應用雙演化演算法於無人飛行載具結構最佳化設計之研究

張永康*, 張維恩, 蘇彤藹
淡江大學航空太空工程研究所

摘 要

本研究提出結合粒子群演算法與差分演化演算法的雙演化演算法於無人飛行載具結構最佳化設計中。粒子群演算法為仿生演算法，其特點為收斂速度快、參數設定少、搜尋範圍廣及具有記憶性。差分演化演算法為演化式演算法，其優勢在於參數設定及架構簡單、能維持母體的多樣性、高效能及高精確度等。雙演化演算法則是利用粒子群演算法與差分演化演算法兩者同時進行運算，優點在於互相補足缺點，利用差分演化演算法的多樣性使其跳脫區域最佳解，而粒子群演算法的記憶性使局部搜尋更加完善，應用兩者不同的搜尋方式，並將兩者演算法之最佳值做比較及分享以獲得最佳值。本研究中針對粒子群演算法提出變速因子的改良機制，藉由判斷粒子的區域最佳解與全域最佳解的距離來改變搜尋的步伐，以改善搜尋過程之收斂效率。本研究在差分演化演算法中選取適合的突變方式可增加解的多樣性以彌補粒子群演算法之不足。由數值分析範例之結果，顯示雙演化演算法求出的解比單獨使用粒子群演算法和差分演化演算法求出的解為佳且應用在無人飛行載具結構之最佳化設計上皆可得到不錯的結果。

關鍵字：雙演化演算法、無人飛行載具、最佳化設計。

Optimum Design of UAV Structures by Dual Evolution Strategy

Y.K. Chang *, W.E. Chang and T.W. Su
Department of Aerospace Engineering, Tamkang University, Taiwan, R.O.C

Abstract

The PSO-DE Dual Evolution Strategy was applied to optimum design of UAV structures in this study. Particle Swarm Optimization(PSO) algorithm is a bionic technique which has fast convergence, less parametric setting and wide search range with memory. Differential Evolution(DE) algorithm is an evolutional technique that has advantages of easy to implement, little parameter tuning requirement, and also exhibits reliable, accurate and fast convergence. In this study, a Dual Evolution Strategy that includes an improved Particle Swarm Optimization algorithm and a Differential Evolution algorithm is proposed for structural optimal design. The improved Particle Swarm Optimization algorithm adopts an alternation velocity factor that changes with the particle distance between the local and the global optimal solution. In the Differential Evolution algorithm, an appropriate mutation is selected to increase domain flexibility and improve the deficiency of Particle Swarm Optimization algorithm. This Dual Evolution Strategy utilizes the domain flexibility offered by the Differential Evolution algorithm and the local search memory of the Particle Swarm Optimization algorithm. The two algorithms are computed independently, the best result is obtained and shared between the two algorithms at each iteration. Numerical analysis showed that the results obtained from the Dual Evolution Strategy are better than those obtained from the Particle Swarm Optimization algorithm or Differential Evolution algorithm.

Keywords: Dual Evolution Strategy, Unmanned Aerial Vehicle, Optimum Design

一、前言

無人飛行載具(Unmanned Aerial Vehicle, UAV)是近年來航太產業發展的重點項目之一。早期的無人飛行載具僅用於在軍事上的需求，從演習時的靶機演進為無人偵察機，之後逐漸演進成軍民通用的多樣化選擇。其與遙控飛機最大的差異在於 UAV 可用來執行特別任務，例如偵查、救援、氣象探測等等。由於無人飛行載具包含飛具

本體，發動機、控制系統、任務酬載、地面導控及無線數據等次系統之整合。為了使無人飛行載具能減少消耗能量，達到延長滯空時間的目的，減輕整體飛機重量則是一大重點。而本研究針對機體結構來做輕量化設計，藉由機體結構輕量化來達到減輕能量消耗及延長滯空時間的效果。同時在設計上為了避免結構共振問題，本研究也將 UAV 結構自然振動頻率最大化做為研究之目標。

