ATFX SIGNAL READER API PYTHON SETUP AND DEMO SCRIPTS

August 4, 2022

Document ver. 1.1

Contents

Installing Python

The Python.NET package current version works with certain versions of Python, such as **3.8** or **2.7**. There are pre-release versions of Python.NET that can work with the latest versions of Python, such as 3.10, but this document will focus on installing the Python version 3.8. If you already have a preferred Python version installed that is compatibility with Python.NET, then skip to Installing Python Packages.

Below link is for installing Python **3.8.10**:

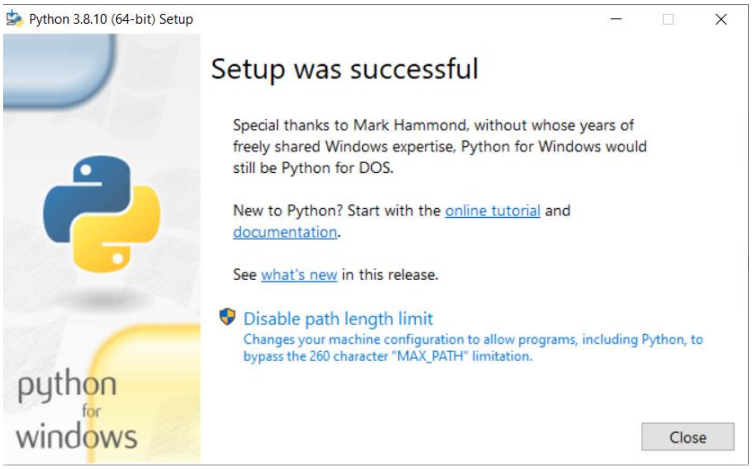
https://www.python.org/downloads/release/python-3810/

Click the **Windows installer (64-bit)** (or whichever installer for the current computer OS).

Run the installer.

Make sure to click the check box "**Add Python 3.8 to PATH**"

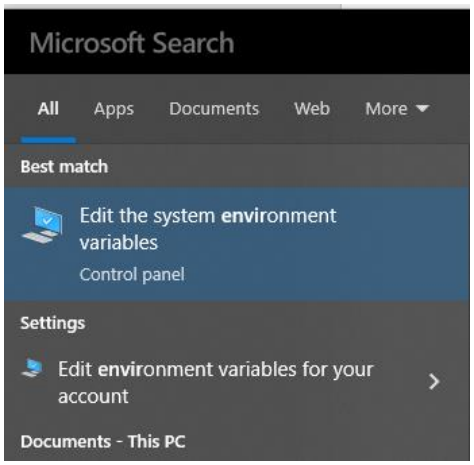
After Python installs and proceeds to the next window, click "**Disable path length limit**"



Make Sure Environment Variables Path are Correct

Clicking the "**Add Python 3.8 to PATH**" in the installation step should add the proper Python command to the environment variables, but just in case, let's check if it is there. Or if you didn't click that option.

Windows Search > Type in "**environment variables**" > Any of the two results will work, click them.



System Properties should pop up > **Environment Variables…** > Select **Path** in **User variables** > Click **Edit**

Two lines should have something to reference to the Python 3.8 installation folders.

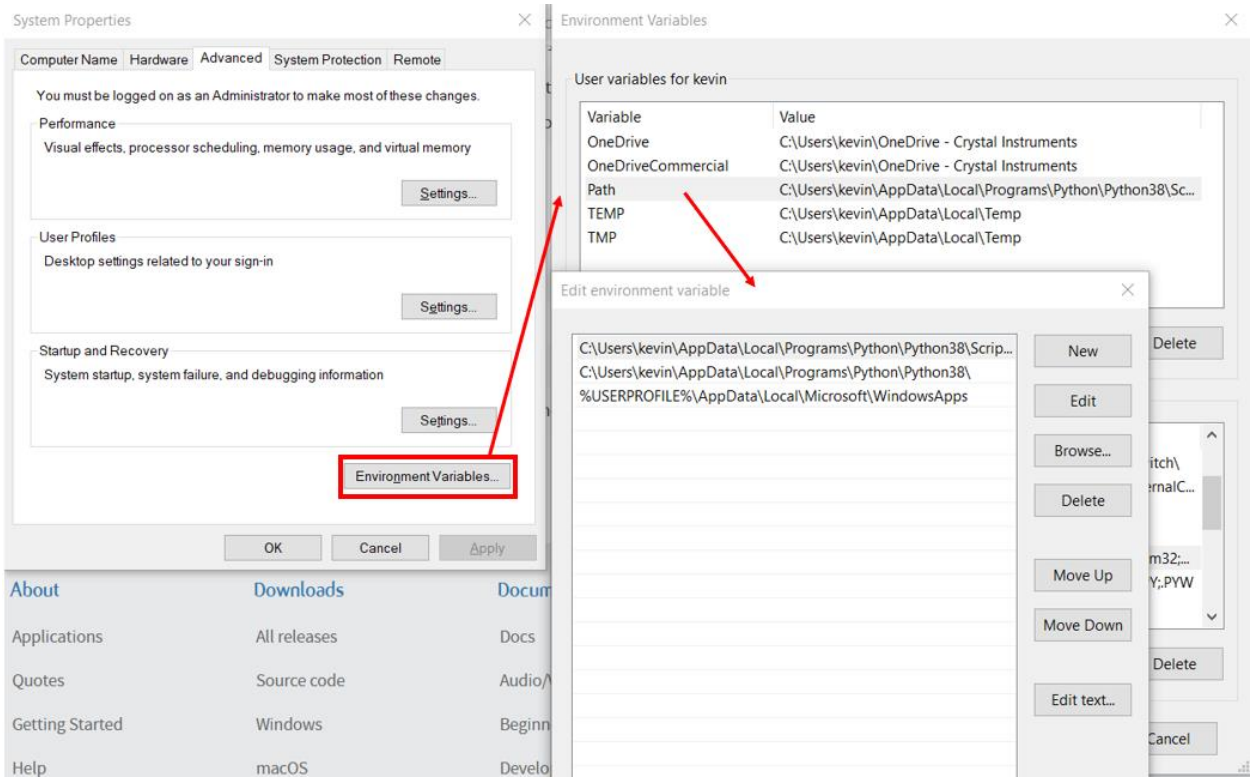
If not, click **New** > click **Browse…** > locate the Python 3.8 installation folder

Or click **New** and type in the file path to the Python 3.8 installation folder, such as

C:\Users\[Your computer username]\AppData\Local\Programs\Python\Python38\Scripts

And

C:\Users\[Your computer username]\AppData\Local\Programs\Python\Python38



Installing Python Packages

Python.NET

After Python has been installed and verified to work, next is installing the Python.NET package which will allow importing C# DLL files that the ATFX API are built in.

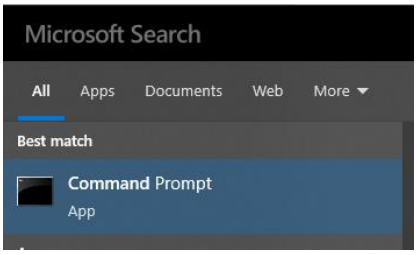
This package is the most important in installing as it offers integration between Python and C# .NET 4.0+ Common Language Runtime (CLR) on **Windows** and Mono runtime on **Linux** and **OSX**. There is another package called IronPython that can provide the same features, but this document and all Python Demo Scripts will focus on Python.NET.

