A Surface Mesh Movement Algorithm for Aerodynamic Optimization of the Nacelle Position on Wing-Body-Nacelle-Pylon Configuration

Gao Yisheng, *Wu Yizhao* *, *Xia Jian*, *Tian Shuling*

College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P. R. China

(Received 19 May 2016; revised 13 July 2016; accepted 5 August 2016)

Abstract: A surface mesh movement algorithm, combining surface mesh mapping with Delaunay graph mapping, is proposed for surface mesh movement involving complex intersections, like wing/pylon intersections. First, surface mesh mapping is adopted for the movement of intersecting lines along the spanwise direction and the wing surface mesh, and then Delaunay graph mapping is utilized for the deformation of the pylon surface mesh, guaranteeing consistent and smooth surface meshes. Furthermore, the corresponding surface sensitivity procedure is implemented for accurate and efficient calculation of the surface sensitivities. The proposed surface mesh movement algorithm and the surface sensitivity procedure are integrated into a discrete adjoint-based optimization framework to optimize the nacelle position on the DLR-F6 wing-body-nacelle-pylon configuration for drag minimization. The results demonstrate that the strong shock on the initial pylon surface is nearly eliminated and the optimal nacelle position can be obtained within less than ten iterations.

Key words: aerodynamic optimization; aerodynamics; aircraft; surface mesh; adjoint

CLC number: V211.3 **Document code**: A **Article ID**: 1005-1120(2016)06-0657-13

0 Introduction

The wing-body-nacelle-pylon (WBNP) configuration has been widely used in current transport aircrafts, thanks to its advantages such as easy access to service and wing bending relief. However, the interactions between wing/fuselage/nacelle/pylon may cause large interference drag, and in turn poor aerodynamic performance. In order to reduce the interference drag, computational fluid dynamics (CFD) has been coupled with numerical optimization techniques to determine the optimal shape and/or position of each component. For example, Koc et al.[1] conducted aerodynamic design optimization for the DLR-F6 WBNP configuration, utilizing a three-dimensional unstructured Euler solver and its discrete adjoint code. Saitoh et al.[2] used a similar approach for multi-point design of WBNP configuration, reducing the drag coefficients at both the low-lift condition and the cruise condition. Li et al.[3] developed an arbitrary space-shape free form deformation(FFD) method for the integral parameterization of nacelle-pylon geometry and optimized the vertical and horizontal locations of the nacelle on the DLR-F6 WBNP configuration. These and others' contributions[4-6] indicate that CFD based numerical optimization methods can improve the design of such a configuration.

Although the optimizations of the vertical and the horizontal (streamwise) locations of a nacelle are not uncommon, the optimization of the spanwise location (as shown in Fig. 1) is still a challenge. In fact, when the nacelle is moved along the spanwise direction, the pylon should be moved accordingly along the spanwise direction,

* Corresponding author, E-mail address: wyzao@nuaa.edu.cn.

How to cite this article: Gao Yisheng, Wu Yizhao, Xia Jian, et al. A surface mesh movement algorithm for aerodynamic optimization of the nacelle position on wing-body-nacelle-pylon configuration[J]. Trans. Nanjing Univ. Aero. Astro., 2016, 33(6):657-669.

http://dx.doi.org/10.16356/j.1005-1120.2016.06.657

leading to the movement of wing/pylon intersecting lines along the wing surface. In this case, a naive application of the commonly used FFD method[7] to the movement of the nacelle/pylon may result in nonsmooth or inconsistent wing-nacelle-pylon surface meshes. Although CAD-based tools may be utilized to address the issue, coupling sophisticated CAD systems with optimization system is not a trivial task. Furthermore, the current proprietary CAD systems seldom provide accurate and efficient surface sensitivity analysis, which is indispensable for gradient-based optimization. On the other hand, some in-house CAD-based tools have been developed for gradient-based shape optimization[8-9], but the capabilities to handle complex intersections are not clear.

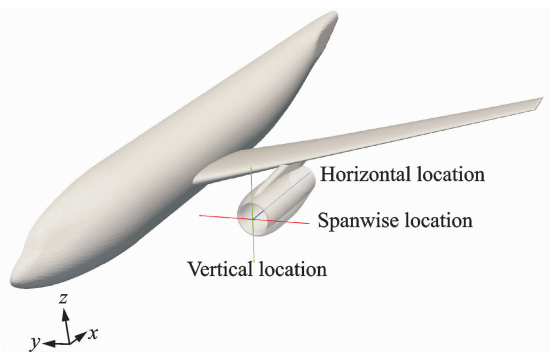


Fig. 1 Representation of the nacelle location

In this paper, an alternative algorithm for the surface mesh movement involving complex intersections without CAD intervention is proposed. The algorithm is based on surface mesh mapping in conjunction with Delaunay graph mapping. First, surface mesh mapping[10] is introduced for the movement of wing/pylon intersections along the spanwise direction and the wing surface mesh. As the nacelle is moved along the spanwise direction, this approach ensures that the wing/pylon intersecting lines are moved accordingly and lie on the wing surface properly, preserving consistency with the underlying wing geometry during the movement of the nacelle. After the determination of the new nacelle location and the new wing/pylon intersection location, Delaunay graph mapping[11] is applied for the surface mesh deformation of the pylon, guaranteeing the smoothness of the deformed surface mesh. Note that the optimization of the pylon geometry itself is not accounted for in this paper. Thus, the pylon is deformed according to the movement of the nacelle/pylon and wing/pylon intersections.

To develop a gradient-based aerodynamic optimization framework for the WBNP configuration, a procedure for the calculation of surface sensitivities is also implemented. The codes of computing the surface sensitivities of the wing points with respect to the spanwise variation are generated by the forward mode of automatic differentiation (AD)[12]. The proposed surface mesh movement approach and the surface sensitivity procedure are integrated into a discrete adjoint-based optimization framework[13], with a linear elasticity-based volume mesh deformation approach, a parallel full implicit flow solver for the Euler equations, a duality-preserving discrete adjoint solver and a gradient-based optimization algorithm. The effectiveness and efficiency of the proposed optimization framework are demonstrated through the optimization of the nacelle position on the DLR-F6 WBNP configuration for drag minimization.

1 Surface Mesh Movement Algorithm

1.1 Surface mesh mapping

Given the vertical and horizontal locations of the nacelle as design variables, only the deformations of the nacelle and pylon surface meshes are required, without the movement of the wing/pylon intersections. However, if the spanwise location of the nacelle is also designated as a design variable, the translation of the nacelle along the spanwise direction requires the consistent translation of the pylon, leading to the problem of the determination of the new wing/pylon intersection location on the wing surface. It is not an easy task due to the requirement of preserving the new wing/pylon intersections lying on the wing surface and the smoothness of the deformed wing-nacelle-pylon surface meshes. Murayama et al.[10]

proposed a simple and robust surface mesh mapping for surface mesh movement in juncture regions, and demonstrated its capability by the change of the position and deflection angle of the horizontal tail wing. Therefore, it is applied here to guide the movement of the wing surface mesh as the change of the wing/pylon intersections in the optimization process.

