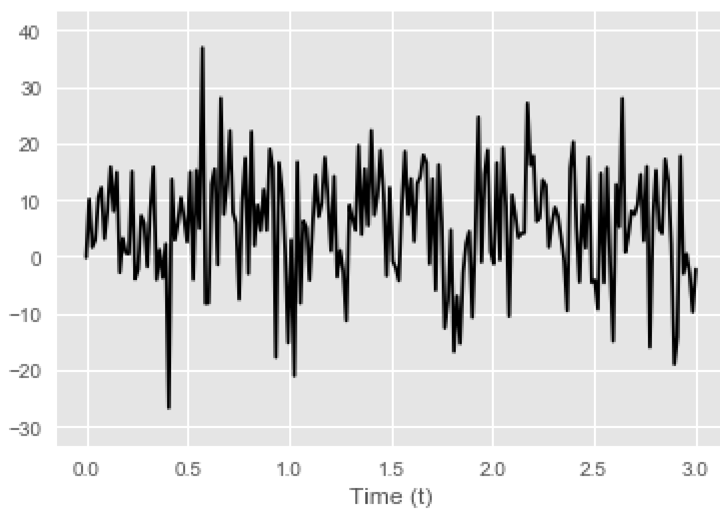
Handout 23: Power Spectral Density

Note: Gaussian process functionality is still in experimental phase and not available in the official Symbulate release.

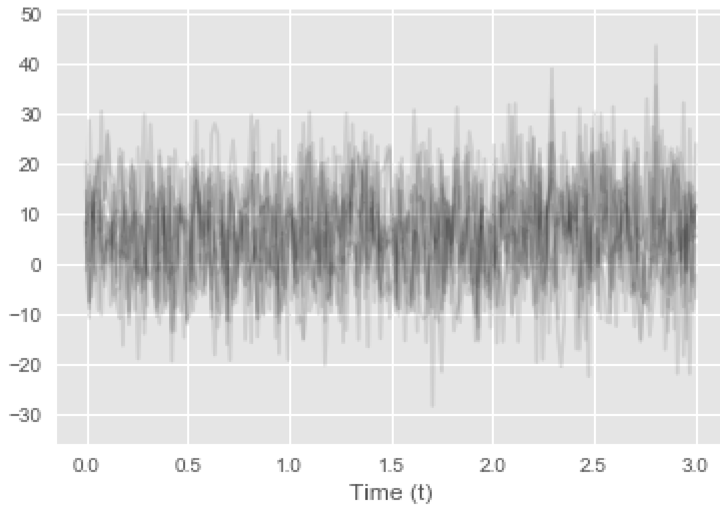
In [1]:
```
from symbulate import *
%matplotlib inline
```

Exer 23.2

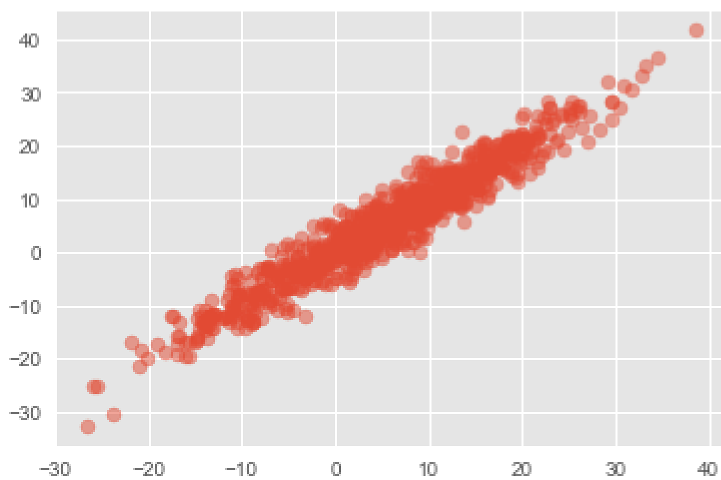
In [2]:
```
P = GaussianProcess(mean_fn=lambda t: sqrt(30), cov_fn=lambda s, t: 100 *
exp(-40000 * (s - t) ** 2))
X = RandomProcess(P, TimeIndex(fs=inf))
X.sim(1).plot(alpha=1, tmax=3)
```



