Using Pandas with a Cartesian 3D Vector Class

https://github.com/t-o-k/scikit-vectors (https://github.com/t-o-k/scikit-vectors)

In [1]:
```
from datetime import datetime
import numpy as np
import pandas as pd

from skvectors import create_class_Cartesian_3D_Vector
```

In [2]:
```
date_rng = pd.date_range(start='2017-01-01', end='2017-01-08', freq='H')

date_rng
```

Out[2]:
```
DatetimeIndex(['2017-01-01 00:00:00', '2017-01-01 01:00:00',
               '2017-01-01 02:00:00', '2017-01-01 03:00:00',
               '2017-01-01 04:00:00', '2017-01-01 05:00:00',
               '2017-01-01 06:00:00', '2017-01-01 07:00:00',
               '2017-01-01 08:00:00', '2017-01-01 09:00:00',
               ...
               '2017-01-07 15:00:00', '2017-01-07 16:00:00',
               '2017-01-07 17:00:00', '2017-01-07 18:00:00',
               '2017-01-07 19:00:00', '2017-01-07 20:00:00',
               '2017-01-07 21:00:00', '2017-01-07 22:00:00',
               '2017-01-07 23:00:00', '2017-01-08 00:00:00'],
              dtype='datetime64[ns]', length=169, freq='H')
```

In [3]:

```
S3 = \
    create_class_Cartesian_3D_Vector(
        name = 'S3',
        component_names = 'xyz',
        brackets = '<>',
        sep = ', ',
        cnull = pd.Series(0, index=date_rng),
        cunit = pd.Series(1, index=date_rng),
        functions = \
            {
                'not': np.logical_not,
                'and': np.logical_and,
                'or': np.logical_or,
                'all': np.all,
                'any': np.any,
                'min': np.minimum,
                'max': np.maximum,
                'abs': np.absolute,
                'int': np.rint,
                'ceil': np.ceil,
                'copysign': np.copysign,
                'log10': np.log10,
                'cos': np.cos,
                'sin': np.sin,
                'atan2': np.arctan2,
                'pi': np.pi
            }
    )
```

In [4]:

```
S3.component_null().head()
```

Out[4]:

```
2017-01-01 00:00:00    0
2017-01-01 01:00:00    0
2017-01-01 02:00:00    0
2017-01-01 03:00:00    0
2017-01-01 04:00:00    0
Freq: H, dtype: int64
```

In [5]:
```
S3.component_unit().head()
```

Out[5]:
```
2017-01-01 00:00:00    1
2017-01-01 01:00:00    1
2017-01-01 02:00:00    1
2017-01-01 03:00:00    1
2017-01-01 04:00:00    1
Freq: H, dtype: int64
```

In [6]:
```
clength = len(date_rng)

clength
```

Out[6]:
```
169
```

In [7]:
```
u = \
    S3(
        np.random.randint(0, 100, size=clength),
        np.random.randint(0, 100, size=clength),
        np.random.randint(0, 100, size=clength)
    )
u -= 50

u(pd.Series.head)
```

Out[7]:
```
S3(x=2017-01-01 00:00:00    29
2017-01-01 01:00:00   -48
2017-01-01 02:00:00   -42
2017-01-01 03:00:00    -8
2017-01-01 04:00:00   -30
Freq: H, dtype: int64, y=2017-01-01 00:00:00   -36
2017-01-01 01:00:00   -20
2017-01-01 02:00:00    49
2017-01-01 03:00:00    34
2017-01-01 04:00:00    39
Freq: H, dtype: int64, z=2017-01-01 00:00:00   -26
2017-01-01 01:00:00     8
2017-01-01 02:00:00   -23
2017-01-01 03:00:00    -1
2017-01-01 04:00:00   -41
Freq: H, dtype: int64)
```

In [8]:
```
v = S3(1, 2, 3)

v(pd.Series.tail)
```

Out[8]:
```
S3(x=2017-01-07 20:00:00    1
2017-01-07 21:00:00    1
2017-01-07 22:00:00    1
2017-01-07 23:00:00    1
2017-01-08 00:00:00    1
Freq: H, dtype: int64, y=2017-01-07 20:00:00    2
2017-01-07 21:00:00    2
2017-01-07 22:00:00    2
2017-01-07 23:00:00    2
2017-01-08 00:00:00    2
Freq: H, dtype: int64, z=2017-01-07 20:00:00    3
2017-01-07 21:00:00    3
2017-01-07 22:00:00    3
2017-01-07 23:00:00    3
2017-01-08 00:00:00    3
Freq: H, dtype: int64)
```

