APPENDICES

Appendix 1: Elliott Wave Patterns

The following are the basic patterns described by the Elliott Wave Principle, reproduced with kind permission from Robert Prechter, Elliott Wave International.

Figure A1.1 – Horizontal triangles

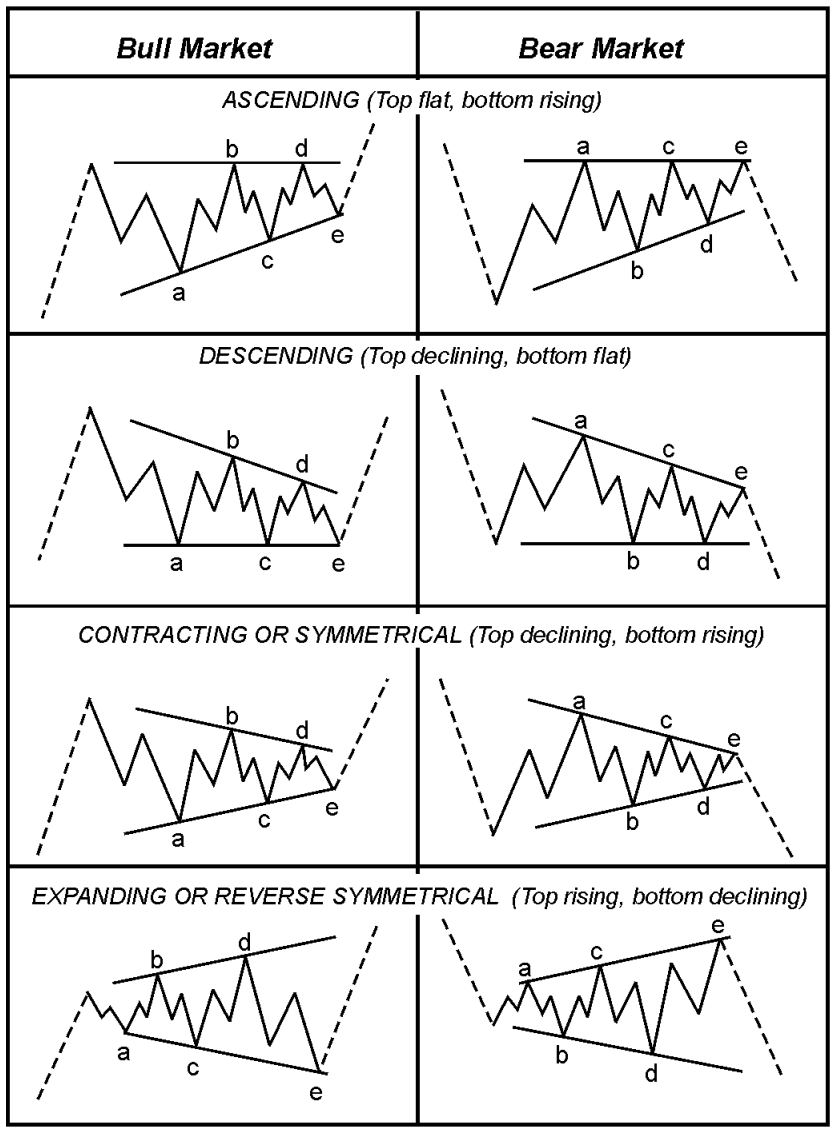


Figure A1.2 – Bull and bear triangles

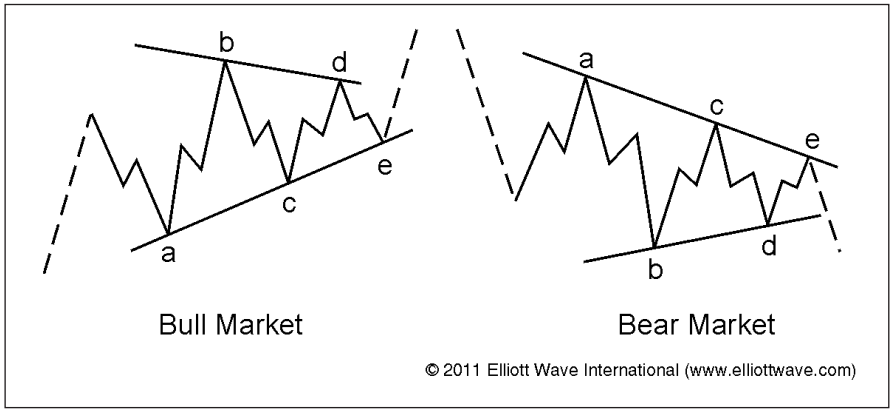


Figure A1.3 – Combination – type one correction

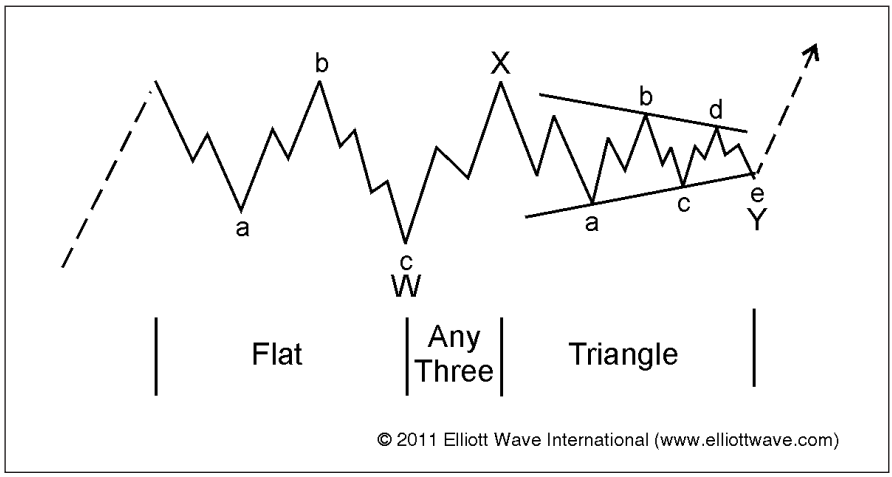


Figure A1.4 – Combination – type two correction

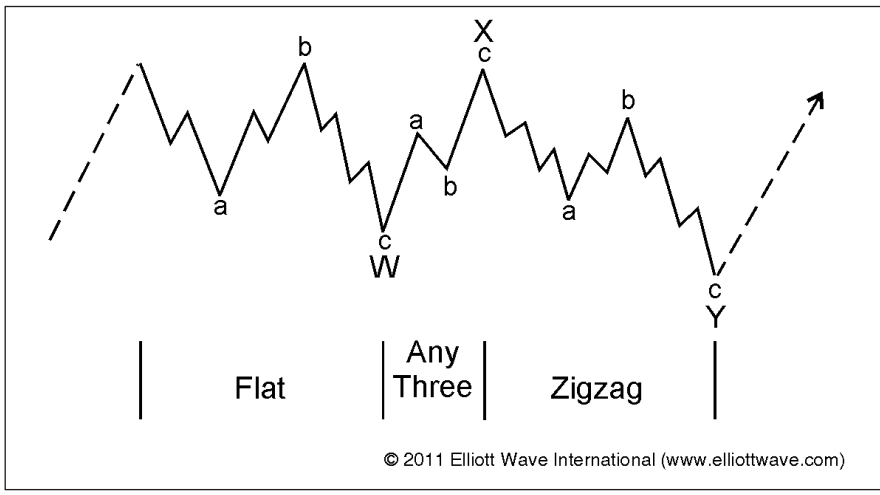


Figure A1.5 – Five-wave impulsive advance and channel

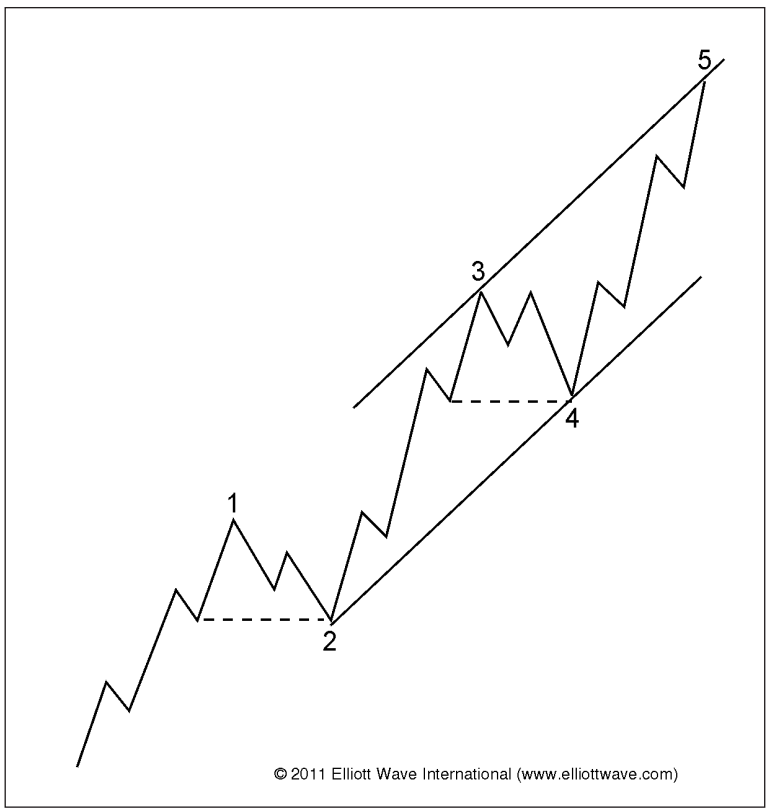


Figure A1.6 – 62% retracement (wave two)

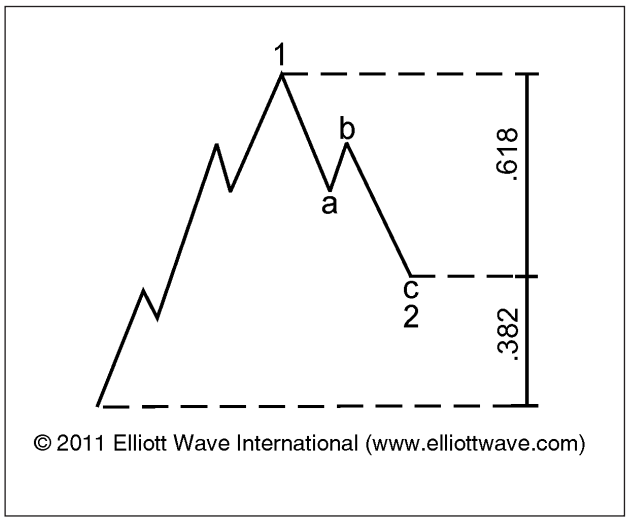


Figure A1.7 – 38% retracement (wave four)

