10. USING FORMULA DOCUMENTS TO PERFORM CALCULATIONS ON IMAGES

Aim: To introduce you to how formula documents work and some applications of formulas to perform calculations on images.

Objectives: By the end of this tutorial you will have learnt how to construct simple formulas, apply them to one or more images to perform useful tasks, and to utilise flag information that comes with some images.

Formula documents provide a powerful tool for modelling and performing sophisticated manipulations and calculations using images. For the non-mathematical they can seem rather daunting. This section thus starts with some background information on Formula documents, which, after reading through, you should use largely for reference. This background explains the syntax used in formulas, gives some advice on best-practice for using formula documents (using comments and constants), and provides reference information on mathematical and logical operators and functions which can be used in formulas. Those of you with some knowledge of using formulas in spreadsheets (e.g. in *Excel*) or of computer programming will find Formula documents fairly straightforward. Those without such a background will probably find them a little difficult! The information here can also be obtained using the *Bilko* on-line Help facility. While going through the background, you will be taken through some routine uses of formulas so that you can see them in action. The tutorial finishes by introducing you to how to use bit-wise operators to interrogate and utilise the "flag" information that comes with various image types (e.g., ENVISAT data).

Background

Each Formula document consists of a series of executable statements, each terminated by a semi-colon '**;**'. Images are referred to in formulas by the **@** symbol followed by a number which refers to the tile **Selector** toolbar buttons or stack **Selector** list number and maps to the appropriate image in the connected images window. Unless only part of a referenced image has been selected, the Formula document will operate on every pixel in the images so referenced. Unless only a single image is being operated on, images have to be connected and the Selector mapping set up, before they can be operated on by Formula documents.

This tutorial introduces you to:

1. Use of comments,
2. Operators,
3. Functions,
4. Setting up of constants,
5. Conditional statements, and
6. Bit-wise operators for utilising "flag" information.

1. Comments

Comments make formulae comprehensible both to others and (when you come back to them after some time) to yourself. It is good practice to include copious comments in your formula documents.

Comments are any lines preceded by the hash symbol **#**.

An example of a set of comments at the start of a formula document is given below:

```
# Start of Bilko formula document to radiometrically correct Landsat-5 TM bands
# 1-3 collected over the Turks and Caicos on 22 June 1990.
#    Converting DN to at satellite spectral radiance (L) using formulae of the type:
#             Lmin + (Lmax/254 - Lmin/255) * @n ;
#
#    Input values
#    ==========
#    Lmin: TM1 = -0.116, TM2 = -0.183, TM3 = -0.159
#    Lmax: TM1 = 15.996; TM2 = 31.776; TM3 = 24.394
```

All comment lines are ignored by the program that processes formula documents. They help other people to understand your formulas and help you to understand them when you come back to them after not using them for some time.

2. Operators

Formula documents support a range of arithmetic, relational (comparison), and logical "operators" to manipulate images. The most important ones are listed below.

Arithmetic	**Relational**	**Logical**
Exponentiation (^)	Equality (==)	AND
Multiplication (*)	Inequality (<>)	OR
Division (/)	Less than (<)	NOT (can also use the "!" sign)
Addition (+)	Greater than (>)	
Subtraction (–)	Less than or Equal to (<=)	
	Greater than or Equal to (>=)	

Do not confuse the equality operator used in comparisons (==) with the assignment operator (=) used in constant or other statements. The difference between the two is illustrated in the section on conditional statements. When several operations occur in a Formula document statement, each part is evaluated in a predetermined order. That order is known as operator precedence. Parentheses (brackets) can be used to override the order of preference and force some parts of a statement to be evaluated before others. Operations within parentheses are always performed before those outside. Within parentheses, however, normal operator precedence is maintained. When Formula document statements contain operators from more than one category, arithmetic operators are evaluated first, relational operators next, and logical operators are evaluated last. Among the arithmetic operators, exponentiation (^) has preference (is evaluated first) over multiplication/division (*, /), which in turn have preference over addition/subtraction (+, –). When multiplication and division (or addition and subtraction) occur together in a statement, each operation is evaluated as it occurs from left to right (unless parentheses direct otherwise).