粒子群演算法(Particle Swarm Optimization, PSO)是由 Eberhart 和 Kennedy[1,2]兩位學者於 1995 年所提出，是藉由觀察鳥類覓食的行為，透過個體間傳遞訊息的方式，使整個鳥群朝同一方向或目標而去，藉此來達到所要的最佳化目標值，是種具有群體智慧的仿生演算法。此演算法的特色是搜尋範圍廣以及收斂速度快，且只需少數參數即可設定調整。因此有許多學者投入此演算法的研究，經過測試發現，此演算法仍有一些缺點，例如容易落入區域最佳解等。因此學者們希望能藉由對粒子群演算法作一些改良，以求得更佳的解。1998 年 Eberhart 和 Shi [3]提出慣性權重(Inertia Weight)的想法，藉由改變粒子的速度，找出粒子在全域搜尋與區域搜尋兩者間的平衡點，以便能用更快的速度求得全域最佳解。郭信川等人[4]於 2004 年提到，針對問題的不同，選擇合適的慣性權重，可以有效的減少粒子的搜尋時間。 Clerc[5]於 1999 年時，針對粒子群演算法的收斂情形，提出了一個稱做壓縮因子(Constriction Factor)的機制，此種含有壓縮因子的粒子群演算法稱 CPSO(Constriction Particle Swarm Optimization)，其出發點與慣性權重不同，希望能藉由壓縮因子的加入來縮短粒子的移動速度，讓粒子能改善區域搜尋的能力，使搜尋效果更佳。2004 年時 He 等人[6] 提出了被動聚集因子(Passive Congregation)的想法，此方法即是在粒子的速度更新中，加入一個被動聚集的部分。此部分代表粒子群中任何一個粒子對其他粒子的影響，藉由被動聚集因子的加入，當粒子在搜尋過程停滯時，該方法能夠改善粒子的停滯不前，使粒子具有較大的搜尋能力，能增加求得最佳解的機率。粒子群演算法被提出後，因其優點在於收斂速度快及參數設定較少，所以吸引了許多學者的關注。這些學者針對粒子群演算法做了一些相關測試，也從粒子群演算法中發現一些收斂特性的問題，所以學者們做了些相關的研究，希望能藉由對粒子群演算法加以改良，進而提升求解的品質及速度。改良方法大致可分為兩種，一是藉由改變參數設定，使其有更佳的搜尋能力；另一種則是結合不同演算法的優點，搭配粒子群演算法來做使用，期望能找到更佳的設計值。

差分演化演算法(Differential Evolution, DE)是由 Storn 及 Price[7]兩位學者於 1996 年所提出的，此演算法是近年來熱門的最佳化演算法之一[8]。由過去的文獻中發現，差分演化演算法在各領域中皆能表現出不錯的成效，其演算的概念類似於演化式演算法，具備有隨機搜尋的能力，以及還包括了突變、重組及選擇運算等機制。差分演化演算法與其他較主流的演化式演算法相較下，具備的優勢有參數設定簡單、整體的強健性、實作容易、高準確性以及收斂速度快等等特性[9,10]。由過去文獻可得知，差分演化演算法具有比其他改良的演化式演算法之求解效能較佳的優勢。雖然差分演化演算法有上述之優點，但是還是會有容易落入區域最佳解或是跳脫出解空間等問題。

雙演化演算法是結合粒子群演算法與差分演算法來執行的演算法。粒子群演算法是觀察鳥類覓食行為而發展出的演算法，而差分演算法，其前身是演化式演算法的，演化式演算法是模擬生物的演化機制所提出的的方法。雙演化演算法結合兩者不同的搜尋方式期望求得較佳的解，並將其解分享給另一演算法以求得最佳的解。本研究即利用上述演算法之優點求解最佳化的問題，期望能求得全域最佳解。

二、粒子群演算法

粒子群演算法(Particle Swarm Optimization, PSO)是由 Eberhart 和 Kennedy[1,2]於 1995 年所提出，藉由觀察鳥類覓食的生態而得到啟發，將其模擬成一種最佳化演算法，藉由個體與個體間的互動規則，傳遞訊息的方式，以及覓食的行為所提出的演算法，是一種具有群體智慧概念的仿生演算法[11]。其主要概念是假設一群小鳥於空間中隨機飛行尋找食物。原先不知道食物的所在地，由小鳥原身的經驗及直覺尋找食物，飛往自身學得有可能存在的地點覓食，而每隻小鳥的經驗與直覺不同。當找到食物時，他們會傳遞訊息給同伴，使其鳥群飛往此區域。在粒子群演算法中，將「粒子(Particle)」模擬成空間中的小鳥，而每個粒子的移動都是獨立的，所在的位置都是最佳化問題的一個解[4]。每個粒子獨立搜尋，當個體遇到函數最佳解時，其最佳搜尋資訊將被記錄在個體記憶中，亦即每個粒子都擁有本身最佳的搜尋資訊記憶。每個粒子速度不同，其更新方法依據本身不同的速度、自身經歷過的個體最佳解

(Pbest)以及所有個體經歷過的群體最佳解(Gbest)三者來做更新速度的依據。藉由更新速度來決定粒子所要移動的距離及方向，並產生粒子新的位置，再由粒子位置的適應值來判別目前解的好壞，經由此方式，反覆迭代直到找到最佳解。在粒子群演算法中的位置更新公式以及速度更新公式如式(1)及(2)來表示。

$$V_{p_i}^{k+1} = V_{p_i}^{k} + c_1 r_1 \left(P_i^k - X_{p_i}^{k}\right) + c_2 r_2 \left(P_g^k - X_{p_i}^{k}\right) \quad (1)$$

$$X_{p_i}^{k+1} = X_{p_i}^{k} + V_{p_i}^{k+1} \quad (2)$$

其公式中：

V_{p_i}：第 i 個粒子的速度

X_{p_i}：第 i 個粒子的位置

P_i：第 i 個粒子所經歷過的最佳位置

P_g：所有粒子經歷過的最佳位置

r_1、r_2：為[0,1]之間的隨機亂數

c_1、c_2：學習因子

k：迭代次數

其中學習因子c_1、c_2代表粒子朝向個體最佳解以及群體最佳解的權重。當此值太低時，粒子的搜尋次數會增加許多，好處是可以增加搜尋到全域最佳解的機率，壞處是需要耗費相當長的時間；相對的，當值太高時，好處是粒子的搜尋速度加快，壞處是可能因為粒子搜尋速度過快導致錯過了全域最佳解。而 Eberhart 等人[1]在做過相關測試之後，將c_1、c_2值建議設定為 2，此值可以取得較佳的權重配置。由位置更新公式(1)來看，此公式可分為三個部分來說明，第一部分為粒子本身的速度，第二部分為粒子自身的經驗來改變搜尋方向，即為「認知(Cognition)」，第三部分為粒子藉由群體的學習來改變的搜尋方向，即為「社會(Social)」，經由此三部分的組合來互相影響，更新並迭代產生新的速度。經過多次迭代運算後，根據收斂條件來判斷是否求得最佳解 [4]。

