Particle Swarm Excel VBA User Guide

Author: Ashley Marshall

Table of Contents

1. Introduction

This piece of software employs the Particle Swarm Optimisation (PSO) technique to solve continuous minimisation optimisation problems. It is capable of solving optimisation problems with multiple variables and gives the user freedom to control various parameters to tune the algorithm.

Section 2 provides an overview of the workbook, Section 3 provides an overview of the function of each subroutine and Section 4 explains how to use the programme. The Appendix contains all the code contained within the programme.

2. Overview of Workbook

2.1. Home

The Home worksheet is where the user can tune the algorithm, input boundaries for variables and insert an objective function for the programme to solve. The user is also able to run the various subroutines from this sheet by clicking the buttons on the left-hand side.

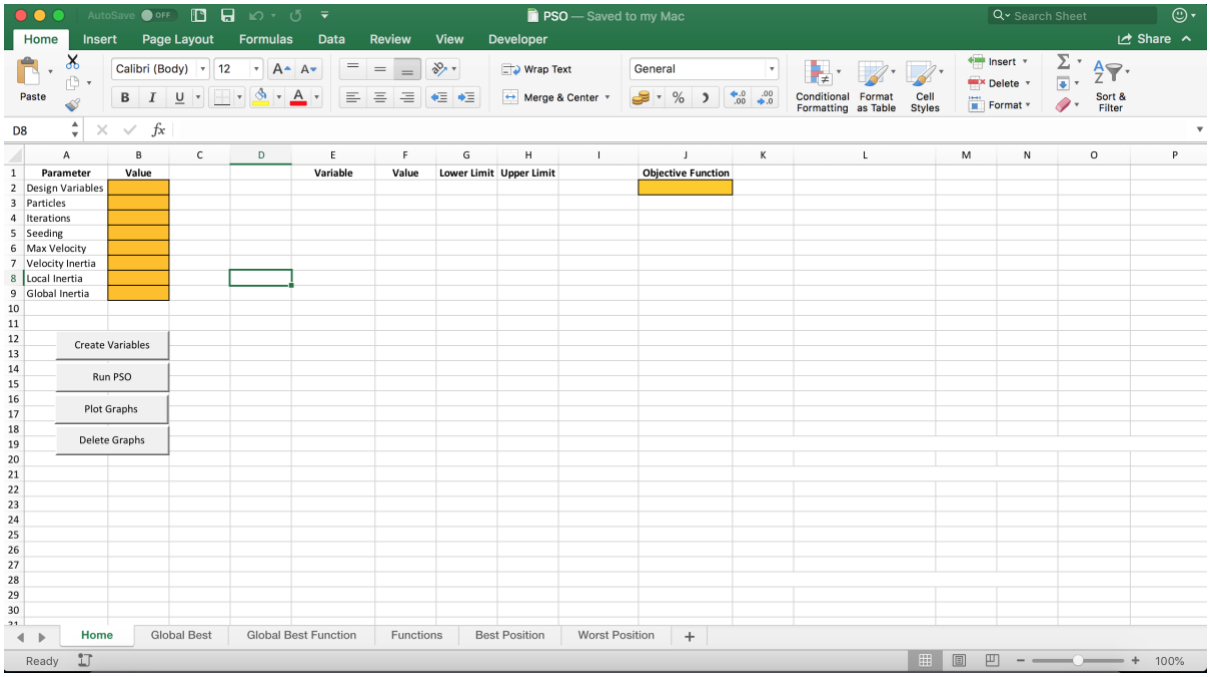

Figure 2.1: Home worksheet

2.2. Global Best

The global best for each variable is written to this worksheet every iteration. Column A corresponds to variable one, column B corresponds to variable two, and so on.

2.3. Global Best Function

The best objective function values from each iteration are written to this worksheet.

2.4. Functions

Once the programme is run, the Functions worksheet is where the worst and best function values, for each iteration, are stored. Column A corresponds to the worst value and column B corresponds to the best value.

