

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

В. Б. Таранчук

**ДВУМЕРНАЯ ГРАФИКА
СИСТЕМЫ *MATHEMATICA*.
ВИЗУАЛИЗАЦИЯ ФУНКЦИЙ**

**Учебные материалы
для студентов факультета
прикладной математики и информатики**

**МИНСК
2022**

УДК 519.67(075.8)
ББК 22.19я73-1
Т19

Рекомендовано советом
факультета прикладной математики и информатики БГУ
22 ноября 2022 г., протокол № 4

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

Таранчук, В. Б.
Т19 Двумерная графика системы *Mathematica*. Визуализация функций : учеб. материалы для студентов фак. прикладной математики и информатики / В. Б. Таранчук. – Минск : БГУ, 2022. – 52 с.

Изложены основные функции, опции и директивы, используемые при визуализации в системе *Mathematica* 2D карт изолиний и плотности. Примерами иллюстрируются возможности формирования и оформлении иллюстраций. Приведены практические задания, включены вопросы и задачи для самостоятельного изучения и выполнения.

Предназначено для студентов факультета прикладной математики и информатики.

УДК 519.67(075.8)
ББК 22.19я73-1

© Таранчук В. Б., 2022
© БГУ, 2022

ПРЕДИСЛОВИЕ

В учебных материалах изложены теоретические и методические вопросы, примеры решения возникающих в практике вычислительных экспериментов, интеллектуального анализа данных задач визуализации функций с использованием инструментов системы компьютерной алгебры *Wolfram Mathematica* ContourPlot и DensityPlot. Названные темы (Изолинии, Карты плотности) 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 изложенных материалах лекций и заданиях практических занятий, в процессе преподавания акценты сделаны на:

- освоение обучаемыми методов функционального программирования и инструментов создания в *Wolfram Mathematica* интерактивных программных модулей с возможностями приближенных и точных символьных вычислений; правил и приёмов работы с интерактивной справочной системой; средств визуализации функций и данных; экспорта формируемых динамических изображений;
- приобретение студентами навыков программирования и отладки блокнотов системы *Mathematica*, выполнения символьных вычислений, интерактивного графического представления результатов математических преобразований и расчётов;
- контроль знаний и навыков средствами компьютерного тестирования (на каждой лекции – 2-5 контрольных вопросов или кратких заданий, на каждом практическом занятии – 5-6 вопросов и 7-9 заданий функционального программирования).

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



Двумерная графика системы *Mathematica*. Визуализация функций

Таранчук Валерий Борисович

БГУ,

факультет прикладной математики и информатики

Учебные материалы, инструкции и рекомендации пользователям системы
компьютерной алгебры *Mathematica*, обучающие примеры и упражнения
(оригинал документа предоставляется студентам в формате PDF)

План лекции, лабораторного занятия “Изолинии” >>>>

- ✓ Общее о 2D визуализации функций
- ✓ Изолинии. Определения
- ✓ Аналитическая функция, используемая в примерах
- ✓ Функция ContourPlot. Основные опции
- ✓ Изолинии. Вывод по умолчанию, масштаб по осям
- ✓ Функция ContourPlot. Примеры задания атрибутов изолиний
- ✓ ContourPlot. Варианты режима цветного вывода
- ✓ Оформление контурной графики индивидуальной цветовой схемой
- ✓ Изолинии, варианты вывода легенды (принятой шкалы цветов)
- ✓ Карты изолиний, примеры специальных настроек
- ✓ Тестовые задания для самоконтроля

Общее о 2D визуализации функций

Напомним, что в материалах лекции и практического занятия 10 [1] изложена принятая классификация и обозначены следующие типы (категории) графических объектов системы *Mathematica*:

- Графики аналитически задаваемых функций одной переменной
- Визуализация массивов одномерных данных
- Гистограммы, столбиковые, круговые, секторные диаграммы
- Специальные функции графики пакетов-приложений
- 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.