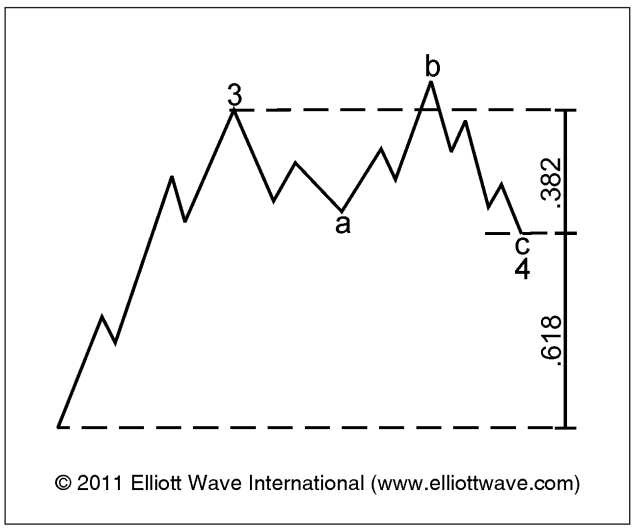


Figure A1.8 – First iteration – Motive and Corrective waves

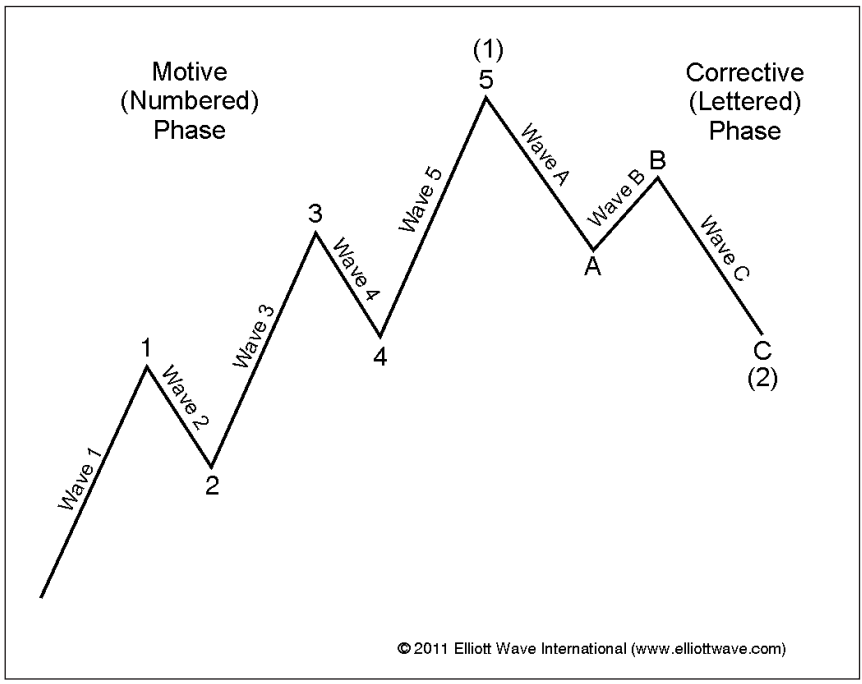


Figure A1.9 – Second iteration – Motive and Corrective waves further subdivided

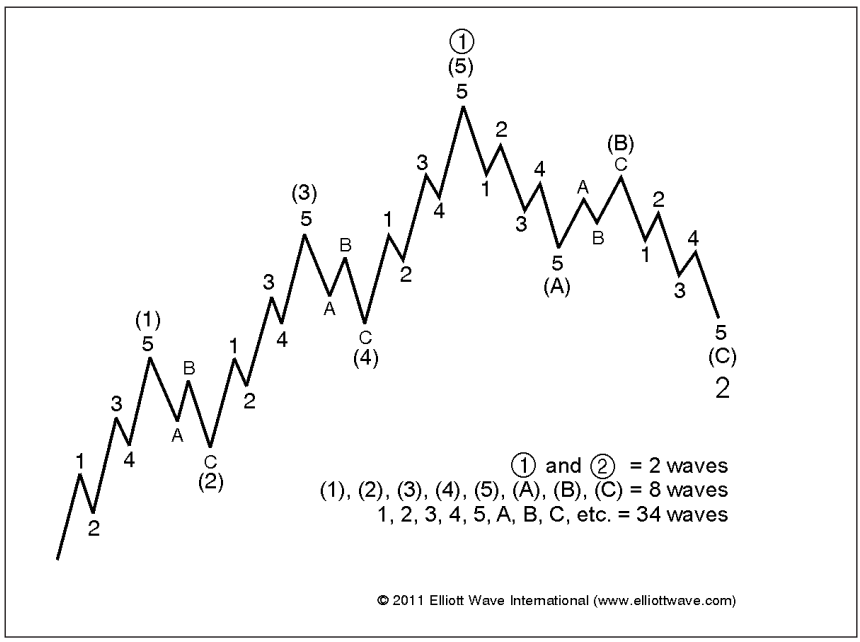


Figure A1.10 – Ending diagonal

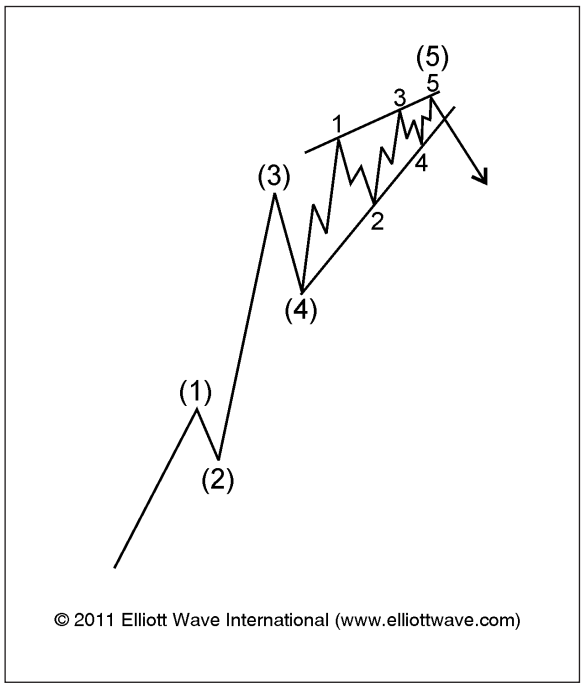


Figure A1.11 – Extensions

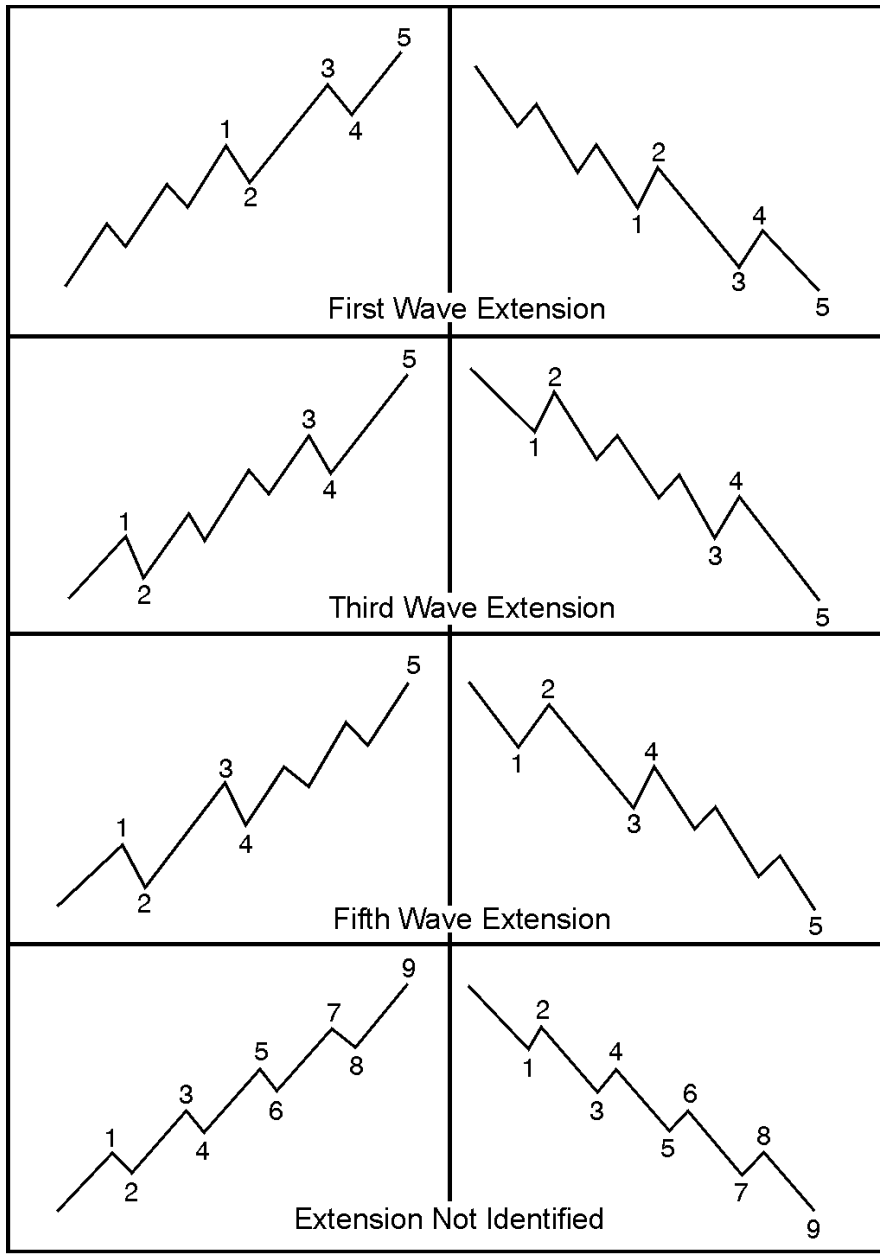


Figure A1.12 – Zigzag

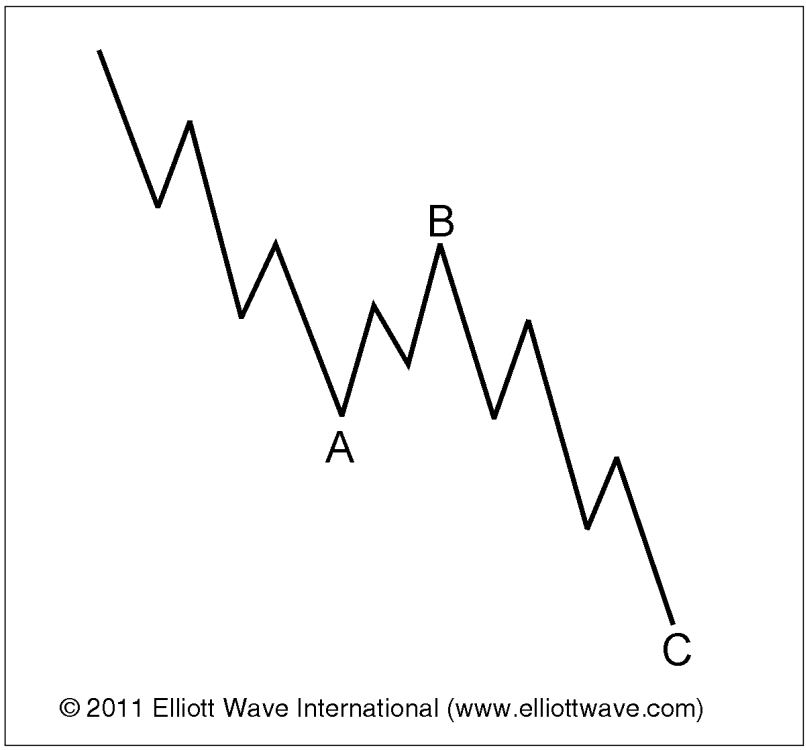


Figure A1.13 – Double zigzag

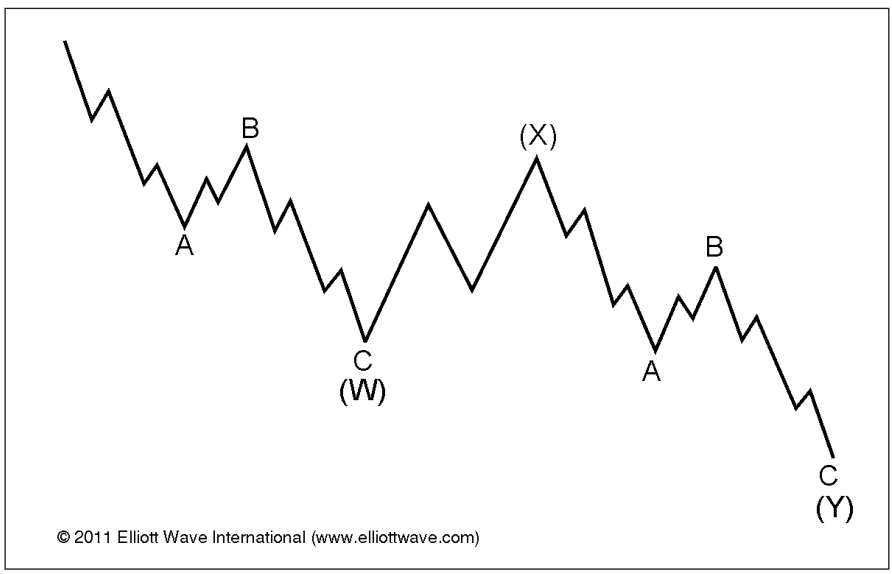


Figure A1.14 – Flat

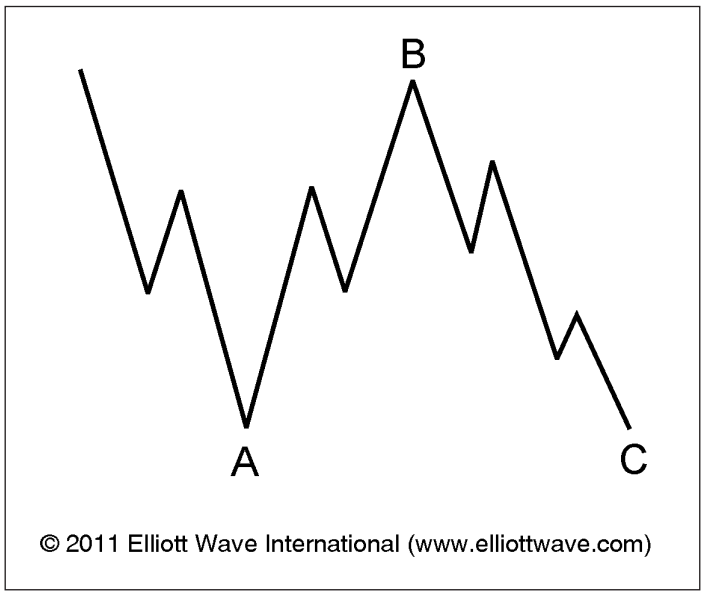
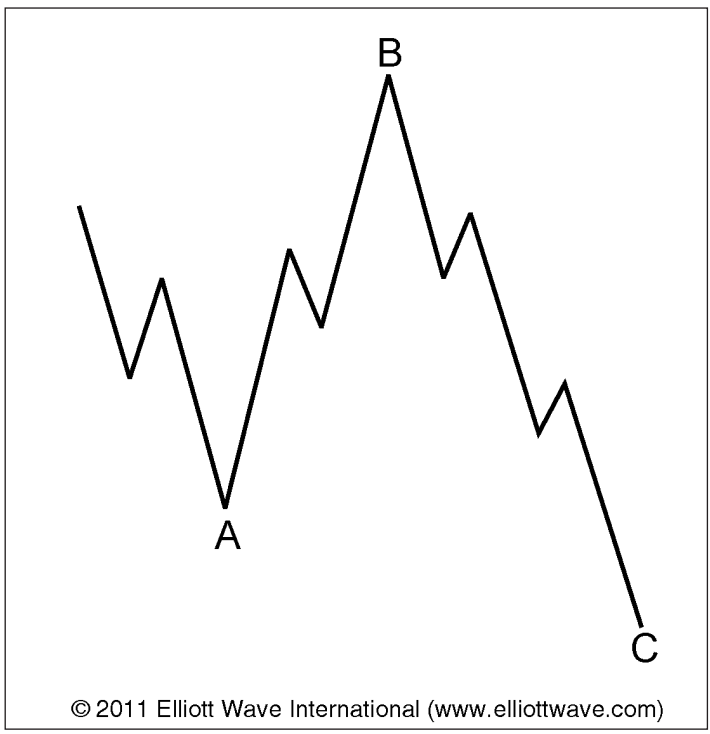


Figure A1.15 – Expanded flat



Appendix 2: FLD Code

The code for the FLD – as well as all of the code used in this book – is provided in Updata's proprietary language as well as TradeStation's Easy Language. For users of other trading platforms, you will either need to translate the code yourself or contact the development team at your vendor.

Updata

```
Programmer – Christopher Grafton
NAME FLD
PARAMETER #PERIOD=80
PARAMETER #SETBACK=20
PARAMETER #R=0
PARAMETER #G=0
PARAMETER #B=0

INDICATORTYPE TOOL
DISPLAYSTYLE  DASH
COLOUR RGB(#R,#G,#B)

@FLD=0

FOR #CURDATE=#PERIOD TO #LASTDATE+(#PERIOD/2)
@FLD=HIST((HIGH+LOW)/2,(#PERIOD/2)+1)

IF #CURDATE>#LASTDATE-#SETBACK

@PLOT=@FLD

ENDIF

NEXT
```

Notes

- You will need to run the code separately for each FLD you wish to display on the chart. The setback period as well as the actual cycle period and colour will need to be changed manually each time.
- #PERIOD refers to the nominal cycle period. Stick to the time units of the chart you are looking at. For example if you want to plot an 18-month FLD on a weekly chart, you need to express the FLD in weeks, thus #PERIOD=80. On the monthly chart it is just 18. If you want to plot the

80-week FLD on a daily chart, then you need to express weeks in trading days. To do this simply multiply by five (the number of trading days in a week): thus in this case 80W = 18M = 400 days.

- Remember: you need to use the average cycle periods actually derived from your phasing analysis or scans. If you just use the nominal periods (80W, 40W and so on), the FLDs will be out of kilter with the actual cycles on the chart. The Nominal Model in months, weeks and days for both calendar and trading time can be found in Chapter 1, Table 1.3.
- Also remember that FLDs on the daily chart always need to be expressed in trading days. If you express them in calendar days, the FLDs will be too long and will not do their job. The conversion again is: calendar days to trading days multiply by 0.7; trading days to calendar days divide by 0.7.
- #SETBACK refers to how much of the FLD in past time you wish to see. If you are plotting a historical FLD, then this value needs to be at least the same as the number of bars displayed on the chart. If you are doing current FLDs you may find you need to adjust between 0 and 20. It is a matter of personal preference. If the setback is too long, then the chart can look too busy. If the setback is too short, then it will convey too little information.
- #DISPLAYSTYLE THICK2 is the thickness of the FLD line. If this setting is too thick, then change it to #DISPLAYSTYLE LINE.