This method includes the following steps:

(1) Create an appropriate structured background mesh for the clean wing geometry, covering the wing/pylon intersections, as shown in Fig. 2. The clean wing geometry is usually already available or can be generated by using "untrim" operation in CAD systems. The background mesh can be generated for part of the clean wing surface, rather than the complete wing surface. This background mesh will be used for the mapping between physical space and parameter domain, as shown in Fig. 3, where ξ and η represent parametric coordinates in parameter domain.

(2) Determine the mapping (parametric coordinates) of vertices on the wing surface mesh covered by the background mesh. Consider any vertex P on the surface mesh, as shown in Fig. 4.

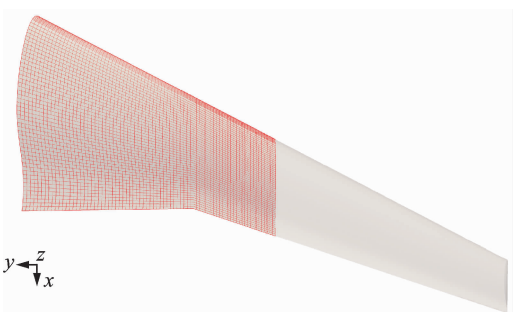


Fig. 2 The clean wing geometry and the background mesh

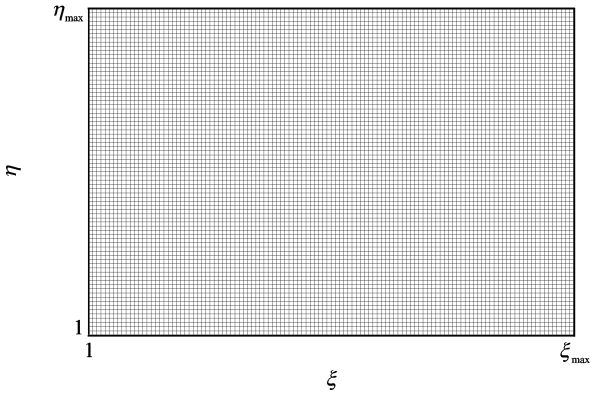


Fig. 3 Parameter domain of the background mesh

The Cartesian coordinates of the vertex P can be given by

$$x_P = \sum_{i=1}^{4} \Phi_i r_i \tag{1}$$

where x_P represents the Cartesian coordinates of the vertex P, r_i the Cartesian coordinates of the vertex of the donor element where the vertex P is located, and Φ_i the interpolation weights calculated by the standard shape function for four-node quadrilateral

$$\begin{cases} \Phi_1 = \dfrac{1-\xi_l}{2}\dfrac{1-\eta_l}{2} \\ \Phi_2 = \dfrac{1+\xi_l}{2}\dfrac{1-\eta_l}{2} \\ \Phi_3 = \dfrac{1+\xi_l}{2}\dfrac{1+\eta_l}{2} \\ \Phi_4 = \dfrac{1-\xi_l}{2}\dfrac{1+\eta_l}{2} \\ \xi_l = 2(\xi_P - i_P) - 1 \\ \eta_l = 2(\eta_P - j_P) - 1 \end{cases} \tag{2}$$

where ξ_P and η_P are the parametric coordinates of the vertex P in parameter domain, ξ_l and η_l the local parametric coordinates of the vertex P, i_P and j_P the donor element indexes of the vertex P. In order to obtain the parametric coordinates ξ_P and

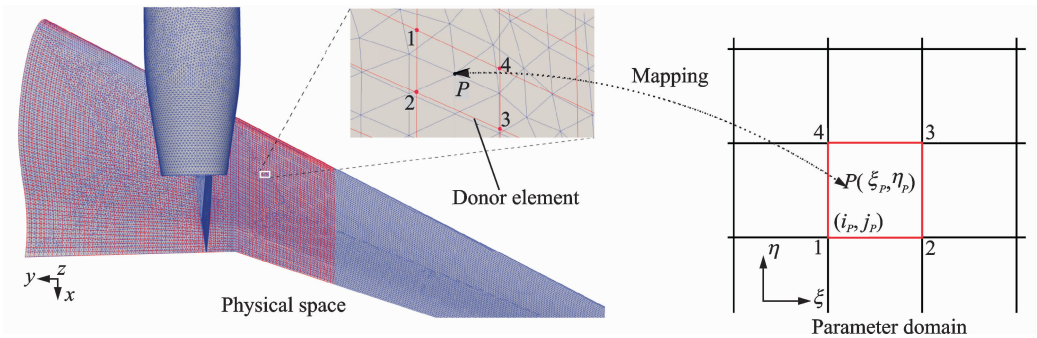


Fig. 4 Mapping of the vertex P

η_P, the donor element on the background mesh is first identified. Rather than the neighbor-to-neighbor jump-search algorithm used in Murayama′s method[10], a simple direct search is applied here. In the direct search, each quadrilateral element on the background mesh is split into two triangular elements, and the projected point is calculated to find the donor element[14]. Although it is not as efficient as the neighbor-to-neighbor jump-search algorithm, the direct search is carried out only once. After the donor element is identified, Newton-Raphson iteration is adopted to obtain the parametric coordinates[15]. The convergence to machine precision can be achieved within 5—6 iterations. Once the parametric coordinates are obtained, the part of the wing surface mesh in physical space, as shown in Fig. 5, is mapped onto a 2D mesh in parameter domain which is shown in Fig. 6.

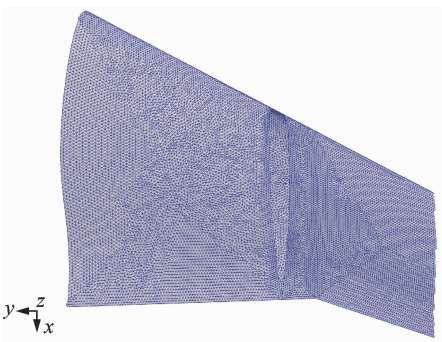


Fig. 5 The part of the wing surface mesh in physical space for mapping

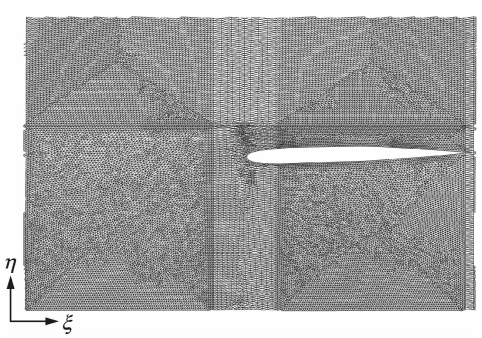


Fig. 6 Mapped mesh in parameter domain

(3) According to the specified movement of the nacelle and the pylon along spanwise direction, move the wing/pylon intersections in parameter domain. Since the movement of the wing/pylon intersections in parameter domain is unknown, a special Newton-Raphson iteration is used to decide the movement, as depicted by pseudocode in Fig. 7.