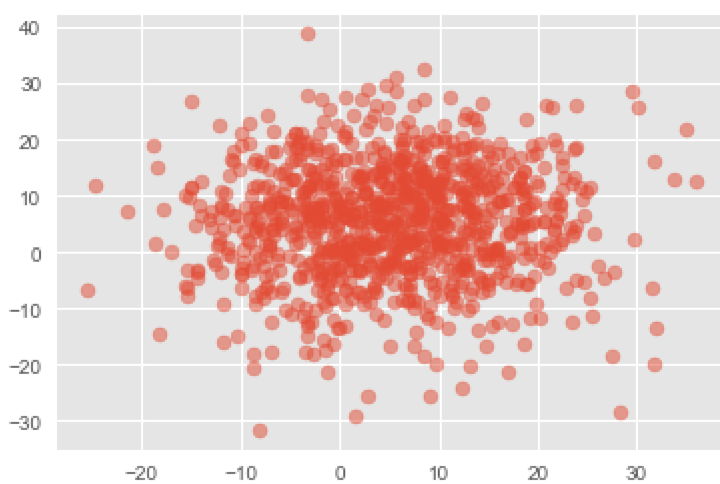
In [3]:
```
X.sim(10).plot(tmax=3)
```



In [4]:
```
(X[0] & X[0.001]).sim(1000).plot()
```



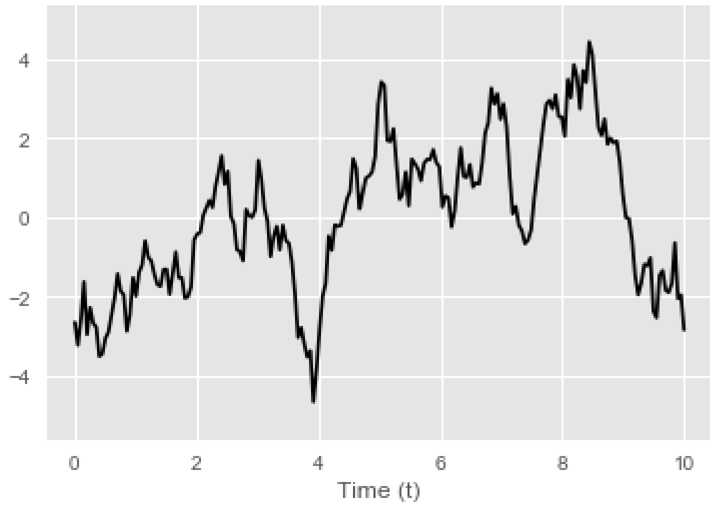
In [5]:
```
(X[0] & X[0.01]).sim(1000).plot()
```



Exer 23.3

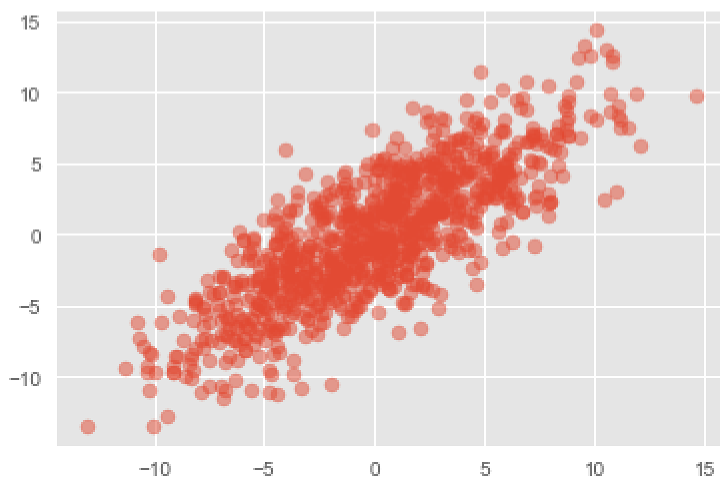
In [6]:
```
def R1(s, t):
    return 20 * max(0, 1 - abs(s-t) / 5)

P1 = GaussianProcess(mean_fn=lambda t: 0, cov_fn=R1)
X1 = RandomProcess(P1, TimeIndex(fs=inf))
X1.sim(1).plot(alpha=1)
```



