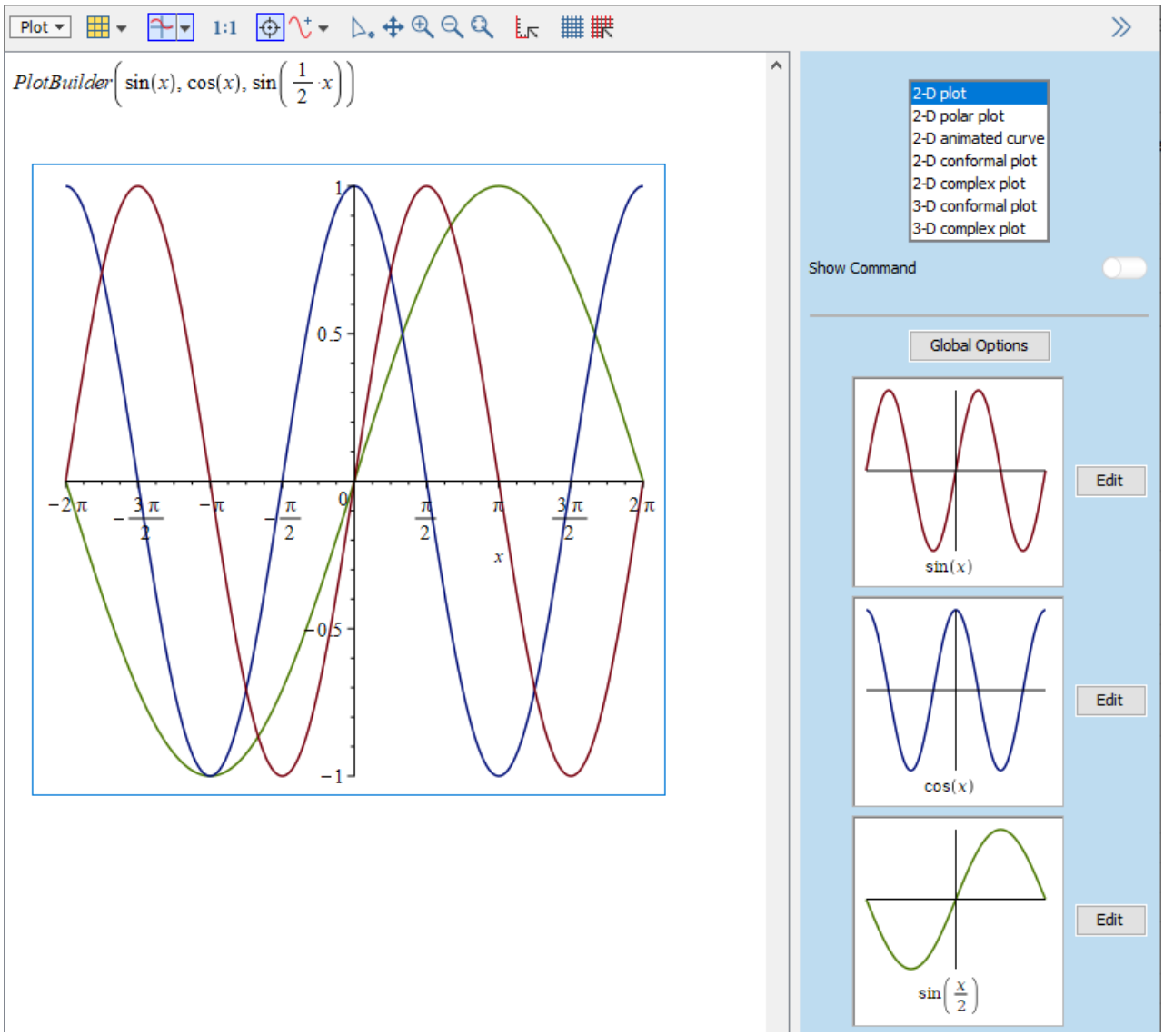
Visualization Updates in Maple 2022

Enhancements to the Plot Builder

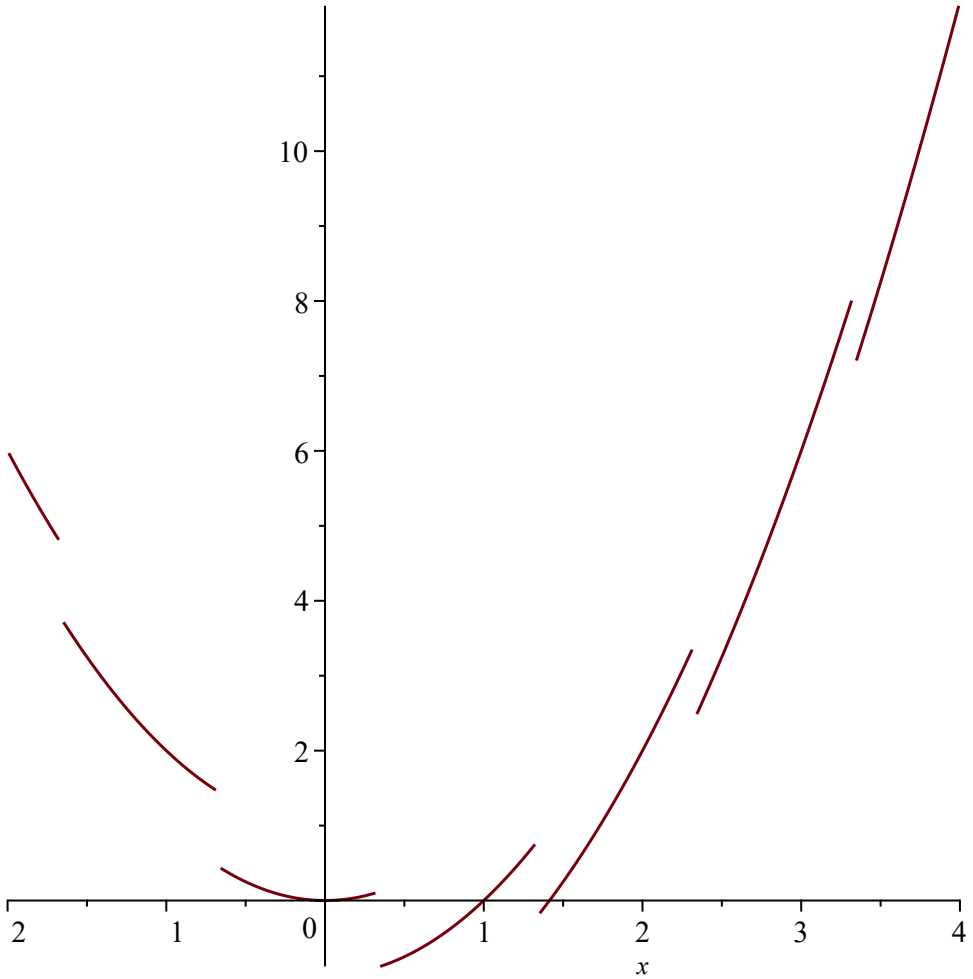
- The Plot Builder provides interactive functionality to build and display 2-D and 3-D plots. In Maple 2022 the Plot Builder has been enhanced to:
  - Support multiple expressions
  - Automatically select an initial choice of plot type, thereby showing a plot immediately
- For details, see Improvements to the Plot Builder in Maple 2022.



Smarter Adaptive Plotting

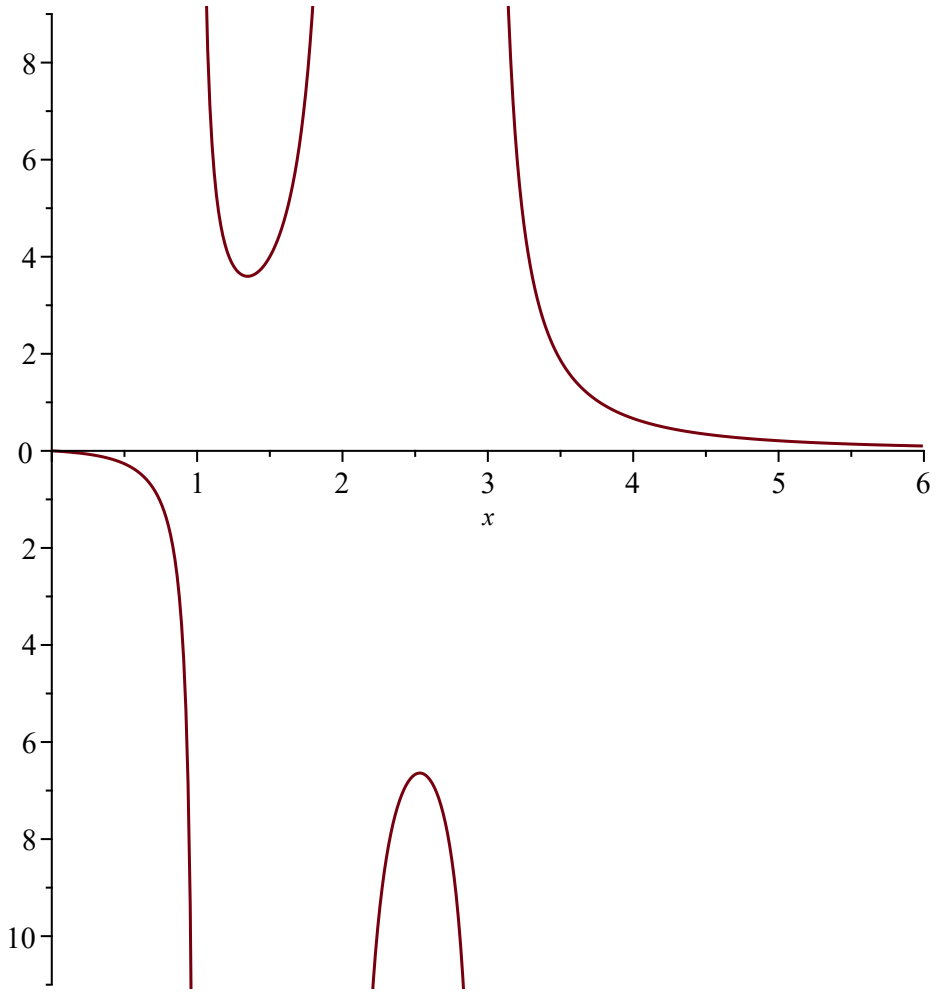
- Plotting in Maple now uses smarter adaptive plotting for two-dimensional plots, resulting in automatic detection of discontinuities and better looking plots by default.
- This improvement relies on the improved adaptive option for the plot command, which has two new option values geometric, and default, the latter of which is now the default for this option.
- The adaptive = geometric option uses a new plotting algorithm based on interval arithmetic using RealBox objects that allows for the automatic detection of many discontinuities **without** the use of the more expensive discont option.
- For example, the following plot involving the ceil function now detects discontinuities by default.

```
> plot( x^2 - ceil( x - 1/3 ), x = -2 .. 4 );
```



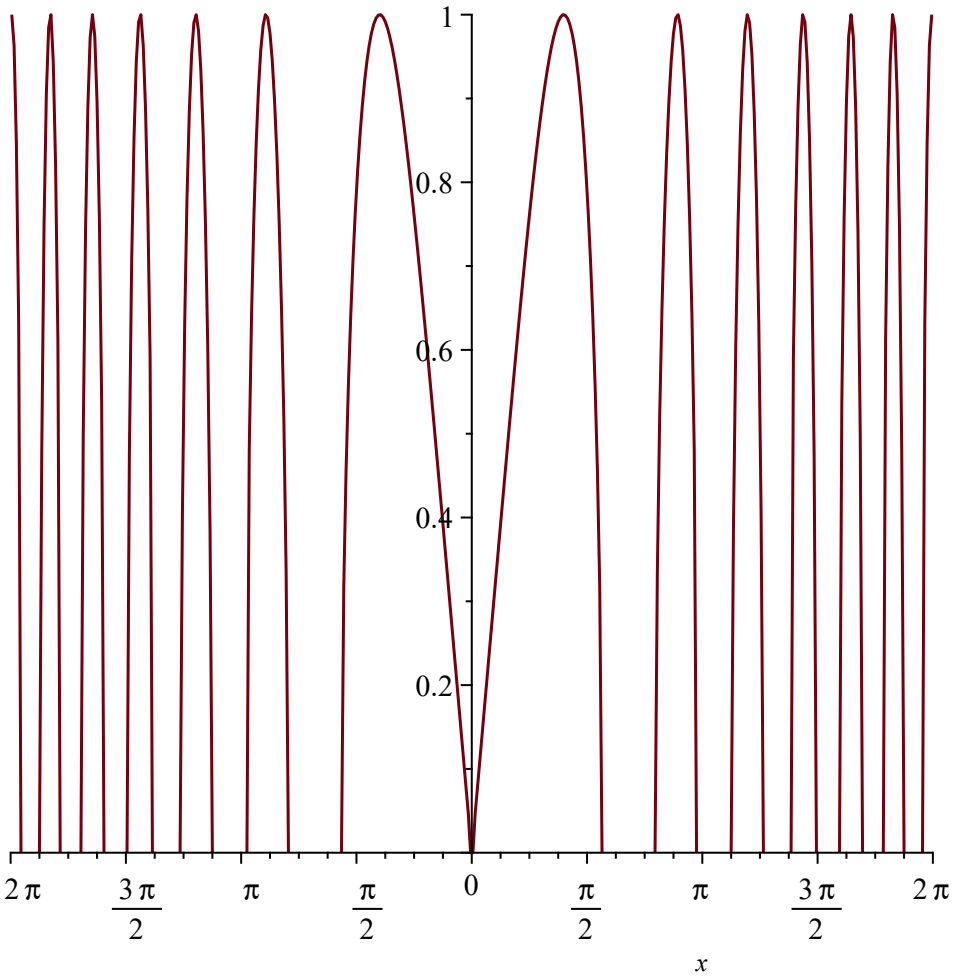
- Similarly, Maple can now detect discontinuities in plots of rational functions without using the discont option.

```
> plot( x/(x-1)/(x-2)/(x-3), x = 0 .. 6 );
```