Thus the formula statement:

0.15 * (@1 – 3.0) ;

will firstly subtract 3.0 from every pixel in the image referenced by @1 and then multiply the result by 0.15 to generate the output image. In this example the parentheses make sure the lower precedence subtraction operation takes place *before* the higher precedence multiplication. If the formula statement had been written:

0.15 * @1 – 3.0 ;

then each pixel of the image referenced by @1 would have been multiplied by 0.15 and afterwards would have had 3.0 subtracted, giving a totally different and wrong! result.

This simple formula can be used to convert pixel DNs in an 8-bit processed National Oceanic and Atmospheric Administration (NOAA) AVHRR Oceans Pathfinder sea surface temperature (SST) image (freely available on the internet) to values corresponding to SST in degrees Celsius. [*Note:* The output image type needs to be 32-bit floating point to store these numbers]. You will now do this in case you are getting bored.

Activity: Open the image file ***Global_SST_Dec2000.gif***. Right-click on the image and select the **Zoom** option from the menu; set the zoom to 45% of full size so that you can see the whole image at once. Note that all land and areas with no data (clouded during December 2000) are set to black (0) and the sea to varying shades of grey. Use a histogram to confirm that pixel values in the sea have values between 1 and 240. [*Hint:* you will need to check **Ignore zero:** in the **Options, Scale** dialog box for the histogram because of all the black land pixels]. Open the formula document ***Pathfinder_SST.frm***. Study the formula and note that because only one image is open *Bilko* knows that this image must be @1.

The formula is not valid for land and cloud areas (which are set to 0), so the IF statement checks if each pixel is 0 and, if it is, sets the output image to 0. If it isn't (ELSE), the formula converts the DN value of the pixel to a temperature in degrees Celsius. Such "conditional" statements are covered in detail in section 5 of this tutorial. [*Note:* If more than one image were open then a connected images window would need to have been set up using **Image, Connect** so that the program knew which image was designated @1.]

Activity: Use the **Options!** menu (which appears when the formula is the active window) to set the **Output Image Type:** to 32-bit floating point so that decimal degrees Celsius values for temperature can be displayed. [Ignore the other settings for now.] Copy the formula by clicking on the Copy button and then click on the image and paste the formula on it by clicking on the Paste button. A new image will be generated with pixel values equal to the sea surface temperature in °C.

The new image is too dark and needs to be stretched to see it properly. Right-click on it and select **Redisplay** from the menu. In the **Redisplay Image** dialog box select check the **Null Values(s):** box to set null value as zero and select Equalize as the stretch to be used. Check out the new image, noting how all land and cloud areas are represented as a "?" on the Status Bar, whereas sea areas show SSTs in degrees Celsius. To check the range of temperatures, open a histogram of the new SST image.

Question 1: Using the histogram, find out what are the upper and lower values of the "bin" with the highest temperature in the sea (to one decimal place). How many pixels have this temperature?

Question 2: Using the histogram, find out what the modal (most common) temperature (to two decimal places) of tropical oceans was in December 2000. [*Hint:* Find the value of the most frequent "bin" at the warmer end of the SST distribution.] How many pixels are in this temperature "bin"?

Activity: When finished, close all images and the formula document.

3. Functions

The following functions are supported within Formula documents. The "argument" that you wish the function to act on is enclosed in parentheses. For the trigonometric functions (sin, cos, tan and arctan) the argument is in **radians** (not degrees). 180° = π radians so you can convert degrees to radians using the equation: $\text{Radians} = \frac{\text{Degrees} \times \pi}{180}$.

Function	Formula document syntax
Square root	SQRT()
Logarithm to the base 10	LOG()
Logarithm to the base *e* (Natural log)	LN()
Exponential (*e* to the power of)	EXP()
Sine	SIN()
Cosine	COS()
Tangent	TAN()
Arctangent	ATAN()

A few functions which may be of use in remote sensing (e.g. in radiometric correction or bathymetric mapping) and which can be easily derived from these basic functions are listed below.

