БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ
И ИНФОРМАТИКИ
Кафедра компьютерных технологий и систем

В. Б. Таранчук

WOLFRAM *MATHEMATICA*. ПРОГРАММИРОВАНИЕ ИНТЕРАКТИВНОЙ 3D ГРАФИКИ

Учебные материалы
для студентов факультета
прикладной математики и информатики

МИНСК
2024

УДК 519.67(075.8)
ББК 22.19я73
Т19

Рекомендовано советом
факультета прикладной математики и информатики БГУ
29 июня 2023 г., протокол № 12

Р е ц е н з е н т
доктор физико-математических наук *Н. Н. Гринчик*

Таранчук, В. Б.

Т19 Wolfram *Mathematica*. Программирование интерактивной 3D графики : учеб. материалы для студентов фак. прикладной математики и информатики / В. Б. Таранчук. – Минск : БГУ, 2024. – 52 с.

Изложены основные функции, опции и директивы построения, оформления, вывода графиков 3D в системе *Mathematica*. Включены примеры, которые поясняют возможности формирования, манипулирования, визуализации различными способами функций, описывающих поверхности, и примеры подготовки иллюстраций с использованием разных математических описаний в разных системах координат.

Предназначено для студентов факультета прикладной математики и информатики.

УДК 519.67(075.8)
ББК 22.19я73

ПРЕДИСЛОВИЕ

В учебных материалах изложены теоретические и методические вопросы, примеры решения возникающих в практике вычислительных экспериментов, интеллектуального анализа данных задач 3D визуализации функций с использованием инструментов системы компьютерной алгебры Wolfram *Mathematica* Plot3D, ContourPlot3D, RegionPlot3D, ParametricPlot3D, SliceContourPlot3D, RevolutionPlot3D. Изложенные в учебном пособии темы «Основы, примеры оформления трехмерных графиков», «Визуализации математических поверхностей» изучаются в дисциплинах специализации «Когнитивная визуализация» (учебная дисциплина для специальности 1-31 03 03 «Прикладная математика», направление специальности 1-31 03 03-01 «Прикладная математика, научно-производственная деятельность»), «Компьютерный анализ и визуализация» (учебная дисциплина для специальности 1-31 03 07 «Прикладная информатика», направление специальности 1-31 03 07-01 «Прикладная информатика, программное обеспечение компьютерных систем»), «Технологии интерактивной визуализации» (учебная дисциплина для специальности 1-31 03 04 «Информатика»), «Интерактивные вычисления и визуализация» (учебная дисциплина для специальности 1-31 03 03 «Прикладная математика», направление специальности 1-31 03 03-01 «Прикладная математика, научно-производственная деятельность»).

В материалах акценты сделаны на:

- развитие навыков функционального программирования средствами системы Wolfram *Mathematica*;
- освоение обучаемыми инструментов создания в *Mathematica* интерактивных программных модулей с возможностями точных символьных вычислений, интерактивного графического представления результатов математических преобразований и расчётов.

Советы и критические замечания по изданию просьба направлять на taranchuk@bsu.by.

ОСНОВЫ, ПРИМЕРЫ ОФОРМЛЕНИЯ ТРЕХМЕРНЫХ ГРАФИКОВ

О графических объектах 3D системы *Mathematica*

Подробное изложение обозначенной выше темы дано в [1]. Все отмеченное там позиции актуальны, более того, отдельные получили существенное развитие, что будет отмечено ниже. Графика системы *Mathematica* всегда считалась для многих пакетов и систем эталоном и во многом способствовала ее высокой репутации мирового лидера среди подобных программных приложений. Графические возможности достигаются, как обилием встроенных функций, так и средствами их модификации с помощью директив, опций, примитивов, языка программирования Wolfram Language. *Mathematica* позволяет строить любые виды математических графиков, причем обычного пользователя в большинстве случаев удовлетворяют графики, параметры которых не надо прописывать в блокноте, а задаются они по умолчанию. В системе поддерживается работа практически со всеми форматами графики, их импорт и экспорт.

Концептуально графики в *Mathematica* являются объектами, которые создаются (возвращаются) соответствующими графическими функциями. Они охватывают построение всех типов математических графиков, иллюстраций деловой графики – гистограмм, двумерных и трехмерных секторных и столбчатых, пузырьковых диаграмм, специфических графиков для таких областей, как финансы и статистика, теория графов, управляющие системы. Достигается многообразие за счет применения функций, опций и директив. Графическим объектам можно присваивать имена, а затем оперировать как любыми объектами.

Программирование графики в *Mathematica* относится к функциональному типу. Почти каждый ввод в системе является функцией (командой), каждая имеет определённый набор атрибутов и может иметь набор опций, директив. Атрибуты важны для эффективной работы системы и могут использоваться при программировании. Пользователи *Mathematica* могут применять опции для управления работой отдельных функций. Опции влияют на результат выполнения функции, различные варианты опций могут дать совершенно разные по виду результаты.

Mathematica содержит множество средств для визуализации функций и данных. Система автоматизирует процесс подбора и указания разных деталей построения иллюстраций. Применение опций, число которых очень большое, позволяет оформлять графические образы в любом масштабе, виде, дизайне. У каждой графической функции опция всегда имеет значение по умолчанию, которое определяет результат применения функции, если не указано другое. Несмотря на то, что используемые по умолчанию

параметры визуализации тщательно подобраны, чтобы наилучшим образом (с точки зрения разработчиков) подходить для большинства типичных случаев, *Mathematica* обеспечивает пользователей возможностями индивидуальной настройки.

Более того, как уже отмечалось ранее, пользователь может в Out-секции блокнота *Mathematica* изменить размер графического изображения, 3D объекты можно масштабировать, вращать и перемещать. Отдельно ниже будут рассмотрены примеры и варианты формирования интерактивных графических объектов, точнее – будут иллюстрации применения инструментов, позволяющих “оживить”, сделать выводимые графические объекты управляемыми, интерактивными с разными альтернативами не только оформления, но и изменения параметров визуализации.

Используя оригинальные алгоритмы, разработанные специалистами компании Wolfram, отлажены и включены в систему многочисленные функции, которые автоматизируют процесс создания содержательных и эстетически оформляемых иллюстраций функций, данных (структурированных и неструктурированных), причём не только для точек, линий и поверхностей, но и для графов и сетей. Каждое графическое изображение в системе *Mathematica* называется графическим объектом. Условно (к настоящему моменту нет общепринятой терминологии) формируемые системой графические объекты в [1] предложено классифицировать на следующие категории:

- Графики аналитически задаваемых функций одной переменной;
- Визуализация массивов одномерных данных;
- Гистограммы, столбиковые, круговые, секторные диаграммы;
- Финансовые диаграммы;
- Средства 2D графики аналитически задаваемых функций;
- Функции интерполяции и визуализации 2D графики массивов данных;
- Функции формирования и вывода объектов 2D графики;
- Средства 3D графики аналитически задаваемых функций;
- Функции интерполяции и визуализации 3D графики массивов данных;
- Функции сбора, формирования и вывода объектов 3D графики, географические визуализации и вычисления.

Следуя предлагаемой классификации, в данном пособии излагаются и иллюстрируются примерами инструменты 3D графики. Именно инструменты, и считается, что читатели знают основы 3D-моделирования (каркасное, поверхностное, твердотельное моделирование).

Поскольку экран дисплея ПК плоский, то на самом деле объемность фигур лишь имитируется: используются известные способы наглядного представления пространственных объектов, в частности, в виде аксонометрического графика. Более того, вместо построения всех элементов фигуры обычно строится ее каркасная или лоскутная модель, содержащая линии разреза фигуры по взаимно перпендикулярным

плоскостям. В результате фигура представляется в виде совокупности из множества криволинейных четырехугольников. Для придания фигуре большей естественности используется алгоритм удаления невидимых линий каркаса и функциональная закраска четырехугольников по правилу бокового освещения фигуры.

Конкретно далее будут рассмотрены вопросы формирования, оформления, интерактивной работы с функциями трехмерной графики, основные из которых следующие:

▫ Средства 3D графики аналитически задаваемых функций:

- Plot3D – 3D визуализация функции двух переменных, заданной явно аналитически в декартовых координатах; с графиком можно работать интерактивно, обеспечиваются: изменение масштаба, повороты и перемещения (интерактивность поддерживается для всех перечисленных ниже функций выводимых 3D изображений); как и для всех случаев ниже, возможна визуализация нескольких функций, обязательно задать область определения;
- ContourPlot3D – контурный график явно заданной в декартовых координатах функции в пространстве; трехмерный контурный график включает также расположенные в пространстве линии равного уровня, показывающие границы слоев трехмерной фигуры в секущих плоскостях, расположенных параллельно опорной плоскости фигуры (расположенные в пространстве линии равного уровня, полученные при расслоении трехмерной фигуры рядом секущих плоскостей, расположенных параллельно опорной плоскости фигуры);
- SliceContourPlot3D – пространственный контурный график функции на срезах;
- RegionPlot3D – визуализация геометрической фигуры в пространстве;
- DensityPlot3D – пространственный график, иллюстрирующий распределение в пространстве заданной функции от двух переменных на основе непрерывной или специально задаваемой цветовой схемы;
- SliceDensityPlot3D – пространственный плотностный график функции на срезах (сечениях);
- ParametricPlot3D – график параметрически заданной аналитическими выражениями в трехмерном пространстве поверхности или пространственной кривой;
- VectorPlot3D – векторное поле в пространстве (совокупность векторов); должно быть определено, как вектор-функция трех координат;
- SliceVectorPlot3D – генерируется пространственный векторный график функции на плоскостях-срезах (только в указанных плоскостях, в остальном пространстве вектора не отрисовываются);

- StreamPlot3D – потоковая диаграмма в 3D; по умолчанию показывает определенное количество линий тока для достижения примерно равномерной плотности по всему графику, не показывает фонового скалярного поля; обрабатывает переменные *x*, *y* и *z* как локальные;
- RevolutionPlot3D – график поверхности вращения (трехмерные объекты, полученные вращением кривых) с использованием заданного выражения;
- SphericalPlot3D – функция строит график трехмерной поверхности, заданной в сферических координатах.

▫ Функции интерполяции и визуализации 3D графики массивов данных:

- ListPlot3D – трехмерная диаграмма разброса данных, представление в пространстве поверхности по массиву значений высот в точках;
- ListContourPlot3D – контурный 3D-график по массиву значений;
- ListSurfacePlot3D – 3D-график поверхности, восстановленной по списку точек;
- ListPointPlot3D – 3D-диаграмма разброса данных, в пространстве генерируются изображения точек с тремя заданными координатами;
- ListDensityPlot3D – пространственный плотностный график по данным;
- ListSliceContourPlot3D – 3D контурный график данных на срезах;
- ListSliceDensityPlot3D – 3D плотностный график данных на срезах;
- ListVectorPlot3D – векторная 3D-диаграмма по данным;
- DiscretePlot3D – график дискретной функции двух переменных;
- RectangleChart3D – трёхмерная столбиковая/брусковая диаграмма;
- BarChart3D – трёхмерная столбиковая диаграмма;
- Histogram3D – гистограмма 3D;
- SmoothHistogram3D – сглаженная гистограмма в пространстве;
- SectorChart3D – секторная диаграмма в пространстве;
- PieChart3D – пространственная круговая диаграмма.

▫ Функции сбора, формирования и вывода объектов 3D графики, географические визуализации и вычисления:

- GeoGraphics – географические информационные системы (в сочетании с GeoRange, GeoProjection, GeoPosition, GeoBackground, GeoPath, GeoDistance, GeoDisplacement, GeoModel, GeoScaleBar, GeoZoomLevel, GeologicalPeriodData, EarthquakeData, GeoBounds, CountryData, CityData, GeoStyling, ListPlot3D, ReliefPlot);
- GeoListPlot – обеспечивает вывод на географической картооснове указанных объектов (границы, города, пути и другое);
- GeoRegionValuePlot – картограмма с раскраской географических областей указанными цветами;
- GeoElevationData – данные о высоте над уровнем моря (в сочетании с $GeoLocation, GeoPosition, GeoZoomLevel, GeoGraphics, ReliefPlot);
- MountainData – данные о высоте гор (в сочетании с GeoGraphics).

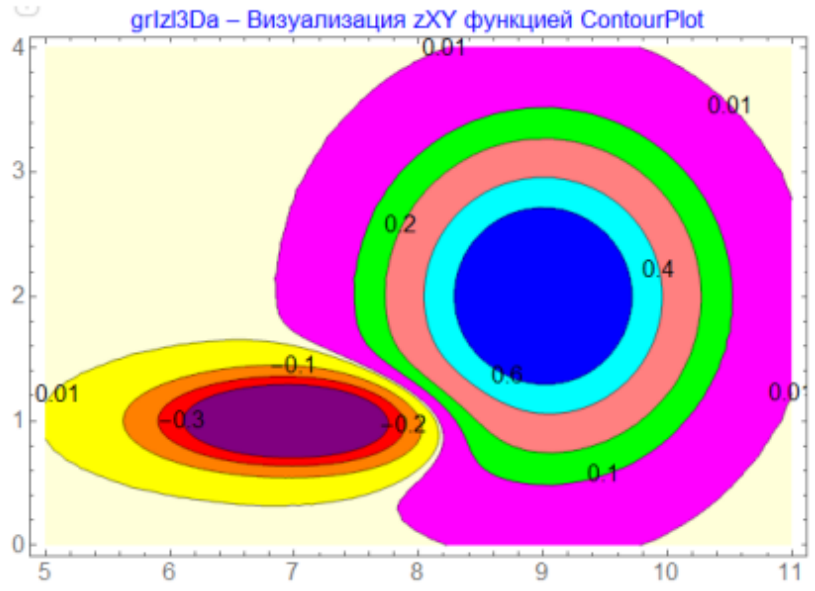
Аналитическая функция *zXY*, используемая в примерах

В настоящем изложении в примерах визуализации, описаниях возможностей формирования и инструментов Wolfram *Mathematica* категории "Средства 3D графики аналитически задаваемых функций" используется выражение $zXY(x,y)$ такое же, как и в "Средства 2D графики аналитически задаваемых функций" ([2]), определяемое аналитически:

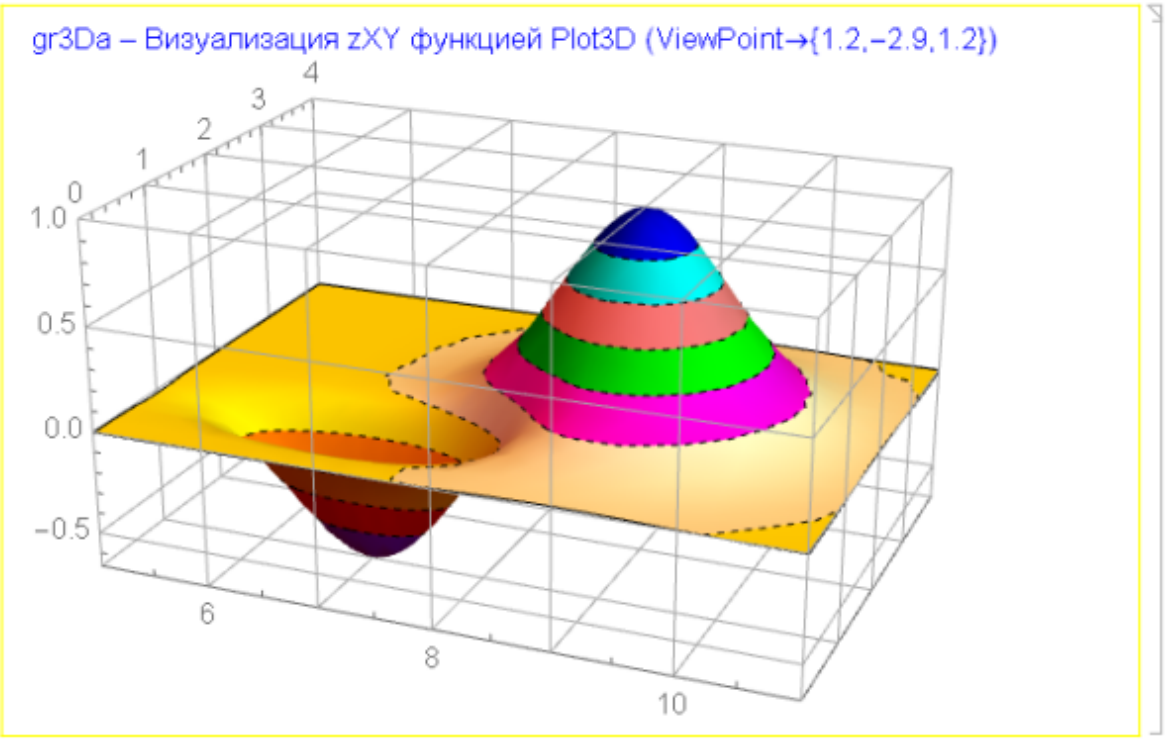
$$zXY(x,\ y) = e^{-(x-9)^2-(y-2)^2} - 2/3\ e^{-(x-7)^2-(3y-3)^2}$$

Вид функции $zXY(x,y)$, иллюстрируют графики grIzl3Da и gr3Da (как они формируется пояснено ниже)

```
xMin := 5; xMax := 11; yMin := 0; yMax := 4;
aspRat = (yMax - yMin) / (xMax - xMin);
zWINmin = -2 / 3; zWINmax = 1; zScale = 3;

zXY[x_, y_] := E^(-(x - 9)^2 - (y - 2)^2) -
    (2 / 3) * E^(-(x - 7)^2 - (3 * y - 3)^2);

zIzl = {-0.3, -0.2, -0.1, -0.01, 0.01, 0.1, 0.2, 0.4, 0.6};
grIzl3Da = ContourPlot[zXY[x, y], {x, xMin, xMax},
   {y, yMin, yMax}, ClippingStyle → Automatic,
   BaseStyle → {Black, 16}, AspectRatio → aspRat,
   (*PlotPoints→60, MaxRecursion→6,*)
   Contours → zIzl, ContourLabels → All,
   ContourShading → {Purple, Red, Orange, Yellow,
     LightYellow, Magenta, Green, Pink, Cyan, Blue},
   PlotLabel → Style
     ["grIzl3Da - Визуализация zXY функцией ContourPlot",
     17, Blue], ImageSize → 550]
```



```
gr3Da = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  FaceGrids → All, (*AxesLabel→{"L","D","H"},*)
  PlotRange → {{xMin, xMax}, {yMin, yMax}, {zWINmin, zWINmax}},
  BoxRatios → {xMax - xMin, yMax - yMin, zScale},
  BaseStyle → {Black, 16},
  Mesh → 9, MeshFunctions → {#3 &}, MeshStyle → {Dashed, Black},
  MeshShading → {Purple, Red, Orange, Yellow,
    LightYellow, Magenta, Green, Pink, Cyan, Blue},
  BaseStyle → {Gray, 14}, ViewPoint → {1.2, -2.9, 1.2},
  PlotLabel → Style[
    "gr3Da - Визуализация zXY
      функцией Plot3D (ViewPoint→{1.2,-2.9,1.2})",
    17, Blue], ImageSize → 550]
```



Оформление трехмерных графиков – основные опции, директивы

В системе *Mathematica* реализован общий механизм задания опций Каждая опция имеет свое название. В качестве последнего аргумента, для большинства графических функций системы, можно указать последовательность правил в виде название->значение, чтобы задать значения для различных опций. Всякая опция, для которой пользователем не указано конкретное значение, рассматриваться системой как имеющая значение "по умолчанию". Вообще говоря, у каждой графической функции есть свой перечень опций, он приведен в подсистеме помощи. Часто не все возможности перечисляются в списках для конкретных функций, потому

что считается, что есть базовые опции, которые действуют везде. Таковые описаны в списке для функции Plot – в перечне этой функции для версии *Mathematica* 11 число опций равно 110.

В материалах лекции и практического занятия 12 [3] в качестве основных, наиболее часто используемых отмечены: AspectRatio – аспектное отношение; Axes – следует ли отображать оси; AxesLabel – обозначения на осях (подписи осей) графика; BaseStyle – базовый стиль; FormatType – тип форматирования, используемый для текста в графике; Frame – следует ли отобразить рамку по периметру графика (окаймлять); FrameLabel – подписи, размещаемые по периметру графика; FrameTicks – какие метки использовать для рамки по периметру графика; GridLines – какие линии сетки добавить; Mesh – опорные (используемые) узлы, кривые в случае поверхностей; PlotLabel – текст, выводимый в качестве надписи графика (заголовок); PlotLegends – легенды графика, используется при визуализации нескольких функций, наборов данных; PlotRange – диапазон координат в границах которого строится график; Ticks – какие метки использовать для осей. Эти и некоторые другие опции, имеющие особенности в графике 3D, поясняются и иллюстрируются ниже.

