Matrices, Geometry&*Mathematica*

Authors: Bruce Carpenter, Bill Davis and Jerry Uhl ©2001

Producer: Bruce Carpenter

Publisher: Math Everywhere, Inc.

MGM.04 SVD Analysis of 2D Matrices

TUTORIALS

T.1) The issue of rank. When an SVD stretch factor is 0

□T.1.a.i) Only one positive stretch factor

Here is the matrix

$$A = \begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1.0 \end{pmatrix}$$

and an SVD analysis of its stretch factors:

$$A = \begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1.0 \end{pmatrix};$$

```
MatrixForm[A]
```

$$\begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1. \end{pmatrix}$$

```
stretches = SingularValues[A][[2]]
```

{1.92354}

1 <-----> hangerframe   2 <-----> stretches   3 <-----> alignerframe

Mathematica reports only one stretch factor.
How do you interpret this?

□Answer:

When *Mathematica* reports just one stretch factor, *Mathematica* is telling you that there are really two stretch factors but one of them is 0. *Mathematica* does not bother to give you the 0 stretch factor.

This signals that the given matrix A is not invertible.

Check:

```
Inverse[A]
```

Inverse::luc : Result for Inverse of badly conditioned matrix {{-0.7, 1.4}, {0.5, -1.}} may contain significant numerical errors.

$\{\{-3.46431\times10^{16}, -4.85003\times10^{16}\}, \{-1.73215\times10^{16}, -2.42502\times10^{16}\}\}$

This checks.

□T.1.a.ii) Only one hangerframe vector, only one aligner frame vector, only one stretch factor

Stay with the same matrix

$$A = \begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1.0 \end{pmatrix}$$

and look at the full SVD analysis of it:

$$A = \begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1.0 \end{pmatrix};$$

```
MatrixForm[A]
```

$$\begin{pmatrix} -0.7 & 1.4 \\ 0.5 & -1. \end{pmatrix}$$

```
alignerframe = SingularValues[A][[3]]
```

{{0.447214, -0.894427}}

```
stretches = SingularValues[A][[2]]
```

{1.92354}

```
hangerframe = SingularValues[A][[1]]
```

{{-0.813733, 0.581238}}

1 <-----> hangerframe   2 <-----> stretches   3 <-----> alignerframe

Mathematica reports
 only one alignerframe vector,
 only one stretch factor, and
 only one hangerframe vector.
What information do you get from this?

□Answer:

Because A is not invertible,you know that the matrix A hangs all its hits on a line through {0,0}.

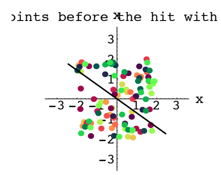
This is the line through {0,0} defined by the lone vector in the hangerframe.

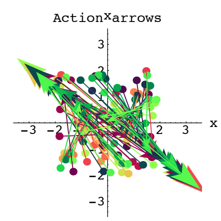
```
Clear[hangerframe];
hangerframe[1] = SingularValues[A][[1, 1]]
```

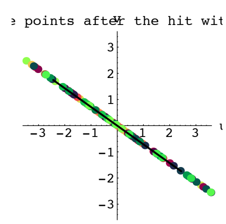
{-0.813733, 0.581238}

See it happen by taking some random points and hitting A on them.

```
a = 2;
points =
  Table[{Random[Real, {-a, a}], Random[Real, {-a, a}]}, {k, 1, 100}];
pointcolor[k_] = RGBColor[0.5 (Sin[(2 π k)/(Length[points])] + 1),
    0.5 (Cos[(2 π k)/(Length[points])] + 1), 0.3];
pointplot = Table[Graphics[{PointSize[0.04], pointcolor[k],
     Point[points[[k]]]}], {k, 1, Length[points]}];
hitpointplot = Table[Graphics[{PointSize[0.04], pointcolor[k],
     Point[A.points[[k]]]}], {k, 1, Length[points]}];
actionarrows = Table[Arrow[A.points[[k]] - points[[k]], Tail → points[[k]],
    VectorColor → pointcolor[k]], {k, 1, Length[points]}];
b = 3;
lineplot = Graphics[
   {Thickness[0.01], Line[{-b hangerframe[1], b hangerframe[1]}]}];
ranger = 1.8 a;
before =
  Show[pointplot, lineplot, Axes → True, AxesLabel → {"x", "x"},
   PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   PlotLabel → "Points before the hit with A"];
Show[before, actionarrows, PlotRange → {{-ranger, ranger},
    {-ranger, ranger}}, PlotLabel → "Action arrows"];
after = Show[hitpointplot, lineplot, Axes → True, AxesLabel →
    {"u", "v"}, PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   PlotLabel → "Same points after the hit with A"];
```







Grab and animate.

The plotted line is the line through {0,0} defined by the lone hangerframe vector:

```
hangerframe = SingularValues[A][[1]]
```

{{-0.813733, 0.581238}}

□T.1.a.iii) Rank

Stay with the same matrix as in parts i) and ii).
Some folks say that this the rank of this matrix is 1.
The same folks say that the rank of any invertible 2D matrix is 2.
What do they mean by this?

□Answer:

The rank of a matrix is the number of dimensions the matrix uses to hang its hits.

The matrix in parts i) and ii) hangs all its hits on a one dimensional straight line. That's why folks say the rank of this matrix is 1.

On the other hand, invertible 2D matrices use all of 2D to hang their hits. That's why folks say that the rank of an invertible 2D matrix is 2.

