Projekt 2 – Optymalizacja profilu – Podstawy teoretyczne

Niniejszy projekt obejmuje optymalizację profilu płata pod względem różnych kryteriów, zdefiniowanych podczas wykonywania projektu. Celem ćwiczenia jest zdefiniowanie funkcji celu dla czterech zagadnień, przeprowadzenia procesu optymalizacji omówienie i przygotowanie wyników w postaci raportu.

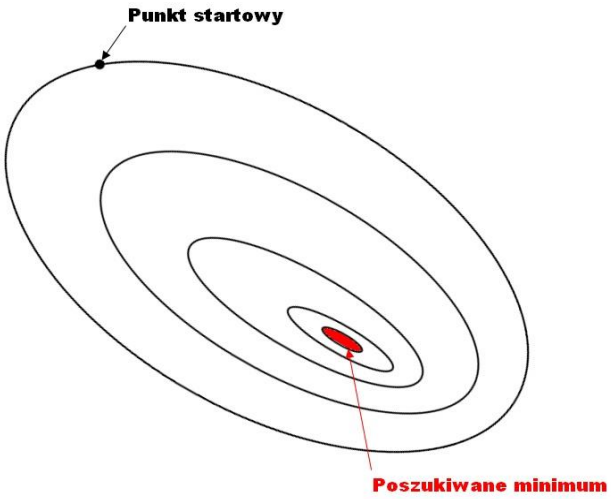
1 Wstęp

Modelowym przykładem zastosowania optymalizacji numerycznej w lotnictwie jest optymalizacja właściwości aerodynamicznych profilu. Przedstawiając zastosowanie nowych metod optymalizacji numerycznej, najczęściej wykorzystywanym przypadkiem testowym, jest właśnie optymalizacja profilu. Zależnie od typu samolotu i celu jaki inżynier chce osiągnąć, profil lotniczy można optymalizować na różne sposoby. Poprzez minimalizację oporu, maksymalizację doskonałości w warunkach przelotowych, maksymalizację współczynnik siły nośnej podczas manewru ciasnego zakrętu, itd. Zawsze należy pamiętać, że poprawa jednego parametru, odbywa się kosztem innego. W ten sposób maksymalizując współczynnik siły nośnej wzrośnie moment pochylający, a co za tym idzie potrzebne siły na usterzeniu poziomym, do zachowania równowagi samolotu i wzrost oporu całej konfiguracji. Świadomy projektant musi krytycznie podchodzić do koncepcji optymalizacji, którą wstępnie nakreślił. Musi zastanowić się, czy wybrana funkcja celu jest najlepszą z możliwych. Czy postawione przez projektanta zadanie optymalizacyjne daje rzeczywiste wyniki. Czy technologia wykonania, którą dysponuje jest wystarczająca do zrealizowania projektu i czy koszty realizacji nie są zbyt wysokie.

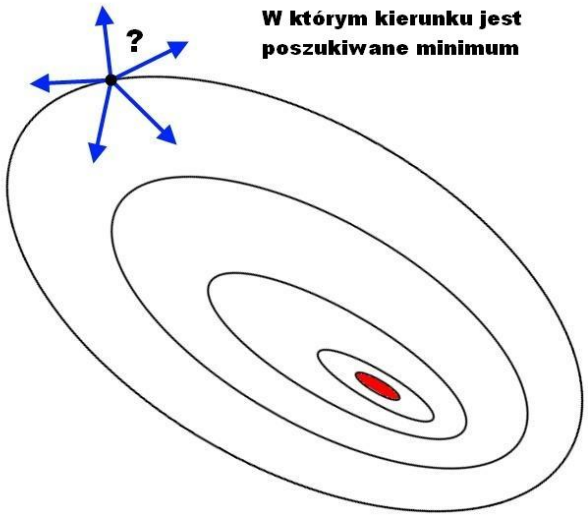
2 Opis programu i algorytmu optymalizującego