Для оформления трехмерных графиков, кроме упомянутых выше, могут использоваться (приведены только специфичные для 3D, полный перечень есть в системе помощи, и, например, для функции Graphics3D включены 75 опций, Plot3D – 99) следующие опции:

- AxesEdge – расположение осей на гранях, определяет, на каких гранях ограничительного параллелепипеда (ящика) должны выводиться оси.
- Boxed – показать грани ограничительной рамки, указывает, следует ли рисовать контуры (ребра, грани) ограничительного параллелепипеда в трехмерном изображении.
- BoxRatios – соотношение показываемых длин сторон ограничительной коробки, задает значение отношений длин сторон для ограничительной рамки трехмерного изображения.
- BoxStyle – стиль прорисовки граней ограничительной коробки, задает прорисовку ограничительной рамки.
- ClipFill – определяет, как отсекаемые части поверхности должны выводиться.
- ClippingStyle – стиль отсечения.
- DefaultBoxStyle – стиль рамки по умолчанию.
- DefaultFaceGridsStyle – стиль по умолчанию для сетки на гранях.
- Epilog – эпилог, опция для графических функций, дающая список графических примитивов, которые должны воспроизводиться после воспроизведения главной части графики.
- FaceGrids – сетки на гранях, устанавливает вывод линий сетки на гранях (лицевых сторонах) ограничительной рамки.
- FaceGridsStyle – стиль сетки на гранях.

■ FaceForm – стиль грани, определяет, необходимо или нет удалять невидимые линии каркаса.

■ Lighting – освещение, задает функциональную засветку от постоянного источника света с заданными координатами, указывает, следует ли использовать моделируемую освещенность (simulated illumination) в трехмерных изображениях.

■ Specularity – зеркальность, указывает, какие поверхности объектов трехмерной графики, которые следуют за ними, должны иметь зеркальность.

■ SphericalRegion – оставлять зазор для сферы, описанной вокруг ограничительной рамки, указывает, следует ли конечный образ масштабировать так, чтобы сфера рисовалась вокруг трехмерного отображения.

■ PlotRegion – визуализация геометрического региона.

■ Prolog – пролог, это опция для графических функций, которая предоставляет список графических примитивов, которые должны быть выведены до того, как будет отрисована основная часть графики.

■ RenderAll – указывает, должен или нет генерироваться PostScript для всех многоугольников.

■ Shading – опция, указывающая, следует ли выполнять затенение поверхностей, параметр для графических функций, в котором задается функция определения цветов элементов.

■ TextureCoordinateFunction – функция координат текстуры.

■ TextureCoordinateScaling – масштабирование координат текстуры.

■ VertexColors – цвет вершины.

■ VertexTextureCoordinates – текстурные координаты вершины, опция для графических примитивов, которая определяет координаты текстуры назначенным вершинам.

■ ViewCenter – центр обзора, задает масштабные координаты точки, оказывающейся в центре области отображения в окончательном (последнем, целевом) графике.

■ ViewMatrix – матрица обзора, это опция, которая может использоваться для задания пары явных однородных матриц преобразования и проекции для трехмерных координат, может использоваться для определения матрицы преобразования явной гомогенной (однородной) перспективы.

■ VertexNormals – нормали в вершинах.

■ ViewPoint – точка обзора, меняет точку пространства, из которой рассматривается объект.

■ ViewProjection – проекция, определяет тип проецирования.

■ ViewVector – вектор обзора.

■ ViewVertical – направление вертикали отображаемого объекта, устанавливает, какое направление в относительных координатах должно быть вертикальным в окончательном образе.

Директивы трехмерной графики (отмечены специфические для 3D графики). Помимо опций для трехмерной графики используется ряд графических директив и функций. Наиболее часто применяемые в 3D:

◆ EdgeForm[g] – стиль ребер, указывает, что грани многоугольников должны быть нарисованы с применением графической директивы или списка директив.

◆ FaceForm[gf,gb] – стиль грани, указывает, что передние грани (лицевые) многоугольников должны выводиться с применением графического примитива gf, а задние грани (невидимые поверхности) – посредством gb.

◆ Glow – свечение, графическая директива, которая указывает, что поверхности следующих объектов 3D-графики должны светиться цветом col.

◆ Texture – текстура.

Примеры использования Plot3D для визуализации функции *zXY*

Как отмечено выше, в примерах визуализации и демонстрации вариантов оформления используется аналитическое выражение

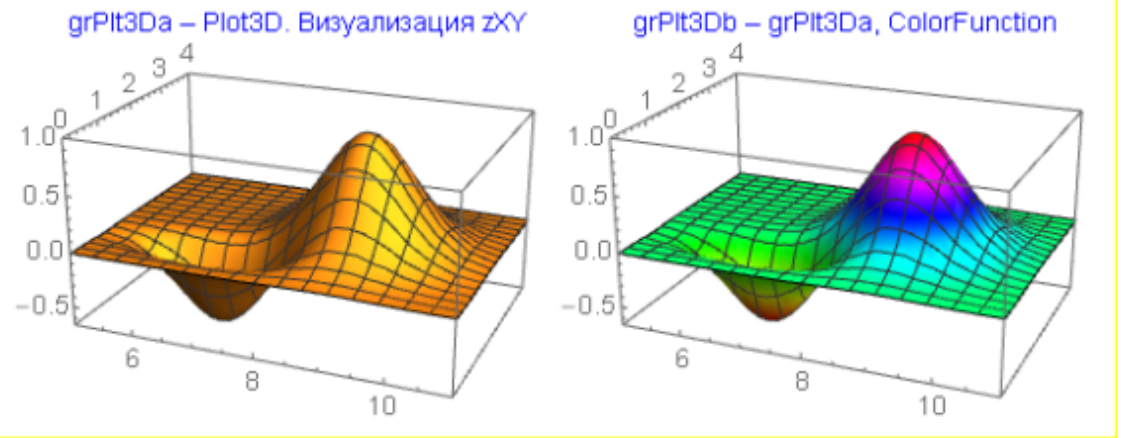
$$zXY(x, y) = e^{-(x-9)^2-(y-2)^2} - 2/3\, e^{-(x-7)^2-(3y-3)^2}$$

Результаты ниже получены с применением разных цветовых схем для раскраски, приведены в нескольких ракурсах осмотра поверхности; в отдельных случаях дополнительно выводятся линии (пунктирные) уровней.

Первая серия иллюстраций – минимум настроек, только задание соотношения показываемых длин сторон ограничительной рамки трехмерного изображения (BoxRatios, rBoxRatios ={xMax-xMin,yMax-yMin,zScale}), точки обзора, из которой рассматривается объект (ViewPoint = {1.2,-2.9,1.2}). Показаны варианты, с раскраской по умолчанию и задаваемой простейшей цветовой схемой (ColorFunction = Hue):

```
xMin := 5; xMax := 11; yMin := 0; yMax := 4;
aspRat = (yMax - yMin) / (xMax - xMin);
zWINmin = -2 / 3; zWINmax = 1; zScale = 3;

zXY[x_, y_] := E^(-(x - 9)^2 - (y - 2)^2) -
    (2 / 3) * E^(-(x - 7)^2 - (3 * y - 3)^2);
rPlotRange = {{xMin, xMax}, {yMin, yMax}, {zWINmin, zWINmax}};
rBoxRatios = {xMax - xMin, yMax - yMin, zScale};
rViewPoint = {1.2, -2.9, 1.2};

grPlt3Da = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    PlotLabel → Style[
      "grPlt3Da - Plot3D. Визуализация zXY", 14, Blue],
    ImageSize → 275];
```

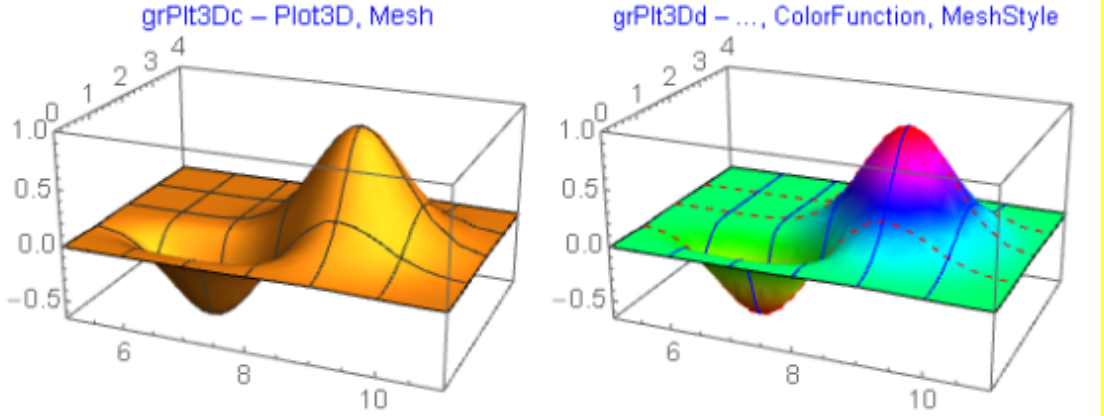
```
grPlt3Db = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    ColorFunction → Hue,
    PlotLabel → Style[
      "grPlt3Db – grPlt3Da, ColorFunction", 14, Blue],
    ImageSize → 275];
Row[{grPlt3Da, grPlt3Db}, Spacer[5]]
```



Следует обратить внимание, что в обоих вариантах выводятся линии каркаса поверхности, и, даже в варианте без указания способа раскраски, цвет поверхности неоднородный.

Оформление, задание стиля сеточных линий. Отметим, что число выводимых линий каркаса, их стиль можно задавать (детали описаны подробно в [2]). В примере grPlt3Dc задано только число линий, в примере grPlt3Dd заданы число линий и дополнительно уточнены их атрибуты (синие линии для *x*-ов, красные пунктирные для *y*-ов):

```
grPlt3Dc = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    Mesh → {5, 3},
    PlotLabel → Style[
      "grPlt3Dc – Plot3D, Mesh", 14, Blue], ImageSize → 275];
grPlt3Dd = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    ColorFunction → Hue,
    Mesh → {5, 3}, MeshStyle → {Blue, {Red, Dashed}},
    PlotLabel → Style[
      "grPlt3Dd – ..., ColorFunction, MeshStyle", 13, Blue],
    ImageSize → 275];

Row[{grPlt3Dc, grPlt3Dd}, Spacer[5]]
```



Адаптивная сетка цифрового поля, используемая при визуализации.

В ряде прикладных задач, в частности, в задачах с разрывными решениями, расчётная область характеризуется наличием разномасштабных элементов сложной неоднородной структуры. Часто достаточно большие зоны имеют малые или умеренные градиенты параметров решения. Вместе с тем встречаются сравнительно узкие области, градиенты параметров решения в которых достигают больших величин.

Для получения достоверного численного решения задач такого типа необходимо использовать расчётные сетки с малыми пространственными шагами. Вычислительные затраты при этом становятся столь значительными, что из-за ограничений вычислительной техники не всегда удаётся получить решение задач с требуемой точностью. В подобных случаях желательно применение динамически адаптивных сеток, позволяющих использование малых пространственных шагов сетки, где это необходимо.

Адаптивная сетка отличается от обычной тем, что размещение ее узлов является частью решения задачи расчета. Решением задачи формирования адаптивной сетки может быть экспорт расчетной сетки, используемой алгоритмами Wolfram *Mathematica* в 3D графике.

Пример предлагаемой к экспорту сетки на иллюстрации grPlt3Dm1a:

```
grPlt3Dm1 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 16},
    ColorFunction → Hue,
    MeshStyle → White, Mesh → All];
grPlt3Dm1a = Show[grPlt3Dm1, ViewPoint → {-0.7, -1.3, 2.9},
  PlotLabel → Style[
    "grPlt3Dm1a - Адаптивная сетка. Mesh→All
ViewPoint→{-0.7, -1.3, 2.9}", 18, Blue],
  ImageSize → 550]
```

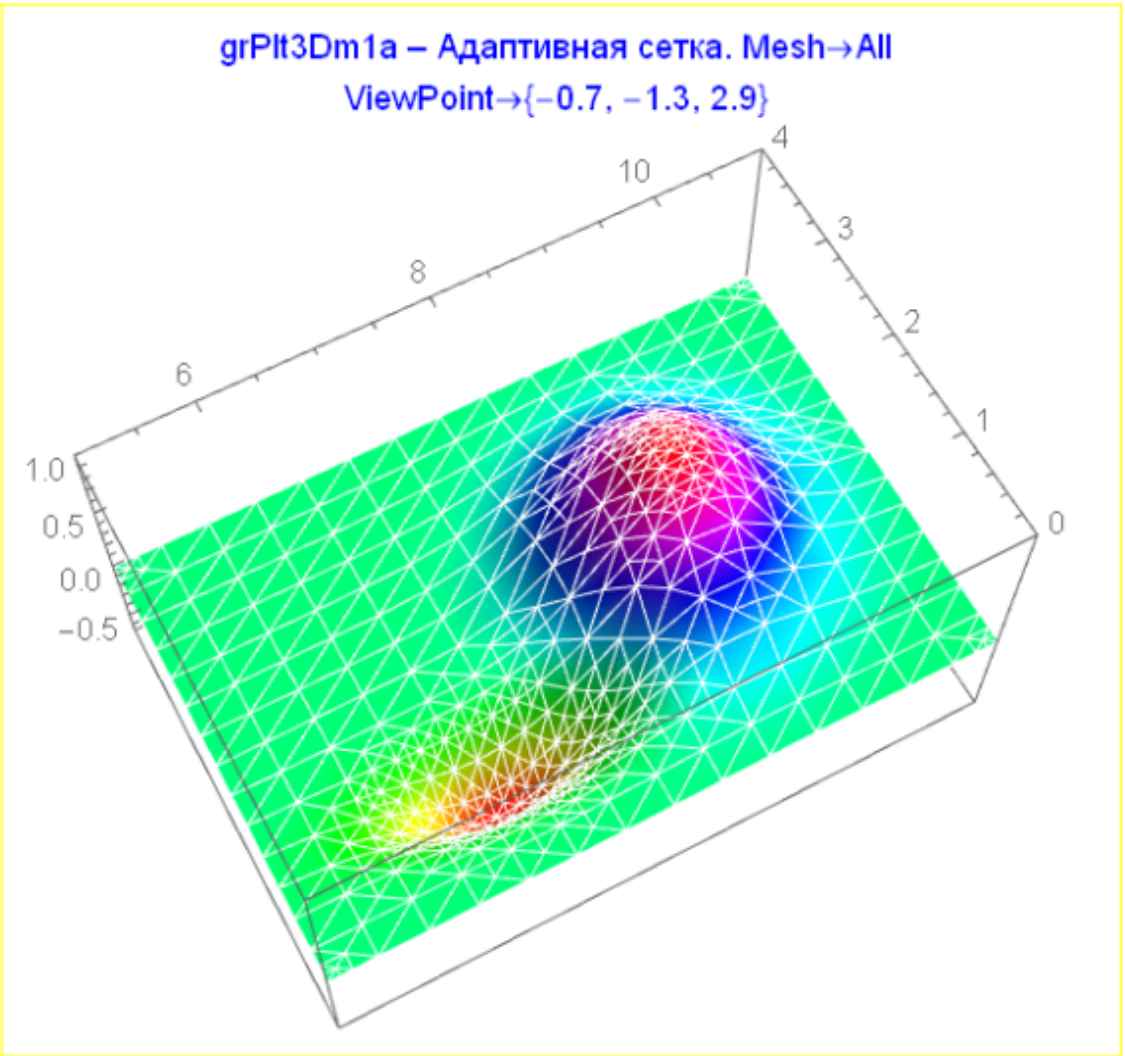
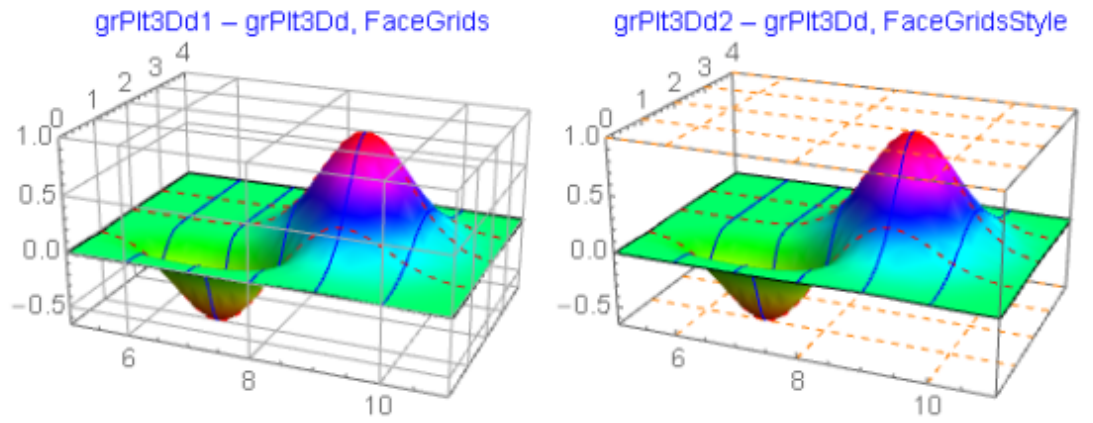


Иллюстрация выше – это также пояснение, на какой сетке реально рассчитывается цифровое поле при подготовке карт контурных, цветовых распределений.

Следует различать сетку для расчетов цифровых распределений на площади, по поверхности (это узлы) и линии(сеть) разметки координатных плоскостей параллелепипеда, в котором выводится изображение 3D (линии координатной сетки, отрезки). Рисунки grPlt3Dd1, grPlt3Dd2 – иллюстрации оформления с выводом калибровки на гранях сетки. Показаны варианты, с прорисовкой сетки на гранях ограничительной коробки, grPlt3Dd1 – стиль по умолчанию, grPlt3Dd2 – стиль задается, причем, только на горизонтальных гранях (FaceGrids устанавливает вывод линий сетки на лицевых сторонах ограничительной рамки):

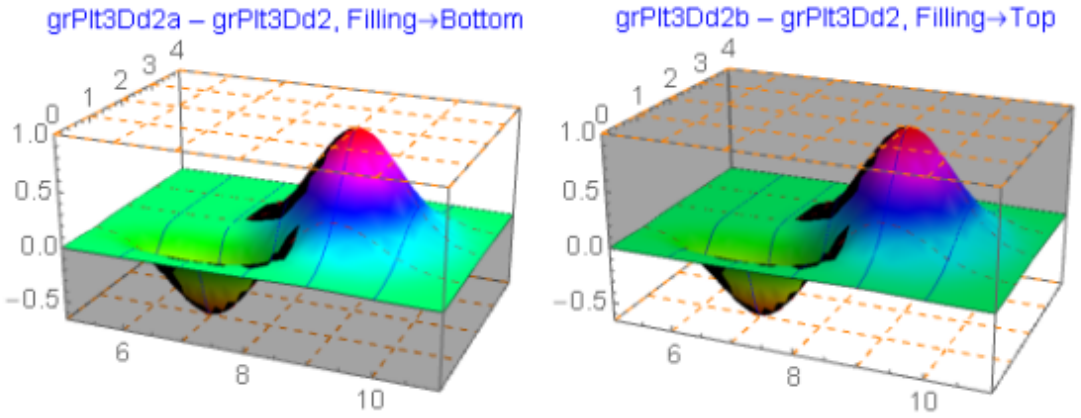
```
grPlt3Dd1 = Show[grPlt3Dd, FaceGrids → All,
    PlotLabel → Style[
      "grPlt3Dd1 - grPlt3Dd, FaceGrids", 14, Blue],
    ImageSize → 275];
```

```
grPlt3Dd2 = Show[grPlt3Dd, (*FaceGrids→All,*)
    FaceGrids → {{0, 0, 1}, {0, 0, -1}},
    FaceGridsStyle → Directive[Orange, Dashed(*Dotted*)],
    PlotLabel → Style[
      "grPlt3Dd2 – grPlt3Dd, FaceGridsStyle", 14, Blue],
    ImageSize → 275];

Row[{grPlt3Dd1, grPlt3Dd2}, Spacer[5]]
```



Варианты заливки нижней и верхней частей стен сцены показаны на grPlt3Dd2a, grPlt3Dd2b:

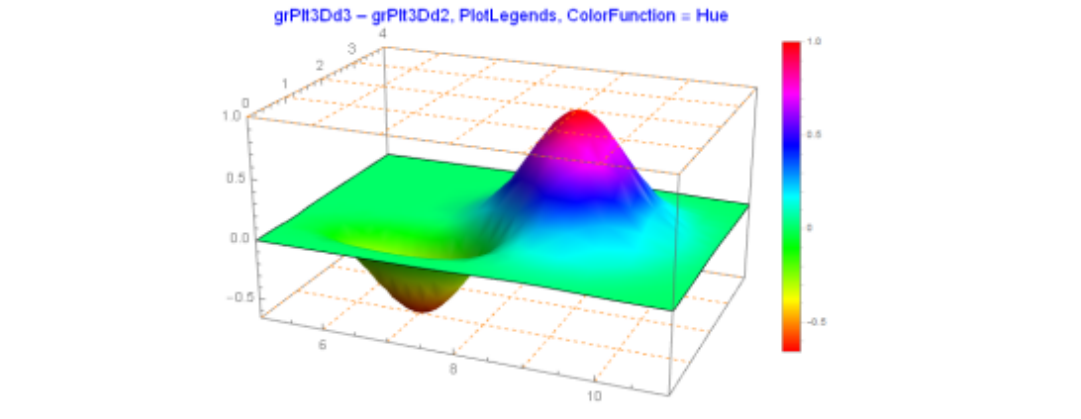
```
grPlt3Dd2a = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    ColorFunction → Hue, Filling → Bottom,
    Mesh → {5, 3}, MeshStyle → {Blue, {Red, Dashed}},
    FaceGrids → {{0, 0, 1}, {0, 0, -1}},
    FaceGridsStyle → Directive[Orange, Dashed],
    PlotLabel → Style[
      "grPlt3Dd2a – grPlt3Dd2, Filling→Bottom", 14, Blue],
    ImageSize → 275];
grPlt3Dd2b = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
    ColorFunction → Hue, Filling → Top,
    Mesh → {5, 3}, MeshStyle → {Blue, {Red, Dashed}},
    FaceGrids → {{0, 0, 1}, {0, 0, -1}},
    FaceGridsStyle → Directive[Orange, Dashed],
    PlotLabel → Style[
      "grPlt3Dd2b – grPlt3Dd2, Filling→Top", 14, Blue],
    ImageSize → 275];

Row[{grPlt3Dd2a, grPlt3Dd2b}, Spacer[5]]
```



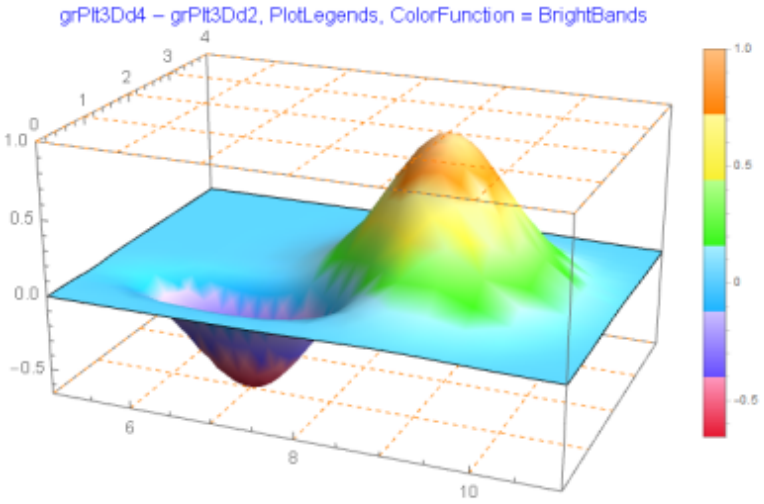
Следует обратить внимание, что в вариантах grPlt3Dd2a, grPlt3Dd2b имеют место искажения цветовой раскраски (например, на рисунках слева у основания поднятия), которые надо устранять заданием PlotPoints, MaxRecursion (отдельно, ниже).