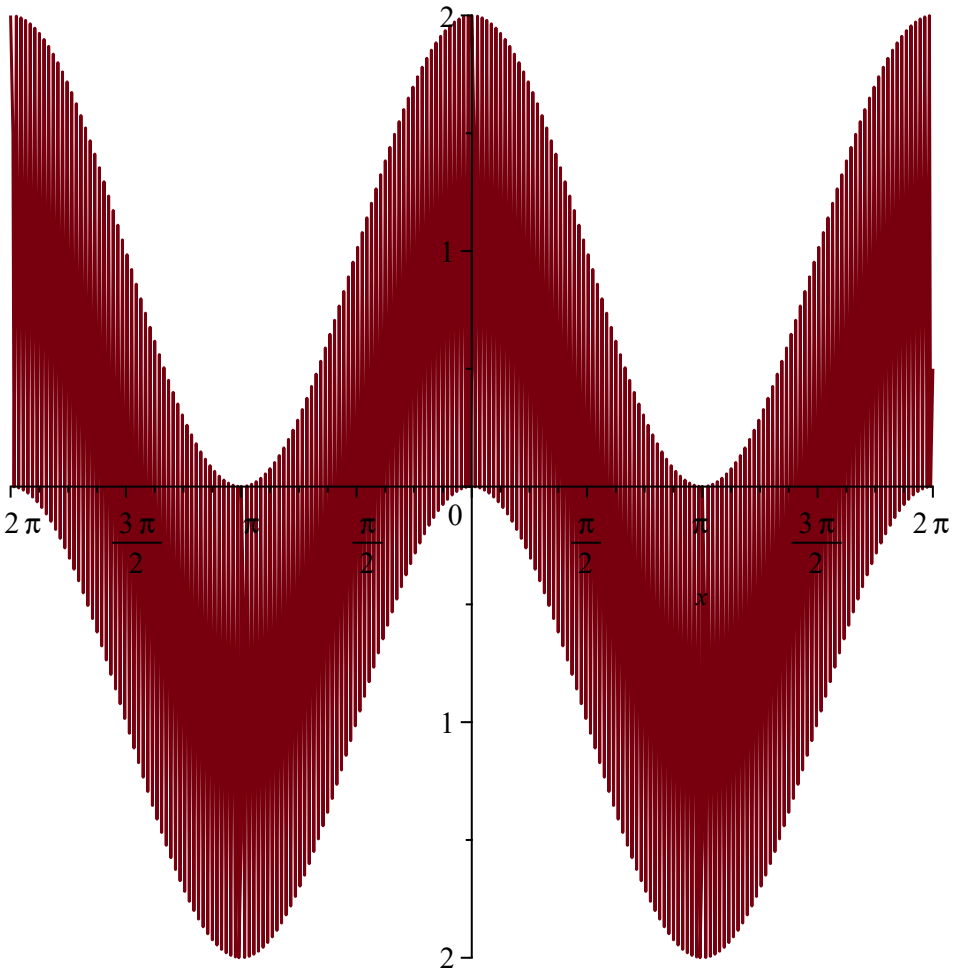


- In addition, many plots look better, in part due to the *a posteriori* geometric analysis employed by the new default.

```
> plot( sqrt( sin( x^2 ) ), x = -2*Pi .. 2*Pi );
```



```
> plot( sin( 100*x ) + cos( x ), x = -2*Pi .. 2*Pi );
```

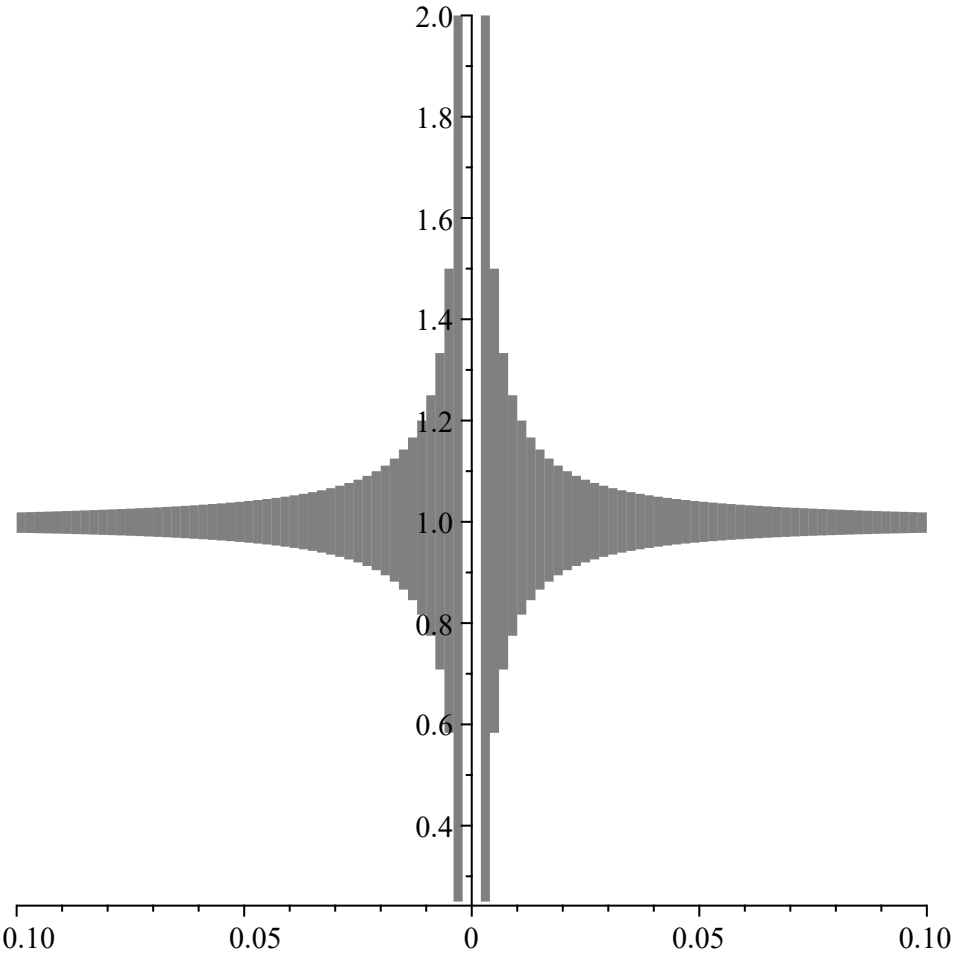


- The new default adaptive = default selects the adaptive = geometric option in most cases, but uses adaptive = true for polynomials and for expressions deemed too large for the new geometric plotting algorithm, which does have additional overhead relative to adaptive plotting.

- One interesting example is the function $\frac{\sin(x)}{x}$. This plot looks good both with the new algorithm and with the previous one. However, with an approach purely based on interval arithmetic, a plot of this function would be inaccurate near the origin. For example, with $\epsilon > 0$ being sufficiently small, consider x in the interval $[\epsilon, 2\,\epsilon]$. A straightforward application of interval arithmetic would evaluate both x and $\sin(x)$ to approximately $[\epsilon, 2\,\epsilon]$, and their ratio then to $\left[\frac{1}{2}, 2\right]$. This is not enough information to infer that the result should be very close to 1. We emulate such an approach with intervals only below:

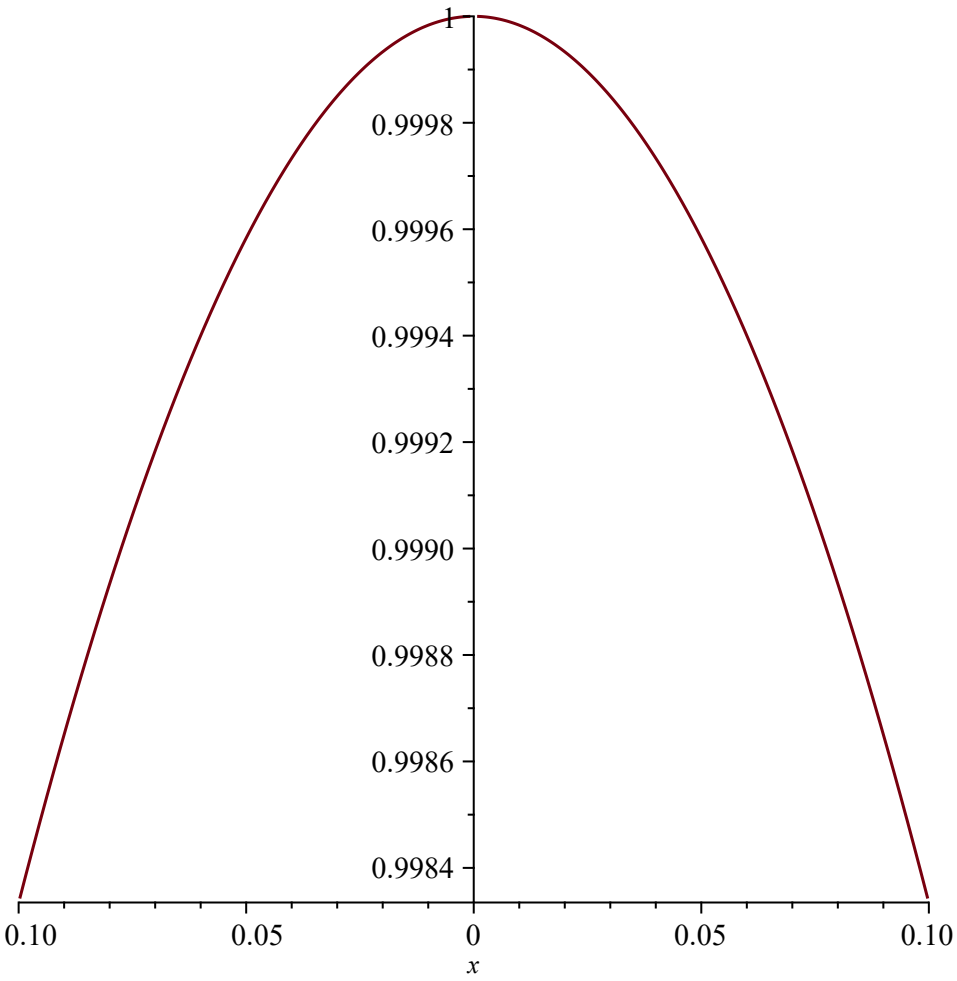
```
> plot_intervals := proc(expr :: algebraic, rng :: name = range, $)
    local x := lhs(rng);
    local (lo, hi) := op(rhs(rng));
    local n_boxes := 100;
    local radius := (hi - lo) / n_boxes / 2;
    local i;
    local intervals := [seq(RealBox(i, radius), i = lo + radius .. hi
  - radius,
        'numelems' = n_boxes)];
    local values := eval~(expr, x =~ intervals);

    local lo_x := map(Centre, intervals) -~ radius;
    local hi_x := map(Centre, intervals) +~ radius;
    local lo_y := map(Centre, values) -~ map(Radius, values);
    local hi_y := map(Centre, values) +~ map(Radius, values);

    return plots:-display(seq(
        plottools:-rectangle([lo_x[i], lo_y[i]], [hi_x[i], hi_y[i]],
  'style'='polygon', 'transparency'=0.5),
        i = 1 .. n_boxes));
  end proc:
```

```
> plot_intervals( sin(x)/x, x=-0.1 .. 0.1 );
```



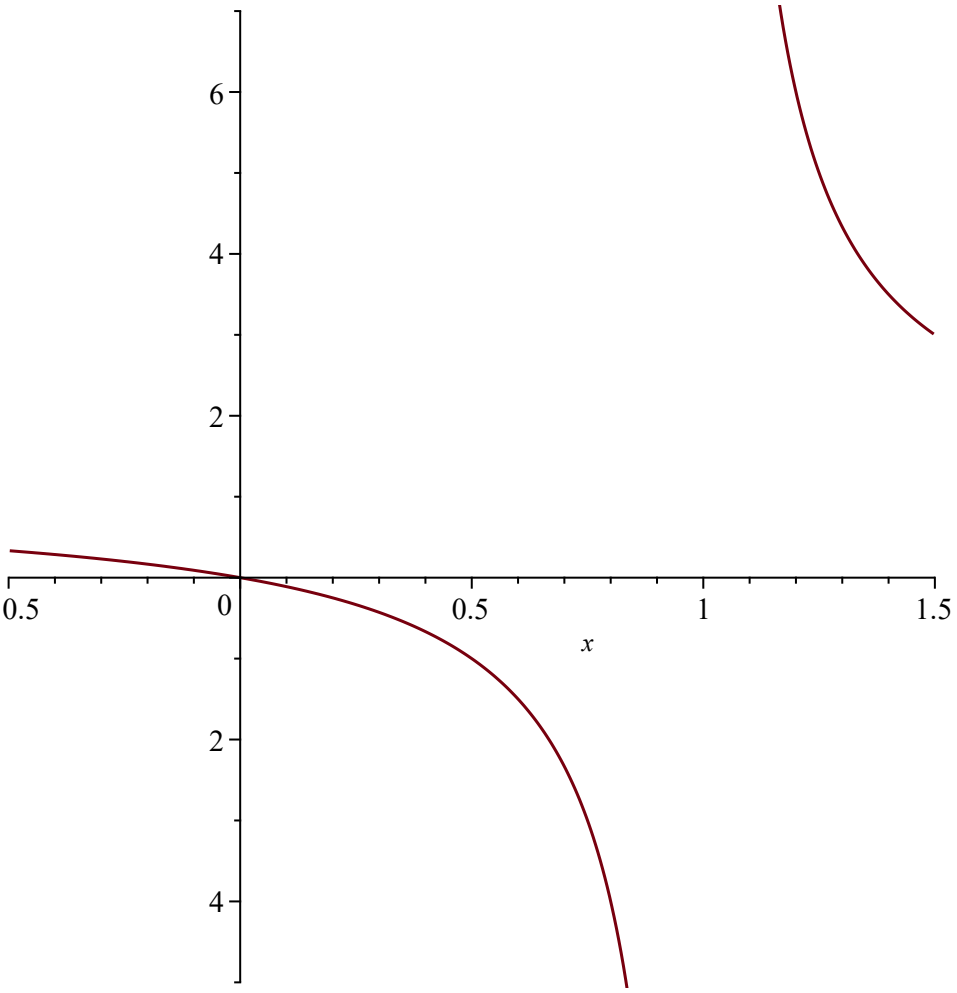
- The a posteriori geometric analysis allows us to plot this curve very accurately, despite these limitations of interval arithmetic.

```
> plot( sin( x )/x, x = -0.1 .. 0.1 );
```

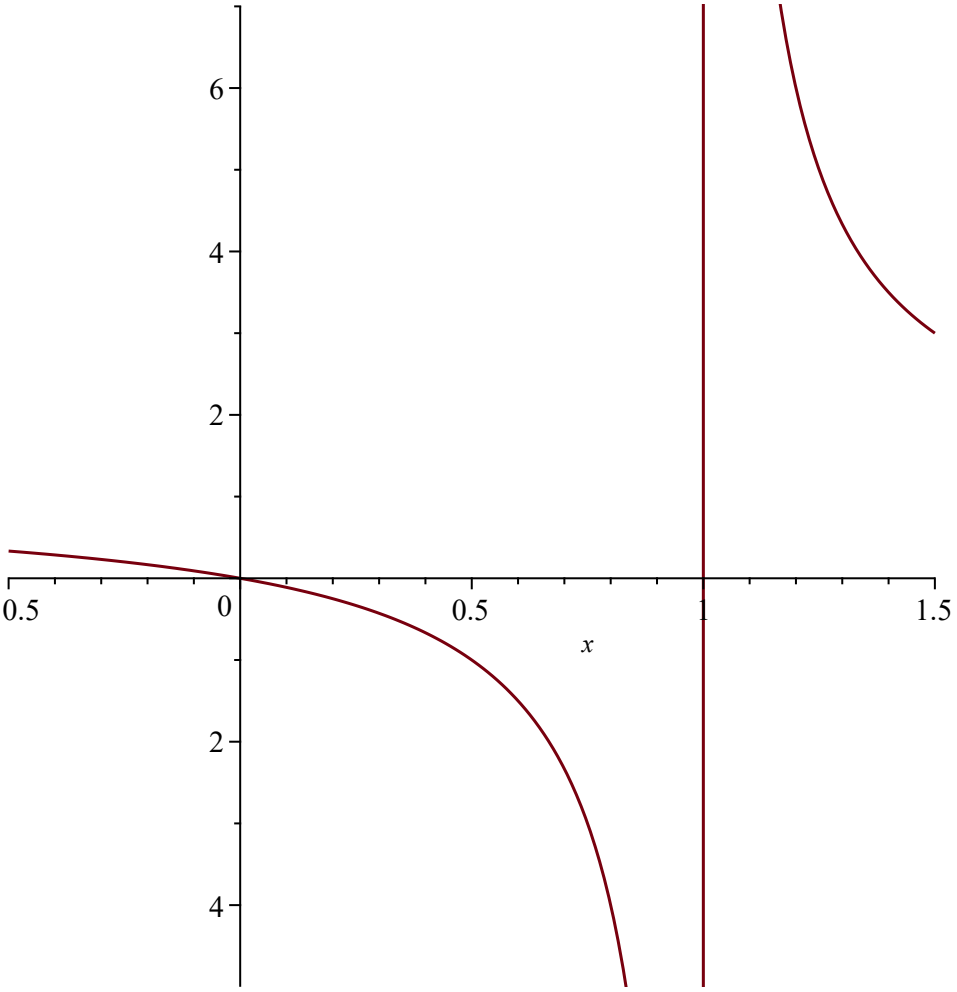


• You can restore the previous behavior by using the option adaptive = true.

```
> plot(x / (x-1));
```