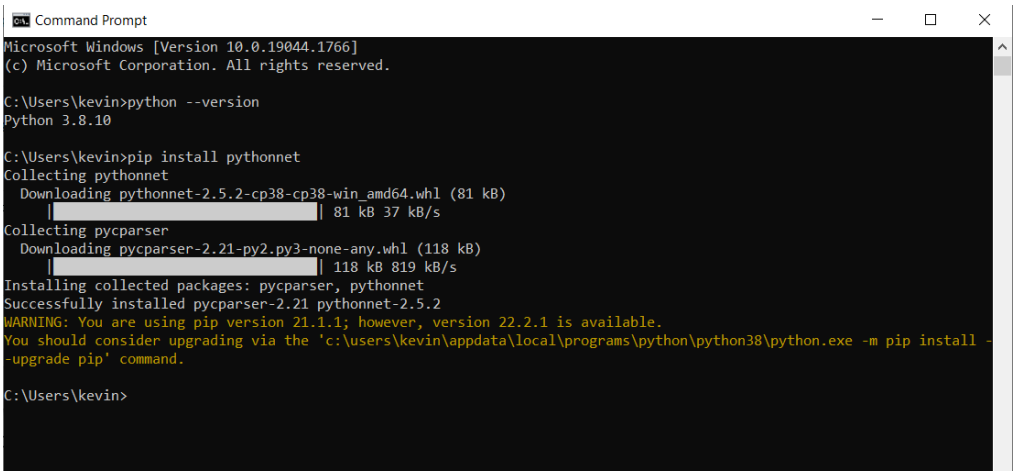
Open **Command Prompt** and type in:

pip install pythonnet

If there is a need to download the Python.NET package to work with Python versions greater than 3.8, then type in:

pip install --pre pythonnet





Here is a link to the Python.NET github page:

https://github.com/pythonnet/pythonnet

Matplotlib & Numpy

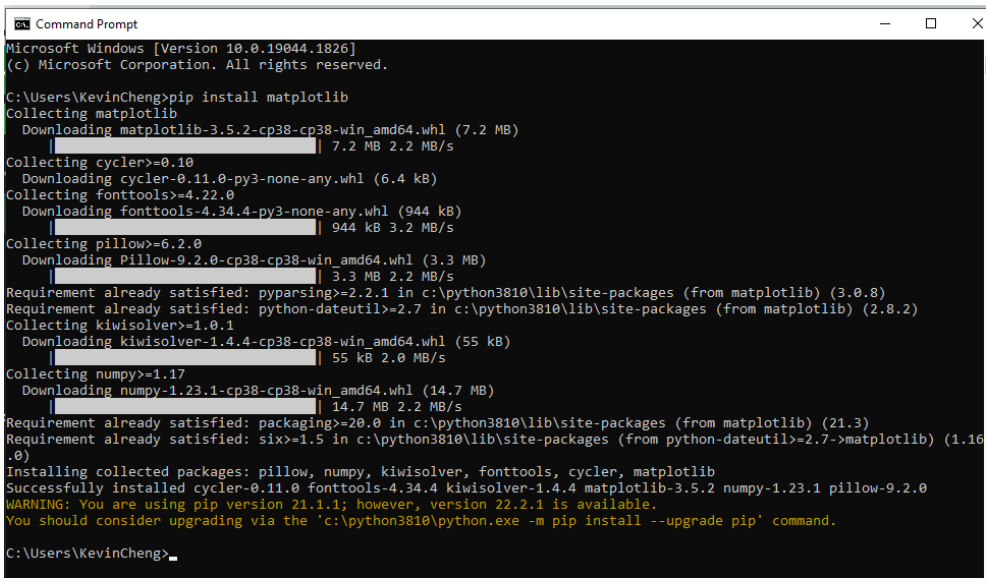
https://matplotlib.org/stable/users/installing/index.html

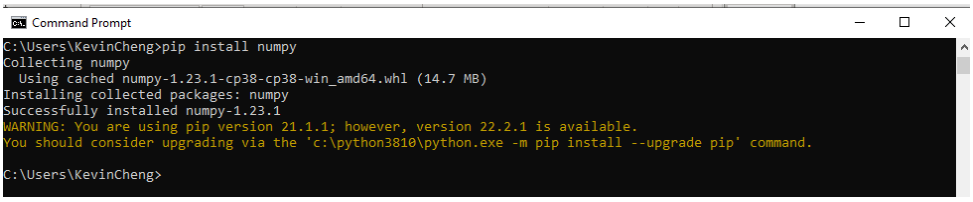
Same as installing Python.NET, go to command prompt and type in:

pip install matplotlib

Installing matplotlib should install numpy as well.

pip install numpy





Fatal Error in Launcher Unable to Create Process Using ‘ ” ” ’

If for some reason the pip install command returns a fatal error in launcher unable to create process using ‘ ” ” ’ then adding python -m to the pip install will work around the issue.

python -m pip install matplotlib

python -m pip install numpy

python -m pip install pythonnet

Downloading & Installing ATFX API

To download the ATFX API, go to the CI site and click on ATFX Signal Reader API.

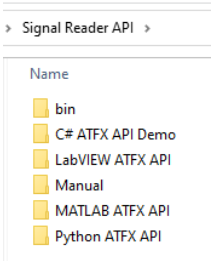
This will download an installer file that will run the process of installing the ATFX API to the computer’s program file location.

C:\Program Files\Crystal Instruments\Signal Reader API

https://www.crystalinstruments.com/programming-corner

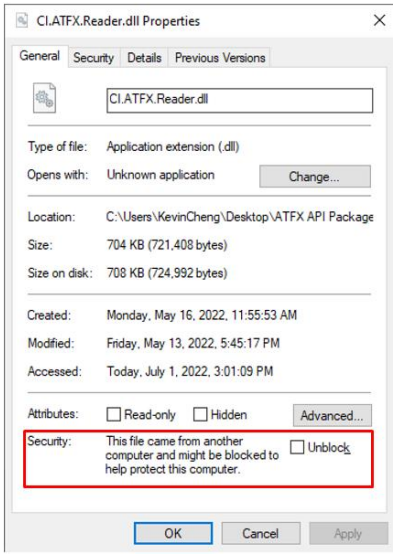
Demo scripts in are their respective coding languages folder, such as C#, LabVIEW, Python and Matlab.

There are also the DLL files in the bin folder and manuals in the Manual folder.



It is recommended to move the entirety of the Signal Reader API **outside of the program files** and into anywhere else such as Downloads, Desktop or C:\. This is so the computer doesn’t ask about admin permissions every time when editing and saving scripts.

Be sure to check the CHM and DLL files via right-click > **Properties** > **Unblock** if the option is there.



Running ATFX API Python Scripts

After Python.NET has been installed and identified, the ATFX API Python Demo Scripts should now work.

Here are some links to the CI site blog posts on ATFX API Python:

https://www.crystalinstruments.com/blog/2022/5/23/how-to-read-ci-data-files-in-python

https://www.crystalinstruments.com/blog/2022/6/10/how-to-read-time-data-from-ci-data-files-in-python-amp-matlab

Integrated Development Environment (IDE)

A simple IDE software is **Visual Studio Code** (**VSCode**).

https://code.visualstudio.com/download

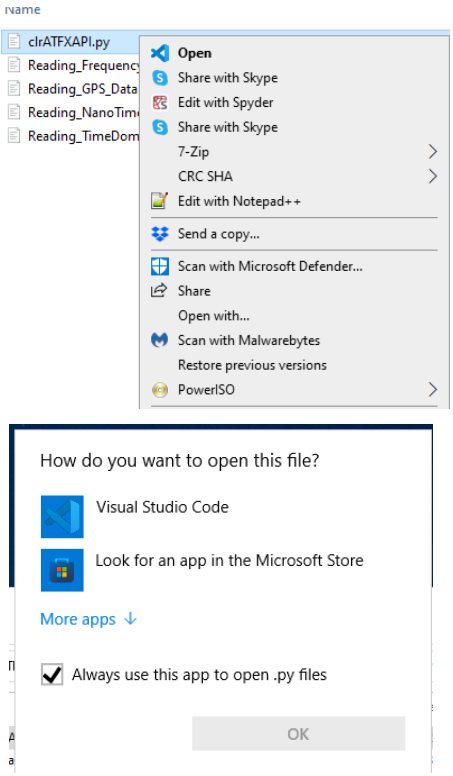
Download it, **install** it, then **open** the coding language scripts in VSCode. The scripts should open in VSCode by default after installation, but just in case it doesn't, **right click** see if the **Open** with VSCode icon is there.

