

## 翼型几何外形的反优化

### 一. 优化问题解释

该问题需要对基准翼型的几何外形进行优化,以获得最接近于目标翼型外形的翼型。在该模型中,选用 NACA0012 作为基准翼型,步骤如下:

1.对翼型进行参数化,利用几个关键的参数来描述翼型形状,该参数即为设计变量,通过改变翼型参数值来实现对目标函数的优化。

2.将目标翼型的 x 坐标代入基准翼型上下翼面,分别求得 y 坐标后与目标翼型相对应的点进行作差取绝对值,即为本问题需要优化的目标函数。

3.使用准牛顿法、遗传算法等优化算法,令构造的纵坐标差值绝对值取最小,即可获得两个非常接近的翼型,进而得以求解。

### 二. 数学模型建立

#### 2.1 CST 参数化翼型

CST 方法通过类别函数和形状函数相结合的方式描述翼型几何形状,其中,类别函数描述了翼型的通用形状特征,形状函数则通过一组 Bernstein 多项式系数来对翼型的具体形状进行调整,通过将这两个函数相乘,再加上后缘厚度修正项,就可以得到翼型上下表面的坐标。

在进行参数化时,选取 NACA0012 作为初始翼型,作为常见的圆头尖尾翼型,将其类别函数指数  $N_1$ 、 $N_2$  和后缘厚度修正  $\delta_{te}$  设置为默认值,该数值分别为 0.5、1、0。在空气动力学中,用于描述翼型通用形状特征的表达式为

$$C_i = x_i^{N_1} \times (1 - x_i)^{N_2}$$

在上述类别函数中,假设  $X_i$  为一个向量  $X = [x_1, x_2, \dots, x_n]$ ,  $N_1$  和  $N_2$  是标量值,  $C_i$  为计算结果向量中的第  $i$  个元素,通过不同  $N_1$  和  $N_2$  的取值可以得到不同的基础形状。

在计算形状函数时,  $S(x)$  由一组 Bernstein 多项式组成。Bernstein 多项式是一种常用的多项式基函数,其表达式为  $B = nchoosek(n, i) \times x^i \times (1 - x)^{n-i}$ , 其中  $nchoosek(n, i)$  是组合数,表示从  $n$  个元素中选择  $i$  个元素的组合方式。

为了确保翼型的后缘闭合,将上表面和下表面的最后一个坐标值取平均值,作为修正后的后缘坐标。该算法通过 CST 方法,利用类别函数和形状函数的组合,结合后缘厚度修正,生成了翼型的上下表面坐标。通过调整输入的 Bernstein

多项式系数，可以灵活地改变翼型的形状，适用于翼型设计和优化等领域。

### 2.2 黄金分割法

在进行目标翼型的反参数化过程中，需要将基准翼型 NACA0012 和目标翼型的纵坐标进行作差取绝对值，将该值通过翼型参数化的过程转换为变量，接着通过优化算法寻优进行求解。在构造求解变量时，将给定目标翼型上下翼面的  $x$  坐标代入基准翼型中求得  $y$  坐标，再分别对应进行作差取绝对值，由此可得该模型的设计变量。

黄金分割法的基本思想为在函数极值存在的区间内取若干点，也称试验点，计算这些点的函数值并进行比较，然后可以根据函数值确定存在函数极值的更小的区间；在这个小区间内重新取若干点再次比较函数值，从而确定更小的存在函数极值的区间；重复上述过程直至存在函数极值的区间足够小时就可以用区间内的点某点来近似函数的极值。在分割区间时，采用黄金分割比例。该方法寻优流程如下：

1. 设定目标函数的绝对误差精度  $\varepsilon_1$  和自变量的绝对误差精度  $\varepsilon_2$ ，以及最大的迭代步数限制数  $M$ 。

2. 在给定的变量区间  $[a, b]$  内，取两个试验点  $x_1$  和  $x_2$ ，利用黄金分割比例来表达分别为  $x_1 = b - q(b - a)$  和  $x_2 = a + q(b - a)$ ，并计算两点处的函数值分别为  $\varphi_1 = f(x_1)$  和  $\varphi_2 = f(x_2)$ 。