本研究利用壓縮因子的概念所提出變速因子的方法，壓縮因子是藉由縮短各粒子在搜尋過程中速度的移動範圍，導致搜尋時間過長，於是本研究設計了一個變速因子R，結合一個判斷公式α，以調整粒子的搜尋速度，減少搜尋的時間。以下為使用變速因子後之速度更新公式(3) ：

$$V_{p_i}^{k+1} = R\left[V_{p_i}^{k} + c_1 r_1 \left(P_i^k - X_{p_i}^{k}\right) + c_2 r_2 \left(P_g^k - X_{p_i}^{k}\right)\right] \quad (3)$$

$$\alpha = \left|\frac{P_g^k - P_i^k}{P_g^k}\right| \quad (4)$$

$$R = \begin{cases} 0.8 & \text{if} \quad \alpha \geq 0.5 \\ 0.5 & \text{if} \quad 0.2 \leq \alpha \leq 0.5 \\ 0.3 & \text{if} \quad \alpha \leq 0.2 \end{cases} \quad (5)$$

式中的 R 為變速因子，設定大小參考常數型慣性權重[3,12]的設定以及壓縮因子[13]的最小值，藉由此值變化速度的大小。α為判斷式，藉由判斷式可知個體最佳解與群體最佳解的距離，當$\alpha > 0.5$時，代表個體最佳解與群體最佳解距離還很遠，所以將步伐調大，使搜尋範圍加大。當 α 介於 0.2 到 0.5 之間，代表兩者數值有在接近的趨勢，當$\alpha < 0.2$時，代表距離相當接近，已經接近最佳解，所以將步伐調小，促使搜尋更加精準。

三、差分演化演算法

差分演化演算法(Differential Evolution, DE)是由學者Storn 及Price[7]於1995年所提出，對於解決全域最佳化是一種強而有力的演算法。其基礎概念源自於學者為了求解切比雪夫多項式(Chebychev Polynomial)的問題所啟發，利用差向量來搜尋以及迭代的構想所提出。經過多年來的研究與發展，差分演化演算法已經能應用於電力調度、地震震央位置判定、醫療X光片分析及最佳化等等各領域。目前還有許多學者仍然對差分演化演算法進行研究，期望能使差分演化演算法變得更為強健、更有效率。差分演化演算法優勢在於參數設定簡單、架構簡單、程式容易編寫以及高效能等優點。雖然有以上優點，但是有時仍有落入區域最佳解的可能性以及收斂性不穩定的情形發生。於是學者針對此缺點進行改良，目前主要改良的方法分為三種，一種是以差分演化演算法與其他演算法做結合[14]，目的是以其他演算法的優點加強差分演化演算法的搜尋或演算部分；另一種則是改良內部參數[15]，例如改變突變權重等等；最後一種則是更改差分演化演算法架構流程[16]。以上三種改良法最主要的目的都是在於避免落入區域最佳解、使收斂更穩定以及能更精準的找到全域最佳解。

差分演化演算法是以向量為基礎，透過母體間個體的差異性，並以隨機搜尋 (Stochastic Direct Search) 的方式將差異向量加入個體中，並透過突變 (Mutation) 、 重 組 (Recombination) 、 選 擇 (Selection)三個步驟進行演化，透過迭代不斷地進行運算，以期能在搜尋空間中找到最佳解。

差分進化演算法的主要使用的參數包括有群體大小、突變權重因子及交換率，茲說明如下[17]：

(1) 群體大小 (NP) ：

群體的大小將影響迭代中運算的向量總數。越大的群體運算所需的時間越長；相對的，越小的群體雖然運算時間快，但是過小的群體在一個較大的解空間中，搜尋效果會較差。

(2) 突變權重因子 (F) ：

突變權重因子是影響解向量的擾動量。當突變權重因子的值越大時，向量在解空間中的變化幅度越大，也代表者群體中有較好的能力能跳出區域最佳解；相反的，當突變權重因子的值越小，收斂速度會越快，但是會提高陷入區域最佳解的機率。一般來說，突變權重因子其值通常設定為0到2 之間；大部份的學者以經驗法則將F 的值設為0.4 到1 之間。

(3) 交換率 (CR) ：

交換率決定擾動後的合成向量與原始向量交換比例的多寡。交換率越高則代表兩個向量中有越高被互相交換的比例。

差分演化演算法的概念和基因演算法相似，主要的演算流程有初始化、突變、重組、選擇，根據文獻其流程步驟如下[7]：

(1) 初始化 (Initialization)

