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
GIVE IT A TRY!

G.1) Inverse fundamentals*

G.1.a) Using SVD to build the inverse matrix

Here's a 2D matrix:

```
A = (3.2  -4.5
     2.7   0.8);
MatrixForm[A]
```

$\begin{pmatrix} 3.2 & -4.5 \\ 2.7 & 0.8 \end{pmatrix}$

Do SVD analysis of A by writing A in the form
A = hanger. stretcher. aligner:

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

$\begin{pmatrix} -0.97801 & -0.208556 \\ -0.208556 & 0.97801 \end{pmatrix}$

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

$\begin{pmatrix} 5.61825 & 0 \\ 0 & 2.61825 \end{pmatrix}$

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

$\begin{pmatrix} -0.657275 & 0.753651 \\ 0.753651 & 0.657275 \end{pmatrix}$

Check:

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

$\begin{pmatrix} 3.2 & -4.5 \\ 2.7 & 0.8 \end{pmatrix}$

$\begin{pmatrix} 3.2 & -4.5 \\ 2.7 & 0.8 \end{pmatrix}$

Here's *Mathematica*'s calculation of the inverse A^{-1} of A:

```
MatrixForm[Inverse[A]]
```

$\begin{pmatrix} 0.0543848 & 0.305914 \\ -0.183549 & 0.217539 \end{pmatrix}$

Use what you see above to give your own calculation of A^{-1}.

G.1.b) Using SVD to recognize a non-invertible matrix

Here's a 2D matrix:

```
A = (3.3  -4.5
     1.1  -1.5);
MatrixForm[A]
```

$\begin{pmatrix} 3.3 & -4.5 \\ 1.1 & -1.5 \end{pmatrix}$

Here's *Mathematica*'s attempt at a calculation of the inverse A^{-1} of A:

```
MatrixForm[Inverse[A]]
```

```
Inverse::luc : Result for Inverse of badly conditioned matrix
  {{3.3, -4.5}, {1.1, -1.5}} may contain significant numerical errors.
```

$\begin{pmatrix} -2.72945\times10^{15} & 8.18836\times10^{15} \\ -2.0016\times10^{15} & 6.0048\times10^{15} \end{pmatrix}$

Garbage.
Do an SVD analysis of A and use the result to explain why *Mathematica* balked at calculating the inverse of this matrix.

G.1.c) A given 2D matrix A is invertible if Det[A] ≠ 0

A given 2D matrix A is not invertible if Det[A] = 0

Lots of folks like to say that a given 2D matrix A is:
-> invertible if Det[A] ≠ 0
-> not invertible if Det[A] = 0.
Explain why they are right.

G.1.d) Looking at the stretch factors

Here is a 2D matrix A:

```
A = ( 1.6  -2.8
     -0.8   1.4);
MatrixForm[A]
```

$\begin{pmatrix} 1.6 & -2.8 \\ -0.8 & 1.4 \end{pmatrix}$

Intent on determining whether A is invertible you begin your SVD analysis:

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

$\begin{pmatrix} 3.60555 \end{pmatrix}$

At this point, you look at the stretch factors and announce that A is not invertible.
How did you know?

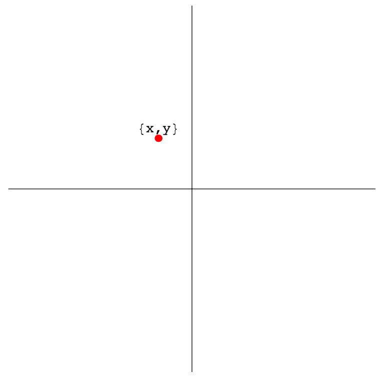
G.1.e) If both stretch factors of A are positive, can there be an {x,y} with {x,y} ≠ {0,0} and with A.{x,y} = {0,0}?

You are given a 2D matrix A and after you do your SVD analysis of it, you learn that both xstretch and ystretch are positive.
You make the call:
Can there be an {x,y} with
{x,y} ≠ {0,0}
and with
A.{x,y} = {0,0}?
Explain your response.

Click on the right for a heavy tip.

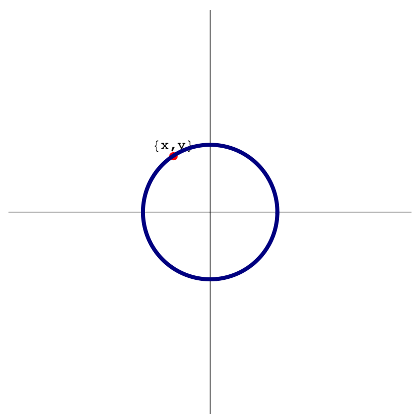
Take out a piece of paper and draw a point {x,y} ≠ {0,0}

It will look something like this:



Draw a circle centered at the origin running through {x,y}.

It will look something like this:



Now draw what you think happens when you hit this circle with a matrix with two positive stretch factors.

Can the resulting ellipse go through {0,0}?

G.1.e.ii) If Det[A] ≠ 0, can there be an {x,y} with {x,y} ≠ {0,0} and with A.{x,y} = {0,0}?

You are given a 2D matrix A and you let *Mathematica* calculate the determinant Det[A] of A and find that
Det[A] ≠ 0.
You make the call:
Can there be an {x,y} with
{x,y} ≠ {0,0}
and with
A.{x,y} = {0,0}?
Explain your response.

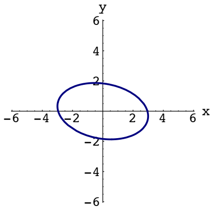
G.2) Area measurements and related matters*

G.2.a.i) The roles of the hanger frame, the stretch factors and the aligner frame

Here's the ellipse you get when you hit the unit circle with the 2D matrix

$A = \begin{pmatrix} 2.7 & 1.3 \\ 0.5 & -1.8 \end{pmatrix}$:

```
A = (2.7   1.3
     0.5  -1.8);
ranger = 2 Max[SingularValues[A][[2]]];
Clear[t];
ellipseplot = ParametricPlot[A.{Cos[t], Sin[t]}, {t, 0, 2 π},
  PlotStyle -> {{Thickness[0.01], NavyBlue}}, AxesLabel -> {"x", "y"},
  PlotRange -> {{-ranger, ranger}, {-ranger, ranger}}];
```



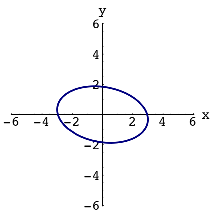
You have the SVD analysis tools to come up with:
->The length of the long axis of this ellipse
->The length of the short axis of this ellipse
->The perpendicular frame that defines the long and the short axes of this ellipse
->The area enclosed by this ellipse.

Do it.

□G.2.a.ii) Modifying the ellipse

Stay with the same ellipse as in part i)

```
Show[ellipseplot];
```

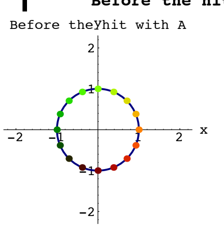


Plot the new ellipse that you get by
-> keeping the short axis of this new ellipse the same as it is in the ellipse plotted above but
->making the long axis of the new ellipse 2 times longer than it is in the ellipse plotted above.
How is the area of the region enclosed by the new ellipse related to the area enclosed by the ellipse plotted above?