If not, click **Open with…** > **Visual Studio Code**

If that option is not there, click **More apps** > **Look for another app on this PC** and browse for VSCode exe file.

C:\Users\KevinCheng\AppData\Local\Programs\Microsoft VS Code

Code.exe



An even better IDE would be spyder, which requires a separate installation guide posted on our CI site.

There are some aspects of coding in spyder that makes coding between Python and C# easier, as some variables are assumed to be, especially when calling Python variables in C# methods and vice versa.

https://www.crystalinstruments.com/blog/2022/5/27/installing-pythonnet-for-spyder

But in general use cases, VSCode will work fine.

Changing File Paths

There are two main file paths to change inside the Python scripts (Same with matlab).

From the top of the file, locate parentPath and change the file path to match wherever the ATFX API DLL files are. Make sure to follow the format regarding the double slashes (\\) for every slash.

```
#---Pythonnet clr import
import clr
# Change file path here to whereever the DLL files are
parentPath =
"C:\\MyStuff\\DevelopmentalVer\\bin\\AnyCPU\\Debug\\Utility\\CIATFXReader\\"
```

```
clr.AddReference(parentPath + "CI.ATFX.Reader.dll")
clr.AddReference(parentPath + "Common.dll")
clr.AddReference('System.Linq')
clr.AddReference('System.Collections')
```

The next file path is wherever an ATFX file is located via the recordingPath.

In the below example copied from the main Python script, clrATFXAPI.py, there are 3 other atfx file paths. The first is a regular ATFX file, the second is a regular ATFX with TimeStamp file and the third is a regular ATFX with GPS file.

Only one is necessary to send into the RecordingManager.Manager.OpenRecording()

```
#---Main Code
# Uncomment any prints and function calls to extract data from file
print("Running Main Code")
recordingManager = RecordingManager

# Change file path here to whereever signal or recording files are
recordingPath = "C:\\Users\\KevinCheng\\Downloads\\gps test example\\"
# Here are multiple strings for different files, change any of them to contain
the file name and correctly
#  reference it in RecordingManager.Manager.OpenRecording
recordingPathRegular = recordingPath + "SIG0000.atfx"
recordingPathTS = recordingPath + "{4499520}_REC_{20220419}(1).atfx"
recordingPathGPS = recordingPath + "REC0041.atfx"

#OpenRecording(string, out IRecording)
# dummy data is required for the OpenRecording for it to correctly output data
# Make sure to reference the correct file string
dummyTest1, recording = RecordingManager.Manager.OpenRecording(recordingPathTS,
None)
```

Key Points about ATFX API Python Scripts

All key points will be referenced from the main Python demo script, clrATFXAPI.py. There are 4 other Python scripts that are much shorter and focused on a specific data calling method. These specific topic scripts are explained in the ATFX API manual Quickstart section.

The key points here will be listed to help familiarize with various aspects of the ATFX API Python scripts. For more detailed information, please refer to the ATFX API manual.

File Paths

The file paths listed here for both the DLL files and the ATFX files are important for the Python script to know where to reference the mentioned file types. The double slashes are part of the formatting for strings in Python.

```
# Change file path here to whereever the DLL files are
parentPath =
"C:\\MyStuff\\DevelopmentalVer\\bin\\AnyCPU\\Debug\\Utility\\CIATFXReader\\"

clr.AddReference(parentPath + "CI.ATFX.Reader.dll")
clr.AddReference(parentPath + "Common.dll")
```

```
# Change file path here to whereever signal or recording files are
recordingPath = "C:\\Users\\KevinCheng\\Downloads\\gps test example\\"
# Here are multiple strings for different files, change any of them to contain
the file name and correctly
#  reference it in RecordingManager.Manager.OpenRecording
recordingPathRegular = recordingPath + "SIG0000.atfx"
recordingPathTS = recordingPath + "{4499520}_REC_{20220419}(1).atfx"
recordingPathGPS = recordingPath + "REC0041.atfx"
```

Imports

After the initial file path for the DLL files, is the imports, such as import clr.

The clr is the Python.NET package import package name that lets Python to import C# System classes and C# DLL files.

```
#---Pythonnet clr import
import clr
# Change file path here to whereever the DLL files are
parentPath =
"C:\\MyStuff\\DevelopmentalVer\\bin\\AnyCPU\\Debug\\Utility\\CIATFXReader\\"

clr.AddReference(parentPath + "CI.ATFX.Reader.dll")
clr.AddReference(parentPath + "Common.dll")
clr.AddReference('System.Linq')
clr.AddReference('System.Collections')

import numpy as np
import matplotlib.pyplot as plt

#---C# .NET imports & dll imports
from EDM.Recording import *
from EDM.RecordingInterface import *
from ASAM.ODS.NVH import *
from EDM.Utils import *
from Common import *
```

```
from Common import _SpectrumScalingType
from Common.Spider import *
from System import *
from System.Diagnostics import *
from System.Reflection import *
from System.Text import *
from System.IO import *
```

Functions

The functions here are to save space and elegant coding in the next couple of sections. These functions call upon certain ATFX API C# objects or methods to return data from the ATFX file.