3. 如果  $\varphi_1 < \varphi_2$ ，则令  $b = x_2$ ， $x_2 = x_1$ ，同时更新  $x_1$  为  $x_1 = b - q(b - a)$ ；如果  $\varphi_1 \geq \varphi_2$ ，则令  $a = x_1$ ， $x_1 = x_2$ ， $x_2 = a + q(b - a)$ 。

4. 判断以下收敛条件之一是否满足： $|\varphi_2 - \varphi_1| \leq \varepsilon_1$ ， $|x_2 - x_1| \leq \varepsilon_2$ ，和  $k > M$ 。如果满足，则终止迭代过程，并在区间  $[a, b]$  内选择函数值较小的试验点作为极值点；否则，返回步骤 2) 进行迭代。

### 2.3 牛顿法

牛顿法基于目标函数在搜索点处利用目标函数的二阶 Taylor 展开式近似目标函数，求解展开式的极值点作为新的搜索点，重复迭代过程直至满足收敛标准。进行 Taylor 二次展开的前提是，目标函数在展开点处二次可微，并且存在正定对称的 Hessian 矩阵。

目标函数 $f(x)$ 在展开点 $x^k$ 的展开式为:

$$f(x_k + d) = f(x_k) + \nabla^T f(x_k) d + \frac{1}{2} d^T H_k d + o(|d|^2) \quad (1)$$

其中 $d = x - x_k$

求解目标函数的最小值即转化为求 (1) 式的最小值。

根据极值存在必要条件, (1) 式中目标函数对  $d$  的导数为 0, 则需要满足

$$-\nabla f(x_k) = H_k d \quad (2)$$

(2) 式解得

$$d_k = -H_k^{-1} \nabla f(x_k) \quad (3)$$

(3) 式即为牛顿法的搜索方向, 又称牛顿方向。

更新当前搜索点为

$$x_{k+1} = x_k + d_k \quad (4)$$

接着计算新搜索点的目标函数值进行迭代计算直至满足终止条件。

根据以上叙述可以看出, 牛顿法对目标函数的 Hessian 矩阵十分依赖, 矩阵非正定, 不能保证下降方向, 且迭代求解搜索方向时需要不断计算 Hessian 矩阵的逆矩阵, 计算量大, 甚至如果 Hessian 矩阵不可逆的话迭代将无法进行。为了解决这个问题, 下面提出准牛顿法进行改进。

## 2.4 准牛顿法

准牛顿法的基本思想是以牛顿迭代法为基础, 不对 Hessian 矩阵直接或通过分解法求逆, 而是直接通过梯度信息建立 Hessian 逆矩阵的近似矩阵, 代替牛顿迭代中的 $H^{-1}x$ , 进行搜索迭代, 直至满足收敛条件。

准牛顿法的搜索方向为

$$d_k = -\alpha^k A^k \nabla f(x^k) \quad (5)$$

式中,  $\alpha^k$ 为步长,  $A^k$ 为变尺度矩阵, 在迭代过程中逐渐逼近牛顿迭代中的 $H^{-1}x_k$ 。为了满足 $A^k$ 可以收敛至 $H^{-1}x_k$ , 需要满足

$$\Delta x^k = A^{k+1} \Delta g^k \quad (6)$$

式中,  $\Delta x^k = x^{k+1} - x^k, \Delta g^k = \nabla f(x^{k+1}) - \nabla f(x^k)$

变尺度矩阵在迭代过程中是不断变化的, 故此会产生一个序列 $\{A^k\}$ 。所以构造变尺度矩阵时必须给定一个初始的 $A^0$ , 且 $A^0$ 必须是正定对称的, 可以认为 $A^1 =$

$A^0 + \Delta A^0$ ，其中 $\Delta A^0$ 为校正矩阵，因此变尺度矩阵求解的迭代公式为

$$A^{k+1} = A^k + \Delta A^k \quad (7)$$

下面给出 BFGS 迭代公式计算变尺度矩阵

$$A^{k+1} = \frac{1}{(\Delta x^k)^T \Delta g^k} \begin{Bmatrix} \Delta x^k (\Delta x^k)^T + \frac{\Delta x^k (\Delta x^k)^T (\Delta g^k)^T A^k \Delta g^k}{(\Delta x^k)^T \Delta g^k} \\ -A^k \Delta g^k (\Delta x^k)^T - \Delta x^k (\Delta g^k)^T A^k \end{Bmatrix} \quad (8)$$