Algorithm 1 Newton-Raphson iteration for determining translation in the η direction

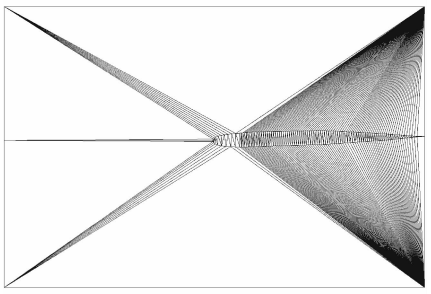
Input:
  y_0—local parametric coorinates
  d_y—translation in the y direction
  i_0, j_0—initial donor element index
  ξ_0, η_0—initial parametric coordinates
  y_{bk}—coordinates of background mesh
Output:
  d_η—translation in the η direction

```
1:procedure Newton-Raphson(y0,dy,i0,j0,ξ0,η0, ybk, dη)
2:    ξ1←2(ξ0−i0)−1.0                          ▷ local parametric coordinates
3:    η1←2(η0−i0)−1.0
4:    i←i0
5:    j←j0
6:    ξ← ξ1
7:    η← η1
8:    for m←1,miter do
9:        y1←ybk,(i,j)                         ▷ y coordinates of donor element vertices
10:       y2←ybk,(i+1,j)
11:       y3←ybk,(i+1,j+1)
12:       y3←ybk,(i,j+1)
13:       y ←1/4(1−ξ)(1−η)y1+1/4(1+ξ)(1−η)y2+1/4(1+ξ)(1+η)y3+
              1/4(1−ξ)(1+η)y4
14:       dd←−1/4 (1−ξ)y1−1/4(1+ξ)y2+1/4(1+ξ)y3+1/4(1−ξ)y4
15:       δη← (y0+dy−y)/dd
16:       η ←η+δη                              ▷ update local parametric coordinate
17:       while |η|>1 do
18:        if η>1 then
19:          j←j+1                             ▷ update donor element
20:          η←η−2                             ▷ adjust local parametric coordinate
21:        else
22:          j←j−1
23:          η←η+2
24:        end if
25:       end while
26:       if |y−y0−dy|<tol. then exit
27:       end if
28:   end for
29:   dη ←j −j0+0.5(η− η0)                     ▷ translation in the η direction
30: End procedure
```

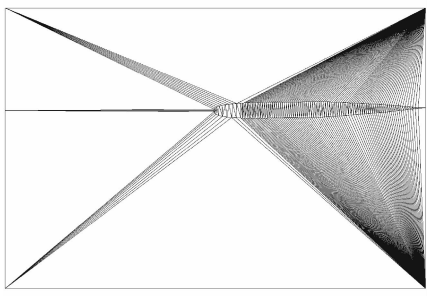
Fig. 7 Newton-Raphson iteration for determining the translation in the η direction

(4) After the movement of wing/pylon intersections in parameter domain, deform the mapped mesh in parameter domain. Since it is essentially a 2D mesh deformation problem in parameter domain, various mesh deformation algorithms can be utilized. Delaunay graph mapping is adopted here due to its simplicity. Delaunay graph is generated by GEOMPACK[16], a FORTRAN library for Delaunay triangulation. The initial and deformed Delaunay graphs and mapped meshes are shown in Figs. 8, 9, respectively. Here the translation of 20% chord length along the direction of

the outboard wing is assumed, and two auxiliary points are added to the left middle side and the right middle side of Delaunay graphs, respectively, in order to prevent possible intersections between the graph elements.

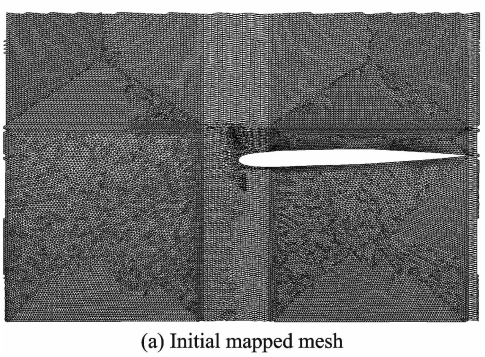


(a) Initial Delaunay graph

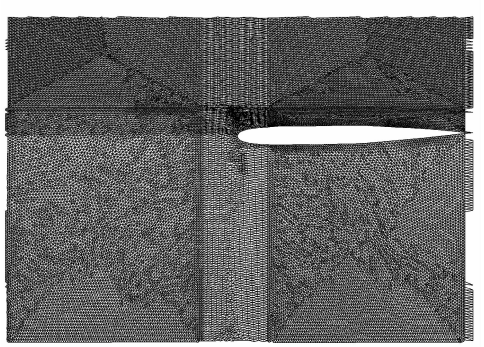


(b) Deformed Delaunay graph

Fig. 8 Initial and deformed Delaunay graphs



(a) Initial mapped mesh



(b) Deformed mapped mesh

Fig. 9 Initial and deformed mapped meshes

(5) According to the deformed mapped mesh in parameter domain, calculate the new coordinates of the vertices on the wing surface using Eq. (2). The resulting wing surface mesh in physical space is shown in Fig. 10. Compared to Fig. 5, it can be seen that the wing/pylon intersections are moved smoothly across the boundary of the inboard wing and the outboard wing.

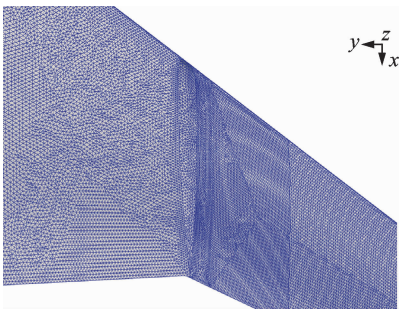


Fig. 10 The resulting wing surface mesh in physical space

1.2 Delaunay graph mapping

Since the vertical, horizontal and spanwise locations are designated as design variables, once the new locations are specified, the nacelle surface mesh is moved rigidly and the wing surface mesh is deformed by the above method. Therefore, the next step is the smooth deformation of the pylon surface mesh. Widely used physical analogy based mesh deformation strategies, such as spring analogy[17] or linear elasticity-based method[18], may be not suitable for this case, because of the difficulty in association with the static equilibrium formulation for fictitious spring network or linear elasticity on curved surfaces. In fact, noncoplanar tension spring network cannot be static equilibrium, so that it is not capable of surface mesh movement on curved surfaces. To overcome the difficulty, Delaunay graph mapping is employed to deform the pylon surface mesh according to the movement of wing/pylon intersections and nacelle/pylon intersections. Delaunay graph mapping has been successfully applied to 2D planar and 3D volumetric mesh deformation[11]. Since the mesh movement depends on the movement of the underlying graph element, as long as the movement of the graph element is smooth, the resulting mesh is moved smoothly. Thus, Delaunay graph mapping is also applicable for this case. In the following, the pylon surface mesh deformation according to the horizontal

movement of the nacelle is presented to illustrate it.

