Aqwa AQL Manual

Ansys
ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
http://www.ansys.com
(T) 724-746-3304
(F) 724-514-9494

Release 2024 R1
January 2024

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

Copyright and Trademark Information

Disclaimer Notice

U.S. Government Rights

Third-Party Software

Published in the U.S.A.

Table of Contents

Chapter 1: Overview

AQL is an interface from Aqwa to Excel. It is intended to facilitate the development of Excel spreadsheets that require data from Aqwa databases. The interface consists of a series of Excel functions that can be accessed within the spreadsheet to recover data directly from the Aqwa database. A typical example might be an application that tabulates motions of a particular structure and then undertakes a post-processing exercise.

Key features are:

- Direct access to the Aqwa database information using Excel Function calls
- Recovery of RAOs using structure number, frequency and direction
- Recovery of time history results
- Error trapping
- Simple installation

Chapter 2: Compatibility

AQL is available for Excel variants (32-bit or 64-bit) designed to run under Windows 10. The interface program does not require an Ansys license to operate.

Chapter 3: Installing AQL

Installation of AQL is straightforward. It is assumed that the reader is reasonably familiar with Windows. The notes in this section relate to Excel 2016. Earlier or later versions may vary slightly from the suggested installation.

3.1. Modifying your Excel Installation

Different files are required for 32-bit or 64-bit versions of Excel. Note that it is quite common to have a 32-bit version of Excel installed on a 64-bit computer. The AQL files used should correspond to the Excel version.

32-bit

The required files will be found in the directory `\ANSYS Inc\v241\aqwa\utils\win32\aql32.dll` and the supporting DLLs `libifcoremd.dll`, `libifportmd.dll`, `libmmd.dll`, `svml_dispmd.dll` must be copied to the same directory as the MS Excel executable EXCEL.EXE (for example `C:\Program Files (x86)\Microsoft Office\root\Office16\`).

64-bit

The required files will be found in the directory `\ANSYS Inc\v241\aqwa\utils\winx64\aql64.dll` and the supporting DLLs `libifcoremd.dll`,`libifportmd.dll`, `libmmd.dll`, `svml_dispmd.dll` must be copied to the same directory as the MS Excel executable EXCEL.EXE (for example `C:\Program Files\Microsoft Office\root\Office16\`).

3.2. Configuring Excel

For 32-bit, the `aql32.xla` file is best placed in:

```
Program Files (x86)\Microsoft Office\root\Office16\Library
```

For 64-bit, the `aql64.xla` file is best placed in:

```
Program Files\Microsoft Office\root\Office16\Library
```

Once installed, it is necessary to set up the location in Excel: navigate to File>Options>Add-ins, select Aql32/64 in the table and then select Go. If a spreadsheet that already has calls to AQL in it is copied from another computer, the spreadsheet may not run properly on the new computer if AQL is installed in different directories on the two machines.

Chapter 4: Running AQL

AQL is reasonably simple to use as follows:

1. Start Excel as usual
2. If the `aql32/64.xla` worksheet file is not auto-loaded at startup, open the `aql32/64.xla` file (using the normal Excel File Open command)
3. Use the AQL functions in a standard Excel worksheet

4.1. Using AQL Functions

When you load the `aql32/64.xla` file is loaded nothing appears on the screen; this is normal. If the file is not loaded, any worksheet formulae using AQL functions will display as #REF!. If a function is used and #VALUE! is displayed, text has probably been entered for an argument which should be a number or vice versa. If #N/A appears, one of the arguments has probably been omitted.

AQL functions are included in formulae just like ordinary Excel functions (such as sine, sum, max, etc.). If you cannot remember the order or meaning of the arguments to the function, use Excel's function wizard - the AQL functions are listed in a category called AQWA-Excel32/64.