基于 BFGS 的准牛顿法优化流程如下：

1. 给定初始迭代点  $x^0$  和收敛精度  $\varepsilon$ 。
2. 计算  $\nabla f(x^0)$ ，取  $A^0 = I$ ，令  $k = 0$ 。
3. 构造搜索方向  $s^k = -A^k \nabla f(x^k)$ 。
4. 沿着构造的方向进行一维搜索，求最优搜索步长  $\alpha^k$ ，使

$$f(x^k + \alpha^k s^k) = \min f(x^k + \alpha s^k)$$

5. 更新搜索点  $x^{k+1} = x^k + \alpha^k s^k$ 。

6. 判断收敛条件  $|\nabla f(x^k)| < \varepsilon$ ，满足则终止迭代；否则更新变尺度矩阵  $A^{k+1}$  和迭代步  $k = k + 1$ ，回到 3) 迭代。

## 2.5 遗传算法

### 2.5.1 算法原理

遗传算法是一种模拟生物进化过程的全局优化算法，它基于达尔文的“自然选择”和孟德尔的“遗传机制”，通过模拟种群中个体的选择、交叉、变异等操作来逐步逼近问题的最优解。其核心思想涵盖适应度（Fitness），用于衡量个体优劣（如目标函数值）；染色体编码，将解空间映射为基因型（如二进制串）；以及种群进化，通过迭代生成更优的下一代种群。该算法具有无需梯度信息的特点，适用于不可导、非凸、多峰函数；具备全局搜索能力，可通过随机性避免陷入局部最优；还拥有并行性，能同时评估多个解。

### 2.5.2 关键操作步骤

遗传算法的优化流程如下：

#### 1. 编码与初始化

编码方式：将设计变量转换为染色体（如二进制、实数编码）。

初始化种群：随机生成  $n_p$  个个体，形成初始种群。

#### 2. 选择

从当前种群中选择适应度高的个体进入交配池，常用方法：

轮盘赌选择 (Roulette Wheel)：个体被选中的概率  $p_i = \frac{f_i}{\sum f_j}$ ，适应度越高概率越大。

### 3.交叉

单点交叉：随机选择交叉点，交换父代部分基因。

### 4.变异

引入随机扰动，增加种群多样性。

单点变异：随机翻转某一位基因（如  $0 \rightarrow 1$ ）。

### 5.终止条件

最大迭代次数：达到预设代数  $(i)_{\max}$ 。

适应度收敛：种群适应度标准差  $S_f \leq (S_f)_{\max}$ 。

最优解稳定：连续多代最优解无显著改进。

## 三. 参数化方法的选择原因

### 3.1 贝塞尔曲线的缺点

#### 1.局部控制不精确

贝塞尔曲线通过控制点来定义曲线，但这些控制点的影响具有全局性，即改变某一个控制点时，曲线的其他部分也会发生变化。这在复杂翼型设计中，尤其是需要精确控制某个局部形状时，会显得不够灵活。对于需要局部精细调整（如前缘或后缘的细节）的设计，贝塞尔曲线难以满足设计师的需求。

#### 2.难以描述复杂翼型

尽管贝塞尔曲线可以创建平滑的曲线，但它的表达能力有限，尤其是在面对更复杂的翼型（例如具有多个拐点或突变的形状）时，贝塞尔曲线无法有效地表示这些复杂的特征。对于翼型的形状要求较高的设计，贝塞尔曲线可能无法完美表达，尤其在翼型厚度分布或弯度变化较大的情况中更为明显。

#### 3.形状控制不直观

贝塞尔曲线的控制点与曲线形状之间的关系是非线性的，这使得设计者在调整时难以预见到每个控制点对最终形状的影响。因此，在优化过程中，设计者需要反复调整控制点，增加了计算和调试的复杂度。而 CST 方法则提供了更直观的控制，设计者可以直接修改特定的设计参数（如厚度分布、前缘角度等）来控制翼型的外形。