Student otrzymuje do dyspozycji kod źródłowy programu *OptoFoil*, w celu lepszego zrozumienia zasad działania oprogramowania bazującego na gradientowych metodach optymalizacji numerycznej. Program został napisany w języku C++ i skompilowany kompilatorem *gcc*, w środowisku graficznym *Dev-cpp*. Środowisko *Dev-cpp* jest łatwe do opanowania i bardzo podobne do środowiska graficznego *Visual Studio,* znanego z ćwiczeń informatyki. Autor starał się, by kod był jak najprostszy i przejrzysty, a oprogramowanie potrzebne do kompilacji darmowe i ogólnodostępne. Algorytm zawiera wszystkie niezbędne moduły wykorzystywane w profesjonalnych programach do optymalizacji. Program można dowolnie modyfikować i rozwijać, w powiązaniu z oprogramowaniem dedykowanym do analizy, w którym zaimplementowano możliwość wykorzystania skryptów i makr. Przykładowo: *Xfoil, AVL, Panukl, SDSA, Calculix, Fluent, Ansys,* itd...

Metody gradientowe należą do szerszej grupy kierunkowych metod poszukiwania minimum. W metodach kierunkowych wyznacza się punkt startowy, przyjęty arbitralnie, lub na podstawie jakichś przesłanek: wnioskując z analizy trendów, na podstawie doświadczenia inżynierskiego, na podstawie wstępnych obliczeń. Im lepiej zostanie dobrany punkt startowy tym większe będzie prawdopodobieństwo znalezienia globalnego minimum, a obliczenia będą trwały krócej. Tak postawione zadanie, dla dwóch zmiennych na płaszczyźnie, z nakreślonymi izoliniami wartości funkcji celu i zaznaczonym minimum, pokazuje *Rys. 1*. W rzeczywistości projektant nie wie jak wyglądają izolinie wartości funkcji celu, ani gdzie znajduje się minimum, gdyż wtedy nie potrzebna byłaby optymalizacja. W którym kierunku należałoby teraz się przemieścić (zmienić zmienne decyzyjne), aby znaleźć minimum *Rys. 2.*



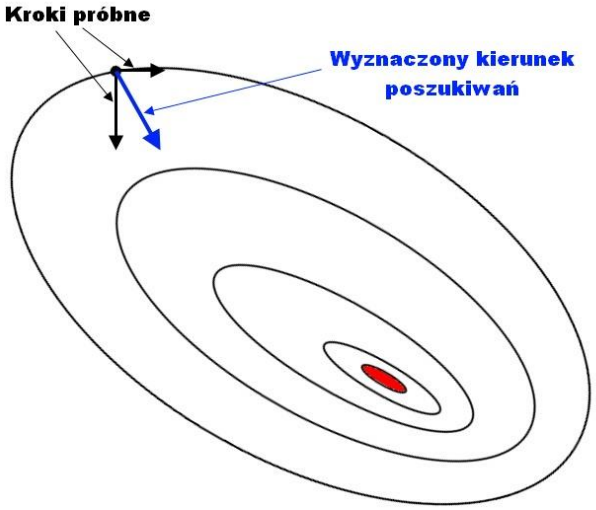
Rys. 1 Izolinie wartości funkcji celu, z zaznaczonym minimum i punktem startowym.



Rys. 2 Poszukiwanie kierunku optymalizacji

Wykonuje się małe kroki próbne, na podstawie których liczone są wartości funkcji celu, po wykonaniu kroku. Następnie, w zależności od zaimplementowanego algorytmu poszukiwania kierunku, ustalany jest kierunek zmian Rys. 3. Metody zerowego rzędu (bezgradientowe), korzystają jedynie z informacji o wartościach funkcji celu po wykonaniu kroków próbnych . Metody gradientowe pierwszego rzędu wyliczają pochodne kierunkowe pierwszego stopnia, dające informację o szybkości spadku wartości funkcji celu w danym kierunku. Natomiast metody gradientowe drugiego rzędu liczą również pochodne kierunkowe drugiego stopnia co daje informację o charakterystyce spadku wartości funkcji celu. Więcej informacji o funkcji celu powoduje szybsze zbieganie się algorytmu optymalizującego, ale zdobycie informacji może być bardzo kosztowne obliczeniowo. Wyliczenie numerycznych pochodnych kierunkowych stopnia pierwszego wymaga wywołania w programie funkcji celu dla każdej zmiennej, rezultatem będzie wektor pochodnych kierunkowych. Chcąc uzyskać wartości drugiej pochodnej należałoby wywołać funkcję celu jeszcze wielokrotnie, gdyż rezultatem jest symetryczna macierz pochodnych drugiego rzędu, łącząca zależności każdej zmiennej z każdą. Istnieją metody typu *adjoint*, uzyskujące informację o gradientach funkcji celu w trakcie pojedynczej analizy zagadnienia i nie zależą od ilości zmiennych decyzyjnych.

