90_GenerateMDFs

March 22, 2020

1 Generate MDF Files

Generate MDF Files.

If your MDF files are locked away, prorietary, or otherwise unavailable. Generate simple signals for analysis.

[1]:
```
import IPython.display as display
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
```

2 Signal Generators.

Simplified versions generators with 3 inputs:

- **t**ime [s]
- **A**mplitude [Unit]
- **f**requency [Hz]

[2]:
```
def sine(t, A=1, f=1):
    """Generate sine wave.
    t: time vector [s]
    A: amplitude [Unit]
    f: frequency (Hz)
    """
    sine_ = A * np.sin(
        2 * np.pi * f * t
    )
    return sine_


def cos(t, A=1, f=1):
    """Generate cosine wave.
    t: time vector [s]
    A: amplitude [Unit]
    f: frequency (Hz)
```

```
    """
    cos_ = A * np.sin(
        2 * np.pi * f * t
    )
    return cos_


def square(t, A=1, f=1):
    """Generate square wave.
    t: time vector [s]
    A: amplitude [Unit]
    f: frequency (Hz)
    """
    square_ = A * scipy.signal.square(
        2 * np.pi * f * t
    )
    return square_


def sawtooth(t, A=1, f=1):
    """Generate sawtooth wave.
    t: time vector [s]
    A: amplitude [Unit]
    f: frequency (Hz)
    """
    sawtooth_ = A * scipy.signal.sawtooth(
        2 * np.pi * f * t,
        width=1,
    )
    return sawtooth_


def triangle(t, A=1, f=1):
    """Generate triange wave.
    t: time vector [s]
    A: amplitude [Unit]
    f: frequency (Hz)
    """
    triangle_ = A * scipy.signal.sawtooth(
        2 * np.pi * f * t,
        width=0.5,
    )
    return triangle_
signal_generators = [sine, cos, square, sawtooth, triangle]
```

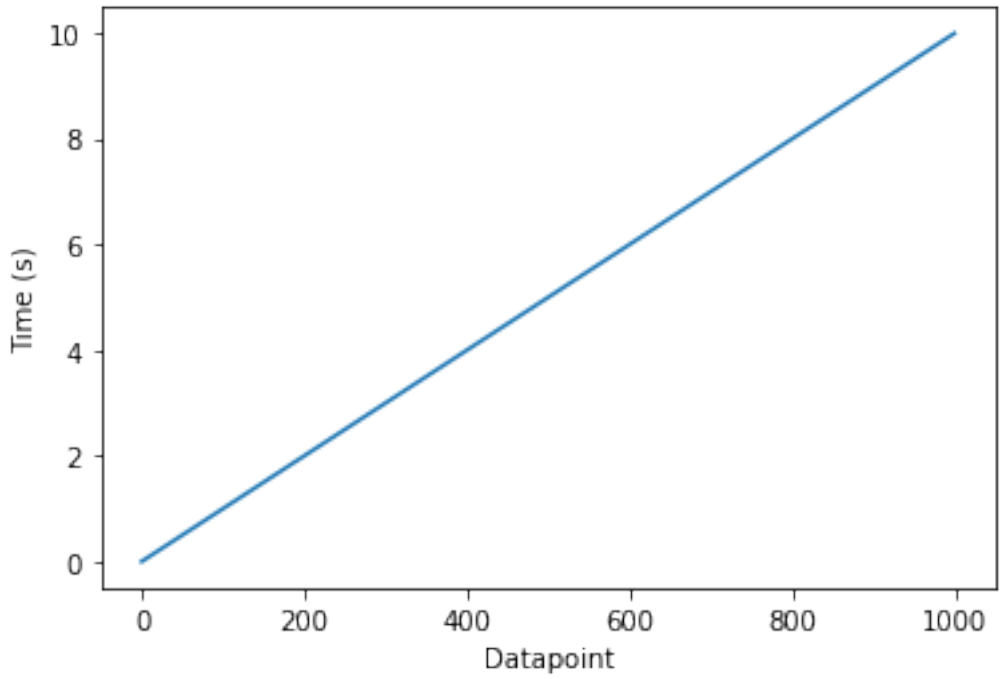
2.1 Signal Generator Usage

Generate a 100 Hz signal from 0 to 10 seconds.