在初始化過程中，需要產生隨機參數向量 (Parameter Vectors) $X_d(t)$，公式表示如下：

$$X_d(t) = \left\{X_{d_{i,1}}, X_{d_{i,2}}, \dots, X_{d_{i,D}}\right\},\ i = 1,2,\dots,NP \quad (6)$$

NP：為群體大小

D：為向量數量

此組數值多寡將影響到搜尋空間大小以及運算時間長短。

(2) 突變 (Mutation) ：

產生參數向量後，需隨機由母體中選取三個參數向量，分為$X_{d_{r1,G}}, X_{d_{r2,G}}$及$X_{d_{r3,G}}$，$r1, r2, r3 \in \{1,2,3,\dots,NP\}$，透過突變因子可得到突變向量$V_{d_{i,G+1}}$。本研究使用差分演化演算法的突變公式如下：

$$V_{d_{i,G+1}} = X_{d_{r1,G}} + F(X_{d_{r2,G}} - X_{d_{r3,G}}) \quad (7)$$

F：突變權重因子

G：迭代次數

此公式為基本突變公式，利用一變異量$(X_{d_{r2,G}} - X_{d_{r3,G}})$，乘上突變權重因子$F$，$F$ 則介於[0,2] 之間並由使用者自行設定，主要是控制變異量之突變率；若F 設定過大，則所產生的突變向量會呈現過於離散之狀態，反之則無法發揮變異量的效用。

(3) 重組 (Recombination) ：

重組是把突變步驟產生的突變向量，與原先參數向量$X_d(t)$重組，來產生一組新的重組向量$u_{i,G+1}$。此機制需設定交換率CR值，利用CR值與一隨機選取的亂數值 (rand) 做比較，若此亂數值小於初始設定之CR值，則新的重組向量中則放入$V_{d_{i,G+1}}$值；反之，若大於CR值則由$X_{d_{i,G}}$取代，如式(8)所述：

$$u_{i,G+1} = \begin{cases} V_{d_{i,G+1}} & if\ rand \le CR \\ X_{d_{i,G}} & if\ rand > CR \end{cases} \quad (8)$$

其中，rand是由0與1之間的均勻隨機亂數所選取。CR值若越小，重組向量中則保留大部分的參數向量，導致重組向量與參數向量的相似度越高；反之，若CR值越大，則重組向量中會得到更多的突變向量，這代表重組向量與參數向量的差異性越大。

(4) 選擇 (Selection) ：

在選擇的步驟中，藉由計算適應值來評估該選擇重組向量，或是原始的參數向量進入下一次迭代。因此在完成突變及重組機制產生的重組向量$u_{i,G+1}$後，計算其適應值，與原始的參數向量$X_{i,G}$之適應值做比較，適應值較佳者會被保留當成下一迭代的參數向量。其判斷公式如式(9)所述：

$$X_{d_{i,G+1}} = \begin{cases} u_{i,G+1} & if\ F(u_{i,G+1}) \le F(X_{d_{i,G}}) \\ X_{d_{i,G}} & otherwise \end{cases} \quad (9)$$

其中$F(X_{d_{i,G}})$為參數向量之適應值，$F(u_{i,G+1})$為重組向量之適應值。

四、雙演化演算法

粒子群演算法與差分演算法有一共通之處，就是兩者的搜尋都是採取隨機搜尋的方式，有時容易導致落入區域最佳解，而為了預防落入區域最佳解，所以過去學者採取了一些方法改良此兩種演算法。2007 年 Xu[18]等人提出以差分演化演算法為主，粒子群演算法為輔的演算方式，命名為 DE-SI 架構，是以在差分演算法的架構中加入粒子群演算法的速度更新式，以達到快速收斂的效果，此架構與 Storn 及 Price 在 1996[7]年提出的差分演化演算法的演算方式類似，主要是利用合併演算法[14]來改良粒子群的速度更新公式，希望藉由改良能快速的找到最佳解。Zhang 及 Xie[46]在 2003 年提出以粒子群演算法為主，差分演化演算法為輔的機制，命名為 DEPSO，此演算法是在粒子群演算法的架構中加入差分演化演算法的加入交配的步驟。Omran[19]等人在 2007 年提出以粒子群演化演算法為主，差分演化演算法為輔的機制，此方式是在粒子群演算法中加入差分演化演算法的三個主要機制，分別為突變、合併、選擇，來進行演算，其目的是讓粒子群演算法擁有跳脫機制，藉由此三個機制可以避免落入區域最佳解，可以解決粒子群演算法會有過早收斂的缺點[20]。

本研究之雙演化演算法是與上述兩種方式截然不同的方法，由上述可知兩種方法不外乎都以

合併為主，可做個簡單的整理，如下：

1. DE(差分演化演算法)為主，PSO(粒子群演算法)為輔[19]
2. PSO(粒子群演算法)為主，DE(差分演化演算法)為輔 [18]

而雙演化演算法是以兩者同時進行，此方式於 2009 年吳盈志[20]提出，兩者並重，並分享彼此粒子的資訊，把最好的結果在迭代中利用交換分享方式給另一演算法，在利用此結果繼續執行演算法，直到滿足條件為止。此方法能同時擁有差分演化演算法的突變優勢，使得求解時不易若入區域最佳解，也擁有粒子群演算法的搜尋能力，使得在區域搜尋上能更精確。