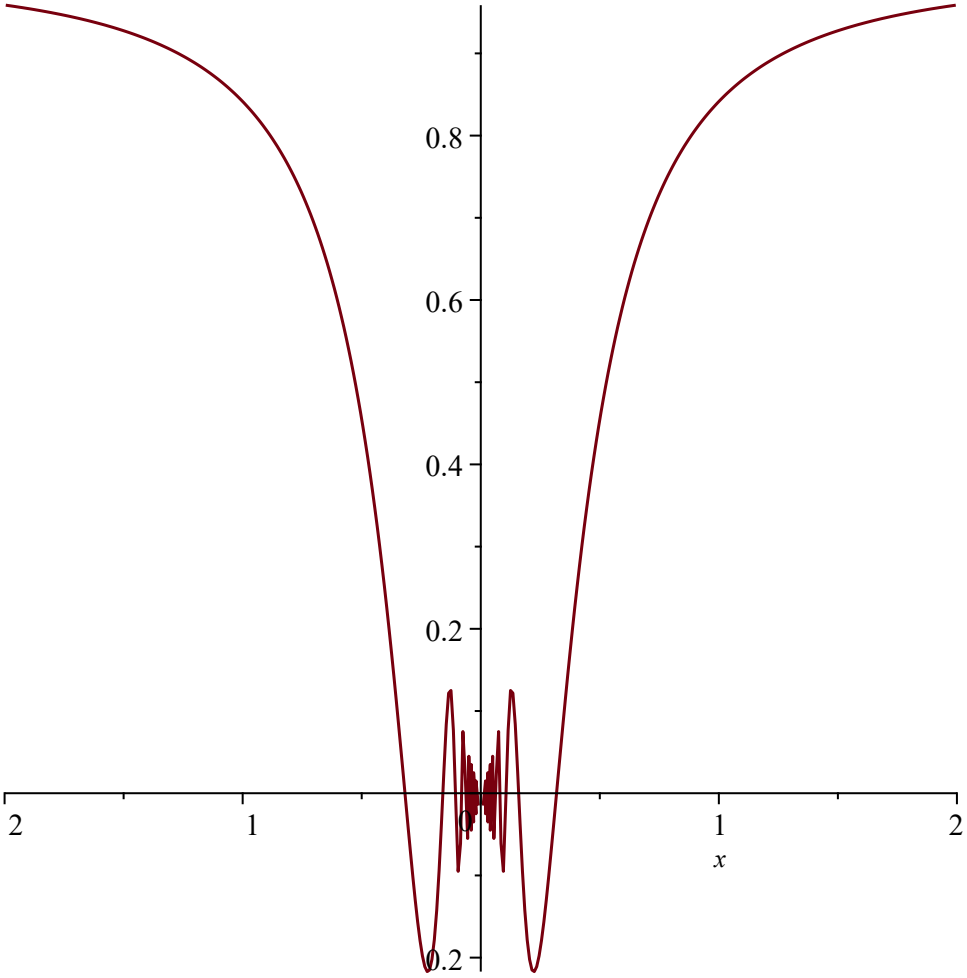
```
> plot(x / (x-1), 'adaptive' = true);
```



▼ Resize, Zoom, and Pan Improvements

- Manipulating a plot by resizing, zooming, or panning it now results in the plot being redrawn with missing details if necessary.
- For example, the following plot only contains enough details for a correct appearance at its default size. Previously, when it was resized or zoomed in, it would be apparent that some details were too coarse. Now, the plot is redrawn at a higher resolution instead of re-using the original resolution in a higher resolution context:

```
> plot(x*sin(1/x), x=-2..2);
```

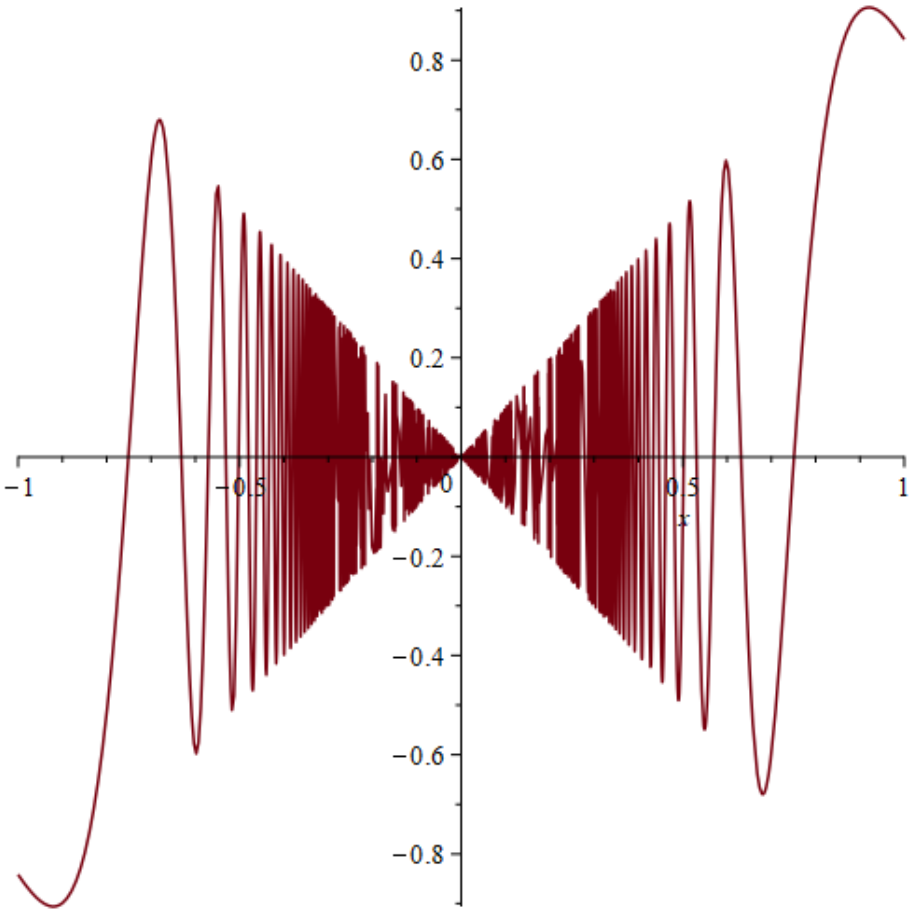


- Similarly, when the user pans the plot's view either to the left or right, the plot is now automatically redrawn in order to show new parts of the graph which were not previously computed because they were not part of the original view.

Resize improvements

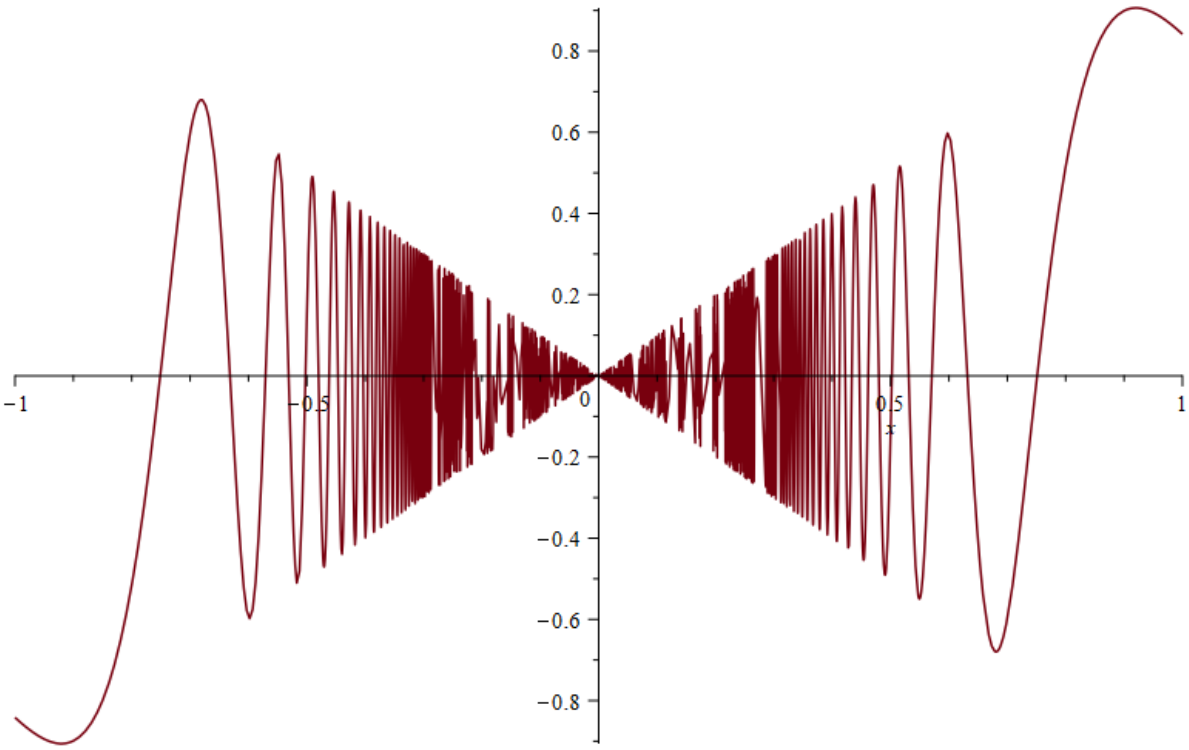
- Here follow some screenshots of the difference after resizing a plot in Maple2021 vs. Maple2022:
- A plot output in Maple2021:

> $plot\left(x\cdot\sin\left(\frac{1}{x^4}\right), x=-1..1\right);$



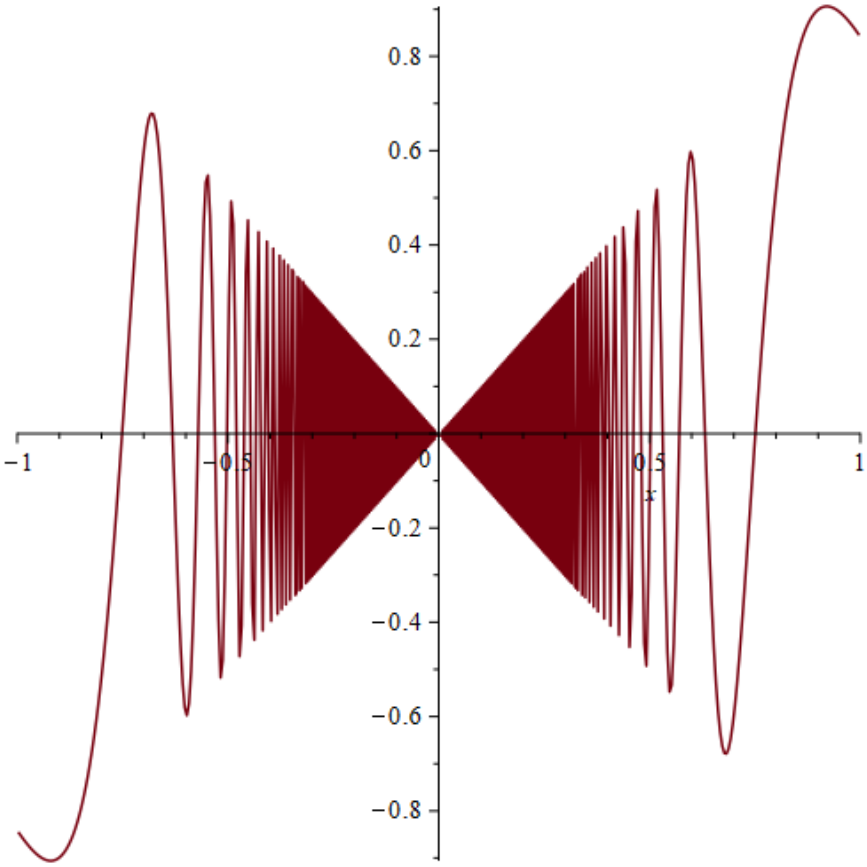
• The same Maple2021 plot output after resizing the plot window:

$$> \ plot\left(x \cdot \sin\left(\frac{1}{x^4}\right), x = -1..1\right);$$



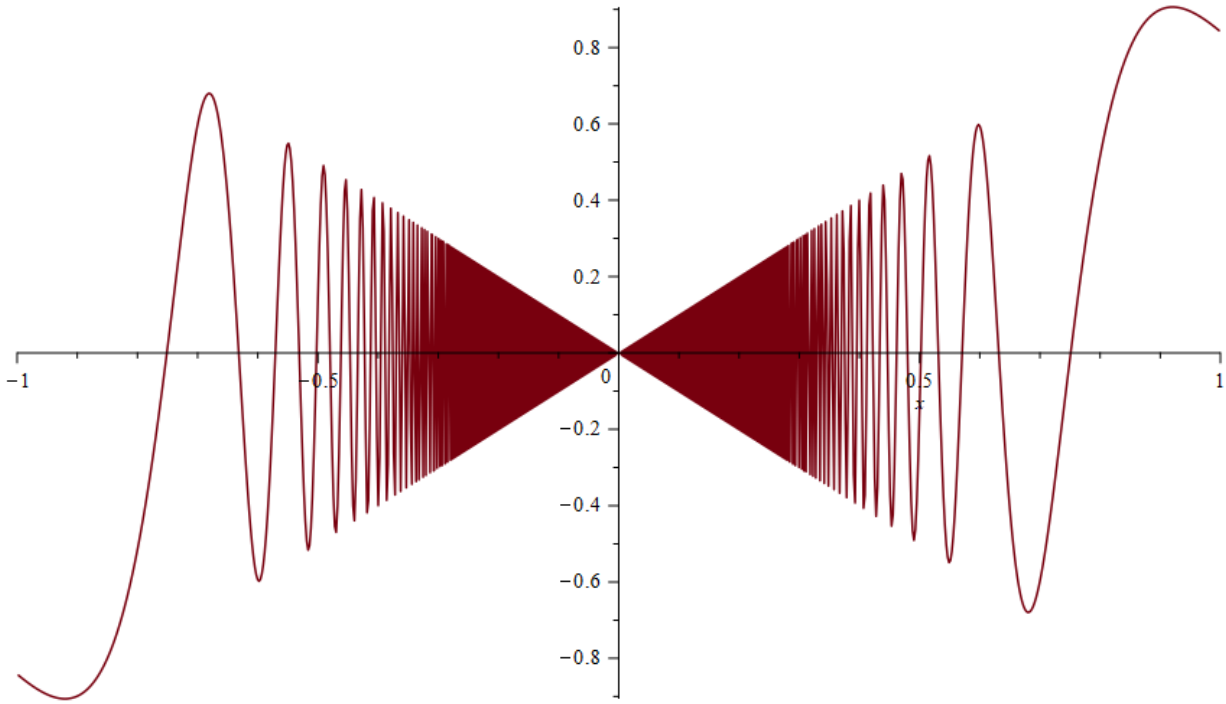
• A plot output in Maple2022:

> $plot\left(x \cdot \sin\left(\frac{1}{x^4}\right), x = -1..1\right);$



• The same Maple2022 plot output after resizing the plot window:

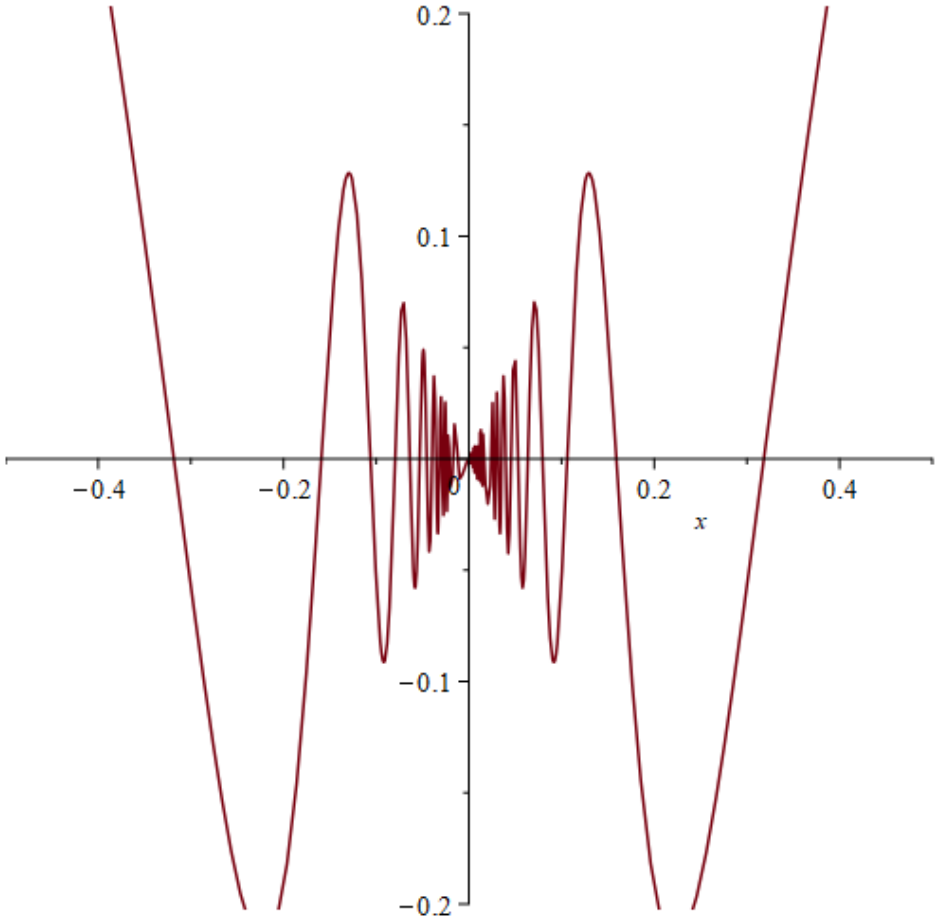
```
> plot(x·sin(1/x^4), x=-1..1);
```



Zoom improvements

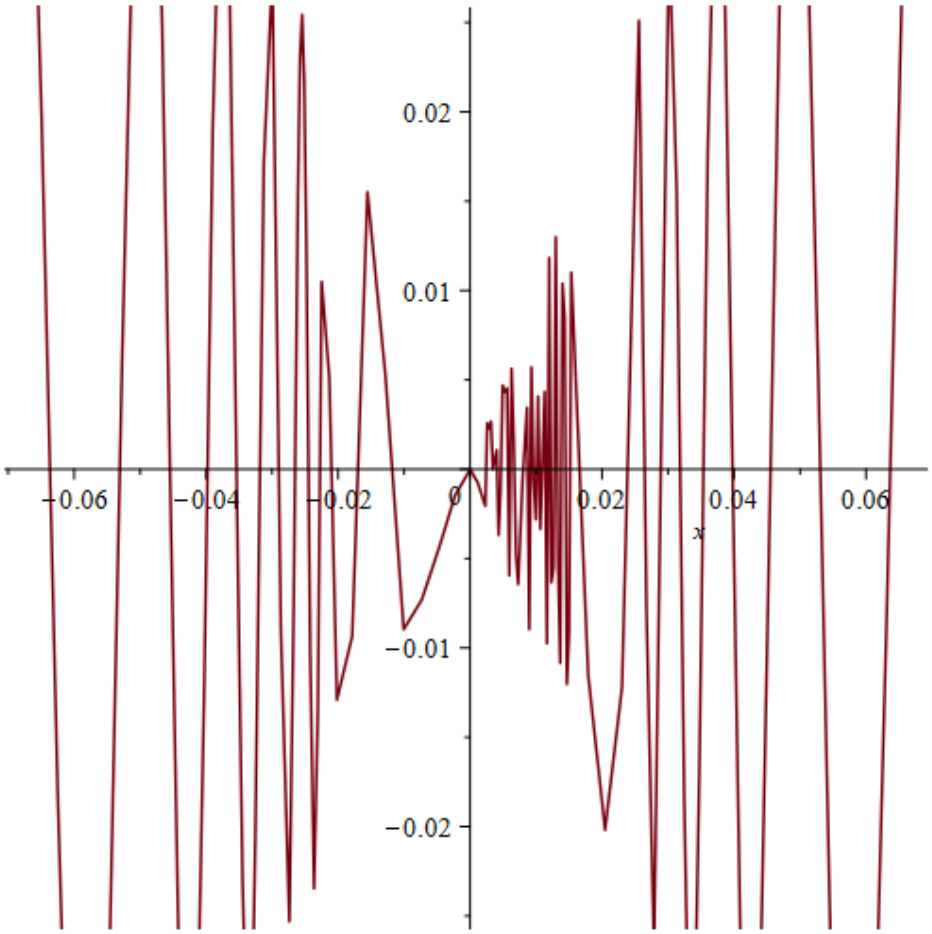
- Here follow some screenshots of the difference after zooming a plot in Maple2021 vs. Maple2022:
- A plot output in Maple2021:

$$> \ plot\left(x \cdot \sin\left(\frac{1}{x}\right), x = -1 \ldots 1, view = \left[-\frac{1}{2} \ldots \frac{1}{2}, -\frac{1}{5} \ldots \frac{1}{5}\right]\right);$$



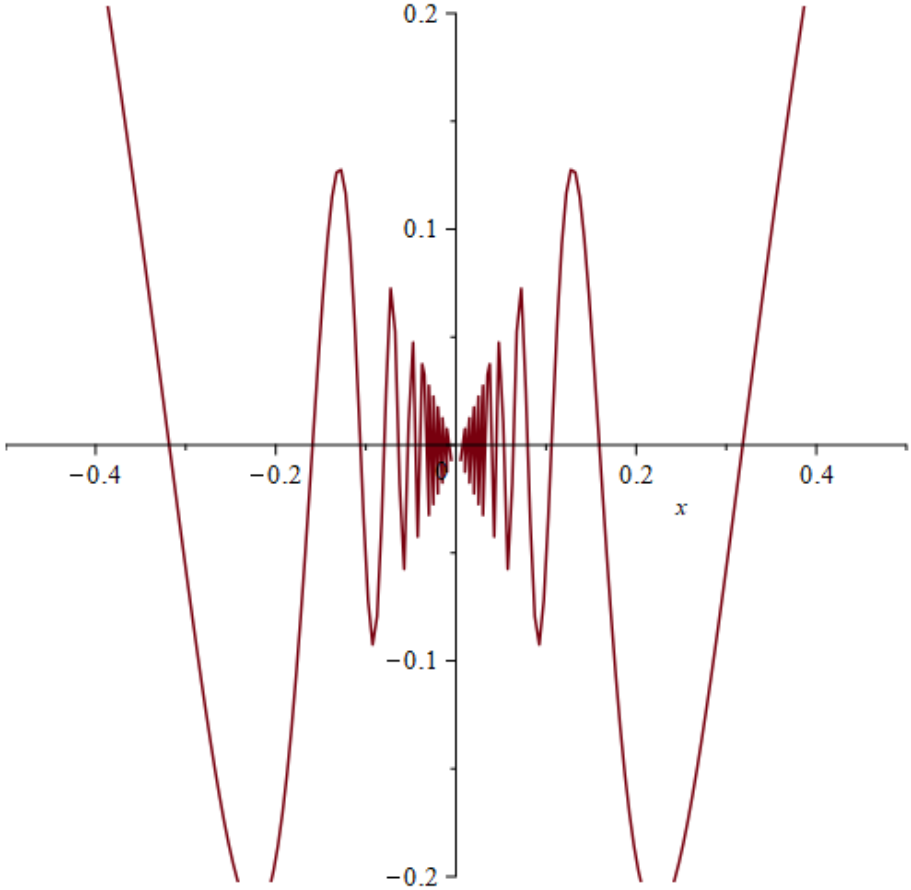
• The same Maple2021 plot output after zooming into the plot:

> $plot\left(x \cdot \sin\left(\frac{1}{x}\right), x = -1 \ldots 1, view = \left[-\frac{1}{2} \ldots \frac{1}{2}, -\frac{1}{5} \ldots \frac{1}{5}\right]\right);$



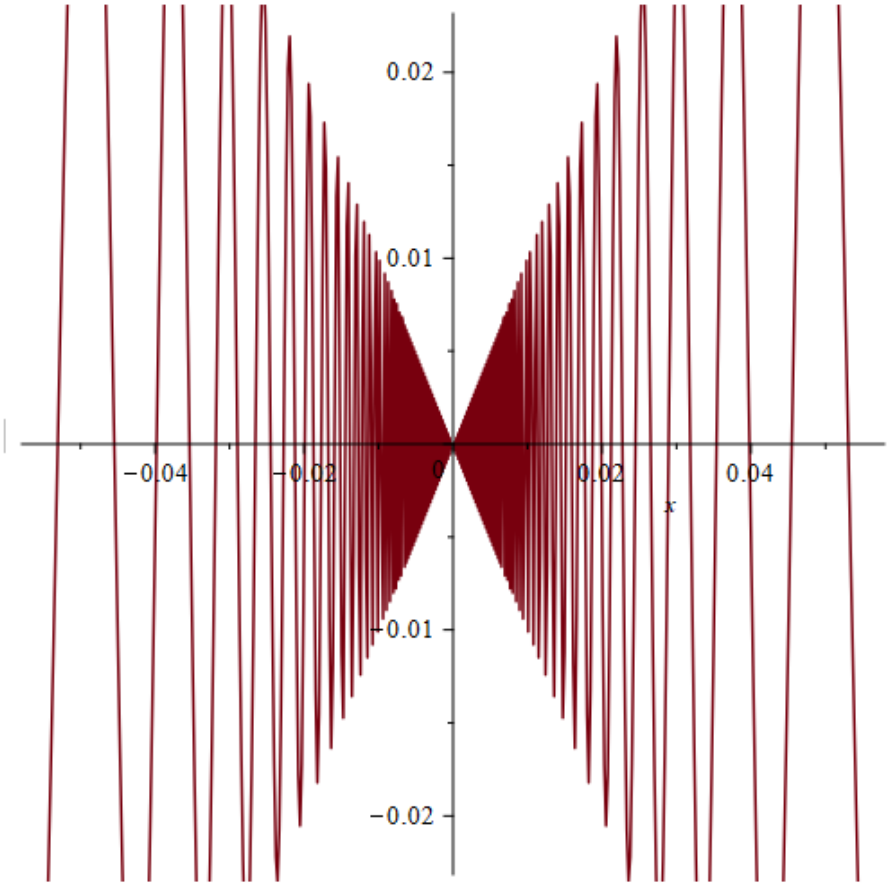
• A plot output in Maple2022:

> $plot\left(x \cdot \sin\left(\frac{1}{x}\right), x = -1 \ldots 1, view = \left[-\frac{1}{2} \ldots \frac{1}{2}, -\frac{1}{5} \ldots \frac{1}{5}\right]\right);$



• The same Maple2022 plot output after zooming into the plot:

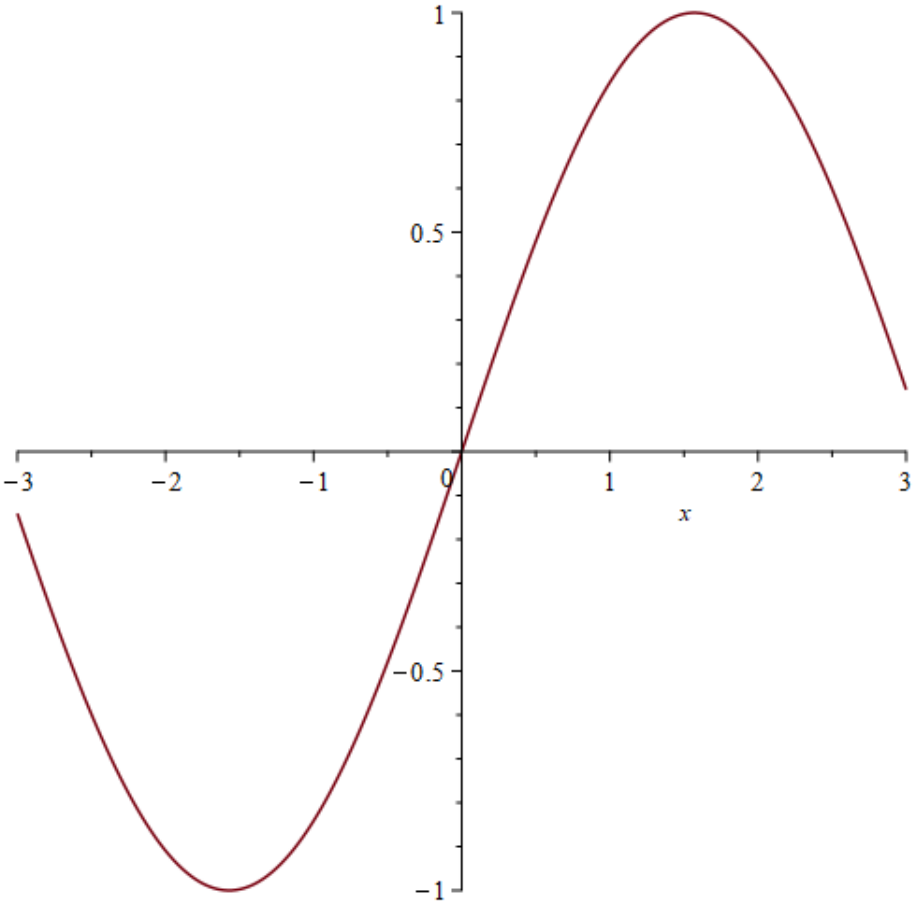
$$> \ plot\left(x \cdot \sin\left(\frac{1}{x}\right), x = -1 \ldots 1, view = \left[-\frac{1}{2} \ldots \frac{1}{2}, -\frac{1}{5} \ldots \frac{1}{5}\right]\right);$$



