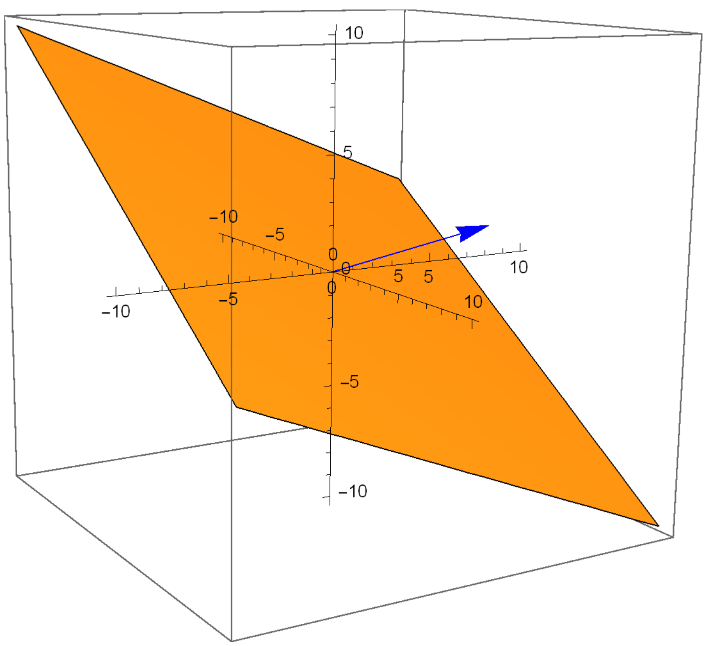
SVD

Example : Reflexion across plane

Let's reflect vector $v = \{3,\ 6,\ 7\}$ across plane $3\,x\ +\ 2\,y\ +\ 5\,z\ = 0$

In[248]:=
```
a = {0, 0, 0};
b = {3, 6, 2};
v =
  Graphics3D[{Blue, Arrow[{a, b}]}, Axes → True, Boxed -> True, AxesLabel → {x, y, z}];
plane = ContourPlot3D[3 x + 2 y + 5 z == 0, {x, -10, 10}, {y, -10, 10},
   {z, -10, 10}, AxesOrigin → {0, 0, 0}, Mesh → None];
Show[
 plane,
 v]
```

Out[252]=



Finding transformation that reflects vectors across plane

First, it is necessary to find the basis $\beta = \{w_1,\ w_2,\ w_3\}$ with all basis vectors perpendicular to each other.

The first one is $w_1\ = \{3,\ 2,\ 5\}$, the normal vector to the plane.

Normalizing it:

In[195]:=
```
w1 = Normalize[{3, 2, 5}]
```

$$\left\{\frac{3}{\sqrt{38}}, \sqrt{\frac{2}{19}}, \frac{5}{\sqrt{38}}\right\}$$

We can also take an arbitrary vector on the plane and it will be perpendicular to the normal vector. So, let $w_2 = \{1,\ 1,\ -1\}$ normalized:

In[198]:=
```
w2 = Normalize[{1, 1, -1}]
```

Out[198]= $\left\{\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right\}$

Finally, we can take the cross product between these two vector to make up our third basis vector, $w_3 = w_1 \times w_2$

In[201]:=
```
w3 = Cross[w1, w2]
```

Out[201]= $\left\{-\frac{7}{\sqrt{114}}, 4\sqrt{\frac{2}{57}}, \frac{1}{\sqrt{114}}\right\}$

So we have our basis β:

