Maintenance Manual

Anthony Chapman

2014

The source code of the PSO and portfolio optimisation (together with an example asset file) could be found in the 'source' folder in project submission.

Haskell Requirements

You will need the latest version of Haskell and it can be downloaded from *http://www.haskell.org/platform/*. *Notepad++* would be my preferred Windows editor and can be installed for free from *http://notepad-plus-plus.org/download/v6.6.2.html*

Instructions on how to compile and run the algorithm:

Makes sure that the source Haskell files (portOpt.hs, PSO.hs) are located in the same folder as the asset data files you wish to use for the portfolio.
To compile and run the code on a Windows environment:

1. Open a command line and move to the folder containing the source files, ie the Haskell code.
2. Type the following command into the command line and press enter:

   % ghc –make portOpt.hs

3. You have now created an executable file, called 'portOpt.exe'
4. Open a windows folder and move to the folder containing the source files, double click on 'portOpt.exe'
5. The program will stat in a new window.

The User Manual provides a guide for using the program.
To compile and run the source code on a Unix/Linux environment:

1. Open a terminal and move to the folder containing the source files, ie the Haskell code.
2. Type the following command into the terminal and press enter:

   % ghc -O2 –make portOpt.hs -threaded -rtsopts

3. You have now created an executable file, now type into the terminal the following and press enter:

   % ./portOpt

4. The program will start in the same terminal window.

The User Manual provides a guide for using the program.

Organisation of System Files, including directory structures

```
chapman_anthony/
    executables/ - folder containing executable files for portfolio optimisation
        portOptWin.exe
        portOptUnix
        assetTest.txt - an example asset file
    source/ - folder containing source code files for portfolio optimisation
        portOpt.hs
        PSO.hs
    readMe.txt
    MaintananceManual
```

Space and Memory Requirements

2 GB of RAM, 100 MB of space would be recommended.

List of source code files, with a summary of their role

File	Description
portOpt.hs	Portfolio optimisation implementation using PSO
portOptUnix	Unix executable file
portOptWin.exe	Windows executable file
PSO.hs	Main PSO optimisation module
assetTest.txt	Example asset file

Program Flow

This section provides a general description of the program flow from a technical side.

Initialisation

Once the input parameters are defined, the body of the function is created. Firstly, we use a function *initialize* to randomly create the list of initial particles. It needs to create the number of particles the user specified at the beginning of the program randomly distributed inside the search space defined by the boundings, in our case it is the the variable *weightBounds* and it is currently set to a minimum of 0.05 and a maximum of 0.35, this represents the minimum and maximum percentage (5% to 35%) every asset in a portfolio has to be obtained. The algorithm needs adjustment velocity parameters which are given by *wpg1*, these are used to control the particles' new velocities once they are updated.

Optimisation

Once the particles are initialised, the function *pso'* deals with the iterations

of the algorithm, the number of iterations was defined at the beginning of the program by the user. Note that each particle contains its position, its velocity, the best solution found by the particle, and the best solution found by any particle. *pso'* runs recursively on the number of iterations, if there are no more iterations to be performed then it just returns best particle found. Otherwise, it applies one iteration step (by using an auxiliary function *oneStep*) and then it recursively call itself with one iteration less.

After each step, each particle is updated using a function *updateParticle*, it applied the new velocity equation described in Chapter 2. *updateParticle* also checks if the fitness of the new position is better than the personal and global best found so far.

Changing Parameters

Changing WPG Parameter
These acceleration coefficients are defined at the beginning of portOpt.hs:

```
wpg1 :: (Double,Double,Double)
wpg1 = (-0.16,1.89,2.12)
```

Changing Penalty Parameter Note: changing penalty parameter, *penPara*, will automatically change the penalty value, *penVal*.

```
--Penalty parameter
penPara = 0.1
--Penalty value
penVal = 1/penPara
```

Changing the Boundaries

```
--Weight bound is set to this to induce diversification
weightBounds = replicate nAssets (0.05,0.35)
```

Changing the Output File Name Here, *time t* gives the date as a string and then that string is concatenated to the end of the another string *"output"*.

```
appendFile ("output-" ++ (time t))
```