In [2]:
```
import scipy
import pandas as pd
import matplotlib.pyplot as plt
```

In [13]:
```
import numpy as np
```

autocorrelation

Shift and calculate correlation directly

In [22]:
```
a = np.array([0,1,2,1]*2)
values = pd.DataFrame(a)

# using shift function to shift the values.
dataframe = pd.concat([values.shift(2), values], axis=1)
```

In [24]:
```
dataframe.head(10)
```

Out[24]:

	0	0
0	NaN	0
1	NaN	1
2	0.0	2
3	1.0	1
4	2.0	0
5	1.0	1
6	0.0	2
7	1.0	1

In [25]:
```
dataframe.corr()
```

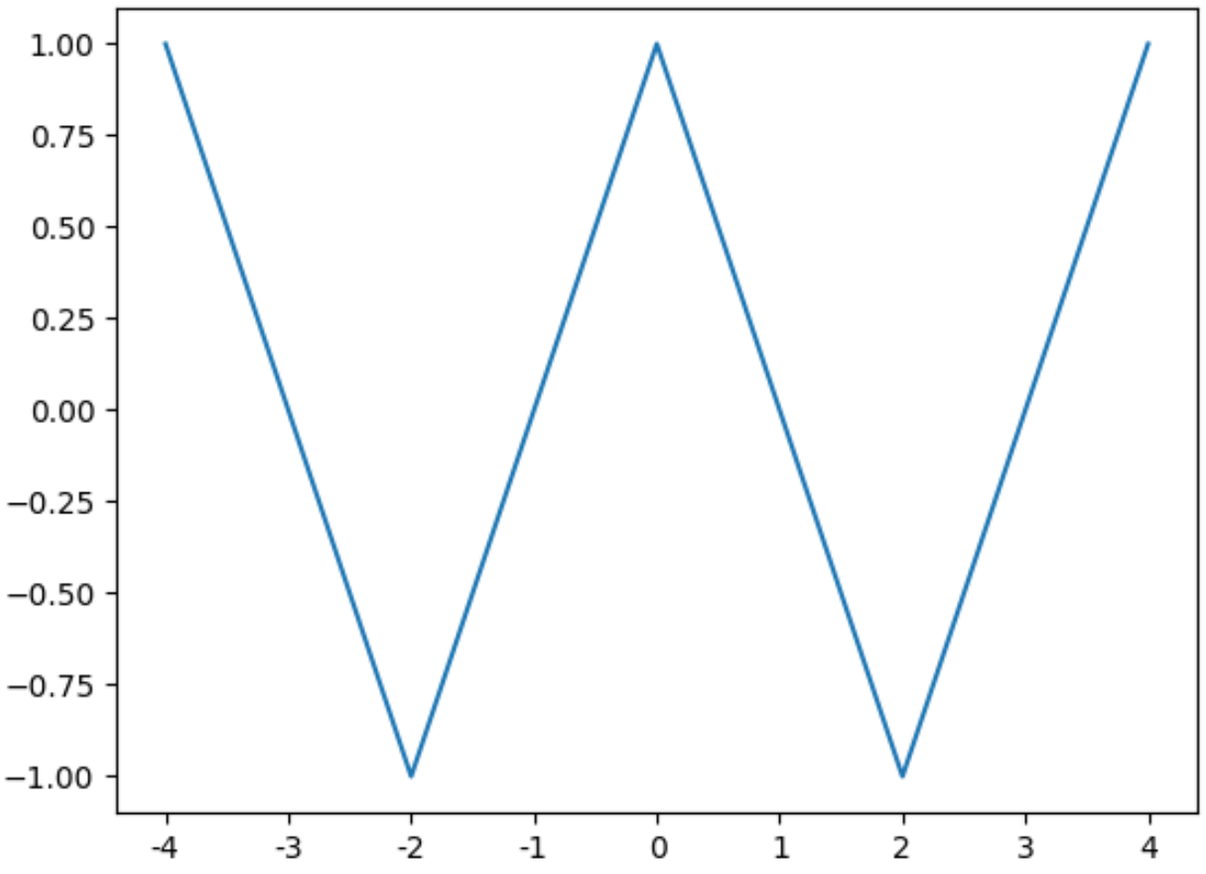
Out[25]:

