Comparison of Graphics Capabilities Mathematica 13.2 and Maple 2022

Summary

As in many areas of functionality, apparent similarities in visualization capabilities between Maple and Mathematica are only skin deep.

- Mathematica automates more of the graphic creation to give more accurate or more understandable results in more cases.
- Mathematica automates more sensible aesthetic choices for professional-looking results.
- Mathematica supports a much wider range of visualization routines.
- Mathematica visualizations make more use of dynamic elements for richer electronic presentations.
- Mathematica visualizations support a greater range of inputs.

If you want to produce professional, publication-quality graphics that are clear and accurate with the minimum amount of effort, then Mathematica is the obvious choice.

Clarity of presentation

The central purpose of visualization is to present information in an easily interpreted form. Important features must be clearly presented, accurate and easy to read.

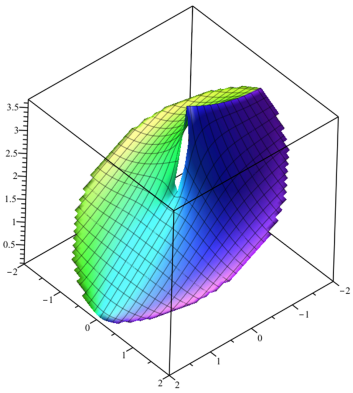
Adaptive sampling

Mathematica function visualizations use adaptive resampling to achieve smooth results even where functions are rapidly changing, without the computational overhead of extra sampling where the function is not changing rapidly.

Some Maple visualizations do this but not all. For example, without this capability, Maple is unable to establish a smooth boundary for this complex plot.

Maple

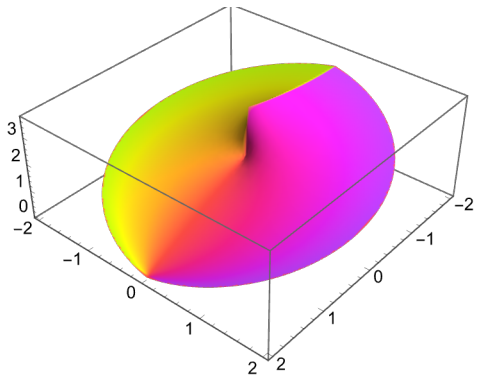
```
f := z ↦ if abs(z) < 2 then 2 − abs(z) + I argument(z) else NaN end if
                    f := z ↦ if |z| < 2 then 2 − |z| + I·arg(z) else NaN end if
complexplot3d(f, −2 − 2 I..2 + 2 I)
```



Mathematica

Mathematica automatically traces the edges and steep parts of the surfaces more carefully.

```
ComplexPlot3D[If[Abs[z] < 2, 2 - Abs[z] - I Arg[z], Missing[]], {z, -2 - 2 I, 2 + 2 I}]
```



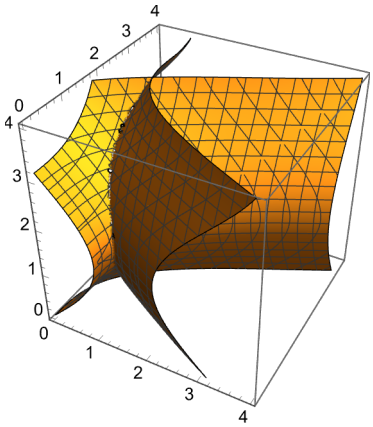
Using mesh lines to enhance interpretation

Mesh lines are a very important part of conveying meaning in 3D graphics, but in Maple they are simply a side effect of sampling.

In this simple implicit equation plot, Mathematica's choice of mesh lines enhances our understanding of the surface curvature.

Mathematica

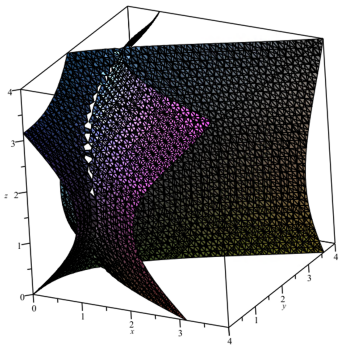
```
ContourPlot3D[Sin[x + Sin[y]] == Sin[y + Sin[z]], {x, 0, 4}, {y, 0, 4}, {z, 0, 4}]
```



Maple

However, in Maple, many unnecessary mesh lines obscure that information. Combined with the lack of adaptive sampling, especially around the intersection of the two surfaces, this makes the plot difficult to interpret.

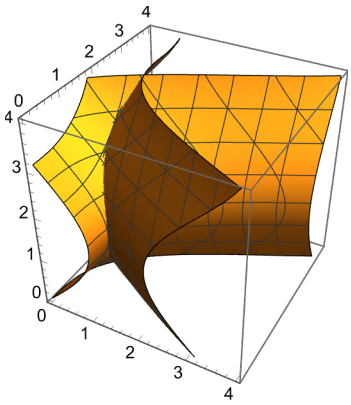
$implicitplot3d(\sin(x + \sin(y)) = \sin(y + \sin(z)), x = 0..4, y = 0..4, z = 0..4)$



Mathematica

In Maple, increasing the sampling introduces even more mesh lines until they obscure the plot itself. The only recourse is to turn mesh lines off completely. In Mathematica, mesh lines are independent of sampling, so we can simultaneously increase the quality of the sampling and choose a sparser set of mesh lines.

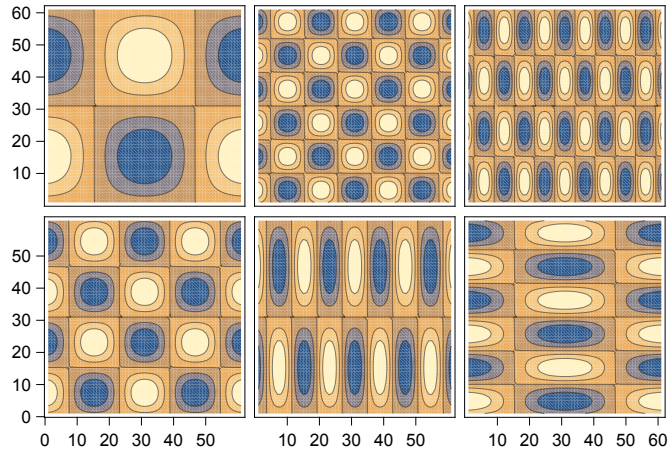
```
ContourPlot3D[Sin[x + Sin[y]] == Sin[y + Sin[z]],
 {x, 0, 4}, {y, 0, 4}, {z, 0, 4}, PlotPoints → 25, Mesh → 6]
```



Axes control

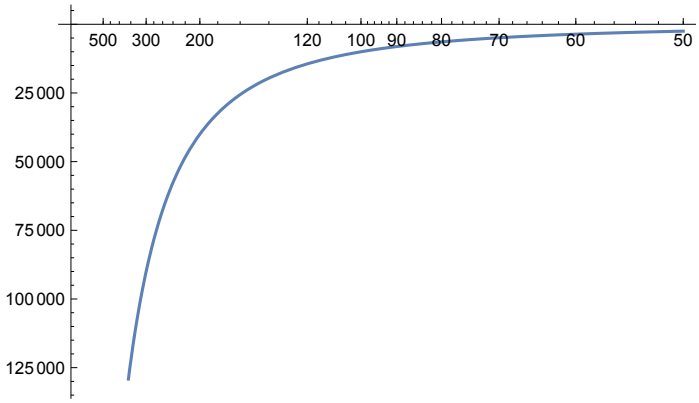
Many Mathematica graphics support paneled layout with shared axes.

```
ListContourPlot[{sin(x) cos(y), sin(2 x) cos(2 y), sin(3 x) cos(3 y), sin(x) cos(4 y),
  sin(2 x) cos(5 y), sin(3 x) cos(y)}, ..., PlotLayout → {"Column", UpTo[4]}]
```



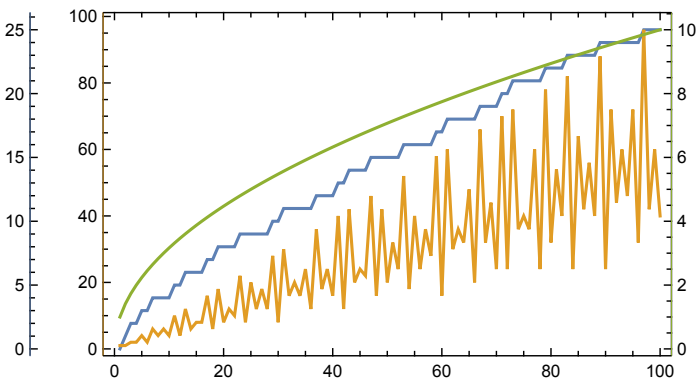
Axes can take individual scaling directives, which can include `"Log"`, `"Reverse"`, `"Reciprocal"`, `"Date"`, `NominalScale`, `OrdinalScale` or an abitrary function. Maple supports only `"log"`.

```
Plot[x^2, {x, 50, 1000}, ScalingFunctions → {"Reciprocal", "Reverse"}]
```



Many plots support any number of axes.

```
ListLinePlot[{PrimePi[Range[100]], EulerPhi[Range[100]], Sqrt[Range[100]]},
 MultiaxisArrangement → All]
```

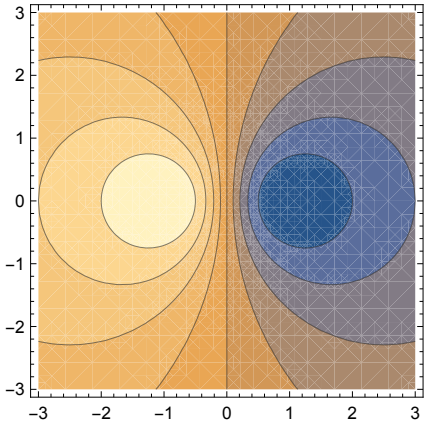


Enhancing interpretation with color

Simple use of shading makes it easier to understand that this contour plot represents a peak on the left and a valley on the right.

Mathematica

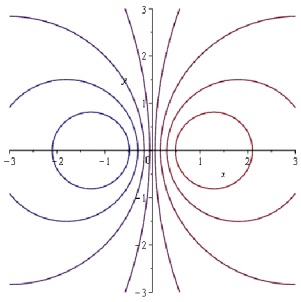
```
ContourPlot[-(5 x)/(x^2 + y^2 + 1), {x, -3, 3}, {y, -3, 3}]
```



Maple

In Maple, default coloring is more subtle and uses lighter shades to represent lower values.

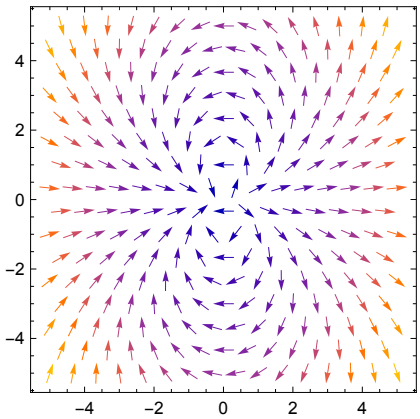
$with(plots) : contourplot\left(-\frac{5\,x}{x^2+y^2+1}, x=-3..3, y=-3..3\right)$



Mathematica

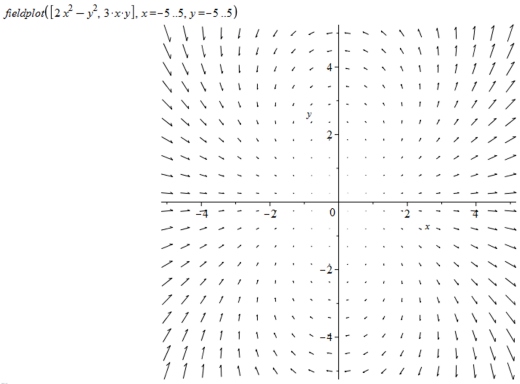
In vector plots, Mathematica makes use of color, by default, to indicate magnitude.

```
VectorPlot[{2 x^2 - y^2, 3 x y}, {x, -5, 5}, {y, -5, 5}]
```



Maple

In Maple, little can be understood from the central area of the plot.

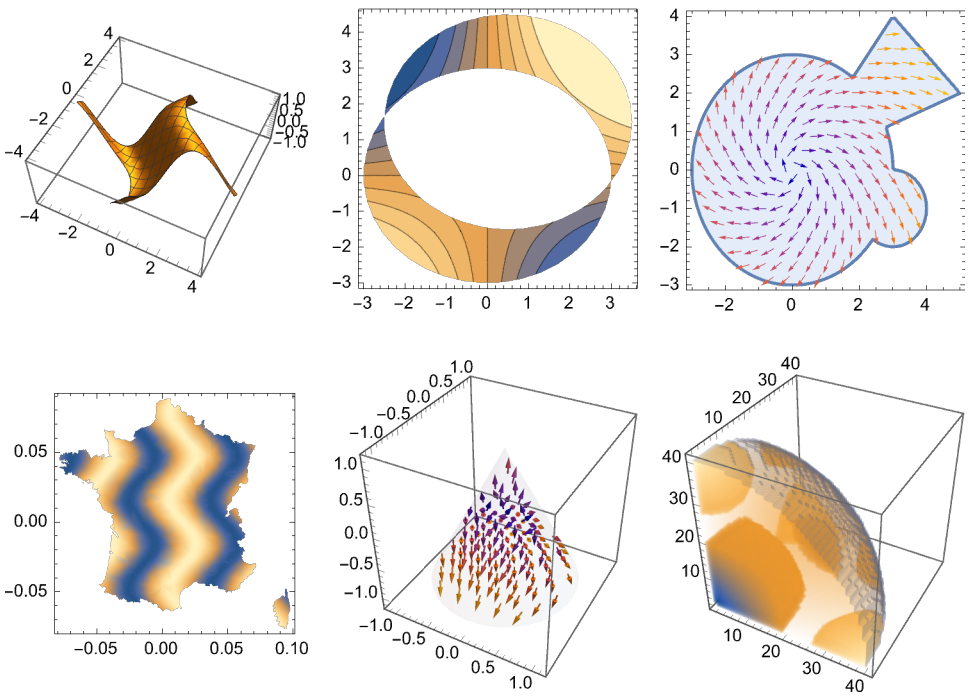


Plot region support

Often, the region in which data or functions are valid is an important piece of context, and visualizations should be able to present that information. In Maple, the region over which visualizations can be created is always either rectangular or (for function plots) one in which the second independent variable is an interval that is a function of the first variable. Mathematica visualizations can be created over any region, specified implicitly or explicitly, or by using geometric constructs or arbitrary meshes. The following visualizations would be very hard to create in Maple.

Mathematica

```
GraphicsGrid[{{Plot3D[Sin[x + Sin[y]],
    {x, y} ∈ ImplicitRegion[Abs[x] < 4 && Abs[y] < 4 && Abs[x y] < 1, {x, y}], ViewPoint → {1, -2, 3}],
   ContourPlot[Sin[x y / 5], {x, y} ∈ RegionSymmetricDifference[Disk[{0, 0}, 3], Disk[{0.5, 1.5}, 3]]], VectorPlot[
    {x + y, y - x}, {x, y} ∈ RegionUnion[Disk[{0, 0}, 3], Disk[{3, -1}, 1], Polygon[{{0, 0}, {5, 2}, {3, 4}}]]]},
  {DensityPlot[Sin[100 x + Sin[100 y]], {x, y} ∈ CountryData["France", "Shape"][[1, 3]]],
   VectorPlot3D[{x y, y, z}, {x, y, z} ∈ Cone[], VectorPoints → 8, RegionBoundaryStyle → Opacity[0.05]],
   ListDensityPlot3D[data, RegionFunction → Function[{x, y, z}, x^2 + y^2 + z^2 ≤ 1680]]}}, ImageSize → 500]
```

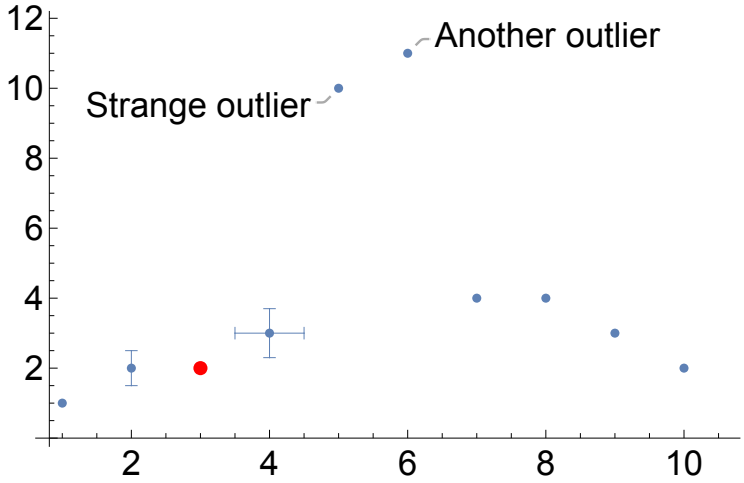


Labeling

While both Maple and Mathematica can be programmed to place text on a graphic, only Mathematica automates this process with symbolic wrappers for data points. Labels, callouts, tooltips, status area

updates, error bars and mouseover effects are supported. Callout placement is automated to avoid overlapping text. Maple supports only tooltips.

