Optimisation of Ducted Propellers for Hybrid Air Vehicles Using High-Fidelity CFD

M. Biava

G. N. Barakos
George.Barakos@glasgow.ac.uk
CFD Laboratory, School of Engineering
James Watt South Building, University of Glasgow
Glasgow
United Kingdom

ABSTRACT

This paper presents performance analysis and design of ducted propellers for lighter-than-air vehicles. High-fidelity computational fluid dynamics simulations were first performed on a detailed model of the propulsor, and the results were in very good agreement with available experimental data. Additional simulations were performed using a simplified geometry, to quantify the effect of the duct and of the blade twist on the propeller performance. It was shown that the duct is particularly effective at low flight speed and that the blades with relatively high twist have better performance over the flight envelope. Design of the optimal twist distribution and of the duct shape was also attempted by coupling the flow solver with a quasi-Newton optimisation method. Flow gradients were computed by solving the discrete adjoint of the Reynolds-averaged Navier-Stokes equations using a fixed-point iteration scheme or a nested Krylov method with deflated restarting. The results show that the ducted propeller propulsive efficiency can be increased by 2%.

Keywords: Hybrid air vehicles; ducted propellers; optimisation; adjoint method; computational fluid dynamics

NOMENCLATURE

a_w	ducted propeller expansion ratio ($a_w = A_e/A_R$)
A_e	area of the exit section of the duct
A_R	area of the propeller disc
c_{root}	blade root chord
C_P	propeller power coefficient ($C_P = P/\rho_\infty n^3 D^5$)
C_T	propeller thrust coefficient ($C_T = T/\rho_\infty n^2 D^4$)
D	propeller diameter
I	cost function [Eq. 9]
J	Jacobian matrix, advance ratio ($J = V_\infty/(nD)$)
$K_{r,n}$	binomial coefficient
M_∞	free-stream Mach number
n	propeller rotational speed (revolutions per second)
P	propeller power
$\boldsymbol{P}$	flow solution vector, primitive variables
R	propeller radius
$\boldsymbol{R}$	flow equation residual vector
Re	Reynolds number ($Re = V_{tip}c_{root}/\nu_\infty$)
T	propeller thrust
V	mesh cell volume
V_∞	free-stream velocity
V_{tip}	blade tip velocity
$\boldsymbol{W}$	flow solution vector, conservative variables

Greek Symbols

$\boldsymbol{\alpha}$	design variables vector [Eq. 9]
β	blade pitch angle
θ	blade twist angle
$\boldsymbol{\lambda}$	adjoint variable vector [Eq. 10]
η	propeller propulsive efficiency ($\eta = J\,C_T/C_P$)
ν_∞	free-stream kinematic viscosity
ρ_∞	free-stream density

Acronyms

ADT	Alternating Digital Tree
CFD	Computational Fluid Dynamics
CFL	Courant–Friedrichs–Lewy
CST	Class/Shape Transformation
FGMRES-DR	Flexible Generalised Minimum Residual with Deflated Restarting
GCG	Generalised Conjugate Gradient
HAV	Hybrid Air Vehicles Ltd

HMB	Helicopter Multi-Block
HPC	High-Performance Computing
IDW	Inverse Distance Weighting
LTA	Lighter-Than-Air vehicle
NS	Navier-Stokes
RANS	Reynolds-Averaged Navier-Stokes
SLSQP	Sequential Least-Squares Quadratic Programming

Subscripts

| ∞ | free-stream value |
| i, j, k | mesh cell indices |

Superscripts

| n | time level |
| T | matrix transpose operator |

1.0 INTRODUCTION

Propellers are very efficient in generating propulsive forces for low-speed aircraft, such as Lighter-Than-Air vehicles (LTAs). Their performance can be augmented by enclosing them in ducts[20], that drive the expansion of their wake, decreasing the final wake velocity and reducing the induced power. Moreover, ducts reduce noise emissions and protect the propeller blades. The ideal power reduction $P_{\text{ducted}}/P_{\text{free}}$ (for a fixed thrust) due to the duct is given by momentum theory[22] and is a function of the ratio a_{w} between the area of the exit section of the duct A_{e} and the propeller disc area A_{R} (see Fig. 1):

$$\frac{P_{\text{ducted}}}{P_{\text{free}}} = \frac{1}{\sqrt{2a_{\text{w}}}} \qquad \dots (1)$$

The higher the value of a_{w}, the higher the power reduction. In reality, however, the value of a_{w} is limited by the adverse pressure gradient inside the duct diffuser, where flow separation may also take place. It follows that accurate predictions of the propulsor aerodynamics require high-fidelity Computational Fluid Dynamics (CFD) to capture the complex propeller flow and the interaction between the propeller wake and the duct. The aerodynamic analysis and the optimisation of a ducted propeller using high-fidelity flow modelling are pursued in this paper.

A representative geometry of a ducted LTA propeller with 1.2 m radius was provided by *Hybrid Air Vehicles Ltd* (HAV), which includes the rotor blades, the duct, the nacelle enclosing the piston engine and the cooling devices (see Fig. 2). The performance of this baseline design was studied by solving the Reynolds-Averaged Navier-Stokes (RANS) equations with the k-ω turbulence model[33] by means of the HMB3 flow solver[4,31]. The obtained numerical results were compared with experimental data available from static tests conducted by HAV on a dedicated test rig (see Fig. 3), and very good agreement was observed between the predicted and measured power and thrust.
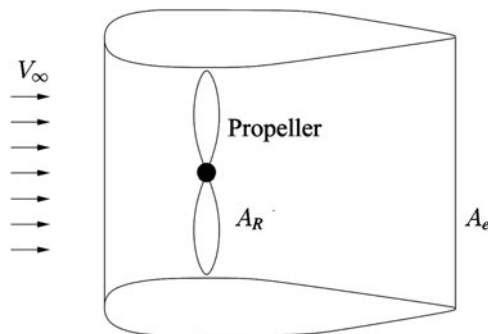
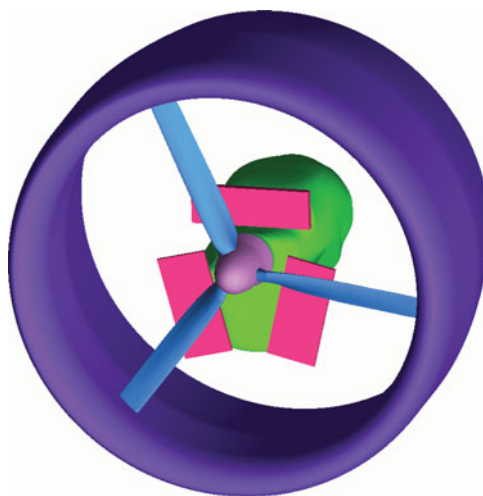Figure 1. Schematic of a ducted propeller.

Figure 2. (Colour online) Ducted propeller with nacelle, radiators and cooler.

The CFD model for the detailed geometry of the propulsor is, however, computationally expensive. Therefore, a simplified geometry was also considered, where the cooling devices were removed and the engine nacelle was replaced by an axisymmetric body (Fig. 4). This simpler geometry was used to quantify the effect of the duct by comparing the predicted thrust and power consumption with those of the unducted (or 'free') propeller (Fig. 5). The results revealed that the duct led to a substantial power reduction in low-speed operations, which cover a significant part of the LTA flight envelope.

The shape of the baseline rotor blade was optimised without the duct, and one question is whether this design is also optimal for the ducted configuration. This work focuses on the twist distribution, while other blade design parameters, such as aerofoils, chord distribution and sweep, are kept constant. To gain a basic understanding of the twist effect, the performance of the baseline blade, which is highly twisted, was compared with a low-twist blade. The low-twist blade design was the outcome of an optimisation study for low speeds, using a simple panel method[1]. However, the high-fidelity CFD results presented in this paper show that the high-twist blade is superior at low and high speed.

Figure 3. (Colour online) Propulsor test rig (Source *Hybrid Air Vehicles Ltd*).

Figure 4. (Colour online) Ducted propeller.

An optimisation of the blade twist distribution with CFD was also attempted using the simplified geometry. The HMB3 flow solver was coupled with an external optimiser and an in-house parametrisation software for the blade surface. The computational mesh is updated at each optimisation cycle using an advanced deformation algorithm based on Inverse Distance Weighting (IDW)[29]. To minimise the expensive flow solution evaluations, a quasi-Newton optimisation method was employed. The optimiser used a Sequential Least-Squares Quadratic Programming (SLSQP) method[17], as provided in the software library *NLopt*[16]. The required flow derivatives with respect to the design variables were computed using an
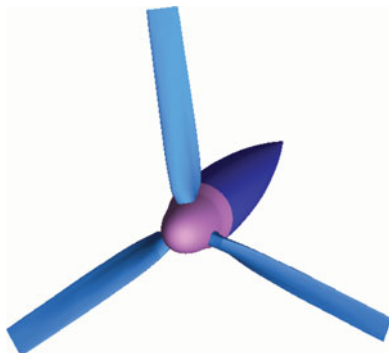
Figure 5. (Colour online) Free propeller.

implicit Fixed-Point Iteration (FPI) scheme[6] or a recently developed Flexible Generalised Minimum Residual solver with Deflated Restarting (FGMRES-DR) nested with GMRES-DR (or standard GMRES) as a pre-conditioner[13,21,26]. While the FPI scheme has low memory usage, the FGMRES-DR method is more robust and faster to converge but more expensive in memory. The blade with optimised twist distribution has 1% higher propulsive efficiency with respect to the baseline design. The propulsive efficiency for the ducted propeller is defined as

$$\eta = J \frac{C_T}{C_P}, \qquad \qquad \dots (2)$$

where the thrust coefficient C_T includes the contribution of both the propeller and the duct. The reference area used for computing the coefficients is the propeller disc area, which is kept fixed throughout the optimisation process. To further increase the propulsor performance, the duct diffuser shape and dimensions were also considered for the optimisation, in addition to the twist distribution, and the resulting design has 2% more propulsive efficiency than the baseline.