□T.1.a.iv) Hangers with just one column and aligners with just one row

Here's a new matrix

$$A = \begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix}:$$

and a look at its SVD stretch factors:

$$A = \begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix};$$

```
stretches = SingularValues[A][[2]]
```

{3.32415}

This matrix is rank 1.
Here are the SVD aligner, stretcher and hanger for A:

```
hanger = Transpose[SingularValues[A][[1]]];
MatrixForm[hanger]
```

$\begin{pmatrix} -0.83205 \\ 0.5547 \end{pmatrix}$

```
stretcher = DiagonalMatrix[SingularValues[A][[2]]];
MatrixForm[stretcher]
```

$\begin{pmatrix} 3.32415 \end{pmatrix}$

```
aligner = SingularValues[A][[3]];
MatrixForm[aligner]
```

$\begin{pmatrix} 0.21693 & -0.976187 \end{pmatrix}$

Check:

```
MatrixForm[hanger.stretcher.aligner]
MatrixForm[A]
```

$\begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix}$

$\begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix}$

What happened here?

Answer:

Mathematica took:

```
MatrixForm[hanger]
```

$\begin{pmatrix} -0.83205 \\ 0.5547 \end{pmatrix}$

This is not a "column" vector.
This is a matrix with two rows and one column.

```
MatrixForm[stretcher]
```

$\begin{pmatrix} 3.32415 \end{pmatrix}$

This is not a number.
This is a matrix with one row and one column.

```
MatrixForm[aligner]
```

$\begin{pmatrix} 0.21693 & -0.976187 \end{pmatrix}$

This is not a vector.
This is a matrix with one row and two columns.

and multiplied them out to duplicate A:

```
MatrixForm[hanger.stretcher.aligner]
MatrixForm[A]
```

$\begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix}$

$\begin{pmatrix} -0.6 & 2.7 \\ 0.4 & -1.8 \end{pmatrix}$

You can do this by hand if you like.

hanger.stretcher.aligner

$$= \begin{pmatrix} -0.83205 \\ 0.5547 \end{pmatrix} . (3.32415) . (0.21693 \quad -0.976187)$$

$$= \begin{pmatrix} -0.83205 \\ 0.5547 \end{pmatrix} . (3.32415 \times 0.21693,\ 3.32415 \times\ -0.976187)$$

$$= \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

with

a = (-0.83205) (3.32415) (0.21693)
b = (-0.83205) (3.32415) (−0.976187)
c = (0.5547) (3.32415) (0.21693)
d = (0.5547) (3.32415) (−0.976187).

T.2) Det[A .B] = Det[A] Det[B]

If A is a 2D diagonal matrix, then Det[A] = product of diagonal entries

Why $Det[A^{-1}] = \frac{1}{Det[A]}$

If A is a 2D hanger or aligner based on a right hand frame, then Det[A] = 1.

If A is a 2D hanger or aligner based on a left hand frame, then Det[A] = −1.

Why A and A^t have the same stretch factors and why $Det[A^t] = Det[A]$

T.2.a) Det[A .B] = Det[A] Det[B]

Look at these calculations based on the formula

$$Det[\begin{pmatrix} a & b \\ c & d \end{pmatrix}] = a\,d - b\,c$$

involving cleared matrices:

```
Clear[a, b, c, d, r, s, t, u, w, x, y, z];
A = (a b
     c d);
P = (r s
     t u);
T = (w x
     y z);
Expand[Det[A .P]] == Expand[Det[A] Det[P]]
True
Expand[Det[A .P.T]] == Expand[Det[A] Det[P] Det[T]]
True
```

What's the message?

Answer:

Look again:

```
Expand[Det[A .P]] == Expand[Det[A] Det[P]]
True
```

This tells you that the determinant of the product of two matrices is the product of their determinants.

Now look at this one:

```
Expand[Det[A .P.T]] == Expand[Det[A] Det[P] Det[T]]
True
```

This tells you that the determinant of the product of three matrices is the product of their determinants.

The same story holds up for any number of matrices.

T.2.b) If A is diagonal matrix , then Det[A] = product of diagonal entries

Here's a random diagonal matrix

```
Clear[diagonalentry];
diagonalentry[1] = Random[Real, {-2, 2}];
diagonalentry[2] = Random[Real, {-2, 2}];
diagonalmatrix = (diagonalentry[1]        0
                        0        diagonalentry[2]);
MatrixForm[diagonalmatrix]
```

$\begin{pmatrix} -0.0246945 & 0 \\ 0 & 0.772789 \end{pmatrix}$

Det[diagonalmatrix] is:

```
Det[diagonalmatrix]
-0.0190836
```

Compare:

```
diagonalentry[1] diagonalentry[2]
-0.0190836
```

Explain why the same thing happens for any and all diagonal matrices.

Answer:

The easiest way to see this is to use the formula

$$Det[\begin{pmatrix} a & b \\ c & d \end{pmatrix}] = a\,d - b\,c$$

from the Basics.

Applying this formula to

$$Det[diagonalmatrix] = Det[\begin{pmatrix} diagonalentry[1] & 0 \\ 0 & diagonalentry[2] \end{pmatrix}]$$

gives

Det[diagonalmatrix] = diagonalentry[1] diagonalentry[2] - 0 0 = diagonalentry[1] diagonalentry[2].

T.2.c) Why $Det[A^{-1}] = \frac{1}{Det[A]}$

Look at these calculations of $Det[A^{-1}]$ and $\frac{1}{Det[A]}$ for a random matrix A:

```
A = (Random[Real, {-3, 3}]  Random[Real, {-3, 3}]
     Random[Real, {-3, 3}]  Random[Real, {-3, 3}]);
Det[Inverse[A]]
-0.153845
```

```
1/Det[A]
```

-0.153845

Apparently, $\text{Det}[A^{-1}] = \frac{1}{\text{Det}[A]}$.

Explain why this is guaranteed for any and all invertible 2D matrices A.

Answer:

Identity = $A^{-1}.A$

So

$1 = \text{Det}[\text{Identity}] = \text{Det}[A^{-1}]\,\text{Det}[A]$.

