Polar Decomposition Example: Matrix Notation

Define a matrix **F** to decompose.

```
F = Array[f, {3, 3}];
f[1, 1] = 2; f[1, 2] = 3; f[1, 3] = 5;
f[2, 1] = f[3, 1] = f[3, 2] = 0;
f[2, 2] = 4; f[2, 3] = 6; f[3, 3] = 7;
F // MatrixForm
```

```
( 2   3   5 )
( 0   4   6 )
( 0   0   7 )
```

Check that it is nonsingular by calculating its determinant.

```
Det[F]
```

```
56
```

Form **FT F**. I denote that by CC because **C** is a reserved word in *Mathematica*. The Transpose operator calculates the transpose of a matrix, and the dot means matrix multiplication.

```
CC = Transpose[F].F;
CC // MatrixForm
```

```
( 4    6    10  )
( 6    25   39  )
( 10   39   110 )
```

This is a symmetric, positive definite matrix. The symmetry is obvious by inspection. We will determine the eigenvalues and discover that they are all positive, which will verify that the matrix is positive definite.
I need a 3x3 identity matrix, and the next thing I do is define that.

```
I33 = Array[i33, {3, 3}];
Do[i33[i, j] = 0, {i, 1, 3}, {j, 1, 3}]
Do[i33[i, i] = 1, {i, 1, 3}]
I33 // MatrixForm
```

```
( 1   0   0 )
( 0   1   0 )
( 0   0   1 )
```

The eigenvalue relation comes from the usual formula. I denote the eigenvalues by λ^2 because I intend to take a square root eventually. (Note that *Mathematica* can find the eigenvalues and eigenvectors directly, but I'd rather run through the process to reinforce what is happening.) So, I calculate the matrix equation

```
EVR = CC - λ^2 I33;
EVR // MatrixForm
```

$$\begin{pmatrix} 4-\lambda^2 & 6 & 10 \\ 6 & 25-\lambda^2 & 39 \\ 10 & 39 & 110-\lambda^2 \end{pmatrix}$$

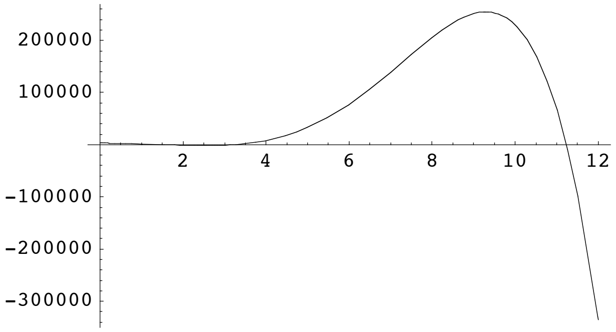
Take the determinant (letting *Mathematica* do that for me), and write the characteristic polynomial, simplifying it for easy reading.

```
Det[EVR];
cp = Collect[%, λ, Simplify]
```

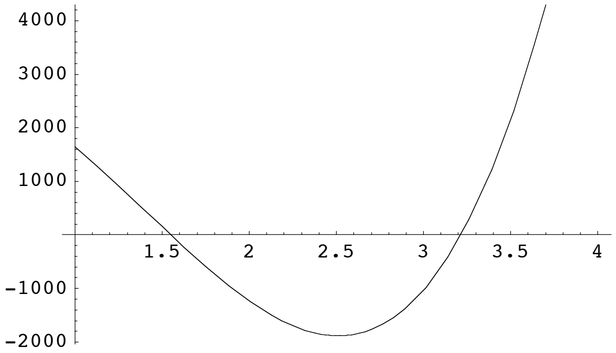
$$3136 - 1633\,\lambda^2 + 139\,\lambda^4 - \lambda^6$$

Mathematica can probably find the roots for me, but we'll be better off looking for them graphically and then finding them numerically. The two plots help me to get close.

```
Plot[cp, {λ, 0, 12}]
Plot[cp, {λ, 1, 4}]
```



```
- Graphics -
```



```
- Graphics -
```

Then I can use *Mathematica*'s FindRoot command, which searches for a root of the specified unknown in the specified interval. I denote the roots by $\lambda 1$, $\lambda 2$, $\lambda 3$.

```
FindRoot[cp == 0, {λ, 1.5, 1.6}]
```

```
{λ → 1.55038}
```

/. is a substitution operator. % means the previous answer, and [[1]] means the first element of the answer, here the term in braces. The statement means that λ1 is obtained by taking the general expression λ and substituting the value we found using FindRoot.

```
λ1 = λ /. %[[1]]
```

```
1.55038
```

Repeat for the other two roots.

```
FindRoot[cp == 0, {λ, 3.1, 3.3}]
```

```
{λ → 3.21448}
```

```
λ2 = λ /. %[[1]]
```

```
3.21448
```

```
FindRoot[cp == 0, {λ, 11, 12}]
```

```
{λ → 11.2367}
```

```
λ3 = λ /. %[[1]]
```

```
11.2367
```

The following is a clever way to get eigenvectors for three dimensional systems. Note that the eigenvectors must be perpendicular to all three rows of the matrix EVR. We can find a general vector that is perpendicular to the first two rows by simply using the cross product of those two rows. This will hold for all values of λ, not just the eigenvalue. The condition that the vector is perpendicular to the third row of EVR turns out to be identical to the eigenvalue equation, and so it satisfied if λ is equal to any of the eigenvectors.
genEV is a vector that is perpendicular to the first two rows of EVR.

```
Clear[v1, v2]; V1 = Array[v1, 3];
Do[v1[i] = EVR[[1, i]], {i, 1, 3}]
V2 = Array[v2, 3];
Do[v2[i] = EVR[[2, i]], {i, 1, 3}]
genEV = Cross[V1, V2]
```

$\{-16 + 10\,\lambda^2,\ -96 + 39\,\lambda^2,\ 64 - 29\,\lambda^2 + \lambda^4\}$

ev1, ev2 and ev2 are realizations of genEV for the specific values of λ

```
ev1 = genEV /. λ → λ1
ev2 = genEV /. λ → λ2
ev3 = genEV /. λ → λ3
```

```
{8.0368, -2.25648, 0.0709563}

{87.3289, 306.983, -128.885}

{1246.63, 4828.27, 12344.8}
```

In order to form the matrix **P** we need to have normalized eigenvectors, and the next steps normalize ev1, ev2 and ev3 (and replaces the unnormalized vectors by their normalized counterpart).

```
den = 0;
Do[den += ev1[[i]]^2, {i, 1, 3}]
den = Sqrt[den];
Do[ev1[[i]] /= den, {i, 1, 3}];
ev1 // MatrixForm
```

```
( 0.962737   )
( -0.270306  )
( 0.00849993 )
```

```
den = 0;
Do[den += ev2[[i]]^2, {i, 1, 3}]
den = Sqrt[den];
Do[ev2[[i]] /= den, {i, 1, 3}];
ev2 // MatrixForm
den = 0;
Do[den += ev3[[i]]^2, {i, 1, 3}]
den = Sqrt[den];
Do[ev3[[i]] /= den, {i, 1, 3}];
ev3 // MatrixForm
```

```
( 0.253713  )
( 0.891864  )
( -0.374445 )

( 0.0936338 )
( 0.362648  )
( 0.92721   )
```

The transpose of **P** is made up of the eigenvectors as columns. **P** is thus the matrix made up of the eigenvectors as rows, and that can be defined conveniently in *Mathematica* as follows: **P** is an array of row vectors.

```
P = {ev1, ev2, ev3};
P // MatrixForm
```

```
( 0.962737   -0.270306  0.00849993 )
( 0.253713   0.891864   -0.374445  )
( 0.0936338  0.362648   0.92721    )
```

I need the diagonal matrix of the eigenvalues, which I will call Λ

```
Clear[λ];
Λ = Array[λ, {3, 3}];
Do[λ[i, j] = 0, {i, 1, 3}, {j, 1, 3}]
λ[1, 1] = λ1;
λ[2, 2] = λ2;
λ[3, 3] = λ3;
Λ // MatrixForm
```

$$\begin{pmatrix} 1.55038 & 0 & 0 \\ 0 & 3.21448 & 0 \\ 0 & 0 & 11.2367 \end{pmatrix}$$

Its square enters the decomposition as well, so I calculate that

```
Λ2 = Λ.Λ;
Λ2 // MatrixForm
```

$$\begin{pmatrix} 2.40368 & 0. & 0. \\ 0. & 10.3329 & 0. \\ 0. & 0. & 126.263 \end{pmatrix}$$

Verify that we have properly decomposed **C**

```
Transpose[P].Λ2.P;
Simplify[%];
% // MatrixForm
```

$$\begin{pmatrix} 4. & 6. & 10. \\ 6. & 25. & 39. \\ 10. & 39. & 110. \end{pmatrix}$$

We can find the matrix **U**

```
U = Transpose[P].Λ.P;
% // MatrixForm
```

$$\begin{pmatrix} 1.74242 & 0.705458 & 0.682857 \\ 0.705458 & 4.14793 & 2.7013 \\ 0.682857 & 2.7013 & 10.1112 \end{pmatrix}$$

We need the inverse of **U**, which we can find directly, or we can find the inverse of **Λ** and construct the inverse of **U**. (The inverse of Λ is simple because Λ is a diagonal matrix, sI can calculate it directly or let *Mathematica* do it.)

```
Λm1 = Inverse[Λ];
Λm1 // MatrixForm
```

$$\begin{pmatrix} 0.645003 & 0. & 0. \\ 0. & 0.311092 & 0. \\ 0. & 0. & 0.0889941 \end{pmatrix}$$

and finally I can calculate the rotational matrix, **R**.

```
Um1 = Transpose[P].Λm1.P;
R = F.Um1;
R // MatrixForm
```

$$\begin{pmatrix} 0.871211 & 0.352729 & 0.341429 \\ -0.477044 & 0.772434 & 0.419255 \\ -0.115848 & -0.528136 & 0.841221 \end{pmatrix}$$

We check that **R** is orthogonal by verifying that its transpose is its inverse (within round-off error).

```
Simplify[Transpose[R].R];
% // MatrixForm
```

$$\begin{pmatrix} 1. & 1.03642\times 10^{-15} & -8.07669\times 10^{-15} \\ 1.03642\times 10^{-15} & 1. & 1.37101\times 10^{-15} \\ -8.07669\times 10^{-15} & 1.37101\times 10^{-15} & 1. \end{pmatrix}$$

Now we can calculate **V**.

```
V = F.Transpose[R];
% // MatrixForm
```

$$\begin{pmatrix} 4.50775 & 3.45949 & 2.39 \\ 3.45949 & 5.60526 & 2.93478 \\ 2.39 & 2.93478 & 5.88854 \end{pmatrix}$$

and verify that the two decompositions do indeed give the original matrix **F**. The following matrices should be zero, and are within reasonable round-off error.

```
Simplify[F - V.R] // MatrixForm
Simplify[F - R.U] // MatrixForm
```

$$\begin{pmatrix} 3.44169\times 10^{-14} & -6.21725\times 10^{-15} & 1.5099\times 10^{-14} \\ 4.39259\times 10^{-14} & -4.44089\times 10^{-15} & -1.77636\times 10^{-15} \\ 5.6414\times 10^{-14} & -9.54783\times 10^{-15} & 4.44089\times 10^{-15} \end{pmatrix}$$

$$\begin{pmatrix} 2.22045\times 10^{-15} & 8.88178\times 10^{-16} & 1.06581\times 10^{-14} \\ 5.89214\times 10^{-15} & 3.55271\times 10^{-15} & -7.99361\times 10^{-15} \\ 1.19727\times 10^{-14} & -1.34624\times 10^{-15} & -6.21725\times 10^{-15} \end{pmatrix}$$