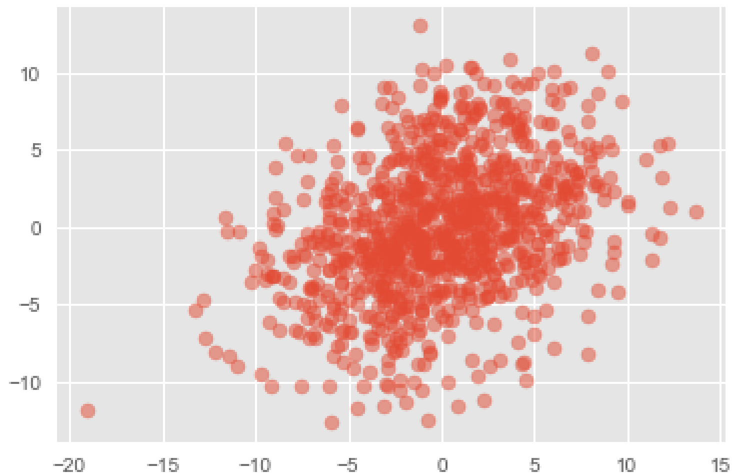
In [7]:
```
xy = (X1[1] & X1[2]).sim(1000)
xy.plot()
xy.cov()
```

Out[7]: 16.765642709184025



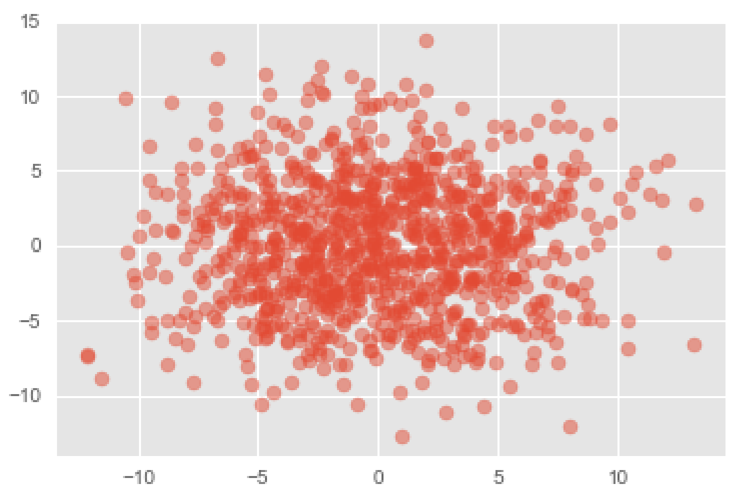
In [8]:
```
xy = (X1[1] & X1[4]).sim(1000)
xy.plot()
xy.cov()
```

Out[8]: 6.9284262499788278



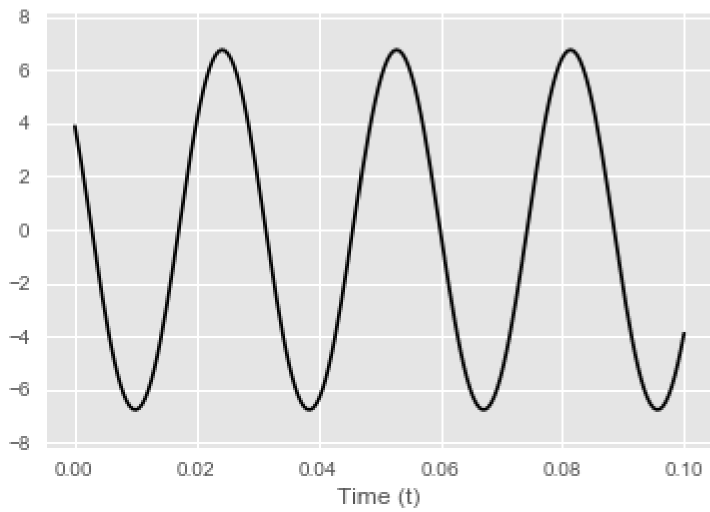
In [9]:
```
xy = (X1[1] & X1[6]).sim(1000)
xy.plot()
xy.cov()
```

