```
// MultiCharts - Raising the Trading Standard                                    http://www.multicharts.com
//  Custom Optimization Criteria: Wrapping it up; Combining Custom Criteria and exporting for further analysis.
//  Source: http://www.multicharts.com/discussion/viewtopic.php?f=5&t=8838

/* Choose the objective function, the performance metric on which you want to perform the Genetic Optimization, below.
    Give the variable 'myObjectiveFunction' a number corresponding with your choice:
    1: Sunny Harris' CPC Index;
    2: Robert Pardo's Pessimistic Return on Margin (PROM);
    3: Robert Pardo's Required Capital;
    4: Pessimistic Return On Required Capital (PRORC);
    5: Michael Harris' Probability Rule.

    For example, if you'd like to optimize on the CPC Index, give variable 'myObjectiveFunction' a value of 1.
*/
var myObjectiveFunction = 1;

/* Settings start here; change these to your preferences */
    /*  Change the variable 'myFileLocation' to the location to where you want to write your file.
        Note: Before running a new optimization, you'll need to remove, rename or delete this file.
        Else it will get overwritten and/or will the header of the text file be unsuitable for importing in Excel.
    */
var myFileLocation = "c:\\Temp\\CustomCriteria_Output.txt";
var AccountSize = 100000;
var SafetyFactor = 3.00;            // The SafetyFactor times the MaxDrawDown gives a buffer for the Required Capital.
var CapitalStopLoss = 0.40;         // The percentage loss after which you'll stop trading the strategy (0.40 = 40%).
var TargetedProfitFactor = 2.00;    // The Profit Factor the trader want to achieve.
var CommissionPenalty = 0.60;       // Penalty used to calculate if the Profit Factor is achieved; enter a one (1) here to disable ↵
 this.

/*
                This is the end of the settings
                        The code below can be left untouched
*/

// Strategy performance variables
var avgWinTrade         = StrategyPerformance.AvgWinningTrade;
var avgLoseTrade        = StrategyPerformance.AvgLosingTrade;
```

```
37 var percentProfitable   = StrategyPerformance.PercentProfitable;
38 var profitFactor        = StrategyPerformance.GrossProfit / Math.abs(StrategyPerformance.GrossLoss);
39 var winLossRatio        = StrategyPerformance.AvgWinningTrade / Math.abs(StrategyPerformance.AvgLosingTrade);
40 var numWinningTrades    = StrategyPerformance.WinningTrades;
41 var numLosingTrades     = StrategyPerformance.LosingTrades;
42 var maxDrawdown         = StrategyPerformance.MaxStrategyDrawDown;
43 var grossProfit         = StrategyPerformance.GrossProfit;
44 var grossLoss           = StrategyPerformance.GrossLoss;
45 var netProfit           = StrategyPerformance.NetProfit;
46 var numOfTrades         = StrategyPerformance.TotalTrades;
47 var avgTrade            = StrategyPerformance.AvgTrade;
48 var maxConsWins         = StrategyPerformance.MaxConsecWinners;
49 var maxConsLoss         = StrategyPerformance.MaxConsecLosers;
50 var avgBarWinner        = StrategyPerformance.AvgBarsInWinningTrades;
51 var avgBarLoser         = StrategyPerformance.AvgBarsInLosingTrades;
52 var returnOnAcc         = StrategyPerformance.ReturnOnAccount;
53 var returnOnAccSize     = (netProfit / AccountSize) * 100;              // Percentage profit on the AccountSize provided by the
    user
54 var maxDDPerc           = (maxDrawdown / AccountSize) * 100;            // The MaxDrawDown in percentage
55
56 // Objective functions variables
57 var cpcIndex, prom, reqCapital, prorc, reqProfitability, headerString, outputString, sep;
58
59 var sqrtWins            = Math.sqrt(numWinningTrades);
60 var sqrtLosses          = Math.sqrt(numLosingTrades);
61 var pessResult          = (avgWinTrade * (numWinningTrades - sqrtWins)) - (Math.abs(avgLoseTrade) * (numLosingTrades +
    sqrtLosses));
62
63 sep = ";";          // The seperator between values exported in the text file; change this if you have trouble importing the data
    with your Locale/Region settings
64 var decimals = 4;       // Number of decimals to export
65 var curDecimals = 2;    // Number of decimals for currency values
66
67 /* The calculation of the Custom Criteria starts below */
68
69 // Sunny Harris' CPC Index
70 if (avgWinTrade == 0 || avgLoseTrade == 0) {
```

```
    cpcIndex = 0;
}
else {
    cpcIndex = (percentProfitable / 100) * winLossRatio * profitFactor;
}


// Robert Pardo's Pessimistic Return on Margin (PROM)
if (AccountSize < 1 || numWinningTrades == 0 || numLosingTrades == 0) {
    prom = 0;
}
else {
    prom = (pessResult / AccountSize) * 100;
}


// Robert Pardo's Required Capital
if (CapitalStopLoss < 0.01 || SafetyFactor < 0.01 || maxDrawdown == 0) {