The paper is structured as follows. Section 2 describes HMB3 and adjoint method, as well as the coupling strategy with the external optimiser. Section 3 presents the numerical results. The performance of the baseline propeller is firstly analysed with the detailed model, and a comparison with experimental data is presented. The effect of the duct and of the twist distribution is then assessed using the simplified geometry. The results of the twist and duct optimisations are finally presented. Conclusions are given in Section 4.

2.0 NUMERICAL METHODS

2.1 The optimisation tool chain

Gradient-based optimisation is an efficient and widely used method for aerodynamic shape optimisation problems, since it minimises the required number of flow solutions. It requires, however, the flow derivatives with respect to the design variables, which can be very expensive to compute when high-fidelity CFD is employed. An economic way to obtain the flow gradients with CFD is represented by the adjoint method, which reduces the cost of flow gradients evaluation to about twice the cost of the base flow solution, regardless of the number of design variables.
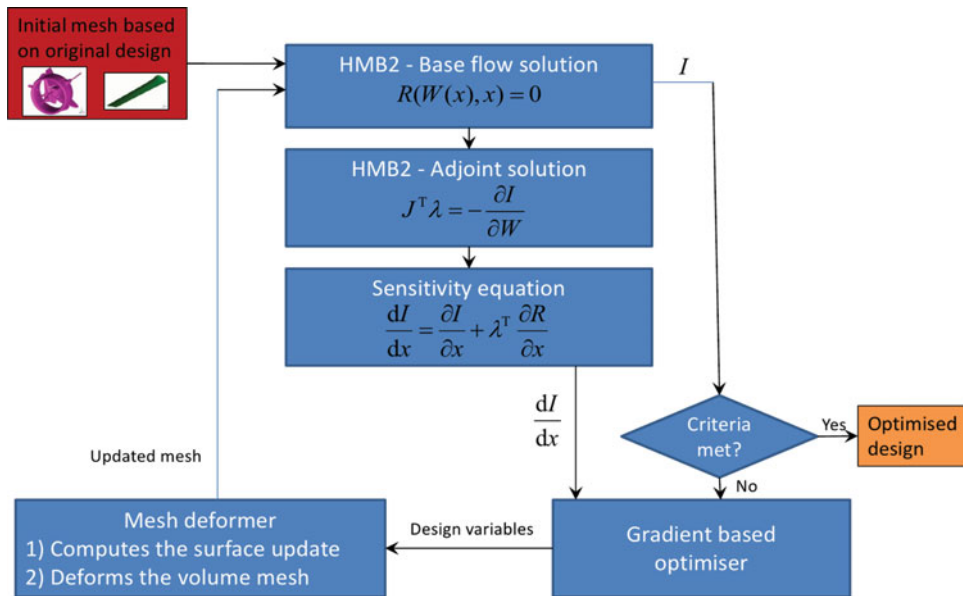
Figure 6. (Colour online) Flow chart of the optimisation process.

The HMB3 flow solver employs two adjoint solvers based on FPI scheme[6] and FGMRES-DR[13,21], which can be interfaced to any gradient-based optimisation tool to solve design problems. The current implementation employs a Sequential Quadratic Programming (SQP) optimisation algorithm[17], as implemented in the software library *NLopt*[16]. It represents the objective function as a quadratic approximation to the Lagrangian and uses a dense SQP algorithm to minimise a quadratic model of the problem. It is intended for problems with up to a few hundred constraints and variables, since the required computer memory scales with the square of the problem dimension. The SQP algorithm is implemented in a separate tool, and it is interfaced with HMB3 using files for data exchange.

The mapping between the design space and the shape to be optimised is problem specific. In aerofoil design, usually a polynomial basis, such as Chebyshev or Bernstein polynomials, is used to represent the aerofoil. For three-dimensional objects (propeller blades, propulsor duct, etc.) more complex parametrisations may be required.

The design optimisation tool chain is described in Fig. 6 and can be summarised as follows.

1 The first step solves the flow around the baseline geometry of the shape under investigation (e.g. aerofoil, blade).

2 The cost functional I (e.g. drag to lift ratio, torque to thrust ratio) is evaluated from the flow solution.

3 The adjoint problem is solved to compute the gradient $\mathrm{d}I/\mathrm{d}\alpha$ of the cost functional with respect to the design variables vector α.

4 The cost functional and its gradient are fed to the gradient-based optimiser, which produces a new set of design variables, corresponding to a design candidate in the search direction.

5 An external parametrisation software updates the aerodynamic shape.

6   A mesh deformation algorithm computes the new volume mesh.

7   HMB3 solves the new flow problem and recomputes the cost functional I and its gradient $dI/d\alpha$.

Steps 2-7 are repeated for several design cycles until determined convergence criteria are met.

2.2 The Flow Solver

The following contains a brief outline of the approach used in HMB3. The Navier-Stokes (NS) equations are discretised using a cell-centred finite-volume approach. The computational domain is divided into a finite number of non-overlapping control-volumes V_{ijk}, and the governing equations are applied to each cell. Also, the NS equations are re-written in a curvilinear co-ordinate system which simplifies the formulation of the discretised terms since body-conforming meshes are adopted here. The spatial discretisation of the NS equations leads to a set of ordinary differential equations in real time:

$$\frac{d}{dt}\left(\boldsymbol{W}_{ijk}V_{ijk}\right) = -\boldsymbol{R}_{ijk}\left(\boldsymbol{W}\right), \qquad \ldots (3)$$

where $\boldsymbol{W}$ and $\boldsymbol{R}$ are the vectors of cell conserved variables and residuals, respectively. The convective terms are discretised using Osher's upwind scheme[11] for its robustness, accuracy and stability properties. Monotonic Upstream-Centred Scheme for Conservation Laws (MUSCL) variable extrapolation[14] is used to provide second-order accuracy with the Van Albada limiter[32] to prevent spurious oscillations around shock waves. Alternatively, the classical Roe's scheme[25], an all-Mach variant of the Roe's scheme[24], and the AUSM+ scheme[19], can be used. Boundary conditions are set using ghost cells on the exterior of the computational domain. At the far-field, ghost cells are set at the free-stream conditions. At solid boundaries, the no-slip condition is set for viscous flows.

The integration in time of Equation (3) to a steady-state solution is performed using a fully implicit time-marching scheme by:

$$\frac{\boldsymbol{W}_{ijk}^{n+1} - \boldsymbol{W}_{ijk}^{n}}{\Delta t} = -\frac{1}{V_{ijk}}\boldsymbol{R}_{ijk}\left(\boldsymbol{W}_{ijk}^{n+1}\right), \qquad \ldots (4)$$

where $n+1$ denotes the real time $(n+1)\Delta t$. For steady-state problems, the real time is replaced by a pseudo-time (τ), which is also used for unsteady problems according to the dual-time stepping scheme of Jameson[15]. Equation (4) represents a system of non-linear algebraic equations, and to simplify the solution procedure, the flux residual $\boldsymbol{R}_{ijk}(\boldsymbol{W}_{ijk}^{n+1})$ is linearised in time as follows:

$$\boldsymbol{R}_{ijk}\left(\boldsymbol{W}^{n+1}\right) \approx \boldsymbol{R}_{ijk}^{n}\left(\boldsymbol{W}^{n}\right) + \frac{\partial \boldsymbol{R}_{ijk}}{\partial \boldsymbol{W}_{ijk}}\Delta \boldsymbol{W}_{ijk}, \qquad \ldots (5)$$

where $\Delta W_{ijk} = W_{ijk}^{n+1} - W_{ijk}^n$. Equation (4) now becomes the following linear system:

$$\left[\frac{V_{ijk}}{\Delta t} \mathbf{I} + \frac{\partial \mathbf{R}_{ijk}}{\partial \mathbf{W}_{ijk}} \right] \Delta W_{ijk} = -\mathbf{R}_{ijk}^n (\mathbf{W}^n) \qquad \ldots (6)$$

The left-hand side of Equation (6) is then rewritten in terms of primitive variables $\mathbf{P}$:

$$\left[\left(\frac{V_{ijk}}{\Delta t} \right) \frac{\partial \mathbf{W}_{ijk}}{\partial \mathbf{P}_{ijk}} + \frac{\partial \mathbf{R}_{ijk}}{\partial \mathbf{P}_{ijk}} \right] \Delta P_{ijk} = -\mathbf{R}_{ijk}^n (\mathbf{W}^n) \qquad \ldots (7)$$

and the resulting linear system is solved with a Generalised Conjugate Gradient (GCG) iterative solver[3]. Since at steady state the left-hand side of Equation (7) must be zero, the Jacobian $\partial \mathbf{R}/\partial \mathbf{P}$ can be approximated by evaluating the derivatives of the residuals with a first-order scheme for the inviscid fluxes. The first-order Jacobian requires less storage and, being more dissipative, ensures a better convergence rate to the GCG iterations.