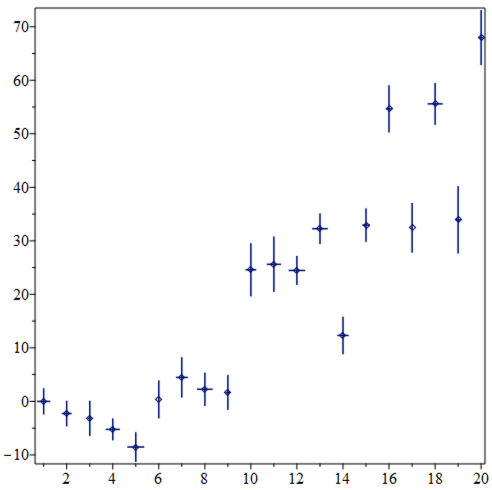
```
ListPlot[{1, Around[2, 0.5], Style[2, Red, PointSize[0.02]],
  Around[3, {{0.5, 0.5}, {0.7, 0.7}}], Callout[10, "Strange outlier"],
  Callout[11, "Another outlier"], 4, 4, 3, 2}, PlotRange → All, BaseStyle → 18]
```



Error bars

Maple provides only the most basic support for error bars. Errors must be uncorrelated, data must be equally spaced and only a single type of visualization and error bar is provided.

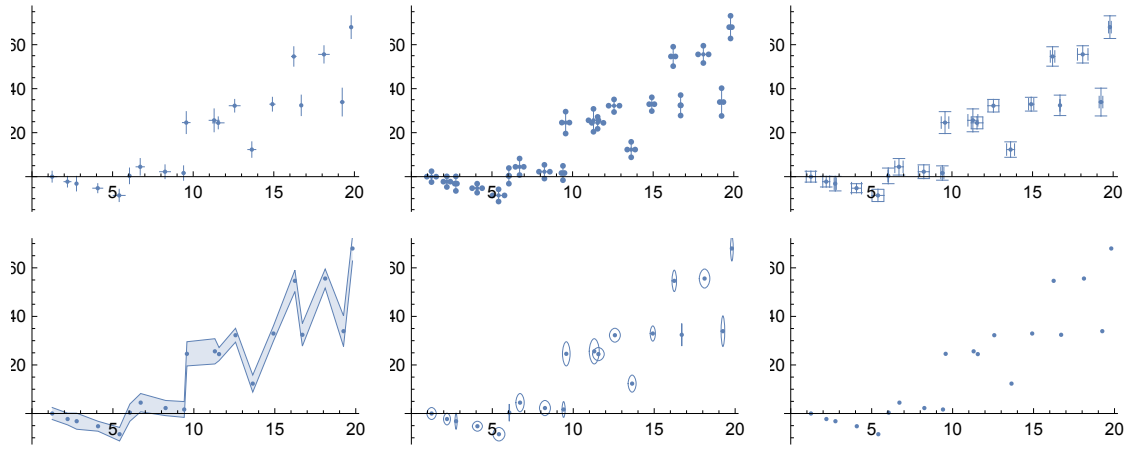
Maple



Mathematica

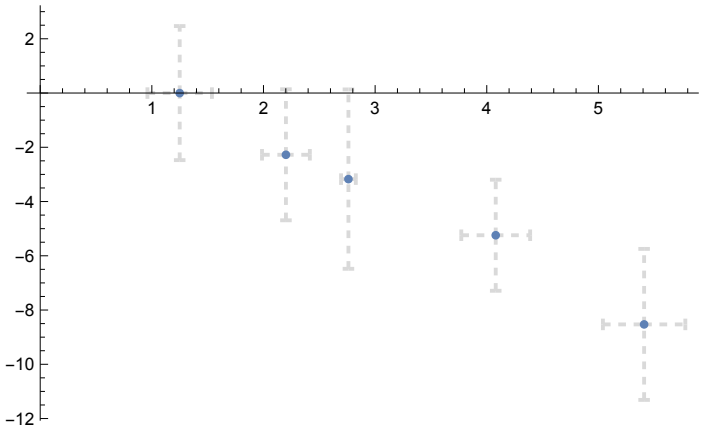
Like many of Mathematica's visualization capabilites, interval markers are provided in a range of built-in styles.

```
GraphicsGrid[Partition[Table[ListPlot[Data, IntervalMarkers → i, PlotStyle → Thin],
    {i, {"Bars", "Points", "Fences", "Bands", "Ellipses", None}}], 3]]
```



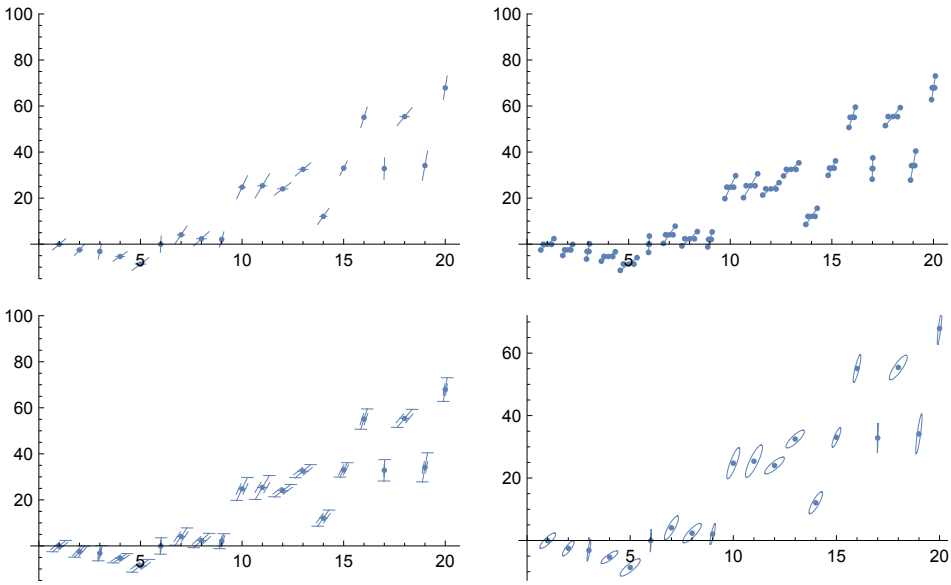
They are fully customizable, while Maple only lets you choose their color.

```
ListPlot[Take[Data, 5], IntervalMarkersStyle → Directive[ LightGray, Thick, Dashed]]
```



Mathematica supports correlated errors.

```
GraphicsGrid[Partition[Table[ListPlot[Data, IntervalMarkers → i],
    {i, {"Bars", "Points", "Fences", "Ellipses"}}], 2]]
```



As well as `ListPlot`, uncertain data is supported in other Mathematica visualizations.

Robustness of application

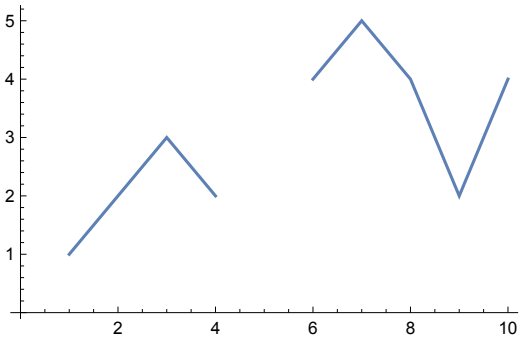
Mathematica visualizations are designed to handle real-world problems that are not always ideally posed. They robustly handle all kinds of potential problems in the data or functions.

Missing data

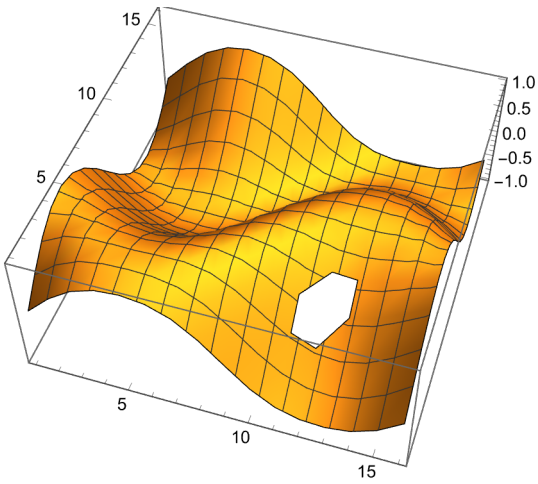
Mathematica

Data plots in Mathematica automatically skip over unplottable points such as symbols or NaN values in data plots.

```
ListLinePlot[{1, 2, 3, 2, Missing[], 4, 5, 4, 2, 4}]
```




```
data = Table[Sin[x + Sin[y]], {x, 0, 6, 0.4}, {y, 0, 6, 0.4}];
data[[4, 12]] = x;
ListPlot3D[data]
```



Maple

In each case, a single bad value in a dataset causes Maple to abandon the entire visualization. It is your responsibility to check and clean data before attempting to visualize it in Maple.

```
> listplot([1, 2, 3, 2, missing( ), 4, 5, 4, 2, 4])
Error, (in plots:-pointplot) points cannot be converted to floating-point values
```

$data := [seq([seq(\sin(x + \sin(y)), x = 0..6, 0.4)], y = 0..6, 0.4)] : data_{4,12} := x :$

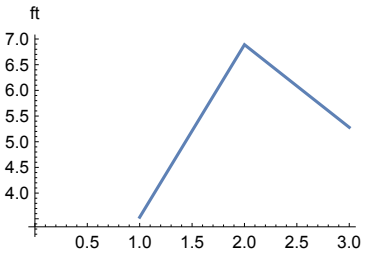
```
with(plots) : matrixplot(data)
Error, (in plots/matrixplot) cannot convert first argument to a floating-point
matrix
```

Units

Mathematica graphics can accept data with associated units, automatically converting to a common unit system.

Mathematica

```
ListLinePlot[{3.53 ft , 2.1 m , 0.001 mi }, AxesLabel → Automatic]
```



Maple

While 2D function plots in Maple can use units to label axes, data plots cannot handle data with associated units. Maple requires that you specify the units you wish to convert to, then strip out the units before plotting and then specify the units back into the axes labels.

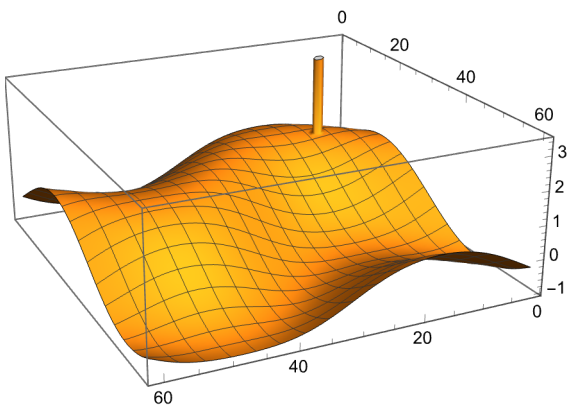
```
> listplot([3.53 [['ft']], 4.1 [['m']]])
Error, (in plots:-pointplot) points cannot be converted to floating-point
values
```

Automatic plot ranges

Automatic plot range choices ensure that the maximum amount of useful information is included in the plot. In this example, we generate a table of values for a curved surface with a single, strong outlier. Mathematica's automatic plot range preserves most of the interesting detail in the image at the expense of the outlier.

Mathematica

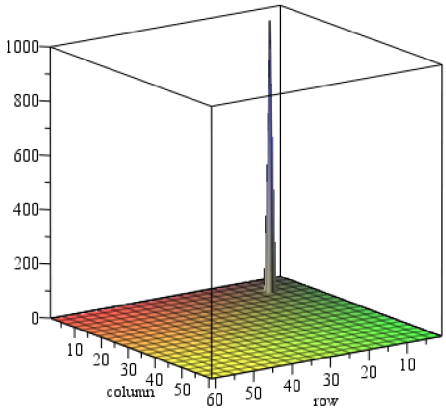
```
data = Table[Sin[x + Sin[y]], {x, 0, 6, 0.1}, {y, 0, 6, 0.1}];
data[[10, 10]] = 1000;
ListPlot3D[data]
```



Maple

However, all useful detail in the Maple plot is lost in order to include the single outlier.

```
data := [seq([seq(sin(x + sin(y)), x = 0..6, 0.1)], y = 0..6, 0.1)] :
data[10, 10] := 1000 :
with(plots) :
matrixplot(data)
```



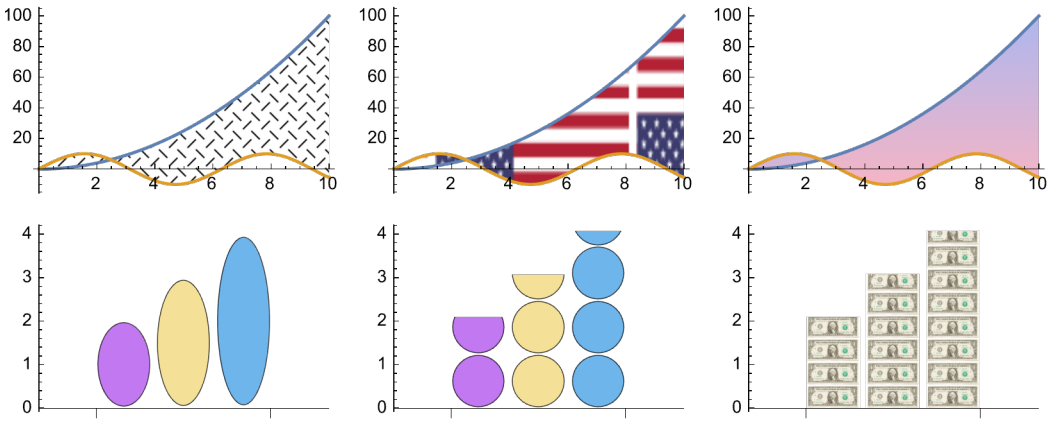
Aesthetic control

As well as ensuring that you get reliable informative visualizations, Mathematica is the right choice for creating beautiful, impactful and professional-looking visualizations. In addition to the carefully designed automatic choices for each visualization, Mathematica provides more detailed customization over the appearance of every detail, so that you can make your visualizations appear exactly the way that you want them. Some of the options that Maple does not provide its users include:

Mathematica

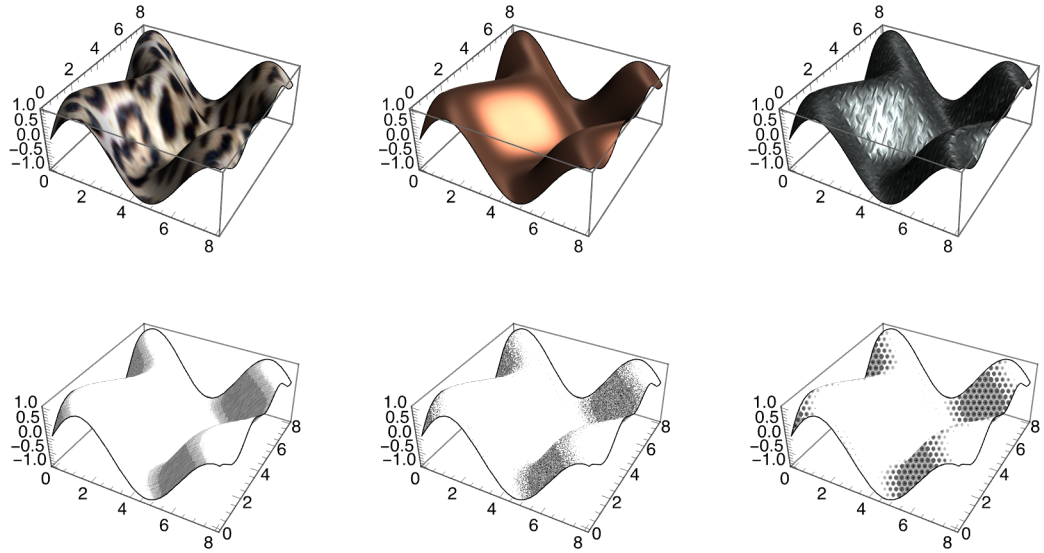
Many 2D plots allow arbitrary fills.

```
GraphicsGrid[{Plot[{x^2, 10 Sin[x]}, {x, 0, 10}, Filling → {1 → {2}}, FillingStyle → #] & /@
    {PatternFilling["DiamondPlate", 10],
     PatternFilling[], LinearGradientFilling[{■, ■}, Top]},
   BarChart[{2, 3, 4}, ChartElements → #, ChartStyle → "Pastel"] & /@
    {{●, All}, ●, }}]
```



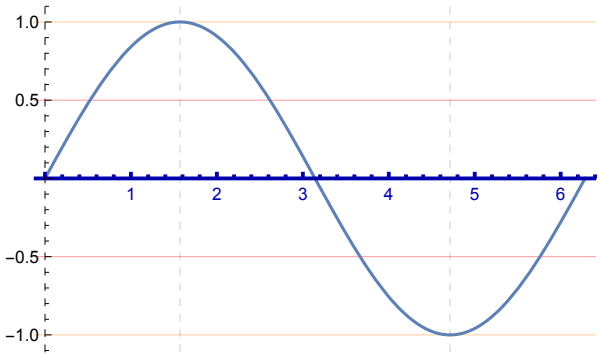
Mathematica provides tools for shading 3D plots for both color and monochrome output.

```
GraphicsGrid[{Plot3D[Sin[x + Sin[y]], {x, 0, 8}, {y, 0, 8}, PlotStyle → MaterialShading[#],
    Mesh → False, Lighting → "ThreePoint", PlotPoints → 80] & /@
   {<|"BaseColor" → Texture[ ]|>, "Copper", <|"BaseColor" → Texture[ ],
    "SurfaceNormals" → Texture[ ], "RoughnessCoefficient" → 0.65,
    "MetallicCoefficient" → Texture[ ]|>},
  Plot3D[Sin[x + Sin[y]], {x, 0, 8}, {y, 0, 8}, PlotStyle → #,
    Lighting → "Neutral", Mesh → None, PlotTheme → "Monochrome"] & /@
   {HatchShading[], StippleShading[], HalftoneShading[]}}]
```