And so

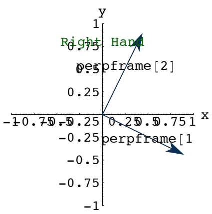
$\frac{1}{\text{Det}[A]} = \text{Det}[A^{-1}]$.

That's all there is to it.

T.2.d.i) If A is a hanger or aligner based on a right hand frame, then Det[A] = 1

Here's a random right hand perpendicular frame:

```
s = Random[Real, {0, 2 π}];
Clear[perpframe];
{perpframe[1], perpframe[2]} =
  {{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}};
rightframeplot = Show[Table[Arrow[perpframe[k], Tail → {0, 0},
     VectorColor → Indigo, HeadSize → 0.2], {k, 1, 2}],
   Graphics[Text["perpframe[1]", 0.6 perpframe[1]]],
   Graphics[Text["perpframe[2]", 0.6 perpframe[2]]],
   Graphics[{GosiaGreen, Text["Right Hand", {0, 0.8}]}], Axes → True,
   AxesLabel → {"x", "y"}, PlotRange → {{-1, 1}, {-1, 1}}];
```



The hanger matrix based on this perpendicular frame is

```
hanger = Transpose[{perpframe[1], perpframe[2]}];
MatrixForm[hanger]
```

$$\begin{pmatrix} 0.897359 & 0.441302 \\ -0.441302 & 0.897359 \end{pmatrix}$$

Its determinant is:

```
Det[hanger]
```

1.

The aligner matrix based on this perpendicular frame is

```
aligner = {perpframe[1], perpframe[2]};
MatrixForm[aligner]
```

$$\begin{pmatrix} 0.897359 & -0.441302 \\ 0.441302 & 0.897359 \end{pmatrix}$$

Its determinant is:

```
Det[aligner]
```

1.

Explain why the same thing happens for all aligners and hangers based on right hand perpendicular frames.

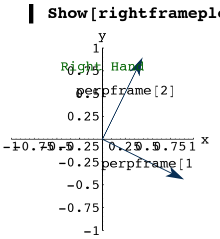
Answer:

Hits with hangers or aligners don't change any area measurements so both their stretch factors are equal to 1.

So |Det[aligner]| = |Det[hanger]| = 1.

To see why Det[hanger] = 1, look at

```
Show[rightframeplot];
```



The first column of hanger is perpframe[1] and the second column of hanger is perpframe[2].

Because {perpframe[1],perpframe[2]} is a right hand frame, the columns of hanger are positively oriented.

So Det[hanger] gets the **plus** sign.

In other words Det[hanger] = 1 .

To see why Det[aligner] = 1, remember

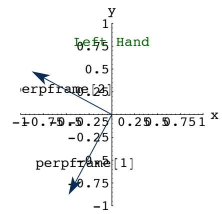
aligner = hanger^{-1}.

This gives $\text{Det}[\text{aligner}] = \frac{1}{\text{Det}[\text{hanger}]} = \frac{1}{1} = 1$.

T.2.d.ii) If A is a hanger or aligner based on a left hand frame, then Det[A] = −1

Here's a random left hand perpendicular frame:

```
s = Random[Real, {0, 2 π}];
Clear[perpframe];
{perpframe[1], perpframe[2]} =
  {{Cos[s], Sin[s]}, -{Cos[s + π/2], Sin[s + π/2]}};
leftframeplot = Show[Table[Arrow[perpframe[k], Tail → {0, 0},
     VectorColor → Indigo, HeadSize → 0.2], {k, 1, 2}],
   Graphics[Text["perpframe[1]", 0.6 perpframe[1]]],
   Graphics[Text["perpframe[2]", 0.6 perpframe[2]]],
   Graphics[{GosiaGreen, Text["Left Hand", {0, 0.8}]}], Axes → True,
   AxesLabel → {"x", "y"}, PlotRange → {{-1, 1}, {-1, 1}}];
```



The hanger matrix based on this perpendicular frame is

```
hanger = Transpose[{perpframe[1], perpframe[2]}];
MatrixForm[hanger]
```

$$\begin{pmatrix} -0.474597 & -0.880203 \\ -0.880203 & 0.474597 \end{pmatrix}$$

Its determinant is:

```
Det[hanger]
```

-1.

The aligner matrix based on this perpendicular frame is

```
aligner = {perpframe[1], perpframe[2]};
MatrixForm[aligner]
```

$$\begin{pmatrix} -0.474597 & -0.880203 \\ -0.880203 & 0.474597 \end{pmatrix}$$

Its determinant is:

```
Det[aligner]
```

-1.

Explain why the same thing happens for all aligners and hangers based on left hand perpendicular frames.

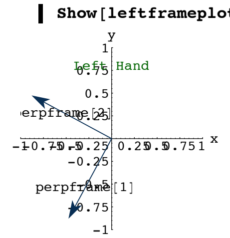
Answer:

Hits with hangers or aligners don't change any area measurements so both their stretch factors are equal to 1.

So |Det[aligner]| = |Det[hanger]| = 1.

To see why Det[hanger] = -1, look at

```
Show[leftframeplot];
```



The first column of hanger is perpframe[1] and the second column of hanger is perpframe[2].

Because {perpframe[1],perpframe[2]} is a left hand frame, the columns of hanger are **negatively** oriented.

So Det[hanger] gets the **minus** sign.

In other words Det[hanger] = -1 .

To see why Det[aligner] = -1, remember

aligner = hanger^{-1}.

This gives $\text{Det}[\text{aligner}] = \frac{1}{\text{Det}[\text{hanger}]} = \frac{1}{-1} = -1$.

