Max Kramer

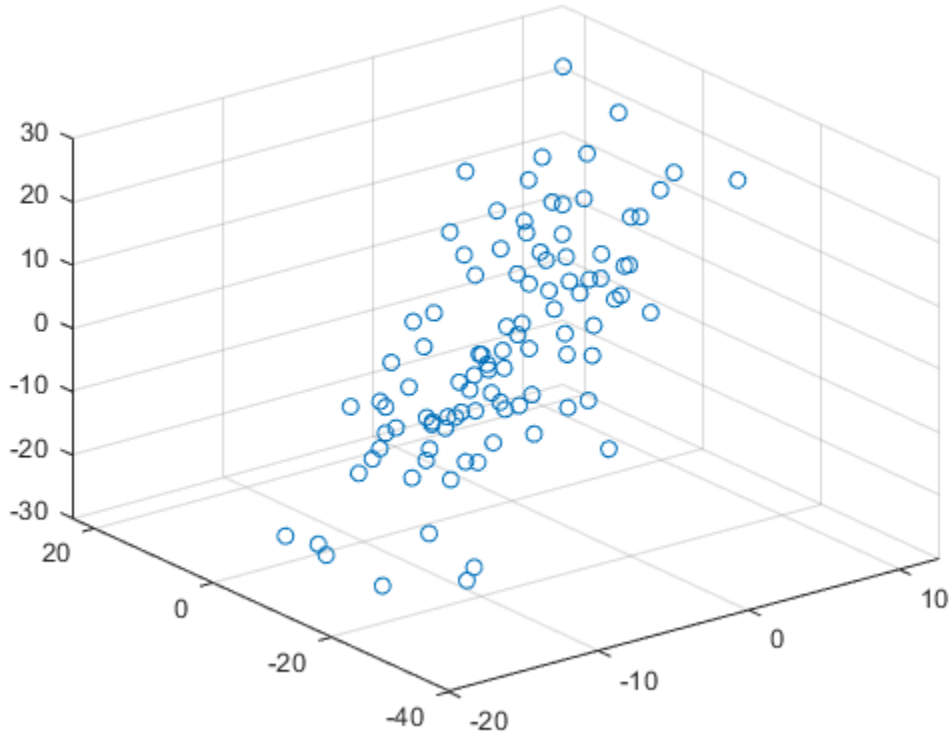
I affirm that I have adhered to the honor code on this assignment.

Hello again, scientist! I'll do all my writing in italics, and problems for you will be in ***bold.*** *Comment your code, and* explain your ideas in plaintext. *As a general rule, I expect you to do at least as much writing as I do. Code should be part of your solution, but I expect variables to be clear and explanation to involve complete sentences. Cite your sources; if you work with someone in the class on a problem, that's an extremely important source. Don't work alone.*

Problem F.04: Rotate the ellipse.

Let's look at a point cloud.

```
A = csvread('ell20.csv');
figure;
scatter3(A(:,1),A(:,2),A(:,3))
```



This is a collection of 100 points in R^3, sampled from an ellipse of three axis lengths, centered at the origin. (Okay, technically an ellipse is a two-dimensional object. This thing is called an ellipsoid. But, y'know, that's really just a fancy ellipse.) What's weird about this data is that it looks "almost" 2-dimensional; let's make some sense of that.

```
[U,S,V] = svd(A,'econ'); % 'econ' just means it skips the whole chunk
 of 0s, since they'll be 0ed out anyway
```

Explain how I know that the ratio of the length of the largest axis to the smallest is about 77. *I mean, okay, I made the ellipse. But assume that I forgot.*

