Matrices, Geometry&*Mathematica*

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

Publisher: Math Everywhere, Inc.

MGM.09 Eigensense: Diagonalizable Matrices, Matrix Exponential, Matrix Powers and Dynamical Systems

GIVE IT A TRY!

G.1) Eigensense*

A kD vector X is an eigenvector of a kD matrix A if

A.X

points in the same or the opposite direction as X.

In this case there is a number s (called corresponding eigenvalue) so that

A.X = s X.

Sometimes eigenvectors and eigenvalues involve complex numbers.

□G.1.a.i) Eigenvectors with positive eigenvalues

Here's a matrix A:

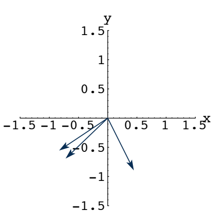
$$A = \begin{pmatrix} 0.9 & -0.3 \\ -0.4 & 1.3 \end{pmatrix};$$

```
MatrixForm[A]
```

$$\begin{pmatrix} 0.9 & -0.3 \\ -0.4 & 1.3 \end{pmatrix}$$

Here are its eigenvectors shown with a random unit vector:

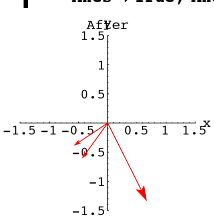
```
Clear[eigenvector];
{eigenvector[1], eigenvector[2]} = Eigenvectors[A];

s = Random[Real, {0, N[2 π]}];
randomvector = {Cos[s], Sin[s]};
ranger = 1.5;

before =
  Show[Arrow[eigenvector[1], Tail → {0, 0}, VectorColor → Indigo],
   Arrow[eigenvector[2], Tail → {0, 0}, VectorColor → Indigo],
   Arrow[randomvector, Tail → {0, 0}, VectorColor → Indigo],
   Axes → True, PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   AxesLabel → {"x", "y"}];
```



If you don't see three distinct clear vectors, then rerun.

Here's what you get when you hit all three plotted vectors by the matrix A:

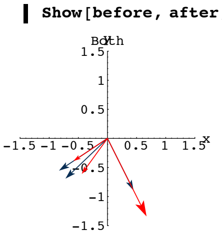
```
after = Show[Arrow[A.eigenvector[1], Tail → {0, 0}, VectorColor → Red],
   Arrow[A.eigenvector[2], Tail → {0, 0}, VectorColor → Red],
   Arrow[A.randomvector, Tail → {0, 0}, VectorColor → Red],
   PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   Axes → True, AxesLabel → {"x", "y"}, PlotLabel → "After"];
```



Grab both plots, align and animate.

Here are both plots:

```
Show[before, after, PlotLabel → "Both"];
```



Describe how this plot reveals which two of the original vectors are the eigenvectors of A.

Describe how this plot reveals that both eigenvalues of A are positive with one eigenvalue smaller than 1 and the other eigenvalue bigger than 1.

□G.1.a.ii) One eigenvector with a positive eigenvalue; one eigenvector with a negative eigenvalue

Here's a new matrix A:

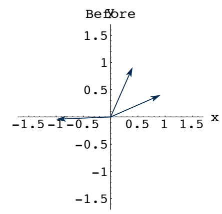
$$A = \begin{pmatrix} 1.1 & -0.7 \\ 0.7 & -0.8 \end{pmatrix};$$

```
MatrixForm[A]
```

$$\begin{pmatrix} 1.1 & -0.7 \\ 0.7 & -0.8 \end{pmatrix}$$

Here are its eigenvectors shown with a random unit vector:

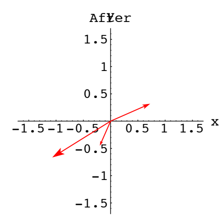
```
Clear[eigenvector];
{eigenvector[1], eigenvector[2]} = Eigenvectors[A];

s = Random[Real, {0, N[2 π]}];
randomvector = {Cos[s], Sin[s]};
ranger = 1.7;

before =
  Show[Arrow[eigenvector[1], Tail → {0, 0}, VectorColor → Indigo],
   Arrow[eigenvector[2], Tail → {0, 0}, VectorColor → Indigo],
   Arrow[randomvector, Tail → {0, 0}, VectorColor → Indigo],
   Axes → True, PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   AxesLabel → {"x", "y"}, PlotLabel → "Before"];
```



If you don't see three distinct clear vectors, then rerun.

Here's what you get when you multiply all three plotted vectors by the matrix A:

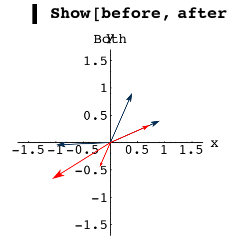
```
after = Show[Arrow[A.eigenvector[1], Tail → {0, 0}, VectorColor → Red],
   Arrow[A.eigenvector[2], Tail → {0, 0}, VectorColor → Red],
   Arrow[A.randomvector, Tail → {0, 0}, VectorColor → Red],
   PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   Axes → True, AxesLabel → {"x", "y"}, PlotLabel → "After"];
```



Grab both plots and animate.

Here are both plots:

```
Show[before, after, PlotLabel → "Both"];
```



Describe how this plot reveals which two of the original vectors are the eigenvectors of A.

Describe how this plot reveals that one of the eigenvalues is positive and that the other is negative.

□G.1.b.i) Pick out the eigenvectors

Here's a new matrix A:

$$A = \begin{pmatrix} 1.425 & -0.129904 \\ -0.129904 & 1.275 \end{pmatrix};$$

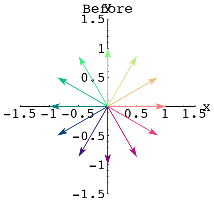
```
MatrixForm[A]
```

$$\begin{pmatrix} 1.425 & -0.129904 \\ -0.129904 & 1.275 \end{pmatrix}$$

Here are a bunch of color-coded unit vectors:

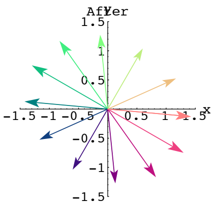
```
Clear[vectorcolor, t];
vectorcolor[t_] = RGBColor[0.5 (Cos[t] + 1), 0.5 (Sin[t] + 1), 0.5];

unitvector[t_] = {Cos[t], Sin[t]};

before = Show[Table[
```

```
Arrow[unitvector[t], Tail → {0, 0}, VectorColor → vectorcolor[t]],
  {t, 0, 2 π, π/6}], Axes → True, AxesLabel → {"x", "y"},
PlotRange → {{-1.5, 1.5}, {-1.5, 1.5}}, PlotLabel → "Before"];
```



Here are the same vectors after they have been multiplied by A.

```
after = Show[Table[Arrow[A.unitvector[t], Tail → {0, 0},
    VectorColor → vectorcolor[t]], {t, 0, 2 π, π/6}], Axes → True,
  AxesLabel → {"x", "y"}, PlotRange → {{-1.5, 1.5}, {-1.5, 1.5}},
  PlotLabel → "After"];
```

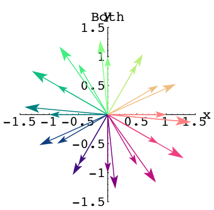


In this plot, A.unitvector[t] is the same color as unitvector[t] in the first plot.

Grab,align and animate slowly.

Here are both plots together:

```
Show[before, after, PlotLabel → "Both"];
```



In this plot, you see enough information to pick out the two eigenvectors of A and their negatives. Say how you made the identification.

G.1.b.ii) Rotation matrices can't have real eigenvectors

Here is a random number s with $0 < s < \frac{\pi}{2}$ together with the matrix A whose hits rotate everything about {0,0} by s counterclockwise radians :

```
s = Random[Real, {0.1, π/2 - 0.1}];
A = Transpose[{{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}}];
MatrixForm[A]
```

$$\begin{pmatrix} 0.696216 & -0.717832 \\ 0.717832 & 0.696216 \end{pmatrix}$$

Here's what happens when you diagonalize A:

```
Clear[eigenvector];
spanners = Eigenvectors[A];
SpannerMatrix = Transpose[spanners];
MatrixForm[SpannerMatrix]
```

$$\begin{pmatrix} 0.707107 & 0.707107 \\ -0.707107\, i & 0.707107\, i \end{pmatrix}$$

The diagonal matrix comes from the eigenvalues of A:

```
diagonalmatrix = DiagonalMatrix[Eigenvalues[A]];
MatrixForm[diagonalmatrix]
```

$$\begin{pmatrix} 0.696216 + 0.717832\, i & 0 \\ 0 & 0.696216 - 0.717832\, i \end{pmatrix}$$

Here's a look at

SpannerMatrix.diagonalmatrix.SpannerMatrix^{-1}:

```
MatrixForm[SpannerMatrix.diagonalmatrix.Inverse[SpannerMatrix]]
```

$$\begin{pmatrix} 0.696216 & -0.717832 \\ 0.717832 & 0.696216 \end{pmatrix}$$

Here's a look at A:

```
MatrixForm[A]
```

$$\begin{pmatrix} 0.696216 & -0.717832 \\ 0.717832 & 0.696216 \end{pmatrix}$$

So:

A = SpannerMatrix.diagonalmatrix.SpannerMatrix^{-1}

and A is diagonalizable.

Notice that complex numbers were needed to do the diagonalization.

Explain why you could have been predicted this in advance.

G.2) Diagonalizations, matrix exponentials and matrix powers

G.2.a.i) Diagonalizing a 2D matrix

Here's a 2D matrix A:

```
A = ( 2.1972  -0.9859
      0.7394  -1.7804 );
MatrixForm[A]
```

$$\begin{pmatrix} 2.1972 & -0.9859 \\ 0.7394 & -1.7804 \end{pmatrix}$$

Here are the eigenvalues of A:

```
Clear[eigenvector, eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Eigenvalues[A]
```

{2.0046, -1.5878}

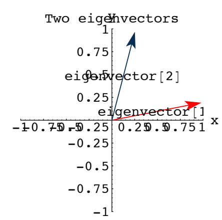
And the corresponding unit eigenvectors of A:

```
{eigenvector[1], eigenvector[2]} = Eigenvectors[A]
```

{{0.981449, 0.191726}, {0.252065, 0.96771}}

See them:

```
eigenplot =
   {Arrow[eigenvector[1], Tail -> {0, 0}, VectorColor -> Red],
    Arrow[eigenvector[2], Tail -> {0, 0}, VectorColor -> Indigo]};

labels =
  {Graphics[{Black, Text["eigenvector[1]", 0.5 eigenvector[1]]}],
   Graphics[{Black,
    Text["eigenvector[2]", 0.5 eigenvector[2]]}]};
ranger = 1;

Show[eigenplot, labels, Axes -> True, AxesLabel -> {"x", "y"},
      PlotRange -> {{-ranger, ranger}, {-ranger, ranger}},
      PlotLabel -> "Two eigenvectors"];
```



The eigenvectors span all of 2D and this tells you that A is diagonalizable.

Diagonalize A by coming up with SpannerMatrix and diagonalmatrix so that

SpannerMatrix.diagonalmatrix.SpannerMatrix^{-1}

duplicates A.

G.2.a.ii) Building the matrix exponential $E^{A t}$

Stay with the same matrix A as in part i):

```
A = ( 2.1972  -0.9859
      0.7394  -1.7804 );
MatrixForm[A]
```

$$\begin{pmatrix} 2.1972 & -0.9859 \\ 0.7394 & -1.7804 \end{pmatrix}$$

In part i), you diagonalized A by coming up with SpannerMatrix and diagonalmatrix so that

A = SpannerMatrix.diagonalmatrix.SpannerMatrix^{-1}.

Stay with the same SpannerMatrix, but come up with newdiagonalmatrix so that

$E^{A t}$ = SpannerMatrix.newdiagonalmatrix.SpannerMatrix^{-1}.

Check yourself by activating:

```
Clear[t];
MatrixForm[Simplify[MatrixExp[A t]]]
```

$$\begin{pmatrix} -0.0536118\, e^{-1.5878\, t} + 1.05361\, e^{2.0046\, t} & 0.27444\, e^{-1.5878\, t} - 0.27444\, e^{2.0046\, t} \\ -0.205823\, e^{-1.5878\, t} + 0.205823\, e^{2.0046\, t} & 1.05361\, e^{-1.5878\, t} - 0.0536118\, e^{2.0046\, t} \end{pmatrix}$$

G.2.b.i) Reading the eigenvalues of A from the formula for $E^{A t}$

Here's new 2D matrix A:

```
A = ( -1.48  -1.43
      -2.55   3.82 );
MatrixForm[A]
```

$$\begin{pmatrix} -1.48 & -1.43 \\ -2.55 & 3.82 \end{pmatrix}$$

Look at this calculation of $E^{A t}$:

```
Clear[t];
MatrixForm[Simplify[MatrixExp[A t]]]
```

```
( 0.905652 e^(-2.09634 t) + 0.0943476 e^(4.43634 t)   0.218899 e^(-2.09634 t) - 0.218899 e^(4.43
( 0.390345 e^(-2.09634 t) - 0.390345 e^(4.43634 t)   0.0943476 e^(-2.09634 t) + 0.905652 e^(4.43
```

If you are clued in, you can read the eigenvalues of A from the output directly above.
Type them here:

eigenvalue[1] = ...

eigenvalue[2] = ...

□G.2.b.ii) Another one

Here's new 2D matrix A:

```
A = ( -1.48  -1.43
      2.55   0.82 );
MatrixForm[A]
```

$$\begin{pmatrix} -1.48 & -1.43 \\ 2.55 & 0.82 \end{pmatrix}$$

Look at this calculation of $E^{A t}$:

```
MatrixForm[Simplify[ComplexExpand[MatrixExp[A t]]]]
```

```
( e^(-0.33 t) (1. Cos[1.52447 t] - 0.754362 Sin[1.52447 t])          -0.938033
(            1.67272 e^(-0.33 t) Sin[1.52447 t]             e^(-0.33 t) (1. Cos[1.524
```

If you are clued in, you can read the eigenvalues p + I q and p − I q of A from the output directly above.
Type them here:

eigenvalue[1] = ...

eigenvalue[2] = ...

□G.2.c) The square root of a matrix

Here's new 2D matrix A:

```
A = ( 2.48  1.03
      1.13  0.82 );
MatrixForm[A]
```

$$\begin{pmatrix} 2.48 & 1.03 \\ 1.13 & 0.82 \end{pmatrix}$$

Apply the square root function to this matrix to make a new matrix B with

B.B = A

```
Eigenvalues[A]
```

{3.01118, 0.288824}

□G.2.d) The derivative of $E^{A t}$ with respect to t

Here's new 2D matrix A:

```
A = ( Random[Real, {-3, 3}]  Random[Real, {-3, 3}]
      Random[Real, {-3, 3}]  Random[Real, {-3, 3}] );
MatrixForm[A]
```

$$\begin{pmatrix} -0.0650792 & -2.19121 \\ 1.73617 & 0.735183 \end{pmatrix}$$

Look at this calculation of D[$E^{A t}$,t] (the derivative of $E^{A t}$ with respect to t):

```
MatrixForm[Simplify[ComplexExpand[D[MatrixExp[A t], t]]]]
```

```
( -0.0650792 e^(0.335052 t) Cos[1.90898 t] - 1.97921 e^(0.335052 t) Sin[1.90898 t]   -2.
(  1.73617 e^(0.335052 t) Cos[1.90898 t] + 0.304722 e^(0.335052 t) Sin[1.90898 t]
```

Look at this calculation of $A.E^{A t}$:

```
MatrixForm[Simplify[ComplexExpand[A.MatrixExp[A t]]]]
```

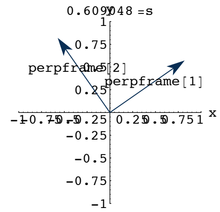
```
( -0.0650792 e^(0.335052 t) Cos[1.90898 t] - 1.97921 e^(0.335052 t) Sin[1.90898 t]   -2.
(  1.73617 e^(0.335052 t) Cos[1.90898 t] + 0.304722 e^(0.335052 t) Sin[1.90898 t]    0.
```

Rerun all three cells several times and then say why you are not surprised with what you see.

□G.2.e.i) Matrix powers A^k for rotation matrices A

Here is a random right hand perpendicular frame:

```
s = Random[Real, {0.3, 0.7}];
ranger = 1;

Clear[perpframe];
{perpframe[1], perpframe[2]} =
  {{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}};

frameplot = {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]]]};

before =
  Show[frameplot, PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
    Axes → True, AxesLabel → {"x", "y"}, PlotLabel → s "=s"];
```



When you want to rotate everything by s counterclockwise radians you hit with this matrix:

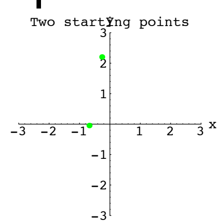
```
A = Transpose[{perpframe[1], perpframe[2]}];
MatrixForm[A]
```

$$\begin{pmatrix} 0.820193 & -0.572087 \\ 0.572087 & 0.820193 \end{pmatrix}$$

See what happens when you take two random starting points and hit them with increasing matrix powers of A:

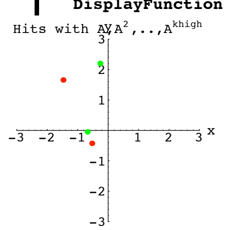
```
Clear[iterationplot, interationplotter, scaler, point,
  pointcolor, starter, khigh, k, j, iterationpoints];
point[k_, starter_] := MatrixPower[A, k].starter;
pointcolor[k_, khigh_] := RGBColor[
  Sin[(Pi/2) k/(khigh + 0.1)], Cos[(Pi/2) k/(khigh + 0.1)], 0]

iterationpoints[starter_, khigh_] :=
Table[Graphics[ {pointcolor[k, khigh], PointSize[0.03],
    Point[MatrixPower[A, k].starter]}], {k, 0, khigh}];

iterationplotter[khigh_] :=
  Show[Table[iterationpoints[starter[j], khigh], {j, 1, 2}],
   Axes -> True, AxesLabel -> {"x", "y"},
   PlotLabel -> "Hits with A,A^2,..,A^khigh",
       PlotRange -> {{-3, 3}, {-3, 3}},
   AspectRatio -> Automatic, DisplayFunction -> Identity];


ss = Random[Real, {0, 2 Pi}];
sss = Random[Real, {0, 2 Pi}];
starter[1] = Random[Real, {0.5, 1}] {Cos[ss], Sin[ss]};
starter[2] =  Random[Real, {1.5, 3}] {Cos[sss], Sin[sss]};


khigh = 0;
```

```
Show[iterationplotter[khigh], PlotLabel -> "Two starting points",
  DisplayFunction -> $DisplayFunction];
```



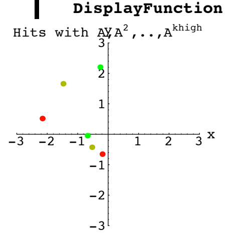
Here are the same two starter points together with the points you get when you hit each of the original starting points with A:

```
khigh = 1;
Show[iterationplotter[khigh],
  DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with A and A.A:

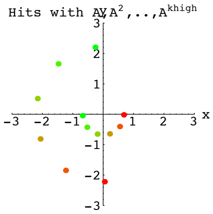
```
khigh = 2;
Show[iterationplotter[khigh],
  DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with
A, A.A = A^2, A.A.A = A^3, A.A.A.A = A^4, and A.A.A.A.A = A^5:

All of these powers are done with matrix multiplication.

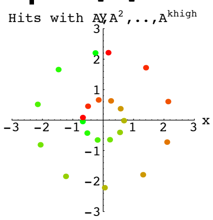
```
khigh = 5;
Show[iterationplotter[khigh],
  DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with A, A.A = A^2, A.A.A = A^3, . . ., A^9, and A^{10}:

All of these powers are done with matrix multiplication.

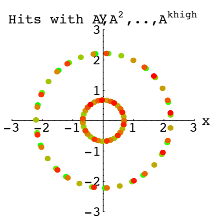
```
khigh = 10;
Show[iterationplotter[khigh],
  DisplayFunction -> $DisplayFunction];
```



Here are the same four starter points together with the points you get when you hit each of the original starting points with A, A.A = A^2, A.A.A = A^3, . . ., A^{49}, and A^{50}:

All of these powers are done with matrix multiplication.

```
khigh = 50;
Show[iterationplotter[khigh],
  DisplayFunction -> $DisplayFunction];
```

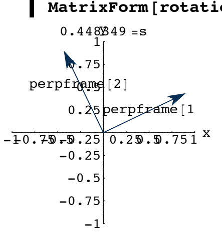


Describe what you see and explain why you see it.
Is every integral power of A (such as A^4) also a rotation matrix?

□G.2.e.ii) If A is rotation matrix, does $E^{A\,t}$ oscillate on closed curves as t advances

Here's a new right hand perpendicular frame and the rotation matrix based on it:

```
s = Random[Real, {0.3, 0.7}];
ranger = 1;

Clear[perpframe];
{perpframe[1], perpframe[2]} =
  {{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}};

frameplot = {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]]]};

before =
  Show[frameplot, PlotRange → {{-ranger, ranger}, {-ranger, ranger}},
   Axes → True, AxesLabel → {"x", "y"}, PlotLabel → s "=s"];
rotationmatrix = Transpose[{perpframe[1], perpframe[2]}];
MatrixForm[rotationmatrix]
```



$$\begin{pmatrix} 0.901164 & -0.433479 \\ 0.433479 & 0.901164 \end{pmatrix}$$

When you want to use this perpendicular frame to rotate, you hit with this matrix:

```
A = rotationmatrix;
MatrixForm[A]
```

$$\begin{pmatrix} 0.901164 & -0.433479 \\ 0.433479 & 0.901164 \end{pmatrix}$$

Look at the eigenvalues of A:

```
Eigenvalues[A]
```

{0.901164 + 0.433479 i, 0.901164 - 0.433479 i}

Use what you see to answer this question:
If A is a rotation matrix, does $E^{A\,t}$ oscillate on closed curves as t advances?

□G.2.e.iii) For which rotation matrices A does $E^{A\,t}$ oscillate on closed curves as t advances

Here is a cleared rotation matrix:

```
Clear[perpframe, s];
{perpframe[1], perpframe[2]} =
{{Cos[s], Sin[s]}, {Cos[s + π/2], Sin[s + π/2]}};
A = Transpose[{perpframe[1], perpframe[2]}];

MatrixForm[A]
```

$$\begin{pmatrix} \mathrm{Cos}[s] & -\mathrm{Sin}[s] \\ \mathrm{Sin}[s] & \mathrm{Cos}[s] \end{pmatrix}$$

Here is *Mathematica*'s calculation of the eigenvalues of A:

```
Eigenvalues[A]
```

{Cos[s] - i Sin[s], Cos[s] + i Sin[s]}

The question is this:
What values of s (with $0 \le s < 2\pi$) guarantee that $E^{A\,t}$ oscillates on closed curves as t advances?
Show off your answer with some plots.

Click on the right for some code to play with.

Fill in your pick for s and run this code.

```
s = ?? ?? ?? ;

Clear[trajectoryplot, eigenplot,
    scaler, eigenvector, starter, thigh, k, j];

trajectoryplot[thigh_, starter_] :=
   ParametricPlot[MatrixExp[A t].starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},

    DisplayFunction -> Identity];

pointer[thigh_, starter_] :=
   ArrowHead[MatrixExp[A thigh].starter,
    (D[MatrixExp[A t], t] /. t -> thigh).starter,
               HeadSize -> 0.6, VectorColor -> Black,
    Aperture -> 0.4];

starterplots = Table[Graphics[
     {NavyBlue, PointSize[0.03], Point[starter[j]]}], {j, 1, 6}];

starter[1] = {Random[Real, {4, 10}], Random[Real, {4, 10}]};
starter[2] = {Random[Real, {-10, -4}], Random[Real, {-10, -4}]};
starter[3] = {Random[Real, {-10, -4}], Random[Real, {4, 10}]};
starter[4] = {Random[Real, {4, 10}], Random[Real, {-10, -4}]};
starter[5] = {Random[Real, {-10, -4}], 0};
starter[6] = {Random[Real, {4, 10}], 0};

thigh = 0.01;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 6}] ,
   starterplots, PlotLabel -> "Starting points",
   DisplayFunction -> $DisplayFunction];
```

```
Information::ssym : «1» is not a symbol or a string.
Information::ssym : «1» is not a symbol or a string.
```

```
Information[
 Information[Information[;, LongForm → True], LongForm → True]
  Clear[trajectoryplot, eigenplot, scaler, eigenvector,
   starter, thigh, k, j]; trajectoryplot[thigh_, starter_] :=
  ParametricPlot[MatrixExp[A t].starter, {t, 0, thigh}, PlotStyle →
    {{Thickness[0.01], CadmiumOrange}}, DisplayFunction → Identity];
 pointer[thigh_, starter_] := ArrowHead[MatrixExp[A thigh].starter,
   (∂t MatrixExp[A t] /. t → thigh).starter,
   HeadSize → 0.6, VectorColor → Black, Aperture → 0.4];
 starterplots = Table[- Graphics -, {j, 1, 6}];
 starter[1] = {Random[Real, {4, 10}], Random[Real, {4, 10}]};
 starter[2] = {Random[Real, {-10, -4}], Random[Real, {-10, -4}]};
 starter[3] = {Random[Real, {-10, -4}], Random[Real, {4, 10}]};
 starter[4] = {Random[Real, {4, 10}], Random[Real, {-10, -4}]};
 starter[5] = {Random[Real, {-10, -4}], 0};
 starter[6] = {Random[Real, {4, 10}], 0}; thigh = 0.01;
 Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 6}],
  starterplots, PlotLabel → Starting points,
  DisplayFunction → $DisplayFunction];, LongForm → True]
```

Here is how the curves
$E^{A t}$.starter[1], $E^{A t}$.starter[2], $E^{A t}$.starter[3],
$E^{A t}$.starter[4], $E^{A t}$.starter[5], $E^{A t}$.starter[6]
plot out as t advances from 0 to 8

```
thigh = 2;

Show[Table[trajectoryplot[thigh, starter[k]], {k, 1, 6}] ,
    Table[pointer[thigh, starter[k]], {k, 1, 6}],
  starterplots, PlotRange -> All,
  PlotLabel -> "Plots of E^A\\\t.starter[k] for 0 < t" < thigh,
  DisplayFunction -> $DisplayFunction];
```

Show::gcomb : An error was encountered in combining the graphics objects in Show[≪1≫].

G.3) Using eigenvalues to analyze behavior of solutions of systems of linear differential equations

If you haven't had a course in differential equations, don't be scared off by this problem.
If you can handle the matrix exponential, then you can handle systems of linear differential equations.
Just check them out in the Tutorials.

G.3.a.i) A system of linear diffeq's

To get a 2D system of linear differential equations, you go with a 2D matrix coefficient matrix A and put
{x′[t], y′[t]} = A.{x[t], y[t]}.
Here's one:

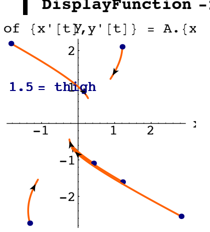
```
A = ( -1.4  0.8
      0.5  -0.7 );
MatrixForm[A]
Clear[x, y, t];
linearsystem = ({x'[t], y'[t]} == A.{x[t], y[t]});
ColumnForm[Thread[linearsystem]]
```

$$\begin{pmatrix} -1.4 & 0.8 \\ 0.5 & -0.7 \end{pmatrix}$$

$x'[t] == -1.4\, x[t] + 0.8\, y[t]$
$y'[t] == 0.5\, x[t] - 0.7\, y[t]$

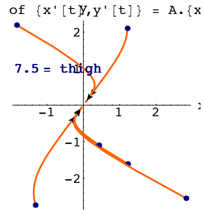
Look at this plot of six trajectories coming from solutions
{x[t], y[t]} = $E^{A t}$.starter
of this system of differential equations starting at six random points as t advances from 0 to 1.5:

```
Clear[trajectoryplot, starterplot, starter, thigh, matrixexpA];
matrixexpA[t_] = MatrixExp[A t];

    trajectoryplot[starter_, thigh_] :=
     {ParametricPlot[ matrixexpA[t] .starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
              PlotLabel ->
     "Solutions of {x'[t],y'[t]} = A.{x[t],y[t]}",
              PlotRange -> All, AxesLabel -> {"x", "y"},
    AxesOrigin -> {0, 0}, DisplayFunction -> Identity],

        ArrowHead[matrixexpA[0.85 thigh] .starter,
    matrixexpA '[0.85 thigh] .starter,
                HeadSize -> 0.2, VectorColor -> Black,
    Aperture -> 0.4],

  Graphics[{NavyBlue, PointSize[0.03], Point[starter]}],
   Graphics[{NavyBlue, Text[thigh "= thigh", {-0.7, 1}]}]};

starter[1] = {Random[Real, {1, 3}], Random[Real, {1, 3}]};
starter[2] = {Random[Real, {-3, -1}], Random[Real, {-3, -1}]};
starter[3] = {Random[Real, {-3, -1}], Random[Real, {1, 3}]};
starter[4] = {Random[Real, {1, 3}], Random[Real, {-3, -1}]};
starter[5] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
starter[6] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};

thigh = 1.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 7.5:

```
thigh = 7.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here are the eigenvalues of the coefficient matrix A:

```
Eigenvalues[A]
```

{-1.77284, -0.327158}

Use the eigenvalue information to explain why the plots come out the way they do.
How will trajectories coming from other solutions look?
Are they all attracted to {0,0} as t gets large?

G.3.a.ii) Swirling around {0,0}

To get a 2D system of linear differential equations, you go with a 2D matrix coefficient matrix A and put
{x′[t], y′[t]} = A.{x[t], y[t]}.
Here's another one:

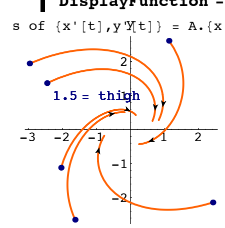
```
A = ( -1.4  1.6
      -1.5  -0.7 );
MatrixForm[A]
Clear[x, y, t];
linearsystem = ({x'[t], y'[t]} == A.{x[t], y[t]});
ColumnForm[Thread[linearsystem]]
```

$$\begin{pmatrix} -1.4 & 1.6 \\ -1.5 & -0.7 \end{pmatrix}$$

$x'[t] == -1.4\, x[t] + 1.6\, y[t]$
$y'[t] == -1.5\, x[t] - 0.7\, y[t]$

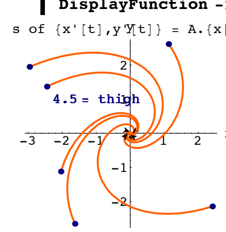
Look at this plot of six trajectories coming from solutions
{x[t], y[t]} = $E^{A t}$.starter
starting at six random points as t advances from 0 to 1.5:

```
Clear[trajectoryplot, starterplot, starter, thigh, matrixexpA];
matrixexpA[t_] = MatrixExp[A t];
    trajectoryplot[starter_, thigh_] :=
     {ParametricPlot[ matrixexpA[t] .starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
              PlotLabel ->
     "Solutions of {x'[t],y'[t]} = A.{x[t],y[t]}",
              PlotRange -> All, AxesLabel -> {"x", "y"},
    AxesOrigin -> {0, 0}, DisplayFunction -> Identity],

        ArrowHead[matrixexpA[0.85 thigh] .starter,
    matrixexpA '[0.85 thigh] .starter,
                HeadSize -> 0.2, VectorColor -> Black,
    Aperture -> 0.4],

  Graphics[{NavyBlue, PointSize[0.03], Point[starter]}],
   Graphics[{NavyBlue, Text[thigh "= thigh", {-1, 1}]}]};

starter[1] = {Random[Real, {1, 3}], Random[Real, {1, 3}]};
starter[2] = {Random[Real, {-3, -1}], Random[Real, {-3, -1}]};
starter[3] = {Random[Real, {-3, -1}], Random[Real, {1, 3}]};
starter[4] = {Random[Real, {1, 3}], Random[Real, {-3, -1}]};
starter[5] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
starter[6] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};

thigh = 1.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 4.5:

```
thigh = 4.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here are the eigenvalues of the coefficient matrix A:

```
Eigenvalues[A]
```

{-1.05 + 1.50914 i, -1.05 - 1.50914 i}

Use the eigenvalue information to explain why the plots come out the way they do.
How will trajectories coming from other solutions look?
Are they all attracted to {0,0} as t gets large?

☐ G.3.a.iii) Moving away from {0,0}

To get a 2D system of linear differential equations, you go with a 2D matrix coefficient matrix A and put

{x′[t], y′[t]} = A.{x[t], y[t]}.

Here's another yet another one:

```
A = (1.4  -0.4
     -0.5  0.6);
MatrixForm[A]
Clear[x, y, t];
linearsystem = ({x'[t], y'[t]} == A.{x[t], y[t]});
ColumnForm[Thread[linearsystem]]
```

$$\begin{pmatrix} 1.4 & -0.4 \\ -0.5 & 0.6 \end{pmatrix}$$

```
x′[t] == 1.4 x[t] - 0.4 y[t]
y′[t] == -0.5 x[t] + 0.6 y[t]
```

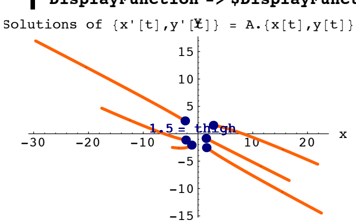
Look at this plot of six trajectories coming from solutions

{x[t], y[t]} = $E^{A t}$.starter

starting at six random points as t advances from 0 to 1.5:

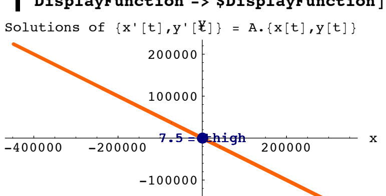
```
Clear[trajectoryplot, starterplot, starter, thigh, matrixexpA];
matrixexpA[t_] = MatrixExp[A t];
    trajectoryplot[starter_, thigh_] :=
     {ParametricPlot[ matrixexpA[t].starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
            PlotLabel ->
     "Solutions of {x'[t],y'[t]} = A.{x[t],y[t]}",
            PlotRange -> All, AxesLabel -> {"x", "y"},
    AxesOrigin -> {0, 0}, DisplayFunction -> Identity],

        ArrowHead[matrixexpA[0.85 thigh].starter,
    matrixexpA'[0.85 thigh].starter,
              HeadSize -> 0.2, VectorColor -> Black,
    Aperture -> 0.4],

  Graphics[{NavyBlue, PointSize[0.03], Point[starter]}],
   Graphics[{NavyBlue, Text[thigh "= thigh", {-1, 1}]}]};

starter[1] = {Random[Real, {1, 3}], Random[Real, {1, 3}]};
starter[2] = {Random[Real, {-3, -1}], Random[Real, {-3, -1}]};
starter[3] = {Random[Real, {-3, -1}], Random[Real, {1, 3}]};
starter[4] = {Random[Real, {1, 3}], Random[Real, {-3, -1}]};
```

```
starter[5] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
starter[6] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};

thigh = 1.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 7.5:

```
thigh = 7.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here are the eigenvalues of the coefficient matrix A:

```
Eigenvalues[A]
```

```
{1.6, 0.4}
```

Use this information to help to identify a unit vector pointing in the same direction as the line you see above.

☐ G.3.a.iv) Swirling away from {0,0}

To get a 2D system of linear differential equations, you go with a 2D matrix coefficient matrix A and put

{x′[t], y′[t]} = A.{x[t], y[t]}.

Here's another one:

```
A = (0.4  -0.8
     0.5   0.3);
MatrixForm[A]
Clear[x, y, t];
linearsystem = ({x'[t], y'[t]} == A.{x[t], y[t]});
ColumnForm[Thread[linearsystem]]
```

$$\begin{pmatrix} 0.4 & -0.8 \\ 0.5 & 0.3 \end{pmatrix}$$

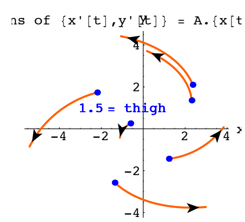
```
x′[t] == 0.4 x[t] - 0.8 y[t]
y′[t] == 0.5 x[t] + 0.3 y[t]
```

Look at this plot of six trajectories coming from solutions

{x[t], y[t]} = $E^{A t}$.starter

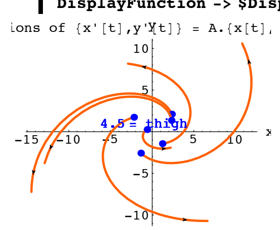
starting at six random points as t advances from 0 to 1.5:

```
Clear[trajectoryplot, starterplot, starter, thigh, matrixexpA];
matrixexpA[t_] = MatrixExp[A t];
    trajectoryplot[starter_, thigh_] :=
     {ParametricPlot[ matrixexpA[t].starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
            PlotLabel ->
     "Solutions of {x'[t],y'[t]} = A.{x[t],y[t]}",
            PlotRange -> All, AxesLabel -> {"x", "y"},
    AxesOrigin -> {0, 0}, DisplayFunction -> Identity],

        ArrowHead[matrixexpA[0.95 thigh].starter,
    matrixexpA'[0.95 thigh].starter,
              HeadSize -> 0.6, VectorColor -> Black,
    Aperture -> 0.4],

  Graphics[{Blue, PointSize[0.03], Point[starter]}],
   Graphics[{Blue, Text[thigh "= thigh", {-1, 1}]}]};

starter[1] = {Random[Real, {1, 3}], Random[Real, {1, 3}]};
starter[2] = {Random[Real, {-3, -1}], Random[Real, {-3, -1}]};
starter[3] = {Random[Real, {-3, -1}], Random[Real, {1, 3}]};
starter[4] = {Random[Real, {1, 3}], Random[Real, {-3, -1}]};
starter[5] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
starter[6] = {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};

thigh = 1.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 4.5:

```
thigh = 4.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
DisplayFunction -> $DisplayFunction];
```



Here are the eigenvalues of the coefficient matrix A:

```
Eigenvalues[A]
```

```
{0.35 + 0.630476 i, 0.35 - 0.630476 i}
```

Use the eigenvalue information to explain why the plots come out the way they do.
How will trajectories coming from other solutions look?

☐ G.3.a.v) Merging onto a line

To get a 2D system of linear differential equations, you go with a 2D matrix coefficient matrix A and put

{x′[t], y′[t]} = A.{x[t], y[t]}.

Here's another one:

```
A = (1.4   0.8
     0.5  -0.9);
MatrixForm[A]
Clear[x, y, t];
linearsystem = ({x'[t], y'[t]} == A.{x[t], y[t]});
ColumnForm[Thread[linearsystem]]
```

$$\begin{pmatrix} 1.4 & 0.8 \\ 0.5 & -0.9 \end{pmatrix}$$

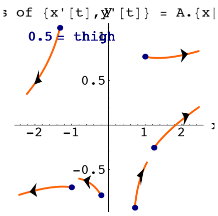
```
x'[t] == 1.4 x[t] + 0.8 y[t]
y'[t] == 0.5 x[t] - 0.9 y[t]
```

Look at this plot of six trajectories coming from solutions

$\{x[t], y[t]\} = E^{A\,t}.\text{starter}$

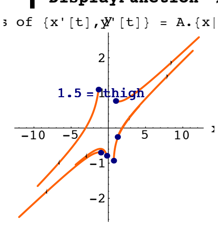
starting at six random points as t advances from 0 to 1.5:

```
Clear[trajectoryplot, starterplot, starter, thigh, matrixexpA];
matrixexpA[t_] = MatrixExp[A t];
    trajectoryplot[starter_, thigh_] :=
     {ParametricPlot[ matrixexpA[t] .starter, {t, 0, thigh},
    PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
             PlotLabel ->
     "Solutions of {x'[t],y'[t]} = A.{x[t],y[t]}",
             PlotRange -> All, AxesLabel -> {"x", "y"},
    AxesOrigin -> {0, 0}, DisplayFunction -> Identity],

        ArrowHead[matrixexpA[0.85 thigh] .starter,
    matrixexpA'[0.85 thigh] .starter,
              HeadSize -> 0.2, VectorColor -> Black,
    Aperture -> 0.4],

  Graphics[{NavyBlue, PointSize[0.03], Point[starter]}],
   Graphics[{NavyBlue, Text[thigh "= thigh", {-1, 1}]}]}];

starter[1] = 0.5 {Random[Real, {1, 3}], Random[Real, {1, 3}]};
starter[2] = 0.5 {Random[Real, {-3, -1}], Random[Real, {-3, -1}]};
starter[3] = 0.5 {Random[Real, {-3, -1}], Random[Real, {1, 3}]};
starter[4] = 0.5 {Random[Real, {1, 3}], Random[Real, {-3, -1}]};
starter[5] = 0.5 {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};
starter[6] = 0.5 {Random[Real, {-3, 3}], Random[Real, {-3, 3}]};

thigh = 0.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
  AspectRatio -> 1,
DisplayFunction -> $DisplayFunction];
```



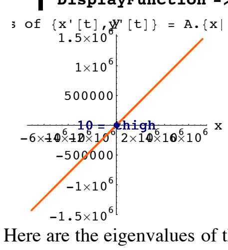
Here's what happens as t advances from 0 to 1.5:

```
thigh = 1.5;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
  AspectRatio -> 1,
DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 10:

```
thigh = 10;
Show[Table[trajectoryplot[starter[k], thigh], {k, 1, 6}],
  AspectRatio -> 1,
DisplayFunction -> $DisplayFunction];
```



Here are the eigenvalues of the coefficient matrix A:

```
Eigenvalues[A]
```

{1.56244, -1.06244}

Use this information to help to identify a unit vector pointing in the same direction as the line you see above.

G.4) The determinant of any diagonalizable matrix is the product of its eigenvalues

□G.4.a) The determinant of any diagonalizable matrix is the product of its eigenvalues

One day, one of your friends comes by, holds out a dusty book and says, "This book I checked out of the library says that the determinant of any diagonalizable matrix is the product of its eigenvalues."
You decide to look into this and type:

```
dim = Random[Integer, {2, 8}];
A = Table[Random[Real, {-4, 4}], {i, 1, dim}, {j, 1, dim}];
MatrixForm[A]
```

2.01379	1.48114	-3.28559	-2.39423	-1.23976	-1.45989	3.7937
-0.674636	3.55977	2.19411	-3.29294	1.51269	-0.429963	0.02786
0.730059	3.11436	-1.87438	3.26839	-1.02888	-1.44459	2.655
1.37421	-0.338023	3.23452	-3.35839	3.89307	-1.05244	1.6287
1.88136	1.35296	-0.846204	-1.69661	2.32159	3.15885	-1.553
0.790706	-1.24844	-0.869016	1.71668	1.67635	-3.37406	-0.1374
-1.25445	-0.879064	-2.02946	2.48838	3.08358	-0.113589	-2.671

Then you calculate the product of the eigenvalues of A:

```
product = Times @@ Eigenvalues[A]
```

28457.2

And then you calculate the determinant of A:

```
determinant = Det[A]
```

28457.2

Bingo.
You try another:

```
dim = Random[Integer, {2, 8}];
A = Table[Random[Real, {-4, 4}], {i, 1, dim}, {j, 1, dim}];
MatrixForm[A]
```

2.59531	0.136011	2.25766	-0.552434	-2.75765	-3.01779	-0.0457
-1.9165	2.53548	3.16356	-1.62558	2.95252	-3.18119	-2.512
-0.910076	2.07325	2.36627	3.77794	0.601541	2.98968	-1.520
2.00623	-1.14633	0.222206	-0.998555	0.763876	-2.12855	-3.732
-1.31963	-0.664028	-2.89561	-0.498942	-0.272144	-1.48283	3.6171
-3.36207	0.443913	-2.7491	-2.02536	0.0363901	1.45424	2.7710
2.03016	-1.39943	-1.45116	-3.47582	-2.73372	-3.27088	-1.719
2.58591	1.39314	-2.82349	3.14765	-1.14195	-1.12402	-2.440

```
product = Times @@ Eigenvalues[A]
```

-45754.

```
determinant = Det[A]
```

-45754.

You say: "Let me see that book."
In the book it says, "That the determinant of any diagonalizable matrix is the product of its eigenvalues is a quick consequence of these facts:

- Fact 1: If A,B and C are square matrices each with k rows and k columns, then
  Det[A.B.C] = Det[A] Det[B] Det[C]
- Fact 2: The determinant of any diagonal matrix is the product of its diagonal entries."

You say: "Taking these facts for granted, I see how to use these two facts to explain why the determinant of any diagonalizable matrix is the product of its eigenvalues."
Your friend says, "Explain it to me."
Do it.

G.5) Continuous dynamical systems: Physical and Chemical Sciences

Analyzing the effect of resistance in simple parallel electrical circuits.

Chemical reactions

□G.5.a.i) Analyzing the effect of resistance in simple parallel electrical circuits

You don't have to understand the electrical jargon to be able to do this problem.

You are fortunate enough to have your own computer set-up right in your dorm room. One night, Brian, the EE student who lives across the hall comes in and says, "Part of my EE 250 homework for tomorrow is to analyze the effect of varying the size of the resistance on the current in a simple parallel electrical circuit. I don't have a clue about where to start."

At first, you think, "I'm scared because I'm a life science major and I don't know diddly squat about electrical circuits." But you don't let on. Instead, you say, "Let me see that EE 250 textbook." It turns out to be thicker and heavier than you had expected. In the book you see that in a simple parallel electrical circuit, the main measurements are

x[t] = voltage drop across the capacitor and
y[t] = current through the inductor

Reading on, you spot that these measurements are related through the linear system

$$x'[t] = \frac{y[t]}{L}$$
$$y'[t] = -\frac{x[t]}{c} - \frac{y[t]}{c\,r}$$

where L, c, r are given positive numbers with

L = inductance
c = capacitance, and
r = resistance.

You say, "Brian I don't even know what those words mean, but I think I can help you."
You say that the coefficient matrix for this linear system is:

```
Clear[A, L, r, c];
A[L_, r_, c_] = {{0, 1/L}, {-1/c, -1/(c r)}};
MatrixForm[A[L, r, c]]
```

$$\begin{pmatrix} 0 & \frac{1}{L} \\ -\frac{1}{c} & -\frac{1}{c\,r} \end{pmatrix}$$

Now you ask Brian, "Is it OK to go with c = 0.8, L = 5.0?
Eagerly, he says, "Sure!"

You enter c = 0.8, L = 5.0 and then look at the eigenvalues of A[L, r, c] for r running from 0.1 to 3.7

```
c = 0.8;
L = 5.0;
Clear[r];

ColumnForm[
 Table[{r "= r", (Eigenvalues[A[L, r, c]])}, {r, 0.1, 3.7, 0.4}]]
```

```
{0.1 = r, {-12.48, -0.0200321}}
{0.5 = r, {-2.39564, -0.104356}}
{0.9 = r, {-1.17637, -0.212518}}
{1.3 = r, {-0.480769 + 0.137335 i, -0.480769 - 0.137335 i}}
{1.7 = r, {-0.367647 + 0.338874 i, -0.367647 - 0.338874 i}}
{2.1 = r, {-0.297619 + 0.401775 i, -0.297619 - 0.401775 i}}
{2.5 = r, {-0.25 + 0.433013 i, -0.25 - 0.433013 i}}
{2.9 = r, {-0.215517 + 0.451168 i, -0.215517 - 0.451168 i}}
{3.3 = r, {-0.189394 + 0.462742 i, -0.189394 - 0.462742 i}}
{3.7 = r, {-0.168919 + 0.470602 i, -0.168919 - 0.470602 i}}
```

You say that the current function y[t] will plot out a lot differently for the smaller positive resistance measurements r than for larger resistance measurements.
What is it about the output immediately above that gave you this idea?
What do you mean?
How are the shapes of the plots related to the eigenvalues above?

❑ G.5.a.ii) When the resistance measurement r is very big

Now you ask Brian whether he is interested in what happens when the resistance measurement r is very big? He replies," I never thought about that. What do you think?" You say, "Let's see."

```
c = 0.8;
L = 5.0;
Clear[r];
Table[{r, Eigenvalues[A[L, r, c]]}, {r, 10, 510, 50}]
```

```
{{10, {-0.0625 + 0.496078 i, -0.0625 - 0.496078 i}},
 {60, {-0.0104167 + 0.499891 i, -0.0104167 - 0.499891 i}},
 {110, {-0.00568182 + 0.499968 i, -0.00568182 - 0.499968 i}},
 {160, {-0.00390625 + 0.499985 i, -0.00390625 - 0.499985 i}},
 {210, {-0.00297619 + 0.499991 i, -0.00297619 - 0.499991 i}},
 {260, {-0.00240385 + 0.499994 i, -0.00240385 - 0.499994 i}},
 {310, {-0.00201613 + 0.499996 i, -0.00201613 - 0.499996 i}},
 {360, {-0.00173611 + 0.499997 i, -0.00173611 - 0.499997 i}},
 {410, {-0.00152439 + 0.499998 i, -0.00152439 - 0.499998 i}},
 {460, {-0.0013587 + 0.499998 i, -0.0013587 - 0.499998 i}},
 {510, {-0.00122549 + 0.499998 i, -0.00122549 - 0.499998 i}}}
```

You say that when you go with very large r's, the plots of the corresponding current functions y[t] don't change much as you change r.
What is it about the output immediately above that gives you this idea?
What do you mean?

❑ G.5.b.i) An eigenvalue = 0

When you come across this linear system

x'[t] = −0.75 x[t] + 0.25 y[t]
y'[t] = 0.75 x[t] − 0.25 y[t],

you immediately write down this coefficient matrix:

```
A = (-0.75   0.25
      0.75  -0.25);
MatrixForm[A]
```

$$\begin{pmatrix} -0.75 & 0.25 \\ 0.75 & -0.25 \end{pmatrix}$$

You check it out:

```
Clear[x, y, t];
linearsystem = {x'[t], y'[t]} == A.{x[t], y[t]};

ColumnForm[Thread[linearsystem]]
```

```
x'[t] == -0.75 x[t] + 0.25 y[t]
y'[t] == 0.75 x[t] - 0.25 y[t]
```

You ask *Mathematica* for the eigenvectors and eigenvalues of A:

```
Clear[eigenvector];
{eigenvector[1], eigenvector[2]} = Eigenvectors[A]
```

```
{{-0.707107, 0.707107}, {-0.316228, -0.948683}}
```

```
Clear[eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Chop[Eigenvalues[A]]
```

```
{-1., 0}
```

As explained in the tutorials, solution pairs {x[t], y[t]} are given by

$\{x[t], y[t]\} = E^{A t}$.starter.

Explain this:
If the starter is not a multiple of eigenvector[1] and

$\{x[t], y[t]\} = E^{A t}$.starter

is the corresponding solution pair, then as t gets large $\frac{y[t]}{x[t]}$ tends to:

$$\frac{-0.948683}{-0.316228}$$

```
3.
```

❑ G.5.b.ii) Chemical reactions

Given quantities of two reactants X and Y are mixed together. Agree that x[t] measures amount of X at time t units after the reaction begins. And agree that y[t]measures the amount of Y at time t units after the reaction begins. In the reaction under study here,

x'[t] = − k1 x[t] + k2 y[t]
y'[t] = k1 x[t] − k2 y[t].

where k1 and k2 are positive reaction rates.

This system of linear differential equations reflects the fact that X reacts with Y to produce more Y and Y reacts with X to produce more X. Both reactions are reversible.
Look at the eigenvalues and eigenvectors of the cleared coefficient matrix:

```
Clear[k1, k2];
A = (-k1   k2
      k1  -k2);
MatrixForm[A]
```

$$\begin{pmatrix} -k1 & k2 \\ k1 & -k2 \end{pmatrix}$$

```
Eigenvalues[A]
```

```
{0, -k1 - k2}
```

```
Eigenvectors[A]
```

$$\left\{\left\{\frac{k2}{k1}, 1\right\}, \{-1, 1\}\right\}$$

The question here is:
What is the ultimate ratio Y to X as t goes to infinity.
Does it depend on the starting values x[0] and y[0]?

Click for a comment.

You start with x[0] and y[0] both positive; so it is impossible to start on a multiple of {-1,1}.

G.6) Continuous dynamical systems: Life Sciences

Reservoir Models for drug metabolization

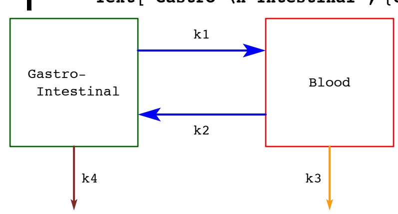
Population models and the ultimate sex ratio

❑ G.6.a.i) Reservoir Models for drug metabolization

Lots of folks deal with the hot issue of models for drug metabolization. The simplest models for these systems are two compartment models such as this one:

```
Clear[box, color];
box[a_, color_] :=
 Graphics[{color, Line[{{a[[1]] - 1, a[[2]] - 1}, {a[[1]] - 1, a[[2]] + 1},
     {a[[1]] + 1, a[[2]] + 1}, {a[[1]] + 1, a[[2]] - 1}, {a[[1]] - 1, a[[2]] - 1}}]}]
```

```
generalmodel = Show[box[{0, 0}, GreenDark], box[{4, 0}, Red],
        Arrow[{2, 0}, Tail → {1, 1/2}],
    Arrow[{-2, 0}, Tail → {3, -1/2}],
        Arrow[{0, -1}, Tail → {4, -1}, VectorColor → CadmiumYellow],
        Arrow[{0, -1}, Tail → {0, -1}, VectorColor → Brown],
        AspectRatio → Automatic,
    Epilog → {Text["k1", {2, 3/4}], Text["k2", {2, -3/4}],
      Text["k3", {15/4, -3/2}], Text["k4", {1/4, -3/2}],
      Text["Gastro-\n Intestinal", {0, 0}], Text["Blood", {4, 0}]}];
```

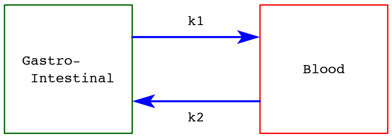


The arrows indicate that the amount leaving or entering each reservoir (depending on the direction of the arrow)
is proportional to the amount in the reservoir.

Agree that
→ x[t] measures the amount of the substance in the gastro-intestinal tract
→ y[t] measures the amount of the substance in the blood t time units after the substance has been ingested into the gastro-intestinal track.

In the case that k3 = 0 and k4 = 0, you get this simplified model:

```
Show[box[{0, 0}, GreenDark],
  box[{4, 0}, Red], Arrow[{2, 0}, Tail → {1, 1/2}],
  Arrow[{-2, 0}, Tail → {3, -1/2}], AspectRatio → Automatic,
  Epilog → {Text["k1", {2, 3/4}], Text["k2", {2, -3/4}],
    Text["Gastro-\n Intestinal", {0, 0}], Text["Blood", {4, 0}]}];
```



This model neglects urination and defecation.

The corresponding linear system is

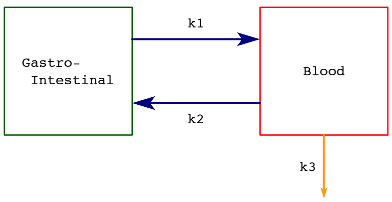
$x'[t] = -k1\, x[t] + k2\, y[t]$

$y'[t] = k1\, x[t] - k2\, y[t].$

This is the same as the chemical model that you'll find in the problem on when an eigenvalue is 0.

This simplified model isn't realistic because it doesn't allow for elimination through urination or defecation. This following diagram sets up the model for elimination through urination.

```
urinemodel = Show[box[{0, 0}, GreenDark], box[{4, 0}, Red],
   Arrow[{2, 0}, Tail → {1, 1/2}, VectorColor → NavyBlue],
   Arrow[{-2, 0}, Tail → {3, -1/2}, VectorColor → NavyBlue],
   Arrow[{0, -1}, Tail → {4, -1}, VectorColor → CadmiumYellow],
   AspectRatio → Automatic, Epilog → {Text["k1", {2, 3/4}],
     Text["k2", {2, -3/4}], Text["k3", {15/4, -3/2}],
     Text["Gastro-\n Intestinal", {0, 0}], Text["Blood", {4, 0}]}];
```

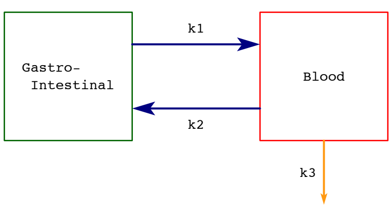


Write down the linear system resulting from this diagram.

□G.6.a.ii) Trying it out

Stay with the model:

```
Show[urinemodel, AspectRatio → Automatic];
```



Go with specific the specific numbers

$k1 = 0.5$, $k2 = 0.5$, and $k3 = 0.6$

Find formulas for x[t] and y[t] given starting data

$x[0] = 5$ (milligrams) and $y[0] = 0$.

Plot {x[t], y[t]} and give the limiting values of x[t] and y[t] as t goes to infinity.

□G.6.b.i) Linear population models

This problem was adapted from original research articles by
Leo Goodman (Biometrics 9,1953) and
David G. Kendall (Journal of the Royal Statistical Society B.11, 1949).
Both articles appear in the book Mathematical Demography
(Springer-Verlag Biomathematics Series Volume 6) edited by top-notch population biologists
David Smith and Nathan Keyfitz
(Springer-Verlag, New York, 1977).

Agree that female[t] measures the population of females and that male[t] measures the population of males t years after a given reference year. One linear model for this setup is

$male'[t] = -dm\, male[t] + bm\, (male[t] + female[t])$

$female'[t] = -df\, female[t] + bf\, (male[t] + female[t]).$

In this model, bm, bf, dm, and df are positive numbers.

bm measures the relative birth rate of males,
bf measures the relative birth rate of females,
dm measures the relative death rate of males;
df measures the relative death rate of females.

This model assumes that the relative growth rates of both the male and female populations are proportional to the whole population (male[t] + female[t]).

The coefficient matrix for this linear population model is:

```
Clear[dm, bf, dm, df];
A = ( bm - dm   bm
      bf        bf - df );
MatrixForm[A]
```

$$\begin{pmatrix} bm - dm & bm \\ bf & bf - df \end{pmatrix}$$

Check:

```
Clear[male, female, t];
popmodel = {male'[t], female'[t]} == A.{male[t], female[t]};

ColumnForm[Thread[popmodel]]
```

```
male'[t] == bm female[t] + (bm - dm) male[t]
female'[t] == (bf - df) female[t] + bf male[t]
```

This checks.

When you go with $bm = bf$ and $dm = df$, so that males and females are equally likely to be born and equally likely to die, then you get

```
ColumnForm[Thread[popmodel /. {bm -> bf, dm -> df}]]
```

```
male'[t] == bf female[t] + (bf - df) male[t]
female'[t] == (bf - df) female[t] + bf male[t]
```

And:

```
Eigenvalues[A /. {bm → bf, dm → df}]
```

```
{2 bf - df, -df}
```

```
Eigenvectors[A /. {bm → bf, dm → df}]
```

```
{{1, 1}, {-1, 1}}
```

When you go with

$bm = bf$ and $dm = df$,

as above, and you go with

$2\, bf > df$,

This means that the birth rate of females is at least half the death rate..

which eigenvector is the ultimate eigenvector in the sense that as t gets large {male[t],female[t]} gravitates to the line through {0,0} determined by this eigenvector?

Click for a comment.

Remember that bm, bf, dm, and df are all positive.

And because populations cannot be negative, it is impossible to start on a multiple of {-1,1}.

□G.6.b.ii) Ultimate sex ratio = 1 if $2\, bf > df$

Explain:

When you go with

$bm = bf$ and $dm = df$,

and you go with

$2\, bf > df$,

then this model predicts that both sexes prosper, growing approximately exponentially, with ultimate sex ratio

$\frac{male[t]}{female[t]} \to 1$ as $t \to \infty$.

So that any huge initial excess of males or females disappears over the course of time regardless of starting data.

□G.6.b.iii) Extinction if $2\, bf < df$

Explain:

When you go with

$bm = bf$ and $dm = df$,

and you go with

$2\, bf < df$,

then this model predicts that both sexes are headed towards extinction in the sense that

$male[t] \to 0$ and $female[t] \to 0$ as $t \to \infty$.

□G.6.c.i) When females are population dominant

From an article by acclaimed evolutionary biologist Jared Diamond of the UCLA Medical School writing in the magazine Natural History (September, 1994):

"Herds of wild horses consist of one stallion and up to a half dozen mares. . ."

In certain populations, the females are population dominant in the sense that the growth of the populations of both sexes depends only on the number of females. Examples are the wild horse population and some pure-bred dog populations in which very few males are allowed to breed while females are bred almost indiscriminately.

Discuss why an appropriate linear population model for a female population dominant society is

$male'[t] = -dm\, male[t] + bm\, female[t]$

$female'[t] = -df\, female[t] + bf\, female[t].$

□G.6.c.ii) Female population dominant society with $bf > df$

Go with the linear population model for a female population dominant society:

$male'[t] = -dm\, male[t] + bm\, female[t]$

$female'[t] = -df\, female[t] + bf\, female[t].$

Set up the matrix:

```
Clear[dm, bf, dm, df];
A = ( -dm   bm
       0    bf - df );
MatrixForm[A]
```

$$\begin{pmatrix} -dm & bm \\ 0 & bf - df \end{pmatrix}$$

Check:

```
Clear[male, female, t];
linpopmodel = {male'[t], female'[t]} == A.{male[t], female[t]};

ColumnForm[Thread[linpopmodel]]
```

```
male'[t] == bm female[t] - dm male[t]
female'[t] == (bf - df) female[t]
```

Now do as you are compelled:

```
Clear[eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Eigenvalues[A]
```
{bf - df, -dm}
```
Clear[eigenvector];
{eigenvector[1], eigenvector[2]} = Eigenvectors[A]
```
$\{\{\frac{bm}{bf - df + dm}, 1\}, \{1, 0\}\}$

If bf > df, then which eigenvector is the ultimate eigenvector ?
How do you know that if bf > df, then this model predicts that the both male and female populations eventually grow approximately exponentially and the ultimate sex ratio $\frac{male[t]}{female[t]} \rightarrow \frac{(dm + bf - df)}{bm}$ as $t \rightarrow \infty$?
Does this allow for the possibility that the male population tends to 0 because very few of them are needed to sustain the overall populations?

Click for a comment

Remember that bm, bf, dm,and df are all positive.

G.6.c.iii) Controlling the ultimate sex ratio in a fascist female population dominant society

Imagine that you are a big shot in a fascist female population dominant society and imagine that you can control the size of dm (percentage death rate of males). How would you set dm in terms of bf, df and bm to guarantee that the ultimate sex ratio $\frac{male[t]}{female[t]} \rightarrow 1$ as $t \rightarrow \infty$?

G.6.c.iv) What the model predicts if bf < df

Look again at the linear population model for a female population dominant society:

male'[t] = − dm male[t] + bm female[t]
female'[t] = − df female[t] + bf female[t].

Set up the matrix:
```
Clear[dm, bf, dm, df];
A = ( -dm    bm
       0   bf - df );
MatrixForm[A]
```
$\begin{pmatrix} -dm & bm \\ 0 & bf - df \end{pmatrix}$

Check:
```
Clear[male, female, t];
linpopmodel = {male'[t], female'[t]} == A.{male[t], female[t]};

ColumnForm[Thread[linpopmodel]]
```
male′[t] == bm female[t] - dm male[t]
female′[t] == (bf - df) female[t]

Now do as you are compelled:
```
Clear[eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Eigenvalues[A]
```
{bf - df, -dm}

What does the model predict if bf < df?

G.7) Discrete dynamical systems: Life Sciences

Population dynamics and Mendelian genetic dynamics

G.7.a) Population dynamics

This problem is suggested by work on page 120 of the book "Demography through Problems" by Nathan Keyfitz and John Beekman, Springer-Verlag, New York, 1984.

Agree that
x[n] = the fraction of the human population living in urban areas in year 10 n
and
y[n] is the fraction of the population living in rural areas in year 10 n;
so x[n + 1] is the fraction of the population living in urban areas and y[n + 1] is the fraction of the population living in rural areas in year 10 years later.

The relationship between
{x[n], y[n]} and {x[n + 1], y[n + 1]}
is expressed in this discrete dynamical system:
x[n + 1] = 0.98 x[n] + 0.12 y[n]
y[n + 1] = 0.02 x[n] + 0.88 y[n] .

Interpretation:
- Over a ten year period, 98% of the urban dwellers stay in urban areas and 12% of the rural dwellers move to urban areas. And
- Over a ten year period, 2% of the urban dwellers move to rural areas and 88% of the rural dwellers continue to live in rural areas.

Some folks like to call this type of problem by the name Markov Chain analysis.

Set up a good matrix and calculate its eigenvalues and eigenvectors.
Then use the results to calculate the ultimate ratio $\frac{x[n]}{y[n]}$ as n gets large.
In the long run, what percentage of the population lives in urban areas?

G.7.b.i) Discrete dynamical systems: Mendelian genetic dynamics: Continued continued crossings with a hybrid

This problem was adapted from the work in Chapter 5 of Fred Roberts's book "Discrete Mathematical Models", Prentice-Hall, New York, 1976.

Gregor Mendel (1822-84) laid the base for the modern theory of genetics by putting it on a mathematical basis.
In the Mendelian system, a gene pair is:
- dominant (D) if both of its slots are dominant,
- recessive (R) if both of its slots are recessive or
- hybrid (H) if it has one dominant and one recessive slot.

Even though the hybrids carry both dominant and recessive slots, hybrids exhibit the dominant trait but retain the ability to pass on the recessive trait.
A good example of dominant versus recessive is
dominant = Not albino
and
recessive = Albino.

Here are the predicted outcomes according to Mendelian theory:

Parents ------> Offspring
D - D--------> D 100%
D - H --------> D 50% and H 50%
D - R---------> H 100%
H - H---------> D 25% , H 50% and R 25%
H - R---------> H 50% and R 50 %
R - R ---------> R 100%

Seedgrowers often practice the procedure known as continued crossings with a hybrid (H).
This means each new plant has one hybrid parent and one parent from the general population.
Generally the hybrid parent has a certain desirable quality.
Start the process of continued crossings with a hybrid and put
Put
D[n] = fraction of dominants in generation n of the process,
H[n] = fraction of hybrids in generation n of the process, and
R[n] = fraction of recessives in generation n of the process.
This gives:

D[n + 1] = 0.5 D[n] + 0.25 H[n] + 0.0 R[n]

Reason:
D - H ---------------------> D 50% and H 50%
H - H ---------> D 25% , H 50% and R 25%
R - H ------------>D 0 %, H 50% and R 50 %

H[n + 1] = 0.5 D[n] + 0.5 H[n] + 0.5 R[n]

Reason:
D - H ---------------------> D 50% and H 50%
H - H ---------> D 25% , H 50% and R 25%
R - H ---------------->H 50% and R 50 %

R[n + 1] = 0.0 D[n] + 0.25 H[n] + 0.5 R[n]

,Reason:
D - H ---------------------> D 50% and H 50% and R 0 %
H - H ----------------> D 25% , H 50% and R 25%
R - H ---------------->D 0 % H 50% and R 50 %

Summary:
Go with continued crossings with a hybrid, and put

D[n] = fraction of dominants in generation n.
H[n] = fraction of hybrids in generation n.
R[n] = fraction of recessives in generation n.
This leads to the discrete dynamical system:
D[n + 1] = 0.5 D[n] + 0.25 H[n] + 0.0 R[n],
H[n + 1] = 0.5 D[n] + 0.5 H[n] + 0.5 R[n],
R[n + 1] = 0.0 D[n] + 0.25 H[n] + 0.5 R[n].
Set up a good matrix A and calculate its eigenvalues and eigenvectors.
Then use the results to predict the limiting values of D[n], H[n] , and R[n] as n gets large.

G.7.b.ii) Mendelian genetic dynamics: Starting with a hybrid-dominant mating

Look at these calculations which show A^k.{1, 0, 0} for generations k = 0,1,2,...,18.
```
A = ( 0.5  0.25  0
      0.5  0.5   0.5
      0    0.25  0.5 );
starter = {1, 0, 0};
ColumnForm[Table[{k , MatrixPower[A, k].starter}, {k, 0, 18}]]
```
{0, {1, 0, 0}}
{1, {0.5, 0.5, 0}}
{2, {0.375, 0.5, 0.125}}
{3, {0.3125, 0.5, 0.1875}}
{4, {0.28125, 0.5, 0.21875}}
{5, {0.265625, 0.5, 0.234375}}
{6, {0.257813, 0.5, 0.242188}}
{7, {0.253906, 0.5, 0.246094}}
{8, {0.251953, 0.5, 0.248047}}
{9, {0.250977, 0.5, 0.249023}}
{10, {0.250488, 0.5, 0.249512}}
{11, {0.250244, 0.5, 0.249756}}
{12, {0.250122, 0.5, 0.249878}}
{13, {0.250061, 0.5, 0.249939}}
{14, {0.250031, 0.5, 0.249969}}
{15, {0.250015, 0.5, 0.249985}}
{16, {0.250008, 0.5, 0.249992}}
{17, {0.250004, 0.5, 0.249996}}
{18, {0.250002, 0.5, 0.249998}}

The generation number k is in the left slot and. A^k.starter is on the right.

That
starter = {1, 0, 0}
says that in generation 0,the hybrids were always mated to a dominant D.
Are there any recessives in the first generation?
What percentage of recessives are expected in generation 3?

◻G.7.b.iii) Mendelian genetic dynamics: Continued crossings with a dominant

Go back to:

Parents ------> Offspring
D - D--------> D 100%
D - H --------> D 50% and H 50%
D - R---------> H 100%
H - H---------> D 25% , H 50% and R 25%
H - R---------> H 50% and R 50 %
R - R ---------> R 100%

This time investigate the procedure known as continued crossings with a dominant (D).
This means each new plant has one D parent and one parent from the general population.

D[n] = fraction of dominants in generation n of the process,
H[n] = fraction of hybrids in generation n of the process, and
R[n] = fraction of recessives in generation n of the process.

Write down the resulting dynamical system. Set up a good matrix and calculate its eigenvalues and eigenvectors.
Then use the results to calculate to predict the limiting values of D[n], H[n] , and R[n] as n gets large.

Click for a tip on how to start.

Look at:

D - D--------> D 100%
D - H --------> D 50% and H 50%
D - R---------> H 100%
H - H---------> D 25% , H 50% and R 25%
H - R---------> H 50% and R 50 % ,
R - R ---------> R 100%

For one thing,

D[n + 1] = 1.0 D[n] + 0.50 H[n] + 0.0 R[n].

Also

H[n+1] = ????????? (You fill this in.)

And

R[n + 1] = 0.0 D[n] + 0.0 H[n] + 0.0 R[n].

```
   AxesLabel → {"trace", "det"},
   PlotRange → {-6, 6}, DisplayFunction → Identity];
cutofflabel = Graphics[Text[FontForm[
     "\!\(trace\^2\) - 4 det == 0", {"Times", 10}], {3.0, -4.0}]];

pointer =
  Arrow[{2, 1} - {3.0, -3.6}, Tail → {3.0, -3.6}, VectorColor → Black];

twosuckercutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{-5, 0}, {0, 0}}]}];
twosuckerlabel = Graphics[Text[FontForm[
     " Both real and \n both negative", {"Times", 10}], {-4, 1}]];

twopropcutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{0, 0}, {5, 0}}]}];
twoproplabel = Graphics[Text[FontForm[
     " Both real and \n both positive", {"Times", 10}], {4, 1}]];

propswirlcutoff = Graphics[
   {CadmiumOrange, Thickness[0.01], Line[{{0, 0}, {0, 6}}]}];
propswirllabel = Graphics[Text[
    FontForm[" p + I q \n and p - I q\n with p > 0 \n and q ≠ 0",
     {"Times", 10}], {2, 4}]];

pureswirlerlabel =
  Graphics[Text[FontForm[" p\n u\n r ", {"Times", 12}], {0, 7}]];

suckswirlcutoff = Graphics[
   {CadmiumOrange, Thickness[0.01], Line[{{0, 0}, {0, 6}}]}];
suckswirllabel = Graphics[Text[
    FontForm["p + I q \n and p - I q\n with p < 0 \n and q ≠ 0",
     {"Times", 10}], {-2, 4}]];

negdetcutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{-5, 0}, {5, 0}}]}];
negdetlabel = Graphics[
   Text[FontForm["Both real and \n one > 0 and one < 0",
     {"Times", 10}], {0, -1.5}]];

pureswirlerlabel = Graphics[
   Text[FontForm["0 + I q and 0 - I q on positive vertical axis",
     {"Times", 10}], {0, 7}]];

chart = Show[cutoff, cutofflabel, pointer,
   twosuckercutoff, twosuckerlabel, twopropcutoff,
   twoproplabel, propswirlcutoff, propswirllabel,
   suckswirlcutoff, suckswirllabel, negdetcutoff, negdetlabel,
   pureswirlerlabel, DisplayFunction → $DisplayFunction];
```

G.8) The Fibonacci difference equation

◻G.8.a.i) Fibonacci numbers come from the difference equation

y[k + 2] = y[k + 1] + y[k]

You get the Fibonacci numbers from the difference equation
y[k + 2] = y[k + 1] + y[k], starting with y[0] = 0 and y[1] = 1.

Come up with a matrix A (called the coefficient matrix) to write this difference equation in matrix form

{y[k], y[k + 1]} = A^k.{y[0], y[1]}

for all k = 0, 1, 2, 3, 4, 5, ...

◻G.8.a.ii) Slamming out some Fibonacci numbers

Stay with the same setup as in part i)
Use the matrix form of the difference equation to slam out
y[2], y[3], , . . , y[49] and y[50]
in terms of y[0] = 0 and y[1] = 1.

◻G.8.a.iii) The golden ratio

Check the eigenvalues and eigenvectors of your coefficient matrix A and use the result to predict the limiting value
of $\frac{y[k+1]}{y[k]}$ as k gets large.

FYI : The limiting value of $\frac{y[k+1]}{y[k]}$ as k gets large is known as the Golden Ratio.

```
N[GoldenRatio]
```

1.61803

◻G.8.a.iv) Different values of y[0] and y[1]

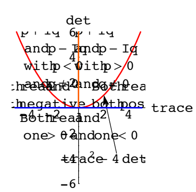
When you go with the Fibonacci difference equation
y[k + 2] = y[k + 1] + y[k] and start with y[0] > 0 and y[1] > 0
(so that y[0] is not necessarily 0 and y[1] is not necessarily 1),
then what is the limiting value of $\frac{y[k+1]}{y[k]}$ as k gets large?

G.9) Using the cheat sheet to analyze the behavior of E^{At}.starter

◻G.9.a.i)

Here is the cheat sheet chart from the Tutorials:

```
cutoff =
  Plot[trace^2/4, {trace, -5, 5}, PlotStyle → {{Red, Thickness[0.01]}},
```



To analyze a matrix A = $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$,

you put

trace = a + d

and

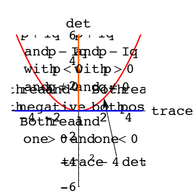
det = a d − b c

and then plot the point {trace,det} on the chart.

◻G.9.a.i) Going to {0,0}

Take another look at the chart.

```
Show[chart];
```



Agree or disagree:

If matrix A = $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is a given matrix and

trace = a + d < 0,

and

det = a d − b c > 0,

then

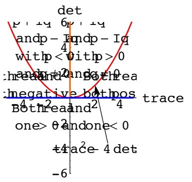
E^{At}.starter → {0, 0}

no matter where the starter point is.

◻G.9.a.ii) Going to infinity

Take another look at the chart.

```
Show[chart];
```



Agree or disagree:

If matrix $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is a given matrix and

$\text{trace} = a + d > 0$,

and

$\det = a\,d - b\,c > 0$,

then

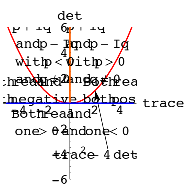
$\| E^{A\,t}.\text{starter} \| \to$ infinity

no matter where the starter point is (as long as starter $\neq \{0,0\}$).

G.9.a.iii) Oscillating on closed curves if trace = 0

Take another look at the chart.

```
Show[chart];
```



Agree or disagree:

If matrix $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ is a given matrix and

$\text{trace} = a + d = 0$,

then as t advances

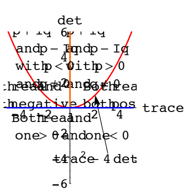
$E^{A\,t}.\text{starter}$

oscillates on closed curves centered at {0,0}.

G.9.a.iv) Set a constant

Take another look at the chart.

```
Show[chart];
```



Here is a matrix A with one cleared entry:

```
Clear[a];
A = (a  -2
     1   1);
MatrixForm[A]
```

$\begin{pmatrix} a & -2 \\ 1 & 1 \end{pmatrix}$

This time you have three jobs:

Go with starter $\neq \{0,0\}$.

- Job 1: Set a so that $E^{A\,t}.\text{starter} \to \{0, 0\}$ as t gets large.

Put answer here:

- Job 2: Set a so that $\| E^{A\,t}.\text{starter} \| \to \infty$ as t gets large.

Put answer here:

- Job 3: Set a so that as t advances $E^{A\,t}$.starter oscillates on closed curves centered at {0,0}.

G.9.b.i) Round off instability

Here's a new matrix A:

```
A = (-1.2  1.5
     -1.3  1.2);
MatrixForm[A]
```

$\begin{pmatrix} -1.2 & 1.5 \\ -1.3 & 1.2 \end{pmatrix}$

Check the eigenvalues of A:

```
Clear[eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Eigenvalues[A]
```

{0.714143 i, -0.714143 i}

The form is p + I q with

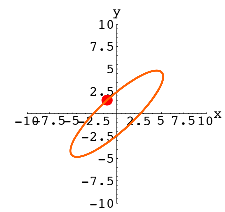
$p = 0$ and $q \neq 0$.

This means as t advances $E^{A\,t}$.starter is guaranteed to advance on an ellipse centered at {0,0}.

```
ranger = 10;
starter = {Random[Real, {-ranger/4, ranger/4}],
   Random[Real, {-ranger/4, ranger/4}]};

MatrixExp[A t].starter
```

$\{1.52004\,(1.05021\,i\,e^{-0.714143\,i\,t} - 1.05021\,i\,e^{0.714143\,i\,t}) - 1.11242\,((0.5 - 0.840168\,i)\,e^{-0.714143\,i\,t} + (0.5 + 0.840168\,i)\,e^{0.714143\,i\,t}),$
$-1.11242\,(-0.910182\,i\,e^{-0.714143\,i\,t} + 0.910182\,i\,e^{0.714143\,i\,t}) + 1.52004\,((0.5 + 0.840168\,i)\,e^{-0.714143\,i\,t} + (0.5 - 0.840168\,i)\,e^{0.714143\,i\,t})\}$

```
endtime = 24;

trajectoryplot = ParametricPlot[MatrixExp[A t].starter,
   {t, 0, endtime}, PlotStyle -> {{CadmiumOrange, Thickness[0.01]}},
   DisplayFunction -> Identity];

starterpoint = starter;
starterplot =
  Graphics[{Red, PointSize[0.06], Point[starterpoint]}];

Show[starterplot, trajectoryplot,
  PlotRange -> {{-ranger, ranger}, {-ranger, ranger}}, Axes -> True,
  AxesLabel -> {"x", "y"}, DisplayFunction -> $DisplayFunction];
```



Now take the coefficient matrix A and build in some random noise like this and compare:

```
h = 0.049;
radomnoise = {{Random[Real, {-h, h}], Random[Real, {-h, h}]},
   {Random[Real, {-h, h}], Random[Real, {-h, h}]}};
noisyA = A + radomnoise;

MatrixForm[noisyA]
MatrixForm[A]
```

$\begin{pmatrix} -1.24522 & 1.4929 \\ -1.27325 & 1.16298 \end{pmatrix}$

$\begin{pmatrix} -1.2 & 1.5 \\ -1.3 & 1.2 \end{pmatrix}$

NoisyA is guaranteed to round off nicely to A.

Now look at a plot of

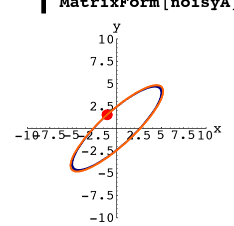
$E^{\text{noisyA}\,t}$.starter

together with a plot of

$E^{A\,t}$.starter:

Both trajectories have the same starting point.

```
h = 0.049;
randomnoise = {{Random[Real, {-h, h}], Random[Real, {-h, h}]},
   {Random[Real, {-h, h}], Random[Real, {-h, h}]}};
noisyA = A + randomnoise;

noisytrajectoryplot = ParametricPlot[MatrixExp[noisyA t].starter,
   {t, 0, endtime}, PlotStyle -> {{NavyBlue, Thickness[0.01]}},
   DisplayFunction -> Identity];

Show[starterplot, noisytrajectoryplot, trajectoryplot,
  PlotRange -> {{-ranger, ranger}, {-ranger, ranger}}, Axes -> True,
  AxesLabel -> {"x", "y"}, DisplayFunction -> $DisplayFunction];

MatrixForm[A]
MatrixForm[noisyA]
```



$\begin{pmatrix} -1.2 & 1.5 \\ -1.3 & 1.2 \end{pmatrix}$

$\begin{pmatrix} -1.23017 & 1.54771 \\ -1.34712 & 1.22912 \end{pmatrix}$

The original trajectory is orange. The noisy trajectory is blue.

Rerun at least seven times.

Each time you rerun, you get a different noisyA which rounds off to A.

Describe what you see.

Comment on the statement:

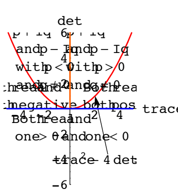
→ Even though
1) the eigenvalues of A are of the form p + I q and p − I q with p = 0 and q ≠ 0 and
2) noisyA rounds off to A, it is quite unlikely that the eigenvalues of noisyA are of the form are of the form p + I q and p − I q with p = 0 and q ≠ 0.

□ G.9.b.ii) Explanation

One of the goals of mathematics is to explain why things work out the way they do.
To this end, take anther look at look at the cheat sheet chart from one of the Tutorials.

```
cutoff =
  Plot[trace^2/4, {trace, -5, 5}, PlotStyle → {{Red, Thickness[0.01]}},
   AxesLabel → {"trace", "det"},
   PlotRange → {-6, 6}, DisplayFunction → Identity];
cutofflabel = Graphics[Text[FontForm[
     "\!\(trace\^2\) - 4 det == 0", {"Times", 10}], {3.0, -4.0}]];

pointer =
  Arrow[{2, 1} - {3.0, -3.6}, Tail → {3.0, -3.6}, VectorColor → Black];

twosuckercutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{-5, 0}, {0, 0}}]}];
twosuckerlabel = Graphics[Text[FontForm[
     " Both real and \n both negative", {"Times", 10}], {-4, 1}]];

twopropcutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{0, 0}, {5, 0}}]}];
twoproplabel = Graphics[Text[FontForm[
     " Both real and \n both positive", {"Times", 10}], {4, 1}]];

propswirlcutoff = Graphics[
   {CadmiumOrange, Thickness[0.01], Line[{{0, 0}, {0, 6}}]}];
propswirllabel = Graphics[Text[
    FontForm[" p + I q \n and p - I q\n with p > 0 \n and q ≠ 0",
     {"Times", 10}], {2, 4}]];

pureswirlerlabel =
  Graphics[Text[FontForm[" p\n u\n r ", {"Times", 12}], {0, 7}]];

suckswirlcutoff = Graphics[
   {CadmiumOrange, Thickness[0.01], Line[{{0, 0}, {0, 6}}]}];
suckswirllabel = Graphics[Text[
    FontForm["p + I q \n and p - I q\n with p < 0 \n and q ≠ 0",
     {"Times", 10}], {-2, 4}]];

negdetcutoff =
  Graphics[{Blue, Thickness[0.01], Line[{{-5, 0}, {5, 0}}]}];
```

```
negdetlabel = Graphics[
   Text[FontForm["Both real and \n one > 0 and one < 0",
     {"Times", 10}], {0, -1.5}]];

pureswirlerlabel = Graphics[
   Text[FontForm["0 + I q and 0 - I q on positive vertical axis",
     {"Times", 10}], {0, 7}]];

chart = Show[cutoff, cutofflabel, pointer,
   twosuckercutoff, twosuckerlabel, twopropcutoff,
   twoproplabel, propswirlcutoff, propswirllabel,
   suckswirlcutoff, suckswirllabel, negdetcutoff, negdetlabel,
   pureswirlerlabel, DisplayFunction → $DisplayFunction];
```



Given a coefficient matrix:

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

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

The part of the chart corresponding eigenvalues of the form p + I q and p − I q with p = 0 and q ≠ 0 is

trace = a + d = 0

and

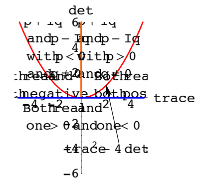
det = a d − bc > 0.

Try to use this information to explain:
If you start with any coefficient matrix A whose eigenvalues are of the form p + I q and p − I q with p = 0 and q ≠ 0, and you make the slightest random error in entering A, then the resulting matrix is all but guaranteed to have eigenvalues not of the form that p + I q and p − I q with p = 0 and q ≠ 0.

□ G.9.b.iii) Other possibilities

Take another look at the chart:

```
Show[chart];
```



If you are given a matrix with

trace < 0 and det = 0,

then use the chart to predict what wildly different things can happen when very slight random errors are made when you enter the matrix.

G.10) Matrix exponential $E^{A\,t}$ for matrices A with eigenvalue[1] = 0 and 0 > eigenvalue[2]. Matrix powers A^k for matrices A with eigenvalue[1] = 1 and 1 > | eigenvalue[2] |

□ G.10.a) Matrix exponential $E^{A\,t}$ for matrices A with eigenvalue[1] = 0 and 0 > eigenvalue[2].

Here are facts from the Basics:

□ **If A is any diagonalizable matrix and the absolute values all eigenvalues of A are negative , then no matter what X is, you are guaranteed that**

$E^{A\,t}.X \to \{0, 0\}$ as t gets large.

□ **If A is any diagonalizable matrix and the absolute values of the of both eigenvalues of A are both positive, then no matter what as long as X ≠ {0, 0}, you are guaranteed that**

(|| E)$^{A\,t}$.X || → infinity as t gets large.

□ **If eigenvalue[1] > eigenvalue[2], then the ultimate direction of $E^{A\,t}$.X as t gets large is in the direction of eigenvector[1] (unless X is a multiple of eigenvector[2])**

This does not give info about the case in which

eigenvalue[1] = 0 and 0 > eigenvalue[2].

It is possible to predict exactly what will happen. If this intrigues you, go on.

Here's a 2D matrix A with eigenvalue[1] = 0 and 0 > eigenvalue[2]