2.5. Best Position

The best position from each iteration is written here. Column A corresponds to variable one, column b refers to variable two, and so on.

2.6. Worst Position

The worst position from each iteration is written here. Column A corresponds to variable one, column b refers to variable two, and so on.

3. Subroutines

The programme contains four subroutines, stored within a single module named PSO, which can be executed by clicking the buttons on the left-hand side of the Home worksheet. The module contains the declarations *Option Explicit* and *Option Base 1.* The code contained within the PSO module can be found in Appendix A.

3.1. PSO_Initialise

Executed by clicking the *Create Variables* button. The function of this subroutine is to prime the Home worksheet, by creating the number of variables, specified by the user in cell B2.

3.2. PSO_Run

Executed by clicking the *Run PSO* button. This subroutine performs the PSO. It reads user parameters from the left-hand side of the Home worksheet and the limits for each variable. While executing, this subroutine writes current variable values onto the Home worksheet and reads the function value, from cell J2, that results from the current set of variables. Upon completion, the subroutine writes the optimum values for each variable into their respective cell in column F, which then displays the final solution value in cell J2.

3.3. PSO_Graphs

Executed by clicking the *Plot Graphs* button. The function of this subroutine is to generate graphs. Its execution results in five graphs being created in their own worksheet. The first is called *Function Chart* and it plots the two function values against iterations. The second is called *Global Best Chart* and it plots the global best values against number against iterations. The third is called *Best Positions* Chart and it plots the best positions against iterations. The fourth is called *Worst Positions Chart* and it plots the worst positions against iterations. The fifth is *Global Best Variables Chart* and it plots the global best values against iterations. Once the graphs have been generated, this subroutine cannot be executed again until the graphs have been deleted, using the Delete_Graphs subroutine.

3.4. Delete_Graphs

Executed by clicking the *Delete Graphs* button. The function of this subroutine is to delete the generated graphs, thus allowing new ones to be created. This subroutine can only be executed if the previously mentioned graphs have been created.

4. Using the Programme

4.1. Initialising

Enter the number of variables contained within the problem into cell B2 and click the *Create Variables* button. Enter the lower and upper limits for each variable into column G and H, respectively. Enter the objective function to be minimised into cell J2, with reference to the variable values contained in column F. Table 4.1 explains which user parameters to insert into the worksheet.

Table 4.1: List of user parameters

Cell	Parameter
B2	Number of design variables contained within the problem
B3	Number of particles to be used by the PSO algorithm
B4	Number of iterations to be completed by the PSO algorithm
B5	Enter the character, Y, if seeding is desired
B6	The maximum initial velocity to be used by the PSO algorithm
B7	The inertia weighting of the velocity component in the velocity calculation
B8	The inertia weighting of the local best component in the velocity calculation
B9	The inertia weighting of the global best component in the velocity calculation

4.2. Running the Algorithm

Once initialised, the PSO can be executed by clicking the *Run PSO* button on the Home worksheet.

4.3. Viewing Results

To view the results click the *Plot Graphs* button on the Home worksheet.

If graphs have already been plotted and the algorithm has since been run, click the *Delete Graphs* button before clicking the *Plot Graphs* button.

