Some surprises in Mathematica

December 1, 2005
Andrej Bauer
http://andrej.com

Mathematica is a great application. But like most Computer Algebra Systems, it does not always give you correct answers. Sometimes this is so because giving the correct answer in all cases would require Mathematica to solve undecidable problems. Sometimes it just a badly designed feature.

Badly designed syntax

■ The meaning of whitespace

Do you know how to factor $x^2 + 2\,x\,y + y^2$? Does Mathematica?

Factor[x^2 + 2 xy + y^2]

$x^2 + 2\,xy + y^2$

It looks like it does not even know hot to factor a simple expression like that. But how about something more complicated?

Factor[2 x^3 y - y^4 + 2 x^2 y^5 - x]

$(-1 + 2\,x^2\,y)\,(x + y^4)$

What is going on? Juxtaposition is interpreted by Mathematica as multiplication:

2 x 3 x y

$6\,x^2\,y$

It is easy to miss a missing space. If you write **xy** instead of **x y**, it might take you a while before you figure out what is going on. This is what was wrong above. It all works once you notice the missing space:

Factor[x^2 + 2 x y + y^2]

$(x + y)^2$

- **Function application**

Here is a rather baffling answer given by Mathematica:

```
Integrate[Sin (x^2), x]
```

$$\frac{\text{Sin}\, x^3}{3}$$

The problem is that **`Sin(x^2)`** should have been **`Sin[x^2]`** because in Mathematica square brackets are used for function application. If you write **`Sin(x^2)`**, it is understood as "constant **`Sin`** multiplied by **`x`** squared". The correct answer is:

```
Integrate[Sin[x^2], x]
```

$$\sqrt{\frac{\pi}{2}}\, \text{FresnelS}\left[\sqrt{\frac{2}{\pi}}\, x\right]$$

While it may be argued that it is user's fault for not knowing how to apply a function, this surely counts as bad design. What if I, an experienced user, wrote this by mistake in the middle of a larger computation? I would get no warning at all that things have gone wrong.

The limits of `Limit`

- **Limits with parameters**

When a free parameter occurs in an expression, Mathematica (usually) does not try to check whether things are different at special values of the parameter. For example:

$$\text{Limit}\left[\frac{x}{x-a}, x \to 0\right]$$

0

However, at $a = 0$ the answer is 1:

$$\text{Limit}\left[\frac{x}{x-0}, x \to 0\right]$$

1

It is understandable that Mathematica behaves this way. After all, the general problem of detecting special values is undecidable. However, if you read documentation for **`Limit`** there is no hint that it is user's responsebility to worry about parameters. While in the above case it is easy to guess that $a = 0$, the following example is not so easy:

$$\texttt{Limit}\left[\frac{(1+4x^2)^{1/4}-(1+5x^2)^{1/5}}{a^{-x^2/2}-\texttt{Cos}[x]},\ x\to 0\right]$$

0

Unless we practically compute this limit by hand, which is exactly what we try to avoid when we use Mathematica, you will never guess that $a = e$ is special:

$$\texttt{Limit}\left[\frac{(1+4x^2)^{1/4}-(1+5x^2)^{1/5}}{e^{-x^2/2}-\texttt{Cos}[x]},\ x\to 0\right]$$

6

▪ Computing limits by l'Hospital rule

Do you know l'Hospital rule for computing limits? I mean, *really* know it, including the side conditions? It goes as follows.

Let $a \in [-\infty, \infty]$ and $J \in \mathbb{R}$ a finite or infinite interval which intersects every neighborhood of a. Let $f, g : J \setminus \{a\} \to \mathbb{R}$ be continuous and differentiable functions such that

$$\lim_{x\to a} f[x] = \lim_{x\to a} g[x] = 0$$

or

$$\lim_{x\to a} |f[x]| = \lim_{x\to a} |g[x]| = \infty.$$

Suppose $g[x] \neq 0$ and $g'[x] \neq 0$ for every $x \in J \setminus \{a\}$ which is close enough to a. If the limit

$$\lim_{x\to a} \frac{f'[x]}{g'[x]}$$

exists then

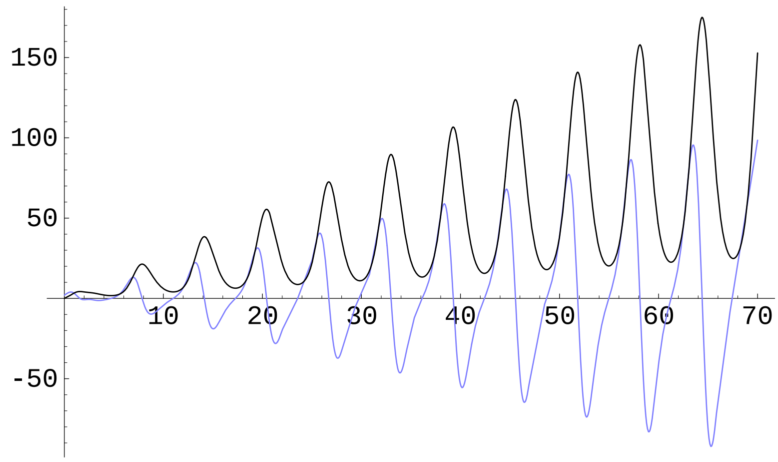
$$\lim_{x\to a} \frac{f[x]}{g[x]} = \lim_{x\to a} \frac{f'[x]}{g'[x]}$$