- #COLOUR: Updata uses the additive primary colour RGB system. You can choose whatever colours you like for FLDs and the full table of colours can be found at **www.tayloredmktg.com/rgb**. Remember though that your colour code needs to be uniform across all of your analysis components: FLDs, VTLs and phasing diamonds. For example, a 40-day (nominal) FLD – which is blue in the scheme used in this book – has to be the same colour as both the 40-day diamond and VTL. If you do not do this, you will get into a dreadful muddle. Finally, if you choose to use the colour scheme shown in this book (shown in Table A2.1) and prefer to display your charts with a black background, you will need to change the 80W to white RGB (255,255,255), otherwise it will not show up on your chart.
- The actual Updata file can be downloaded from: **vectisma.com/resources/downloads/fldcode**

Colour coding

Table A2.1 – RGB colour codes for cycles in the Nominal Model

Nom.			Colour	RGB code	
18Y			Brown	139,69,19	
9Y			Turqoiuse	0,134,139	
54M	230W		Grey	105,105,105	
18M	80W		Black	0,0,0	
9M	40W		Purple	148,0,211	
	20W		Olive	107,142,35	
	10W	80D	Red	255,0,0	
		40D	Blue	0,0,255	
		20D	Green	0,100,0	
		10D	Orange	255,165,0	
		5D	Light Blue	0,191,255	

TradeStation

```
/// Program Name - FLD ///

/// Programmer - Mark Cotton ///

/// Website - www.7GTradingTools.com ///

/// Skype - markcottons.skype ///

/// Email - markcottons.email@gmail.com ///


inputs: int PERIOD( 80 ) ;
inputs: int SETBACK( 0 ) ;
inputs: int COLOR( RGB( 241, 31, 15 ) ) ;
inputs: int THICKNESS( 0 { 0 to 6 } ) ;


variables: bool vFIRSTBAR( true ) ;
variables: int vTEXTID( 0 ), double vPREVLASTBARDATE(
JuliantoDate( LastCalcJDate ) ) ;
variables: int vCOUNT( 0 ), int vb( 0 ) ;
if vFIRSTBAR = true then

  begin

    while vPREVLASTBARDATE = JuliantoDate(
LastCalcJDate )

      begin

        vCOUNT = vCOUNT + 1 ;

        vTEXTID = Text_New( JuliantoDate( LastCalcJDate
- vCOUNT ), time, high + ( high - low ), " " ) ;

        Text_SetStyle( vTEXTID, 1, 1 ) ;

        vPREVLASTBARDATE = Text_GetDate( vTEXTID ) ;

      end ;
```

```
    vFIRSTBAR = false ;

  end ;


if GetAppInfo( aiSpaceToRight ) >= ( ( PERIOD * 0.5 ) +
1 ) then

  begin

    if SETBACK > 0 then

      begin

        if LastBarOnChart = true then

          begin

            for vb = 0 to ( SETBACK + ( ( PERIOD * 0.5
) + 1 ) ) - 1

              begin

                  plot1[( -( PERIOD * 0.5 ) + 1 ) +
vb]( ( high[vb] + low[vb] ) * 0.5, "FLD1", COLOR,
default, THICKNESS ) ;

              end ;

          end ;

      end

    else

      begin

        plot2[-( PERIOD * 0.5 ) + 1]( ( high + low ) *
0.5, "FLD2", COLOR, default, THICKNESS ) ;

      end ;

  end
```

```
else

  begin

    value1 = Text_SetString( vTEXTID, “Set Spaces to
the Right to “ + numtostr( ( ( PERIOD * 0.5 ) + 1 ), 0
) + “ in Format Window” ) ;

  end ;
```

Appendix 3: Inverse Moving Average Code

Updata

```
Programmer — Christopher Grafton
NAME INVERSE MOVING AVERAGE
PARAMETER @PERIOD=20

INDICATORTYPE CHART
DISPLAYSTYLE HISTOGRAM

@SMA=0
@CLOSE=0
@DETREND=0

FOR #CURDATE=@PERIOD TO #LASTDATE
@SMA=HIST(MAVE(@PERIOD),(-@PERIOD/2))
@CLOSE=CLOSE

@DETREND=@CLOSE-@SMA

IF @DETREND>0
  COLOUR RGB(0,0,255)
ENDIF

IF @DETREND<0
  COLOUR RGB(255,0,0)
ENDIF

@PLOT=@DETREND
NEXT

FOR #CURDATE=(#LASTDATE-@PERIOD/2)+1 TO #LASTDATE
@PLOT=-10000
NEXT
```

Notes

- This indicator, also called a Detrender, is displayed as a histogram in the lower window of the chart. When the close is above the moving average, the histogram bars are coloured blue; and when the close is below the moving average, the bars are coloured red.