□G.2.b.i) Hitting and measuring

Here's the unit circle:

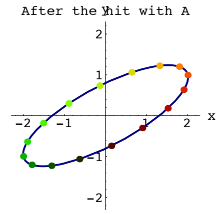
```
Clear[x, y, t];
{x[t_], y[t_]} = {Cos[t], Sin[t]};
{tlow, thigh} = {0, 2 π};
ranger = 2.3;

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 matrix A and a plot of the ellipse that results from hitting the unit circle with A:

```
A = (1.8  -0.9
     1.2   0.3);
MatrixForm[A]
after = Show[hitplotter[A],
  hitpointplotter[A], PlotLabel → "After the hit with A",
  DisplayFunction → $DisplayFunction];
```

$$\begin{pmatrix} 1.8 & -0.9 \\ 1.2 & 0.3 \end{pmatrix}$$

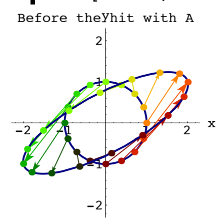


Given that the area inside the unit circle measures out to π square units, do an SVD analysis of the matrix A and use your analysis to calculate the area measurement of the region inside and on the plotted ellipse.

□G.2.b.ii) More bunched at the ends

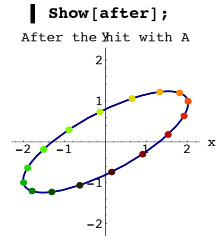
Stay with the same set-up as in part i) and look at this plot showing the unit circle, the ellipse and some action arrows

```
Clear[actionarrows];
actionarrows[matrix2D_] :=
  Table[Arrow[matrix2D.{x[t], y[t]} - {x[t], y[t]}, Tail → {x[t], y[t]}
    VectorColor → pointcolor[t]], {t, tlow, thigh - jump, jump}];

Show[before, actionarrows[A], after];
```



Notice that the plotted points on the circle are evenly spaced along the circle but after you hit these points with A, the resulting points are not evenly spaced along the ellipse.

```
Show[after];
```

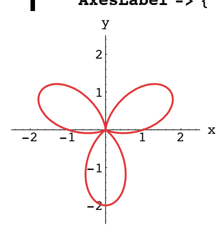


Try to use the hanger frame and the stretch factors to explain why the hit points at sharply curved part of the ellipse are bunched more closely together than those on the flat part.

□G.2.c) Hitting with A and its transpose

Here's a closed curve

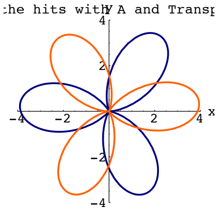
```
Clear[x, y, t];
{x[t_], y[t_]} = {Cos[t] (1 + Sin[3 t]), Sin[t] (1 + Sin[3 t])};
{tlow, thigh} = {0, 2 π};
ranger = 2.5;

curveplot = ParametricPlot[{x[t], y[t]}, {t, tlow, thigh},
   PlotStyle -> {{RoseMadder, Thickness[0.01]}},
PlotRange -> {{-ranger, ranger}, {-ranger, ranger}},
   AxesLabel -> {"x", "y"}];
```



Here are the two curves you get by hitting the curve above with a random 2D matrix A and hitting with the transpose A^t of A:

```
A = (Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
     Random[Real, {-2, 2}]  Random[Real, {-2, 2}]);
MatrixForm[A]

{xstretch, ystretch} = SingularValues[A][[2]];
        ranger = 2 Max[{1.0, Max[{xstretch, ystretch}]}];

ParametricPlot[{A.{x[t], y[t]}, Transpose[A].{x[t], y[t]}},
 {t, tlow, thigh}, PlotStyle →
  {{NavyBlue, Thickness[0.01]}, {CadmiumOrange, Thickness[0.01]}},
 PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
 AxesLabel → {"x", "y"},
 PlotLabel → "After the hits with A and Transpose[A]"];
```

$$\begin{pmatrix} 1.77271 & -0.826654 \\ 0.873094 & 1.79319 \end{pmatrix}$$

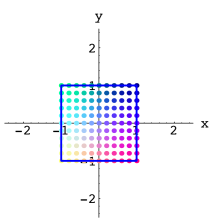


Rerun both cells a couple of times. Each time you run it, you can say with great authority that the area enclosed by one of these curves measures out to the same value as the area enclosed by other curve.
What fact backs up this observation?

□ G.2.d.i) Hitting on a square

Here's the square with corners at {-1,-1}, {1,-1}, {1,1} and {-1,1}:

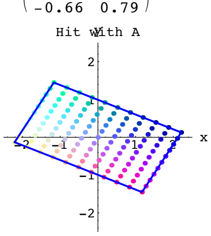
```
jump = 0.1;
Clear[parallelogramplotter, basepoint, side1, side2, pointcolor];
ranger = 2.5;

pointcolor[r_, t_] =
  RGBColor[0.5 (Cos[π t] + 1), 0.5 (Cos[π r] + 1), 0.5 (Sin[π t] + 1)];

parallelogramplotter[basepoint_, side1_, side2_] :=
  {Table[Graphics[{PointSize[0.025],
      pointcolor[r, t], Point[basepoint + t side1 + r side2]}],
    {t, 0, 1, jump}, {r, 0, 1, jump}], Graphics[
    {Thickness[0.01], Blue, Line[{basepoint, basepoint + side1,
       basepoint + side1 + side2, basepoint + side2, basepoint}]}]};

basepoint = {-1, -1};
side1 = {0, 2};
side2 = {2, 0};

Show[parallelogramplotter[basepoint, side1, side2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  Axes → True, AxesLabel → {"x", "y"}];
```



Here's a matrix A and the parallelogram that results from hitting this square and the points inside it with A:

```
A = ( 1.69   0.52 );
    ( -0.66  0.79 )
MatrixForm[A]

Ahit = Show[parallelogramplotter[A.basepoint, A.side1, A.side2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  PlotLabel → "Hit with A", Axes → True, AxesLabel → {"x", "y"}];
```

$$\begin{pmatrix} 1.69 & 0.52 \\ -0.66 & 0.79 \end{pmatrix}$$

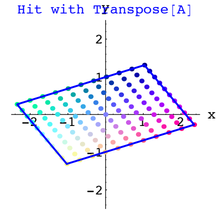


Do an SVD analysis of A and use what you get to help you to measure the area of this parallelogram.

□ G.2.d.ii) Hitting the transpose on the same square

Here is what you get when you hit the original square in part i) above with A^t, the transpose of A:

```
B = Transpose[A];
Transposehit =
 Show[parallelogramplotter[B.basepoint, B.side1, B.side2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  PlotLabel → "Hit with Transpose[A]",
  Axes → True, AxesLabel → {"x", "y"}];
```



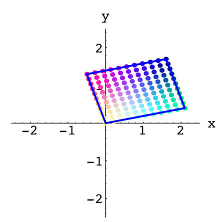
Grab both plots and animate briefly.

Although this is not the same parallelogram as in part i), clued-in matrix folks know that the area of this parallelogram is guaranteed to measure out to the same value as the area of the parallelogram in part i).
How do the clued-in matrix folks know this?

□ G.2.e.i) Using a parallelogram to define a matrix

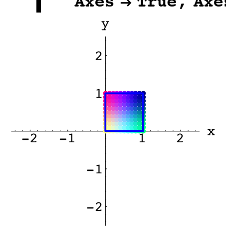
Here's a parallelogram with lots of points inside:

```
jump = 0.1;
Clear[parallelogramplotter, basepoint, side1, side2, pointcolor];
ranger = 2.5;
pointcolor[r_, t_] =
  RGBColor[0.5 (Cos[π t] + 1), 0.5 (Cos[π r] + 1), 0.5 (Sin[π t] + 1)];

parallelogramplotter[basepoint_, side1_, side2_] :=
  {Table[Graphics[{PointSize[0.025],
      pointcolor[r, t], Point[basepoint + t side1 + r side2]}],
    {t, 0, 1, jump}, {r, 0, 1, jump}], Graphics[
    {Thickness[0.01], Blue, Line[{basepoint, basepoint + side1,
       basepoint + side1 + side2, basepoint + side2, basepoint}]}]};

basepoint = {0, 0};
paraside1 = {2.1, 0.4};
paraside2 = {-0.5, 1.3};