Function	Formula document syntax
Secant	1 / COS()
Cosecant	1 / SIN()
Cotangent	1 / TAN()

4. Setting up of constants

Constants, like comments, make formulae easier to understand, particularly complex formulae, and can make formulae much easier to alter so that they can be applied to different images. Constant statements should be inserted at the start of formula documents. The form of the statement is:

CONST name = value ;

Where the keyword CONST alerts the program processing the formula document to the fact that this is a constant declaration. The name can be any string of alphanumeric characters (but beginning with a letter), that signifies what the constant refers to; the value is an integer or decimal number that you wish to be assigned to the constant name. The constant statement assigns the value to the name so that in formula you can use the name instead of typing in the value. If the constant is a long string of numbers like -1.0986543 and appears lots of times in the formula document, or you wish to experiment with several different values for a constant, having only to change one constant statement can be very useful. Some examples of constant statements are given below.

```
CONST Lmin = -0.116 ;          CONST Lmax = 15.996 ;
CONST pi =3.14152 ;
CONST d_squared = 1.032829 ;
CONST SunZenithAngle = 32 ;
CONST ESUN = 195.7 ;
#
#   Converting L to exoatmospheric reflectance (ER) with formula of the type:
#   pi * L * d² / (ESUN * cos(SunZenithAngle)) ;
    @2 = pi * (Lmin + (Lmax/254 - Lmin/255)*@1) * d_squared / (ESUN *
    COS(SunZenithAngle/180*pi)) ;
```

In the example above, involving radiometric correction, the values of Lmin, Lmax and ESUN vary for different wavebands. Just changing three of the constant values allows the same formula to be used for several different wavebands, minimising the chance of an error. Note also that more than one statement can be typed on one line.

You can also use constant statements to set up labels for images. For example,

CONST AVHRR_band4 = @1 ;
CONST AVHRR_band5 = @2 ;

This assigns a pair of connected images (@1 and @2) to two constant names which make it clear that the formula is designed to work on thermal infra-red bands #4 and #5 of AVHRR scenes which are used to calculate sea surface temperature. [Note that you cannot use the # symbol for bands in formulae as it is reserved for comments].

Using constants improves the understandability of formula documents. They also allow you to more easily apply a complex formula to a set of images; only having to change the constant statement assignments at the beginning to use with different connected images.

Constant names may be made up of letters (uppercase and lowercase), numbers and the underscore character. However, they may not begin with an underscore.

5. Conditional statements

To illustrate the use of conditional (IF ... ELSE ...) statements you will look at how they could be used to make a "land mask" from ***AVHRR_Mulls#02.bmp***. By land mask we mean an image made up of only zeros and ones (a Boolean image) with zeros lying over land and ones lying over water. If you wanted to process water areas separately in ***AVHRR_Mulls#02.bmp*** or ***AVHRR_Mulls#04.bmp*** you would just have to multiply these images by the land mask to set all land areas to 0 while leaving water pixels as they were.

Sea pixels in ***AVHRR_Mulls#02.bmp*** have values of 45 or less. Land pixels have values in excess of 45. We will set up this threshold value as a constant called "Threshold".

The following conditional statement which has one equality comparison (==), one inequality (<), one logical (OR) and two assignment operations (=) makes a "land mask" from ***AVHRR_Mulls#02.bmp***.

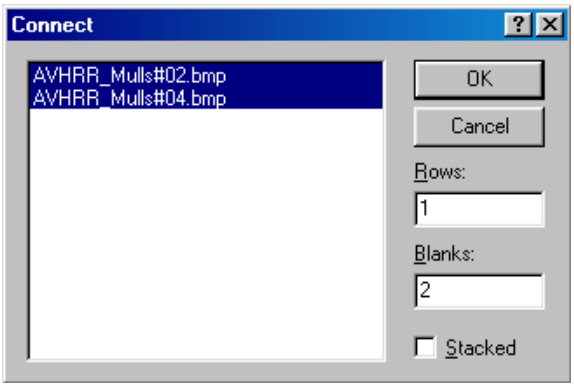
CONST Threshold = 45 ; assigns the value 45 to the constant name "Threshold"

CONST Mulls2 = @1 ; assigns the image designated @1 by the Selector toolbar to the constant name "Mulls2" (this should be ***AVHRR_Mulls#02.bmp***)

CONST Landmask = @3 ; assigns the image designated @3 by the Selector toolbar to the constant name "Landmask" (this will be the output image)

IF (Mulls2 == Threshold OR Mulls2 < Threshold) Landmask = 1
ELSE Landmask = 0 ;

The statement says: IF the pixel value in @1 image (= ***AVHRR_Mulls#02.bmp***) is equal to the constant **Threshold** (= 45) or is less than the constant **Threshold**, then assign a value of 1 to the corresponding pixel in @3 (the blank image to which the name **Landmask** has been assigned) ELSE assign a value of 0 to the @3 image. Thus all sea pixels (equal to or below the threshold) are set to 1 and land pixels (above the threshold) are set to zero.