In the functions, ShowFrameData and ShowFrameDataSpectrumEU, the for loop can be uncommented to return all data in a signal frame data.

```
#---Functions
def ShowGPSInfo(recordingPath):
    recording = ODSNVHATFXMLRecording(recordingPath)

    if type(recording) is ODSNVHATFXMLRecording:
        nvhRec = recording
        nvhMeasurement = nvhRec.Measurement
        nvhEnvironment = nvhRec.Environment
        bGPS = nvhMeasurement.GPSEnabled
        if bGPS:
            print("GPS Enabled: ", bGPS)
            print("Longitude: ", nvhMeasurement.Longitude)
            print("Latitude: ", nvhMeasurement.Latitude)
            print("Altitude: ", nvhMeasurement.Altitude)
            print("Nanoseconds Elapsed: ", nvhMeasurement.NanoSecondElapsed)

        if not String.IsNullOrEmpty(nvhEnvironment.TimeZoneString):
            print("Time Zone: ", nvhEnvironment.TimeZoneString)

        print("Created Time (Local): ", nvhRec.RecordingProperty.CreateTime)
        print("Created Time (UTC): ",
Utils.GetUTCTime(nvhRec.RecordingProperty.CreateTime, None))
        dateTimeNano = DateTimeNano(nvhRec.RecordingProperty.CreateTime,
UInt32(nvhMeasurement.NanoSecondElapsed))
        print("DateTimeNano Object: ", dateTimeNano)

def ShowFrameData(signal):
    frame = signal.GetFrame(0)
```

```
    frameX = np.fromiter(frame[0], float)
    frameY = np.fromiter(frame[1], float)
    # print(frame[0].Length)

    # Print all data in frame or adjust the range to print a certain amount of
data
    # for data in range(frame[0].Length):
    #     print("X: ", frame[0][data])
    #     print("Y: ", frame[1][data])

    # Print the first 3 pairs of the frame data
    print("X: ", frame[0][0])
    print("Y: ", frame[1][0])
    print("X: ", frame[0][1])
    print("Y: ", frame[1][1])
    print("X: ", frame[0][2])
    print("Y: ", frame[1][2])

    plt.plot(frameX,frameY,'r')
    plt.xlabel(signal.Properties.xQuantity + " (" + signal.Properties.xUnit +
")")
    plt.ylabel(signal.Properties.yQuantity + " (" + signal.Properties.yUnit +
")")
    plt.title("Plot of the " + signal.Name)
    plt.legend(signal.Name)
    plt.show()

def ShowFrameDataSpectrumEU(signal, spectrum, engiUnit):
    frame = signal.GetFrame(0, spectrum, engiUnit)

    frameX = np.fromiter(frame[0], float)
    frameY = np.fromiter(frame[1], float)
    # print(frame[0].Length)

    # Print all data in frame or adjust the range to print a certain amount of
data
    # for data in range(frame[0].Length):
    #     print("X: ", frame[0][data])
    #     print("Y: ", frame[1][data])

    # Print the first 3 pairs of the frame data
    print("X: ", frame[0][0])
    print("Y: ", frame[1][0])
    print("X: ", frame[0][1])
    print("Y: ", frame[1][1])
```

```
    print("X: ", frame[0][2])
    print("Y: ", frame[1][2])

    plt.plot(frameX,frameY,'r')
    plt.xlabel(signal.Properties.xQuantity + " (" + signal.Properties.xUnit +
")")
    plt.ylabel(signal.Properties.yQuantity + " (" + signal.Properties.yUnit +
")")
    plt.title("Plot of the " + signal.Name)
    plt.legend(signal.Name)
    plt.show()
```

RecordingManager

The RecordingManager is the ATFX API method of opening a ATFX file and returning a data object that contains all the recording properties, signal data, test configuration and input channel data in a ATFX file.

For Python, this is the way that a C# method must be coded as. This is due to Python not having a similar concept to C# method out parameter. Thus the dummyTest1 contains nothing of importance, while the recording is the ATFX file data object.

```
#OpenRecording(string, out IRecording)
# dummy data is required for the OpenRecording for it to correctly output data
# Make sure to reference the correct file string
dummyTest1, recording = RecordingManager.Manager.OpenRecording(recordingPathTS,
None)
```

Print Statements

In the following section is blocks of print statements that print various data regarding the ATFX file, from the recording properties, signal properties, signal frame data, a list of signals and so on.

More data can be printed by referencing the ATFX API manual C# code example section.

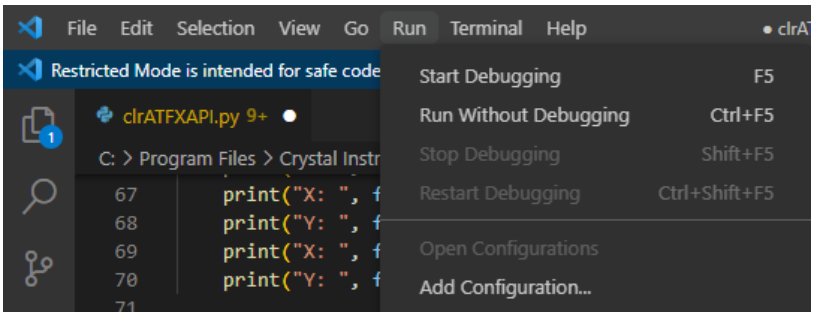
```
#---Print Statements
# Comment or uncomment any print statements below
print(recording)
print("\nRecording Properties\n")
for prop in Utility.GetListOfProperties(recording.RecordingProperty):
    print(prop[0], prop[1])

# Make sure to reference the correct file string
print("\nRecording GPS Properties\n")
ShowGPSInfo(recordingPathTS)
```

```
print("\nSignal 1 Properties\n")
for prop in Utility.GetListOfProperties(recording.Signals[0].Properties):
    print(prop[0], prop[1])

print("\nSignal 1 Properties GeneratedTime\n")
for prop in
Utility.GetListOfProperties(recording.Signals[0].Properties.GeneratedTime):
    print(prop[0], prop[1])

# print("\nRecording List\n")
# recordingList = Utility.GetListOfAllRecordings(recording)
# for record in recordingList:
#     print(record)

# print("\nODS Recording Properties\n")
# for prop in Utility.GetListOfProperties(recordingList[0].RecordingProperty):
#     print(prop[0], prop[1])

# Prints properties of a TimeStamp or GPS recording if the ATFX file references
those files
# print("\nTS Recording Properties\n")
# for prop in Utility.GetListOfProperties(recordingList[1].RecordingProperty):
#     print(prop[0], prop[1])

print("\nSignal List\n")
signalList = Utility.GetListOfAllSignals(recording)
for signal in signalList:
    print(signal)

print("\nSignal Frame Data\n")
print(signalList[0])
ShowFrameData(signalList[0])

# Prints frame data for the 5th signal in the list
# This can be any other signal, such as the 1st signal in the list
print(signalList[4])
ShowFrameData(signalList[4])
print("-----------")
# GetFrameData with pass in parameters for spectrum and engineering unit
# The spectrum type and engineering unit can be changed by referencing the
class library CHM file for valid values
# The Convert.ToInt32 is necessary for the the enum AccelerationUnitType to be
read as a int instead of a string
ShowFrameDataSpectrumEU(signalList[4], _SpectrumScalingType.EUPeak,
```

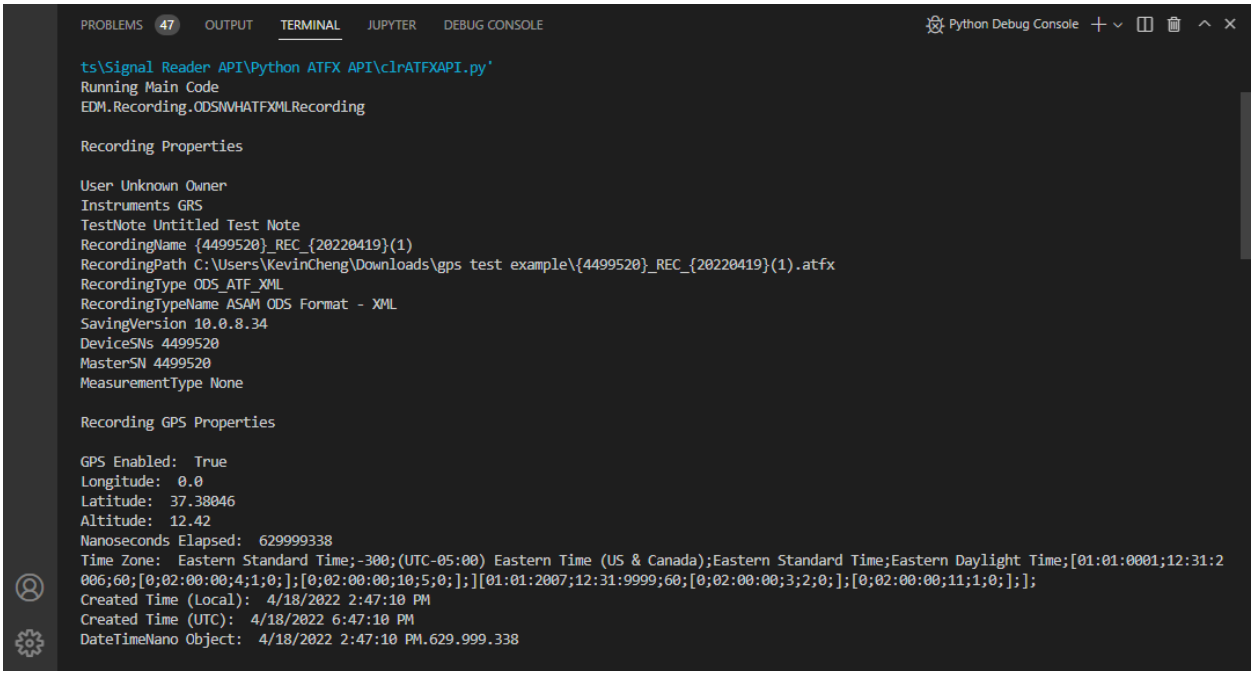
```
                                        AccelerationUnitEnumString.ArrayString[
Convert.ToInt32(AccelerationUnitType.g)])
print("-----------")
ShowFrameDataSpectrumEU(signalList[4], _SpectrumScalingType.EUPeak,
                                        AccelerationUnitEnumString.ArrayString[
Convert.ToInt32(AccelerationUnitType.m_s_2)])
```