A simple example is shown below:

```
=AQLposition("C:\Alpha\adtest01”, 1, z, 100)
```

This function returns the Z displacement at time-step 100 for structure number 1, in model adtest01, stored in directory `C:\Alpha`. The directory and model name (in this case "C:\Alpha\adtest01") are collectively referred to as model in the function descriptions given in Function Descriptions (p. 11)

To avoid typing such long formulae, use absolute and relative row and column references in Excel as much as possible. This is illustrated in the example below where you require the displacement of one structure in all six degrees of freedom for every thousand time-steps. The directory and the model name are entered into cells A1 and A2 (splitting these up makes changing to a different model easy). These items are then concatenated into a single string in cell A3. The number of the structure is entered into cell B4. The six degrees of freedom, namely X, Y, Z, RX, RY and RZ are entered into cells B5, C5, D5, E5, F5 and G5. The time-steps are entered into cells A6, A7, A8, A9, A10, A11 and A12.

The full formula for displacement in cell B6 would be:

```
=AQLposition("C:\Alpha\adtest01”, 1, x, 1)
```

Which is entered using cell references as:

```
=AQLposition(A3,B4,B5,A6)
```

This is much shorter and can be entered by clicking on the appropriate cells. However, if this formula is copied from cell B6 to cell C7, it becomes:

=AQLposition(B4,C5,C6,B7)

This is incorrect. The required formula in cell C7 is:

=AQLposition(A3,B4,C5,A7)

Column and row numbers in a formula can be made absolute by prefixing them with dollar signs. Therefore the formula in cell B6 can be rewritten as:

=AQLposition(A3,B4,B$5,$A6)

Which when copied to C7 gives:

=AQLposition(A3,B4,C$5,$A7)

Which, as required, evaluates to:

=AQLposition("C:\Alpha\adtest01”,1,”Y”,1000)

The use of named cells may also be helpful when requiring invariant data access. By naming cell A3 'model', use can be made of this name in the function calls and it behaves as though giving an absolute address. Thus 'model' is synonymous with A3.

The formula in cell B6 becomes:

=AQLposition(model,B4,B$5,$A6)

The formula in cell B6 can be copied over the whole of the table B6:G12 to give the required table as follows:

	A:	B:	C:	D:	E:	F:	G:
1:	C:\ALPHA\						
2:	ADTEST01						
3:	=(A1&A2)						
4:	Structure Number	1					
5:	Time-Step	X	Y	Z	RX	RY	RZ
6:	1	=AQLposition(model,B4,B$5,$A6))					
7:	1000						
8:	2000						
9:	3000						
10:	4000						
11:	5000						
12:	6000						

4.2. Example

There is an example spreadsheet in the training directory that is included with the standard Aqwa installation (`ANSYS Inc\241\aqwa\training\`). This spreadsheet displays several results from the "takbuy" model in the training directory. It will probably be necessary to modify the path in cell B4. You will need to run the file `Run_Training.bat` (found in the training directory) to perform a simple Aqwa analysis before trying to view the spreadsheet `takbuy.xls`.

Note:

Run_Training.bat can be run from the command line. You may need to move the "training" folder to another location on your computer if you get a write permission error when you try to execute **Run_Training.bat**. If you move the training folder, rather than executing **Run_Training.bat**, execute the following command from the command line while in the training folder:

```
"C:\Program Files\Ansys Inc\v241\aqwa\bin\winx64\Aqwa.exe" std
training.com
```

Chapter 5: AQL Functions

The functions are designed to be relatively straightforward to use. All arguments are in terms of user numbers (structure number, frequency number, etc.) rather than Aqwa internal numbers. The AQL interface requires no special configuration, no start up macros, etc.

5.1. Functional Interface

The Aqwa interface for Excel consists of two main parts:

- A dynamic link library called `aql32/64.dll`
- An Excel add-in macro sheet called `aql32/64.xla`

The dynamic link library (or DLL) is a set of functions written in the FORTRAN language. These functions understand enough about the Aqwa file system to extract information from it. They do all the hard work.

The add-in macro sheet provides a set of spreadsheet functions that use the dynamic link library to access the Aqwa file system. These spreadsheet functions can be used like any other functions in an Excel spreadsheet (or 'worksheet' in Microsoft's terminology). All the functions have names starting with the prefix *AQL*.

