AFGROW Workshop 2019

Using AFGROW in MatLab

Jimmy Lambert, Alex Litvinov,
LexTech, Inc.

LexTech

Problem

AFGROW .NET

- Find a way to run AFGROW prediction within MATLAB
- Needs the ability to pass parameter values by reference
- Needs to be able to catch events such as PredictFinished
- Need to be able to access classes such as AfgrowRetardationModels

History

- MatLab’s OLE automation server method is the recommended method for COM
- Afgrow did not work with this method originally, because MatLab uses late binding, and AFGROW did not
- The .Net method of using AFGROW within MatLab worked, but required some extra steps
- Afgrow was updated to be compatible with the actxserver method

.Net Method

- The .Net Method Can Be Used With All Versions of AFGROW
- In order to use the .Net method, the user must have a visual studio developer pack.
- Events can not be handled in this version, meaning methods like RunPredict are not useful.
- Step-By-Step guide can be found at https://www.afgrow.net/userarea/default.aspx in the document “Using MATLAB to Perform a Life Prediction in AFGROW version prior to 5.3.3”

OLE Automation Server Method

- Can be used with versions of AFGROW after version 5.3.3
- Requires nothing but AFGROW and MatLab
- This method can handle the PredictFinished event
- A step by step guide can be found at the AFGROW downloads page in the document “Using MATLAB to Perform a Life Prediction in AFGROW version 5.3.3 or higher”

.Net COM Example

- Add a .Net Reference to AFGROW
- Create an Afgrow.ApplicationClass object
- Use Either ConstAmplitudeSpectrum or OpenSpectrum to set the spectrum.
- Set any additional options for the specimen and spectrum
- Call RunPredict or RunFrozPredict to run a prediction.

```
reference = NET.addAssembly('C:\Program Files\AFGROW\Afgrow.dll');
afgrow = Afgrow.ApplicationClass;
afgrow.ConstAmplitudeSpectrum(0.0)
%Add additional options such as model number, crack length,
% retardation, smf or much more here.
[ret,Cycles,FinalC,finalKc,finalA,finalKa,finalCt,finalKct] =
afgrow.RunFrozPredict()
```

OLE Automation Server Method

Simple Example

- Create an AFGROW COM object using actxserver
- Use Either ConstAmplitudeSpectrum or OpenSpectrum to set the spectrum.
- Set any additional options for the specimen and spectrum
- Call RunPredict or RunFrozPredict to run a prediction.

```
afgrow = actxserver('Afgrow.Application');
afgrow.ConstAmplitudeSpectrum(0.0);
%Add additional options such as model number, crack length,
% retardation, smf or much more here.
[ret,Cycles,FinalC,finalKc,finalA,finalKa,finalCt,finalKct] =
afgrow.RunFrozPredict();
```

OLE Automation Server Method

Events Example

- This example will run a prediction similar to the previous example
- This example will use the ‘PredictFinished’ and ‘TransitionInfo’ events in order to receive the data

OLE Automation Server Method

Events Example – Main Script

- This script functions similarly to the actxserver example, but uses events
- ‘registerevent’ is used to register the ‘PredictFinished’ and ‘TransitionInfo’ events to the corresponding event handler functions.
- The model 1070 (Corner Crack at Edge) , it is a part through crack
- ‘RunPredict’ starts a prediction much like ‘RunFrozPredict’ but asynchronously

```
%TransitionInfoExample.m
clear;
afgrow = actxserver('Afgrow.Application');
afgrow.Visible = true;
registerevent(afgrow, {'PredictFinished' @HandlePredictFinished});
registerevent(afgrow, {'TransitionInfo' @HandleTransitionInfo});
afgrow.Model = 1070;
afgrow.ConstAmplitudeSpectrum(0.0);
afgrow.RunPredict;
```

OLE Automation Server Method

Events Example – Event Handler Functions

- The first function handles events for when the crack transition from part-through to through crack
- The second function handles the event when the prediction is finished.
- Each function should be in its' own file in the same folder as the script

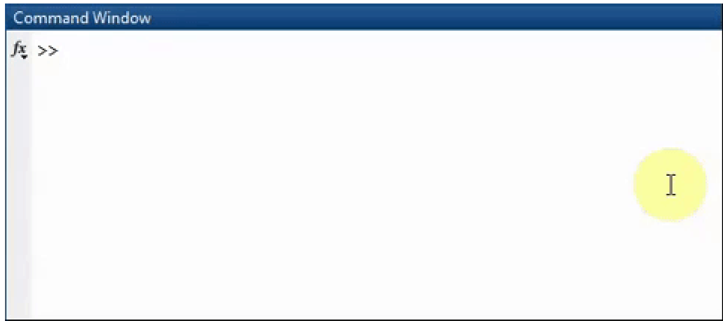
```
function HandleTransitionInfo(varargin)
    disp("Transition Occurred");
    disp(["Passes",varargin{8};"Cycles", varargin{7}]);
end
```

```
function HandlePredictFinished(varargin)
    disp("Prediction Finished");
    disp(["Cycles", varargin{4}]);
end
```