Оформление, задание схемы раскраски, легенда. Напомним, что вопросы выбора цветовой схемы раскраски контурных карт подробно рассмотрены в [3], подключение и оформление легенды с примерами описаны в [1]. Визуализация графиков 3D выполняется аналогично, ниже даны пример подключения легенды, использования иной схемы раскраски:

```
grPlt3Dd3 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → rPlotRange, BoxRatios → rBoxRatios,
  BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
  ColorFunction → Hue, Mesh → None,
  FaceGrids → {{0, 0, 1}, {0, 0, -1}},
  FaceGridsStyle → Directive[Orange, Dashed],
  PlotLegends → BarLegend[Automatic,
    LegendMarkerSize → {20, 350}],
  PlotLabel → Style[
    "grPlt3Dd3 - grPlt3Dd2, PlotLegends, ColorFunction = Hue",
    18, Blue], ImageSize → 550]
```



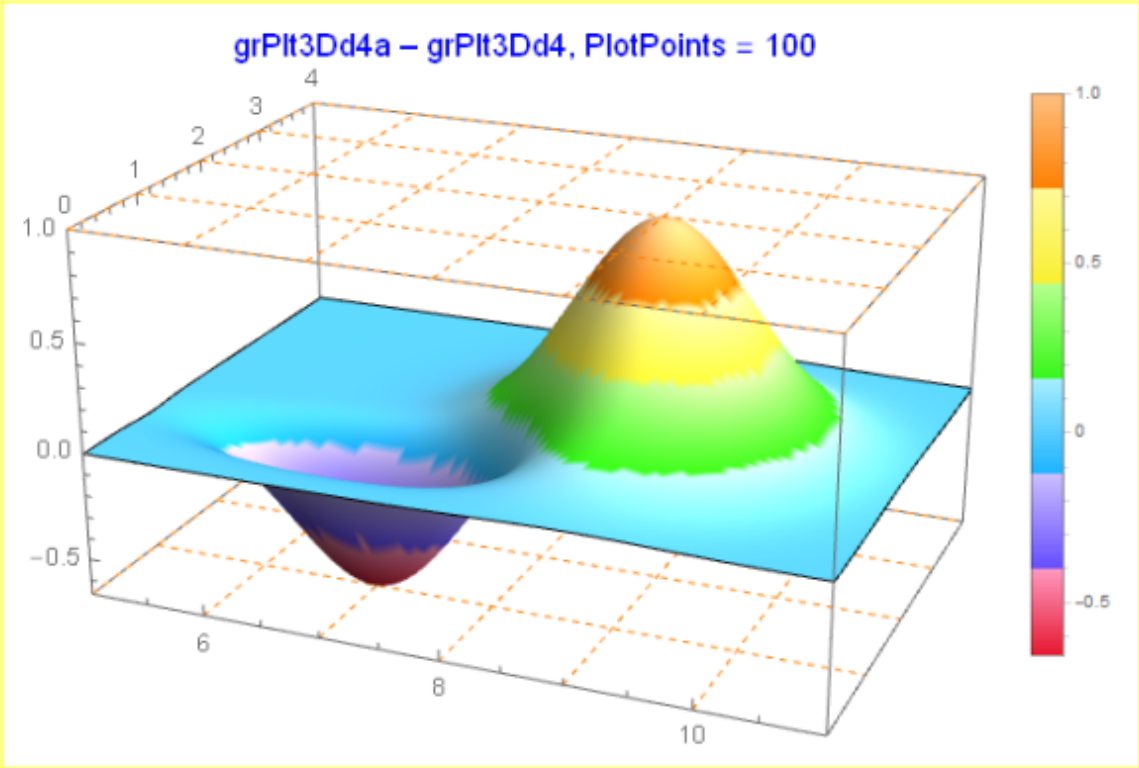
```
grPlt3Dd4 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → rPlotRange, BoxRatios → rBoxRatios,
  BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
  ColorFunction → "BrightBands", Mesh → None,
  (* Pastel  TemperatureMap LakeColors ThermometerColors *)
  FaceGrids → {{0, 0, 1}, {0, 0, -1}},
  FaceGridsStyle → Directive[Orange, Dashed],
  PlotLegends → BarLegend[Automatic,
    LegendMarkerSize → {20, 350}],
  PlotLabel → Style[
    "grPlt3Dd4 - grPlt3Dd2, PlotLegends, ColorFunction
      = BrightBands", 17, Blue], ImageSize → 550]
```



В обоих вариантах grPlt3Dd3, grPlt3Dd4 имеют место заметные искажения однородности окраски в переходных зонах изменения цветов – “выбросы”.

Вариант уменьшения “выбросов” заданием PlotPoints иллюстрирует график grPlt3Dd4a, при этом расчеты с разными значениями PlotPoints показали, что для данных схемы раскраски, масштаба это число по умолчанию примерно 12.

```
grPlt3Dd4a = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → rPlotRange, BoxRatios → rBoxRatios,
  BaseStyle → {Black, 14}, ViewPoint → rViewPoint,
  ColorFunction → "BrightBands", Mesh → None,
  FaceGrids → {{0, 0, 1}, {0, 0, -1}},
  FaceGridsStyle → Directive[Orange, Dashed],
  PlotLegends → BarLegend[Automatic,
    LegendMarkerSize → {20, 350}],
  PlotLabel → Style[
    "grPlt3Dd4a - grPlt3Dd4, PlotPoints = 100", 18, Blue],
  PlotPoints → 100, ImageSize → 550]
```

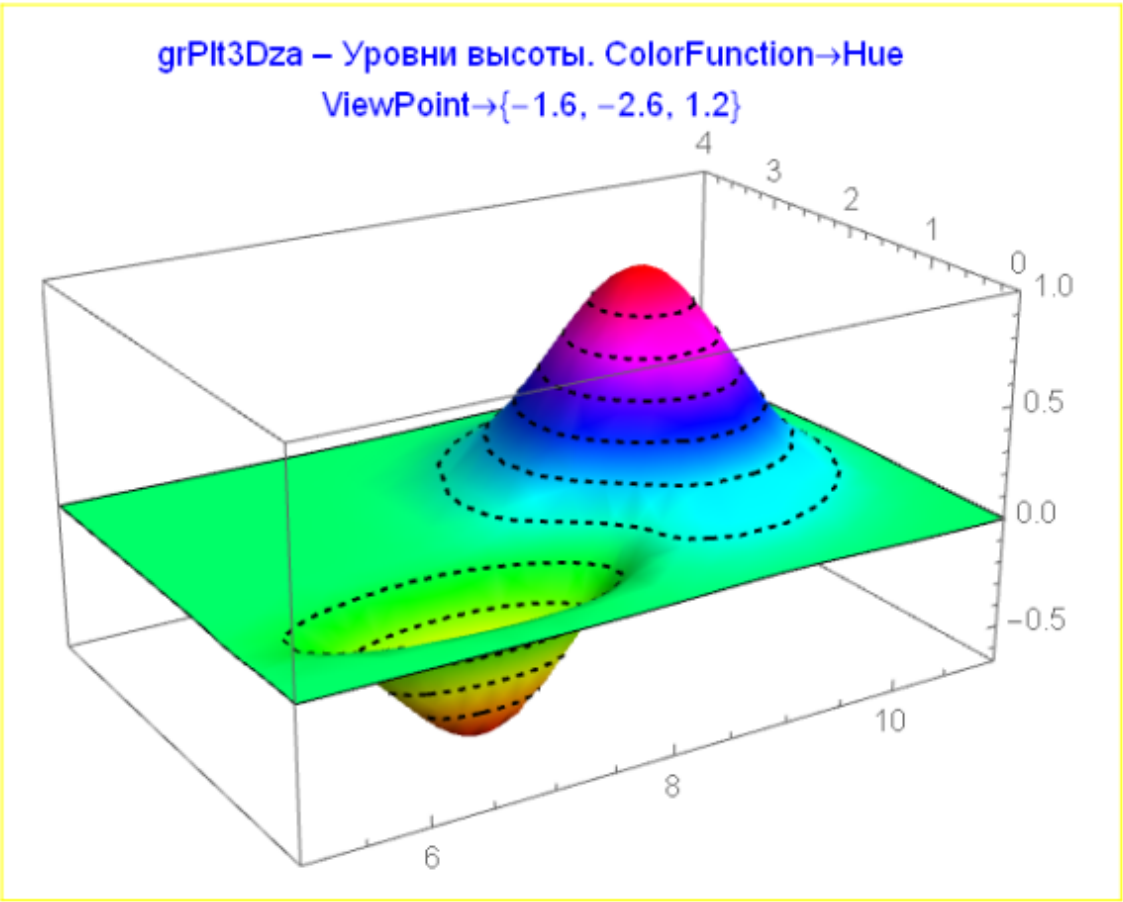


Применение, варианты расчетов с разными значениями MaxRecursion (вплоть до максимально допустимого значения 15) показали, что в случае PlotPoints по умолчанию, а также при задании числа точек в диапазоне 50 - 100 заметного эффекта изменений MaxRecursion нет.

Цветовая раскраска, отметки уровней по высоте, отсечения и разрезы

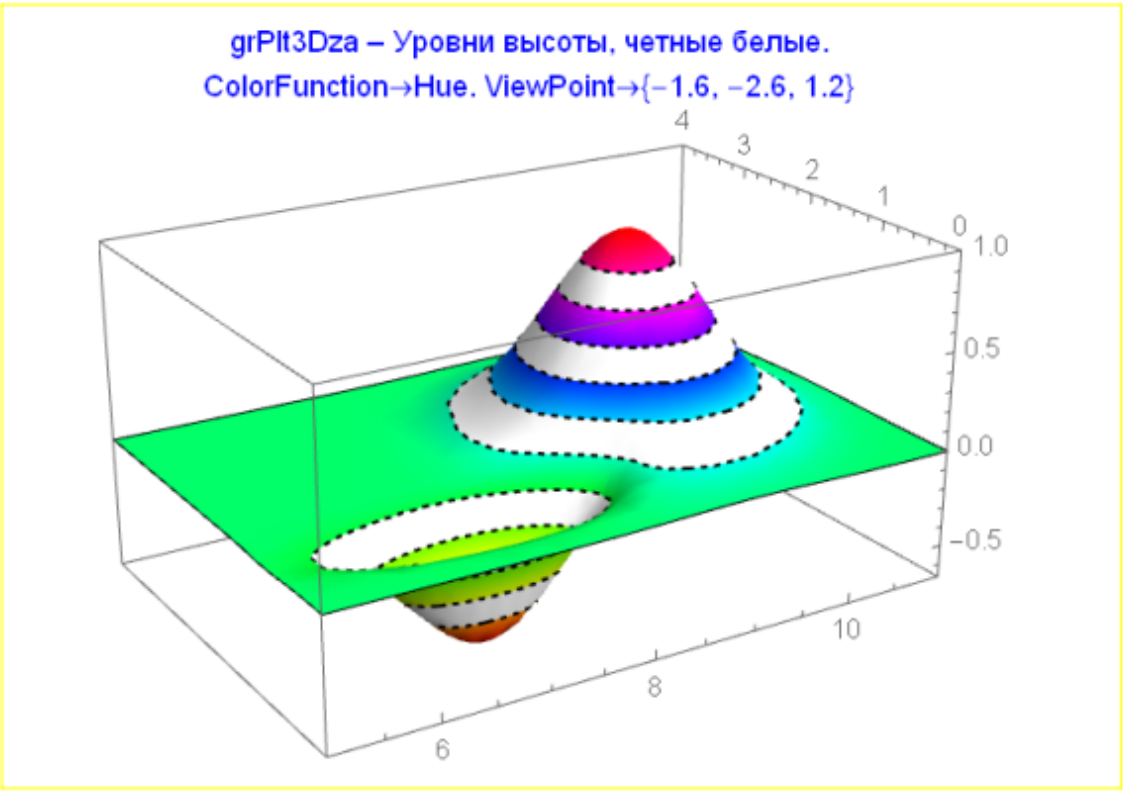
Результаты, приведенные изображения выше получены с применением разных цветовых схем для раскраски, но нет выделения конкретных отметок уровней. Пример ниже показывает вариант, как такое дополнение оформления можно сделать. В примере grPlt3Dza задано число дискретов nMesh = 10:

```
nMesh = 10;
grPlt3Dz0 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    BaseStyle → {Black, 16},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    ColorFunction → Hue,
    Mesh → nMesh, MeshFunctions → {#3 &},
    MeshStyle → {Dashed, Black, Thick}];
grPlt3Dza = Show[grPlt3Dz0, ViewPoint → {-1.6, -2.6, 1.2},
   PlotLabel → Style[
     "grPlt3Dza - Уровни высоты. ColorFunction→Hue
ViewPoint→{-1.6, -2.6, 1.2}", 18, Blue], ImageSize → 550]
```



В примере grPlt3Dzb задано, что каждый четный слой белый (зрелищное выделение интервалов):

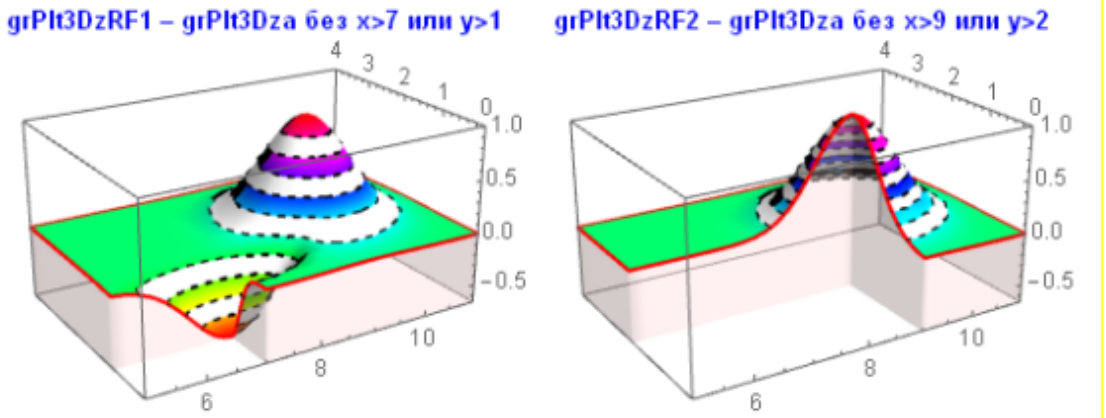
```
grPlt3Dz0 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange,
    BoxRatios → rBoxRatios,
    BaseStyle → {Black, 16},
    ColorFunction → Hue,
    Mesh → nMesh,
    MeshFunctions → {#3 &},
    MeshStyle → {Dashed, Black, Thick},
    MeshShading → {Automatic, White}];

grPlt3Dzb = Show[grPlt3Dz0,
   ViewPoint → {-1.6, -2.6, 1.2},
   PlotLabel → Style[
     "grPlt3Dzb - Уровни высоты, четные белые.
ColorFunction→Hue. ViewPoint→{-1.6, -2.6, 1.2}",
     18, Blue],
   ImageSize → 550]
```



Используя инструменты региональной функции (RegionFunction), можно делать разнообразные разрезы, просматривать фрагменты изображений с устранением закрывающих нужное частей. Примеры даны для графика grPlt3Dza вариантами grPlt3DzaRF1 и grPlt3DzaRF2, в которых для повышения зрелищности показаны частично прозрачной заливкой (Opacity[0.3]) светло красного цвета (LightRed) площади отсечений и красными жирными линиями итоговую границу (BoundaryStyle→ Directive[Red,Thick]) поверхности с учетом скрытого (отсечения):

```
grPlt3DzRF1 = Plot3D[zXY[x, y], {x, xMin, xMax}, y, yMin, yMax},
  PlotRange → rPlotRange, BoxRatios → rBoxRatios,
  BaseStyle → {Black, 12},
  ColorFunction → Hue,
  Mesh → nMesh, MeshFunctions → {#3 &},
  MeshStyle → {Dashed, Black, Thick},
  MeshShading → {Automatic, White},
  Filling → Bottom, FillingStyle →
   Directive[Opacity[0.3], LightRed],
  RegionFunction → Function[{x, y, z}, x > 7 || y > 1],
  BoundaryStyle → Directive[Red, Thick],
  ViewPoint → {-1.6, -2.6, 1.2},
  PlotLabel → Style[
    "grPlt3DzRF1 - grPlt3Dza без x>7 или y>1", 13, Blue, Bold],
  ImageSize → 275];
```

```
grPlt3DzRF2 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 12}, ColorFunction → Hue,
    Mesh → nMesh, MeshFunctions → {#3 &}, MeshStyle →
     {Dashed, Black, Thick}, MeshShading → {Automatic, White},
    Filling → Bottom, FillingStyle →
     Directive[Opacity[0.3], LightRed],
    RegionFunction → Function[{x, y, z}, x > 9 || y > 2],
    BoundaryStyle → Directive[Red, Thick],
    ViewPoint → {-1.6, -2.6, 1.2},
    PlotLabel → Style[
      "grPlt3DzRF2 – grPlt3Dza без x>9 или y>2", 13, Blue, Bold],
    ImageSize → 275];

Row[{grPlt3DzRF1, grPlt3DzRF2}, Spacer[5]]
```



Базовые варианты представления 3D графики

Графическое изображение пространственных объектов отличается тем, что включает построение геометрической проекции трёхмерной модели сцены на плоскость (например, экран компьютера). С созданием и внедрением 3D-дисплеев и 3D-принтеров трёхмерная графика не обязательно включает в себя проецирование на плоскость. Но во всех случаях процесс создания трёхмерной модели объекта предполагает необходимость разработать зрительный объёмный образ желаемого объекта. При этом модель может, как соответствовать объектам из реального мира (автомобили, здания, мебель, цунами, астероид), так и быть полностью абстрактной (проекция четырёхмерного фрактала).