Niestety metoda *adjoint* wymaga bezpośredniej integracji z kodem źródłowym do analizy numerycznej.



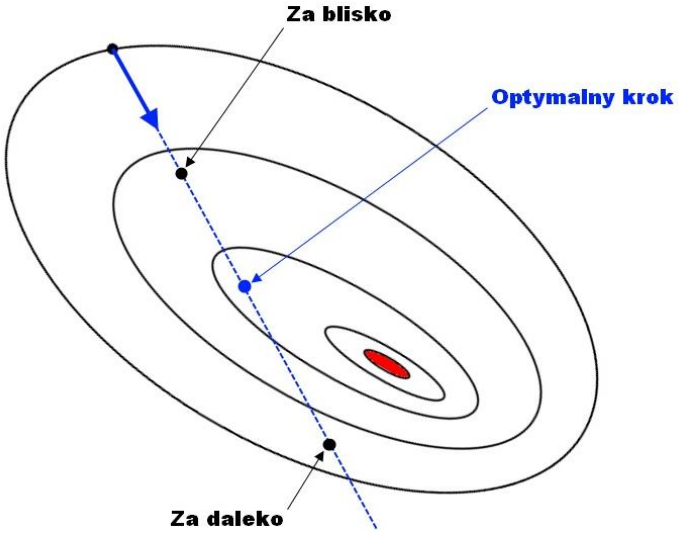
Rys. 3 Wyznaczanie kierunku poszukiwań minimum funkcji celu.

Mając wyznaczony kierunek, w którym chcemy poszukiwać minimum, należy podjąć decyzję jak duży wykonać krok w wyznaczonym kierunku. Zbyt krótkie kroki spowodują, że algorytm będzie się zbiegał bardzo długo, za długie, że można ”ominąć minimum, które założyliśmy, iż jest w pobliżu punktu startowego. Do określenia optymalnej długości kroku przeprowadza się poszukiwanie jednokierunkowe. Zakładamy, że nowe wartości zmiennych decyzyjnych będą wynikały ze wzoru (1), gdzie p_k jest wyznaczonym uprzednio wektorem kierunku, x_k to wektor zmiennych decyzyjnych, α_k poszukiwany skalar wzmocnienia, które definiuje długość kroku, oraz indeks k mówiący o aktualnie wykonywanej iteracji.

$$x_{k+1} = x_k + \alpha_k \cdot p_k \qquad (1)$$

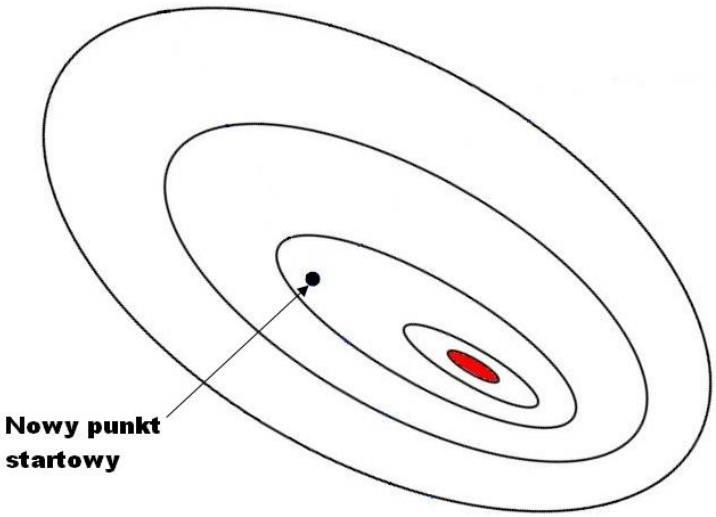
Nowe wartości zmiennych decyzyjnych wyliczane są poprzez dodanie do starych zmiennych decyzyjnych iloczynu wektora kierunku zmian p_k i wzmocnienia α_k. W ten sposób zakładamy, że nowe wartości zmiennych decyzyjnych zależą jedynie od wzmocnienia α_k, gdyż p_k zostało już jednoznacznie określone. Co za tym idzie funkcja celu jest funkcją jednej zmiennej, dlatego nazywamy ten proces poszukiwaniem w jednym kierunku Rys. 4.