First, a bounding box within which the initial pylon geometry and possible deformed pylon geometry lie is created, as shown in Fig. 11. The eight corner points of the bounding box and the points along the wing/pylon intersections and the nacelle/pylon intersections (red points shown in Fig. 11) are defined as boundary points for Delaunay tetrahedralization, which is yield by an open source tetrahedral mesh generator TetGen[19]. Then for any point P on the pylon surface, the coordinates of the point P can be calculated by

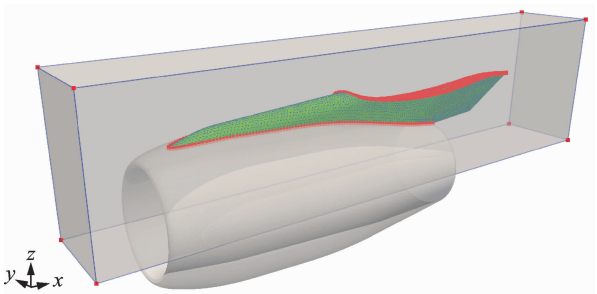


Fig. 11 Bounding box

$$\begin{aligned} x_P &= \sum_i e_i x_i \\ y_P &= \sum_i e_i y_i \\ z_P &= \sum_i e_i z_i \end{aligned} \tag{3}$$

where (x_i, y_i, z_i), $i = 1,2,3,4$ are the nodal points of the graph element where the point P located, shown in Fig. 12, and e_i, $i = 1,2,3,4$ the four relative volume coefficients. This graph element and the corresponding relative volume coefficients are also provided by TetGen, so additional search procedure is avoided. When the nacelle movement, for example 10% chord length along the negative horizontal direction, is specified, the points along the nacelle/pylon intersections are moved accordingly, leading to the movement of the graph element, as shown in Fig. 13. Therefore, the new coordinates of the point P is given by

$$\begin{aligned} x'_P &= \sum_i e_i x'_i \\ y'_P &= \sum_i e_i y'_i \\ z'_P &= \sum_i e_i z'_i \end{aligned} \tag{4}$$

where (x'_i, y'_i, z'_i), $i = 1,2,3,4$ represent the new coordinates of the nodal points.

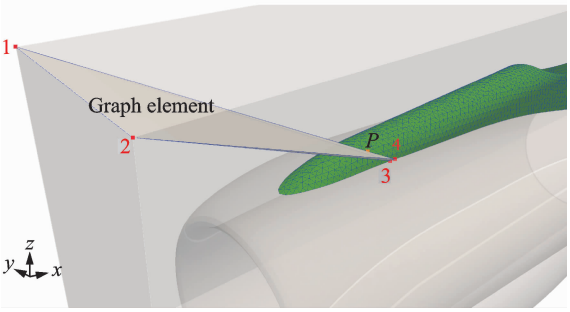


Fig. 12 Graph element

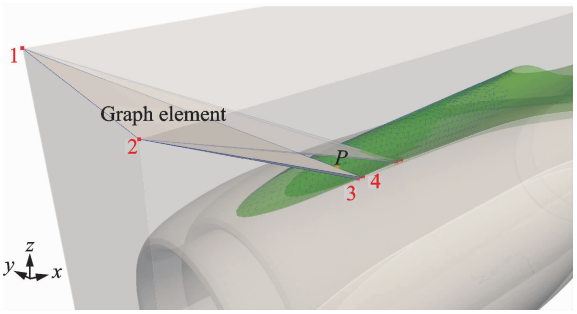


Fig. 13 Movement of graph element

The application of Delaunay graph mapping to the case of the vertical or spanwise movement of the nacelle is exactly the same.

1.3 Complete procedure and applications

In summary, the complete procedure of the proposed surface mesh movement algorithm can be described as:

(1) Move the nacelle rigidly according to the specified vertical, horizontal and spanwise movement;

(2) According to the spanwise movement of the nacelle, determine the movement of the wing/pylon intersections and deform the wing surface mesh using surface mesh mapping;

(3) According to the new locations of the wing/pylon intersections and the nacelle/pylon intersections, apply Delaunay graph mapping to deform the pylon surface mesh.

Figs. 14(a—f) demonstrate the surface mesh movement for different nacelle positions, including ±3% translations along the vertical direction, ±10% translations along the horizontal di-

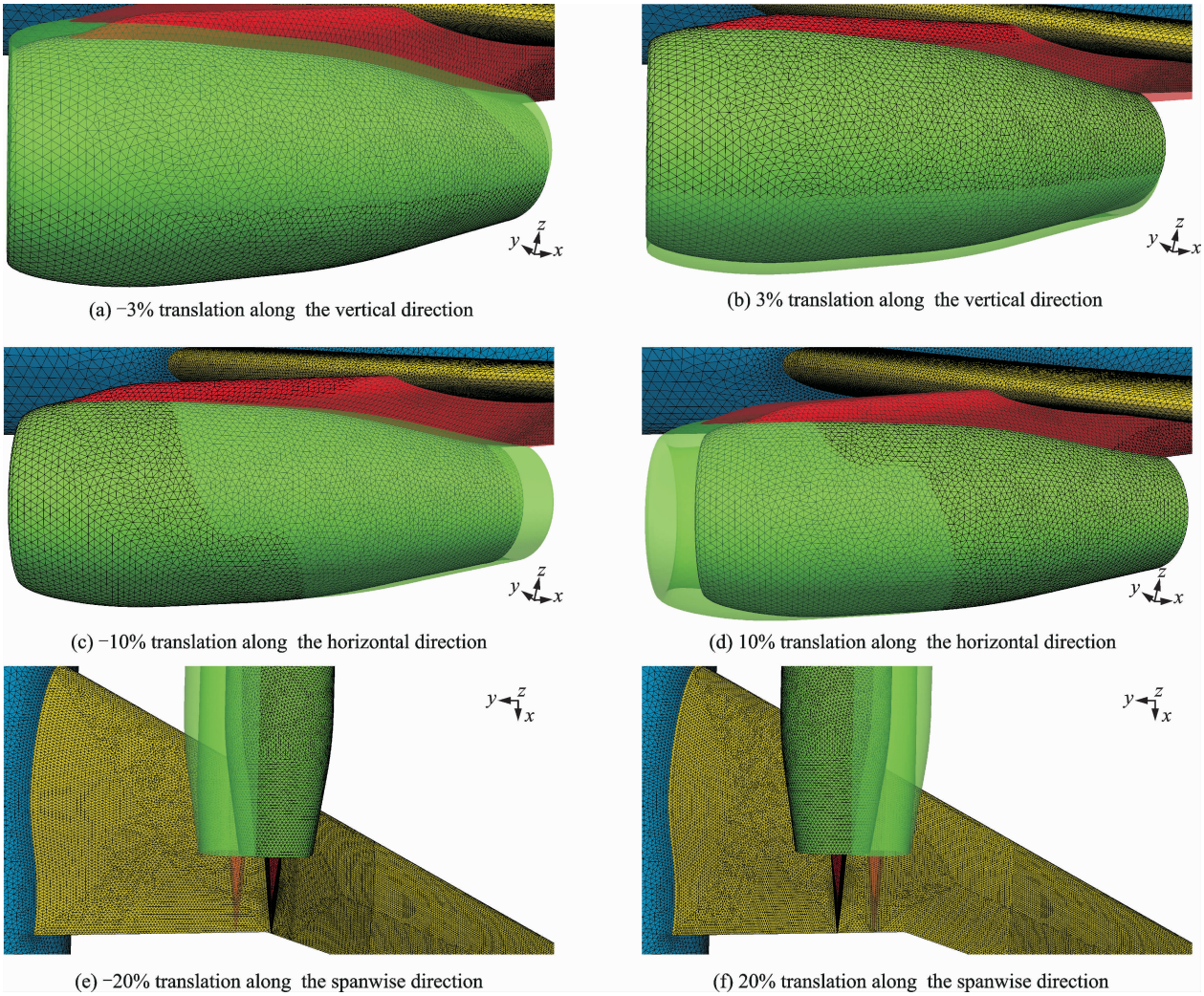


Fig. 14 Surface mesh movement for different nacelle positions (transparent: initial, solid: deformed)