- The closing price has been used, although you could equally use any other price level: for example High or Low. In which case, simply change the line: @CLOSE=CLOSE to @CLOSE=HIGH or @CLOSE=LOW.
- Similarly, the moving average does not need to be a simple moving average. If you wish to use an exponential moving average, for example, replace MAVE with EAVE in the line @SMA=HIST(MAVE(@PERIOD),(-@PERIOD/2))
- The Parameter @PERIOD is a manual input and the choice of moving average period is yours.
- The Updata code can be downloaded from: **vectisma.com/downloads/inversema**

Appendix 4: Cycle Envelope Code

Updata

```
Programmer – Christopher Grafton
NAME CYCLE ENVELOPE
PARAMETER @PERIOD=20
PARAMETER @ADJUST=1.5
PARAMETER #R=0
PARAMETER #G=0
PARAMETER #B=0

INDICATORTYPE TOOL
DISPLAYSTYLE 3LINES
PLOTSTYLE LINE RGB(#R,#G,#B)
PLOTSTYLE2 LINE RGB(#R,#G,#B)
PLOTSTYLE3 DASH RGB(#R,#G,#B)

@SMA=0
@UPPER=0
@LOWER=0
@HIGH=0
@LOW=0
@MID=0
@MID1=0
@MID2=0
@RANGE=@PERIOD/2
@AVG=0
#SUM=0
#ADD=0
@BAND=0
@DEV=0
@DEN=0

FOR #CURDATE=@PERIOD TO (#LASTDATE-@PERIOD/2)
@DEV=MAX(@DEV,(HIGH(0)-LOW(0)))
@DEN=@DEN+1
NEXT
@BAND=@DEV/@ADJUST
FOR #CURDATE=@PERIOD TO (#LASTDATE-@PERIOD/2)
@SMA=HIST(MAVE(@PERIOD),(-@PERIOD/2))
@UPPER=@SMA+@BAND
@LOWER=@SMA-@BAND
@RANGE=(@PERIOD/2)
@PLOT=@UPPER
@PLOT2=@LOWER
@PLOT3=@SMA
```

```
NEXT

FOR #CURDATE=(#LASTDATE-@PERIOD/2)+1 TO (#LASTDATE-1)
@PLOT=-10000
@PLOT2=-10000
@PLOT3=-10000
NEXT

#CURDATE=#LASTDATE
@MID=MAVE(@RANGE)
@MID1=MAVE(@RANGE)+@BAND
@MID2=MAVE(@RANGE)-@BAND
DRAWLINESTYLE DASH
DRAWCOLOUR RGB(#R,#G,#B)
DRAWLINE @RANGE,@UPPER,0,@MID1
DRAWLINE @RANGE,@LOWER,0,@MID2
DRAWLINE @RANGE,@SMA,0,@MID
```

Notes

This tool produces centred moving average bands, but with straight lines drawn from the last plot of each line (centred MA, upper band and lower band) to the simple moving average value at the end of the data, thereby plotting in an approximation for the final lagging range.

- #PERIOD refers to the moving average period. This needs to be between one-quarter and a half of the cycle around which you wish to draw the envelope. For example, a 40-day cycle will need an envelope period of between 10 and 20 days. Half is standard, but sometimes this may be too smooth. If the envelope period is the same length as the cycle, the envelope will not pick out the fluctuation of the cycle of interest.
- Any number of envelopes can be drawn and they can be differentiated by choosing different colours. Remember, you need to input the appropriate RGB number in each case, for example red would be #R=255, #G=0, #B=0.
- #ADJUST: this automatically adjusts the width of the envelope to enclose most of the price action. The default setting is 1.5, however sometimes you will find that the bands are too narrow. In this case, decrease the number. If the bands are too wide, then increase the number. Often it is a case of experimenting to get the best fit.
- The Updata code can be downloaded from: **vectisma.com/downloads/envelope**

TradeStation

```
/// Program Name - Cycle Envelope ///
/// Programmer - Mark Cotton ///
/// Website - www.7GTradingTools.com ///
/// Skype - markcottons.skype ///
/// Email - markcottons.email@gmail.com ///


inputs: int PERIOD( 20 ) ;
inputs: int CHANNELWIDTH( 30 ) ;
inputs: int COLOR( RGB( 241, 31, 15 ) ) ;
inputs: int COLORPRO( RGB( 20, 241, 15 ) ) ;


variables: double vDEV( 0 ), int vb( 0 ) ;
variables: double vSMA( 0 ), double vUPPER( 0 ), double
vLOWER( 0 ) ;
variables: double vCLOSETOT( 0 ), double vAVGPRO( 0 ),
double vAVGPROPLOT( 0 ) ;
variables: double vINCREMENT( 0 ) ;


vSMA = Average( close, PERIOD ) ;
vUPPER = vSMA + ( CHANNELWIDTH * 0.5 ) ;
vLOWER = vSMA - ( CHANNELWIDTH * 0.5 ) ;


plot1[( Period * 0.5 )]( vSMA, "SMA", COLOR ) ;
plot2[( Period * 0.5 )]( vUPPER, "UPPER", COLOR ) ;
plot3[( Period * 0.5 )]( vLOWER, "LOWER", COLOR ) ;


vCLOSETOT = 0 ;

for vb = 0 to ( PERIOD * 0.5 ) - 1

  begin

    vCLOSETOT = vCLOSETOT + close ;

  end ;

vAVGPRO = vCLOSETOT / ( PERIOD * 0.5 ) ;
```

```
vINCREMENT = absvalue( vAVGPRO - vSMA ) / ( PERIOD *
0.5 ) ;


for vb = 0 to ( PERIOD * 0.5 ) - 1

  begin

    if vAVGPRO >= vSMA then vAVGPROPLOT = vAVGPRO - (
vb * vINCREMENT ) else vAVGPROPLOT = vAVGPRO + ( vb *
vINCREMENT ) ;

    plot1[vb]( vAVGPROPLOT, “SMA”, COLORPRO ) ;

    plot2[vb]( vAVGPROPLOT + ( CHANNELWIDTH * 0.5 ),
“UPPER”, COLORPRO ) ;

    plot3[vb]( vAVGPROPLOT - ( CHANNELWIDTH * 0.5 ),
“LOWER”, COLORPRO ) ;

  end ;
```

Appendix 5: Diamonds Grid and Numbers

Updata

Diamonds Grid

```
Programmer – Christopher Grafton
NAME DIAMONDS GRID
PARAMETER "GAPSIZE" #GAPSIZE=10
PARAMETER "PROJECT" #PROJECT=50
PARAMETER "DAY" #D=8
PARAMETER "MONTH" #M=2
PARAMETER "YEAR" #Y=2010
PARAMETER "WHITE" #WHITE=1
PARAMETER "DAILY" #DAILY=1
PARAMETER "ADJUST" #GAP=20

INDICATORTYPE TOOL

#GAPHOR=10
#PERIOD0=#LASTDATE-DATE(#D,#M,#Y)

#CURDATE=#LASTDATE-1
@LOW=PLOW(LOW,#PERIOD0)
@HIGH=PHIGH(HIGH,#PERIOD0)
@LOWEST=PLOW(LOW,#PERIOD0+#GAPHOR)
@HIGHEST=PHIGH(HIGH,#PERIOD0+#GAPHOR)
@GAPHIGH=0.05 * (@HIGHEST-@LOWEST)
@GAPLOW=0.1 * (@HIGHEST-@LOWEST)
@GAPLEVEL=0.035 * (@HIGHEST-@LOWEST)
@LOWEST=@LOWEST - @GAPLOW
@HIGHEST=@LOWEST + @GAPHIGH
@LEVEL=@LOWEST-@GAPLEVEL*0.5
@LEVEL=@LOWEST-@GAPLEVEL*1.5
@LEVEL=@LOWEST-@GAPLEVEL*2.5
@LEVEL=@LOWEST-@GAPLEVEL*3.5
@LEVEL=@LOWEST-@GAPLEVEL*4.5
@LEVEL=@LOWEST-@GAPLEVEL*5.5
@LEVEL=@LOWEST-@GAPLEVEL*6.5

#I=0
@LEVEL=0.0
```

```
FOR #I=0 TO 7
@LEVEL=@LOWEST-#I*@GAPLEVEL
IF #WHITE=0
DRAWLEVEL DOT,@LEVEL,RGB(105,105,105)
ELSEIF #WHITE=1
DRAWLEVEL DOT,@LEVEL,RGB(0,0,0)
ENDIF
NEXT

IF #WHITE=0
COLOUR RGB(255,255,255)
ELSEIF #WHITE=1
COLOUR RGB(0,0,0)
ENDIF
#CURDATE=DATE(#D,#M,#Y)- #GAP
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*0.5
DRAWTEXT @LEVEL AT "80W (18M)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*0.5
DRAWTEXT @LEVEL AT "18Y"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*1.5
DRAWTEXT @LEVEL AT "40W (200D)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*1.5
DRAWTEXT @LEVEL AT "9Y (460W)"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*2.5
DRAWTEXT @LEVEL AT "20W (100D)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*2.5
DRAWTEXT @LEVEL AT "54M (230W)"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*3.5
DRAWTEXT @LEVEL AT "80D (55D)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*3.5
DRAWTEXT @LEVEL AT "18M (80W)"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*4.5
DRAWTEXT @LEVEL AT "40D (28D)"
```

```
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*4.5
DRAWTEXT @LEVEL AT "9M (40W)"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*5.5
DRAWTEXT @LEVEL AT "20D (14D)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*5.5
DRAWTEXT @LEVEL AT "20W (100D)"
ENDIF
IF #DAILY=1
@LEVEL=@LOWEST-@GAPLEVEL*6.5
DRAWTEXT @LEVEL AT "10D (7D)"
ELSEIF #DAILY=0
@LEVEL=@LOWEST-@GAPLEVEL*6.5
DRAWTEXT @LEVEL AT "10W (55D)"
ENDIF

@LEVEL = @LOWEST + @GAPLEVEL
IF #WHITE=0
COLOUR RGB(255,255,255)
ELSEIF #WHITE=1
COLOUR RGB(0,0,0)
ENDIF

#COUNT=0
#COUNT1=0
FOR #CURDATE=(#LASTDATE-
#PERIOD0) TO (#LASTDATE+#PROJECT)
  IF MOD(#COUNT,#GAPSIZE) != 0
    DRAWTEXT @LEVEL AT "."
  ELSEIF MOD(#COUNT,#GAPSIZE) = 0
    DRAWTEXT @LEVEL AT #COUNT
  ENDIF
    #COUNT=#COUNT+1
NEXT
```

Notes

The diamonds grid superimposes at the bottom of the price chart and does not create a new sub-window. You will need to manually adjust the chart to fit the screen once the grid has been placed.

- #GAPSIZE: this is the gap between the sequence of consecutive numbers running along the top of the grid and serving as a proxy for date. You can

choose any number, but for most purposes 10 is sufficient. This will just count up in tens, leaving dots instead of numbers in-between. If you are looking at shorter periods, then you should choose a lower value, say 1 or 5. If you are looking at longer periods choose 20 or more.