Pan improvements

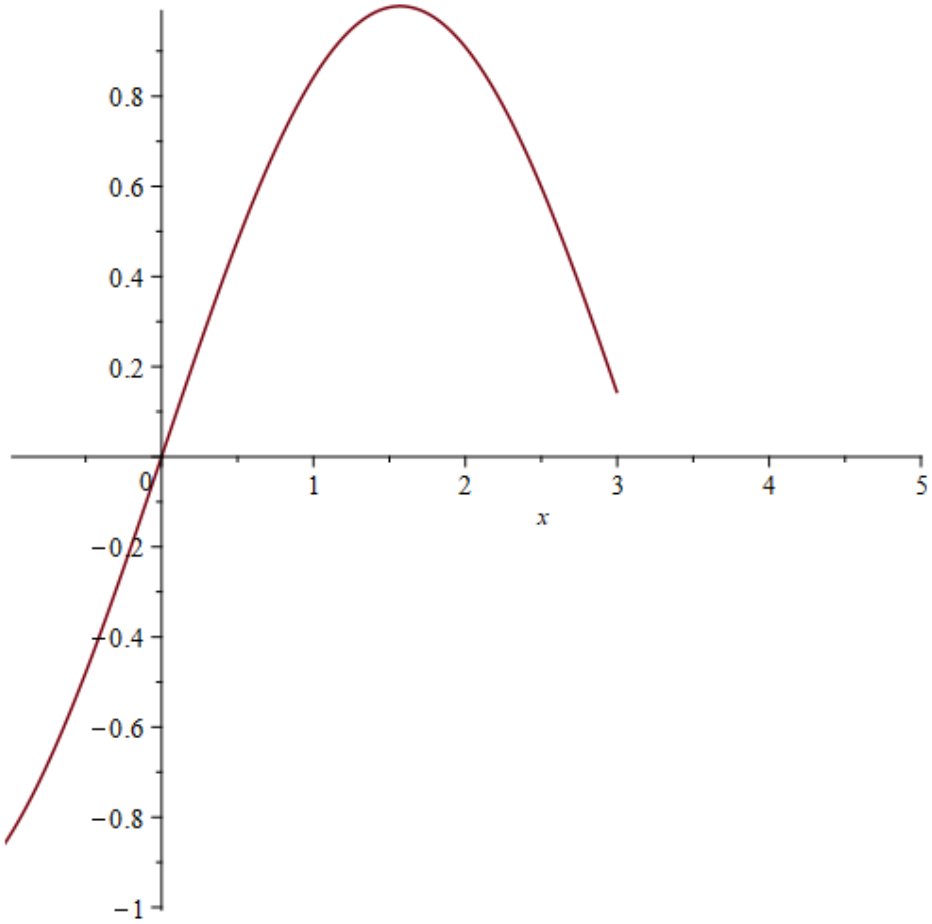
- Here follow some screenshots of the difference after panning a plot in Maple2021 vs. Maple2022:
- A plot output in Maple2021:

```
> plot(sin(x), x=-3..3);
```



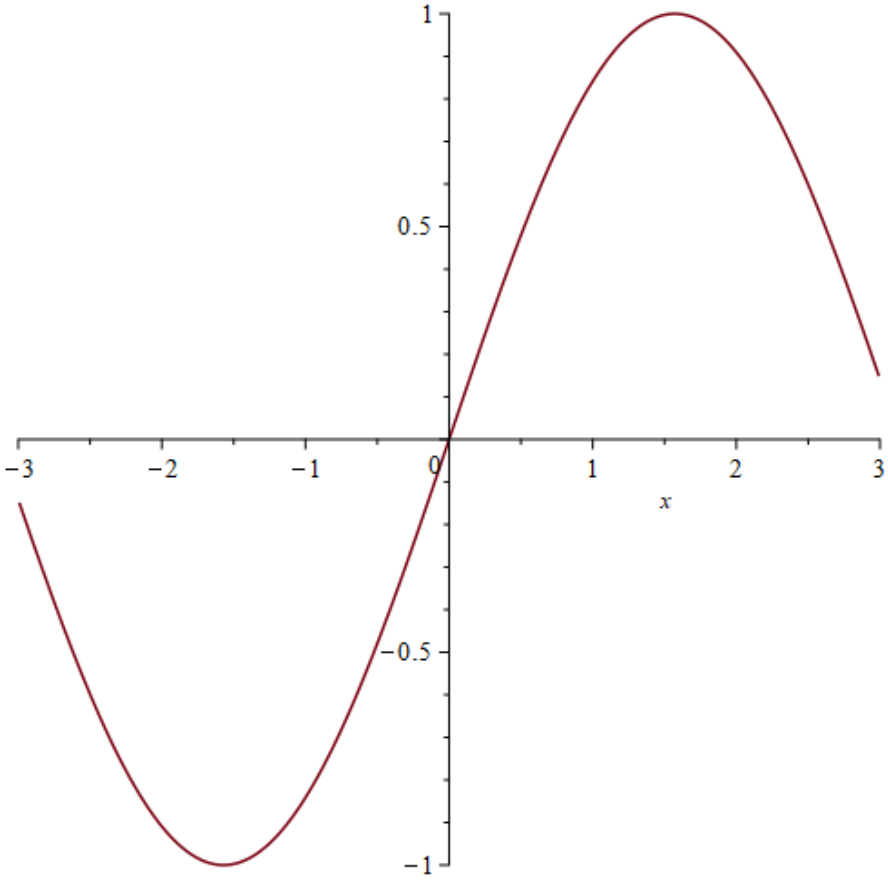
• The same Maple2021 plot output after panning the plot's view to the right:

```
> plot(sin(x), x = -3 ..3 );
```



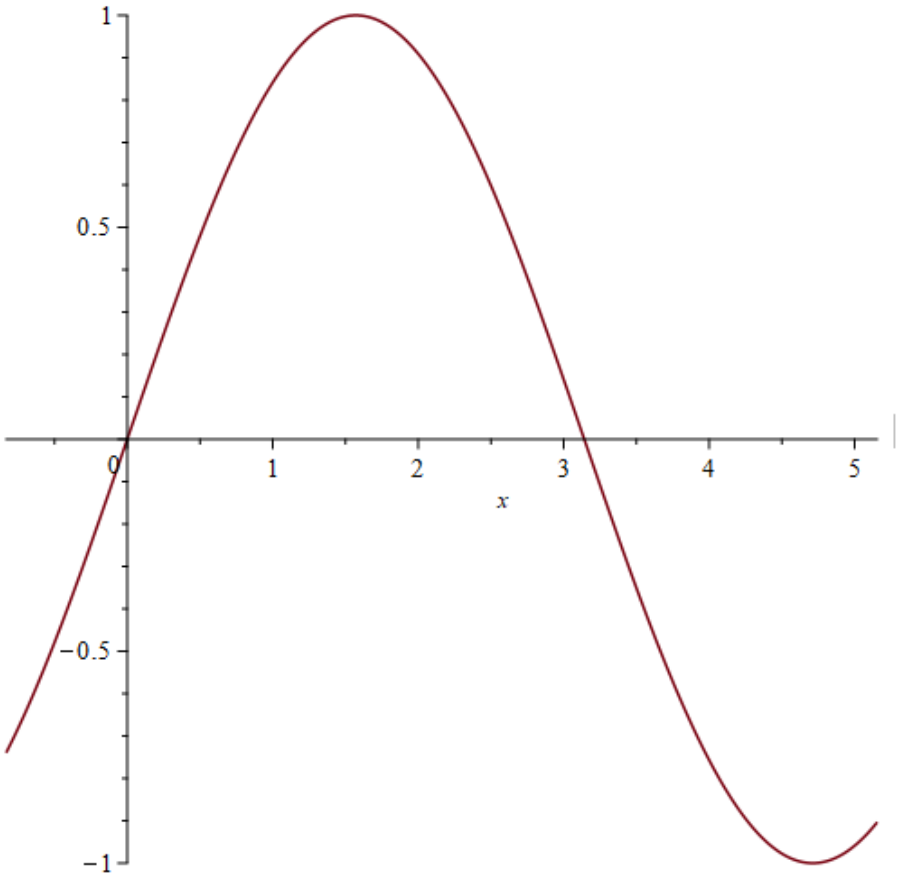
• A plot output in Maple2022:

> $plot(\sin(x), x = -3..3);$



• The same Maple2022 plot output after panning the plot's view to the right:

```
> plot(sin(x), x = -3 .. 3);
```

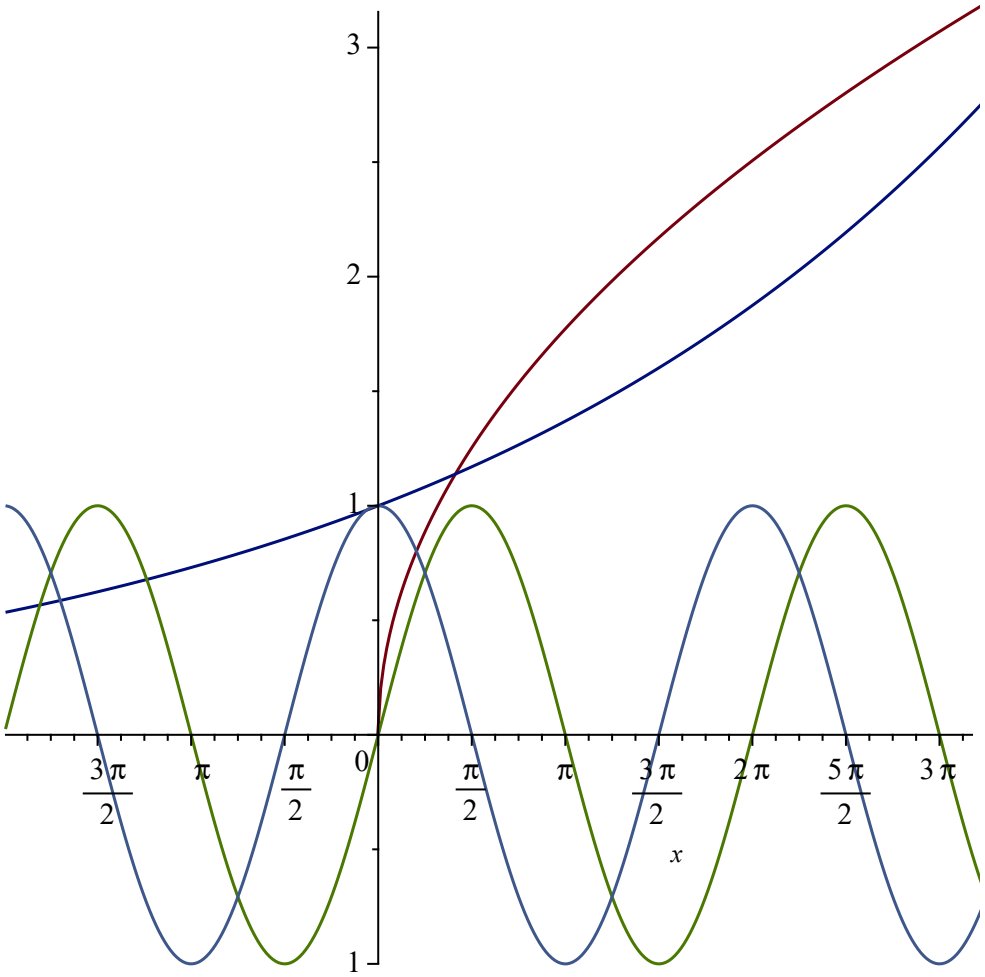


Improved Display of Multiple Plots Using plots:-display

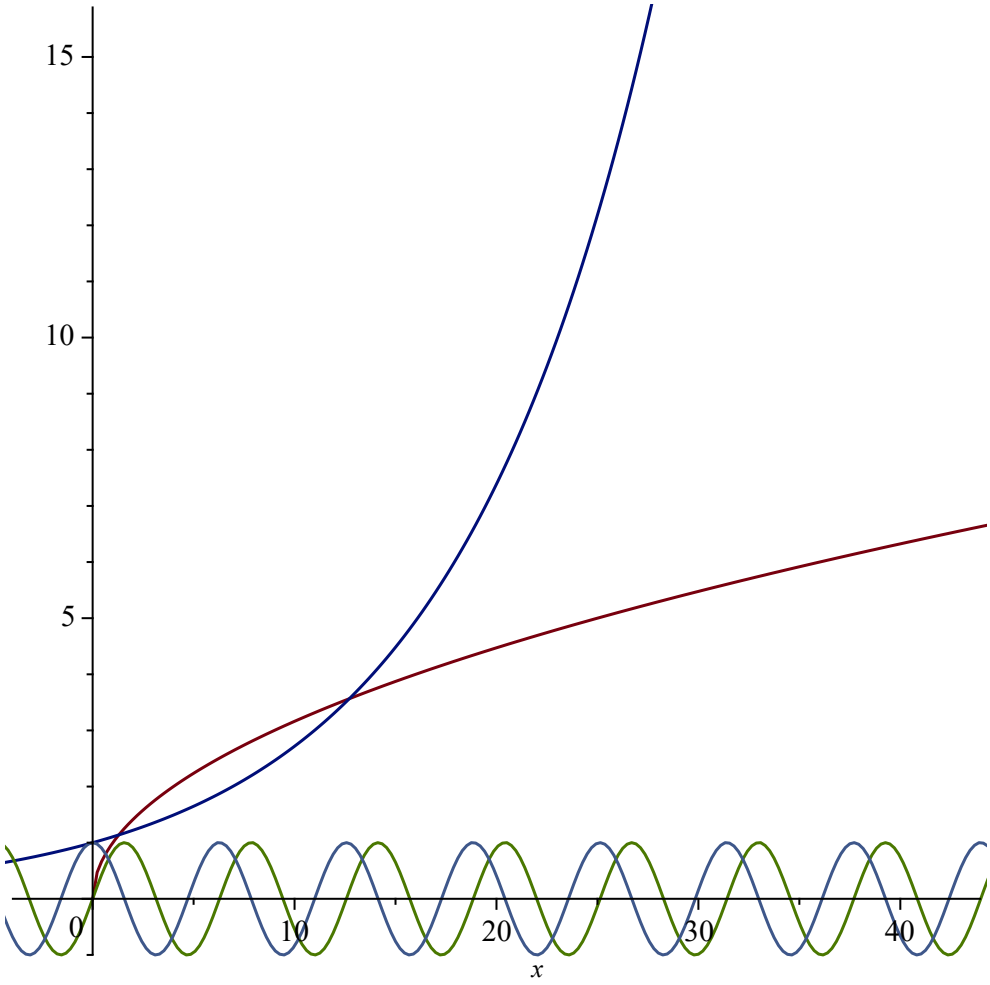
- The command plots:-display is used to display multiple plots in the same plot region. In Maple 2022, when you use plots:-display on a collection of 2-D static plots created using the plot command, it is now converted by default to a single plot command and then redrawn.
- The resulting plot:
  - Has a seamless look because there is enough data to fill the view despite component plots potentially having been computed for different views.
  - Can be resized, zoomed, or panned via the interactive controls in the plotting toolbar without causing any gaps in the data being shown or requiring the user to re-execute any plot commands.
- As a result of this redrawing, some features of the component plots which were left unspecified may change. For example, plots with unspecified colors might have the same color when displayed individually but be shown with different colors in order to distinguish them when combined using plots:-display. This redrawing can be turned off by setting the option redraw to false. Sometimes it is not possible to combine a collection of plots into a single plot command without losing some crucial information. In such a case, the plots are only displayed together rather than redrawn.