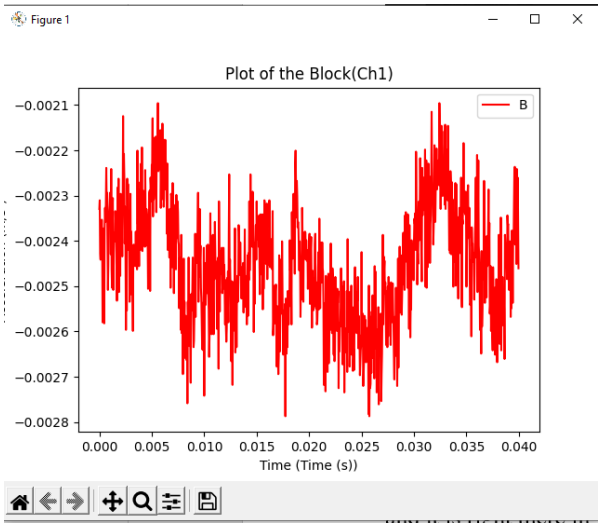
Running the Scripts

To run the scripts in VSCode, go to Run > Run Without Debugging



There will be a terminal that pops up at the bottom of VSCode that will display lines of statements regarding the ATFX file data.

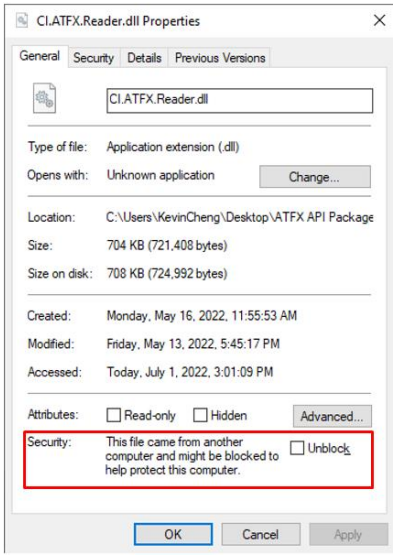




DLL Files not Found Despite Correct File Path

When running the script, there is a chance it may error despite the DLL file path being correct and it is right there in file system. This is due to the computer Windows OS protecting itself from outside files.

Be sure to check the DLL files via right-click > **Properties** > **Unblock** if the option is there.



Module not found: clr, numpy, matplotlib

When running the script and an error ModuleNotFound pops up, then the Python you are currently using may not have the packages installed. This could be because the computer has multiple other Python versions installed and the Python the computer is using is the first Python version you installed.

There are many ways to solve multiple Python Installations, such as uninstalling the version you do not need or assigning a different python environment for the IDE.

Installing Pre-Release Python.NET Package

If the current Python version, such as 3.9 or 3.10 is necessary to keep, then installing the pre-release version of the Python.NET package will solve this problem. Please refer back to Installing Python Packages – Python.NET.

Uninstall a Specific Python

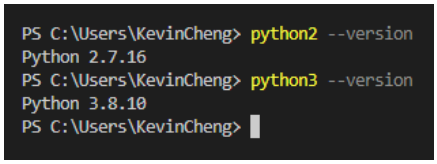
If any version of Python does not matter, then uninstalling any unnecessary Python versions and keeping the preferred 3.8 version will solve this problem.

It is easy to uninstall a Python version by using the OS program uninstaller option. This will completely remove the specific Python from the computer and resetting the path commands.

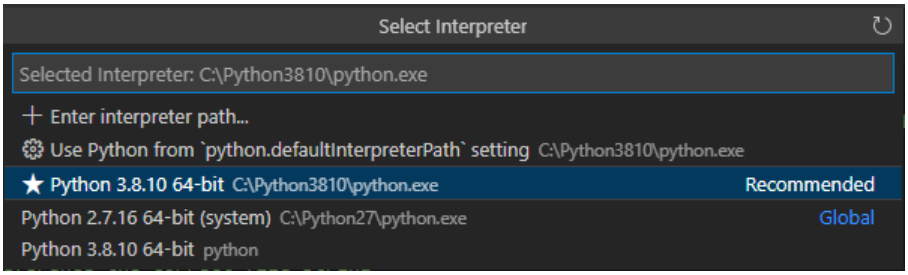
Selecting a Specific Python Version

If multiple Python versions are needed to be installed and cannot be removed, then the simplist way to select a specific Python version would be to rename the python.exe for each Python version installed and call the new names.

Such as Python 2.7 python.exe to **python2**.exe and Python 3.8 python.exe to **python3**.exe and using either python2 or python3 to install packages and run python scripts.



It is also possible to for VSCode to change the Python interpreter by pressing **Ctrl + Shift + P**, type in **interpreter** and select **Python interpreter** which is display a list of available Python versions.

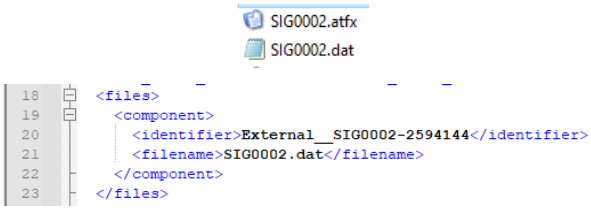


FileNotFoundException & NullReference: Recording

An error indicating that the Python scipt could not find an ATFX file with certain external files, such as a .dat, .ts or .gps file. Or that the recording object is None and any method passing in this recording object sends a NullReference error. This is because in the ATFX file it contains a

reference to any of the mentioned file types. It is important that the .atfx and .dat file are together as the .dat file contains the bulk of the signal data and properties.

```
Running Main Code
Traceback (most recent call last):
  File "c:\MyStuff\DevelopmentalVer\UtilityMode\Other\Python ATFX API\clrATFXAPI.py", line 128, in <module>
    dummyTest1, recording = RecordingManager.Manager.OpenRecording(recordingPathRegular, None)
System.IO.FileNotFoundException:
   at EDM.Recording.RecordingManager.OpenRecording(String recordingPath, IRecording& recording) in C:\MyStuff\DevelopmentalV
er\UtilityMode\ATFXReaderLib\ATFX\RecordingManager.cs:line 1092
PS C:\MyStuff\DevelopmentalVer\UtilityMode\Other\Python ATFX API>
```



END USER LICENSE AGREEMENT FOR CRYSTAL INSTRUMENTS SOFTWARE

--- Updated May 11, 2022

IMPORTANT – READ CAREFULLY. This End User License Agreement ("the Agreement") is a legally binding agreement between you ("the Licensee") and Crystal Instruments Corporation ("Crystal Instruments") for the Crystal Instruments EDM (Engineering Data Management) software, PA (Post Analyzer), EDM Cloud, CI Store, EDC (Embedded Device Control), various API, or the embedded software installed in CoCo, Spider and other series hardware, which includes software components and tools and written documentation ("Software") that accompanies this Agreement. This Agreement contains **WARRANTY AND LIABILITY DISCLAIMERS**.