Activity: Open the files ***AVHRR_Mulls#02.bmp*** and ***AVHRR_Mulls#04.bmp***. Use **Image, Connect** to bring up the **Connect** dialog box. Connect the two files and set the **Blanks:** box to **2** to add two blank images to the connected images window. Use the **Selector** toolbar to designate ***AVHRR_Mulls#02.bmp*** as image 1 (and thus @1 in formulas), ***AVHRR_Mulls#04.bmp*** as image 2 (and thus @2 in formulas) and the two blanks as images 3 and 4 (@3 and @4 in formulas).

Click on **File, New** and create a new Formula document. Enter the three constant assignment statements and the conditional statement described above into the formula document. Use the **Options!** menu to set the **Output Image Type:** to Same as @1, if need be. Then copy and paste the completed formula to the connected tiled images window and the first blank will turn black as it becomes the land mask. Apply an automatic linear stretch to the connected tiled images (remember to select all *(<Ctrl>+A*) of the @1 image first) to see the mask, which should look like the image above. Check that the land is set to zero and the sea to 1 (underlying image data values, not display values) in the land mask image.

Note: To modify the formula so that the mask is produced as a new image not in the connected images window, the formula would be modified thus:

```
CONST Threshold = 45 ;
CONST Mulls2 = @1 ;
IF (Mulls2 == Threshold OR Mulls2 < Threshold) 1
ELSE  0 ;
```

Thus if no image *@n* is assigned for the output of the Formula document then a new image is created to receive it. **Do not do this now**, although please feel free to experiment later!

The next step is to apply this land mask (which is most easily generated from the near infra-red band #2 image due to sharp land:sea boundary) and apply it to the ***AVHRR_Mulls#04.bmp*** image so that the sea surface temperature information can be analysed separately. To do this, two lines (see below) should be added to the formula document:

```
CONST Mulls4 = @2 ;
```

and

```
@4 = Mulls4 * Landmask ;
```

Activity: Insert the new constant assignment after the "**CONST Mulls2 = @1 ;**" statement and insert the second statement, which multiplies the AVHRR band #4 image by the land mask, at the end of the formula document. Once you've checked the new lines carefully to make sure there are no errors, copy the updated formula and paste it to the connected tiled images window. Check the fourth tiled image, which should now contain a masked band #4 image, to make sure that the land has now indeed been set to 0.

[*Note:* If you wanted to save a copy of the masked band #4 image (@4), you would need firstly to select all (*<Ctrl>+A*) of the fourth tiled image and then select **File, Save**].

Activity: When you have finished, close all the image files. You may wish to save the formula document for reference.

Question 3: How could you simplify the conditional statement in the formula (refer to the table of operators in section 2 above)?

6. Bit-wise operators for utilising "flag" information

Images from many sensors may come with "flag" data, which are usually in the form of an image that contains flag attributes for each pixel. Examples are the AVHRR (Advanced Very High Resolution Radiometer) sensor of the United States' National Oceanographic and Atmospheric Administration (NOAA), and the MERIS (Medium Resolution Imaging Spectrometer) and AATSR (Advanced Along Track Scanning Radiometer) sensors of the European Space Agency's ENVISAT satellite. The flag attributes indicate such things as where there is land, water, cloud, coastline, sun glint, invalid data, etc. on the image.

Activity: Open the file ***MER_RR__1_Mediterranean.n1,*** which is a MERIS image of the central Mediterranean around Italy. Click on the ***Flag Codings*** folder in the left hand pane of the hierarchical file window and then double-click on the ***I1_flags*** text file. This opens and tells you what the 8 flag codes (attributes) are, that can be applied to each pixel in this dataset. In this instance, since there are only 8 flags, the flag information can be stored as an 8-bit unsigned integer image; each flag being represented by one bit (Table 10.1).

Table 10.1. Flag Codings that can be represented by an 8-bit unsigned integer. Each of the 8 bits of each pixel's flag codings can be set to 1 to represent a different attribute.

