LexTech

AFGROW.NET

AFGROW Workshop 2020

Using AFGROW with Python

Jimmy Lambert, Alex Litvinov,
LexTech, Inc.

LexTech

Goals

AFGROW.NET

- Demonstrate the ability to use a Python script to setup and run AFGROW predictions
- Create a practical life prediction example using AFGROW COM interface and the Python language

LexTech

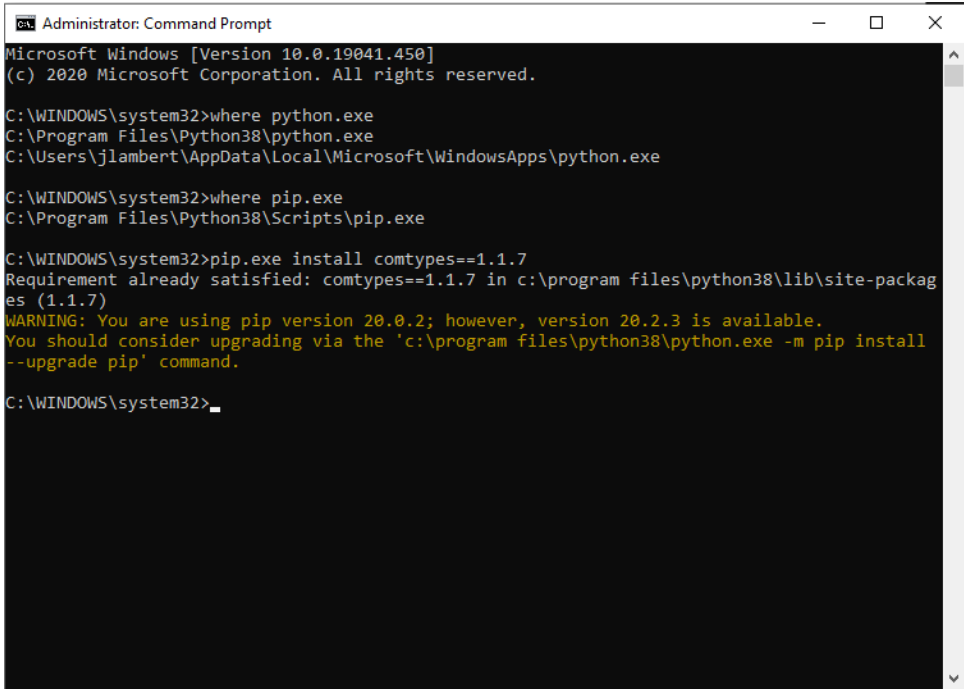
Requirements

AFGROW.NET

- AFGROW version 5.3.5 or newer
- The latest version of Python from https://www.python.org/downloads/
- Python Comtypes Library version 1.1.7

Python Setup Steps

- Download and install Python
- Verify Python installation by:
  - Open Command Prompt as an Administrator
  - Type “where python.exe” and “where pip.exe” to verify that python has installed correctly
- Install COM Type Python Library by typing “*pip.exe install comtypes==1.1.7*” . This will install the library that will allow Python to interact with AFGROW and Spectrum Manager through the COM Interfaces.



LexTech

Steps to Setup a Simple Test Project

AFGROW.NET

- Create a folder and add a new empty Python file
- In the python file, paste or type the example code
- To run the python script, type “python.exe nameoffile.py” in the command prompt

LexTech

Used in this Example

AFGROW.NET

- Comtypes Python Library
- AFGROW object and AFGROW Events
- Comtypes CreateObject Method
- Comtypes GetEvents Method
- Comtypes PumpEvents Method