```
A = ( -0.50   0.40
       0.25  -0.20 );

MatrixForm[A]
```

$\begin{pmatrix} -0.5 & 0.4 \\ 0.25 & -0.2 \end{pmatrix}$

Check the eigenvalues and eigenvectors:

```
Eigenvalues[A]
```

{-0.7, 0}

Enter them so that eigenvalue[1] = 0 and 0 > eigenvalue[2]

```
Clear[eigenvalue];
{eigenvalue[2], eigenvalue[1]} = Eigenvalues[A]
```

{-0.7, 0}

Enter the eigenvectors the corresponding way so that

A.eigenvector[1] = eigenvalue[1] eigenvector[1]

and

A.eigenvector[2] = eigenvalue[2] eigenvector[2] :

```
Clear[eigenvector];
{eigenvector[2], eigenvector[1]} = Eigenvectors[A]
```

{{-0.894427, 0.447214}, {-0.624695, -0.780869}}

Note that the labels are reversed.

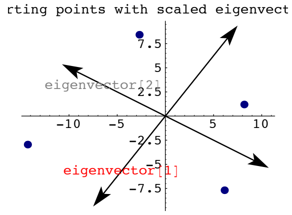
Here are four random points

{starter[1], starter[2], starter[3], starter[4]}

in 2D shown with scaled versions of the eigenvectors of A:

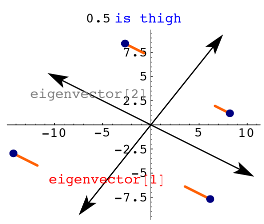
```
Clear[trajectoryplot, eigenplot, scaler, starter, thigh, k, j];

trajectoryplot[thigh_, starter_] :=
  ParametricPlot[MatrixExp[A t].starter, {t, 0, thigh},
   PlotStyle -> {{Thickness[0.01], CadmiumOrange}},
   DisplayFunction -> Identity];

starterplots = Table[Graphics[
     {NavyBlue, PointSize[0.03], Point[starter[j]]}], {j, 1, 4}];

scaler = 12;


eigenlabels = {Graphics[
     {Red, Text["eigenvector[1]", 0.6 scaler eigenvector[1]]}],
       Graphics[{GrayLevel[0.5],
      Text["eigenvector[2]", 0.6 scaler eigenvector[2]]}]};

eigenplot[j_] :=
```

```
    {Arrow[scaler eigenvector[j],
    Tail -> {0, 0}, VectorColor -> Black],
     Arrow[-scaler eigenvector[j], Tail -> {0, 0},
    VectorColor -> Black]};

starter[1] = Random[Real, {4, 10}] eigenvector[1] +
   Random[Real, {4, 10}] eigenvector[2];
starter[2] = Random[Real, {-10, -4}] eigenvector[1] +
   Random[Real, {-10, -4}] eigenvector[2];
starter[3] = Random[Real, {-10, -4}] eigenvector[1] +
   Random[Real, {4, 10}] eigenvector[2];
starter[4] = Random[Real, {4, 10}] eigenvector[1] +
   Random[Real, {-10, -4}] eigenvector[2];

thigh = 0.01;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
  starterplots, eigenplot[1], eigenplot[2], eigenlabels,
  PlotLabel -> "Starting points with scaled eigenvectors",
  DisplayFunction -> $DisplayFunction];
```



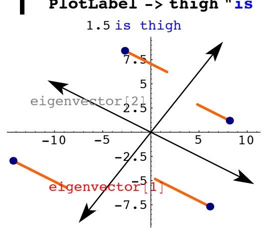
Here is how the curves
$E^{A t}$.starter[1], $E^{A t}$.starter[2], $E^{A t}$.starter[3], $E^{A t}$.starter[4]
plot out as t advances from 0 to 0.5 shown with scaled eigenvectors of A:

```
thigh = 0.5;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
 starterplots, eigenplot[1], eigenplot[2], eigenlabels,
 PlotLabel -> thigh "is thigh", DisplayFunction -> $DisplayFunction];
```



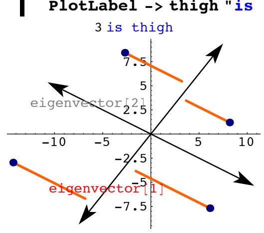
Here's what happens as t advances from 0 to 1.5 shown with scaled eigenvectors of A:

```
thigh = 1.5;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
 starterplots, eigenplot[1], eigenplot[2], eigenlabels,
 PlotLabel -> thigh "is thigh", DisplayFunction -> $DisplayFunction];
```



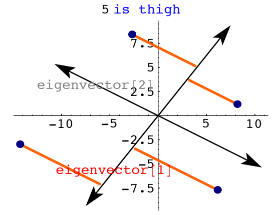
Here's what happens as t advances from 0 to 3 shown with scaled eigenvectors of A:

```
thigh = 3;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
 starterplots, eigenplot[1], eigenplot[2], eigenlabels,
 PlotLabel -> thigh "is thigh", DisplayFunction -> $DisplayFunction];
```



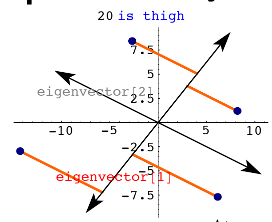
Here's what happens as t advances from 0 to 5 shown with scaled eigenvectors of A:

```
thigh = 5;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
 starterplots, eigenplot[1], eigenplot[2], eigenlabels,
 PlotLabel -> thigh "is thigh", DisplayFunction -> $DisplayFunction];
```



Here's what happens as t advances from 0 to 20:

```
thigh = 20;
Show[Table[trajectoryplot[thigh, starter[j]], {j, 1, 4}] ,
 starterplots, eigenplot[1], eigenplot[2], eigenlabels,
 PlotLabel -> thigh "is thigh", DisplayFunction -> $DisplayFunction];
```



As k gets large, the points $E^{A t}$.starter stall on the line through {0,0} defined by eigenvector[1].
You could have predicted this in advance because eigenvalue[1] = 0 and
0 > eigenvalue[2]:.
What is different is:
-> as k gets large, $E^{A t}$.starter does not go to {0, 0} and
-> as k gets large, (|| $E^{A t}$.starter || does not go to infinity.

Instead $E^{A t}$ starter gravitates to a certain specific point on the line through {0,0} defined by eigenvector[1].
The fact of the matter is that if
eigenvalue[1] = 0 and 0 > eigenvalue[2]:.
and
starter = s eigenvector[1] + t eigenvector[2]
then you are guaranteed that as t gets large
$E^{A t}$.starter → s eigenvector[1].
Your job is to explain why this is guaranteed.

Click on the right for a heavy tip.

In one of the Basics, this fact is explained:

$E^{A t}$.(s eigenvect[1] + t eigenvect[2])

$= s\, E^{\text{eigenval[1]} t}$ eigvect[1] + t $E^{\text{eigenval[2]} t}$ eigvect[2]

$= E^{\text{eigenval[1]} t}$ (s eigvect[1] + t $E^{(\text{eigenval[2]} - \text{eigenval[1]}) t}$ eigvect[2]

In this setup eigval[1] = 0 and 0 > eigval[2].

This gives

$E^{A t}$.(s eigenvect[1] + t eigenvect[2])

= s eigvect[1] + t $E^{(\text{eigenval[2]}) t}$ eigvect[2]

Use the the fact that eigval[2] < 0 to make your explanation.

G.10.b.i) Matrix powers A^k for matrices A with eigenvalue[1] = 1 and 1 > | eigenvalue[2] |:

Here are facts freom the Basics:

- **If A is any diagonalizable matrix and the absolute values all eigenvalues of A are less than 1, then no matter what X is, you are guaranteed that**
  A^k.X → {0, 0}as k gets large.

- **If A is any diagonalizable matrix and the absolute values of the of both eigenvalues of A are both bigger than 1, then no matter what as long as X ≠ {0, 0}, you are guaranteed that**
  (|| A)^k.X || → infinity as k gets large.

- **If | eigenvalue[1] | > | eigenvalue[2] |, then the ultimate direction of A^k.X as k gets large is in the direction of eigenvector[1] (unless X is a multiple of eigenvector[2])**

This does not give info about the case in which
eigenvalue[1] = 1 and 1 > | eigenvalue[2] |:
It is possible to predict exactly what will happen. If this intrigiues you, go on.

Here's a random 2D matrix A with eigenvalue[1] = 1 and 1 > | eigenvalue[2] |

```
a = Random[Real, {0.1, 0.99}];
b = Random[Real, {0.1, 0.99}];
c = 1 - a;
d = 1 - b;
A = ( a  b
      c  d );

MatrixForm[A]
```

$$\begin{pmatrix} 0.49284 & 0.720137 \\ 0.50716 & 0.279863 \end{pmatrix}$$

Check the eigenvalues:

```
Clear[eigenvalue];
{eigenvalue[1], eigenvalue[2]} = Eigenvalues[A]
```

{1., -0.227296}

This gives eigenvalue[1] = 1 and 1 > | eigenvalue[2] |:

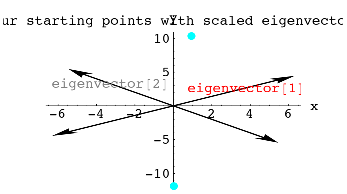
Here are two random points

{starter[1], starter[2]}

in 2D shown with scaled versions of the eigenvectors of A:

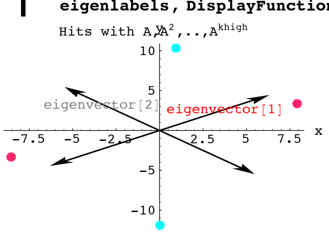
```
Clear[iterationplot, interationplotter,
  eigenplot, scaler, point, pointcolor, eigenvector,
  eigenlabels, starter, khigh, k, j, iterationpoints];

point[k_, starter_] := MatrixPower[A, k] .starter ;
pointcolor[k_, khigh_] := RGBColor[Sin[(Pi / 2) k / (khigh + 0.1)],
  Cos[(Pi / 2) k / (khigh + 0.1)], E^(-k / (khigh + 0.1))]

iterationpoints[starter_, khigh_] :=
Table[Graphics[ {pointcolor[k, khigh], PointSize[0.03],
     Point[MatrixPower[A, k] .starter]}], {k, 0, khigh}];

iterationplotter[khigh_] :=
  Show[Table[iterationpoints[starter[j], khigh], {j, 1, 2}],
   Axes -> True, AxesLabel -> {"x", "y"},
   PlotLabel -> "Hits with A,A^2,..,A^khigh",
        AspectRatio -> 1 / GoldenRatio, DisplayFunction -> Identity];

scaler := 0.9 Sqrt[point[1, starter[1]].point[1, starter[1]] ];

{eigenvector[1], eigenvector[2]} = Eigenvectors[A];
eigenplot[j_] :=
    {Arrow[scaler eigenvector[j],
    Tail -> {0, 0}, VectorColor -> Black],
     Arrow[-scaler eigenvector[j], Tail -> {0, 0},
    VectorColor -> Black]};
```

```
eigenlabels = {Graphics[
    {Red, Text["eigenvector[1]", 0.6 scaler eigenvector[1]]}],
        Graphics[{GrayLevel[0.5],
     Text["eigenvector[2]", 0.6 scaler eigenvector[2]]}]};

starter[1] = Random[Real, {4, 10}] eigenvector[1] +
   Random[Real, {4, 10}] eigenvector[2];
starter[2] = Random[Real, {-10, -4}] eigenvector[1] +
   Random[Real, {-10, -4}] eigenvector[2];

khigh = 0;
Show[iterationplotter[khigh],
  eigenplot[1], eigenplot[2], eigenlabels,
  PlotLabel -> "Four starting points with scaled eigenvectors",
  DisplayFunction -> $DisplayFunction];
```



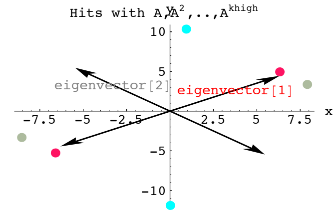
Here are the same two starter points together with the points you get when you hit each of the original starting points with A:

```
khigh = 1;
Show[iterationplotter[khigh], eigenplot[1], eigenplot[2],
  eigenlabels, DisplayFunction -> $DisplayFunction];
```



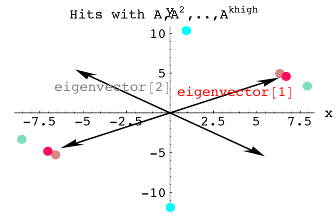
Here are the same two starter points together with the points you get when you hit each of the original starting points with A and A.A:

```
khigh = 2;
Show[iterationplotter[khigh], eigenplot[1], eigenplot[2],
  eigenlabels, DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with A, A.A and A.A.A:

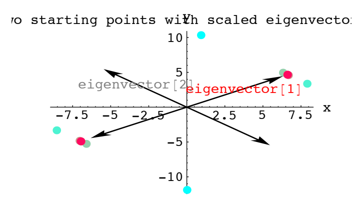
```
khigh = 3;
Show[iterationplotter[khigh], eigenplot[1], eigenplot[2],
  eigenlabels, DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with A, A.A, A^3, A^4, and A^5:

All of these powers are done with matrix multiplication.

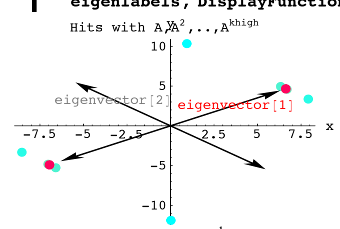
```
khigh = 5;
Show[iterationplotter[khigh],
  eigenplot[1], eigenplot[2], eigenlabels,
  PlotLabel -> "Two starting points with scaled eigenvectors",
  DisplayFunction -> $DisplayFunction];
```



Here are the same two starter points together with the points you get when you hit each of the original starting points with A, A.A, A^3, A^4, . . ., A^9, A^{10}:

All of these powers are done with matrix multiplication.