#### 4.优化效率低

在反优化过程中，贝塞尔曲线通常需要较多的计算和迭代，尤其是在复杂翼型设计中，优化过程中容易出现收敛速度慢的问题，尤其是与 CST 方法相比，它的优化效率较低。

### 3.2 CST 方法的优点

#### 1. 更高的形状表达能力

CST 方法通过将基础形状（类别函数）与形状变换函数（通过 Bernstein 多项式表示）结合，可以灵活地表达复杂的翼型。通过对 Bernstein 多项式系数的调整，CST 方法能够精细控制翼型的前缘、后缘、曲率、厚度分布等特征。相比贝塞尔曲线，CST 方法在复杂翼型的表达上具有更强的能力，能够更准确地满足气动性能要求。

#### 2. 精确的局部控制

CST 方法通过多个参数（如 Bernstein 多项式系数）来调节形状，这些参数能够精确地控制翼型的局部区域。比如，可以单独控制翼型的前缘圆度、后缘厚度等，不会像贝塞尔曲线那样引起全局形状的剧烈变化。这使得 CST 方法在需要局部精细调整的设计中具有显著优势。

#### 3. 更直观和易于优化

CST 方法通过类别函数和形状函数的组合，使得翼型的设计和优化过程更加直观。设计者可以直接通过调整少量的控制参数（如 Bernstein 多项式的系数）来控制翼型的整体和局部特性。与贝塞尔曲线的非线性控制不同，CST 方法的控制参数与形状的关系更加直观，优化过程通常更为高效和精确。

#### 4. 高效的反优化与多目标优化

CST 方法适用于复杂的反优化问题，特别是在涉及多个设计目标（如气动性能、结构强度、制造成本等）的情况下。由于 CST 方法具有较强的全局控制性，它能够高效地引导优化算法收敛到期望的设计，减少了迭代次数和计算开销。此外，CST 方法能够处理复杂的形状变换，非常适合多目标优化问题。

#### 5. 适应性强和灵活性高

CST 方法能够根据设计需求灵活地改变翼型的各个部分（如前缘、后缘、翼厚分布等）。它不仅能够表达传统的对称或非对称翼型，还能够处理更为复杂的翼型设计，满足不同的气动性能需求。设计者可以方便地在 CST 方法框架下尝试不同的设计策略。

#### 6. 易于进行参数化设计与制造

CST 方法能够将翼型设计的几何形状与性能目标紧密结合,且参数化程度高。这使得设计师可以将翼型设计与数值模拟或制造工艺进行紧密耦合,优化设计过程中的各项参数,最终达到最优的设计效果。

### 3.3 总结

1.贝塞尔曲线的缺点主要在于局部控制不精确、难以描述复杂形状、优化效率低等方面,尤其是在复杂翼型设计和精细调节的场景下,贝塞尔曲线可能无法满足高精度的设计需求。

2.CST 方法提供了更高的形状表达能力、精确的局部控制和更直观的优化过程,因此在复杂翼型的设计和反优化中具有显著优势,尤其是在多目标优化和高效计算方面表现出色。

## 四. 优化算法的选择原因及参数调整

### 4.1 准牛顿法-黄金分割法

#### 4.1.1 黄金分割法选择原因

1.黄金分割法是一种一维搜索方法,它只需要比较目标函数在不同点的函数值大小,而不需要计算目标函数的导数。这对于一些难以求导或者导数计算成本较高的目标函数来说非常有用。

2.黄金分割法的收敛速度是线性的,虽然不是很快,但它的收敛速度比较稳定,不会出现剧烈的波动。在最速下降法中使用黄金分割法进行一维搜索,可以保证在每次迭代中都能找到一个相对较好的步长,从而保证算法的稳定性。

3.黄金分割法的实现非常简单,只需要按照固定的比例缩小区间范围,不断迭代直到满足停止条件即可。这种简单的算法结构使得它在实际应用中易于实现和调试。

#### 4.1.2 准牛顿法选择原因

1.相较于只使用一阶导数(梯度)信息的最速下降法,准牛顿法尝试利用目标函数的二阶导数(Hessian 矩阵)信息。二阶导数能反映函数的曲率,帮助算法更准确地把握函数的局部形状,从而更有效地确定搜索方向和步长,使得算法在接近最优解时具有更快的收敛速度。