rection and ±20% translations along the spanwise direction (in unit of mean aerodynamic chord). As can be seen, even for large movement, the consistency of the underlying wing geometry and the quality of the deformed surface meshes are maintained.

2 Discrete Adjoint-Based Optimization Framework

Discrete adjoint method is among the most efficient optimization strategies, due to the cost of gradient computation essentially independent of the number of design variables. In order to develop a discrete adjoint-based optimization framework, a procedure for the computation of the surface sensitivity with respect to design variables is required. In this section, the surface sensitivity procedure is implemented and integrated into a discrete adjoint-based optimization framework.

2.1 Discrete adjoint formulation

Provided an objective function L, for instance, drag coefficient C_D and design variables $\boldsymbol{D}$, according to the duality principle[20], adjoint model for the computation of the sensitivity can be formulated as[21]

$$\left[\frac{\mathrm{d}L}{\mathrm{d}\boldsymbol{D}}\right]^{\mathrm{T}} = \left[\frac{\partial \boldsymbol{x}_{\mathrm{surf}}}{\partial \boldsymbol{D}}\right]^{\mathrm{T}} [\boldsymbol{K}]^{-\mathrm{T}} \left(\left[\frac{\partial L}{\partial \boldsymbol{x}_{\mathrm{all}}}\right]^{\mathrm{T}} - \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}_{\mathrm{all}}}\right]^{\mathrm{T}} \left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}\right]^{-\mathrm{T}} \left[\frac{\partial L}{\partial \boldsymbol{w}}\right]^{\mathrm{T}}\right) \quad (5)$$

where $\boldsymbol{x}_{\mathrm{surf}}$ represents the surface mesh point coordinates, $\boldsymbol{x}_{\mathrm{all}}$ the volume mesh point coordinates, $\boldsymbol{w}$ the flow variables, $\boldsymbol{R}$ the residuals of the discretized flow equations, and $\boldsymbol{K}$ the stiffness matrix emerged from volume mesh motion problem.

Except for the $[\partial \boldsymbol{x}_{\mathrm{surf}}/\partial \boldsymbol{D}]^{\mathrm{T}}$ term, the computational components for all other terms in Eq. (5) have been implemented in our previous work[13].

The $[\partial L/\partial \boldsymbol{x}_{\text{all}}]^{\text{T}}$ and $[\partial L/\partial \boldsymbol{w}]^{\text{T}}$ terms are constructed explicitly according to the definition of the objective function. Once the $[\partial L/\partial \boldsymbol{w}]^{\text{T}}$ term is ready, let $\Lambda_w = [\partial \boldsymbol{R}/\partial \boldsymbol{w}]^{-\text{T}}[\partial L/\partial \boldsymbol{w}]^{\text{T}}$, then the flow adjoint equations are given by

$$\left[\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{w}}\right]^{\text{T}} \Lambda_w = \left[\frac{\partial L}{\partial \boldsymbol{w}}\right]^{\text{T}} \tag{6}$$

The $[\partial \boldsymbol{R}/\partial \boldsymbol{w}]^{\text{T}}$ term, the transpose of the flow Jacobian, represents a very complicated matrix structure. Thus, this term is not explicitly constructed, only implicitly computed by matrix-vector multiplication with AD tools. The flow adjoint equations are solved by multi-color Gauss-Seidel method[22]. The $[\partial \boldsymbol{R}/\partial \boldsymbol{x}_{\text{all}}]^{\text{T}}$ term constitutes a fairly complicated sparse matrix, also implicitly calculated by matrix-vector multiplication with AD tools. The term $[\boldsymbol{K}]^{-\text{T}}$ is associated with the mesh adjoint problem, which can be solved by the identical procedure for the volume mesh motion problem.

2.2 Surface sensitivity procedure

The remaining $[\partial \boldsymbol{x}_{\text{surf}}/\partial \boldsymbol{D}]^{\text{T}}$ term, referred to as surface sensitivities, denotes the sensitivities of the surface points with respect to the design variables, related to the above surface mesh movement algorithm. In the following, the surface sensitivities of different components are considered respectively.

(1) For any point on the nacelle surface, since the nacelle is moved rigidly, the surface sensitivities can be easily calculated as

$$\frac{\partial \boldsymbol{x}_{\text{nacelle}}}{\partial \boldsymbol{D}} = \boldsymbol{I} \tag{7}$$

where $\boldsymbol{x}_{\text{nacelle}}$ represents the coordinates of the point and $\boldsymbol{I}$ represents a 3×3 identity matrix.

(2) For the wing surface, according to Section 1.1, three routines should be differentiated: the movement of intersecting lines in parameter domain, 2D mesh deformation in parameter domain and the calculation of the new coordinates in physical space. Although handed derivation may be used, the application of AD tools is preferred, because AD tools can generate routines to compute accurate sensitivities in an efficient and automatic manner without human intervention, reducing the implementation time. In this work, AD tool Tapenade[23] is applied to generate the differentiated routines as it is free available and supports Fortran 90 language. The result of the application of AD tool Tapenade to Newton-Raphson iteration used in the movement of the wing/pylon intersecting lines is shown in Fig. 15. In this case, d_y is set as the independent input variable and d_η as the dependent output variable. If d_y_d is set to 1, d_η_d will give the required sensitivities with respect to d_y. Other differentiated routines can be obtained in an identical manner.