LexTech

Sample Code

AFGROW.NET

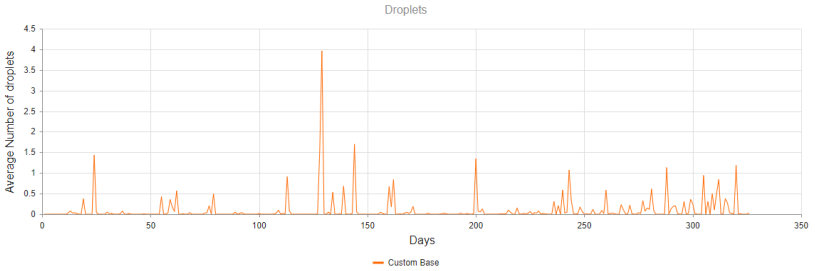
```
    # Import comtypes
    from comtypes.client import CreateObject
    from comtypes.client import GetEvents
    from comtypes.client import PumpEvents

    # Create class and function that will handle events coming from AFGROW
    class EventSink(object):
        def IAfgrowEventSink_PredictFinished(self, this, retval, cycles, kc, ka, kct, c, a, ct):
            print("predict finished")
            print(retval)
            print(cycles)
            print(kc)
            print(ka)
            print(kct)
            print(c)
            print(a)
            print(ct)

    # Create AFGROW instance and set parameters
    af = CreateObject("Afgrow.Application")
    af.Visible = True
    af.ConstAmplitudeSpectrum(0.0)

    # Connect class in order to process AFGROW events
    sink = EventSink()
    connection = GetEvents(af, sink)

    # Start the AFGROW Prediction
    af.RunPredict()

    # Pump events to asyncronously wait for AFGROW events, in seconds.
    PumpEvents(30)
```

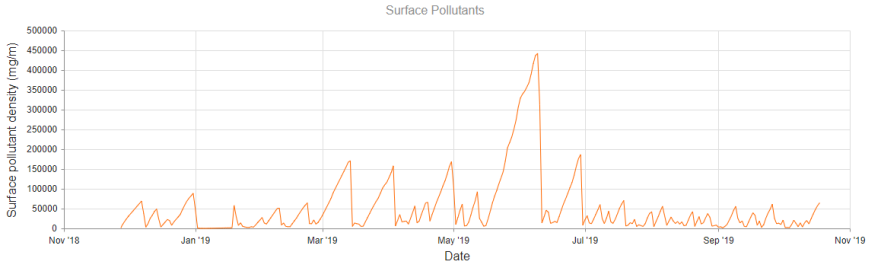
Practical Example Background

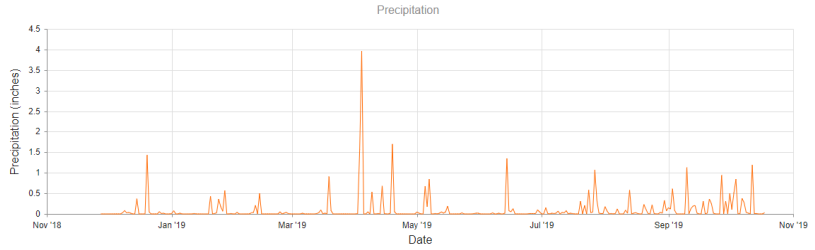
We used data and prediction results of the application that was developed by *Corrosion Prognostics, LLC* as part of the SBIR:

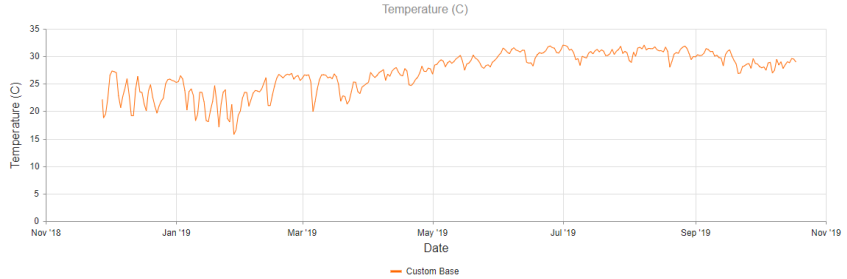
"Computational corrosion modeling for Condition Based Maintenance Plus (CBM+)/aircraft environment tracking".

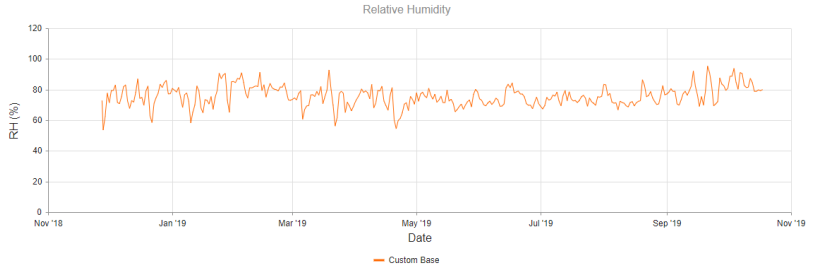
The application uses meteorological data driven modeling capability, surface wetness and corrosion kinetics to prediction the steel and aluminum general corrosion.