К категории “Средства 2D графики аналитически задаваемых функций” отнесены:

- ContourPlot – контурный график на плоскости, который показывает линии равного уровня (изолинии); контурные графики (или графики в горизонталях) наиболее часто используются в топографии для представления на плоскости объемного рельефа местности;
- DensityPlot – плотностный график (карта зон), значение функции в каждой точке отображается при помощи окрашивания в определённый цвет, можно задавать интервалы и соответствующие им цвета, можно применять плавные цветовые переходы (градиентная заливка);
- RegionPlot – визуализация геометрической фигуры на плоскости;
- ParametricPlot – используется для рисования на плоскости графиков функций, заданных параметрически;
- StreamPlot – линии тока на плоскости (диаграмма потоков); для построения линий тока используются тангенциальные компоненты векторных переменных; в гидромеханике – линия, направление касательной к которой в каждой точке совпадает с направлением скорости частицы жидкости в этой точке (в каждый момент времени частица движется вдоль линии тока);
- StreamDensityPlot – линии тока на плоскости с фоном плотности функции (плотностная диаграмма потоков);
- VectorPlot – векторное поле на плоскости, графическое изображение при помощи направленных отрезков – векторов;
- VectorDensityPlot – векторно-плотностная диаграмма;
- LineIntegralConvolutionPlot – диаграмма потоков над изображением по LIC-методу (Line Integral Convolution – линейная интегральная свертка).

Общие правила и основные опции для функций графики изложены в материалах лекции и практического занятия 12 [2]. Напомним, что система Wolfram *Mathematica* содержит много различных инструментов и настроек для содержательной и зрелищной визуализации. Модули графики обеспечивают высококачественное статическое или динамическое представление функций, данных, диаграмм, изображений или аннотаций, определяя баланс между скоростью вычислений, вывода и визуальным соответствием виду объектов.

При формировании иллюстраций *Mathematica* автоматизирует с подбором оптимальных вариантов параметры построения графика, обеспечивает корректное отображение деталей поведения функций на всех участках (фокусировку, разбиение на регионы), выбор масштаба; для пространственных изображений (3D-графики являются интерактивными) – как правило, изображения по умолчанию (без уточняющих опций пользователя) выводятся наглядно и эстетично с хорошими установками ракурса обзора и параметров освещения, причем, это при возможности изменять параметры камеры, освещения в реальном времени. Используемые

по умолчанию настройки графики тщательно подобраны, что наилучшим образом подходит для подавляющего большинства случаев. Но дополнительно, *Mathematica* позволяет делать настройки визуализации в соответствии с потребностями пользователя, система предоставляет сотни опций, позволяющих контролировать каждый аспект отображения, включая размер рисунка, метки осей, отображение сетки, разбиение, заливку, многое другое. Если нужно улучшить внешний вид графика, если требуется получить наилучшую графику для решения конкретной специальной задачи, надо уточнять настройки вывода, выбирать не типовые варианты, экспериментировать, последовательно пробуя различные параметры (директивы) для тех или иных опций.

Рассмотрение инструментов 2D-графики системы *Mathematica* начнем с описания возможностей и иллюстрации вариантов настроек формирования и вывода изображений функцией *ContourPlot*, которая показывает линии равного уровня – изолинии.

Изолинии. Определения

Контурные графики или изолинии, или линии равных высот являются линиями пересечения трехмерной поверхности $z=f(x,y)$ с секущими горизонтальными плоскостями $z=const$, расположенными параллельно друг под или над другом.

В большом энциклопедическом словаре даётся определение: **ИЗОЛИНИИ** (от изо...) – линии равного значения какой-либо величины на географической карте, вертикальном разрезе или графике. Изолинии дают характеристику непрерывных явлений в некоторый период или момент времени (напр., изобары, изобаты). Применяются при картографировании природных и социально-экономических явлений; могут быть использованы для получения их количественной характеристики и для анализа корреляционных связей между ними.