In [9]:
```
w = u.cross(v).normalize()

w(pd.Series.tail)
```

Out[9]:
```
S3(x=2017-01-07 20:00:00   -0.922814
2017-01-07 21:00:00    0.282078
2017-01-07 22:00:00    0.434147
2017-01-07 23:00:00    0.682810
2017-01-08 00:00:00   -0.227508
Freq: H, dtype: float64, y=2017-01-07 20:00:00    0.381548
2017-01-07 21:00:00    0.752207
2017-01-07 22:00:00    0.676028
2017-01-07 23:00:00   -0.692163
2017-01-08 00:00:00   -0.773527
Freq: H, dtype: float64, z=2017-01-07 20:00:00    0.053239
2017-01-07 21:00:00   -0.595497
2017-01-07 22:00:00   -0.595401
2017-01-07 23:00:00    0.233839
2017-01-08 00:00:00    0.591520
Freq: H, dtype: float64)
```

In [10]:
```
c = 2.5 * w(np.ceil)

c(pd.Series.tail)
```

Out[10]:
```
S3(x=2017-01-07 20:00:00   -0.0
2017-01-07 21:00:00    2.5
2017-01-07 22:00:00    2.5
2017-01-07 23:00:00    2.5
2017-01-08 00:00:00   -0.0
Freq: H, dtype: float64, y=2017-01-07 20:00:00    2.5
2017-01-07 21:00:00    2.5
2017-01-07 22:00:00    2.5
2017-01-07 23:00:00   -0.0
2017-01-08 00:00:00   -0.0
Freq: H, dtype: float64, z=2017-01-07 20:00:00    2.5
2017-01-07 21:00:00   -0.0
2017-01-07 22:00:00   -0.0
2017-01-07 23:00:00    2.5
2017-01-08 00:00:00    2.5
Freq: H, dtype: float64)
```

In [11]:
```
w.x.tail()
```

Out[11]:
```
2017-01-07 20:00:00   -0.922814
2017-01-07 21:00:00    0.282078
2017-01-07 22:00:00    0.434147
2017-01-07 23:00:00    0.682810
2017-01-08 00:00:00   -0.227508
Freq: H, dtype: float64
```

In [12]:
```
type(w.x)
```

Out[12]:
```
pandas.core.series.Series
```

In [13]:
```
w.x.index[-5:]
```

Out[13]:
```
DatetimeIndex(['2017-01-07 20:00:00', '2017-01-07 21:00:00',
               '2017-01-07 22:00:00', '2017-01-07 23:00:00',
               '2017-01-08 00:00:00'],
              dtype='datetime64[ns]', freq='H')
```

In [14]:
```
w.x.values[-5:]
```

Out[14]:
```
array([-0.92281444,  0.28207761,  0.43414662,  0.6828098 , -0.22750788])
```

In [15]:
```
type(w.x.values)
```

Out[15]:
```
numpy.ndarray
```

In [16]:
```
df = pd.DataFrame(w.as_dict())

df.tail()
```

Out[16]:

	x	y	z
2017-01-07 20:00:00	-0.922814	0.381548	0.053239
2017-01-07 21:00:00	0.282078	0.752207	-0.595497
2017-01-07 22:00:00	0.434147	0.676028	-0.595401
2017-01-07 23:00:00	0.682810	-0.692163	0.233839
2017-01-08 00:00:00	-0.227508	-0.773527	0.591520

In [17]:
```
a = np.array(w).T

a[-5:]
```

Out[17]:
```
array([[-0.92281444,  0.38154828,  0.05323929],
       [ 0.28207761,  0.75220697, -0.59549718],
       [ 0.43414662,  0.67602831, -0.59540108],
       [ 0.6828098 , -0.69216336,  0.23383897],
       [-0.22750788, -0.77352678,  0.59152048]])
```

In [ ]:
```

```