Appendix: PSO Module

```
'----- PSO -----

'Version History:
   '21/02/2018 - Module complete

'Author: Ashley Marshall

Option Explicit
Option Base 1
Sub PSO_Initialise()

Dim variables As Double
Dim i As Double

variables = Worksheets("Home").Cells(2, 2)

Range("E2:H100").Delete
Worksheets("Home").Cells(2, 10).Delete

For i = 1 To variables
   Worksheets("Home").Cells((i + 1), 5) = "x" & i
   Worksheets("Home").Cells((i + 1), 6).Interior.ColorIndex = 43
   Worksheets("Home").Cells((i + 1), 7).Interior.ColorIndex = 37
   Worksheets("Home").Cells((i + 1), 8).Interior.ColorIndex = 37
   Worksheets("Home").Cells(2, 10).Interior.ColorIndex = 44
   Range("J2").Borders.LineStyle = xlContinuous
   Range("E2:H" & (i + 1)).Borders.LineStyle = xlContinuous
Next i

End Sub
Sub PSO_Run()

'----- Variables -----
Dim variables As Double
Dim swarm_size As Double
Dim iterations As Double
Dim maxvel As Double
Dim diff As Double
Dim min_val As Double
Dim max_val As Double
Dim i_min As Double
Dim i_max As Double
Dim position() As Double
Dim velocity() As Double
Dim local_best() As Double
Dim global_best() As Double
Dim fn_local_best() As Double
Dim fn_position() As Double
Dim xmax() As Double
Dim xmin() As Double
Dim global_best_fn As Double
```

```
Dim w, c1, c2 As Double
Dim i, j, k, L, n As Double

'----- Set Values -----
variables = Worksheets("Home").Cells(2, 2)
iterations = Worksheets("Home").Cells(4, 2)
swarm_size = Worksheets("Home").Cells(3, 2)
maxvel = Worksheets("Home").Cells(6, 2)
w = Worksheets("Home").Cells(7, 2)
c1 = Worksheets("Home").Cells(8, 2)
c2 = Worksheets("Home").Cells(9, 2)

'----- Resize Arrays -----
ReDim position(variables, swarm_size)
ReDim velocity(variables, swarm_size)
ReDim local_best(variables, swarm_size)
ReDim global_best(variables)
ReDim fn_local_best(swarm_size)
ReDim fn_position(swarm_size)
ReDim xmax(variables)
ReDim xmin(variables)

'----- Run -----
If Worksheets("Home").Cells(5, 2) = "Y" Then
   Randomize (1)
End If

Worksheets("Global Best").Range("A1").CurrentRegion.Delete
Worksheets("Functions").Range("A1").CurrentRegion.Delete
Worksheets("Global Best Function").Range("A1").CurrentRegion.Delete
Worksheets("Best Position").Range("A1").CurrentRegion.Delete
Worksheets("Worst Position").Range("A1").CurrentRegion.Delete
Worksheets("Home").Range("F2:F" & (variables + 1)).ClearContents

For j = 1 To variables
   xmax(j) = Worksheets("Home").Cells((j + 1), 8)
   xmin(j) = Worksheets("Home").Cells((j + 1), 7)
Next j

For n = 1 To iterations

   If n = 1 Then

      For i = 1 To swarm_size
         For j = 1 To variables
            velocity(j, i) = (Rnd() * 2 * maxvel) - maxvel
            position(j, i) = xmin(j) + (Rnd() * (xmax(j) - xmin(j)))
            Worksheets("Home").Cells((j + 1), 6) = position(j, i)
            local_best(j, i) = position(j, i)
         Next j
         fn_position(i) = Worksheets("Home").Cells(2, 10)
         fn_local_best(i) = fn_position(i)
         global_best_fn = Application.Min(fn_local_best)
         If i = 1 Then
            global_best_fn = fn_local_best(i)
```

```
            For j = 1 To variables
               global_best(j) = local_best(j, i)
            Next j
         Else
            If fn_local_best(i) < global_best_fn Then
               global_best_fn = fn_local_best(i)
               For j = 1 To variables
                  global_best(j) = local_best(j, i)
               Next j
            End If
         End If
      Next i

   Else

      For i = 1 To swarm_size
         For j = 1 To variables
            velocity(j, i) = (w * velocity(j, i)) + (c1 * Rnd() * (local_best(j, i) - position(j, i))) + (c2
* Rnd() * (global_best(j) - position(j, i)))
            position(j, i) = position(j, i) + velocity(j, i)
            If position(j, i) > xmax(j) Then
               position(j, i) = xmax(j)
            End If
            If position(j, i) < xmin(j) Then