[3]:
```
tf = 10
f  = 100
# Multiple time vectors
t = np.arange(0, tf, 1/f, dtype=np.float32)
```

[4]:
```
plt.plot(t)
plt.xlabel("Datapoint")
plt.ylabel("Time (s)")
```

[4]:
```
Text(0, 0.5, 'Time (s)')
```



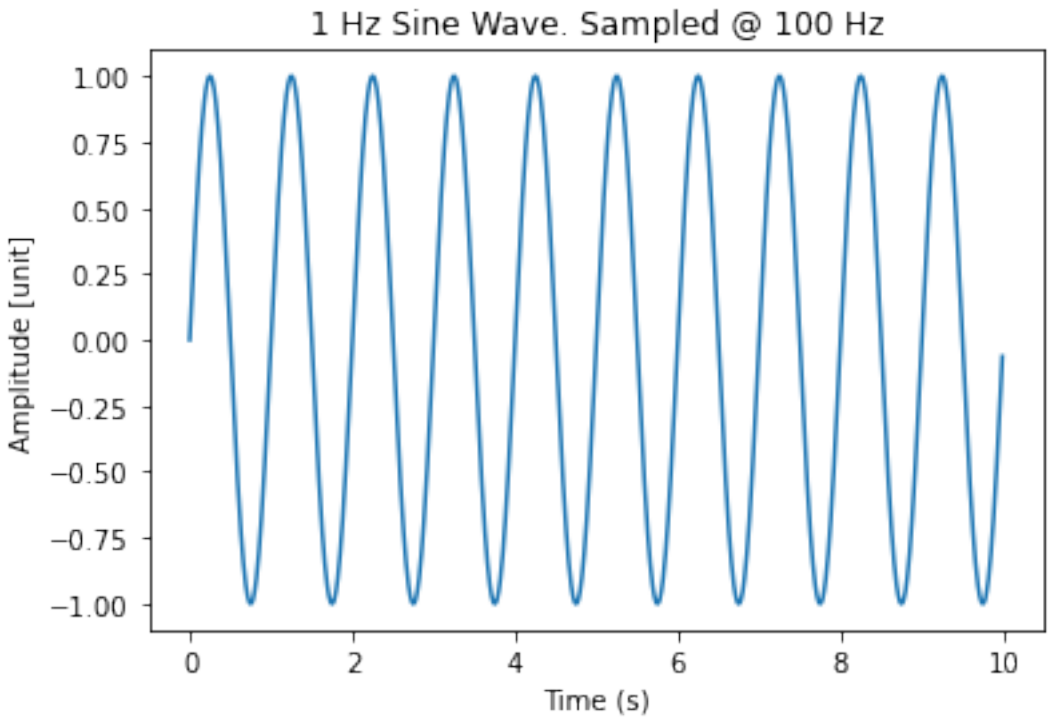
2.1.1 Sine Wave

Generate & plot a 1 Hz sine wave with amplitude 1.

[5]:
```
sine_signal = sine(t, A=1.0, f=1.0)
```

[6]:
```
plt.plot(t, sine_signal)
plt.xlabel("Time (s)")
plt.ylabel("Amplitude [unit]")
plt.title("1 Hz Sine Wave. Sampled @ 100 Hz")
```

[6]: Text(0.5, 1.0, '1 Hz Sine Wave. Sampled @ 100 Hz')



2.2 Save MDF File

[7]:
```
import asammdf
```

[8]:
```
from asammdf import MDF, Signal
```

[9]:
```
mdf = MDF(
        version='4.11',
    )
mdf.append(
    signals=signals,
    common_timebase=False,
)
o = mdf.save(
    dst=str(channel_path),
    overwrite=True,
    compression=2,
)
```

```
␣
↪----------------------------------------------------------------------------
```

```
NameError                                 Traceback (most recent call␣
↪last)

<ipython-input-9-bbac902337e1> in <module>
  3     )
  4 mdf.append(
----> 5     signals=signals,
  6     common_timebase=False,
  7 )


NameError: name 'signals' is not defined
```