1. SCOPE OF THE LICENSE RIGHT
1.1 By installing, copying, or using the Software, the Licensee agrees to be bound by the terms of this Agreement.
1.2 Subject to the terms and conditions of this Agreement, Crystal Instruments hereby grants to the Licensee a non-exclusive, non-transferable, right to use the Software, as ordered by the Licensee, solely for the Licensee's own use and solely with the Crystal Instruments hardware for which it is intended.
1.3. The Licensee shall not be entitled to copy or distribute the Software or parts thereof; publish the Software for others to copy; sell, rent, lease, or lend the Software; or transfer or assign the Software or the license rights to the Software to a third party in any other way whatsoever.
1.4 The Licensee shall, however, be entitled to make back-up copies of the Software to the extent that applicable law expressly permits. The use of the back-up copy shall be subject to the terms of this Agreement.
1.5 The Licensee shall ensure that the Software is stored in such a manner that third parties do not have access to it and that a third party does not come into possession of the Software in any other way. The Licensee shall make all employees who have access to the Software fully aware of this obligation.

2. CHANGES TO THE SOFTWARE
2.1 The Licensee shall not be entitled to make any changes to the Software, or reverse engineer, decompile, or disassemble the Software, except and only to the extent that applicable law expressly permits.
2.2 In the event of the Licensee or a third party interfering with or making any changes to the Software, Crystal Instruments may terminate the Agreement with immediate effect, and Crystal Instruments hereby disclaims any liability for the consequences of such interference or change.

3. INTELLECTUAL PROPERTY RIGHTS
3.1 The Software is protected by copyright law and other intellectual property laws. Crystal Instruments or its suppliers own all copyright and any other intellectual property rights in the Software. The Licensee shall respect Crystal Instruments' and its suppliers' rights and the Licensee shall be fully liable in the event of any violation of these rights, including unauthorized passing on of the Software or any part of it to a third party.
3.2 The Licensee shall not be entitled to break, change or delete any security codes or license keys, nor shall the Licensee be entitled to change or remove statements in the Software or on the media on which the Software is delivered regarding copyrights, trademarks, or any other proprietary notices.
3.3 Information and data supplied by Crystal Instruments with the Software, such as, but not limited to, user manuals and documentation, are proprietary to Crystal Instruments or its suppliers. Such information is furnished solely to assist the Licensee in the installation, operation and use of the Software and the Licensee agrees not to reproduce or copy such information, except as is reasonably necessary for proper use of the Software.

4. TRADEMARKS
4.1 The Licensee acknowledges Crystal Instruments' and its suppliers' sole ownership of any trademarks including service marks, logos and other proprietary marks submitted with the Software, and all associated goodwill. This Agreement does not grant the Licensee any rights to the trademarks of Crystal Instruments and its suppliers.
4.2 The Licensee agrees not to use the trademarks in any manner that will diminish or otherwise damage Crystal Instruments' or its suppliers' goodwill in the trademarks. The Licensee agrees not to adopt, use, or register any corporate name, trade name, trademark, domain name, service mark, certification mark, or other designation similar to, or containing in whole or in part, the trademarks of Crystal Instruments.

5. CLOUD SERVICE PROVIDED BY CRYSTAL INSTRUMENTS
5.1 Data Location When cloud service is enabled, Crystal Instruments Corporation may process and store the customer data anywhere Crystal Instruments Corporation or its agents maintain facilities and services.
5.1.1 Facilities All facilities used to store and process an application and customer data will adhere to reasonable security standards no less protective than the security standards at facilities where Crystal Instruments Corporation processes and stores its own information of a similar type.

5.2 Data Processing and Security
5.2.1 Scope of Processing By entering into this agreement, customer instructs Crystal Instruments Corporation to process customer personal data and other data related to its services only in accordance with applicable law: (a) to provide the cloud services; (b) as further specified by customer via customer's use of the cloud services (including the admin console and other functionality of the services); (c) as documented in the form of this agreement, including these terms; and (d) as further documented in any other written instructions given by customer and acknowledged by Crystal Instruments Corporation as constituting instructions for purposes of these Terms.
5.2.2 Data Security Crystal Instruments Corporation will use third party technical measures to protect customer data against accidental or unlawful destruction, loss, alteration, unauthorized disclosure or access. Crystal Instruments Corporation is not responsible or liable for the deletion of or failure to store any customer data and other communications maintained or transmitted through use of the services. In addition, Crystal Instruments is not responsible or liable for unauthorized access of the customer data. Customer is solely responsible for securing and backing up data. Crystal

Instruments Corporation does not warrant that the operation of the software or the services will be error-free or uninterrupted. Neither the software nor the services are designed, manufactured, or intended for high risk activities.

5.2.3 Data Deletion

Deletion by Customer: Crystal Instruments Corporation will enable Customer to delete Customer Data during the Term in a manner consistent with the functionality of the Services.

Deletion on Termination. On expiry of the Term, Crystal Instruments would delete all Customer Data. Customer acknowledges and agrees that Customer will be responsible for exporting, before the Term expires, any Customer Data it wishes to retain afterwards.

5.3 Accounts Customer must have an account to use the services, and is responsible for the information it provides to create the account, the security of passwords for the account, and for any use of its account. If customer becomes aware of any unauthorized use of its password or its account, Customer will notify Crystal Instruments Corporation as promptly as possible. Crystal Instruments Corporation has no obligation to provide customer multiple accounts.

5.4 Payment Terms for Cloud Service

5.4.1 Free Quota Certain services are provided to customer without charge up to the fee threshold, as applicable.

5.4.2 Online Billing At the end of the applicable fee accrual period, Crystal Instruments Corporation will issue an electronic bill to customer for all charges accrued above the fee threshold based on (i) Customer's use of the Services during the previous fee accrual period; (ii) any additional units added; (iii) any committed purchases selected; and/or (iv) any package purchases selected. For use above the fee threshold, customer will be responsible for all fees up to the amount set in the account and will pay all fees in the currency set forth in the invoice. If customer elects to pay by credit card, debit card, or other non-invoiced form of payment, Crystal Instruments Corporation will charge (and customer will pay) all fees immediately at the end of the fee accrual period. If customer elects to pay by invoice (and Crystal Instruments Corporation agrees), all fees are due as set forth in the invoice. Customer's obligation to pay all fees is non-cancellable. Crystal Instruments Corporation's measurement of Customer's use of the services is final. Crystal Instruments Corporation has no obligation to provide multiple bills. Payments made via wire transfer must include the bank information provided by Crystal Instruments Corporation.

5.4.3 Payment Information Crystal Instruments Corporation will not store any payment related information on its facilities. All payment information, including recurring payments are stored at a third party facility. Crystal Instruments will not be responsible or liable for unauthorised access to this information.

