Load data

```
import cdt
eisen = cdt.ExpressionProfile("supp2data.cdt")
```

```
X = array(eisen.num)
X.shape
```

```
(2467, 79)
```

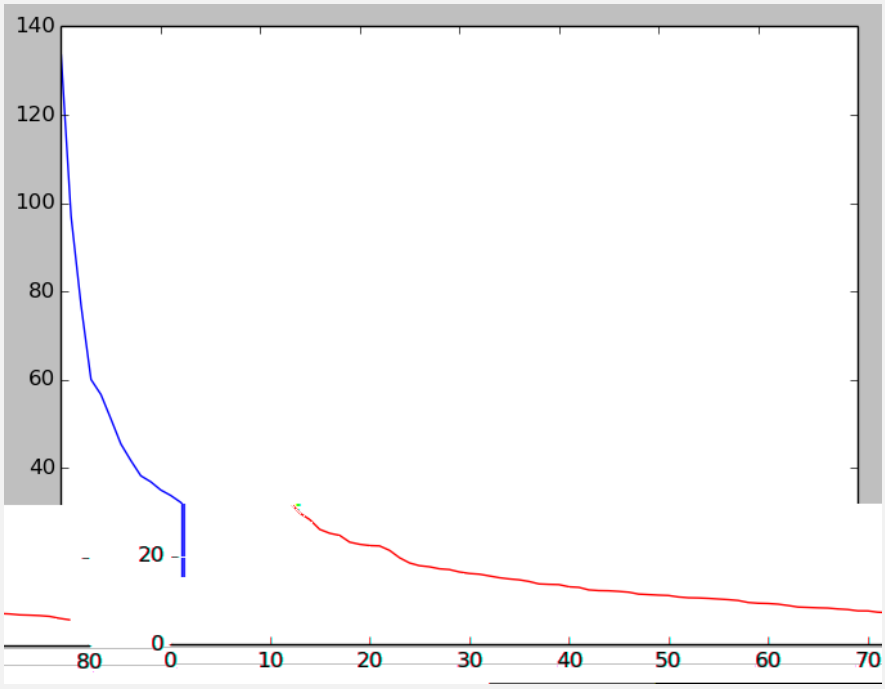
Gene oriented SVD without scaling

```
%time (u,s,v) = svd(X, full_matrices = False)
```

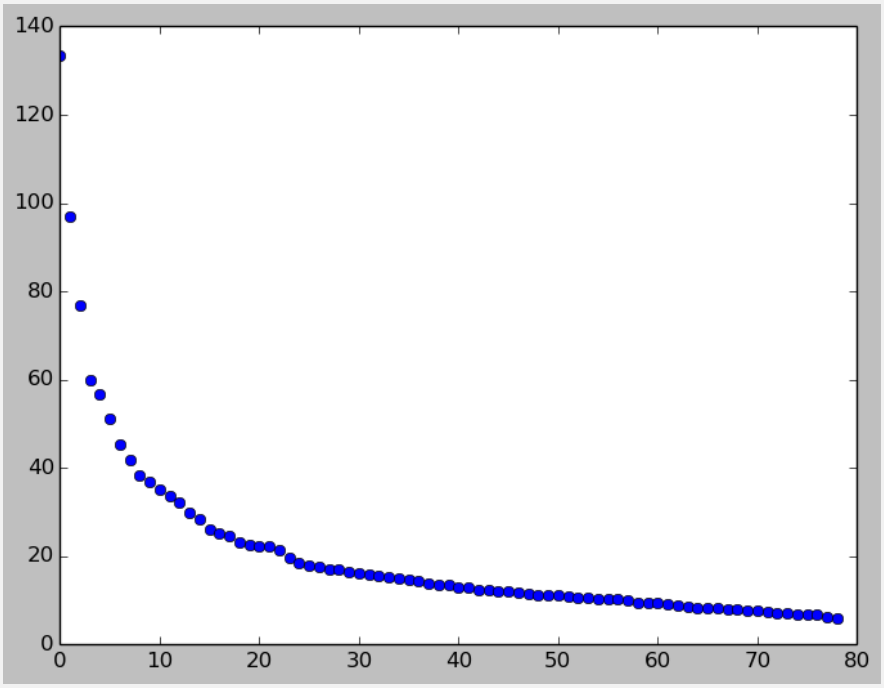
```
CPU times: user 44 ms, sys: 0 ns, total: 44 ms
Wall time: 44 ms
```

Plot singular values

```
fig = figure()
plot(s)
display(fig)
```



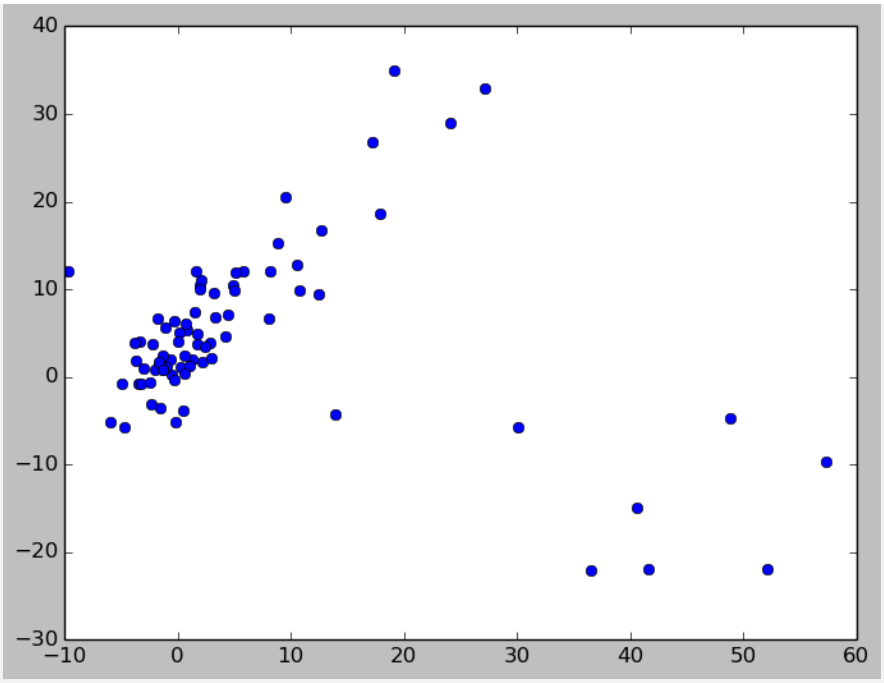

```
fig = figure()
plot(s,"bo")
display(fig)
```



Project arrays into gene space

```
p1 = dot(u[:,0],X)
p2 = dot(u[:,1],X)
```

```
fig = figure()
plot(p1,p2,"bo")
display(fig)
```



Lets color by data set

```
print eisen.expCond
```

```
['alpha 0', 'alpha 7', 'alpha 14', 'alpha 21', 'alpha 28', 'alpha 35', 'alpha 42', 'alpha 49', 'alpha 56
```

```
sorted(set(i.split()[0] for i in eisen.expCond))
```

```
['Elu', 'alpha', 'cdc15', 'cold', 'diau', 'dtt', 'heat', 'spo', 'spo-', 'spo5']
```

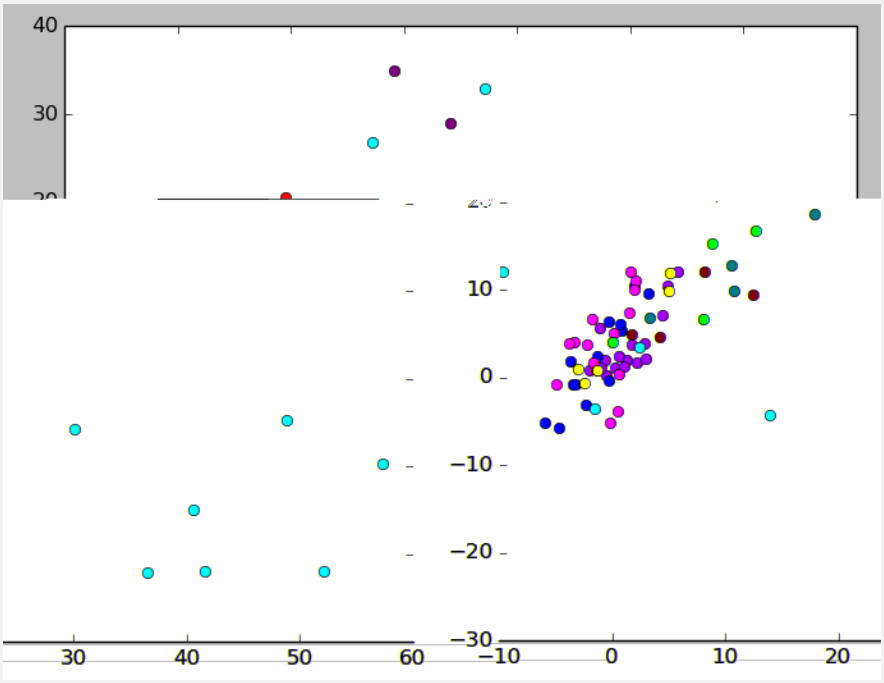
```
colors = []
for i in eisen.expCond:
    if(i.startswith("Elu")):
        colors.append("red")
    elif(i.startswith("alpha")):
        colors.append("orange")
    elif(i.startswith("cdc15")):
        colors.append("yellow")
    elif(i.startswith("cold")):
        colors.append("green")
    elif(i.startswith("diau")):
        colors.append("cyan")
    elif(i.startswith("dtt")):
        colors.append("blue")
    elif(i.startswith("heat")):
        colors.append("purple")
    elif(i.startswith("spo")):
        colors.append("magenta")
```

```
    else:
        raise ValueError
```

```
len(colors), len(p1)
```

```
(79, 79)
```

```
fig = figure()
for (i,j,c) in zip(p1,p2,colors):
    plot([i],[j], color = c, marker = "o")
display(fig)
```