	0	0
0	1.0	-1.0
0	-1.0	1.0

Use pandas autocorr function

In [26]:
```
signal_A = pd.Series(a)
```

In [27]:
```
plt.plot([signal_A.autocorr(i) for i in range(-4,5)])
plt.xticks(range(9), range(-4,5))
plt.show()
```

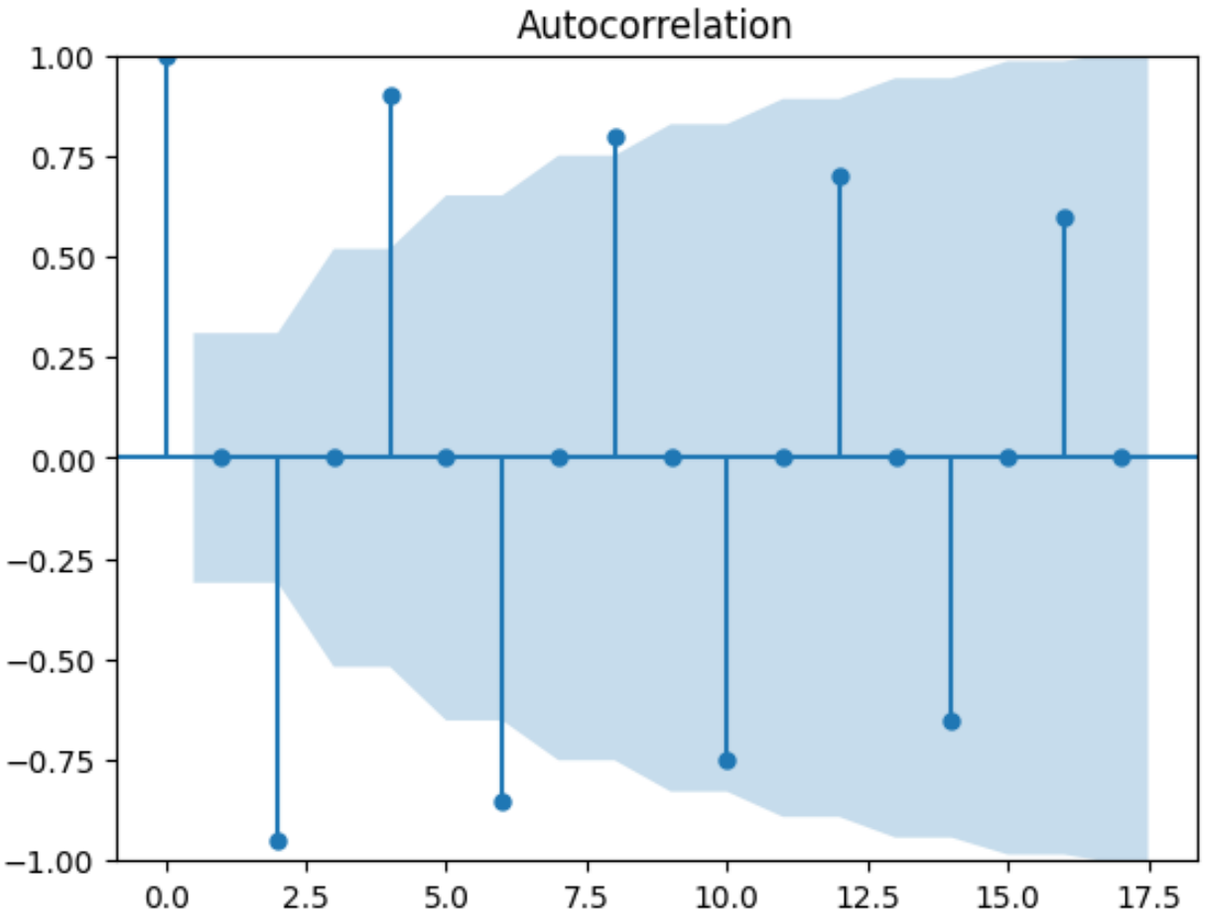


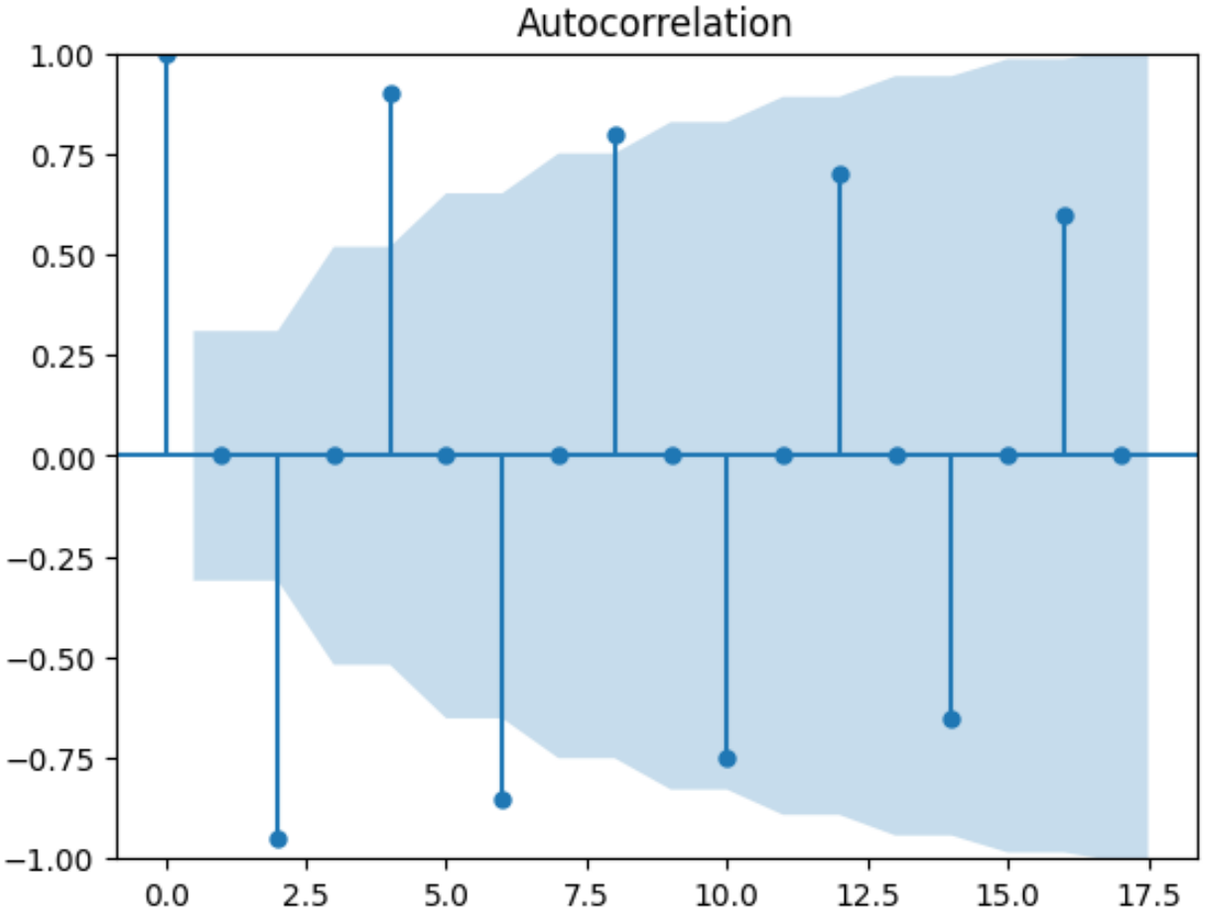
Use plot_acf directly from statsmodels

In [1]:
```
from statsmodels.graphics.tsaplots import plot_acf
```

In [12]:
```
plot_acf(signal_A)
```

Out[12]:


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


Autocorrelation
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

2D Gaussian

https://juanitorduz.github.io/multivariate_normal/

In [29]:
```
import seaborn as sns; sns.set()
```

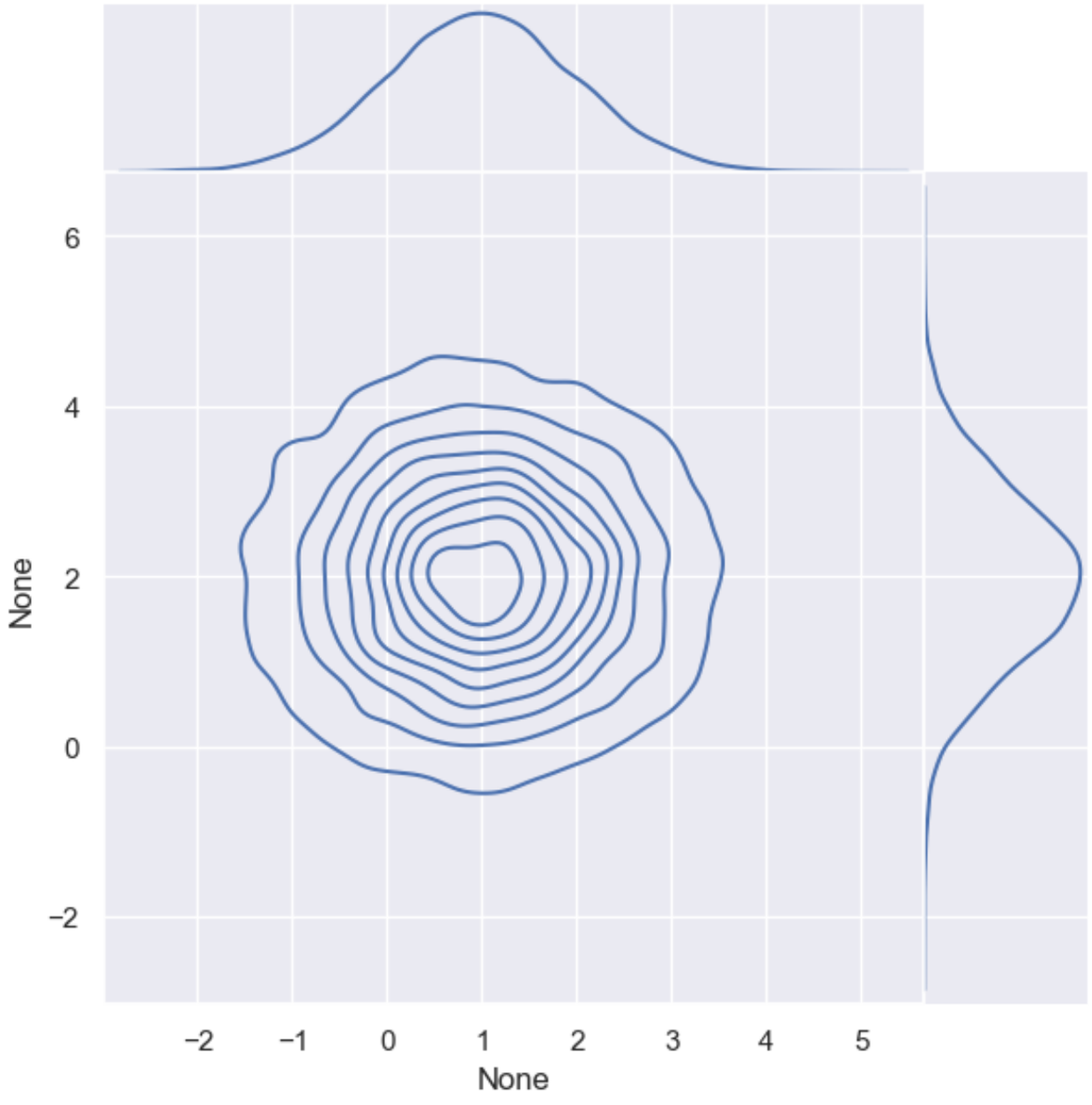
In [30]:
```
# Define dimension.
d = 2

# Set mean vector.
m = np.array([1, 2]).reshape(2, 1)
```

In [33]:
```
def generate_samples(K_0, m, d):
    epsilon = 0.0001
    K = K_0 + epsilon*np.identity(d)
    L = np.linalg.cholesky(K)
    n = 10000
    u = np.random.normal(loc=0, scale=1, size=d*n).reshape(d, n)

    x = m + np.dot(L, u)
    return x
```

