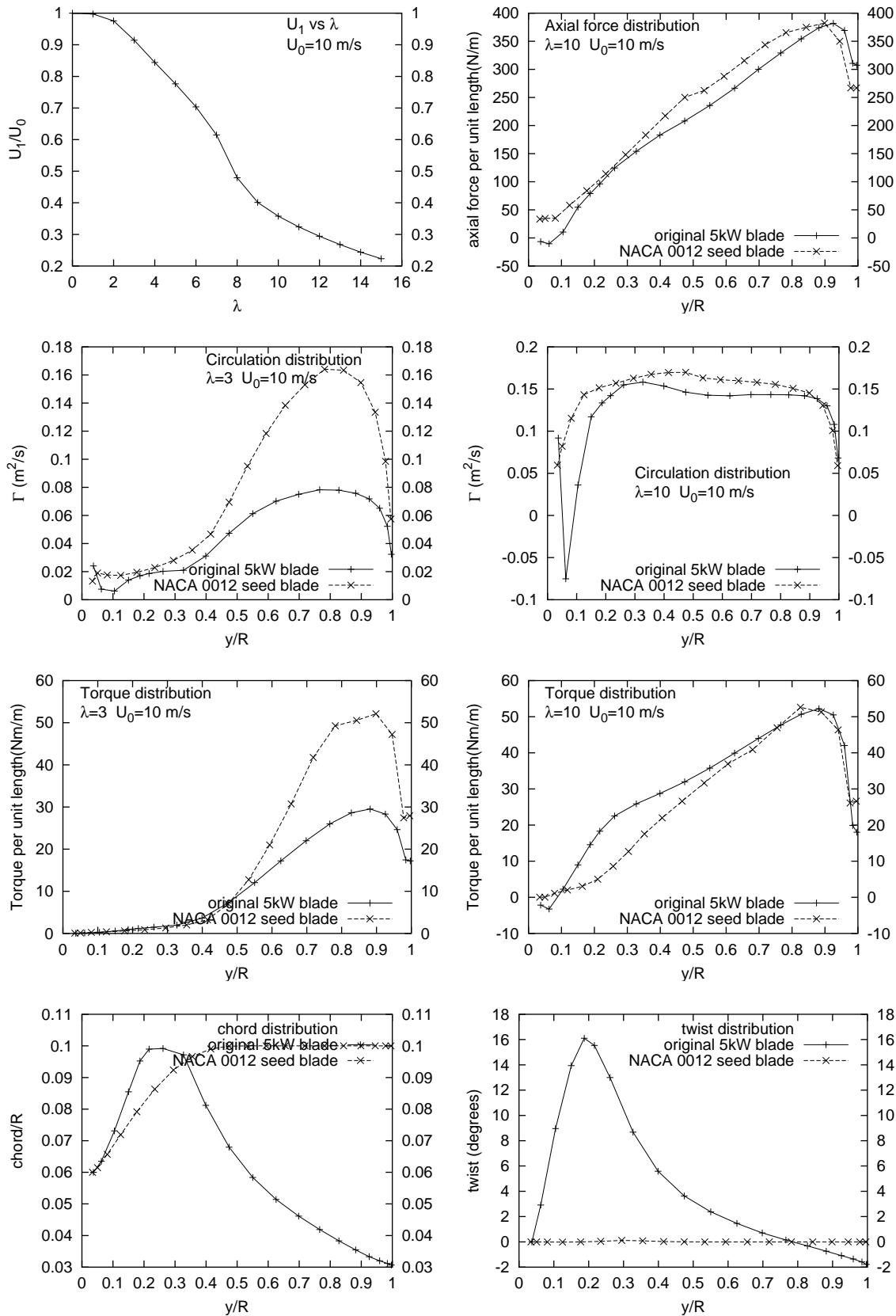


Figure 6.19: Performance results for NACA 0012 seed blade

### **6.7.3 Evolving a population of B-spline wind turbine blades**

To demonstrate the capabilities of this evolutionary optimisation technique, a population of 50 bi-cubic B-spline blade surfaces was generated. Each blade surface was based on the untapered, untwisted NACA 0012 seed blade geometry described above, with small Gaussian random perturbations added to the co-ordinates of its sectional B-spline aerofoils (including the y co-ordinate along the blade's span). Knot values in the common knot vector were similarly perturbed. To maintain a common root section, no perturbation of the "hub" aerofoil section's parameters was performed, and no subsequent "evolution" on these parameters was allowed to occur.

The multiobjective DE algorithm was applied to this primordial soup of blade designs at generation 0, and the population allowed to evolve for 2000 generations. The primary reason for stopping at generation 2000 was a time constraint on the current project - approximately seven days of computing time on a small PC-based cluster computer was required for the 100,000 objective function evaluations undertaken in the example run described here. To keep solution times to an absolute minimum, blade mesh density was chosen as  $30 \times 10$ , which is a rather coarse mesh. The justification for this choice, and the accompanying reduction in absolute solution accuracy, is that the *same* prediction performance degradation occurs during each blade evaluation. As a result, the blade performance figures accompanying the following results differ from evaluations of the same blades at higher mesh densities, with the most noticeable examples being the rather high  $C_P$  predictions, in some cases in excess of the Betz limit (Figure (6.20.1)).

As described previously, two separate blade populations were stored: the evolving blade population; and a Pareto optimal blade set. The Pareto set contains the population of "optimal" trade-off designs and is described in detail in the following section. Various performance figures for the evolving blade population appear in Figures 6.20.1 to 6.20.4. Unlike the blades in the Pareto set, those in the evolving population were permitted to remain "unviable" to maintain genomic diversity and, as described previously, were only discarded if they were superseded by a better comparison vector (either through a better viability measure or via the Pareto selection operator). Since one of the mechanisms for punishing unviable designs was via a degradation in their performance "fitness" values, this had the effect of distorting the "true" shape of the

evolving performance figure visualisations.

Nevertheless, interesting information about how the optimisation progressed can still be gleaned from Figures 6.20.1 to 6.20.4. On each plot, “best”, “mean” and “maximum” values for each measure across the evolving population were recorded (with “maximum” replaced by “minimum” for the evolving  $I_{xx}$  data). The “best” performance measure reflects the value of the single member of the evolving population which was judged to be the best member, via the following (arbitrary) aggregate fitness criterion:

$$bias = \begin{cases} 1.0 & \text{for } mean\_C_P < 0.45 \\ 3.0 & \text{otherwise} \end{cases}$$

$$member[i].fitness = bias \left( \frac{member[i].acceleration}{max\_acceleration} \right) + 5.0 \left( \frac{member[i].C_P}{max\_C_P} \right) \quad (6.10)$$

with the member with the highest aggregate fitness score tagged the “best”. The criterion simply biases in favour of better  $C_P$  values at the start of the optimisation, with slightly less bias as the mean  $C_P$  value across the evolving population exceeds the arbitrarily-chosen threshold of 0.45. Tagging one member of the population in such a way serves no purpose other than to provide another measure of how the population is evolving. It certainly plays no part in evolutionary activity itself.

The most important features evident on these evolving performance plots are the initial burst of evolutionary activity before generation 1000, where offspring consistently outperform their parents and are accepted into the evolving population, followed by a plateau where improvement takes place relatively slowly, all the way up to the completion of the run. The initial burst of activity involves competition between a few good “species” of blades for footholds in the limited space available via the removal of the least fit (or viable) designs. This “coarse” evolutionary activity gradually gives way to “refinement” - a period during which pockets of very dissimilar “species families” inter-breed, resulting in a large increase in the number of sub-species in the population.

A most striking strategy for improving acceleration performance (Figure 6.20.2) is shown to be a rapid decrease in the blade’s second moment of inertia (Figure 6.20.4). Although this necessarily results in a decrease in low- $\lambda$  torque (Figure 6.20.3), the degraded torques are still favourably comparable to those offered by the existing 5 kW Newcastle turbine blade design.

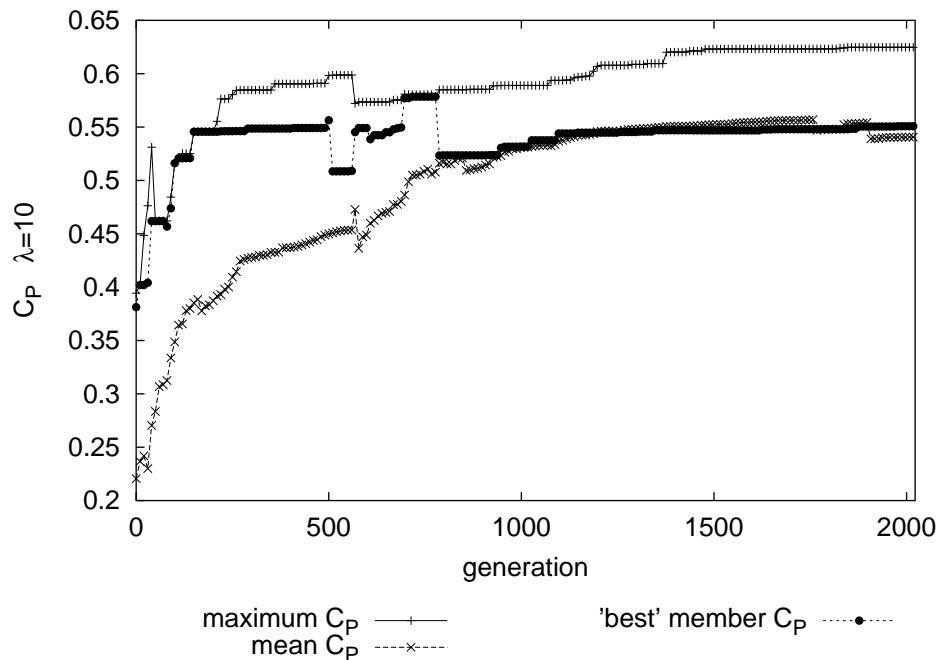
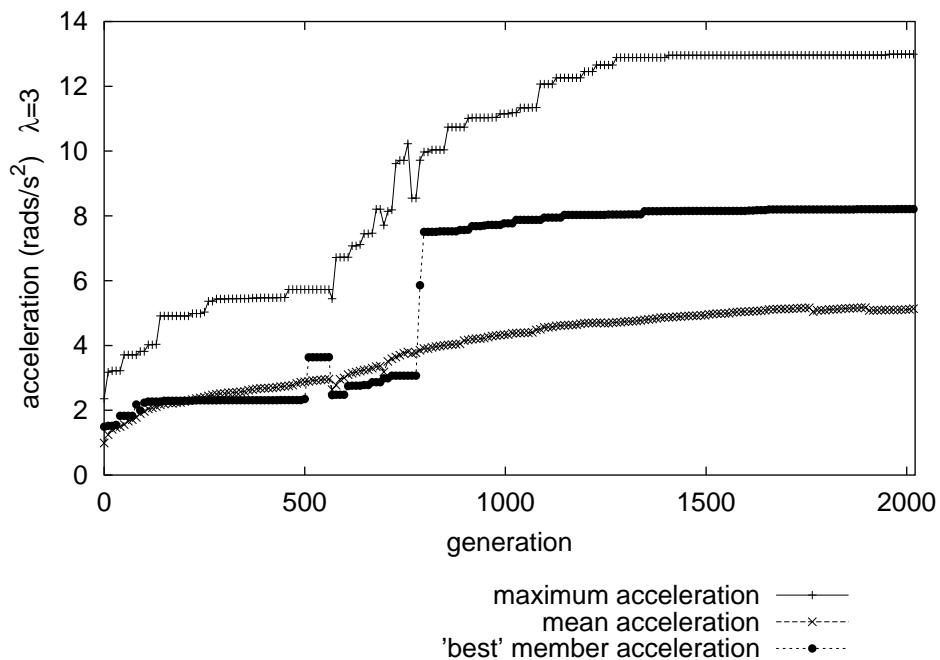
6.20.1: Evolving  $C_p$  performance ( $\lambda=10$ ) over 2000 generations6.20.2: Evolving acceleration performance ( $\lambda=3$ ) over 2000 generations

Figure 6.20: Performance figures for the evolving population over 2000 generations

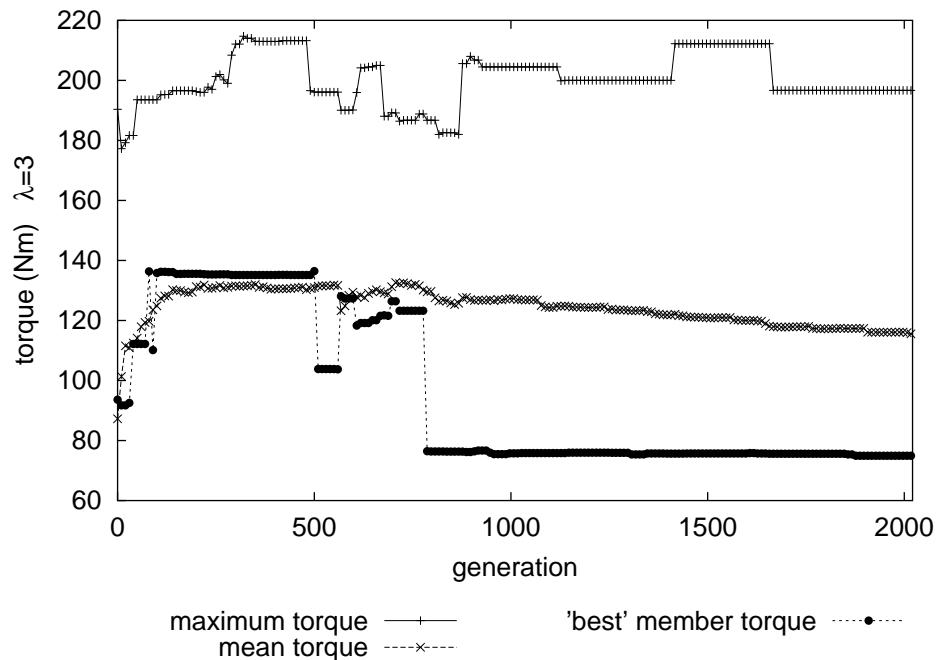
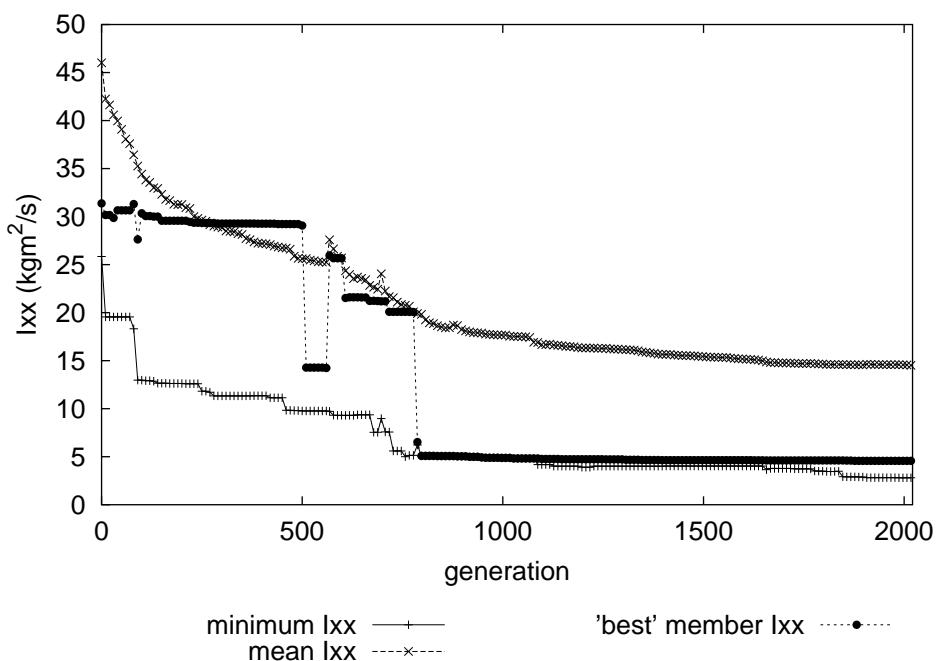
6.20.3: Evolving torque performance ( $\lambda=3$ ) over 2000 generations6.20.4: Evolving  $I_{xx}$  over 2000 generations

Figure 6.20: Performance figures for the evolving population over 2000 generations

### 6.7.4 A Pareto optimal set of wind turbine blades

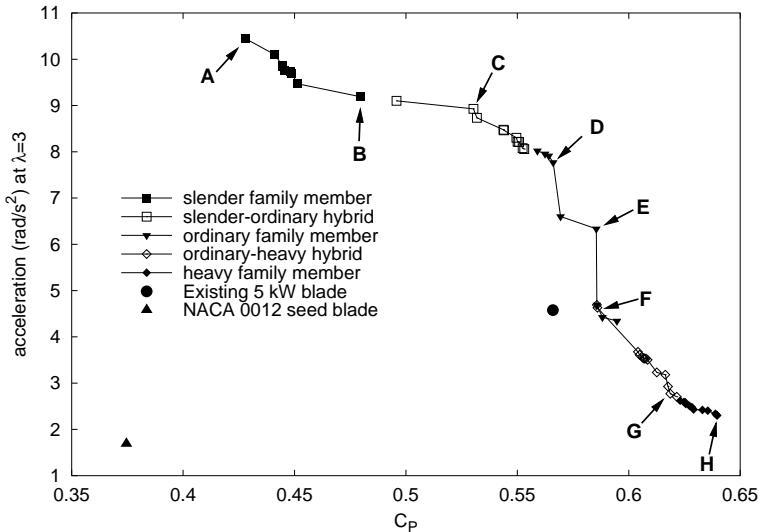


Figure 6.21: Pareto front after 2000 generations

Figure 6.21 shows the Pareto optimal front after 2000 generations (100,000 objective function evaluations, with a population size of 50). As described in the previous section, the optimisation progressed from the vicinity of the seed NACA 0012 blade at the bottom left of Figure 6.21. By generation 800, the Pareto front had passed through the point marked by the performance of the existing 5 kW Newcastle turbine blade design, and at the termination of the run had stabilised to the front shown. For approximately half of the optimisation run - 1000 generations - the Pareto set was relatively sparse, and contained less than twenty “unique” blade designs. This is an expected occurrence: as each new Pareto set member was added it replaced all of the current members it dominated. This meant that large numbers of blades, in fact whole species, faced the risk of being eradicated by a new dominant design after each evaluation of every generation. The stabilisation of the Pareto set during the second-half of the optimisation run suggests that the “true” Pareto front was being approached at generation 2000 (at least over the current fitness landscape defined by the “low”-resolution objective functions employed). Evolutionary activity at this later stage was characterised by less “forward movement” - large advancements in objective function values - and more “side-ways movement”, or inter-species activity, which filled in the gaps between

disparate designs and produced a variety of “sub-species”.

The Pareto front in Figure 6.21 contains 54 “unique” blade designs divided into a number of categories, with each blade belonging to a “*species*” or sub-species of wind turbine blade design. A species was loosely defined as containing “blades that share a major physical attribute”. Three distinct species of small wind turbine blade were identified:

- **The “Slender” family:** Slender blades are, as the name suggests, very light-weight blades with a small, almost-constant chord distribution along the span. Their performance is characterised by poor power production across the whole range of studied tip speed ratios, coupled with exceptional starting performance due to their tiny second-moments of inertia and surprisingly good low- $\lambda$  torque. (Figure 6.22).
- **The “Ordinary” family:** Ordinary blades have, in general, a decreasing taper and twist from root to tip. These are characteristics shared with current commercial state-of-the-art small wind turbine blade designs. Their performance is characterised by “good” power production and “good” low- $\lambda$  acceleration performance - desirable properties which have seen the design philosophy dominate the small wind turbine market. (Figure 6.24).
- **The “Heavy” family:** Heavy blades have large, almost-constant chords along their spans. Their outstanding torque performance across the entire range of studied tip speed ratios makes them very good power producers, but is critically undermined by the huge second-moments of inertia inherent in bulky designs, a flaw which results in poor starting performance. (Figure 6.26).

Between these categories lie two sub-species: **Slender-Ordinary hybrids** and **Ordinary-Heavy hybrids**. These hybrid families “fill the gaps” between the distinct species - both geometrically and in terms of performance. The Slender-Ordinary family appears in Figure 6.23 and the Ordinary-Heavy family appears in Figure 6.25. Each blade was numbered from 1 to 54 and, via the definition of Pareto dominance, is thus ranked on the basis of monotonically- increasing  $C_P$  performance and monotonically decreasing  $\dot{\Omega}$  performance. It should be emphasised that this order is dependent on the “low”-resolution aerodynamic evaluation technique used during the optimisation

- magnitudes of performance figures will be different in subsequent higher-resolution evaluations. This is especially evident at the higher-power design conditions, and it should be understood that the very high  $C_P$  figures, some exceeding the Betz limit, are erroneous artifacts of the low-resolution evaluation technique. In fact, after consideration of the *quite low* higher-resolution  $C_P$  figures of the Ordinary-Heavy hybrid family it becomes clear that, had the optimisation run included a  $40 \times 20 \times 5 \times 20$  resolution aerodynamic objective function (at an unacceptable time-cost to the current project), many members of the “Ordinary-Heavy” sub-species *as shown here* would not even have evolved. This does not destroy the instructive-value of the “heavier” blade designs, however, and certainly does not negate the validity of the optimisation run: Slender, Slender-Ordinary and Ordinary blade designs all evaluate very well under subsequent higher-resolution evaluation, and it is perhaps fortunate that these are the very blade designs that address the problem statement of the current project. Even the “Heavy” blade species (which consists of blades which share remarkably similar geometries) evaluates “adequately” under the higher-resolution mesh, but it should be noted that, again, these Heavy blades *as presented here* would not exist if evolution took place with a higher-resolution aerodynamic objective function, simply because their geometries elicit erroneous pressure-distribution predictions from the panel solver with the higher-resolution mesh and would be tagged as “unviable” by the Differential Evolution objective functions described in earlier sections. In any case, the finer panel mesh applied to nearly all of the following Pareto optimal blade designs offered performance predictions indicating that these blades performed *more* favourably against the existing 5 kW baseline design - a result which attests to the power of both the B-spline surface representation scheme and the Differential Evolution optimisation technique.

Eight interesting blade designs were chosen from this Pareto set to demonstrate their geometry and performance. They appear in Figure 6.21 labelled alphabetically from “A” to “H”, and are compared against the existing 5 kW Newcastle blade design in more detail in the following sections.

Each B-spline blade was evaluated by first generating it from its evolved object vector. It was then discretised into a  $40 \times 20$  quadrilateral surface panel mesh. A wake mesh with 5 turns and 20 radial panels per turn was added. The panel method was applied across the range of tip speed ratios spanning from  $\lambda=0$  to  $\lambda=15$ , and a variety of performance figures stored.

### 6.7.5 Pareto set: “Slender” blade family

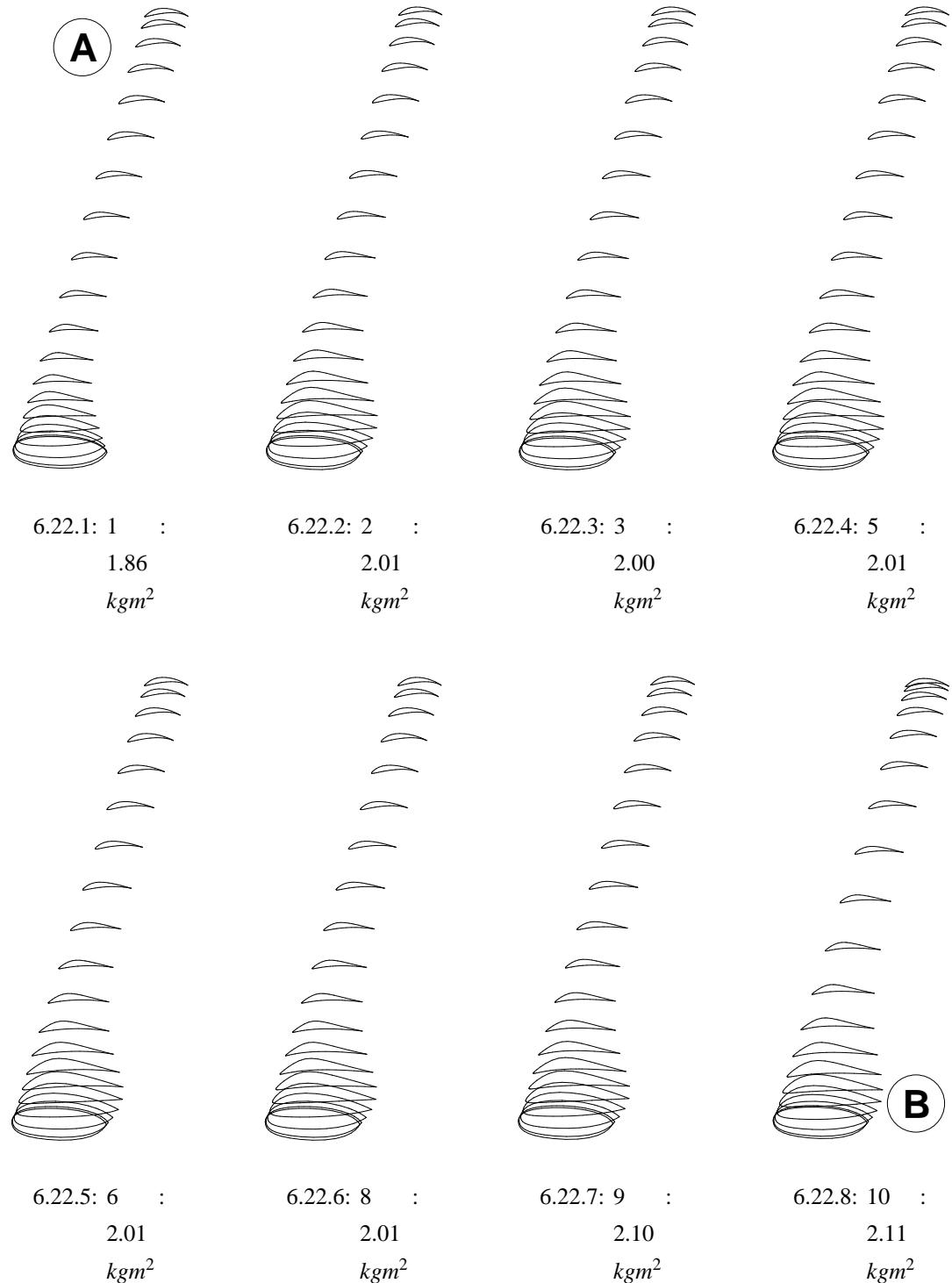


Figure 6.22: Span-wise cross-sections through the “Slender” blade family

### 6.7.6 Pareto set: Slender-Ordinary hybrids

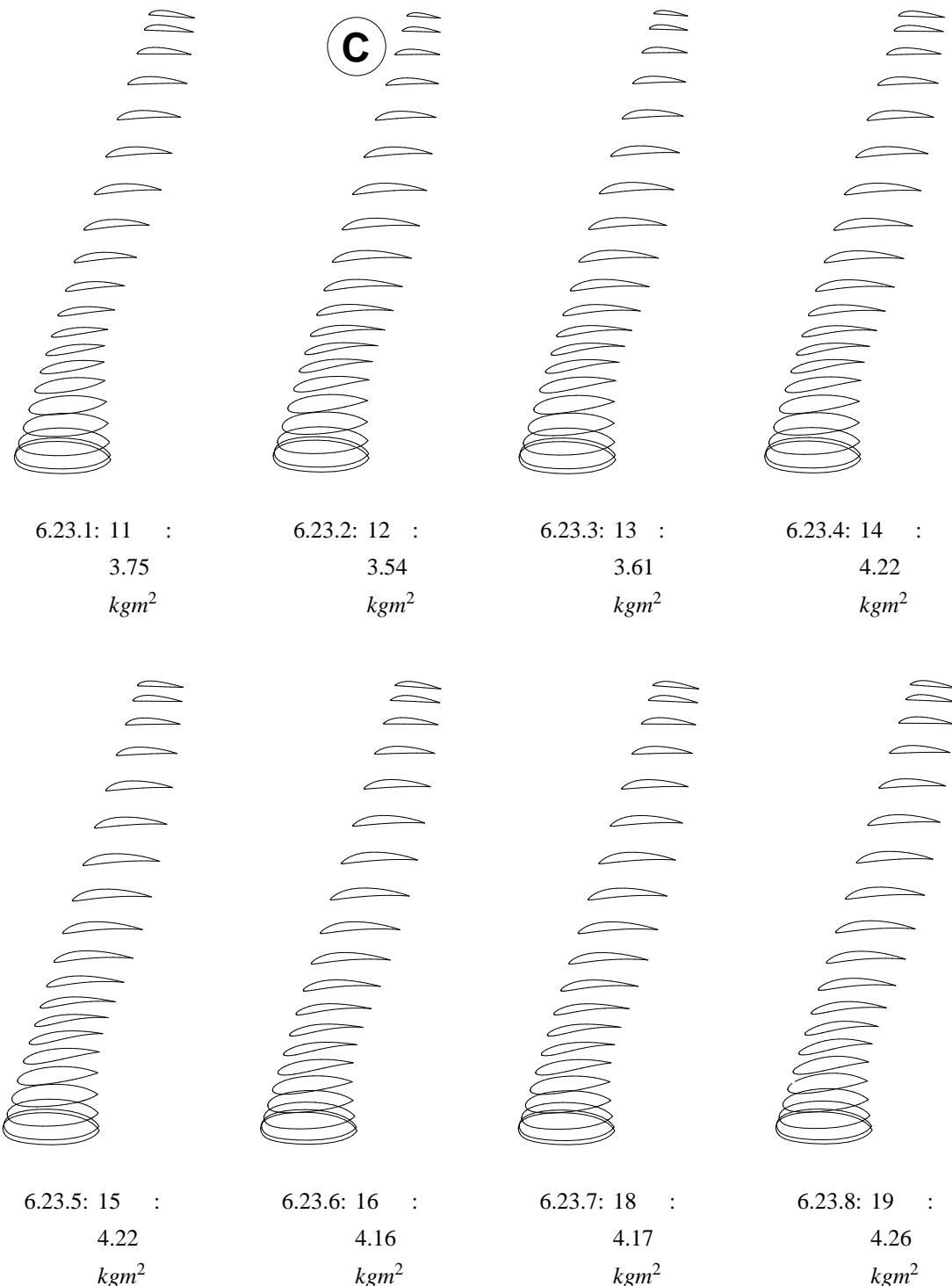


Figure 6.23: Span-wise cross-sections through the Slender-Ordinary hybrid series

### 6.7.7 Pareto set: “Ordinary” blade family

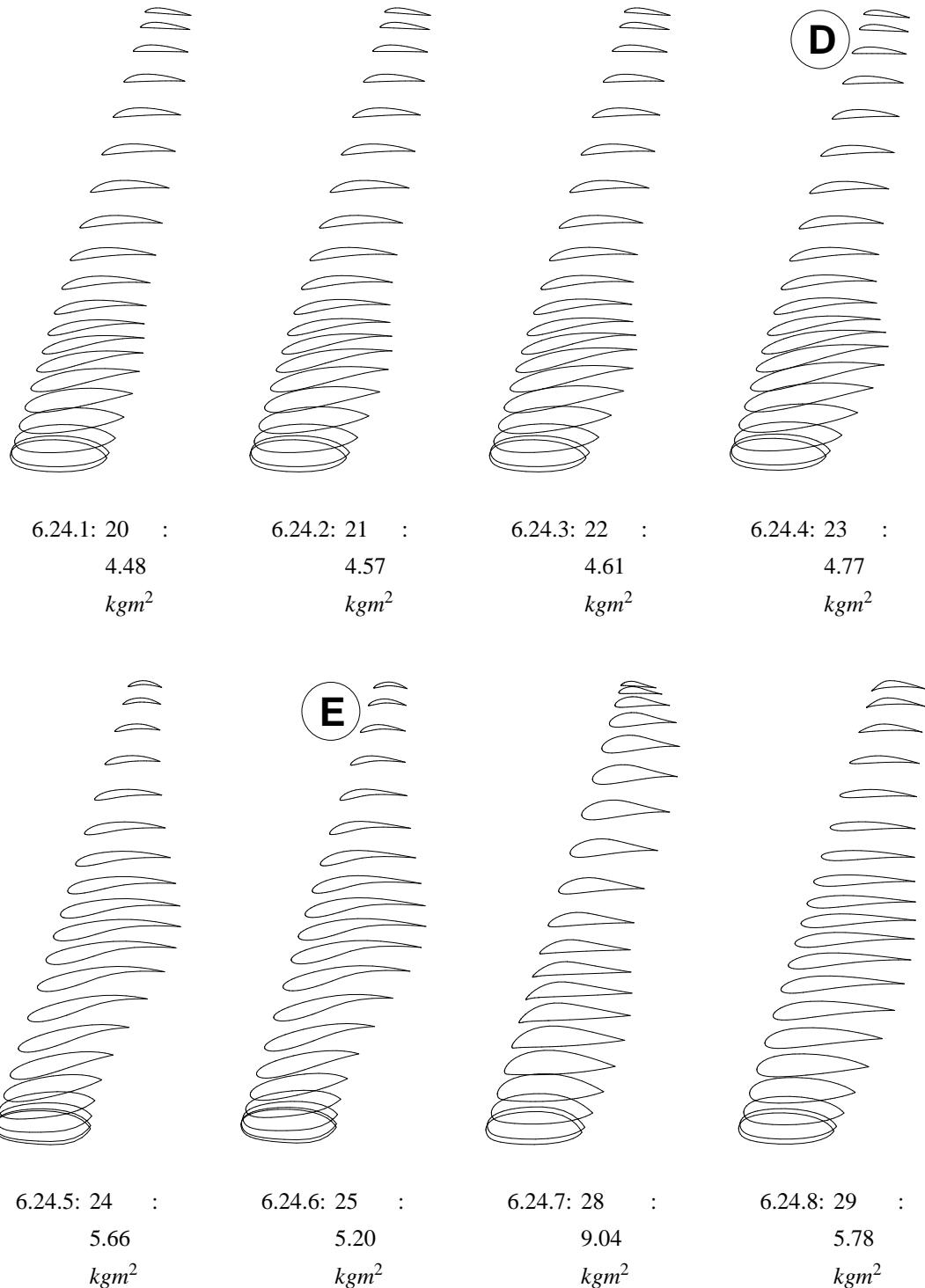
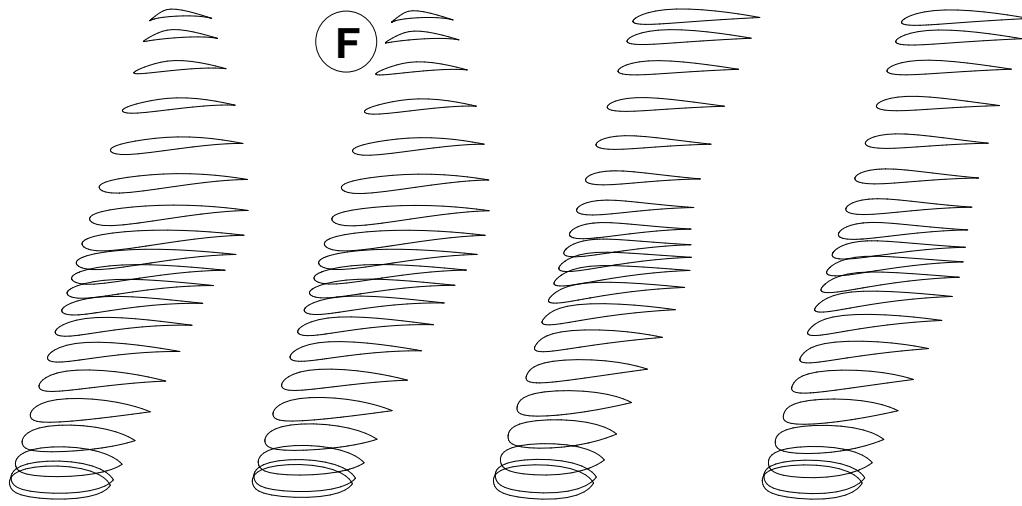
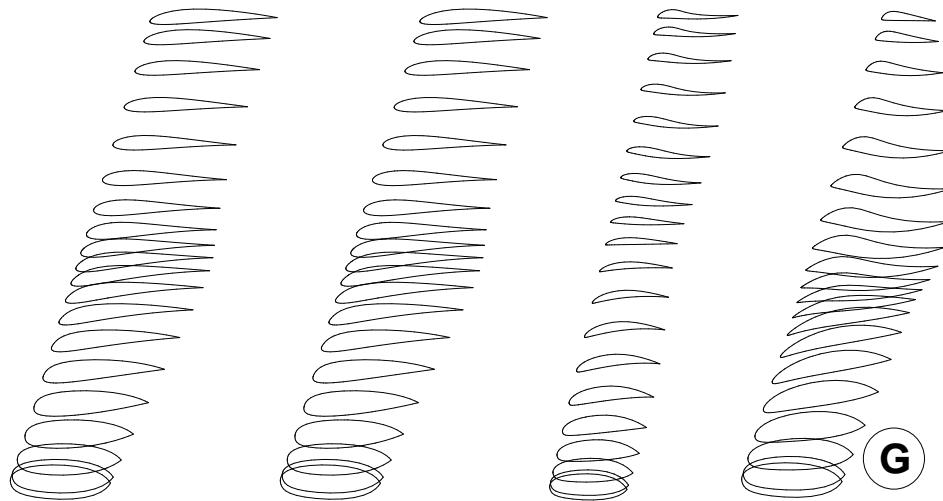


Figure 6.24: Span-wise cross-sections through the “Ordinary” blade family

### 6.7.8 Pareto set: Ordinary-Heavy hybrids



6.25.1: 26 :	6.25.2: 27 :	6.25.3: 30 : 12.84	6.25.4: 32 : 13.75
10.79 $kgm^2$	10.93 $kgm^2$	$kgm^2$	$kgm^2$



6.25.5: 34 : 13.80	6.25.6: 36 : 13.80	6.25.7: 38 : 8.30	6.25.8: 40 :
$kgm^2$	$kgm^2$	$kgm^2$	11.10 $kgm^2$

Figure 6.25: Span-wise cross-sections through the Ordinary-Heavy hybrid series

### 6.7.9 Pareto set: “Heavy” blade family

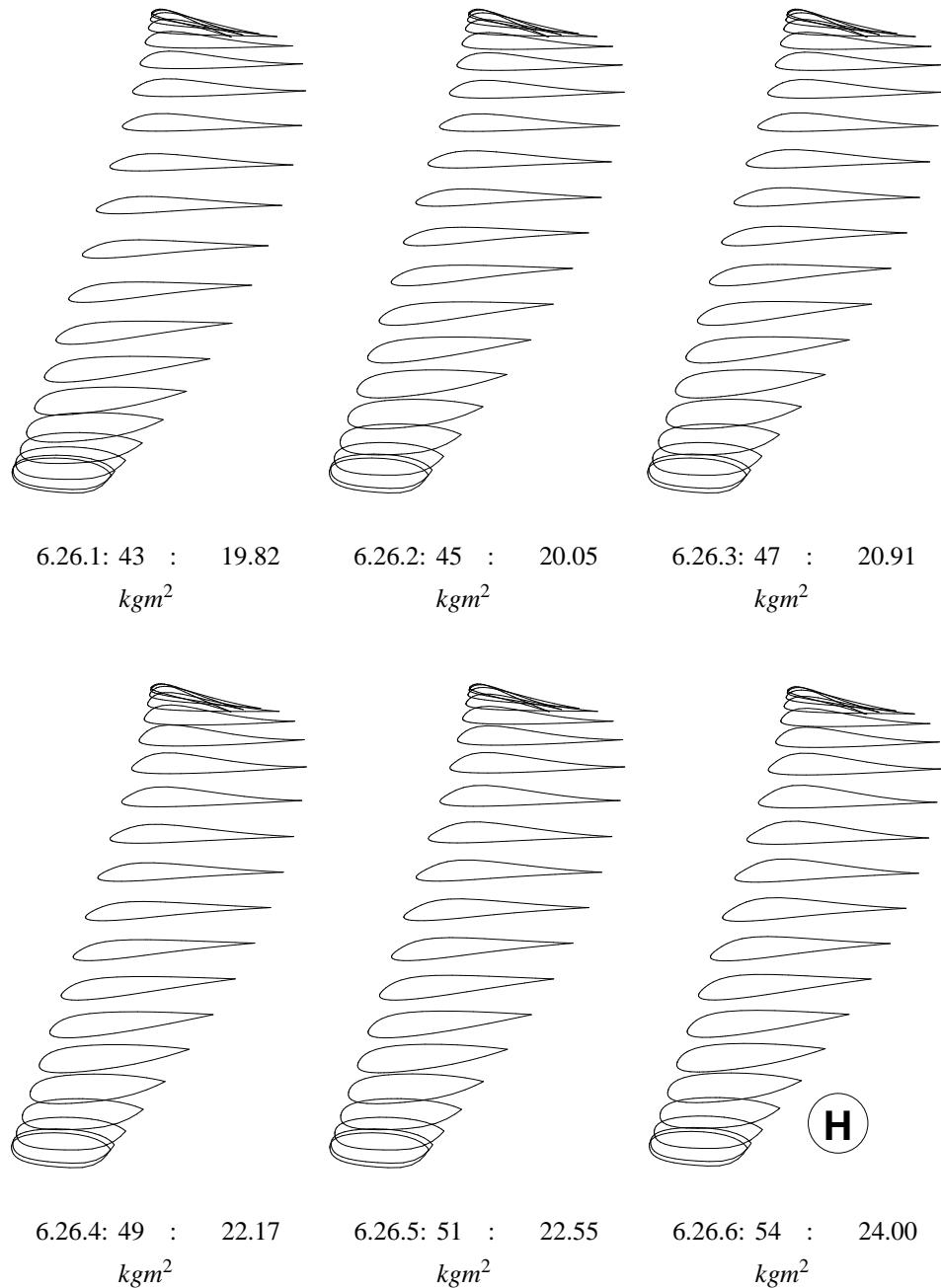


Figure 6.26: Span-wise cross-sections through the “Heavy” blade family

### 6.7.10 Pareto set: member [A]

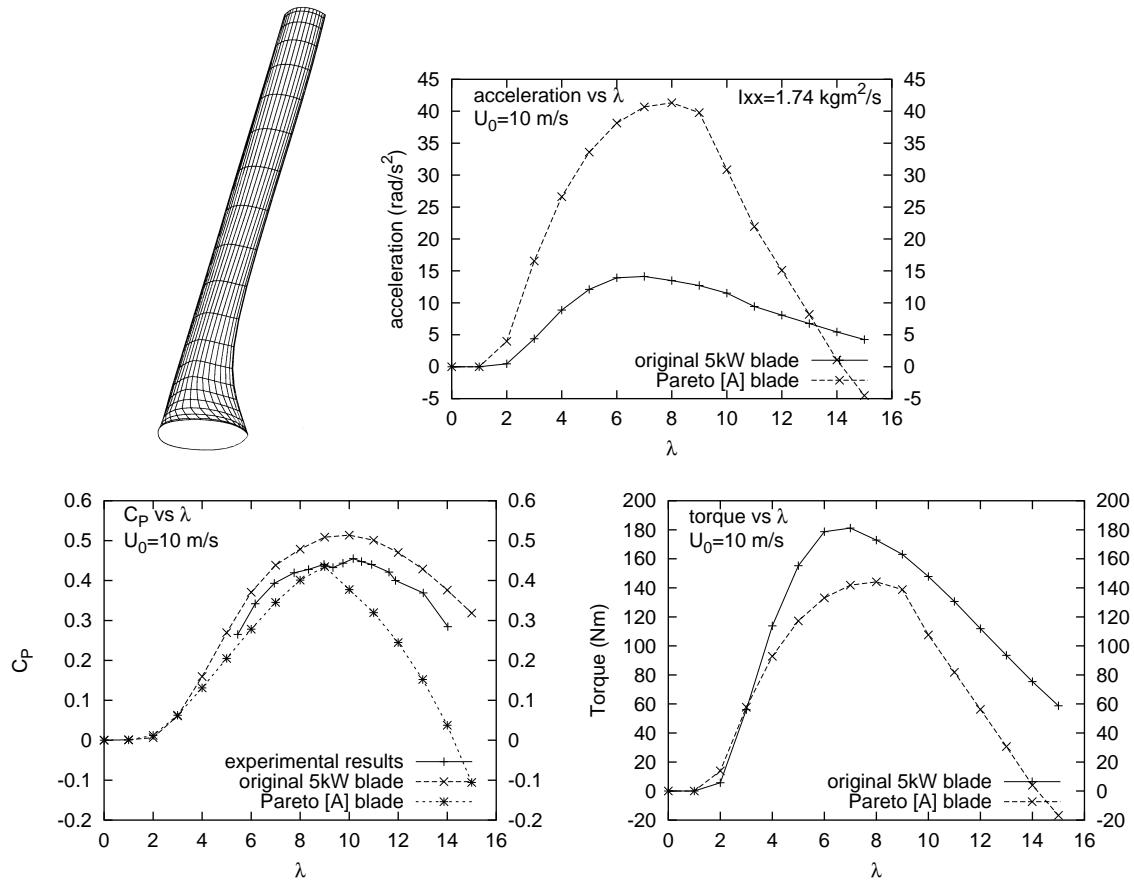


Figure 6.27: Pareto [A] blade - Geometry and performance results

Figure 6.27 shows the meshed geometry of the [A] member of the current Pareto set, together with important performance results. Member [A] is an extreme example of the first of the three “families” of blade designs highlighted by the current study: the *slender blade family*. Its performance is characterised by rather poor power production across the entire range of tip speed ratios, together with outstanding acceleration performance - a result of the *very low second-moment of inertia* of this blade, together with surprisingly good low- $\lambda$  torque. Figure 6.28 shows the blade is constructed from cambered aerofoils, with camber increasing from root to tip. Also evident are the large stress concentrations near the root at high- $\lambda$  - a problem caused by large thrust loads (large root moments) and slender root geometry. Figures 6.29 and 6.30 show the sectional pressure and span-wise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , respectively. The

most notable feature of figure 6.31 is that tip-wise torques at low- $\lambda$  are quite high - an obvious contributor to good starting performance. The chord and twist distributions displayed in Figure 6.31 show that the blade is very slender indeed, with an almost constant chord of around 70 millimetres (for a 2.5 metre blade) and an almost constant twist of around +3° along its span.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.28.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.28.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
1.7442	0.3776	4463	924	61.75	16.558	1.50