• Consider the following output from plots:-display:

```
> plots:-display(plot([sqrt(x), exp(x/10)]), plot([sin(x), cos(x)]));
```



If the user zooms out and pans to the upper right using the manipulators found on the toolbar, the view will now change to the following:



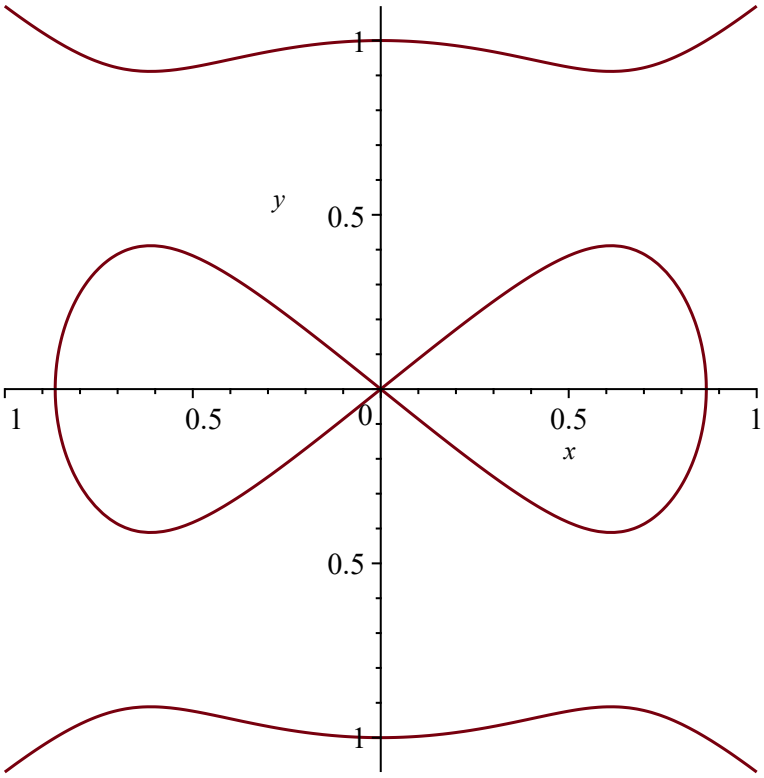
There is no need to re-execute any plot or plots:-display command to achieve this.

Domain for Implicit Plots

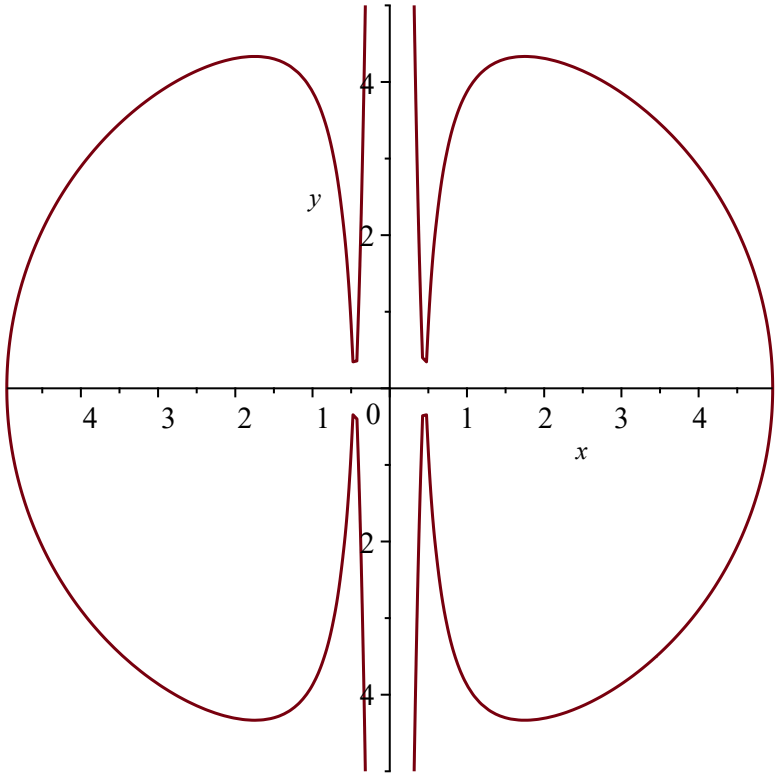
- In Maple 2021, we improved Maple's ability to choose a domain for an expression passed to the plot and plot3d commands. For Maple 2022, we expanded this to the implicitplot and implicitplot3d commands.

```
> with(plots):
```

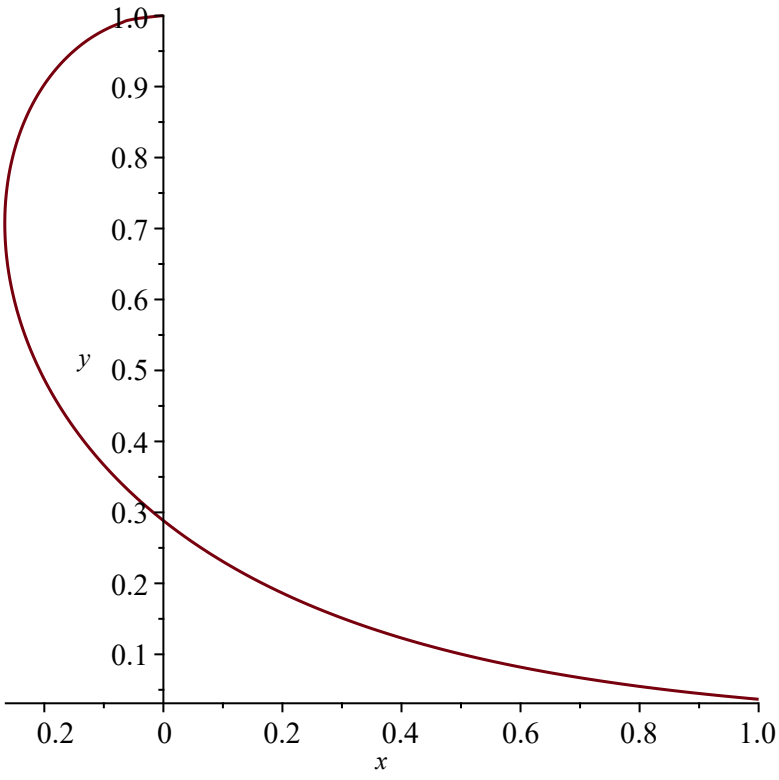
```
> implicitplot(y^2*(y^2-1) - x^2*(x^2-3/4));
```



```
> implicitplot(x^4*(x^2 + y^2) - (5*x^2 - 1)^2);
```

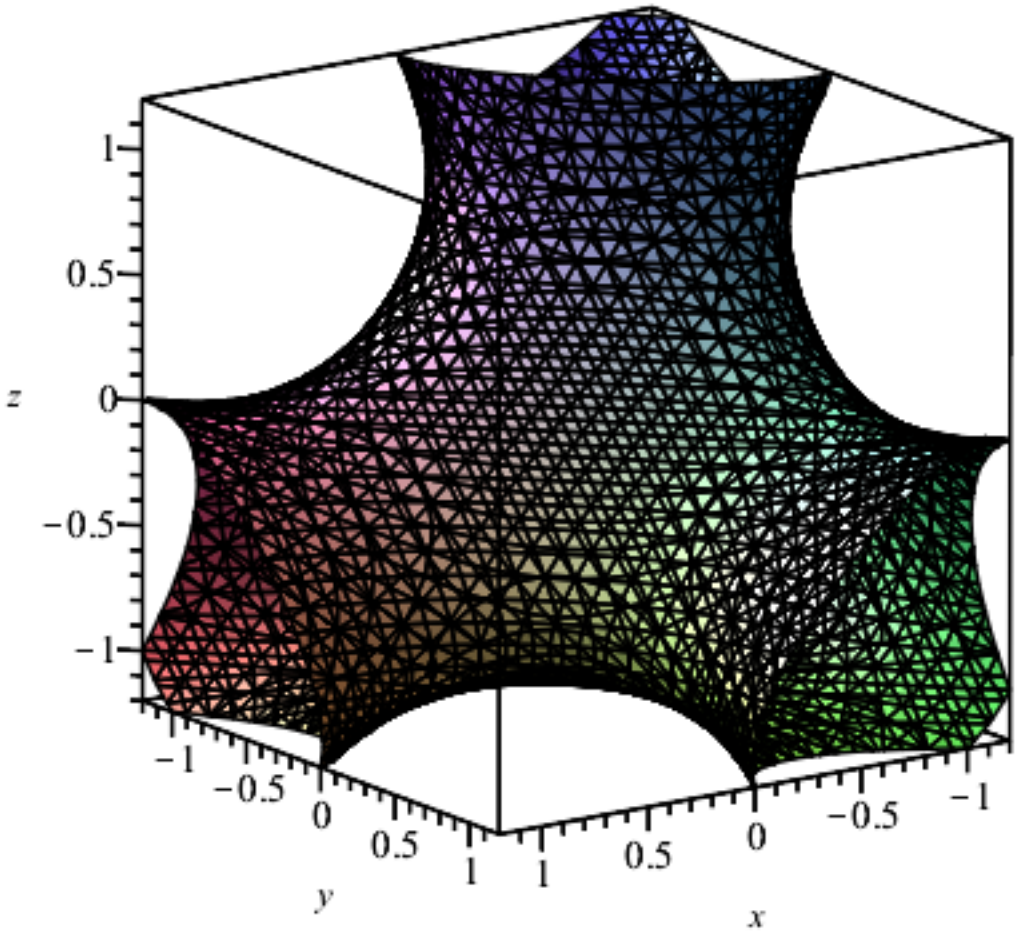



```
> implicitplot(x + sqrt(1 - y^2) - ln((1 + sqrt(1 - y^2))/y)/2);
```

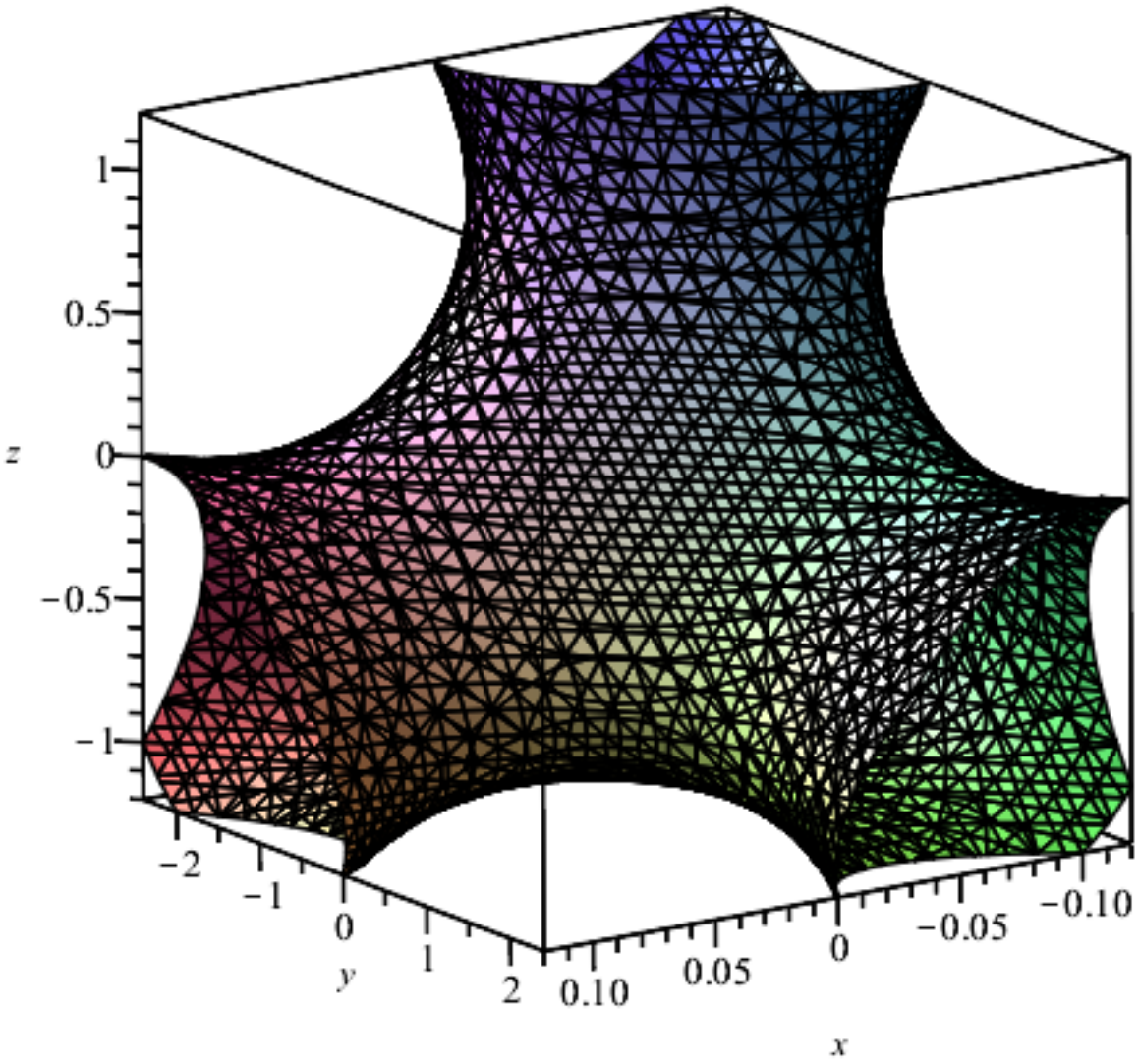


• Below, we see that Maple scales the domain with the scaling factors for x and y.

```
> implicitplot3d(x^3 + y^3 + z^3 + 1 = (x + y + z + 1)^3, size = [700,
  700]);
```



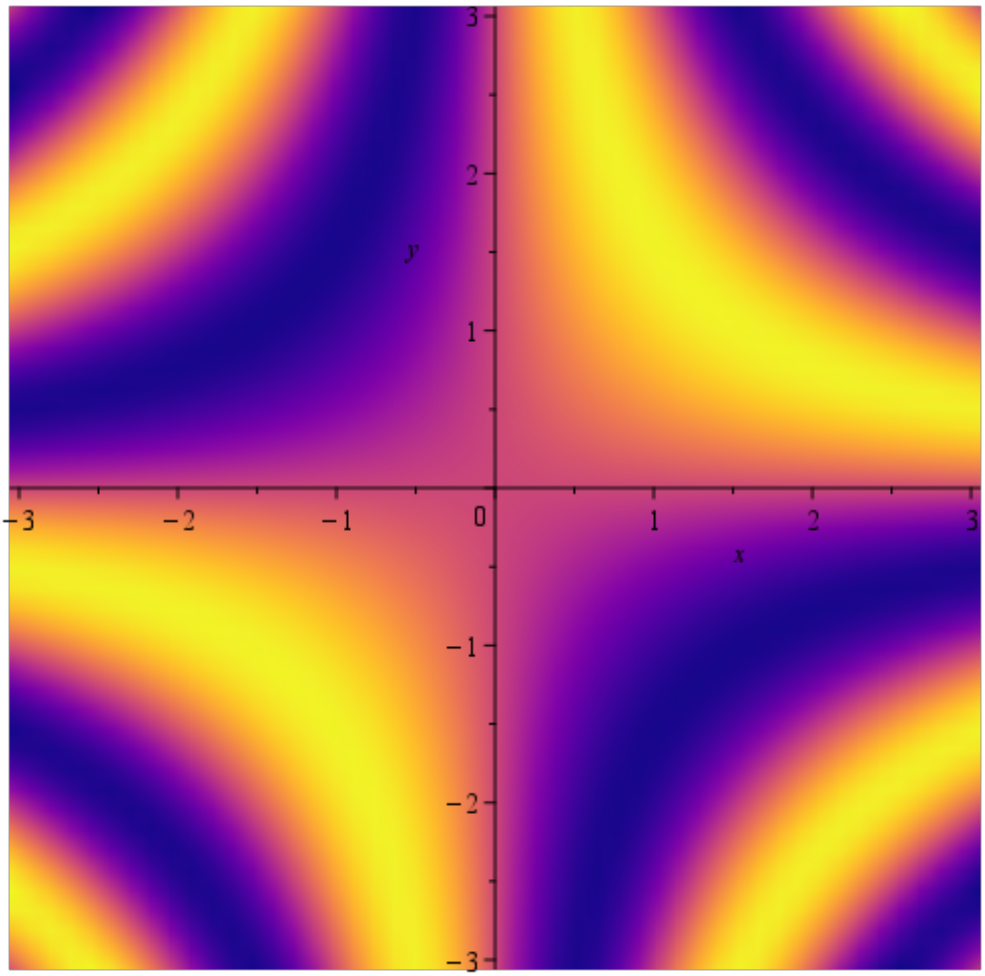
```
> implicitplot3d((10*x)^3 + (y/2)^3 + z^3 + 1 = (10*x + y/2 + z + 1)
  ^3, size = [700, 700]);
```



