CCD ALGORITHM
Application to EU Style Factors ERC or RISK PARITY portfolios In Excel VBA

This short note is a direct application of the Juan-Carlos Richard and Thierry Roncalli article available on ssrn. The cyclical coordinate descent (CCD) they describe is a fantastic tool in that you can solve many convex optimization problems without the need for an optimiser/solver, just using an iterative algorithm.

We apply here the cyclical coordinate descent (CCD) algorithm to compute the weight of the ERC portfolio in an Excel VBA framework.

Note: We make no distinction between Equally Weighted Risk Contribution (ERC) and Risk Parity portfolios. Both define here a portfolio where all its components have the same budget of risk.

CCD Algorithm

In short going from the optimisation program of a Risk Budgeted portfolio

From a mathematical point of view, a risk budgeting (or RB) portfolio is defined as follows (Roncalli, 2013):

$$\begin{cases} \mathcal{RC}_i(x) = b_i \mathcal{R}(x) \\ b_i > 0 \\ x_i > 0 \\ \sum_{i=1}^n b_i = 1 \\ \sum_{i=1}^n x_i = 1 \end{cases}$$

From differentiating the Lagrangian function they deduce an iterative algorithm to compute the RB weigths

$$x_i^\star = \frac{-(\Sigma x)_i + x_i\sigma_i^2 + \sqrt{((\Sigma x)_i - x_i\sigma_i^2)^2 + 4\sigma_i^2 b_i \sigma(x)}}{2\sigma_i^2}$$

Portfolio of EU Style Premia

We implement the **CCD** iterative algorithm in Excel VBA (see appendix for command lines).

We now have an efficient tools to compute easily the ERC portfolio in Excel.

Let s illustrate this with a portfolio aiming at investing in 5 Style Premia on EU stocks. : Low Beta, Momentum, Quality, Low Size, Value. This can be easily extended to

Below are the volatility and correlation matrix (weekly return, since 2011).
We would note that the Value style premia presents interesting diversification properties counterbalancing its higher volatility.

	Low Beta	Momentum	Quality	Low Size	Value
σ	0.79%	0.95%	1.03%	1.11%	1.65%

	Quality	Low Beta	Momentum	Low Size	Value
Low Beta	1.0	0.6	0.5	0.1	-0.4
Momentum	0.6	1.0	0.4	0.0	-0.4
Quality	0.5	0.4	1.0	0.0	-0.3
Low Size	0.1	0.0	0.0	1.0	0.6
Value	-0.4	-0.4	-0.3	0.6	1.0

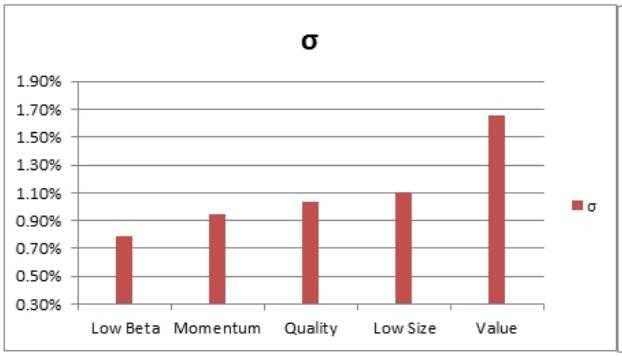
Excel VBA ERC_CCD() function

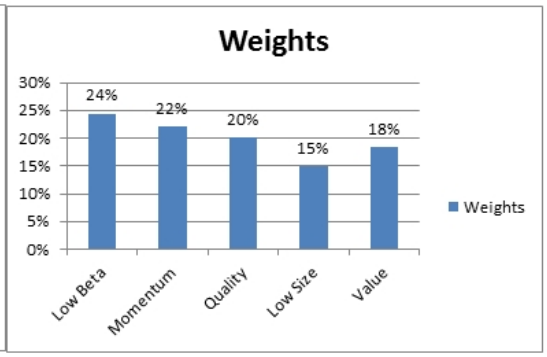
We can know compute the covariance matrix and the ERC portfolio using the ERC_CCD() function developed in VBA.

	ERC CCD Weights	Contrib. σi
Low Beta	=ERC_CCD(O6:S10)	0.11%
Momentum	22%	0.11%
Quality	20%	0.11%
Low Size	15%	0.11%
Value	18%	0.11%
Total	100%	0.56%
σPTF	0.56%	

Covariance Matrix

	Quality	Low Beta	Momentum	Low Size	Value
Low Beta	6.3E-05	4.6E-05	3.9E-05	8.9E-06	-4.5E-05
Momentu	4.6E-05	9.0E-05	4.1E-05	8.9E-07	-5.7E-05
Quality	3.9E-05	4.1E-05	1.1E-04	5.1E-07	-4.7E-05
Low Size	8.9E-06	8.9E-07	5.1E-07	1.2E-04	1.2E-04
Value	-4.5E-05	-5.7E-05	-4.7E-05	1.2E-04	2.7E-04





We'll note that despite a higher volatility the ERC portfolio would have a greater weight in Value vs. Low Size. This is due to the good diversification power of the Value factor.