本研究利用 ANSYS 執行結構分析，結合雙演化演算法其進行結構最佳化設計，其執行流程如圖 1 所示。

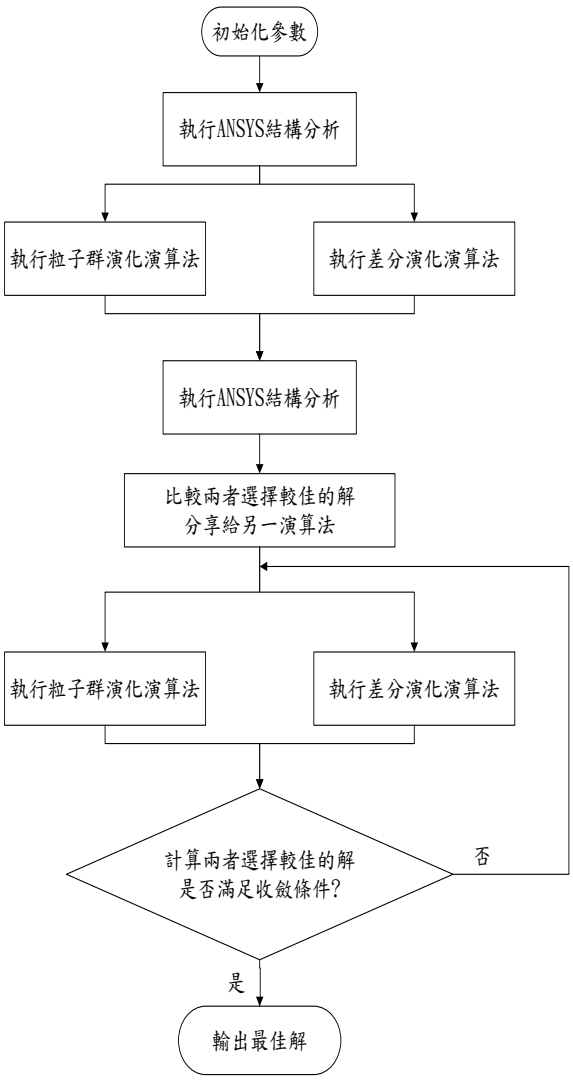

圖 1 雙演化執行流程

五、數值分析

範例一：十桿件桁架結構輕量化設計

本範例為雙演化演算法應用前的測試案例，十桿件桁架結構外型及尺寸如圖 2 所示。桁架上方負載 P_1 為 $50{,}000\,lb$，下方負載 P_2 為 $150{,}000\,lb$。結構之材料性質為楊式係數 $E = 10^7\,psi$、材料密度 $\rho = 0.1\,lb/in^3$ 和蒲松比 $v = 0.3$。其限制條件為節點 2 在 y 方向之位移量 u_{2y} 不超過 2.0 in 之要求。設計參數為各桿件之截面積 A_i，其上下限分別為 $26.0\ in^2$和 $0.1\ in^2$ 以及桿各件最大應力值不可超過 $25\ ksi$。本範例選擇 Link1 為其在 ANSYS 有限元素分析軟體內的元素。

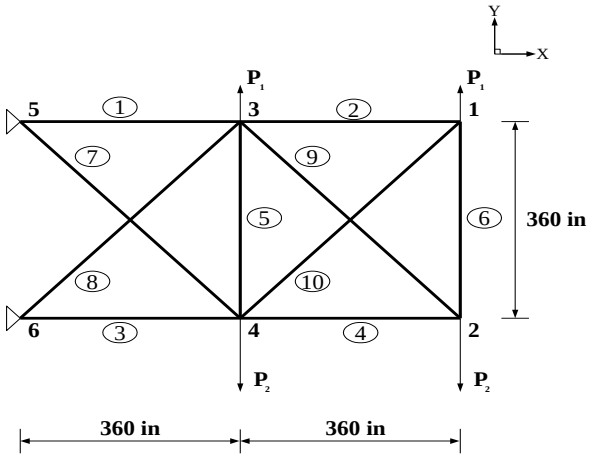

圖 2 範例一 十桿件桁架結構尺寸圖

表 1 範例一 最佳值比較

	初始值	文獻[21]	本研究
$A_1(in^2)$	10.0	23.370	24.578
$A_2(in^2)$	10.0	0.183	0.100
$A_3(in^2)$	10.0	25.740	23.981
$A_4(in^2)$	10.0	14.380	14.957
$A_5(in^2)$	10.0	0.100	0.100
$A_6(in^2)$	10.0	1.973	1.041
$A_7(in^2)$	10.0	12.240	4.160
$A_8(in^2)$	10.0	12.950	16.101
$A_9(in^2)$	10.0	19.980	19.787
$A_{10}(in^2)$	10.0	0.157	0.100
Weight(lb)	4196.47	4675.64	4375.26
$u_{2y} \le 2.0$ in	3.940	2.000	1.998
stress	20.463	24.964	24.989