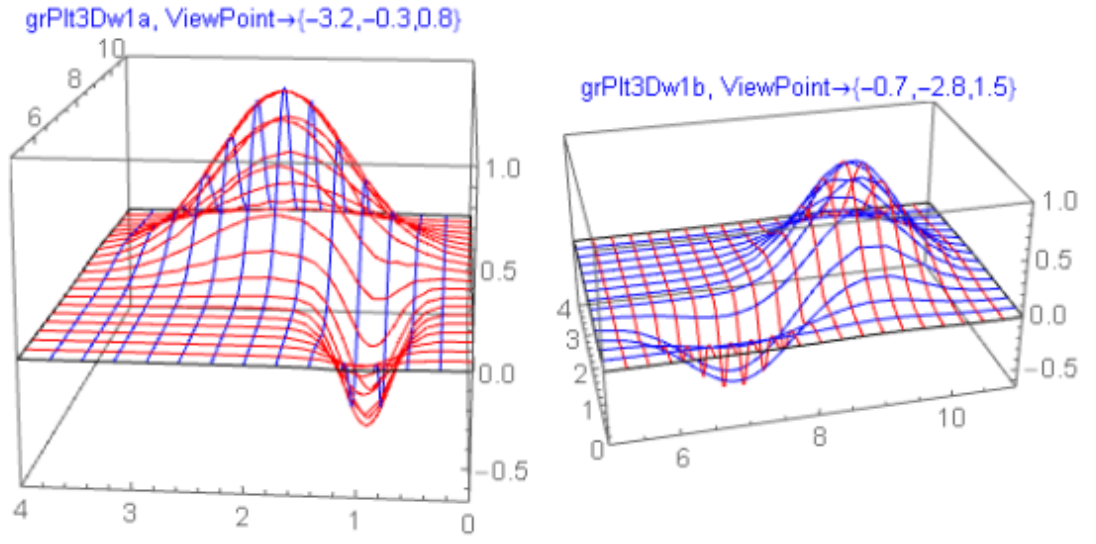
Результаты, приведенные изображения выше получены с применением разных цветовых схем для раскраски, но такое доступно при наличии возможностей управления и отображения цвета. Это не всегда можно реализовать. Есть иные подходы визуализации 3D. Приведем имитации

нескольких из их, причем будем использовать записанное выше математическое описание поверхности.

Каркасное представление (Wireframe representation). Фактически таковое уже проиллюстрировано в примерах с сеточными линиями.

Иллюстрации grPlt3Dw1, grPlt3Dw1a, grPlt3Dw1b включают линии каркаса поверхности, уравнение которой задано. Приведены разные ракурсы (сверху с угла, слева, справа), что частично поясняет возможности интерактивной работы с получаемым графическим изображением:

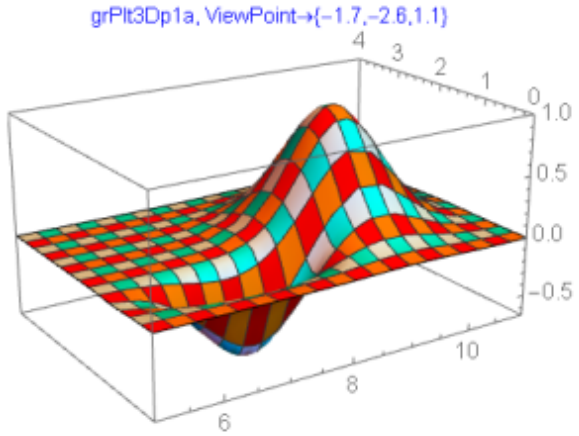
```
grPlt3Dw0 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 16},
    FaceGrids → None, PlotStyle → None,
    Mesh → {19, 13},
    MeshStyle → {Directive[Red, Thin], Directive[Blue, Thin]}];
grPlt3Dw1a = Show[grPlt3Dw0, ViewPoint → {-3.2, -0.3, 0.8},
    PlotLabel → Style["grPlt3Dw1a, ViewPoint→{-3.2,-0.3,0.8}",
      13, Blue], ImageSize → 275];
grPlt3Dw1b = Show[grPlt3Dw0, ViewPoint → {-0.7, -2.8, 1.5},
    PlotLabel → Style["grPlt3Dw1b, ViewPoint→{-0.7,-2.8,1.5}",
      13, Blue], ImageSize → 275];

Row[{grPlt3Dw1a, grPlt3Dw1b}, Spacer[5]]
```



Лоскутное представление (Patchwork representation).

Иллюстрации grPlt3Dp1, grPlt3Dp1a, grPlt3Dp1b включают криволинейные четырехугольники лоскутков, приведены разные ракурсы, в частности вид не только сверху, но и снизу:

```
grPlt3Dp0 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange, BoxRatios → rBoxRatios,
    BaseStyle → {Black, 16},
    FaceGrids → None, PlotStyle → None,
    Mesh → {19, 13},
    MeshShading → {{Red, Orange}, {LightBlue, Cyan}}];
grPlt3Dp1a = Show[grPlt3Dp0, ViewPoint → {-1.7, -2.6, 1.1},
  PlotLabel → Style[
    "grPlt3Dp1a, ViewPoint→{-1.7,-2.6,1.1}", 15, Blue],
  ImageSize → 400]
```




```
grPlt3Dp1b = Show[grPlt3Dp0, ViewPoint → {1.2, -2.9, -0.9},
  PlotLabel → Style[
    "grPlt3Dp1b, ViewPoint→{1.2, -2.9, -0.9}", 15, Blue],
  ImageSize → 400]
```

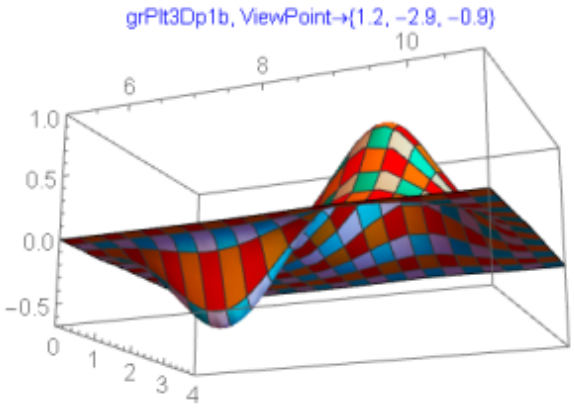


Иллюстрация grPlt3Dp2a включает криволинейные четырехугольники лоскутков, которые в шахматке через один прозрачные, задана типовая схема раскраски градиентная Hue, дополнительно обозначены координатные линии на нижней координатной плоскости:

```
grPlt3Dp2 = Plot3D[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    PlotRange → rPlotRange,
    BoxRatios → rBoxRatios,
    BaseStyle → {Black, 16},
    ColorFunction → Hue,
    Mesh → {19, 13},
    FaceGrids → {{0, 0, -1}},
    FaceGridsStyle → Directive[Red, Dashed],
    MeshShading → {{Automatic, None}, {None, Automatic}}];

grPlt3Dp2a = Show[grPlt3Dp2,
  ViewPoint → {-1.6, -2.6, 1.2},
  PlotLabel → Style[
    "grPlt3Dp2a – Лоскутное представление. ColorFunction→Hue
ViewPoint→{-1.6, -2.6, 1.2}", 18, Blue],
  ImageSize → 550]
```

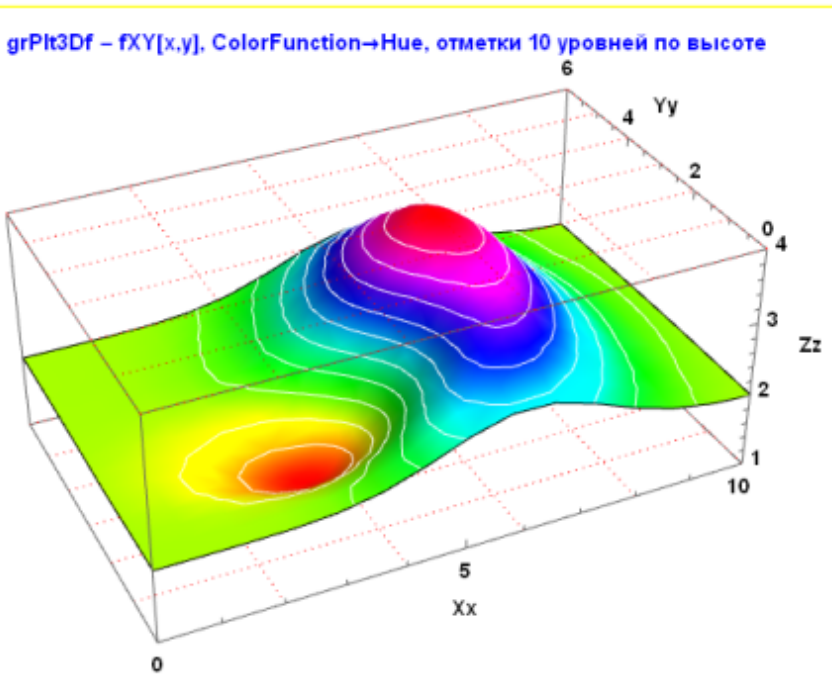
Воксельное представление.

Воксельная графика является растром в 3-мерном исполнении. Воксел расшифровывается как “объемный пиксель”. Обычно, под вокселем

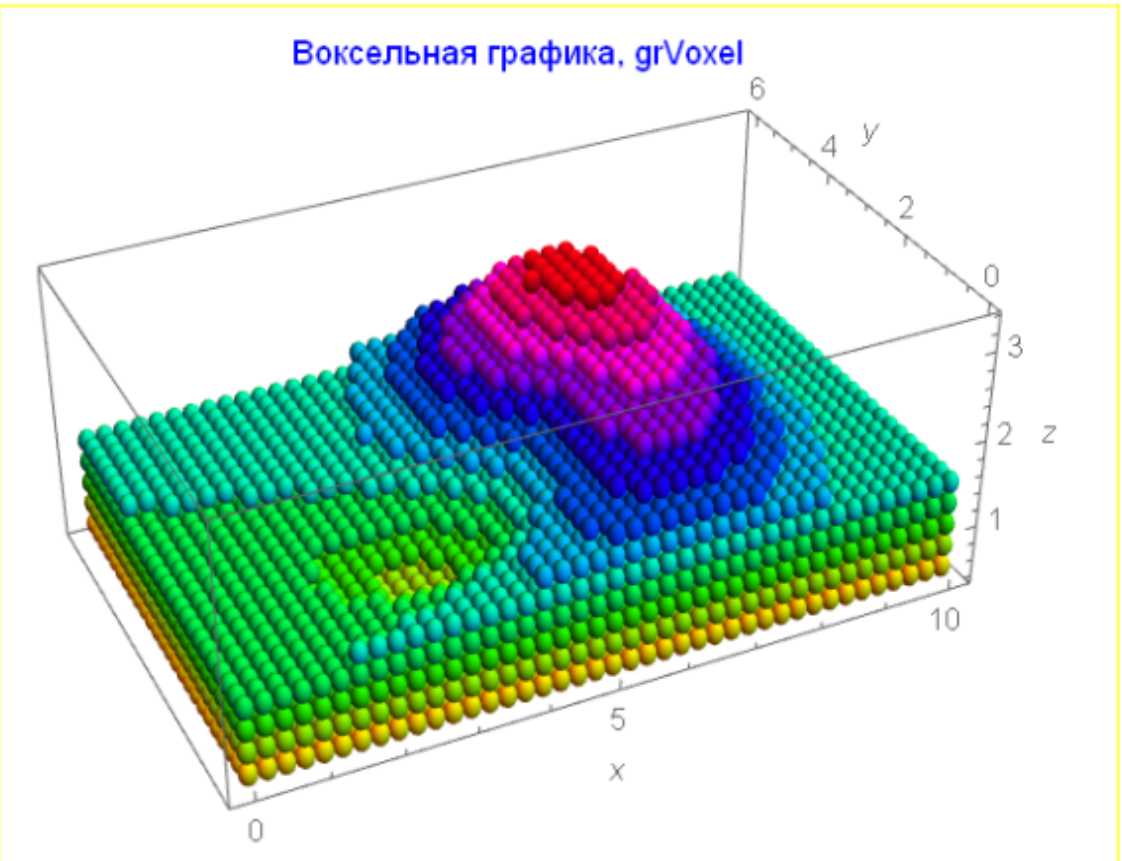
понимается объемный примитив, чаще всего шар, или куб, который имеет задаваемый размер и цвет. Пример визуализации в варианте воксельного представления grVoxel дан для распределения fXY[x,y] изображение которого иллюстрируется графиком grPlt3Df:

```
xmin := 0;
xmax := 10;
ymin := 0;
ymax := 6;
aspRat = ymax / xmax;
zWINmin = 1;
zWINmax = 4;
zScale = 4;
rBoxRatios = {xmax - xmin, ymax - ymin, zScale};
nMesh = 10;
fXY[x_, y_] :=
   2 - E^(-(x / 2 - 2)^2 - (y - 2)^2) + 2 E^(-(x / 2 - 3)^2 - (y / 3 - 1)^2);
```

```
grPlt3Df = Plot3D[fXY[x, y], {x, xmin, xmax}, {y, ymin, ymax},
  PlotRange → {{xmin, xmax}, {ymin, ymax}, {zWINmin, zWINmax}},
  BoxRatios → rBoxRatios,
  BaseStyle → {Black, 16},
  AxesLabel → {"Xx", "Yy", "Zz"}, LabelStyle → Directive[Black, Bold, 14],
  FaceGrids → {{0, 0, 1}, {0, 0, -1}},
  FaceGridsStyle → Directive[Red, Dotted],
  ColorFunction → Hue,
  Mesh → nMesh, MeshFunctions → {#3 &}, MeshStyle → {White, Thin},
  PlotLabel →
   Style["grPlt3Df - fXY[x,y], ColorFunction→Hue, отметки 10 уровней
      по высоте", 15, Blue],
  ViewPoint → {-1.2, -2.5, 1.8}, ImageSize → 550]
```



```
zmax = First[Maximize[fXY[x, y], {x, y}]];
zmin = First[Minimize[fXY[x, y], {x, y}]];
(*nX =1/2; nY =1/2; nZ =1/2; nX =1/3; nY =1/3; nZ =1/3;*)
nX = 1 / 4; nY = 1 / 4; nZ = 1 / 4;

Table[{p = {i, j, k}},
   {i, xmin, xmax, nX}, {j, ymin, nY}, {k, (*zmin*)1, fXY[i, j], nZ}];
grVoxel = Graphics3D[Table[With[{p = {i, j, k}},
     {Hue[0.01 + k * 0.30], (*Opacity[0.75],*)
      Ball[p, nX / 2]}],
    {i, xmin, xmax, nX},
    {j, ymin, ymax, nY}, {k, 0.5, fXY[i, j] - 0.5, nZ}],
  Axes → True, AxesLabel → {x, y, z},
  BaseStyle → {Black, 16},
  BoxRatios → rBoxRatios,
  ViewPoint → {-1.2, -2.5, 1.8}, PlotLabel →
   Style["Воксельная графика, grVoxel", 18, Blue], ImageSize → 550]
```

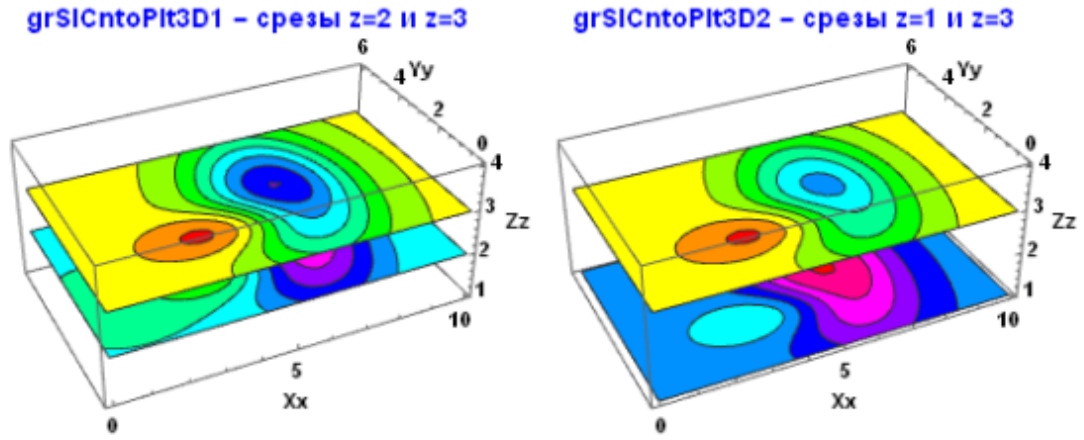


Пространственный контурный график функции на срезах

В системе *Mathematica* реализован эффективный механизм вывода карт изолиний, проецирования на координатные плоскости. Несколько примеров группы grSlCntoPlt3D использования функции SliceContourPlot3D (пространственный контурный график функции на срезах) достаточно

иллюстративны для понимания, записанные опции оформления пояснялись в примерах выше:

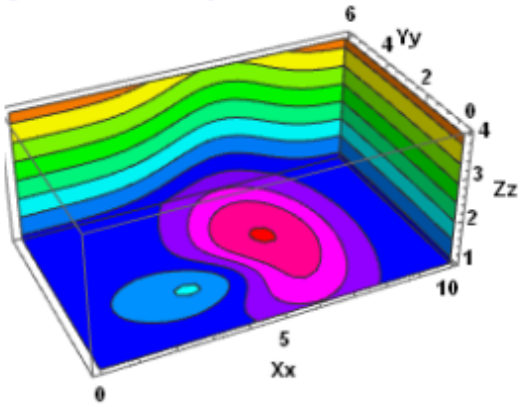
```
grSlCntoPlt3D1 = SliceContourPlot3D[fXY[x, y] - z, {z == 2, z == 3},
    {x, xmin, xmax}, {y, ymin, ymax}, {z, zWINmin, zWINmax},
    BoxRatios → rBoxRatios,
    BaseStyle → {Black, 11},
    AxesLabel → {"Xx", "Yy", "Zz"},
    LabelStyle → Directive[Black, Bold, 11],
    ColorFunction → Hue, Contours → nMesh,
    ViewPoint → {-1.2, -2.5, 1.8},
    PlotLabel → Style["grSlCntoPlt3D1 - срезы z=2 и z=3", 14, Blue],
    ImageSize → 275];
grSlCntoPlt3D2 = SliceContourPlot3D[fXY[x, y] - z, {z == 1, z == 3},
    {x, xmin, xmax}, {y, ymin, ymax}, {z, zWINmin, zWINmax},
    BoxRatios → rBoxRatios,
    BaseStyle → {Black, 11},
    AxesLabel → {"Xx", "Yy", "Zz"},
    LabelStyle → Directive[Black, Bold, 11],
    ColorFunction → Hue, Contours → nMesh,
    PlotLabel → Style["grSlCntoPlt3D2 - срезы z=1 и z=3", 14, Blue],
    ViewPoint → {-1.2, -2.5, 1.8}, ImageSize → 275];

grSlCntoPlt3D12 = Row[{grSlCntoPlt3D1, grSlCntoPlt3D2}, Spacer[5]]
```



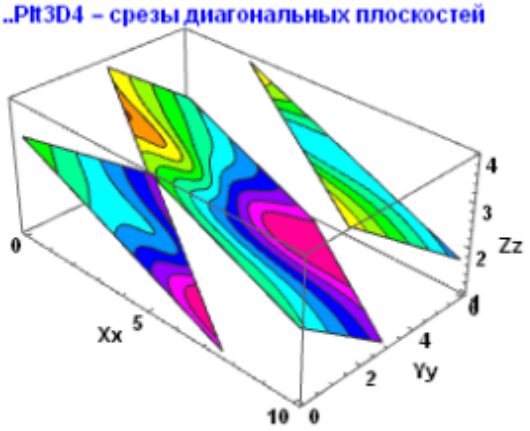

```
grSlCntoPlt3D3 =
  SliceContourPlot3D[fXY[x, y] - z, "BackPlanes", {x, xmin, xmax},
   {y, ymin, ymax}, {z, zWINmin, zWINmax}, BoxRatios → rBoxRatios,
   BaseStyle → {Black, 11},
   AxesLabel → {"Xx", "Yy", "Zz"},
   LabelStyle → Directive[Black, Bold, 11],
   ColorFunction → Hue, Contours → nMesh,
   ViewPoint → {-1.2, -2.5, 1.8},
   PlotLabel →
    Style["grSlCntoPlt3D3 - срезы плоскостей тыла", 12, Blue],
   ImageSize → 275];
```

```
grSlCntoPlt3D4 = SliceContourPlot3D[fXY[x, y] - z, "DiagonalStackedPlanes",
    {x, xmin, xmax}, {y, ymin, ymax}, {z, zWINmin, zWINmax},
    BoxRatios → rBoxRatios, BaseStyle → {Black, 11},
    AxesLabel → {"Xx", "Yy", "Zz"},
    LabelStyle → Directive[Black, Bold, 11],
    ColorFunction → Hue, Contours → nMesh,
    ViewPoint → {2., -2., 1.7},
    PlotLabel →
     Style["..Plt3D4 - срезы диагональных плоскостей", 12, Blue],
    ImageSize → 275];

grSlCntoPlt3D23 = Row[{grSlCntoPlt3D3, grSlCntoPlt3D4}, Spacer[20]]
```

grSlCntoPlt3D3 – срезы плоскостей тыла



..Plt3D4 – срезы диагональных плоскостей



ВИЗУАЛИЗАЦИИ МАТЕМАТИЧЕСКИХ ПОВЕРХНОСТЕЙ

Поверхности второго порядка, примеры оформления

Трехмерные графики функций задаваемых явно.

Выше отмечены основные возможности графической функции ContourPlot3D. Напомним основное:

- ContourPlot3D – контурный график явно заданной в декартовых координатах функции в пространстве; трехмерный контурный график включает также расположенные в пространстве линии равного уровня, показывающие границы слоев трехмерной фигуры в секущих плоскостях, расположенных параллельно опорной плоскости фигуры (расположенные в пространстве линии равного уровня, полученные при расслоении трехмерной фигуры рядом секущих плоскостей, расположенных параллельно опорной плоскости фигуры).

Первый аргумент ContourPlot3D – выражение, график которого будет сформирован. Следующие аргументы определяют границы изменения переменных, задаваемые в виде списков.

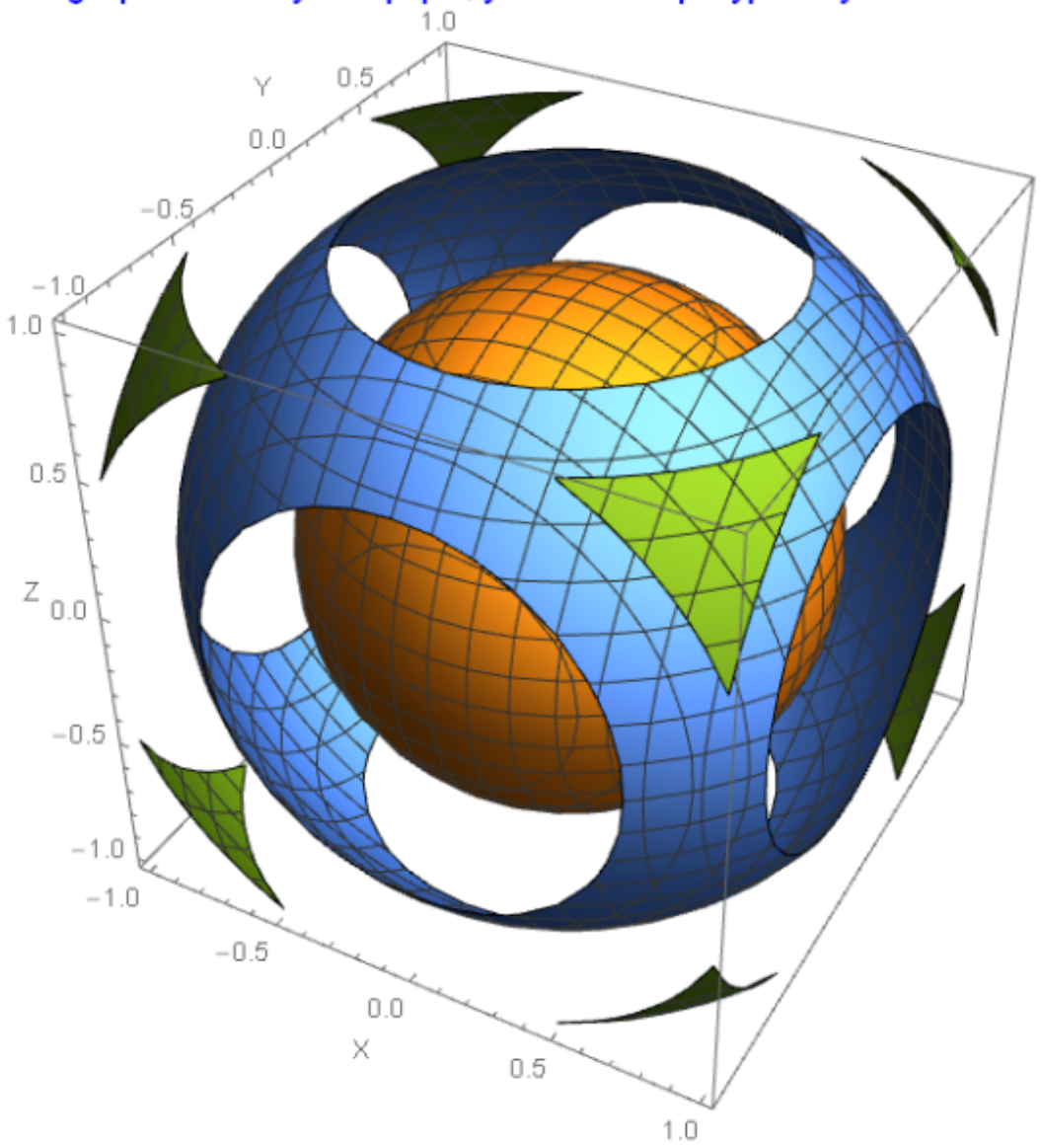
Mathematica по умолчанию размещает график 3D в ящик (ограничительную коробку), ориентация которого определяется положением точки, из которой он рассматривается. Если не указано, ориентация имеет следующие настройки: шкала x почти горизонтальная, находится впереди слева, x увеличивается направо; шкала y проходит спереди назад вправо, y увеличивается назад; шкала z – вертикальная слева, значения z возрастают вверх.

Записывая выражение $x^2 + y^2 + z^2$ для описания поверхности сферы, применяя функцию ContourPlot3D с установками по умолчанию, получим изображение grSph0a.

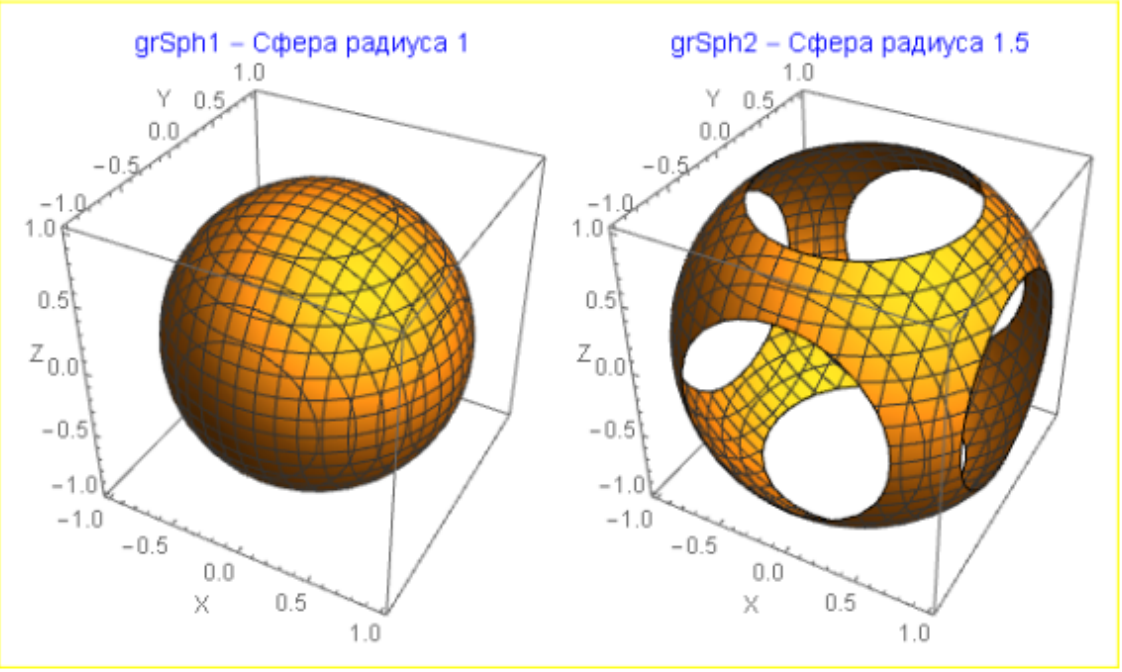
Следует понимать, подтвердим это иллюстрацией, покажем, что при применении функции ContourPlot3D выводятся поверхности. Для этого приведены построенные изображения grSph1, grSph2. На рисунке справа (grSph2) часть сферы радиуса большего, чем размер ящика, в который она помещена, выполнено отсечение. В пустотах (что отсечено) просматриваются внутренности:

```
grSph0 = ContourPlot3D[x^2 + y^2 + z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, BaseStyle → {Black, 14},
    Axes → True, AxesLabel → {"X", "Y", "Z"}];
grSph0a = Show[grSph0, BaseStyle → {Black, 14},
  PlotLabel → Style
     ["grSph0a – Рисуем сферы, установки и ракурс по умолчанию",
     18, Blue],
  ImageSize → 550]
```

grSph0a – Рисуем сферы, установки и ракурс по умолчанию




```
grSph1 = ContourPlot3D[x^2 + y^2 + z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {1},
    Axes → True, AxesLabel → {"X", "Y", "Z"}, PlotLabel → Style
      ["grSph1 - Сфера радиуса 1", 14, Blue],
    BaseStyle → {Black, 12}, ImageSize → 275];
grSph2 = ContourPlot3D[x^2 + y^2 + z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {1.5},
    Axes → True, AxesLabel → {"X", "Y", "Z"}, PlotLabel → Style
      ["grSph2 - Сфера радиуса 1.5", 14, Blue],
    BaseStyle → {Black, 12},
    ImageSize → 275];

Row[{grSph1, grSph2}, Spacer[5]]
```

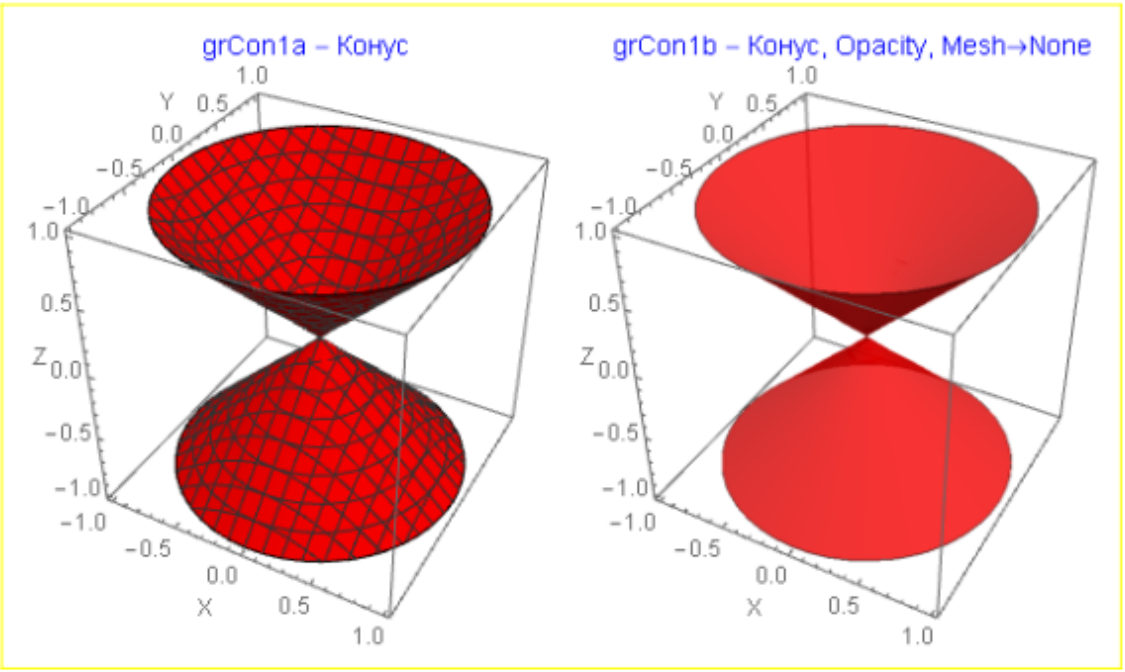


Такая же ситуация, виды во всех результатах применения ContourPlot3D. Основные опции этой графической функции описаны и иллюстрируются примерами.

Поверхность конуса, разные условия прозрачности и окраски. Примеры оформления.

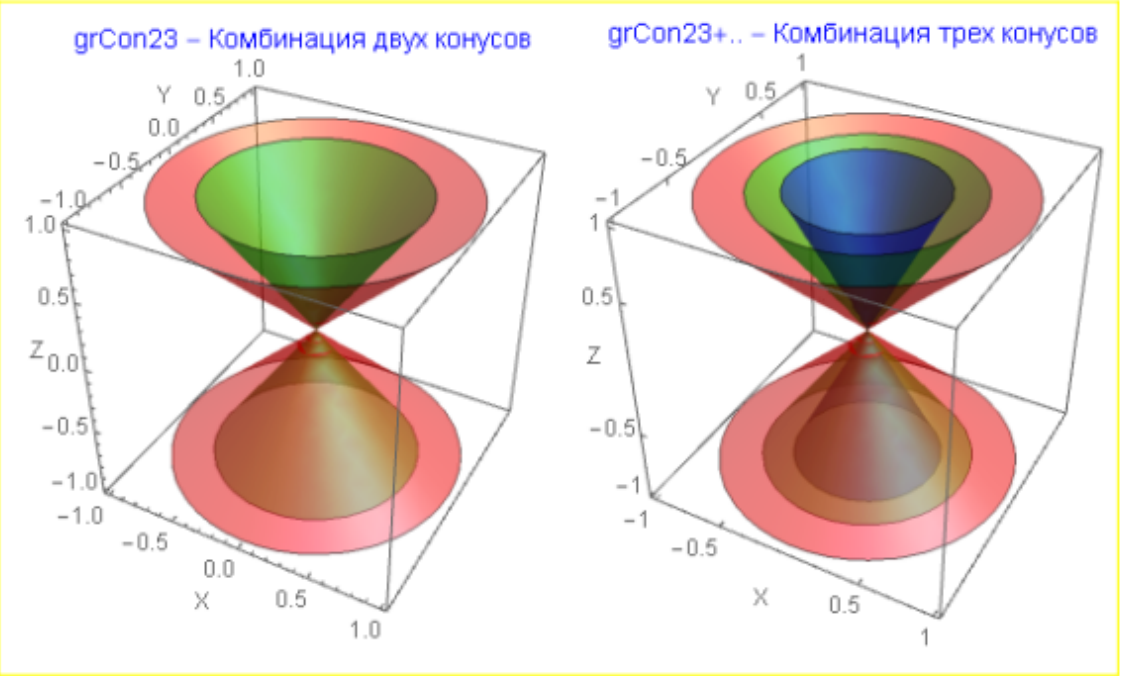
Иллюстрации задания цвета раскраски, прозрачности и отмены вывода сеточных линий, графики grCon1a, grCon1b:

```
grCon1a = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    ContourStyle → Red,
    PlotLabel → Style
["grCon1a - Конус", 14, Blue],
    BaseStyle → {Black, 12},
    ImageSize → 275];
grCon1b = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    ContourStyle → Directive[Red, Opacity[0.8]],
    Mesh → None,
    PlotLabel → Style
["grCon1b - Конус, Opacity, Mesh→None", 14, Blue],
    BaseStyle → {Black, 12},
    ImageSize → 275];

Row[{grCon1a, grCon1b}, Spacer[5]]
```



Варианты комбинирования. Также на иллюстрации справа (grCon23+..) изменен способ калибровки осей (опцией Ticks), заданы конкретные метки:

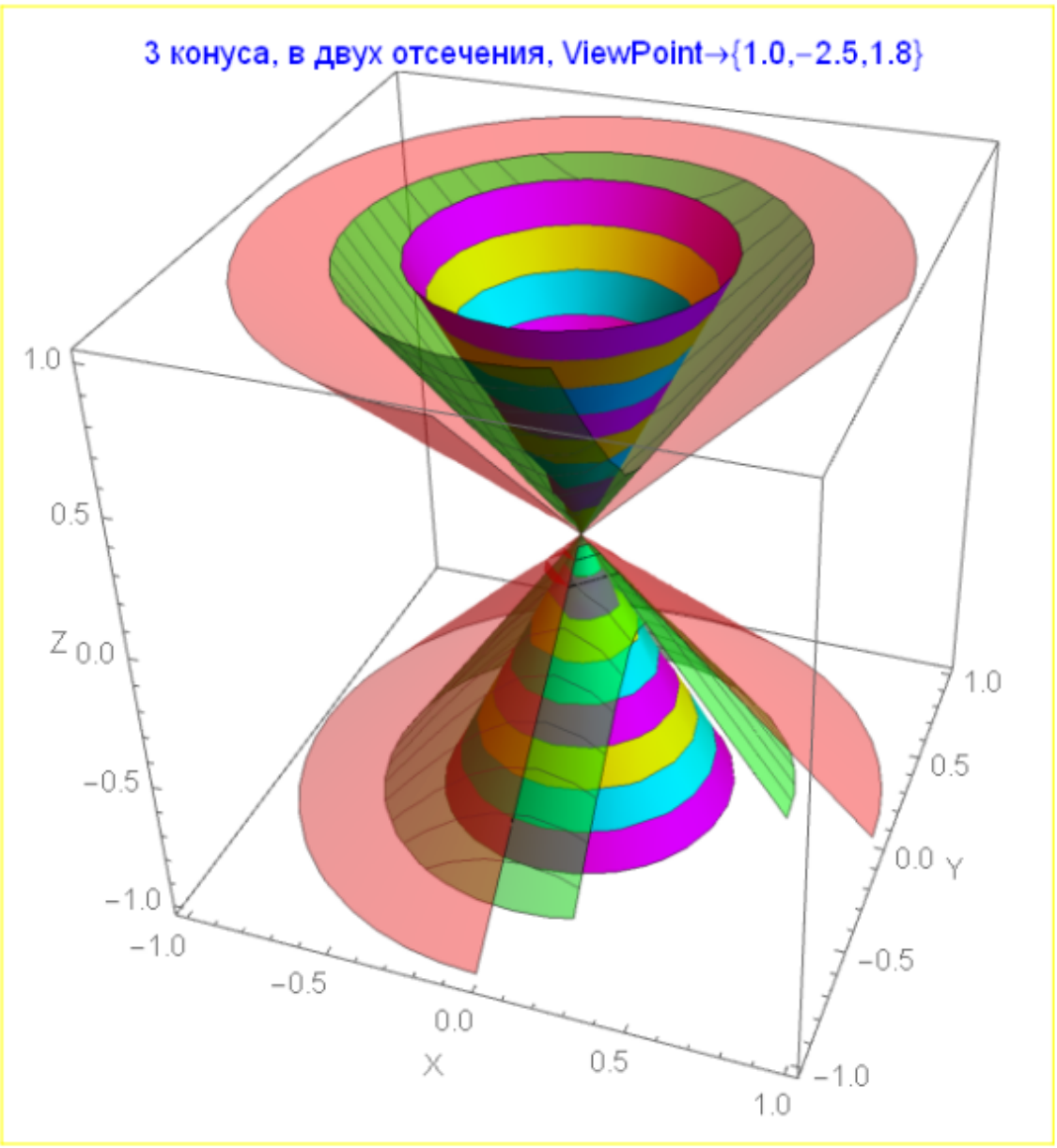
```
grCon2 = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    Mesh → None,
    ContourStyle → Directive[Red, Opacity[0.5]]];
grCon3 = ContourPlot3D[2 * x^2 + 2 y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    Mesh → None, ContourStyle → Directive[Green, Opacity[0.5]]];
grCon23 = Show[grCon2, grCon3, BaseStyle → {Black, 12},
    PlotLabel → Style
["grCon23 – Комбинация двух конусов", 14, Blue],
    ImageSize → 275];
grCon4 = ContourPlot3D[4 * x^2 + 4 y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1},
    Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"}, Mesh → None,
    ContourStyle → Directive[Blue, Opacity[0.5]]];

Row[{grCon23,
  Show[grCon2, grCon3, grCon4, Ticks → {{-1, -0.5, 0.5, 1},
     {-1, -0.5, 0.5, 1}, {-1, -0.5, 0.5, 1}},
    PlotLabel → Style
["grCon23+.. – Комбинация трех конусов", 14, Blue],
    BaseStyle → {Black, 12}, ImageSize → 275]}, Spacer[10]]
```



Примеры оформления по участкам, уровням. Отсечение (отмена вывода) части поверхности.