The steady-state solver for turbulent flows is formulated and solved in an identical manner to that of the mean flow. The eddy-viscosity is calculated from the latest values of k and ω, for example, and is used to advance the mean and the turbulent flow solutions. An approximate Jacobian is used for the source term of the models by only taking into account the contribution of their dissipation terms, i.e. no account of the production terms is taken on the left-hand side of the system.

2.3 The adjoint solvers

An efficient way to compute the derivatives of the cost functional with respect to design variables is via solving the *sensitivity equation* casted in adjoint form. The underlying idea is to write explicitly the cost functional I as a function of the flow variables W and of the design variables α, that is, $I = I(W(\alpha), \alpha)$. The flow variables are subject to satisfy the fluid dynamics governing equations (e.g. the NS equations) written in compact form as:

$$R(W(\alpha), \alpha) = 0 \qquad \ldots (8)$$

Formally, taking the derivative of I with respect to α, we obtain:

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha} = \frac{\partial I}{\partial \alpha} + \frac{\partial I}{\partial W} \frac{\partial W}{\partial \alpha} \qquad \ldots (9)$$

By introducing the adjoint variable λ as the solution of the following linear system:

$$\left(\frac{\partial \mathbf{R}}{\partial W} \right)^{\mathrm{T}} \lambda = - \left(\frac{\partial I}{\partial W} \right)^{\mathrm{T}} \qquad \ldots (10)$$

Equation (9) can be rewritten as:

$$\frac{\mathrm{d}I}{\mathrm{d}\alpha} = \frac{\partial I}{\partial \alpha} + \lambda^{\mathrm{T}} \frac{\partial \mathbf{R}}{\partial \alpha} \qquad \ldots (11)$$

The computational cost of the dual-sensitivity problem (10)-(11) scales with the number of outputs, since the right-hand side of (10) depends on I, but it is independent of the input

parameters. The adjoint form of the sensitivity equation is particularly efficient for cases where the number of (output) cost functionals is small, while the number of (input) design variables is large.

2.3.1 Fixed-point iteration scheme

The linear system (10) for computing the adjoint variable can be solved with a FPI scheme, where an approximation $\hat{J}$ of the linear system matrix J, with better condition number, is introduced as a pre-conditioner to advance the solution at each iteration[10,12]. The FPI scheme reads:

$$\hat{J}^{\mathrm{T}} \Delta \boldsymbol{\lambda}^{n+1} = -\left(\frac{\partial I}{\partial \boldsymbol{W}}\right)^{\mathrm{T}} - J^{\mathrm{T}} \boldsymbol{\lambda}^n, \qquad \dots (12)$$

where

$$J = \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{W}}, \qquad \dots (13)$$

$$\hat{J} = \frac{V}{\mathrm{CFL}\, \Delta t} \boldsymbol{I} + \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{W}}\right]^{\mathrm{1st}}, \qquad \dots (14)$$

$$\Delta \boldsymbol{\lambda}^{n+1} = \boldsymbol{\lambda}^{n+1} - \boldsymbol{\lambda}^n \qquad \dots (15)$$

The pre-conditioner $\hat{J}$ is the matrix used for the base flow implicit update (7) and consists of the sum of the stabilising pseudo-time derivative term and of the first-order residual Jacobian (e.g. the Jacobian evaluated using a first-order scheme for the inviscid fluxes and having the sparsity pattern induced by the inviscid discretisation scheme of the equations). The CFL number in the pseudo-time derivative term can be adjusted to obtain the desired convergence speed of the FPI (12), which is solved using a GCG iterative scheme[3].

Note that the iterative scheme (12) does not require the full, exact Jacobian J, but only the matrix-vector product $J^{\mathrm{T}} \boldsymbol{v}$. The computer code to perform the product is obtained by automatic differentiation in reverse mode of the flow steady residual function. This avoids storing J, and hence the computation of the adjoint sensitivities adds only a small memory overhead to the base solver. More details about the development of the fully implicit FPI scheme are given in Ref.[6].

The steady RANS equations for the flows considered in this work are not always asymptotically converging. In fact, they sometimes enter limit cycles, because small residual unsteadiness is present in the flow field, due to the complexity of the geometry. This flow unsteadiness leads to the presence of unstable modes in the associated adjoint problem, which may hinder the convergence of the FPI[8]. To stabilise the FPI, HMB3 embeds the Recursive Projection Method (RPM)[9,30], which, for mildly unstable flows, allows to detect and isolate the unstable modes and to restore the FPI convergence.

2.3.2 Nested Krylov-subspace method

The FPI scheme presented in the previous section has very low memory requirements, and it is suited to large applications on High-Performance Computing (HPC) systems with low memory per computing core. For stiff problems, however, the convergence rate of the FPI scheme can be poor, or it may even diverge. For this reason, a new Krylov-subspace solver based on the Generalised Minimum Residual (GMRES) method[28] is introduced here.

GMRES is a robust method for the solution of a non-symmetric linear system, based on the minimisation of the equations residual over a Krylov subspace generated using Arnoldi's method[2]. A well-known theoretical result states that GMRES is guaranteed to converge in at most n steps, where n is the dimension of the system matrix[27]. The GMRES is easily parallelizable, as it involves only matrix-vector products and vector inner products. The drawback of the method is its memory requirements, because it needs to store all the base vectors of the Krylov subspace, whose size increases by one at each Arnoldi's iteration. To limit the memory demand, a restarted version of the method is employed, where the Krylov subspace is discarded after a predefined number m of Arnoldi's iterations, and the process is restarted from the last solution. The method is denoted as GMRES(m), and its memory requirement is equal to m vectors of size n. Since the information contained in the Krylov subspace is lost after a restart, the restarted GMRES has poorer convergence than full GMRES, and the process may even stagnate if the size m of the Krylov subspace is not sufficiently large.

The robustness and efficiency of Krylov subspace methods relies on the choice of a suitable pre-conditioner[27]. In HMB3, the no-fill Incomplete Lower-Upper decomposition, or ILU(0), is used[5]. For parallel computations, the ILU(0) terms are computed locally on each processor because of the poor scalability of a fully coupled decomposition. For large and tough linear problems, however, neglecting the coupling terms may prevent convergence of the linear solver. In our tests, GMRES(m) with decoupled ILU(0) running on 128 cores failed to converge the adjoint problem associated to the large, three-dimensional, RANS flow solutions considered in this work, for values of m up to 500, which was the maximum allowed by the used HPC system memory.

A significant improvement over the restarted GMRES is GMRES with deflated restarting (GMRES-DR)[21]. At the end of a cycle, k approximate eigenvectors associated with the smallest eigenvalues are extracted from the Krylov subspace, and used to augment the subspace at the next cycle. This operation is referred to as *deflation* of the eigenvalues. The method is denoted as GMRES-DR(m,k), and the memory requirement is the same as that of GMRES(m). Note that GMRES-DR($m,0$), i.e. with no deflation, reduces to standard GMRES. The deflation of the eigenvalues greatly improves the convergence rate of GMRES, and the minimum size m of the Krylov subspace to avoid stagnation can also be significantly reduced. Nevertheless, even GMRES-DR was stagnating or having slow convergence for the considered adjoint problems using less than 500 Krylov base vectors. This was mainly due to the reduced accuracy of the decoupled ILU(0) pre-conditioner, which is particularly severe when a large number of parallel processes is used.