Show[parallelogramplotter[basepoint, paraside1, paraside2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  Axes → True, AxesLabel → {"x", "y"}];
```



Here's the square with corners at {0,0}, {1,0},{1,1} and {0,1}

```
basepoint = {0, 0};
squareside1 = {1, 0};
squareside2 = {0, 1};

Show[parallelogramplotter[basepoint, squareside1, squareside2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  Axes → True, AxesLabel → {"x", "y"}];
```



Use the sides of the parallelogram to define a matrix A this way:

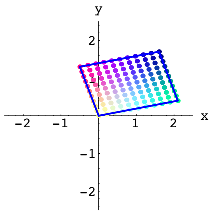
```
A = Transpose[{paraside1, paraside2}];
MatrixForm[A]
```

$$\begin{pmatrix} 2.1 & -0.5 \\ 0.4 & 1.3 \end{pmatrix}$$

The vertical columns of A are the vectors that define the parallelogram.

Here's what you get when you hit the square with A:

```
Show[
 parallelogramplotter[A.basepoint, A.squareside1, A.squareside2],
 PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
 Axes → True, AxesLabel → {"x", "y"}];
```



Determine the relationship between this parallelogram and the original parallelogram.

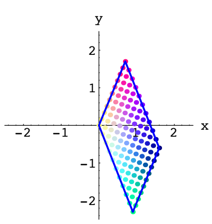
□G.2.e.ii) Measuring the area of that parallelogram

Do an SVD analysis of matrix A in part i) to help to measure the area enclosed within the original parallelogram.

□G.2.e.iii) Using another parallelogram to define another matrix

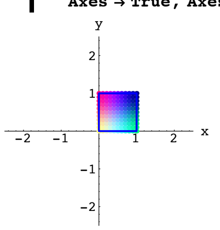
Here's a new parallelogram with lots of points inside:

```
jump = 0.1;
Clear[parallelogramplotter, basepoint, side1, side2, pointcolor];
ranger = 2.5;
pointcolor[r_, t_] =
  RGBColor[0.5 (Cos[π t] + 1), 0.5 (Cos[π r] + 1), 0.5 (Sin[π t] + 1)];

parallelogramplotter[basepoint_, side1_, side2_] :=
  {Table[Graphics[{PointSize[0.025],
      pointcolor[r, t], Point[basepoint + t side1 + r side2]}],
    {t, 0, 1, jump}, {r, 0, 1, jump}], Graphics[
    {Thickness[0.01], Blue, Line[{basepoint, basepoint + side1,
       basepoint + side1 + side2, basepoint + side2, basepoint}]}]};

basepoint = {0, 0};
paraside1 = {0.9, -2.3};
paraside2 = {0.7, 1.7};

Show[parallelogramplotter[basepoint, paraside1, paraside2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  Axes → True, AxesLabel → {"x", "y"}];
```



Here's the square with corners at {0,0}, {1,0},{1,1} and {0,1}

```
basepoint = {0, 0};
squareside1 = {1, 0};
squareside2 = {0, 1};

Show[parallelogramplotter[basepoint, squareside1, squareside2],
  PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
  Axes → True, AxesLabel → {"x", "y"}];
```



Make a matrix A so that hitting this square with A gives the parallelogram. Do an SVD analysis of A to help you to measure the area enclosed within the parallelogram.

G.3) Linear Algebra: Using 2D matrices to try to solve linear equations*

□G.3.a) Success when the coefficient matrix is invertible

Use what you know about matrices and their inverses to try come up with the x and the y that solve the simultaneous linear equations:

$2.37\,x + 1.23\,y = -9.81$
$1.83\,x - 0.94\,y = 3.59.$

□G.3.b.i) Failure when the coefficient matrix is not invertible

Here's a matrix which is not invertible:

```
A = ( 2.   -1.2 );
    ( -0.5  0.3 )
MatrixForm[A]
```

$\begin{pmatrix} 2. & -1.2 \\ -0.5 & 0.3 \end{pmatrix}$

```
Inverse[A]
```

Inverse::sing : Matrix {{2., -1.2}, {-0.5, 0.3}} is singular.

Inverse[{{2., -1.2}, {-0.5, 0.3}}]

As you know, in spite of this, the corresponding linear system, for given numbers u and v,

$2.0\,x - 1.2\,y = u$
$-0.5\,x + 0.3\,y = v$

might have many or no solutions for x and y, depending on where the point {u,v} is located.

Go with

{u,v} = {2.0, 0.0},

and explain how you can tell that the linear system

$2.0\,x - 1.2\,y = u$
$-0.5\,x + 0.3\,y = v$

has no solution for x and y.

□G.3.b.ii) Success

Go with

{u,v} = {1.8, -0.45},

and explain how you can tell that the linear system

$2.0\,x - 1.2\,y = u$
$-0.5\,x + 0.3\,y = v$

has a solution for x and y.

□G.3.b.iii) More solutions

Stay with

{u,v} = {1.8, -0.45},

describe where all the solutions of

$2.0\,x - 1.2\,y = u$
$-0.5\,x + 0.3\,y = v$

come from.

□G.3.c.i) Linear systems and lines

Here are two linear equations:

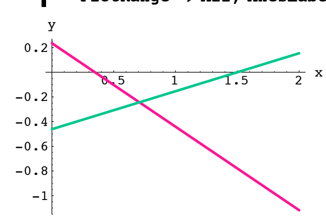
```
Clear[x, y];
equation1 = 2.3 x + 3.4 y == 0.8
equation2 = 0.4 x - 1.3 y == 0.6
```

2.3 x + 3.4 y == 0.8

0.4 x - 1.3 y == 0.6

Each equation defines a line. Here is a plot of both lines:

```
y1sol[x_] = y /. Solve[equation1, y][[1]];
y2sol[x_] = y /. Solve[equation2, y][[1]];

Plot[{y1sol[x], y2sol[x]}, {x, 0, 2}, PlotStyle →
   {{DeepPink, Thickness[0.01]}, {TurquoiseBlue, Thickness[0.01]}},
  PlotRange -> All, AxesLabel → {"x", "y"}];
```



The question here is:

How is the solution of the linear system

$2.3\,x + 3.4\,y = 0.8$
$0.4\,x - 1.3\,y = 0.6$

related to the point at which the two lines cross?

□G.3.d) Determinants and linear systems

Here's a totally cleared linear system:

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

ColumnForm[Thread[linearsystem = A.{x, y} == {u, v}]]
```

a x + b y == u

c x + d y == v

The coefficient matrix is:

```
MatrixForm[A]
```

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

Remembering that |Det[A]| is the product of the SVD stretch factors xstretch and ystretch for A, agree or disagree with these statements:

• When you go with specific a, b, c and d that make Det[A] ≠ 0, then for each choice of {u,v}, the corresponding linear system has exactly one solution.

Put answer here.

• When you go with a, b, c and d that make Det[A] = 0, then for each choice of {u,v}, the

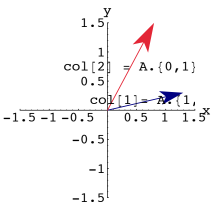
corresponding linear system either has no solution (overdetermined) or many solutions (underdetermined).
Put answer here.