Out[9]: 0.43592533732092192



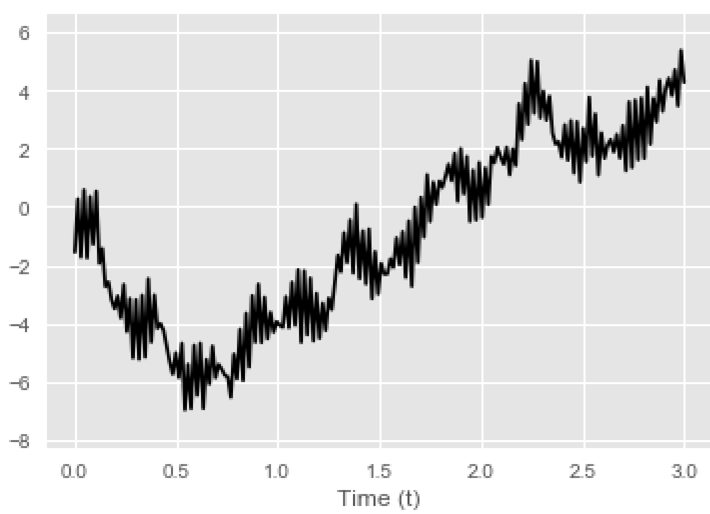
In [10]:
```
def R2(s, t):
    return 6 * cos(2 * pi * 35 * (s - t))

P2 = GaussianProcess(mean_fn=lambda t: 0, cov_fn=R2)
X2 = RandomProcess(P2, TimeIndex(fs=inf))
X2.sim(1).plot(alpha=1, tmax=0.1)
```



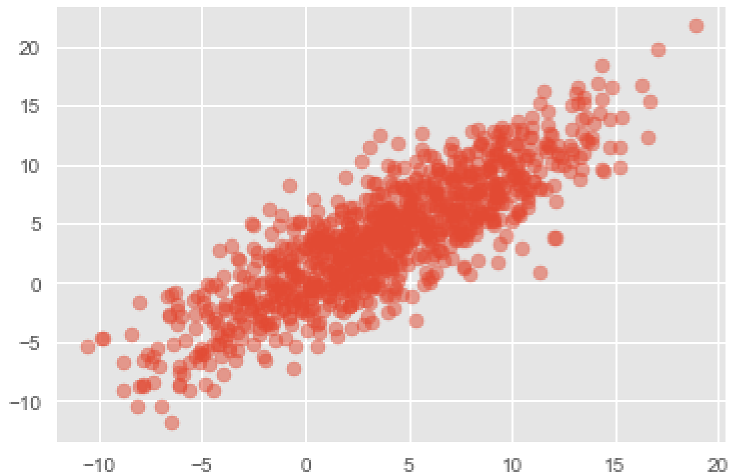
In [11]:
```
X1, X2 = AssumeIndependent(X1, X2)
Y = X1 + X2 + 4
```

In [12]:
```
Y.sim(1).plot(alpha=1, tmax=3)
```



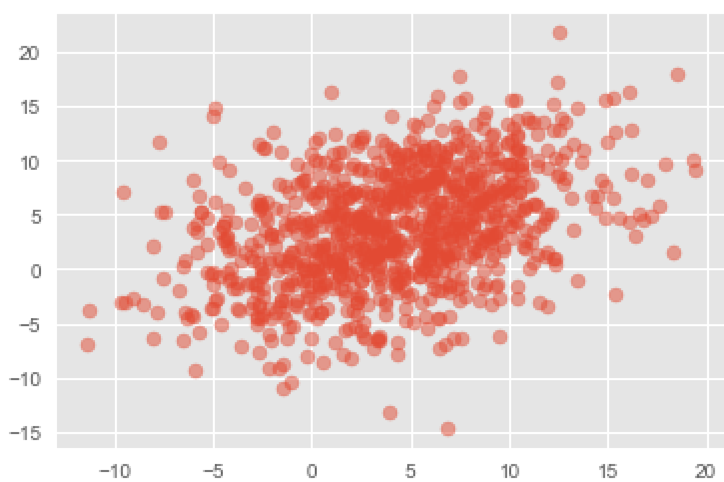
In [13]:
```
xy = (Y[1] & Y[2]).sim(1000)
xy.plot()
xy.cov()
```