To restore the effectiveness of the pre-conditioner, while maintaining the scalability of the method, we introduced a nested GMRES-DR solver as the pre-conditioner for the (outer) GMRES-DR cycles. To allow for this, a flexible variant of the outer solver was needed[26], at the cost of higher memory requirement, since two vectors need to be stored at each Arnoldi's iteration. The development of flexible GMRES-DR (FGMRES-DR) is described in Ref. 13. The inner GMRES-DR solver inverts the pre-conditioning matrix $\hat{J}$ defined in (14), and uses, in turn, the decoupled ILU(0) pre-conditioner. The CFL number in the pseudo-time derivative term can be used to regulate the diagonal dominance of $\hat{J}$ and to obtain the desired convergence rate for the inner cycles. The whole scheme may be denoted as FGMRES-DR(m,k)-GMRES-DR(m_p,k_p), or FGMRES-DR(m,k)-GMRES(m_p) when standard GMRES is used for the inner cycles. This nested Krylov-subspace solver proved to be able to converge tough adjoint problems using about 200 Krylov base vectors for the outer cycles, where GMRES(m) and GMRES-DR(m,k) with decoupled ILU(0) pre-conditioning were failing.
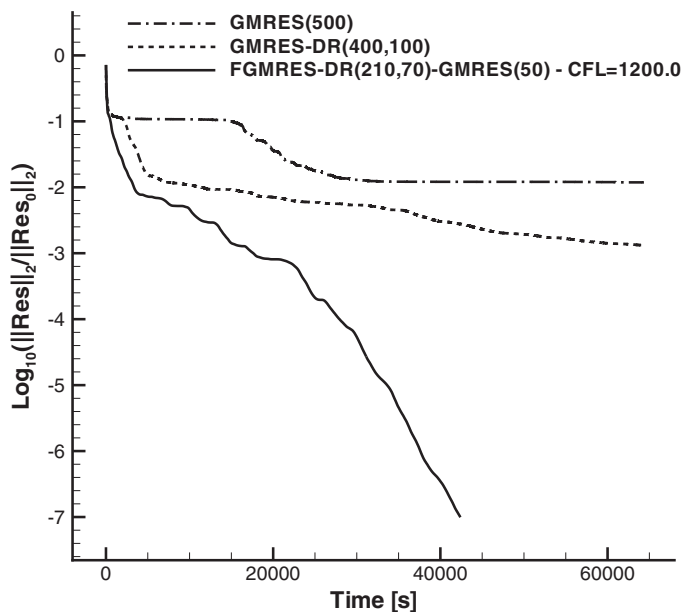
Figure 7. Convergence history of the Krylov-subspace solvers.

Like for the FPI scheme, the implementation of the nested Krylov-subspace solver does not require the Jacobian matrix J, but only the matrix-vector product $J^T v$. Therefore, J is never stored explicitly, and the product $J^T v$ is computed by means of automatically differentiated code of the residual function.

Figure 7 shows the convergence history of the developed Krylov-subspace methods for the solution of the adjoint equations (10) of a three-dimensional RANS flow. In particular, we considered the flow through the simplified model of the ducted propeller (see Fig. 4) at cruise condition, solved using a grid of about 9 m cells. The k-ω turbulence model was used, and the derivatives of its terms were fully accounted for in the system matrix J, thus leading to a tough linear problem with 63 m degrees of freedom. The adjoint solvers were run on 128 Xeon E5-2650 2 GHz cores, and 4 GB of memory was available for each core. The GMRES(500) failed to converge, and it stagnated after dropping the residual by 2 orders of magnitude, whereas GMRES-DR(400,100) showed very slow convergence. The nested Krylov-subspace solver FGMRES-DR(210,70)-GMRES(50), with $\hat{J}$ computed using a CFL equal to 1,200, was instead able to fully converge the solution. The performance of the nested method were also competitive in terms of run time, as it took 12 hours to drop the residual by 7 orders of magnitude, which is half of the time that was needed to converge the base flow.

2.4 Mesh deformation

To adapt the volume mesh to the surface generated by the parametrisation software at each optimisation cycle, an advanced mesh deformation algorithm has been implemented in HMB3, based on Inverse Distance Weighting (IDW)[29]. IDW is an interpolation method that calculates the values at given points with a weighted average of the values available at a set of known points. The weight assigned to the value at a known point is proportional to the inverse of the distance between the known and the given point.

Given N samples $\boldsymbol{u}_i = u(\boldsymbol{x}_i)$ for $i = 1, 2, \ldots, N$, the interpolated value of the function $\boldsymbol{u}$ at a point $\boldsymbol{x}$ using IDW is given by:

$$\boldsymbol{u}(\boldsymbol{x}) = \begin{cases} \dfrac{\sum\limits_{i=1}^{N} w_i(\boldsymbol{x})\boldsymbol{u}_i}{\sum\limits_{i=1}^{N} w_i(\boldsymbol{x})}, & \text{if } d(\boldsymbol{x}, \boldsymbol{x}_i) \neq 0 \text{ for all } i \\ \boldsymbol{u}_i, & \text{if } d(\boldsymbol{x}, \boldsymbol{x}_i) = 0 \text{ for some } i \end{cases}, \qquad \ldots (16)$$

where

$$w_i(\boldsymbol{x}) = \frac{1}{d(\boldsymbol{x}, \boldsymbol{x}_i)^p} \qquad \ldots (17)$$

In the above equations, p is any positive real number (called the *power parameter*), and $d(\boldsymbol{x}, \boldsymbol{y})$ is the Euclidean distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ (but any other metric operator could be considered as well).

The method in its original form tends to become extremely expensive as the sample data set gets large. An alternative formulation of the Shepard's method, which is better suited for large-scale problems, has been proposed by Renka[23]. In the new formulation, the interpolated value is calculated using only the k-nearest neighbours within the R-sphere (k and R are given, fixed, parameters). The weights are slightly modified in this case:

$$w_i(\boldsymbol{x}) = \left(\frac{\max(0, R - d(\boldsymbol{x}, \boldsymbol{x}_i))}{Rd(\boldsymbol{x}, \boldsymbol{x}_i)} \right)^2, \quad i = 1, 2, \ldots, k \qquad \ldots (18)$$

If this interpolation formula is combined with a fast spatial search structure for finding the k-nearest points, it yields an efficient interpolation method suitable for large-scale problems.

The modified IDW interpolation formula is used in HMB3 to implement mesh deformation in an efficient and robust way. The known displacement of points belonging to solid surfaces represent the sample data, while the displacements at all other points of the volume grid are computed using Equation (16) with the weights (18). For fast spatial search of the sample points, an Alternating Digital Tree (ADT) data structure[7] is used. A blending function is also applied to the interpolated displacements so that they smoothly tend to zero as the distance from the deforming surface approaches R.

3.0 NUMERICAL RESULTS

3.1 CFD mesh and problem set-up

To model the propulsor, a system of grids is used along with the sliding mesh method available in HMB3. The sliding grids allow for the front and rear part of the intake to be meshed independently of the blades or other parts of the geometry, making the parameterisation and optimisation of the propulsor easier. This allows, for instance, to compare the performance of different blades, or of the ducted and unducted propellers, without the need to re-mesh the complete geometry.
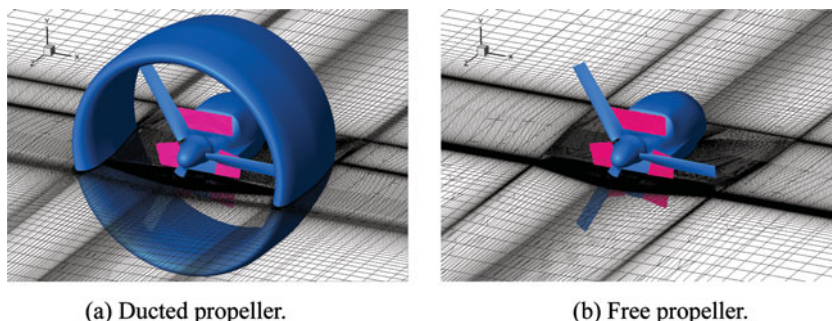
(a) Ducted propeller. (b) Free propeller.

Figure 8. (Colour online) Propulsor grids including nacelle and coolers.
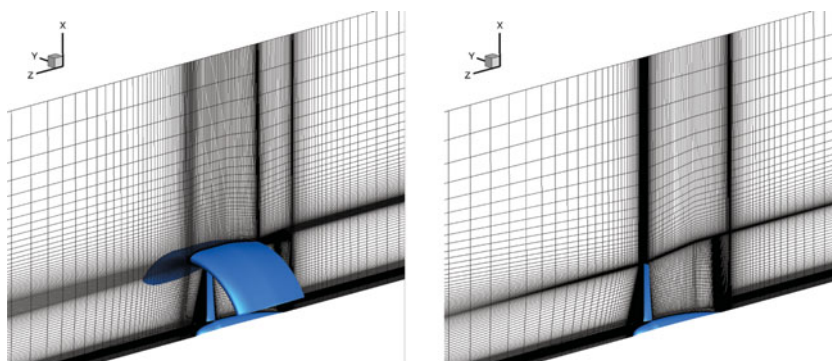
(a) Ducted propeller. (b) Free propeller.

Figure 9. (Colour online) Propulsor grids for the simplified cases.

The CFD grid for the detailed model of Fig. 2 counts 1,171 blocks and 33.5 m cells (Fig. 8a). The radiators and coolers were included in the CFD simulations as infinitely thin walls, by setting up a solid wall boundary at the interface between two adjacent grid blocks[34]. The location of these thin walls, which must have rectangular shape, is specified by giving their corner point coordinates. HMB3 automatically localises the set of block boundary cells that approximate best the rectangular region and flags these cells as solid boundaries. The grid for the unducted case (Fig. 8(b)) counts 921 blocks and 31.3 m of cells. The commercial software Ansys ICEMCFD was used to generate the meshes.