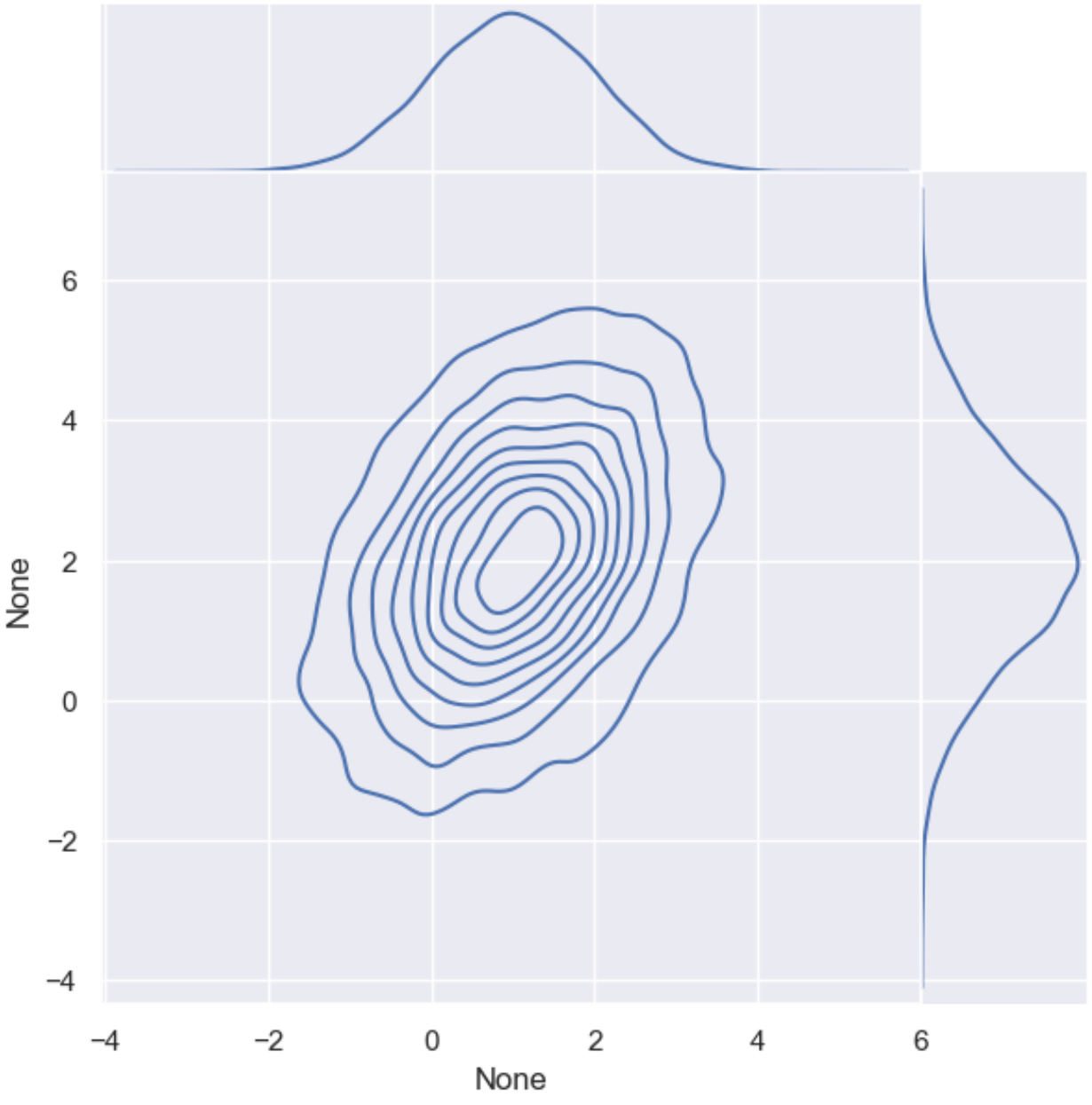
In [34]:
```
K_0 = np.array([[1, 0],
                [0, 1]])
x = generate_samples(K_0, m, d)
sns.jointplot(x=x[0], y=x[1], kind="kde", space=0);
```