本範例之初始設計變數 A_i為 $10\ in^2$。運用雙演化演算法運算後可得到結構最佳值之重量由原先的 4196.468 lb 提升到 4375.255 lb，而節點 2 在 y 方向的位移量 u_{2y}由 3.940 in 下降到 1.988 in，其桿件最大應力值也在 25 ksi 以內，皆滿足限制條件之要求，如表 1 所示。本研究最佳化後整體結構之總重量也較文獻[21]為輕。

範例二：無人飛行載具機翼主樑輕量化設計

本範例為無人飛行載具機翼主樑結構輕量化設計，其外型尺寸如圖 3 所示。水平桿件長度為 $L_1 = 3.5\ m$，兩桿距離為 $L_2 = 0.5\ m$，其材料特性為楊氏系數 $E = 70\ GPa$、材料密度 $\rho = 2700\ Kg/m^3$ 和蒲松比$v = 0.33$，並沿著 x 軸施加均佈力 $q = 112\ N/m$。本研究期望尋求無人飛行載具機翼結構斜桿放置之最佳位置距離 X 以滿足限制條件要求並達到輕量化之目標。主樑與斜桿截面為空心圓管，水平桿外徑為D_1內徑為d_1，及斜桿外徑為D_2內徑為d_2。其限制條件為水平端點在 Y 方向之位移量u_{tip}不超過18 cm，以及主樑結構兩空心管壁面厚度不得小於2 mm，並限制桿件最大應力不得超過 189 MPa，此為70%之降伏應力。

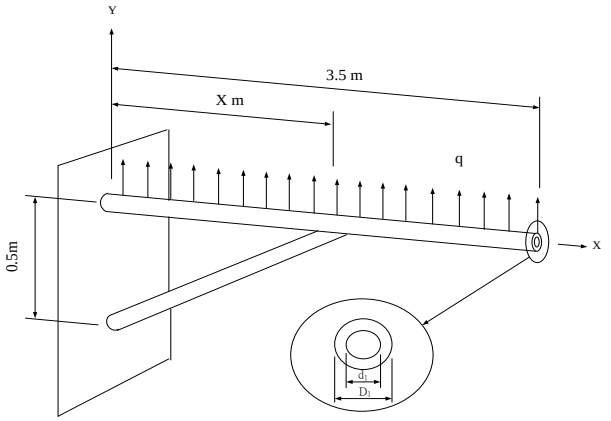

圖 3 範例二 無人飛行載具主樑結構外型圖

本範例初始設計參數為水平桿外徑 $D_1 = 5.0\ cm$、水平桿內徑$d_1 = 4.0\ cm$、斜桿外徑 $D_2 = 5.0\ cm$、斜桿內徑$d_2 = 4.0\ cm$和X $= 200\ cm$。初始結構之總重量10608 g，位移為 0.9 cm，最大應力173 MPa。經過雙演化演算法最佳化運算後重量下降至 3074 g，位移為 17.92 cm 和桿件最大應力值為 112.20 MPa，皆滿足限制條件之要求，如表 2 所示。結果顯示，本研究雙演化演算法所得到的最佳值較文獻[22]為佳。

表 2 範例二 最佳值比較

	初始	文獻[22]	本研究
水平桿外徑(cm)	5.00	5.00	2.97
水平桿內徑(cm)	4.00	4.60	2.27
斜桿外徑(cm)	5.00	2.40	1.10
斜桿內徑(cm)	4.00	2.00	0.38
距離 X(cm)	200	54.50	161.94
Wight(g)	10608	3557	3074
$u_{tip} \leq 18$ cm	0.90	17.90	17.92
$\|\sigma\| \leq 189$MPa	173.00	77.00	112.20

範例三：無人飛行載具主樑承受扭矩輕量化設計

本範例為考慮無人飛行載具機翼控制面操作時，機翼若承受扭矩過大會使機翼變型以致破壞空氣動力特性，因此本範例為機翼主樑承受扭矩之輕量化設計，其外型尺寸如圖 4 所示。機翼全長為 $L = 3.5\ m$，翼肋寬度0.5 m，並在翼肋後端施加一均佈力$q = 8.4\ N/m$，結構之材料性質為楊氏係數$E = 70\ GPa$、材料密度 $\rho = 2700\ Kg/m^3$ 和蒲松比 $v = 0.33$。其限制條件為翼肋端點u_{tip}位移不能超過5.0 cm，並限制桿件最大應力不得超過 189 MPa，此為 70%之降伏應力。

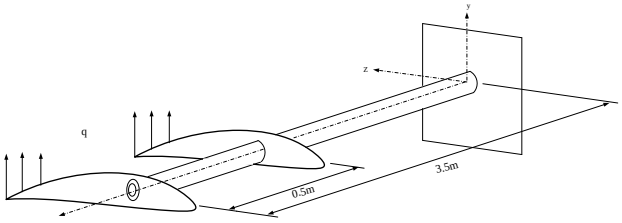

圖 4 受扭矩之無人飛行載具主樑結構外型圖

初始設計參數為圓管外徑$D = 5.0\ cm$和圓管內徑$d = 3.0\ cm$，初始結構之總重量為5065 g，位移為5.5 cm，最大應力為23 MPa。本研究利用雙演化演算法將重量降至3086 g，位移為5.0 cm，桿件最大應力值為27.71 MPa，皆滿足限制條件之要求如表 3 所示。結果顯示本研究之最佳值比文獻[21]最佳值較佳。