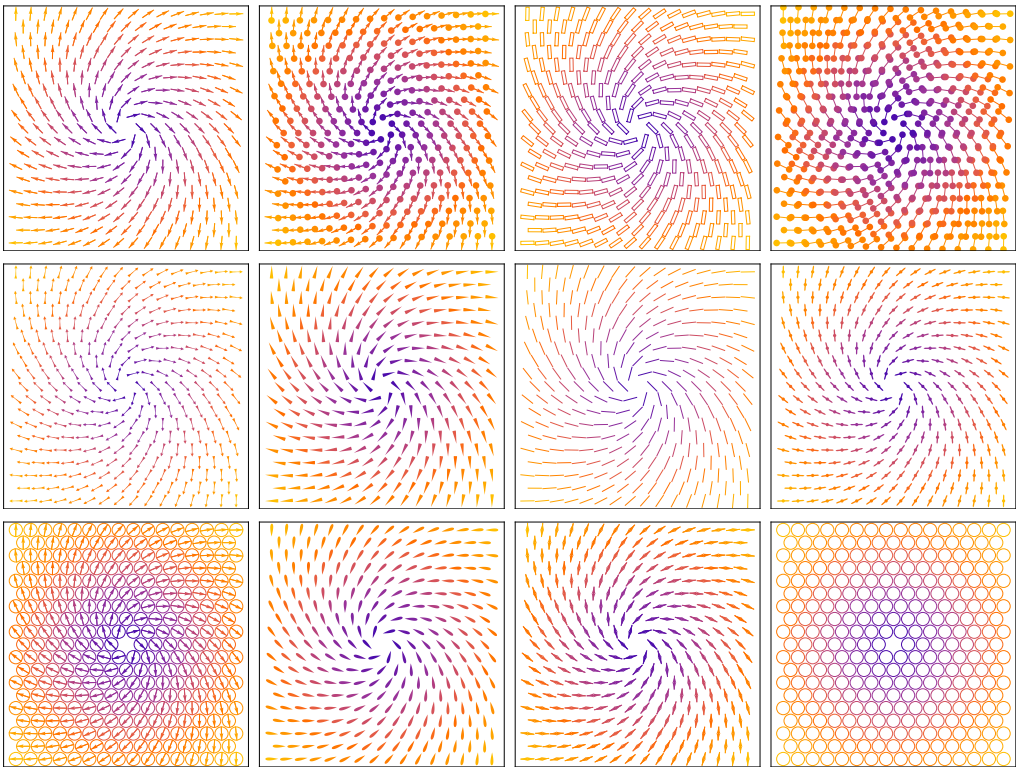
Grid lines and axes can be individuall specified in both location and style.

```
Plot[Sin[x], {x, 0, 2 π}, AxesStyle → {Directive[Thick, Darker@Blue], Dashed},
 GridLines → {{{Pi / 2., Dashed}, {3 / 2 Pi, Dashed}},
   {{-1, Orange}, {-0.5, Red}, {0.5, Red}, {1, Orange}}}]
```

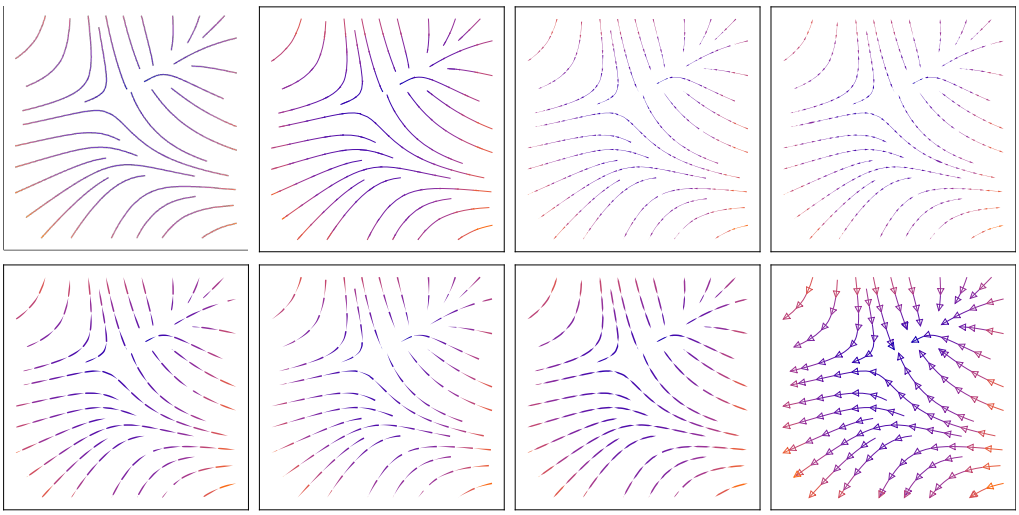


Vector and stream plots provide a range of different vector marker styles.

```
Multicolumn[
 Table[VectorPlot[{x + y, -x + y}, {x, -3, 3}, {y, -3, 3}, VectorMarkers → s, ImageSize → 128,
   FrameTicks → None], {s, {"Arrow", "Dart", "CircleArrow", "DotArrow", "Pointer",
    "Drop", "Box", "Segment", "ArrowArrow", "DotDot", "BarDot", "Circle"}}]]
```



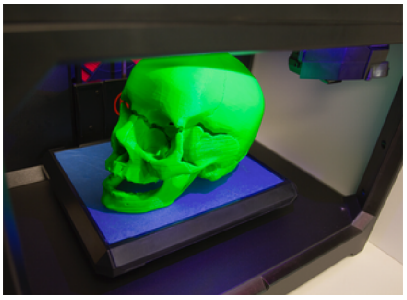
```
Multicolumn[
 Table[StreamPlot[{-1 - x^2 + y, 1 + x - y^2}, {x, -3, 3}, {y, -3, 3},
   StreamStyle → s, FrameTicks → None, ImageSize → 128, StreamPoints → Coarse],
  {s, {"Line", "Arrow", "Dart", "DoubleDart", "Drop", "Pointer", "Toothpick",
    Arrowheads[{{.025, 1, {▷, 0}}}]}}], 4, Appearance → "Horizontal"]
```



3D printing support

Mathematica provides fully integrated capabilities to directly 3D print geometric models, using either an online printing service or your own printer. You can algorithmically generate geometric models or import and transform 3D models from files and immediately output physical 3D objects.

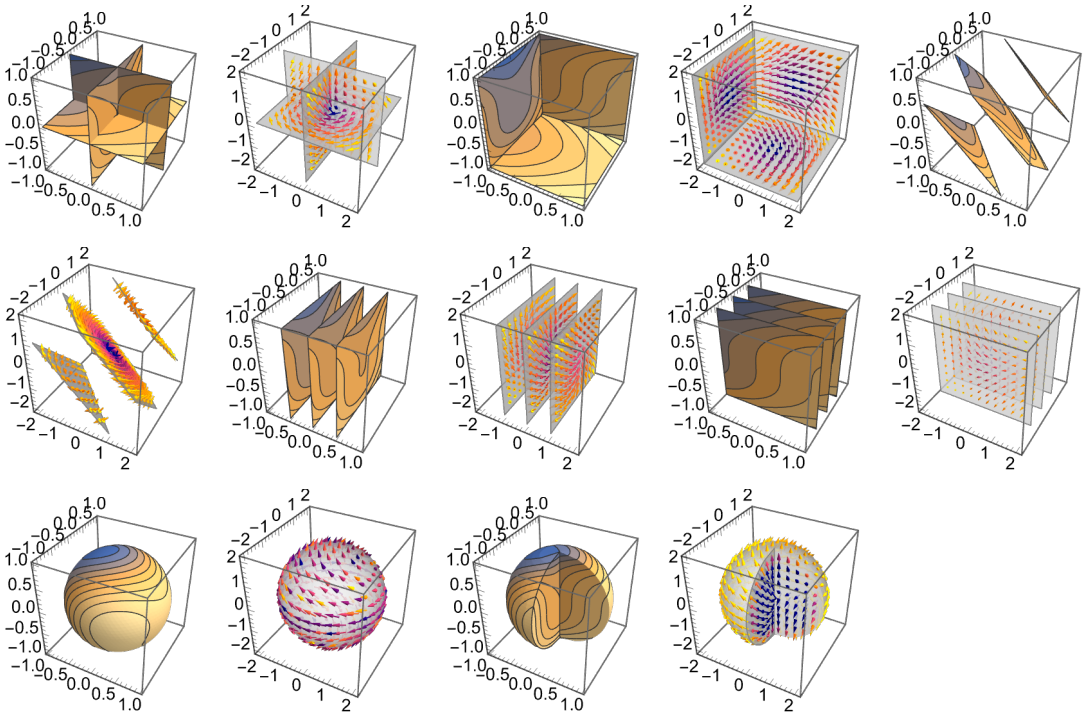
```
Printout3D[  ]
```



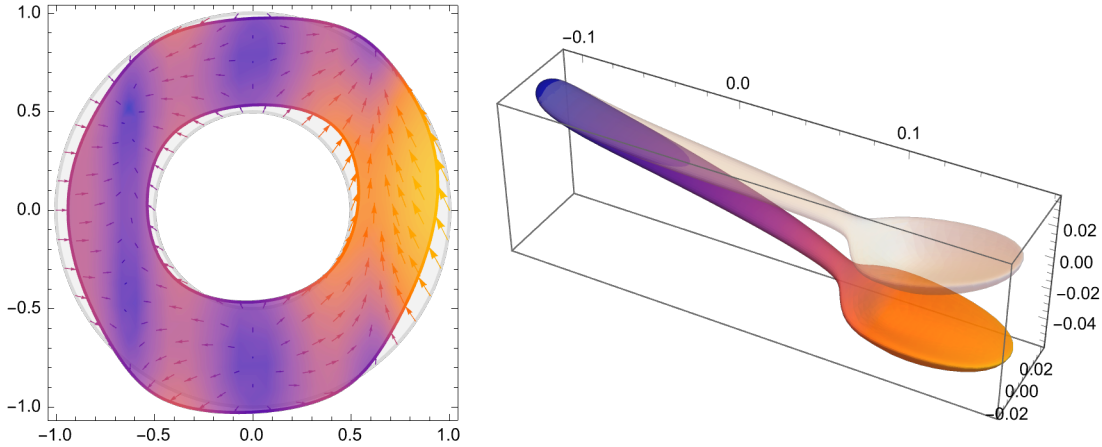
Breadth of capability

The range of built-in visualization types is much larger in Mathematica than Maple.
*Maple has no direct way to produce **any** of the following visualizations.*

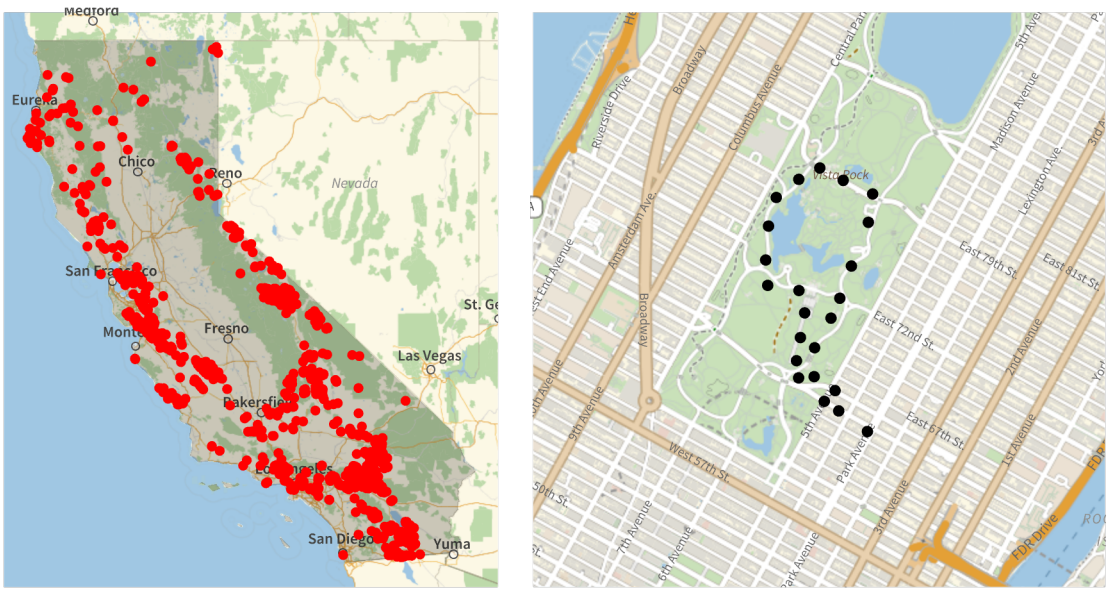
```
Multicolumn[Flatten[Table[{SliceContourPlot3D[
      Sin[x] + y^2 - z^3, sl, {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, ImageSize → 110],
     SliceVectorPlot3D[{y, -x, z}, sl, {x, -2, 2}, {y, -2, 2}, {z, -2, 2}, ImageSize → 110]},
    {sl, {"CenterPlanes", "BackPlanes", "DiagonalStackedPlanes", "XStackedPlanes",
      "YStackedPlanes", "CenterSphere", "CenterCutSphere"}}]], 5, Appearance → "Horizontal"]
```



```
GraphicsRow[{VectorDisplacementPlot[
    {Sin[5 x], x + Cos[2 y]}, {x, y} ∈ Annulus[], VectorPoints → Automatic],
  VectorDisplacementPlot3D[displacements, {x, y, z} ∈ ]}]
```

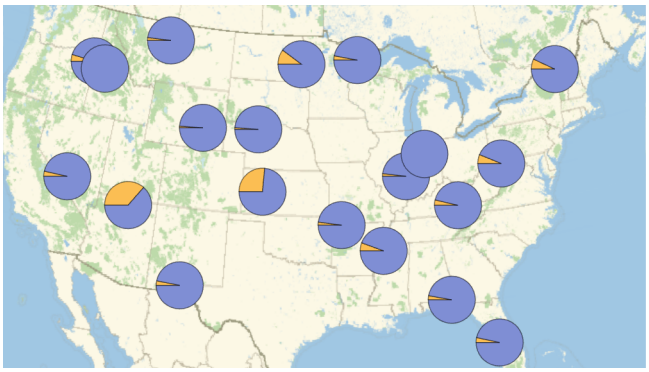



```
GraphicsRow[
 {GeoGraphics[{Polygon[California, United States ADMINISTRATIVE DIVISION], Red, PointSize[.02],
    Point[EarthquakeData[California, United States ADMINISTRATIVE DIVISION,
      4, {{1980}, {2015}}, "Position"]["Values"]]}],
  GeoGraphics[{PointSize[0.02], Point[GPS Data]}, GeoRange → 1 mi ]}]
```

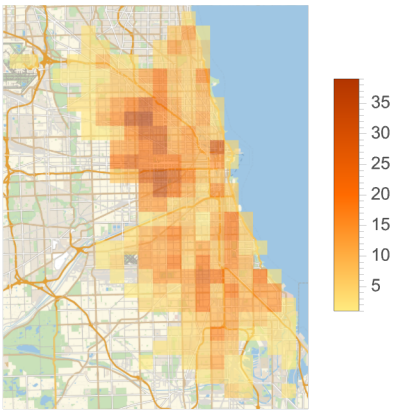


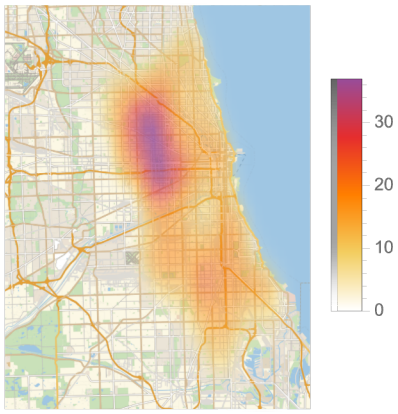
```
GraphicsRow[GeoBubbleChart[Data +, GeoBackground → #] & /@
  {"StreetMap", "Satellite", "ReliefMap"}, 3]
```


```
PointValuePlot[Table[RandomGeoPosition[United States COUNTRY] →
    {RandomInteger[3], RandomReal[100]}, 20], PlotLegends → Automatic]
```

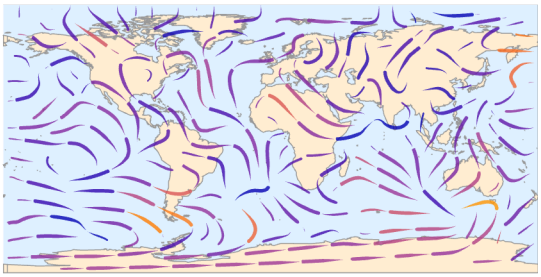


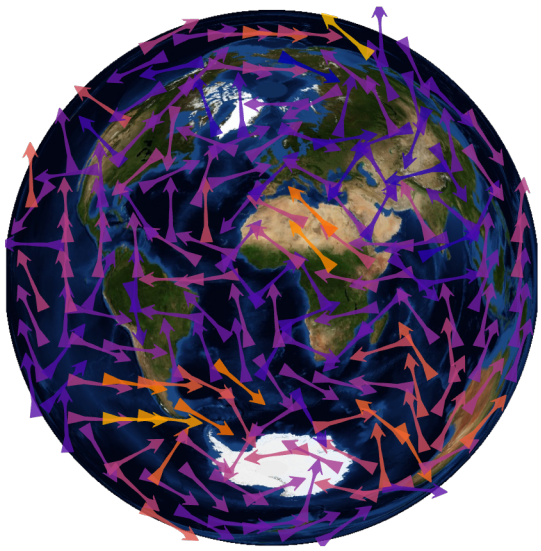
```
GraphicsRow[{GeoHistogram[Auto Theft Locations +,
    {"Rectangle", Quantity[1, "Miles"]}, PlotLegends → Automatic],
  GeoSmoothHistogram[Auto Theft Locations +, PlotLegends → Automatic]}]
```