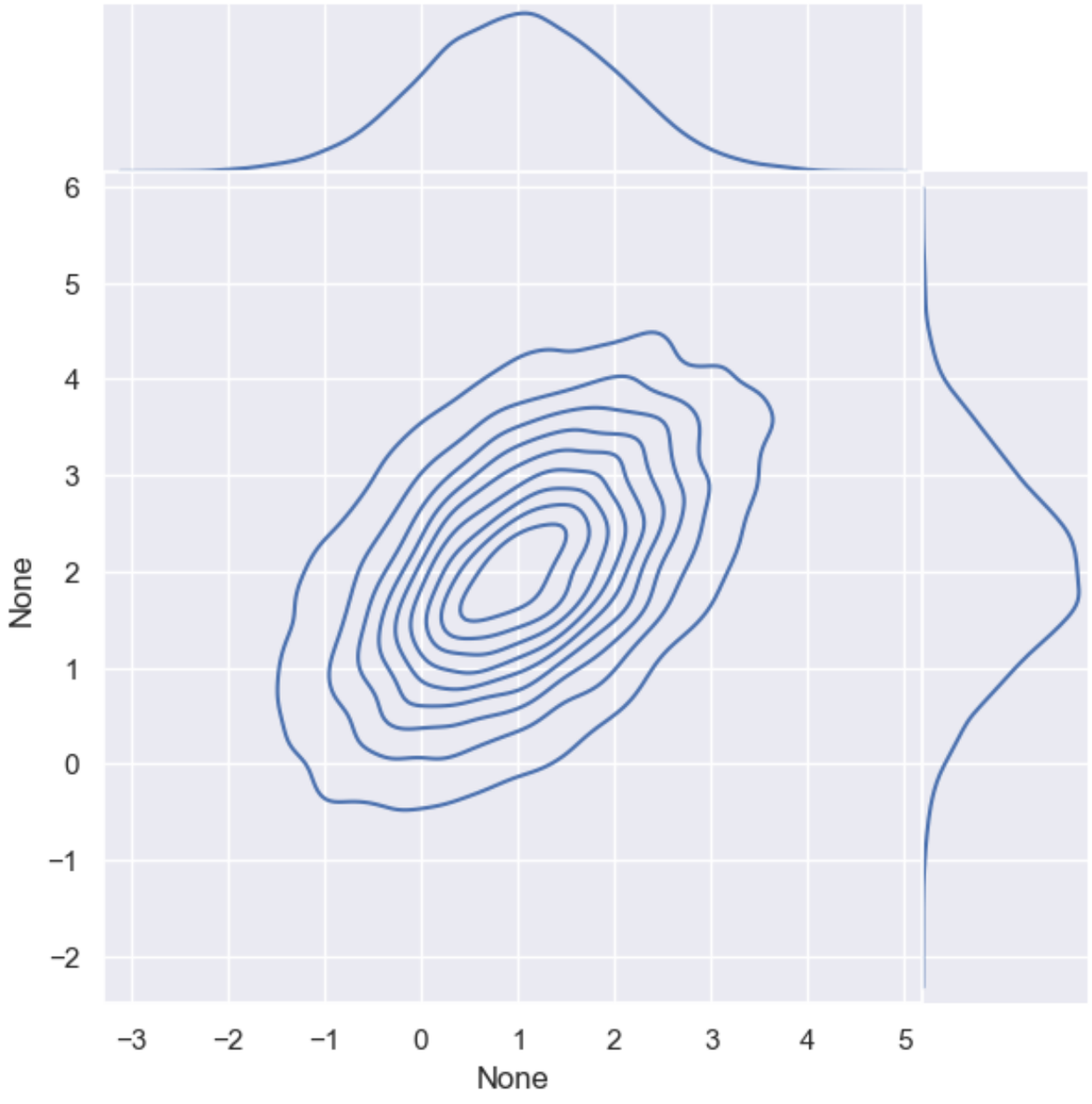
Rotation and translation

See
https://en.wikipedia.org/wiki/Gaussian_function#Meaning_of_parameters_for_the_genera
regarding the meaning of the covariance matrix.