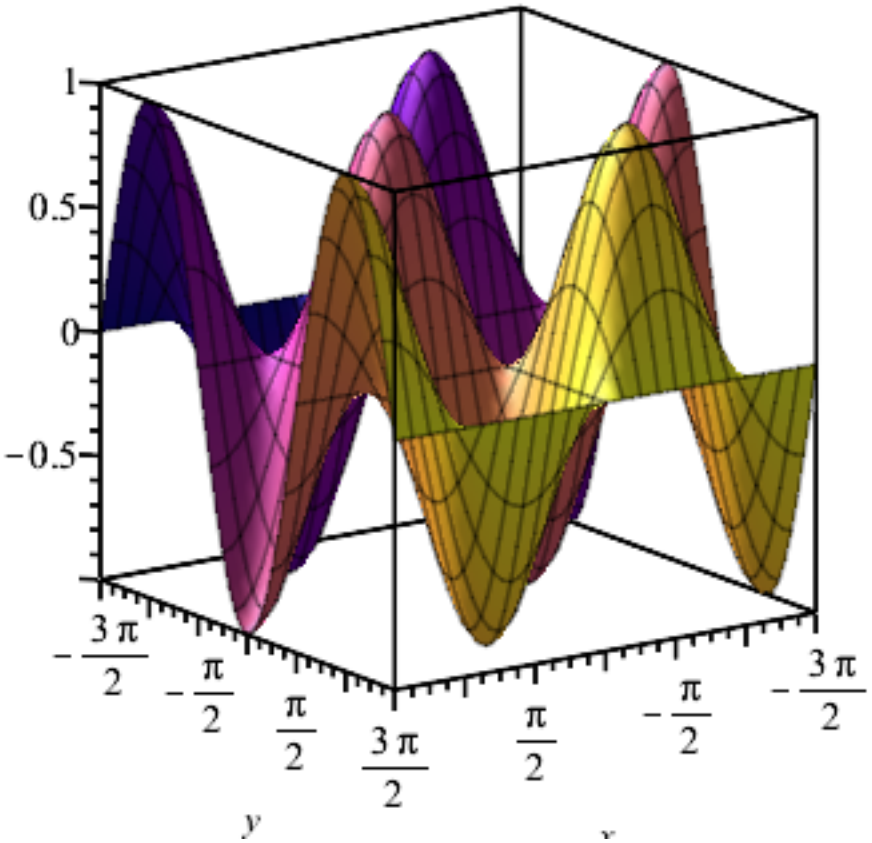
New Shortcuts for Specifying Plot Colors

- The plot gradient colorschemes now support using ColorTools[Palette] objects or names instead of a list of colors.

```
> plots:-densityplot(sin(x*y), x = -3..3, y = -3..3, 'colorscheme'=
  "Plasma", 'style'='surface');
```

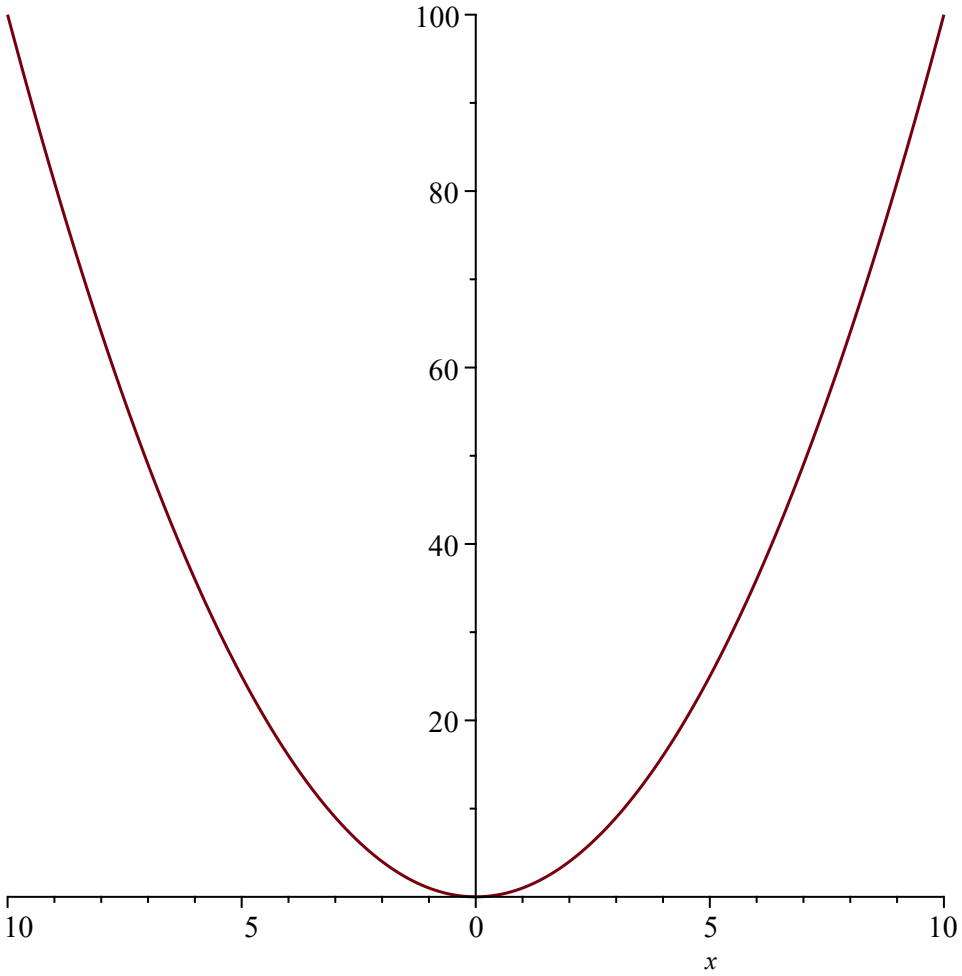


```
> plot3d(sin(x)*cos(y), x=-3/2*Pi..3/2*Pi, y=-3/2*Pi..3/2*Pi,
  'colorscheme'=["ygradient", "Plasma"]);
```



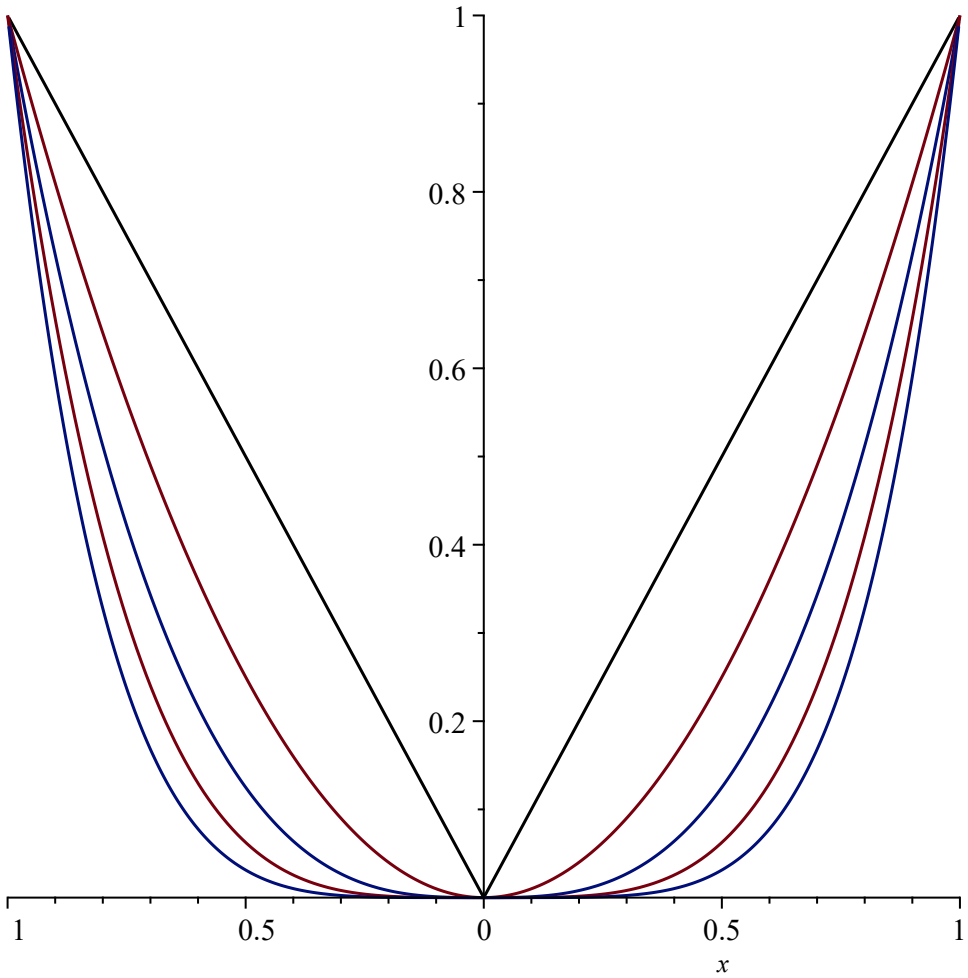
- The color option now accepts integers to select colors from the default list of colors set with plot,setcolors.

```
> plot(x^2, 'color'=1);
```



- This new syntax is especially useful for reusing the same colors when plotting multiple curves.

```
> plot([abs(x), x^2, abs(x^3), x^4, abs(x^5)], x=-1..1, color=[0, 1,
  2, 1, 2]);
```

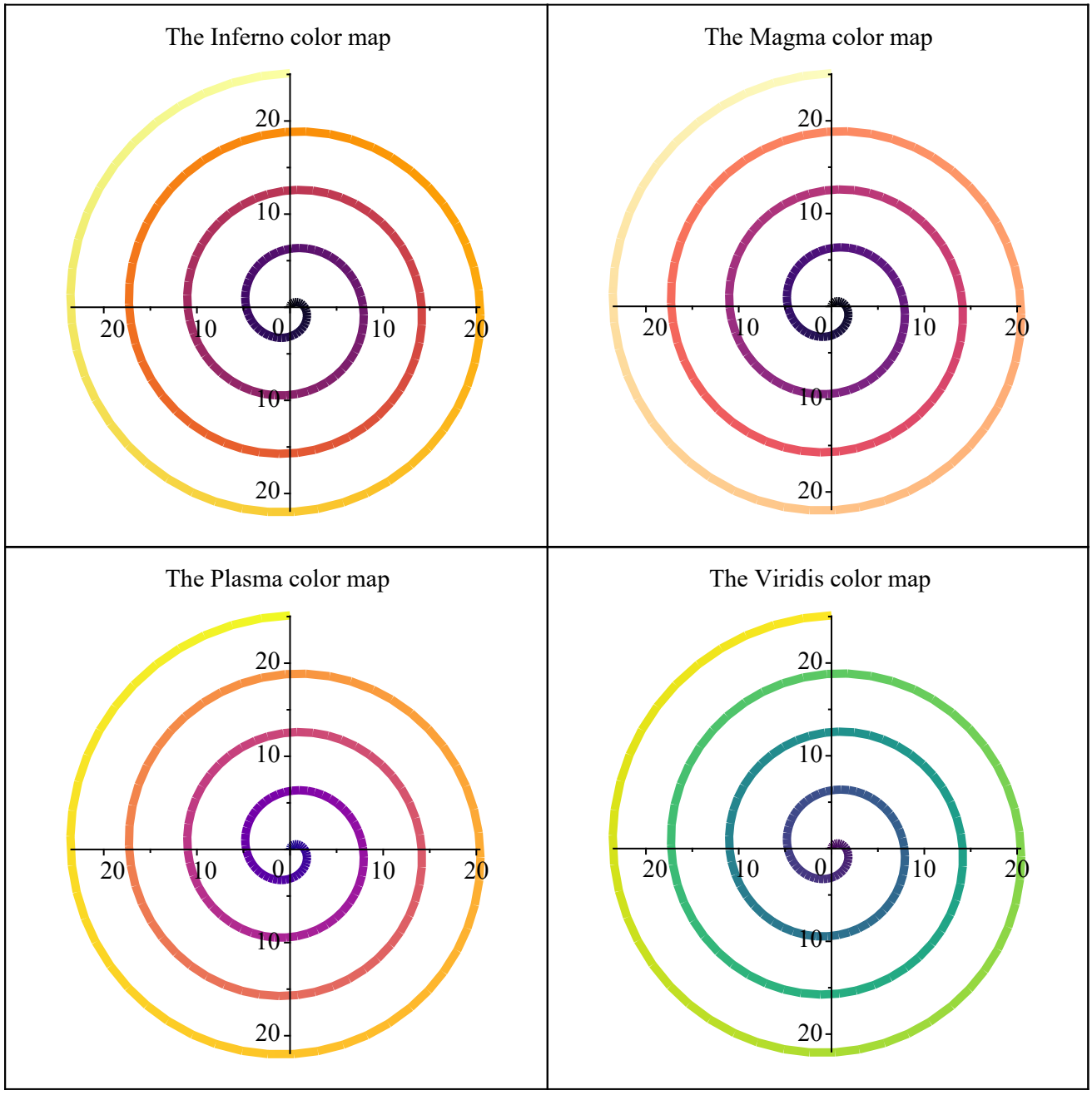


- For details, see plot/color.