So, the second singular vector is separating out a lot of the sporulation data – let's color and label this series

```
eisen.expCond.index("spo 0"), eisen.expCond.index("heat 0")
```

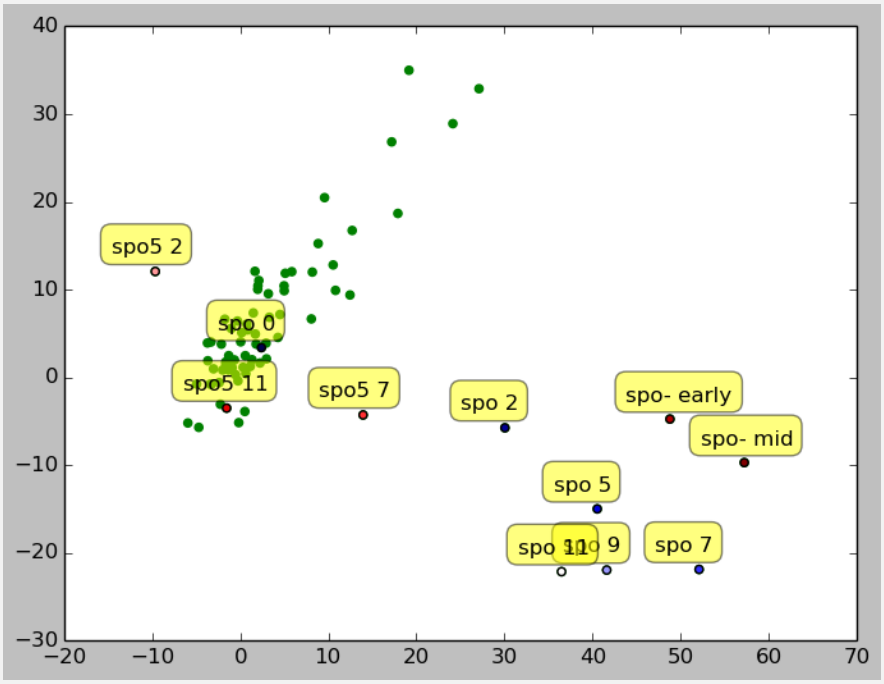
```
(47, 58)
```

```
58-47
```

```
11
```

```
(fig,ax) = subplots(1,1)
ax.scatter(p1,p2,color ="green", marker= "o")
ax.scatter(p1[47:58],p2[47:58],c = arange(47,58), cmap = get_cmap("seismic"))
for (x, y, label) in zip(p1[47:58], p2[47:58], eisen.expCond[47:58]):
```

