1. **重力搜尋演算法(Gravitational Search Algorithm, GSA)**

重力搜尋演算法是一種可以用來解決最佳化問題的演算法，將問題等化成物理上的重

力，質量間彼此會互相吸引，產生力量，並藉彼此吸引的程度找出最佳解。

**| 實驗設計步驟**

1. 產生原始族群

隨機產生50組搜尋範圍內的原始族群所在位置，這裡的原始族群即為物體質量。

1. 計算適應度

將50組原始族群分別計算其累加誤差平方和的值，經過平均後將其取平方根，再由適應函數計算求出各質量適應度。



1. 求出各質量所佔比例(mass\_Calculation.m)

使用式(3.1)求出個別質量，再利用式(3.2)求出每個質量所佔的比例，其概念類似GA演算法的複製率。若是要找出最大質量，則Best為適應度最大的，worst則為適應度最小的。

|  |  |
| --- | --- |
|  | (3.1) |

|  |  |
| --- | --- |
|  | (3.2) |

1. 算出吸引力(Gfield.m)

將彼此間的質量與距離代入改良式重力公式，如式(3.3)，即可求得彼此間的吸引力。

其中，G(t)值大小會隨時間改變，變化方式如式(3.4)。F取最好的值數量本實驗則是設定不會隨著時間減少而減少，如式(3.5)。Ma為影響別人的強度(Active gravitational mass)，Mp為被影響的強度(Passive gravitational mass)，而Mi為自己的質量，Mii為第i個族群的慣性質量，本實驗將這些質量都設定為自己的質量，即Mi = Mp = Ma = Mii。R為兩個族群間的歐幾里得距離。Ԑ=8.85x10-12

|  |  |
| --- | --- |
|  | (3.3) |

|  |  |
| --- | --- |
|  | (3.4) |

|  |  |
| --- | --- |
|  | (3.5) |

1. 算出加速度並更新速度及位置

由式(3.6)即可算出加速度，並藉由式(3.7)及式(3.8)分別更新速度及位置。

|  |  |
| --- | --- |
|  | (3.6) |

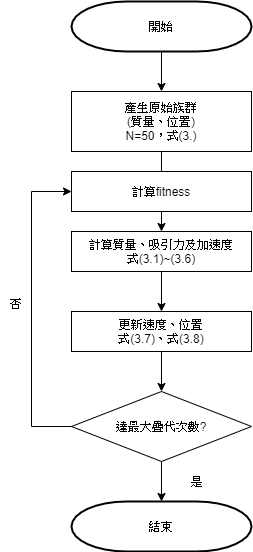
|  |  |
| --- | --- |
|  | (3.7) |

|  |  |
| --- | --- |
|  | (3.8) |

1. 搜尋範圍限定(space\_bound.m)

依照題目制定的搜尋範圍，避免族群在進行GSA演算法的過程中更新位置時，新一代的族群超出上下限。

1. 演算法流程圖



圖三 GSA演算法程式流程圖

1. 參數設定

N = 50; 群體個數

iter = 50; 遞迴次數

alpha = 20; 重力常數之參數

G0 = 100; 重力常數之初值

**實驗與模擬結果**



圖 1 、第一組初始值之搜尋結果



圖 2 、第一組初始值之實際追蹤軌跡(y)與參考值(y𝑑)軌跡



圖 3 第一組初始值之追蹤誤差(e)



圖 4 第一組初始值之力量輸入(u)



圖 5 第二組初始值之搜尋結果



圖 6 第二組初始值之實際追蹤軌跡(y)與參考值(y𝑑)軌跡



圖 7 第二組初始值之追蹤誤差(e)



圖 8 第二組初始值之力量輸入(u)

****

圖 9 第三組初始值之搜尋結果

****

圖 10 第三組初始值之實際追蹤軌跡(y)與參考值(y𝑑)軌跡

****

圖 11 第三組初始值之追蹤誤差(e)

****

圖 12 第三組初始值之力量輸入(u)

Q3. Please explain why the fitness function, all the parameters and settings that are chosen.

Matlab Code