6.28.3: RESULTS

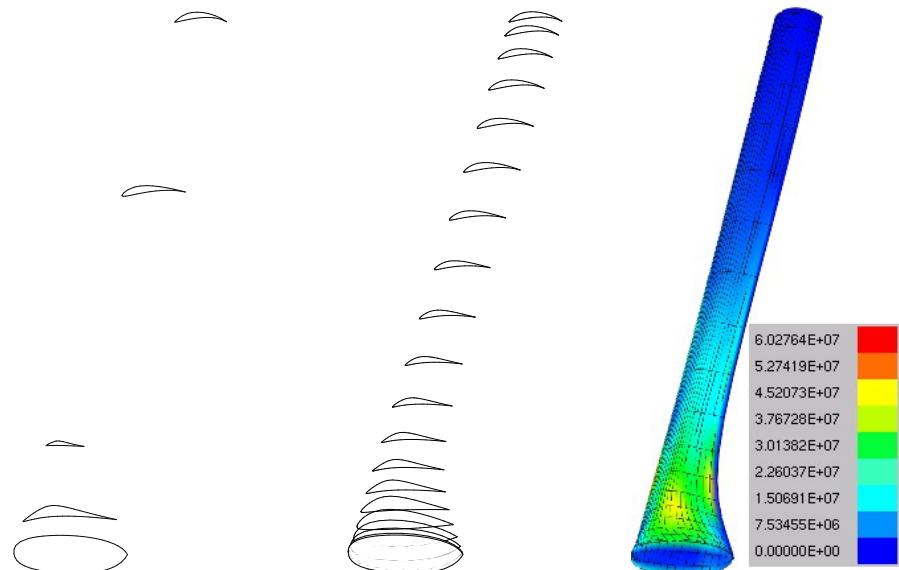
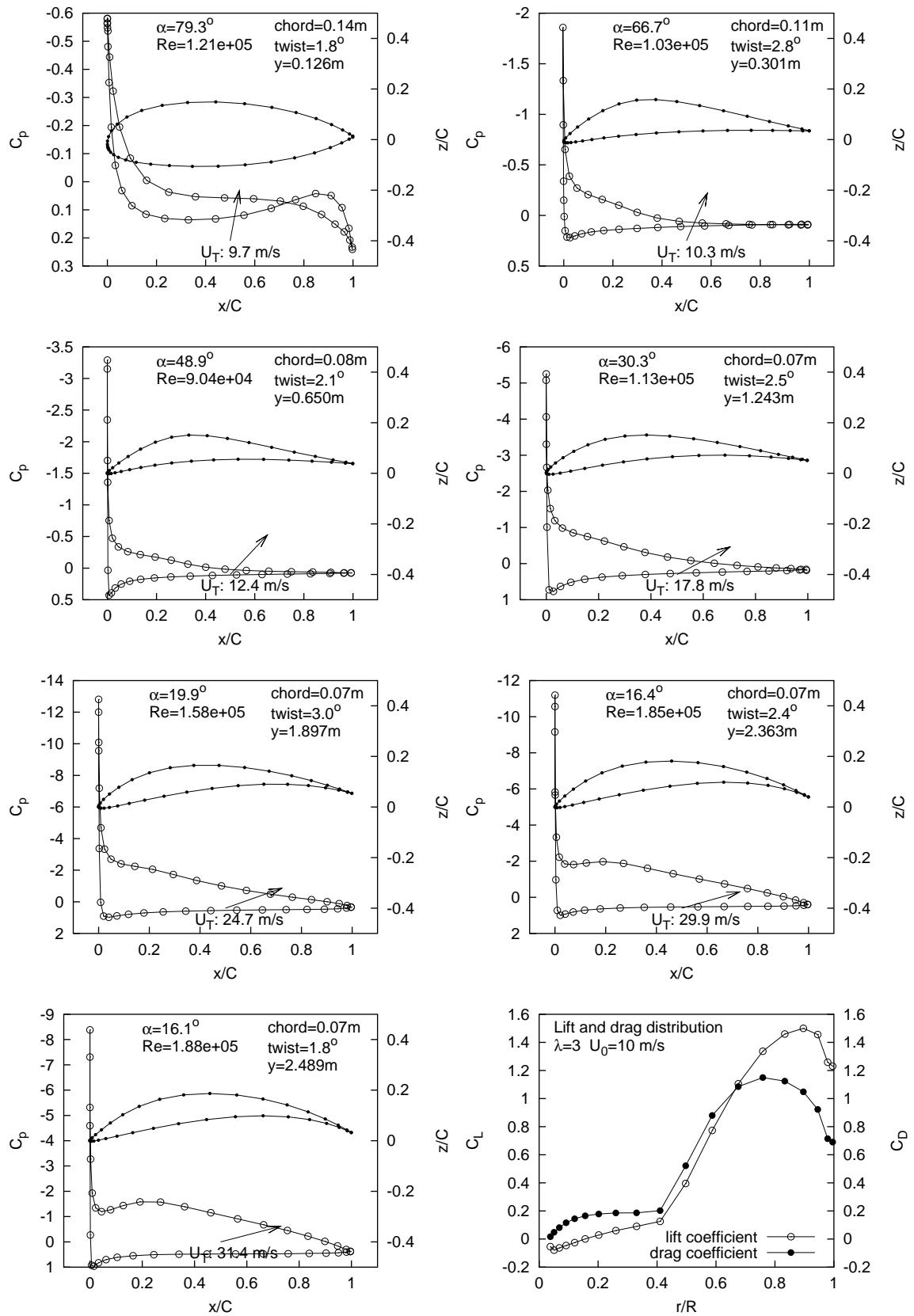
6.28.4: B-spline  
aerofoils6.28.5: Radial  
x-sections6.28.6: von-Mises  
equiv. stress

Figure 6.28: Pareto [A] blade - Geometry and performance results (continued)



Figure 6.29: Pressure and force distributions along Pareto [A] blade @  $\lambda=3$

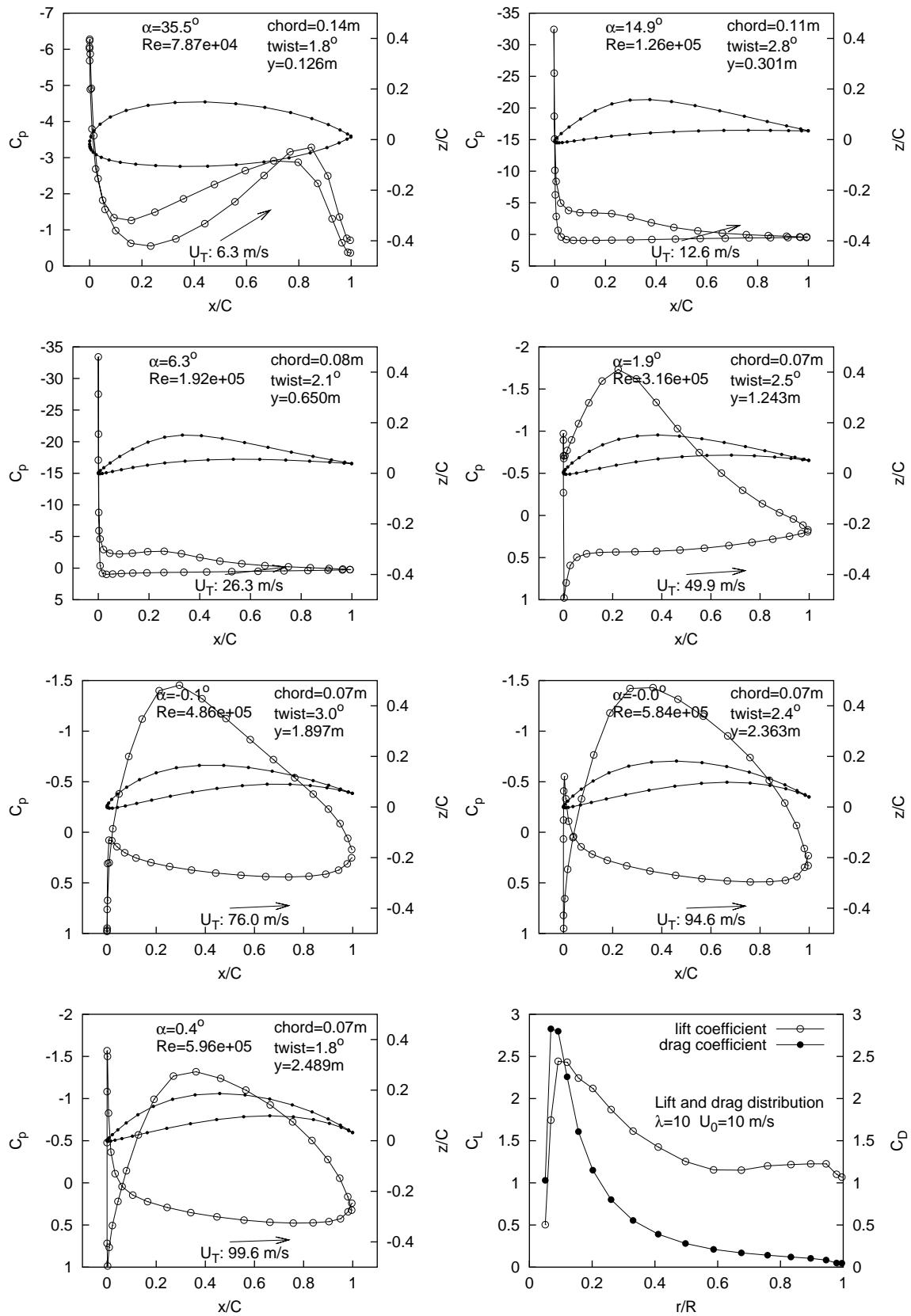


Figure 6.30: Pressure and force distributions along Pareto [A] blade @  $\lambda=10$

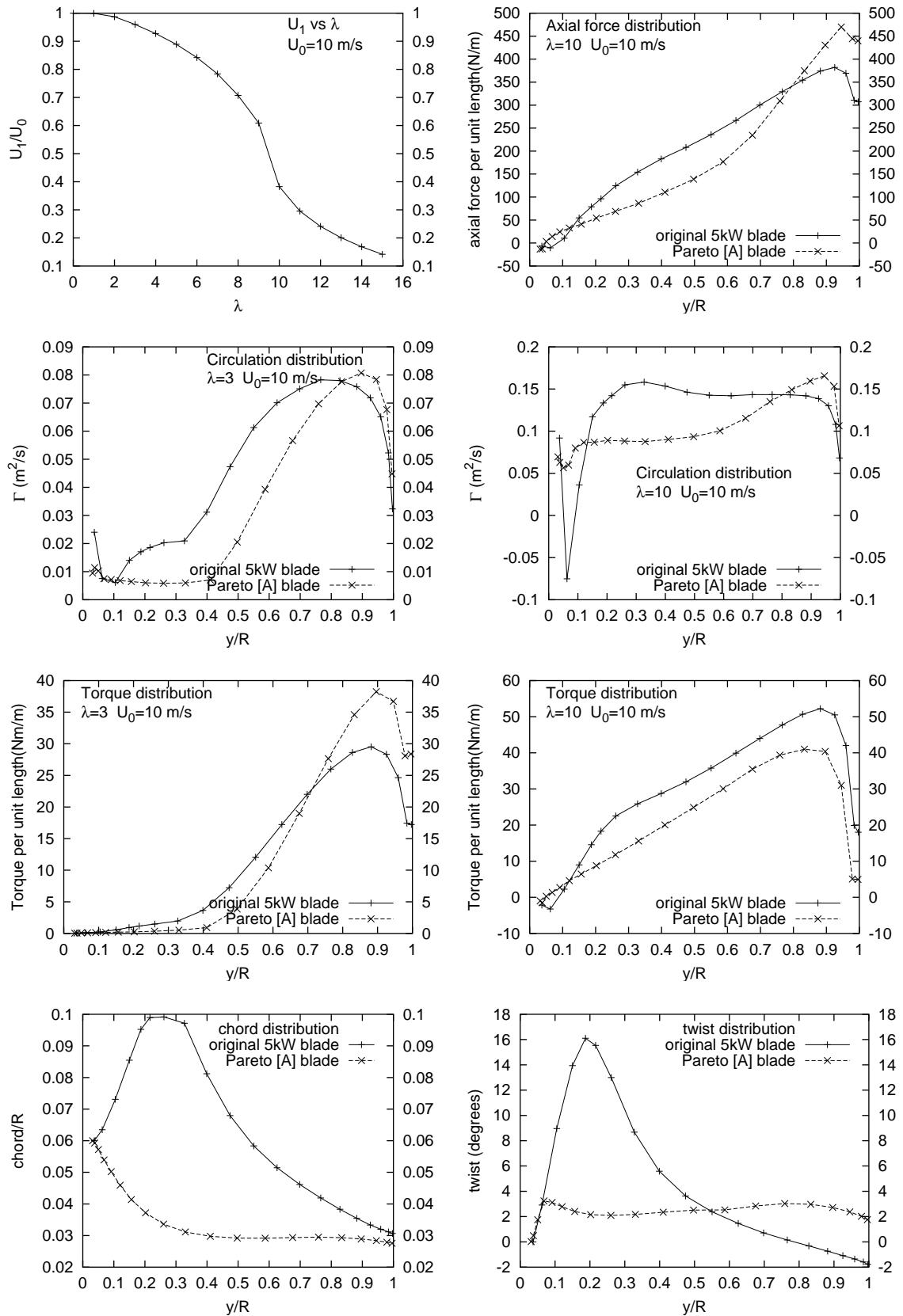


Figure 6.31: Performance results for Pareto [A] blade

### 6.7.11 Pareto set: member [B]

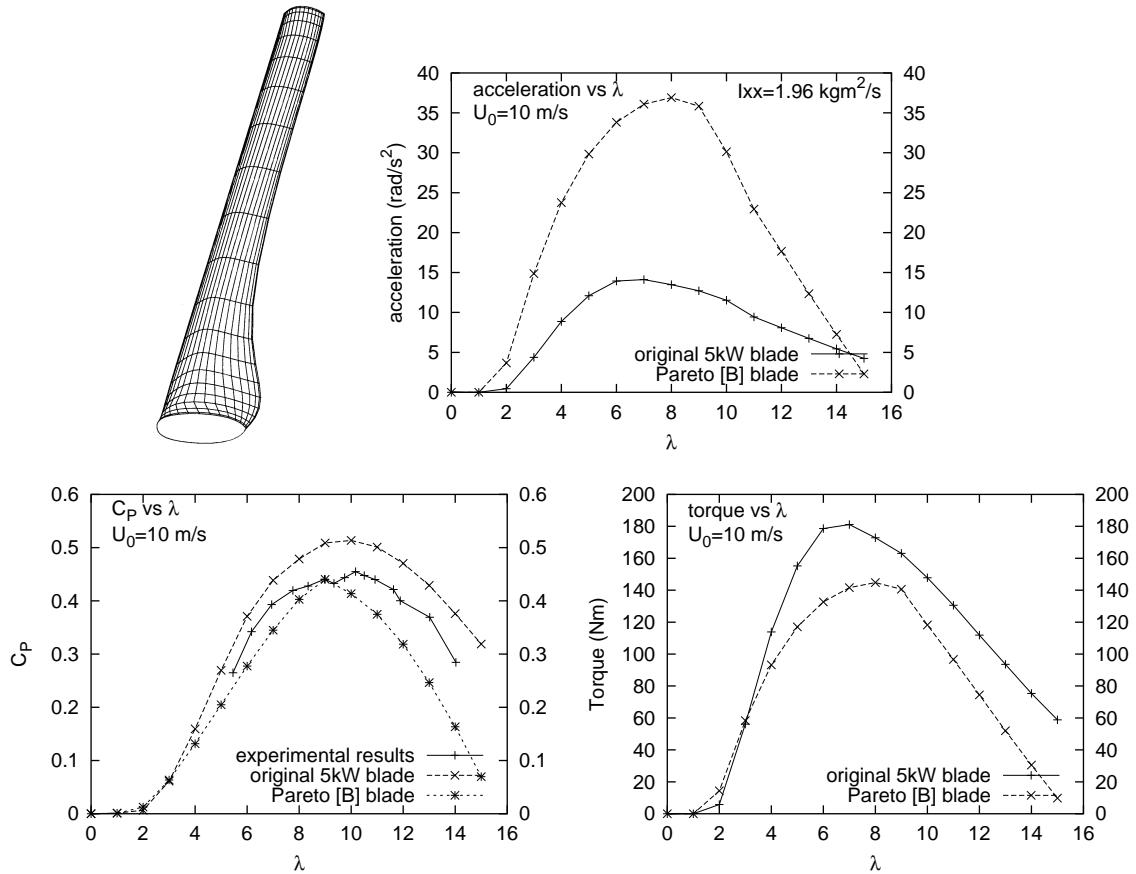


Figure 6.32: Pareto [B] blade - Geometry and performance results

Figure 6.32 shows the meshed geometry of the [B] member of the current Pareto set, together with important performance results. Member [B] is an example of the dominant design strategy employed by the *slender blade family* - a larger chord at the hub than the [A] member (around 160 mm) but otherwise similarly very slender for remainder of the span. This addition considerably improves its  $\lambda=10$  torque/power performance, but at the expense of slightly increased second-moment of inertia. Other performance results are shown in Figure 6.36 and are similar to those of the [A] member. Figures 6.34 and 6.35 show the sectional pressure and span-wise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , respectively.

B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

## 6.33.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

## 6.33.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
1.9613	0.4135	4887	923	62.34	14.798	2.10

## 6.33.3: RESULTS

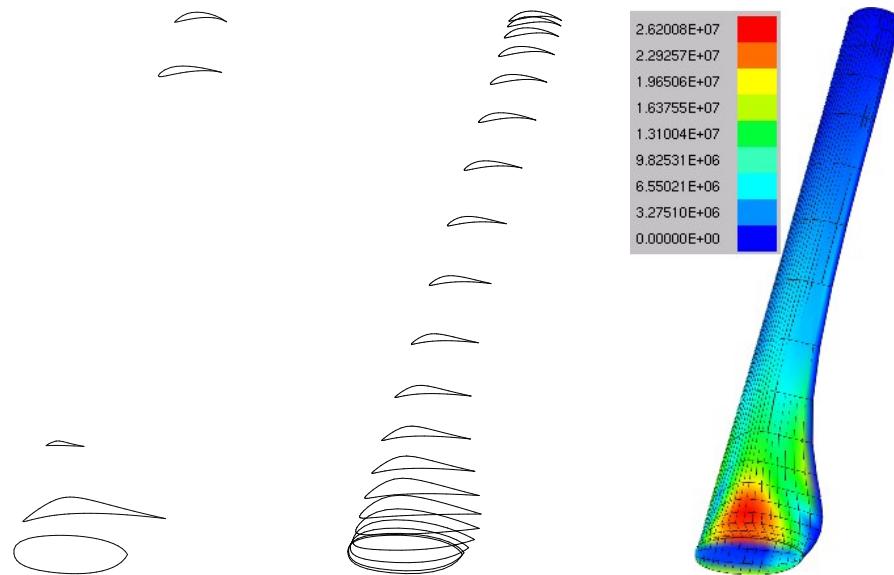
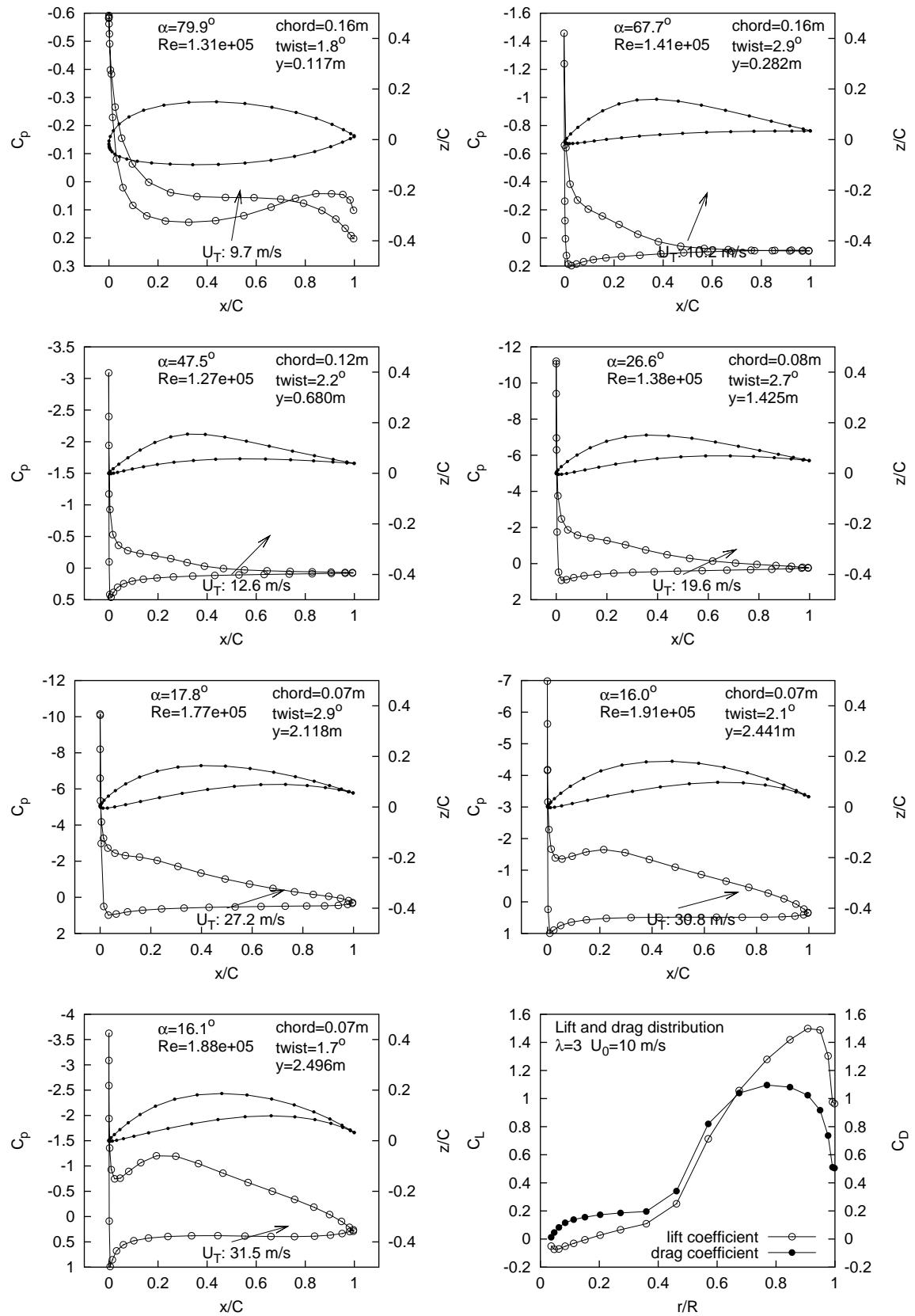
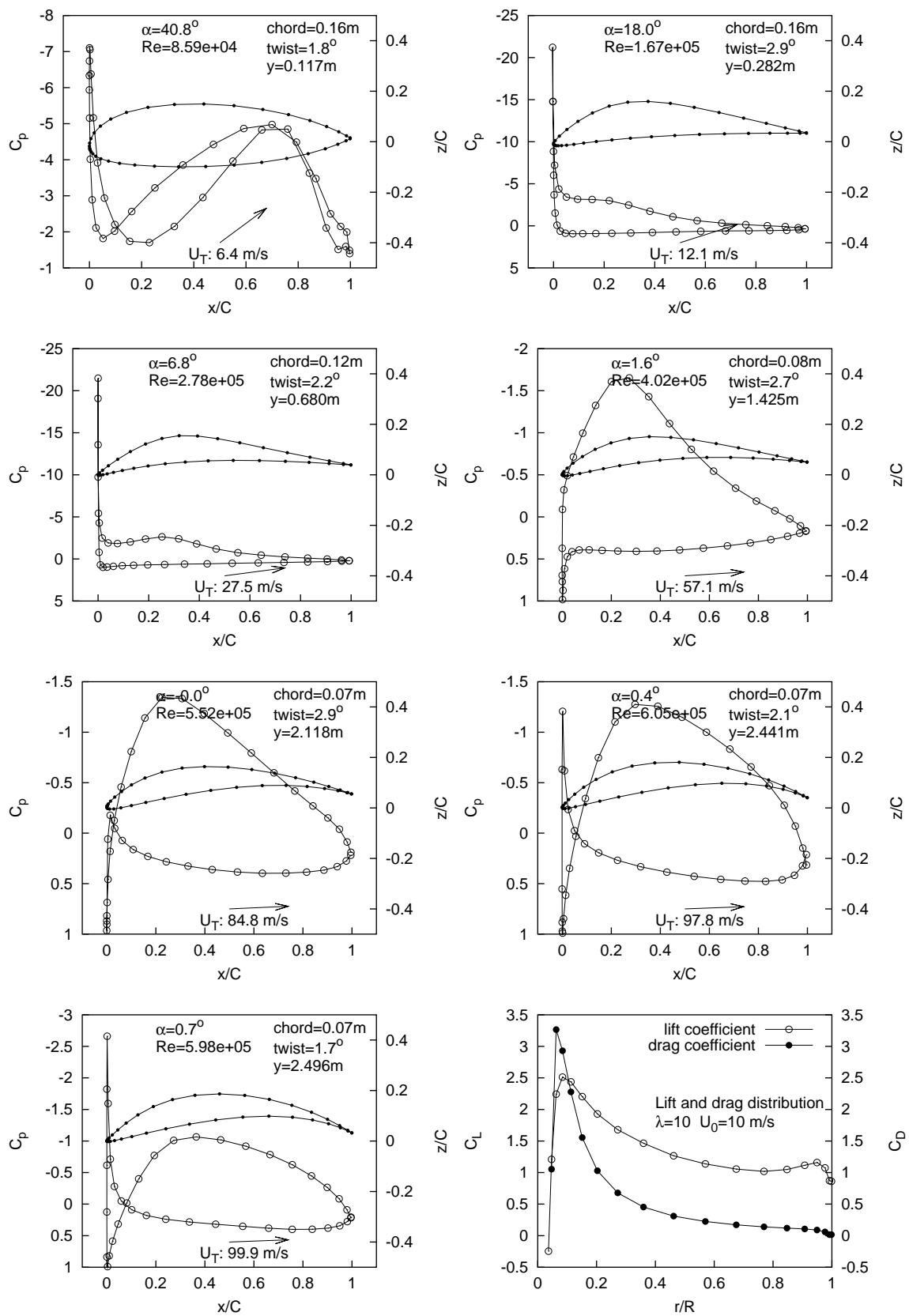
6.33.4: B-spline  
aerofoils6.33.5: Radial  
x-sections6.33.6: von-Mises  
equiv. stress

Figure 6.33: Pareto [B] blade - Geometry and performance results (continued)



Figure 6.34: Pressure and force distributions along Pareto [B] blade @  $\lambda=3$

Figure 6.35: Pressure and force distributions along Pareto [B] blade @  $\lambda=10$

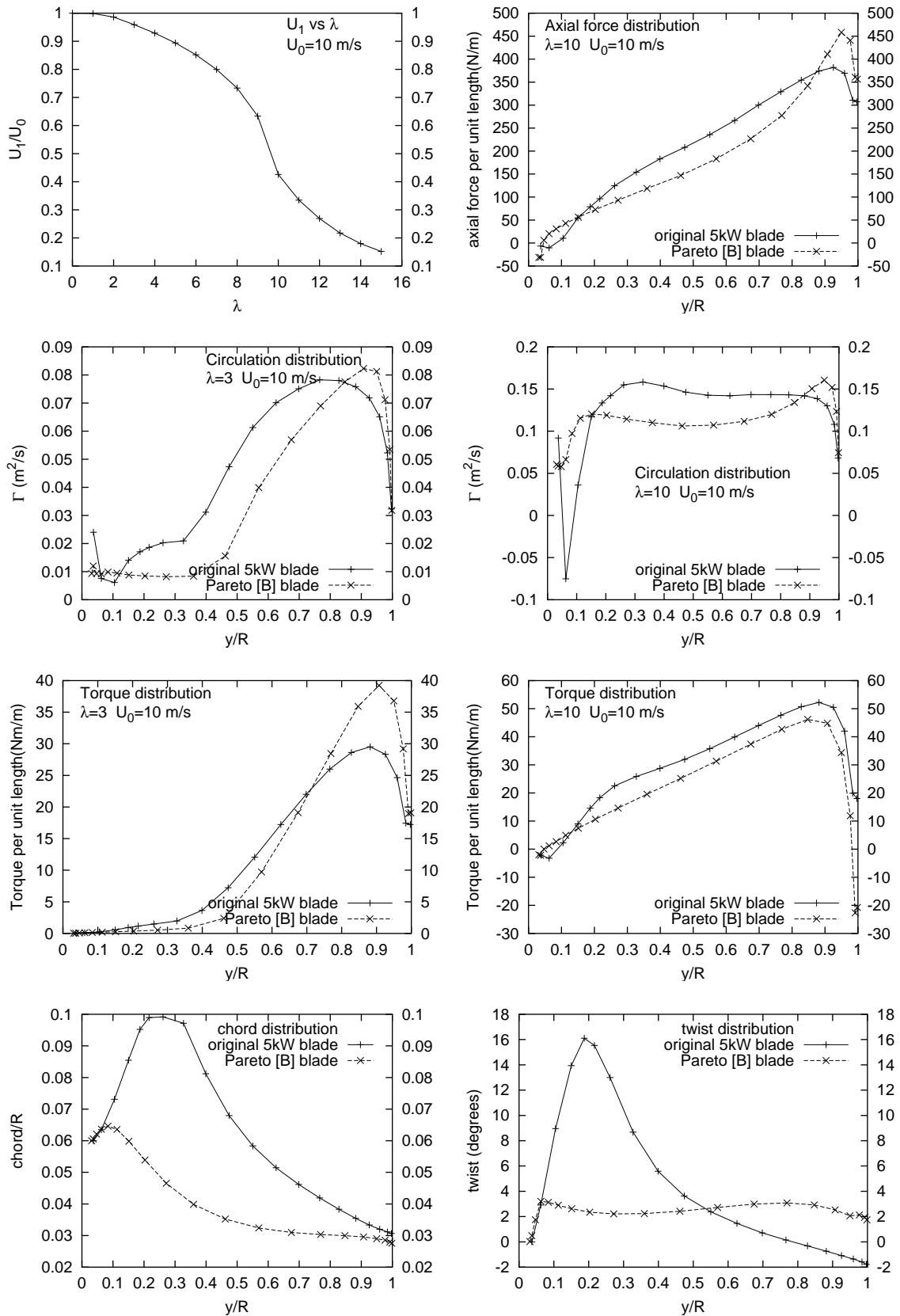


Figure 6.36: Performance results for Pareto [B] blade

### 6.7.12 Pareto set: member [C]

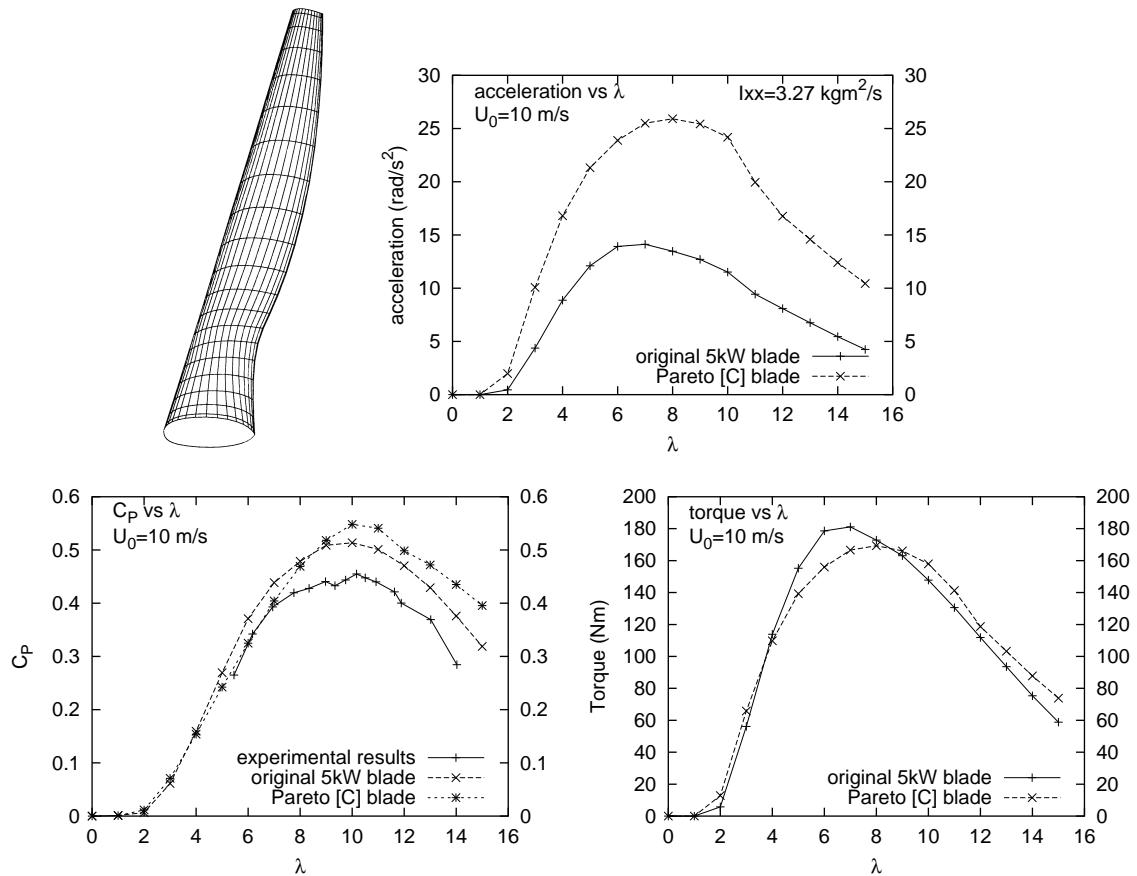


Figure 6.37: Pareto [C] blade - Geometry and performance results

Figure 6.37 shows the meshed geometry of the [C] member of the current Pareto set, together with important performance results. Member [C] is an hybrid of the first two “families” of blade designs identified in the current study: the “*slender*” blade family, and the “*ordinary*” blade family. Its performance is characterised by surprisingly good power production across the entire range of tip speed ratios, together with very good comparative acceleration. The acceleration is obviously the result of the low second-moment of inertia of this blade, together with rather good low- $\lambda$  torque. Whilst power production is very slightly lower at lower tip speed ratios (conditions where operational power extraction is not intended to occur, anyway), power production is *better* than the existing 5 kW Newcastle blade at high- $\lambda$  (at least in the predictions offered by this  $40 \times 20 \times 5 \times 20$  blade mesh). This was not expected, especially

since the blade is so slender, and appears in the Pareto set calculation ( $30 \times 10 \times 5 \times 10$ ) at a significantly lower power figure. One of these results could very well be a gross error, but the detailed higher-density mesh results look convincing. Figure 6.38 shows the blade is constructed from aerofoils with rather less camber than the Pareto [A] and [B] members. Stress concentrations near the root at high- $\lambda$  are significantly lower than for the slender blades. Figures 6.39 and 6.40 show the sectional pressure and span-wise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , respectively. Figure 6.41 shows that, like the very slender [A] and [B] blades, tip-wise torques at low- $\lambda$  are quite high contributing to good starting performance. The chord and twist distributions displayed in Figure 6.41 show that mid-span chords are approaching those of the existing 5 kW Newcastle blade, whilst still allowing for a slender, low- $I_{xx}$  design which also delivers larger chords where the majority of blade torque is generated. A very interesting twist distribution identifies this blade as the hybrid that it is: the strategy of keeping chord lengths low is a slender-family trait, as is the constant, small mid-span twist angle; *but* the increased hub-wise twist distribution is very much a “standard” wind turbine blade design feature.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.38.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.38.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
3.2660	0.5481	6478	973	69.82	9.860	2.94

6.38.3: RESULTS

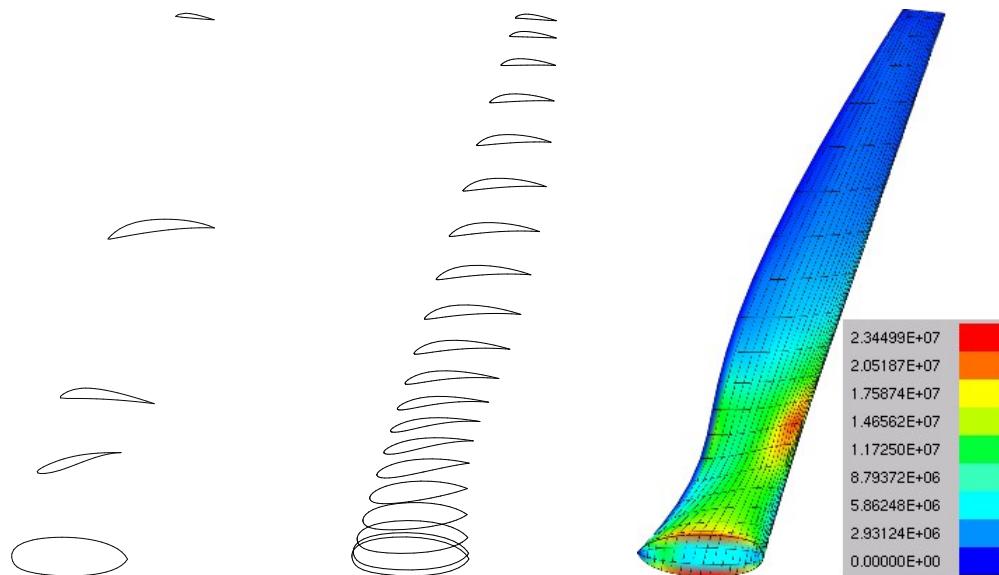
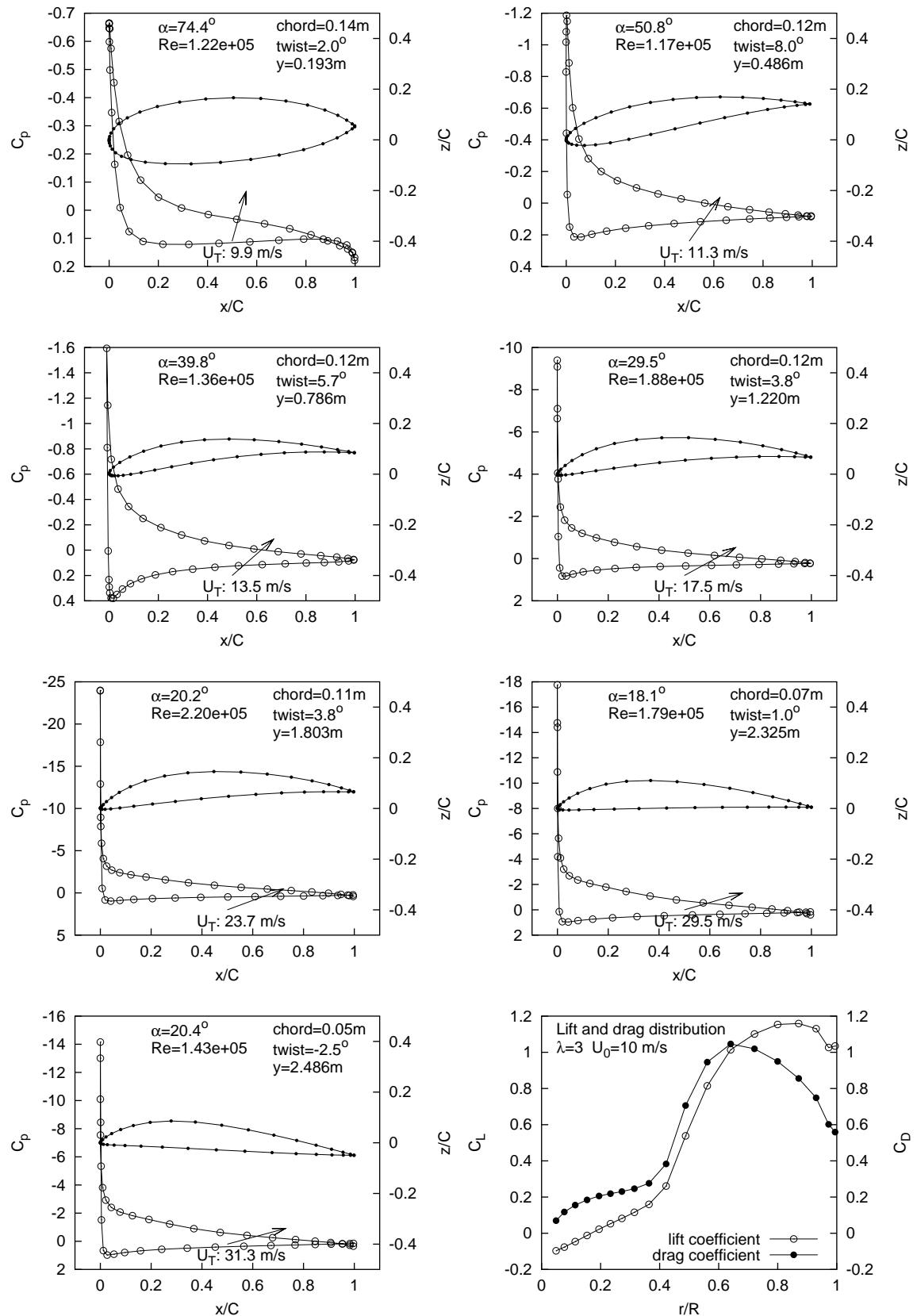
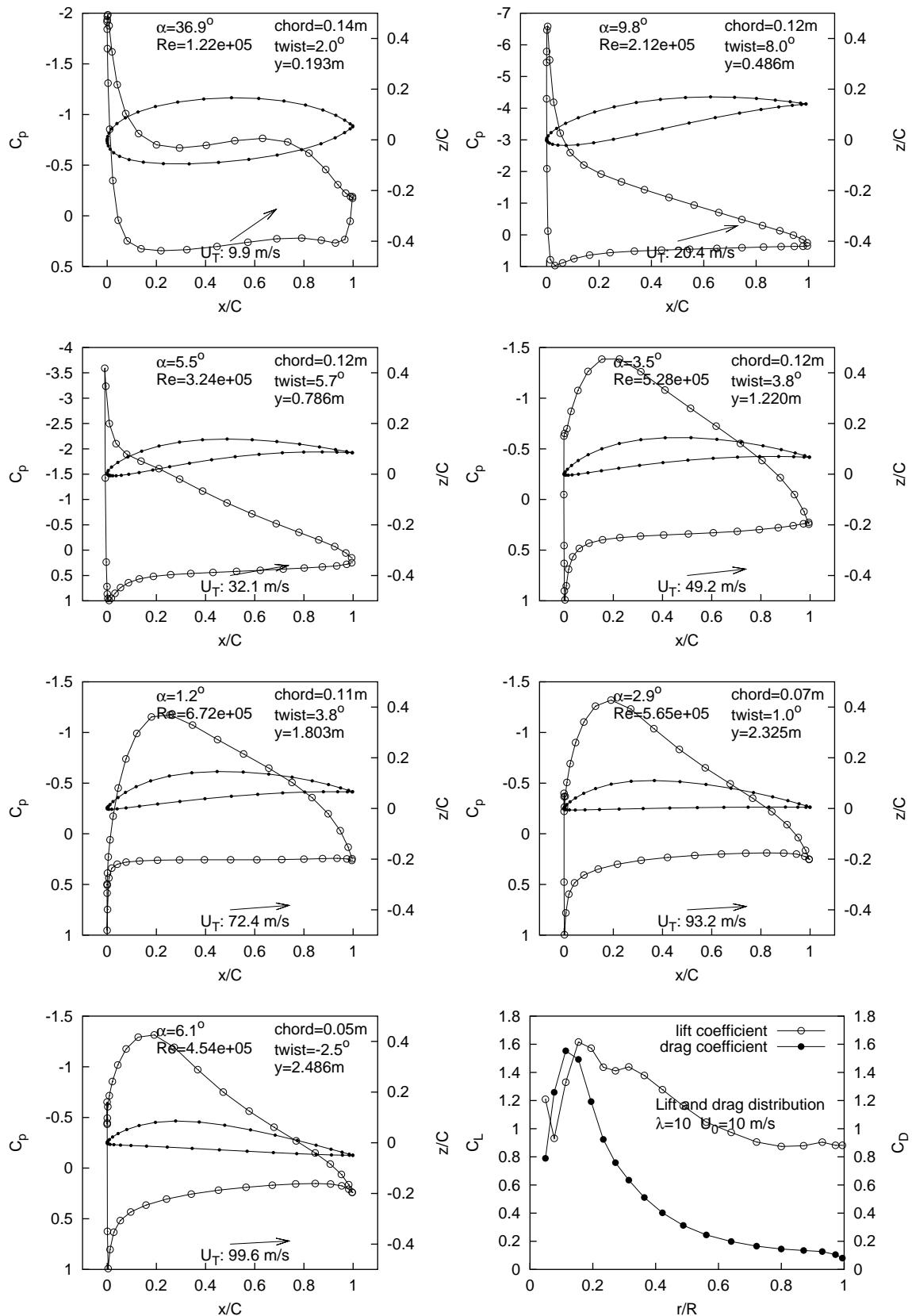
6.38.4: B-spline  
aerofoils6.38.5: Radial  
x-sections6.38.6: von-Mises  
equiv. stress