◻T.2.e) Why A and A^t have the same stretch factors and why Det[A^t] = Det[A]

Look at these calculations of Det[A] and Det[A^t] for random 2D matrices A:

```
A = ( Random[Real, {-5, 5}]  Random[Real, {-5, 5}]
      Random[Real, {-5, 5}]  Random[Real, {-5, 5}] );
Det[A]
Det[Transpose[A]]
```

1.04853

1.04853

Rerun many times.

This is strong evidence that when you go with any 2D matrix A , then both A and A^t have the same determinant.
Explain why this is guaranteed.

◻Answer:

Go with any 2D matrix

A = hanger.stretcher.aligner.

This gives

A^t = aligner^t.stretcher.hanger^t.

This tells you why A and A^t have the same stretch factors

So

Det[A] = Det[hanger] Det[stretcher] Det[aligner]

and

Det[A^t] = Det[aligner^t] Det[stretcher] Det[hanger^t].

But Det[aligner^t] = Det[aligner] and Det[hanger] = Det[hanger^t]

Reasons: If the aligner frame is a right hand frame, then aligner^t is a hanger based on the same right hand frame.
So Det[aligner] = Det[aligner^t] = 1.
If the aligner frame is a left hand frame, then aligner^t is a hanger based on the same left hand frame.
So Det[aligner] = Det[aligner^t] = -1.
If the hanger frame is a right hand frame, then hanger^t is a aligner based on the same right hand frame.
So Det[hanger] = Det[hanger^t] = 1.
If the hanger frame is a left hand frame, then hanger^t is a aligner based on the same left hand frame.
So Det[hanger] = Det[hanger^t] = -1.

The upshot: Det[A] and Det[A^t] are both the product of the same three numbers.

This makes them equal.

T.3) Hits with 2D matrices with positive determinants preserve orientation. Hits with 2D matrices with negative determinants reverse orientation

◻T.3.a.i) If A is a 2D matrix with a positive determinant, then hits with A preserve orientation

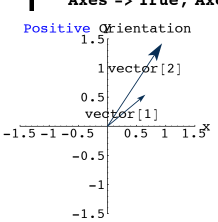
Here is a 3D matrix A with a positive determinant

```
A = ( 1.2  0.5
     -0.1  1.6 );
Det[A]
```

1.97

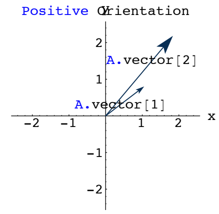
Here are random two vectors in 2D:

```
Clear[vector];
Clear[a];
a[i_] := ((-1)^Random[Integer,{0,1}]) Random[Real, {0.5, 1.5}];

vector[1] = {a[1], a[2]};
vector[2] = {a[3], a[4]};
ranger = 1.5;
before = Show[Table[
  Arrow[vector[k],
   Tail -> {0, 0}, VectorColor -> Indigo], {k, 1, 2}],
  Graphics[Text["vector[1]", 0.4 vector[1]]],
  Graphics[Text["vector[2]", 0.7 vector[2]]],
  PlotLabel -> If[ Det[{vector[1], vector[2]}] > 0,
    "Positive Orientation", If[Det[{vector[1], vector[2]}] < 0,
     "Negative Orientation", "No Orientation"]],
   PlotRange -> {{-ranger, ranger}, {-ranger, ranger}},
  Axes -> True, AxesLabel -> {"x", "y"}];
```



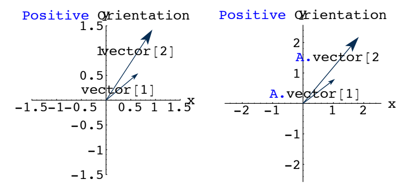
Now look at A.vector[1] and A.vector[2]:

```
newranger = Max[SingularValues[A][[2]]] ranger;
after = Show[Table[
  Arrow[A.vector[k],
   Tail -> {0, 0}, VectorColor -> Indigo], {k, 1, 2}],
  Graphics[Text["A.vector[1]", 0.4 A.vector[1]]],
  Graphics[Text["A.vector[2]", 0.7 A.vector[2]]],
  PlotLabel -> If[ Det[{A.vector[1], A.vector[2]}] > 0,
    "Positive Orientation", If[Det[{A.vector[1], A.vector[2]}] < 0,
     "Negative Orientation", "No Orientation"]],
   PlotRange -> {{-newranger, newranger}, {-newranger, newranger}},
  Axes -> True, AxesLabel -> {"x", "y"}];
```



See them both.

```
Show[GraphicsArray[{before, after}]];
```



Notice that {A.vector[1],A.vector[2]} have the same orientation as {vector[1],vector[2]}.

Explain this:
If A is a 2D matrix with a positive determinant ,then hits with A preserve orientation in the sense that:
- If {vector[1],vector[2]} are positively oriented, then so are {A.vector[1],A,vector[2]}.
- If {vector[1],vector[2]} are negatively oriented, then so are {A.vector[1],A,vector[2]}.

◻Answer:

Go with a matrix A with Det[A] > 0.

Make a new matrix

$$B = \begin{pmatrix} \text{vector[1]} & \text{vector[2]} \\ \downarrow & \downarrow \end{pmatrix}.$$

vector[j] is in column[j] of B

The sign of Det[B] determines the orientation (positive or negative) of {vector[1],vector[2]}.

$$\text{The product A.B } = \begin{pmatrix} \text{A.vector[1]} & \text{A.vector[2]} \\ \downarrow & \downarrow \end{pmatrix}.$$

The sign of Det[A.B] determines the orientation (positive or negative) of {A.vector[1],A.vector[2]}.

But

Det[A.B] = Det[A] Det[B].

And because Det[A] > 0, you are guaranteed that Det[A.B] has the sign of Det[B].

The upshot:
- If {vector[1],vector[2]} are positively oriented, then so are {A.vector[1],A,vector[2]}.
- If {vector[1],vector[2]} are negatively oriented, then so are {A.vector[1],A,vector[2]}.

And you're out of here.