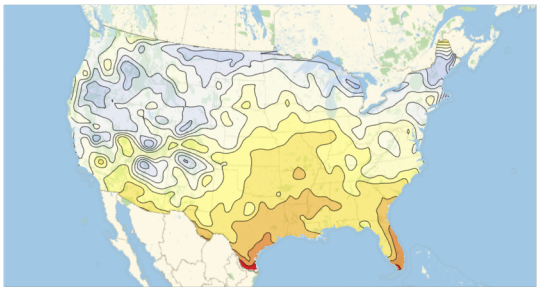


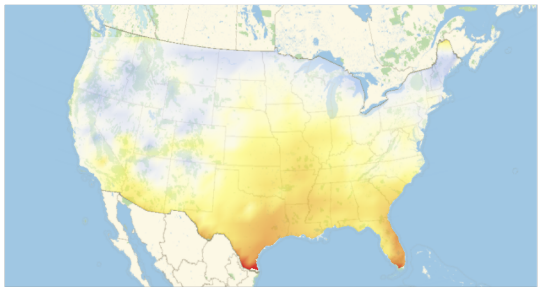
```
GraphicsRow[{GeoStreamPlot[Data, StreamMarkers → "Drop",
    StreamStyle → Blue, GeoBackground → "Coastlines"],
  GeoVectorPlot[Data, VectorMarkers → "Dart", VectorScale → Large, VectorStyle →
    Yellow, GeoProjection → "LambertAzimuthal", GeoBackground → "Satellite"]}]
```



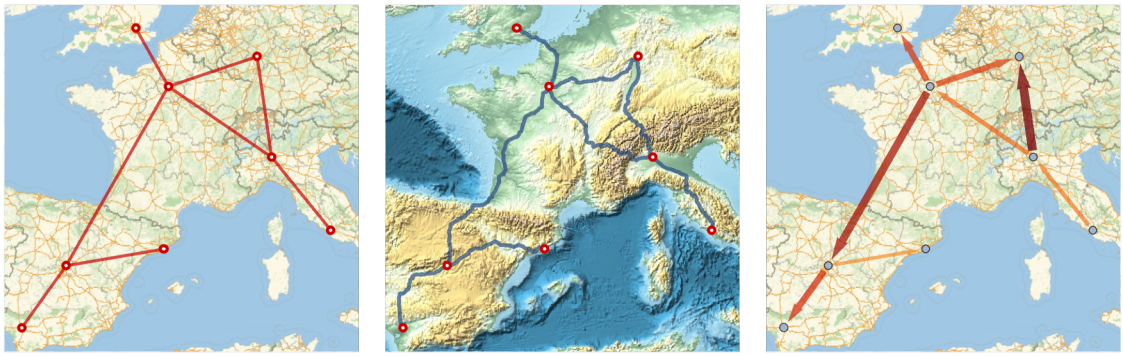


```
GraphicsRow[{GeoContourPlot[temperatures,
    ColorFunction → "TemperatureMap", RegionFunction → United States COUNTRY,
    Contours → 12, ContourShading → True, InterpolationOrder → 5],
  GeoDensityPlot[temperatures,
    ColorFunction → "TemperatureMap", RegionFunction → United States COUNTRY]}]
```

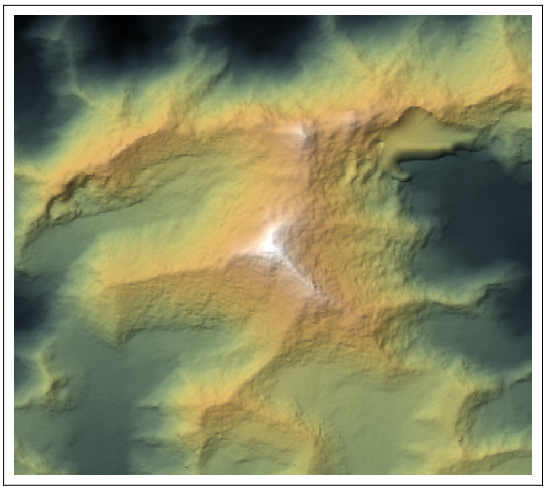




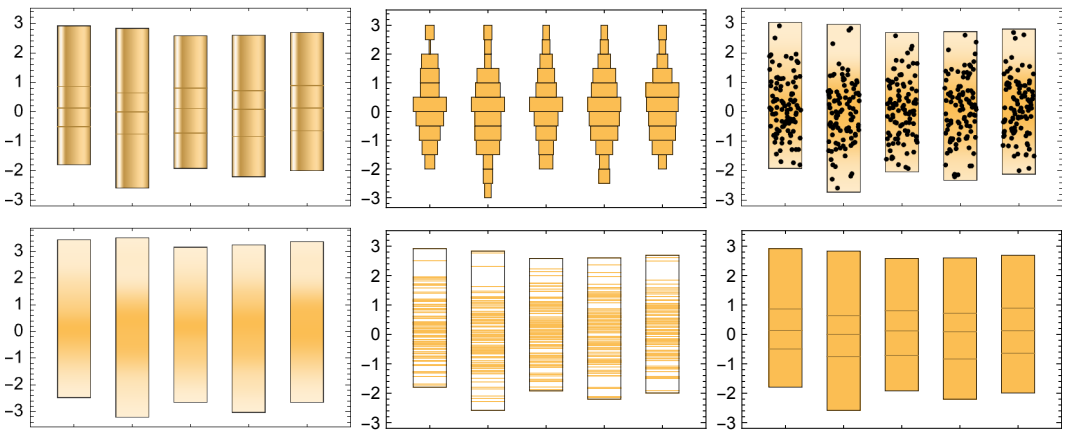
```
GraphicsRow[{GeoGraphPlot[
   network = {Rome CITY ⟷ Milan CITY, ⋯, Madrid CITY ⟷ Barcelona CITY}],
  GeoGraphPlot[network, GraphLayout → "Driving",
   EdgeStyle → Thick, GeoBackground → GeoStyling["ReliefMap"]],
  GeoGraphValuePlot[
   {{Rome CITY, Milan CITY, 2.2}, ⋯, {Madrid CITY, Barcelona CITY, 1.8}}]}]
```



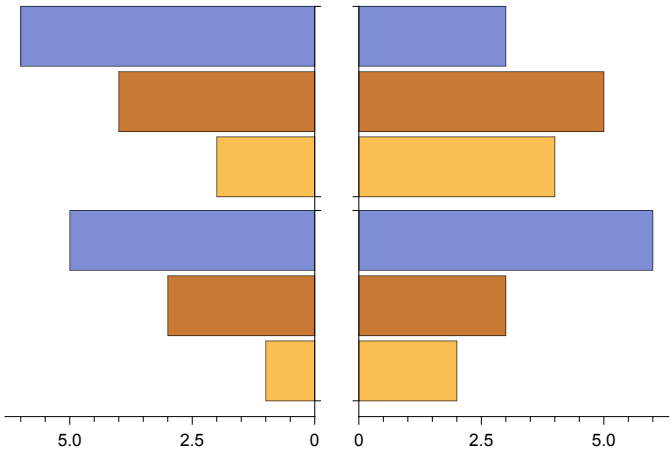
```
ReliefPlot[GeoElevationData[Mount Everest MOUNTAIN, GeoRange → 6 km],
 ColorFunction → "GreenBrownTerrain"]
```



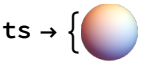
```
Multicolumn[
 Table[DistributionChart[Data, ChartElementFunction → f], {f, {"GlassQuantile",
    "Density", "HistogramDensity", "LineDensity", "PointDensity", "Quantile"}}], 3]
```



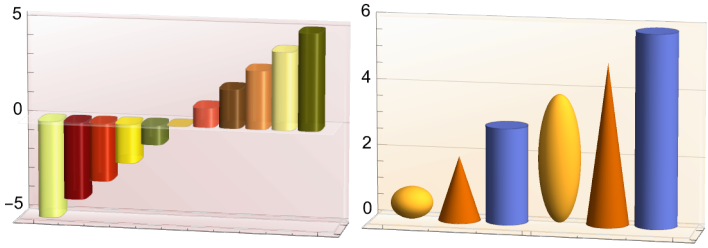
```
PairedBarChart[{{1, 3, 5}, {2, 4, 6}}, {{2, 3, 6}, {4, 5, 3}}]
```



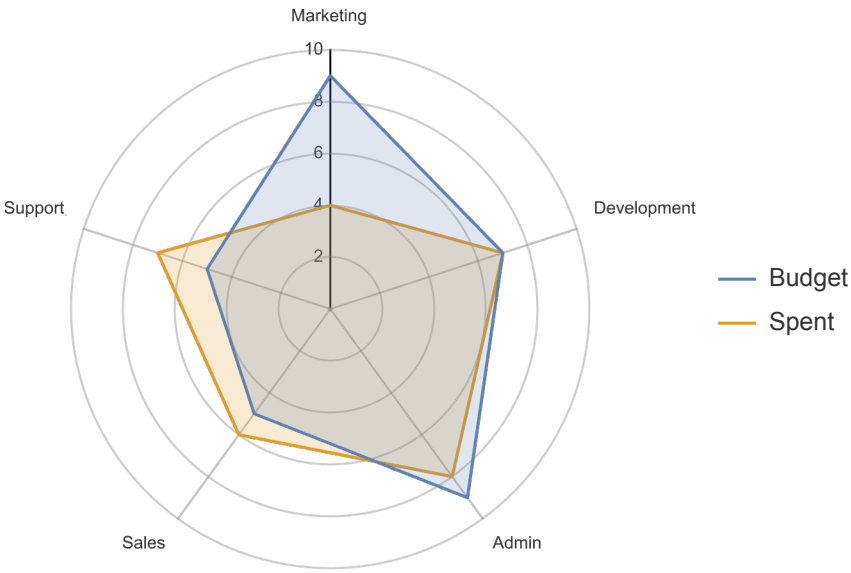

```
Multicolumn[
 {BarChart3D[Range[-5, 5], ChartStyle → 53, ChartElementFunction → "ProfileCube",
   ChartBaseStyle → Directive[EdgeForm[Gray], Opacity[0.8], Specularity[White, 30]]],
  BarChart3D[{{1, 2, 3}, {4, 5, 6}}, ChartElements → {
```



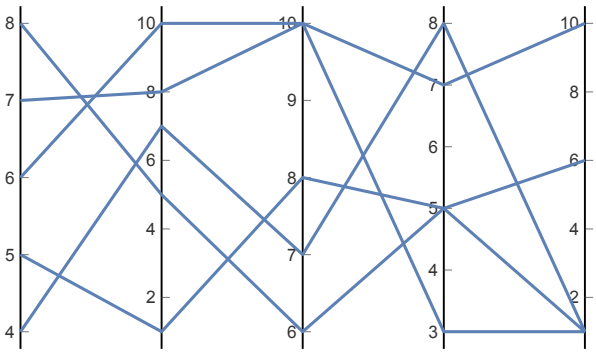
```
}]}, 2]
```



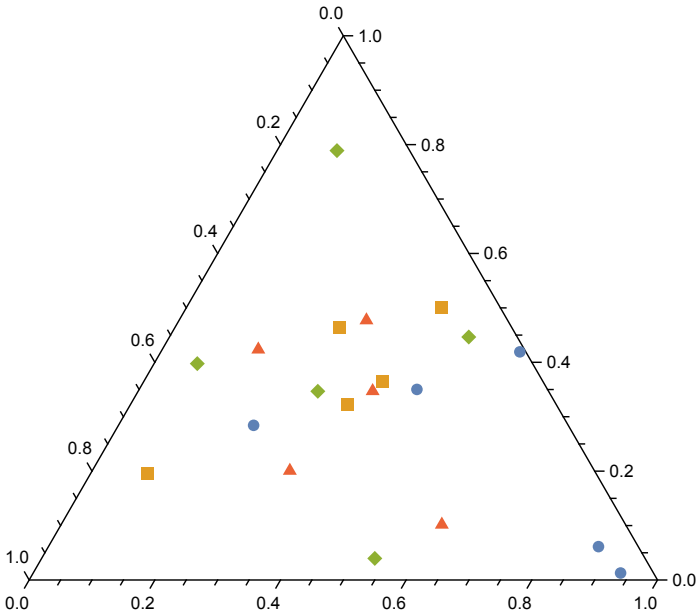

```
RadialAxisPlot[{Legended[<|"Marketing" → 9,
    "Development" → 7, "Admin" → 9, "Sales" → 5, "Support" → 5|>, "Budget"],
  Legended[<|"Marketing" → 4, "Development" → 7,
    "Admin" → 8, "Sales" → 6, "Support" → 7|>, "Spent"]}]
```



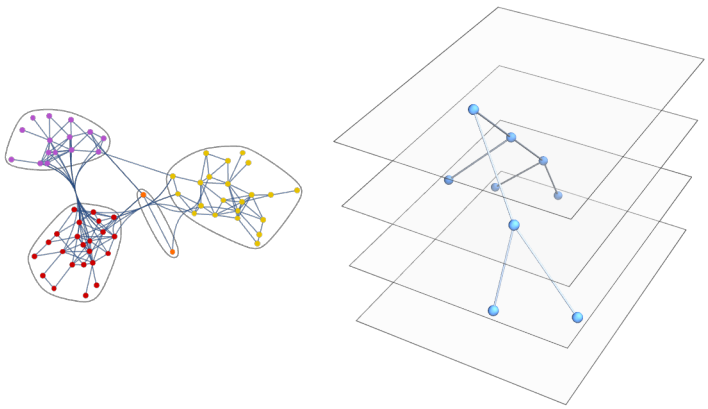
```
ParallelAxisPlot[RandomChoice[Range[10], {5 , 5}]]
```



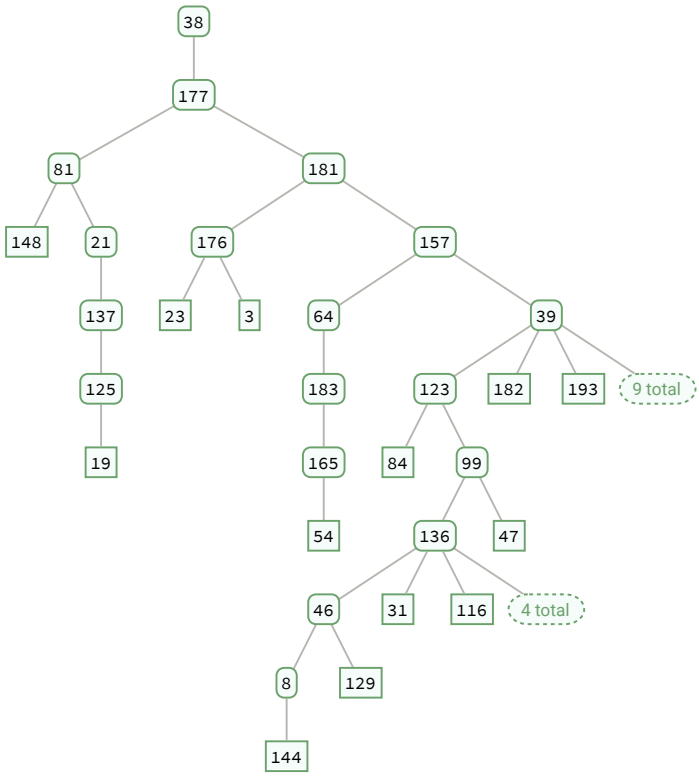

```
TernaryListPlot[RandomReal[1, {4, 5, 3}], PlotMarkers → Automatic]
```



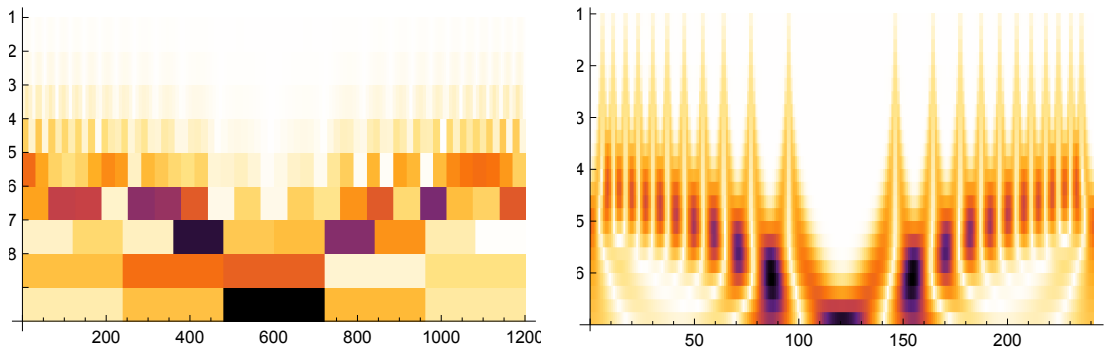

```
GraphicsRow[{CommunityGraphPlot[ ], LayeredGraphPlot3D[KaryTree[9]]}]
```