The simplified ducted and free propellers of Figs 4 and 5 are axisymmetric and, consequently, only a cylindrical sector around one blade was meshed. Periodic boundary conditions were used to account for the other blades. The grid for the ducted and free propellers are shown in Figs 9(a) and 9(b), respectively. The former grid has 324 blocks and 9.8 m cells, while the latter has 256 blocks and 9.3 m cells.

For both the detailed and the simplified models, to reduce the computation of the flow to the solution of a steady problem, the flow equations were formulated in the rotating reference frame attached to the blade. This is obvious in the case of the free propeller. For the ducted case, the duct is rotating with the propeller angular velocity, but in the opposite sense, in the blade attached reference frame. This is accounted for through the boundary conditions by imposing this rotational velocity to the boundary faces on the duct surface. The duct geometry

(a) Axial velocity in a slice of the flow field (ducted propeller, point A of Figure 11).

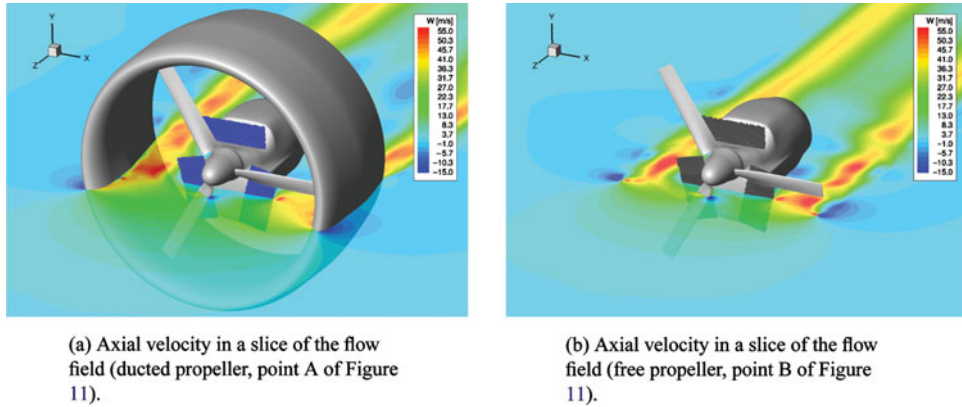(b) Axial velocity in a slice of the flow field (free propeller, point B of Figure 11).

Figure 10. (Colour online) CFD simulations of the ducted and free propellers with nacelle, radiators and cooler.

is invariant under rotations around the propeller axis, so the flow can be still modelled as steady in the blade reference frame.

The detailed model is not invariant under rotations, due to the presence of the cooling devices and the shape of the nacelle. The formulation as a steady problem represents an approximation in this case. The comparison with experimental data shown in the next section confirms the validity of this assumption.

3.2 Assessment of the numerical model

Experimental data is available from static tests of the ducted propeller conducted by *Hybrid Air Vehicles Ltd*. The test conditions were replicated in CFD simulations for both ducted and free propeller cases, using the detailed and the simplified geometries. The flow was modelled using the RANS equations endowed with the k-ω turbulence model. The choice of the model was dictated by its simplicity and robustness and also by the good agreement between available static test data for the propulsor and our preliminary computations. The Reynolds number, based on the blade root chord and blade tip speed, was between 0.795×10^6 and 3.18×10^6, while the tip Mach number was between 0.175 and 0.7.

The complex flow through the ducted and free propellers is shown in Fig. 10, where a significant blockage effect due to the nacelle, radiators and cooler can be observed. The predicted performances are reported in Fig. 11, which also shows the static test measurements for the ducted case. Note that the data is scaled with a reference value to respect the confidentiality of the real-world design. The plot reveals the importance of using a more detailed representation of the real geometry. Indeed, there is a good agreement between the CFD results for the realistic model and the experiments, while the simplified model significantly underestimates the required power. Nonetheless, the simplified model is significantly more economic and was selected to study the effect of the duct, and for the analysis and optimisation of the blade twist distribution. The analysis of the results reveals also the beneficial contribution of the duct, that allows to increase significantly the thrust for a given power (compare, for instance, points A and B in Fig. 11).
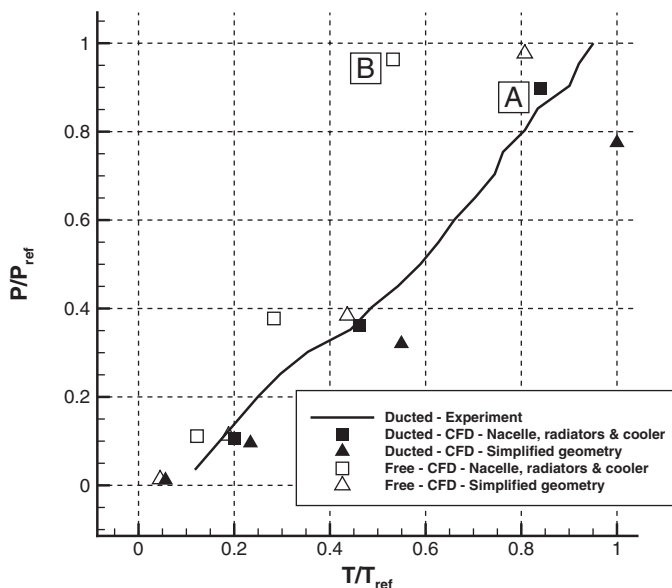
Figure 11. Computed and experimental performance of the ducted and free propellers.

3.3 Effect of the Duct

To assess the effect of the duct, the performance of the ducted and free propellers were computed using the baseline high-twist blade and the simplified geometry of Figs 4 and 5. The flow was modelled using the RANS equations and the k-ω turbulence model. The Reynolds number was 3.18×10^6 and the tip Mach number was 0.7. Three advance ratios were considered, that is, $J = 0, 0.136, 0.540$, which correspond to static conditions, cruise speed and dash speed, respectively.

Figures 12(a) and 12(b) show, respectively, the wake visualised using the Q-criterion and the surface pressure coefficient (based on the free-stream velocity) on the ducted high-twist propeller at dash speed. The blade loading shows the importance of the outboard part of the blades with high suction produced near 60% of the blade radius. The complex swirling flow is also well resolved with the current set of CFD grids.

Figures 13(a) and 13(b) compare the flow through the ducted and unducted propellers at the same flight conditions. The differences in the flow-field are substantial and well captured by the simulation. It is clearly shown how the rear part of the duct operates as a diffuser, decreasing the propeller wake speed and increasing the static pressure, so as to induce a positive contribution to the overall thrust.

The predicted performance of the ducted and free propellers are given in Table 1. In static conditions ($J = 0$), the performance of the ducted propeller is significantly higher than that of the free propeller. For instance, at a low blade pitch angle the ducted propeller gives 24% more thrust with 25% less power. At cruise speed ($J = 0.136$), the contribution of the duct is still important, and at a medium pitch angle, the ducted propeller gives 10% more thrust with 22% less power.

When the speed is higher, the positive effect of the duct reduces due to the increase of the duct drag that compensates the positive contribution of the static pressure in the diffuser. Consider, for example, the flight condition at dash speed ($J = 0.54$). At fixed blade pitch

Table 1
Computed performance of the propulsor

Twist	Configuration	J [-]	β	$C_T/C_{T,\text{ref}}$ [-]	$C_P/C_{P,\text{ref}}$ [-]	η/η_{ref} [-]
High	Ducted	0.000	Low	5.417	1.588	0.000
High	Ducted	0.000	Medium	6.861	2.206	0.000
High	Ducted	0.136	Medium	5.278	2.206	0.586
High	Ducted	0.540	Low	1.333	1.059	1.241
High	Ducted	0.540	Medium	2.056	1.588	1.276
High	Ducted	0.540	High	2.444	1.882	1.276
High	Free	0.000	Low	4.361	2.000	0.000
High	Free	0.136	Medium	4.778	2.676	0.431
High	Free	0.540	Low	2.361	1.824	1.276
Low	Free	0.136	Medium	5.278	2.412	0.534
Low	Free	0.540	Low	1.000	1.000	1.000

(a) Wake visualisation using Q-criterion ($Q = 10^{-3}$).

(b) Surface pressure coefficient.
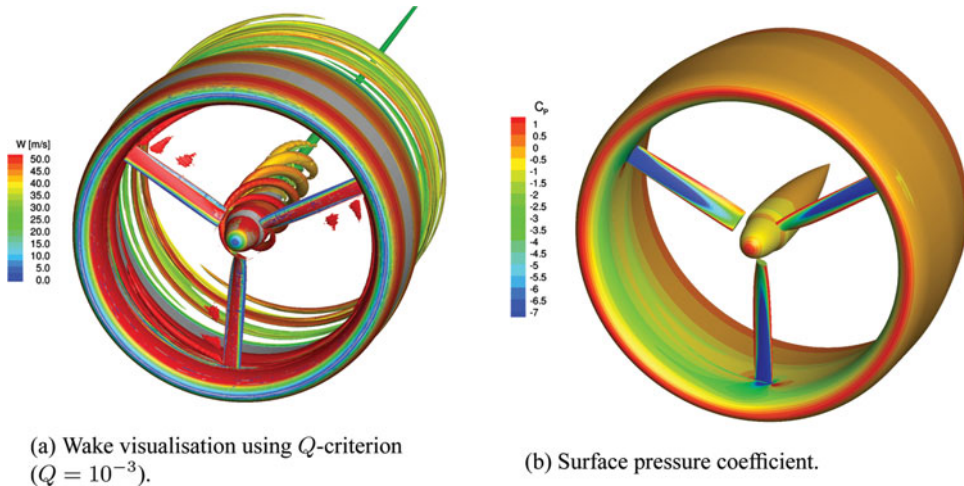
Figure 12. (Colour online) CFD simulation of the ducted propeller at dash speed.