W celu jak najszybszego znalezienia optymalnego kroku, zmiany funkcji celu, w zależności od α_k, aproksymuje się krzywą drugiego, lub nawet trzeciego stopnia i znajduje się jej minimum. Procedury poszukiwania optymalnego kroku są jednymi z najbardziej zagnieżdżonych w kodzie programu i często powtarzanych, dlatego muszą być bardzo efektywne.



Rys. 4 Wyznaczanie optymalnego kroku

Trzeba zapewnić zmniejszenie wartości funkcja celu w każdej iteracji, a dodatkowo, że algorytm będzie się szybko zbiegał. W tym celu stosuje się warunki (Armijo, Wolfa), które po spełnieniu zatwierdzają ostatni wykonywany krok z próbnym α_k. Zmienne decyzyjne są uaktualniane według wzoru (1) i wyliczana jest nowa wartość funkcji celu. W ten sposób otrzymany został nowy punkt startowy Rys. 5, do którego ponownie stosuje się opisywaną procedurę.



Rys. 5 Wyznaczenie nowego punktu startowego

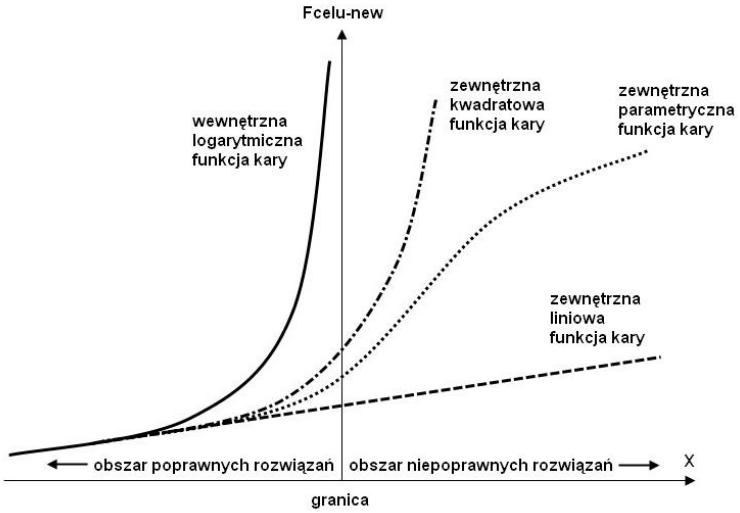
4 Nakładanie ograniczeń na funkcję celu

Optymalizacja w zastosowaniach inżynierskich często wymaga nałożenia ograniczeń, w celu otrzymania poprawnych i możliwych do zrealizowania rozwiązań. Ograniczenia można wprowadzić w dwojaki sposób, bezpośrednio lub przy pomocy funkcji kary. W naszym przykładzie zajmiemy się funkcją kary, która jest dość intuicyjna w zastosowaniu. Polega na zastąpieniu pierwotnej funkcji celu nową funkcją, która jest sumą pierwotnej funkcji celu i wartości funkcji kary (2). Jeżeli ograniczenia nie są naruszone, to wartość funkcji celu wynosi

zero. Takie podejście sprawia, że zadanie z ograniczeniami transformowane jest do zadania bez ograniczeń z nową funkcją celu.