Figure 6.38: Pareto [C] blade - Geometry and performance results (continued)



Figure 6.39: Pressure and force distributions along Pareto [C] blade @  $\lambda=3$

Figure 6.40: Pressure and force distributions along Pareto [C] blade @  $\lambda=10$

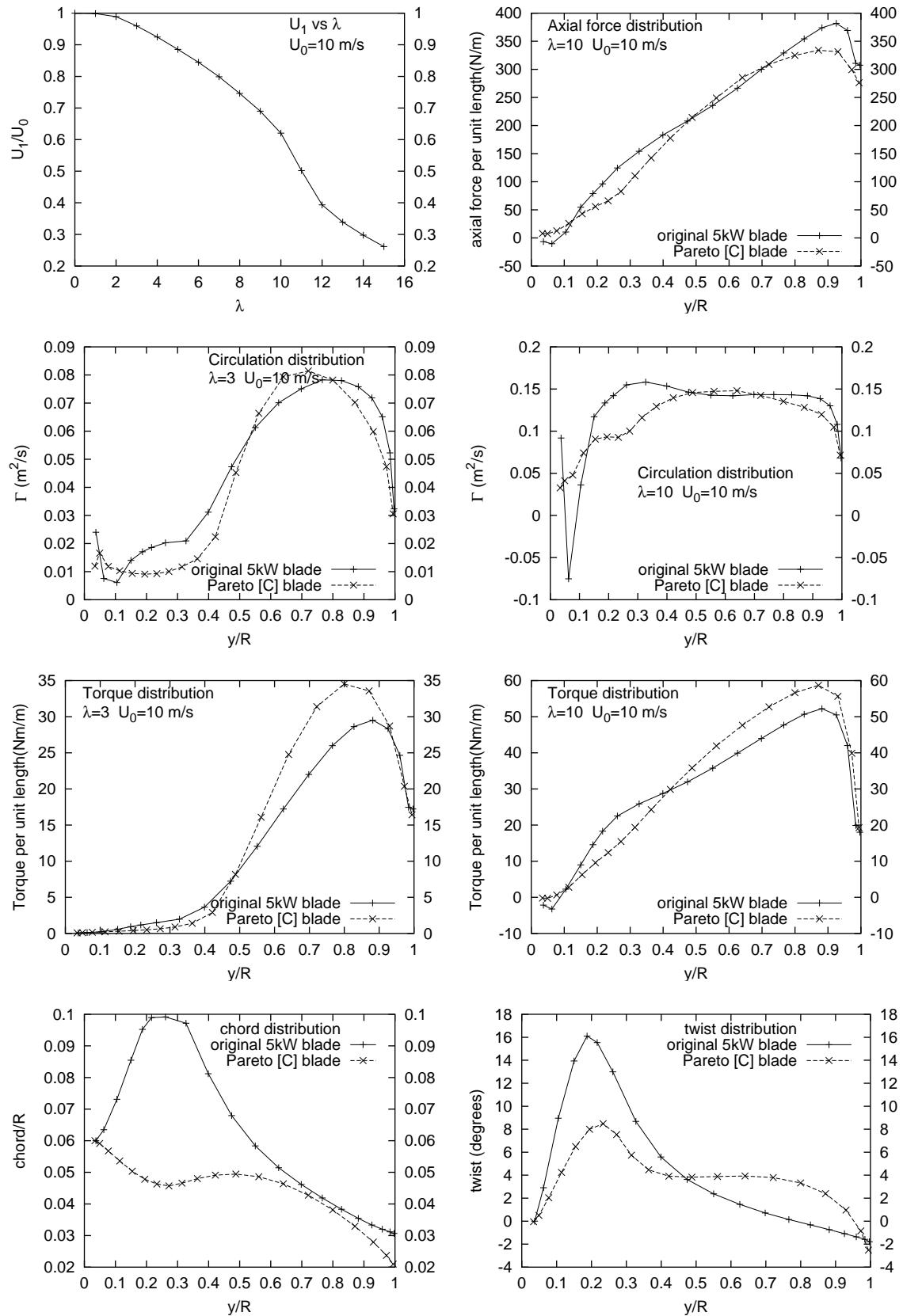


Figure 6.41: Performance results for Pareto [C] blade

### 6.7.13 Pareto set: member [D]

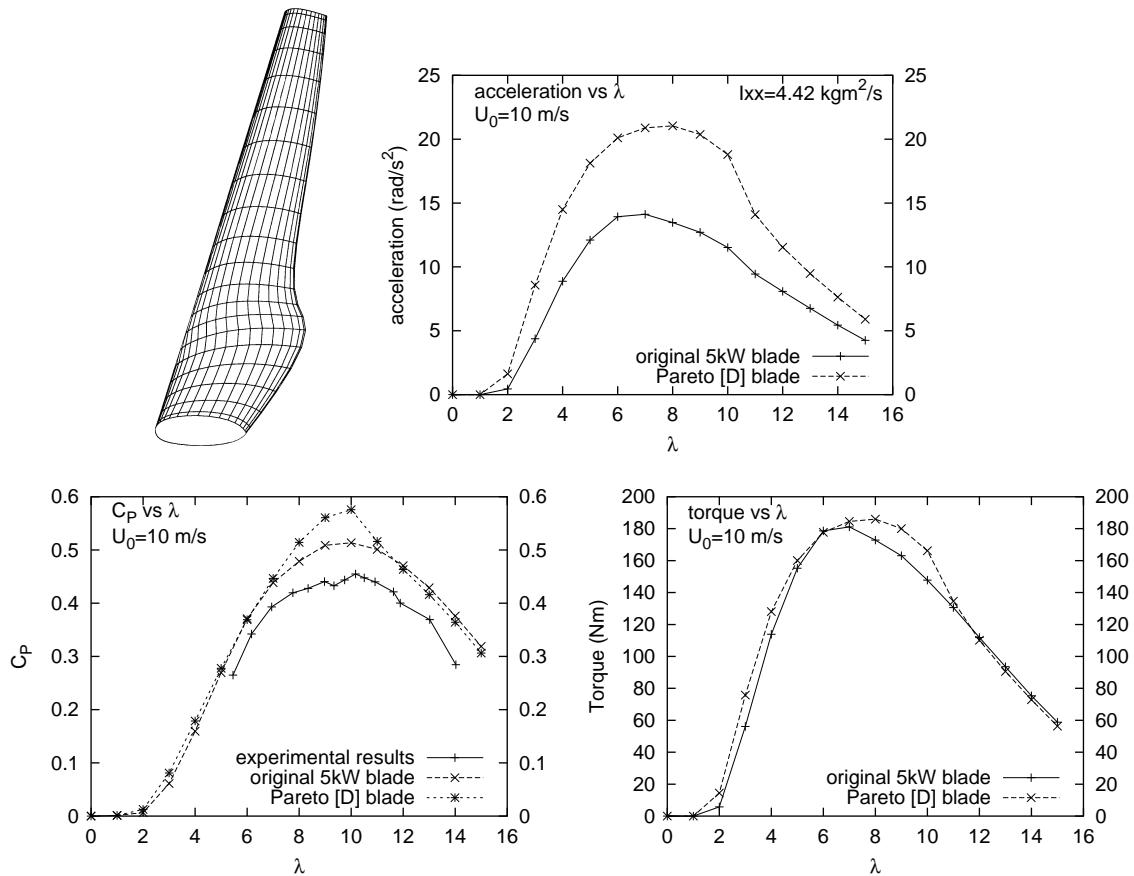


Figure 6.42: Pareto [D] blade - Geometry and performance results

Figure 6.42 shows the meshed geometry of the [D] member of the current Pareto set, together with important performance results. Member [D] is a very interesting example of the “ordinary” blade family. Like all members of the “ordinary” blade family, its performance is characterised by good power production across the entire range of tip speed ratios - with Figure 6.42 demonstrating that geometric similarity with the Newcastle 5 kW blade translates into very similar  $C_P$  and high- $\lambda$  torque figures as well. Acceleration performance benefits greatly from the significantly lower second-moment of inertia of this design, a result attained through increased slenderness towards the blade root and thinner cross-sectional “aerofoils” throughout. Figure 6.43 shows the blade is constructed from “regular-looking” aerofoils. Relatively good stress concentrations near the root at high- $\lambda$  point to the advantages of a more

substantial root geometry. Figures 6.44 and 6.45 show the sectional pressure and spanwise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , respectively. Low- $\lambda$  circulation and torque results in Figure 6.46 demonstrate that good tip-wise lift and torque is being generated, augmenting the already reduced second-moment of inertia. The chord and twist distributions in Figure 6.31 show the similarity of this blade's geometry with the existing 5 kW Newcastle blade. This fact is notable, especially when the geometry of the “seed” blade is taken into consideration and given that the evolutionary optimisation process had no *a priori* knowledge of existing state-of-the-art blade designs.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.43.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.43.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
4.4184	0.5758	6806	1097	79.80	8.359	4.52

6.43.3: RESULTS

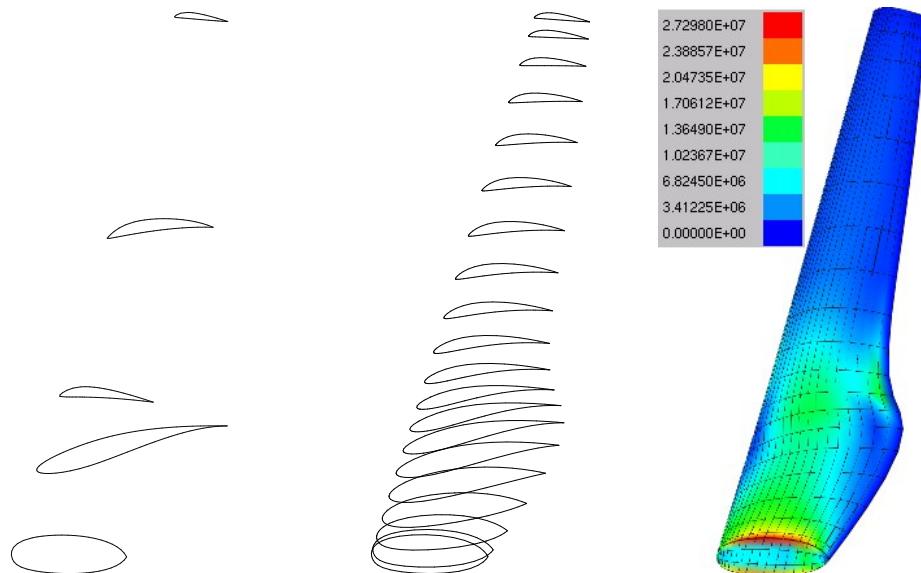
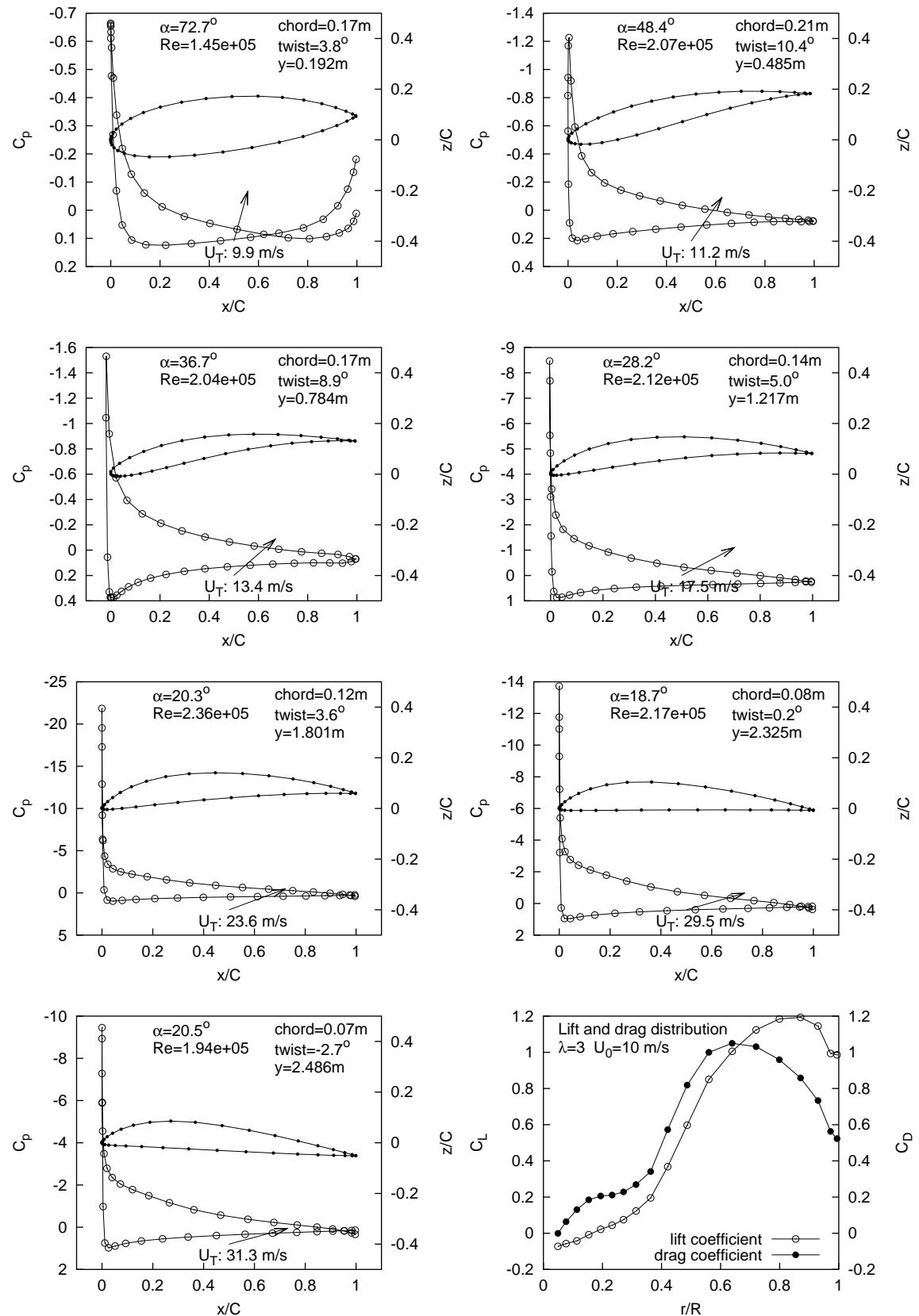
6.43.4: B-spline  
aerofoils6.43.5: Radial  
x-sections6.43.6: von-Mises  
equiv. stress

Figure 6.43: Pareto [D] blade - Geometry and performance results (continued)



Figure 6.44: Pressure and force distributions along Pareto [D] blade @  $\lambda=3$

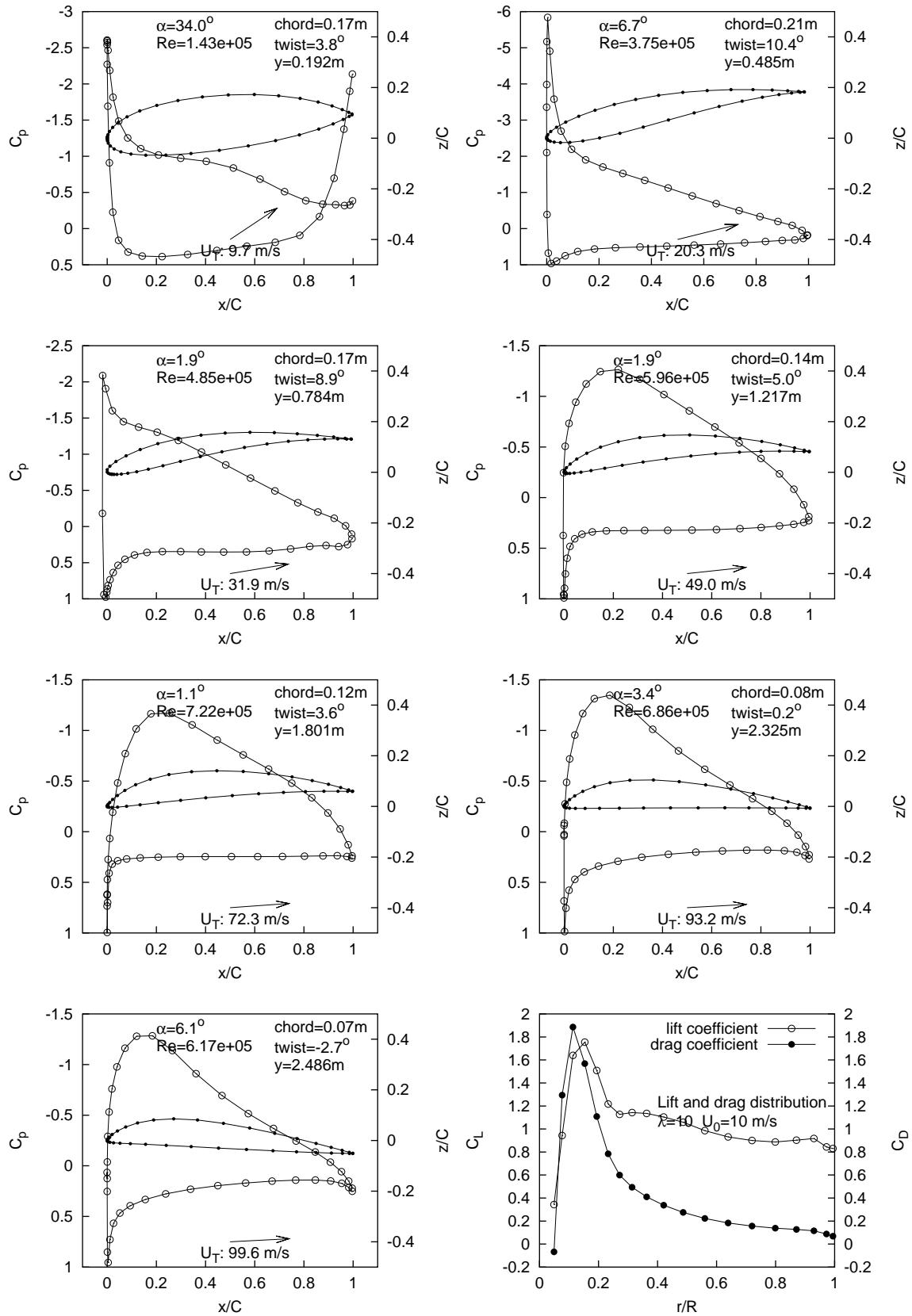


Figure 6.45: Pressure and force distributions along Pareto [D] blade @  $\lambda=10$

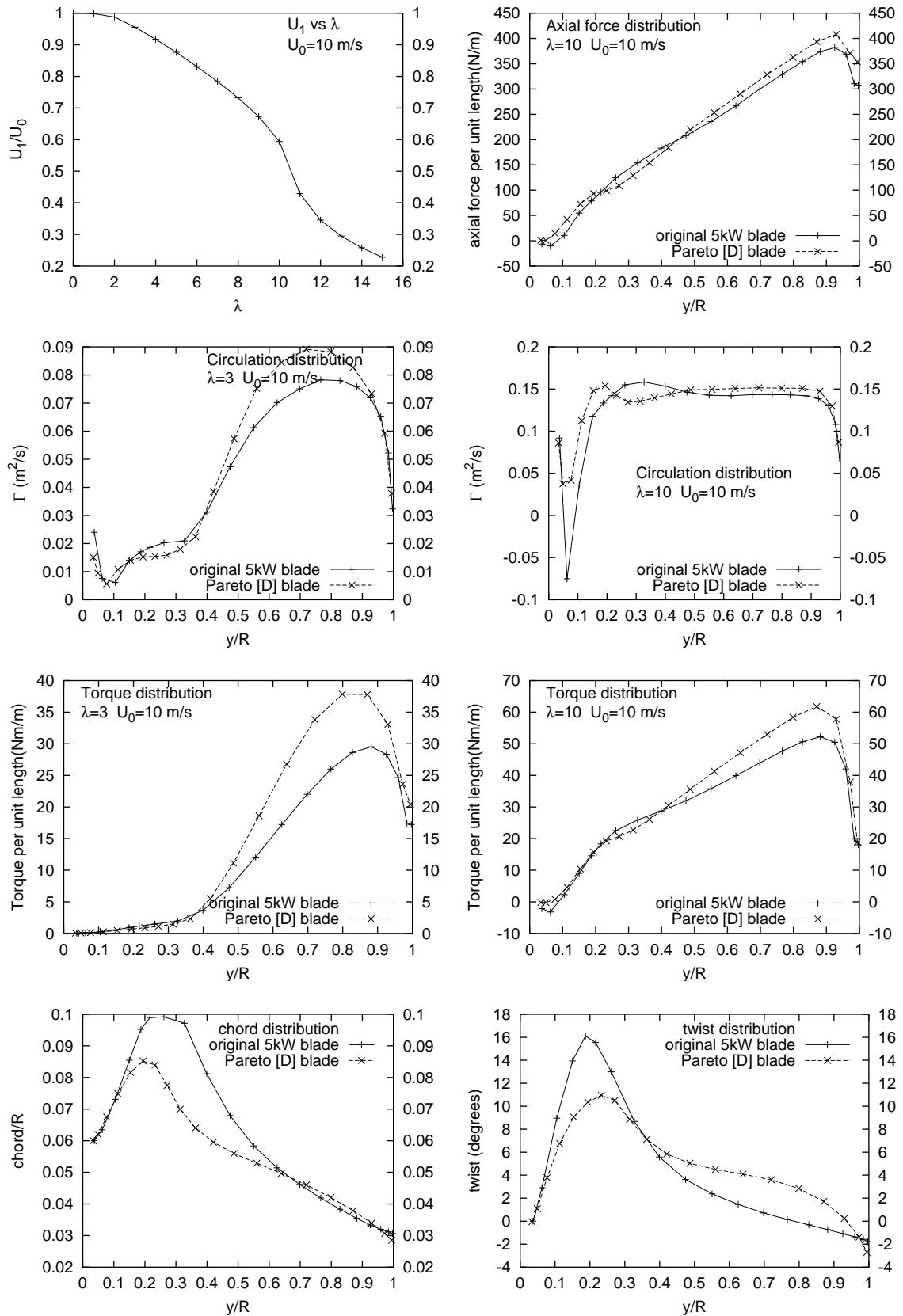


Figure 6.46: Performance results for Pareto [D] blade

### 6.7.14 Pareto set: member [E]

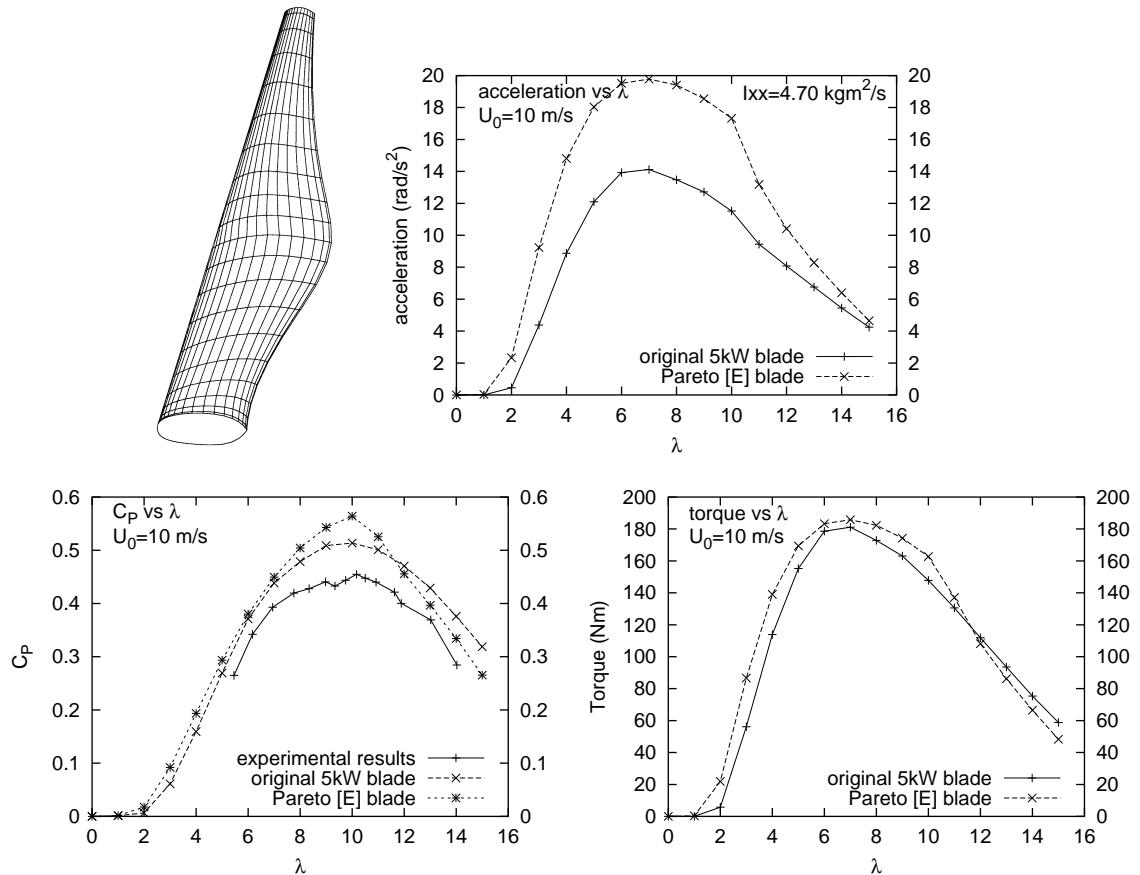


Figure 6.47: Pareto [E] blade - Geometry and performance results

Figure 6.47 shows the meshed geometry of the [E] member of the current Pareto set, together with important performance results. Member [E] is a member of the “*ordinary*” blade family, and shares the family’s generally good power production and starting performance. Two geometric features make this blade unusual: cross-sections along its span (especially around the mid-span) do not look like the “standard” low-speed aerofoil sections normally found in small wind turbines (see Figure 6.48 and the pressure-distribution plots in Figures 6.49 and 6.50); and the chord distribution is skewed so that it favours larger chords near the mid-span, rather than the “standard” larger chords closer to the hub with a simple taper (see the chord distribution in Figure 6.46). This results in an approximate 7% increase in second-moment of inertia, compared with the Pareto [D] blade.

B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.48.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.48.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
4.6962	0.5643	6670	1011	90.67	8.721	4.94

6.48.3: RESULTS

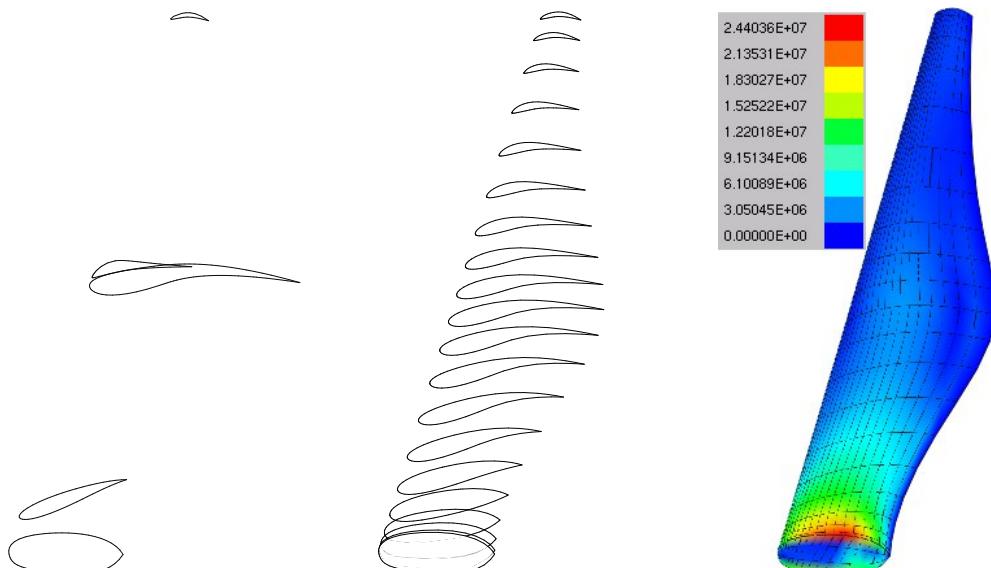
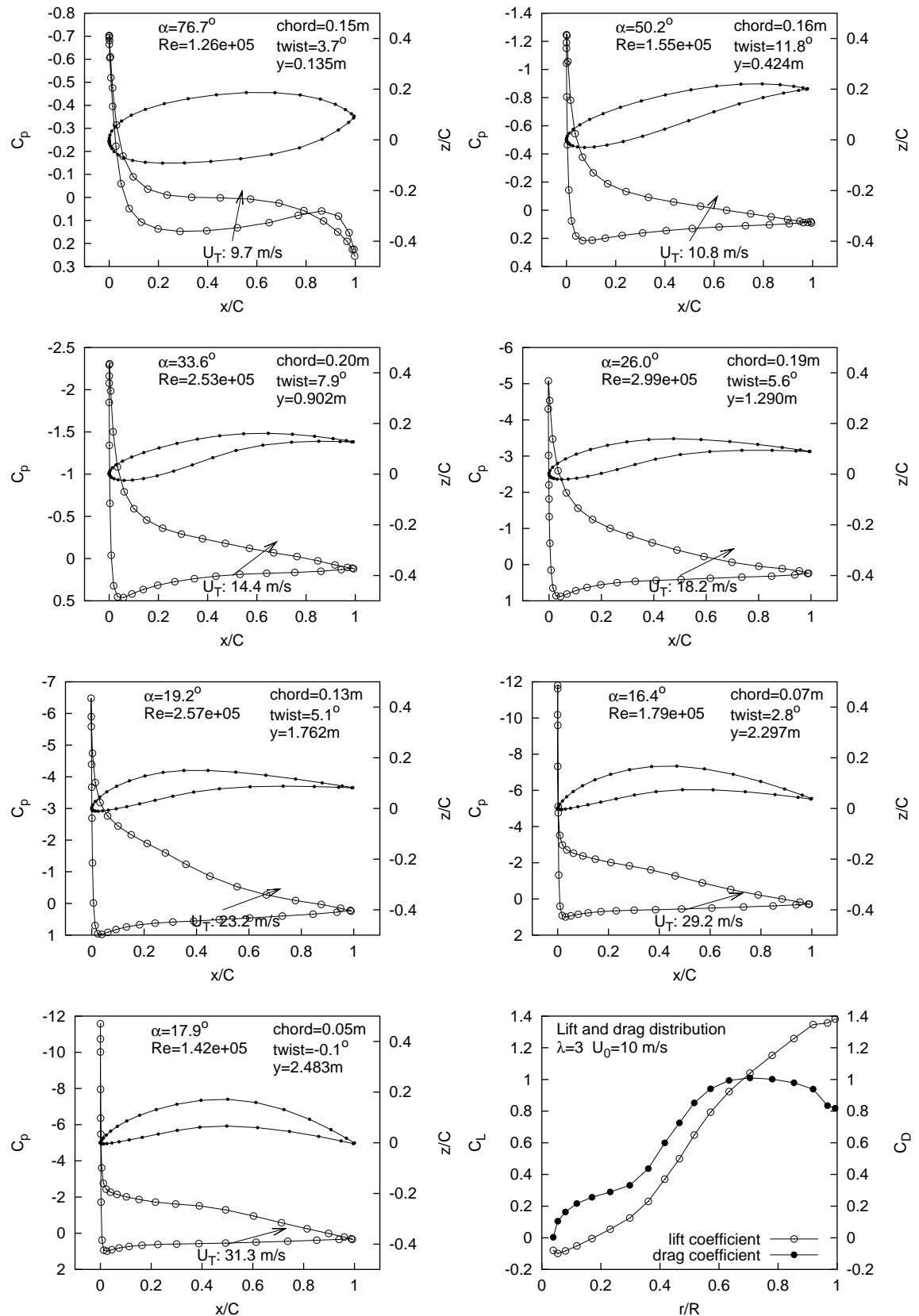
6.48.4: B-spline  
aerofoils6.48.5: Radial  
x-sections6.48.6: von-Mises  
equiv. stress

Figure 6.48: Pareto [E] blade - Geometry and performance results (continued)



Figure 6.49: Pressure and force distributions along Pareto [E] blade @  $\lambda=3$

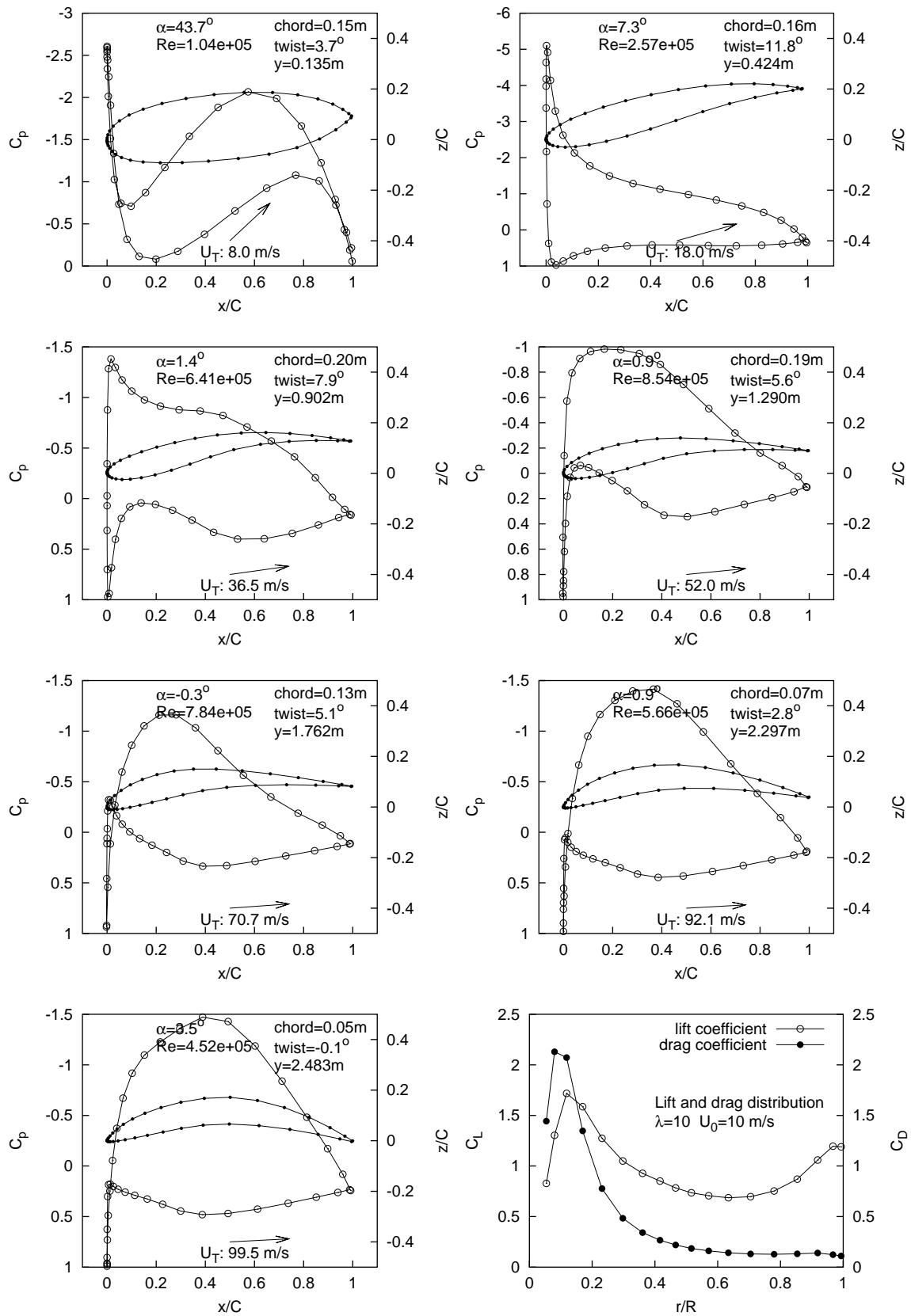


Figure 6.50: Pressure and force distributions along Pareto [E] blade @  $\lambda=10$

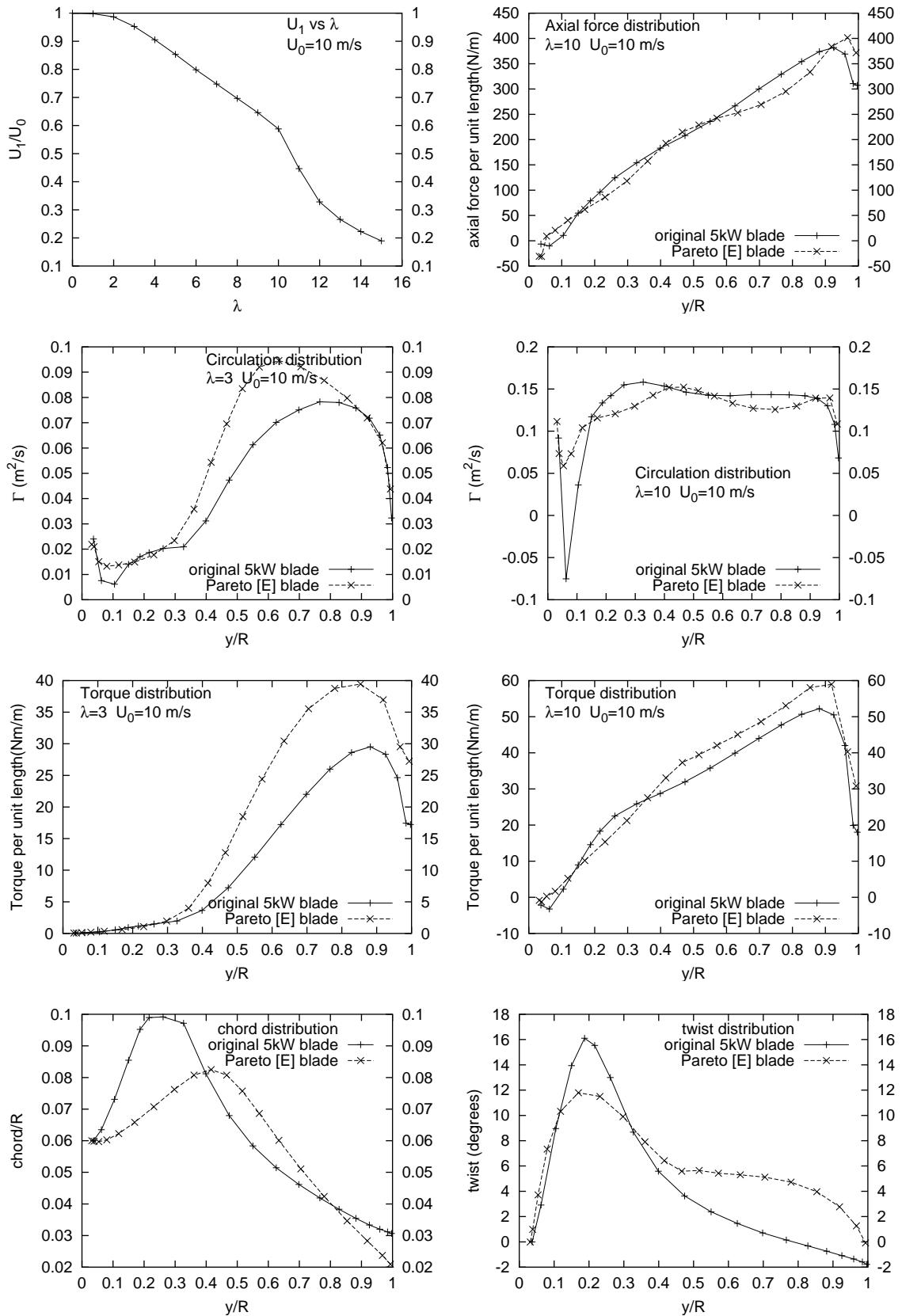


Figure 6.51: Performance results for Pareto [E] blade

### 6.7.15 Pareto set: member [F]

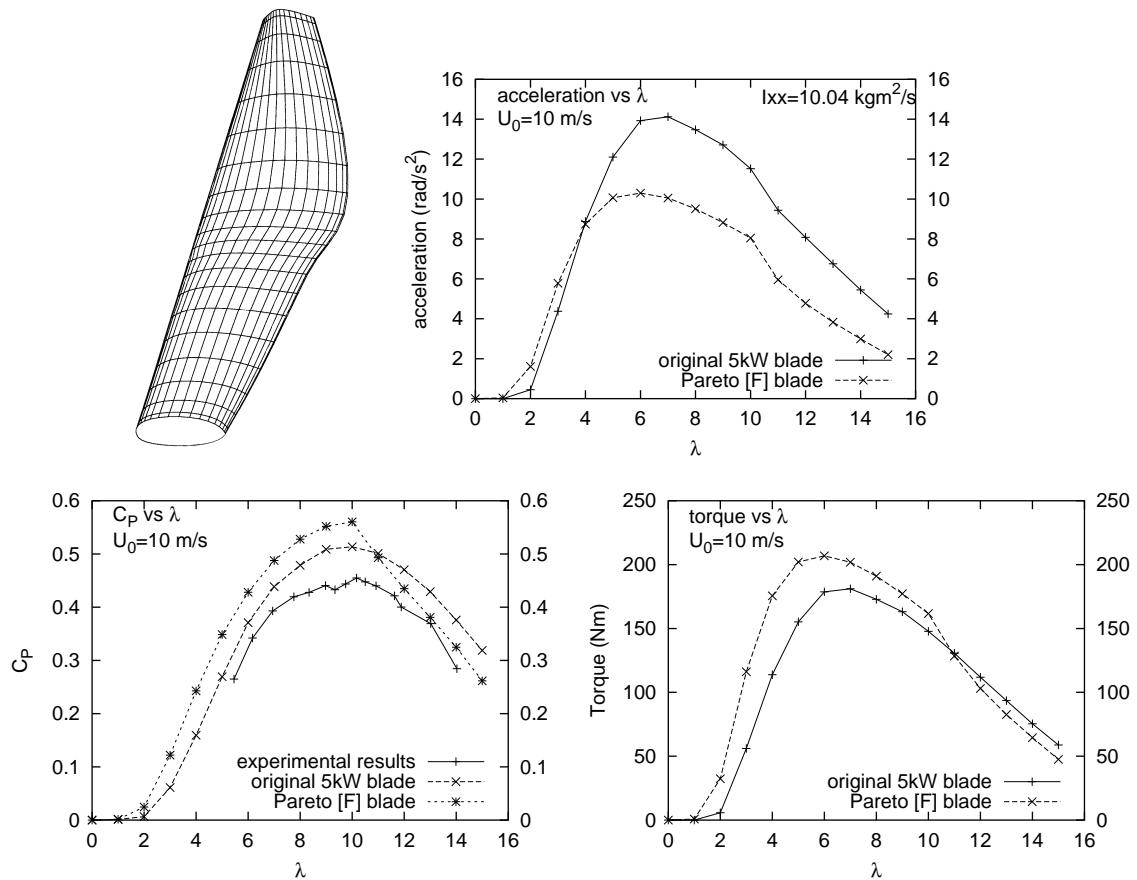


Figure 6.52: Pareto [F] blade - Geometry and performance results

Figure 6.52 shows the meshed geometry of the [F] member of the current Pareto set, together with important performance results. Member [F] is an “*Ordinary-Heavy*” hybrid, and its geometry reflects the fact via a decreasing taper from the mid-span to the tip. As described in the introduction to this results section, the performance of *Ordinary-Heavy* and *Heavy* blade families is not Pareto optimal, owing to the inadequacies of the lower-resolution aerodynamic objective function which allowed them to evolve. Nevertheless, the existence of this blade is instructive for no reason other than its power production is good (but not optimal), and that it shares a common evolution history with the “lighter” blade families identified in the current project. This last is an important point: blades like this will undoubtedly appear in the evolving population of *all* evolutionary optimisation runs carried out using the current optimisation

method. They *will* influence the evolution of other blade designs that ultimately win their way into the Pareto set, simply because of the very good low- $\lambda$  torques afforded by very-large mid-span and tip chords. These are demonstrated in Figure 6.52 and Figure 6.31. The advantages of these torques are largely overwhelmed by the increased second-moment of inertia figures, though.

Figure 6.53 shows the blade is constructed from some interesting conventional-looking aerofoils, especially hub-wise from mid-span. Figures 6.54 and 6.55 show the sectional pressure and span-wise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , respectively. Figure 6.31 also shows that the blade possesses a very large mid-span chord and a “flat” twist distribution.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.53.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.53.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
10.0411	0.5605	6625	1113	120.11	5.566	7.36

6.53.3: RESULTS

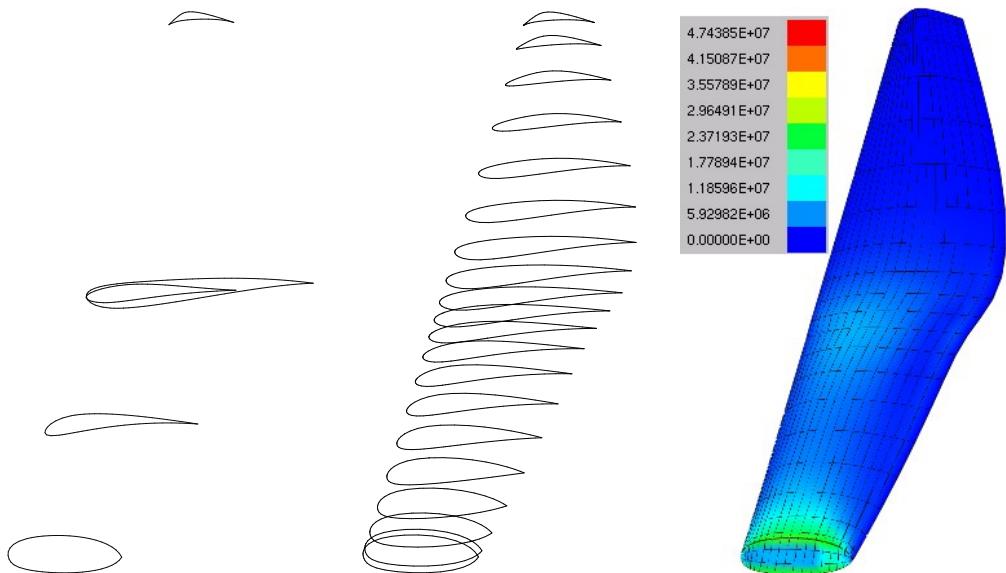
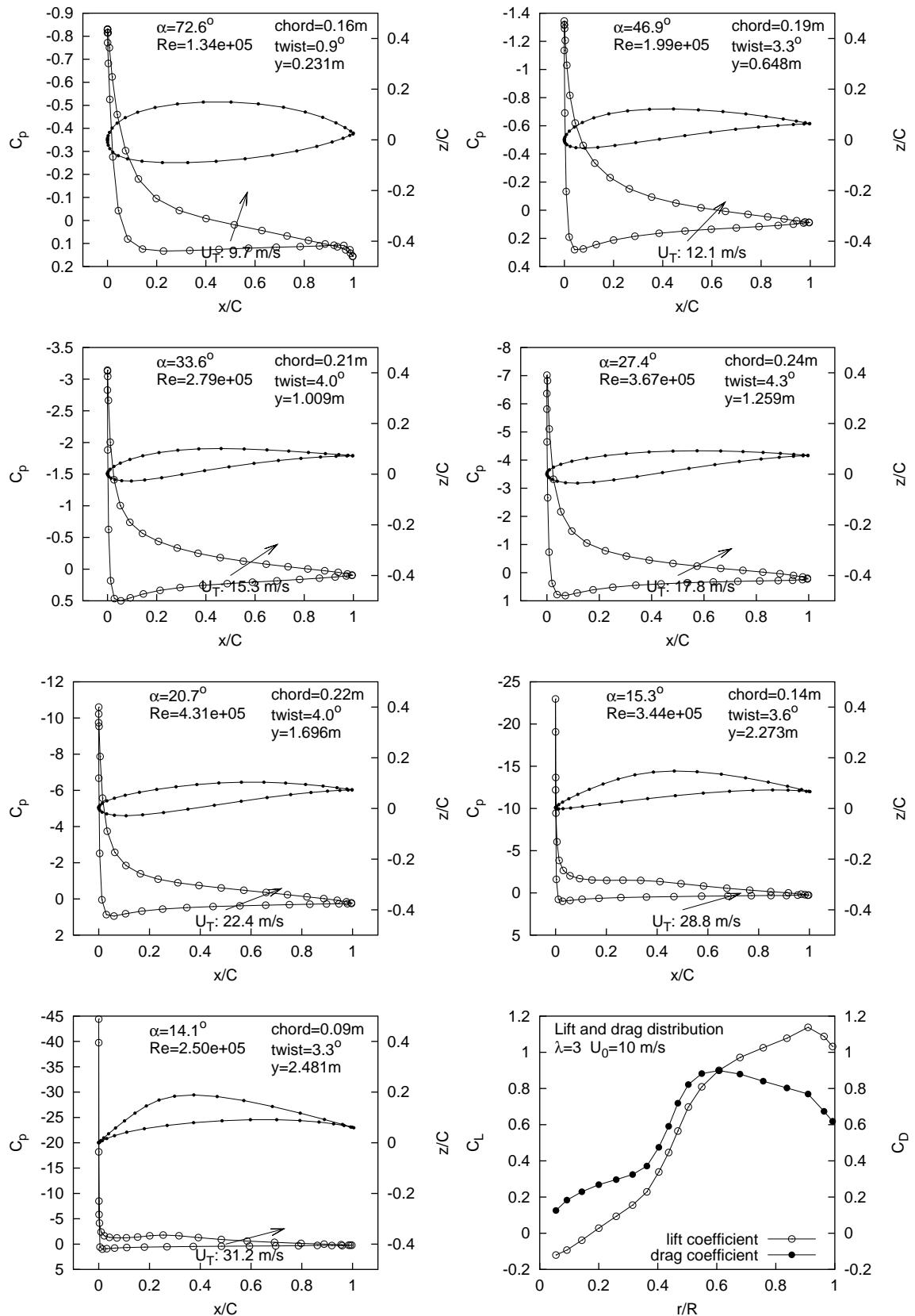
6.53.4: B-spline  
aerofoils6.53.5: Radial  
x-sections6.53.6: von-Mises  
equiv. stress

Figure 6.53: Pareto [F] blade - Geometry and performance results (continued)



Figure 6.54: Pressure and force distributions along Pareto [F] blade @  $\lambda=3$

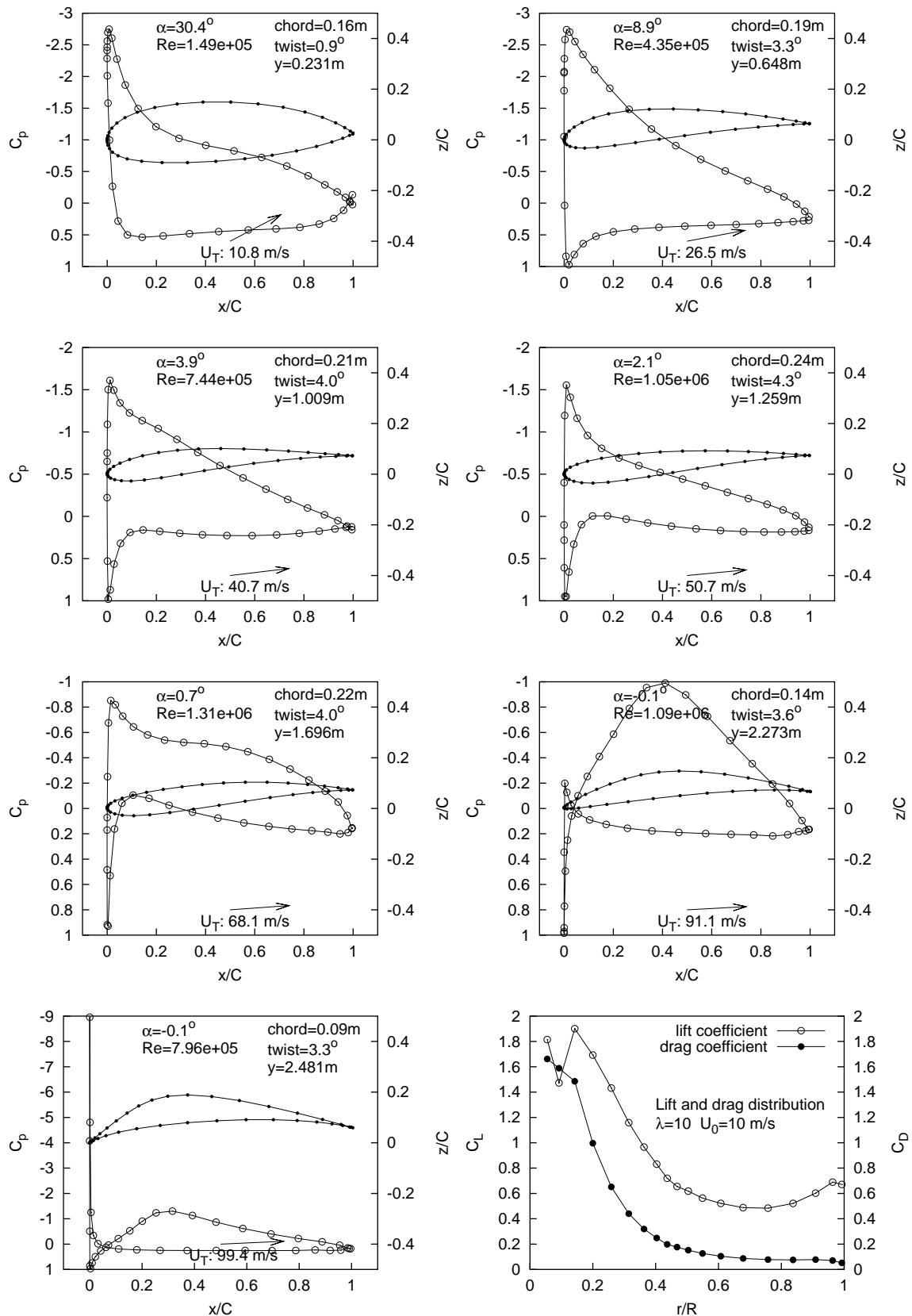


Figure 6.55: Pressure and force distributions along Pareto [F] blade @  $\lambda=10$

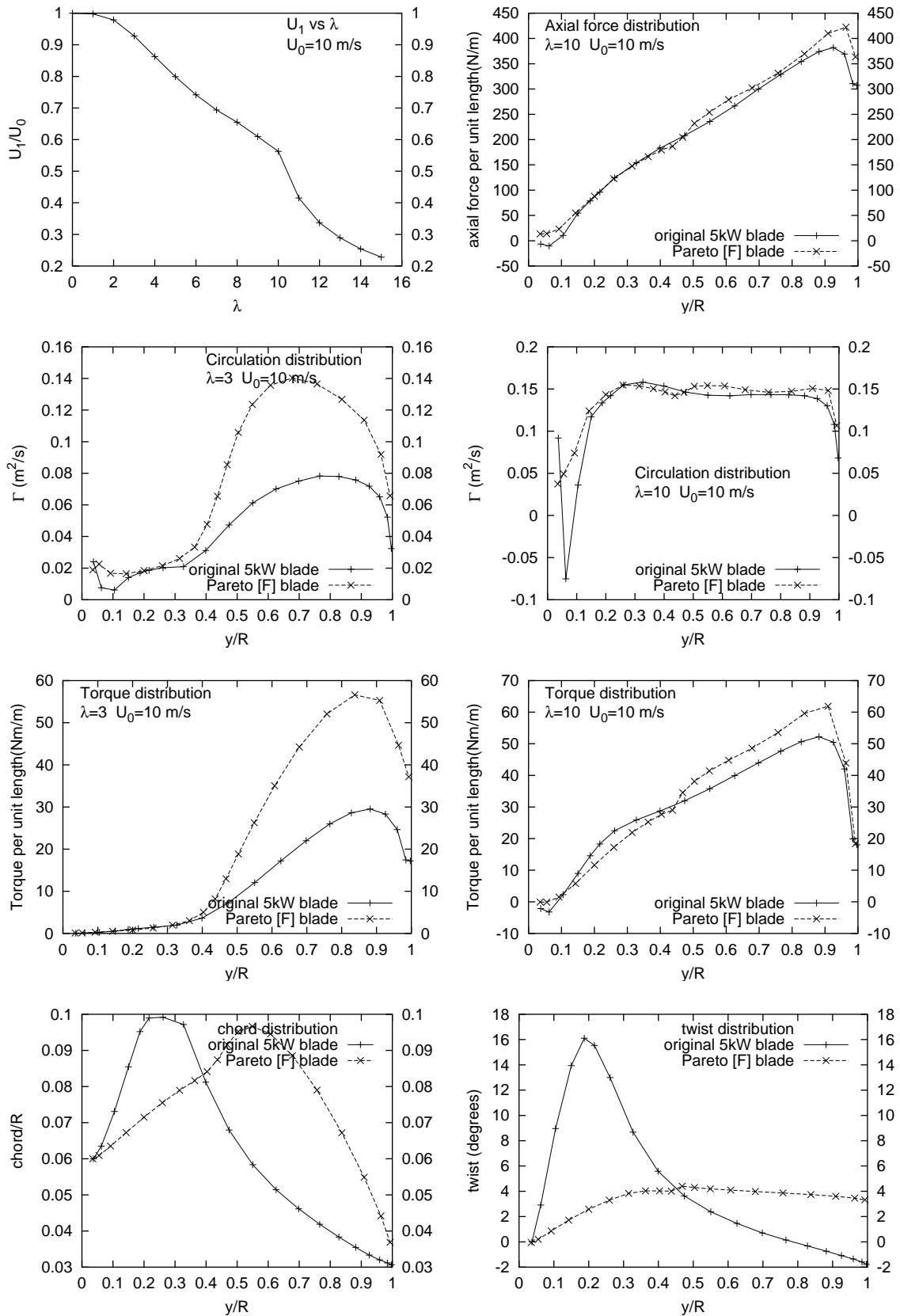


Figure 6.56: Performance results for Pareto [F] blade

### 6.7.16 Pareto set: member [G]

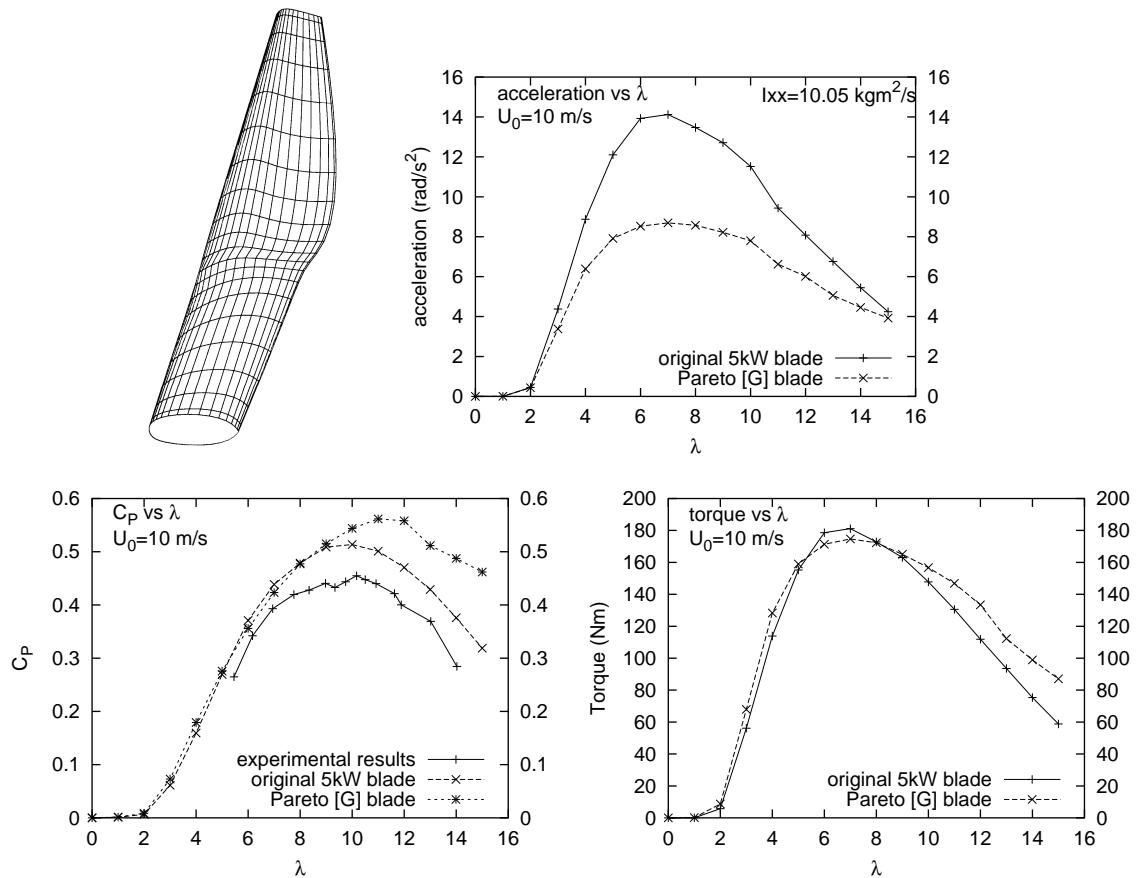


Figure 6.57: Pareto [G] blade - Geometry and performance results

Figure 6.57 shows the meshed geometry of the [G] member of the current Pareto set, together with important performance results. Member [G] is another “*Ordinary-Heavy*” hybrid, and was chosen for study simply because of the strange “aerofoil” sections it possesses. The advantages of high low- $\lambda$  torques and “good” power production, which were the defining performance features of the [F] “*Ordinary-Heavy*” member and in fact are its only redeeming features, are not evident in this blade, which seems to have suffered a large survival disadvantage - via the loss of its excellent predicted power production - after subsequent aerodynamic evaluation at the higher panel mesh resolution.

Figure 6.58 shows the blade is constructed from a set of decidedly strange aerofoils. Figures 6.59 and 6.60 display the sectional pressure and span-wise lift/drag

distributions at  $\lambda=3$  and  $\lambda=10$ , respectively, and demonstrate that even unconventional aerodynamic shapes can provide “good” aerodynamic performance, as evidenced by the span-wise lift and drag distributions. Figure 6.61 shows that the blade possesses a twist distribution which is remarkably similar to that of the existing 5 kW Newcastle blade, a feature that appears to have a beneficial bearing on the  $\lambda=10$  span-wise torque distribution along the hub-wise half of the blade span.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.58.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.58.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
10.0462	0.5438	6428	868	71.93	3.243	7.83

6.58.3: RESULTS

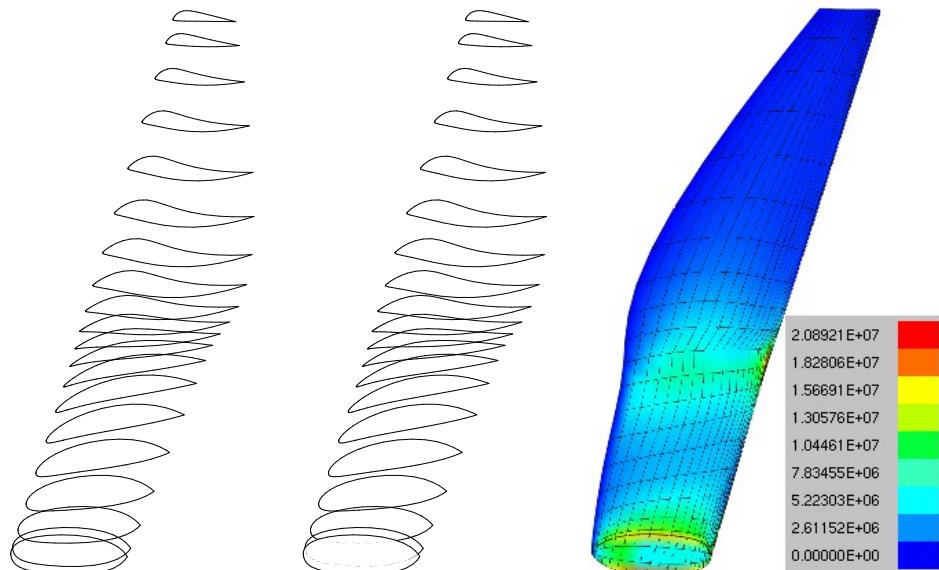
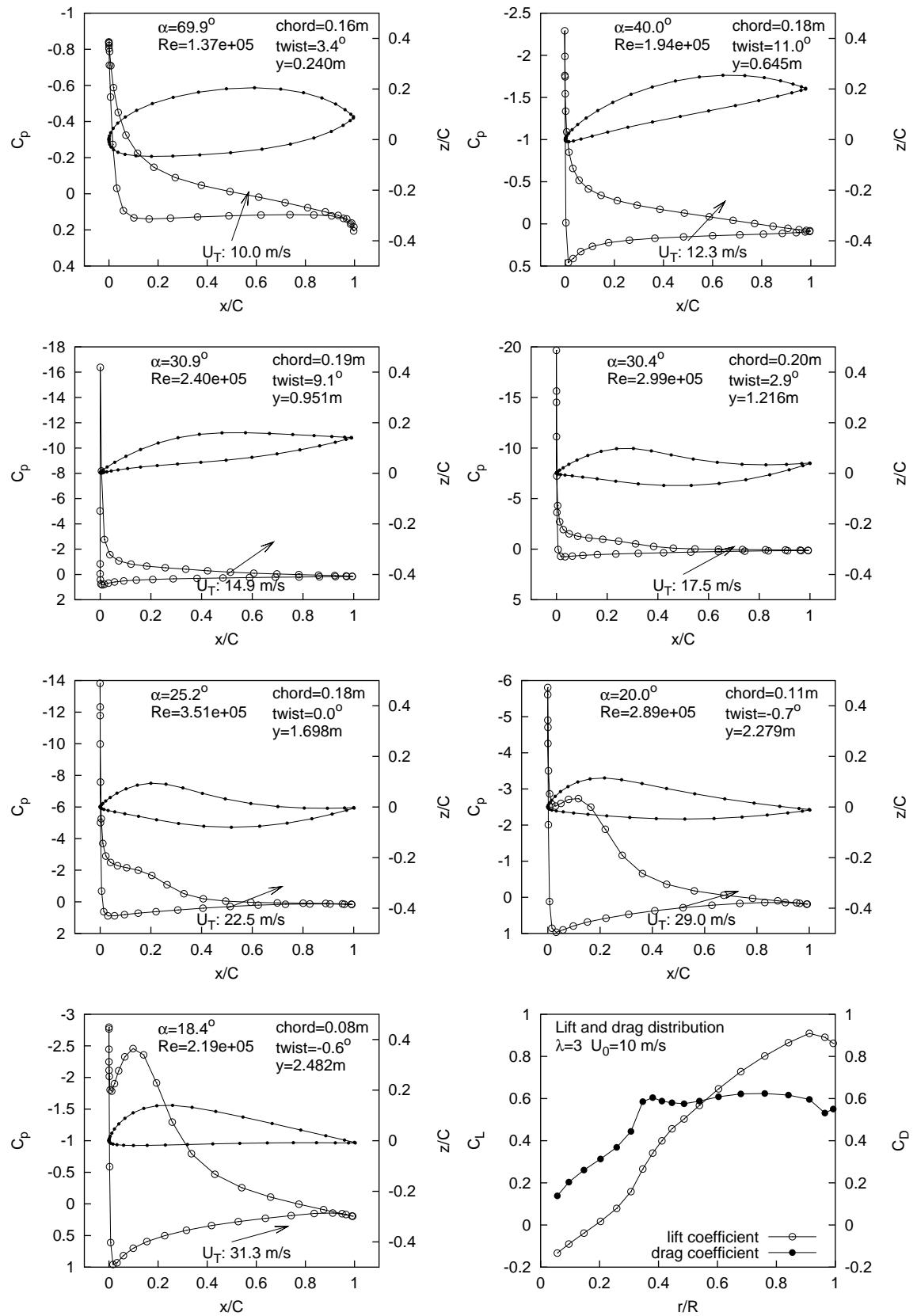
6.58.4: B-spline  
aerofoils6.58.5: Radial  
x-sections6.58.6: von-Mises  
equiv. stress

Figure 6.58: Pareto [G] blade - Geometry and performance results (continued)



Figure 6.59: Pressure and force distributions along Pareto [G] blade @  $\lambda=3$

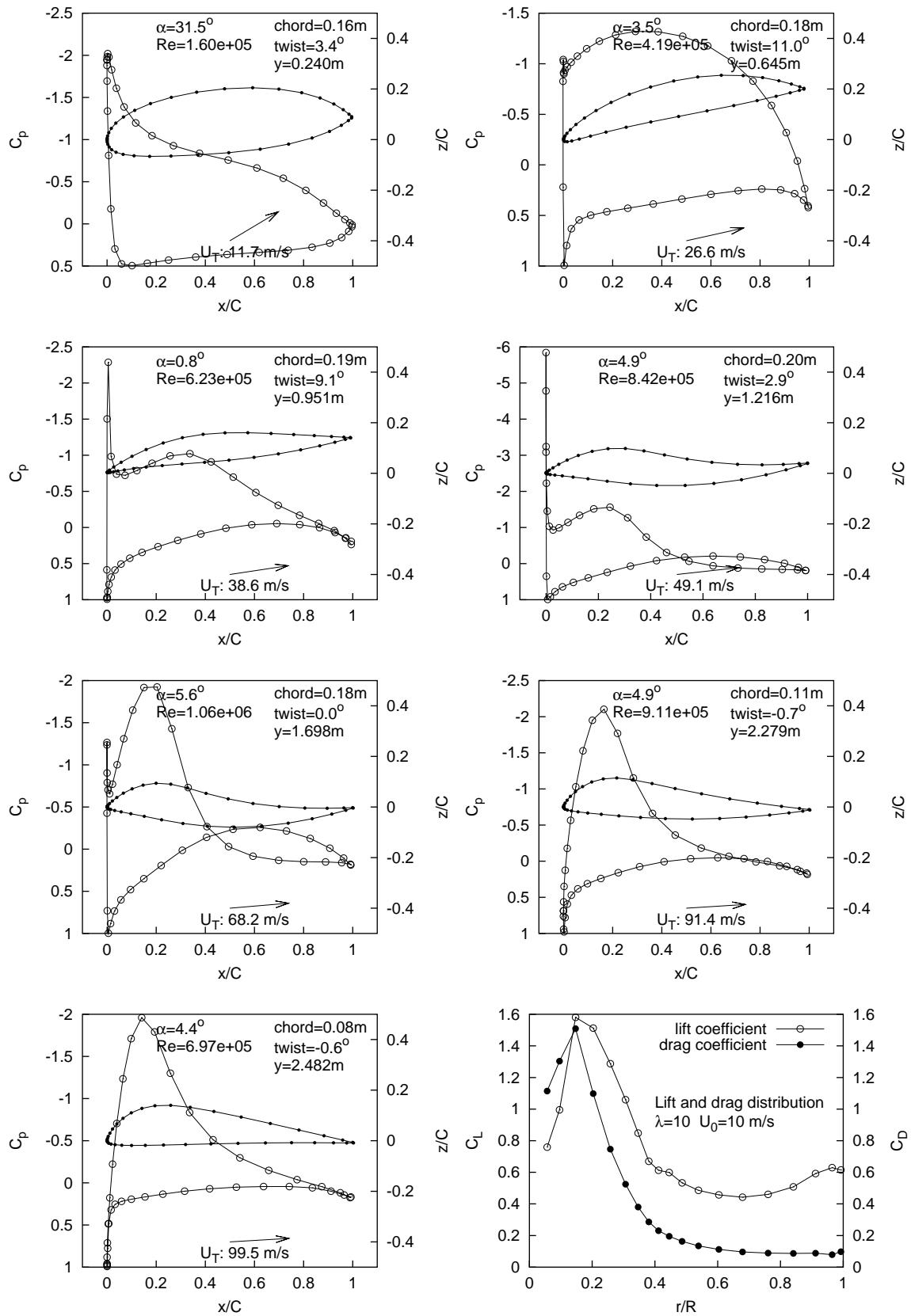


Figure 6.60: Pressure and force distributions along Pareto [G] blade @  $\lambda=10$

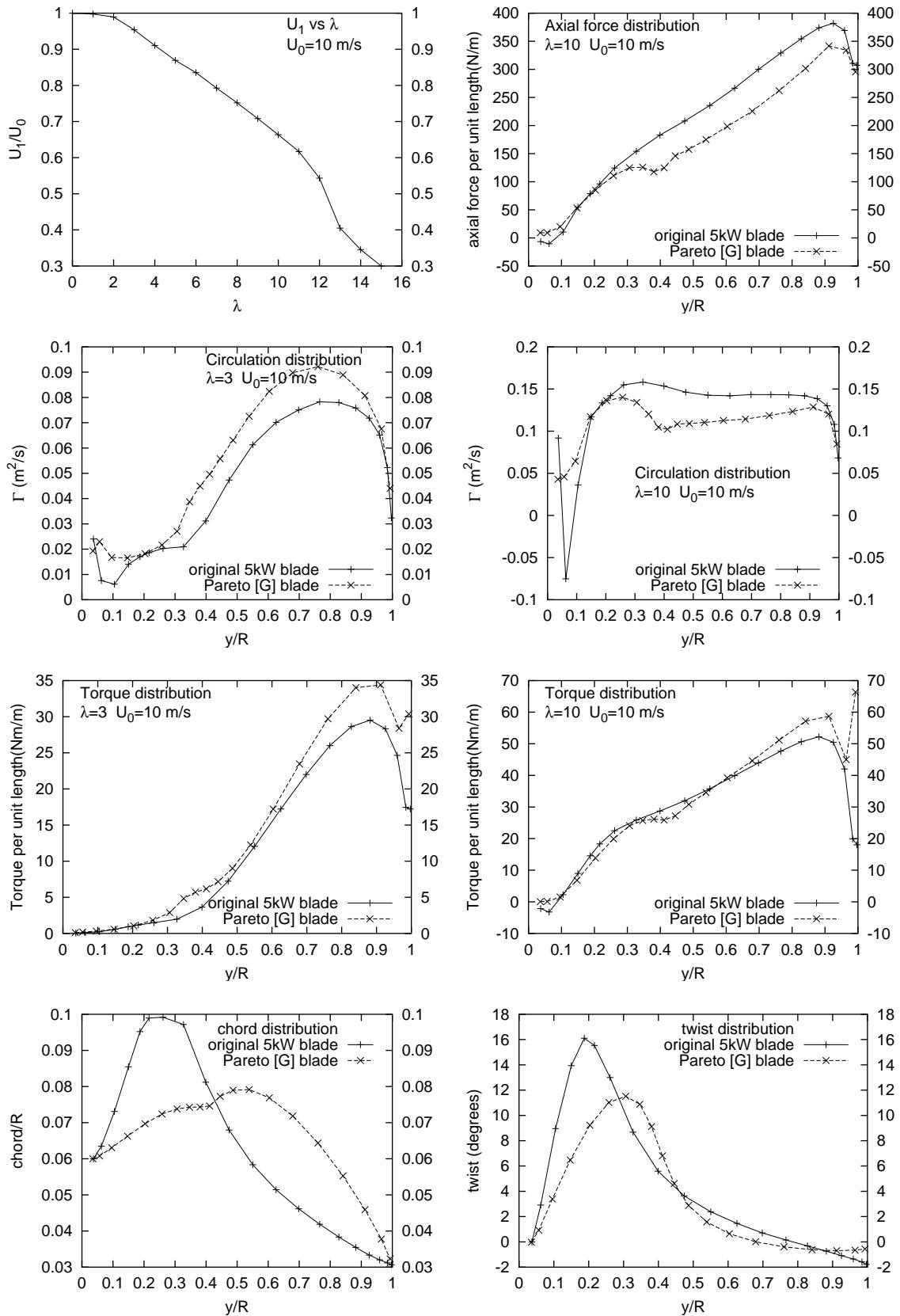


Figure 6.61: Performance results for Pareto [G] blade

### 6.7.17 Pareto set: member [H]

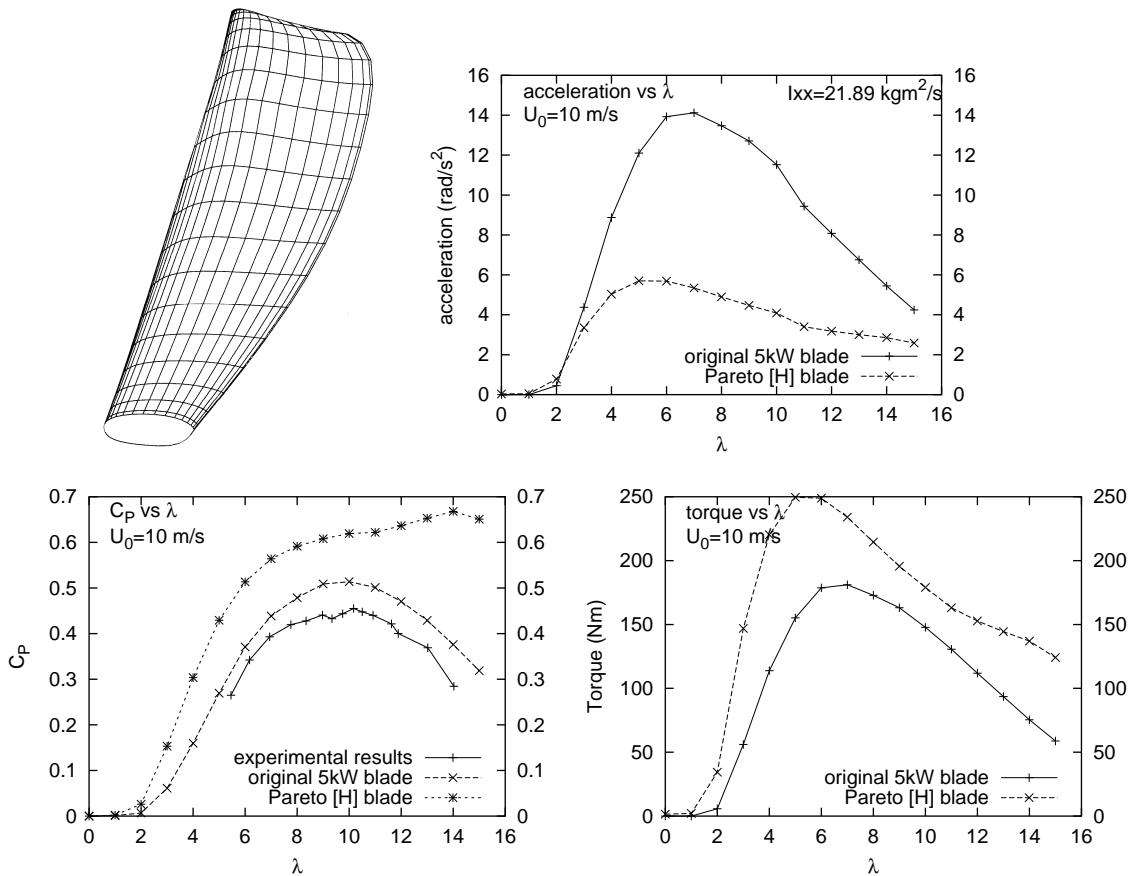


Figure 6.62: Pareto [H] blade - Geometry and performance results

Figure 6.62 shows the meshed geometry of the [H] member of the current Pareto set, together with important performance results. Member [H] is the final member of the Pareto set and belongs to the “*Heavy*” family of small wind turbine blade designs identified by the current project. All Heavy family blades share very similar geometry, the major feature of which is a large “sail-like” chord distribution. Figure 6.63 shows the blade is constructed from a set of very interesting aerofoils. Figures 6.64 and 6.65 display the sectional pressure and span-wise lift/drag distributions at  $\lambda=3$  and  $\lambda=10$ , which also highlights a major flaw in the performance predictions for this blade: trailing-edge pressure equalisation was not attained along a significant portion of this blade’s span, a problem which increases towards the blade tip. This Kutta-condition mis-convergence *may* render the results unreliable, at least when comparing it to blades

whose pressure predictions converged adequately.

At the same time, Figures 6.64 and 6.65 also present a fact that may be a saving grace: the most severe mis-convergences (at both studied tip speed ratios) occurred in only the final *tip-wise* 5% of the span. The chord and twist distributions in Figure 6.66 show why: the blade evolved by opportunistically exploiting a flaw in the error-checking of the aerodynamic objective function - with the result that high- $\lambda$  torques were erroneously over-predicted. The cause was a very sudden tip-wise geometry change - chord plummets from approximately 228 millimetres to 125 millimetres in the last 124 millimetres of the tip. Over the same interval, the twist changes by approximately  $15^\circ$ . The panel method cannot hope to handle these sharp geometry gradients, especially when the rather coarse surface discretisations employed here are used. The saving grace comes from the fact that the *influence* of these inaccuracies is also localised to the final 5% of the blade span. This is demonstrated by the  $\lambda=10$  torque distribution in Figure 6.66.

Even after discounting the value of these prediction results for use in comparisons with the existing 5 kW Newcastle blade at tip speed ratios greater than, say,  $\lambda=6$ , the calculated performance of this “Heavy” blade is what might be expected from such a geometry: superior torque at low tip speed ratios, very good power production and very poor starting acceleration because of the huge second-moment of inertia. The existence of the blade and its presentation here is worthwhile for no reason other than the evolved aerofoils presented in Figure 6.63 and displayed in more detail in Figures 6.64 and 6.65 are very “nice-looking” aerodynamic shapes. It should be remembered that their distant ancestor was a NACA 0012, an aerofoil which, if inserted in place of these shapes in the current blade geometry, would certainly not result in a blade with better performance.



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

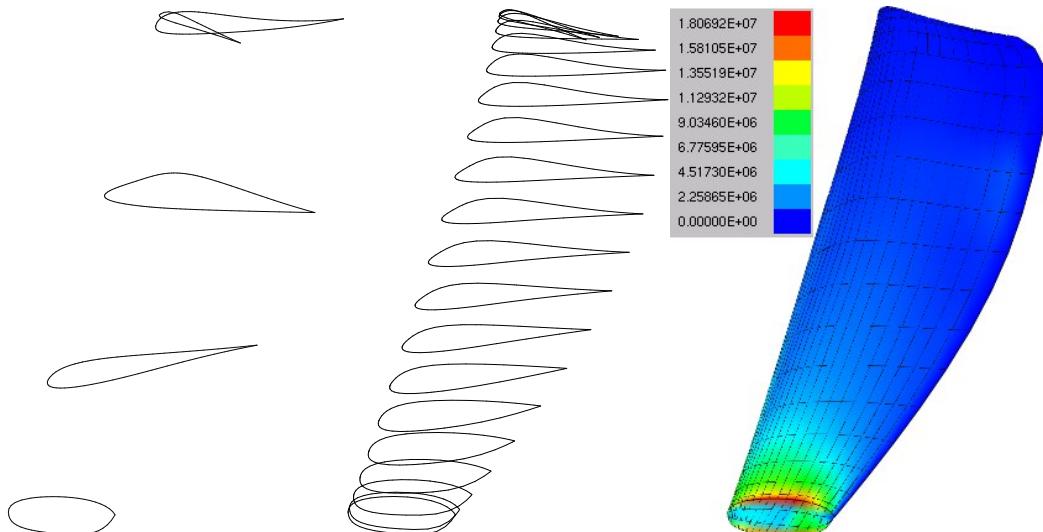
6.63.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
2	40	20	5	20

6.63.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
21.8859	0.6194	7321	993	150.94	3.145	13.49

6.63.3: RESULTS



6.63.4: B-spline aerofoils

6.63.5: Radial

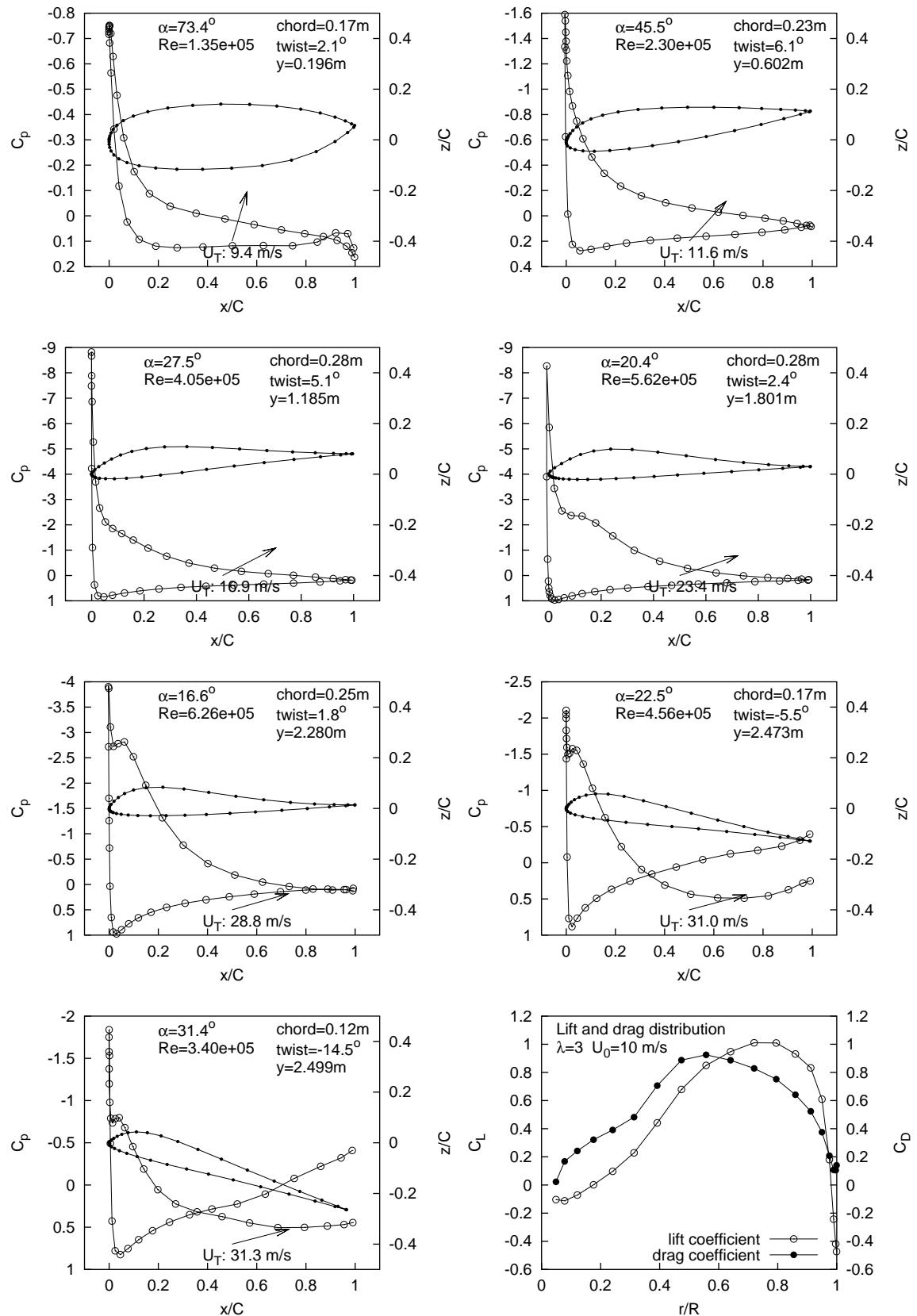
x-sections

6.63.6: von-Mises

equiv. stress

Figure 6.63: Pareto [H] blade - Geometry and performance results (continued)



Figure 6.64: Pressure and force distributions along Pareto [H] blade @  $\lambda=3$

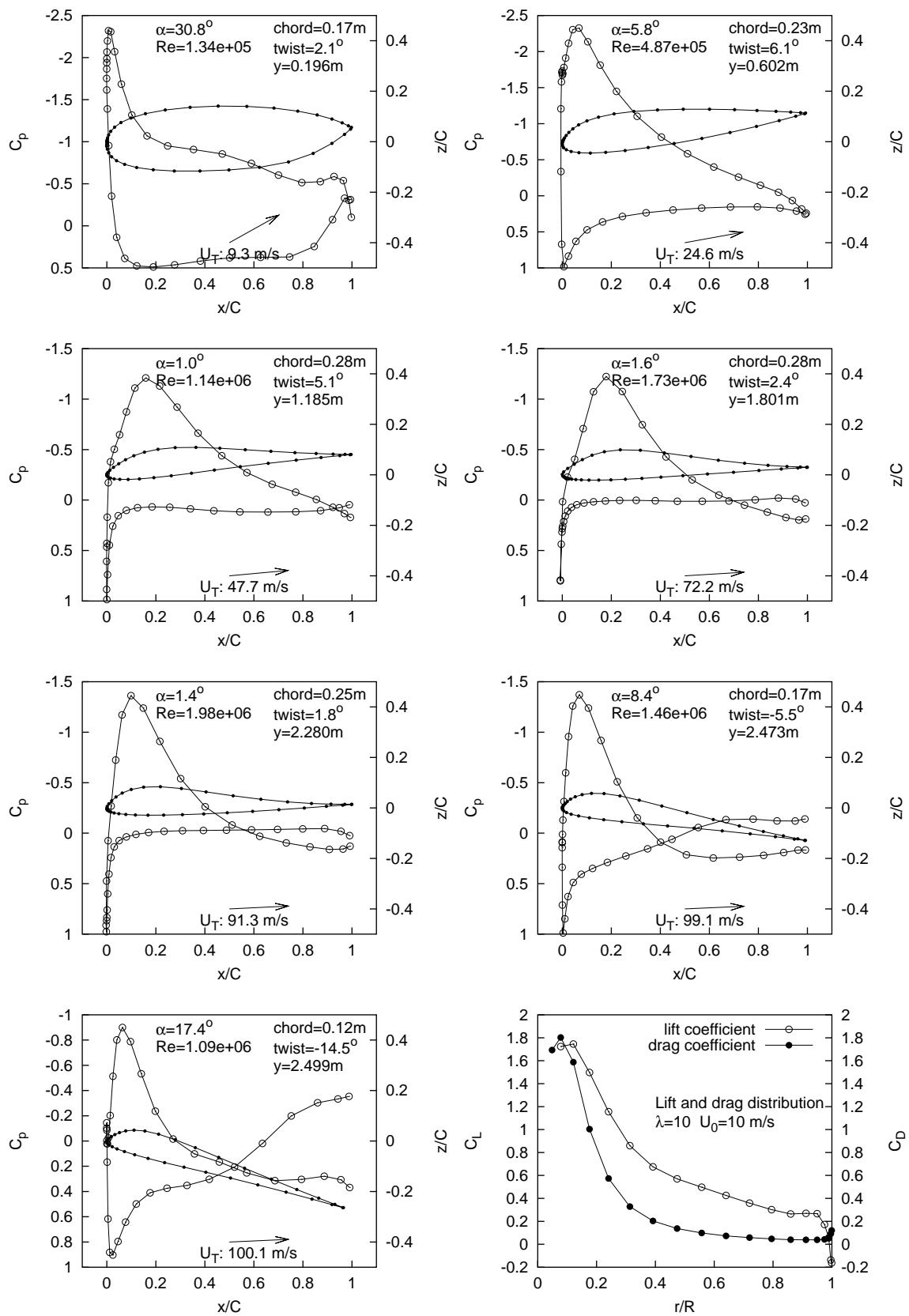


Figure 6.65: Pressure and force distributions along Pareto [H] blade @  $\lambda=10$

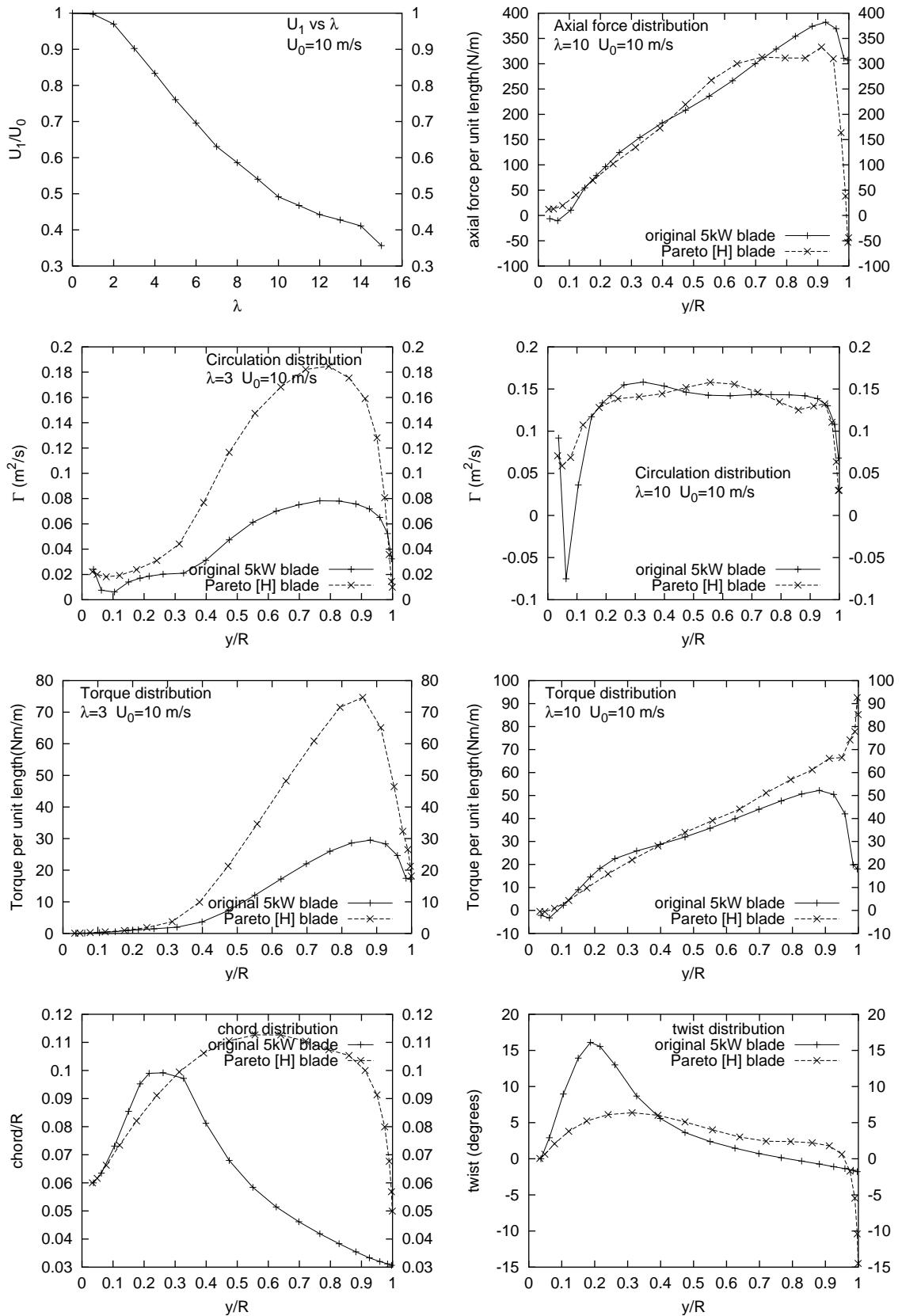


Figure 6.66: Performance results for Pareto [H] blade

## 6.8 Discussion

The results of this optimisation run seem to indicate that a strong negative correlation exists between starting performance and peak-power production. That is, better power extraction very definitely comes at a cost of weaker starting performance (and vice versa), an observation supported by the trade-off surface shown in Figure 6.21.

A dominant design strategy for improving starting performance seems to be to *lessen blade second-moment of inertia whilst maintaining good starting torque*. This may seem to be a trivial assertion, but the potentially-valid counter-strategy of improving starting torque by increasing sectional chord lengths (with a related increase in second-moment of inertia) *does not* seem to be a workable option, or at least an option for producing optimally accelerating designs. Lessening blade second moment of inertia is such a dominant survival strategy that earlier trial runs of the optimisation algorithm resulted in blade cross-sections which bore no resemblance to a finite-thickness “aerofoil”: very thin plates with camber were favoured. This would seem to be the preferred evolutionary route, since the lift produced by cambered plates, whilst inferior to that provided by “real” aerofoil sections, is complemented by very low enclosed volumes and second-moments of inertia, which leads the evolutionary algorithm to favour larger and larger cambered “sail-like” blades. This design route was abandoned on the grounds that thin shapes constructed from low-density materials (like fibre reinforced plastic) could not be expected to cope with the fatiguing rigours faced by real wind machines in constant use, with the possibility of using stronger, tougher and invariably denser materials negating the value of slenderness. A lower-limit was thus placed on the thickness of the evolved span-wise B-spline curves used to represent the blade surface, a strategy which seems to have been successful. Most of the cross-sections through evolved blade surfaces resemble finite-thickness “aerofoil” sections with quarter-chord thickness very close to the 10% minimum threshold. A second inertia-reduction strategy seems to involve a reduction in chord near the hub. The evolved Pareto [D] blade, for example, is a “standard” blade design with tip-wise chords closely resembling the current Newcastle 5 kW blade, but with substantially reduced hub chords. The resulting blade geometry has a second moment of inertia approximately 25% lower than the current blade design, but with roughly equivalent torque figures across the range of studied tip speed ratios - including “starting” torque.

It is evident, in fact, that taper distribution is the major feature distinguishing one Pareto “species” from another, with very slender blades invariably being those with better starting performance.

Power extraction performance seems to be maintained (if not actively improved) by obeying the general taper/twist rules of classic wind turbine blade design. That is, better power production is seen in blades with relatively large chords/twists towards the root which diminish gradually toward the tip. The evolved Pareto [D] blade is again perhaps the best example of this design philosophy in the current study, with chord and twist distributions being qualitatively similar to the existing 5 kW blade design. This is to be expected: the design of modern wind turbine blade designs has been an iterative process spanning the last several decades - a de facto, conscious “evolutionary optimisation” if you will - with the result that peak power extraction is probably very close to optimal in current state-of-the-art blade designs. In fact, taking the predictive accuracy of the current aerodynamic solution method into account, the performance point representing the existing 5 kW blade design in Figure 6.21 is probably closer to the “real” Pareto front (along the  $C_p$  axis) than indicated. It is likely that a blade which shares this taper/twist distribution philosophy but which, through reductions in second-moment of inertia and a “better” choice of aerodynamic section(s), is able to incrementally improve on current starting performance will also be able to maintain state-of-the-art power extraction. The Pareto [D] blade and its close siblings are thus interesting candidates for further experimental study.

The aerodynamic cross-section of evolved blades is the single most influential geometric feature of the evolved blades. One of the major strengths of the current optimisation procedure is that the “aerofoils” evolved satisfy the constraints of being slender (low  $I_{xx}$  with finite-thickness) whilst also performing very well aerodynamically. It should be mentioned that these aerofoil shapes are *smooth* curves which, for the most part, would probably lead a casual observer to believe that they were “designed” for the purpose. This issue needs to be emphasised: previous studies employing evolutionary optimisation to evolve aerofoils have resulted in curve shapes that were “irregular”. Jones et al. (2000), for example, presents a range of evolved aerofoils that meet the desired performance requirements, but are also very “bumpy”, and quite unlike the familiar streamlined shapes expected of aerofoils. Their aerofoil curve representation scheme used two cubic B-spline curves (upper and lower surfaces) defined by a to-

tal of twenty control points - many more than the current method - a decision which undoubtedly sprang from the desire to represent the widest possible range of curve shapes. This raises a possible criticism of the current method: the apparent smoothness and regularity of the evolved blade cross sections stems from the use of very few defining B-spline control points - seven, in fact. Could it be that this compactness runs the risk of ignoring small, fundamental geometry changes, thus destroying its value as an aerodynamic modelling technique? The blade cross sections depicted in the current Pareto set show that the compact geometric modelling technique presented here is capable of representing a very wide range of aerofoil curve shapes, and that these are indeed more than capable of meeting the aerodynamic requirements of the current application. This success, whilst seemingly trivial, is an important result of the current project. Specifically, the flexibility of cubic B-spline curves *should* be leveraged by ruthlessly *reducing* the number of control points (and knots) in curves representing aerofoil shapes undergoing evolutionary optimisation, for two complementary and important reasons: it virtually guarantees that the curves will be smooth and regular; and it ensures that the object vector length is at a minimum.

The Ordinary-Heavy and Heavy evolved species are obviously pathological designs. Their very existence highlights a problem with the current optimisation methodology: the use of low-density aerodynamic evaluations (in this case, with a  $30 \times 10$  panel mesh) can result in the emergence of evolved designs whose *evaluated* performance does not bear close resemblance to their actual performance capabilities. A “successful” blade evaluation was judged, during the current project, on the basis of how close the performance predictions of the low-density panel mesh models ( $30 \times 10 \times 5 \times 10$ ) matched the predictions of the same blades discretised with a higher-density panel mesh ( $40 \times 20 \times 5 \times 20$ ). The three “lightest” species of blades pass this test well with, for the most part, well-behaved, converged aerodynamic solutions being the norm and  $C_P$  figures at  $\lambda=10$  consistently around 10% too high (sometimes exceeding the Betz limit) for the low-density solutions. The two “heavy” species fail the test, with fast, unpleasant geometry changes near the blade tips evaluating very badly under the higher-density evaluations. Three possible solutions for this problem are thus: evolve blades using higher-density evaluations; develop better geometry generation/checking routines to weed out pathological designs before they even get a chance to be evaluated; and develop more fault-tolerant aerodynamic evaluation routines.

The largest possible criticism of the current project is that no attempt has been made to replicate the 5 kW design results over many different optimisation runs. A standard way to verify stochastic optimisation techniques is to complete *many* identical runs (on the order of 1000) and average the results. Constraints on time during the current project prevented this type of statistical validation. The 5 kW optimisation run can thus probably best be described as a demonstration of the capabilities of the current optimisation technique, rather than a definitive exposition of the true 5 kW Pareto blade set. In this context, the method can be seen to be very successful: blade designs were developed which exceeded the performance of current state-of-the-art geometries, a result which points to the likelihood of the technique meeting the requirements of rigorous future validation tests.

Two common evolved geometry features may raise certain operational and manufacturing concerns: “sharp” leading edge geometries seem to be quite popular in the evolving populations, and some designs seem to favour a “cusped” trailing edge. The sharp leading edge phenomenon can easily be addressed by culling potential blade designs which exceed some leading-edge “roundness” criterion. This is expanded upon in the following section, and used in the subsequent 600 Watt blade optimisation run. The cusped trailing edge issue was not addressed here since it did not offer problems to the aerodynamic solver. This type of geometry may be difficult to manufacture successfully, and users hoping to create physical models of blade designs may wish to devise a simple method of culling blades with cusped trailing edges before they have a chance to enter the evolving population.

It should also be emphasised that the computational blade models employed in the current project used vastly simplified “materials” to those that would be employed in a physical blade. The fibre-reinforced plastic “skin” and foam “core” were modelled as bulk materials - a far cry from the Resin Transfer Moulded 5 kW Newcastle blade with its complex layup of different types of glass and carbon fibre reinforcement. It is obvious then that the second moment of inertia of any optimised blade design will be dependent on the process and materials used in its manufacture. Users hoping to evolve blades with strongly predictable inertial properties will thus need to develop blade material models mirroring the output of their chosen manufacturing processes. A related issue is that the process and materials chosen to manufacture a particular blade geometry will likely *be dependent on that geometry*. For example, the cusped

pressure-side of some blade cross-sections may cause mould release problems in some processes, very thin aerofoil sections may prove difficult to manufacture with finite-thickness fibre plys and small trailing-edge angles may be impossible to create without resulting in very fragile blades. Users with a clear understanding of the constraints of their possible manufacturing processes can thus formulate geometric and performance constraints, and incorporate these into the current optimisation method. The extreme flexibility afforded by both B-spline curves and surfaces and Evolutionary Optimisation techniques make the inclusion of geometric constraints tailored for a particular end-use simple.

## **6.9 Blades and aerofoils: aerodynamic differences**

The justification for employing a relatively expensive three-dimensional aerodynamic solver for use in the current project, rather than a faster two-dimensional method (such as the Blade Element Method), was based on the assertion that cross-sections through three-dimensional blades and the aerofoils they resemble have *different aerodynamic properties*. Considerable evidence exists to support this.

The complex effects of *rotation* on lifting surfaces have been studied by numerous researchers hoping to explain the physical observation that wind turbine blades often produce *more* power than is predicted by methods that employ two-dimensional empirical aerofoil performance data. The impetus for these studies is the phenomenon of *stall delay*, or the tendency of the flow on a rotating turbine blade to remain attached at Reynolds numbers and angles of attack at which similar aerofoil flows separate. Stall delay was first reported by Himmelskamp (1945) in his study of the sectional aerodynamics of aircraft propellers. It has subsequently been suggested that separation and stall are delayed because the boundary layer of a rotating lifting surface undergoes both thinning in the radial direction due to centrifugal forces, and chord-wise acceleration due to the Coriolis effect. Stall delay remains a complex, vigorously studied yet poorly understood phenomenon. Recent reviews are provided by Burton et al. (2001) and Schreck et al. (2000). A defining feature of stall is the “loss of leading-edge suction peak” as seen on chord-wise pressure coefficient plots. Barnsley and Wellicome (1992) provides a graphic example of the differences between stalled and unstalled pressure profiles along a 1 metre test blade. Between a range of tip speed ratios for this

turbine ( $2 \leq \lambda \leq 4.2$ ), stall can be seen to progress from tip to the root with decreasing  $\lambda$ , with stall characterised by a “flat” region of near-constant suction extending from the leading edge, without a well defined leading-edge suction peak. Stall delay is thus characterised by pressure distributions which exhibit good leading-edge suction peaks and smooth, gradual, pressure-recovery slopes towards the trailing edge. Ronsten (1992) showed that leading-edge suction peaks were maintained at 30% and 55% span positions on a rotating wind turbine blade studied at a single  $\lambda$ , and that this contrasted greatly with the measurements on the non-rotating blade. Specifically, the non-rotating results at these span positions were heavily stalled, with no leading-edge suction peak and a remarkably flat suction profile from leading-edge to trailing edge.

Despite the uncertainties, useful results have appeared which suggest that empirical models of the stall delay effect may be possible, and that these potentially could be applied to correct physical aerofoil data or the calculated results of numerical prediction techniques. Du and Selig (1998) and Du and Selig (2000), for example, were based on the results of Snel et al. (1994), and presented a simplification of the three-dimensional incompressible boundary layer equations as an algebraic correction to sectional lift and drag coefficients for a rotating aerofoil as a function of sectional solidity. The method may be capable of providing useful quantities such as the location of separation on the rotating blade and an estimate of boundary layer thickness, and is perhaps a starting point for the development of a boundary layer extension to a three-dimensional inviscid panel code. A review of stall delay prediction is provided by Duque et al. (2000).

Rotation and its effect upon the boundary layer and inviscid three-dimensional flow field may be but one of many physical phenomena contributing to the differences between two-dimensional and three-dimensional aerodynamic performance of wind turbine blades. Madsen and Christensen (1990), for example, showed that significant aerodynamic differences exist between a wind turbine blade and the aerofoil section from which it was constructed in *both* stationary and rotating conditions. They reported that the effect of rotation was “minor” for the 100 kW turbine studied, and stated that “three-dimensional flow effects caused by the finite aspect ratio of the blade and spanwise pressure gradients along the blade surface” contributed most significantly to the measured performance differences. Interestingly, these effects occurred in both stalled and unstalled conditions, which would seem to suggest that the “three-dimensionality” of the blade surface (and thus three-dimensionality of the flow field) was heavily im-

plicated.

Stall, separation, and stall delay are physical processes associated with the boundary layer of a lifting surface, and are thus inherently viscous phenomena. The danger exists, then, that all two-dimensional and three-dimensional aerodynamic differences related to wind turbine blades will be attributed to viscous effects. This is not necessarily the case, with inviscid pressure distributions in the current project and elsewhere demonstrating that the *geometry* of the three-dimensional blade and the trailing helical wake cause significant performance differences compared with similar aerofoil cases. Wood (1991), for example, studied panel-method simulations of a small wind turbine blade (using a three-dimensional solver) and the aerofoil section on which the blade was based (using a two-dimensional solver), and showed that significant differences exist between the two geometries in the leading-edge suction peak at similar angles of attack. Specifically, it was suggested that the consistently *reduced* leading-edge suction peaks closer to the hub as predicted by the three-dimensional solver led to stall delay, and that this was brought on by the increasing solidity. The mechanism for this change was most likely the influence of the trailing vorticity, this being significantly different (and its interaction with the blades significantly more complex) than the influence of the aerofoil's trailing wake. The implication is that the three-dimensionality of the rotor/wake system causes inviscid surface velocities which differ from those of the two-dimensional case. This suggests that a three-dimensional aerodynamic solver is perhaps an appropriate choice for studying wind turbine performance, even the simple inviscid code employed in the current project. Importantly, inviscid flow in the radial direction would seem to be an inherent feature of wind turbine blade flows, and gives rise to the "centrifugal" and "cross-flow" surface velocities which presumably account for the viscous effects of boundary layer thinning and separation-point delay reported in the stall-delay literature. A three-dimensional inviscid solver with an accurate empirical correction for stall delay (as a function of blade solidity) would thus seem to be a good choice for the next generation of wind turbine performance codes.

In an effort to verify that the use of a three-dimensional aerodynamic solver did indeed generate different results for two-dimensional and three-dimensional geometries, the current panel method was applied to the 5 kW Newcastle blade (the "original" 5 kW blade analysed earlier) to study the surface pressure distributions over the three-dimensional blade and the (two-dimensional) SD7062 aerofoil on which this blade is

based. These pressure distributions are compared in Figure 6.67. As with the two-dimensional code validation studies presented in Section 4.9, the SD7062 aerofoil was modelled as a straight “wing” with very long aspect ratio. The surface pressure distribution at the centre of this wing’s span will approach that of the aerofoil as the aspect ratio tends to infinity. To ensure that the empirical viscous correction was applied to similar flow conditions, the two-dimensional “wing” was scaled to the chord of the “blade element” to which it was compared. Similarly, the flow speed for this wing ( $U_0$ ) was matched to the  $U_T$  of its comparable blade element and its angle of attack set to the blade element’s  $\alpha$ . As with the previous two-dimensional study, a very long planar trailing wake mesh emanating from the wing’s trailing edge and aligned with the free stream was added to the wing mesh model. Comparisons were performed for the two operational conditions studied for the rotating blade: at  $\lambda=3$  and at  $\lambda=10$ . The three-dimensional “blade” simulation results appear on the plots labelled as “3D”, whilst the two-dimensional “wing” results are labelled “2D”.

It is obvious from the comparisons that the “blade” pressure distributions are rather different to those of the SD7062 aerofoil, although the reduction in leading-edge suction peak for span positions greater than 65% for both of the rotational speeds studied seems to contradict the results of Wood (1991), whose suction-peak calculations increasingly differed toward the blade hub. The point is made, however, that 2D/3D aerodynamic differences exist. The implication for the evaluation of wind turbine blade performance, at least in the current application, is clear: two-dimensional aerofoil performance data, whether empirically gathered or numerically determined, differ from the results of a three-dimensional aerodynamic solver, even when both solvers are inviscid. The simple empirical viscous correction employed by the current code did not substantially alter these discrepancies. It is possible that the three-dimensionality of the blade geometry may influence this difference: the discrete surface pressure evaluations on the blade were taken at the collocation points of the strip of panels comprising each blade element, with the possibility of the “aerofoils” at each end of these panel strips having different chords and twist angles. The interesting point to make about this, however, is that the greatest differences between the two-dimensional and three-dimensional evaluations occur *nearer to the blade tip*, where chord and twist variations are minor compared with those closer to the hub, thus giving rise to blade elements which are comparatively uniform along their span. As with the results of Wood (1991),

the helical wake is thus likely to be the reason for the disparities, suggesting that any numerical method, whether two-dimensional or three-dimensional, hoping to simulate the performance of wind turbine blades absolutely must account for the effects of the *geometry* of the trailing vorticity. This is a task greater than the “conservation of mass” wake models ubiquitously employed in the Blade Element Method, which use a control volume simplification to balance the blade and wake torque and thrust loads. A panel method that models the complex three-dimensional geometries of both the blade *and* the trailing wake, as used here, is thus well placed to also model the complex *interaction* between blade and wake that probably gives rise to the differences between blade and aerofoil pressure distributions which contribute to stall delay.

A similar comparison of two-dimensional and three-dimensional surface pressures was conducted for one of the promising evolved geometries: the Pareto[D] blade (page 236). An additional step was necessitated by the non-standard section along the blade: unlike the “original” 5 kW blade design which used a single SD7062 aerofoil, every evolved blade has a significantly different blade cross sections at any point along its span. To enable a two-dimensional geometry to be successfully compared with its blade element counterpart, the following procedure was employed: the position of each blade element collocation point was tabulated; the “aerofoil” that these discrete points represented was then rotated back to zero pitch and scaled to unit-chord; and the versatile Differential Evolution B-spline curve fitting technique described in section 3.4 was applied to a “simplified” SD7062 B-spline aerofoil curve (a curve with just three knots and seven control points), with the position of the control points evolved to create a curve-fit to each Pareto[D] aerofoil surface co-ordinate set. A selection of these new B-spline aerofoil curves appear in Figures 6.68 to 6.74, alongside the standard (non-simplified) SD7062 aerofoil for comparative purposes. These aerofoil sections were scaled and rotated to match their blade element counterparts in the Pareto[D] blade, and the procedure used to evaluate the aerodynamic performance of the SD7062 aerofoil (as described above) was applied to each. Comparisons between the three-dimensional blade surface pressures at each corresponding blade element (evaluated at the “operational-speed” condition,  $\lambda=10$ ) and the predicted two-dimensional surface pressures also appear in Figures 6.68 to 6.74.

The most significant feature of the pressure distribution comparisons is, again, that the two-dimensional and three-dimensional evaluations are quite different, and the as-

sertion that a three-dimensional aerodynamic solution routine is the most appropriate choice in the current application seems valid. However, it is evident that the reduction in leading-edge suction peak for “blade” evaluations does not generalise to all blade geometries: only the very tip of the blade shows such a pressure reduction. Since the only difference between the evaluations of the “original” 5 kW blade and the Pareto[D] blade is the blades themselves, it is reasonable to state that the “three-dimensionality” of the blade geometry may significantly affect the leading-edge suction peak, sometimes to the detriment of aerodynamic performance. This result was unexpected, and no attempt to account for it was considered during the construction of the geometry evolution routines. It may be possible to use the result as the basis of another type of “geometry check”, so that candidate blade designs with increased sectional leading-edge suction peaks (or, more usefully, whose geometry *predicts* that such increased suction peaks will likely occur) can be eliminated before they get the chance to enter the evolving population. Future researchers hoping to evolve blade geometries with more robust aerodynamic performance characteristics may thus wish to study the effect in more detail.

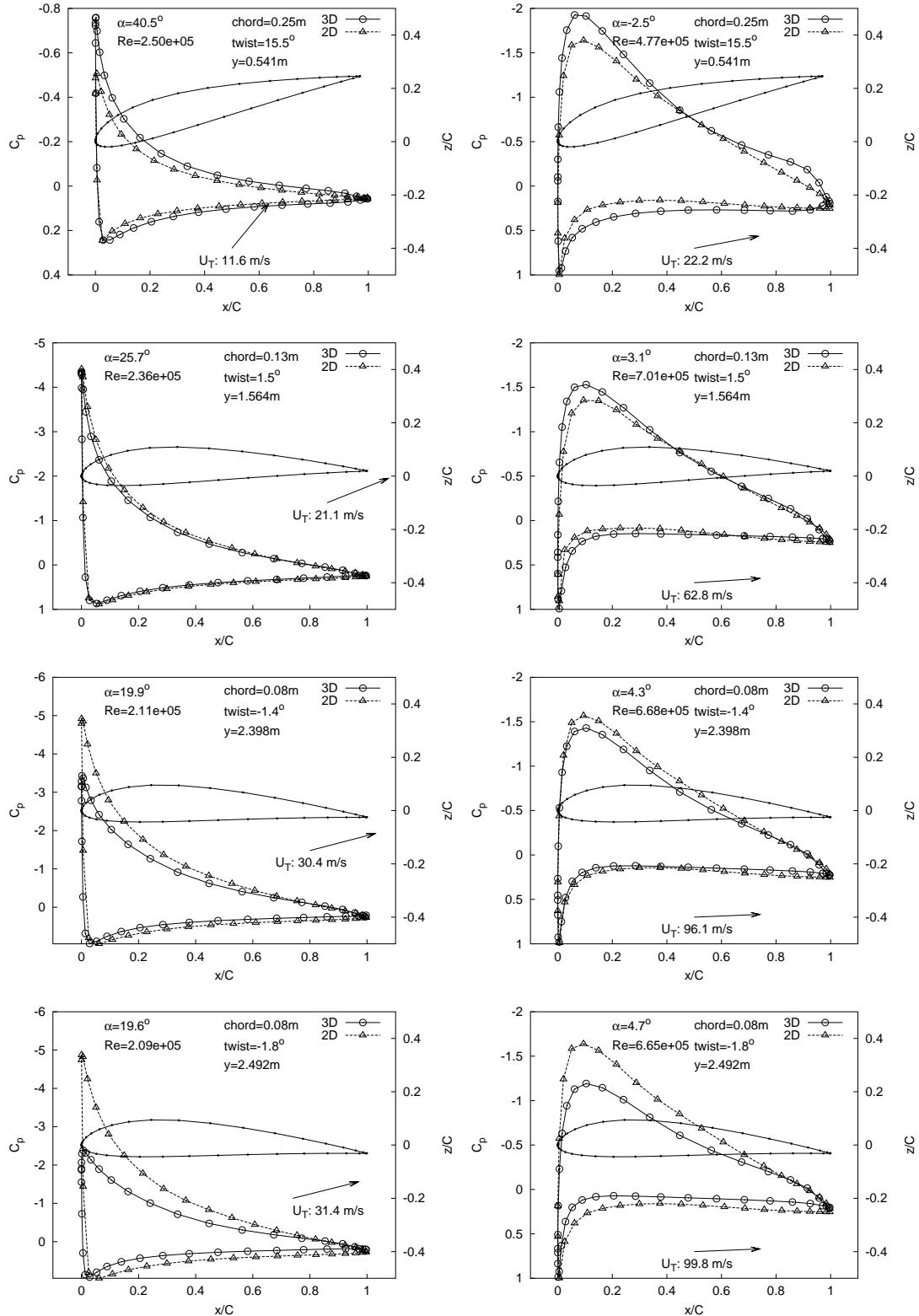
6.67.7:  $\lambda=3$ 6.67.8:  $\lambda=10$ 

Figure 6.67: Original 5 kW blade (SD7062 section): 2D and 3D pressure comparisons

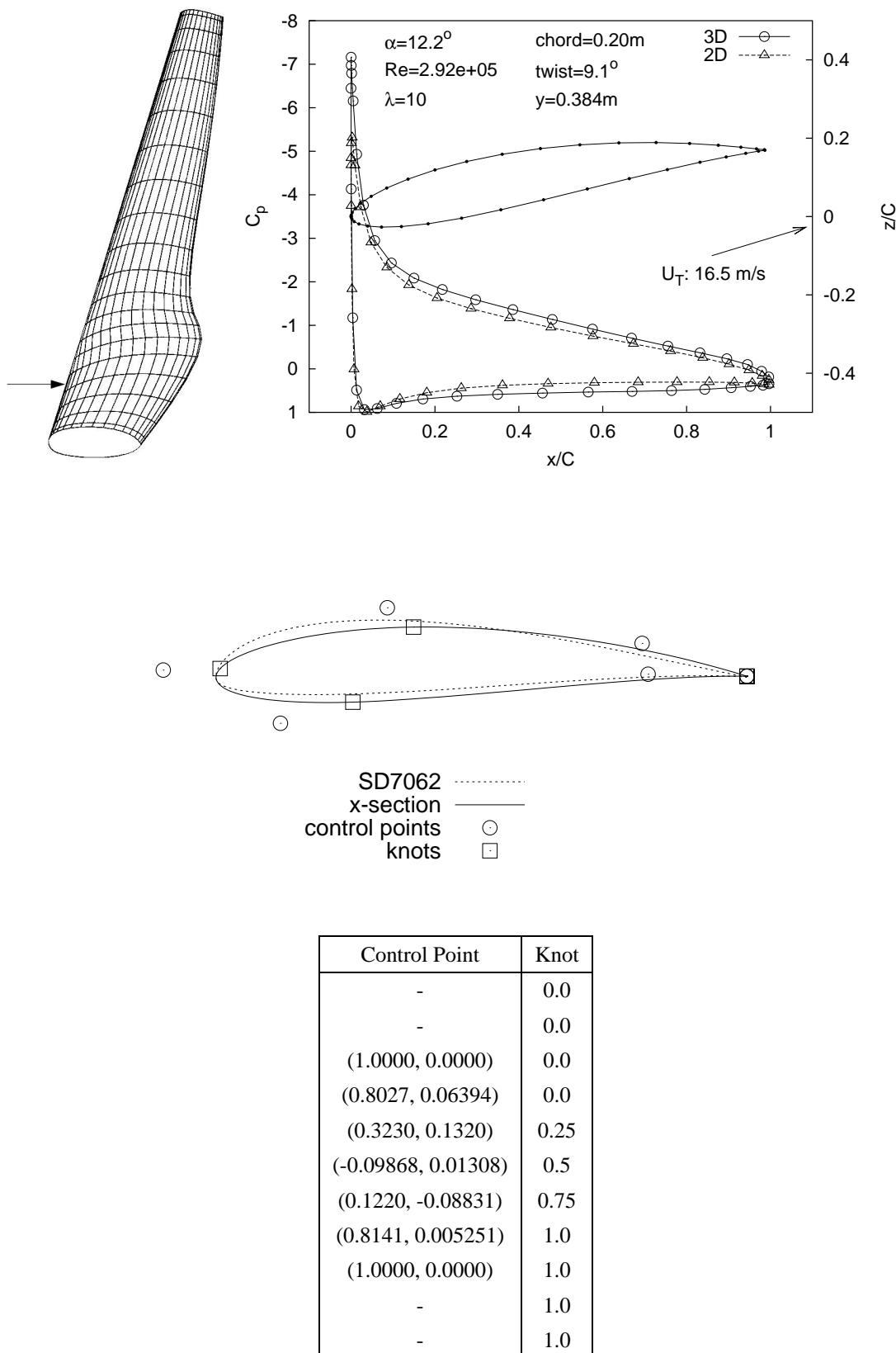


Figure 6.68: Pareto [D] blade: cross-section (1 of 7)

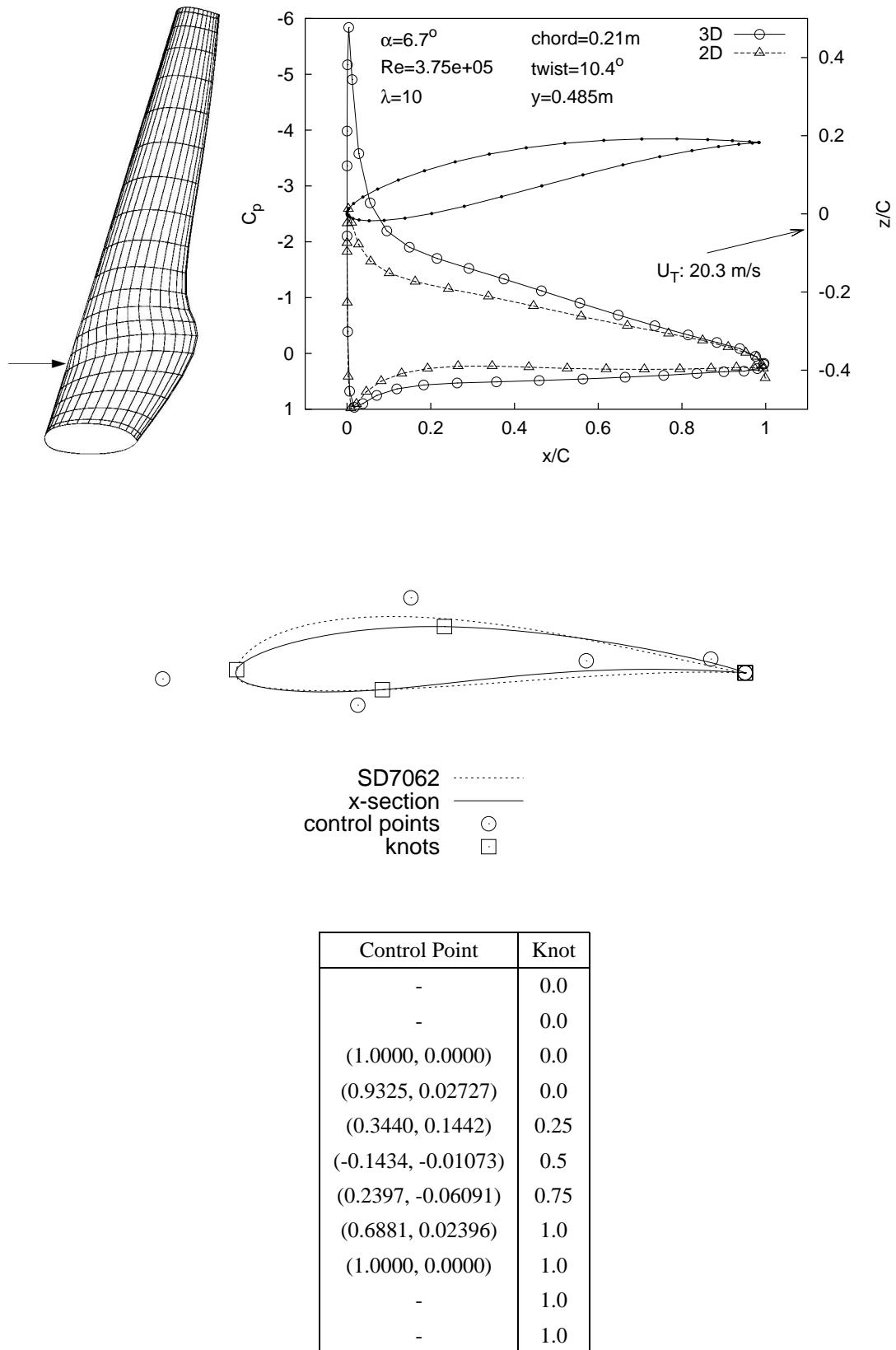


Figure 6.69: Pareto [D] blade: cross-section (2 of 7)

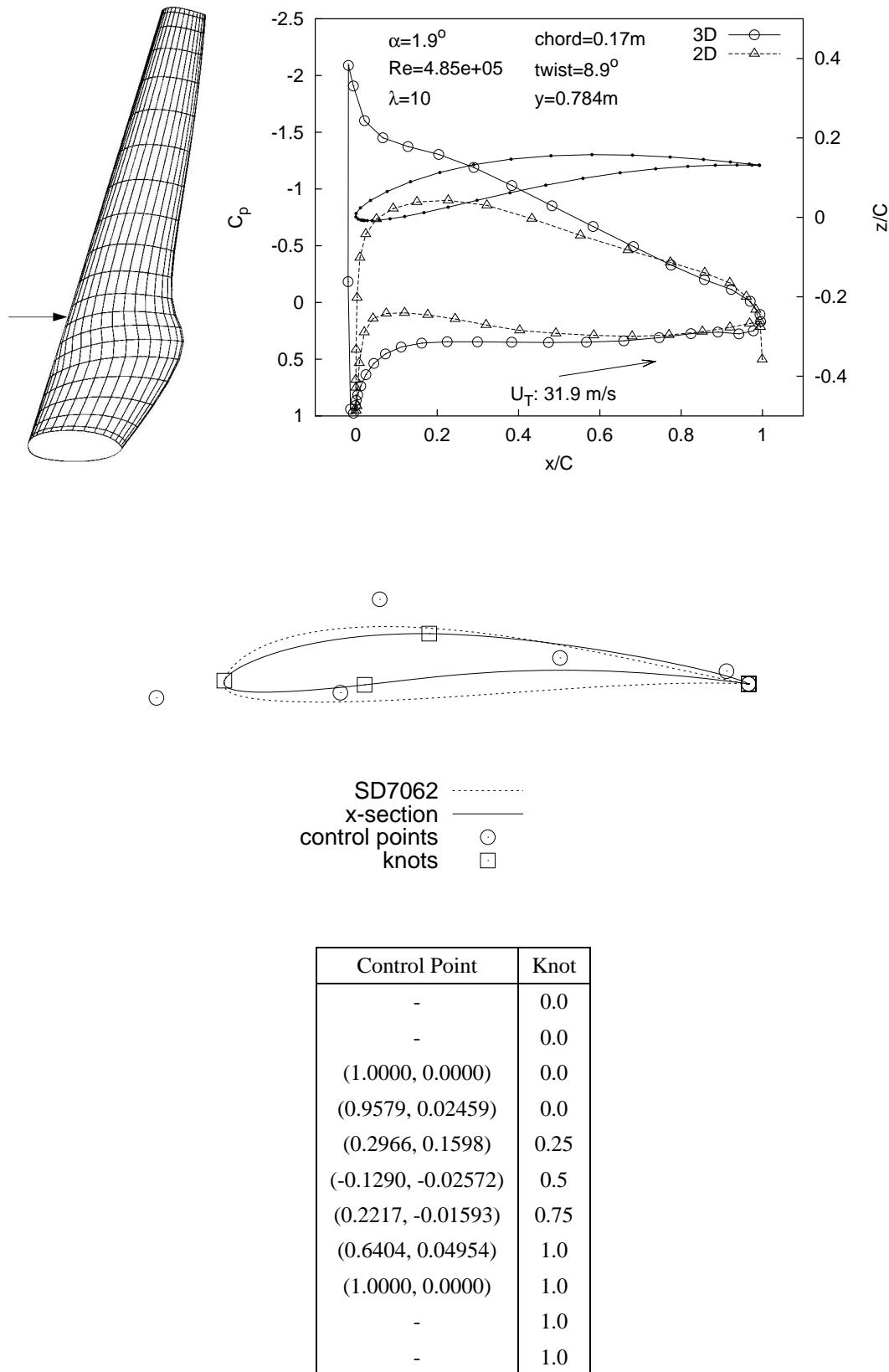
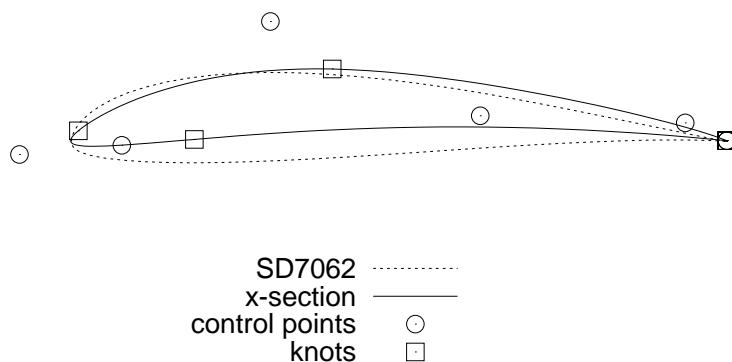
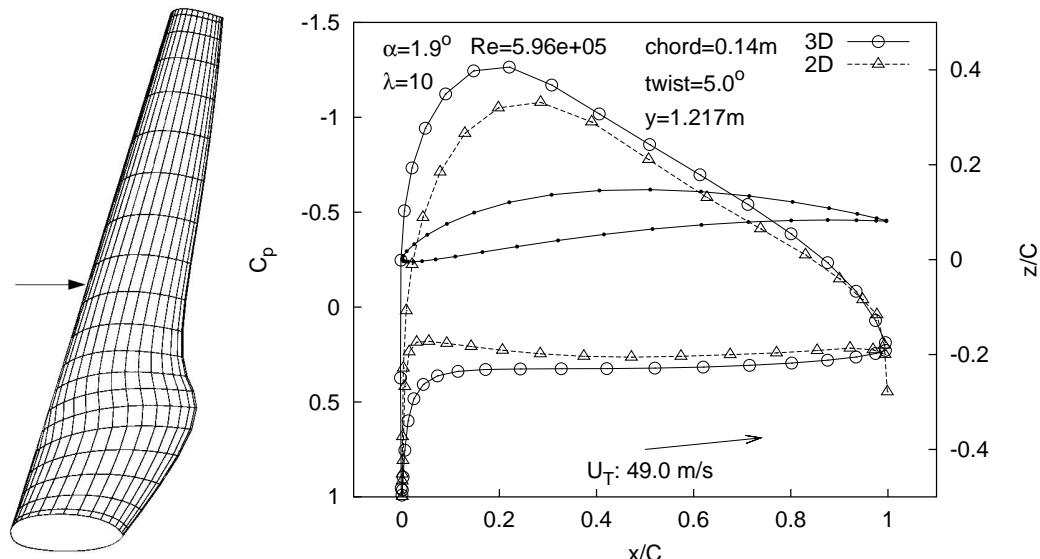


Figure 6.70: Pareto [D] blade: cross-section (3 of 7)



Control Point	Knot
-	0.0
-	0.0
(1.0000, 0.0000)	0.0
(0.9364, 0.02906)	0.0
(0.3037, 0.1888)	0.25
(-0.07892, -0.02069)	0.5
(0.07714, -0.006034)	0.75
(0.6238, 0.04038)	1.0
(1.0000, 0.0000)	1.0
-	1.0
-	1.0

Figure 6.71: Pareto [D] blade: cross-section (4 of 7)

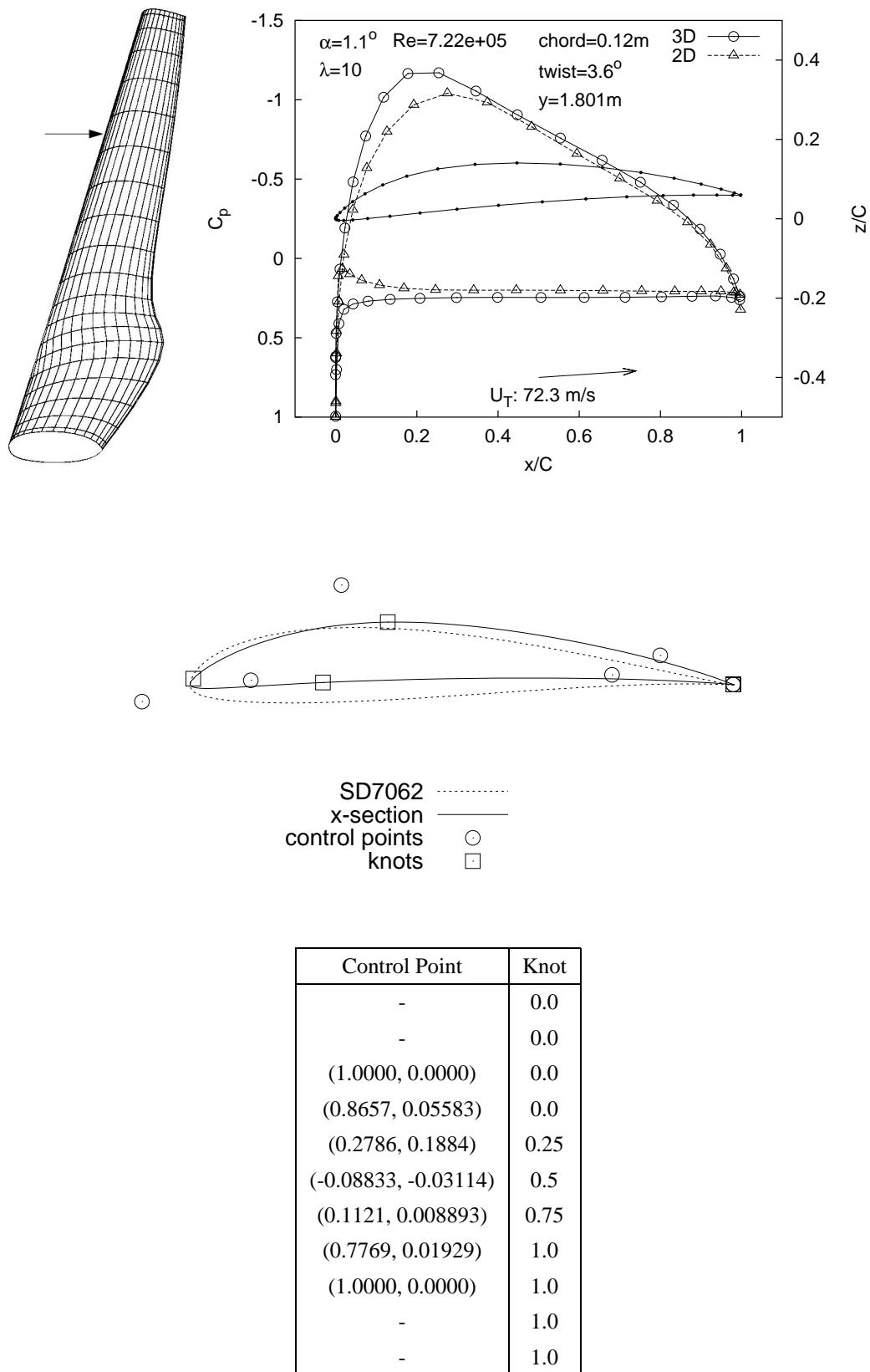


Figure 6.72: Pareto [D] blade: cross-section (5 of 7)

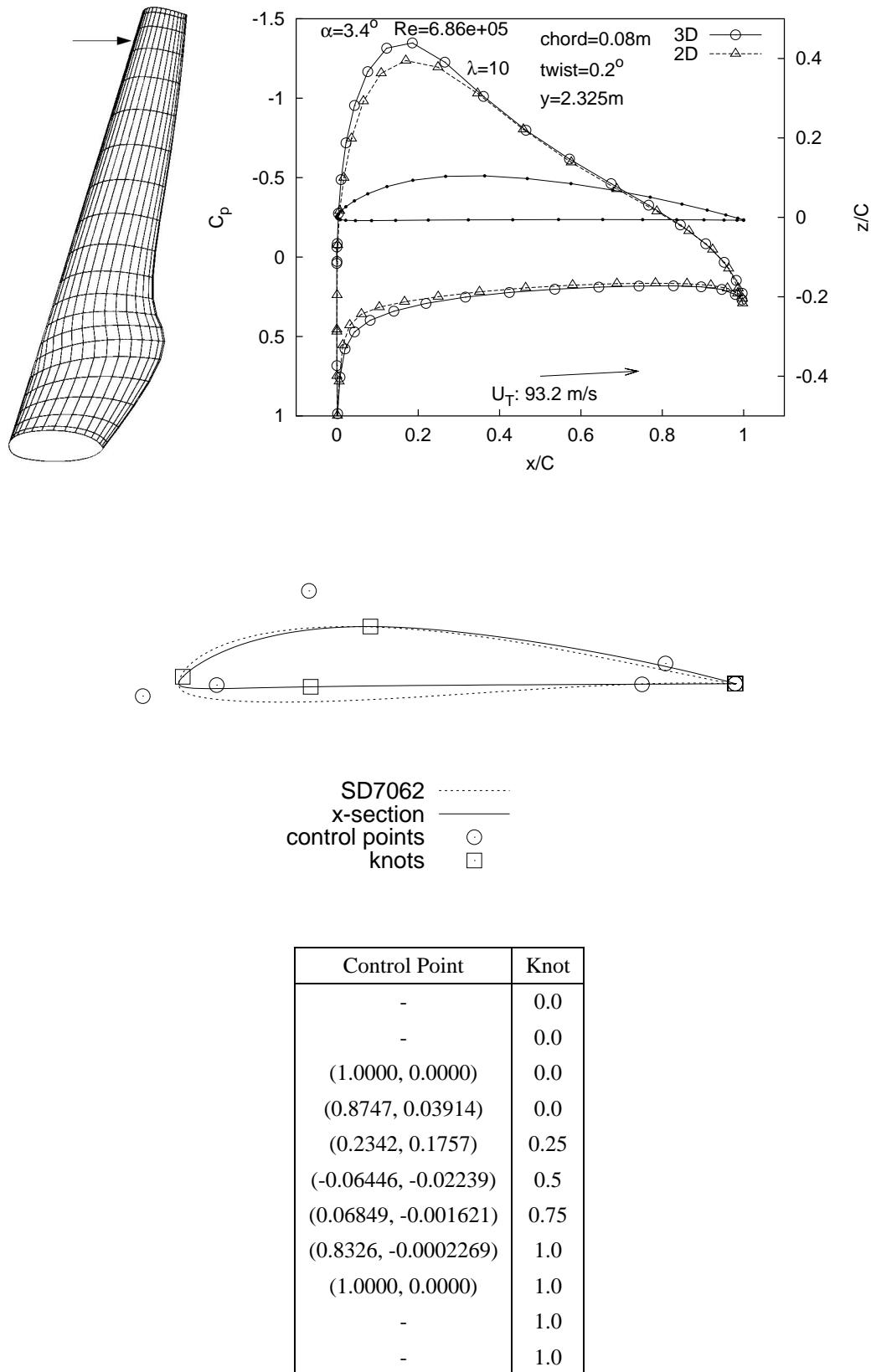


Figure 6.73: Pareto [D] blade: cross-section (6 of 7)

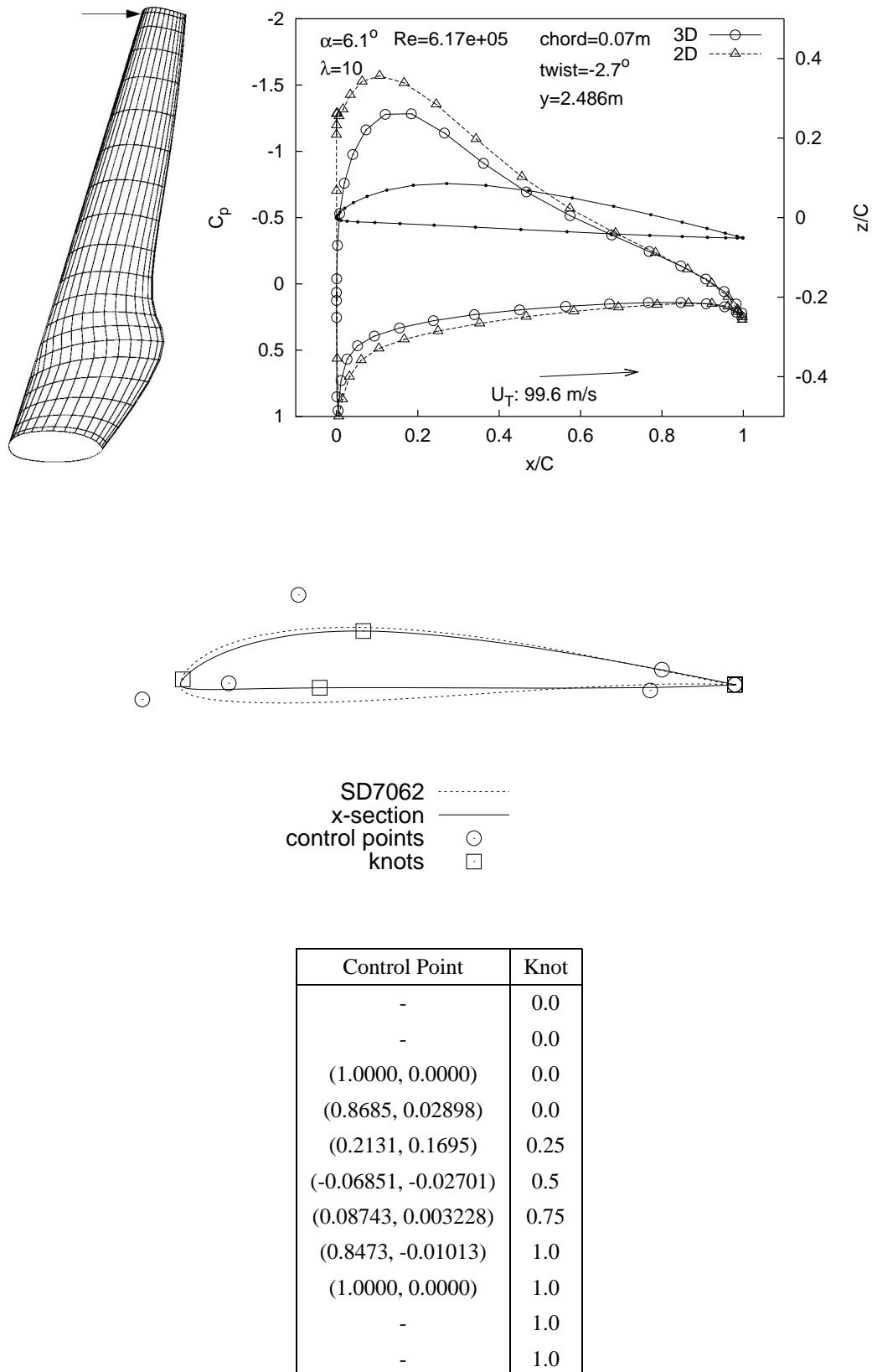


Figure 6.74: Pareto [D] blade: cross-section (7 of 7)

## 6.10 Obtaining a “rounder” leading edge

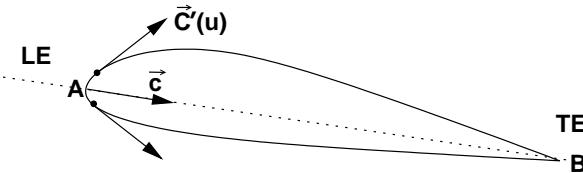


Figure 6.75: Scheme for ensuring a “round” leading edge

Preliminary runs of the evolutionary optimiser highlighted another geometry problem: blades with “sharp” leading-edges seemed to have an artificial survival advantage, perhaps because they favour the prediction of a large LE pressure-spike. These spikes were especially evident at the low- $\lambda$  operating condition. To counter this problem, the leading-edge of the four B-spline “aerofoils” evolved in each blade design was constrained to be “round”, with “roundness” defined via the vectors shown in Figure 6.75. This was accomplished for each curve by first finding the Cartesian co-ordinates of the leading and trailing edges (points A and B in Figure 6.75), with unit-chord vector then formed as

$$\vec{c} = \frac{B - A}{|B - A|} \quad (6.11)$$

Two knot parameters ( $u$ ) offset by 0.01 on either side of the leading edge knot parameter were then defined. These were used to form two unit-vectors representing the first-derivative of the curve at these knot parameters ( $\vec{C}'(u)$ , calculated using the method outlined in Appendix A.1, where  $\vec{C}$  is a two-dimensional B-spline “aerofoil” curve). A LE “roundness” measure at each of these points was then calculated via

$$LE\_roundness = \vec{C}'(u) \cdot \vec{c} \quad (6.12)$$

with values closest to 1.0 giving a LE which is “very sharp”, and values closest to 0.0 giving a LE which is “very round”. The aerofoil side with the largest roundness measure (“sharpest” LE measure) was chosen as the representative score for each aerofoil.

The measure was implemented as a geometry constraint alongside the other “sane geometry” constraints, and was thus used to discard all potential blade designs with a “sharp” LE prior to the expensive aerodynamic evaluation. A maximum roundness measure of 0.4 was arbitrarily chosen after viewing the cross-sectional  $C_p$  predictions

of a number of evolved blades. Aerofoils with a roundness measure below 0.4 tended not to show the large LE pressure spike evident on sharper LE blades. Both the NACA 0012 and SD7062 aerofoils have roundness scores of about 0.2.

## 6.11 Optimisation results: 600 Watt blade

One final small blade design task was undertaken to further demonstrate the effectiveness of the current evolutionary blade design technique. The goal was to improve a current blade design for a one-metre long blade for use on a three-bladed, 600 Watt wind turbine platform currently in development by the Wind Energy Group at the University of Newcastle. This existing blade design was, in essence, a scaled-down version of the 2.5 metre 5 kW blade mentioned previously, with a chord/radius ratio 1.4 times that of the 2.5 metre blade, and an increase in pitch angle of 5°. These changes, along with the use of three blades, were intended to improve the blade's low- $\lambda$  performance at the expense of maximum  $C_P$ . See Clausen and Wood (2000) and Wood and Rowbotham (1999).

Since the emphasis on this particular run was to improve an existing design rather than evolve one from scratch, a simplified version of the existing blade was modelled as a collection of five B-spline aerofoil sections (again, four simplified SD7062 B-spline aerofoil curves and one rounded "hub" section curve), and this used as a "seed" curve for the initial population. Choosing an existing blade design as the initial seed geometry introduces the possibility that the initial population may be very close to a "local" optimum in the search space. This significantly reduces the likelihood of radically different (and perhaps radically better) designs emerging over the course of the optimisation run. This would have been a distinct disadvantage during the primary optimisation run reported on earlier: the extreme diversity of blade geometries in the 5 kW Pareto optimal set was in large part caused by an initial "primordial soup" of remarkably unfit seed blade designs which existed far from the culling influence of local optima.

Each blade rotor in the initial (generation zero) population consisted of:

- Three blades
- Blade length ( $R$ ) of 1 m
- 4 simplified B-spline aerofoils
- $N\_PANELS = 30$
- $M\_PANELS = 10$

- TURNS = 5
- W\_PANELS = 10

with each being a slightly-perturbed version of the seed blade. One normalised real-valued deviate on the range [-1,1] was generated by a Gaussian-random number routine for each parameter to be perturbed. This was then scaled according to the type of parameter to which it was added:

- MAX\_CHORD\_DELTA =  $0.001 \times \text{span}$
- MAX\_TWIST\_DELTA = 1.0 degree
- MAX\_SPAN\_DELTA =  $0.01 \times \text{span}$
- MAX\_KNOT\_DELTA = 0.01 (on knot span [0,1])
- MAX\_CONTROL\_POINT\_DELTA =  $0.01 \times \text{normalised chord}$

Each blade in the evolving population was then evaluated at:

- $U_0 = 10 \text{ m/s}$
- $\lambda_{min} = 3$
- $\lambda_{max} = 10$
- F=0.8 CR=0.3

Again, F is the DE differential multiplier and CR is the crossover probability. In the interests of quicker object function evaluations and a shorter overall optimisation run, a POP\_SIZE of 20 was set. This is far too low for an objective *global* search of the parameter space, but for this local optimisation task proved adequate.

As mentioned previously, the run was characterised by reduced diversity due to the lack of genomic diversity in the seed population, as well as the small population size. This resulted in a final Pareto set with very little genomic diversity, which is acceptable when viewed in the current context of *local* geometry improvement. As such, at the completion of the run, one blade was chosen from the Pareto set as a representative example of the run.

Four different blade designs are presented in the following sections:

1. The existing 600 Watt blade design. This was modelled as a spanwise sequence of 29 SD7062 B-spline aerofoil curves and one rounded “hub” curve.
2. A model of the same blade as above, but with the  $5^\circ$  pitch angle removed.
3. A “seed” blade model based on the un-pitched blade, but defined by just five B-spline aerofoil curves (one of which is a static “hub” curve”).
4. The evolved “improved” curve (labelled as Pareto [A]). In essence, this is a modified “seed” blade surface, and as such is represented by just four new “aerofoil” curves and one static “hub” curve.

To compare the performance of these blade geometries, each was discretised into a  $40 \times 20$  quadrilateral panel mesh and supplied with a  $20 \times 10 \times 5$  quadrilateral wake panel mesh. The panel method described in Chapter 4 was applied to each blade mesh across a range of tip speed ratios spanning from  $0 \leq \lambda \leq 15$  at a constant co-axial wind speed of  $U_0 = 10$  m/s. Performance results from these evaluations appear in the following pages.

### 6.11.1 Existing 600 Watt blade

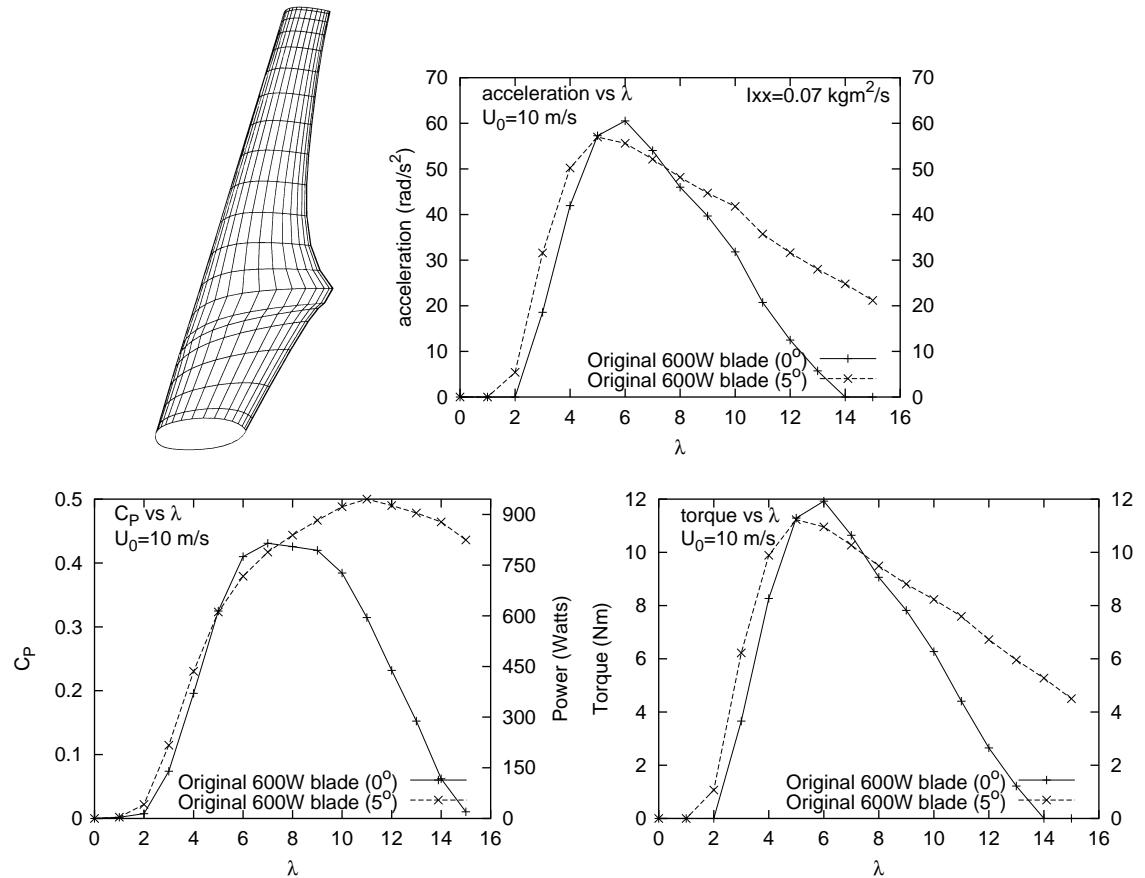


Figure 6.76: 600 Watt original blade - Geometry and performance results



B-spline curves	degree	control points	U-knots	V-knots
30	3	7	11	34

6.77.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
3	40	20	5	20

6.77.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
0.0657	0.4881	923	132	7.22	33.976	0.47

6.77.3: RESULTS

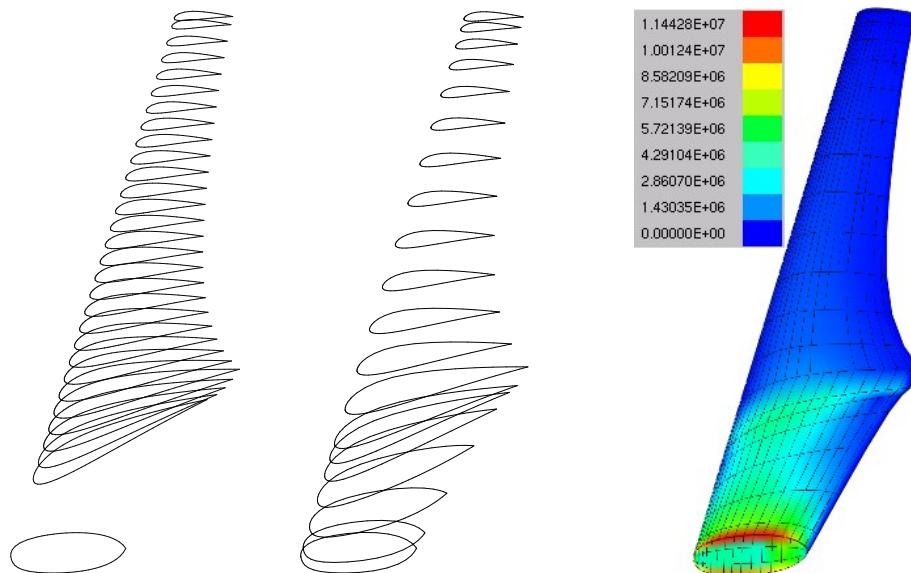
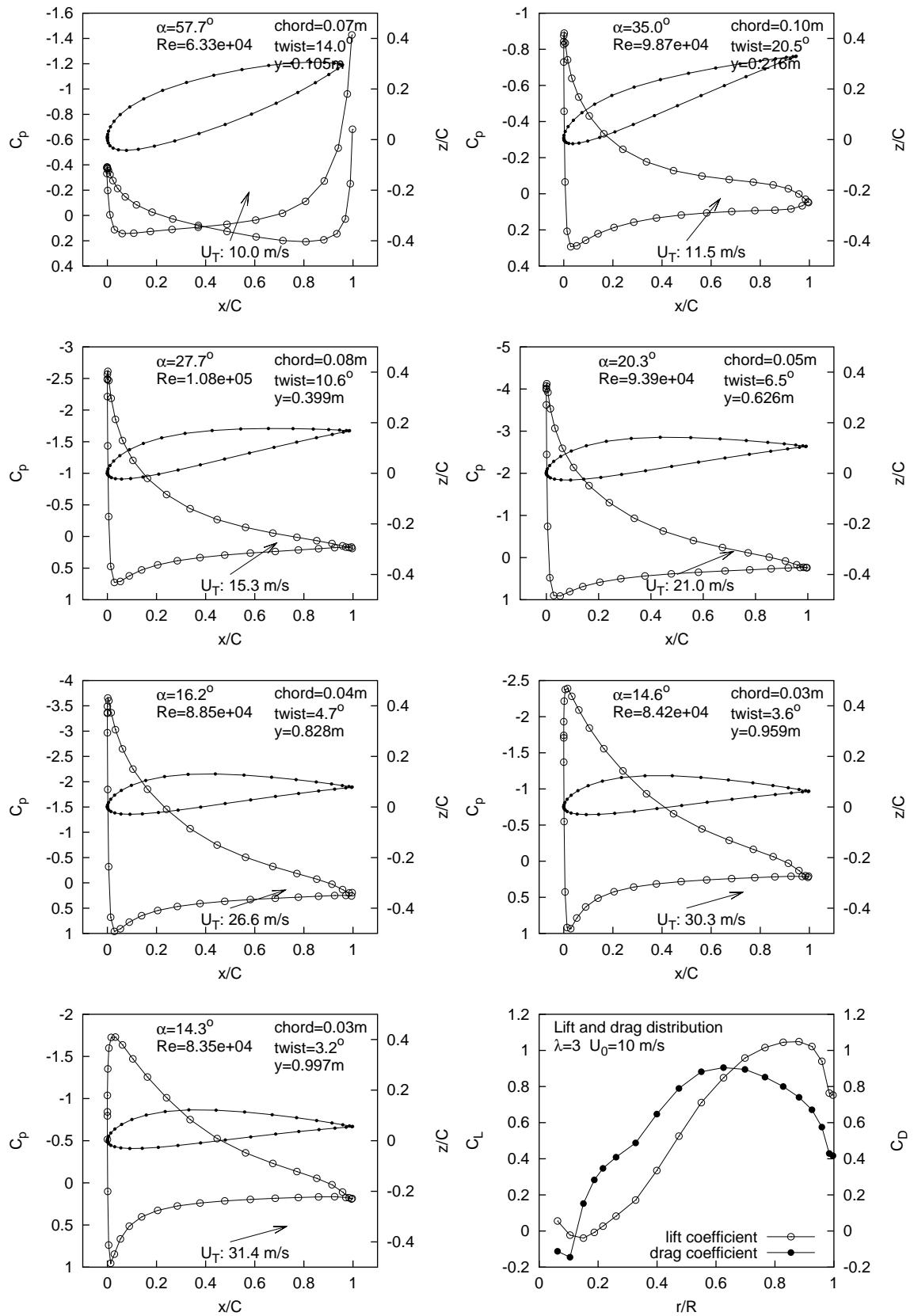
6.77.4: B-spline  
aerofoils6.77.5: Radial  
x-sections6.77.6: von-Mises  
equiv. stress

Figure 6.77: 600 Watt original blade - Geometry and performance results (continued)



Figure 6.78: Pressure and force distributions along 600 Watt original blade @  $\lambda=3$

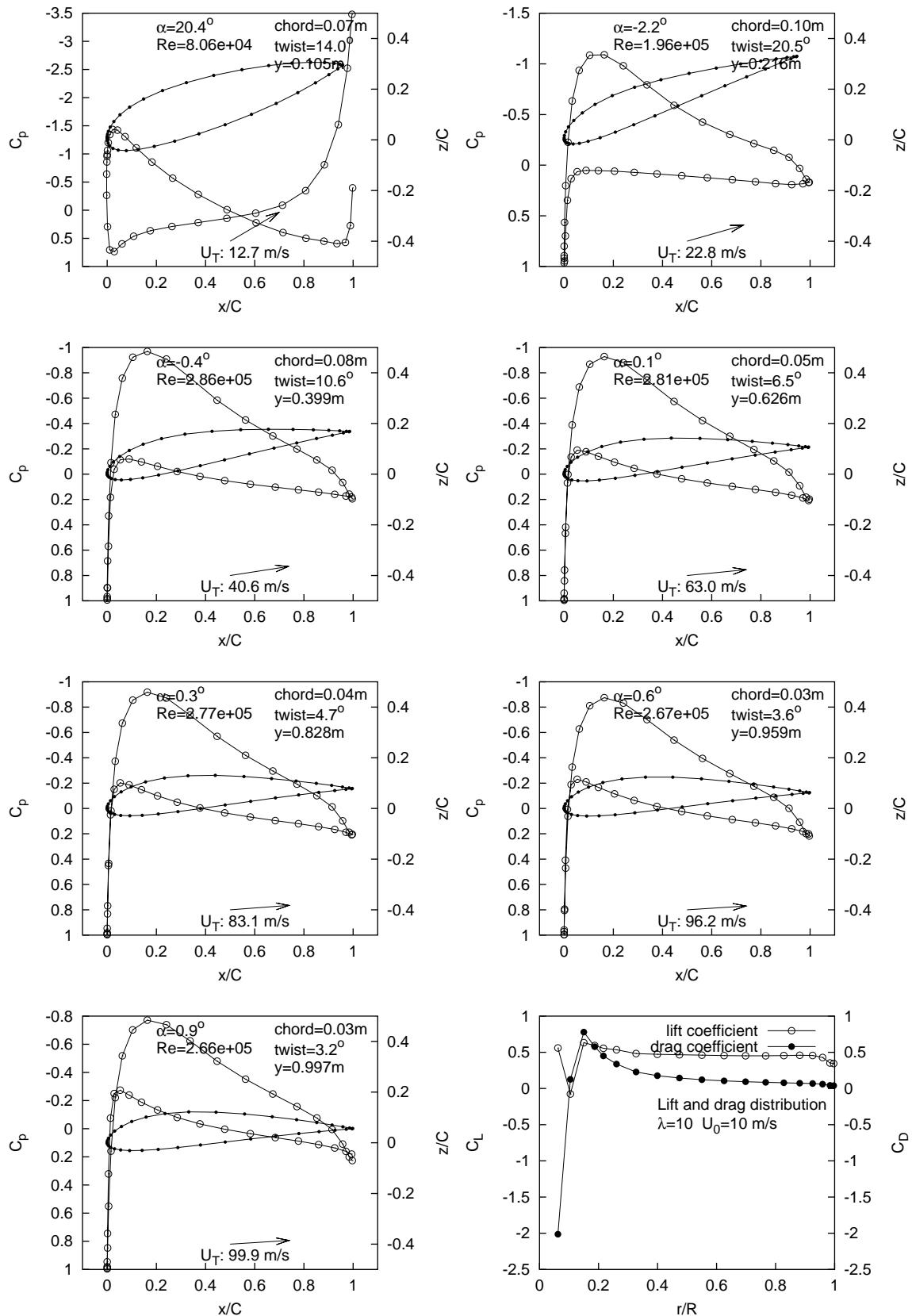


Figure 6.79: Pressure and force distributions along 600 Watt original blade @  $\lambda=10$