5.2. Function Descriptions

All the function information in this section can be obtained using the help button from the Function Wizard in Excel. The list of functions below is complete at the time of writing, but more functions will be added when necessary.

FUNCTIONS	DESCRIPTION
Aqlcogcoord	Returns x, y or z coordinate of the center of gravity of a structure
Aqlcoord	Returns x, y or z coordinate of a node
Aqlelcent	Returns x, y, or z coordinate of the centroid of a diffracting element
Aqlelcent_moonpool	Returns x, y, or z coordinate of the centroid of a diffracting element on a moonpool
Aqlelpres	Returns pressure at element centroid
Aqlelpres_moonpool	Returns pressure at element centroid on a moonpool
Aqlfrequency	Returns wave frequencies analyzed in Aqwa-Line
Aqlglobal	Returns global parameters
Aqlmessage	Returns an error message for a given aql error flag number
Aqlndirns	Returns the number of wave directions for a structure
Aqlnfreqs	Returns the number of wave frequencies for a structure

FUNCTIONS	DESCRIPTION
Aqlnlines	Returns the number of mooring lines in a model
Aqlnmoonpools	Returns the number of moonpools in a model
Aqlnstructs	Returns the number of structures in a model
Aqlposition	Returns the position of a structure for a time history analysis
Aqlrao	Returns RAO information from the hydrodynamic database
Aqlrao2	Returns interpolated RAO information from the hydrodynamic database
Aqlstatmooring	Returns mooring line information for a static (Aqwa-Librium) analysis
Aqlstatposcog	Returns position of the CoG for a static (Aqwa-Librium) analysis
Aqlstatposnod	Returns position of a specified node for a static (Aqwa-Librium) analysis
Aqlthacccog	Returns the acceleration of the cog for a time history analysis
Aqltharticulation	Returns articulation reactions for a time history analysis
Aqlthfender	Returns fender information for a time history analysis
Aqlthmooring	Returns mooring line information for a time history analysis
Aqlthnumsteps	Returns number of time steps for a time history analysis
Aqlthposcog	Returns the position of the CoG for a time history analysis
Aqlthtime	Returns the time associated with a time step in a time history analysis
Aqlthvelcog	Returns the velocity of the cog for a time history analysis
Aqlwavedirn	Returns wave directions analyzed in Aqwa-Line
Aqlzcge	Returns the ZCGE parameter for a structure in an Aqwa-Line analysis

Aqlcogcoord

Purpose:	Returns the intial coordinate of the center of gravity
Syntax:	aqlcogcoord(model,structure,freedom)
Notes:	Only one coordinate is returned according to the freedom specified
Example:	aqlcogcoord(Model,1,x) returns the x coordinate of the center of gravity of structure 1 in the model specified in the name cell Model

Aqlcoord

Purpose:	Returns the initial coordinate of a node
Syntax:	aqlcoord(model,structure,node,freedom)
Notes:	Only one coordinate is returned according to the freedom specified
Example:	aqlcoord(Model,1,101,x) returns the x coordinate of the element 101 on structure 1 in the model specified in the named cell Model.

Aqlelcent

Purpose:	Returns the initial coordinate of the centroid of a diffracting element
Syntax:	aqlelcent(model,structure,element,xsym,ysym,freedom)

Notes:	Only one coordinate is returned according to the freedom specified. Xsym and ysym are used to define the region in a model with symmetry. Use 1 to specify the modelled region, 2 to specify the reflected region
Example:	aqlelcent(Model,1,101,2,1,x) returns the x-coordinate of element 101 on structure 1 in the model specified in the named cell Model. The element will be in the region reflected in the X-Z plane (usually on the starboard side for a conventional ship model with one axis of symmetry and nodes defined with positive y-coordinates)

Aqlelcent_moonpool

Purpose:	Returns the initial coordinate of the centroid of a diffracting element on a moonpool
Syntax:	Aqlelcent_moonpool(model,structure,moonpool,element,xsym,ysym,freedom)
Notes:	Only one coordinate is returned according to the freedom specified. moonpool is the global sequence number of the required moonpool, element is the sequence number of element among the elements of the required moonpool. xsym and ysym are used to define the region in a model with symmetry. Use 1 to specify the modelled region, 2 to specify the reflected region
Example:	Aqlelcent_moonpool(Model,1,2,101,2,1,x) returns the x-coordinate of element 101 on the moonpool of global number of 2 associated with structure 1 in the model specified in the named cell Model. The element will be in the region reflected in the X-Z plane (usually on the starboard side for a conventional ship model with one axis of symmetry and nodes defined with positive y-coordinates)

Aqlelpres

Purpose:	Returns the pressure at the centroid of an element
Syntax:	aqlelpres(model,structure,direnum,freqnum,element,xsym,ysym,component,amp/pha)
Notes:	xsym and ysym are used to define the region in a model with symmetry. Use 1 to specify the modelled region, 2 to specify the reflected region. Valid components are Hydrostatic, Incident, Diffraction, Radiation, Total. The names can be abbreviated to the first 3 letters. Hydrostatic is the dynamically varying component of hydrostatic pressure caused by the motion of the ship
Example:	aqlelpres(Model,1,2,5,101,2,1,Tot,Amp) returns the amplitude of the total pressure at the centroid of element 101 on structure 1 in the model specified in the named cell Model, for the 2nd direction and the 5th frequency

Aqlelpres_moonpool

Purpose:	Returns the pressure at the centroid of an element on a moonpool
Syntax:	aqlelpres_moonpool(model,structure,moonpool,direnum,freqnum,element,xsym,ysym,component,amp/pha)
Notes:	xsym and ysym are used to define the region in a model with symmetry. Use 1 to specify the modelled region, 2 to specify the reflected region. Valid components are Hydrostatic, Incident, Diffraction, Radiation, Total. The names can be abbreviated to the first 3 letters. Hydrostatic is the dynamically varying component of hydrostatic pressure caused by the motion of the ship
Example:	Aqlelpres_moonpool(Model,1,2,2,5,101,2,1,Tot,Amp) returns the amplitude of the total pressure at the centroid of element 101 on the moonpool of global number of 2 associated with structure 1 in the model specified in the named cell Model, for the 2nd direction and the 5th frequency

Aqlfrequency

Purpose:	Returns the frequencies utilized in the radiation/diffraction analysis
Syntax:	aqlfrequency(model,structure,freqnum)
Notes:	freqnum is the frequency identifier as specified in the Aqwa-Line analysis

Aqlglobal

Purpose:	Returns a global parameter
Syntax:	aqlglobal(model,parameter)
Notes:	Valid parameters are DPTH, DENS, ACCG for extracting water depth, water density, acceleration due to gravity

Aqlmessage

Purpose:	Returns the definition of an AQL error code
Syntax:	aqlmessage(Errorcode)
Example:	aqlmessage(#AQL:101) returns the error message connected with AQL error number 101

Aqlndirns

Purpose:	Returns the number of wave directions for a structure
Syntax:	aqlndirns(model,structure)

Aqlnfreqs

Purpose:	Returns the number of wave frequencies for a structure
Syntax:	aqlnfreqs(model,structure)

Aqlnlines

Purpose:	Returns the number of mooring lines in a model
Syntax:	aqlnlines(model)

Aqlnmoonpools

Purpose:	Returns the number of moonpools in a model
Syntax:	aqlnmoonpools(model)

Aqlnstructs

Purpose:	Returns the number of structures in a model
Syntax:	aqlnstructs(model)

Aqlposition

Purpose:	Returns the position of the center of gravity of a structure at a given time
Syntax:	aqlposition(model,structure,freedom,timestep)
Notes:	Only applies to time domain programs, Naut and Drift. Freedom is in terms of the Fixed Reference Axis. Timestep is the timestep number rather than the actual time
Example:	aqlposition(Model,1,x,2) returns the x-coordinate of the center of gravity of structure 1 at timestep number 2 in the model specified in the named cell Model