◻T.3.a.ii) If A is a 2D matrix with a negative determinant, then hits with A reverse orientation

Here is a 3D matrix A with a negative determinant

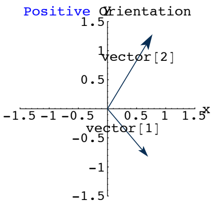
```
A = ( 0.4  1.8
      1.5  0.3 );
Det[A]
```

-2.58

Here are random two vectors in 2D:

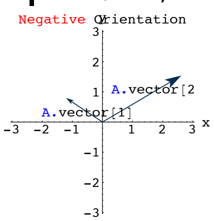
```
Clear[vector];
Clear[a];
a[i_] := ((-1)^Random[Integer,{0,1}]) Random[Real, {0.5, 1.5}];

vector[1] = {a[1], a[2]};
vector[2] = {a[3], a[4]};
ranger = 1.5;
before = Show[Table[
  Arrow[vector[k],
   Tail -> {0, 0}, VectorColor -> Indigo], {k, 1, 2}],
  Graphics[Text["vector[1]", 0.4 vector[1]]],
  Graphics[Text["vector[2]", 0.7 vector[2]]],
```

```
PlotLabel -> If[ Det[{vector[1], vector[2]}] > 0,
  "Positive Orientation", If[Det[{vector[1], vector[2]}] < 0,
   "Negative Orientation", "No Orientation"]],
 PlotRange -> {{-ranger, ranger}, {-ranger, ranger}},
Axes -> True, AxesLabel -> {"x", "y"}];
```



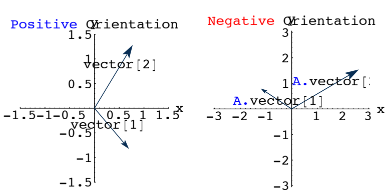
Now look at A.vector[1] and A.vector[2]:

```
newranger = Max[SingularValues[A][[2]]] ranger;
after = Show[Table[
  Arrow[A.vector[k],
    Tail -> {0, 0}, VectorColor -> Indigo], {k, 1, 2}],
  Graphics[Text["A.vector[1]", 0.4 A.vector[1]]],
  Graphics[Text["A.vector[2]", 0.7 A.vector[2]]],
  PlotLabel -> If[ Det[{A.vector[1], A.vector[2]}] > 0,
    "Positive Orientation", If[Det[{A.vector[1], A.vector[2]}] < 0,
     "Negative Orientation", "No Orientation"]],
   PlotRange -> {{-newranger, newranger}, {-newranger, newranger}},
 Axes -> True, AxesLabel -> {"x", "y"}];
```



See them both.

```
Show[GraphicsArray[{before, after}]];
```



Notice that the orientation of {A.vector[1],A.vector[2]} is opposite to the orientation of {vector[1],vector[2]}.

Explain this:
If A is a 2D matrix with a negative determinant ,then hits with A reverse orientation in the sense that:

- If {vector[1],vector[2]} are positively oriented, then {A.vector[1],A,vector[2]} are negatively oriented.
- If {vector[1],vector[2]} are negatively oriented, then {A.vector[1],A,vector[2]} are positively oriented.

□ Answer:

This is a copy,paste and edit of the answer to part i) above.
The edits are highlighted in green.

Go with a matrix A with Det[A] < 0.

Make a new matrix

$$B = \begin{pmatrix} \text{vector[1]} & \text{vector[2]} \\ \downarrow & \downarrow \end{pmatrix}.$$

vector[j] is in column[j] of B

The sign of Det[B] determines the orientation (positive or negative) of {vector[1],vector[2]}.

$$\text{The product A.B} = \begin{pmatrix} \text{A.vector[1]} & \text{A.vector[2]} \\ \downarrow & \downarrow \end{pmatrix}.$$

The sign of Det[A.B] determines the orientation (positive or negative) of {A.vector[1],A.vector[2]}.

But

Det[A.B] = Det[A] Det[B].

And because Det[A] < 0, you are guaranteed that Det[A.B] has the sign opposite to the sign of Det[B].

The upshot:

- If {vector[1],vector[2]} are positively oriented, then {A.vector[1],A,vector[2]} are negatively oriented.
- If {vector[1],vector[2]} are negatively oriented, then {A.vector[1],A,vector[2]} are positively oriented.

And you're really out of here.

T.4) Revealing the cross product formula:

If X = {a, b, c} and Y = {r, s, t}, then

$$X \times Y = \{\text{Det}[\begin{pmatrix} b & c \\ s & t \end{pmatrix}], -\text{Det}[\begin{pmatrix} a & c \\ r & t \end{pmatrix}], \text{Det}[\begin{pmatrix} a & b \\ r & s \end{pmatrix}]\}.$$

□ T.4.a.i) The formula for the cross product of two 3D vectors

Look at this:

```
Clear[a, b, c, r, s, t];
X = {a, b, c};
Y = {r, s, t};
{Det[(b c
      s t)], -Det[(a c
                   r t)], Det[(a b
                               r s)]}
```

```
{-c s + b t, c r - a t, -b r + a s}
```

Compare with *Mathematica*'s calculation of X × Y :

```
Cross[X, Y]
```

```
{-c s + b t, c r - a t, -b r + a s}
```

What's the message?