Algorithm 2 Differentiation of Newton-Raphson iteration

Input:
y_0—local parametric coorinates
d_y—translation in the y direction
i_0, j_0—initial donor element index
ξ_0, η_0—initial paremetric coordinates
y_{bk}—coordinates of background mesh
Output:
d_η —translation in the η direction
d_η —d-sensitivity with respect to d_y

1:procedure Newton-Raphson(y_0,dy,i_0, j_0, ξ_0,η_0, y_{bk}, d_η)
2: $\xi_1 \leftarrow 2(\xi_0 - i_0) - 1.0$ ▷local parametric coordinates
3: $\eta_1 \leftarrow 2(\eta_0 - i_0) - 1.0$
4: $i \leftarrow i_0$
5: $j \leftarrow j_0$
6: $\xi \leftarrow \xi_1$
7: $\eta \leftarrow \eta_1$
8: η-d$\leftarrow 0$
9: for $m \leftarrow 1$,miter do
10: $y_1 \leftarrow y_{bk}(i, j)$ ▷y coordinates of donor element vertices
11: $y_2 \leftarrow y_{bk}(i+1, 1)$
12: $y_3 \leftarrow y_{bk}(i+1, j+1)$
13: $y_4 \leftarrow y_{bk}(i, j+1)$
14: $y\_d \leftarrow -\frac{1}{4}(1-\xi)y_1 - \frac{1}{4}(1+\xi)y_2 + \frac{1}{4}(1+\xi)y_3 + \frac{1}{4}(1-\xi)y_4$
15: $y \leftarrow \frac{1}{4}(1-\xi)(1-\eta)y_1 + \frac{1}{4}(1+\xi)(1-\eta)y_2 + \frac{1}{4}(1+\xi)(1+\eta)y_3 + \frac{1}{4}(1-\xi)(1+\eta)y_4$
16: $dd \leftarrow -\frac{1}{4}(1-\xi)y_1 - \frac{1}{4}(1+\xi)y_2 + \frac{1}{4}(1+\xi)y_3 + \frac{1}{4}(1-\xi)y_4$
17: $\delta\eta\_d \leftarrow \frac{dy\_d - y\_d}{dd}$
18: $\delta\eta \leftarrow \frac{y_0 + dy - y}{dd}$
19: $\eta\_d \leftarrow \eta\_d + \delta\eta\_d$
20: $\eta \leftarrow \eta + \delta\eta$ ▷update local parametric coordinate
21: while $|\eta| > 1$ do
22: if $\eta > 1$ then
23: $j \leftarrow j+1$ ▷update donor element
24: $\eta \leftarrow \eta - 2$ ▷adjust local parametric coordinate
25: else
26: $j \leftarrow j-1$
27: $\eta \leftarrow \eta + 2$
28: end if
29: end while
30: if $|y - y_0 - d_y| <$ tol. then exit
31: end if
32: end for
33: $d_\eta\_d \leftarrow 0.5\,\eta\_d$
34: $d_\eta \leftarrow j - j_0 + 0.5(\eta - \eta_0)$ ▷translation in the η direction
35: End procedure

Fig. 15 Differentiation procedure of Newton-Raphson iteration

(3) For any point P on the pylon surface, the new coordinates are calculated by Eq. (4). Since the relative volume coefficients are only dependent of the initial coordinates of the nodal points and independent of the movement of the nodal points, these coefficients are constants for the computation of surface sensitivities. Therefore, the surface sensitivities can be evaluated as

$$\begin{aligned}\frac{\partial x'_P}{\partial \boldsymbol{D}} &= \sum_i e_i \frac{\partial x'_i}{\partial \boldsymbol{D}}\\ \frac{\partial y'_P}{\partial \boldsymbol{D}} &= \sum_i e_i \frac{\partial y'_i}{\partial \boldsymbol{D}}\\ \frac{\partial z'_P}{\partial \boldsymbol{D}} &= \sum_i e_i \frac{\partial z'_i}{\partial \boldsymbol{D}}\end{aligned} \tag{8}$$

where $\partial x'_i/\partial \boldsymbol{D}$, $\partial y'_i/\partial \boldsymbol{D}$, $\partial z'_i/\partial \boldsymbol{D}$ are zero or the sensitivities already obtained in the previous two steps.

2.3 Adjoint sensitivity verification

The complete sensitivity (the $\mathrm{d}L/\mathrm{d}\boldsymbol{D}$ term in Eq. (5)) is verified with three methods: Central finite difference, the complex step method[24] and tangent model[21].

Finite difference is the most common method for estimating the derivative (sensitivity). The second-order central finite difference for approximating the first-order derivative of an objective function $f(x)$ is given by

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \frac{f(x+h) - f(x-h)}{2h} + O(x^2) \tag{9}$$

where h is the step size. Nevertheless, this method often suffers from the "step-size dilemma". The step size should be small enough to minimize truncation error while not so small that round-off error dominates the result. In practice, a suitable step size is usually determined by trial and error.

The complex step method is an accurate and robust derivative approximation method and has been applied to exact linearizations of complicated real-valued residual operators[25]. For a real-valued function $f(x)$, if the input becomes a complex value $x + ih$, where h is a small step size, the Taylor series of the function $f(x+ih)$ can be written as

$$f(x+ih) = f(x) + ihf'(x) + O(h^2) \tag{10}$$

Accordingly, the derivative $f'(x)$ can be approximated by

$$f'(x) = \frac{\mathrm{Im}(f(x+ih))}{h} \tag{11}$$

This approximation is also second-order accurate and no differencing involved, avoiding the "step-size dilemma". So this method can provide a very accurate derivative approximation, if a small enough step size is given.

The tangent model is exact linearization of a vector function with respect to one variable. It is commonly used for sensitivity analysis and verification of adjoint model. If both tangent model and adjoint model are implemented exactly, the duality principle assures that the sensitivities calculated by two models are consistent within machine precision. Even a tiny inexact linearization in adjoint model will immediately manifest lack of consistency. Accordingly, the tangent model is very useful for the verification of adjoint implementation.

The sensitivities of lift coefficient C_{L} and drag coefficient C_{D} with respect to the horizontal movement D_x, the spanwise movement D_y and the vertical movement D_z, calculated by different methods, are shown in Tables 1,2, respectively. The finite difference step size for the horizontal movement is 10^{-6}. The finite difference step size for the spanwise movement is 1.4×10^{-9}. The finite difference step size for the vertical movement is 10^{-7}. The complex step size is 10^{-20}. All equations involved are converged to machine precision to avoid any algebraic error. The adjoint sensitivities yield an agreement of 9—11 significant figures when compared to the results of the complex step method and the tangent model, and an agreement of 4—8 significant figures when compared to the results of the finite difference method. It is evident that the adjoint sensitivities obtained by the current approach are highly accurate for the gradient based optimization process.

Table 1 Adjoint sensitivity verification for lift coefficient with respect to design variables

Method	$\frac{\partial C_L}{\partial \boldsymbol{D}_x}$	$\frac{\partial C_L}{\partial \boldsymbol{D}_y}$	$\frac{\partial C_L}{\partial \boldsymbol{D}_z}$
Finite difference	$-9.451\,502\,889\,790\,29\times10^{-1}$	$-8.139\,215\,492\,062\,95\times10^{-2}$	$7.558\,051\,451\,256\,52\times10^{-1}$
Complex-step	$-9.451\,502\,894\,684\,83\times10^{-1}$	$-8.139\,456\,108\,942\,67\times10^{-2}$	$7.758\,051\,024\,375\,74\times10^{-1}$
Tangent	$-9.451\,502\,894\,697\,32\times10^{-1}$	$-8.139\,456\,108\,909\,56\times10^{-2}$	$7.758\,051\,024\,325\,96\times10^{-1}$
Adjoint	$-9.451\,502\,894\,683\,65\times10^{-1}$	$-8.139\,456\,108\,907\,21\times10^{-2}$	$7.758\,051\,024\,323\,85\times10^{-1}$