2.避免精确计算 Hessian 矩阵,可以降低对目标函数计算复杂度。对于大规模优化问题,精确计算目标函数的 Hessian 矩阵及其逆矩阵的计算量非常大,存储 Hessian 矩阵也需要大量的内存空间。

3.稳定性较好, 准牛顿法在迭代过程中会根据目标函数的局部特性自适应地调整搜索方向。近似的 Hessian 矩阵或其逆矩阵能够反映函数的曲率变化, 使得算法在遇到函数的复杂地形(如局部极小值、鞍点等)时, 能够更好地绕过这些陷阱, 朝着全局最优解或局部最优解前进。

4.与一些对初始点敏感的优化算法相比, 准牛顿法通常具有较强的鲁棒性。它对初始点的选择不太苛刻, 在不同的初始点下都能有较好的收敛性能。这使得准牛顿法在实际应用中更加可靠, 能够处理各种不同类型的优化问题。将两种方法结合使用, 既实现了全局搜索过程, 又极大减少了计算量, 充分发挥各算法的优势。

### 4.1.3 参数调整

#### 1.容忍度 tol

容忍度 tol 是用来控制算法收敛的精度。当目标函数在当前点的梯度的范数小于 tol 时, 算法认为已经找到了一个满足精度要求的解, 从而停止迭代。其调整策略为:

1) 如果 tol 设置得太小, 算法会进行更多的迭代, 以达到更高的精度, 但这会增加计算时间。

2) 如果 tol 设置得太大, 算法可能会过早停止迭代, 导致找到的解不够精确。

因此, 在本算法中取了一个折中的容忍度值  $10^{-6}$ , 使得计算时间相对合理, 也可以达到较高的精度。

#### 2.最大迭代次数 max\_iter

最大迭代次数的作用是为了防止算法陷入无限循环。如果算法在达到最大迭代次数之前还没有满足收敛条件, 就会强制停止迭代。其调整策略为:

1) 如果 max\_iter 设置得太小, 算法可能在还没有收敛到一个较好的解时就停止迭代。

2) 如果 max\_iter 设置得太大, 会增加不必要的计算时间。对于一些简单的目标函数, 可能在较少的迭代次数内就能收敛到较好的解, 过大的 max\_iter 值会浪费计算资源, 延长计算时间。

因此, 在本算法中设置 max\_iter 的值为 500, 既保证算法有足够的迭代次数找到较优解, 又不过多浪费计算资源。

#### 3.中心差分步长 h

中心差分步长 h 用于计算目标函数的数值梯度。它的大小会直接影响数值梯度的计算精度, 进而影响最速下降法的收敛性能。在本算法中取  $10^{-6}$  这样一个



值,是在考虑计算精度和计算机数值精度限制的情况下,经过权衡得到的一个较为合适的值,能够在一定程度上保证数值梯度的计算精度,进而提高算法的性能。

### 4.2 遗传算法

#### 4.3.1 选择原因

1.遗传算法模拟生物进化过程,通过选择、交叉和变异等操作在解空间中进行搜索。它不依赖于目标函数的梯度信息,能够在复杂的、多峰的解空间中探索不同区域,具有很强的全局搜索能力,相比基于梯度的方法,更有可能找到全局最优解,而不是陷入局部最优陷阱。

2.对于目标函数的连续性、可导性等没有严格要求,无论是连续函数、离散函数,还是具有噪声、不连续的函数,遗传算法都能适用。在处理一些传统优化方法难以解决的复杂问题,如组合优化问题、多目标优化问题时,遗传算法展现出良好的适应性和鲁棒性。

3.遗传算法每次迭代操作是对一个种群进行处理,种群中的多个个体可以并行计算,这种天然的并行性使得遗传算法可以充分利用多核处理器或分布式计算环境,提高计算效率,尤其适用于大规模优化问题。

#### 4.3.2 参数调整

1. 种群规模决定了遗传算法在每一代中包含的个体数量。较大的种群规模意味着算法能够在更广泛的解空间中进行搜索,增加找到全局最优解的可能性,但同时也会增加计算量和计算时间;较小的种群规模计算效率较高,但可能导致算法过早收敛,陷入局部最优。因此,在综合考虑问题的规模和复杂度,种群数量设置为 300。

2. 交叉操作是遗传算法中产生新个体的主要方式,交叉概率决定了种群中个体进行交叉操作的比例。较高的交叉概率可以增加种群的多样性,促进算法在解空间中进行更广泛的搜索,但过高的交叉概率可能导致优良个体的基因被破坏;较低的交叉概率则会使算法的搜索能力下降,收敛速度变慢,在本算法中将交叉概率设置为 0.9。

3. 变异操作是遗传算法中维持种群多样性的重要手段,适当的变异概率可以防止算法过早收敛,帮助算法跳出局部最优解;但变异概率过高会使算法退化为随机搜索,降低收敛速度。为了使得算法形成稳定的进化方向,增加对种群多样性的贡献程度,在本算法中变异概率为 0.01。

4.最大迭代次数分析同准牛顿迭代法,经过综合考虑,最大迭代次数取 1000 次。

## 五. 优化历程和最优解的分析

### 5.1 优化历程

#### 5.1.1 拟牛顿法

在本算法优化过程中，拟牛顿法展现出卓越的性能。它优化速度极快，能迅速得到接近目标翼型的结果，运行时间相当短。这是因为拟牛顿法通过构造近似海森矩阵的逆矩阵，极大减少计算量，在每次迭代中能高效地确定搜索方向，从而快速逼近最优解。

随着迭代次数的增加，拟牛顿法对翼型的调整幅度逐渐变小，仅做一些微小的优化。这体现了拟牛顿法的收敛稳定性，每一次迭代都在朝着最优解稳步前进。

最终，拟牛顿法得到的翼型与目标翼型的近似程度非常高，说明它在局部搜索方面具有很强的能力，能够精准地找到最优解附近的精确位置，提高了优化效率和准确性。

#### 5.1.2 遗传算法

在本次遗传算法优化过程中，设定迭代次数为 1000 次，算法每运行 10 代输出一次当前最优适应度值。观察其优化轨迹发现，当迭代至第 50 代时，遗传算法的收敛曲线已趋于平稳，这一现象充分验证了种群规模、交叉概率、变异概率等参数设置的合理性，表明所选参数能够促使算法在较短时间内逼近最优解。

在翼型优化实践中，优化翼型与目标翼型的形态演变并非呈现单调趋近的规律，期间偶尔会出现差异显著的翼型形态。这种现象源于遗传算法内在的随机性，其通过选择、交叉、变异等操作对解空间进行探索，使得优化过程具有一定的不确定性和跳跃性。随着迭代的持续推进，算法逐渐收敛，最终输出的优化翼型能够较好地契合目标要求，证明了算法在复杂优化问题中的有效性和鲁棒性。

### 5.2 最优解分析

#### 5.2.1 拟牛顿法

在 500 次迭代过程中，拟牛顿法展现出强大的寻优能力，最终确定最优解位于区间 (0.180221, 0.190728) 内。该方法凭借对海森矩阵的近似构建，有效降低了计算复杂度，在每次迭代中都能快速调整搜索方向，使得算法以极高的效率逼近最优解。当目标函数值收敛至 0.000071 时，表明算法已精准捕捉到满足高精度要求的解，这一结果不仅体现了拟牛顿法在局部搜索上的高效性，更印证了

其求解的准确性。

### 5.2.2 遗传算法

从遗传算法的收敛结果来看，最优适应度值最终稳定在 0.0007。观察算法迭代过程，在仅进行 150 次迭代时，适应度值便已不再波动，持续保持在该数值水平。通过对比每 10 次迭代输出的适应度数据与整体收敛曲线可知，算法在早期迭代阶段就已完成收敛过程。这不仅印证了算法参数设置的合理性，也表明该算法成功找到了符合预期的高质量最优解，在翼型优化问题上取得了理想的求解效果。

## 六、优化结果分析

### 6.1 拟牛顿法

运行 E2.m 可得基准翼型在优化后得到目标翼型的图像，见图 1。从运行过程中可得，初始迭代时就已经得到了非常接近目标翼型的优化结果，随着迭代次数不断增加，二者的差异呈现指数级缩减，外形轮廓拟合效果非常好。