- #PROJECT: this sets how far into the future the numbers keep counting. You should leave a space at the right side of your chart to allow the FLDs to project into and place future potential diamonds.
- #DAY, #MONTH, #YEAR: you need to choose the start date of the grid, inputting day, month and year. For the year, do not abbreviate – write 2009, 2010, etc.
- #WHITE: this sets the colour of the lines and the numbers in the grid. If you are using a white background chart, then you need to select #WHITE=1. If you are using a black background chart, select #WHITE=0.
- #DAILY: if you are conducting an analysis on a daily chart you need to select #DAILY=1. If you are conducting an analysis on a weekly chart, select #DAILY=0.
- #ADJUST: this adjusts the position of the nominal periods on the far left of the grid either closer to the start (0) or further away, depending on the time frame or the size of your chart. The default setting is 10, which means the values are written ten bars back from the zero. Five is closer, 20 is further away.
- The actual Updata file can be downloaded from: **vectisma.com/downloads/diamondgrid**

Numbers from fixed date

You may just want to have the sequence of consecutive numbers at the bottom of the chart but not the grid, for example if you are scanning through charts to pick out likely candidates for further analysis. In this case you can run the following code. The parameters are the same, but there is no need to distinguish daily from weekly.

```
Programmer – Christopher Grafton
NAME NUMBERS
PARAMETER "GAPSIZE" #GAPSIZE=10
PARAMETER "PROJECT" #PROJECT=50
PARAMETER "DAY" #D=9
PARAMETER "MONTH" #M=3
PARAMETER "YEAR" #Y=2009
PARAMETER "WHITE" #WHITE=1

INDICATORTYPE TOOL
DISPLAYSTYLE 2LINES
PLOTSTYLE1 LINE RGB(0,0,0)
PLOTSTYLE2 LINE RGB(0,0,0)
#GAPHOR=10
#PERIOD0=#LASTDATE-DATE(#D,#M,#Y)

#CURDATE=#LASTDATE-1
@LOW=PLOW(LOW,#PERIOD0)
@HIGH=PHIGH(HIGH,#PERIOD0)
@LOWEST=PLOW(LOW,#PERIOD0+#GAPHOR)
@HIGHEST=PHIGH(HIGH,#PERIOD0+#GAPHOR)
@GAPHIGH=0.05 * (@HIGHEST-@LOWEST)
@GAPLOW=0.1 * (@HIGHEST-@LOWEST)
@GAPLEVEL=0.035 * (@HIGHEST-@LOWEST)
@LOWEST=@LOWEST - @GAPLOW
@HIGHEST=@LOWEST + @GAPHIGH
@LEVEL9Y=@LOWEST-@GAPLEVEL*0.5
@LEVEL54M=@LOWEST-@GAPLEVEL*1.5
@LEVEL18M=@LOWEST-@GAPLEVEL*2.5
@LEVEL9M=@LOWEST-@GAPLEVEL*3.5
@LEVEL20W=@LOWEST-@GAPLEVEL*4.5
@LEVEL10W=@LOWEST-@GAPLEVEL*5.5

#I=0
@LEVEL=0.0
FOR #I=0 TO 1
```

```
@LEVEL=@LOWEST-#I*@GAPLEVEL
NEXT

@LEVEL = @LOWEST - @GAPLEVEL*0.5

#COUNT=0
#COUNT1=0
FOR #CURDATE=(#LASTDATE-
#PERIOD0) TO (#LASTDATE+#PROJECT)
IF MOD(#COUNT,#GAPSIZE) != 0
DRAWTEXT @LEVEL AT "."
ELSEIF #WHITE=1
COLOUR RGB(0,0,0)
ENDIF
IF MOD(#COUNT,#GAPSIZE) = 0
DRAWTEXT @LEVEL AT #COUNT
ELSEIF #WHITE=0
COLOUR RGB(255,255,255)
ENDIF
#COUNT=#COUNT+1
NEXT
```

Numbers from period back

Instead of starting the numbers from a fixed date, you may wish to choose a period back, for example to start counting 125 bars back from the last date. The code for this is as follows:

```
NAME NUMBERS PERIOD
PARAMETER "GAPSIZE" #GAPSIZE=10
PARAMETER "PROJECT" #PROJECT=30
PARAMETER "PERIOD" #PERIOD1=125
PARAMETER "WHITE" #WHITE=1

INDICATORTYPE TOOL
DISPLAYSTYLE 2LINES
PLOTSTYLE1 LINE RGB(0,0,0)
PLOTSTYLE2 LINE RGB(0,0,0)
#GAPHOR=10
#PERIOD0=#PERIOD1

#CURDATE=#LASTDATE-1
@LOW=PLOW(LOW,#PERIOD0)
@HIGH=PHIGH(HIGH,#PERIOD0)
```

```
@LOWEST=PLOW(LOW,#PERIOD0+#GAPHOR)
@HIGHEST=PHIGH(HIGH,#PERIOD0+#GAPHOR)
@GAPHIGH=0.05 * (@HIGHEST-@LOWEST)
@GAPLOW=0.1 * (@HIGHEST-@LOWEST)
@GAPLEVEL=0.035 * (@HIGHEST-@LOWEST)
@LOWEST=@LOWEST - @GAPLOW
@HIGHEST=@LOWEST + @GAPHIGH
@LEVEL9Y=@LOWEST-@GAPLEVEL*0.5
@LEVEL54M=@LOWEST-@GAPLEVEL*1.5
@LEVEL18M=@LOWEST-@GAPLEVEL*2.5
@LEVEL9M=@LOWEST-@GAPLEVEL*3.5
@LEVEL20W=@LOWEST-@GAPLEVEL*4.5
@LEVEL10W=@LOWEST-@GAPLEVEL*5.5

#I=0
@LEVEL=0.0
FOR #I=0 TO 1
@LEVEL=@LOWEST-#I*@GAPLEVEL
NEXT

@LEVEL = @LOWEST - @GAPLEVEL*0.5

#COUNT=0
#COUNT1=0
FOR #CURDATE=(#LASTDATE-
#PERIOD1) TO (#LASTDATE+#PROJECT)
IF MOD(#COUNT,#GAPSIZE) != 0
DRAWTEXT @LEVEL AT "."
ELSEIF #WHITE=1
COLOUR RGB(0,0,0)
ENDIF
IF MOD(#COUNT,#GAPSIZE) = 0
DRAWTEXT @LEVEL AT #COUNT
ELSEIF #WHITE=0
COLOUR RGB(255,255,255)
ENDIF
#COUNT=#COUNT+1
NEXT
```

- Updata code can be downloaded from **vectisma.com/downloads/numbers**

TradeStation

```
/// Program Name - Diamonds Grid ///
/// Programmer - Mark Cotton ///
/// Website - www.7GTradingTools.com ///
/// Skype - markcottons.skype ///
/// Email - markcottons.email@gmail.com ///


inputs: int GAPSIZE( 10 ) ;
inputs: int STARTDAY( 6 ) ;
inputs: int STARTMONTH( 3 ) ;
inputs: int STARTYEAR( 2009 ) ;
inputs: int TEXTPOSITION( 25 { Bars back from start
date } ) ;
inputs: int TEXTCOLOR( White ) ;
inputs: int LINEGAPSIZE( 100 ) ;
inputs: int LINEPROJECT( 5 { Bars into future } { Set
Spaces to the Right to 50 in Format Window } ) ;
inputs: int LINECOLOR( DarkGray ) ;


variables: bool vFIRSTBAR( true ) ;
variables: int vTEXTID( 0 ), double vPREVLASTBARDATE(
JuliantoDate( LastCalcJDate ) ) ;
variables: bool vHILOCALC( False ), int vPERIOD(
TEXTPOSITION ) ;
variables: double vSTARTDATE( 0 ), double vSTARTTIME( 0
) ;
variables: double vLOWEST( 999999999 ), intrabarpersist
bool vFINISHED( False ) ;
variables: double vTICK( ( MinMove / PriceScale ) *
iff( Category = 12, 10, 1 ) ) ;
variables: int vPLOT( 0 ), int vCOUNT( 0 ), int vTEXT(
0 ), int va1( 0 ), int va2( 0 ) ;


arrays: string aTEXT[7]( "" ) ;
arrays: int aTEXTID[]( 0 ) ; Array_SetMaxIndex(
aTEXTID[], LINEPROJECT * 40 ) ;
arrays: double aTEXTDATEID[]( 0 ) ; Array_SetMaxIndex(
aTEXTDATEID[], LINEPROJECT + 1 ) ;
```

```
/// Retrieve Date previous to Last Chart Date ///

if vFIRSTBAR = true then

  begin

    while vPREVLASTBARDATE = JuliantoDate(
LastCalcJDate )

      begin
        vCOUNT = vCOUNT + 1 ;
        vTEXTID = Text_New( JuliantoDate( LastCalcJDate
- vCOUNT ), time, high + ( high - low ), " " ) ;
        Text_SetStyle( vTEXTID, 1, 1 ) ;
        vPREVLASTBARDATE = Text_GetDate( vTEXTID ) ;
      end ;


    vFIRSTBAR = false ;

  end ;


if GetAppInfo( aiSpaceToRight ) < LINEPROJECT then
  begin
    value1 = Text_SetString( vTEXTID, "Set Spaces to
the Right to " + numtostr( LINEPROJECT, 0 ) + " in
Format Window" ) ;
  end ;


/// Find grid start position / Calculate Lowest Low to
plot grid below ///

if date >= ELDate( STARTMONTH, STARTDAY, STARTYEAR )
and vHILOCALC = false and GetAppInfo( aiSpaceToRight )
>= LINEPROJECT then

  begin
    if vPERIOD = TEXTPOSITION then
      begin
        vSTARTDATE = date[TEXTPOSITION] ;
        vSTARTTIME = time[TEXTPOSITION] ;
      end ;
```

```
    vPERIOD = vPERIOD + 1 ;


    if low < vLOWEST then vLOWEST = low ;

    if date = vPREVLASTBARDATE then vHILOCALC = true ;
  end ;


/// Draw Grid ///

if vHILOCALC = true and vHILOCALC[1] = false and
barstatus( 1 ) = 2 then

  begin

    /// Numeric Series ///

    vCOUNT = 0 ;

    for vPLOT = vPERIOD - ( TEXTPOSITION + 1 ) downto -
LINEPROJECT

      begin

        /// Dots History ///
        if mod( vCOUNT, GAPSIZE ) <> 0 and vPLOT >= 0
then
          begin

            value1 = Text_New( date[vPLOT],
time[vPLOT], vLOWEST - ( 1 * ( LINEGAPSIZE * vTICK ) ),
“.” ) ;
            Text_SetColor( value1, TEXTCOLOR ) ;
            Text_SetStyle( value1, 2, 2 ) ;
          end ;


        /// Numbers History ///

        if mod( vCOUNT, GAPSIZE ) = 0 and vPLOT >= 0
then

          begin
```

```
            value1 = Text_New( date[vPLOT],
time[vPLOT], vLOWEST - ( 1 * ( LINEGAPSIZE * vTICK ) ),
numtostr( vCOUNT, 0 ) ) ;
            Text_SetColor( value1, TEXTCOLOR ) ;
            Text_SetStyle( value1, 2, 1 ) ;
          end ;


        /// Dots Future / Calculate Future Dates ///

        if vPLOT = -1 then

          begin
            va1 = 0 ; va2 = 0 ;
            while va2 < LINEPROJECT
              begin
                   va1 = va1 + 1 ;
                   aTEXTID[va1] = Text_New(
JuliantoDate( DateToJulian( Date ) + va1 ) , time,
vLOWEST - ( 1 * ( LINEGAPSIZE * vTICK ) ), ".” ) ;

                   Text_SetColor( aTEXTID[va1],
TEXTCOLOR ) ;
                   Text_SetStyle( aTEXTID[va1], 2, 2 ) ;

                   if va1 = 1 or Text_GetDate(
aTEXTID[va1] ) <> Text_GetDate( aTEXTID[va1 - 1] ) then

                        begin
                              va2 = va2 + 1 ;
                              aTEXTDATEID[va2] =
aTEXTID[va1] ;


                        end ;


                   if va1 > 1 and Text_GetDate(
aTEXTID[va1] ) = Text_GetDate( aTEXTID[va1 - 1] ) then

                        begin
                              Text_SetString(
aTEXTID[va1], “ “ ) ;
                              Text_SetColor(
aTEXTID[va1], TEXTCOLOR ) ;
```

```
                                Text_SetStyle(
aTEXTID[va1], 2, 2 ) ;
                          end ;

              end ;

          end ;



        /// Numbers Future ///

        if mod( vCOUNT, GAPSIZE ) = 0 and vPLOT < 0
then

          begin
            Text_SetString( aTEXTDATEID[ absvalue(
vPLOT ) ], numtostr( vCOUNT, 0 ) ) ;
            Text_SetColor( aTEXTDATEID[ absvalue( vPLOT
) ], TEXTCOLOR ) ;
            Text_SetStyle( aTEXTDATEID[ absvalue( vPLOT
) ], 2, 1 ) ;
          end ;

        vCOUNT = vCOUNT + 1 ;
      end ;


    /// Lines 1 to 7 ///

    for vPLOT = vPERIOD downto -LINEPROJECT

      begin
        plot1[vPLOT]( vLOWEST - ( 2 * ( LINEGAPSIZE *
vTICK ) ), "Line1", LINECOLOR ) ;
        plot2[vPLOT]( vLOWEST - ( 3 * ( LINEGAPSIZE *
vTICK ) ), "Line2", LINECOLOR ) ;
        plot3[vPLOT]( vLOWEST - ( 4 * ( LINEGAPSIZE *
vTICK ) ), "Line3", LINECOLOR ) ;
        plot4[vPLOT]( vLOWEST - ( 5 * ( LINEGAPSIZE *
vTICK ) ), "Line4", LINECOLOR ) ;
        plot5[vPLOT]( vLOWEST - ( 6 * ( LINEGAPSIZE *
vTICK ) ), "Line5", LINECOLOR ) ;
        plot6[vPLOT]( vLOWEST - ( 7 * ( LINEGAPSIZE *
vTICK ) ), "Line6", LINECOLOR ) ;
        plot7[vPLOT]( vLOWEST - ( 8 * ( LINEGAPSIZE *
```

```
vTICK ) ), “Line7”, LINECOLOR ) ;
      end ;



    /// Grid Text ///

    if bartype = 2 then

      begin
        aTEXT[1] = “20W (100D)” ;
        aTEXT[2] = “80D (55D)” ;
        aTEXT[3] = “40D (28D)” ;
        aTEXT[4] = “20D (14D)” ;
        aTEXT[5] = “10D (7D)” ;
        aTEXT[6] = “5D (3.5D)” ;
      end ;


    if bartype = 3 then

      begin
        aTEXT[1] = “54M (230W)” ;
        aTEXT[2] = “18M (80W)” ;
        aTEXT[3] = “9M (40W)” ;
        aTEXT[4] = “20W (100D)” ;
        aTEXT[5] = “10W (55D)” ;
        aTEXT[6] = “5W (28D)” ;
      end ;



    for vTEXT = 3 to 9

      begin
        value1 = Text_New( vSTARTDATE, vSTARTTIME,
vLOWEST - ( vTEXT * ( LINEGAPSIZE * vTICK ) ),
aTEXT[vTEXT - 2] ) ;
        Text_SetColor( value1, TEXTCOLOR ) ;
Text_SetStyle( value1, 0, 1 ) ;
      end ;

  end ;
```