Aqlrao

Purpose:	Returns RAO data
Syntax:	aqlrao(model,structure,direnum,freqnum,freedom,Amp/Phase)
Notes:	direnum is the wave direction number rather than the actual wave angle
Example:	aqlrao(Model,1,2,5,rx,Amp) returns the amplitude of the roll RAO of structure 1 in waves of frequency no. 5 and direction no. 2

Aqlrao2

Purpose:	Returns interpolated RAO data.
Syntax:	aqlrao2(model,structure,dire,freq,freedom,Amp/Phase)
Notes:	This function interpolates between data points in the Aqwa database. *Dire* is the wave angle (degrees) and *freq* is the wave frequency (rad/s). Compare with **aqlroa** above.
Example:	aqlrao2(Model, 1, 15, 0.35, x, Amp) returns the RAO based motion of structure 1 in the x direction, in waves with a frequency of 0.35 rad/s and direction 15 deg.

Aqlstatmooring

Purpose:	Returns the mooring line information from a static (Aqwa-Librium) analysis.
Syntax:	aqlthmooring(model,structure,configuration,spectrum,line,freedom)
Notes:	Valid freedoms are X, Y, Z, tension, laid length, and uplift. Long freedom names can be abbreviated to the first four letters. Requires the .PLD file for the model. Aqwa-Librium does not produce a .PLD file so it is necessary to open the AGS and read in the .PLT file. A .PLD file will then be generated automatically.
Example:	aqlstatmooring(Model,1,2,2,4,TENS) returns the tension in line 4 connected to structure 1 for mooring configuration no.2 and spectrum no.2.

Aqlstatposcog

Purpose:	Returns the final position of the center of gravity from a static (Aqwa-Librium) analysis
Syntax:	aqlthmooring(model,structure,configuration,spectrum,freedom)
Notes:	Valid freedoms are X, Y, Z, RX, RY, RZ. Requires the .PLD file for the model. Aqwa-Librium does not produce a .PLD file, so it is necessary to open the AGS and read in the .PLT file. A. PLD file will then be generated automatically.
Example:	aqlstatposcig(Model,1,2,2,RX) returns the angle about the X-axis of the center of gravity of structure 1 for mooring configuration no. 2 and spectrum no. 2.

Aqlstatposnod

Purpose:	Returns the final position of a specified node from a static (Aqwa-Librium) analysis.
Syntax:	aqlthmooring(model,structure,configuration,spectrum,node,freedom)
Notes:	Valid freedoms are X, Y, Z, RX, RY, and RZ. Output for the specified node must have been requested in the Aqwa-Librium run using the NODE card in deck 18. Requires the .PLD file for the model. Aqwa-Librium does not produce a .PLD file so it is necessary to open the AGS and read in the .PLT file. A .PLD file will then be generated automatically.
Example:	aqlstatposnod(Model,1,2,2,548,Y) returns the y-position of node 548 on structure 1 for mooring configuration no.2 and spectrum no.2.

Aqlthacccog

Purpose:	Returns the acceleration of a structures center of gravity at a given time step in a time history analysis.
Syntax:	aqlthacccog(model,structure,freedom,timestep)
Notes:	Requires the .PLD file for the model. The freedom refers to the FRA. Valid freedoms are X, Y, Z, RX, RY, and RZ. Note that these do not necessarily refer to surge, sway, heave, etc.

Aqltharticulation

Purpose:	Returns the articulation reactions associated with a given time step in a time history analysis
Syntax:	aqltharticulation(model,structure,articulation,freedom,timestep)
Notes:	Requires the .PLD file for the model. Valid freedoms are FX, FY, FZ, MX, MY, MZ

Aqlthfender

Purpose:	Returns the fender information associated with a given time step in a time history analysis.
Syntax:	aqlthfender(model,structure,line,result,timestep)
Notes:	Requires the .PLD file for the model. Valid results are listed below. Long result names can be abbreviated to the 1st four letters.