Иллюстрации и примеры этой группы получаются путем использования MeshFunctions→{#2&} (отрисовка уровней второго аргумента – *y*), MeshFunctions→{#3&} (отрисовка уровней третьего аргумента – *z*, высоты). Также в примере при формировании изображения grb следует обратить внимание на запись выносимого в заголовок текста rPlotLabel = “3 конуса, в двух отсечения, ViewPoint→{1.0,-2.5,1.8}”:

```
gr1a = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    ContourStyle → Directive[Red, Opacity[0.4]], Mesh → None,
    RegionFunction → Function[{x, y, z}, x < 0 || y > 0]];
gr2b = ContourPlot3D[2 * x^2 + 2 y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    ContourStyle → Directive[Green, Opacity[0.5]],
    MeshFunctions → {#2 &},
    RegionFunction → Function[{x, y, z}, x < 1 / 5 || y > 0]];
gr3b = ContourPlot3D[4 * x^2 + 4 y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1}, Contours → {0},
    MeshFunctions → {#3 &},
    MeshShading → {Magenta, Cyan, Yellow}];
rPlotLabel =
  "3 конуса, в двух отсечения, ViewPoint→{1.0,-2.5,1.8}";
grb = Show[gr1a, gr2b, gr3b, BaseStyle → {Black, 16},
  ViewPoint → {1.0, -2.5, 1.8},
  PlotLabel → Style[rPlotLabel, 18, Blue], ImageSize → 550]
```

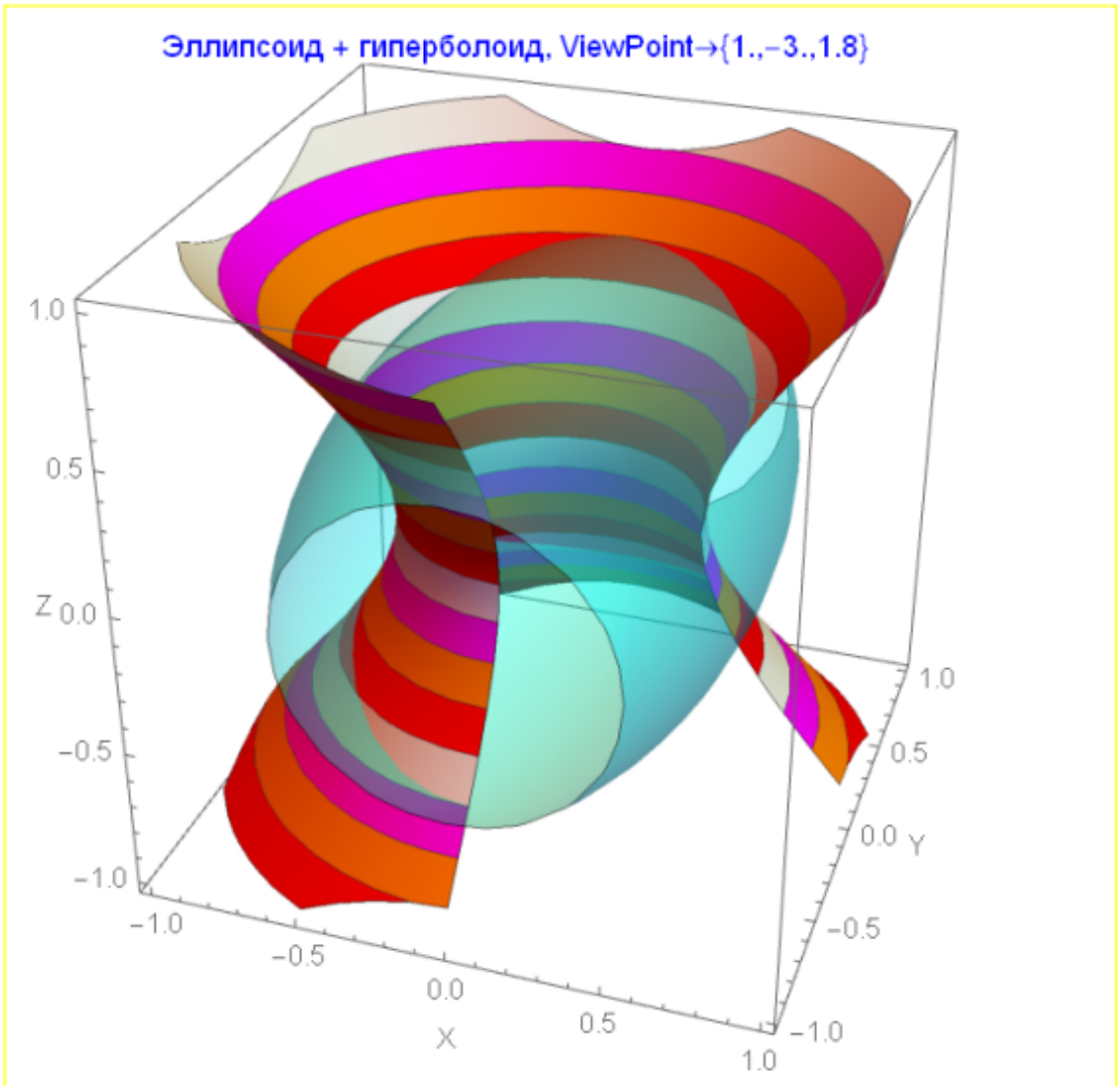


Примеры синтеза, визуализации разных аналитических выражений.

Эллипсоид ($x^2 + y^2/4 + z^2$, Contours→{0.55}) и гиперболоид ($x^2 + y^2 - z^2$, Contours→{0.25}), разные условия окраски (MeshShading→{Red, Orange, Magenta, LightBrown}, RegionFunction→Function[{x,y,z},x<0||y>0]):

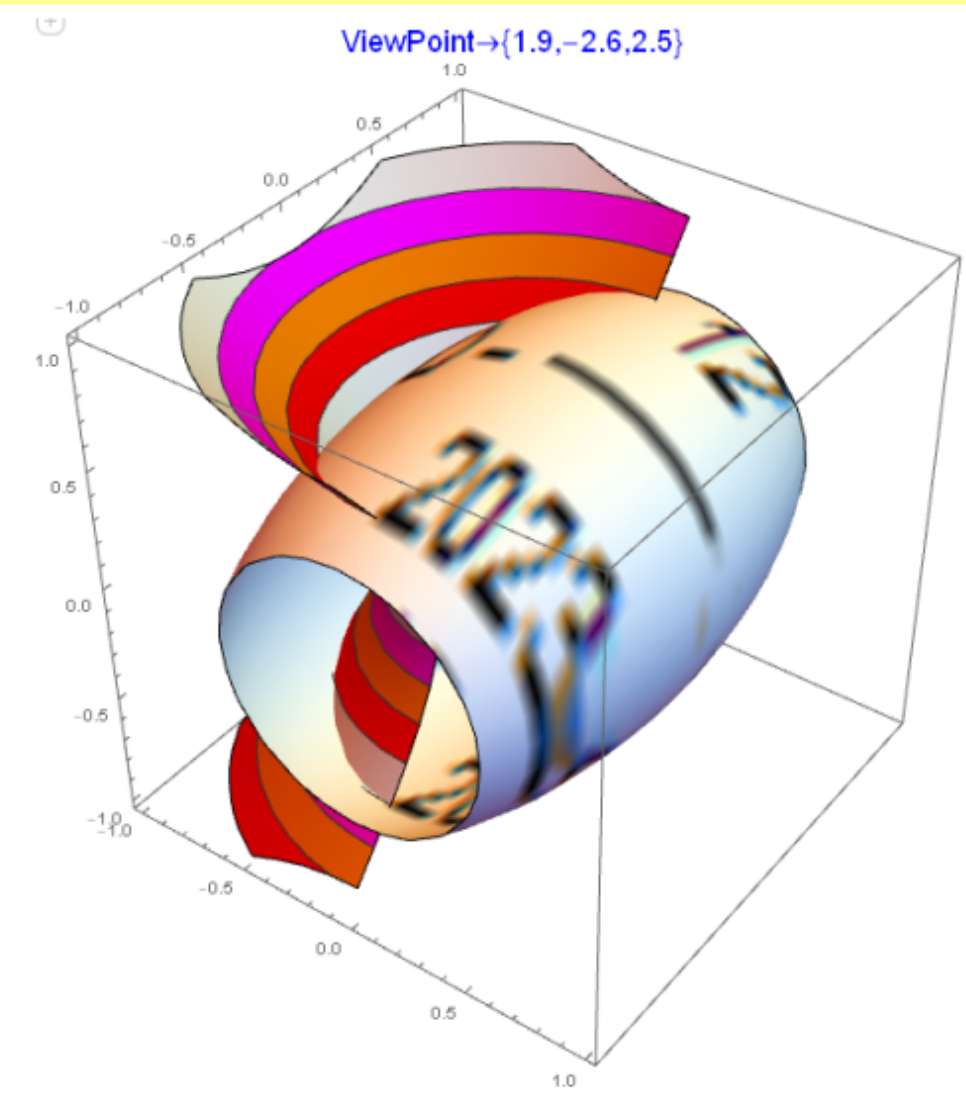
```
gr6 = ContourPlot3D[x^2 + y^2 / 4 + z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1},
    Contours → {0.55},
    PlotPoints → 10,
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    Mesh → None,
    ContourStyle → Directive[Cyan, Opacity[0.4]]];
```

```
gr7 = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1},
    Contours → {0.25},
    PlotPoints → 10, MeshFunctions → {#3 &},
    MeshShading → {Red, Orange, Magenta, LightBrown},
    RegionFunction → Function[{x, y, z}, x < 0 || y > 0]];

Show[gr6, gr7, BaseStyle → {Black, 16},
 ViewPoint → {1., -3., 1.8}, PlotLabel →
  Style["Эллипсоид + гиперболоид, ViewPoint→{1.,-3.,1.8}",
   18, Blue, FontFamily → "Arial"],
 ImageSize → 550]
```



Примеры синтеза, комбинирования с использованием разных аналитических выражений. Разные схемы раскраски, текстуры и подписи по поверхностям.

```
gr8 = ContourPlot3D[x^2 + y^2/4 + z^2, {x, -1, 1},
    {y, -1, 1}, {z, -1, 1}, Contours → {0.55},
    PlotPoints → 10, ContourStyle → Texture[3 D - 12 / 2023],
    Mesh → None,
    ContourStyle → Directive[Red, Opacity[0.5]]];

gr9 = ContourPlot3D[x^2 + y^2 - z^2,
    {x, -1, 1}, {y, -1, 1}, {z, -1, 1},
    Contours → {0.25}, PlotPoints → 10,
    Axes → True, AxesLabel → {"X", "Y", "Z"},
    MeshFunctions → {#3 &},
    MeshShading → {Red, Orange, Magenta, LightBrown},
    RegionFunction → Function[{x, y, z}, x < 0 || y > 1]];

gr89 = Show[gr8, gr9,
  ViewPoint → {1.9, -2.6, 2.5}, PlotLabel →
    Style["ViewPoint→{1.9,-2.6,2.5}", 18, Blue], ImageSize → 550]
```



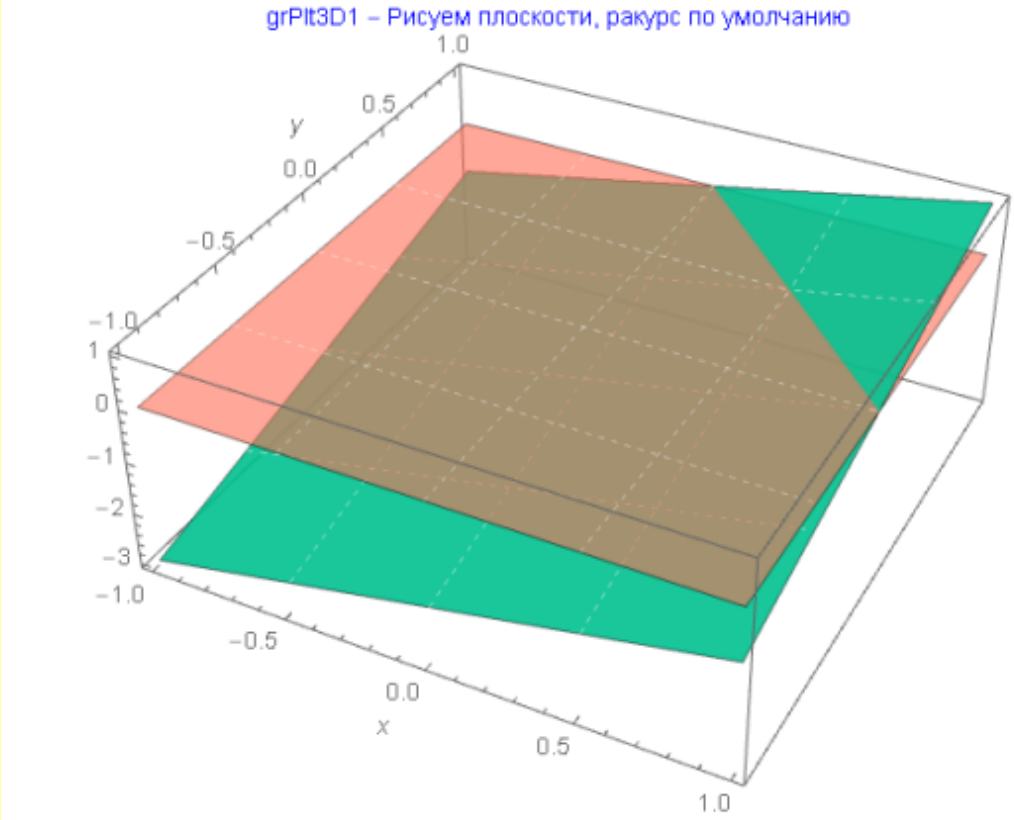
Как рисовать плоскости

Напоминание:

- Plot3D – 3D визуализация функции двух переменных, заданной явно аналитически в декартовых координатах ...

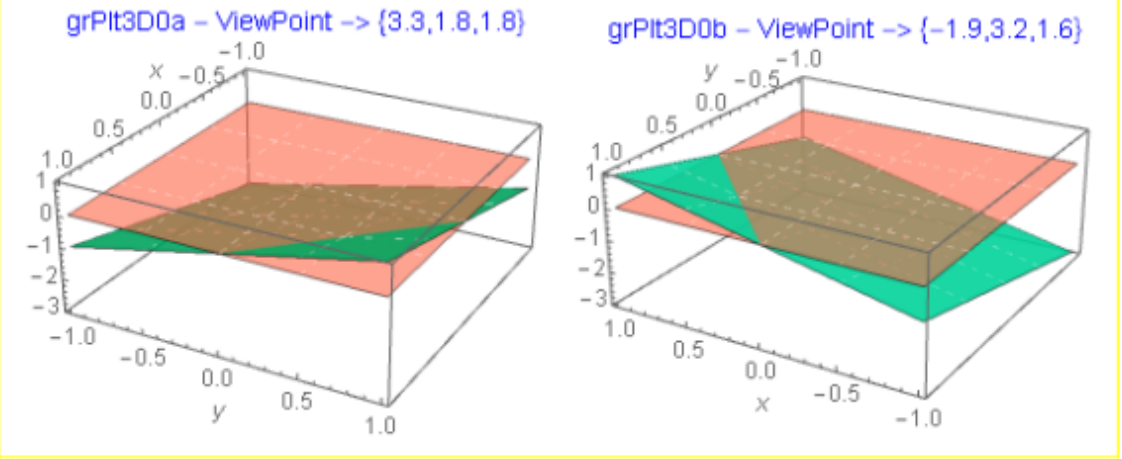
Нарисуем плоскости $x+y=1$ и $x=0$, $y=0$; направление (ракурс) осмотра не задается (по умолчанию); разными заданы прозрачности; на плоскостях рисуются линии сетки:

```
f[x_, y_] = 0;
grPlt3D0 = Plot3D[{f[x, y], x + y - 1}, {x, -1, 1}, {y, -1, 1},
    AxesLabel → Automatic,
    Mesh → 3, MeshStyle → Directive[Dashed, White],
    PlotStyle → {
       {Directive[Pink, Opacity[0.6]]},
       {Directive[Cyan, Opacity[0.9]]}}];
grPlt3D1 = Show[grPlt3D0, BaseStyle → {Black, 14},
  PlotLabel → Style
     ["grPlt3D1 - Рисуем плоскости, ракурс по умолчанию",
     14, Blue],
  ImageSize → 550]
```



Рисунки grPlt3D0a и grPlt3D0b – иллюстрации выводимой сцены, плоскостей в двух разных ракурсах осмотра:

```
grPlt3D0a = Show[grPlt3D0, BaseStyle → {Black, 12},
    ViewPoint → {3.3, 1.8, 1.8},
    PlotLabel → Style
      ["grPlt3D0a - ViewPoint -> {3.3,1.8,1.8}", 14, Blue],
    ImageSize → 275];
grPlt3D0b = Show[grPlt3D0, BaseStyle → {Black, 12},
    ViewPoint → {-1.9, 3.2, 1.6}, PlotLabel → Style
      ["grPlt3D0b - ViewPoint -> {-1.9,3.2,1.6}", 14, Blue],
    ImageSize → 275];
Row[{grPlt3D0a, grPlt3D0b}, Spacer[5]]
```



Примеры визуализации с использованием RegionPlot3D

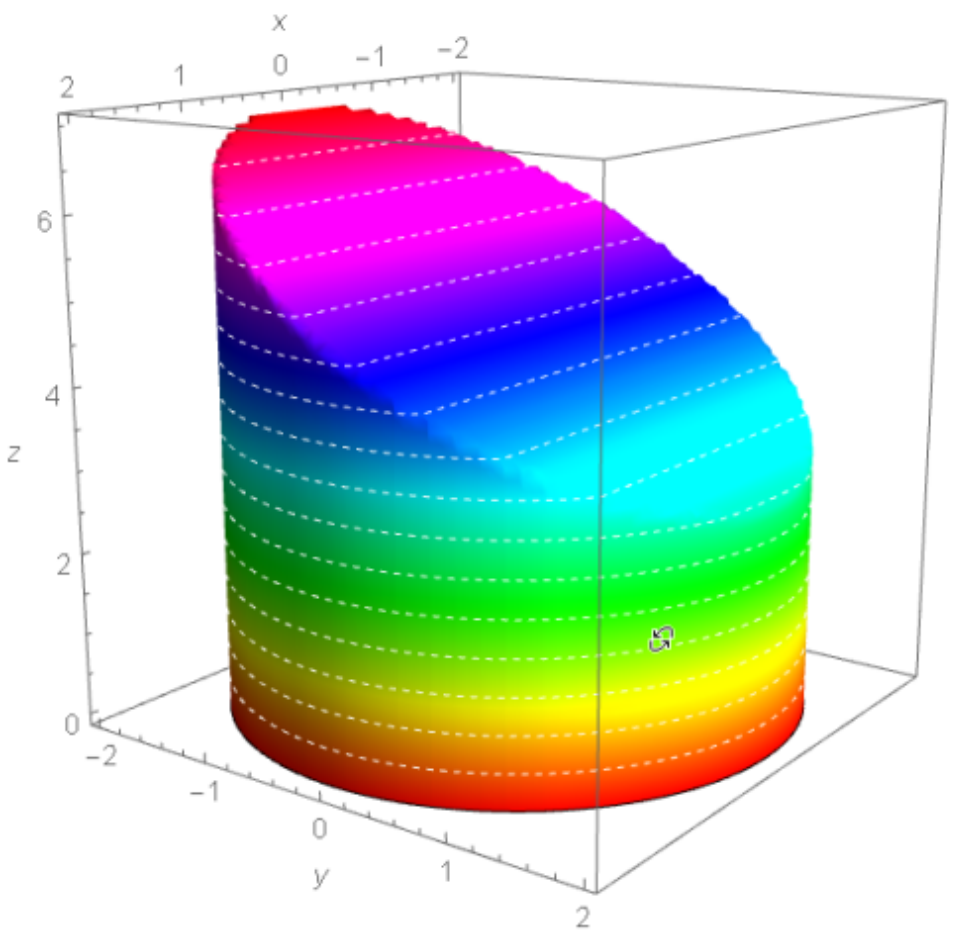
Напоминание:

- RegionPlot3D – визуализация геометрической фигуры в пространстве;

Визуализация формы усеченного цилиндра. Варианты задания цветовой раскраски. Пример grRP1 – иллюстрация способа раскраски с использованием градиентной цветовой схемы Hue; дополнительно белыми пунктирными линиями (MeshStyle→{{Dashed,White}) выводятся уровни высоты (MeshFunctions→{#3&}):

```
Print[
 Style["grRP1, ColorFunction→Hue, ViewPoint→{2.6,1.9,0.8}",
  Blue, 16]]
grRP1 = RegionPlot3D[x^2 + y^2 ≤ 4 && z + y ≤ 5,
  {x, -2, 2}, {y, -2, 2}, {z, 0, 7},
  BaseStyle → {Black, 16}, Lighting → "Neutral",
  AxesLabel → Automatic, PlotPoints → 60,
  ColorFunction → Hue,
  MeshFunctions → {#3 &}, MeshStyle → {{Dashed, White}},
  BoxRatios → {4, 4, 4}, ViewPoint → {2.6, 1.9, 0.8},
  ImageSize → 550]
```

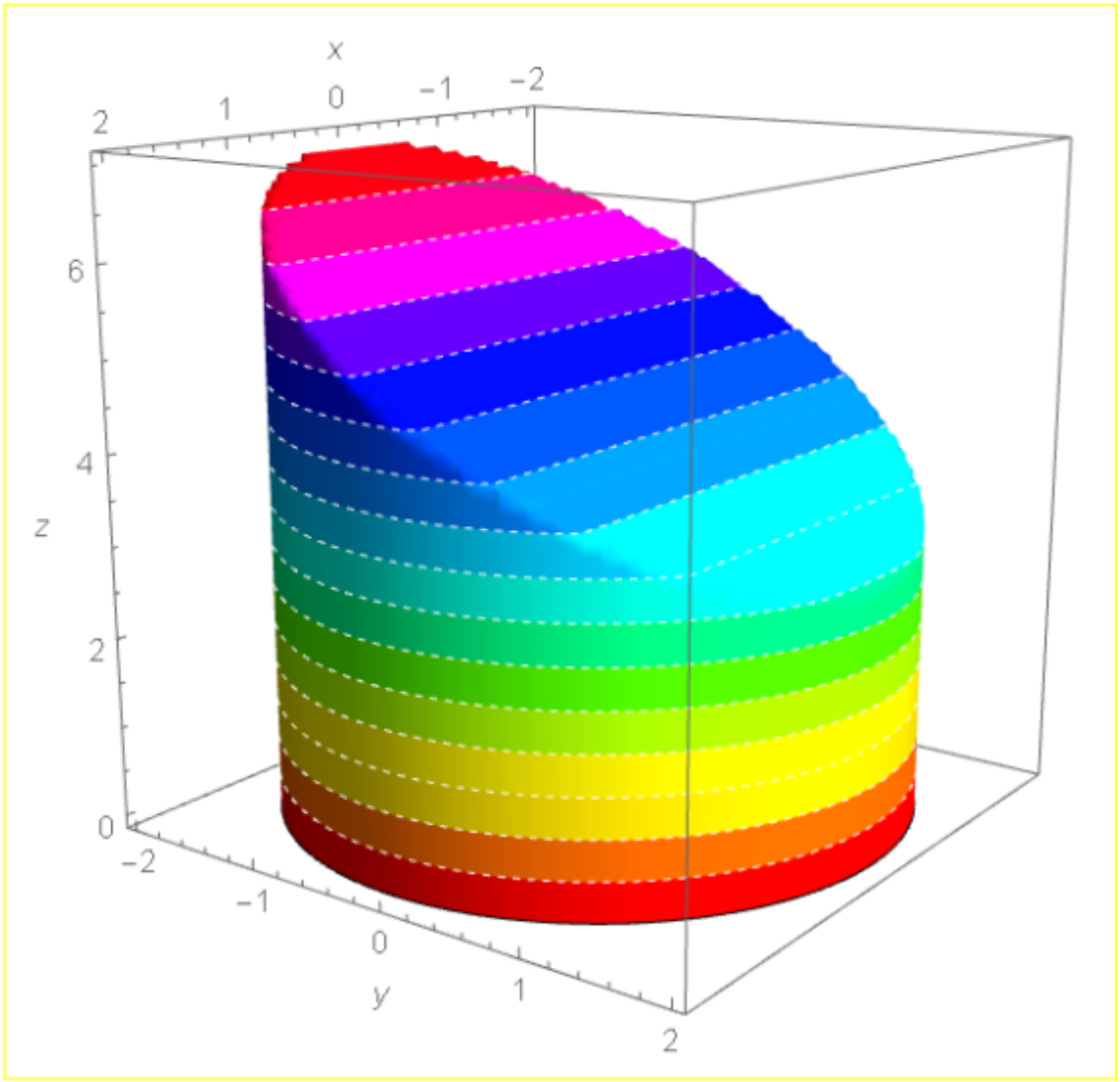
```
grRP1, ColorFunction→Hue, ViewPoint→{2.6,1.9,0.8}
```