Appendix 6: Diamond Placement Code

Updata

Diamond placement on monthly chart

```
Programmer – Christopher Grafton

NAME DIAMOND - MONTHLY
PARAMETER "PROJECT" #PROJECT=50
PARAMETER "STARTGAP" #STARTGAP=-10
PARAMETER "ENDGAP" #ENDGAP=20
INDICATORTYPE TOOL
DISPLAYSTYLE 3LINES
PLOTSTYLE LINE RGB(255,0,0)
PLOTSTYLE2 LINE RGB(255,255,255)
PLOTSTYLE3 LINE RGB(0,0,0)

'Copy the contents of the Code Variables column from
the Phasing Model spreadsheet and paste this into the
space between the two rows of hash marks below. Before
you do this you will need to Save As the code here and
name it for the security being analysed. You will also
need to change the NAME in row 0 from DIAMOND MONTHLY
to that of the security, for example in this case it
would be $ UKX MONTHLY. The pasted contents will need
to overwrite whatever is in the space already. Take
cake not to overwrite anything below the second row of
hash marks. Also, there is no need to copy the entire
column from the spreadsheet, just the rows containing
the code variables. For example copy A3:A43, not A:A.

'###################################

#D=31
#M=3
#Y=1995
#18YAV=168.857142857143
#9YAV=84.4285714285714
#54MAV=42.2142857142857
#18MAV=14.0714285714286
#LASTGRID=199
```

```
#18YLAST=96
#9YLAST=168
#54MLAST=168
#18MLAST=197
#18Y1=96
#18Y2=-50
#9Y1=0
#9Y2=168
#9Y3=-50
#9Y4=-50
#9Y5=-50
#9Y6=96
#9Y7=-50
#54M1=43
#54M2=135
#54M3=-50
#54M4=-50
#54M5=-50
#54M6=-50
#54M7=-50
#54M8=0
#54M9=168
#54M10=-50
#54M11=-50
#54M12=-50
#54M13=96
#54M14=-50
#18M1=16
#18M2=31
#18M3=61
#18M4=78
#18M5=113
#18M6=122
#18M7=154
#18M8=144
#18M9=184
#18M10=197
#18M11=-50
#18M12=-50
#18M13=-50
#18M14=-50
#18M15=43
#18M16=135
#18M17=-50
#18M18=-50
```

```
#18M19=-50
#18M20=-50
#18M21=-50
#18M22=0
#18M23=168
#18M24=-50
#18M25=-50
#18M26=-50
#18M27=96
#18M28=-50


'##################################

@REM=0
@NEWNUM=0
@REM1=0
@NEWNUM1=0
@REM2=0
@NEWNUM2=0
@REM3=0
@NEWNUM3=0
#GAP=10
#GAPSIZE=10
#GAPHOR=10
#PERIOD0=#LASTDATE-DATE(#D,#M,#Y)
@LEVEL1=0

#CURDATE=#LASTDATE-1
@LOW=PLOW(LOW,#PERIOD0)
@HIGH=PHIGH(HIGH,#PERIOD0)
@LOWEST=PLOW(LOW,#PERIOD0+#GAPHOR)
@HIGHEST=PHIGH(HIGH,#PERIOD0+#GAPHOR)
@GAPHIGH=0.05*(@HIGHEST-@LOWEST)
@GAPLOW=0.1*(@HIGHEST-@LOWEST)
@GAPLEVEL=0.035*(@HIGHEST-@LOWEST)
@LOWEST=@LOWEST-@GAPLOW
@HIGHEST=@LOWEST+@GAPHIGH
@LEVEL=@LOWEST-@GAPLEVEL*0.5
@LEVEL=@LOWEST-@GAPLEVEL*1.5
@LEVEL=@LOWEST-@GAPLEVEL*2.5
@LEVEL=@LOWEST-@GAPLEVEL*3.5
@LEVEL=@LOWEST-@GAPLEVEL*4.5
@LEVEL18Y=@LOWEST-@GAPLEVEL*0.5
@LEVEL9Y=@LOWEST-@GAPLEVEL*1.5
```

```
@LEVEL54M=@LOWEST-@GAPLEVEL*2.5
@LEVEL18M=@LOWEST-@GAPLEVEL*3.5

#I=0
@LEVEL=0.0
FOR #I=0 TO 5
@LEVEL=@LOWEST-#I*@GAPLEVEL
DRAWCOLOUR RGB(0,0,0)
DRAWLEVEL DOT,@LEVEL,RGB(0,134,139)
NEXT

@LEVEL=@LOWEST+@GAPLEVEL
COLOUR RGB(0,0,0)
#COUNT=0
#COUNT1=0
FOR #CURDATE=DATE(#D,#M,#Y) TO #LASTDATE+#PROJECT
   IF MOD(#COUNT,#GAPSIZE) != 0
       DRAWTEXT @LEVEL AT "."
  ELSEIF MOD(#COUNT,#GAPSIZE) = 0
      DRAWTEXT @LEVEL AT #COUNT
  ENDIF
  #COUNT=#COUNT+1
NEXT

COLOUR RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#STARTGAP
@LEVEL1=@LOWEST+@GAPLEVEL*0.3
DRAWTEXT @LEVEL1 AT "Nom"
@LEVEL=@LOWEST-@GAPLEVEL*0.5
DRAWTEXT @LEVEL AT "18Y"
@LEVEL=@LOWEST-@GAPLEVEL*1.5
DRAWTEXT @LEVEL AT "9Y"
@LEVEL=@LOWEST-@GAPLEVEL*2.5
DRAWTEXT @LEVEL AT "54M"
@LEVEL=@LOWEST-@GAPLEVEL*3.5
DRAWTEXT @LEVEL AT "18M"

@REM=MOD((#18YAV/12)*10,INT((#18YAV/12)*10))
IF @REM>0.5
   @REM=1
ELSE
   @REM=0
ENDIF
@NEWNUM=(INT((#18YAV/12)*10)+@REM)/10
```

```
@REM1=MOD((#9YAV/12)*10,INT((#9YAV/12)*10))
IF @REM1>0.5
   @REM1=1
ELSE
   @REM1=0
ENDIF
@NEWNUM1=(INT((#9YAV/12)*10)+@REM1)/10

@REM2=MOD((#54MAV)*10,INT((#54MAV)*10))
IF @REM2>0.5
   @REM2=1
ELSE
   @REM2=0
ENDIF
@NEWNUM2=(INT((#54MAV)*10)+@REM2)/10

@REM3=MOD((#18MAV*4.333)*10,INT((#18MAV*4.333)*10))
IF @REM3>0.5
   @REM3=1
ELSE
   @REM3=0
ENDIF
@NEWNUM3=(INT((#18MAV*4.333)*10)+@REM3)/10

COLOUR RGB(0,0,0)
#CURDATE=#LASTDATE+#ENDGAP
@LEVEL1=@LOWEST+@GAPLEVEL*0.3
DRAWTEXT @LEVEL1 AT "Avg"
@LEVEL=@LOWEST-@GAPLEVEL*0.5
DRAWTEXT @LEVEL AT @NEWNUM "Y"
@LEVEL=@LOWEST-@GAPLEVEL*1.5
DRAWTEXT @LEVEL AT @NEWNUM1 "Y"
@LEVEL=@LOWEST-@GAPLEVEL*2.5
DRAWTEXT @LEVEL AT @NEWNUM2 "M"
@LEVEL=@LOWEST-@GAPLEVEL*3.5
DRAWTEXT @LEVEL AT @NEWNUM3 "W"

'18Y DIAMONDS
#CURDATE=DATE(#D,#M,#Y)+#18Y1
DRAWIMAGE AT,#CURDATE,@LEVEL18Y,Diamond 8,RGB(139,69,19)
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18Y2
```

```
DRAWIMAGE AT,#CURDATE,@LEVEL18Y,Diamond 8,RGB(139,69,19)
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)

'9Y DIAMONDS
#CURDATE=DATE(#D,#M,#Y)+#9Y1
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y2
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y3
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y4
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y5
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y6
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9Y7
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,Diamond 8,RGB(0,134,139)

DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)

'54M DIAMONDS
#CURDATE=DATE(#D,#M,#Y)+#54M1
```

```
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M2
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M3
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M4
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M5
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M6
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M7
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M8
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M9
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M10
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M11
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M12
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M13
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54M14
DRAWIMAGE AT,#CURDATE,@LEVEL54M,Diamond 8,RGB(105,105,105)
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)

'18M DIAMONDS
#CURDATE=DATE(#D,#M,#Y)+#18M1
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
```

```
#CURDATE=DATE(#D,#M,#Y)+#18M2
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M3
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M4
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M5
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M6
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M7
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M8
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M9
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M10
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M11
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M12
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M13
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M14
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M15
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M16
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M17
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M18
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M19
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M20
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M21
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M22
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M23
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M24
```

```
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M25
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M26
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M27
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18M28
DRAWIMAGE AT,#CURDATE,@LEVEL18M,Diamond 8,RGB(0,0,0)

'FUTURE DIAMONDS
#CURDATE=DATE(#D,#M,#Y)+#18YLAST+#18YAV
DRAWIMAGE AT,#CURDATE,@LEVEL18Y,DOT 8,RGB(255,0,0)
#CURDATE=DATE(#D,#M,#Y)+#9YLAST+#9YAV
DRAWIMAGE AT,#CURDATE,@LEVEL9Y,DOT 8,RGB(255,0,0)
#CURDATE=DATE(#D,#M,#Y)+#54MLAST+#54MAV
DRAWIMAGE AT,#CURDATE,@LEVEL54M,DOT 8,RGB(255,0,0)
#CURDATE=DATE(#D,#M,#Y)+#18MLAST+#18MAV
DRAWIMAGE AT,#CURDATE,@LEVEL18M,DOT 8,RGB(255,0,0)


FOR #CURDATE=#LASTDATE TO #LASTDATE+#PROJECT
   @PLOT=@HIGHEST
NEXT
FOR #CURDATE=0 TO #LASTDATE
   @PLOT2=@HIGHEST
NEXT
#CURDATE=#LASTDATE
DRAWLINESTYLE DOT
DRAWCOLOUR RGB(255,0,0)
DRAWLINE 0,LOW,0,@LOWEST
```

Notes

This is the code for semi-automatic diamond placement on a monthly chart – in this case FTSE 100 (UKX) from March 1995 to October 2011. There are separate codes for the weekly and daily charts. The Code Variables are copied and pasted from a spreadsheet and of course vary from analysis to analysis.