5.4.4 Taxes for Cloud Services

(a) Customer is responsible for any taxes, and customer will pay Crystal Instruments Corporation for the services without any reduction for taxes. If Crystal Instruments Corporation is obligated to collect or pay taxes, the taxes will be invoiced to customer, unless customer provides Crystal Instruments Corporation with a timely and valid tax exemption certificate authorized by the appropriate taxing authority. In some states the sales tax is due on the total purchase price at the time of sale and must be invoiced and collected at the time of the sale. If customer is required by law to withhold any taxes from its payments to Crystal Instruments Corporation, customer must provide Crystal Instruments Corporation with an official tax receipt or other appropriate documentation to support such withholding. If under the applicable tax legislation the services are subject to local VAT and the customer is required to make a withholding of local VAT from amounts payable to Crystal Instruments Corporation, the value of services calculated in accordance with the above procedure will be increased (grossed up) by the customer for the respective amount of local VAT and the grossed up amount will be regarded as a VAT inclusive price. Local VAT amount withheld from the VAT-inclusive price will be remitted to the applicable local tax entity by the customer and customer will ensure that Crystal Instruments Corporation will receives payment for its services for the net amount as would otherwise be due (the VAT inclusive price less the local VAT withheld and remitted to applicable tax authority).

(b) If required under applicable law, customer will provide Crystal Instruments Corporation with applicable tax identification information that Crystal Instruments Corporation may require to ensure its compliance with applicable tax regulations and authorities in applicable jurisdictions. Customer will be liable to pay (or reimburse Crystal Instruments Corporation for any taxes, interest, penalties or fines arising out of any mis-declaration by the Customer.

5.4.5 Invoice Disputes and Refunds Any invoice disputes must be submitted prior to the payment due date. If the parties determine that certain billing inaccuracies are attributable to Crystal Instruments Corporation, Crystal Instruments Corporation will not issue a corrected invoice, but will instead issue a credit memo specifying the incorrect amount in the affected invoice. If the disputed invoice has not yet been paid, Crystal Instruments Corporation will apply the credit memo amount to the disputed invoice and Customer will be responsible for paying the resulting net balance due on that invoice. To the fullest extent permitted by law, customer waives all claims relating to fees unless claimed within thirty days after charged (this does not affect any customer rights with its credit card issuer). Refunds (if any) are at the discretion of Crystal Instruments Corporation and will only be in the form of credit for the services. Nothing in this Agreement obligates Crystal Instruments Corporation to extend credit to any party.

5.4.6 Delinquent Payments; Suspension Late payments may bear interest at the rate of 1.5% per month (or the highest rate permitted by law, if less) from the payment due date until paid in full. customer will be responsible for all reasonable expenses (including attorneys' fees) incurred by Crystal Instruments Corporation in collecting such delinquent amounts. If customer is late on payment for the services, Crystal Instruments Corporation may suspend the services or terminate the account(s) and services(s) for breach

5.5 Account Term & Termination

5.5.1 Account Term The term of the account will begin on the effective date and continue until the agreement is terminated.

5.5.2 Termination for Breach Crystal Instruments Corporation may terminate account for breach if: (i) the account(s) is in material breach of the agreement; or (ii) the customer ceases its business operations or becomes subject to insolvency proceedings and the proceedings are not dismissed within ninety days.

5.5.3 Termination for Convenience Customer may stop using the cloud service at any time. Customer may terminate the account(s) and services for its convenience at any time on prior written notice and upon termination, must cease use of the applicable services.

Crystal Instruments Corporation may terminate the account(s) or services for its convenience at any time without liability to Customer.

5.5.4 Effect of Termination If the account(s) or services(s) are terminated, then: (i) the rights granted by one party to the other will immediately cease; (ii) all fees owed by customer to Crystal Instruments Corporation are immediately due upon receipt of the final electronic bill; (iii) customer will delete the software, any application and any data; and (iv) upon request, each party will use commercially reasonable efforts to return or destroy all confidential information of the other party.

5.6 Customer Obligations for Cloud Services

5.6.1 Compliance Customer is solely responsible for account information and data and for making sure its usage of services is consistent with the terms of the services. Crystal Instruments Corporation reserves the right to review the data for compliance.

5.6.2 Restrictions

Customer will not, and will not allow third parties under its control to: (a) copy, modify, create a derivative work of, reverse engineer, decompile, translate, disassemble, or otherwise attempt to extract any or all of the source code of the services (except to the extent such restriction is expressly prohibited by applicable law); (b) sublicense, resell, or distribute any or all of the services; or (c) create multiple account(s) to simulate or act as a single account or otherwise access the services in a manner intended to avoid incurring fees or exceed usage limits or quotas;

5.6.3 Third Party Components

Third party components (which may include open source software) of the services may be subject to separate license agreements. To the limited extent a third party license expressly supersedes this agreement, that third party license governs customer's use of that third party component.

6. EXPORT RESTRICTIONS

The Software may be subject to the export control laws and regulations of the United States. The Licensee must comply with all domestic and international export control laws and regulations that apply to the Software. These laws include restrictions on destinations, end users, and end use.

7. THE LICENSEE'S CHOICE OF SOFTWARE

The Software is a standard product, which is delivered by Crystal Instruments with the functions that are specified in the accompanying documentation. Any assistance provided by Crystal Instruments in connection with the choice of the Software will be based on the Licensee's information about the Licensee's business provided to Crystal Instruments. The Licensee shall be responsible for both the completeness and the accuracy of such information. Crystal Instruments makes no representations or warranties as to whether the Software meets the functionality or other requirements of the Licensee and assumes no liability therefor.

8. WARRANTIES AND DISCLAIMERS

8.1 The Licensee shall be under obligation to examine and test the Software immediately after installation of the Software.

8.2 On condition that Crystal Instruments is fully paid for the Software that Customer purchased, Crystal Instruments warrants that the Software will be free of material defects for a period of 12 months after the delivery of the Software to Licensee (the "Warranty Period"). A defect in the Software shall be regarded as material if it has a material adverse effect on the functionality of the Software as a whole or if it prevents operation of the Software. Minor bugs or functions that can be improved are not viewed as a defect.

8.3 If the Licensee documents that there is a material defect in the Software, and notifies Crystal Instruments of the defect within the Warranty Period, Crystal Instruments will, at its discretion, without charge: (a) deliver a new version of the Software without the material defect, or (b) remedy the defect, or (c) provide Licensee with instructions for procedures or methods (workarounds) which result in the defect not having a significant effect on the Licensee's use of the Software. If Crystal Instruments fails to do any of the above within 30 days (or such longer period of time as is reasonably necessary given the nature of the defect), the Licensee may terminate this Agreement upon notice to Crystal Instruments, in which event Crystal Instruments will refund to Licensee a pro-rated portion of the license fee paid by Licensee for the Software (based on the portion of the Warranty Period remaining as of the date Licensee notified Crystal Instruments of the defect), provided Licensee returns to Crystal Instruments all the Licensee's versions and copies of the Software, and all manuals and accompanying documentation. This paragraph states the sole obligations of Crystal Instruments, and the sole remedy of Licensee, for defects in the Software, and the parties shall not be entitled to bring further claims against each other.

8.4 EXCEPT FOR THE EXPRESS WARRANTY IN SECTION 7.2 ABOVE, THE SOFTWARE IS PROVIDED "AS IS" AND WITHOUT ANY OTHER WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO WARRANTIES OF ACCURACY, COMPATIBILITY WITH OTHER SOFTWARE OR HARDWARE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. CRYSTAL INSTRUMENTS DOES NOT WARRANT THAT THE OPERATION OF THE SOFTWARE WILL BE WITHOUT INTERRUPTIONS, DEFECT-FREE, OR ERROR-FREE OR THAT PRODUCT DEFECTS OR ERRORS CAN OR WILL BE REMEDIED OR CORRECTED.