Результаты формирования изображения grRP2 – иллюстрация способа раскраски с использованием списка задаваемых цветов (заливки между уровнями делаются указываемыми однородными цветами):

```
Print[Style[
  "grRP2, MeshShading, ViewPoint→{2.6,1.9,0.8}", Blue, 16]]

grRP2 = RegionPlot3D[x^2 + y^2 ≤ 4 && z + y ≤ 5,
  {x, -2, 2}, {y, -2, 2}, {z, 0, 7},
  BaseStyle → {Black, 16}, Lighting → "Neutral",
  AxesLabel → Automatic, PlotPoints → 60,
MeshShading → {Hue[0], Hue[0.07], Hue[0.15],
    Hue[0.16], Hue[0.22], Hue[0.28], Hue[0.42],
    Hue[0.48], Hue[0.53], Hue[0.58], Hue[0.62], Hue[0.66],
    Hue[0.72], Hue[0.80], Hue[0.92], Hue[0.99]},
  MeshFunctions → {#3 &}, MeshStyle → {{Dashed, White}},
  BoxRatios → {4, 4, 4}, ViewPoint → {2.6, 1.9, 0.8},
  ImageSize → 550]
```

```
grRP2, MeshShading, ViewPoint→{2.6,1.9,0.8}
```



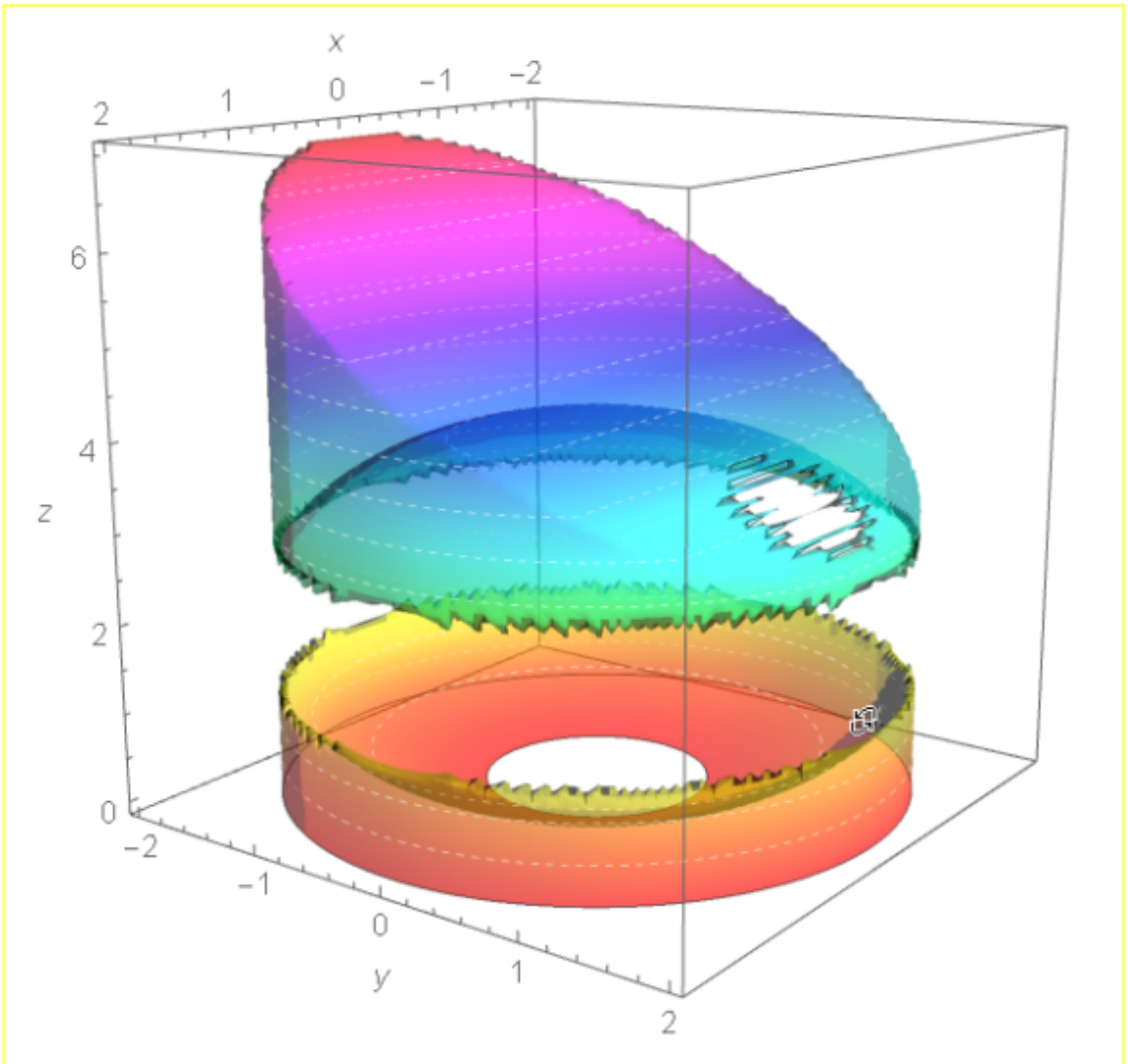
Отметим, что вариант grRP2 в ряде случаев предпочтительней, так как его можно использовать без дополнений показа уровней линиями:

Изображение grRP3 – результат манипуляций с графическими объектами, когда в рассматриваемом цилиндре “вырезали” шар, что обеспечивает выражение $x^2+y^2\leq 4\,\&\&\,z+y\leq 5\,\&\&\,x^2+y^2+(z-2)^2\geq 4.5$, в котором записаны формулы цилиндра $x^2+y^2\leq 4$, отсекающей плоскости $z+y\leq 5$, сферы $x^2+y^2+(z-2)^2\geq 4.5$ диаметра 4.5 с центром в (0,0,2). Для повышения визуального восприятия в этом варианте (градиентная раскраска) при отображении усеченного цилиндра добавлены опции прозрачности (PlotStyle→Opacity[0.4]) и освещения (Lighting) по модели Neutral:

```
Print[Style[
  "grRP3, ColorFunction→Hue, Opacity[0.4]", Blue, 16]]
```

```
grRP3 = RegionPlot3D[x^2 + y^2 ⩽ 4 && z + y ⩽ 5 && x^2 + y^2 + (z - 2)^2 ⩾ 4.5,
  {x, -2, 2}, {y, -2, 2}, {z, 0, 7},
  PlotStyle → Opacity[0.4],
  BaseStyle → {Black, 16}, Lighting → "Neutral",
  AxesLabel → Automatic, PlotPoints → 80,
  ColorFunction → Hue,
  MeshFunctions → {#3 &}, MeshStyle → {{Dashed, White}},
  BoxRatios → {4, 4, 4}, ViewPoint → {2.6, 1.9, 0.8},
  ImageSize → 550]
```

```
grRP3, ColorFunction→Hue, Opacity[0.4]
```

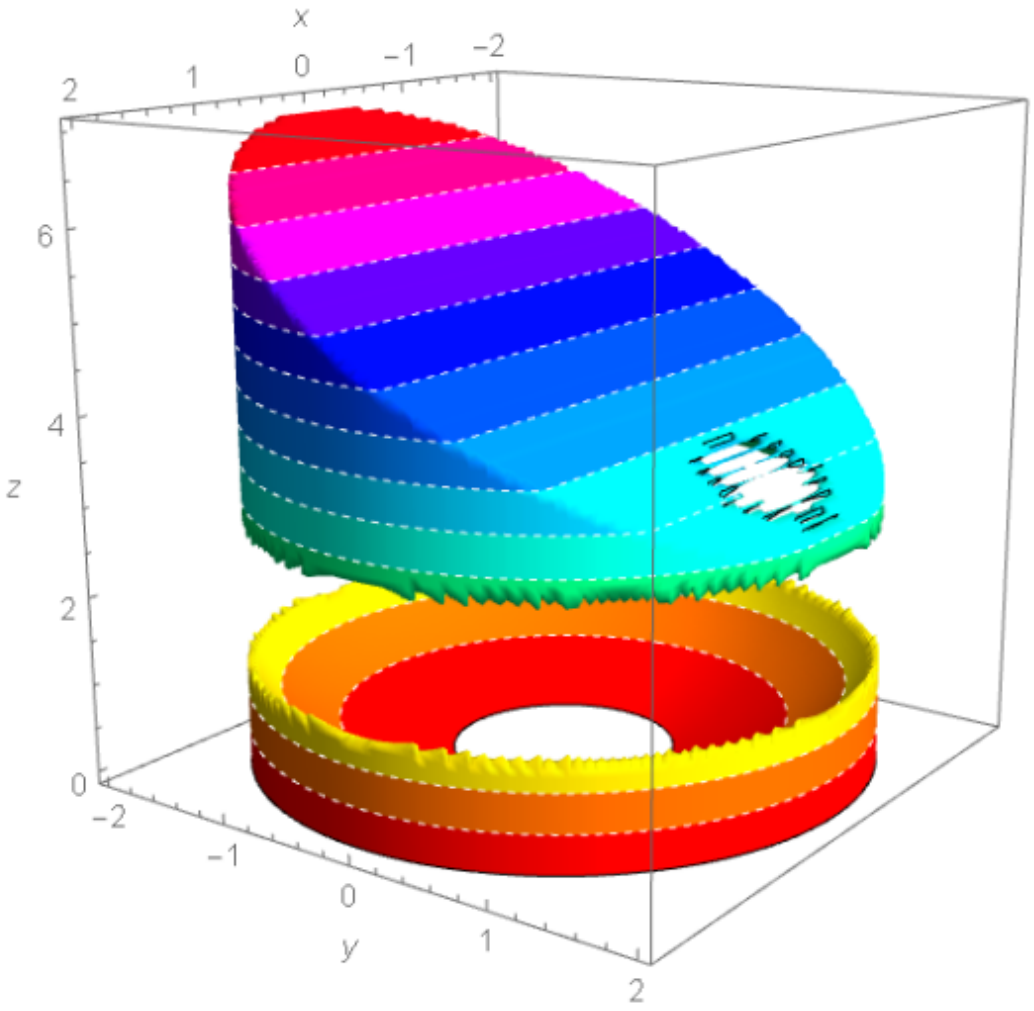


Следует обратить внимание, что даже задание PlotPoints→80 не достаточно для устранения искажений поверхности фигуры после “вырезали” по контуру сферы.

График grRP4 – результат таких же манипуляций, когда цветовая окраска выполнена послойно однородными цветами из списка:

```
Print[Style["grRP4, MeshShading, PlotPoints→90", Blue, 16]]

grRP4 = RegionPlot3D[x^2 + y^2 ⩽ 4 && z + y ⩽ 5 && x^2 + y^2 + (z - 2)^2 ⩾ 4.5,
  {x, -2, 2}, {y, -2, 2}, {z, 0, 7},
  BaseStyle → {Black, 16}, Lighting → "Neutral",
  AxesLabel → Automatic, PlotPoints → 90,
  MeshFunctions → {#3 &}, MeshStyle → {{Dashed, White}},
  MeshShading → {Hue[0], Hue[0.07], Hue[0.15], Hue[0.16],
    Hue[0.22], Hue[0.28], Hue[0.42], Hue[0.48],
    Hue[0.53], Hue[0.58], Hue[0.62], Hue[0.66],
    Hue[0.72], Hue[0.80], Hue[0.92], Hue[0.99]},
  BoxRatios → {4, 4, 4}, ViewPoint → {2.6, 1.9, 0.8},
  ImageSize → 550]
```

```
grRP4, MeshShading, PlotPoints→90
```



Примеры использования ParametricPlot3D

В вступительной части отмечены основные возможности графической функции ParametricPlot3D. Напомним, что она может использоваться для

формирования, оформления, вывода графиков параметрически заданной аналитическими выражениями в трехмерном пространстве поверхности или пространственной кривой. Математически параметрическое задание для описания кривых и поверхностей в пространстве имеет широчайшие возможности, инструментарий этой функции в системе Wolfram *Mathematica* тоже много функциональный. Например, основных описанных в системе помощи опций более 70.

Понятно, что полное изложение такой мощной графической функции потребовало бы большого объема, здесь отметим только несколько часто употребляемых настроек и директив.

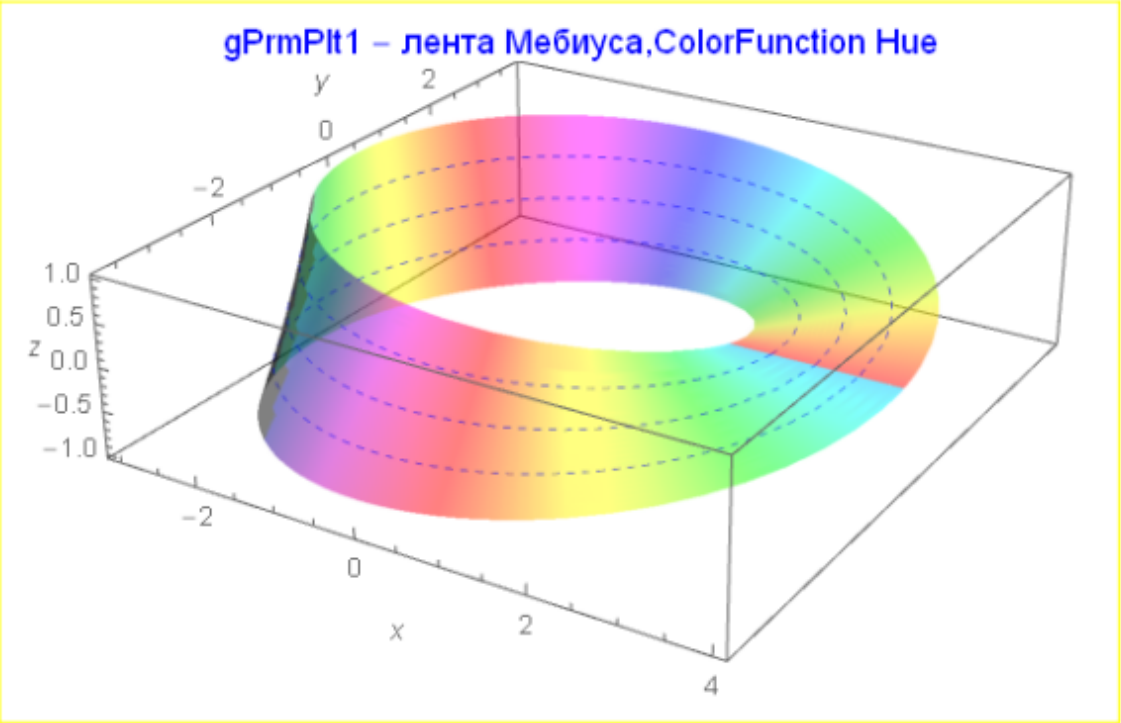
Варианты работы с функцией ParametricPlot3D приведем на примерах визуализации ленты Мебиуса и тора.

Лента Мебиуса – одна из наиболее известных в математике поверхностей, простейшая не ориентируемая, которая является односторонней в трёхмерном пространстве (поверхность Мёбиуса относят к непрерывным топологическим объектам).

Тор – поверхность вращения, получаемая вращением образующей окружности вокруг оси, лежащей в плоскости этой окружности и не пересекающей её. В рассмотренных ниже примерах используется уравнение тора с заданием параметрически. Но также можно использовать непараметрическое уравнение, которое имеет четвёртую степень, другими словами поверхность тора имеет четвёртый порядок.

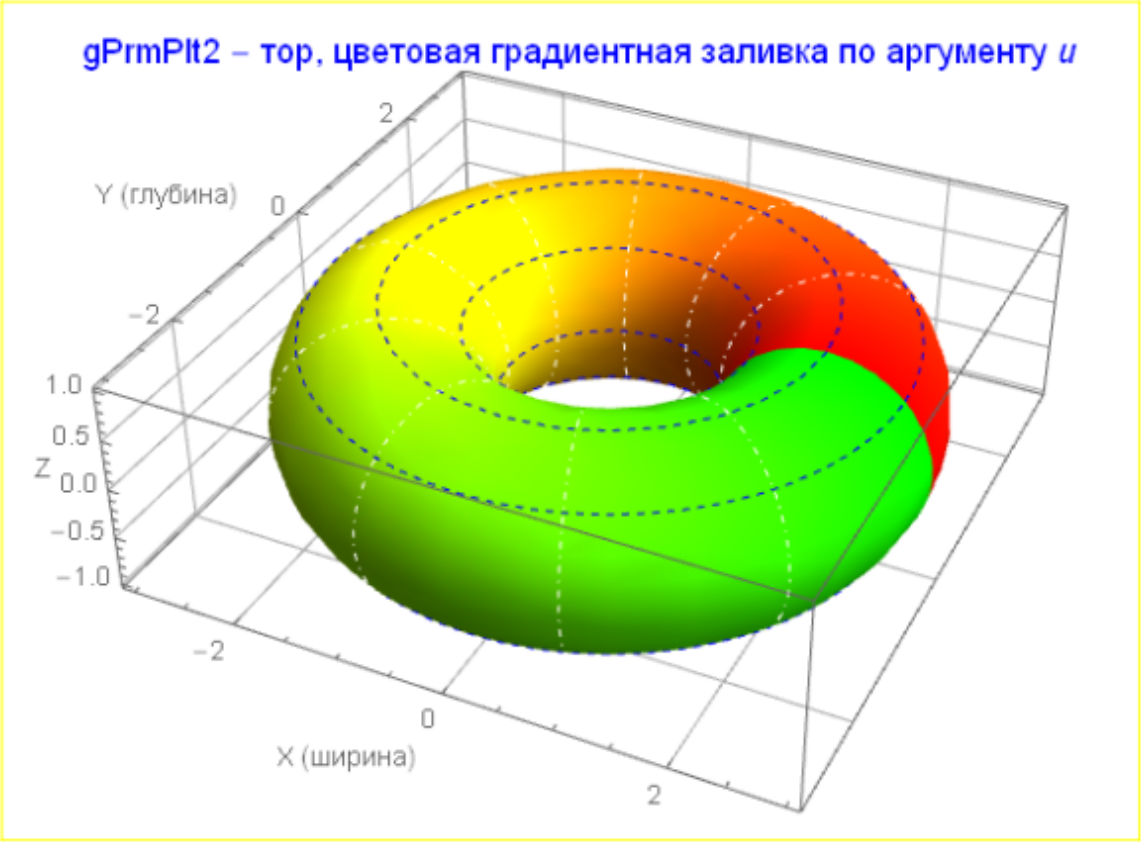
Рисунок gPrmPlt1 иллюстрирует поверхность Мебиуса с цветовой раскраской ColorFunction Hue, пунктирными синими линиями выводятся 3 уровня второго параметра *v*. Следует обратить внимание на необходимость использования опции MaxRecursion -> 6, при значениях ниже 6 заметны искажения в цветах раскраски. Применение опции с заданием прозрачности повышает зрелищность в подтверждении, что поверхность не двухсторонняя:

```
gPrmPlt1 = ParametricPlot3D[{Cos[v] (3 + u Cos[v / 2]),
     Sin[v] (3 + u Cos[v / 2]), u Sin[v / 2]},
   {u, -1, 1}, {v, 0, 2 Pi},
   AxesLabel → {x, y, z}, BaseStyle → {Black, 14},
   MaxRecursion → 6,
   Mesh → 3, MeshStyle → {Blue, Dashed},
   MeshFunctions → Function[{x, y, z, u, v}, u],
   PlotStyle → Directive[Yellow, Opacity[0.5]],
   ColorFunction → Function[{x, y, z, u, v}, Hue[v / (Pi / 8)]]];
Show[gPrmPlt1,
 PlotLabel → Style
   ["gPrmPlt1 - лента Мебиуса,ColorFunction Hue", 18, Blue],
 ViewPoint → {1.6, -2.6, 1.4},
 ImageSize → 550]
```