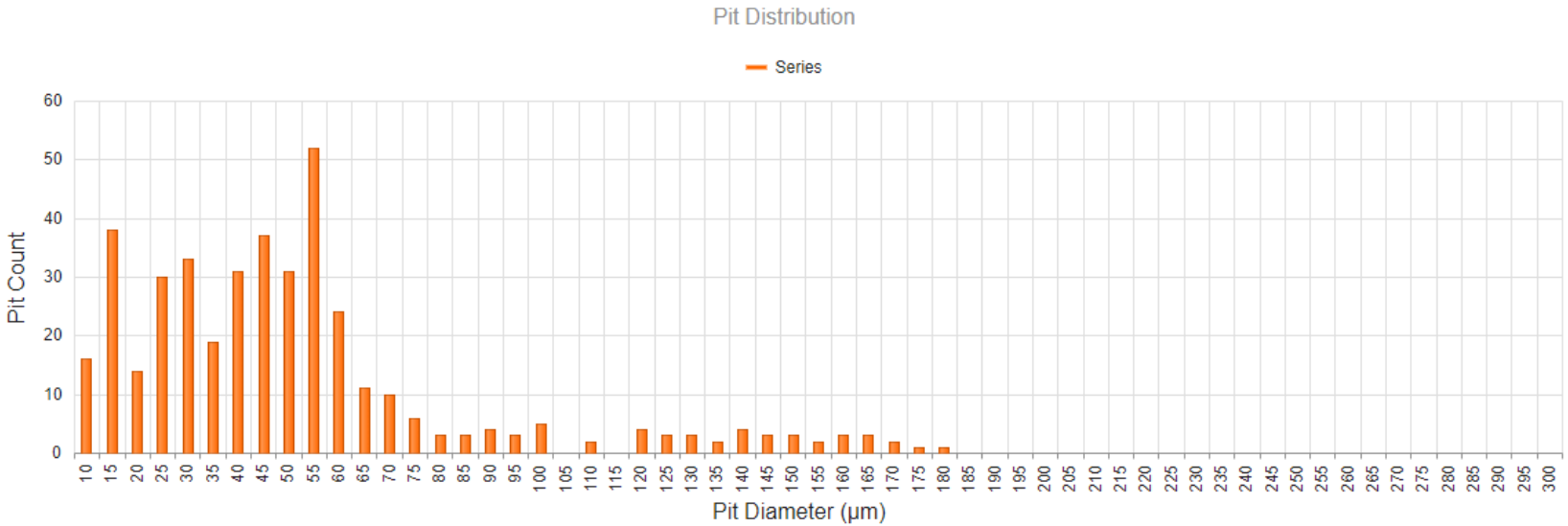


Problem Setup

- Used the application generated distribution of corrosion pits in aluminum as the input to generate the distribution of life.
- Corner Crack at Hole Model

- Pit radius was used as the initial crack size
- Used Falstaf and Constant Amplitude Spectrum
- SMF = 25