表 3 範例三 最佳值比較

	初始值	文獻[21]	本研究
圓管外徑(cm)	5.00	4.70	5.22
圓管內徑(cm)	3.00	4.10	4.81
Wight(g)	5065	3970	3086
$u_{tip} \leq 5.0$ cm	5.50	4.90	5.00
$\|\sigma\| \leq 189$MPa	23.00	23.00	27.71

範例四：太陽能無人飛機之結構輕量化設計

本範例為無人飛機之結構輕量化設計，其結構外型如圖 5 所示。機身與機翼主樑之材料為碳纖維。主翼與水平尾翼與垂直尾翼為巴爾沙木材料。其太陽能板以及電子儀器的重量利用塊狀質量的方式分佈在機翼內。本研究在機翼主樑上施加一均佈力$q = 38.86\ N/m$，以代表飛行時所產生的升力。其限制條件為機翼主樑端點位移$u \le 1.0\ cm$ 以及限制桿件最大應力σ不得超過$49\ MPa$。

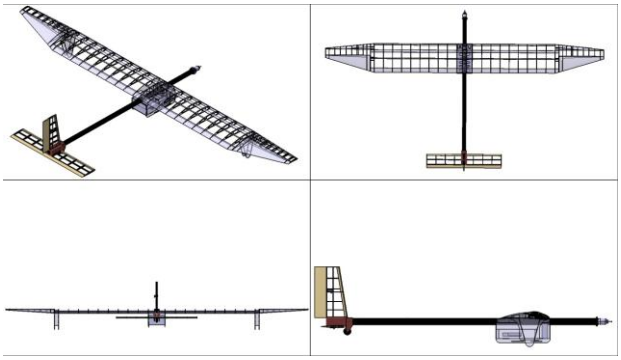

圖 5 太陽能動力飛機「鷺鷥」之結構設計圖

本研究將太陽能無人飛機於 ANSYS 中建模並對其做靜態分析。吾人初始設計參數給予主樑外徑$D_1 = 3.2\ cm$和主樑內徑$d_1 = 3.0\ cm$。初始結構重量為$6450\ g$，經過雙演化演算法最佳化運算後重量下降至$6294g$，顯示雙演化演算法應用在太陽能無人飛行載具上使的整體重量減輕了 2.4%，而初始主樑端點的位移為 0.389cm，最佳化運算後將位移提升為 1.00cm，並未違反限制條件之要求，如表 4 所示。結果顯示應用雙演化演算法比單獨使用 PSO 有較佳的結果，並能達到太陽能無人飛機結構輕量化之目標。

表 4 範例四 最佳值之比較

	初始值	PSO	本研究
主樑外徑(cm)	3.200	2.000	2.021
主樑內徑(cm)	3.000	1.790	1.821
端點位移 $u\le$ 1.0cm	0.389	0.996	1.000
$\lvert\sigma\rvert \le 49MPa$	7.747	15.348	16.343
Weight(g)	6450	6303	6294

範例五：太陽能無人飛機自然振動頻率最大化

本範例為太陽能無人飛機之自然振動頻率最大化設計。其目標為最大化結構第一模態之自然振動頻率f_1，其結構外型如圖 5 所示。初始第一模態之自然振動頻率為$69.724Hz$。結構之總重量為$6.45\ kg$，限制條件為增加之總質量 M 不得超過$64.5\ g \cdot s^2/m$(約為原始總質量之 1%)，本研究將太陽能無人飛行載具結構中主翼部分之節點塊狀集中質量(lumped mass)作為設計變數。

本範例將太陽能無人飛機於 ANSYS 中建模並對其作動態分析，經過雙演化演算法做最佳化運算之後，總質量增加了 $10.998\ g \cdot s^2/m$，而第一模態之自然振動頻率提高為$87.121Hz$。表 5 為初始值與最佳值之比較，結構增加之總質量滿足限制條件之要求，而第一模態之自然振動頻率也為之提高。範例二證明了雙演化演算法應用於太陽能無人飛機結構上，有良好的成效。

表 5 範例五 最佳值之比較

	初始值	PSO	DE	本研究
f_1(Hz)	69.724	75.384	71.533	87.121
M ≤ 64.50	0.000	30.217	30.565	10.998

六、結論

本研究成功地應用雙演化演算法於無人飛行載具結構之最佳化設計。研究中將雙演化演算法利用 FORTRAN 語法並結合 ANSYS 中的 APDL 語法成為一系統程式，使其能自動執行最佳化與 ANSYS 分析之流程。本研究所使用的雙演化演算法，是結合粒子群演算法及差分演化演算法，優點在於利用兩者不同的搜尋方式，不只可避免落入區域最佳解，而得到全域最佳解。此兩種不同的演算法參數設定簡單，因此在迭代的過程中可節省不少計算時間。本文中針對粒子群演算法提出變速因子的改良機制，藉由判斷粒子的區域最佳解與全域最佳解的距離來改變搜尋的步伐，而步伐的大小參考慣性權重的數值，分別設定為 0.8、0.5 和 0.3，依照距離遠近來做變化，因此能得到更準確的解。在差分演化演算法中，本文選擇使用三種隨機值組合的突變方式，目的是讓突變向量不會受到最佳解的影響，使其隨機搜尋，增加解的多樣性，可彌補粒子群演算法之不足。本研究所採取之雙演化演算法乃同時執行兩種演算法並將兩者的解求出並做分享，優點在於可以互相補足缺點，利用差分演化演算法使解能跳脫區域最佳解，而利用粒子群演算法能針對局部搜尋更佳完善，兩者互相分享資訊，能更快的求得全域