(a) Ducted propeller.
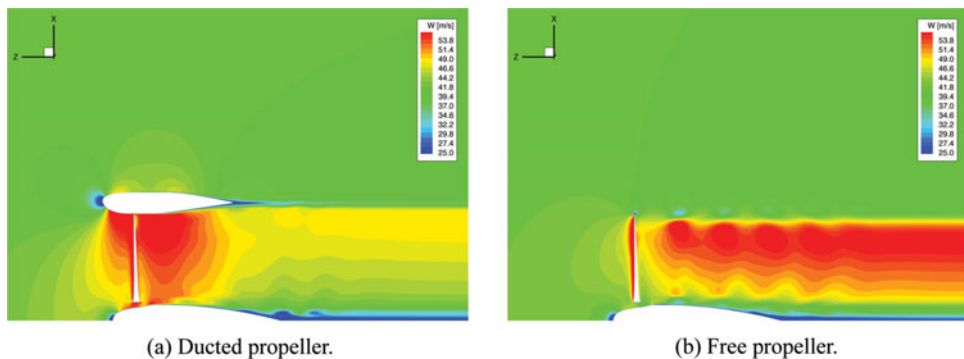
(b) Free propeller.

Figure 13. (Colour online) Axial velocity in the vertical symmetry plane of the ducted and free propellers at dash speed.

Table 2
Force breakdown on the ducted and free propellers

		Thrust			Torque		
J [-]	Configuration	Blades [%]	Spinner [%]	Duct [%]	Blades [%]	Spinner [%]	Duct [%]
0.136	Ducted	54.6	0.3	45.1	100.2	0.0	−0.2
0.136	Free	100.6	− 0.6	–	100.0	0.0	–
0.540	Ducted	77.2	1.1	21.7	100.1	0.1	−0.2
0.540	Free	99.8	0.2	–	99.9	0.1	–

angle, the thrust and the torque of the ducted propeller are considerably lower, since the acceleration of the flow at the duct inlet reduces the effective angle of attack of the blade. Therefore, to compare the performance at this speed, we alter the angle of attack of the ducted propeller blades to match the thrust of the free propeller. By interpolating results for the ducted and free propellers in Table 1, it is deduced that the torque in the ducted case is only 1% lower than the free propeller value. At this higher advance ratio, the duct is ineffective.

To better understand the role of the duct in the thrust generation, Table 2 reports the force breakdown of the ducted and free propellers at cruise and dash speeds. At cruise speed, the duct contribution is 45% of the overall thrust, and the diffuser effect overcompensates the duct drag force and the reduced effective angle of attack of the propeller blades. At dash speed, the contribution is reduced to 22%, and it is barely sufficient to balance the negative effects. For higher speeds, the duct is detrimental to the propulsor performance.

3.4 Effect of the blade twist

The propulsor analysed in the previous section is equipped with highly twisted blades. Their twist distribution was designed for the free propeller, and it is therefore interesting to verify if it is also optimal for the ducted configuration. To better understand the effect of the blade twist on the propulsor performance, a new set of blades was considered, having the same sectional shape but a lower twist value (around one-fourth).

The thrust and torque obtained with CFD for the ducted propeller with low-twist blades are given in Table 1. The comparison with the high-twist blade results reveals that the propulsive efficiency of the low-twist blades is lower both at cruise and dash speeds. These results contradict those obtained with the simple panel method[1], and highlight the necessity of using high-fidelity CFD to predict accurately the performance of a ducted propeller. In light of these results, an optimisation of the blade twist distribution was attempted, using a gradient based method coupled to the HMB3 flow solver.

3.5 Optimisation of the twist distribution

The performance of a ducted propeller depends mainly on the blade geometry and on the expansion ratio of the duct. But there are also other factors that might influence the performance, such as

(a) the shape of the duct diffuser, which needs to be designed to avoid separations due to the adverse pressure gradient;

(b)   the shape of the duct inlet lips, whose design determines the behaviour at off-design working conditions such as, for instance, when the flow is not perfectly axial;

(c)   the clearance between the blade tip and the duct inner surface, which affects the blade-tip losses;

(d)   the blockage induced by non-aerodynamic parts, such as purely structural component, engine and cooling devices.

This work focuses on the blade and duct geometries. Initially, the twist distribution alone was optimised, and at a second stage, both the blade twist distribution and the duct shape were included in the optimisation process. The results of the twist optimisation are given in this section, while the optimisation including the duct shape follows.

The ducted propeller with the high-twist blades with medium pitch, and with advance ratio corresponding to the LTA cruise speed was considered for the twist optimisation exercise. The baseline performance of the propulsor at these conditions are listed in Table 1, from which we get $C_T/C_{T,\text{ref}} = 6.861$ and $C_P/C_{P,\text{ref}} = 2.206$. The optimisation objective was to maximise the thrust coefficient C_T, with the power coefficient C_P constrained to the value of the baseline design.

The shape of the duct was fixed, as well as the shape of the blade sections. The only permitted modification to the blade geometry was a perturbation of the initial twist distribution. This perturbation was described by a Bernstein polynomial expansion of order n:

$$\Delta\theta(\xi) = \sum_{r=0}^{n} \alpha_r K_{r,n} \xi^r (1 - \xi)^{n-r}, \qquad \ldots (19)$$

where $K_{r,n}$ is given by

$$K_{r,n} = \binom{n}{r} = \frac{n!}{r!(n-r)!} \qquad \ldots (20)$$

and ξ is the blade span-wise normalised coordinate, which has a value of 0 at the rotation axis and value of 1 at the blade tip.

The gradients of the thrust and power coefficients with respect to the design variables were computed by means of FGMRES-DR(210,70)-GMRES(50). The number of inner iterations was limited to 50, and the CFL number used for computing the pre-conditioning matrix $\hat{J}$, given by Equation (14), was 1,200. The linear solver was run on 128 Intel Xeon E5-2650 2 GHz cores, and it took about 12 hours and 1,637 outer iterations to converge the solution of the adjoint equations to a relative residual of 10^{-7}, which is almost half of the time needed to solve the base flow (see Fig. 14). The developed GMRES method proved to be particularly effective for the solution of the adjoint equations that represent a particularly challenging problem in this work, since the derivatives of the two-equation k-ω turbulence model are included. Note also that the base flow is not asymptotically converging but instead enters a limit cycle (see Fig. 14), and this is known to lead to an even tougher linear system for the adjoint variables[8].

Figures 15 and 16 show, respectively, the distribution of the adjoint variable of the thrust and power coefficients relative to the pressure equation for the baseline design. Even if a direct interpretation of the adjoint variable is not easy, by Equation (11) we can infer that
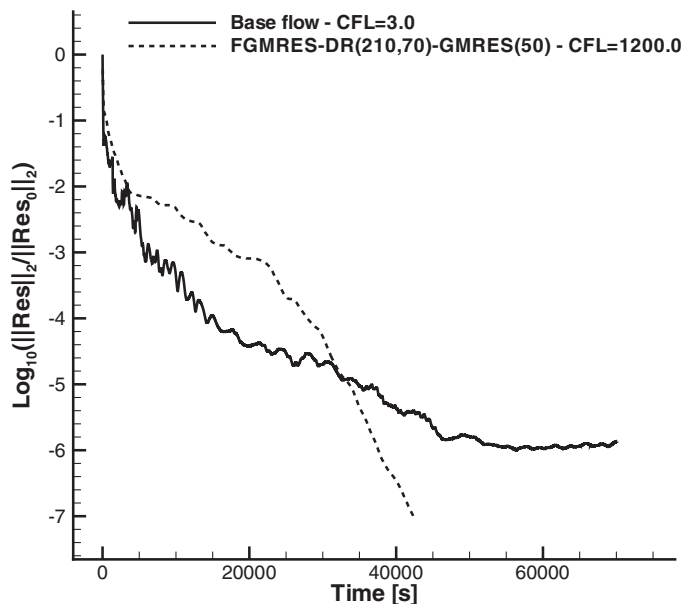
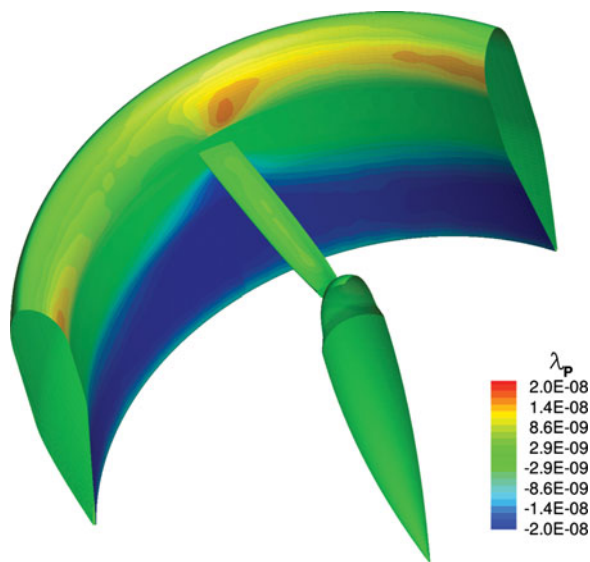Figure 14. Convergence history of the base flow and of the adjoint equations.