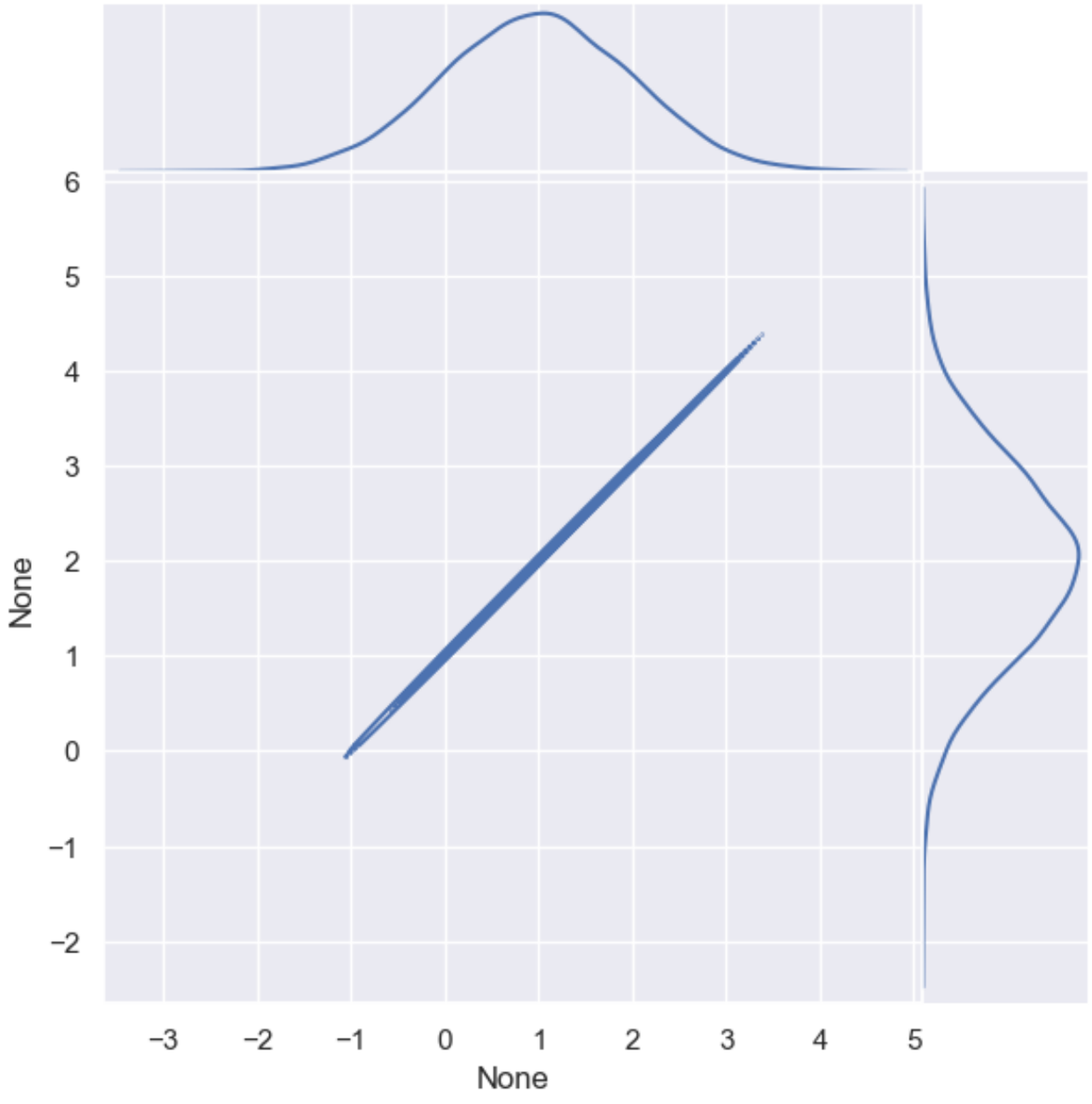
In [45]:
```
K_0 = np.array([[1, 0.5],
                [0.5, 2]])
x = generate_samples(K_0, m, d)
sns.jointplot(x=x[0], y=x[1], kind="kde", space=0);
```