Result name	Description
FX	Force in global X axis
FY	Force in global Y axis
FZ	Force in global Z axis
Total	Total force
Compression	Reduction in size of fender
Elastic	Force due to compression
Damping	Damping force
Friction	Friction force
Horizontal	Horizontal movement
Vertical	Vertical movement

Aqlthmooring

Purpose:	Returns the mooring line information associated with a given time step in a time history analysis.
Syntax:	aqlthmooring(model,structure,line,result,timestep)
Notes:	Requires the .PLD file for the model. Valid freedoms are X, Y, Z, tension, laid length, and uplift. Long freedom names can be abbreviated to the 1st four letters.

Aqlthnumsteps

Purpose:	Returns the number of timesteps in a time history analysis
Syntax:	aqlthnumsteps(model)

Aqlthposcog

Purpose:	Returns the position of a structures center of gravity associated with a given time step in a time history analysis.
Syntax:	aqlthposcog(model,structure,freedom,timestep)
Notes:	Requires the .PLD file for the model. The freedom refers to the FRA. Valid freedoms are X, Y, Z, RX, RY, and RZ. Note that these do not necessarily refer to surge, sway, heave, etc.

Aqlthtime

Purpose:	Returns the time (in seconds) associated with a given time step in a time history analysis
Syntax:	aqlthtime(model,timestep)
Notes:	Requires the .PLD file for the model

Aqlthvelcog

Purpose:	Returns the velocity of a structures center of gravity associates with a given time step in a time history analysis.
Syntax:	aqlthvelcog(model,structure,freedom,timestep)
Notes:	Requires the .PLD file for the model. The freedom refers to the FRA. Valid freedoms are X, Y, Z, RX, RY, and RZ. Note that these do not necessarily refer to surge, sway, heave, etc.

Aqlwavedirn

Purpose:	Returns the wave directions utilized in the radiation/diffraction analysis, units are degrees
Syntax:	aqlwavedirn(model,structure,direnum)
Notes:	Direnum is the direction identifier as specified in the Aqwa-Line analysis

Aqlzcge

Purpose:	Returns the ZCGE parameter for a structure in an Aqwa-Line analysis
Syntax:	aqlzcge(model,structure)

5.3. Troubleshooting

When using AQL functions, several different types of problem might arise. These fall into three main groups:

- Errors in function names and arguments
- Errors in values given to a function
- Errors in accessing the Aqwa files

Some problems may cause Excel to display one of its standard error indicators. #NAME? usually means that you have mis-typed the name of the function, or that the `aql32/64.xla` add-in has not been loaded. #N/A usually means that too few arguments have been given to a function. #VALUE! usually means that the wrong type of arguments have been given (such as text where a number is expected) or a wrong value for an argument with a restricted range of valid values (such as "Q" or 8 for a freedom name, where only "X", "Y", "Z", would be acceptable).

Some general suggestions for solving problems are:

- #NAME? - check that the function name is spelled correctly and that the `aql32/64.xla` add-in has been loaded
- #N/A - check that the correct number and type of arguments have been given to the function - use the Formula|Paste Function... command in Excel to check what the arguments mean
- #VALUE! - if the cell contains a formula which does a calculation, the AQL function may be returning an error code and Excel is signaling that it cannot perform arithmetic on text values. If the cell simply calls a function check that arguments which can only take a few possible values, such as axis names, force names, etc. have been correctly specified
- #REF! - this error is generally nothing to do with AQL
- If the Aqwa analysis has been re-run since the spreadsheet was last used, make a null edit of the cell where the Aqwa files are defined (simply put the cursor on the cell, press F2, enter) and Excel will re-calculate as if you had changed the cell (even though you know that you have not). This may clear up problems caused by Excel displaying out-of-date results.
- Many errors in tables come from confusion about absolute and relative cell addresses, so check that a formula that you have copied does reference the appropriate rows and columns for element, node, etc. numbers.