Example

Running Multiple Predictions Asynchronously

- Will be implemented as a function in MatLab
- Outputs Prediction Results to the Screen
- Run Asynchronously with RunPredict and Events
- Runs Predictions Based on Input Array of Crack Lengths
- Is run with the Signature RunMultiPredict(afgrow, [.2,.3,.4])


Command Window

```
fx >>
```

Example

Running Multiple Predictions Asynchronously

```
function RunMultiPredict(afgrow, inputArr)
        global inputs;
        global globalAfgrow;
        globalAfgrow = afgrow;
        inputs = inputArr;


        registerevent(afgrow, {'PredictFinished' @PredictFinished_Multi_Handler});
        afgrow.Visible = true;
        afgrow.Units = 'UnitsEnglish';
        afgrow.Model = 'aCenterThrough';
        afgrow.SpecimenThickness = 1;
        afgrow.ConstAmplitudeSpectrum(0.0);
        afgrow.SMF = 10;
        afgrow.CrackLengthC = inputArr(1);
        afgrow.RunPredict;
end
```

inputArr will contain varying initial crack lengths in this example.

These variables are made global so the event handler will have access to them.

LexTech

Example

Running Multiple Predictions Asynchronously

AFGROW .NET

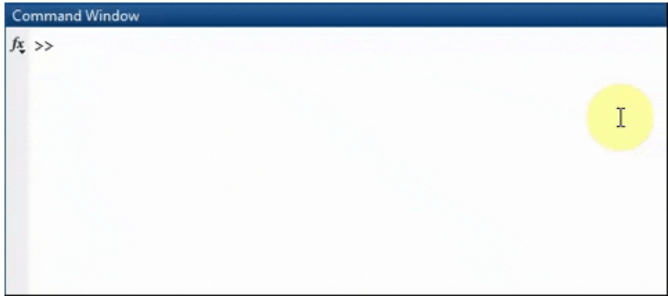
```
function PredictFinished_Multi_Handler(varargin)
        global inputs;
        global globalAfgrow;
        disp(["Initial Crack Length",inputs(1);"Cycles",varargin{4}]);
        if size(inputs,2) > 1
                inputs(1) = [];
                globalAfgrow.CrackLengthC = inputs(1);
                globalAfgrow.RunPredict;
        end
end
```

Changes to the model parameters are made in these lines. The input the was used previously is deleted, then, the crack length is updated with the next value. Finally, globalAfgrow.RunPredict is called. This routine is then called recursively with each PredictFinished event.

Example

Advanced Models

- Will be implemented as a function in MatLab
- Outputs Prediction Results to the Screen
- Run with RunFrozPredict inline
- Runs Predictions Based on Input Array of Crack Lengths
- Is run with the Signature afgrow.RunFrozPredict()



```
fx >>
```

Example

Advanced Models

```
function [RetVal, Cycles, FinalC, FinalKc] = AdvancedRunFrozPredict(Thickness, HoleOffset, IsPartThrough, CrackLengthC,
CrackLengthA)
        afgrow = actxserver('Afgrow.Application');
        afgrow.Visible = true;
        afgrow.Units = 'UnitsEnglish';
        afgrow.Model = 'mAdvanced';



        afgrow.AdvancedModel.AddHole("hole1", .5, HoleOffset);


        %Continued on next page
```

In this example we set the model to ‘mAdvanced’ which will allow us to access AFGROW’s Advanced Model Interface.

After a hole is added to the model, corner cracks, through cracks, and slots can be added to it.

Example

Advanced Models

```
    if IsPartThrough
            afgrow.AdvancedModel.AddCrackToHole("hole1", 'AfgrowCornerCrack', 'HRight', CrackLengthC, CrackLengthA);
    Else
            afgrow.AdvancedModel.AddCrackToHole("hole1", 'AfgrowThroughCrack', 'HRight', CrackLengthC);
    end
```

These lines evaluate the boolean value 'IsPartThrough' that is a parameter of the function. Then either a through crack or a part through crack is added to "hole1".

```
    afgrow.AdvancedModel.SetModelProperty('propThickness', Thickness);
    afgrow.ConstAmplitudeSpectrum(0.0);
    afgrow.SMF = 14;
    [RetVal,Cycles,FinalC,FinalKc,~,~,~,~] = afgrow.RunFrozPredict;
End
```

LexTech

Conclusion

AFGROW .NET

- The .Net method should be used for AFGROW versions before 5.3.3
- The OLE Automation Server Method should be used with versions of AFGROW version 5.3.3 or newer
- MatLab version 2019a contains a fix that allows events such as HandleTransitionInfo to be used
- For more information about using MatLab with AFGROW, see the My AFGROW page on afgrow.net