HyperLogic
TECHNICAL NOTES
A series of informal papers on technical topics for customers and friends of HyperLogic Corporation
Number 3
Revision 3

Extending Microsoft Excel with CubiCalc RTC's Fuzzy DLL

Excel, the popular Windows-based spreadsheet from Microsoft, lets you call functions in external Dynamic Link Libraries. This technical note discusses using CubiCalc RTC's fuzzy inference run-time DLL with Excel. The article is primarily for current CubiCalc RTC users, but it provides background information so that potential users of RTC can follow it too. Readers should be familiar with Excel.

Windows Dynamic Linking

Dynamic link libraries are files containing software procedures that can be invoked from separate, independent programs. The name refers to way stand-alone programs link to the DLL functions "on the fly."

The three critical pieces of information Excel needs to call a DLL are the (1) DLL file name, (2) the name of the function within the DLL, and (3) the data types of the function arguments and its return value. You provide all three as arguments to either the Excel CALL function or the REGISTER function. This tells Excel *how* to call the function; you must also provide the DLL function arguments so Excel knows *what* to pass to it.

The file name is just a standard file name string. The function name is also a character string. The data types are represented by a string of characters, one for the return value and one for each argument.

CALL presents everything in one long expression; it says both how to call the function and what values to pass. REGISTER provides just the linkage information, which Excel remembers to use when you later call the function. Once a function is registered, you can use it as you would a built-in Excel function.

This would be a good time to read about CALL and REGISTER in your Excel documentation. We'll show some examples later in this article.

CubiCalc RTC's Run-Time DLL

CubiCalc RTC includes a dynamic link library version of its fuzzy inference engine. The DLL contains callable functions for evaluating a fuzzy rule base. It reads the rule base from a file created by the run-time generator.

Although other functions are included in CBCRUN20.DLL, only three are of interest here.

The *wfz_init* function takes as its argument a string containing the name of the rule base definition file. It returns an integer value; if nonnegative, it is a "channel number" or ID for the rule base. If negative, it is an error number. You can load multiple rule bases in the DLL, so the channel number lets you distinguish the different ones. You use the channel number in other calls.

The *wfz_close* function takes a single argument, the channel number identifying the rule base. This function frees up memory that was allocated for that rule base.

The *wfz_eval* function takes two arguments. The first is the address of an array of IEEE 8-byte floating point numbers, one for each fuzzy input variable. The second argument is a similar array for fuzzy outputs. When you call *wfz_eval*, the DLL evaluates the rule base for the input values and returns the results.

Unfortunately, when multiple fuzzy variables are used in the rule base, none of Excel's data representations correspond exactly to the calling convention for *wfz_eval*. To translate, HyperLogic has created CBXRUN20.DLL.

The first new function is a simple translator. Called *wfz_xeval*, it has exactly the same calling convention as *wfz_eval* but formats its array arguments the way Excel wants them.

The second is an all-in-one function, *wfz_all*, to initialize the rule base, evaluate the rules, and close the rule base, returning the output array. Its calling convention is similar to that of *wfz_xeval* and *wfz_eval*, but instead of a channel number it takes a file name string as its first argument. When you use *wfz_all*, you don't need to call *wfz_init* or *wfz_close.*

Excel's Argument Representation

Excel uses a set of single-letter codes to specify the argument types for an external function. There is an argument called *type_text* in the CALL and REGISTER functions. The first character of the *type_text* string describes the function's return value; successive characters describe the data types for the function's arguments.

Table 1 lists *type_text* strings for functions in CBCRUN20.DLL and CBXRUN20.DLL. As documented in the RTC manual, many of the functions formally return an integer-valued status or error code, but the strings in Table 1 are set up to return the requested data, such as the output array, a weight, or an activation value. Excel allows only a single return value, so you can't get back both the data and the error code at once.

Single-Output Rule Bases

If your CubiCalc project uses only one output variable, you can call either *wfz_all* or *wfz_xeval* as if it

wfz_close	II
wfz_eval	3IEE
wfz_init	IC
wfz_get_activation	3IIE
wfz_get_index	4IICM
wfz_get_weight	3IIE
wfz_set_weight	3IIB
wfz_xeval*	3IKK
wfz_all*	3FKK

Table 1. type_text strings

* These are in CBXRUN20.DLL.

```
=CALL("CBXRUN20.DLL", "wfz_all", "3FKK", "ORDER1.CBD",
 A1:A3, 0)
```

Example 1. Single-output rule base with CALL

```
=REGISTER("CBXRUN20.DLL", "wfz_all", "3FKK")
                 ...
=wfz_all("ORDER1.CBD", A1:A3, 0)
```

Example 2. Single-output rule base with REGISTER

```
=CALL("CBXRUN20.DLL", "wfz_all", "3FKK","ORDER2.CBD",
 A1:A3, {0,0})
```

Example 3. Multiple-output rule base using CALL
Type this in with multiple cells selected, then press CTRL+SHIFT+ENTER

```
=INDEX(CALL("CBXRUN20.DLL", "wfz_all", "3FKK",
 "ORDER3.CBD",A1:A3,{0,0,0}), 3)
```

Example 4. Obtaining the Third Output with INDEX

returned just one value. Excel automatically converts the array to a single number.

Suppose you've created a rule base with three inputs and a single output. You can invoke the all-in-one evaluation function by entering the formula shown in Example 1. (You might need to add directory information to the DLL and data file name argument strings.)

Another way to call the function is to register it first. A good way to do this is to run a macro that registers all the DLL functions. Such a macro is provided with the distribution for CBXRUN20.DLL. Example 2 shows the registration and a subsequent use of *wfz_all*.

Multiple-Output Rule Bases

If your CubiCalc project uses multiple fuzzy outputs, the rule base result is an array of values, one for each fuzzy output variable. The returned output values occupy multiple cells in the sheet.

To enter the function, first select a range of cells, one cell per fuzzy output. The range can be either horizontal or vertical. With that range still selected, type in a formula like that shown in Example 3. Finish by pressing the key combination CTRL+SHIFT+ENTER. This tells Excel that you have entered an array formula.

If you've done everything correctly, Excel will enclose the formula in braces to indicate that it is an array. The values in the cell range will be the output values from the fuzzy rule base evaluation.

You can also retrieve just a single value from the output array using the INDEX function. Example 4 shows how. The first argument to INDEX is the returned array from the CALL to *wfz_all*; the second argument is a 3, which says to select the third element of the array.

Rule Default Values

Notice the last argument passed to *wfz_all* in the examples. We've used zero or an array of zeroes, one for each output, in every case. The zeroes are default output values. You should use whatever is appropriate for your own problem.

Default values for the outputs are used whenever the rule base has no rules to handle the input situation. Because of the way fuzzy rules are computed, the fuzzy output is completely undefined if none of the rules apply. There is no way to for the fuzzy engine to "guess" what the result should be, so you must provide a default value.

Improving Performance

These examples all used *wfz_all*. Although it's okay once or twice, it may be too slow if your worksheet uses *wfz_all* in many cells.

To initialize a rule base, the inference engine must read it from a disk file and perform various computations and memory allocations; this is a time-consuming process. A better approach uses the *wfz_xeval* function for evaluating the rules. An example provided with the CBXRUN20.DLL distribution has a macro sheet showing how to do it.

Summary

With Microsoft Excel and the RTC DLL, even non-programmers can create stand-alone fuzzy rule-based applications.

Registered CubiCalc RTC users can get CBXRUN20.DLL from HyperLogic technical support.