Pit Distribution Plot



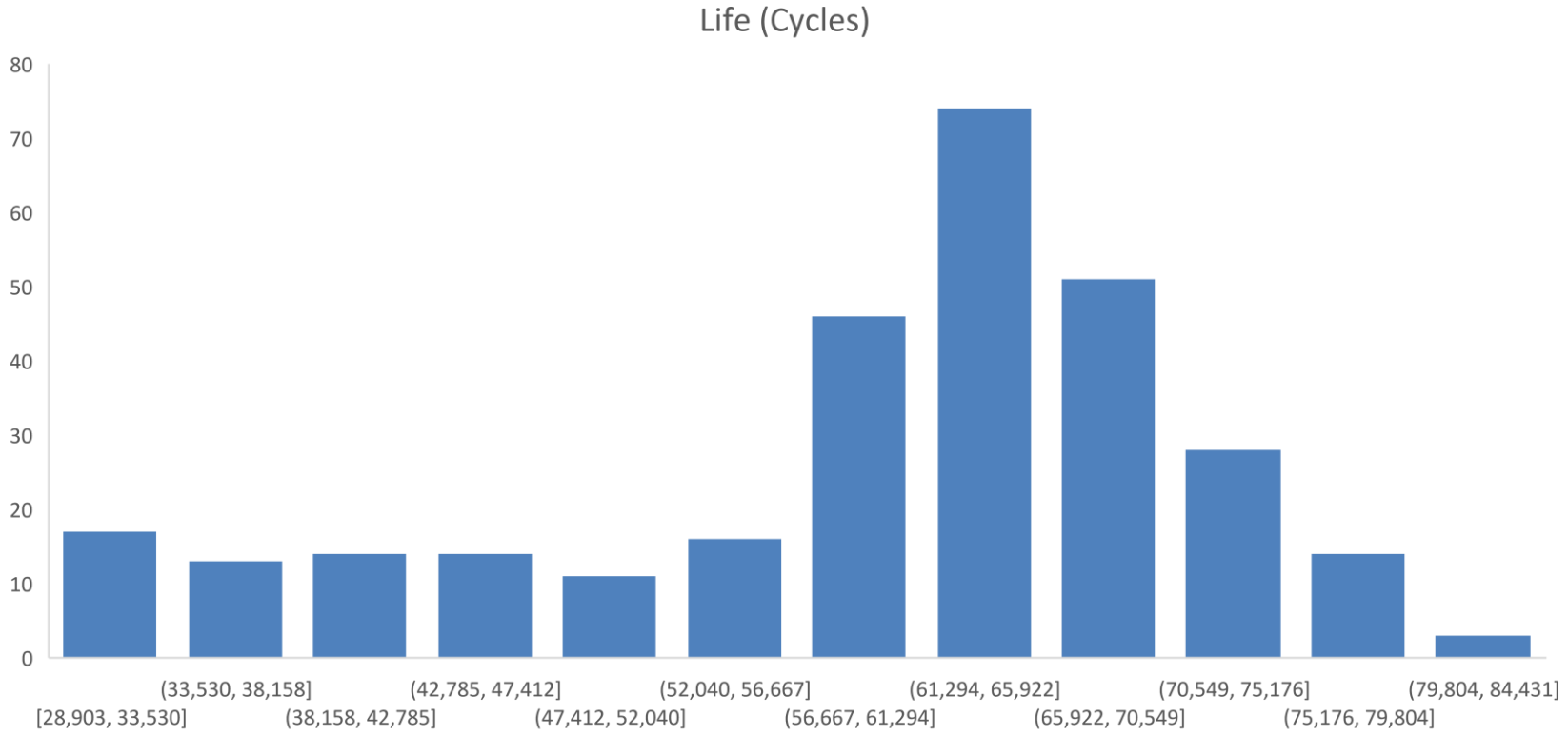
Pit Data Sample Code (1/2)

```
    # Import comtypes
    from comtypes.client import CreateObject
    from comtypes.client import GetEvents
    from comtypes.client import PumpEvents

    #Get path to file with pit data, radius in inches
    path = input("Path to converted pit data: ")
    ConvertedPitData = open(path, mode='r')
    pitLines = ConvertedPitData.readlines()

    #parse file for the data
    for i in range(len(pitLines)):
        pitLines[i] = float(pitLines[i])

    #Open file to output life in cycles
    outfile = open("AFCycles.txt", 'w')

    #Create AFGROW object and set parameters
    af = CreateObject("Afgrow.Application")
    af.Visible = True
    af.Model = 1030
    af.SMF = 25
    af.OpenSpectrumFile("C:\\Users\\YourName\\Documents\\AFGROW\\Spectra\\dfstaf.sp3")
    first = True
```

