Orthogonal Projection

In [1]:

```
#keep
import numpy as np
import numpy.linalg as la
```

In [2]:

```
#keep

# for in-line plots
%matplotlib inline

# for plots in a window
# %matplotlib qt

import matplotlib.pyplot as pt
from mpl_toolkits.mplot3d import Axes3D
```

Make two random 3D vectors:

In [3]:

```
#keep
np.random.seed(13)
x = np.random.randn(3)
y = np.random.randn(3)
```

Make them orthonormal:

In [4]:

```
y = y - y.dot(x)/x.dot(x)*x
x = x/la.norm(x)
y = y/la.norm(y)
```

Check:

In [5]:

```
#keep
print(y.dot(x))
print(la.norm(x))
print(la.norm(y))
```

```
7.6327832943e-17
1.0
1.0
```

Plot the two vectors:

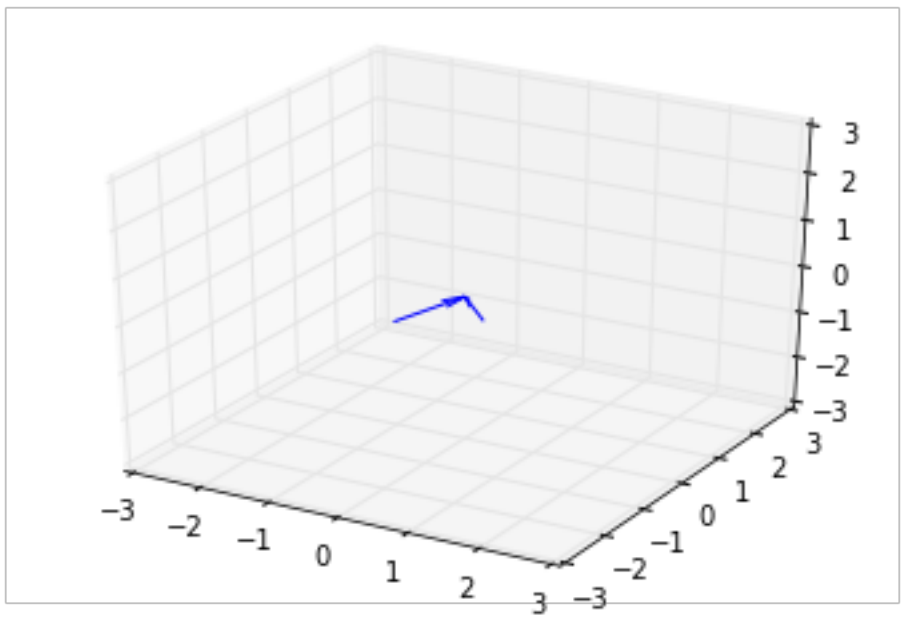
In [6]:

```
#keep
fig = pt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.set_xlim3d([-3, 3])
ax.set_ylim3d([-3, 3])
ax.set_zlim3d([-3, 3])

xy = np.array([x, y]).T
ax.quiver(
   0, 0, 0,
   xy[0], xy[1], xy[2],)
```

Out[6]:

```
<mpl_toolkits.mplot3d.art3d.Line3DCollection at 0x7fb167041cf8>
```



Make an array with the cornerpoints of a cube:

In [7]:

```
#keep
points = np.array([
    [-1,-1,-1],
    [-1,-1,1],
    [-1,1,-1],
    [-1,1,1],

    [1,-1,-1],
    [1,-1,1],
    [1,1,-1],
    [1,1,1],
])
```

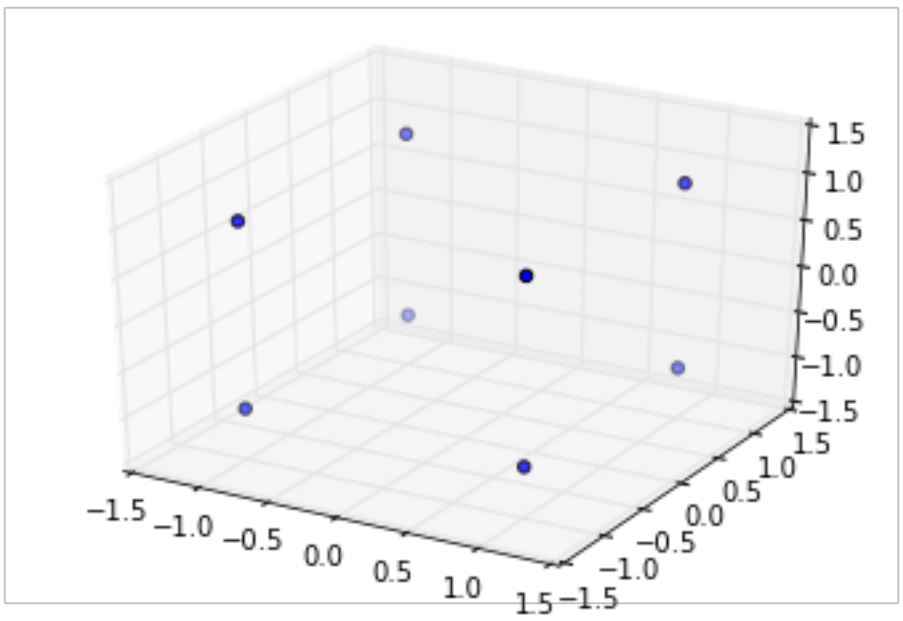
Plot them:

In [8]:

```
#keep
fig = pt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(points[:,0], points[:, 1], points[:, 2])
```

Out[8]:

```
<mpl_toolkits.mplot3d.art3d.Path3DCollection at 0x7fb166f61278>
```



Construct the projection matrix:

In [9]:

```
Q = np.array([
   x,y,np.zeros(3)
]).T
print(Q)
```

```
[[-0.68624693  0.65133881  0.        ]
 [ 0.72610421  0.63968157  0.        ]
 [-0.04286988  0.40812404  0.        ]]
```

In [10]:

```
P = Q.dot(Q.T)
print(P)
```

```
[[ 0.89517709 -0.08163735  0.29524635]
 [-0.08163735  0.93641985  0.22994143]
 [ 0.29524635  0.22994143  0.16840306]]
```

Check that $P^2 = P$:

In [11]:

```
#keep
la.norm(P.dot(P)-P)
```

Out[11]:

```
1.7880278822442372e-16
```

Project the points, assign to `proj_points`:

In [12]:

```
proj_points = np.einsum("ij,nj->ni", P, points)
```

In [13]:

```
#keep
fig = pt.figure()
ax = fig.add_subplot(111, projection='3d')
ax.scatter(points[:,0], points[:, 1], points[:, 2])
ax.scatter(proj_points[:,0], proj_points[:, 1], proj_points[:, 2], color="red")

xy = np.array([x, y]).T
ax.quiver(
   0, 0, 0,
   xy[0], xy[1], xy[2],)
```

Out[13]:

```
<mpl_toolkits.mplot3d.art3d.Line3DCollection at 0x7fb166f94080>
```