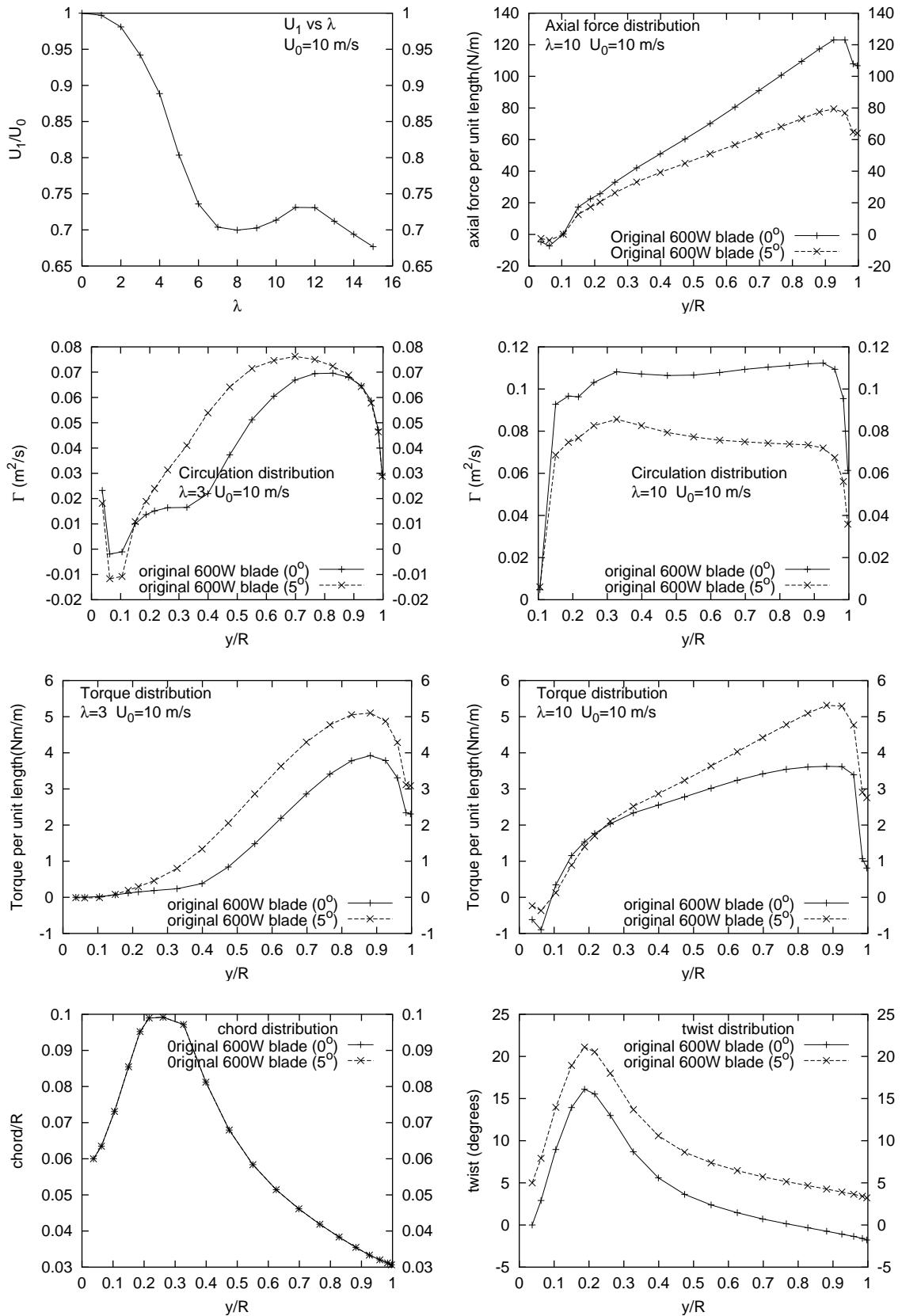


Figure 6.80: Performance results for 600 Watt original blade

### 6.11.2 600 Watt “seed” blade

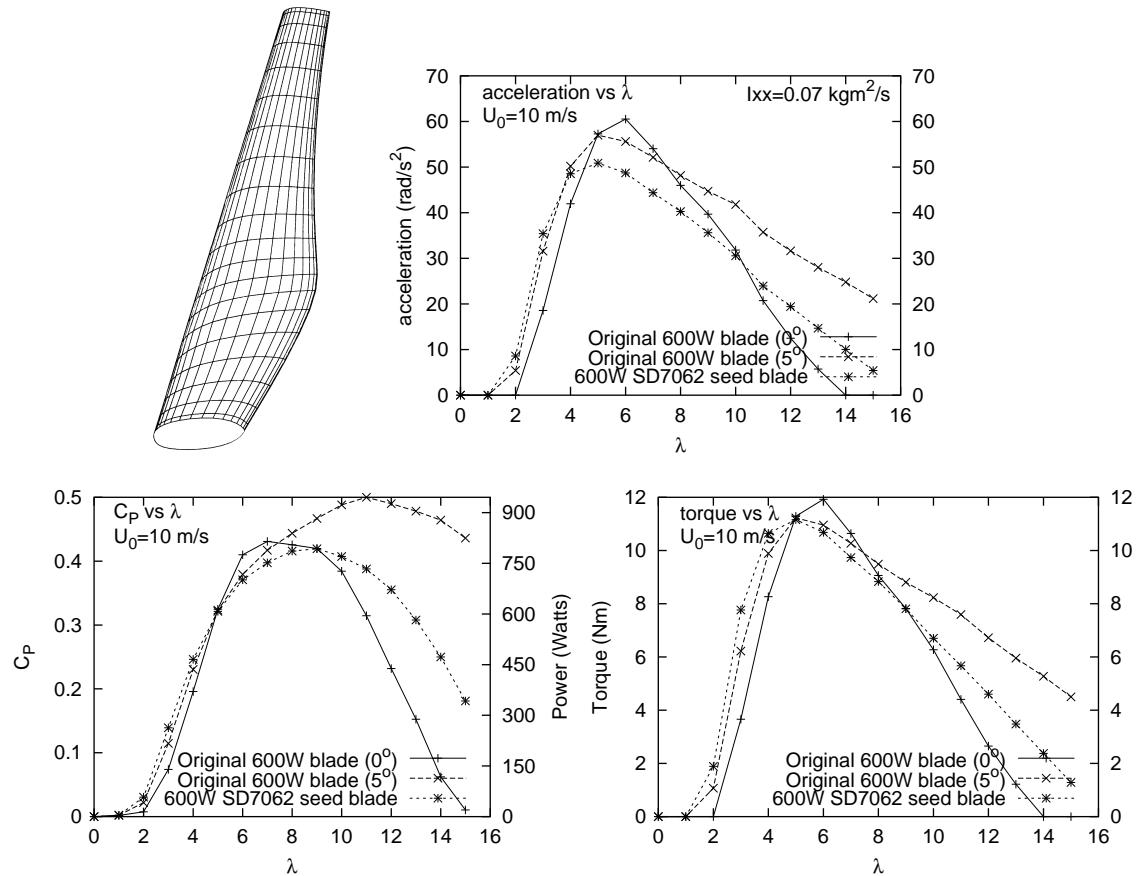


Figure 6.81: 600 Watt seed blade - Geometry and performance results

B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

## 6.82.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
3	40	20	5	20

## 6.82.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
0.0731	0.4073	770	107	8.76	37.016	0.48

## 6.82.3: RESULTS

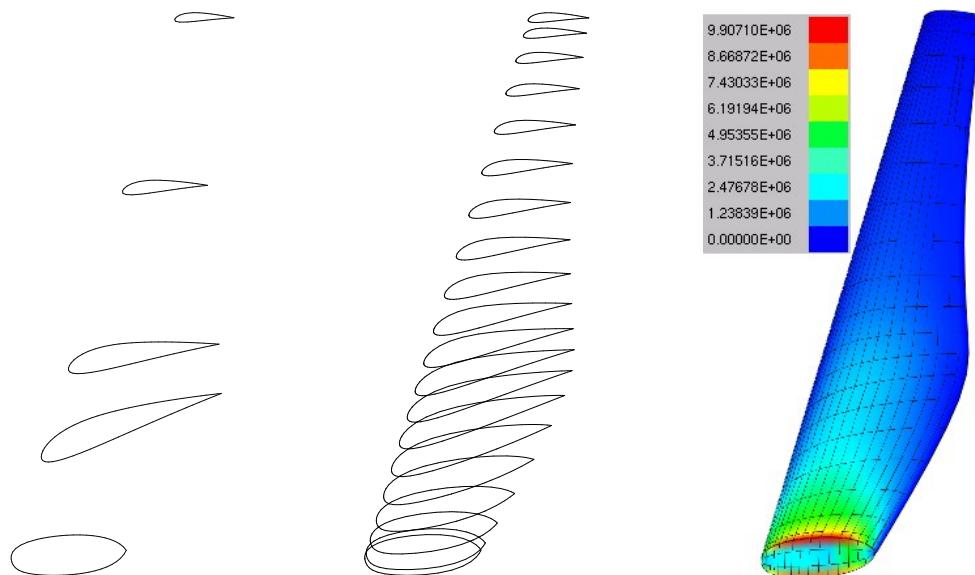
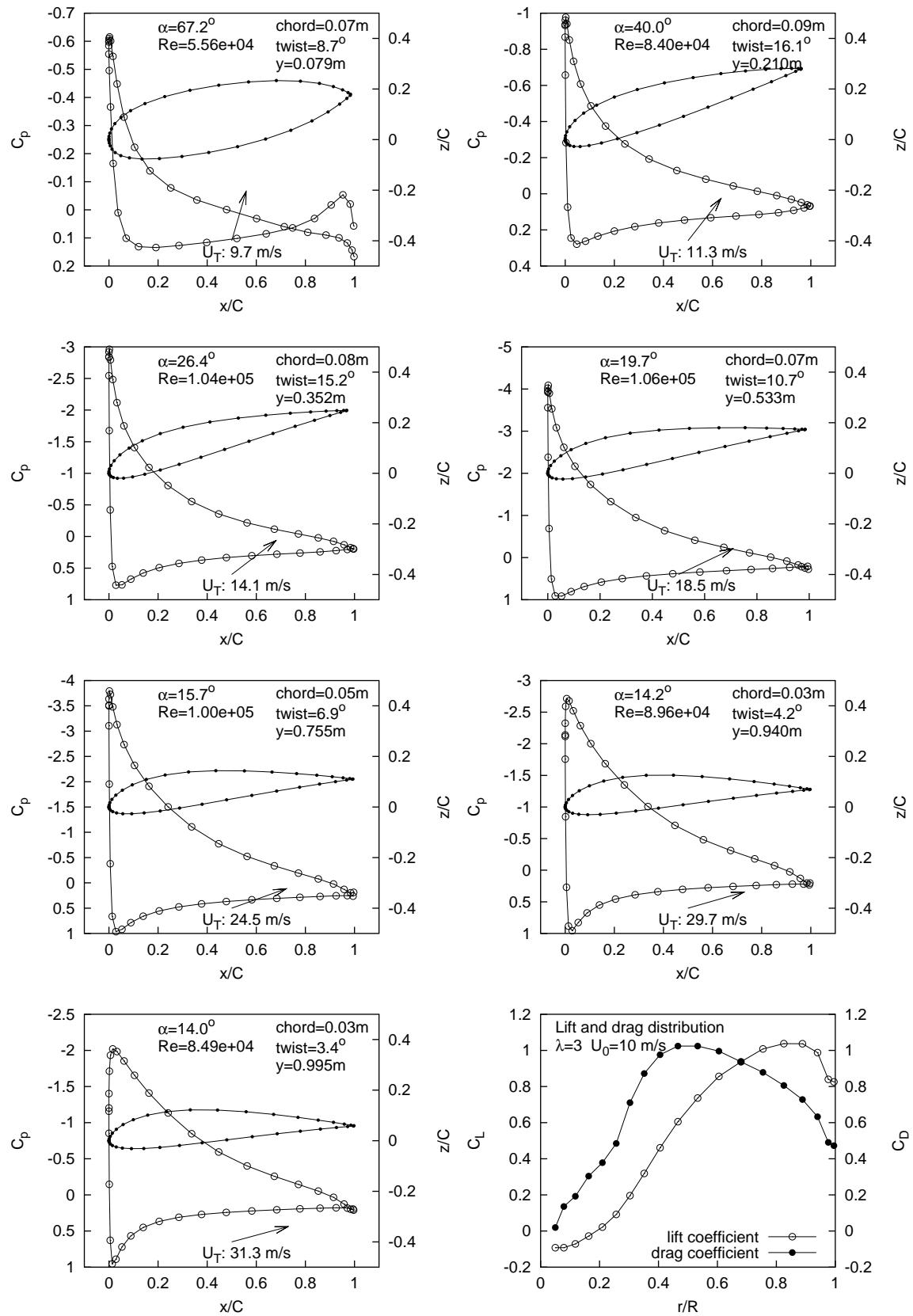
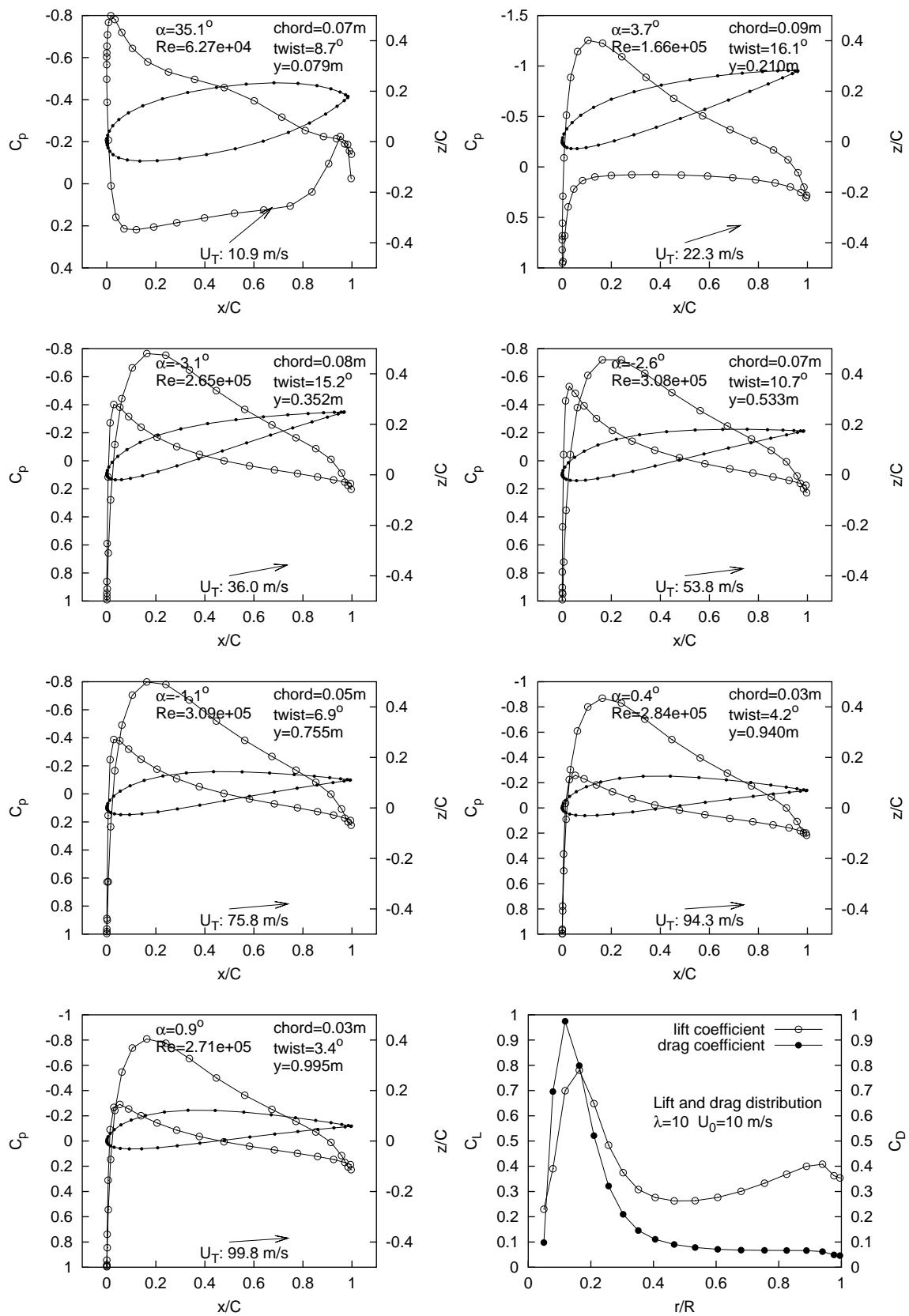
6.82.4: B-spline  
aerofoils6.82.5: Radial  
x-sections6.82.6: von-Mises  
equiv. stress

Figure 6.82: 600 Watt seed blade - Geometry and performance results (continued)



Figure 6.83: Pressure and force distributions along 600 Watt seed blade @  $\lambda=3$

Figure 6.84: Pressure and force distributions along 600 Watt seed blade @  $\lambda=10$

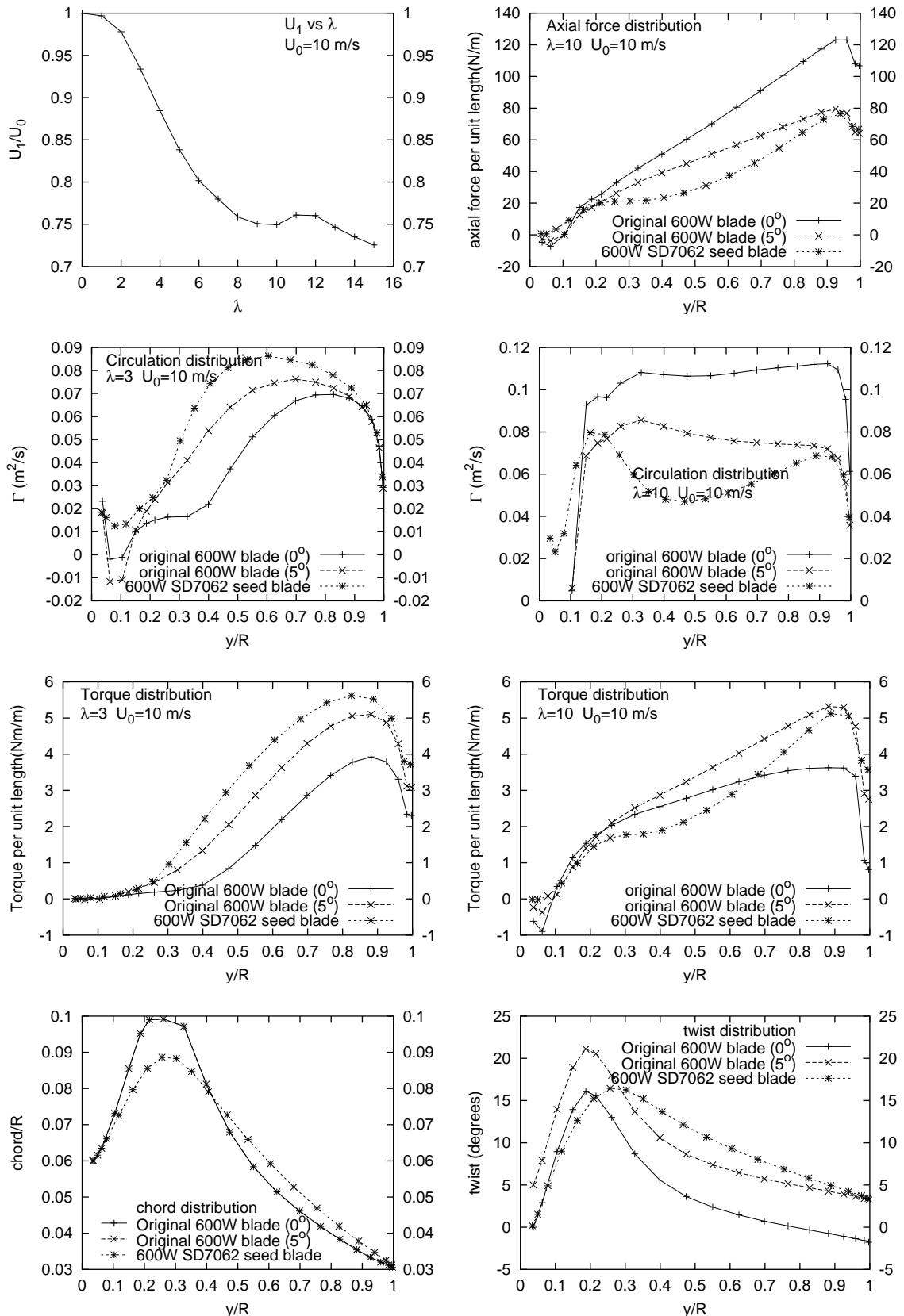


Figure 6.85: Performance results for 600 Watt seed blade

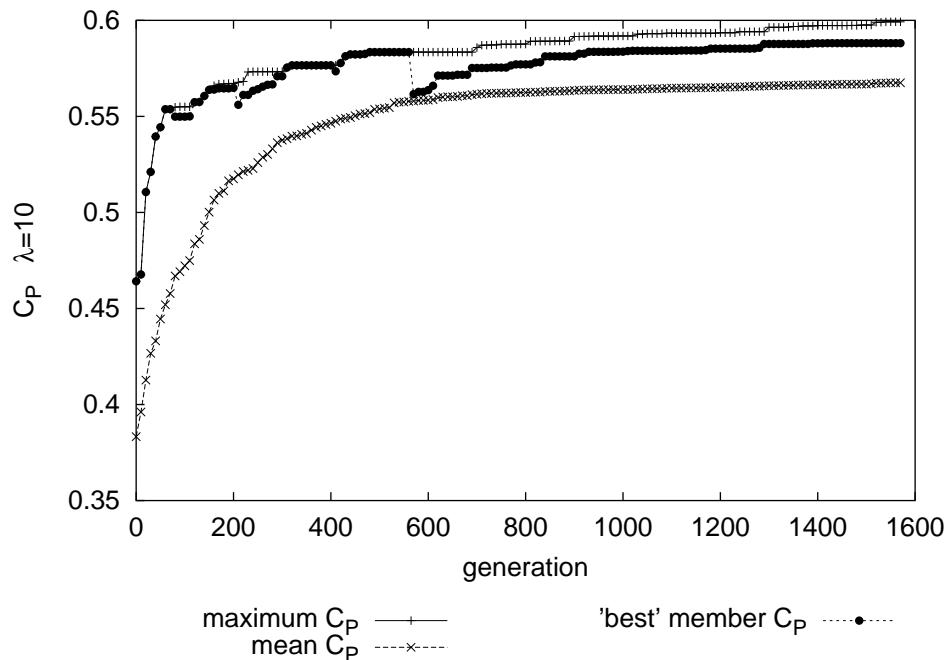
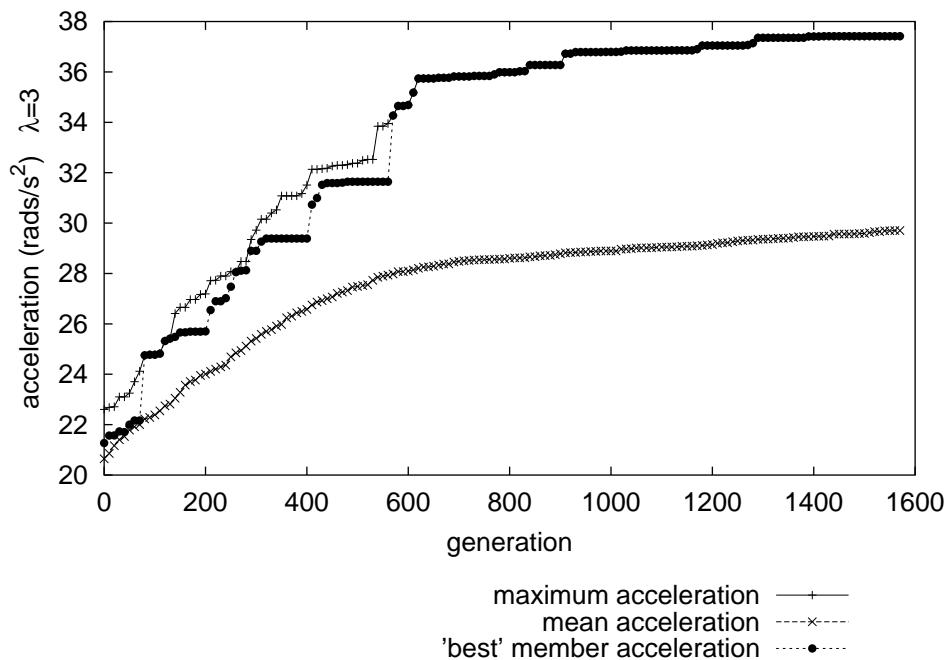
6.86.1: Evolving  $C_P$  performance ( $\lambda=10$ ) over 1600 generations6.86.2: Evolving acceleration performance ( $\lambda=3$ ) over 1600 generations

Figure 6.86: Performance figures for the evolving population over 1600 generations

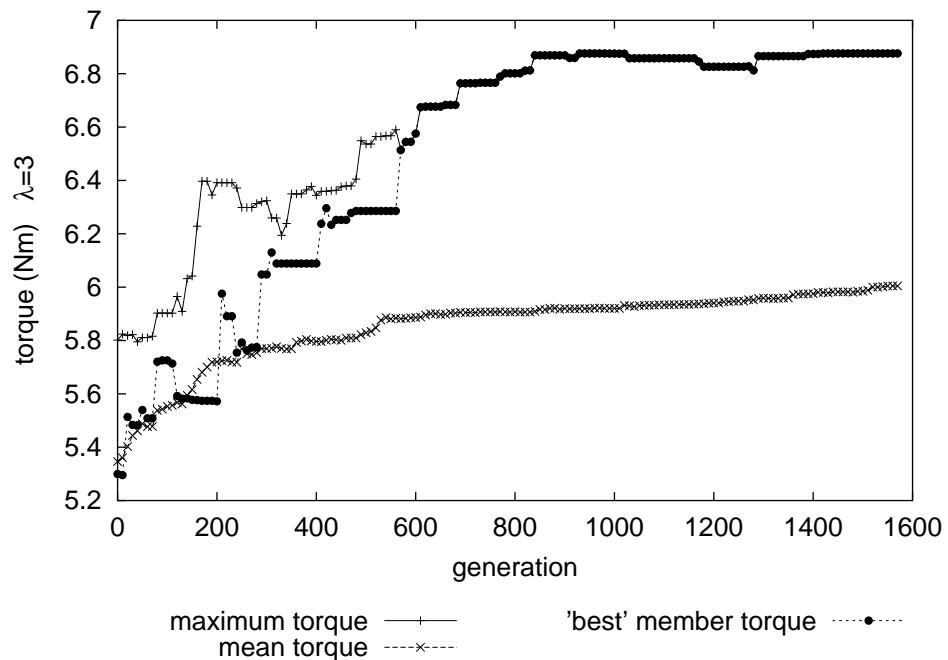
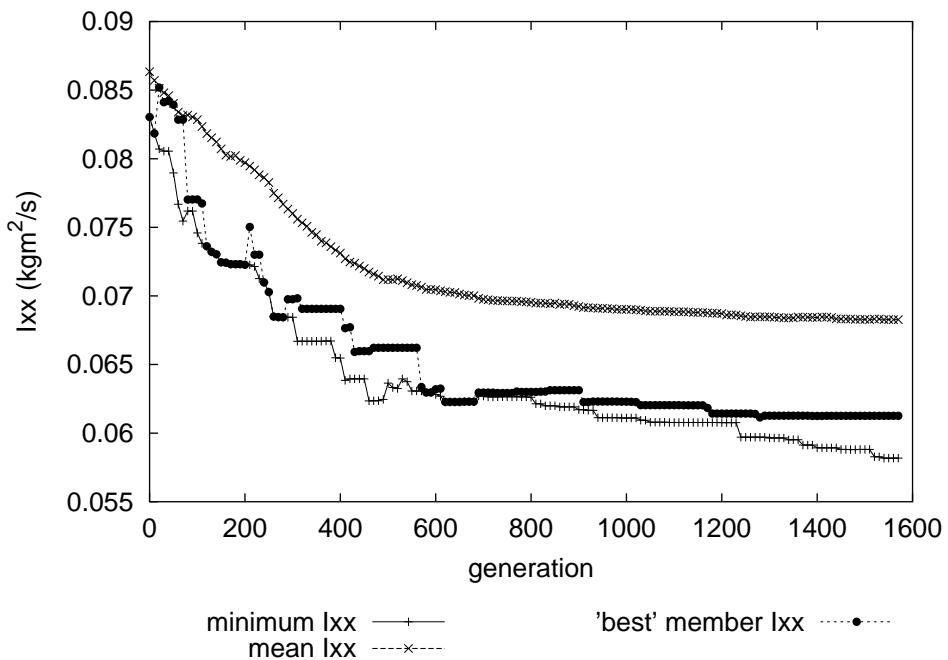
6.86.3: Evolving torque performance ( $\lambda=3$ ) over 1600 generations6.86.4: Evolving  $I_{xx}$  over 1600 generations

Figure 6.86: Performance figures for the evolving population over 1600 generations

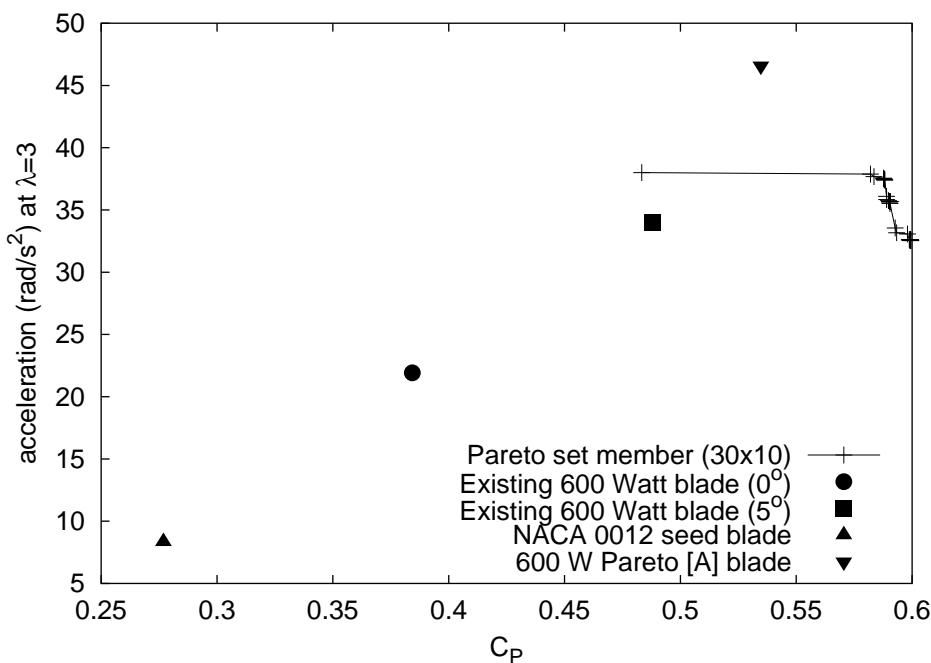


Figure 6.87: 600W Pareto front after 1600 generations ([A] blade typical)

Figure 6.87 compares the acceleration and  $C_P$  performance of the four blades studied in this section. The Pareto front after 1600 generations is included as an illustration of the final progress of the low-resolution run, and should not be directly compared to the four other distinct points on the plot, each of which represents a blade evaluated with a finer blade panel mesh ( $40 \times 20 \times 5 \times 20$ ). The Pareto [A] blade depicted in Figure 6.87 was a typical example of the blades in the final Pareto set (recall that the genomic diversity throughout the run remained quite low), and is used here as the sole result of the 600 Watt “improvement” run.

The [A] blade is a good example of what can be achieved using this method. Both low- $\lambda$  acceleration and peak  $C_P$  performance figures are appreciably higher than those of the existing 600 Watt blade design from which it was spawned. Detailed results in the following pages show that this improvement was largely due to changes in the *aerodynamic cross-section* of the blade. These “thinner” aerodynamic sections seem to perform as well as existing low-speed aerofoils, the SD7062 being the current benchmark. These slender sections drove the [A] blade  $I_{xx}$  approximately 20% lower than that of the existing blade ( $0.0517 \text{ kgm}^2$  versus  $0.0657 \text{ kgm}^2$  using an equivalent  $40 \times 20$  volume mesh), a result with significant ramifications for improved low- $\lambda$  acceleration.

### 6.11.3 600 Watt [A] blade

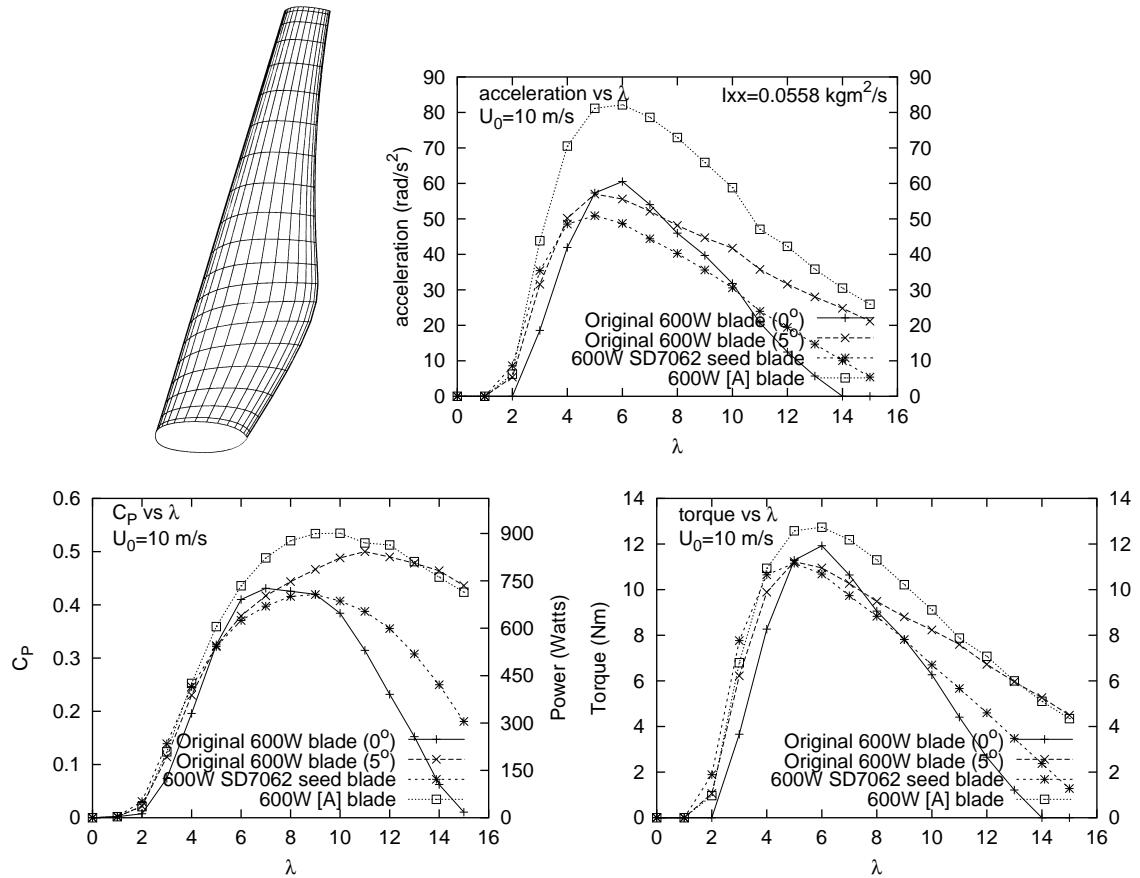


Figure 6.88: 600 Watt Pareto [A] blade - Geometry and performance results



B-spline curves	degree	control points	U-knots	V-knots
5	3	7	11	9

6.89.1: Blade parameters

NB	N_PANELS	M_PANELS	TURNS	W_PANELS
3	40	20	5	20

6.89.2: Mesh parameters

I <sub>xx</sub> (kgm <sup>2</sup> )	C <sub>P</sub> (λ=10)	P(W)(λ=10)	F <sub>x</sub> (N)(λ=10)	Q(Nm)(λ=3)	Ω(rad/s <sup>2</sup> )(λ=3)	M(kg)
0.0517	0.5347	1011	171	7.80	46.581	0.37

6.89.3: RESULTS

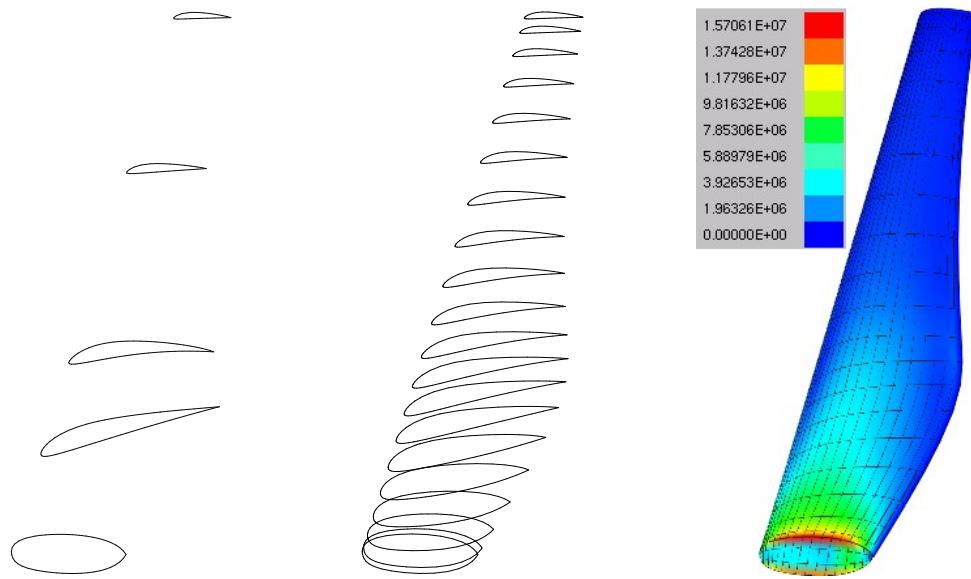
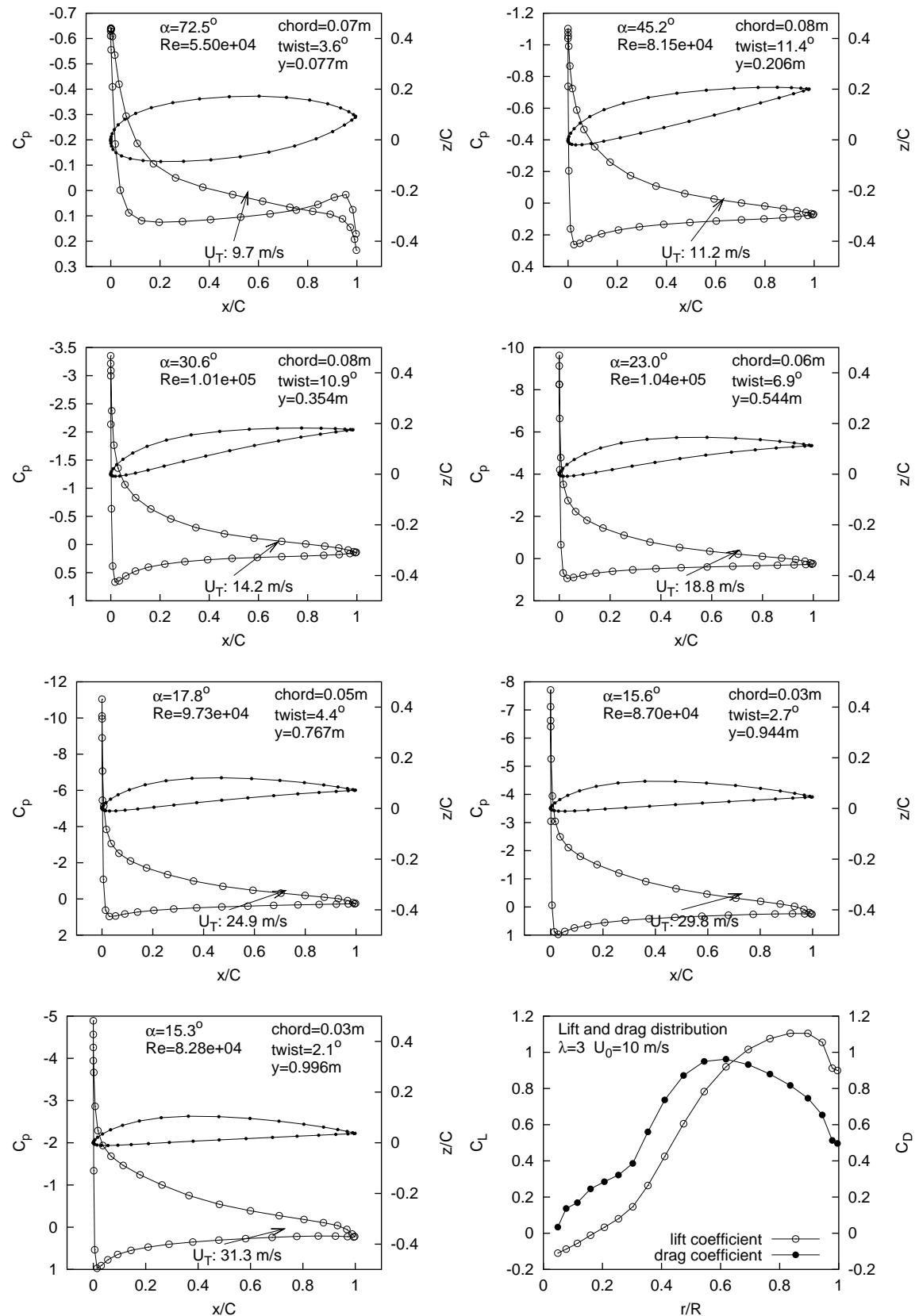


Figure 6.89: 600 Watt Pareto [A] blade - Geometry and performance results (continued)



Figure 6.90: Pressure and force distributions along 600 Watt Pareto [A] blade @  $\lambda=3$

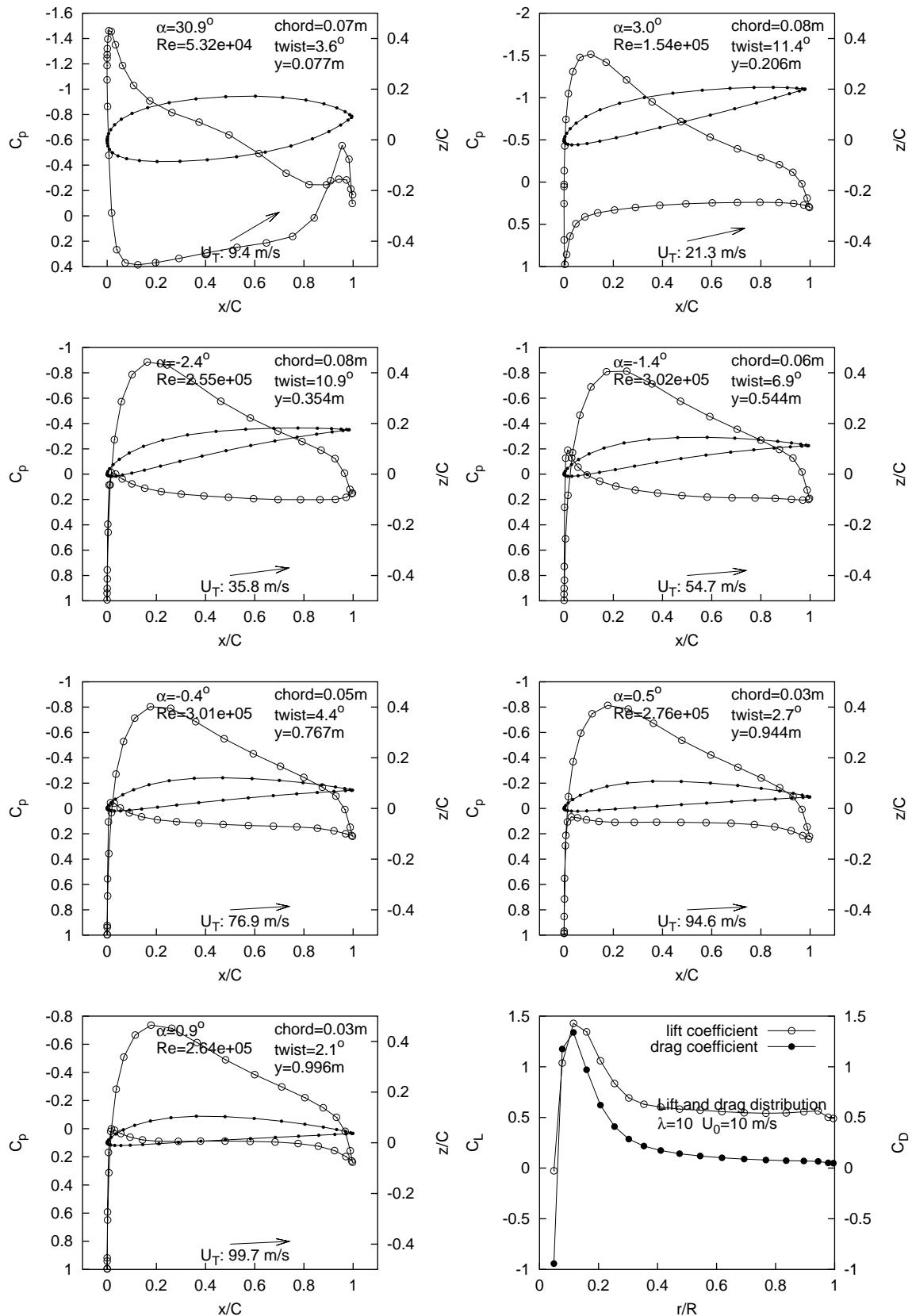


Figure 6.91: Pressure and force distributions along 600 Watt Pareto [A] blade @  $\lambda=10$

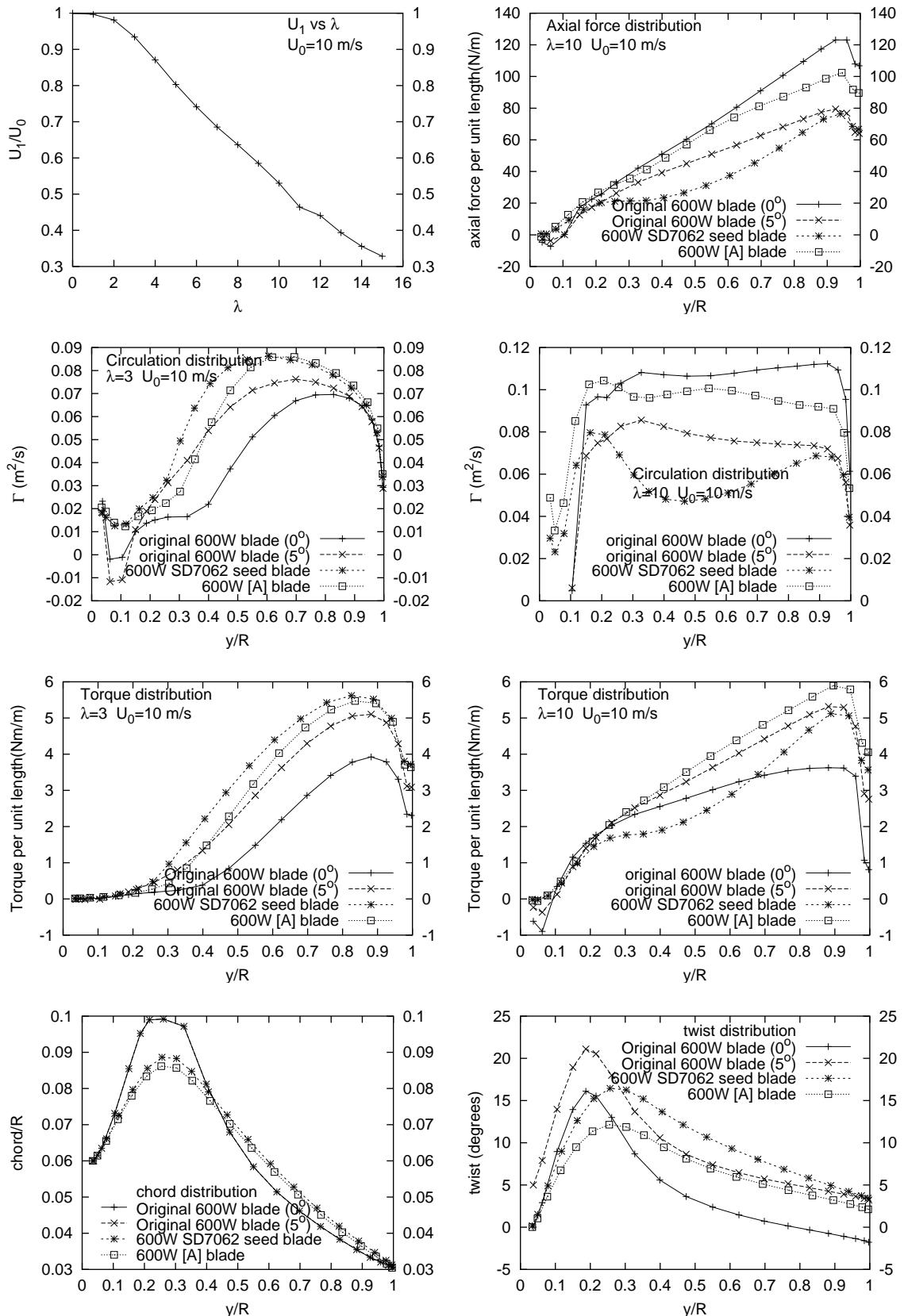


Figure 6.92: Performance results for 600 Watt Pareto [A] blade

## 6.12 Discussion

The results of the 600 Watt run suggest that the method is capable of finding good evolutionary improvements over existing blade designs. Figure 6.86.1 shows that  $C_P$  very quickly converges to near-optimum levels, most likely because the seed blade is very close to the optimal  $C_P$  configuration using the SD7062 aerofoil. By altering the seed's twist distribution (a matter of adjusting just four object variables per blade) the evolution routine was able to quickly “tune” the  $C_P$  performance of the population. It is interesting to note from the twist distribution comparisons in Figure 6.92 show that the evolved blade's twist closely matches that of the existing blade (at 5° pitch) for the tip-wise half of the span. It is reasonable to conclude that the evolutionary optimisation routine is in part serving its intended purpose by quickly finding an optimal- $C_P$  design, and it is somewhat gratifying that this was achieved by mirroring the twist distribution of an experimentally validated blade design with near-optimal peak power performance.