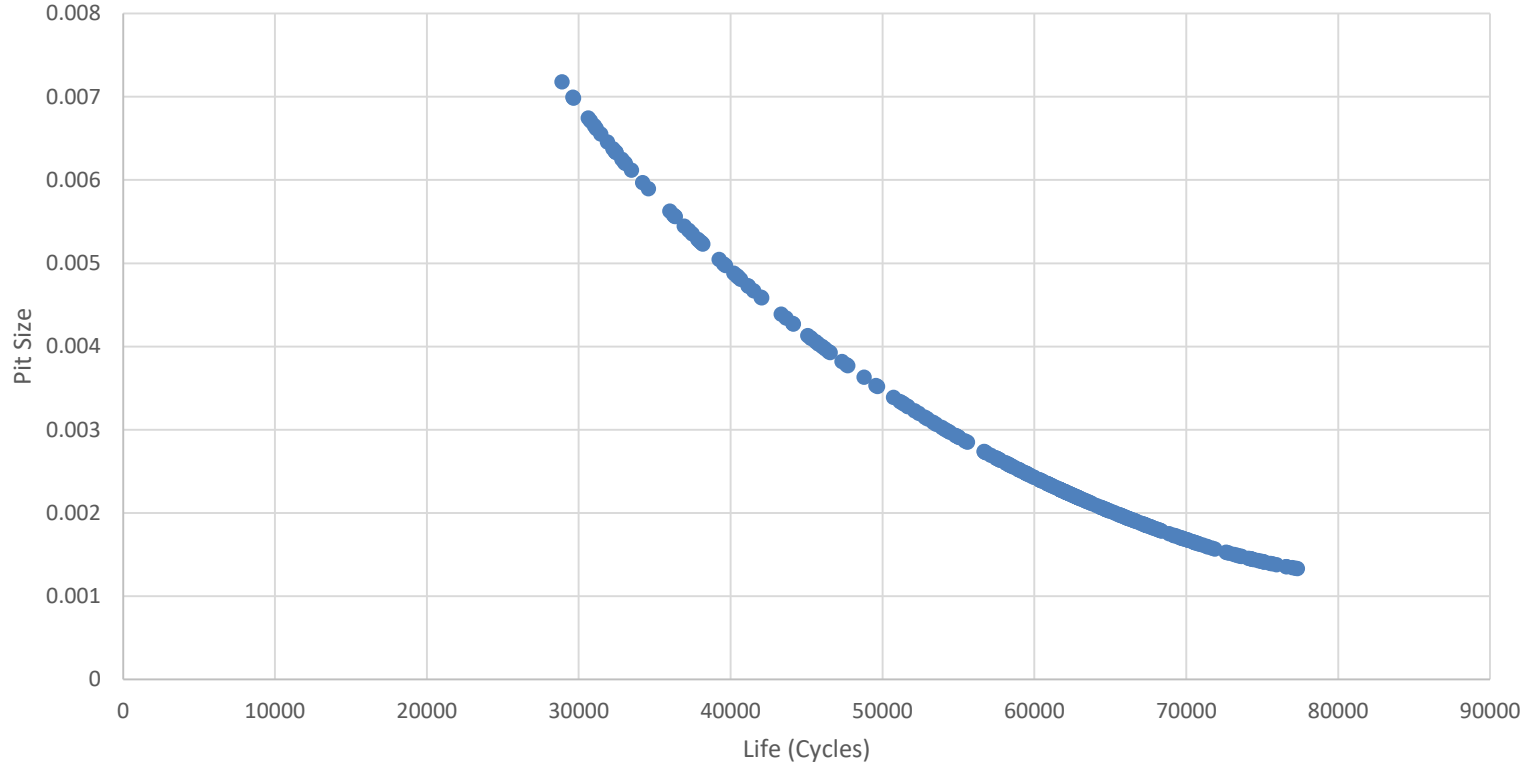
Pit Data Sample Code (2/2)

```
    # Create class and function that will handle events coming from AFGROW
    class EventSink: # this class handles all of the events from afgrow

        count = 0

        def __init__(self, pitsList):
            self.pitsList = pitsList
            self.count = 0

        def IAfgrowEventSink_PredictFinished(self, this, retval, cycles, kc, ka, kct, c, a, ct):
            print(self.count)
            print(cycles)
            outfile.write(str(cycles) + "\n")
            af.CrackLengthA = self.pitsList[self.count]
            af.CrackLengthC = self.pitsList[self.count]
            self.count = self.count + 1
            if not (self.count > len(self.pitsList) - 1):
                af.RunPredict()
            else:
                outfile.close()

    # Connect class in order to process AFGROW events
    sink = EventSink(pitLines)
    connection = GetEvents(af, sink)

    # Start the AFGROW Prediction
    af.RunPredict()

    # Pump events to asyncronously wait for AFGROW events, in seconds.
    PumpEvents(150000) #this method waits for events to arrive for the amount of time determined by the parameter
```