$$F_{CELU-NEW} = F_{CELU} + F_{KARA} \tag{2}$$

Można dokonać generalnego podziału na metody funkcji kary wewnętrzne i zewnętrzne. Wewnętrzne funkcje kary nigdy nie przekraczają założonej granicy i jednocześnie nie pozwalają zbliżyć się do granicy nawet jeśli minimum znajduje się na niej. Dodatkowo tego rodzaju funkcje kary najczęściej realizowane są przez funkcje logarytmiczne, co może skutkować nieciągłością pochodnych nowej funkcji celu na granicy i problemy ze zbieżnością. Zewnętrzna funkcja kary najczęściej realizowana jest poprzez funkcję kwadratową z zachowaniem ciągłości pierwszej pochodnej na granicy, innymi słowy funkcja kary jest styczna do pierwotnej funkcji celu. Nadal mogą jednak występować nieciągłości pochodnych wyższego rzędu i stwarzać problemy w metodach wykorzystujących drugie pochodne, na przykład w klasycznej metodzie Newtona. Wadą funkcji zewnętrznych jest to, że najczęściej ograniczenia nieznacznie są przekraczane, jeśli minimum leży blisko granicy Rys. 6.



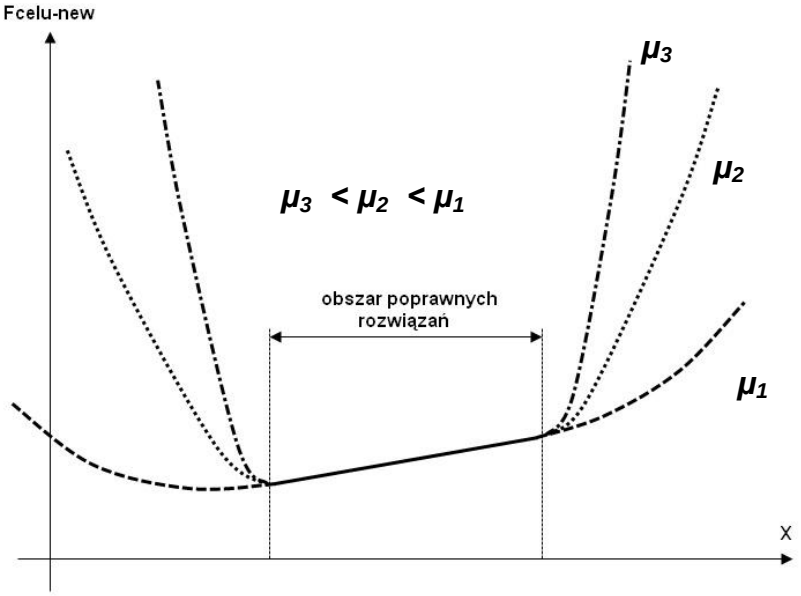
Rys. 6 Zewnętrzna i wewnętrzna funkcja kary

Zastosowanie klasycznej zewnętrznej kwadratowej funkcji celu, dla zadania zdefiniowanego jako (3), wyraża wzór (4).

$$\min F_{CELU} \text{ dla } x > g \tag{3}$$

$$F_{CELU-NEW} = F_{CELU} + \frac{1}{2 \cdot \mu} \sum c^2(x) \text{, gdzie } c = \max[g - x, 0] \tag{4}$$

Naruszanie granic można zmniejszyć, ale wiąże się to z narastającymi problemami z ciągłością funkcji i ze zbieżnością. Parametr μ we wzorze (4) kontroluje „promień" styczności funkcji kary do funkcji celu Rys. 7. Im mniejsze μ tym dokładniej zdefiniowana granica i większe problemy ze zbieżnością, a im większe μ tym bardziej rozmyta granica i mniej problemów ze zbieżnością algorytmu optymalizacji.