G.4) Determinant fundamentals

□G.4.a.i) Columns and the sign of the determinant

Here's a matrix A together with a plot of its columns:

```
A = {{1.3, 0.8}, {0.3, 1.5}};
Clear[columnplotter, matrix];
columnplotter[matrix_] :=
 Show[Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.4], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.5],
  Graphics[{Text["col[1]= A.{1,0}", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2] = A.{0,1}", 0.5 matrix.{0, 1}]}],
  Axes → True, AxesLabel → {"x", "y"},
  PlotRange -> {{-1.5, 1.5}, {-1.5, 1.5}},
  DisplayFunction -> Identity];

Show[columnplotter[A], DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```



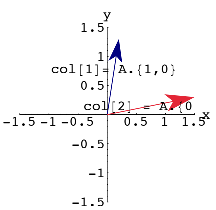
$A = \begin{pmatrix} 1.3 & 0.8 \\ 0.3 & 1.5 \end{pmatrix}$

How does the plot signal that Det[A] > 0?

□G.4.a.ii) Columns and the sign of the determinant

Here's a new matrix A together with a plot of its columns:

```
A = {{0.2, 1.5}, {1.3, 0.3}};
Show[columnplotter[A], DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```



$A = \begin{pmatrix} 0.2 & 1.5 \\ 1.3 & 0.3 \end{pmatrix}$

How does the plot signal that Det[A] < 0?

□G.4.b) Left or right?

Here's a random perpendicular frame and its corresponding hanger matrix:

```
Clear[perpframe];
s = Random[Real, {0, π}];
{perpframe[1], perpframe[2]} = {{Cos[s], Sin[s]},
   ((-1)^Random[Integer,{0,1}]) {Cos[s + π/2], Sin[s + π/2]}};
hanger = Transpose[{perpframe[1], perpframe[2]}];
MatrixForm[hanger]
```

$\begin{pmatrix} -0.586857 & -0.809691 \\ 0.809691 & -0.586857 \end{pmatrix}$

The determinant of this matrix is:

```
Det[hanger]
```

1.

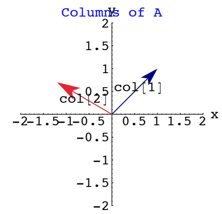
You make the call:
Is this perpendicular frame a right hand or a left hand perpendicular frame?

□G.4.c) Products

Here's a plot of the vertical columns of

$$A = \begin{pmatrix} 1.0 & -1.2 \\ 1.0 & 1.0 \end{pmatrix};$$

```
Clear[columnplotter, matrix];
columnplotter[matrix_] :=
 Show[Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.4], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.5],
  Graphics[{Text["col[1]", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2]", 0.5 matrix.{0, 1}]}], Axes → True,
  AxesLabel → {"x", "y"}, PlotRange -> {{-2, 2}, {-2, 2}},
  DisplayFunction -> Identity];
A = {{1.0, -1.2}, {1.0, 0.7}};

Show[columnplotter[A], PlotLabel -> "Columns of A",
 DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```

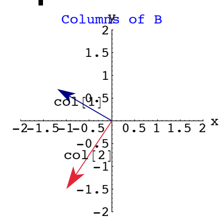


$A = \begin{pmatrix} 1. & -1.2 \\ 1. & 0.7 \end{pmatrix}$

The shorter angle from column[1] of A to column[2] of A is counterclockwise; so the orientation of the columns of A is positive.

Now look at the columns of $B = \begin{pmatrix} -1.2 & -1.0 \\ 0.7 & -1.5 \end{pmatrix}$:

```
B = {{-1.2, -1.0}, {0.7, -1.5}};
Show[columnplotter[B], PlotLabel -> "Columns of B",
 DisplayFunction -> $DisplayFunction];
"B = " MatrixForm[B]
```



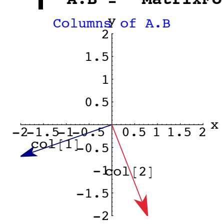
$B = \begin{pmatrix} -1.2 & -1. \\ 0.7 & -1.5 \end{pmatrix}$

The shorter angle from column[1] of B to column[2] of B is counterclockwise; so the orientation of the columns of B is positive.

Now look at the columns of the product

$$A.B = \begin{pmatrix} 1.0 & -1.2 \\ 1.0 & 1.0 \end{pmatrix}.\begin{pmatrix} -1.2 & -1.0 \\ 0.7 & -1.5 \end{pmatrix}:$$

```
Show[columnplotter[A.B], PlotLabel -> "Columns of A.B",
 DisplayFunction -> $DisplayFunction];
"A.B = " MatrixForm[A.B]
```



$A.B = \begin{pmatrix} -2.04 & 0.8 \\ -0.71 & -2.05 \end{pmatrix}$

The shorter angle from column[1] of A.B to column[2] of A.B is counterclockwise; so the orientation of the columns of A.B is positive.
Here you took two matrices A and B each with positively oriented columns and found that the columns of the product A.B are also positively oriented.

Was this just a fluke?
Or is it true that when you go with any two matrices A and B each with positively oriented columns, then the columns of the product A.B are guaranteed to be positively oriented?

On what facts do you base your answer?

□G.4.d) Interchanging the rows of a 2D matrix

In the Basics, you saw that when you interchange the columns of a 2D matrix, you change the sign but not the absolute value of the determinant.
Try it out on a cleared 2D matrix A:

```
Clear[a, b, c, d]
A = {{a, b}, {c, d}};
MatrixForm[A]
```

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

```
colinterchangedA = A.{{0, 1}, {1, 0}};
MatrixForm[colinterchangedA]
```

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

```
-Det[A] == Det[colinterchangedA]
```

True

Now go with a new cleared matrix A:

```
Clear[a, b, c, d]
A = (a b
     c d);
MatrixForm[A]
```

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

Here's how to interchange the rows of A:

```
rowinterchangedA = (0 1
                    1 0).A;
MatrixForm[rowinterchangedA]
```

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

Look at this:

```
-Det[A] == Det[rowinterchangedA]
```

True

Explain why that happened.

Click on the right for a little tip.

```
Det[(0 1
     1 0)]
```

-1

□ G.4.e) The aligner and hanger frames set the sign of the determinant

From the Tutorials:

- If A is a hanger or aligner based on a right hand frame, then Det[A] = 1.
- If A is a hanger or aligner based on a left hand frame, then Det[A] = —1.
- If A is a stretcher, then Det[A] = product of stretch factors.

So if A = hanger.stretcher.aligner, then

Det[A] = Det[hanger] xstretch ystretch Det[aligner].

Use this good information to help to answer these questions:

- How do you know that saying Det[A] < 0 is the same as saying that either
  the aligner frame is a right hand frame and the hangerframe is a left hand frame
  or
  the aligner frame is a left hand frame and the hangerframe is a right hand frame.

Put answer here.

- How do you know that if Det[A] < 0, then a hit with A incorporates a flip?

Put answer here.

- How do you know that if Det[A] < 0, then a hit with A does not preserve orientation?

Put answer here.

- How do you know that saying Det[A] > 0 is the same as saying that either
  the aligner frame is a right hand frame and the hangerframe is a right hand frame
  or
  the aligner frame is a left hand frame and the hangerframe is a left hand frame.

Put answer here.

- How do you know that if Det[A] > 0, then a hit with A incorporates no flip or two flips(resulting in no flip)?

Put answer here.

- How do you know that if Det[A] > 0, then a hit with A preserves orientation?

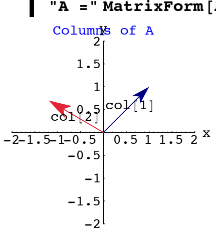
Put answer here.

□ G.4.f) Rows and columns

Here's a plot of the vertical columns of

$$A = \begin{pmatrix} 1.0 & -1.2 \\ 1.0 & 1.0 \end{pmatrix};$$

```
Clear[columnplotter, matrix];
columnplotter[matrix_] :=
 Show[Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.4], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.5],
  Graphics[{Text["col[1]", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2]", 0.5 matrix.{0, 1}]}], Axes → True,
  AxesLabel → {"x", "y"}, PlotRange -> {{-2, 2}, {-2, 2}},
  DisplayFunction -> Identity];
A = (1.0 -1.2
     1.0  0.7);

Show[columnplotter[A], PlotLabel -> "Columns of A",
 DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```

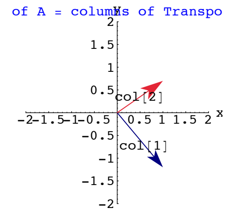


$A = \begin{pmatrix} 1. & -1.2 \\ 1. & 0.7 \end{pmatrix}$

The shorter angle from column[1] of A to column[2] of A is counterclockwise; so the orientation of the columns of A is positive.