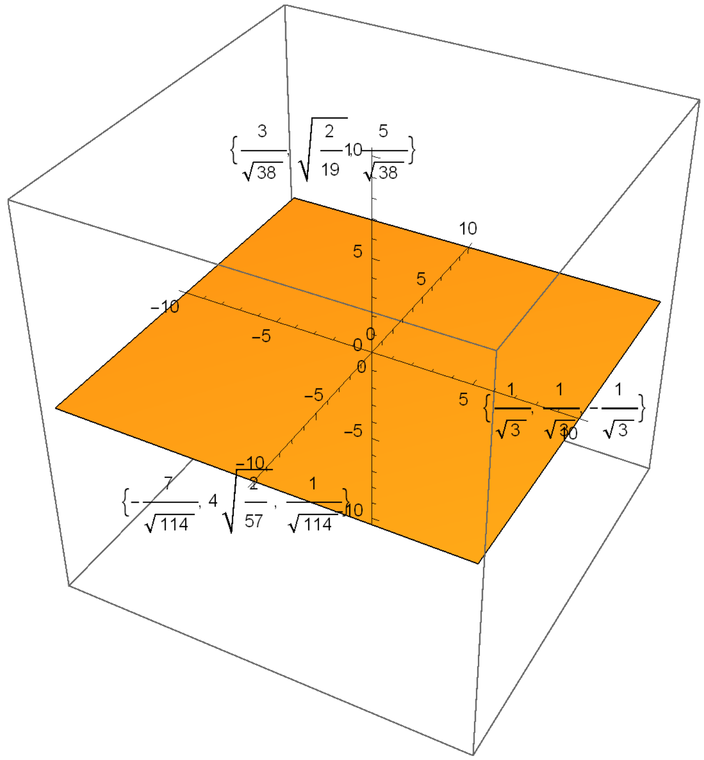
In[226]:=
```
β = {w1, w2, w3}
```

Out[226]= $\left\{\left\{\frac{3}{\sqrt{38}}, \sqrt{\frac{2}{19}}, \frac{5}{\sqrt{38}}\right\}, \left\{\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, -\frac{1}{\sqrt{3}}\right\}, \left\{-\frac{7}{\sqrt{114}}, 4\sqrt{\frac{2}{57}}, \frac{1}{\sqrt{114}}\right\}\right\}$

In this new basis the transformation becomes really easy -- all we need to do is flip the sign of the component of the vector that is normal to the plane, in this case w_1:

In[212]:=
```
new = ContourPlot3D[z == 0, {x, -10, 10}, {y, -10, 10},
   {z, -10, 10}, AxesOrigin → {0, 0, 0}, Mesh → None, AxesLabel → {w2, w3, w1}];
Show[
 new]
```

Out[213]=



Notice on the plot above that the new z axis is the vector normal to the plane, w_1. So, we can write a transformation matrix T in the basis β that inverts the w_1 component of a vector:

In[222]:=
```
T = {{-1, 0, 0}, {0, 1, 0}, {0, 0, 1}};
MatrixForm[T]
```

Out[223]//MatrixForm=

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

We have expressed the transformation that reflects a vector across the plane on the β basis, so now we need to find a way of changing basis from standard to β and vice-versa.
The matrix that takes a vector from β to the standard basis is just the three basis vectors:

In[224]:=
```
stToB = β;
MatrixForm[β]
```

Out[225]//MatrixForm=

$$\begin{pmatrix} \frac{3}{\sqrt{38}} & \sqrt{\frac{2}{19}} & \frac{5}{\sqrt{38}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \\ -\frac{7}{\sqrt{114}} & 4\sqrt{\frac{2}{57}} & \frac{1}{\sqrt{114}} \end{pmatrix}$$

So the last part is getting a vector in standard and converting it β. The matrix that does that is the inverse of the previous matrix.
Since β is orthogonal, its inverse equals its transpose:

In[228]:=
```
BtoSt = Transpose[β];
MatrixForm[BtoSt]
```

Out[229]//MatrixForm=

$$\begin{pmatrix} \frac{3}{\sqrt{38}} & \frac{1}{\sqrt{3}} & -\frac{7}{\sqrt{114}} \\ \sqrt{\frac{2}{19}} & \frac{1}{\sqrt{3}} & 4\sqrt{\frac{2}{57}} \\ \frac{5}{\sqrt{38}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{114}} \end{pmatrix}$$

Therefore, the matrix M that reflects a vector across the plane is:

In[253]:=
```
M = BtoSt .T.stToB;
MatrixForm[MatrixForm[stToB] .MatrixForm[T].MatrixForm[BtoSt]]
MatrixForm[M]
```

Out[254]//MatrixForm=

$$\begin{pmatrix} \frac{3}{\sqrt{38}} & \sqrt{\frac{2}{19}} & \frac{5}{\sqrt{38}} \\ \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \\ -\frac{7}{\sqrt{114}} & 4\sqrt{\frac{2}{57}} & \frac{1}{\sqrt{114}} \end{pmatrix} . \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} \frac{3}{\sqrt{38}} & \frac{1}{\sqrt{3}} & -\frac{7}{\sqrt{114}} \\ \sqrt{\frac{2}{19}} & \frac{1}{\sqrt{3}} & 4\sqrt{\frac{2}{57}} \\ \frac{5}{\sqrt{38}} & -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{114}} \end{pmatrix}$$

Out[255]//MatrixForm=

$$\begin{pmatrix} \frac{10}{19} & -\frac{6}{19} & -\frac{15}{19} \\ -\frac{6}{19} & \frac{15}{19} & -\frac{10}{19} \\ -\frac{15}{19} & -\frac{10}{19} & -\frac{6}{19} \end{pmatrix}$$

The reflected vector is:

In[256]:=
```
v1 = {3, 6, 2};
v2 = v1.M;
MatrixForm[v2]
```

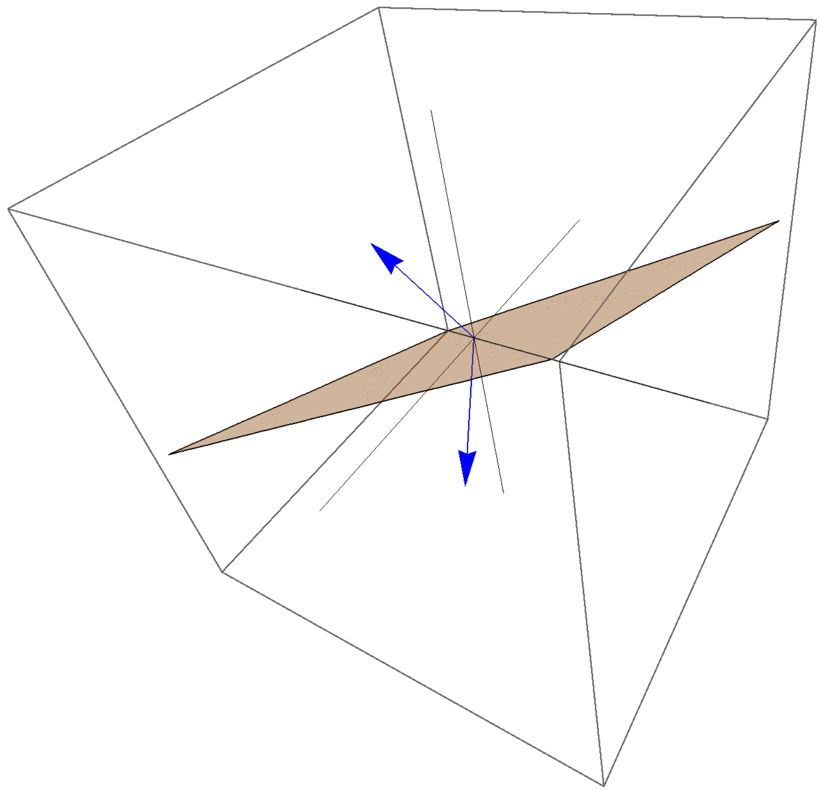
Out[258]//MatrixForm=

$$\begin{pmatrix} -\frac{36}{19} \\ \frac{52}{19} \\ -\frac{117}{19} \end{pmatrix}$$

We can verify that visually by plotting both vectors:

In[428]:=
```
o = {0, 0, 0};
vec1 =
  Graphics3D[{Blue, Arrow[{o, v1}]}, Axes → True, Boxed -> True, AxesLabel → {x, y, z}];
vec2 = Graphics3D[{Blue, Arrow[{o, v2}]}, Axes → True,
   Boxed -> True, AxesLabel → {x, y, z}];
plane = ContourPlot3D[3 x + 2 y + 5 z == 0, {x, -10, 10}, {y, -10, 10}, {z, -10, 10},
   AxesOrigin → {0, 0, 0}, Mesh → None, ContourStyle → Opacity[0.4], Ticks → None];
Show[plane, vec1, vec2, ViewPoint → {0.8, 1, -1}]
```

Out[432]=



Using SVD on an image

In[440]:=
```
image = ImageResize[Import["C:\\Users\\nicol\\Documents\\mathematica\\sydney.jpg"], 500]
```

Out[440]=

Converting image to grayscale:

In[441]:=
```
imageGray = ColorConvert[image, "Grayscale"]
```

Out[441]=

Getting the pixel values into array:

In[442]:=
```
pixels = ImageData[imageGray];
```

Now we use Singular Value Decomposition to break down the pixel matrix into the three matrices described in the previous example: U, Σ, V^T.

```
{u, s, v} = SingularValueDecomposition[pixels];
v = Transpose[v];
```

Dimensions of U, Σ and V^T:

In[453]:=
```
Dimensions[u]
Dimensions[s]
Dimensions[v]
```

Out[453]= {391, 391}

Out[454]= {391, 500}

Out[455]= {500, 500}

We can see that the diagonal matrix Σ has 391 rows, so we can choose any number between 1 and 391 to get the image back. Notice that choosing 391 gives the original image, while choosing lower values will give you an approximation.

In[456]:=
```
original = Image[u.s.v]
```

Out[456]=


Getting an approximation using 100/391 of the image:

In[485]:=
```
u1 = Take[u, {1, 391}, {1, 100}];
s1 = Take[s, {1, 100}, All];
approximation1 = Image[u1.s1.v]
```

Out[487]=

Getting an approximation using 10% of the image:

In[494]:=
```
u2 = Take[u, {1, 391}, {1, 39}];
s2 = Take[s, {1, 39}, All];
approximation2 = Image[u2.s2.v]
```

Out[496]=

Getting an approximation using 10/391 of the image :

In[497]:=
```
u3 = Take[u, {1, 391}, {1, 10}];
s3 = Take[s, {1, 10}, All];
approximation3 = Image[u3.s3.v]
```

Out[499]=