               position(j, i) = xmin(j)
            End If
            Worksheets("Home").Cells((j + 1), 6) = position(j, i)
         Next j
         fn_position(i) = Worksheets("Home").Cells(2, 10)
      Next i

      For i = 1 To swarm_size
         If fn_position(i) < fn_local_best(i) Then
            For j = 1 To variables
               local_best(j, i) = position(j, i)
            Next j
            fn_local_best(i) = fn_position(i)
         End If
         If fn_local_best(i) < global_best_fn Then
            global_best_fn = fn_local_best(i)
            For j = 1 To variables
               global_best(j) = local_best(j, i)
            Next j
         End If
      Next i

   End If

   For i = 1 To swarm_size
      If i = 1 Then
         min_val = fn_position(i)
         i_min = i
         max_val = fn_position(i)
         i_max = i
      Else
```

```
            If fn_position(i) < min_val Then
                i_min = i
                min_val = fn_position(i)
            End If
            If fn_position(i) > max_val Then
                i_max = 1
                max_val = fn_position(i)
            End If
        End If
    Next i

    If n = iterations Then
        For j = 1 To variables
            Worksheets("Home").Cells((j + 1), 6) = global_best(j)
        Next j
    End If

    Worksheets("Functions").Cells(n, 1) = max_val
    Worksheets("Functions").Cells(n, 2) = min_val
    Worksheets("Global Best Function").Cells(n, 1) = global_best_fn

    For j = 1 To variables
        Worksheets("Global Best").Cells(n, j) = global_best(j)
        Worksheets("Best Position").Cells(n, j) = position(j, i_min)
        Worksheets("Worst Position").Cells(n, j) = position(j, i_max)
    Next j

Next n

End Sub
Sub PSO_Graphs()

Dim i As Double
Dim variables As Double

variables = Worksheets("Home").Cells(2, 2)
Dim global_best As Chart
Set global_best = Charts.Add
global_best.SetSourceData Source:=Worksheets("Global
Best").Range("A1").CurrentRegion, PlotBy:=xlColumns
global_best.ChartType = xlLine
For i = 1 To variables
    global_best.SeriesCollection(i).Name = "Variable " & i
Next i
ActiveSheet.Name = "Global Best Variables Chart"

Dim functions As Chart
Set functions = Charts.Add
functions.SetSourceData Source:=Worksheets("Functions").Range("A1").CurrentRegion,
PlotBy:=xlColumns
functions.ChartType = xlLine
functions.SeriesCollection(1).Name = "Worst"
functions.SeriesCollection(2).Name = "Best"
ActiveSheet.Name = "Function Chart"
```

```
Dim global_best_fn As Chart
Set global_best_fn = Charts.Add
global_best_fn.SetSourceData Source:=Worksheets("Global Best
Function").Range("A1").CurrentRegion, PlotBy:=xlColumns
global_best_fn.ChartType = xlLine
global_best_fn.Legend.Delete
ActiveSheet.Name = "Global Best Function Chart"

Dim best_positions As Chart
Set best_positions = Charts.Add
best_positions.SetSourceData Source:=Worksheets("Best
Position").Range("A1").CurrentRegion, PlotBy:=xlColumns
best_positions.ChartType = xlLine
For i = 1 To variables
   best_positions.SeriesCollection(i).Name = "Variable " & i
Next i
ActiveSheet.Name = "Best Position Chart"

Dim worst_positions As Chart
Set worst_positions = Charts.Add
worst_positions.SetSourceData Source:=Worksheets("Worst
Position").Range("A1").CurrentRegion, PlotBy:=xlColumns
worst_positions.ChartType = xlLine
For i = 1 To variables
   worst_positions.SeriesCollection(i).Name = "Variable " & i
Next i
ActiveSheet.Name = "Worst Position Chart"

End Sub
Sub Delete_Graphs()

Dim wks As Chart

For Each wks In Charts
   If wks.Name = "Global Best Variables Chart" Or wks.Name = "Function Chart" Or
wks.Name = "Global Best Function Chart" Or wks.Name = "Best Position Chart" Or
wks.Name = "Worst Position Chart" Then
      wks.Activate
      Application.DisplayAlerts = False
      wks.Delete
      Application.DisplayAlerts = True
   End If
Next wks

End Sub
```