	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Power of 2	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Value	128	64	32	16	8	4	2	1

Activity: Click on the ***Bands*** folder in the left pane. Note that the Data type of the ***I1_flags*** image folder in the right pane is 8-bit unsigned integer and that the flags image is the same size (1121 x 1121 pixels) as the 15 radiance bands of the MERIS image; that is, there is one flag coding for each pixel of the MERIS scene. Right-click on the ***I1_flags*** image folder and select **Open Properties**. The MERIS Level 1b classification and quality flags are listed. These are displayed in Table 10.2 below.

Table 10.2. Level 1b classification and quality flags for MERIS imagery.

Flag descriptors	Value	8-bit unsigned integer (binary)	Binary digit (bit) setting	Result
COSMETIC	= 1	00000001	Bit 1 set	$2^0 = 1$
DUPLICATED	= 2	00000010	Bit 2 set	$2^1 = 2$
GLINT_RISK	= 4	00000100	Bit 3 set	$2^2 = 4$
SUSPECT	= 8	00001000	Bit 4 set	$2^3 = 8$
LAND_OCEAN	= 16	00010000	Bit 5 set	$2^4 = 16$
BRIGHT	= 32	00100000	Bit 6 set	$2^5 = 32$
COASTLINE	= 64	01000000	Bit 7 set	$2^6 = 64$
INVALID	= 128	10000000	Bit 8 set	$2^7 = 128$
Example combinations				
Land and duplicated		00010010	Bits 2 and 5 set	2 + 16 = 18
Bright and land		00110000	Bits 5 and 6 set	16 +32 = 48
Land and coastline		01010000	Bits 5 and 7 set	16 + 64 = 80

Bit-wise operators
And (&)
Or (|)

Bilko provides you with two bit-wise operators that allow you to utilise this flag information. The key one is the bit-wise "And" operator which is represented by an ampersand "&" sign in formula documents. Bit-wise "Or" is also implemented but will not concern us further here. The flags allow one to mask out areas of the image that are unwanted for various reasons (e.g., contain suspect data, bright areas (sensor saturation) due to such features as thick cloud, are land (not ocean) – if one is studying ocean features, are affected by sun glint, etc.). This is best implemented in formula documents by setting up a series of constants for each flag you wish to utilise and then performing a series of bit-wise "and" operations between these constants (flag settings) and the "flags" image to identify areas affected by each flag. This is best explained using an example.

Activity: Double-click on the ***radiance_2*** image folder in the right pane to open the TOA (Top Of Atmosphere) radiance band 2 MERIS image. [If an **Extract** dialog box appears, uncheck the **Extract** checkbox and click OK.] This brings up the **Redisplay Image** dialog box. Check the **Null Value(s): == 0** checkbox and make a note of the maximum pixel value. Select **Equalize** as the stretch and then click the **Apply** button. Note that the image is in 16-bit unsigned integer format. Briefly inspect the image, which shows Italy, the Adriatic and Croatian coast, and the western Mediterranean around Corsica, Sardinia, Sicily and the Balearic Islands. Now right-click on the ***l1_flags*** image folder and select **Open Items** to open the flags image. Inspect this image using the cursor and answer the following question.

Question 4: In the flags image, what values do (1) clear ocean, (2) most land, and (3) most of the bright (cloud) area in the north-east of the image have? Which flags are set to give these values?

Activity: Connect the flags image, the band 2 radiance image and one blank image as a group of three tiled connected images using the **Image, Connect** menu. Using the Selector toolbar designate the flags image as @1 (image 1), the band 2 radiance image as @2, and the blank image as @3. When this is done, open the formula document ***Bitwise.frm***.

Inspect the formula document. Note that (i) it is expecting the flags image to be @1 and the band 2 radiance image to be @2, (ii) constants have been set up for 6 of the flags, and

(iii) that the output image from the formula will appear in the third tile (@3) of the connected images window.

Let us now examine what the formula does. It seeks to mask out (i.e. set to zero) all areas of the image that (i) have glint risk, (ii) are suspect, (iii) are land, (iv) are bright (i.e. have thick cloud or are causing sensor saturation for some other reason), or (v) are invalid. Each bitwise "and" (denoted by a **&**) operation between a flag constant and the flag image creates a Boolean image (that is, one in which all pixels have values of either 0 or 1), with areas where the flag is present having a value of 1 and those where it is absent having the value 0 (Table 10.3).