Acceleration performance evolved more slowly than peak-power performance, a fact demonstrated by Figures 6.86.2, 6.86.3 and 6.86.4. The simplest explanation for the discrepancy between evolution rates for the two optimisation objectives is that the seed blade was closer to the optimal  $C_P$  design than to the optimally accelerating design. Of course this is to be expected, since the seed's geometry was a drastic simplification of the existing optimal-power blade (albeit twisted non-optimally and with a rather bulky non-optimal chord distribution). Figure 6.86.3 shows that the low- $\lambda$  torque performance of the (arbitrarily-weighted) “best” blade in the evolving population is matched by a similar rate of improvement for the “best” second moment of inertia figures (Figure 6.86.4). The noteworthy feature of these plots is not that *both* low- $\lambda$  torque and  $I_{xx}$  improved to levels appreciably better than those of the seed - a predictable and perhaps necessary result, since the seed was specifically chosen to be non-optimal so that the optimisation had “room to move” - but that the rate of evolution of both had stabilised considerably by generation 800. The most likely interpretation of this is that the evolving population had found an optima in the two-dimensional parameter space being searched, and that the low genomic diversity of the population prevented excursions from this optima: the population had “converged”. Without a knowledge of what a truly “optimal” blade design is it is impossible to currently know

whether this converged state is close to the global optimum or whether it is stuck in perhaps one of many local optima. The latter is a rather more likely result.

It is important to re-emphasise that this project is more concerned with finding whether it is *possible* to improve on existing blade designs rather than finding an elusive globally optimal solution. In this context the project, and specifically the current 600 Watt study, is successful. Figure 6.87 shows that the representative result of the run - the Pareto [A] blade - is at least as good at peak-power extraction *whilst providing a predicted low- $\lambda$  acceleration improvement of approximately 25%*. Figure 6.92 shows both that the evolved chord distribution remained essentially unaltered to that of the seed blade, and that the slight improvements in low- $\lambda$  torque are insufficient to account for the significant improvement in low- $\lambda$  acceleration. Figures 6.90 and 6.91 show pressure distributions which are well-behaved, as well as showing cross-sections through the evolved blade which are comparatively thin, “light” shapes which nevertheless at least equal the aerodynamic performance of the SD7062 curve from which they were spawned.

Many more identical optimisation runs are required (and statistical results collated) to verify that the method is capable of consistently evolving better-than-state-of-the-art blade geometries. Further, no claim is made here that the Pareto [A] blade is an optimal blade design, just that the panel method described in the previous chapters evaluates its performance comparatively better than that of a model of a current human-designed blade. Thus the method described in this project is the first iteration in the development a robust, rigorous and repeatable optimisation method. The next step will be to make a working prototype of an evolved blade design (perhaps a set of three one-metre Pareto [A] blades) and test it on a rotor in a wind tunnel and/or in the field. The results from such a test will be invaluable for the next iteration in the development of the method: the validation of the output of the panel method solver with a view to the furthering the development of such things as a better wake model and better viscous-regime handling.



# Chapter 7

## Conclusion

The project documented herein generated a number of significant outcomes:

1. **The development of a B-spline representation of aerofoil curves and wind turbine blade surfaces.** The use of bi-cubic B-spline surfaces was shown to be a convenient, flexible, accurate, and compact way of representing the complex three-dimensional geometry of wind turbine blades. It was shown that bi-cubic B-spline blade surface representations could easily be partitioned into quadrilateral surface panels for use by an aerodynamic “panel method” solver. Further, it was demonstrated how a B-spline curve fit to aerofoil co-ordinate data could easily be accomplished by “evolving” the position of the B-spline curve’s control points and knots.
2. **The development of an aerodynamic panel method solver for general three-dimensional blade geometries.** A fast and reasonably accurate three-dimensional, first-order aerodynamic panel method solver was presented which had the capability to efficiently evaluate the performance of *general three-dimensional blade surface geometries*. A simple iterative wake model was also presented. Through validation of the method against performance measurements (and third-party computational performance calculations) of real wind turbine blades it was shown that, even with the use of many simplifying assumptions, this general aerodynamic solver offers good performance predictions, especially at moderate-to-high tip speed ratios.
3. **A simple yet useful treatment of the starting problem.** The “starting problem”

was defined and the methodology established by which a panel-method calculation of instantaneous aerodynamic torque at a low tip speed ratio, together with an approximation of blade inertia via finite-element blade volume discretisation could be used to provide a relative measure of wind turbine starting performance.

4. **Application of the Differential Evolution algorithm to the problem of evolving general three-dimensional blade geometries.** The Differential Evolution technique was described and its use as a multi-objective, real-valued object vector optimiser presented. A way to encode bi-cubic B-spline blade surfaces as (compact) real-valued one-dimensional vectors of geometry parameters was described, as was a method for “evolving” an initially randomly-generated pool of these into a “Pareto optimal set” of *new* optimal blade designs using the DE optimiser. Results of two computational experiments employing this technique were offered, including a novel Pareto set of blade geometries. A selection of these designs included a handful of exceptional *new* blades, with simulated performance results suggesting that these may out-perform current state-of-the-art small wind turbine blade designs.
5. **Identification of significant differences between blade geometries optimised for starting, and those optimised for power production.** It was asserted at the beginning of this thesis that power production optimisation and starting performance optimisation were *competing* objectives. This was shown to be true by the strong *diversity* of blade designs in the final 5 kW Pareto optimal set. Valid evolved geometries spanned from those with excellent starting performance and relatively poor power extraction (caused by very slender chord distributions and “thin” aerofoil sections, giving vastly reduced second moments of inertia), through to blades with excellent power production and “good” starting characteristics. A diverse range of evolved designs between these extremes illustrated the strong relationship between increasing power performance and increasing second moment of inertia, with the result that improved power production almost invariably comes at a cost of reduced starting performance.

It should be evident that some effort has been spent in pointing out flaws in component of the computational method. The most serious of these is contained in the treatment of the viscous effects of low-Re, high- $\alpha$  flows. Future workers are strongly urged

to devote some effort to treating viscous effects, such as stall, in a much more rigorous way, as the correct simulation (or estimation) of the blade forces these flows induce is essential for the correct determination of starting performance. A first step may be the development of a method to predict when separation has occurred, and to use this as a correction to the inviscid code for improved low-speed predictions. The crude “starting” acceleration determination made use of in this project, whilst useful in the current context, should be improved upon greatly in future treatments dealing with small wind turbine starting. Current and future research into three-dimensional Viscous-Inviscid Interaction methods may hold the key to such computational improvements. However, the perennial questions remain: how can the low-Re, high- $\alpha$  flows commonly experienced by stationary and slowly-rotating small wind turbine blades be adequately simulated? And, just as importantly in the current context, how can these simulations be made to run *quickly* to complement the comparatively low-overhead of the inviscid panel method for the many thousands of aerodynamic evaluations needed per optimisation run?

A simple wake model was utilised in all aerodynamic simulations undertaken in the current project, and it was shown that a new method to determine the wake vortex pitch (based on the work of Wood and Boersma (2001)) is an adequate first-approximation to simulating the wake of an operating wind turbine. In many respects, the model is the *simplest* that could possibly be utilised. Future researchers hoping to expand the utility and accuracy of the method are encouraged to include:

- A model for wake expansion. A constant diameter wake (equal to the rotor diameter) was employed in the current work. This is clearly not physically correct. It is anticipated that accounting for conservation of mass of the column of air through the rotor will improve simulation accuracy.
- An infinite wake. A severely truncated wake (five revolutions) was used in all simulations in the current project. A shorter wake resulted in fewer wake panels, with a corresponding reduction in memory usage and solution times. Future work would most likely benefit from the inclusion of longer wake models and/or the addition of an approximation for the influence of the far wake.
- A means for handling yawed rotors. The scope of the current project called for the simplest possible flow conditions, the first requirement of which was, not

surprisingly, constant, axial flow. As small wind turbines spend a considerable portion of their operational lives yawing into the wind, a future direction may be to expand upon the current code to include wakes which are not perfect helices, and which are not aligned with the rotor axis.

- Further improvement and verification of the wake pitch relaxation technique. Very little experimental work exists which can validate or invalidate the wake pitch model employed in the current project. Future workers hoping to improve wake simulation accuracy will first need to determine what an “accurate” wake pitch is. The computational technique used here, and in particular the rough-and-ready relaxation iteration scheme, can also in all likelihood be improved, with a reduction in the number of expensive solution iterations being the prize for an optimised iteration technique.

The optimisation method used in the current project should only be considered *an example* of the possibilities of an evolutionary approach to aerodynamic shape optimisation. Nothing substantially novel in the field of Evolutionary Computation (EC) was sought or obtained, with the scope of the project centering around the *application* of a number of existing complementary EC techniques to solve a mechanical engineering problem. For improved rigour, future researchers are strongly encouraged to statistically verify the results of the method, a task which will involve averaging over many, many identical optimisation runs. More effort can also conceivably be devoted to the “computer science” part of future projects: an empirical approach was adopted here in the selection of such things as population sizes, object vector lengths and DE parameter settings, with arbitrary settings being chosen to reduce run times, rather than for peak optimisation performance. Strategies which increase population diversity, such as fitness sharing, could likely be used to improve the global-search performance of the DE optimisation method. Future researchers are also encouraged to treat the current handling of fitness and viability evaluation of candidate blade designs as a starting point: many different ways of approaching the multiobjective problem are possible.

In conclusion, it is fair to say that the objectives of the current project have been met: an evolutionary approach to three-dimensional aerodynamic shape optimisation has been shown to be viable. Future researchers interested in expanding upon the method will likely enjoy the benefits of faster, cheaper parallel computing hardware,

and will thus be able to employ finer panel meshes and larger evolving populations. The success of the current, limited, method suggests that the attainment of these two goals will result in exceptionally good optimisation results.



## References

- Althaus, D. (1980). *Profilpolaren fur den Modellflug : Windkanalmessungen an Profilen im kritischen Reynoldszahlbereich*. Neckar-Verlag, Villingen.
- Althaus, D. (1996). *Niedriggeschwindigkeitsprofile : Profilentwicklungen und Polarenmessungen im Laminarwindkanal des Institut für Aerodynamik und Gasdynamik der Universität Stuttgart. Profilpolaren fur den Modellflug : Windkanalmessungen an Profilen im kritischen Reynoldszahlbereich*. Vieweg, Braunschweig, Germany.
- Anderson, M. B., Milborrow, D. J., and Ross, J. N. (1982). Performance and wake measurements on a 3m diameter horizontal axis wind turbine: Comparison of theory, wind tunnel and field test data. In *Fourth International Symposium on Wind Energy Systems*, pages 113–136. BHRA Fluid Engineering, Cranfield, Bedford, England.
- Arsuffi, G., Guj, G., and Morino, L. (1993). Boundary element analysis of unsteady aerodynamics of windmill rotors in the presence of yaw. *Journal of Wind Engineering and Industrial Aerodynamics*, 45:153–173.
- Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice*, chapter 2, pages 66–91. Oxford University Press, New York.
- Bäck, T., Haase, W., Naujoks, B., Onesti, L., and Turchet, A. (1999). Evolutionary algorithms applied to academic and industrial test cases. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 383–398. John Wiley and Sons Ltd., West Sussex, England.

- Bäck, T. and Schwefel, H. P. (1995). Evolution strategies i: Variants and their computational implementation. In Périaux, J. and Winter, G., editors, *Genetic Algorithms in Engineering and Computer Science*, chapter 6. John Wiley and Sons Ltd.
- Barnsley, M. J. and Wellicome, J. F. (1992). Wind tunnel investigation of stall aerodynamics for a 1.0 m horizontal axis rotor. *Journal of Wind Engineering and Industrial Aerodynamics*, 39:11–21.
- Besnard, E., Vermeersch, T., Reboul, G., Schmitz, A., and Cebeci, T. (1998). Prediction of wing flows with separation. AIAA. AIAA paper 98-0404.
- Bossanyi, E. A. (1998). *Bladed for Windows: Theory Manual*. Garrad Hassan and Partners Limited.
- Branum, L. and Tung, C. (1997). Performance and pressure data from a small model tilt-rotor in hover. NASA Technical Memorandum 110441 USAATCOM Technical Report 97-A-003. Aeroflightdynamics Directorate, U. S. Army Aviation and Troop Command, Ames Research Center, Moffett Field, CA.
- Brentner, K. S. (1986). Prediction of helicopter rotor discrete frequency noise. NASA Technical Memorandum 87721.
- Buhl, M. L., Wright, A. D., and Tangler, J. L. (1997). Wind turbine design codes: A preliminary comparison of the aerodynamics. Technical Report NREL/CP-500-23975, National Renewable Energy Laboratory, Golden, Colorado.
- Burton, T., Sharp, D., Jenkins, N., and Bossanyi, E. (2001). *Wind Energy Handbook*. John Wiley and Sons Ltd., Chichester, England.
- Caradonna, F. X. and Tung, C. (1981). Experimental and analytical studies of a model helicopter rotor in hover. *Vertica*, 5:149–161.
- Cebeci, T. (1998). *An engineering approach to the calculation of aerodynamic flows*. Horizons Publishing, Inc., Long Beach, California.
- Cebeci, T. and Besnard, E. (1998). An efficient and accurate approach for analysis and design of high lift configurations. *Canadian Aeronautics and Space Journal*, 44(4):1–17.

- Cebeci, T. and Cousteix, J. (1999). *Modeling and Computation of Boundary-Layer Flows : laminar, turbulent, and transitional boundary layers in incompressible flows*. Horizons Publishing, Inc., Long Beach, California.
- Cerrolaza, M. and Annicchiarico, W. (1999). Genetic algorithms in shape optimization: Finite and boundary element applications. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 283–326. John Wiley and Sons Ltd., West Sussex, England.
- Chang, C. S., Xu, D. Y., and Quek, H. B. (1999). Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system. *IEE Proc.-Electr. Power Appl.*, 146(5):577–583.
- Clausen, P. D. and Wood, D. H. (2000). Recent advances in small wind turbine technology. *Wind Engineering*, 24(3):189–201.
- Coello, C. A. C. (2001). List of references on evolutionary multiobjective optimization. <http://www.lania.mx/~ccoello/EMOO/EMOObib.html>.
- Conlisk, A. T. (1997). Modern helicopter aerodynamics. *Annu. Rev. Fluid. Mech.*, 29:515–567.
- Cvetković, D. (2000). *Evolutionary Multi-Objective Decision Support Systems for Conceptual Design*. PhD thesis, School of Computing, University of Plymouth, Plymouth, UK.
- Deb, K. (1999). Evolutionary multi-criterion optimization. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 135–162. John Wiley and Sons Ltd., West Sussex, England.
- Diveux, T., Sebastian, P., Bernard, D., and Puiggali, J. R. (2001). Horizontal Axis Wind Turbine Systems: Optimization Using Genetic Algorithms. *Wind Energy*, 4:151–171.

- Drela, M. (1989). XFOIL: An analysis and design system for low Reynolds number airfoils. In Mueller, T. J., editor, *Low Reynolds Number Aerodynamics*. Springer-Verlag.
- Drela, M. (2001). XFOIL: Subsonic airfoil development system. <http://raphael.mit.edu/xfoil/>.
- Drela, M. and Giles, M. B. (1987). Viscous-inviscid analysis of transonic and low Reynolds number airfoils. *AIAA Journal*, 25(10):1347–1355.
- Du, Z. and Selig, M. S. (1998). A 3-D stall-delay model for horizontal-axis wind turbine performance prediction. AIAA. AIAA paper 98-0021.
- Du, Z. and Selig, M. S. (2000). The effect of rotation on the boundary layer of a wind turbine blade. *Renewable Energy*, 20(2):167–181.
- Duque, E. P. N., Johnson, W., vanDam, C. P., Cortes, R., and Yee, K. (2000). Numerical predictions of wind turbine power and aerodynamic loads for the NREL Phase II combined experiment rotor. AIAA. AIAA paper 2000-0038.
- Ebert, P. R. and Wood, D. H. (1997). Observations of the starting behaviour of a small horizontal-axis wind turbine. *Renewable Energy*, 12(3):245–257.
- Eppler, R. and Somers, D. M. (1980). A computer program for the design and analysis of low-speed airfoils. NASA TM-80210.
- Fogel, D. B. (1999). An introduction to evolutionary computation and some applications. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 23–42. John Wiley and Sons Ltd., West Sussex, England.
- Fonseca, C. M. and Fleming, P. J. (1998a). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation. *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 28(1):26–37.
- Fonseca, C. M. and Fleming, P. J. (1998b). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part II: Application example.

- IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans*, 28(1):38–47.
- Fuglsang, P. and Madsen, H. A. (1999). Optimization method for wind turbine rotors. *Journal of Wind Engineering and Industrial Aerodynamics*, 80:191–206.
- Giguère, P. and Selig, M. S. (1997). Aerodynamic blade design methods for horizontal axis wind turbines. Presented at the 13th Annual Canadian Wind Energy Association Conference and Exhibition. Quebec City, Quebec.
- Giguère, P. and Selig, M. S. (1998). New airfoils for small horizontal axis wind turbines. *Journal of Solar Energy Engineering*, 120:108–114.
- Giguère, P. and Selig, M. S. (1999). Design of a tapered and twisted blade for the NREL combined experiment rotor. Technical Report NREL/SR-500-26173, National Renewable Energy Laboratory, Golden, Colorado.
- Gouriérès, D. L. (1982). *Wind Power Plants: Theory and Design*. Pergamon Press, Oxford.
- Hampsey, M. and Wood, D. H. (1999). Designing small wind turbine blades for optimal starting and power extraction. *Wind Engineering*, 23(1):31–39.
- Hansen, A. C. and Butterfield, C. P. (1993). Aerodynamics of horizontal-axis wind turbines. *Annu. Rev. Fluid. Mech.*, 25:115–149.
- Hansen, A. C., Butterfield, C. P., and Cui, X. (1990). Yaw loads and motions of a horizontal axis wind turbine. *Journal of Solar Energy Engineering*, 112:310–314.
- Hansen, A. C. and Cui, X. (1989). Analysis and observations of wind turbine yaw dynamics. *Journal of Solar Energy Engineering*, 111:367–371.
- Hansen, N. and Ostermeier, A. (1996). Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In *Proc. of the 1996 IEEE Int. Conf. on Evolutionary Computation*, pages 312–317, Piscataway, NJ. IEEE Service Center.
- Hansen, N., Ostermeier, A., and Gawelczyk, A. (1995). On the adaptation of arbitrary normal mutation distributions in evolution strategies: The generating set adaptation.

- In Eshelman, L. J., editor, *Proc. of the Sixth Int. Conf. on Genetic Algorithms*, pages 57–64, San Francisco, CA. Morgan Kaufmann.
- Harvey, S. A. (1999). Design optimization of turbomachinery blades. Cambridge University Dept. Eng., Cambridge, U. K. MEng Report.
- Hess, J. L. (1974). The problem of three-dimensional lifting potential flow and its solution by means of surface singularity distribution. In *Computer Methods in Applied Mechanics and Engineering 4*, pages 283–319. North-Holland Publishing Company.
- Hess, J. L. (1986). Review of the source panel technique for flow computation. In Shaw, R. P., Periaux, J., Chaudouet, A., Wu, J., Morino, L., and Brebbia, C. A., editors, *Innovative Numerical Methods in Engineering: Proc. 4th Intl. Symp Georgia Tech.*, pages 283–319. Springer-Verlag.
- Hess, J. L. and Smith, M. O. (1966). Calculation of Potential Flow About Arbitrary Bodies. In *Progress in Aeronautical Sciences 8*, pages 1–138. Pergamon Press, New York.
- Himmelskamp, H. (1945). Profiluntersuchungen an einem umlaufenden propeller (profile investigations on a rotating airscrew). Gottingen 1945, Mitt. Max-Planck-Institut fur Stromungsforschung Gottingen Nr. 2, 1950. Ph.D. Dissertation.
- Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.
- Holland, J. (1992). Genetic algorithms. Scientific American. July, pages 44-50.
- Hsiao, C. T. and Pauley, L. L. (1999). Numerical computation of tip vortex flow generated by a marine propeller. *Journal of Solar Energy Engineering*, 121:638–645.
- Jones, B. R., Crossley, W. A., and Lyrintzis, A. S. (2000). Aerodynamic and aeroacoustic optimization of rotorcraft airfoils via a parallel genetic algorithm. *Journal of Aircraft*, 37(6):1088–1096.
- Katz, J. and Plotkin, A. (2001). *Low Speed Aerodynamics, 2nd ed.* Cambridge University Press, Cambridge, UK.

- Kinnas, S. A. and Hsin, C.-Y. (1992). Boundary Element Method for the Analysis of Unsteady Flow Around Extreme Propeller Geometries. *AIAA Journal*, 30(3):688–696.
- Laitone, E. V. (1996). Aerodynamic lift at Reynolds numbers below  $7 \times 10^4$ . *AIAA Journal*, 34(9):1941–1943.
- Lampinen, J. (2001). Multi-constrained nonlinear optimization by the differential evolution algorithm. Presented at the 6th On-line World Conference on Soft Computing in Industrial Applications (WSC6). <http://vision.fhg.de/wsc6>.
- Le, S. (2000). SLFFEA: San le's finite element system. <http://www.geocities.com/slffea/slffea.html>.
- Lyon, C. A., Broeren, A. P., Giguère, P., Gopalarathnam, A., and Selig, M. S. (1998). *Summary of Low-Speed Airfoil Data - Volume 3*. SoarTech Publications, Virginia Beach, VA 23451 USA.
- Madsen, H. A. and Christensen, H. F. (1990). On the relative importance of rotational, unsteady and three-dimensional effects on the HAWT rotor aerodynamics. *Wind Engineering*, 14(6):405–415.
- Marco, N., Lanteri, S., Desideri, J. A., and Périaux, J. (1999). A parallel genetic algorithm for multi-objective optimization in computational fluid dynamics. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 445–456. John Wiley and Sons Ltd., West Sussex, England.
- Maskew, B. (1982). Prediction of Subsonic Aerodynamic Characteristics: A Case for Low-Order Panel Methods. *J. Aircraft*, 19(2):157–163.
- Mayer, C., Bechly, M. E., Hampsey, M., and Wood, D. H. (2001). The starting behaviour of a small horizontal-axis wind turbine. *Renewable Energy*, 22(1-3):411–417.
- Michalewicz, Z. and Shoenauer, M. (1996). Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary Computation*, 4(1):1–32.

- Miley, S. J. (1982). A catalog of low Reynolds number airfoil data for wind turbine applications. Technical Report RFP-3387 VC-60, Rockwell International.
- Morino, L., Chen, L.-T., and Suciu, E. O. (1974). Steady and Oscillatory Subsonic and Supersonic Aerodynamics around Complex Configurations. *AIAA Journal*, 13(3):368–374.
- Newman, J. N. (1986). Distributions of Sources and Normal Dipoles over a Quadrilateral Panel. *Journal of Engineering Mathematics*, 20:113–126.
- Obayashi, S. and Takanashi, S. (1996). Genetic optimization of target pressure distributions for inverse design methods. *AIAA Journal*, 34(5):881–886.
- Piegl, L. and Tiller, W. (1997). *The NURBS Book*. Springer-Verlag, Berlin.
- Press, W., Flannery, B., Teukolsky, S., and Vetterling, W. (1992). *Numerical Recipes in C: The art of scientific computing*. Cambridge University Press, Cambridge, U. K. <http://www.nr.com>.
- Preuss, R. D., Suciu, E. O., and Morino, L. (1980). Unsteady Potential Aerodynamics of Rotors with Applications to Horizontal-Axis Windmills. *AIAA Journal*, 18(4):385–393.
- Ronsten, G. (1992). Static pressure measurements on a rotating and a non-rotating 2.375 m wind turbine blade. comparison with 2d calculations. *Journal of Wind Engineering and Industrial Aerodynamics*, 39:105–118.
- Sato, J. and Sunada, Y. (1995). Experimental research on blunt trailing-edge airfoil sections at low Reynolds numbers. *AIAA Journal*, 33(11):2001–2005.
- Schepers, J. G., Brand, A. J., Bruining, A., Graham, J. M. R., Hand, M. M., Infield, D. G., Madsen, H. A., Paynter, R. J. H., and Simms, D. A. (1997). Final report of IEA Annex XIV : Field Rotor Aerodynamics. Technical Report ECN-C-97-027, Netherlands Energy Research Foundation (ECN). [http://www.ecn.nl/unit\\_de/wind/annexxiv/](http://www.ecn.nl/unit_de/wind/annexxiv/).
- Schreck, S., Robinson, M., Hand, M., and Simms, D. (2000). HAWT dynamic stall response asymmetries under yawed flow conditions. NREL/CP-500-27898 (AIAA paper 2000-0040).

- Schwefel, H.-P. (1995). *Evolution and optimum seeking*. Wiley, New york.
- Selig, M. (2000). UIUC Airfoil Coordinates Database. [http://amber.aae.uiuc.edu/m-selig/ads/coord\\_database.html](http://amber.aae.uiuc.edu/m-selig/ads/coord_database.html).
- Selig, M., McGranahan, B., and Broughton, B. (2001). UIUC Low-Speed Airfoil Tests. [http://amber.aae.uiuc.edu/m-selig/uiuc\\_lsat.html](http://amber.aae.uiuc.edu/m-selig/uiuc_lsat.html).
- Selig, M. S. and Maughmer, M. D. (1992). Multipoint inverse airfoil design method based on conformal mapping. *AIAA Journal*, 30(5):1162–1170.
- Simms, D. A., Hand, M. M., Fingersh, L. J., and Jager, D. W. (1999). Unsteady aerodynamics experiment phases II–IV: Test configurations and available data campaigns. Technical Report NREL/TP-500-25950, National Renewable Energy Laboratory, Golden, Colorado. <http://wind2.nrel.gov/amestest/>.
- Snel, H., Houwink, R., and Bosscher, T. (1994). Sectional prediction of lift coefficients on rotating wind turbine blades in stall. Netherlands Energy Research Foundation. ECN Report, ECN-C-93-052, December.
- Sørensen, J. N., editor (1999). *VISCWIND: Viscous Effects on Wind Turbine Blades*. Dept. Energy Engineering, Technical University of Denmark. ET-AFM-9902.
- Spera, D. A., editor (1994). *Wind turbine technology : fundamental concepts of wind turbine engineering*. ASME Press, New York.
- Storn, R. (1996). On the usage of differential evolution for function optimization. In *North American Fuzzy Information Processing Society (NAFIPS) 1996, Berkley*, pages 519–523.
- Storn, R. and Price, K. (1995). Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley, CA.
- Takahashi, S., Obayashi, S., and Nakahashi, K. (1998). Inverse optimization of transonic wing shape for mid-size regional aircraft. AIAA. AIAA paper 98-0601.

- Tomassini, M. (1999). Parallel and distributed evolutionary algorithms. In Miettinen, K., Mäkelä, M. M., Neittaanmäki, P., and Périaux, J., editors, *Evolutionary Algorithms in Engineering and Computer Science*, pages 113–134. John Wiley and Sons Ltd., West Sussex, England.
- Veldhuizen, D. A. V. and Lamont, G. B. (2000). Multiobjective evolutionary algorithms: Analyzing the state-of-the-art. *Evolutionary Computation*, 8(2):125–147.
- Vicini, A. and Quagliarella, D. (1997). Inverse and direct airfoil design using a multi-objective genetic algorithm. *AIAA Journal*, 35(9):1499–1505.
- Wang, F. S. and Sheu, J. W. (2000). Multiobjective parameter estimation problems of fermentation processes using a high ethanol tolerance yeast. *Chemical Engineering Science*, 55:3685–3695.
- Wayner, P. (1991). Genetic algorithms - programming takes a valuable tip from nature. Byte. January, pages 361-368.
- Wolfe, W. P. and Ochs, S. S. (1997). Predicting aerodynamic characteristics of typical wind turbine airfoils using CFD. Technical Report SAND96-2345 UC-261, Sandia National Laboratories, Albuquerque, NM.
- Wood, D. and Rowbotham, T. (1999). Design and testing of high performance blades for a 600 watt horizontal-axis wind turbine. In *Proceedings of the 1999 Australian Wind Energy Conference*.
- Wood, D. H. (1991). A three-dimensional analysis of stall-delay on a horizontal-axis wind turbine. *Journal of Wind Engineering and Industrial Aerodynamics*, 37:1–14.
- Wood, D. H. (2001). A blade element estimation of the cut-in wind speed of a small turbine. *Wind Engineering*, 25(4):249–255.
- Wood, D. H. and Boersma, J. (2001). On the motion of multiple helical vortices. *J. Fluid Mech.*, 447:149–171.
- Xu, G. and Sankar, L. N. (2000). Computational study of horizontal axis wind turbines. *Journal of Solar Energy Engineering*, 122:35–39.

# Appendix A

## B-spline curve fundamentals

B-spline curves and surfaces are a very convenient way of representing aerofoils and three-dimensional blade surfaces. The following pages document some essential background information, together with C-code implementations used in the current project. All theoretical material was first presented in Piegl and Tiller (1997), and the C-code implementations of fundamental B-spline routines were heavily influenced by their code. The reader is heartily directed to this excellent text for more details.

### A.1 Curve derivatives

Piegl and Tiller (1997) give  $\mathbf{C}^{(k)}(u)$ , the  $k$ th derivative of a B-spline curve  $\mathbf{C}(u)$

$$\mathbf{C}^{(k)}(u) = \sum_{i=0}^n N_{i,p}^{(k)}(u) \mathbf{P}_i \quad (\text{A.1})$$

So finding the curve derivatives at a particular parameter location  $u$  along the curve depends largely on finding the derivatives of the curve's Basis functions at that point

$$N_{i,p}^{(k)}(u) = p \left( \frac{N_{i,p-1}^{(k-1)}}{u_{i+p} - u_i} - \frac{N_{i+1,p-1}^{(k-1)}}{u_{i+p+1} - u_{i+1}} \right) \quad (\text{A.2})$$

Code for calculating these derivatives (based on code by Piegl and Tiller (1997)) accompanies this thesis, and is outlined on page 405.

## A.2 Projection of a point onto a curve

A fundamental task encountered during global curve and surface approximation is finding the point on a curve or surface which is closest to some point in space, like a data point in an aerofoil shape description.

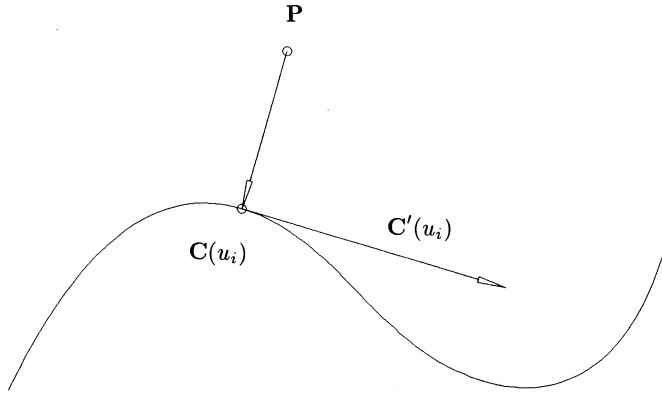


Figure A.1: Projecting a point onto a curve (Piegl and Tiller (1997))

The task is to find the curve parameter,  $u_i$ , which gives the point on the curve closest to the data point  $\mathbf{P}$ . Piegl and Tiller (1997) describe a simple method to do this which uses Newton iteration

1. Evaluate curve points (equation (3.1)) at  $N$  equally-spaced parameter positions,  $u$ , and compute the distance between these points and the data point in question. Select the closest parameter value,  $u_0$  as the starting parameter for the Newton iteration step.
2. Form the dot-product function  $f(u) = \mathbf{C}'(u) \cdot (\mathbf{C}(u) - \mathbf{P})$  and recognise that the distance from any point  $\mathbf{C}(u)$  to any point  $\mathbf{P}$  is minimised when  $f(u) = 0$
3. Let  $u_i$  be the parameter obtained at the  $i$ th Newton iteration, then

$$u_{i+1} = u_i - \frac{f(u_i)}{f'(u_i)} = u_i - \frac{\mathbf{C}'(u_i) \cdot (\mathbf{C}(u_i) - \mathbf{P})}{\mathbf{C}''(u_i) \cdot (\mathbf{C}(u_i) - \mathbf{P}) + |\mathbf{C}'(u_i)|^2} \quad (\text{A.3})$$

with the curve derivatives given by equation (A.1), with a code outline on page 405.

4. Iterate until a predetermined error tolerance has been achieved.

## Appendix B

# Lanchester-Betz limit via actuator disc theory

### B.1 Finding the flow velocity at the rotor

Consider the column of air that passes through the wind turbine's rotor in Figure B.1. The column (with circular cross-section) begins far upstream from the rotor disc with

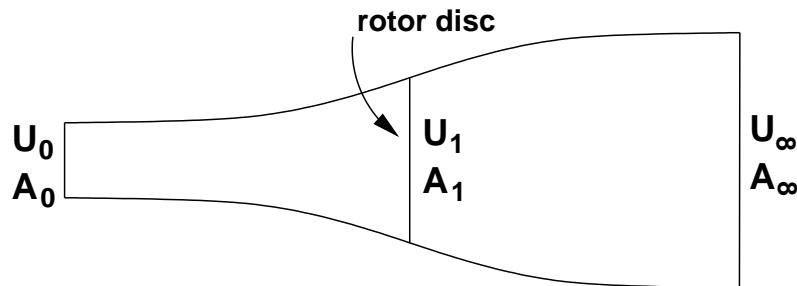


Figure B.1: Lanchester-Betz actuator disc

velocity  $U_0$  and area  $A_0$ . The area of the rotor disk itself is  $A_1$  and the flow velocity through it  $U_1$ . Far downstream of the rotor the flow velocity is  $U_\infty$  with cross-sectional area  $A_\infty$ . We make the following assumptions about this device:

- wind turbine has no hub
- rotor has an infinite number of blades
- machine has no drag

- $U_0, U_1$  and  $U_\infty$  are parallel to the machine's axis
- the machine is a pure energy converter

A proportion of the kinetic energy of the column of air passing through the rotor is extracted and converted by the machine into work. In so doing, it slows the air down ( $U_\infty < U_0$ ).

Assume conservation of mass - the mass flow rate of air through  $A_0, A_1$  and  $A_\infty$  is constant:

$$\begin{aligned} \dot{m} &= \rho AU \\ \therefore A_0 U_0 &= A_1 U_1 = A_\infty U_\infty \end{aligned} \quad (\text{B.1})$$

By the Euler theorem, the thrust force exerted on the rotor is

$$F = \dot{m} \Delta U = \rho A_1 U_1 (U_0 - U_\infty) \quad (\text{B.2})$$

The power absorbed by the rotor is then

$$\begin{aligned} P &= \frac{\text{work}}{\text{time}} = \frac{\text{force} \times \text{distance}}{\text{time}} = F U_1 \\ \therefore P &= \rho A_1 U_1^2 (U_0 - U_\infty) \end{aligned} \quad (\text{B.3})$$

Now, this power comes from the kinetic energy extracted from the column of air

$$\begin{aligned} T_0 &= \frac{1}{2} m U_0^2 \\ &= \frac{1}{2} \rho A_1 U_1 t U_0^2 \quad ; \quad (t = \text{time}) \\ \therefore \Delta T &= T_0 - T_\infty = \frac{1}{2} t \rho A_1 U_1 (U_0^2 - U_\infty^2) \end{aligned} \quad (\text{B.4})$$

Equating (B.3) and (B.4) and solving for  $U_1$  we have

$$\begin{aligned} P &= \frac{\text{work}}{\text{time}} = \rho A_1 U_1^2 (U_0 - U_\infty) = \frac{1}{2} \rho A_1 U_1 (U_0^2 - U_\infty^2) \text{ so} \\ U_1 &= \frac{U_0^2 - U_\infty^2}{2(U_0 - U_\infty)} = \frac{(U_0 + U_\infty)(U_0 - U_\infty)}{2(U_0 - U_\infty)} \\ \therefore \mathbf{U}_1 &= \frac{\mathbf{U}_0 + \mathbf{U}_\infty}{2} \end{aligned} \quad (\text{B.5})$$

The result (B.5) is used in Section 4.4.3 to determine the turbine's wake pitch.

## B.2 Betz limit

Using the result

$$U_1 = \frac{U_0 + U_\infty}{2} \quad (\text{B.6})$$

Equation (B.2) becomes

$$\begin{aligned} F &= \rho A_1 \left( \frac{U_0 + U_\infty}{2} \right) (U_0 - U_\infty) \\ &= \frac{1}{2} \rho A_1 (U_0^2 - U_\infty^2) \end{aligned} \quad (\text{B.7})$$

and Equation (B.3) becomes

$$\begin{aligned} P &= \rho A_1 \left( \frac{U_0 + U_\infty}{2} \right)^2 (U_0 - U_\infty) \\ &= \frac{1}{2} \rho A_1 (U_0^2 - U_\infty^2) (U_0 - U_\infty) \\ &= \frac{1}{2} \rho A_1 (U_0^3 - U_0 U_\infty^2 + U_0^2 U_\infty - U_\infty^3) \end{aligned} \quad (\text{B.8})$$

The maximum power occurs when  $\frac{dP}{dU_\infty} = 0$ :

$$\begin{aligned} \frac{dP}{dU_\infty} &= \frac{1}{2} \rho A_1 (-2U_0 U_\infty + U_0^2 - 3U_\infty^2) \\ \therefore P_{max} \quad \text{when} \quad &3U_\infty^2 + 2U_0 U_\infty - U_0^2 = 0 \end{aligned} \quad (\text{B.9})$$

which makes this a quadratic in  $U_\infty$  with roots

$$U_\infty = \frac{-U_0 \pm 2U_0}{3} \quad (\text{B.10})$$

whose only meaningful root is

$$U_\infty = \frac{U_0}{3} \quad (\text{B.11})$$

Substituting (B.11) back into (B.8) gives

$$P_{max} = \frac{8}{27} \rho A_1 U_0^3 \quad (\text{B.12})$$

In terms of the maximum possible power coefficient this becomes

$$\begin{aligned} C_p^{max} &= \frac{P_{max}}{\frac{1}{2} \rho A_1 U_0^3} \\ &= \frac{16}{27} \approx 0.5926 \end{aligned} \quad (\text{B.13})$$

This result is called the **Betz Limit**, and gives the upper power limit of the ideal wind energy conversion machine modelled by the Lanchester-Betz actuator disc. No actual wind energy conversion system has ever exceeded this limit. Additionally, (B.11) gives the result that, at maximum power, the wind speed through the rotor disc will be

$$U_1^{P_{max}} = \frac{2U_0}{3} \quad (\text{B.14})$$

which, together with the characteristic  $U_1$  vs  $\lambda$  curve of a particular blade design, provides a useful way of checking the validity of the panel method wind turbine performance prediction code.

## **Appendix C**

### **Code Outline (on supplemental CD)**

Pages 345-464 appear as **thesis/code\_outline.pdf** on the supplemental CD.

# Appendix D

## Supplemental CD

A CD-R containing a wide variety of supplemental materials is provided on the back-cover of this thesis:

- **thesis/mark\_thesis.pdf** is this thesis.
- **thesis/code\_outline.pdf** is an outline of the multiobjective evolutionary blade optimisation code developed as part of the current project. It should be considered as an appendix to this thesis. Use the outline to find the relationships between various functions and files when navigating through the code on the CD.
- **3D\_panel\_solver/** is a stand-alone application that combines the 3D panel-method aerodynamic solver with a 3D OpenGL-compatible B-spline blade viewer program. It is accessible by running the **3D\_panel\_solver/test** program. The application will probably need to be built on your system before it will run. See the end of this chapter for more details. Full source code is provided.
- **multiobjective\_blade\_optimisation\_project/pvm3\_blade.tar.bz2** is a complete backup of the master and slave applications used in the current project to evolve the geometries documented earlier, together with lots of miscellaneous stuff. No attempt has been made to make this material user-friendly: more than two years of accumulated results (successful and not-so-successful), dead-ends and mistakes are included, together with the final working code. If you are interested in exploring this material I will help you. See the end of this chapter for details on how to get assistance.

- **multiobjective\_blade\_optimisation\_project/BLADE\_RESULTS.tar.bz2** is a complete archive of the original, seed and evolved Pareto blade designs documented in this thesis and their performance results. As with the **pvm3.blade.tar.bz2** archive mentioned above, no attempt has been made to document or clean up this material, and you will likely need assistance with navigating it. Each blade in the archive is represented by a directory containing a description of the blade, a stand-alone application to evaluate it and a large amount of results-related data. See the end of this chapter for details on how to get help navigating this material.
- **digitise\_aerofoils/** contains a very simple application that can be used to “digitise” scanned-in aerofoil drawings. By using the mouse, a user can click around the profile of an aerofoil, and each point will correspond to an X-Y co-ordinate that will be saved to a data file.
- **digitise\_plots/** similarly, this useful program can be used to digitise X-Y published plots and save the results to file. This was used extensively to compare the results of the current code with published results.
- **software/** some third-party software packages used during the progress of the current project. Some are essential for the functioning of the code on this CD. The packages are:
  - **fltk-1.0.10-source.tar.bz2:** The Fast Light Tool Kit. This was the Graphical User Interface toolkit used to build the front-ends to the applications developed as part of the current project. It was particularly useful because it contains an OpenGL widget that made the 3D blade viewer program possible.
  - **Fltk.ps.gz:** The FLTK programming guide.
  - **pvm3.4.4.tgz:** Parallel Virtual Machine. This was the method used to network slave computers together to achieve a concurrent computing platform. This made the solution of thousands of expensive aerodynamic evaluations considerably faster than would have been the case on a single-processor computer (although the cutting-edge “super-computer” of four

dual-pentium 350 MHz slave computers assembled in 1999 is not very super by 2002 standards!).

- **gnuplot-3.8g.0.tar.gz**: An extremely powerful plotting environment. All of the plots in this thesis were created with gnuplot.
- **gl2ps-0.4.tar.gz**: A very, very nice addition to the 3D blade viewer developed during the current project. All of the 3D images shown in the thesis, including all meshed blade surfaces and spanwise sequences of aerofoils, were generated as encapsulated postscript files from the 3D mesaGL objects by gl2ps.
- **Mesa3D (not included)**: Mesa3D is an open-source OpenGL work-alike. A very powerful application interface for the creation of 3D objects. Visit <http://www.mesa3D.org>
- **beowulf.tar.gz**: some accumulated scripts and information for assembling a Beowulf concurrent computer from inexpensive commodity hardware.
- **slffea-1.1.tar.gz**: San Le’s structural Finite Element package. Used to evaluate the simple eight-node brick blade models.
- **xfoil.tar.gz**: Although not used in the current project, Mark Drela’s XFOIL is very widely used by diverse sections of the aerodynamics community, and is very highly respected. It is included here because I appreciate the gesture of goodwill shown by Professor Drela by making the source code of this excellent piece of software freely available.

## A note on the contents of the CD

You will probably need to install most of the packages contained in the **software/** directory before the bulk of the code can be successfully compiled on your computer. All of the code successfully compiles under gcc (the GNU C/C++ compiler) on RedHat Linux 7.0. Be aware that you will need a video card (and an associated driver) that can handle OpenGL GLX commands for the 3D blade viewer program to function. Building an application will first involve editing the files that start with the word “make”, such as “make\_test”, “make\_master” or “make\_slave”. Change the path to the files and libraries in these files to the paths on your system, then execute the file.

I would very much like to make the source code and results generated as part of my project publicly available. The problem with this, however, is that I do not have the time to clean up and document all of the accumulated work of years of post-graduate study. The CD contains a couple of hundred megabytes of code, results and other stuff, and it would literally take months of work for me to make this useful and user-friendly for another person. If you would like assistance in navigating the contents of the CD please email me at **mark@warpct.com**.