由于拟牛顿法的强鲁棒性，即便初始翼型与目标差异较大，也能凭借合理的迭代策略快速缩小差距，加速收敛进程。此外，拟牛顿法使用近似的 Hessian 矩阵进行迭代计算，可以减少求解逆矩阵的计算量，使得其收敛速度加快，进而得到理想的优化结果。

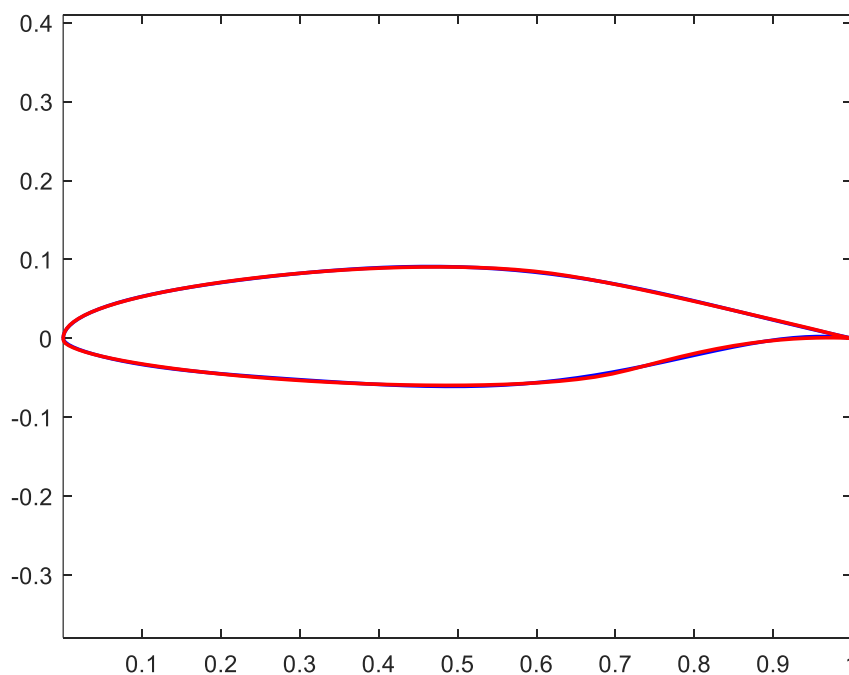


图 1 拟牛顿法优化翼型

### 6.2 遗传算法

在遗传算法的运行过程中，呈现出显著的动态演化特征与高效的收敛性能。算法每轮迭代中，翼型形态均发生显著改变，这主要归因于算法基于生物进化机制设计的随机性与多样性探索策略。遗传算法通过选择、交叉、变异等核心操作生成新个体，其中交叉操作以一定概率将父代个体的基因片段重组，变异操作则对个体基因进行随机扰动，这些随机过程使得种群在解空间中不断探索新的区域，进而导致翼型形态在迭代中持续大幅变动。

尽管存在随机性，但遗传算法展现出较快的收敛速度。当迭代至 70 次时，算法已接近收敛状态，原因如下：

- 1.遗传算法通过初始种群在整个解空间内的广泛分布，具备强大的全局搜索能力，能够快速定位到较优解的大致区域。同时，选择操作基于个体适应度进行筛选，使得更接近目标翼型的个体有更高概率被保留并繁衍后代，推动种群向更优解方向定向进化，有效平衡了探索与利用，避免陷入局部最优，加速了收敛进程。

- 2.在翼型优化场景中，算法依据预先设定的适应度函数对个体进行评估。每一次迭代都以提升适应度为目标，通过不断淘汰低适应度个体、保留和进化高适应度个体，使得种群整体的适应度不断提高，进而加速翼型向目标形态的逼近速度。

- 3.算法中种群规模、交叉概率、变异概率等关键参数的合理设置对收敛速度起到了重要作用。例如，适中的种群规模既能保证种群的多样性以充分探索解空间，又能避免规模过大导致计算效率降低；合适的交叉和变异概率使得算法在保持进化动力的同时，避免过度随机化而无法收敛，从而确保算法在有限的迭代次数内实现高效收敛。遗传算法优化结果见图 2 和图 3。