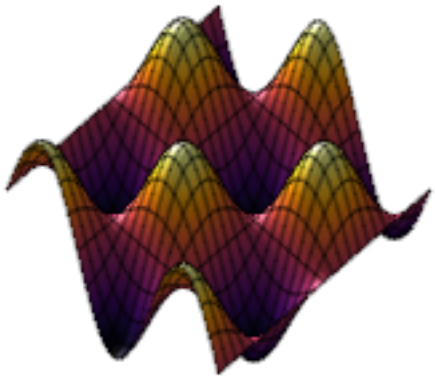
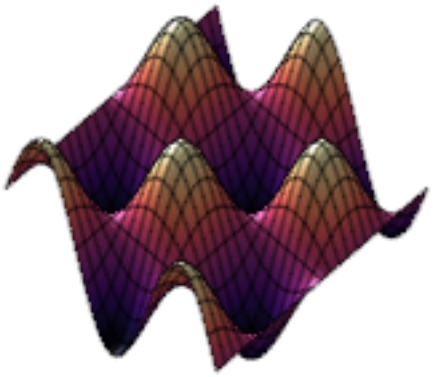
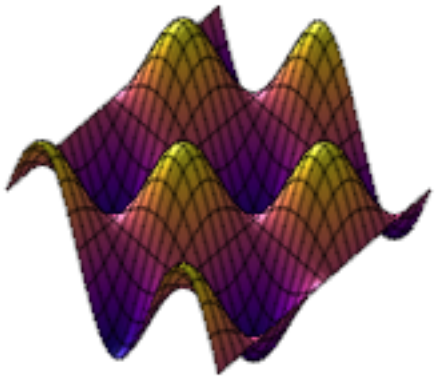
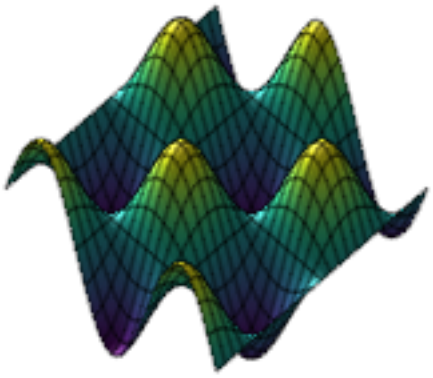
New Colormaps for Gradient Color Schemes

- The colormaps Inferno, Magma, Plasma, and Viridis are popular in many open source visualization libraries and are notable for being perceptually uniform even when viewed by persons with common forms of color vision deficiency and when printed to black-and-white. They are now included in Maple as ColorTools palettes so can be easily used with the colorscheme option.
- Here are all four new colormaps in 2-D and 3-D plots:

```
> plots:-display(Matrix(2,2,[seq(plot([x*sin(x), x*cos(x), x = 0 .. 8*
  Pi], colorscheme = m, thickness = 5, title = cat("The ", m, " color
  map"), scaling = constrained), m in ["Inferno", "Magma", "Plasma",
  "Viridis"])]));
```

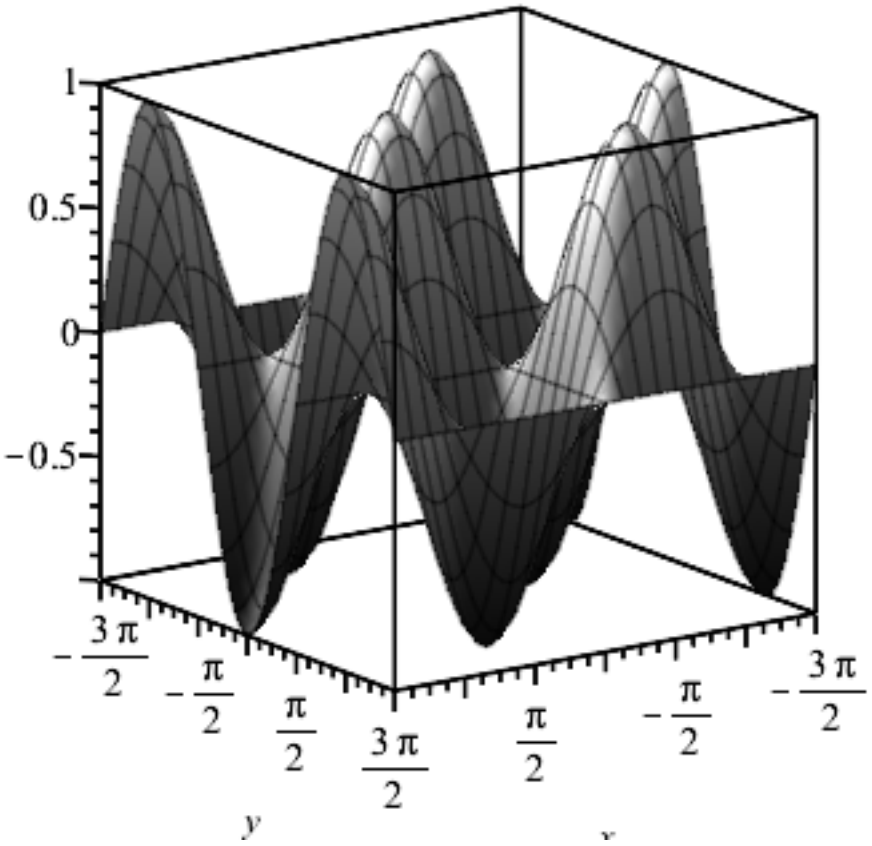


```
> plots:-display(Matrix(2,2,[seq(plot3d(sin(x)*cos(y), x = -3/2*Pi ..
  3/2*Pi, y = -3/2*Pi .. 3/2*Pi, orientation = [45, 30, 0],
  'colorscheme' = m, title = cat("The ", m, " color map"), axes =
  none), m in ["Inferno", "Magma", "Plasma", "Viridis"])]));
```

The Inferno color map 	The Magma color map 
The Plasma color map 	The Viridis color map 

• This also provide a convenient method for creating custom colormaps.

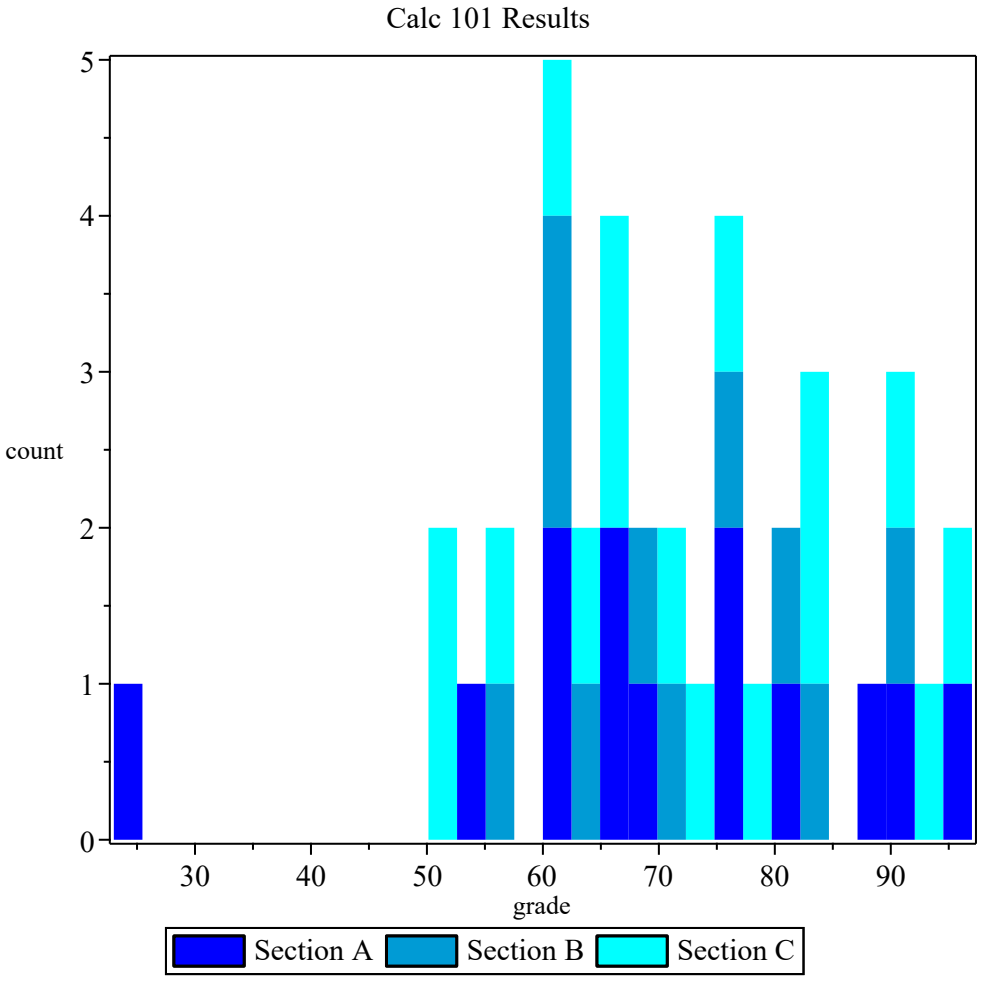
```
> monomap := ColorTools:-Palette([seq(sprintf("Grey %d",floor(3*i)-14)
  =ColorTools:-Color("Lab", [i,0,0]),i=5..90,numelems=256)]):
> plot3d(sin(x)*cos(y), x=-3/2*Pi..3/2*Pi, y=-3/2*Pi..3/2*Pi,
  'colorscheme'=monomap);
```



Stacked Option for Histograms

- The Histogram command in the Statistics package has a new option, format=stacked, which allows the combination of separate data sets into the same column bins on the histogram while maintaining a visual distinction.

```
> A := <23,97,82,77,65,54,61,75,92,67,68,61,88>:

> B := <90,61,83,81,72,77,64,56,60,69>:

> C := <51,97,92,57,75,94,52,66,73,84,67,83,78,72,63,62>:

> Statistics:-Histogram([A,B,C],frequencyscale=absolute,format=
  stacked, color = blue..cyan,labels=["grade","count"],legend=
  ["Section A","Section B","Section C"],title="Calc 101 Results");
```

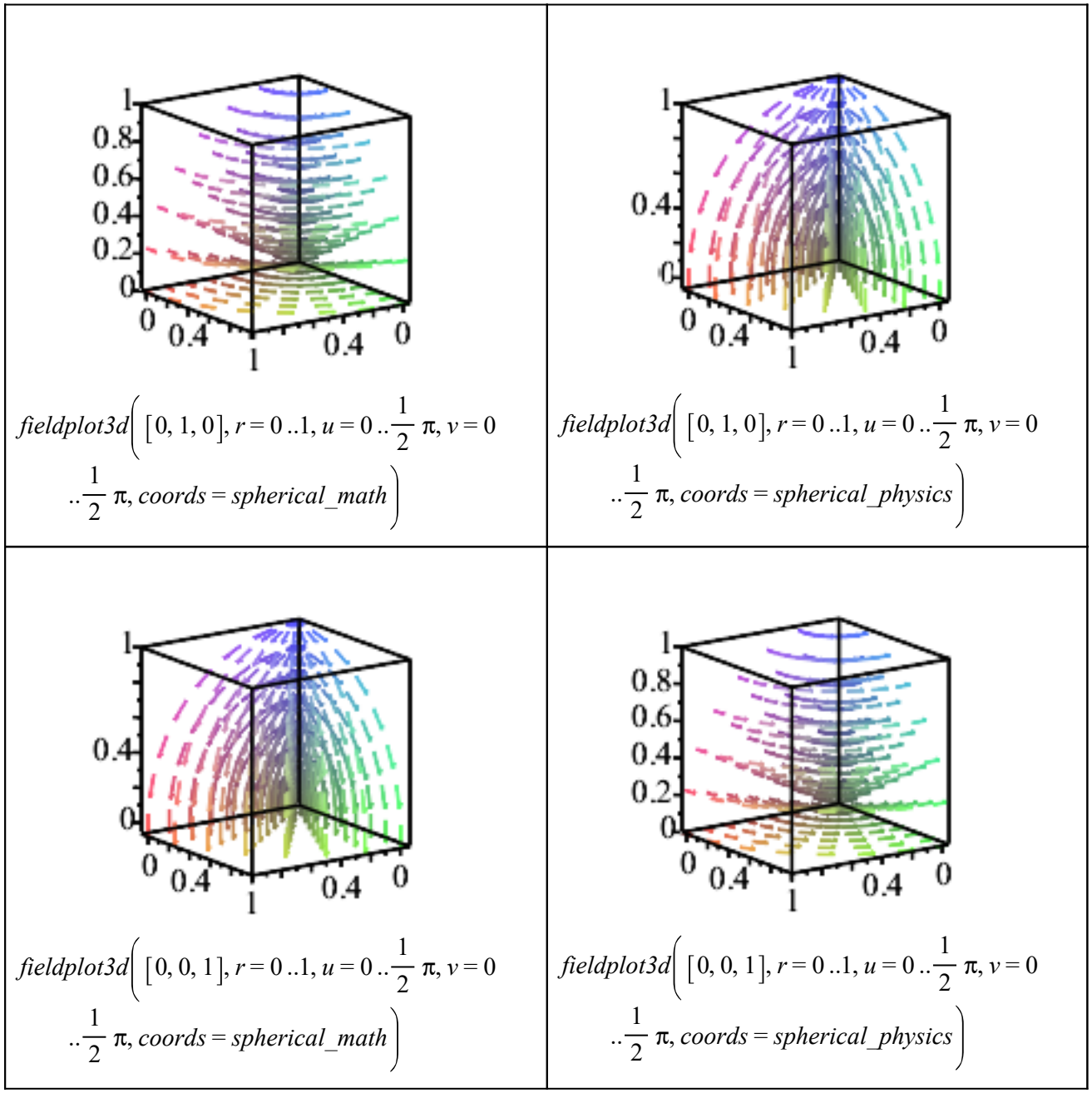


The Spherical Coordinate System

- There are two widely used conventions for how to represent spherical coordinates: one is used more in physics and the other more in mathematics. The only difference is a swap of the second and third coordinate: the vector represented by (u, v, w) in one convention is represented by (u, w, v) in the other.
- The plotting library and the changecoords command now understand two new coordinate systems, called spherical_math and spherical_physics, which can be used to refer to these coordinate systems unambiguously. In the spherical_math coordinate system, the second coordinate is the azimuthal angle (the angle in the XY plane) and the third coordinate is the altitude angle (the angle out of that plane), whereas in the spherical_physics coordinate system, the second coordinate is the altitude and the third is the azimuth.
- The interpretation of the spherical coordinate system has been modified for Maple 2022 in the plotting library and the changecoords command. In previous versions, the plotting library and the changecoords command always interpreted spherical as spherical_math whereas the Physics and VectorCalculus packages always interpreted it as spherical_physics. In Maple 2022:
  - If neither of the Physics and VectorCalculus packages has been loaded (using the with command), then the plotting library and the changecoords command will interpret spherical as spherical_math, as in earlier versions of Maple.
  - If either of the packages Physics or VectorCalculus *has* been loaded, then the plotting library and the changecoords command will interpret spherical as spherical_physics.
  - The Physics and VectorCalculus packages will always interpret spherical as spherical_physics, as in earlier versions of Maple.
- As an illustration, the following shows two different field plots, each obtained in two different ways, by plotting the field given by the azimuth and altitude angles, in both the spherical_physics and spherical_math coordinate systems.

```
> make_fieldplot3d := proc(unit :: identical(2, 3), system :: name, $)
   local arguments := [ifelse(unit = 2, [0, 1, 0], [0, 0, 1]),
     r = 0 .. 1, u = 0 .. Pi/2, v = 0 .. Pi/2, coords = system];
   return plots:-fieldplot3d(op(arguments), caption = ':-fieldplot3d'
  (op(arguments)));
   end proc:

> plot_matrix := Matrix(2, (i, j) -> make_fieldplot3d(i + 1,
  [spherical_math, spherical_physics][j])):
```

```
> plots:-display(plot_matrix);
```




```
>
```

Visualizations with Units

- Many visualizations with units have been updated. See the units update page for more details.