□ Answer:

The message is:

If X = {a, b, c} and Y = {r, s, t}, then

$$X \times Y = \{\text{Det}[\begin{pmatrix} b & c \\ s & t \end{pmatrix}], -\text{Det}[\begin{pmatrix} a & c \\ r & t \end{pmatrix}], \text{Det}[\begin{pmatrix} a & b \\ r & s \end{pmatrix}]\}:$$

```
Clear[a, b, c, r, s, t];
X = {a, b, c};
Y = {r, s, t};
Cross[X, Y] == {Det[(b c
                     s t)], -Det[(a c
                                  r t)], Det[(a b
                                              r s)]}
```

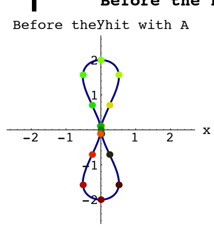
```
True
```

T.5) Using SVD matrix analysis to set the plot range

□ T.5.a.i) Using stretch factors to help set the plot range

Here's a curve:

```
Clear[x, y, t];
{x[t_], y[t_]} = 2 {Cos[t] Sin[t]^5, Sin[t]^3};

{tlow, thigh} = {0, 2 π};
ranger = 2.7;

Clear[hitplotter, hitpointplotter,
  pointcolor, actionarrows, matrix2D];
pointcolor[t_] = RGBColor[0.5 (Cos[t] + 1), 0.5 (Sin[t] + 1), 0];
jump = (thigh - tlow)/16;

hitplotter[matrix2D_] := ParametricPlot[matrix2D.{x[t], y[t]},
   {t, tlow, thigh}, PlotStyle → {Thickness[0.01], NavyBlue}},
   PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   AxesLabel → {"x", "y"}, DisplayFunction → Identity];

hitpointplotter[matrix2D_] :=
  Table[Graphics[{pointcolor[t], PointSize[0.035],
     Point[matrix2D.{x[t], y[t]}]}], {t, tlow, thigh - jump, jump}];

before = Show[hitplotter[IdentityMatrix[2]],
   hitpointplotter[IdentityMatrix[2]], PlotLabel →
    "Before the hit with A", DisplayFunction → $DisplayFunction];
```

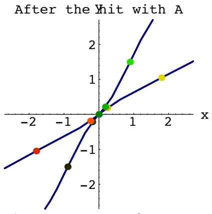


Here's a 2D matrix A and an attempt to plot what a hit with A does to this curve:

```
A = (1.8   1.9
     -0.9  1.8);
MatrixForm[A]
```

```
after = Show[hitplotter[A],
   hitpointplotter[A], PlotLabel → "After the hit with A",
   DisplayFunction → $DisplayFunction];
```

$$\begin{pmatrix} 1.8 & 1.9 \\ -0.9 & 1.8 \end{pmatrix}$$



What went wrong?

Answer:

Look at:

```
ranger
```

2.7

The plot range specification through

PlotRange->{{-ranger,ranger},{-ranger,ranger}}

did not accommodate the full curve in the second plot.

T.5.a.ii) Automatic setting of the PlotRange

Is there a way to set ranger in advance to avoid guessing?

Answer:

Of course, if there were no way this question would not have been asked.

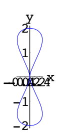
Here's how it goes:

First plot the original curve in true scale using

AspectRatio-> Automatic

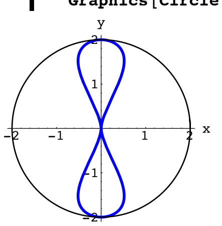
and PlotRange->All:

```
originalcurve = ParametricPlot[{x[t], y[t]},
   {t, tlow, thigh}, PlotStyle → {{Thickness[0.015], Blue}},
   AspectRatio → Automatic, PlotRange → All, AxesLabel → {"x", "y"}];
```



Now eyeball a circle centered on {0,0} that snugly surrounds the curve:

```
radius = 2.0;
Show[originalcurve,
  Graphics[Circle[{0, 0}, radius]], PlotRange → All];
```



Look at the stretch factors:

```
{xstretch, ystretch} = SingularValues[A][[2]]
```

{2.78035, 1.78035}

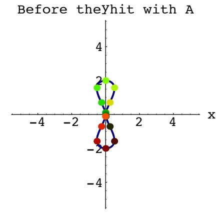
Redefine ranger:

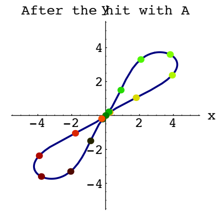
```
ranger = Max[{1.0, Max[{xstretch, ystretch}]}] radius
```

5.5607

Now run the plots:

```
before = Show[hitplotter[IdentityMatrix[2]],
   hitpointplotter[IdentityMatrix[2]], PlotLabel →
    "Before the hit with A", DisplayFunction → $DisplayFunction];

after = Show[hitplotter[A],
   hitpointplotter[A], PlotLabel → "After the hit with A",
   DisplayFunction → $DisplayFunction];
```





Now both plots are accommodated automatically and the same scale is used in both.

T.5.a.iii) The math behind the scene

What guarantees that setting

ranger = radius Max[{xstretch,ystretch,1}]

will always work?

Answer:

Once you identify the radius = r of a circle centered on {0,0} completely enclosing the curve, then you are sure that the hit with A stretches this circle and everything inside it by no more than a factor of

Max[{xstretch,ystretch}].

So the plot of the hit curve must come out inside the circle of radius

r Max[{xstretch,ystretch,1}].

T.6) Hand Calculations:

Mathematica Fat's tips on memorizing the formulas for the determinant and cross product and templates for hand calculation of solutions of linear systems (Cramer's rule), inverse matrices and SVD

HC.a) *Mathematica* Fat's tips on memorizing the formula for the 2D determinant

The formula for the determinant is:

$$\mathbf{Det}\left[\begin{pmatrix} a & b \\ c & d \end{pmatrix}\right] = a\,d - b\,c$$

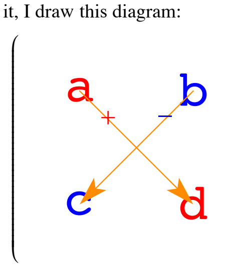
What are *Mathematica* Fats's tips for memorizing it?

Answer:

Fats says the formula is

$$\mathbf{Det}\left[\begin{pmatrix} a & b \\ c & d \end{pmatrix}\right] = a\,d - b\,c$$

To memorize it, I draw this diagram:



The way I think about it:

It's just opposite from the way you want it to be.