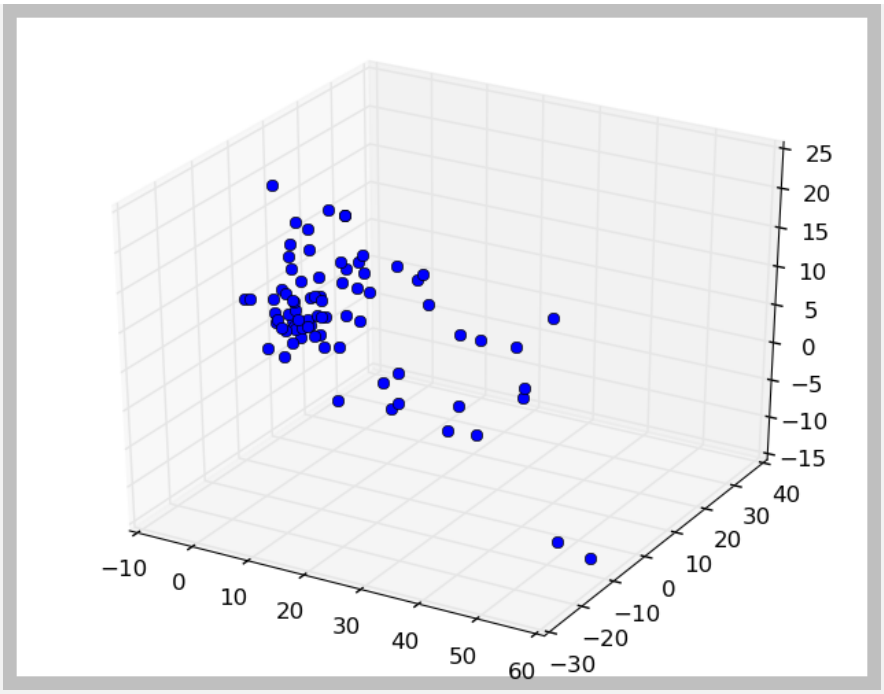
```
    ax.annotate(label, xy = (x, y), xytext = (x-5,y+2),
                bbox = {"fc":'yellow',"alpha":.5,"boxstyle":"round,pad=0.5"})
display(fig)
```



Based on the key, the "spo5" columns are relative to t=5h rather than t=0 (effectively subtracting the spo5 vector) and the "spo-" columns are from an *ndt80* knockout relative to t=2h or t=5h.

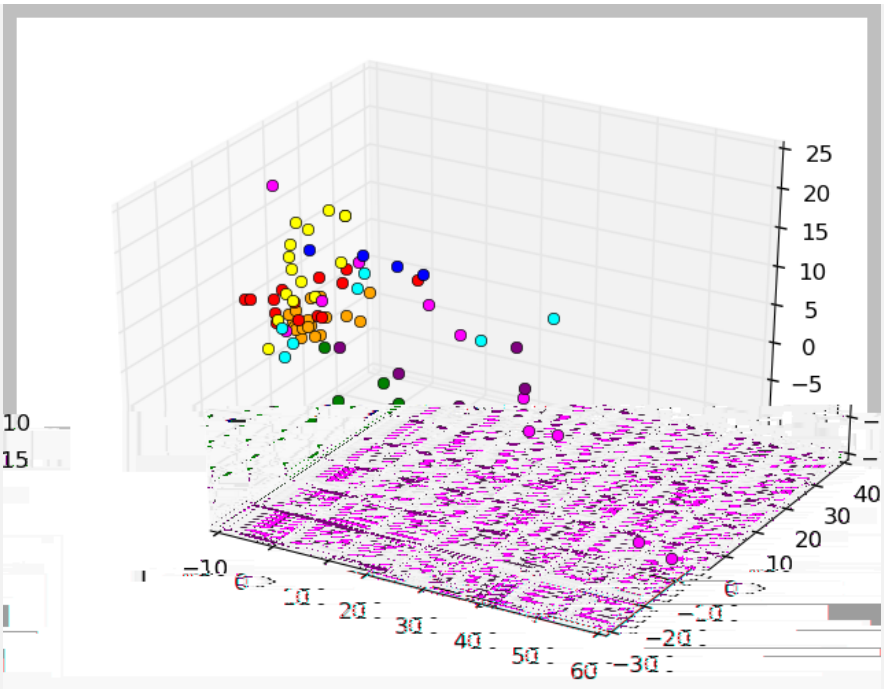
Here's the equivalent plot for the arrays projected onto the first three components (try rotating the plot in IPython)

```
p3 = dot(u[:,2],X)
from mpl_toolkits.mplot3d import Axes3D
fig = figure()
ax = fig.add_subplot(111, projection='3d')
ax.plot(p1,p2,p3,"bo")
display(fig)
```



Or, with our previous coloring:

```
fig = figure()
ax = fig.add_subplot(111, projection='3d')
for (i,j,k,c) in zip(p1,p2,p3,colors):
    ax.plot([i],[j],[k], color = c, marker = "o")
display(fig)
```



Another view, highlighting the separation of the heat-shock data:

```
display(fig)
```