Now look at the rows of A (which are the columns of A^t):

```
Show[columnplotter[Transpose[A]],
 PlotLabel -> "Rows of A = columns of Transpose[A]",
 DisplayFunction -> $DisplayFunction];
"A = " MatrixForm[A]
```



$A = \begin{pmatrix} 1. & -1.2 \\ 1. & 0.7 \end{pmatrix}$

The shorter angle from column[1] of A^t (= row[1] of A) to column[2] of A^t (= row[2] of A) is counterclockwise; so the orientation of the rows of A is positive.

Here you took a matrix A with positively oriented columns and found that the rows of A are also positively oriented.

Was this just a fluke?

Or is it true that when you go with a matrix A with positively oriented columns, then the rows of A are guaranteed to be positively oriented?

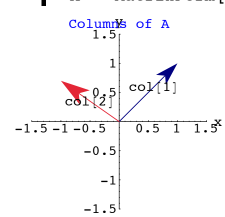
On what facts do you base your answer?

□ G.4.g) Inverses

Here's a plot of the vertical columns of

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

```
Clear[columnplotter, matrix];
columnplotter[matrix_] :=
 Show[Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.4], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.5],

  Graphics[{Text["col[1]", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2]", 0.5 matrix.{0, 1}]}], Axes → True,
  AxesLabel → {"x", "y"}, PlotRange -> {{-1.5, 1.5}, {-1.5, 1.5}},
  DisplayFunction -> Identity];
A = (1.0 -1.0
     1.0  0.7);

Show[columnplotter[A], PlotLabel -> "Columns of A",
 DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```

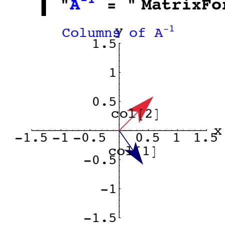


$A = \begin{pmatrix} 1. & -1. \\ 1. & 0.7 \end{pmatrix}$

The shorter angle from column[1] of A to column[2] of A is counterclockwise; so the orientation of the columns of A is positive.

Now look at the columns of A^{-1}

```
Show[columnplotter[Inverse[A]], PlotLabel -> "Columns of A^-1",
 DisplayFunction -> $DisplayFunction];
"A^-1 = " MatrixForm[Inverse[A]]
```



$A^{-1} = \begin{pmatrix} 0.411765 & 0.588235 \\ -0.588235 & 0.588235 \end{pmatrix}$

The shorter angle from column[1] of A^{-1} to column[2] of A^{-1} is counterclockwise; so the orientation of the columns of A^{-1} is positive.

Here you took a matrix A with positively oriented columns and found that the columns of A^{-1} are also positively oriented.

Was this just a fluke?
Or is it true that when you go with a matrix A with positively oriented columns, then the columns of A^{-1} are also guaranteed to be positively oriented?

On what facts do you base your answer?

□ G.4.h.i) Using the determinant formula $\mathrm{Det}\left[\begin{pmatrix} a & b \\ c & d \end{pmatrix}\right] = a\,d - b\,c$

Go with $A = \begin{pmatrix} 4.3 & 5.1 \\ -3.9 & 7.2 \end{pmatrix}$.

The determinant of A is:

```
A = ( 4.3   5.1 );
    ( -3.9  7.2 )
Det[A]
```

50.85

The formula for the determinant is:

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

Use the formula to duplicate this calculation of Det[A].

□ G.4.h.ii) Set a parameter

Go with $A = \begin{pmatrix} 4. & 5. \\ x & 7 \end{pmatrix}$.

The determinant of A is:

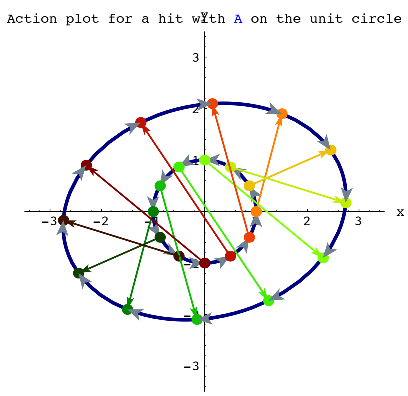
```
Clear[x];
A = ( 2.  4. );
    ( x   6  )
Det[A]
```

12. - 4. x

Use what you see to set x so that A has an SVD stretch factor equal to 0.

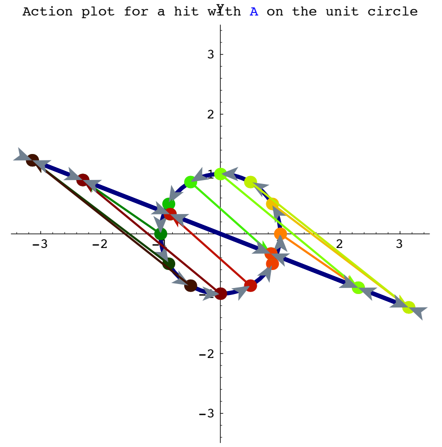
□ G.4.j) Action plots

Here's an action plot showing what a hit with a certain matrix A does to the unit circle:



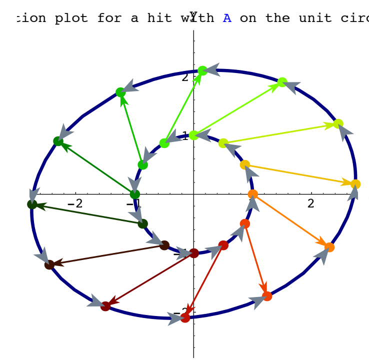
Multiple choice:
Det[A] is
positive........ negative...... zero.............

Here's another action plot showing what a hit with a certain matrix A does to the unit circle:



Multiple choice:
Det[A] is
positive........ negative...... zero.............

Here's another action plot showing what a hit with a certain matrix A does to the unit circle:



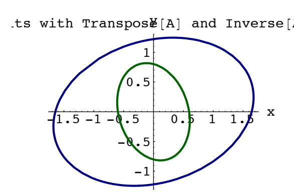
Multiple choice:
Det[A] is
positive........ negative...... zero.............

G.5) $\mathrm{Det}[A] = \mathrm{Det}[A^t]$ and $\mathrm{Det}[A^{-1}] = \frac{1}{\mathrm{Det}[A]}$

□ G.5.a) Hitting A^t and A^{-1} on the unit circle

Here's what you get when you take a random 2D matrix A and hit both A^t and A^{-1} on the unit circle:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
    ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] )

Clear[x, y, t, s];
{tlow, thigh} = {0, 2 π};
{x[t_], y[t_]} = {Cos[t], Sin[t]};

ParametricPlot[{Transpose[A].{x[t], y[t]}, Inverse[A].{x[t], y[t]}},
  {t, tlow, thigh}, PlotStyle → {{Thickness[0.01], NavyBlue},
    {Thickness[0.01], GosiaGreen}}, AxesLabel → {"x", "y"},
  PlotLabel → "Hits with Transpose[A] and Inverse[A]"];
```



Rerun many times.