Conclusion

We show here the command lines of the function to easily compute the ERC / Risk Parity portfolio in Excel.

We illustrate it by computing the ERC portfolio on 5 EU Equities Style Factors. This could be easily extended to any universe.

Appendix : Excel VBA command Lines

```
Function ERC_CDD(COV_Mat, Optional Ret_Typ = 1, Optional Opt_MaxIte = 500, Optional Opt_Precision = 0.0000001)

  'Conversion of a Range to a Matrix/Array
  If TypeName(COV_Mat) = "Range" Then COV_Mat = MatToArray(COV_Mat)
  nR = UBound(COV_Mat, 1)

  'Variables declaration
  Dim n, i
  Dim vol, mr, tr
  Dim x: ReDim x(1 To nR, 1 To 1)
  Dim vol_i2: ReDim vol_i2(1 To nR, 1 To 1)
  Dim Cov_i, L2_1oN_t

  For i = 1 To nR
    'Asset i volatility^2
    vol_i2(i, 1) = COV_Mat(i, i)
    'Initialisation x0
    x(i, 1) = 1 / Sqr(vol_i2(i, 1))
  Next i

  'Distance Portfolio RiskBudgets vs. the ERC target (RB = 1/N)
  L2_1oN_t = L2_1oN(w_TR(COV_Mat, x))
  n = 1
  While (n <= Opt_MaxIte) And Not (L2_1oN_t < Opt_Precision)
  Debug.Print n & " " & L2_1oN_t

    For i = 1 To nR
        vol = Application.WorksheetFunction.MMult(Application.WorksheetFunction.Transpose(x), Application.WorksheetFunction.MMult(COV_Mat, x))
        vol = Sqr(vol(1))

        Cov_i = Application.WorksheetFunction.MMult(COV_Mat, x)
        Cov_i = Cov_i(i, 1)

        x(i, 1) = 1 / (2 * vol_i2(i, 1)) * (-Cov_i + x(i, 1) * vol_i2(i, 1) + Sqr((Cov_i - x(i, 1) * vol_i2(i, 1)) ^ 2 + 4 * vol_i2(i, 1) * vol / nR))
    Next i
    L2_1oN_t = w_TR_L2_1oN(COV_Mat, x)
    n = n + 1
  Wend

  'Weights normalization so that sum(x)=1
  x_sum = Application.WorksheetFunction.Sum(x)
  For i = 1 To nR
    x(i, 1) = x(i, 1) / x_sum
  Next i

  'portfolio volatility and Risk Contribution
  vol = Application.WorksheetFunction.MMult(Application.WorksheetFunction.Transpose(x), Application.WorksheetFunction.MMult(COV_Mat, x))
  tr = w_TR(COV_Mat, x)

  'portfolio volatility and Risk Contribution
  If Ret_Typ = 1 Then ERC_CDD = x
  If Ret_Typ = 2 Then ERC_CDD = vol(1)
  If Ret_Typ = 3 Then ERC_CDD = tr

End Function
```

```
Function w_TR(MMat, x)
    'Compute the Risk Contribution (eg. Risk Budget) of a portfolio x knowing the CovVar matrix
    w_TR_0 = Application.WorksheetFunction.MMult(MMat, x)

    If TypeName(x) = "Range" Then x = x.Value
    For l = LBound(w_TR_0) To UBound(w_TR_0)
        w_TR_0(l, 1) = w_TR_0(l, 1) * x(l, 1)
    Next l

    'Normalized by 100
    w_TR_SUM = Application.WorksheetFunction.Sum(w_TR_0)
    For l = LBound(w_TR_0) To UBound(w_TR_0)
        w_TR_0(l, 1) = w_TR_0(l, 1) / w_TR_SUM
    Next l
    w_TR = w_TR_0
End Function
```

```
Function L2_1oN(x)
    L2_1oN = 0
    If TypeName(x) = "Range" Then x = x.Value
    nR = UBound(x)
    For Each c In x
        L2_1oN = L2_1oN + Abs(c - 1 / nR)
    Next c
End Function
```

```
Public Function MatToArray(R)
    Dim l As Integer
    Dim c
    If VarType(R) = vbString Then Set R = Range(R)

    If Not TypeName(R) = "Range" Then
        MatToArray = R
    Else
        Dim R_Mod: ReDim R_Mod(1 To R.Rows.Count, 1 To R.Columns.Count)
        l = 0
        For Each c In R
            R_Mod(c.Row - R.Row + 1, c.Column - R.Column + 1) = c.Value
            l = l + 1
        Next c
        MatToArray = R_Mod
    End If
End Function
```

Reference

Richard, Jean-Charles and Roncalli, Thierry, Smart Beta: Managing Diversification of Minimum Variance Portfolios (March 2015). Available at SSRN: https://ssrn.com/abstract=2595051