The arrow with the **negative slope** carries a **plus** sign.

The arrow with the **positive slope** carries a **minus** sign.

HC.b) *Mathematica* Fat's tips on memorizing the formula for the cross product

The formula for the cross product is:

If **X** = {**a**, **b**, **c**} and **Y** = {**r**, **s**, **t**}, then

$$\mathbf{X} \times \mathbf{Y} = \{\mathbf{Det}\left[\begin{pmatrix} b & c \\ s & t \end{pmatrix}\right], -\mathbf{Det}\left[\begin{pmatrix} a & c \\ r & t \end{pmatrix}\right], \mathbf{Det}\left[\begin{pmatrix} a & b \\ r & s \end{pmatrix}\right]\}.$$

What are *Mathematica* Fats's tips for memorizing it?

Answer:

Fats says:

Start with

X = {**a**, **b**, **c**} and **Y** = {**r**, **s**, **t**},

and make this box;

a	b	c
r	s	t

To get the first slot of **X** × **Y**, scratch out the first vertical column to get this box:

■	b	c
■	s	t

This gives you a 2D matrix.

The first slot of **X** × **Y** is $\text{Det}[\begin{pmatrix} b & c \\ s & t \end{pmatrix}]$.

To get the **second** slot of slot of **X** × **Y**, go back to

X = {**a**, **b**, **c**} and **Y** = {**r**, **s**, **t**},

and then go back to this box:

a	b	c
r	s	t

Scratch out the **second** vertical column to get this box

a	■	c
r	■	t

This gives you another 2D matrix.

The **second** slot of **X** × **Y** is **-** $\text{Det}[\begin{pmatrix} a & c \\ r & t \end{pmatrix}]$.

`Note the minus sign on the second slot.`

To get the **third** slot of slot of **X** × **Y**,

go back to

X = {**a**, **b**, **c**} and **Y** = {**r**, **s**, **t**},

and then go back to this box

a	b	c
r	s	t

Scratch out the **third** vertical column to get this box

a	b	■
r	s	■

This gives you another 2D matrix.

The **third** slot of **X** × **Y** is $\text{Det}[\begin{pmatrix} a & b \\ r & s \end{pmatrix}]$.

There you go:

If **X** = {**a**, **b**, **c**} and **Y** = {**r**, **s**, **t**}, then

$$X \times Y = \{\text{Det}[\begin{pmatrix} b & c \\ s & t \end{pmatrix}], -\text{Det}[\begin{pmatrix} a & c \\ r & t \end{pmatrix}], \text{Det}[\begin{pmatrix} a & b \\ r & s \end{pmatrix}]\}:$$

□ HC.d) Cramer's rule formulas for solutions of linear systems

In your grandfather's day, Cramer's rule was a hot topic. If you want to talk with your grandfather about matrices, you might want to know about Cramer's rule.
Here it is:

If $\text{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]$ is not 0, then the solutions {x,y} of

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}.\{x,y\} = \{u,v\}$$

are given by

$$x = \frac{\text{Det}[\begin{pmatrix} u & b \\ v & d \end{pmatrix}]}{\text{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]} \quad \text{and} \quad y = \frac{\text{Det}[\begin{pmatrix} a & u \\ c & v \end{pmatrix}]}{\text{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]}$$

Check:

```
Clear[a, b, c, d, u, v];
linsystem = (a b
             c d).{x, y} == {u, v}
```

{a x + b y, c x + d y} == {u, v}

```
x = Det[(u b
         v d)]
    ----------
    Det[(a b
         c d)]
```

$\frac{d u - b v}{-b c + a d}$

```
y = Det[(a u
         c v)]
    ----------
    Det[(a b
         c d)]
```

$\frac{-c u + a v}{-b c + a d}$

```
Simplify[(a b
          c d).{x, y}] == {u, v}
```

True

It works any time the determinant of the coefficient matrix is not 0.
Explain where these formulas come from.

□ Answer:

Go with

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}.\{x,y\} = \{u,v\}$$

```
Clear[a, b, c, d, x, y, u, v];
(a b
 c d).{x, y} == {u, v}
```

{a x + b y, c x + d y} == {u, v}

Put col[1] = {a,c} and col[2] = {b,d} and note that

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}.\{x,y\} = \{u,v\}$$

is the same as the vector equation:

x col[1] + y col[2] = {u,v}:

```
Clear[col];
col[1] = {a, c};
col[2] = {b, d};
 x col[1] + y col[2] == {u, v}
```

{a x + b y, c x + d y} == {u, v}

Make vectors perpendicular to each column:

perpcol[1] = {c,-a}

perpcol[2] = {d,-b}

Now remember

x col[1] + y col[2] = {u,v}

Dot perpcol[2] on both sides to get

perpcol[2]. (x col[1] + y] col[2]) = perpcol[2].{u,v}

This is the same as

x perpcol[2]. col[1] +y perpcol[2].col[2] = perpcol[2].{u,v}

Because perpcol[2] is perpendicular to col[2], this reduces to

x perpcol[2]. col[1] + 0 = perpcol[2].{u,v}

So the solution for x is:

$$x = \frac{\text{perpcol[2].\{u,v\}}}{\text{perpcol[2].col[1]}}$$

Similarly the solution for y is

$$y = \frac{\text{perpcol[1].\{u,v\}}}{\text{perpcol[1].col[2]}}.$$

Try it running on cleared constants:

```
Clear[a, b, c, d, x, u, v];
(a b
 c d).{x[1], x[2]} == {u, v}
```

{a x[1] + b x[2], c x[1] + d x[2]} == {u, v}

```
Clear[col];
col[1] = {a, c};
col[2] = {b, d};
 perpcol[1] = {c, -a};
 perpcol[2] = {d, -b};
x = perpcol[2].{u, v}
    ------------------
    perpcol[2].col[1]
```