    reqCapital = 0;
}
else {
    reqCapital = (Math.abs(maxDrawdown) * SafetyFactor) / CapitalStopLoss;
}


// Pessimistic Return On Required Capital (PRORC); inspired on Pardo's concepts
if (prom == 0 || reqCapital == 0) {
    prorc = 0;
}
else {
    prorc = (pessResult / reqCapital) * 100;
}

// Michael Harris' Probability Rule
if (TargetedProfitFactor < 0.01 || CommissionPenalty < 0 || CommissionPenalty > 1) {
    reqProfitability = 0;
}
```

```
108 else {
109     reqProfitability = percentProfitable - (TargetedProfitFactor / (TargetedProfitFactor + (CommissionPenalty * winLossRatio)) *
        100);
110 }
111
112 // Function to print the data to the file
113 function WriteToFile(stringToWrite) {
114     // Create a new FileSystemObject
115     fsObj = new ActiveXObject("Scripting.FileSystemObject");
116
117     // If the file doesn't exits, create it, write the header {and the first line}
118     if (!fsObj.FileExists(myFileLocation)) {
119
120         // Create the header string
121         headerString =
122             "CPC Index"             + sep +
123             "PROM"                  + sep +
124             "RequiredCapital"       + sep +
125             "PRORC"                 + sep +
126             "ProbabilityRule"       + sep +
127             "GrossProfit"           + sep +
128             "GrossLoss"             + sep +
129             "NetProfit"             + sep +
130             "ReturnOnAccount"       + sep +
131             "ReturnOnAccountSize"   + sep +
132             "PessimisticProfit"     + sep +
133             "TotalTrades"           + sep +
134             "PercentProfitable"     + sep +
135             "WinningTrades"         + sep +
136             "LosingTrades"          + sep +
137             "AvgTrade"              + sep +
138             "AvgWinningTrade"       + sep +
139             "AvgLosingTrade"        + sep +
140             "WinLossRatio"          + sep +
141             "ProfitFactor"          + sep +
142             "MaxStrategyDrawDown"   + sep +
143             "MaxDrawDownPercentage" + sep +
```

```
            "MaxConsecWinners"      + sep +
            "MaxConsecLosers"       + sep +
            "AvgBarsInWinningTrades"+ sep +
            "AvgBarsInLosingTrades"
            ;

        // Create the file
        fsObj.CreateTextFile(myFileLocation);

        // Open stream to write to it
        os = fsObj.GetFile(myFileLocation);
        os = os.OpenAsTextStream(2, 0);

        // Write data to it
        os.WriteLine(headerString);
        //os.WriteLine(stringToWrite);

        // Close the file
        os.Close();
    }
    else {
        os = fsObj.GetFile(myFileLocation);
        os = os.OpenAsTextStream(8, 0);

        os.WriteLine(stringToWrite);

        os.Close();
    }
}

// Create the string with strategy data
outputString =
    cpcIndex.toFixed(decimals)              + sep +
    prom.toFixed(decimals)                  + sep +
    reqCapital.toFixed(curDecimals)         + sep +
    prorc.toFixed(decimals)                 + sep +
    reqProfitability.toFixed(decimals)      + sep +
```

```
    grossProfit.toFixed(curDecimals)        + sep +
    grossLoss.toFixed(curDecimals)          + sep +
    netProfit.toFixed(curDecimals)          + sep +
    returnOnAcc.toFixed(decimals)           + sep +
    returnOnAccSize.toFixed(decimals)       + sep +
    pessResult.toFixed(curDecimals)         + sep +
    numOfTrades.toFixed(0)                  + sep +
    percentProfitable.toFixed(decimals)     + sep +
    numWinningTrades.toFixed(0)             + sep +
    numLosingTrades.toFixed(0)              + sep +
    avgTrade.toFixed(decimals)              + sep +
    avgWinTrade.toFixed(decimals)           + sep +
    avgLoseTrade.toFixed(decimals)          + sep +
    winLossRatio.toFixed(decimals)          + sep +
    profitFactor.toFixed(decimals)          + sep +
    maxDrawdown.toFixed(curDecimals)        + sep +
    maxDDPerc.toFixed(decimals)             + sep +
    maxConsWins.toFixed(0)                  + sep +
    maxConsLoss.toFixed(0)                  + sep +
    avgBarWinner.toFixed(decimals)          + sep +
    avgBarLoser.toFixed(decimals)
    ;

WriteToFile(outputString);

// Return the choosen objective function
if (myObjectiveFunction == 1) {
    return cpcIndex;
}
else if (myObjectiveFunction == 2) {
    return prom;
}
else if (myObjectiveFunction == 3) {
    return reqCapital;
}
else if (myObjectiveFunction == 4) {
    return prorc;
```

```
218 }
219 else if (myObjectiveFunction == 5) {
220     return reqProfitability;
221 }
222 else {      // makes sure that at least something gets returned
223     return netProfit;
224 }
```