It is not absolutely imperative to use the Diamond Placement Code, but it is a good idea. For my own work, I always use it. The advantages are that the finish is neat, polished and consistent. Furthermore, your analyses can be quickly updated as new data comes through and saved for future reference.

In many ways it is easier to save the code for each security and just rerun it than it is to save charts.

The code and inputs for the diamond placement can be found on the Phasing Model spreadsheet which can be downloaded from **vectisma.com/downloads/phasingmodel**. Because considerable time and effort were spent programming this spreadsheet. I will be charging US $199 for the download. The model can be platform and country specific – i.e. to take national holidays into account for future projections. Included for free in this one-off fee are three one-hour-plus long online videos showing a full Hurst cycle analysis on three different securities.

The Updata code for monthly, weekly and daily diamond placements can be found at the following locations, respectively:

- **vectisma.com/downloads/diamondplacement_monthly**;
- **vectisma.com/downloads/diamondplacement_weekly**; and
- **vectisma.com/downloads/diamondplacement_daily**

Appendix 7: Discrete Fourier Transform Code

Updata – Visual Basic DFT (VB.Net)

Discrete Fourier Transform

Developed by John F. Ehlers

Coded in VB.NET by Paolo Barletta Ph.D.

Produces a spectrum measured by a Discrete Fourier Transform (DFT) but with a music-based transform applied to it.

The color in the heatmap indicates the cycle amplitude and the cycle period is the vertical scale, scaled from 8 to 50 bars.

The heatmap is in time synchronism with the barchart.

The additional transformation makes it easier to identify the variable dominant cycle.

Reference: *Stocks & Commodities* Magazine, January 2007.

```
Imports Microsoft.VisualBasic
Imports System.Windows.Forms
Imports System.Drawing
Imports Updata
Imports System.IO
Imports System.Text

Public Class Updata_TestIndicator
  Implements ICustomIndicator

  ' the init() function will be called once each time
the indicator is created, before everything
  ' else is called
  Public Sub init() Implements ICustomIndicator.init
  End Sub

  ' the getLines() function returns the number of lines
displayed, and also
  ' how to display them
' iLineStyles is an array of values from the linestyles
list in the header, eg BAR, CANDLE
' iToolStyles is an array of values from the linestyles
```

```
list in the header, eg CHART, TOOL
‘ sNames is an array of names for each of the
individual lines
‘ iColours is the default up colour, and iColours2 the
default down colour for each bar
  Public Function getLines(ByVal iLineStyles() As
LineStyles, ByVal iChartStyles() As ChartStyles, ByVal
sNames() As String, ByVal iColours() As Integer, ByVal
iColours2() As Integer)  As Integer Implements
ICustomIndicator.getLines
iLineStyles(0) = LineStyles.Dot
    iChartStyles(0) = ChartStyles.Chart
    iColours(0) = Color.Red.ToArgb()
    iColours2(0) = Color.Red.ToArgb()
    sNames(0) = “Discrete Fourier Transform”
    getLines = 1
  End Function


‘ the queryForParameters function needs to fill up the
arrays that are passed in, and return the
  ‘ number of parameters that the indicator refers to.
  ‘ iRets - returns types of the variables required –
eg PRICEVARIABLE, INDICATORVARIABLE
  ‘ sNames - returns the short name for the parameter
  ‘ sDescrips – returns the long descriptive text for
the variable shown on the prompt dialog
  ‘ defaults – returns the default value (must be of
the correct type) for this parameter

  Public Function queryForParameters(ByVal iRets() As
Updata.VariableTypes, ByVal sNames() As String, ByVal
sDescrips() As String, ByVal defaults As Object()) As
Integer Implements ICustomIndicator.queryForParameters
    iRets(0) = VariableTypes.PriceVariable
    sNames(0) = “Period”
    sDescrips(0) = “Period”
    defaults(0) = CType(50, Integer)
    queryForParameters = 1
  End Function

  ‘ these global variables are maintained between the
last calculation, and the current paint
‘ all this information is required to plot the heatmap
  Private Colour1(50, 1) As Integer
```

```
  Private Colour2(50, 1) As Integer
  Private cycledata(50, 1) As Double

  ' recalculateAll does the full recalculation of your
indicator based on source data
  ' you need to fill up the dRet double array with
return values
  ' dSrc: the data of the main chart is drawn on, first
index=point, second=field OHLCV
  '       note that study lines may only return one
point per field
  ' oParams: returns the values of the parameters
specified in queryForParameters, correctly formatted
  '       note that lines and lists will be returned as
price arrays, again OHLCV or just c if a study
  ' dRet: The data to be returned, this has one
Double()() for each line returned.
  '       each Double()() is the same length as the
source data, and each index in
  '       the source data should match the same index
in the dest data
  ' iTradeTypes: return values from the tradetypes
above, or ignore if not a system above,
  ' dTradeOpenPrices: return price to open a new
position at
  ' dTradeClosePrices: return price to close a position
at
  ' iTradeAmount: return size of holding place, leave
blank to use defaults


Public Function recalculateAll(ByVal dSrc()() As Double
, ByVal oParams() As Object, ByVal dRet()()() As
Double, ByVal iTradeTyles()() As Integer, ByVal
dTradeOpenPrices()() As Double, ByVal
dTraderClosePrices()() As Double, ByVal
iTradeAmounr()() As Integer, ByVal dStopLevels()() As
Double) As Boolean Implemets ICustom
Indicator.recalculateAll
          If dSrc.Length = 0 Then
      recalculateAll = False
    End If
    Dim i As Integer
    Dim alpha1 As Double
    Dim HP(dRet(0).Length - 1) As Double
```

```
    Dim CleanedData(dRet(0).Length - 1) As Double
    Dim Period As Integer
    Dim n As Integer
    Dim MaxPwr As Double
    Dim Num As Double
    Dim Denom As Double
    Dim price(dRet(0).Length - 1) As Double
    Dim k As Integer
    'meant to set size of display window
    Dim Window As Integer = CType(oParams(0), Integer)
    Dim CosinePart(50) As Double
    Dim SinePart(50) As Double
    Dim Pwr(50) As Double
    Dim DB(50) As Double

    ReDim cycledata(50, dRet(0).Length)
    ReDim Colour1(50, dRet(0).Length)
    ReDim Colour2(50, dRet(0).Length)

    For i = 2000 To dRet(0).Length - 1

      price(i) = (dSrc(i)(1) + dSrc(i)(2)) / 2

      'Get a detrended version of the data by High Pass
Filtering with a 40 Period cutoff
If i <= 5 Then
        HP(i) = price(i)
        CleanedData(i) = price(i)
      End If

      If i > 5 Then

alpha1 = (1 - System.Math.Sin(360 / 40)) / system.math.
Cos(360 / 40)
            HP(i) = 0.5 * (1 + alpha1) * (price(i) -
price(i — 1)) + alpha1 * (HP(i — 1)
        CleanedData(i) = (HP(i) + 2 * HP(i - 1) + 3 *
HP(i - 2) + 3 * HP(i - 3) + 2 * HP(I — 4) + HP(I — 5))
/ 12
      End If

      If i > Window Then

        'This is the DFT
        For Period = 8 To 50
```

```
          CosinePart(Period) = 0
          SinePart(Period) = 0
          For n = 0 To Window - 1

            CosinePart(Period) = CosinePart(Period) +
CleanedData(i-n) * system.math.Cos(2*system.Math.Pi *
(i - n) / Period)
            SinePart(Period) = SinePart(Period) +
CleanedData(i-n) *
system.math.Sin(2*system.math.Sin(2*system.Math.Pi * (i
- n) / Period)
          Next n

Pwr(Period) = CosinePart(Period) * CosinePart(Period) +
SinePart(Period) * SinePart(Period)
        Next Period

        'Find Maximum Power Level for Normalization
        MaxPwr = Pwr(8)
        For Period = 8 To 50
          If Pwr(Period) > MaxPwr Then
            MaxPwr = Pwr(Period)
          End If
        Next Period

        'Normalize Power Levels and Convert to Decibels
        For Period = 8 To 50
          If MaxPwr > 0 And Pwr(Period) > 0 Then
            DB(Period) = - 10 * system.math.log10(0.01
/ (1 - 0.99 * Pwr(Period) / MaxPwr)) /
system.math.log10(10)
          End If
          If DB(Period) > 20 Then
            DB(Period) = 20
          End If
        Next Period

        'Find Dominant Cycle using CG algorithm
        Num = 0
        Denom = 0
        For Period = 8 To 50
          cycledata(period, i) = 0
          If DB(Period) < 3 Then
            Num = Num + Period * (3 - DB(Period))
            Denom = Denom + (3 - DB(Period))
```

```
          End If

cycledata(period, i) = DB(Period)   '!!!! is this what
we're meant to plot?
        Next Period

        ' store the dominant cycle in period 7
        If Denom <> 0 Then
          cycledata(7, i) = Num / Denom
        End If


        'Plot the Spectrum as a Heatmap
        For Period = 8 To 50
          'Convert Decibels to RGB Color for Display
          If DB(Period) > 10 Then
            Colour1(Period, i) =255 * (2 - DB(Period) /
10)
            Colour2(Period, i) = 0
          End If
          If DB(Period) <= 10 Then
            Colour1(Period, i) = 255
            Colour2(Period, i) = 255 * (1 - DB(Period)
/ 10)
          End If


        Next Period



        ' force a 0 to 50 range
        For k = 0 To dRet(0)(i).Length - 1
          dRet(0)(i)(k) = 8
          If i Mod 2 = 0 Then
            dRet(0)(i)(k) = 50
          End If
        Next k


      End If

    Next i

    recalculateAll = True
```

```
  End Function


  ' reserved for future support
  Public Function recalculateLast(ByVal dSrc()() As
Double, ByVal oParams() As Object, ByVal dRet()()() As
Double) As Boolean Implements
ICustomIndicator.recalculateLast
    recalculateLast = False
  End Function

  ' override the default paint function with this code
  ' return false if you want Updata to paint the line
as per normal
  ' inputs:  g is the Graphics object to draw to
  '          oParams are the current values for the
parameters of this indicator
  '          ds are the current price values for this
indicator
  '          iFirstVisible is the first bar to be
visible on the screen area
  '          iLastVisible is the last bar to be visible
on the screen area
  '       iGraphFunctions is the object that holds info
about the physical chart displayed
  '       iLineNumber is the number of the line being
drawn iLineNumber is the number of the line being drawn

  Public Function
paint(ByVal g As System.Drawing.Graphics, ByVal
oParams() As Object, ByVal ds()()() As Double, ByVal
iFirstVisible As Integer, ByVal iLastVisible As
Integer, ByVal c As iGraphFunctions, ByVal iLineNum As
Integer) As Boolean Implements ICustomIndicator.paint
    If iLineNum = 1 Then
      paint = False
    Else
      'Line Conditions
      Dim x, y, y2, i, Period As Integer
      Dim pLastPoint As Point
      If (iFirstVisible < iLastVisible – 10) Then '
checks to see it's not just refreshing the last point
        ' shade the background in black

Using b As SolidBrush = New SolidBrush(Color.fromArgb(0
, 0, 0))
```

```
g.FillRectangle(b, c.getLocation().x, c.getLocation().y
, c.getSize().Width, c.getSize().Height)
        End Using
      End If

      ' now show the colour spectrum

Using brush As SolidBrush = New SolidBrush(Color.fromAr
gb(0, 0, 0))
        For period = 8 To 50
          y = c.DataToY(period)
          y2 = c.DataToY(period + 1)
          For i = iFirstVisible To iLastVisible

brush.Color = Color.fromArgb(Colour1(period, i), Colour
2(period, i), 0)
            x = c.DataToX(i)
            If i <> iFirstVisible Then
              If y - y2 > 1 Then
                   g.FillRectangle(brush, x, y2, x -
pLastPoint.x + 1, y - y2 + 1)

 'g.FillRectangle(brush, x, y, x, pLastPoint.y)
              Else

g.FillRectangle(brush, x, y2, x - pLastPoint.x + 1, 1)
              End If
            End If
            pLastPoint = New Point(x, y)
          Next i
        Next period
      End Using


      Dim showdc As Boolean

showdc = not True '!!!! change this to not show the
dominant cycle

      If showdc Then  ' show dominant cycle

Using pUp As Pen = New pen(Color.fromArgb(255, 255, 255
))
          pUp.Width = 2
```

```
            period = 7
            y = c.DataToY(period)
            y2 = c.DataToY(period + 1)
            For i = iFirstVisible To iLastVisible
              If period <> 7 Then

pUp.Color = Color.fromArgb(Colour1(period, i),
Colour2(period, i), 0)
              Else
                pUp.Width = 4
              End If
              y = c.DataToY(cycledata(7, i))
              x = c.DataToX(i)
              If i <> iFirstVisible Then

g.DrawLine(pUp, x, y, pLastPoint.x, pLastPoint.y)
              End If
              pLastPoint = New Point(x, y)
            Next i
          End Using
        End If

        paint = True
      End If
    End Function

End Class
```