$\frac{d u - b v}{-b c + a d}$

This is the same as:

```
Cramerx = Det[(u b
               v d)]
          ----------
          Det[(a b
               c d)]
```

$\frac{d u - b v}{-b c + a d}$

Similarly the solution for y is:

```
y = perpcol[1].{u, v}
    ------------------
    perpcol[1].col[2]
```

$\frac{c u - a v}{b c - a d}$

This is the same as:

```
Cramery = Det[( a u ; c v )] / Det[( a b ; c d )]
```

$\frac{-c u + a v}{-b c + a d}$

HC.e) Hand calculation of inverse matrices

Explain this: If $\mathrm{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]$ is not zero, then

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{\mathrm{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Answer:

According to Cramer's rule:

The solutions {x,y} of

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}.\{x,y\} = \{u,v\}$$

are given by

```
Clear[a, b, c, d, u, v]
{x, y} = { Det[( u b ; v d )] / Det[( a b ; c d )] , Det[( a u ; c v )] / Det[( a b ; c d )] }
```

$\left\{\frac{d u - b v}{-b c + a d}, \frac{-c u + a v}{-b c + a d}\right\}$

The matrix you hit on {u,v} to get {x,y} is the inverse of $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$.

From the output above, you can see that that matrix is:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{a d - b c}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

$$= \frac{1}{\mathrm{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

because $a d - b c = \mathrm{Det}[\begin{pmatrix} a & b \\ c & d \end{pmatrix}]$.

HC.f) Hand calculation of SVD: Tedious but possible

Thanks go to Professor Todd Will of Davidson College for suggesting this.

Here is a 2D matrix:

```
A = ( 1.8  -2.1 ; 1.3  0.2 );
MatrixForm[A]
```

$\begin{pmatrix} 1.8 & -2.1 \\ 1.3 & 0.2 \end{pmatrix}$

Use hand-style calculations to duplicate this matrix by coming up by with an aligner frame, stretch factors and a hanger frame so that

A = hanger.stretcher.aligner

Answer:

First notice that no matter what x is,the vectors

{x, 1 - x} and {x - 1 ,x}

are perpendicular:

```
Clear[x];
Expand[{x, 1 - x}.{x - 1, x}]
```

0

Now find a solution x of

(A.{x, 1 − x}).(A.{x − 1, x}) = 0:

```
SVDeqn = Expand[(A.{x, 1 - x}).(A.{x - 1, x})] == 0
```

$3.52 - 7.52\, x + 0.48\, x^2 == 0$

This is an ordinary quadratic equation which is readily solved by the quadratic formula:

```
xsols = Solve[SVDeqn]
```

{{x → 0.482974}, {x → 15.1837}}

Go with one of the solutions:

The smaller one will be the most easily managed.

```
goodx = x /. xsols[[1]]
```

0.482974

Use this x and put

$$\mathrm{alignerframe}[1] = \frac{\{goodx, 1-goodx\}}{(\sqrt{\{goodx, 1-goodx\}.\{goodx, 1-goodx\}})}$$

and

$$\mathrm{alignerframe}[2] = \frac{\{goodx-1, goodx\}}{(\sqrt{\{goodx-1, goodx\}.\{goodx-1, goodx\}})}:$$

```
Clear[alignerframe];
alignerframe[1] =
  {goodx, 1 - goodx} / Sqrt[{goodx, 1 - goodx}.{goodx, 1 - goodx}];
alignerframe[2] = {goodx - 1, goodx} /
  Sqrt[{goodx - 1, goodx}.{goodx - 1, goodx}];
aligner = {alignerframe[1], alignerframe[2]};
MatrixForm[aligner]
```

$\begin{pmatrix} 0.682633 & 0.730761 \\ -0.730761 & 0.682633 \end{pmatrix}$

The stretch factors are:

xstretch = || A.alignerframe[1] ||

$= \sqrt{(A.alignerframe[1]).(A.alignerframe[1])}$

and

ystretch = || A.alignerframe[2] ||

$= \sqrt{(A.alignerframe[2]).(A.alignerframe[2])}$

so your stretcher is:

```
xstretch = √((A.alignerframe[1]).(A.alignerframe[1]));
ystretch = √((A.alignerframe[2]).(A.alignerframe[2]));
stretcher = ( xstretch  0 ; 0  ystretch );
MatrixForm[stretcher]
```

$\begin{pmatrix} 1.07788 & 0 \\ 0 & 2.86674 \end{pmatrix}$

Your hangerframe is:

$\mathrm{hangerframe}[1] = \frac{1}{xstretch}$ A.alignerframe[1]

$\mathrm{hangerframe}[2] = \frac{1}{ystretch}$ A.alignerframe[2]

This makes it automatic that
A.alignerframe[1] = xstretch hangerframe[1]
and
A.alignerframe[2] = ystretch hangerframe[2]

So your hanger is:

```
Clear[hangerframe];
{hangerframe[1], hangerframe[2]} =
  { 1/xstretch A.alignerframe[1], 1/ystretch A.alignerframe[2]};
hanger = Transpose[{hangerframe[1], hangerframe[2]}];
MatrixForm[hanger]
```

$\begin{pmatrix} -0.283759 & -0.958896 \\ 0.958896 & -0.283759 \end{pmatrix}$

Try them out by comparing A with:

```
MatrixForm[hanger.stretcher.aligner]
MatrixForm[A]
```

$\begin{pmatrix} 1.8 & -2.1 \\ 1.3 & 0.2 \end{pmatrix}$

$\begin{pmatrix} 1.8 & -2.1 \\ 1.3 & 0.2 \end{pmatrix}$

It's within the reach of low-level but tedious hand algebra.