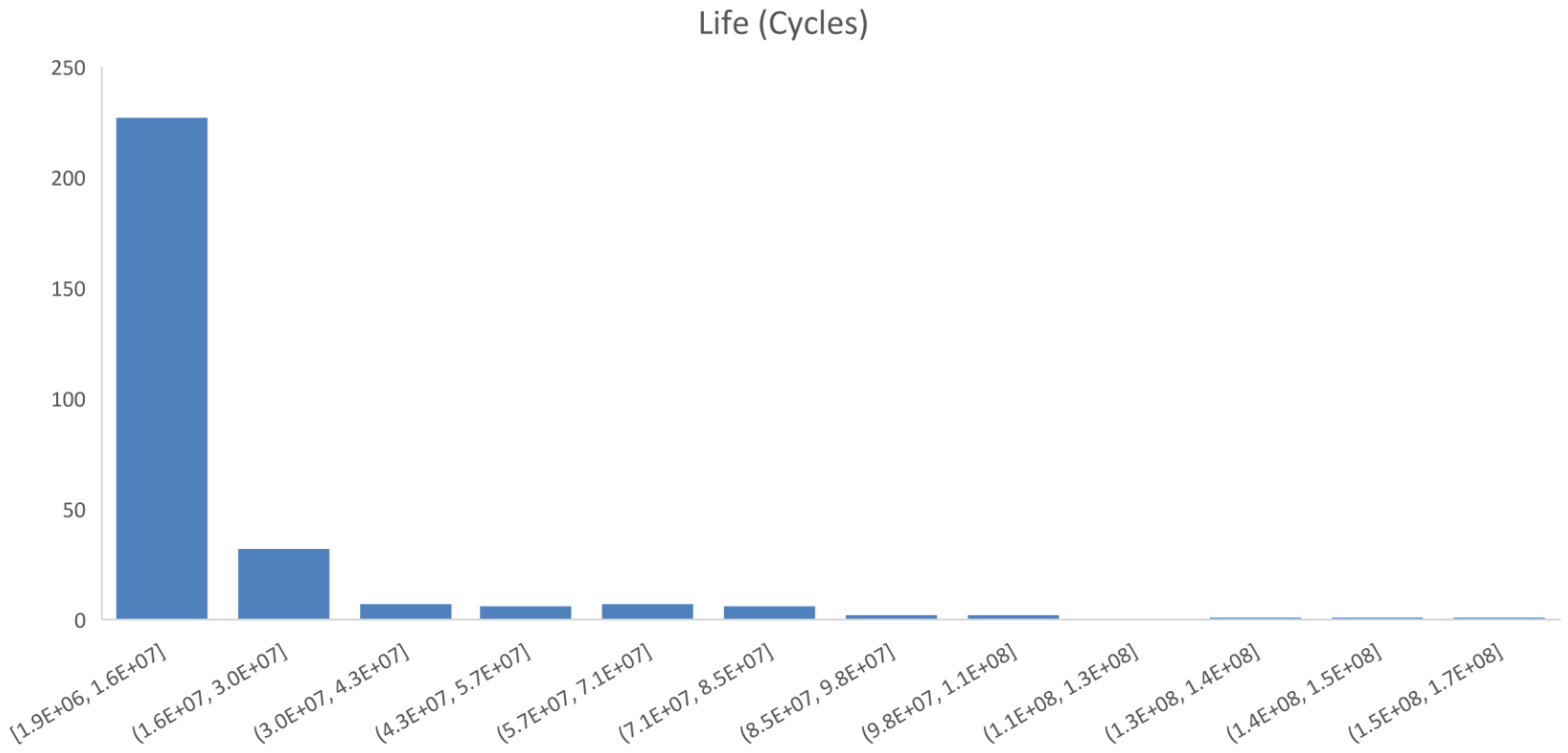
Pit Data Results (Constant Amplitude)



