Lecture 7: Linear Algebra

Reading:
Kreyszig Sections: 13.1, 13.2, 13.5, 13.6 (pages602–606, 607–611, 623–626, 626–629)

Uniqueness and Existence of Linear System Solutions

It would be useful to use the Mathematica Help Browser and look through the section in the Mathematica Book: Advanced Mathematics/ Linear Algebra/Solving Linear Equations

A linear system of m equations in n variables $(x_1, x_2, \ldots, x_n)$ takes the form

$$\begin{aligned} A_{11}x_1 + A_{12}x_2 + A_{13}x_3 + \ldots + A_{1n}x_n &= b_1 \\ A_{21}x_1 + A_{22}x_2 + A_{23}x_3 + \ldots + A_{2n}x_n &= b_2 \\ \vdots &= \vdots \\ A_{k1}x_1 + A_{k2}x_2 + A_{k3}x_3 + \ldots + A_{kn}x_n &= b_k \\ \vdots &= \vdots \\ A_{m1}x_1 + A_{m2}x_2 + A_{m3}x_3 + \ldots + A_{mn}x_n &= b_m \end{aligned} \tag{7-1}$$

and is fundamental to models of many systems.

The coefficients, A_{ij}, form a matrix and such equations are often written in an "index" short-hand known as the Einstein summation convention:

$$A_{ij}x_i = b_j \qquad \text{(Einstein summation convention)} \tag{7-2}$$

where *if an index* (here i) *is repeated in any set of multiplied terms,* (here $A_{ij}x_i$) *then a summation over all values of that index is implied.* Note that, by multiplying and summing in Eq. 7-2, the repeated

3.016 Home

◀◀ ◀ ▶ ▶▶

Full Screen

Close

Quit

index i disappears from the right-hand-side. It can be said that the repeated index in "contracted" out of the equation and this idea is emphasized by writing Eq. 7-2 as

$$x_i A_{ij} = b_j \qquad \text{(Einstein summation convention)} \tag{7-3}$$

so that the "touching" index, i, is contracted out leaving a matching j-index on each side. In each case, Eqs. 7-2 and 7-3 represent m equations $(j = 1, 2, \ldots, m)$ in the n variables $(i = 1, 2, \ldots, n)$ that are contracted out in *each equation*. The summation convention is especially useful when the dimensions of the coefficient matrix is larger than two; for a linear elastic material, the elastic energy density can be written as:

$$E_{\text{elast}} = \frac{1}{2}\epsilon_{ij} C_{ijkl} \epsilon_{kl} = \frac{1}{2}\sigma_{pq} S_{pqrs} \sigma_{rs} \tag{7-4}$$

where C_{ijkl} and ϵ_{ij} are the fourth-rank *stiffness* tensor and second-rank elastic *strain* tensor; where S_{ijkl} and σ_{ij} are the fourth-rank compliance tensor and second-rank stress tensor;

In physical problems, the goal is typically to find the n x_i for a given m b_j in Eqs. 7-2, 7-3, or 7-1. This goal is conveniently represented in matrix-vector notation:

$$\underline{A}\vec{x} = \vec{b} \tag{7-5}$$

Lecture 07 MATHEMATICA® Example 1

Solving Linear Sets of Equations

notebook (non-evaluated) pdf (evaluated) html (evaluated)

This example is kind of backwards. We will take a matrix

$$\underline{A} = \begin{pmatrix} 1 & 2 & 1 & 1 \\ -1 & 4 & -2 & 0 \\ 1 & 2 & 4 & 5 \\ 1 & 0 & 1 & 1 \end{pmatrix} \text{ unknown vector } \vec{x} = \begin{pmatrix} x \\ y \\ z \\ t \end{pmatrix} \text{ and known vector } \vec{b} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} \tag{7-6}$$

and extract four equations for input to `Solve` to obtain the solution to $\vec{x}$.

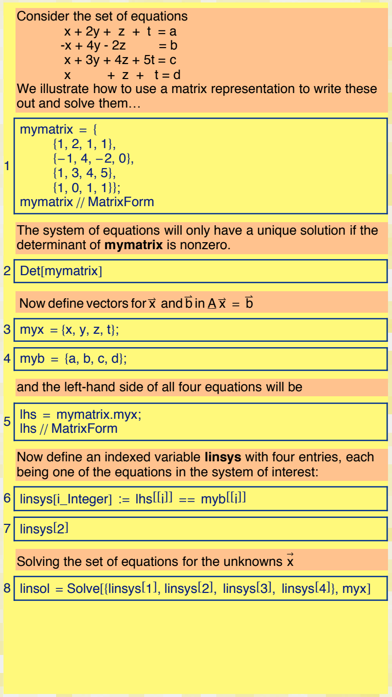
1: The coefficient matrix is a list of row-lists.

2: A unique solution will exist if the determinant, computed with `Det`, is non-zero.

5: The left-hand-side is a column-vector with four entries.

6: This function creates *logical equalities* for each corresponding entry on the left- and right-hand-sides.

8: Here, the function creates the input four equations and the `myx` contains the unknowns.



Consider the set of equations

```
 x + 2y +  z  +  t  = a
-x + 4y - 2z        = b
 x + 3y + 4z + 5t = c
 x          + z  +   t = d
```

We illustrate how to use a matrix representation to write these out and solve them…

```
1  mymatrix = {
          {1, 2, 1, 1},
          {−1, 4, −2, 0},
          {1, 3, 4, 5},
          {1, 0, 1, 1}};
   mymatrix // MatrixForm
```

The system of equations will only have a unique solution if the determinant of **mymatrix** is nonzero.

```
2  Det[mymatrix]
```

Now define vectors for $\vec{x}$ and $\vec{b}$ in $\underline{A}\,\vec{x} = \vec{b}$

```
3  myx = {x, y, z, t};
```

```
4  myb = {a, b, c, d};
```

and the left-hand side of all four equations will be

```
5  lhs = mymatrix.myx;
   lhs // MatrixForm
```

Now define an indexed variable **linsys** with four entries, each being one of the equations in the system of interest:

```
6  linsys[i_Integer] := lhs[[i]] == myb[[i]]
```

```
7  linsys[2]
```

Solving the set of equations for the unknowns $\vec{x}$

```
8  linsol = Solve[{linsys[1], linsys[2], linsys[3], linsys[4]}, myx]
```

3.016 Home

Full Screen

Close

Quit

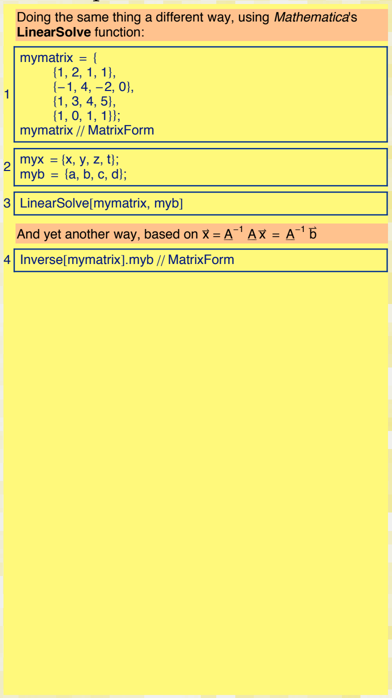
Inverting Matrices or Just Solving for the Unknown Vector

notebook (non-evaluated) pdf (evaluated) html (evaluated)

Continuing the last example, it is much more compact to invert a matrix symbolically or numerically. If a matrix inverse is going to be used over and over again, it is probably best to compute and store the inverse once. However, if a one-time only solution for $\vec{x}$ in $\underline{A}\vec{x} = \vec{b}$ is needed, then computing the inverse is computationally less efficient than using an algorithm designed to solve for $\vec{x}$ directly. Here is an example of both methods.

3: `LinearSolve` can take two arguments, $\underline{A}$ and $\vec{b}$, and returns $\vec{x}$ that solves $\underline{A}\vec{x} = \vec{b}$. It will be noticibly faster than the following inversion method, especially for large matrices.

4: The matrix inverse is obtained with `Inverse` and a subsequent multiplication by the right-hand-side gives the solution.



Doing the same thing a different way, using *Mathematica*'s **LinearSolve** function:

1
```
mymatrix = {
      {1, 2, 1, 1},
      {-1, 4, -2, 0},
      {1, 3, 4, 5},
      {1, 0, 1, 1}};
mymatrix // MatrixForm
```

2
```
myx = {x, y, z, t};
myb = {a, b, c, d};
```

3
```
LinearSolve[mymatrix, myb]
```

And yet another way, based on $\vec{x} = \underline{A}^{-1}\,\underline{A}\,\vec{x} = \underline{A}^{-1}\,\vec{b}$

4
```
Inverse[mymatrix].myb // MatrixForm
```

3.016 Home

Full Screen

Close

Quit

Uniqueness of solutions to the nonhomogeneous system

$$\underline{A}\vec{x} = \vec{b} \tag{7-7}$$

Uniqueness of solutions to the homogeneous system

$$\underline{A}\vec{x_o} = \vec{0} \tag{7-8}$$

Adding solutions from the nonhomogeneous and homogenous systems

You can add any solution to the homogeneous equation (if they exist there are infinitely many of them) to any solution to the nonhomogeneous equation and the result is still a solution to the nonhomogeneous equation.

$$\underline{A}(\vec{x} + \vec{x_o}) = \vec{b} \tag{7-9}$$

Lecture 07 MATHEMATICA® Example 3

Determinants, Rank, and Nullity

notebook (non-evaluated) pdf (evaluated) html (evaluated)

Several examples of determinant calculations are provided to illustrate the properties of determinants. When a determinant vanishes (i.e., det $\underline{A} = 0$, there is no solution to the inhomogeneous equation det $\underline{A} = \vec{b}$, but there will be an infinity of solutions to det $\underline{A} = 0$; the infinity of solutions can be characterized by solving for a number *rank* of the entries of $\vec{x}$ in terms of the *nullity* of other entries of $\vec{x}$

1: A matrix is created where the third row is the sum of $p \times$ first row, $q \times$ second row, and $r \times$ fourth row. In other words, one row is a linear combination of the others.

2: The determinant is computed with `Det`.

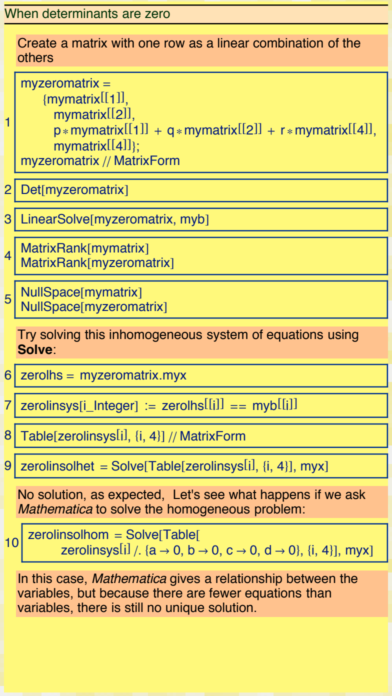
3: An attempt to solve the linear inhomogeneous equation should fail.

4: When the determinant is zero, there may still be some linearly independent rows or columns. The rank gives the number of linearly independent rows or columns and is computed with `MatrixRank`.

5: The *null space* of a matrix, $\underline{A}$, is a set of linearly independent vectors that, if left-multiplied by $\underline{A}$ gives a zero vector. The nullity is how many linearly independent vectors there are in the null space. Sometimes, vectors in the null space are called *killing vectors*.

9: Here, an attempt to use `Solve` for the heterogeneous system is attempted, but of course it is bound to fail...

10: However, this is the solution the singular homogeneous problem ($\underline{A}\vec{x} = \vec{0}$, where det $\underline{A} = 0$. The solution is three (the rank) dimensional surface embedded in four dimensions (the rank plus the nullity). Notice that the solution is a multiple of the null space.



When determinants are zero

Create a matrix with one row as a linear combination of the others

```
1  myzeromatrix =
       {mymatrix[[1]],
         mymatrix[[2]],
         p*mymatrix[[1]] + q*mymatrix[[2]] + r*mymatrix[[4]],
         mymatrix[[4]]};
   myzeromatrix // MatrixForm

2  Det[myzeromatrix]

3  LinearSolve[myzeromatrix, myb]

4  MatrixRank[mymatrix]
   MatrixRank[myzeromatrix]

5  NullSpace[mymatrix]
   NullSpace[myzeromatrix]
```

Try solving this inhomogeneous system of equations using **Solve**:

```
6  zerolhs = myzeromatrix.myx

7  zerolinsys[i_Integer] := zerolhs[[i]] == myb[[i]]

8  Table[zerolinsys[i], {i, 4}] // MatrixForm

9  zerolinsolhet = Solve[Table[zerolinsys[i], {i, 4}], myx]
```

No solution, as expected, Let's see what happens if we ask *Mathematica* to solve the homogeneous problem:

```
10 zerolinsolhom = Solve[Table[
          zerolinsys[i] /. {a → 0, b → 0, c → 0, d → 0}, {i, 4}], myx]
```

In this case, *Mathematica* gives a relationship between the variables, but because there are fewer equations than variables, there is still no unique solution.

3.016 Home

Full Screen

Close

Quit

Properties and Roles of the Matrix Determinant

In example 07-1, it was stated (item 2) that a unique solution exists if the matrix's determinant was non-zero. The solution,

$$\vec{x} = \begin{pmatrix} \frac{2a+2b-4c+18d}{\det \underline{A}} \\ \frac{7a-7d}{\det \underline{A}} \\ \frac{13a-8b+2c-23d}{\det \underline{A}} \\ \frac{-15a+6b+2c+19d}{\det \underline{A}} \end{pmatrix} \tag{7-10}$$

indicates why this is the case and also illustrates the role that the determinant plays in the solution. Clearly, if the determinant vanishes, then the solution is undetermined unless $\vec{b}$ is a zero-vector $\vec{0} = (0,0,0,0)$. Considering the *algebraic equation*, $ax = b$, the determinant plays the role for matrices that the condition $a = 0$ plays for algebra: the inverse exists when $a \neq 0$ or $\det \underline{A} \neq 0$.

The determinant is only defined for square matrices; it derives from the elimination of the n unknown entries in $\vec{x}$ using all n equation (or rows) of

$$\underline{A}\vec{x} = 0 \tag{7-11}$$

For example, eliminating x and y from

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \text{gives the expression}$$
$$\det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \equiv a_{11}a_{22} - a_{12}a_{21} = 0 \tag{7-12}$$

and eliminating x, y, and z from

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

gives the expression

$$\det\underline{A} \equiv a_{11}a_{22}a_{33} - a_{11}a_{32}a_{23} + a_{21}a_{32}a_{13} - a_{21}a12a_{33} + a_{31}a_{12}a_{23} - a_{31}a_{22}a_{13} = 0 \tag{7-13}$$

3.016 Home

Full Screen

Close

Quit

©W. Craig Carter

The following general and true statements about determinants are plausible given the above expressions:

- The terms in the determinant's sum are products of a terms; one term comes from each column.
- Each term is one of all possible the products of an entry from each column.
- There is a plus or minus in front each term in the sum, $(-1)^p$, where p is the number of *neighbor exchanges required to put the rows in order* in each term written as an ordered product of their columns (as in example Eqs. 7-12 and 7-13).

These and the observation that it is impossible to eliminate $\vec{x}$ in Eqs. 7-12 and 7-13 if the information in the rows is redundant (i.e., there is not enough information—or independent equations—to solve for the $\vec{x}$) yield the general properties of determinants that are illustrated in the following example.

Properties of Determinants

notebook (non-evaluated) pdf (evaluated) html (evaluated)

Rules corresponding how det $\underline{A}$ changes when the columns of $\underline{A}$ are permuted, or multiplied by a constant are demonstrated, along with $\det(\underline{AB}) = \det \underline{A} \det \underline{B}$ and $\underline{AB} \neq \underline{BA}$.

2: A matrix with random real entries between -1 and 1 is created.

5: Multiplying *one* column of a matrix by a constant a, multiplies the matrix's determinant by *one factor of* a.

7: Because the matrix has one linearly-dependent column, its determinant should vanish. This example demonstrates what happens with limited numerical precision operations on real numbers. The determinant is not zero, but could be considered *effectively zero.*

8: Problems with numerical imprecision can usually be alleviated with `Chop` which sets small magnitude numbers to zero.

10: Using, `Permutations`, create all possible permutations of two sets of three identical objects for subsequent construction of a symbolic matrix.

12: The symbolic matrix has a fairly simple determinant.

13: A matrix with random rational numbers is created...

14: And, of course, its determinant is also a rational number.

16: This demonstrates that determinant of a product is the product of determinants...

18: And, this (reduntantly) shows that the order of matrix multiplication does not affect the product rule for determinants.

19: However, the result of multiplying two matrix *does* depend on the order of multiplication: $\underline{AB} \neq \underline{BA}$, in general.

```
1  rv[i_] := rv[i] = Table[Random[Real, {-1, 1}], {j, 6}]

2  RandMat = Table[rv[i], {i, 6}]

3  Det[RandMat]

4  Det[{rv[2], rv[1], rv[3], rv[4], rv[5], rv[6]}]

5  Det[{a*rv[2], rv[1], rv[3], rv[4], rv[5], rv[6]}]

6  LiDepVec =
     a*rv[1] + b*rv[2] + c*rv[3] + d*rv[4] + e*rv[5]

7  Det[{rv[1], rv[2], rv[3], rv[4], rv[5], LinDepVec}]

8  Chop[Det[{rv[1], rv[2], rv[3], rv[4], rv[5], LinDepVec}]]

9  SymVec = {a, a, a, c, c, c};

10 Permuts = Permutations[SymVec]
   Permuts // Dimensions

11 SymbMat = {Permuts[[1]], Permuts[[12]], Permuts[[6]],
               Permuts[[18]], Permuts[[17]], Permuts[[9]]};
   SymbMat // MatrixForm

12 DetSymbMat = Simplify[Det[SymbMat]]

13 RandomMat = Table[
        Table[Random[Integer, {-100, 100}]
              ----------------------------- , {i, 6}], {j, 6}];
              Random[Integer, {-100, 100}]
   RandomMat // MatrixForm

14 DetRandomMat = Det[RandomMat]

15 CheckA = Det[SymbMat.RandomMat] // Simplify

16 DetRandomMat*DetSymbMat == CheckA

17 CheckB = Det[RandomMat.SymbMat] // Simplify

18 CheckA == CheckB

19 (RandomMat.SymbMat - SymbMat.RandomMat) //
      Simplify // MatrixForm
```

3.016 Home

Full Screen

Close

Quit

Vector Spaces

Consider the position vector

$$\vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{7-14}$$

The vectors $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$ can be used to generate any general position by suitable scalar multiplication and vector addition:

$$\vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = x \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + y \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} + z \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{7-15}$$

Thus, three dimensional real space is "spanned" by the three vectors: $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$. These three vectors are candidates as "basis vectors for $\Re^3$."

Consider the vectors $(a,-a,0)$, $(a,a,0)$, and $(0,a,a)$ for real $a \neq 0$.

$$\vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \frac{x+y}{2a} \begin{pmatrix} a \\ -a \\ 0 \end{pmatrix} + \frac{x-y}{2a} \begin{pmatrix} a \\ a \\ 0 \end{pmatrix} + \frac{x-y+2z}{2a} \begin{pmatrix} 0 \\ a \\ a \end{pmatrix} \tag{7-16}$$

So $(a,-a,0)$, $(a,a,0)$, and $(0,a,a)$ for real $a \neq 0$ also are basis vectors and can be used to span $\Re^3$.

The idea of basis vectors and vector spaces comes up frequently in the mathematics of materials science. They can represent abstract concepts as well as shown by the following two dimensional basis set:

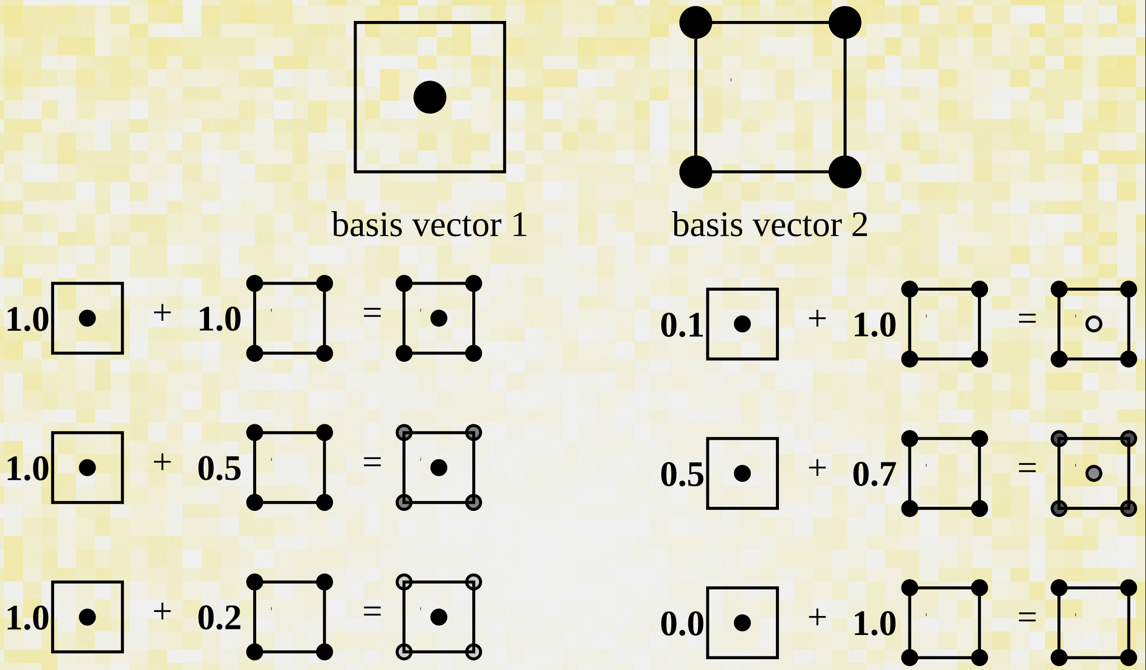


Figure 7-2: A vector space for two-dimensional CsCl structures. Any combination of center-site concentration and corner-site concentration can be represented by the sum of two basis vectors (or basis lattice). The set of all grey-grey patterns is a vector space of patterns.

Linear Transformations

Lecture 07 MATHEMATICA® Example 5

Visualization of linear transformations

notebook (non-evaluated) pdf (evaluated) html (evaluated)

An simple octagon with different colored faces is transformed by operating on all of it vertices with a matrix. This example demonstrates how symmetry operations, like rotations reflections, can be represented as a matrix multiplication, and how to visualize the results of linear transformations generally.

1: The package Polyhedra contains Graphics Objects with the coordinates of many common polyhedra.

2: This demonstrates how an Octahedron can be drawn on the screen.

3: The ViewPoint option to Show allows viewing from different points in 3D space.

4: InputForm reveals how the coordinates of the polydra are stored. . .

5: And, this can be mimicked to create a face-colored polyhedron with the Hue graphics directive.

7: This is a matric which would create the mirror image across the z-axis of any point it multiplies.

8: This is a moderately sophisticated example of rule usage: it looks for triangles (Polygons with three points); names the points; and then multiplies a matrix by each of the points. The result in this case is a mirror operation.

9: This generalizes the previous example, by creating a function that takes a matrix as an argument.

11: This visualizes a rotation of $\pi/4$ around the z-axis.

12: This mirrors across the x- and y-axis and performs a linear expansion by a factor of 5 along the z-direction. The octagon volume increases by the determinant of the transformation matrix.

```
1  << Graphics`Polyhedra`

2  Show[Polyhedron[Octahedron]]

3  Show[Polyhedron[Octahedron],
     ViewPoint −> {−0.007, −1.995, −0.135}]

4  Polyhedron[Octahedron] // InputForm

5  ColOct = Graphics3D[{
           {Hue[0/8], Polygon[{{0, 0, 1}, {1, 0, 0.}, {0, 1, 0.}}]},
           {Hue[1/8], Polygon[{{0, 0, 1}, {0, 1, 0}, {−1, 0, 0}}]},
           {Hue[2/8], Polygon[{{0, 0, 1}, {−1, 0, 0}, {0, −1, 0}}]},
           {Hue[3/8], Polygon[{{0, 0, 1}, {0, −1, 0}, {1, 0, 0}}]},
           {Hue[4/8], Polygon[{{1, 0, 0}, {0, −1, 0}, {0, 0, −1}}]},
           {Hue[5/8], Polygon[{{1, 0, 0}, {0, 0, −1}, {0, 1, 0}}]},
           {Hue[6/8], Polygon[{{0, 0, −1}, {0, −1, 0}, {−1, 0, 0}}]},
           {Hue[7/8], Polygon[{{0, 1, 0}, {0, 0, −1}, {−1, 0, 0}}]}
       }]

6  Show[ColOct, Lighting → False]

7  tmat = {{1, 0, 0}, {0, 1, 0}, {0, 0, −1}};
   tmat // MatrixForm

8  Show[ColOct /. {Polygon[{a_List , b_List , c_List}] →
          Polygon[{tmat.a, tmat.b, tmat.c}]}, Lighting → False]

9  seetrans[tranmat_] :=
     Show[ColOct /. {Polygon[{a_List , b_List , c_List}] →
          Polygon[{tranmat.a, tranmat.b, tranmat.c}]},
       Lighting → False]

10 seetrans[{{1, 0, 0}, {0, 1, 0}, {0, 0, 1}}]

11 seetrans[{{Cos[Pi/4], Sin[Pi/4], 0},
       {Sin[−Pi/4], Cos[Pi/4], 0}, {0, 0, 1}}]

12 seetrans[{{−1, 0, 0}, {0, −1, 0}, {0, 0, 5}}]
```

3.016 Home

Full Screen

Close

Quit

Index

3.016 Home

Full Screen

Close

Quit