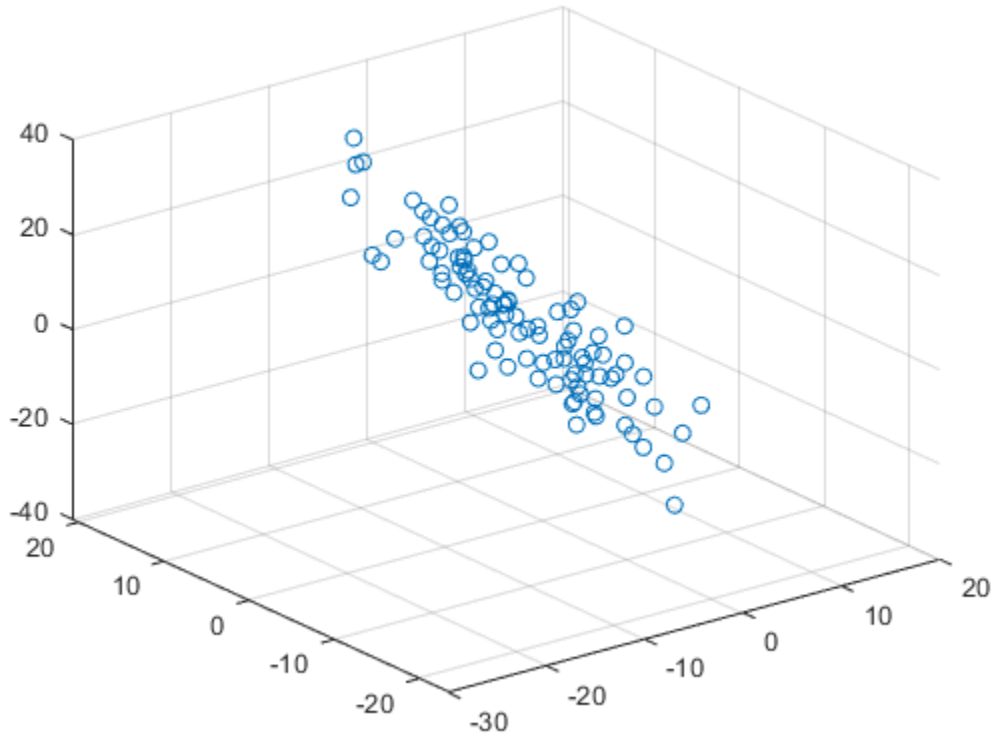
```
S(1,1) / S(3,3)
```

```
ans =

   77.2882
```

The matrix S contains the singular values of A. The largest singular value divided by the smallest singular value is the condition number of the matrix. This condition number is the ratio of the length of the largest axis to the smallest axis.

Rotate A so that the axes of the ellipse are sent to the standard axes for R^3, then plot. *This isn't as hard as it sounds; you can do it in one line, if you know what the SVD is telling you.*

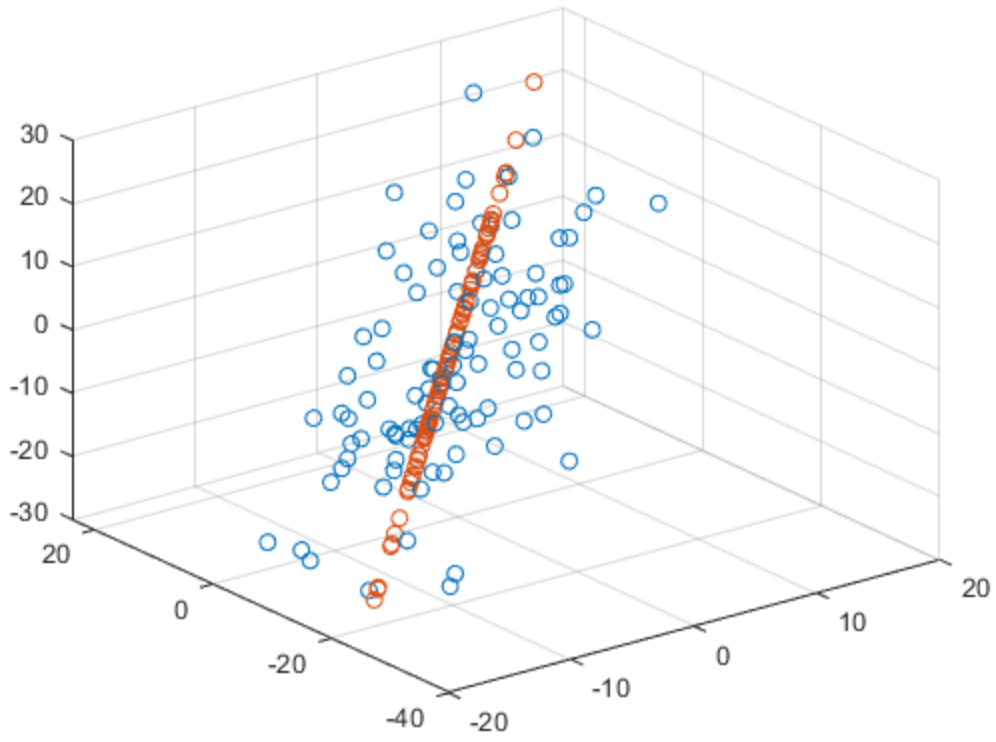
```
A2 = U*S*V;
figure;
scatter3(A2(:,1),A2(:,2),A2(:,3));
```