Table 2 Adjoint sensitivity verification for drag coefficient with respect to design variables

Method	$\frac{\partial C_L}{\partial \boldsymbol{D}_x}$	$\frac{\partial C_L}{\partial \boldsymbol{D}_y}$	$\frac{\partial C_L}{\partial \boldsymbol{D}_z}$
Finite difference	$-5.620\,390\,899\,213\,62\times10^{-3}$	$8.281\,678\,914\,074\,62\times10^{-3}$	$4.726\,797\,349\,363\,65\times10^{-2}$
Complex-step	$-5.620\,390\,514\,898\,58\times10^{-3}$	$8.281\,621\,087\,656\,65\times10^{-3}$	$4.726\,797\,172\,369\,31\times10^{-2}$
Tangent	$-5.620\,390\,515\,007\,32\times10^{-3}$	$8.281\,621\,087\,670\,01\times10^{-3}$	$4.726\,797\,172\,340\,63\times10^{-2}$
Adjoint	$-5.620\,390\,514\,972\,17\times10^{-3}$	$8.281\,621\,087\,725\,22\times10^{-3}$	$4.726\,797\,172\,341\,57\times10^{-2}$

3 Optimization Result

The proposed discrete adjoint-based optimization framework is used for the optimization of the nacelle position on the DLR-F6 WBNP configuration for drag minimization. This optimization problem can be formulated as

$$\begin{aligned} &\min \quad C_D \\ &\text{subject to} \quad C_L \geqslant C_{L\,\text{initial}} \\ &\text{w. r. t.} \quad \boldsymbol{D} \end{aligned} \tag{12}$$

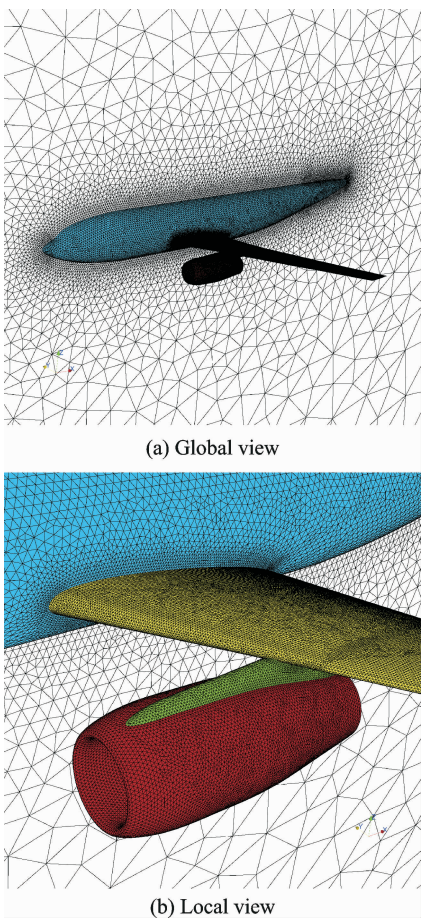
where $C_{L\,\text{initial}}$ represents the lift coefficient of the initial configuration. The mean aerodynamic chord is 0.141 2 m and half model reference area is 0.072 7 m^2. The computational mesh consists of 570 866 points and 3 161 828 tetrahedral cells, as depicted in Fig. 16. The Mach number is 0.75 and the incidence is 1°. The lower and upper limits of the design variables are shown in Table 3.

Table 3 Lower and upper limits of the design variables

%

Design variable	Lower limit	Initial value	Upper limit
Horizontal	−10.0	0	+1.0
Spanwise	−10.0	0	+15.0
Vertical	−10.0	0	+1.0

Optimization package NPSOL[26], which is based on sequential quadratic programming (SQP) method[27], is used to drive the optimization process. Table 4 shows the optimization results. The drag coefficient is reduced from C_D = 0.028 55 to C_D = 0.028 34, while the lift coefficient remains constant. Fig. 17 illustrates pressure coefficient contours on the initial and optimized wing-nacelle-pylon surface. The strong shock on the initial pylon surface has been almost removed, which contributes to the reduced drag. Fig. 18 demonstrates the initial and optimized na-



(a) Global view

(b) Local view

Fig. 16 Computational mesh of the DLR-F6 WBNP configuration

celle positions. The nacelle is moved further forward, downward and inward compared to the initial position, and the resulting surface mesh remains smooth and consistent under a relatively large deformation. Fig. 19 shows the convergence history for the optimization problem, in terms of the merit function in NPSOL. The optimization problem converges within a total of 9 design cycles. The complete optimization takes approximately 4.5 h of wall-clock time, using 48 cores on the TH-1A supercomputer at National Supercomputer Center in Tianjin. This suggests that the optimization framework is highly efficient for this optimization problem and can be used as a design tool for the WBNP configuration.

Table 4 Optimization results

Coefficient	Initial value	Optimized value	Variation/%
C_D	0.028 55	0.028 34	−0.74
C_L	0.758 7	0.758 7	0.0

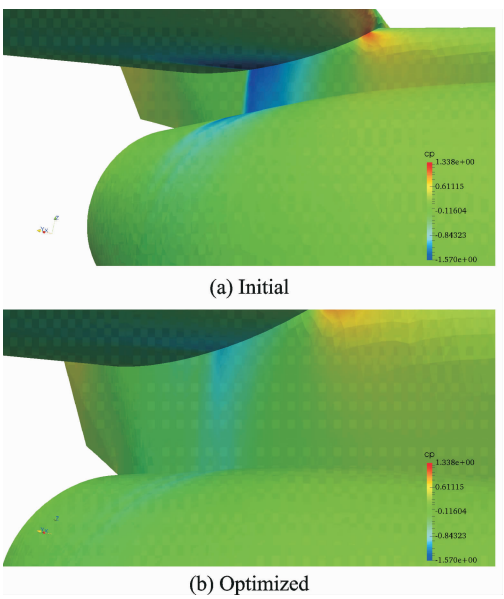


Fig. 17 Pressure coefficient contours on the wing-nacelle-pylon surface

4 Conclusions

A robust and efficient surface mesh movement algorithm for surface mesh involving complex intersections has been presented. The corresponding surface sensitivity procedure is implemented and integrated into a discrete adjoint-based optimization framework. This optimization framework has been successfully applied to the optimization of the nacelle position on the DLR-F6 WBNP configuration. The optimization almost eliminates strong shock on the initial pylon surface, resulting in a reduction of 2 counts of the drag coefficient. The optimal nacelle position is achieved within less than ten iterations, which indicates high efficiency of the proposed optimization framework. Future work will focus on aerodynamic optimization based on the Reynolds averaged Navier-Stokes equations. The optimization concerning wing geometry and nacelle position will be simultaneously considered.