Out[13]: 22.226042672361569



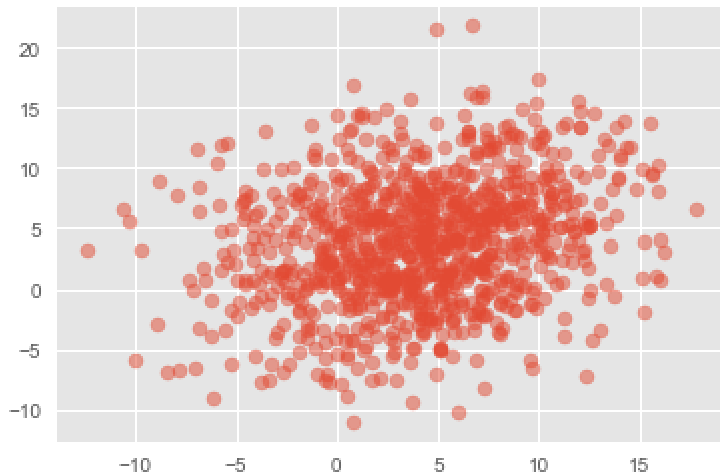
In [14]:
```
xy = (Y[1] & Y[2 + 1/70]).sim(1000)
xy.plot()
xy.cov()
```

Out[14]: 10.903348303071702



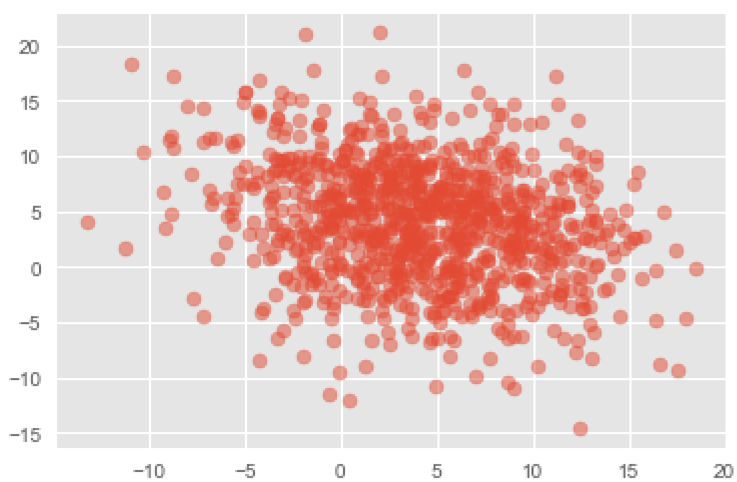
In [15]:
```
xy = (Y[1] & Y[6]).sim(1000)
xy.plot()
xy.cov()
```

Out[15]: 6.1427686541344739



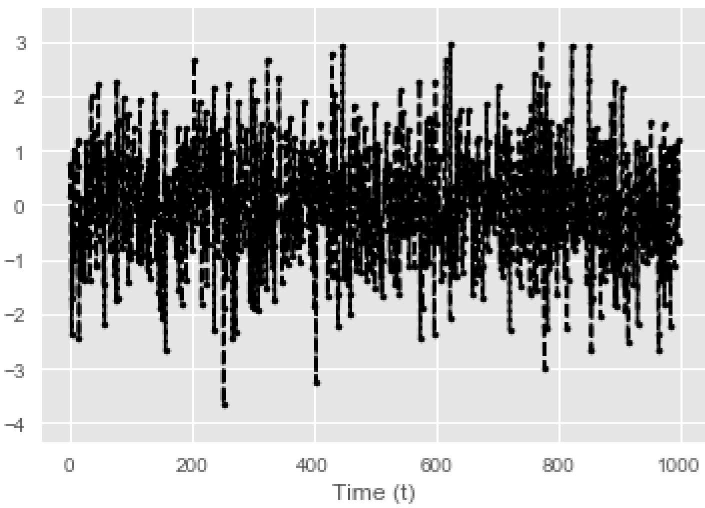
In [16]:
```
xy = (Y[1] & Y[6 + 1/70]).sim(1000)
xy.plot()
xy.cov()
```

Out[16]: -6.799977650748569



Exer 23.4

In [32]:
```
P = Normal(0, 1) ** inf
RandomProcess(P).sim(1).plot(alpha=1,tmax=1000)
```



In [ ]:

In [ ]: