Fourth Tutorial

In this tutorial we will learn how to use some LinearAlgebra commands to compute dot products, lengths and determinants. Also, we will see how to compute definite integrals and learn some tricks using the combinat package.

Start by loading the LinearAlgebra package

> with(LinearAlgebra):

Then define a couple of three dimensional vectors:

> v[1]:=RandomVector(3);
> v[2]:=RandomVector(3);

\[
\begin{bmatrix}
-79 \\
-71 \\
28 \\
-50 \\
30 \\
62
\end{bmatrix}
\]

Obviously, these vectors are not equal. But if you have to compare vectors with hundreds of coordinates the Equal command is handy:

> Equal(v[1],v[2]);

false

We use the DotProduct command to compute... the dot product:

> DotProduct(v[1],v[2]);

3556

How about finding the length of v[1]? A first approach is to use the definition:

> sqrt(DotProduct(v[1],v[1]));

\[ \sqrt{12066} \]

We could also use the Norm command that just evaluates to the length (aka norm) of the vector:

> Norm(v[1],2);

\[ \sqrt{12066} \]

Notice that the Norm command requires a second parameter, that we chose to be 2. If you don’t put a second parameter or you use a different value then Norm will not give you the length of the vector in the usual sense.

Now that we know how to compute lengths and dot products, how about finding the angle between vectors? We know that the cosine of the angle between v[1] and v[2] is

> cosa:=DotProduct(v[1],v[2])/(Norm(v[1],2)*Norm(v[2],2));

\[ \frac{889 \sqrt{12066} \sqrt{1811}}{10925763} \]

so that the actual angle is

> a:=arccos(cosa);

\[ \arccos \left( \frac{889 \sqrt{12066} \sqrt{1811}}{10925763} \right) \]
That is...

> evalf(a);

1.180614562

We could also write a function that, given two vectors, returns the angle between them:

> angle:=(v,w)->arccos(DotProduct(v,w)/(Norm(v,2)*Norm(w,2)));

so we can re-evaluate the angle between v[1] and v[2] with

> angle(v[1],v[2]);

\[
\arccos\left(\frac{889 \sqrt{12066} \sqrt{1811}}{10925763}\right)
\]

Determinants are computed in a very simple way, just use the \texttt{Determinant} command:

> A:=RandomMatrix(3,3);

\[
\begin{pmatrix}
20 & -34 & -21 \\
-7 & -62 & -56 \\
16 & -90 & -8
\end{pmatrix}
\]

> Determinant(A);

\[-92574\]

The \texttt{Determinant} command works for numeric matrices as well as for matrices including letters:

> B:=<<1-L,3>|<2,4-L>>;

\[
\begin{pmatrix}
1-L & 2 \\
3 & 4-L
\end{pmatrix}
\]

> Determinant(B);

\[-2 - 5 L + L^2\]

In some cases, especially for large matrices, it can help Maple do the computations faster if you tell the \texttt{Determinant} function what way to proceed. In the \texttt{Determinant} help page you can see how to do this. For example, if you know that all the entries of your matrix are integer numbers, you may want to use

> Determinant(A,method=integer);

\[-92574\]

Instead of the plain version.

Now we turn to some tricks that are useful when you have to consider lists of vectors. For example, our first task will be to construct all possible vectors in \(\mathbb{R}^2\) with components 1, 2 or 3. We start by defining a list of possible values:

> values:={1,2,3};

\[values := \{1, 2, 3\}\]

Next we load the \texttt{combinat} package:

> with(combinat):

Warning, the protected name Chi has been redefined and unprotected

If you remember the idea of cartesian product, one way of obtaining all the vectors that we want is by taking the cartesian product of the set of possible values with itself, once, because we are working with two dimensional vectors (how would you go about solving a similar problem for vectors in \(\mathbb{R}^3\), etc?).
The following code computes the cartesian product of the values and stores the result as a set of vectors:

```plaintext
> T:=cartprod([values, values]);
  vectors:={};
  while not T[finished] do
    vectors:=vectors union {convert(T[nextvalue](), Vector)};
  end do;
We can see the vectors:
> vectors;

    [1, 1, 1, 2, 2, 3, 3, 3, 3]

How do we see how many vectors are there?
> nops(vectors);

    9

Now suppose that we want to find, not just the vectors whose components have some specified values, but that you want to find all possible pairs of such vectors. For that, you have to choose from your set of possible vectors, two vectors in all possible ways. Again, the combinat package comes to the rescue: use the choose command:

```plaintext
> pairs:=choose(vectors, 2);

```

    [1, 1, 1, 2, 2, 3, 3, 3, 3],
    [2, 2, 2, 3, 3, 3, 3, 3, 3],
    [3, 3, 3, 3, 1, 2, 2, 2, 3],
    [1, 1, 2, 2, 2, 2, 3, 3, 3],
    [2, 2, 2, 3, 3, 3, 3, 3, 3],
    [3, 3, 3, 3, 1, 2, 2, 2, 3],
    [1, 1, 2, 2, 2, 2, 3, 3, 3],
    [2, 2, 2, 3, 3, 3, 3, 3, 3],
    [3, 3, 3, 3, 1, 2, 2, 2, 3],
```

of course, we could have done this by hand, but it takes a while... How many pairs did we get?
> nops(pairs);
Now we want to find, for each pair of vectors (the ones in pairs), the angle between them. The main ingredient for that is the command `map` that applies a function to each element of a list. For example:

```maple
triple := t -> 3*t;
aSet := {0, 3, 4, 5, 7};
map(triple, aSet);
```

```
triple → t → 3 t
```

```
aSet = {0, 3, 4, 5, 7}
   = {0, 9, 12, 15, 21}
```

So, we have to write a function that takes each entry in the pairs set and returns the angle: but we must be careful because each entry in the pairs set is, itself, a set of two vectors:

```maple
angle2 := s -> angle(s[1], s[2]);
```

```maple
angles := map(angle2, pairs);
```

```
angles := {0, arccos(2 \sqrt{10}/5), arccos(9 \sqrt{13} \sqrt{10}/130), arccos(3 \sqrt{2} \sqrt{5}/10), arccos(12/13),
          arccos(5 \sqrt{13} \sqrt{2}/26), arccos(11 \sqrt{13} \sqrt{10}/130), arccos(8 \sqrt{5} \sqrt{13}/65), arccos(7 \sqrt{5} \sqrt{10}/50),
          arccos(7 \sqrt{5} \sqrt{13}/65), arccos(3/5), arccos(\sqrt{5} \sqrt{10}/10), arccos(4/5)}
```

```maple
nops(angles);
```

```
13
```

But: we started with a set (pairs) containing 36 elements and we found only 13 angles! The point is that these are sets, so each element appears only once, so that repeated angles are not considered. A set is determined by braces `{}` while a list is determined by square braces `[]`.

Now we change subject and consider integration in Maple. The easiest case:

```maple
int(x^2 + sin(x), x);
```

```
x^3/3 - cos(x)
```

The previous example computed the integral ("antiderivative") of $x^2+\sin(x)$. If we want to find the definite integral of the same function for $x$ between -1 and 3 we evaluate:

```maple
int(x^2+sin(x), x=-1..3);
```

```
28/3 - cos(3) + cos(1)
```

or,

```maple
evalf(%);
```

```
10.86362814
```

But Maple can also evaluate integrals involving unknown constants:

```maple
int(cos(n*x), x=0..Pi);
```

```
sin(\pi n)/n
```

Notice that this last expression could be simplified to 0 because $\sin(\pi n) = 0$ for any integer $n$. But the
Notice that this last expression could be simplified to 0 because $\sin(\pi n) = 0$ for any integer $n$. But the problem here is that Maple doesn’t know that $n$ should be integer! We can tell Maple about this assumption with

$$\texttt{assume(n::integer);}$$

and then we reevaluate the same integral

$$\texttt{int(cos(n*x),x=0..Pi);}$$

This time we obtained the expected result. You may want to explore the possibilities of the \texttt{assume} command. Maple is capable of computing fairly complex integrals (both numerically and analytically) as well as higher dimensional ones, but we leave it to the interested reader to see the details.