Rerun several times.
Describe what you see and try to explain why you see it.
Some questions to ponder:
Both ellipses seem to be hanging on the same perpendicular frame. What perpendicular frame is it?
Why does the long axis of each ellipse line up with the short axis of the other?

□ G.5.b.i) $\mathrm{Det}[A] = \mathrm{Det}[A^t]$

Here's a random 2D matrix:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
    ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] )
MatrixForm[A]
```

$$\begin{pmatrix} 0.566863 & -0.344305 \\ 1.05833 & 0.600553 \end{pmatrix}$$

Here are calculations of Det[A] and Det[A^t]:

```
Det[A]
```

0.70482

```
Det[Transpose[A]]
```

0.70482

What is it about the relationship between

the aligner frame for A, the stretch factors for A and the hanger frame for A

and

the aligner frame for A^t, the stretch factors for A^t and the hanger frame for A^t

that explains why

Det[A] = Det[A^t]

for any 2D matrix A?

G.5.b.ii) The effect of interchanging rows on the determinant

In the Basics, you saw that when you go with a matrix

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

and interchange the columns A to get

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

then Det[interchangedA] = - Det[A].

What happens to the determinant when you go with a matrix

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

and interchange the rows A to get

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

Click on the right for a friendly tip.

Interchanging the rows of A interchanges to columns of A^t.

G.5.c) Det[A^{-1}] = $\frac{1}{\text{Det[A]}}$

Here's a random 2D matrix:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
MatrixForm[A]
```

$$\begin{pmatrix} -1.53516 & -0.297565 \\ 0.222105 & -1.99826 \end{pmatrix}$$

Here are calculations of

Det[A^{-1}] and $\frac{1}{\text{Det[A]}}$:

```
Det[Inverse[A]]
```

0.319108

```
  1
------
Det[A]
```

0.319108

Rerun the last three input cells a few times.

What is it about the relationship between

the aligner frame for A, the stretch factors for A and the hanger frame for A

and

the aligner frame for A^{-1}, the stretch factors for A^{-1} and the hanger frame for A^{-1}

that explains why

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

for any 2D matrix A?

G.6) Hit and Tell

G.6.a) The plots of the column vectors of A have their tips right on the ellipse

Here's a random 2D matrix A together with

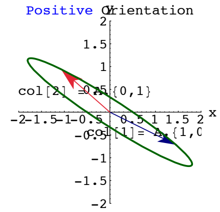
- a plot of the ellipse you get when you hit A on the unit circle

and

- a plot the columns of A:

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

A = ( a[1, 1]  a[1, 2]
      a[2, 1]  a[2, 2] );
ellipseplot = ParametricPlot[A.{Cos[t], Sin[t]},
    {t, 0, 2 Pi}, PlotStyle -> {{GosiaGreen, Thickness[0.01]}},
    DisplayFunction -> Identity];
Clear[columnplotter, matrix];

columnplotter[matrix_] :=
 Show[Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.4], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.5],
  Graphics[{Text["col[1]= A.{1,0}", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2] = A.{0,1}", 0.5 matrix.{0, 1}]}],
  Axes → True, AxesLabel → {"x", "y"}, PlotRange -> {{-2, 2}, {-2, 2}},
  PlotLabel -> If[ Det[A] > 0, "Positive Orientation",
    If[Det[A] < 0, "Negative Orientation", "No Orientation"]],
  DisplayFunction -> Identity];
```

```
Show[columnplotter[A], ellipseplot,
  DisplayFunction -> $DisplayFunction];
"A =" MatrixForm[A]
```



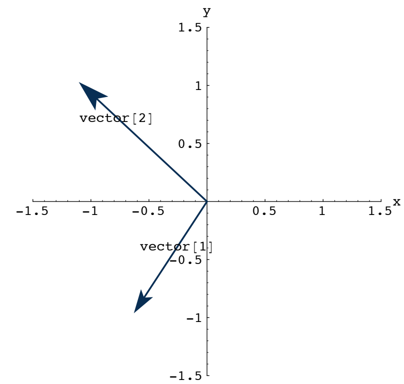
$$A = \begin{pmatrix} 1.45007 & -1.07261 \\ -0.71574 & 0.94263 \end{pmatrix}$$

Rerun several times and then answer this question:

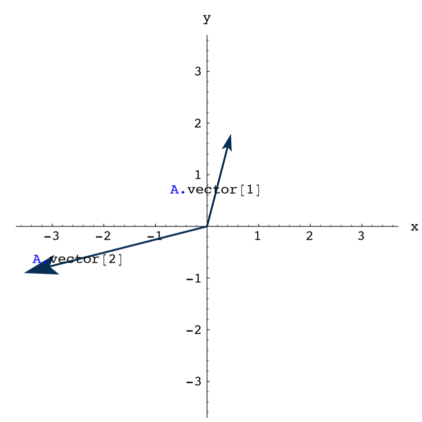
Why do the plots of the column vectors of A have their tips right on the ellipse?

G.6.b) Sign of the determinant

Here are two vectors in 2D:



Here's what happens when you hit these two vectors with a certain matrix A:



You make the call:

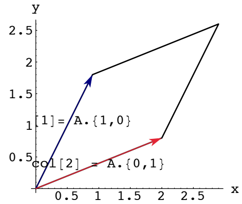
Det[A] is Positive,,,,,,,Zero,,,,,,,Negative,,,,,,,.

On what facts do you base your answer?

G.6.c) Parallelograms and determinants

Here's a 2D matrix A together with the parallelograms you get when you hit A on the unit square with corners at {0,0},{1,0},{1,1} and {0,1}:

```
A = ( 0.9  2.0
      1.8  0.8 );
Clear[hitplotter, matrix];
hitplotter[matrix_] :=
 Show[Graphics[Line[{matrix.{0, 0}, matrix.{1, 0},
     matrix.{1, 1}, matrix.{0, 1}, matrix.{0, 0}}]],
  Arrow[matrix.{1, 0}, Tail → {0, 0}, VectorColor → NavyBlue,
   HeadSize → 0.2], Arrow[matrix.{0, 1}, Tail → {0, 0},
   VectorColor → AlizarinCrimson, HeadSize → 0.2],
  Graphics[{Text["col[1]= A.{1,0}", 0.6 matrix.{1, 0}]}],
  Graphics[{Text["col[2] = A.{0,1}", 0.5 matrix.{0, 1}]}],
  Axes → True, AxesLabel → {"x", "y"}];
hitplotter[A];
```



The determinant of A is:

```
A = ( 0.9  2.0
      1.8  0.8 );
Det[A]
```

-2.88

How is the calculation of Det[A\ related to the plot?

□G.6.d.i) Hitting with A and then hitting with the matrix you get by interchanging the columns of A

Here's a random 2D matrix A:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
MatrixForm[A]
```

$$\begin{pmatrix} -0.676083 & -1.89385 \\ 0.861451 & -0.00218732 \end{pmatrix}$$

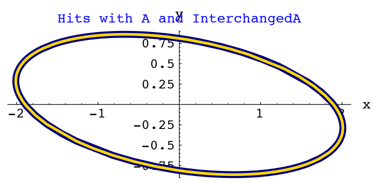
Here's the matrix you get when you interchange the two columns of A:

```
InterchangedA = A.( 0  1
                    1  0 );
MatrixForm[InterchangedA]
```

$$\begin{pmatrix} -1.89385 & -0.676083 \\ -0.00218732 & 0.861451 \end{pmatrix}$$

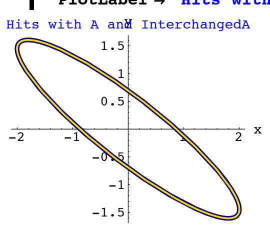
Here's what happens when you hit both of these matrices on the unit circle:

```
Clear[x, y, t, s];
{tlow, thigh} = {0, 2 π};
{x[t_], y[t_]} = {Cos[t], Sin[t]};

ParametricPlot[{A.{x[t], y[t]}, InterchangedA.{x[t], y[t]}},
 {t, tlow, thigh}, PlotStyle → {{Thickness[0.02], NavyBlue},
   {Thickness[0.008], Gold}}, AxesLabel → {"x", "y"},
 PlotLabel → "Hits with A and InterchangedA"];
```