```
khigh = 10;
Show[iterationplotter[khigh], eigenplot[1], eigenplot[2],
  eigenlabels, DisplayFunction -> $DisplayFunction];
```



As k gets large, the points A^k .starter gravitate to the line through {0,0} defined by eigenvector[1].

You could have predicted this in advance because eigenvalue[1] = 1 and 1 > | eigenvalue[2] |:.

What is different is:

-> as k gets large, A^k.starter does not go to {0, 0} and

-> as k gets large, || A^k.starter || does not go to infinity.

Instead A^k starter gravitates to a certain specific point on the line through {0,0} defined by eigenvector[1].

The fact of the matter is that if

eigenvalue[1] = 1 and 1 > | eigenvalue[2] |:.

and

starter = s eigenvector[1] + t eigenvector[2]

then you are guaranteed that

A^k.starter → s eigenvector[1].

Your job is to explain why this is guaranteed.

Click on the right for a heavy tip.

In one of the Basics, this fact is explained:

$A^k.(s\,\text{eigenvect}[1] + t\,\text{eigenvect}[2])$

$$= \text{eigval}[1]^k \left(s\ \text{eigvect}[1] + t\left(\frac{\text{eigval}[2]}{\text{eigval}[1]}\right)^k \text{eigvect}[2]\right)$$

In this setup eigval[1] = 1 and1 > | eigval[2] |.

This gives

$A^k.(s\,\text{eigenvect}[1] + t\,\text{eigenvect}[2])$

$$= (s\ \text{eigvect}[1] + t\,\text{eigval}[2]^k\ \text{eigvect}[2])$$

Use the behavior of $\text{eigval}[2]^k$ as k gets large to make your explanation.

□G.10.b.ii) Identify the straight line

Stay with the same setup as in part i) so that eigenvalues of A are {eigenvalue[1], eigenvalue[2]} and
-> eigenvalue[1] = 1 and 1 > | eigenvalue[2] |.
-> starter = s eigenvector[1] + t eigenvector[2]
then you are guaranteed that as k gets large
$A^k.\text{starter} \to s\ \text{eigenvector}[1]$.
Explain this: As k gets large A^k.starter moves to s eigenvector[1] on a straight line.
Identify this straight line.

G.11) More on the matrix exponential and related matters

$A.E^{A\,t} = E^{A\,t}.A$ is guaranteed.

When you can be sure that A.B = B.A.

When you can be sure that $E^{A+B} = E^A.E^B$

□G.11.a.i) Eigenvalues of E^A

A 2D diagonalizable matrix is any matrix A of the form

$$A = \text{SpannerMatrix}.\begin{pmatrix} \text{eigenvalue}[1] & 0 \\ 0 & \text{eigenvalue}[2] \end{pmatrix}.\text{SpannerMatrix}^{-1}.$$

Here the two vertical columns of SpannerMatrix are linearly independent eigenvectors of A.

You calculate E^A by putting

$$E^A = \text{SpannerMatrix}.\begin{pmatrix} E^{\text{eigenvalue}[1]} & 0 \\ 0 & E^{\text{eigenvalue}[2]} \end{pmatrix}.\text{SpannerMatrix}^{-1}$$

Seeing this, answer this question:

Given a diagonalizable matrix A, how are the eigenvalues of E^A related to the eigenvalues of A?

□G.11.a.ii) Eigenvectors of E^A

A 2D diagonalizable matrix is any matrix A of the form

$$A = \text{SpannerMatrix}.\begin{pmatrix} \text{eigenvalue}[1] & 0 \\ 0 & \text{eigenvalue}[2] \end{pmatrix}.\text{SpannerMatrix}^{-1}.$$

Here the two vertical columns of SpannerMatrix are linearly independent eigenvectors of A.

You calculate E^A by putting

$$E^A = \text{SpannerMatrix}.\begin{pmatrix} E^{\text{eigenvalue}[1]} & 0 \\ 0 & E^{\text{eigenvalue}[2]} \end{pmatrix}.\text{SpannerMatrix}^{-1}$$

Seeing this, answer this question:
Given a diagonalizable matrix A, how are the eigenvectors of E^A related to the eigenvectors of A?

□G.11.b.i) $A.E^{A\,t} = E^{A\,t}.A$

Here are two random 2D matrices A and B:

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

$$\begin{pmatrix} -0.388399 & 1.47348 \\ -1.1683 & 0.379378 \end{pmatrix}$$

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

$$\begin{pmatrix} 0.519735 & 0.984563 \\ -1.937 & 0.431975 \end{pmatrix}$$

Here are A.B and B.A:

```
MatrixForm[A.B]
```

$$\begin{pmatrix} -3.05598 & 0.254103 \\ -1.34206 & -0.986387 \end{pmatrix}$$

```
MatrixForm[B.A]
```

$$\begin{pmatrix} -1.35213 & 1.13934 \\ 0.247648 & -2.69024 \end{pmatrix}$$

Grab and rerun all four cells a couple of times.

As you can see, the outcome
A.B = B.A
is pretty darn rare.
In spite of this, if A is diagonalizable, you can always count on having
$A.E^{A\,t} = E^{A\,t}.A$.
Here's a start on the idea in 2D: The same idea (with more writing) works in all dimensions.
A 2D diagonalizable matrix is any matrix A of the form

$$A = \text{SpannerMatrix}.\begin{pmatrix} \text{eigenvalue}[1] & 0 \\ 0 & \text{eigenvalue}[2] \end{pmatrix}.\text{SpannerMatrix}^{-1}$$

Just to keep everything from bouncing off the screen, use this shorthand:
SM = SpannerMatrix
e[1] = eigenvalue[1]
e[2] = eigenvalue[2].

In this shorthand, a 2D diagonalizable matrix A is of the form

$$A = SM.\begin{pmatrix} e[1] & 0 \\ 0 & e[2] \end{pmatrix}.SM^{-1}.$$

This gives:

$$A\,t = t\,SM.\begin{pmatrix} e[1] & 0 \\ 0 & e[2] \end{pmatrix}.SM^{-1} = SM.\begin{pmatrix} t\,e[1] & 0 \\ 0 & t\,e[2] \end{pmatrix}.SM^{-1}$$

So.

$$E^{A\,t} = SM.\begin{pmatrix} E^{t\,e[1]} & 0 \\ 0 & E^{t\,e[2]} \end{pmatrix}.SM^{-1}.$$

And so

$$A.E^{A\,t} = SM.\begin{pmatrix} e[1] & 0 \\ 0 & e[2] \end{pmatrix}.SM^{-1}.SM.\begin{pmatrix} E^{t\,e[1]} & 0 \\ 0 & E^{t\,e[2]} \end{pmatrix}.SM^{-1}$$

while

$$E^{A\,t}.A = SM.\begin{pmatrix} E^{t\,e[1]} & 0 \\ 0 & E^{t\,e[2]} \end{pmatrix}.SM^{-1}.SM.\begin{pmatrix} e[1] & 0 \\ 0 & e[2] \end{pmatrix}.SM^{-1}.$$

Your job is to take over and finish this explanation of why
$A.E^{A\,t} = E^{A\,t}.A$.

□G.11.b.ii) Simultaneously diagonalizable

Given a diagonalizable 2D matrix A, then you are guaranteed a basis of 2D
{eigenvectA[1], eigenvectA[2]}
of unit eigenvectors of A.
Given another diagonalizable 2D matrix B, then you are guaranteed a basis of 2D
{eigenvectB[1], eigenvectB[2]}
of unit eigenvectors of B.

Old matrix folks know that if A and B are simultaneously diagonalizable in the sense that
eigenvectA[1] = eigenvectB[1]
and
eigenvectA[2] = eigenvectB[2],
then it is certain that
A.B = B.A.
Your job is to explain why this is certain.

Click on the right for a tip.

$$A = \text{SpannerMatrixA}.\begin{pmatrix} \text{eigenvalA}[1] & 0 \\ 0 & \text{eigenvalA}[2] \end{pmatrix}.\text{SpannerMatrixA}^{-1}$$

Here eigenvectA[1] is the first vertical column of SpannerMatrixA and eigenvectA[2] is the second vertical column of SpannerMatrixA.

$$B = \text{SpannerMatrixB}.\begin{pmatrix} \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalB}[2] \end{pmatrix}.\text{SpannerMatrixB}^{-1}$$

Here eigenvectB[1] is the first vertical column of SpannerMatrixB and eigenvectB[2] is the second vertical column of SpannerMatrixB.

Because

eigenvectA[1] = eigenvectB[1]

and

eigenvectA[2] = eigenvectB[2],

you are certain that

SpannerMatrixA = SpannerMatrixB.

Just for shorthand put

SM = SpannerMatrixA = SpannerMatrixB.

In this shorthand,

$$A = SM.\begin{pmatrix} \text{eigenvalA}[1] & 0 \\ 0 & \text{eigenvalA}[2] \end{pmatrix}.SM^{-1}$$

and

$$B = SM.\begin{pmatrix} \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalB}[2] \end{pmatrix}.SM^{-1}.$$

Now you take over and finish the explanation of why A.B = B.A.

□G.11.b.iii) If A is diagonalizable, then A and $E^{A\,t}$ are simultaneously diagonalizable

Given a diagonalizable 2D matrix A, explain why A and $E^{A\,t}$ are simultaneously diagonalizable.

G.11.b.iv) Calculus Cal says $E^{A+B} = E^A.E^B$

One day while you are sitting at your favorite machine, that repulsive Calculus Cal comes in and says:"Given two 2D matrices A and B, you can be sure that $E^{A+B} = E^A.E^B$."

You say:"Where did you get that idea?"

Cal says:"For numbers a and b, everyone knows that $E^{a+b} = E^a\,E^b$ is a basic identity and because matrices are made of numbers, it follows that $E^{A+B} = E^A.E^B$ for any 2D matrices A and B."

You decide to check this out:

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

$$\begin{pmatrix} 0.442949 & 1.79593 \\ -1.43127 & 0.746664 \end{pmatrix}$$

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

$$\begin{pmatrix} 0.999304 & -1.02814 \\ 0.214904 & -1.61429 \end{pmatrix}$$

Here is E^{A+B}:

```
MatrixForm[MatrixExp[A + B]]
```

$$\begin{pmatrix} 3.25243 & 1.09294 \\ -1.73148 & -0.0356465 \end{pmatrix}$$

Here is $E^A.E^B$:

```
MatrixForm[MatrixExp[A].MatrixExp[B]]
```

$$\begin{pmatrix} -0.152665 & 0.48577 \\ -4.15152 & 1.57351 \end{pmatrix}$$

Rerun a couple of times and then tell Cal what you think of his theory.

G.11.b.v) If A and B are simultaneously diagonalizable, then $E^{A+B} = E^A.E^B$.

Given a diagonalizable 2D matrix A, then you are guaranteed a basis of 2D

{eigenvectA[1], eigenvectA[2]}

of unit eigenvectors of A.

Given another diagonalizable 2D matrix B, then you are guaranteed a basis of 2D

{eigenvectB[1], eigenvectB[2]}

of unit eigenvectors of B.

Old matrix folks know that A and B are simultaneously diagonalizable in the sense that

eigenvectA[1] = eigenvectB[1]

and

eigenvectA[2] = eigenvectB[2],

then it is certain that

$E^{A+B} = E^A.E^B$.

Your job is to explain why this is certain.

Click on the right for a big start.

$$A = \text{SpannerMatrixA}.\begin{pmatrix} \text{eigenvalA}[1] & 0 \\ 0 & \text{eigenvalA}[2] \end{pmatrix}.\text{SpannerMatrixA}^{-1}$$

Here eigenvectA[1] is the first vertical column of SpannerMatrixA
and eigenvectA[2] is the second vertical column of SpannerMatrixA.

$$B = \text{SpannerMatrixB}.\begin{pmatrix} \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalB}[2] \end{pmatrix}.\text{SpannerMatrixB}^{-1}$$

Here eigenvectB[1] is the first vertical column of SpannerMatrixB
and eigenvectB[2] is the second vertical column of SpannerMatrixB.

Because

eigenvectA[1] = eigenvectB[1]

and

eigenvectA[2] = eigenvectB[2],

you are certain that

SpannerMatrixA = SpannerMatrixB.

Just for shorthand put

SM = SpannerMatrixA = SpannerMatrixB.

In this shorthand,

$$A = \text{SM}.\begin{pmatrix} \text{eigenvalA}[1] & 0 \\ 0 & \text{eigenvalA}[2] \end{pmatrix}.\text{SM}^{-1}$$

and

$$B = \text{SM}.\begin{pmatrix} \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalB}[2] \end{pmatrix}.\text{SM}^{-1}.$$

In this shorthand,

$$E^A = \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[2]} \end{pmatrix}.\text{SM}^{-1}$$

and

$$E^B = \text{SM}.\begin{pmatrix} E^{\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalB}[2]} \end{pmatrix}.\text{SM}^{-1}.$$

So

$$E^A.E^B$$

$$= \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[2]} \end{pmatrix}.\text{SM}^{-1}.\text{SM}.\begin{pmatrix} E^{\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalB}[2]} \end{pmatrix}.\text{SM}^{-1}$$

$$= \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[2]} \end{pmatrix}.\begin{pmatrix} E^{\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalB}[2]} \end{pmatrix}.\text{SM}^{-1}$$

$$= \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]}\,E^{\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[1]}\,E^{\text{eigenvalA}[2]} \end{pmatrix}.\text{SM}^{-1}$$

$$= \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]+\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[2]+\text{eigenvalB}[2]} \end{pmatrix}.\text{SM}^{-1}.$$

Once you know that

$$A + B = \text{SM}.\begin{pmatrix} \text{eigenvalA}[1] + \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalA}[2] + \text{eigenvalB}[2] \end{pmatrix}.\text{SM}^{-1}.$$

Then you will know that

$$E^{A+B} = \text{SM}.\begin{pmatrix} E^{\text{eigenvalA}[1]+\text{eigenvalB}[1]} & 0 \\ 0 & E^{\text{eigenvalA}[2]+\text{eigenvalB}[2]} \end{pmatrix}.\text{SM}^{-1}$$

$$= E^A\,E^B.$$

To know that

$$A + B = \text{SM}.\begin{pmatrix} \text{eigenvalA}[1] + \text{eigenvalB}[1] & 0 \\ 0 & \text{eigenvalA}[2] + \text{eigenvalB}[2] \end{pmatrix}.\text{SM}^{-1},$$

all you need to explain is:

1) Why

X = eigenvectA[1] = eigenvectB[1]

is an eigenvector of A + B and why

(A + B).X = (eigenvalA[1] + eigenvalB[1]) X

and

2) Why

Y = eigenvectA[2] = eigenvectB[2]

is an eigenvector of A + B and why

(A + B).X = (eigenvalA[2] + eigenvalB[2]) X

Now you take over and explain why these two.