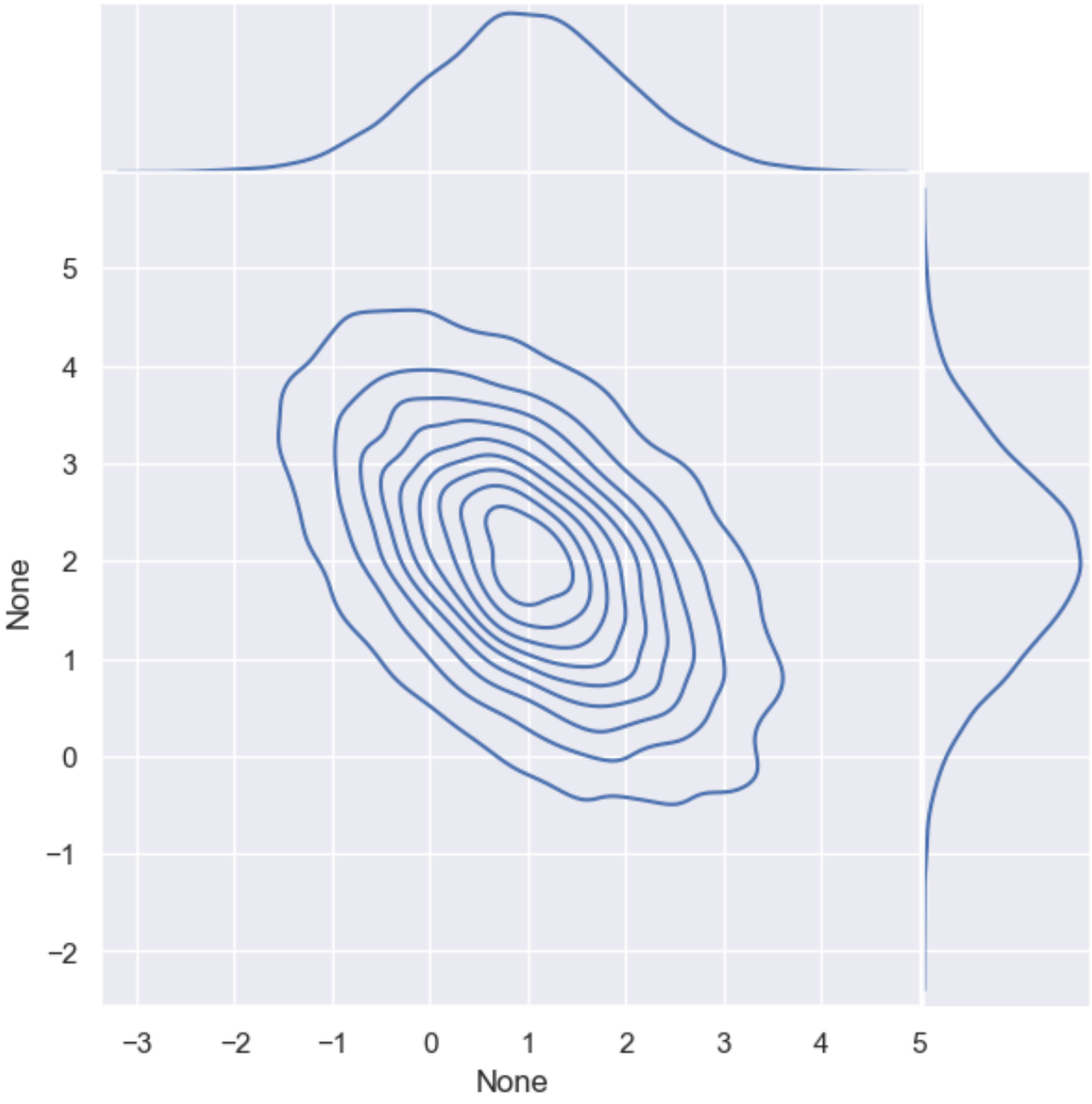
In [43]:
```
K_0 = np.array([[1, 0.5],
                [0.5, 1]])
x = generate_samples(K_0, m, d)
sns.jointplot(x=x[0], y=x[1], kind="kde", space=0);
```



In [37]:
```
K_0 = np.array([[1, 1],
                [1, 1]])
x = generate_samples(K_0, m, d)
sns.jointplot(x=x[0], y=x[1], kind="kde", space=0);
```



In [38]:
```
K_0 = np.array([[1, -0.5],
                [-0.5, 1]])
x = generate_samples(K_0, m, d)
sns.jointplot(x=x[0], y=x[1], kind="kde", space=0);
```


5
4
3
2
1
0
−1
−2
None
−3
−2
−1
0
1
2
3
4
5
None