最佳解。由範例可以得知使用雙演化法求出的解比單一演算法求出的解較佳，顯示了應用雙演化演算法在無人飛行載具結構最佳化上是有成效的。

七、誌謝

本研究承蒙行政院國家科學委員會贊助，計畫編號 NSC 101-2221-E-032-008，特此致謝。

參考文獻

[1] Eberhart, R.C., and Kennedy, J., "Particle swarm optimization," IEEE *international conference on neural networks*, Vol. 4, pp. 1942–1948, 1995.

[2] Eberhart, R.C., and Kennedy, J., "A New Optimizer Using Particle Swarm Theory," *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, IEEE Service Center, pp. 39-43, 1995.

[3] Eberhart, R.C. and Shi, Y., "A modified particle swarm optimizer," *Proceedings of IEEE International Conference on Evolutionary Computation*, 1998.

[4] 郭信川，張建仁，劉清祥，「粒子群演算法於最佳化問題之研究」，*第一屆台灣作業研究學會學術研討會暨 2004 年科技與管理學術研討會*，419~432 頁，2004。

[5] Clerc, M., "The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization," *Proceedings of the Congress on Evolutionary Computation*, Vol. 3, pp. 1951−1957, 1999.

[6] He, S., Wu, Q.H., Wen, J.Y., Saunders, J.R. and Paton, R.C., "A particle swarm optimizerwith passive congregation," *Biosystem*, Vol. 78, pp. 135–47, 2004.

[7] Storn, R. and Price, K., "Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous spaces," *Global Optimization*, pp. 341-359, 1996.

[8]Kicinger, R., Arciszewski, T. and Jong, K.D., "Evolutionary computation and structural design: A survey of the state-of-the-art," *Computers & Structures*, Vol. 83, No. 23-24, pp. 1943-1978, 2005.

[9]Salman, A., Engelbrecht, A.P. and Omran, M.G.H., "Empirical analysis of self-adaptive differential evolution," *European Journal of Operational Research*, Vol. 183, No. 2, pp. 785-804, 2007.

[10] 李維平，江長育，搭配擾動策略之差分演化演算法，中原大學資管研究所論文，2011。

[11]Kicinger, R., Arciszewski, T. and Jong, K.D., "Evolutionary computation and structural design: A survey of the state-of-the-art," *Computers & Structures*, Vol. 83, No. 23-24, pp. 1943-1978, 2005.

[12] 莊玟珊，PSO–SA 混合搜尋法與其他結構最佳化設計之應用，國立中央大學土木工程研究所碩士論文，2007。

[13] Eberhart, R.C. and Shi, Y., "Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization," *In Proceedings of the 2000 Congress on Evolutionary Computation*, Vol. 1,pp.84−88, 2000.

[14]Hao, Z.F., Guo, G.H. and Huang, H., "A Particle Swarm Optimization Algorithm with Differential Evolution," *In Proceedings of the Sixth International Conference on Machine Learning and Cybernetics*,, pp. 1031-1035, 2007.

[15]Abbass, H.A., "The Self-adaptive Pareto Differential Evolution Algorithm. in Evolutionary Computation," *In Proceedings of the 2002 IEEE Congress on Evolutionary Computation*, 2002.

[16] Deng, C., Zhao, B., Deng, A. and Hu, R., "New Differential Evolution Algorithm with a Second Enhanced Mutation Operator," *In Proceedings of the International Workshop on Intelligent Systems and Applications*, pp. 1-4, 2009.

[17] Gong, W., Cai, Z. and Jiang, L., "Enhancing the performance of differential evolution using orthogonal design method," *Applied Mathematics and Computation*, Vol. 206, No. 1, pp. 56-69, 2008.

[18]Xu, R., Venayagamoorthy, G. K. and Wunsch, D. C., "Modeling of gene regulatory networks with hybrid differential evolution and particle swarm optimization," ESCI on *Neural Networks* Vol. 20, pp. 917-927, 2007.

[19]Omran, M. G. H., Engelbrecht, A. P. and Salman, A., "Differential Evolution Based Particle Swarm Optimization," *In Proceedings of the 2007 IEEE on Swarm Intelligence Symposium*, pp. 112–119, 2007.

[20]吳盈志，雙演化演算法之研究，中原大學資訊管理研究所，2009。

[21]劉敬文，結合基因演算法與線性規劃法於結構最佳化設計，淡江大學航空太空工程學系研究所，2010。

[22]王興正，應用粒子群演算法於無人飛行載具結構系統之最佳化設計，淡江大學航空太空工程學系研究所，2012。