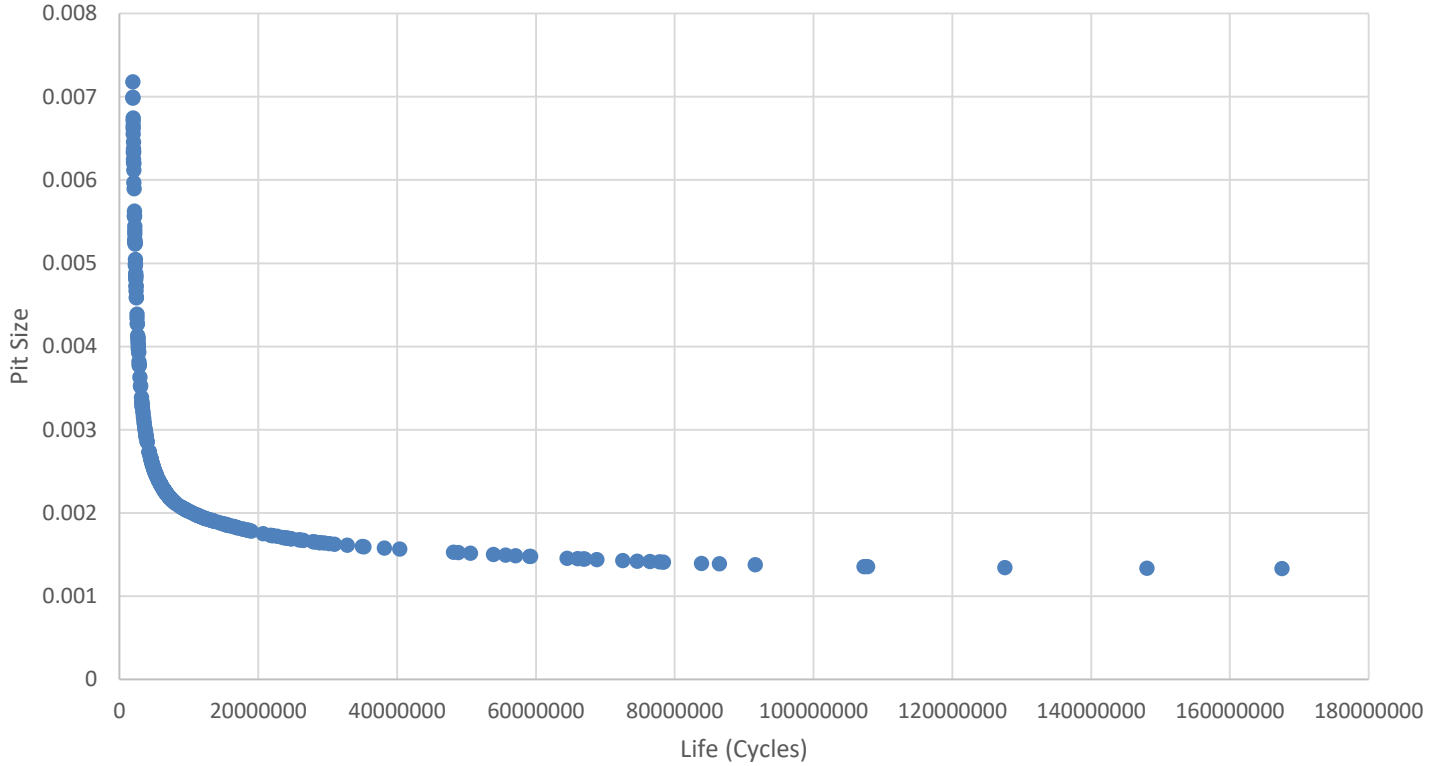
Pit Data Results (Constant Amplitude)



Pit Data Results (FALSTAF)



Pit Data Results



LexTech

Conclusion

AFGROW.NET

- Using comtypes allows AFGROW predictions to be run within a python script
- Using this method allows many predictions to be run automatically with python

LexTech

Links

AFGROW.NET

- AFGROW.net
- Python.org