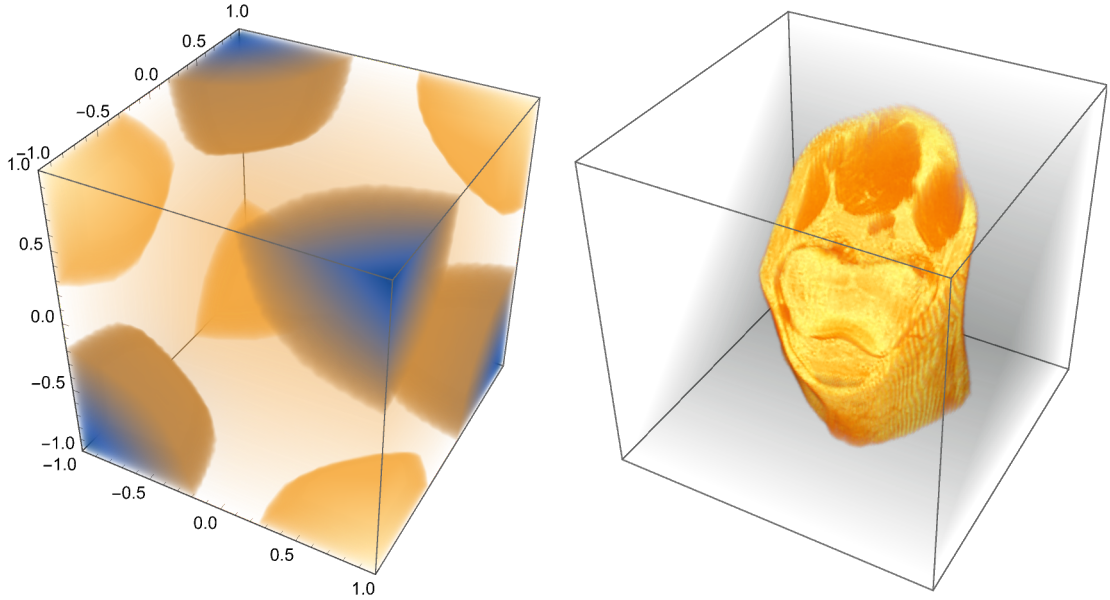
```
SeedRandom[9999]; RandomTree[200, MaxDisplayedChildren → All → 3]
```



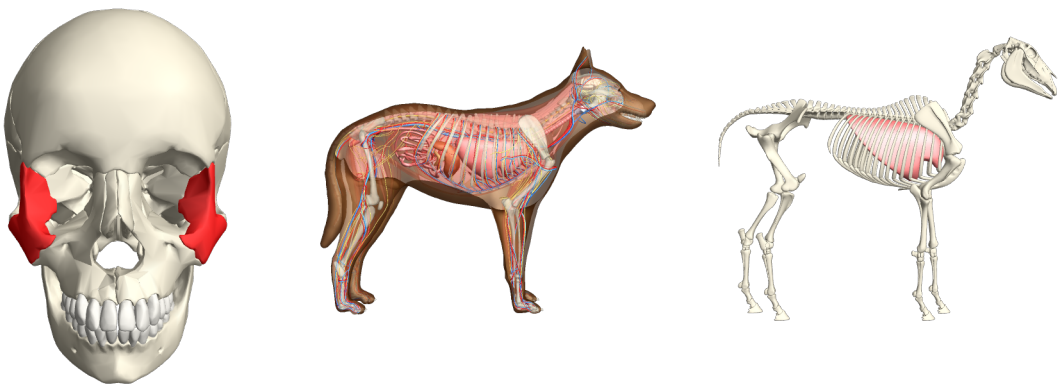

```
GraphicsRow[{WaveletScalogram[DiscreteWaveletTransform[
    Table[Sin[x^2], {x, -6, 6, 0.01}], Automatic, 8]], WaveletScalogram[
   ContinuousWaveletTransform[Table[Sign[Cos[x^2]], {x, -6, 6, 0.05}]]]}]
```



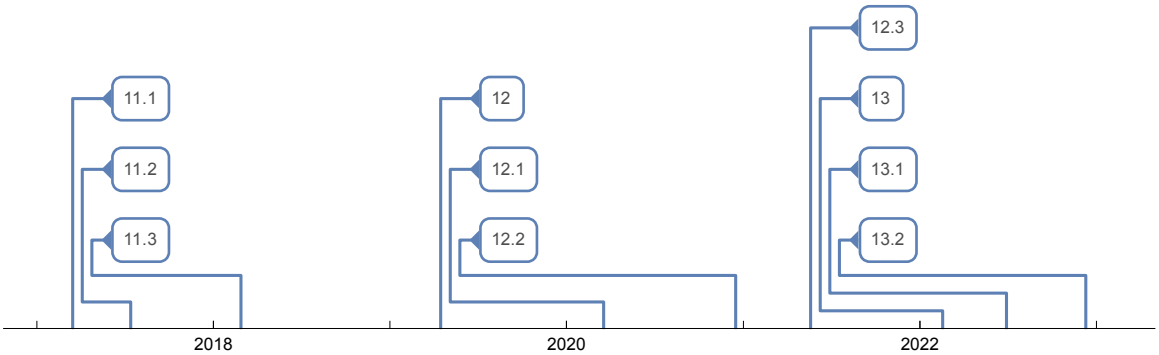
```
GraphicsRow[{DensityPlot3D[x y z, {x, -1, 1}, {y, -1, 1}, {z, -1, 1}]
, Show[ExampleData[{"TestImage3D", "MRknee"}], ClipPlanes → {{0, 1, -1, 0}}]}]
```



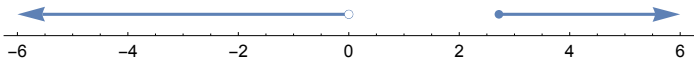

```
GraphicsRow[{AnatomyPlot3D[{ neurocranium ANATOMICAL STRUCTURE , sphenoid bone ANATOMICAL STRUCTURE ,
    nasal bone ANATOMICAL STRUCTURE , maxilla ANATOMICAL STRUCTURE , maxillary dentition ANATOMICAL STRUCTURE ,
    mandible ANATOMICAL STRUCTURE , mandibular dentition ANATOMICAL STRUCTURE , Red,
    zygomatic bone ANATOMICAL STRUCTURE }, PlotRange → skull ANATOMICAL STRUCTURE ],
  AnatomyPlot3D[{ alimentary system (dog) ANIMAL ANATOMICAL STRUCTURE ,
    respiratory system (dog) ANIMAL ANATOMICAL STRUCTURE ,
    cardiovascular system (dog) ANIMAL ANATOMICAL STRUCTURE ,
    nervous system (dog) ANIMAL ANATOMICAL STRUCTURE ,
    Opacity[.5], skeleton (dog) ANIMAL ANATOMICAL STRUCTURE ,
    ClipPlanes → Dynamic[{InfinitePlane[{{-78, -200, 0}, {-78, 300, 0}, {-78 + 50, 0, 500}}]}],
    Opacity[.1], muscular system (dog) ANIMAL ANATOMICAL STRUCTURE ,
    Opacity[.7], skin (dog) ANIMAL ANATOMICAL STRUCTURE }, ViewPoint → Left],
  AnatomyPlot3D[{ set of bones (horse) ANIMAL ANATOMICAL STRUCTURE ,
    lung (horse) ANIMAL ANATOMICAL STRUCTURE }, ViewPoint → Left]}]
```



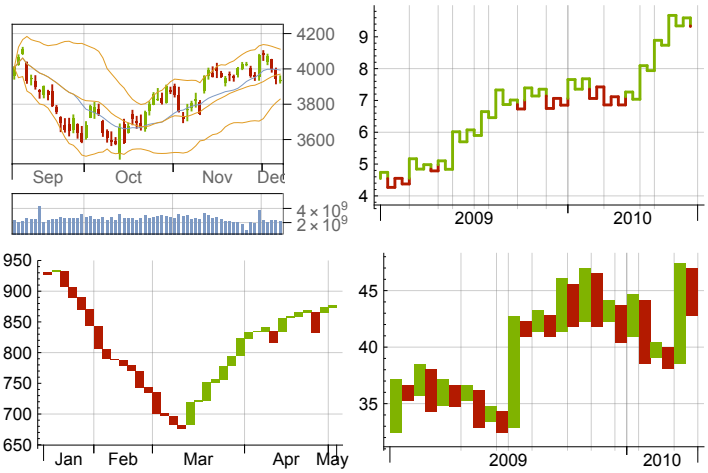
```
TimelinePlot[<|"11.1" → Day: Thu 16 Mar 2017,
  "11.2" → Day: Fri 14 Jul 2017, "11.3" → Day: Tue 27 Feb 2018, "12" → Day: Tue 16 Apr 2019,
  "12.1" → Day: Wed 18 Mar 2020, "12.2" → Wed 16 Dec 2020, "12.3" → Thu 20 May 2021,
  "13" → Thu 17 Feb 2022, "13.1" → Wed 29 Jun 2022, "13.2" → Sat 10 Dec 2022|>]
```




```
NumberLinePlot[FunctionRange[e^x/x, x, y], {y, -5, 5}]
```

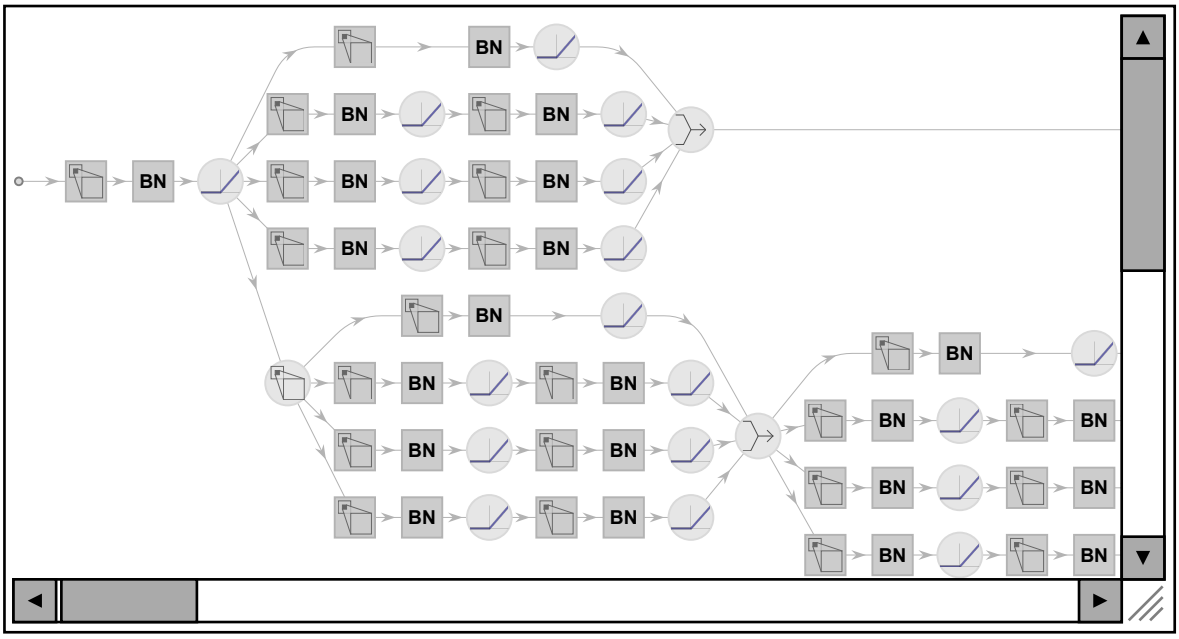



```
Grid[{{
    TradingChart["SP500", {"Volume", "SimpleMovingAverage", "BollingerBands"}],
    KagiChart[FinancialData["AAPL", "Close", {{2009, 5, 1}, {2010, 4, 30}, "Day"}]]},
  {LineBreakChart[{"^GSPC", {{2009, 1, 1}, {2009, 4, 31}}}],
    PointFigureChart[FinancialData["JPM", "Close", {{2009, 5, 1}, {2010, 4, 30}}]]}}]
```



```
WordCloud[EntityValue[CountryData[], {"Name", "Population"}], ♥]
```


```
Framed@Pane[Information[
   NetModel["Single-Image Depth Perception Net Trained on Depth in the Wild Data"],
   "FullSummaryGraphic"], {660, 350}, Scrollbars → True]
```



```
Framed@FeatureSpacePlot[{<image_1>, <image_2>, <image_3>, <image_4>,
  <image_5>, <image_6>, <image_7>, <image_8>, <image_9>}, LabelingSize → 60]
```

<image_1>
<image_2>
<image_3>
<image_4>
<image_5>
<image_6>
<image_7>
<image_8>
<image_9>

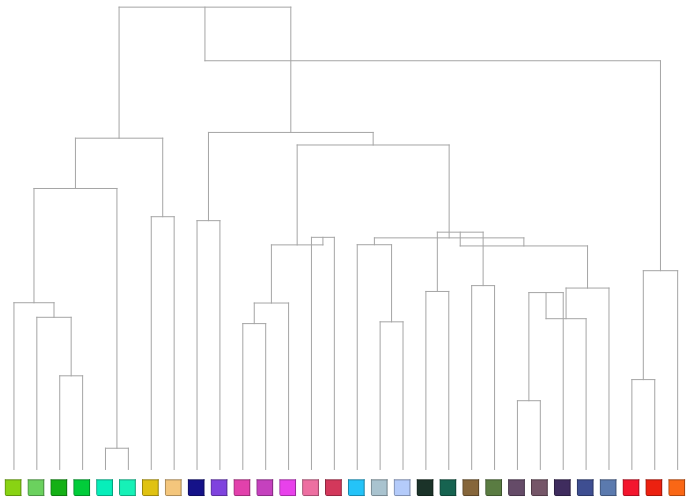
<image_10>

```
i = <image_11>;
HighlightImage[i, ImageBoundingBoxes[i]]
```

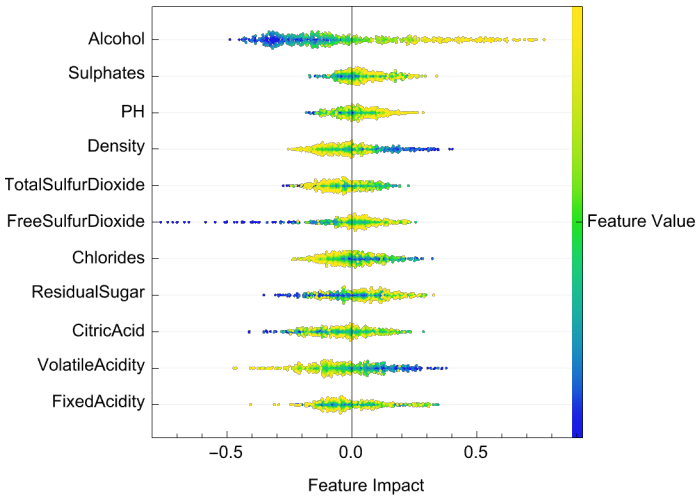
<image_11>

<image_12>

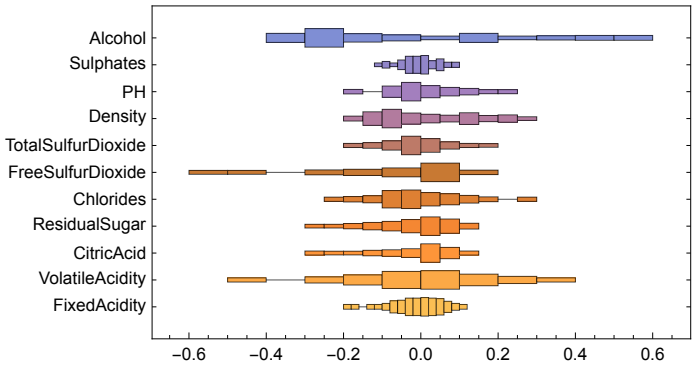
```
Dendrogram[RandomColor[30], ClusterDissimilarityFunction → "Centroid"]
```



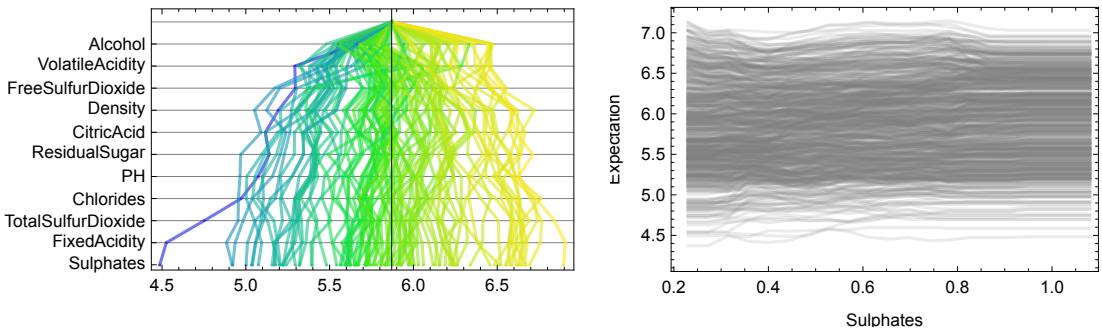
```
train = RandomSample[ResourceData["Sample Data: Wine Quality"], 4000];
test = Complement[ResourceData["Sample Data: Wine Quality"], train];
wineQuality = Predict[train → "WineQuality", PerformanceGoal → "Quality"];
PredictorMeasurements[wineQuality, test, "SHAPPlots"]
```



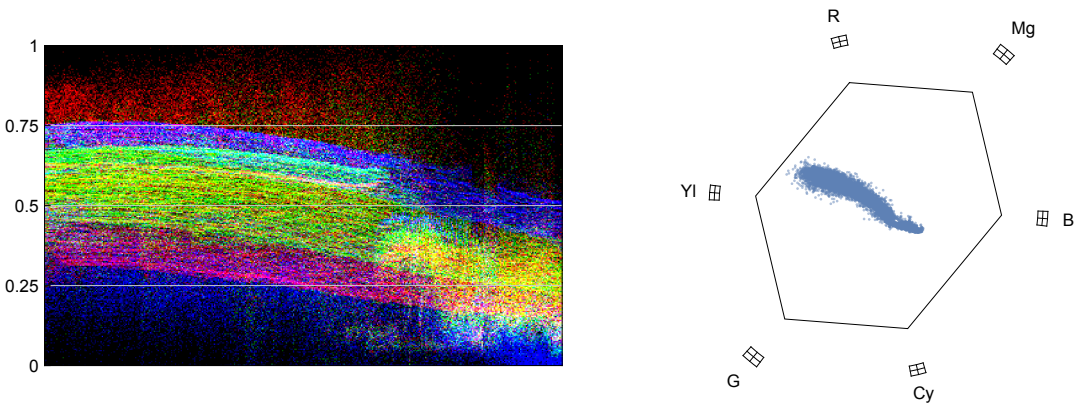

```
FeatureImpactPlot[wineQuality, test]
```