Figure 15. (Colour online) Pressure equation adjoint variable of the thrust coefficient C_T.

modifications to surface regions with higher absolute values of the adjoint variable shall affect more the output functional (C_T or C_P). On the other hand, regions with small absolute values of the adjoint variable have negligible influence on the output functional.

Seven coefficients were used for the parametrisation (19), and each coefficient was constrained within the range $(-5°, 5°)$. The optimisation loop converged after 15 cycles
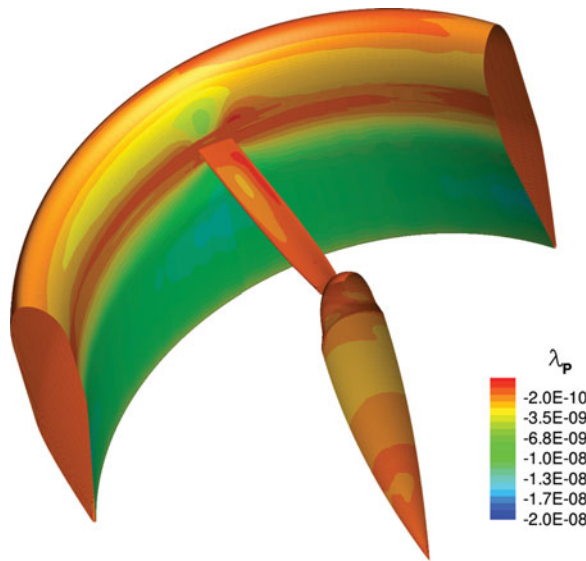
Figure 16. (Colour online) Pressure equation adjoint variable of the power coefficient C_P.
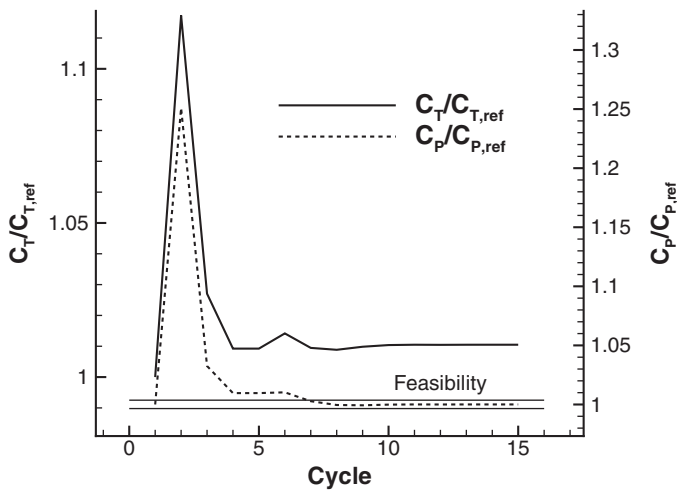
Figure 17. Optimisation history.

(see Fig. 17), yielding an increase in the propulsive efficiency of the propulsor of about 1%. This is a moderate improvement over the baseline design and suggests that the original twist distribution was nearly optimal for the ducted propeller configuration. Figure 18 compares the baseline and optimised blade surfaces and shows how the local pitch has been increased next to the blade tip and at the root and decreased in the sections between 65% and 93% of the blade radius. This leads to a higher loading at the tip of the optimised blade, as shown in Figs 19 and 20.
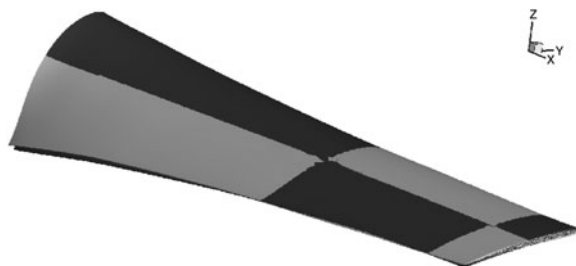
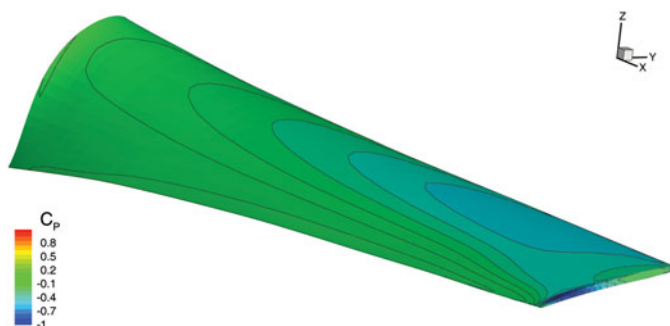Figure 18. Comparison of baseline (light gray) and optimised (dark gray) blade surfaces.

Figure 19. (Colour online) Surface pressure coefficient of the baseline blade
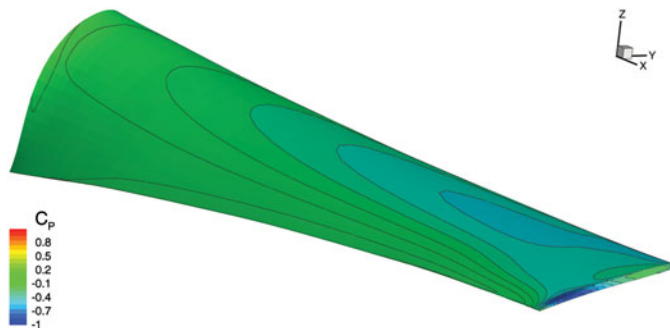(cruise speed, medium blade pitch).

Figure 20. (Colour online) Surface pressure coefficient of the optimised blade
(cruise speed, medium blade pitch).

3.6 Optimisation of the duct shape

A further improvement of the propulsor performance is expected if the duct shape and size are included in the optimisation process. As for the twist optimisation case, the optimisation objective was the maximisation of the thrust coefficient C_T, with the power coefficient C_P constrained to the value of the baseline design.

The blade twist perturbation was described again by the Bernstein polynomial expansion (19). The duct is axisymmetric; therefore, only the section shape needs to be parametrised. The considered design variables allow to vary the shape of the duct diffuser, the radius at the duct outlet and the length of the duct (see Fig. 21). This is done as follows.
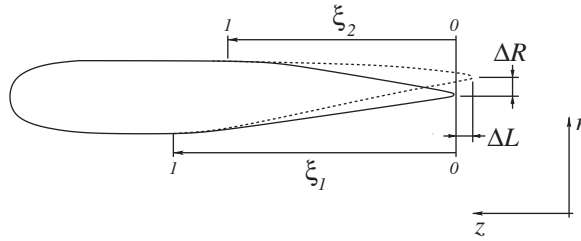
Figure 21. Schematic of the parametrisation of the duct cross-section.

- The shape of the duct diffuser is described using a Class/Shape Transformation (CST) parametrisation[18]:

$$r(\xi_1) = C_{N_1}^{N_2}(\xi_1) \sum_{r=0}^{n} \alpha_r K_{r,n} \xi_1^r (1 - \xi_1)^{n-r}, \qquad \ldots (21)$$

where

$$C_{N_1}^{N_2}(\xi_1) = \xi_1^{N_1} \left(1 - \xi_1^{N_2}\right) \qquad \ldots (22)$$

The symbol r represents the radial coordinate of the duct, while ξ_1 is a nondimensional variable that ranges from 0 at the duct lip to 1 at the inner flat surface of the duct section, as shown in Fig. 21. To represent the rounded lip at $\xi = 0$ and the flat surface at $\xi = 1$, we take $N_1 = 0.5$ and $N_2 = 1.0$.

- The variation of the duct radius ΔR at the outlet is given by the following two functions:

$$\Delta r(\xi_1) = \Delta R[1 - \sin^2(0.5\pi\xi_1)], \qquad \ldots (23)$$

$$\Delta r(\xi_2) = \Delta R[1 - \sin^2(0.5\pi\xi_2)] \qquad \ldots (24)$$

The first function is used to parametrise the duct diffuser surface, whereas the second is used for the duct outer surface.

- The variation of the duct length ΔL is described by the following function:

$$\Delta z(\xi_1) = \Delta L[1 - \sin^2(0.5\pi\xi_1)], \qquad \ldots (25)$$

$$\Delta z(\xi_2) = \Delta L[1 - \sin^2(0.5\pi\xi_2)], \qquad \ldots (26)$$

where z represents the axial coordinates, positive from the aft to the forward side of the propulsor. The first function is used for the diffuser, while the second is for the outer surface.