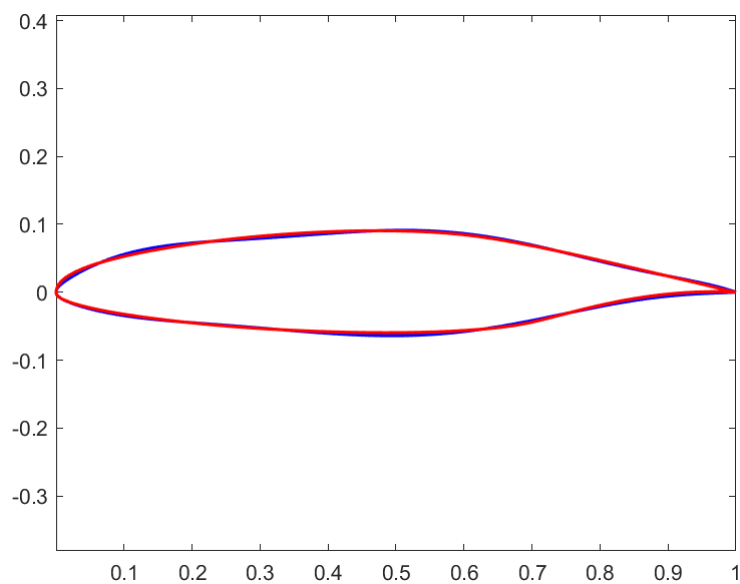


图 2 遗传算法优化翼型

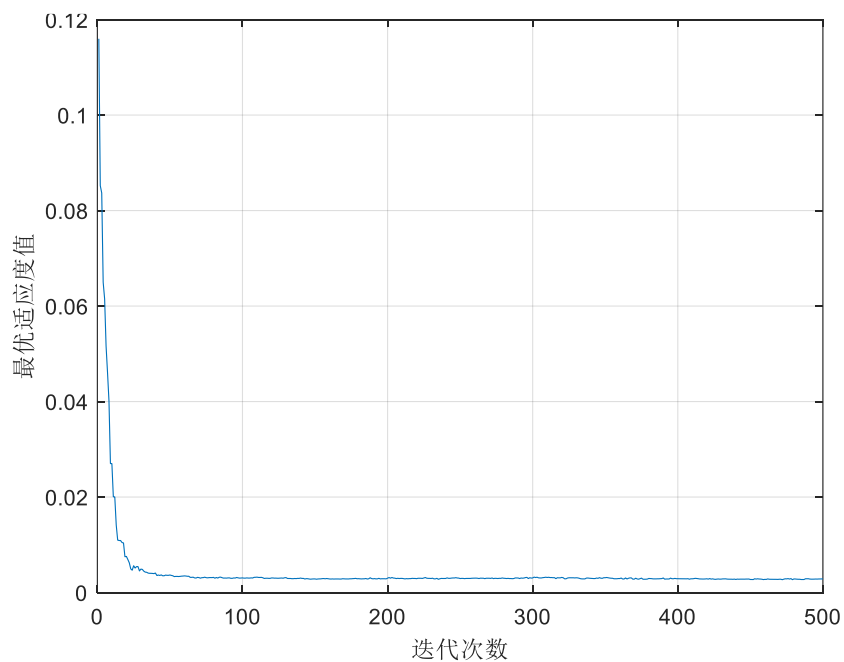


图 3 遗传算法迭代次数

注：关于初始代码的下翼面头部不收敛问题，产生的原因是下翼面的点变异后可能会大于 0，导致收敛困难，产生尖头形状，解决策略是相对减小  $N1$  值，并对目标函数添加限制，当变异后的点的纵坐标大于 0 时进行取反。但有时变异会一直让下翼面头部点的纵坐标大于零，从而无法收敛（极小概率情况）

### 6.3 算法对比

遗传算法基于生物进化机制，从种群出发进行全局搜索，对目标函数的连续性和可导性无要求，相比拟牛顿法虽初期收敛慢，但全局寻优能力强，能有效跳出局部最优；而拟牛顿法依赖目标函数的导数信息易陷入局部最优。此外，遗传算法计算复杂度较高但可并行计算，参数调试要求高；拟牛顿法计算复杂度相对较低，参数依赖相对简单，二者各有利弊。