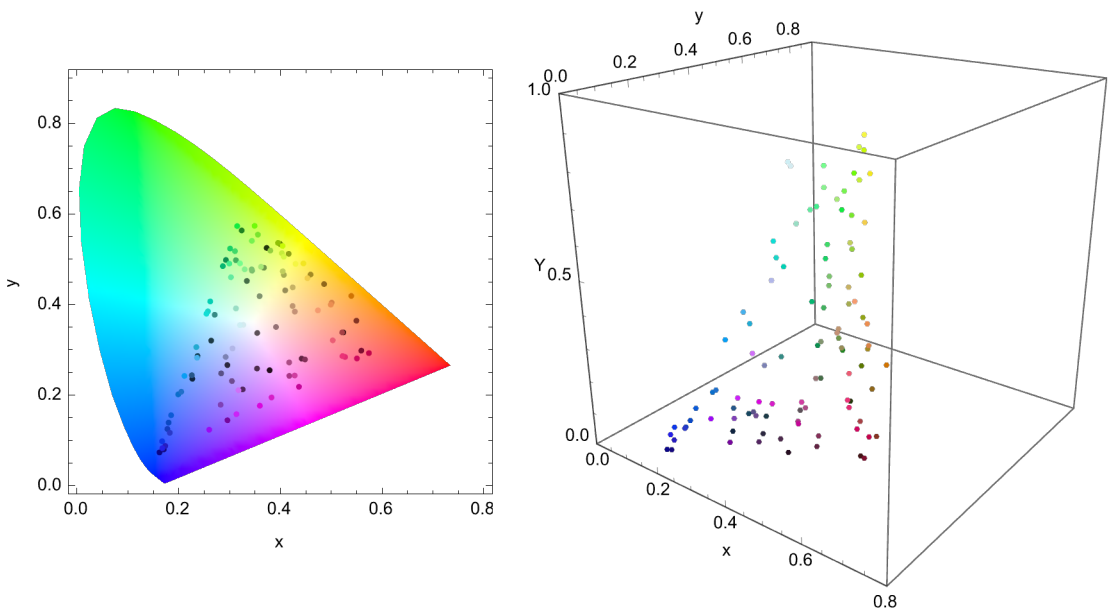
```
GraphicsRow[{CumulativeFeatureImpactPlot[wineQuality, test],
  PredictorMeasurements[wineQuality, test, "ICEPlots"]["Sulphates"]}]
```



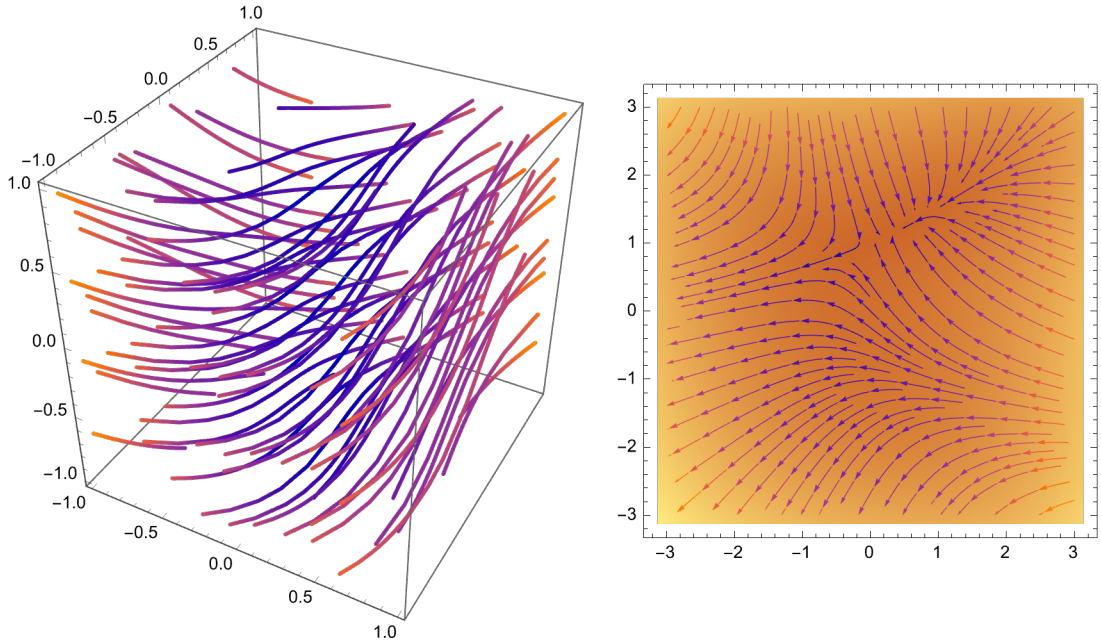

```
GraphicsGrid[{{ImageWaveformPlot[ ], ImageVectorscopePlot[ ]}},
 ImageSize → 600]
```



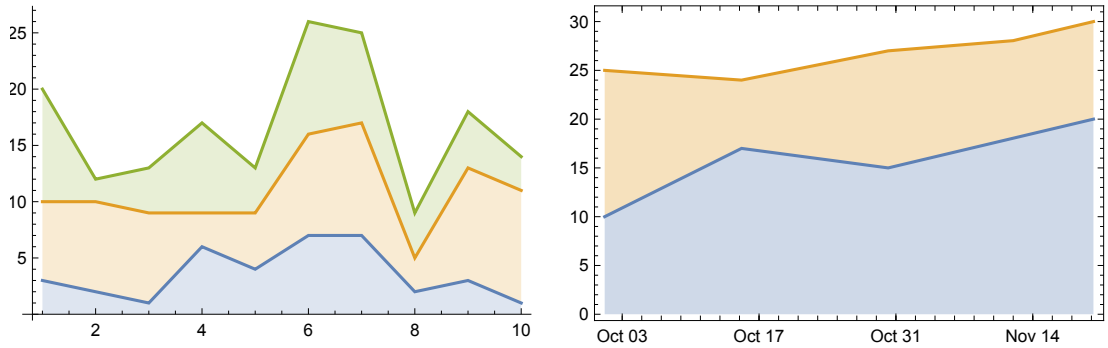

```
With[{cols = RandomColor[100]},
 GraphicsRow[{ChromaticityPlot[cols], ChromaticityPlot3D[cols]}]]
```



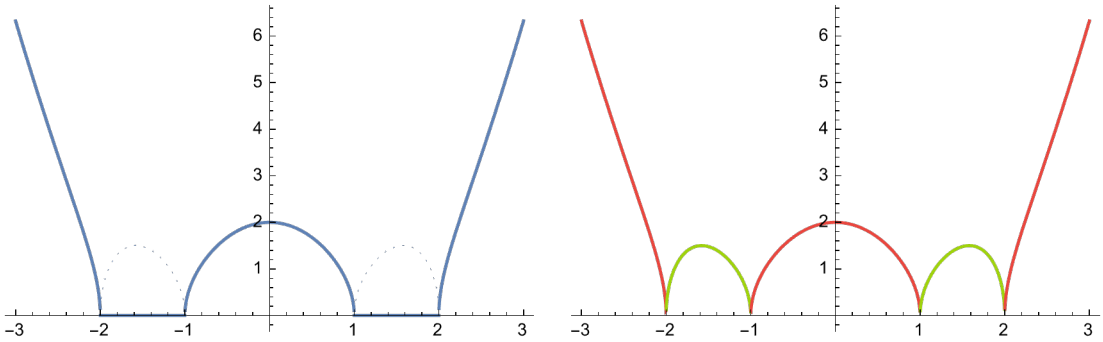
```
GraphicsRow[{StreamPlot3D[{y^2, 1, x}, {x, -1, 1}, {y, -1, 1}, {z, -1, 1}],
  StreamDensityPlot[{-1 - x^2 + y, 1 + x - y^2}, {x, -3, 3}, {y, -3, 3}]}]
```



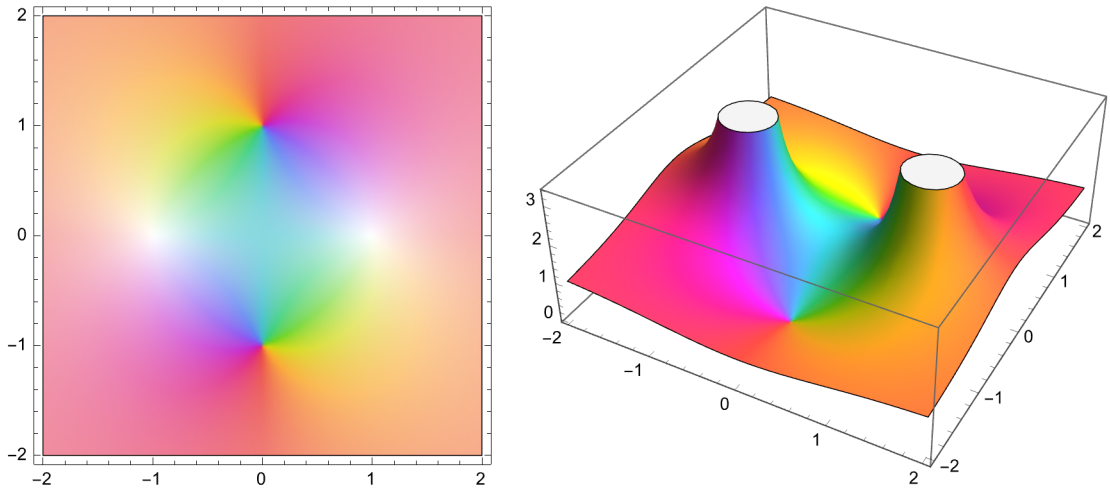

```
GraphicsRow[{StackedListPlot[{{3, 2, 1, 6, 4, 7, 7, 2, 3, 1},
    {7, 8, 8, 3, 5, 9, 10, 3, 10, 10}, {10, 2, 4, 8, 4, 10, 8, 4, 5, 3}}],
  data1 = {{Day: Sat 1 Oct 2016, 10}, {Day: Sat 15 Oct 2016, 17},
    {Day: Sun 30 Oct 2016, 15}, {Day: Sun 20 Nov 2016, 20}};
data2 = {{Day: Sat 1 Oct 2016, 15}, {Day: Sat 15 Oct 2016, 7},
    {Day: Sun 30 Oct 2016, 12}, {Day: Sat 12 Nov 2016, 10}, {Day: Sun 20 Nov 2016, 10}};
StackedDateListPlot[{data1, data2}]}]
```



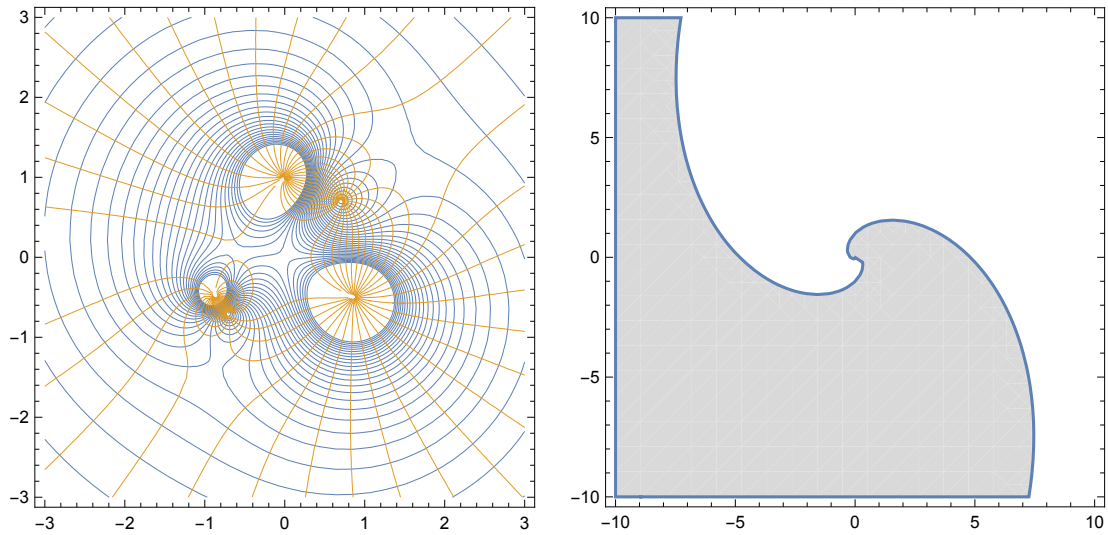
```
GraphicsRow[
 {ReImPlot[Sqrt[(x^2 - 1) (x^2 - 4)], {x, -3, 3}], AbsArgPlot[Sqrt[(x^2 - 1) (x^2 - 4)], {x, -3, 3}]}]
```



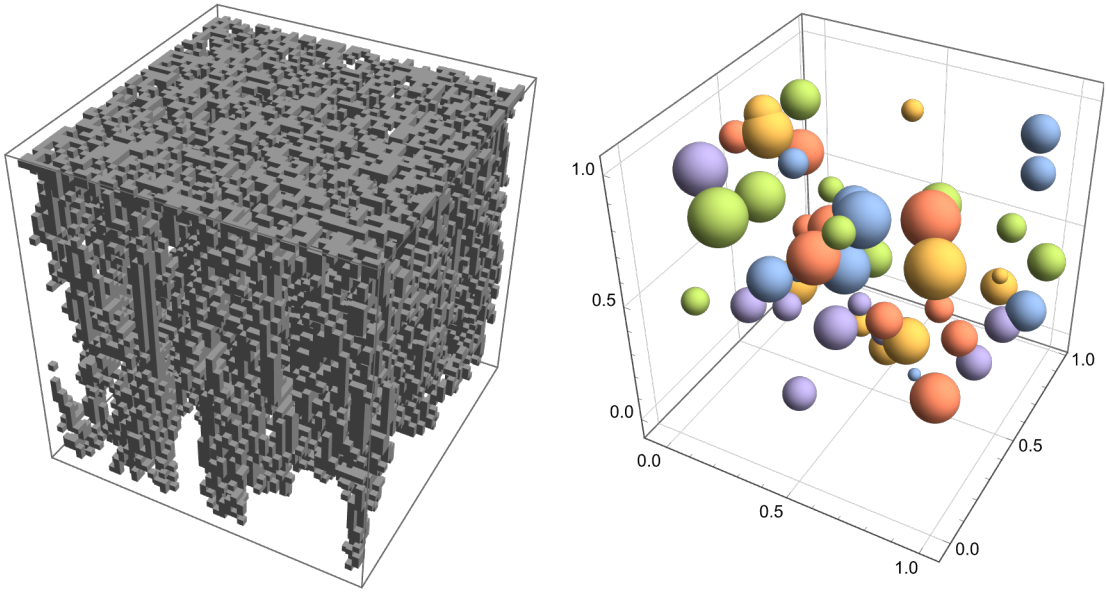
```
GraphicsRow[
 {ComplexPlot[(z^2 + 1)/(z^2 - 1), {z, -2 - 2 I, 2 + 2 I}], ComplexPlot3D[(z^2 + 1)/(z^2 - 1), {z, -2 - 2 I, 2 + 2 I}]}]
```



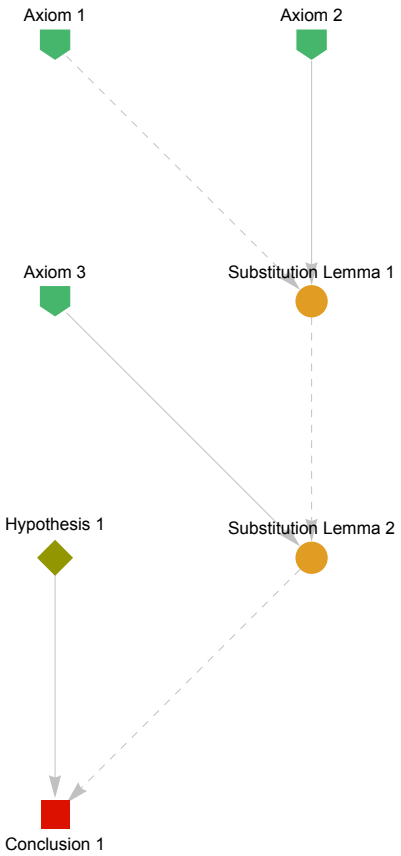
```
GraphicsRow[
 {ComplexContourPlot[AbsArg[(z^2 - I) / (z^3 + I)], {z, -3 - 3 I, 3 + 3 I}, Contours → 30],
  ComplexRegionPlot[Sqrt[z^(2 + 2 I)] == z^(1 + I), {z, 10}, PlotStyle → LightGray]}]
```



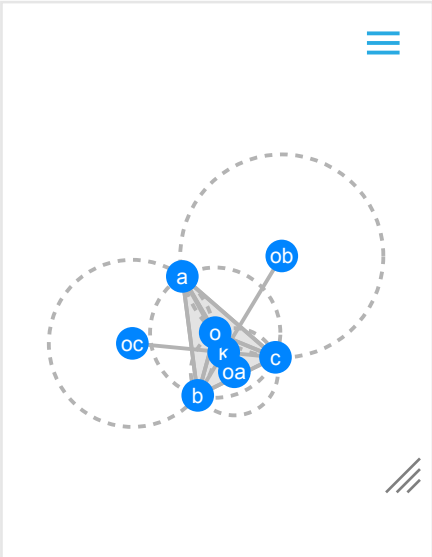
```
GraphicsRow[
 {ArrayPlot3D[CellularAutomaton["GameOfLife", RandomInteger[1, {50, 50}], 50]],
  BubbleChart3D[RandomReal[1, {5, 10, 4}]]}]
```