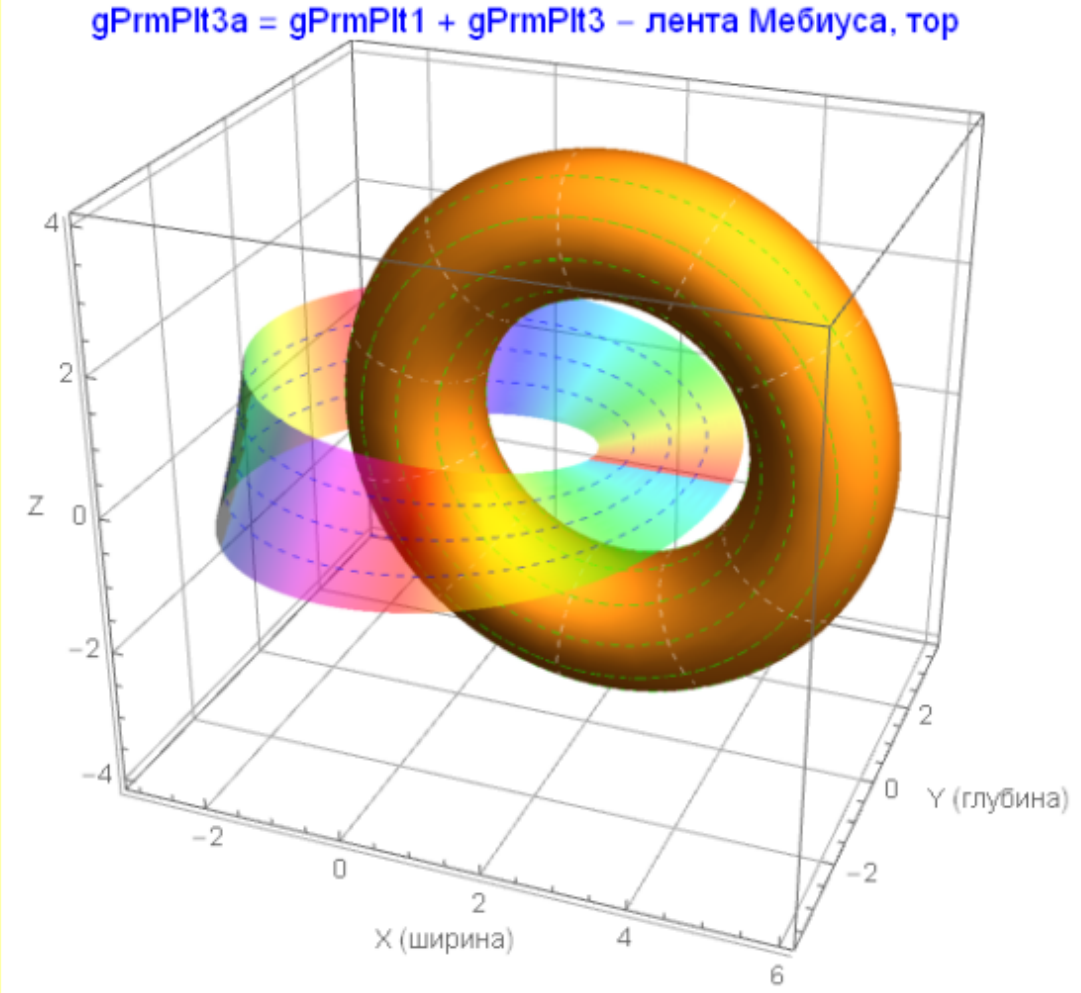
Вариант оформления изображения тора приведен на рисунке gPrmPlt2, в отличие от предыдущих иллюстраций добавлены сетки на гранях (опция FaceGrids, задание в соответствующих позициях 1 или -1 обеспечивает, на какой грани выводится сетка); направления осей сопровождаются текстовыми пояснениями ({“X (ширина)”, “Y (глубина)”, “Z”}); поверхность окрашена по возрастанию аргумента *u* согласно градиентной схеме (ColorFunction->Function[{x,y,z,u,v}) в части цветового круга от красного до зелёного (Hue[u/(6*Pi)]); применение опции нормировать аргумент функции оцвечивания ColorFunctionScaling->False обязательно, иначе выводится однородная окраска поверхности; пунктирными синими ({Blue,Dashed}) и штрихпунктирными белыми ({White,DotDashed}) показаны сеточные уровни, число которых задается в списке {9,7}:

```
ParametricPlot3D[
 {(2 + Cos[v]) Cos[u], (2 + Cos[v]) Sin[u], Sin[v]},
 {u, 0, 2 Pi}, {v, 0, 2 Pi},
 BaseStyle → {Black, 14},
 AxesLabel → {"X (ширина)", "Y (глубина)", "Z"},
 FaceGrids → {{-1, 0, 0}, {0, 1, 0}, {0, 0, -1}},
 Mesh → {9, 7},
 MeshStyle → {{White, DotDashed}, {Blue, Dashed}},
 ColorFunction → Function[{x, y, z, u, v}, Hue[u / (6 Pi)]],
 ColorFunctionScaling → False,
 PlotLabel → Style[
   "gPrmPlt2 – тор, цветовая градиентная заливка по аргументу u",
   18, Blue],
 ViewPoint → {1.2, -2.5, 1.9}, ImageSize → 550]
```



Пример синтеза двух приведенных поверхностей иллюстрирует рисунок gPrmPlt3a:

```
gPrmPlt1d = Show[gPrmPlt1, PlotRange → {-4, 4}];
gPrmPlt3 = ParametricPlot3D[
  {2 + (3 + Cos[v]) * Cos[2 * Pi * u],
     1 + Sin[v], (3 + Cos[v]) * Sin[2 * Pi * u]},
    {u, -1, 1}, {v, 0, 2 * Pi},
    MaxRecursion → 6,
    BaseStyle → {Black, 14},
    Mesh → 10, MeshStyle → {{White, Dashed}, {Green, Dashed}}
    (* ColorFunction→"ThermometerColors",*)
    (* Несколько разных зрелищных цветовых схем:
      BrightBands,  BlueGreenYellow,
    CherryTones, MintColors, Pastel *)];
gPrmPlt3a = Show[gPrmPlt1d, gPrmPlt3,
  FaceGrids → {{-1, 0, 0}, {0, 1, 0}, {0, 0, -1}},
  AxesLabel → {"X (ширина)", "Y (глубина)", "Z"},
  PlotLabel → Style
    ["gPrmPlt3a = gPrmPlt1 + gPrmPlt3 – лента Мебиуса, тор",
    18, Blue],
  ViewPoint → {1.1, -2.8, 1.5}, ImageSize → 550]
```



Примеры использования RevolutionPlot3D

Напоминания:

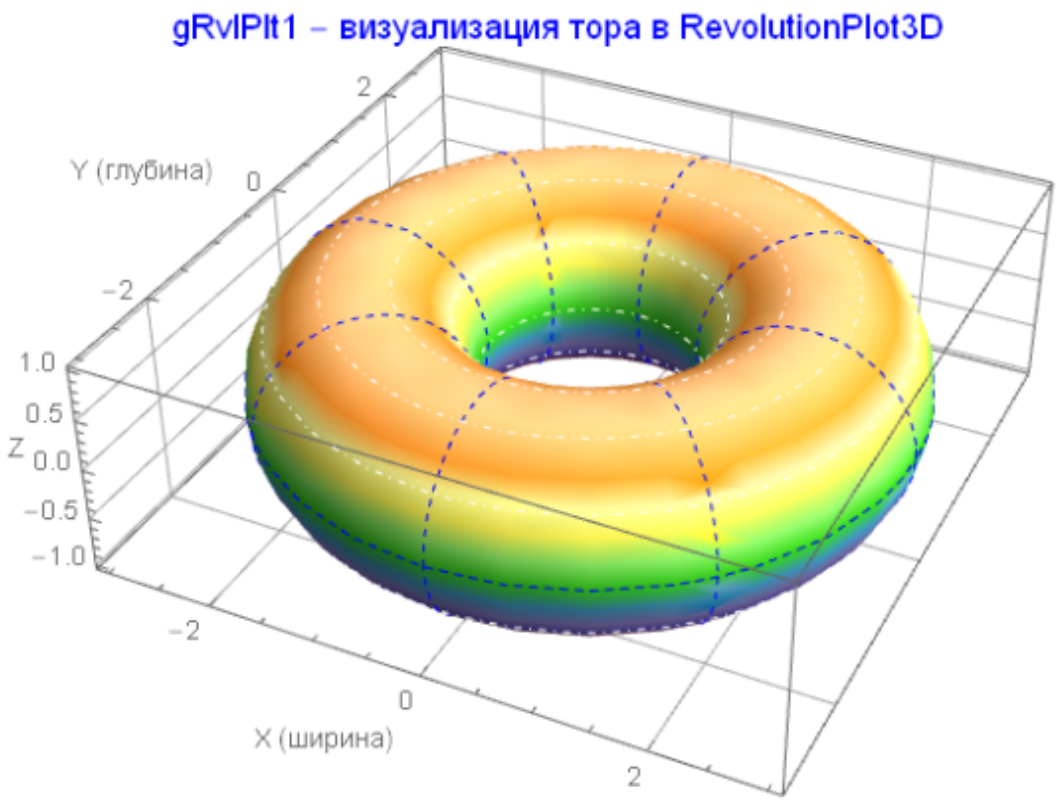
Тор – поверхность вращения, получаемая вращением образующей окружности вокруг оси, лежащей в плоскости этой окружности и не пересекающей её. ...

- RevolutionPlot3D – график поверхности вращения (трехмерные объекты, полученные вращением кривых) с использованием заданного выражения;

Пример получения и оформления изображения тора, как на иллюстрации gPrmPlt2, но здесь с иным оформлением и с использованием графической функции RevolutionPlot3D, которая обеспечивает формирование поверхности вращения для заданного выражения – график gRvlPlt1:

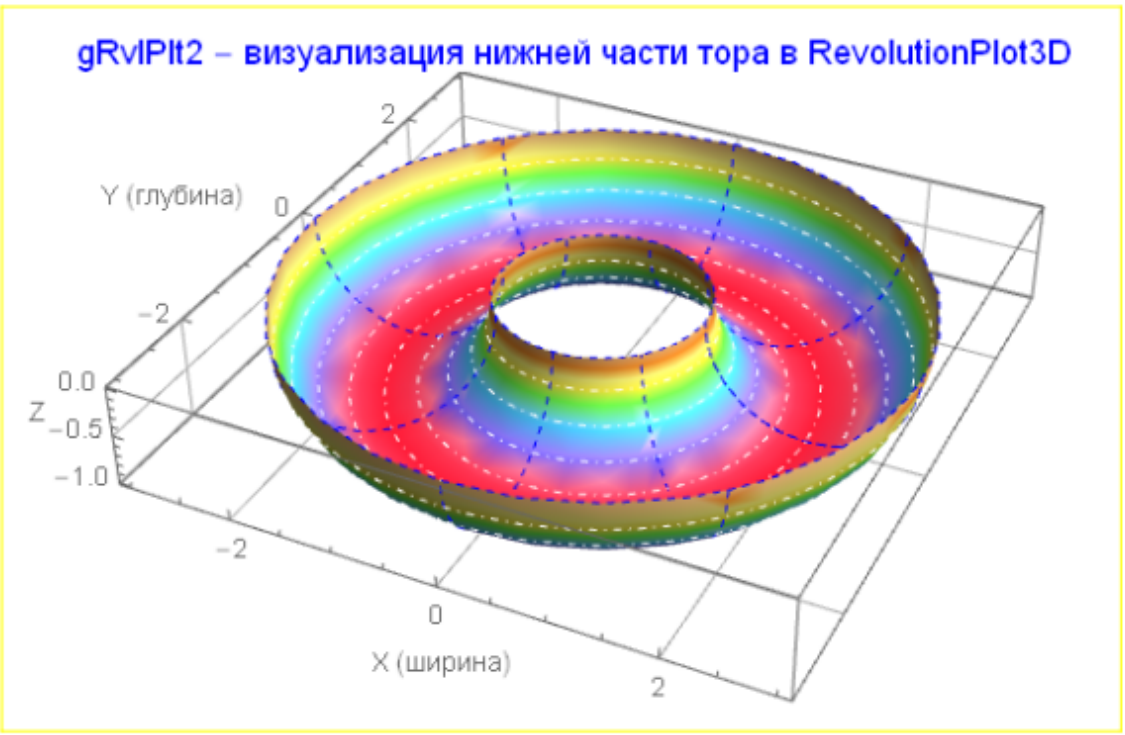
```
gRvlPlt1 = RevolutionPlot3D[{2 + Cos[t], Sin[t]}, {t, 0, 2 Pi},
  BaseStyle → {Black, 14},
```

```
AxesLabel → {"X (ширина)", "Y (глубина)", "Z"},
 FaceGrids → {{-1, 0, 0}, {0, 1, 0}, {0, 0, -1}},
 Mesh → {9, 7},
 MeshStyle → {{White, DotDashed}, {Blue, Dashed}},
 ColorFunction → "BrightBands",
 PlotLabel → Style[
   "gRvlPlt1 - визуализация тора в RevolutionPlot3D", 18, Blue],
 ViewPoint → {1.2, -2.5, 1.9}, ImageSize → 550]
```



Вариант вывода конкретной части изображения (в примере нижней) иллюстрирует gPrmPlt2:

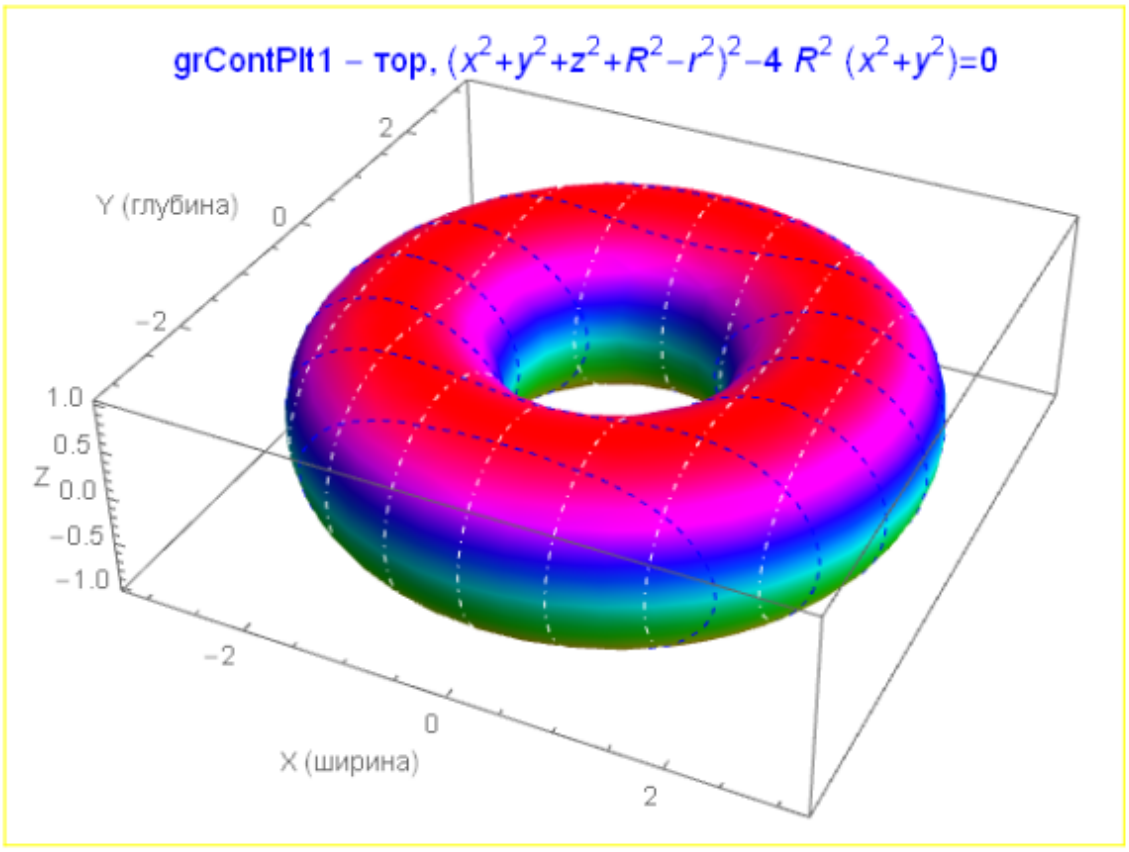
```
gRvlPlt2 = RevolutionPlot3D[{2 + Cos[t], Sin[t]}, {t, 0, 2 Pi},
  BaseStyle → {Black, 14},
  AxesLabel → {"X (ширина)", "Y (глубина)", "Z"},
  FaceGrids → {{-1, 0, 0}, {0, 1, 0}, {0, 0, -1}},
  MaxRecursion → 11,
  Mesh → {9, 7},
  MeshStyle → {{White, DotDashed}, {Blue, Dashed}},
  RegionFunction -> (#4 > Pi &),
  ColorFunction → "BrightBands",
  PlotLabel → Style[
    "gRvlPlt2 - визуализация нижней
      части тора в RevolutionPlot3D", 18, Blue],
  ViewPoint → {1.2, -2.5, 1.9}, ImageSize → 550]
```



Примеры визуализации поверхностей задаваемых неявно

Вопрос обсуждался выше, здесь приводим один из вариантов оформления выводимого тора, когда используется неявное задание, как решение уравнения $\left(x^2+y^2+z^2+R^2-r^2\right)^2-4*R^2\left(x^2+y^2\right)=0$:

```
R = 2; r = 1;
xMin := -Pi; xMax := Pi; yMin := -Pi; yMax := Pi;
zScale = 5 / 8 * Pi;
rBoxRatios = {xMax - xMin, yMax - yMin, zScale};

grContPlt1 = ContourPlot3D[
   (x^2 + y^2 + z^2 + R^2 - r^2) ^2 - 4 * R^2 * (x^2 + y^2) == 0,
   {x, xMin, xMax}, {y, yMin, yMax},
   {z, -1, 1}, BaseStyle → {Black, 14},
   AxesLabel → {"X (ширина)", "Y (глубина)", "Z"},
   BoxRatios → rBoxRatios,
   Mesh → 7, MeshFunctions → {#1 &, #2 &},
   MeshStyle → {{White, DotDashed}, {Blue, Dashed}},
   ColorFunction → Function[{x, y, z}, Hue[(z + 1) / 2]],
   ColorFunctionScaling → False,
   PlotLabel → Style[
     "grContPlt1 - тор, (x^2+y^2+z^2+R^2-r^2)^2-4 R^2 (x^2+y^2)=0",
     18, Blue],
   ViewPoint → {1.2, -2.5, 1.9}, ImageSize → 550]
```


grContPlt1 – тор, $(x^2+y^2+z^2+R^2-r^2)^2-4\ R^2\ (x^2+y^2)=0$
Y (глубина)
2
0
−2
1.0
0.5
Z 0.0
−0.5
−1.0
−2
0
2
X (ширина)

Рекомендуемая литература

1. *Таранчук, В.Б.* **Введение в графику системы *Mathematica*** : учеб. материалы для студентов фак. прикладной математики и информатики / В. Б. Таранчук. - Минск : БГУ, 2017. - 53 с. (http://elib.bsu.by/handle/123456789/186348)

2. *Таранчук, В.Б.* **Двумерная графика системы Mathematica. Визуализация функций :** учеб. материалы для студентов фак. прикладной математики и информатики / В.Б. Таранчук. - Минск : БГУ, 2022. - 52 с. (https://elib.bsu.by/handle/123456789/293313)

3. *Таранчук, В.Б.* **Одномерная графика системы *Mathematica*. Визуализация функций** : учеб. материалы для студентов фак. прикладной математики и информатики / В. Б. Таранчук. – Минск : БГУ, 2019. – 52 с. (http://www.elib.bsu.by/handle/123456789/215369)

СОДЕРЖАНИЕ

Учебное издание

Таранчук Валерий Борисович

WOLFRAM *MATHEMATICA*. ПРОГРАММИРОВАНИЕ ИНТЕРАКТИВНОЙ 3D ГРАФИКИ

Учебные материалы для студентов факультета прикладной математики и информатики

В авторской редакции

Ответственный за выпуск *В. Б. Таранчук*

Подписано в печать 19.01.2024. Формат 60×84/16. Бумага офсетная.
Усл. печ. л. 3,02. Уч.-изд. л. 2,96. Тираж 50 экз. Заказ

Белорусский государственный университет.
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 1/270 от 03.04.2014.
Пр. Независимости, 4, 220030, Минск.

Отпечатано на копировально-множительной технике
факультета прикладной математики и информатики
Белорусского государственного университета.
Пр. Независимости, 4, 220030, Минск.