The blue side−conditionis nasty for Mathematica to check. So it ignores it. But documentation does not give you any hints about that. So let us take the following two functions:

```
f[x_] := x + Sin[2 x] / 2
g[x_] := e^Sin[x] x + e^Sin[x] Sin[2 x] / 2
```

If you look closely, you will see that $g[x] == f[x] \cdot e^{\texttt{Sin}[x]}$. Furthermore, $g[x]$ does not satisfy the conditions for l'Hospital rule, as $g'[x]$ has infinitely many zeroes in neighborhood of ∞ (the blue graph below is $g'[x]$):

```
Plot[{g'[x], g[x]}, {x, 0, 70},
  PlotStyle → {RGBColor[0.5, 0.5, 1.0], RGBColor[0, 0, 0]}];
```



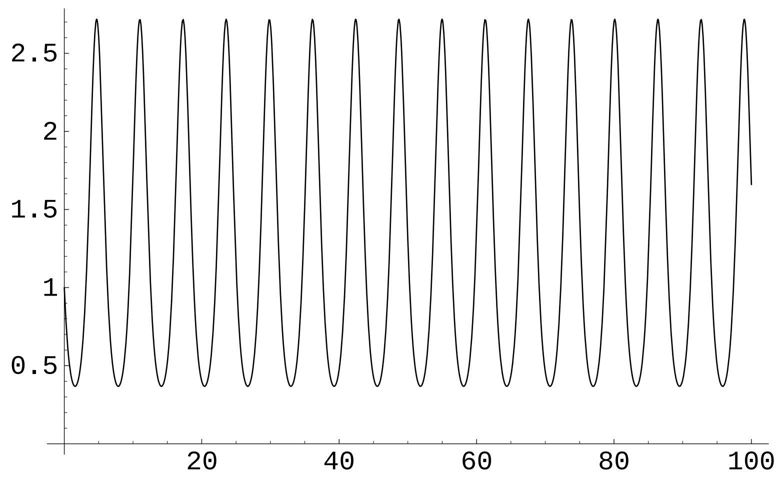
Let us compute the limit $\frac{f[x]}{g[x]}$, which is just $e^{-Sin[x]}$, as $x \to \infty$:

$$\text{Limit}\left[\frac{f[x]}{g[x]},\ x \to \infty\right]$$

0

This is a wrong answer, because the quotient $f[x] / g[x]$ oscillates in neighborhood of $x = \infty$:

$$\text{Plot}\left[\frac{f[x]}{g[x]},\ \{x, 0, 100\}\right];$$



If we simplify $f[x] / g[x]$ before computing its limit, Mathematica can handle it:

$$\frac{f[x]}{g[x]}\ \text{// Simplify}$$

$e^{-Sin[x]}$

```
Limit[Simplify[f[x]/g[x]], {x → ∞}]
```

```
{Interval[{1/e, e}]}
```

The order of things

You can get strange results by changing the order of operations when the order should not matter. These usually involve subtitutions.

Detecting singularities

Suppose I want to compute the sum $\sum_{k=0}^{5} a^k$ (yes, this is a silly example, you can create a smart one). What is the answer? Most people will forget to consider the possibility that $a = 0$ leads to 0^0 which is undefined. What does Mathematica say? It depends on the order of things. If we first compute the sum and then substitute `0` for `a` it forgets, just like people:

```
Sum[a^k, {k, 0, 5}] /. a → 0
```

```
1
```

If we substitute `0` for `a` first, then it tells us something is wrong (again, just like most people would):

```
Sum[0^k, {k, 0, 5}]
```

Power::indet : Indeterminate expression 0^0 encountered.

```
Indeterminate
```

Addition of scalars and vectors

Consider the affine map $f[v] == \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \cdot (v + \{0, 1\})$ which takes a two−dimensional vector `v`, translates it by `{0, 1}` and multiplies it by a 2×2 matrix. We might define it as follows in Mathematica:

In[9]:= $f[v\_] = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} . (v + \{0, 1\});$

This looks quite reasonable, does it not? Let us compute `f[{0, 0}]`:

In[10]:= `f[{0, 0}] // MatrixForm`

Out[10]//MatrixForm=

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

Why, we got a 2×2 matrix as a result instead of a vector?! What if we compute the same thing "by hand"?

In[11]:=
$$\begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} . (\{0, 0\} + \{0, 1\}) \text{ // MatrixForm}$$

Out[11]//MatrixForm=
$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Now it is a vector. So the problem must be with the definition of `f`. Indeed, writing `:=` instead of `=` helps:

In[12]:=
$$g[v\_] := \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} . (v + \{0, 1\})$$
```
g[{0, 0}] // MatrixForm
```

Out[13]//MatrixForm=
$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

But the real problem lies deeper. Mathematica distributes addition of a symbol over a list, which is problematic when the symbol is supposed to represent a vector:

$$a + \{x_0, x_1, x_2\}$$

$$\{a + x_0, a + x_1, a + x_2\}$$

So, it thinks that in the expression `v + {0, 1}` the symbol `v` is a scalar so it gets distributed over the list. When we plug in a vector for `v` we end up with a 2×2 matrix:

In[14]:=
```
v + {0, 1}
```

Out[14]=
```
{v, 1 + v}
```

In[15]:=
```
v + {0, 1} /. v -> {0, 0} // MatrixForm
```

Out[15]//MatrixForm=
$$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}$$

However, if we subtitute `{0, 0}` for `v` immediately, we get a vector:

In[16]:=
```
{0, 0} + {0, 1} // MatrixForm
```

Out[16]//MatrixForm=
$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Once again, we see how certain design choices in Mathematica are reasonable in certain contexts, but cause a lot of trouble in others.