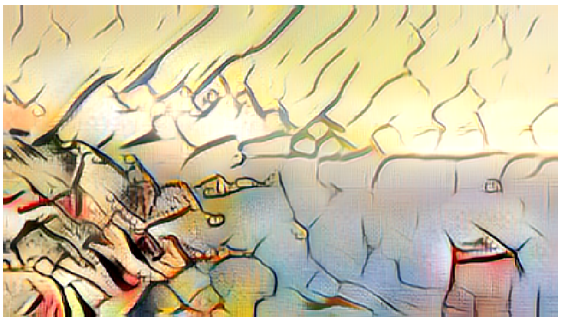

```
Row[{FindEquationalProof[a == d, {a == b, b == c, c == d}]["ProofGraph"],
```



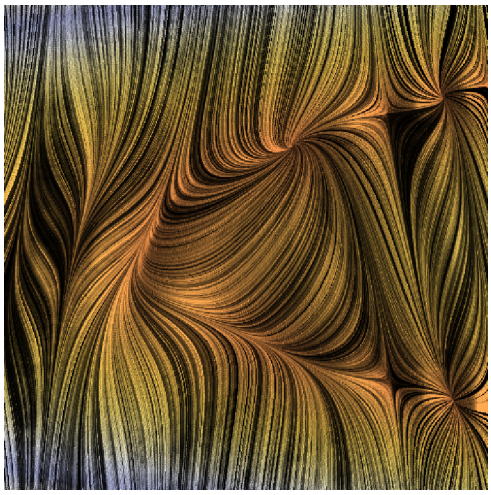
```
RandomInstance[
 GeometricScene[{a, b, c, o, oa, ob, oc, k}, {o == TriangleCenter[{a, b, c}, "Circumcenter"],
   oa == TriangleCenter[{o, b, c}, "Circumcenter"], ob ==
    TriangleCenter[{a, o, c}, "Circumcenter"], oc == TriangleCenter[{a, b, o}, "Circumcenter"],
   Line[{a, k, oa}], Line[{b, k, ob}], Line[{c, oc}]}], RandomSeeding → 6]
```



ImageRestyle[ , ]

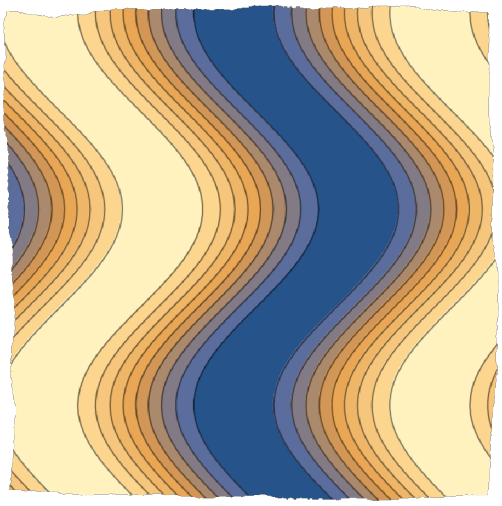


```
LineIntegralConvolutionPlot[{{Cos[x^2 + y], 1 + x - y^2}, {"noise", 500, 500}},
 {x, -3, 3}, {y, -3, 3}, ColorFunction → "BeachColors",
 LightingAngle → 0, LineIntegralConvolutionScale → 3, Frame → False]
```

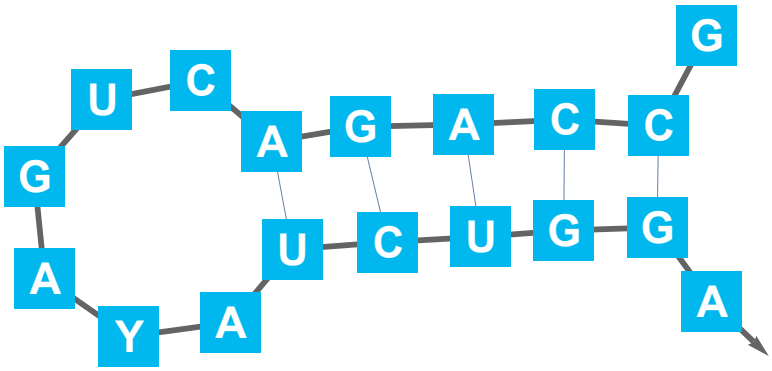


```
GraphicsRow[
 {BarcodeImage["123456789999", "UPC"], BarcodeImage["http://www.wolfram.com", "QR"],
  BarcodeImage["1234", "PDF417"], BarcodeImage["1234", "Aztec"],
  BarcodeImage["1234567890", "ITF"], BarcodeImage["1234", "DataMatrix"]}]
```

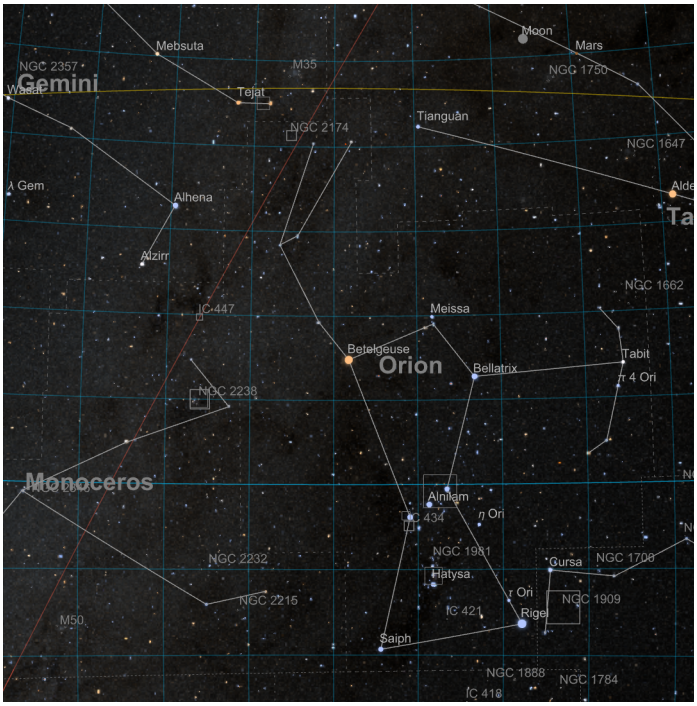
```
ImageEffect[ContourPlot[Sin[x + Sin[y]], {x, 0, 8}, {y, 0, 8}, Frame → None], "TornFrame"]
```



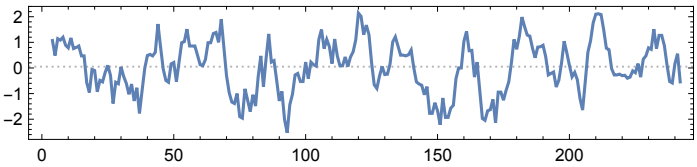
```
BioSequencePlot[BioSequence["RNA", "GCCAGACUGAYAUCUGGA",
  {Bond[{2, 17}], Bond[{3, 16}], Bond[{4, 15}], Bond[{5, 14}], Bond[{6, 13}]}]]
```



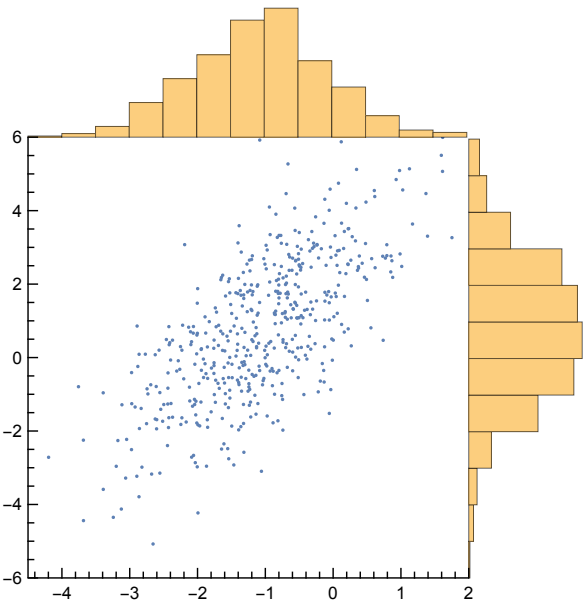
```
AstroGraphics[ Betelgeuse STAR , AstroRange → 20° , AstroReferenceFrame → "Equatorial",
 AstroBackground → "GalacticSky", AstroGridLines → 5° ]
```



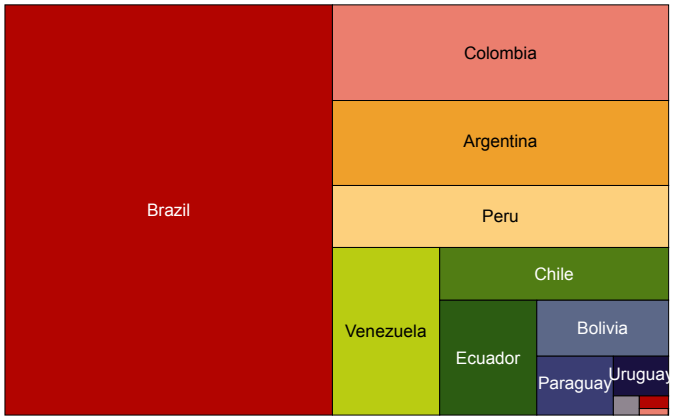

```
ResourceFunction["PeptideHydropathyPlot"][
 "CGVPAIQPVLSGLSRIVNGEEAVPGSWPWQVSLQDKTGFHFCGGSLINENWVVTAAHCGVTTSDVVVAGEFDQGSSSEKIQK⁚
    LKIAKVFKNSKYNSLTINNDITLLKLSTAASFSQTVSAVCLPSASDDFAAGTTCVTTGWGLTRYTNANTPDRLQQASLP⁚
    LLSNTNCKKYWGTKIKDAMICAGASGVSSCMGDSGGPLVCKKNGAWTLVGIVSWGSSTCSTSTPGVYARVTALVNWVQQ⁚
    TLAAN"]
```



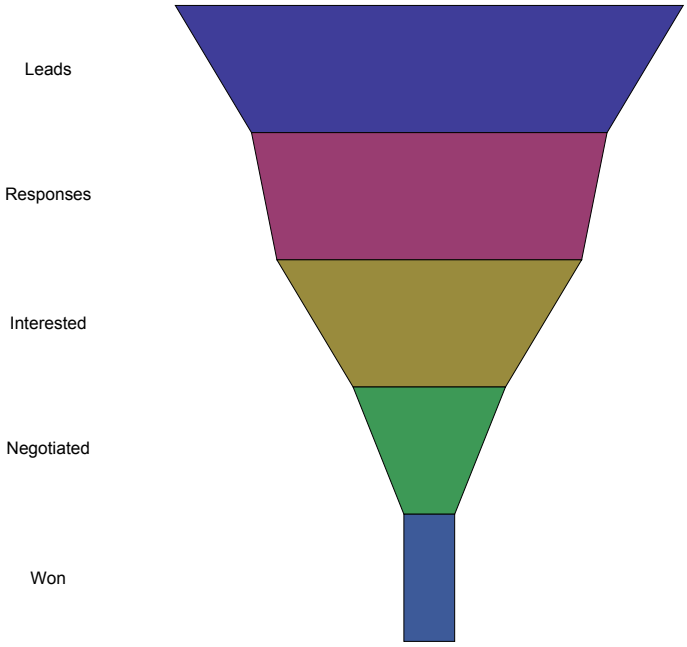

```
data = RandomVariate[BinormalDistribution[{-1, 1}, {1, 2}, 0.7], 500];
ResourceFunction["MarginalPlot"][data]
```



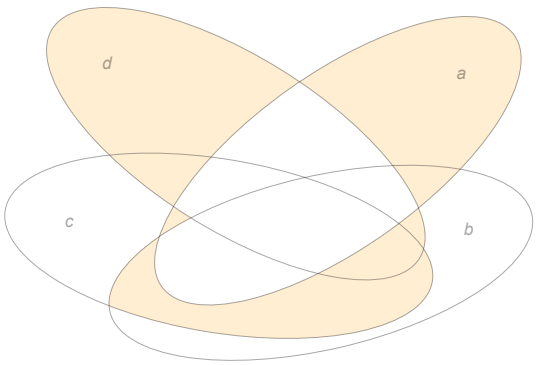
```
ResourceFunction["RectangleAreaChart"][
 If[#["Population"] < Quantity[1 000 000, "People"], None, #["Name"]] →
    #["Population"] & /@
  EntityList[EntityClass["Country", "SouthAmerica"]], "ChartColors" → ColorData[10]]
```



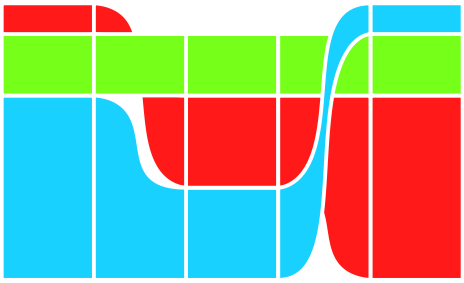

```
ResourceFunction["FunnelChart"][<|"Leads" → 10, "Responses" → 7,
  "Interested" → 6, "Negotiated" → 3, "Won" → 1|>, InterpolationOrder → 1]
```



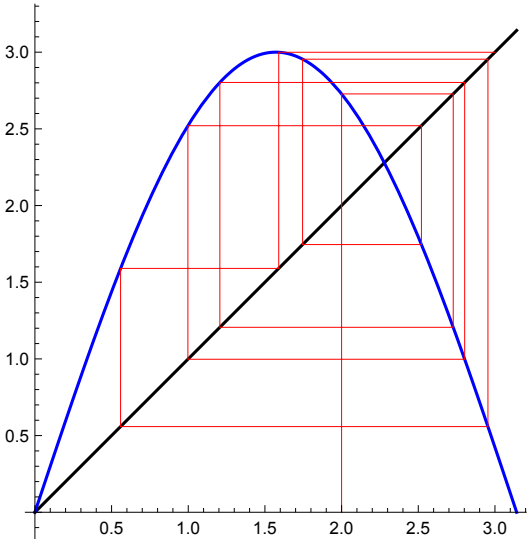

```
ResourceFunction["VennDiagram"][a ⊻ (b ∧ c ∨ d)]
```



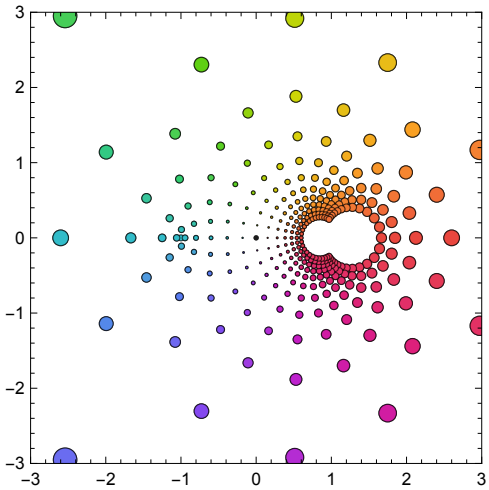
```
gs = Graph[{1 ⟷ 2, 2 ⟷ 3, 3 ⟷ 1}, VertexWeight → #] & /@ {{1, 2, 6}, {3, 2, 3}, {6, 2, 1}};
ResourceFunction["AlluvialChart"][gs, ImageSize → 250]
```



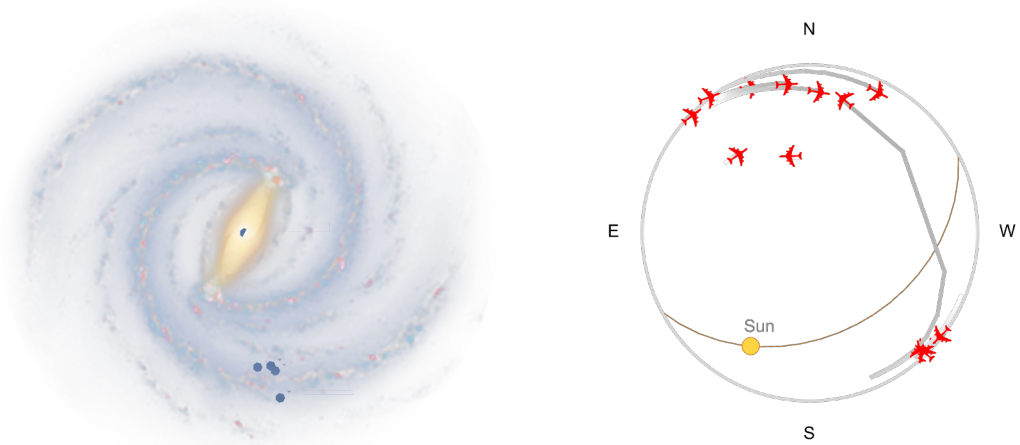
```
ResourceFunction["CobwebPlot"][x ↦ 3 Sin[x], 2, 10, {0, Pi}]
```