Rys. 7 Definicja obszaru rozwiązań

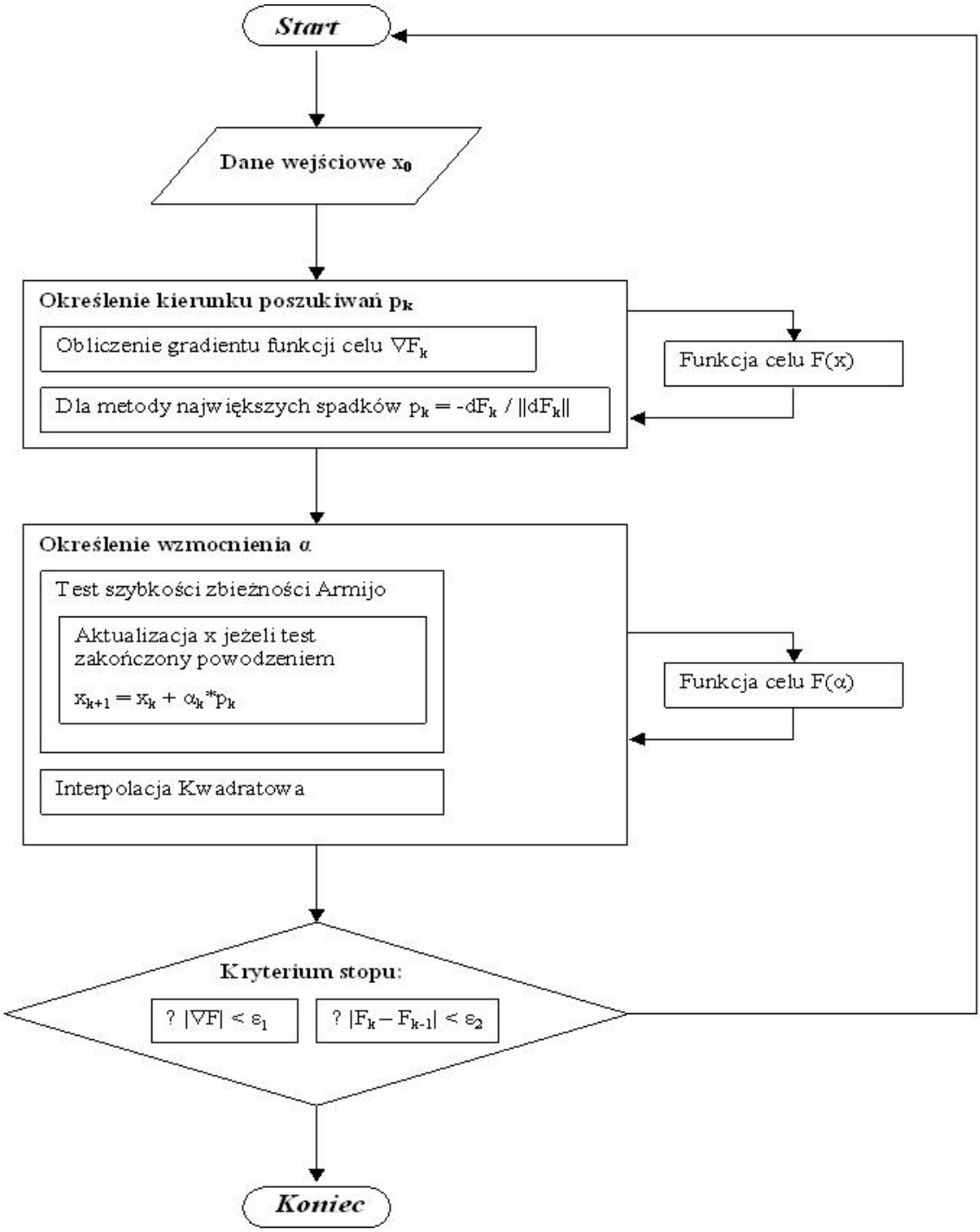
W bardziej wyrafinowanych algorytmach optymalizacji z ograniczeniami, wykorzystuje się metodę funkcji kary ze współczynnikami Lagrange'a, która przy pomocy mnożników Lagrange'a, dostosowuje odpowiednio parametr μ, w kolejnych iteracjach. Początkowo μ ma dużą wartość co zapewnia brak problemów ze zbieżnością. W miarę jak iteracja postępuje i zbliżamy się do minimum kroki algorytmu są coraz mniejsze. Można więc zmniejszyć parametr μ zwiększając dokładność wyznaczania granicy, bez narażania się na problemy ze zbieżnością. Metoda współczynników Lagrange'a sama odpowiednio dostosowuje wartość parametru μ na podstawie historii iteracji.

Ostatnią ważną sprawą jest kryterium stopu, tzn. momentu w którym uznaje się, że funkcja celu została wystarczająco dobrze zoptymalizowana, lub dalsze zmiany są tak powolne, iż ponosimy zbyt duże koszty obliczeniowe. Kryterium stopu można zdefiniować poprzez z góry określoną liczbę iteracji, lub warunki matematyczne, na przykład poprzez brak zmian funkcji celu, lub zerowanie się pochodnych kierunkowych.

Schemat na *Rys. 8* przedstawia kompletny algorytm optymalizacji zastosowany w programie *OptoFoil*.

5 Przebieg ćwiczenia - zadania do wykonania

Student ma do dyspozycji kod źródłowy programu *OptoFoil*, z którym musi się zapoznać, a następnie odpowiednio modyfikować w trakcie ćwiczenia, dostosowując do zadań jakie ma do wykonania. Dzięki dostępnemu kodowi źródłowemu, ma możliwość zapoznania się z prostym algorytmem optymalizacyjnym, który może być wykorzystany do praktycznych zadań. Ćwiczenie polegające na optymalizacji profilu lotniczego pod różnymi względami.



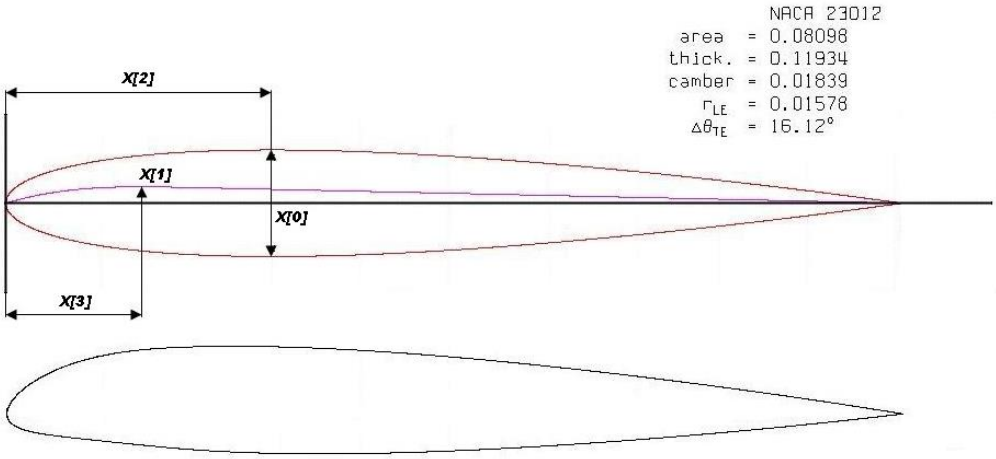
Rys. 8 Schemat optymalizacyjny

Projekt składa się z pięciu etapów. W pierwszym z nich student uczy się zasady na jakiej działa program OptoFoil pisząc skrypt. W pozostałych czterech student musi przeprowadzić optymalizację profilu uwzględniając cel optymalizacji. Następnie opracować wyniki i odpowiedzieć na pytania zawarte w opisie etapów ćwiczenia projektu.

5.1 Optymalizacja profilu – zmienne decyzyjne

Zmienne decyzyjne do optymalizacji profilu zostały zdefiniowane jak na Rys. 9. Rysunek ten w górnej części przedstawia rozkład grubości i krzywiznę szkieletowej dla przykładowego profilu NACA 23012, co w połączeniu daje rzeczywisty profil zobrazowany w dolnej części rysunku. Zmienne decyzyjne są to odpowiednio:

X[0] – maksymalna grubość profilu
X[1] – maksymalna strzałka ugięcia profilu
X[2] – położenie maksymalnej grubości profilu
X[3] – położenie maksymalnej strzałki ugięcia dla profilu



Rys. 9 Zmienne decyzyjne dla optymalizacji profilu

Literatura:

1. H.C. „Skip" Smith, *The Illustrated Guide to Aerodynamics*, TAB Books, McGraw-Hill, Inc., 1992
2. J. Nocedal, S. J. Wright, *Numerical Optimization*, Springer 1999
3. Numerical optimization techniques for engineering design