The individual functions in the DLL are designed to be fairly robust so they will not cause Excel or Windows to crash in the event of problems. They only read from Aqwa files, so cannot corrupt any data. However it is not advisable to attempt to read from an Aqwa database file at the same time as an Aqwa program is trying to write to it.

5.4. Restrictions

- Excel sometimes returns spurious error codes from functional calls. All results should be checked by the user.

- There are no facilities for writing information back to the Aqwa files.
- The DLL functions can be accessed directly from Excel using the CALL function. However, this is not recommended because errors in the parameters to CALL can easily lead to Excel or Windows crashing.

5.5. Performance

The AQL functions are designed for convenience, not speed. The functions in the DLL do include some local caching to reduce the amount of file access when a single model is being used. However, as they comply with the general Windows recommendation for not keeping files open between calls (which also means that they do not require initialization or close down and should not cause system problems if the Aqwa files are updated while Excel is running), they do perform a lot of input from the Aqwa file system to retrieve even small amounts of data.

Excel does have an unpleasant habit of performing a full recalculation when not necessary (especially when a row or column is inserted or deleted, or after moves or copies). If a worksheet has a table containing thousands of numbers retrieved by AQL, this can take some time. One way of avoiding the recalculation is to create the table using AQL functions and then replace the formulae with the results so the cells contain only numbers thereafter - which require no recalculation. To do this, select the main body of the table, then the copy command (Edit | Copy on the menu). Immediately after this give the paste special command specifying values only (Edit | Paste Special | Values | OK on the menu). It's a good idea to save a single copy of the original formula somewhere in case the table has to be reconstructed.

5.6. Model Updates

Like all spreadsheet programs, Excel tries to avoid doing unnecessary recalculation. If nothing has changed in a spreadsheet, it does not automatically recalculate it. Even the Excel command Options | Calculate Calc Now (F9) will not force a spreadsheet to recalculate if nothing has changed in the spreadsheet itself.

This can be a problem with Aqwa results where models may be rerun: as far as Excel is concerned a function asking for (say) the position of structure 1 at a particular time-step does not need recalculating if none of these numbers have changed - it does not take account of the fact that the data in files outside Excel may have changed. This consideration applies even when a spreadsheet is closed and reopened or Excel restarted.

To force all the results to be reloaded, make a null edit of the model name (as used in the first argument to each AQL function) by clicking on the cell, then pressing F2 to edit the cell, then enter to accept the edit. Excel will recalculate the spreadsheet and update all the values.

Chapter 6: Error Codes

The error codes on the left below can be translated into the text on the right with the AQL message function, described previously. Some of these error messages relate to problems within the AQL interface that cannot be corrected by the user. These may arise due to errors within the AQL code, corrupt Aqwa files, or lack of resources such as free memory or file handles.

Error Code	Description
#AQL:101	Invalid or missing node number for specified structure
#AQL:102	Structure does not exist in specified model
#AQL:103	Unable to locate or open .RES file for specified model
#AQL:108	Invalid directory path for specified model
#AQL:109	Invalid or missing frequency for specified structure
#AQL:110	Invalid or missing direction for specified structure
#AQL:111	Phase or Amplitude keyword required
#AQL:202	Phase or Amplitude keyword required
#AQL:204	Unable to locate .POS file for specified model
#AQL:205	Unable to open .POS file for specified model
#AQL:212	Invalid time-step requested
#AQL:213	Results are not available for this time-step
#AQL:215	Invalid freedom requested
#AQL:220	Invalid moonpool requested
#AQL:305	Unable to locate .PLD file for specified model
#AQL:314	Invalid or missing plot information requested
#AQL:402	Unable to locate file

The DLL makes use of several standard Aqwa routines for file access, and these may generate error messages that are not covered by the codes above. In this case the Aqwa error messages are echoed to the spreadsheet. They were not intended for outputting to a spreadsheet, because they are usually too long for the cells, so it is necessary to enlarge the cell so that the message can be read.