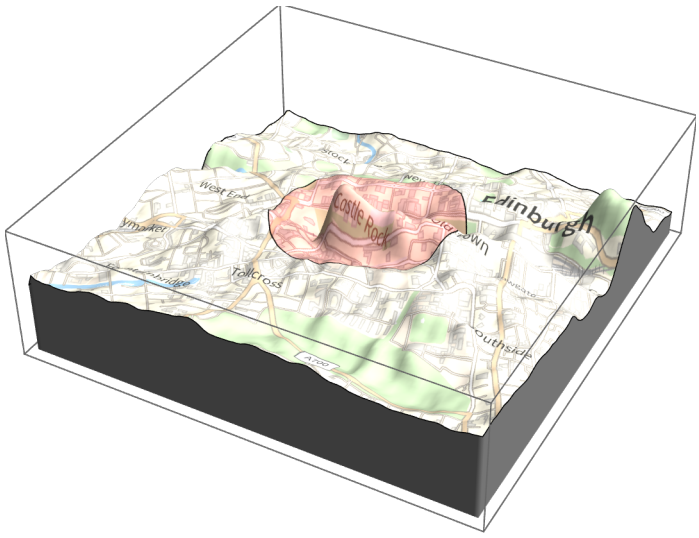

```
ResourceFunction["ComplexBubblePlot"][(z^2 + 1) / (z^2 - 1), {z, -2 - 2 I, 2 + 2 I}]
```



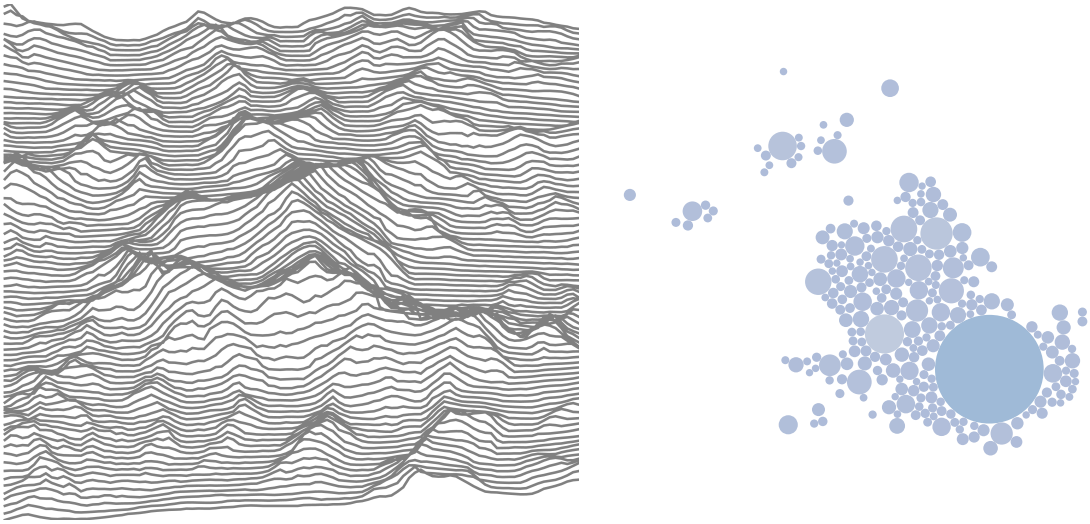
```
GraphicsRow[{ResourceFunction["MilkyWayPlot3D"][
    {Entity["Star", "Deneb"], Entity["AstronomicalRadioSource", "SagittariusAStar"],
     Entity["Nebula", "M42"], Entity["Pulsar", "PSRJ0534Plus2200"]}],
  ResourceFunction["FlightsOverhead"][New York City CITY, "SkyMap"]}]
```



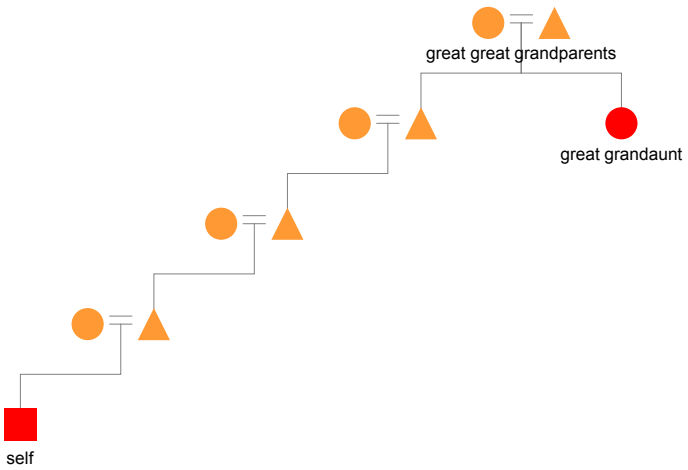

```
ResourceFunction["GeoElevationGraphics3D"][
 {EdgeForm[Black], Red, GeoDisk[Entity["City",
    {"Edinburgh", "Edinburgh", "UnitedKingdom"}], Quantity[500, "Meters"]]},
 GeoBackground → "VectorClassic", GeoRange → Quantity[0.75, "Miles"]]
```



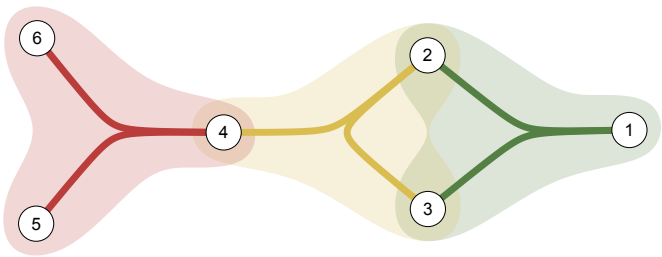
```
GraphicsRow[{ResourceFunction["RidgeLineMap"][
    GeoElevationData[Entity["Mountain", "MountEverest"],
     GeoRange → Quantity[10, "Kilometers"]], "HeightMultiplier" → 0.5],
  ResourceFunction["DorlingCartogram"][
    EntityValue[EntityClass["City", {"Country" → Entity["Country", "UnitedKingdom"],
        "Population" → TakeLargest[250]}], "Entities"] →
     "Population", ColorFunction → "Aquamarine"]}]
```



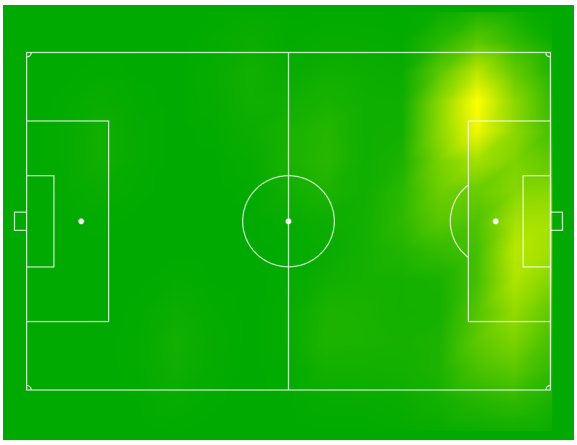
```
ResourceFunction["GenealogyTreePlot"][{"Aunt", "Great" → 2}]
```



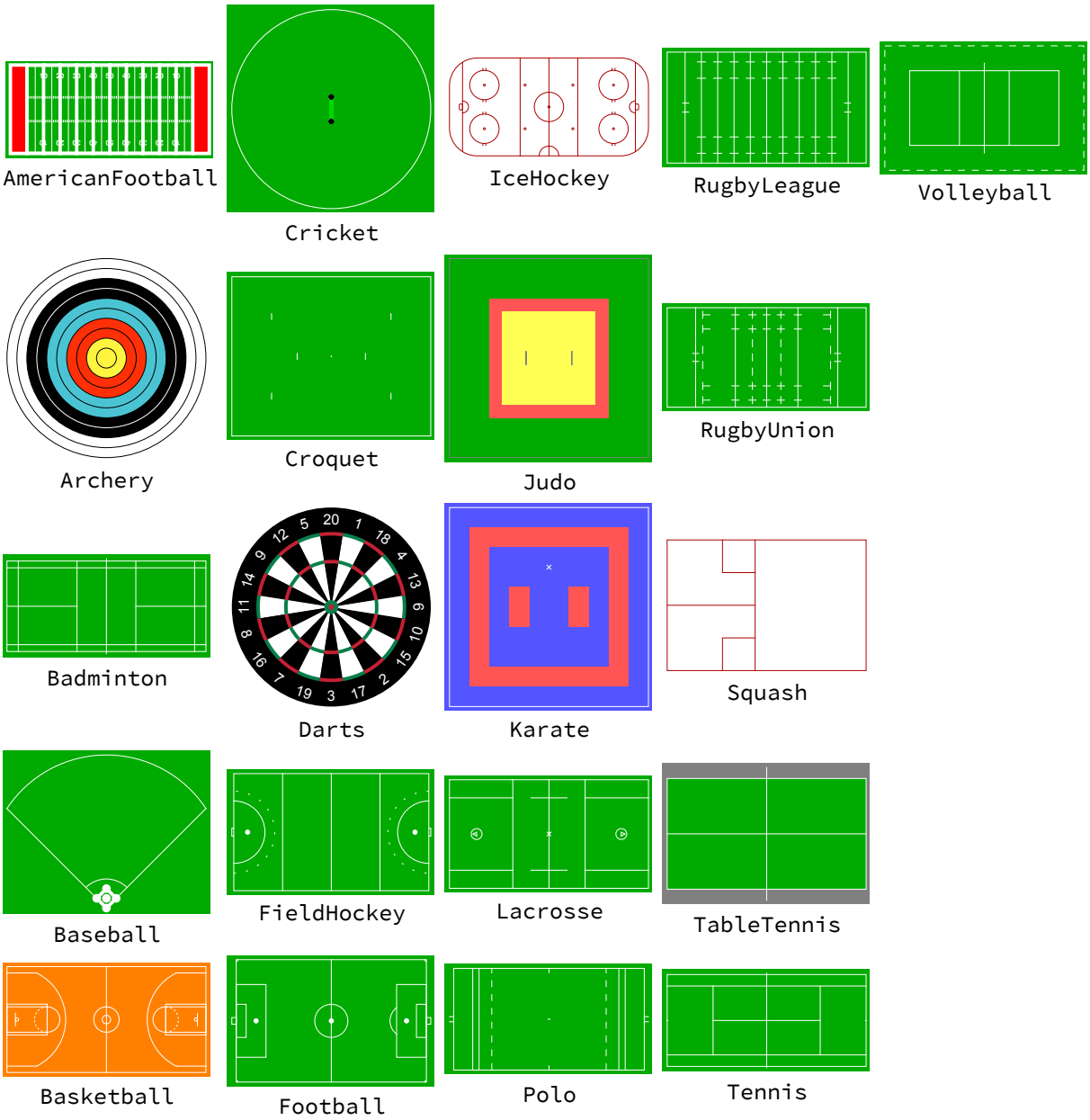

```
ResourceFunction["HypergraphPlot"][{{1, 2, 3}, {2, 3, 5}, {5, 7, 8}}]
```



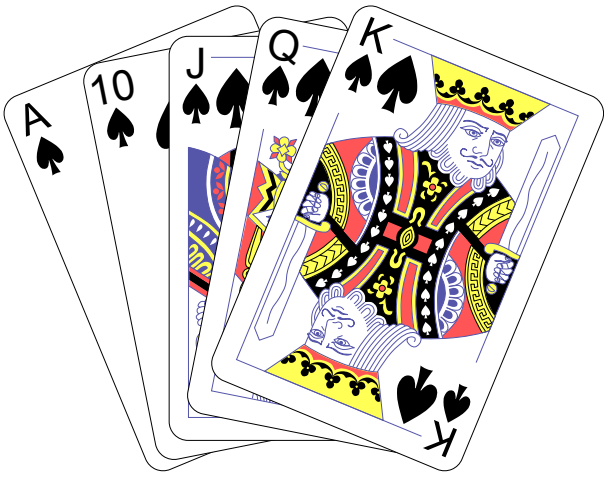
```
ResourceFunction["SportsFieldGraphics"]["Football",
 SmoothDensityHistogram[{…} +, ColorFunction → (Blend[{Darker@Green, Yellow}, #] &),
  PlotRange → {{-53, 53}, {-42, 42}}]]
```

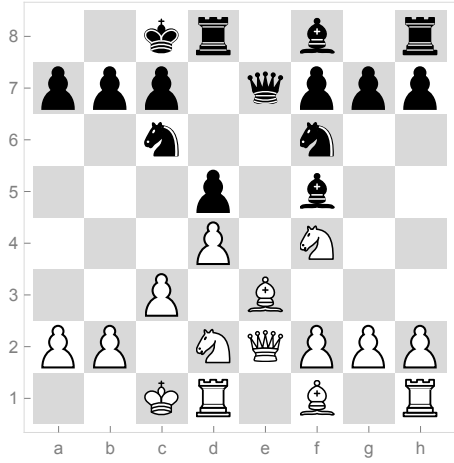


```
Multicolumn[Labeled[ResourceFunction["SportsFieldGraphics"][#, ImageSize → 120], #] & /@
  ResourceFunction["SportsFieldGraphics"][]]
```



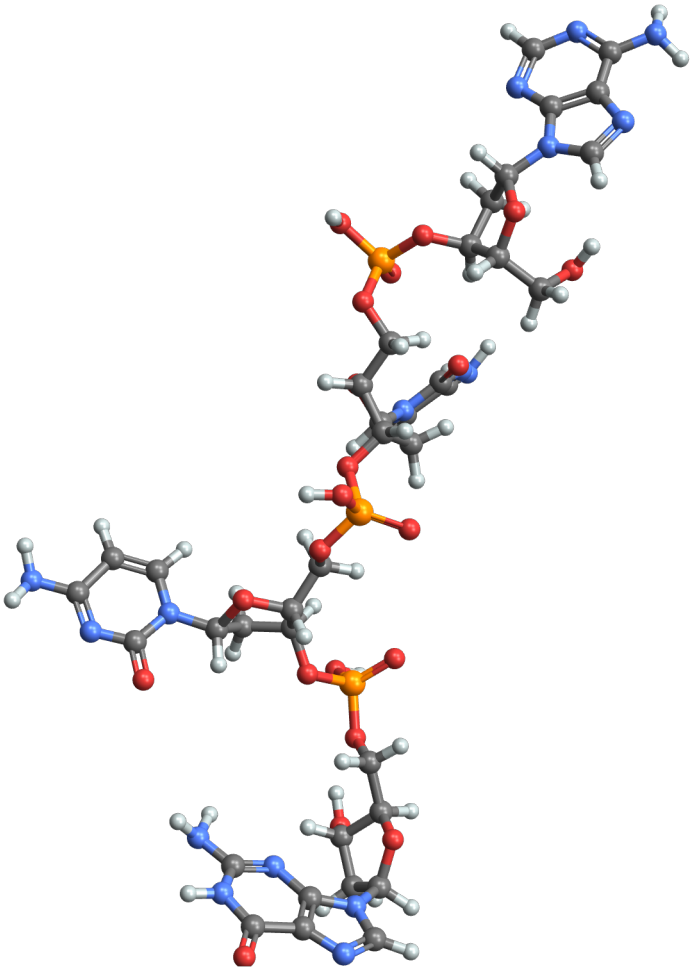
```
GraphicsRow[{ResourceFunction["PlayingCardGraphic"][
    {1, 10, 11, 12, 13}, ImagePadding → {{100, 100}, {30, 30}}],
  ImportString[
    "2kr1b1r/ppp1qppp/2n2n2/3p1b2/3P1N2/2P1B3/PP1NQPPP/2KR1B1R b - - 3 11",
    "FEN"][[1]]}]
```



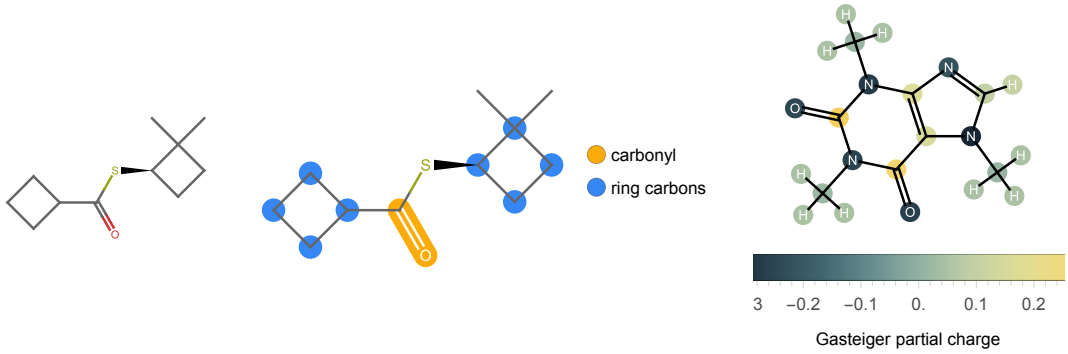


The following graphics are available in Maple only through the purchase of an additional toolbox.

```
MoleculePlot3D[Molecule[BioSequence["ACCG"]]]
```



```
GraphicsRow[{MoleculePlot[Molecule["O=C(C1CCC1)S[C@@H]1CCC1(C)C"]],
  MoleculePlot["O=C(C1CCC1)S[C@@H]1CCC1(C)C", <|"carbonyl" → Bond[{"C", "O"}, "Double"],
    "ring carbons" → Atom["C", "RingAtomQ" → True]|>],
  ResourceFunction["MoleculeValuePlot"][
   Molecule["caffeine"], "GasteigerPartialCharge"]}]
```



- Images were generated using Mathematica 13.1 and Maple 2022.
- Maple images have been copied using a screen capture tool to preserve pixel-level screen rendering. Printing this document will not represent the resolution that printing from the original application would achieve.
- Except where stated, all comparisons use default options. Both systems allow manual control over plot details, and in some cases, with sufficient work, a user may overcome some of the Maple deficiencies described in this comparison.
- Some plots have been manually rotated so they can be compared from similar viewpoints.