The SVD computes A by first rotating the domain to the principal axes from the standard axes (using V'). Then A is scaled (using S) and then the codomain is rotated from the standard axes to the principal axes (using U). To rotate A's domain (R^3), we multiply U*S*V rather than V' as V rotates the principal axes to the standard axes of the domain.

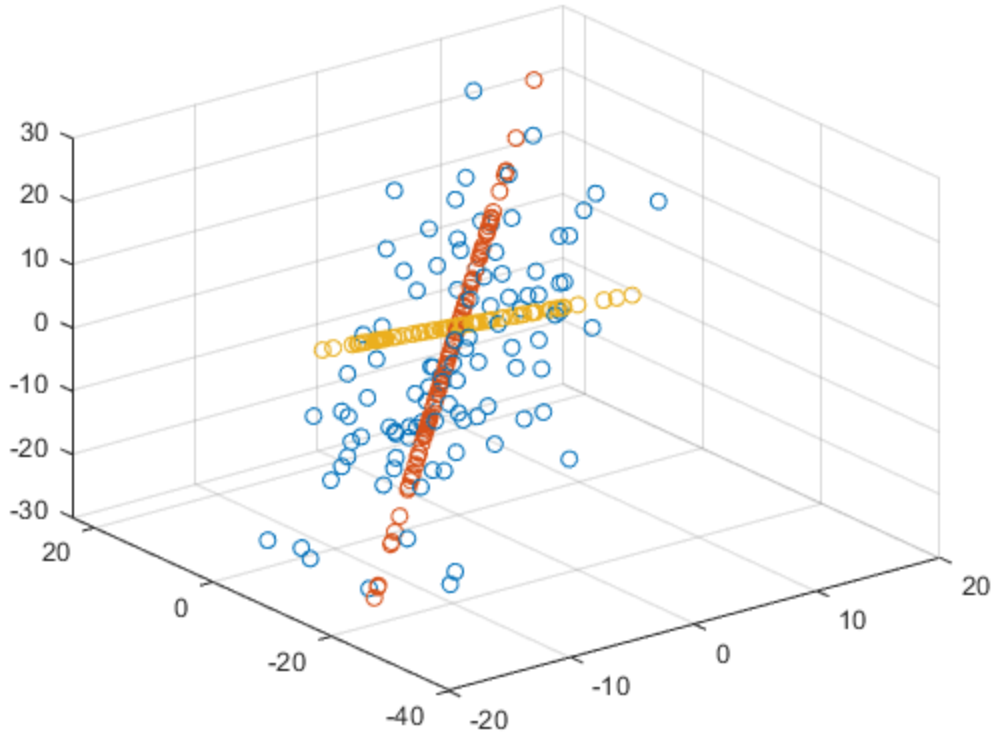
These axes are also called the "principal components" of A. More on that in a bit. For now, let's see what they look like in the original space.

```
proj1 = diag([1 0 0]);
S1 = proj1*S;
A1 = U*S1*V';

figure;
scatter3(A(:,1),A(:,2),A(:,3))
hold on;
scatter3(A1(:,1),A1(:,2),A1(:,3))
```



What I've just done is to project all the data onto the first principal axis. ***Show me, on the same set of axes, what happens when you project the data onto the second principal axis.*** *If you've done this right, your two axes should be orthogonal to each other and span "most" of the data. (Automatically!)*

```
proj2 = diag([0 1 0]);
S2 = proj2*S;
A2 = U*S2*V';

figure;
scatter3(A(:,1),A(:,2),A(:,3))
hold on;
scatter3(A1(:,1),A1(:,2),A1(:,3))
scatter3(A2(:,1),A2(:,2),A2(:,3))
```



The only difference here is the projection matrix. Instead of projecting onto the space of the 1st principal axis, we project onto the space of the second principal axis using the matrix proj2. This axis is in fact orthogonal to the first principal axis and the two axes seem to span the majority of the data.

Published with MATLAB® R2019b