Try it again:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
InterchangedA = A.( 0  1
                    1  0 );

ParametricPlot[{A.{x[t], y[t]}, InterchangedA.{x[t], y[t]}},
 {t, tlow, thigh}, PlotStyle → {{Thickness[0.02], NavyBlue},
   {Thickness[0.008], Gold}}, AxesLabel → {"x", "y"},
 PlotLabel → "Hits with A and InterchangedA"];
```



These plots signal a relationship between the SVD stretch factors of A and interchangedA.
These plots also signal a relationship between the SVD hangerframes of A and interchangedA.
What do you say these relationships are?

□G.6.d.ii) Hitting with A and then hitting with the matrix you get by interchanging the rows of A

Here's a random 2D matrix A:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
MatrixForm[A]
```

$$\begin{pmatrix} -1.03599 & 0.198045 \\ 1.88362 & -0.106736 \end{pmatrix}$$

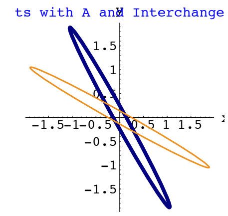
Here's the matrix you get when you interchange the two rows of A:

```
InterchangedA = ( 0  1
                  1  0 ).A;
MatrixForm[InterchangedA]
```

$$\begin{pmatrix} 1.88362 & -0.106736 \\ -1.03599 & 0.198045 \end{pmatrix}$$

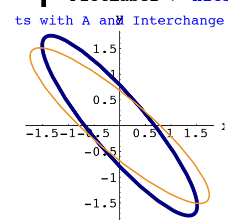
Here's what happens when you hit both of these matrices on the unit circle:

```
Clear[x, y, t, s];
{tlow, thigh} = {0, 2 π};
{x[t_], y[t_]} = {Cos[t], Sin[t]};

ParametricPlot[{A.{x[t], y[t]}, InterchangedA.{x[t], y[t]}},
 {t, tlow, thigh}, PlotStyle → {{Thickness[0.02], NavyBlue},
   {Thickness[0.008], Carrot}}, AxesLabel → {"x", "y"},
 PlotLabel → "Hits with A and InterchangedA"];
```



Try it again:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
InterchangedA = ( 0  1
                  1  0 ).A;

ParametricPlot[{A.{x[t], y[t]}, InterchangedA.{x[t], y[t]}},
 {t, tlow, thigh}, PlotStyle → {{Thickness[0.02], NavyBlue},
   {Thickness[0.008], Carrot}}, AxesLabel → {"x", "y"},
 PlotLabel → "Hits with A and InterchangedA"];
```



These plots signal a relationship between the SVD stretch factors of A and interchangedA.
These plots also signal a relationship between the SVD hanger frames of A and interchangedA.
What do you say these relationships are?

Click on the right for a friendly tip.

To get the matrix resulting from interchanging the rows of A, you go with

InterchangedA $= \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.A.

Hits with $\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.flip about the line y = x:

```
s = π/4;
{perpframe[1], perpframe[2]} =
  {{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}};
Clear[alignerframe];
{alignerframe[1], alignerframe[2]} = {perpframe[1], perpframe[2]};
aligner = {alignerframe[1], alignerframe[2]};
stretcher = {{1, 0}, {0, 1}};
Clear[hangerframe];
{hangerframe[1], hangerframe[2]} = {perpframe[1], -perpframe[2]};
hanger = Transpose[{hangerframe[1], hangerframe[2]}];
flipper = hanger.stretcher.aligner;
MatrixForm[flipper]
```

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

G.7) Area, length, isometries and rotations*

□G.7.a.i) When hits with A do not change area measurements

If A is a 2D matrix and hits with A do not change area measurements, then what is

|Det[A]| = xstretch ystretch

guaranteed to be?

□G.7.a.ii) When hits with A do not change area measurements but do change lengths

Make a 2D matrix A and so that hits with A do change some length measurements but do not change area measurements.

G.7.b.i) Isometries in 2D

Many folks say that a 2D matrix A is an isometry if

||A.X|| = ||X||

for all 2D vectors X.
Saying that a 2D matrix A is an isometry is the same as saying that hits with A do not change length measurements.

If A is a 2D matrix and hits with A do not change length measurements, then what are the two SVD stretch factors of A guaranteed to be?

G.7.b.ii) Saying that a 2D matrix is an isometry is the same as saying that both the SVD stretch factors of A are equal to 1

When you make a 2D matrix A with both SVD stretch factors equal to 1, then you are guaranteed that

A.alignerframe[1] = hangerframe[1]
A.alignerframe[2] = hangerframe[2]

When you take any 2D X and resolve it into components along the aligner frame vectors, you get

$$X = \sum_{j=1}^{2} (\text{X.alignerframe[j]})\ \text{alignerframe[j]}$$

and

$$||\,X\,|| = \sqrt{\sum_{j=1}^{2} (\text{X.alignerframe[j]})^2}\,.$$

When you hit the same 2D X with A, you get

$$\text{A.X} = \sum_{j=1}^{2} (\text{X.alignerframe[j]})\ \text{A.alignerframe[j]}$$

$$= \sum_{j=1}^{2} (\text{X.alignerframe[j]})\ \text{hangerframe[j]}$$

and

$$||\,\text{A.X}\,|| = \sqrt{\sum_{j=1}^{2} (\text{X.alignerframe[j]})^2}\,.$$

Is this enough to tell you that saying that a 2D matrix is an isometry is the same as saying that both the SVD stretch factors of A are equal to 1?

G.7.b.iii) Isometries, rotations and flippers

Explain in detail:

- If A is a 2D isometry matrix, then |Det[A]| = 1.

Put answer here.

- If A is a 2D rotation matrix, then A is an isometry and Det[A] = 1.

Put answer here.

- If A is a 2D isometry matrix and Det[A] = 1, then A is a rotation matrix.

Put answer here.

- If A is a 2D line flipper (reflection matrix) matrix, then A is an isometry and Det[A] = -1.

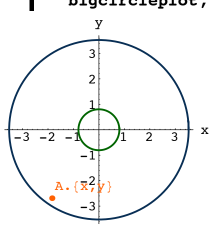
Put answer here.

G.8) Biggest and smallest stretches

G.8.a.i) Min[xstretch,ystretch] ||{x,y}|| ≤ ||A.{x,y}|| ≤ Max[{xstretch,ystretch}] ||{x,y}||.

Look at this plot:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
    ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] )
{xstretch, ystretch} = SingularValues[A][[2]];
MatrixForm[A];
{x, y} = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
xynorm = √({x, y}.{x, y});

hitxyplot =
  Graphics[{CadmiumOrange, PointSize[0.03], Point[A.{x, y}]}];
hitxylabel = Graphics[
   {CadmiumOrange, Text["A.{x,y}", A.{x, y}, {-1, -1.5}]}];

littlecircleplot = Graphics[{GosiaGreen, Thickness[0.01],
   Circle[{0, 0}, xynorm Min[{xstretch, ystretch}]]}];
bigcircleplot = Graphics[{Indigo, Thickness[0.01],
   Circle[{0, 0}, xynorm Max[{xstretch, ystretch}]]}];

Show[hitxyplot, hitxylabel, littlecircleplot,
  bigcircleplot, Axes → True, AxesLabel → {"x", "y"}];
```



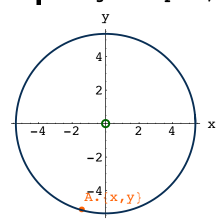
This plot shows:

- A.{x,y} for a random point {x,y} and a random 2D matrix A.
- The circle of
  radius = ||{x,y}|| Min[{xstretch,ystretch}]
  centered at {0,0}.
- The circle of
  radius =||{x,y}|| Max[{xstretch,ystretch}]
  centered at {0,0}.