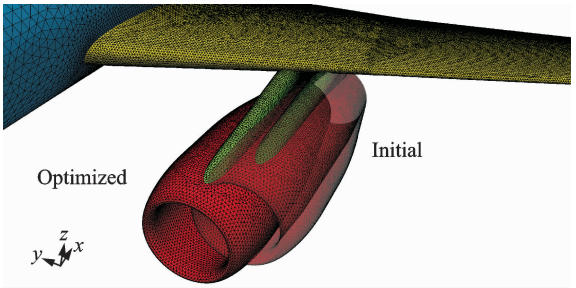


Fig. 18 Initial and optimized nacelle positions

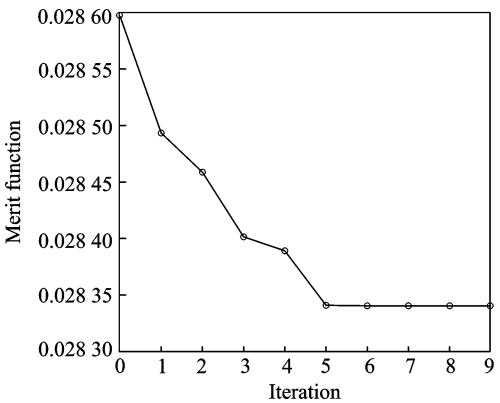


Fig. 19 Convergence history for the optimization problem

Acknowledgement

This study was supported by the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

References:

[1] KOC S, KIM H, NAKAHASHI K. Aerodynamic design of wing-body-nacelle-pylon configuration[C]// 17th AIAA Computational Fluid Dynamics Conference. Reston: AIAA, 2005.

[2] SAITOH T, KIM H, TAKENAKA K, et al. Multi-point design of wing-body-nacelle-pylon configuration [C]// 24th Applied Aerodynamics Conference. Reston: AIAA, 2006.

[3] LI J, GAO Z, HUANG J, et al. Aerodynamic design optimization of nacelle/pylon position on an aircraft[J]. Chinese Journal of Aeronautics, 2013, 26(4): 850-857.

[4] NEMEC M, AFTOSMIS M. Parallel adjoint framework for aerodynamic shape optimization of component-based geometry[C]//49th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition. Reston: AIAA, 2011.

[5] ZHANG Y, CHEN H, ZHANGE W, et al. Wing/engine integrated optimization based on Navier-Stokes equations [C] // 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition. Reston: AIAA, 2012.

[6] HATANAKA K, OBAYASHI S, JEONG S. Application of the variable-fidelity MDO tools to a jet aircraft design[C]// 25th International Congress of the Aeronautical Sciences. Hamburg, Germany: ICAS, 2006.

[7] SEDERBERG T, PARRY S. Free-form deformation of solid geometric models [J]. ACM SIGGRAPH Computer Graphics, 1986, 20(4): 151-160.

[8] TRUONG A, ZINGG D, HAIMES R. Surface mesh movement algorithm for computer-aided-design-based aerodynamic shape optimization[J]. AIAA Journal, 2016, 54(2): 542-556.

[9] HATANAKA K, BREZILLON J, RONZHEIMER A. A fully automated CAD-based framework for shape optimization[C]// 28th International Congress of the Aeronautical Sciences. Valencia, Spain: ICAS, 2012.

[10] MURAYAMA M, NAKAHASHI K, MATSUSHIMA K. A robust method for unstructured volume/surface mesh movement[J]. Transactions of the Japan Society for Aeronautical and Space Sciences, 2003, 46(152): 104-112.

[11] LIU X, QIN N, XIA H. Fast dynamic grid deformation based on Delaunay graph mapping[J]. Journal of Computational Physics, 2006, 211(2): 405-423.

[12] GRIEWANK A, WALTHER A. Evaluating derivatives: Principles and techniques of algorithmic differentiation[M]. Philadelphia: Society for Industrial and Applied Mathematics, 2008.

[13] GAO Yisheng, WU Yizhao, XIA Jian. A discrete adjoint-based approach for airfoil optimization on unstructured meshes[J]. ACTA Aerodynamica Sinica, 2013, 31(2): 244-249. (in Chinese)

[14] SAMAREH-ABOLHASSANI J. Unstructured grids on NURBS surfaces[C]// 11th Applied Aerodynamics Conference. Reston: AIAA, 1993.

[15] TIAN Shuling. Investigation of overset unstructured grids algorithm[D]. Nanjing: Nanjing University of Aeronautics & Astronautics, 2008. (in Chinese)

[16] JOE B. GEOMPACK—A software package for the generation of meshes using geometric algorithms[J]. Advances in Engineering Software, 1991, 13: 325-331.

[17] BATINA J. Unsteady Euler airfoil solutions using unstructured dynamic meshes [J]. AIAA Journal, 1990, 28(2): 1381-1388.

[18] TEZDUYAR T. Stabilized finite element formulations for incompressible flow computations[J]. Advances in Applied Mechanics, 1992, 28(1): 1-44.

[19] SI H. TetGen, a Delaunay-based quality tetrahedral mesh generator[J]. ACM Transactions on Mathematical Software, 2015, 41(2):1-36.

[20] GILES M, PIERCE N. An introduction to the adjoint approach to design[J]. Flow, Turbulence and Combustion, 2000, 65:393-415.

[21] MAVRIPLIS D. Discrete adjoint-based approach for optimization problems on three-dimensional unstructured meshes[J]. AIAA Journal, 2007, 45(4): 740-750.

[22] SATO Y, HINO T, OHASHI K. Parallelization of an unstructured Navier-Stokes solver using a multi-color ordering method for Open MP[J]. Computer & Fluids, 2013, 88:496-509.

[23] HASCOET L, PASCUAL V. The Tapenade automatic differentiation tool: Principles, model, and specification[J]. ACM Transactions on Mathematical Software, 2013, 39(3):1-43.

[24] MARTINS J, STURDZA P, ALONSO J. The complex-step derivative approximation[J]. ACM Transactions on Mathematical Software, 2003, 29(3):245-262.

[25] NIELSEN E, KLEB W. Efficient construction of discrete adjoint operators on unstructured grids using complex variables[J]. AIAA Journal, 2006, 44(4):

827-836.

[26] GILL P, MURRAY W, SAUNDERS M, et al. User's guide for NPSOL 5.0: A FORTRAN package for nonlinear programming: Technical Report SOL 86-1 [R]. California: SOL, 1986.

[27] NOCEDAL J, WRIGHT S. Numerical optimization [M]. Berlin: Springer-Verlag, 1999.

Mr. **Gao Yisheng** is currently a Ph. D. candidate at College of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China. His main research interests include computational fluid dynamics and aerodynamic shape optimization.

Prof. **Wu Yizhao** is currently a professor at Nanjing University of Aeronautics and Astronautics, Nanjing, China. His main research interests include computational fluid dynamics, aerodynamic shape optimization and hypersonic flows.

Prof. **Xia Jian** is currently a professor at Nanjing University of Aeronautics and Astronautics, Nanjing, China. His main research interests include computational fluid dynamics, fluid structure interaction, aerodynamic shape optimization and DSMC.

Dr. **Tian Shuling** is currently an associate professor at Nanjing University of Aeronautics and Astronautics, Nanjing, China. His main research interests include computational fluid dynamics, unsteady aerodynamics and aerodynamic shape optimization.

(Executive Editor: Zhang Bei)