- The VB Code for Updata can be downloaded from: **vectisma.com/downloads/dft**

TradeStation – Easy Language

{Discrete Fourier Transform, Copyright (c) 2006 John F. Ehlers}

```
Inputs:
  Price((H+L)/2),
  Window(50),
  ShowDC(False);

Vars:
  alpha1(0),
  HP(0),
  CleanedData(0),
  Period(0),
  n(0),
  MaxPwr(0),
  Num(0),
  Denom(0),
  DominantCycle(0),
  Color1(0),
  Color2(0);

//Arrays are sized to have a maximum Period of 50 bars
Arrays:
  CosinePart[50](0),
  SinePart[50](0),
  Pwr[50](0),
  DB[50](0);

//Get a detrended version of the data by High Pass
Filtering with a 40 Period cutoff
If CurrentBar <= 5 Then Begin
  HP = Price;
  CleanedData = Price;
End;
If CurrentBar > 5 Then Begin
  alpha1 = (1 - Sine(360/40))/Cosine(360/40);
  HP = .5*(1 + alpha1)*(Price - Price[1]) +
alpha1*HP[1];
  CleanedData = (HP + 2*HP[1] + 3*HP[2] + 3*HP[3] +
2*HP[4] + HP[5])/12;
End;

//This is the DFT
```

```
For Period = 8 to 50 Begin
  CosinePart[Period] = 0;
  SinePart[Period] = 0;
  FOR n = 0 to Window - 1 Begin
    CosinePart[Period] = CosinePart[Period] +
CleanedData[n]*Cosine(360*n/Period);
    SinePart[Period] = SinePart[Period] +
CleanedData[n]*Sine(360*n/Period);
  End;
  Pwr[Period] = CosinePart[Period]*CosinePart[Period] +
SinePart[Period]*SinePart[Period];
End;

//Find Maximum Power Level for Normalization
MaxPwr = Pwr[8];
For Period = 8 to 50 Begin
  If Pwr[Period] > MaxPwr Then MaxPwr = Pwr[Period];
End;

//Normalize Power Levels and Convert to Decibels
For Period = 8 to 50 Begin
  IF MaxPwr > 0 and Pwr[Period] > 0 Then DB[Period] = -
10*LOG(.01 / (1 - .99*Pwr[Period] / MaxPwr))/Log(10);
  If DB[Period] > 20 then DB[Period] = 20;
End;

//Find Dominant Cycle using CG algorithm
Num = 0;
Denom = 0;
For Period = 8 to 50 Begin
  If DB[Period] < 3 Then Begin
    Num = Num + Period*(3 - DB[Period]);
    Denom = Denom + (3 - DB[Period]);
  End;
End;
If Denom <> 0 then DominantCycle = Num/Denom;
If ShowDC = True Then Plot1(DominantCycle, "S1", RGB(0,
0, 255),0,2);

//Plot the Spectrum as a Heatmap
For Period = 8 to 50 Begin
  //Convert Decibels to RGB Color for Display
  If DB[Period] > 10 Then Begin
    Color1 = 255*(2 - DB[Period]/10);
    Color2 = 0;
  End;
```

```
  If DB[Period] <= 10 Then Begin
    Color1 = 255;
    Color2 = 255*(1 - DB[Period]/10);
  End;
  If Period = 8 Then Plot8(8, "S8", RGB(Color1, Color2,
0),0,4);
  If Period = 9 Then Plot9(9, "S9", RGB(Color1, Color2,
0),0,4);
  If Period = 10 Then Plot10(10, "S10", RGB(Color1,
Color2, 0),0,4);
  If Period = 11 Then Plot11(11, "S11", RGB(Color1,
Color2, 0),0,4);
  If Period = 12 Then Plot12(12, "S12", RGB(Color1,
Color2, 0),0,4);
  If Period = 13 Then Plot13(13, "S13", RGB(Color1,
Color2, 0),0,4);
  If Period = 14 Then Plot14(14, "S14", RGB(Color1,
Color2, 0),0,4);
  If Period = 15 Then Plot15(15, "S15", RGB(Color1,
Color2, 0),0,4);
  If Period = 16 Then Plot16(16, "S16", RGB(Color1,
Color2, 0),0,4);
  If Period = 17 Then Plot17(17, "S17", RGB(Color1,
Color2, 0),0,4);
  If Period = 18 Then Plot18(18, "S18", RGB(Color1,
Color2, 0),0,4);
  If Period = 19 Then Plot19(19, "S19", RGB(Color1,
Color2, 0),0,4);
  If Period = 20 Then Plot20(20, "S20", RGB(Color1,
Color2, 0),0,4);
  If Period = 21 Then Plot21(21, "S21", RGB(Color1,
Color2, 0),0,4);
  If Period = 22 Then Plot22(22, "S22", RGB(Color1,
Color2, 0),0,4);
  If Period = 23 Then Plot23(23, "S23", RGB(Color1,
Color2, 0),0,4);
  If Period = 24 Then Plot24(24, "S24", RGB(Color1,
Color2, 0),0,4);
  If Period = 25 Then Plot25(25, "S25", RGB(Color1,
Color2, 0),0,4);
  If Period = 26 Then Plot26(26, "S26", RGB(Color1,
Color2, 0),0,4);
  If Period = 27 Then Plot27(27, "S27", RGB(Color1,
Color2, 0),0,4);
  If Period = 28 Then Plot28(28, "S28", RGB(Color1,
Color2, 0),0,4);
```

```
  If Period = 29 Then Plot29(29, “S29”, RGB(Color1,
Color2, 0),0,4);
  If Period = 30 Then Plot30(30, “S30”, RGB(Color1,
Color2, 0),0,4);
  If Period = 31 Then Plot31(31, “S31”, RGB(Color1,
Color2, 0),0,4);
  If Period = 32 Then Plot32(32, “S32”, RGB(Color1,
Color2, 0),0,4);
  If Period = 33 Then Plot33(33, “S33”, RGB(Color1,
Color2, 0),0,4);
  If Period = 34 Then Plot34(34, “S34”, RGB(Color1,
Color2, 0),0,4);
  If Period = 35 Then Plot35(35, “S35”, RGB(Color1,
Color2, 0),0,4);
  If Period = 36 Then Plot36(36, “S36”, RGB(Color1,
Color2, 0),0,4);
  If Period = 37 Then Plot37(37, “S37”, RGB(Color1,
Color2, 0),0,4);
  If Period = 38 Then Plot38(38, “S38”, RGB(Color1,
Color2, 0),0,4);
  If Period = 39 Then Plot39(39, “S39”, RGB(Color1,
Color2, 0),0,4);
  If Period = 40 Then Plot40(40, “S40”, RGB(Color1,
Color2, 0),0,4);
  If Period = 41 Then Plot41(41, “S41”, RGB(Color1,
Color2, 0),0,4);
  If Period = 42 Then Plot42(42, “S42”, RGB(Color1,
Color2, 0),0,4);
  If Period = 43 Then Plot43(43, “S43”, RGB(Color1,
Color2, 0),0,4);
  If Period = 44 Then Plot44(44, “S44”, RGB(Color1,
Color2, 0),0,4);
  If Period = 45 Then Plot45(45, “S45”, RGB(Color1,
Color2, 0),0,4);
  If Period = 46 Then Plot46(46, “S46”, RGB(Color1,
Color2, 0),0,4);
  If Period = 47 Then Plot47(47, “S47”, RGB(Color1,
Color2, 0),0,4);
  If Period = 48 Then Plot48(48, “S48”, RGB(Color1,
Color2, 0),0,4);
  If Period = 49 Then Plot49(49, “S49”, RGB(Color1,
Color2, 0),0,4);
  If Period = 50 Then Plot50(50, “S50”, RGB(Color1,
Color2, 0),0,4);
End;
```

Appendix 8: Volatility Index code

Updata

```
Programmer - Christopher Grafton
NAME VOLATILITY INDEX
PARAMETER "INDEX" ~INDEX="$UKX-FTSE"
PARAMETER "LEVEL" @LEVEL=0

INDICATORTYPE TOOL

#PERIOD1=14
@SUM=0
@ATRSEC=0
@ATRSECNORM=0
@ATRIND=0
@ATRINDNORM=0
@RV=0
@LOW=0
@HIGH=0
@SMASEC=0
@SMAIND=0
@LASTPRICE=0
@FIRSTPRICE=0
#TIMEFRAME=250
@LEVEL=0
FOR #CURDATE=#LASTDATE-#TIMEFRAME TO #LASTDATE

@LEVEL=PHIGH(HIGH,#TIMEFRAME)
      @ATRSEC=ATR(#PERIOD1)
      @SMASEC=MAVE(#PERIOD1)
      @ATRSECNORM=(@ATRSEC/@SMASEC)*100

      SOURCEDATA ~INDEX
      @ATRIND=ATR(#PERIOD1)
      @SMAIND=MAVE(#PERIOD1)
      @ATRINDNORM=(@ATRIND/@SMAIND)*100
      @RV=@ATRSECNORM/@ATRINDNORM

NEXT
#CURDATE=#LASTDATE
ENDIF
DRAWCOLOUR RGB(0,0,0)
DRAWFONT ARIAL 15
```

```
DRAWTEXT @LEVEL AT "RV" @RV
'DRAWTEXT @LEVEL ABOVE "SEC" @ATRSECNORM
'DRAWTEXT @LEVEL BELOW "INDEX" @ATRINDNORM

QUOTENOTE RV=@RV
```

Notes

The parameter "Index" is the index you wish to compare the security to. In this case it is the FTSE 100 as an ESignal ticker ($UKX-FTSE).

The Updata code can be downloaded from:
vectisma.com/downloads/volatility