Here xstretch and ystretch are the SVD stretch factors of A

See some more:

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
    ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] )
{xstretch, ystretch} = SingularValues[A][[2]];
MatrixForm[A];
{x, y} = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
xynorm = √({x, y}.{x, y});

hitxyplot =
  Graphics[{CadmiumOrange, PointSize[0.03], Point[A.{x, y}]}];
hitxylabel = Graphics[
   {CadmiumOrange, Text["A.{x,y}", A.{x, y}, {-1, -1.5}]}];

littlecircleplot = Graphics[{GosiaGreen, Thickness[0.01],
   Circle[{0, 0}, xynorm Min[{xstretch, ystretch}]]}];
bigcircleplot = Graphics[{Indigo, Thickness[0.01],
   Circle[{0, 0}, xynorm Max[{xstretch, ystretch}]]}];

Show[hitxyplot, hitxylabel, littlecircleplot,
  bigcircleplot, Axes → True, AxesLabel → {"x", "y"}];
```



Rerun several times - each time you get a new matrix A and a new {x,y}

Explain how the plots reflect the fact that

Min[xstretch,ystretch] ||{x,y}|| ≤
||A.{x,y}|| ≤
Max[{xstretch,ystretch}] ||{x,y}||.

Click on the right for a tip.

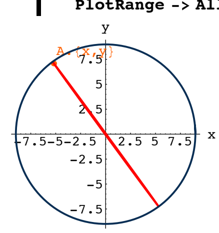
||{x,y}|| $=\sqrt{\{x, y\}.\{x, y\}}$;
this is the distance from {0,0} to {x,y}.

||A.{x,y}|| $= \sqrt{(A.\{x, y\}).(A.\{x, y\})}$;
this is the distance from {0,0} to A.{x,y}.

G.8.a.ii) Why it works

Look at this embellishment of the plot in part i):

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
    ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}] )
{xstretch, ystretch} = SingularValues[A][[2]];
MatrixForm[A];
{x, y} = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
xynorm = √({x, y}.{x, y});

ellipseplot = ParametricPlot[A.{xynorm Cos[t], xynorm Sin[t]},
   {t, 0, 2 π}, PlotStyle -> {{Red, Thickness[0.01]}},
   DisplayFunction -> Identity];

hitxyplot =
  Graphics[{CadmiumOrange, PointSize[0.03], Point[A.{x, y}]}];
hitxylabel = Graphics[
   {CadmiumOrange, Text["A.{x,y}", A.{x, y}, {-1, -1.5}]}];

littlecircleplot = Graphics[{GosiaGreen, Thickness[0.01],
   Circle[{0, 0}, xynorm Min[{xstretch, ystretch}]]}];
bigcircleplot = Graphics[{Indigo, Thickness[0.01],
   Circle[{0, 0}, xynorm Max[{xstretch, ystretch}]]}];

Show[hitxyplot, hitxylabel,
  littlecircleplot, ellipseplot , bigcircleplot,
  PlotRange -> All, Axes → True, AxesLabel → {"x", "y"}];
```



This plot shows:

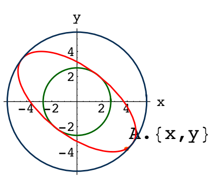
- A.{x,y} for a random point {x,y} and a random 2D matrix A.
- The circle of
  radius = ||{x,y}|| **Min**[{xstretch,ystretch}]
centered at {0,0}.
- The circle of
  radius =||{x,y}|| **Max**[{xstretch,ystretch}]
centered at {0,0}.
- The the ellipse you get when you hit A on the circle centered at {0,0} that runs through {x,y}.

Rerun several times and then say why it is guaranteed that A.{x,y} plots out between the two circles..

□G.8.a.iii) If A has two positive stretch factors and {x,y} is not {0,0}, then A.{x,y} is not {0,0}

Take another look at this embellishment of the plot in part i):

```
A = ( Random[Real, {-2, 2}]  Random[Real, {-2, 2}]
      Random[Real, {-2, 2}]  Random[Real, {-2, 2}] );
{xstretch, ystretch} = SingularValues[A][[2]];
MatrixForm[A];
{x, y} = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
xynorm = √{x, y}.{x, y};

ellipseplot = ParametricPlot[A.{xynorm Cos[t], xynorm Sin[t]},
   {t, 0, 2 π}, PlotStyle -> {{Red, Thickness[0.01]}},
   DisplayFunction -> Identity];

hitxyplot =
  Graphics[{CadmiumOrange, PointSize[0.03], Point[A.{x, y}]}];
hitxylabel = Graphics[
  {Black, Text["A.{x,y}", A.{x, y}, {-1, -1.5}]}];

littlecircleplot = Graphics[{GosiaGreen, Thickness[0.01],
   Circle[{0, 0}, xynorm Min[{xstretch, ystretch}]]}];
bigcircleplot = Graphics[{Indigo, Thickness[0.01],
   Circle[{0, 0}, xynorm Max[{xstretch, ystretch}]]}];

Show[hitxyplot, hitxylabel,
  littlecircleplot, ellipseplot , bigcircleplot,
  PlotRange -> All, Axes → True, AxesLabel → {"x", "y"}];
```



This plot shows:

- A.{x,y} for a random point {x,y} and a random 2D matrix A.
- The circle of
  radius = ||{x,y}|| **Min**[{xstretch,ystretch}]
centered at {0,0}.
- The circle of
  radius =||{x,y}|| **Max**[{xstretch,ystretch}]
centered at {0,0}.
- The the ellipse you get when you hit A on the circle centered at {0,0} that runs through {x,y}.

Rerun several times and then explain this statement:
If A has two positive stretch factors and {x,y} is not {0,0}, then A.{x,y} is not {0,0}.
In other words, a hit with A cannot squash a non-zero vector onto the zero vector.

G.9) $Y \times X = -X \times Y$

□G.9.a) $Y \times X = -X \times Y$

Here are two random 3D vectors X and Y:

```
X = {Random[Real, {-2, 2}],
  Random[Real, {-2, 2}], Random[Real, {-2, 2}]}
Y = {Random[Real, {-2, 2}], Random[Real, {-2, 2}],
  Random[Real, {-2, 2}]}
```

```
{1.36408, -0.370042, -1.50157}
{1.84642, -0.767968, -0.919513}
```

The cross product $X \times Y$ is:

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

```
{-0.812902, -1.51825, -0.364314}
```

But the cross product $Y \times X$ is:

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

```
{0.812902, 1.51825, 0.364314}
```

Evidently

$$Y \times X = -X \times Y.$$

Use the fact that if X = {a, b, c} and Y = {r, s, t}, then

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

to explain why it is certain that $Y \times X = -X \times Y$.

Click on the right for a tip.

Look at:

```
Clear[a, b, c, 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}
```

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

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