9. CONSENT TO USE OF DATA

Licensee agrees that Crystal Instruments and its affiliates may, through Internet connections established by the Software or otherwise, collect technical information related to Licensee's use of the Software, including but not limited to the serial numbers of Crystal Instruments hardware with which the Software is used, email addresses of users, and technical information relating to Licensee's computers, systems, application software, and peripherals. Licensee agrees that Crystal Instruments may use such information to facilitate the provision of Software updates and product support, to improve Crystal Instruments' products and/or services, or to provide products or services to Licensee. Crystal Instruments will not, however, publish or disclose such information in a form that may personally identify Licensee.

10. LIABILITY AND LIMITATION OF LIABILITY

10.1 CRYSTAL INSTRUMENTS SHALL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING BUT NOT LIMITED TO LOSS OF EXPECTED PROFIT, LOSS OF DATA OR THEIR RECOVERY, LOSS OF GOODWILL OR ANY OTHER SIMILAR DAMAGES), UNDER ANY LEGAL THEORY, IN CONNECTION WITH THE USE OF THE SOFTWARE OR THE INABILITY TO USE THE SOFTWARE, REGARDLESS OF WHETHER CRYSTAL INSTRUMENTS HAS BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES.

10.2 IN NO EVENT SHALL THE TOTAL LIABILITY OF CRYSTAL INSTRUMENTS TO LICENSEE ARISING OUT OF OR RELATING TO THE SOFTWARE EXCEED THE LICENSE FEE PAID BY LICENSEE FOR THE SOFTWARE.

10.3 Crystal Instruments shall not be liable for any errors, defects, or deficiencies which are not related to the Software, nor shall Crystal Instruments be liable for the integration or interaction between the Software and the Licensee's existing hardware and software. Crystal Instruments shall not be liable for the effect of any upgrades on existing hardware, software, or adjustments for the Software regardless of whether such adjustments were developed by Crystal Instruments.

10.4 Crystal Instruments shall have no liability of any nature relating to software or content of third parties that may be included in the Software.

10.5 The limitations in this Section 9 will apply even in the event of failure of essential purpose of any remedy.

11. GOVERNMENT USERS

The Software and related documentation are "Commercial Items", as that term is defined at 48 C.F.R. §2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation", as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. The Software and documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein.

12. TERM AND TERMINATION

12.1 The term of this Agreement, and Licensee's license rights, which may be referred to the activation period of license, shall be as indicated in Licensee's order. Such term may be perpetual, or may be of limited duration in the event the Software is provided to Licensee for demonstration, evaluation or other similar purposes. Licensee acknowledges that if Licensee's rights are of limited duration, the license key provided to Licensee to enable use of the Software may cease to allow use of the Software after expiration of such activation period.

12.2 Upon termination of the Agreement for any reason, the Licensee is obliged to immediately return or destroy the Software and all copies thereof as directed by Crystal Instruments and, if requested by Crystal Instruments, to certify in writing as to the destruction or return of the Software and all copies thereof.

13. DEFAULTS

If the Licensee is in default of the Agreement, the Licensee's rights under the Agreement shall terminate with immediate effect, and the Licensee shall be under an obligation to return the Software, including any back-up copies and accompanying documentation, without a right to repayment. In addition, Crystal Instruments shall be entitled to damages for any loss, which Crystal Instruments may suffer, in accordance with the general rules of United States law, including all losses, damages, costs, expenses, etc., without any limitations, incurred or suffered by Crystal Instruments as a result of claims from any third party in relation to the Licensee's breach of the Agreement.

14. UPDATES AND RENEW

14.1 For one year after the delivery of the Software, Crystal Instruments will provide Licensee, free of charge, with any updates to the Software that Crystal Instruments makes generally available to its customers. Licensee may renew such right to receive updates, for additional periods of one year each, by paying Crystal Instruments the support renewal fee in effect at the time of such renewal. Licensee acknowledges that if Licensee elects not to renew the right to receive updates, the license key provided to Licensee to enable use of the Software may thereafter cease to allow installation and use of updates. Notwithstanding the above, Crystal Instruments may charge an additional license fee for any optional upgrades Crystal Instruments may release, which include significant new functionality and which Crystal Instruments does not make available without charge to its customers generally.

14.2 Crystal Instruments and the Licensee can agree on the other term about the period of software update after the sales.

14.3 Crystal Instruments has the rights to control the period of software update through various technical means including online activation or certain algorithm embedded in the license keys. The Licensee has no rights to reverse engineer, decompile, or disassemble the algorithm.

15. CHOICE OF LAW AND COURT OF JURISDICTION

15.1 The Agreement shall be governed by the laws of the State of California, and applicable United States federal law.

15.2 Any suit or proceeding arising out of this Agreement shall be brought only in a court located in Santa Clara County, California, and the parties submit to the exclusive jurisdiction and venue of such courts; provided, however, that Crystal Instruments may seek injunctive relief for any breach of this Agreement by Licensee in any court that would otherwise have jurisdiction over Licensee.

16. GENERAL PROVISIONS

16.1 Failure by Crystal Instruments to exercise or enforce any rights hereunder shall not be deemed to be a waiver of any such rights nor affect the exercise or enforcement thereof at any time or times thereafter.

16.2 If any provision or part of this Agreement is or is held by any court of competent jurisdiction to be unenforceable or invalid, such unenforceability or invalidity shall not affect the enforceability of any other provision.

16.3 This Agreement constitutes the entire agreement between the parties with respect to its subject matter and supersedes all prior or contemporaneous understandings regarding that subject matter. No amendment to or modification of this Agreement will be binding unless in writing and signed by an authorized officer of Crystal Instruments.

16.4 Licensee may not transfer or assign Licensee's rights under this Agreement to any third party without the prior written consent of Crystal Instruments, including by operation of law.

17. THIRD PARTY SOFTWARE LICENSE/NOTICES

Crystal Instruments Software uses a number of software products from 3rd parties that are under one of the following licenses, Apache License, GPL License, LGPL License and MIT License. Please contact Crystal Instruments to obtain the most updated list of 3rd party software that are incorporated in the Software.

License Type Definition

***Apache License**

Apache License is a free software license authored by the Apache Software Foundation (ASF). The Apache License requires preservation of the copyright notice and disclaimer. Like any free software license, the Apache License allows the user of the software the freedom to use the software for any purpose, to distribute it, to modify it, and to distribute modified versions of the software, under the terms of the license, without concern for royalties.

The 2.0 version of the Apache License was approved by the ASF in 2004. The goals of this license revision have been to reduce the number of frequently asked questions, to allow the license to be reusable without modification by any project (including non-ASF projects), to allow the license to be included by reference instead of listed in every file, to clarify the license on submission of contributions, to require a patent license on contributions that necessarily infringe the contributor's own patents, and to move comments regarding Apache and other inherited attribution notices to a location outside the license terms

***GPL License**
The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software. Software that ensures that these rights are retained is called free software. The license was originally written by Richard Stallman of the Free Software Foundation (FSF) for the GNU project.

***LGPL License**
LGPL (formerly the GNU Library General Public License) is a free software license published by the Free Software Foundation (FSF). The LGPL allows developers and companies to use and integrate LGPL software into their own (even proprietary) software without being required (by the terms of a strong copyleft) to release the source code of their own software-parts.

***MIT License**
The MIT License is a permissive free software license originating at the Massachusetts Institute of Technology (MIT)。 The MIT License is compatible with many copyleft licenses, such as the GNU General Public License (GNU GPL). Any software licensed under the terms of the MIT License can be integrated with software licensed under the terms of the GNU GPL.

--- Updated May 11, 2022