Since you want to mask out areas where the flag is present, you want the opposite. To achieve this, each bit-wise "and" is preceded by a NOT operation (in this case using the **!** sign, which can be used interchangeably with NOT in *Bilko* formula documents). This creates a series of masks for each flag such that areas where the flag is present are set to 0, and the remainder of the image is set to 1. Multiplying the band 2 radiance image by this series of masks sets all the unwanted areas to zero.

Table 10.3. Results of bitwise "and" operations for three cases. ***Case 1***: the flags image indicates land with thick cloud (bright); a bitwise "and" with a *bright* flag constant indicates true (i.e. Bit 6 is set in the flags image **and** in the constant) and so the result for that pixel is 1. This is switched to 0 by the NOT (!) operation. ***Case 2***: the flags image indicates land with thick cloud (bright); a bitwise "and" with a *land* flag constant indicates true (i.e. Bit 5 is set in the flags image **and** in the constant) and so the result for that pixel is 1.This is switched to 0 by the NOT (!) operation. ***Case 3***: the flags image indicates open ocean with no problems; a bitwise "and" with a *land* flag constant indicates false (i.e. Bit 5 is **not** set in the flags image) and so the result for that pixel is 0. This is switched to 1 by the NOT (!) operation.

Case	Flags image pixel		Flag constant	Flag descriptor	Result in output image	Result after NOT (!) operation
1	01**1**10000	&	01**1**00000	Bright (32)	0000000**1**	0000000**0**
2	00**1**10000	&	00**1**10000	Land (16)	0000000**1**	0000000**0**
3	00000000	&	00**1**10000	Land (16)	0000000**0**	0000000**1**

Activity: The output image needs to be a 16-bit unsigned integer image like the band 2 radiance image. With the formula document as the active window, select **Options!** from the menu. In the **Formula Options** dialog box, uncheck the **Use special handling for Nulls** checkbox, select 16-bit unsigned integer as the **Output Image Type:** and click on OK. You can now apply the formula to the tiled connected image by copying it and pasting it on the connected images window. Inspect the new image in the third tile of the connected images window and note that all areas with land, with risk of sun glint, with thick bright cloud or with suspect pixels are set to zero and you are only left with data values for water areas.

In the formula document two other ways of achieving the same result are shown as "comments". You may wish to comment out the formula you used and delete the **#** signs before each of the alternative formulae in turn to satisfy yourself that these do the same thing. You will find that the second alternative formula, which uses a series of bitwise "or" operations, works the fastest.

To see exactly where the coastline is on the new image one can draw a bright line to coincide with pixels flagged as coastline. The brightest pixel value in the band 2 radiance image, which you should have noted earlier, was 52547. Let us thus set the coastline to have a value of 60000. To do this you need to add two new statements to the formula document after the alternative formulae. One way of doing this is to add the following two lines:

```
CONST NewCoast = 60000 ;
IF (Flags & Coastline) NewCoast ELSE @3 ;
```

This will create a new image; this will be the @3 image with the coastline added with a value of 60000.

Activity: Add the two new lines to the end of the formula document and rerun the formula. Inspect the new image. You will find that the sea areas are very lacking in contrast and uniformly grey. To rectify this, right-click on the image and select **Redisplay**. In the **Redisplay Image** dialog box select Equalize as the stretch to use and apply the stretch. This greatly improves the image.

When you have finished, close all the image files. You may wish to save the formula document for reference.

Note that some images may come with more sophisticated sets of flags. For example, several ENVISAT image products use 16 flags such that 16-bit unsigned integer data types are required for the flag images. Some even have 24 flags and need 32-bit unsigned integer data types to represent them. These flags can be utilised using the same methods.

These three examples have taken you through the basics of using formula documents and how to make formulas as understandable as possible both for your own and other people's use. Formulas can be used to perform elements of radiometric correction, perform band ratios, calculate vegetation indices, utilise flag data, carry out water column correction and a whole host of both simple and sophisticated manipulations of image data. These capabilities are further explored in various lessons. A number of advanced features and functions have not been covered here but will be covered in a mini-lesson on advanced use of formulas.