Seven coefficients were used for the parametrisation of the twist variation (19). The shape of the duct diffuser was described considering 16 coefficients for the CST parametrisation (22). Two design variables were finally used to define the variation of the duct radius and length in (23) and (25), respectively. There was a total of 25 design variables. The twist coefficients were constrained within the range $(-5°, 5°)$. Both the duct radius and length variations were
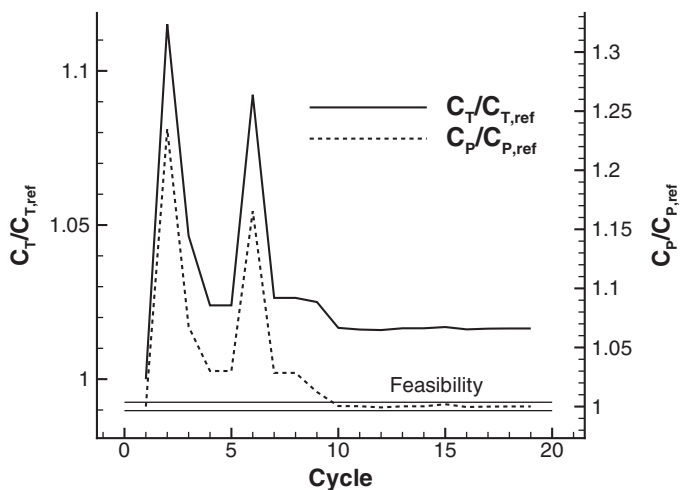
Figure 22. Optimisation history.

Figure 23. Comparison of baseline (light gray) and optimised (dark gray) blade and duct surfaces.

limited to $(-100\,\text{mm}, 100\,\text{mm})$. The optimisation converged after 19 cycles (see Fig. 22), leading to a 2% increase in the propulsive efficiency. Figure 23 shows a comparison of the optimised and baseline surface of the blade and duct. Note that the computed optimal shape of the duct has a larger radius and shorter length. Minor variations have been introduced in the diffuser shape, with the optimised one having a slightly higher curvature near the outflow section.

4.0 CONCLUSIONS

This paper presented the analysis and design of a ducted propeller by means of high-fidelity CFD methods. The numerical results prove the effectiveness of the duct for

low-speed operations, where it significantly increased the overall propulsor efficiency. The effect of the blade twist on the performance was also analysed, and results indicate that blades loaded outboard operate with higher efficiency over a wide range of operating conditions. Optimisation of the twist distribution and of the duct shape was then attempted using a quasi-Newton method and an adjoint solver to provide the required gradients. A moderate increase in the propulsor efficiency was obtained using the blade with optimised twist. Further performance improvement was achieved by combined optimisation of the blade twist and the duct shape, which led to a 2% increase in the propulsor efficiency. The above optimisations were facilitated by the developed nested Krylov-subspace method for solving the adjoint equations.

ACKNOWLEDGEMENTS

REFERENCES

1. Private communication with Hybrid Air Vehicles Ltd., 2015, Bedford, UK.
2. ARNOLDI, W.E. The principle of minimized iterations in the solution of the matrix eigenvalue problem, *Quarterly of Applied Mathematics*, 1951, **9**, (1), pp 17-29.
3. AXELSSON, O. *Iterative Solution Methods*, 1994, Cambridge University Press, Cambridge, Massachusetts, US.
4. BARAKOS, G., STEIJL, R., BADCOCK, K. and BROCKLEHURST, A. Development of CFD capability for full helicopter engineering analysis, 31st European Rotorcraft Forum [CD-ROM], number Paper 91, 2005, Council of European Aerospace Societies.
5. BENZI, M. Preconditioning techniques for large linear systems: a survey. *J. Computational Physics*, 2002, **182**, (2), pp 418-477.
6. BIAVA, M., WOODGATE, M. and BARAKOS, G.N. Fully implicit discrete adjoint methods for rotorcraft applications. *AIAA J.*, 2016, **54**, (2), pp 735-749.
7. BONET, J. and PERAIRE, J. An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems, *Int. J. Numerical Methods in Engineering*, 1991, **31**, (1), pp 1-17.
8. CAMPOBASSO, M.S. and GILES, M.B. Effects of flow instabilities on the linear analysis of turbomachinery aeroelasticity, *J. Propulsion and Power*, 2003, **19**, (2), pp 250-259.
9. CAMPOBASSO, M.S. and GILES, M.B. Stabilization of a linear flow solver for turbomachinery aeroelasticity using recursive projection method, *AIAA J.*, 2004, **42**, (9), pp 1765-1774.
10. CHRISTIANSON, B. Reverse accumulation and implicit functions, *Optimization Methods and Software*, 1998, **9**, (4), 307-322.
11. ENGQUIST, B. and OSHER, S. Stable and entropy satisfying approximations for transonic flow calculations. *Mathematics of Computation*, 1980, **34**, (149), pp 45-75.
12. GILES, M.B. On the iterative solution of adjoint equations, In G. Corliss, C. Faure, A. Griewank, L. Hascoët, and U. Naumann, Eds, *Automatic Differentiation of Algorithms*, 2002, Springer, New York, New York, US, pp 145-151.
13. GIRAUD, L., GRATTON, S., PINEL, X. and VASSEUR, X. Flexible GMRES with deflated restarting, *SIAM J. Scientific Computing*, 2010, **32**, (4), pp 1858-1878.
14. HIRSCH, C. *Numerical Computation of Internal and External Flows: The Fundamentals of Computational Fluid Dynamics*, volume 1, 2007, Butterworth-Heinemann.
15. JAMESON, A. Time dependent calculations using multigrid with application to unsteady flows past airfoils and wings, 10th Computational Fluid Dynamics Conference, AIAA Paper 1991-1596, 1991, Honolulu, HI, US.

16. Johnson, S.G. The NLopt Nonlinear-Optimization Package, http://ab-initio.mit.edu/nlopt. Technical report.
17. Kraft, D. Algorithm 733: TOMP-Fortran modules for optimal control calculations, *ACM Transactions on Mathematical Software*, 1994, **20**, (3), pp 262-281.
18. Kulfan, B.M. Universal parametric geometry representation method, *J. Aircraft*, 2008, **45**, (1), pp 142-158.
19. Liou, M.S. A sequel to AUSM: AUSM+, *J. Computational Physics*, 1996, **129**, (2), pp 364-382.
20. McCormick, B.W. *Aerodynamics of V/STOL Flight*, 1999, Dover Publications, Mineola, New York, USA.
21. Morgan, R.B. GMRES with deflated restarting. *SIAM J. Scientific Computing*, 2003, **24**, (1), pp 20-37.
22. Pereira, J. Hover and Wind-Tunnel Testing of Shrouded Rotors for Improved Micro Air Vehicle Design, PhD thesis, 2008, University of Maryland, College Park, Maryland, US.
23. Renka, R.J. Multivariate Interpolation of Large Sets of Scattered Data. *ACM Transactions on Mathematical Software*, 1988, **14**, (2), pp 139-148.
24. Rieper, F. A low-Mach number fix for Roe's approximate Riemann solver. *J. Computational Physics*, 2011, **230**, (13), pp 5263-5287.
25. Roe, P.L. Approximate Riemann solvers, parameter vectors, and difference schemes, *J. Computational Physics*, 1981, **43**, (2), pp 357-372.
26. Saad, Y. A flexible inner-outer preconditioned GMRES algorithm, *SIAM J. Scientific Computing*, 1993, **14**, (2), pp 461-469.
27. Saad, Y. *Iterative Methods for Sparse Linear Systems*, 1996, PWS Publishing, Boston, Massachusetts, US.
28. Saad, Y. and Schultz, M.H. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *SIAM J. Scientific and Statistical Computing*, 1986, **7**, (3), pp 856-869.
29. Shepard, D. A two-dimensional interpolation function for irregularly-spaced data, Proceedings of the 1968 23rd ACM National Conference, 1968, Association for Computing Machinery, New York, New York, US, pp 517-524.
30. Shroff, G.M. and Keller, H.B. Stabilization of unstable procedures: the recursive projection method, *SIAM J. Numerical Analysis*, 1993, **30**, (4), 1099-1120.
31. Steijl, R., Barakos, G.N. and Badcock, K. A framework for CFD analysis of helicopter rotors in hover and forward flight, *Int. J. Numerical Methods in Fluids*, 2006, **51**, (8), pp 819-847.
32. Van Albada, G.D., Van Leer, B. and Roberts Jr, W.W. A comparative study of computational methods in cosmic gas dynamics, *Astronomy and Astrophysics*, 1982, **108**, pp 76-84.
33. Wilcox, D.C. Re-assessment of the scale-determining equation for advanced turbulence nodels, *AIAA J.*, 1988, **26**, (11), pp 1299-1310.
34. Woodgate, M.A., Pastrikakis, V.A. and Barakos, G.N. Method for calculating rotors with active gurney flaps, *J. Aircraft*, 2016, **53**, (3), pp 605-626.