Следует отметить другие определения, пояснения этого термина:

Изолинии – линии равного значения какой-либо величины в её распределении на поверхности, в частности на плоскости (на географической карте, вертикальном разрезе или графике). Изолинии отражают непрерывное изменение исследуемой величины в зависимости от двух других переменных, например от географической широты и долготы на картах.

Изолиния – линия, соединяющая на карте или диаграмме точки с одинаковыми количественными показателями (давления, температуры, влажности и т.п.). План в изолиниях изображает в виде топографической поверхности характер и степень изменчивости исследуемой величины.

Изолинии – линии равного значения какой-либо величины в её распределении на поверхности, в частности на плоскости (на

географической карте, вертикальном разрезе или графике). Изолинии отражают непрерывное изменение исследуемой величины в зависимости от двух других переменных, например от географической широты и долготы на картах.

Изолиния – линия, соединяющая на карте или диаграмме точки с одинаковыми количественными показателями (давления, температуры, влажности и т.п.).

Изолиния, или линия уровня (функции) – условное обозначение на карте, чертеже, схеме или графике, представляющее собой линию, в каждой точке которой измеряемая величина сохраняет одинаковое значение. Изолинии – способ представления скалярной функции от двух переменных на плоскости.

Изолинии на картах наиболее широко используются для характеристики значений непрерывных и постепенно изменяющихся в пространстве величин (например, температуры воздуха) – распределений по площади, но также выполняют значительно более разнообразные функции. С помощью изолиний на картах показывают изменение количественных характеристик явлений во времени (например, вековые изменения составляющих земного магнетизма), параметры изменения контролируемых процессов (например, уровень концентрации загрязняющих веществ), время наступления каких-либо явлений (например, сроки первых осенних заморозков), продолжительность явлений (например, число дней со снежным покровом), уровень каких-либо физических величин (например, магнитного склонения), повторяемость или вероятность явлений (например, повторяемость гроз).

Для метеорологических и климатических показателей – это наиболее часто изолинии температуры, влажности воздуха и почвы, суммы активных температур, продолжительности беззаморозкового периода, абсолютных высот над уровнем моря и других абиотических факторов среды (субстрат и его состав, влажность, температура, свет и другие виды излучений в природе). Как правило, изолинии дают характеристику природных явлений в определенный период или момент времени. Применительно к потребности ряда производств, для оценки конкретного элемента климата, слово “изолинии” заменяется изотермой, изоатмой (испаряемость за определенный промежуток времени), изогией (количество осадков за какие-либо период) и др.

Изолинии, примеры характеристик конкретных распределений: изобары, изобаты, изогипсы, изопакеты, изотермы. В частности, изогипсы (горизонталы) – высота земной поверхности над уровнем моря, изобаты – изолинии глубин водного бассейна на карте, изогипсы – линии на карте, соединяющие точки с равными высотами земной поверхности над уровнем моря (передают плановые очертания форм рельефа суши), изопакеты – линии на карте, проведенные через точки, соответствующие

одинаковой мощности пласта горной породы, изотермы – линии, изображающие на термодинамической диаграмме один из изопроцессов – изотермический (процесс, происходящий при неизменной температуре), другое определение – кривые линии на карте, соединяющие точки с равными значениями температуры.

Изолинии высоты можно рассматривать, как контуры разреза поверхности горизонтальными плоскостями (поэтому для изолиний высоты часто применяется название “горизонтали”). Описание поверхности изолиниями высоты часто используется, например в картографии. Для описания поверхности можно использовать не только изолинии высоты, но и другие, например, x - или y -изолинии.

Аналитическая функция, используемая в примерах

В настоящем изложении в примерах визуализации, описаниях возможностей формирования и инструментов Wolfram *Mathematica* категории “Средства 2D графики аналитически задаваемых функций” используется выражение $zXY(x,y)$, 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 И.1 и И.2 (как они формируется пояснено ниже)

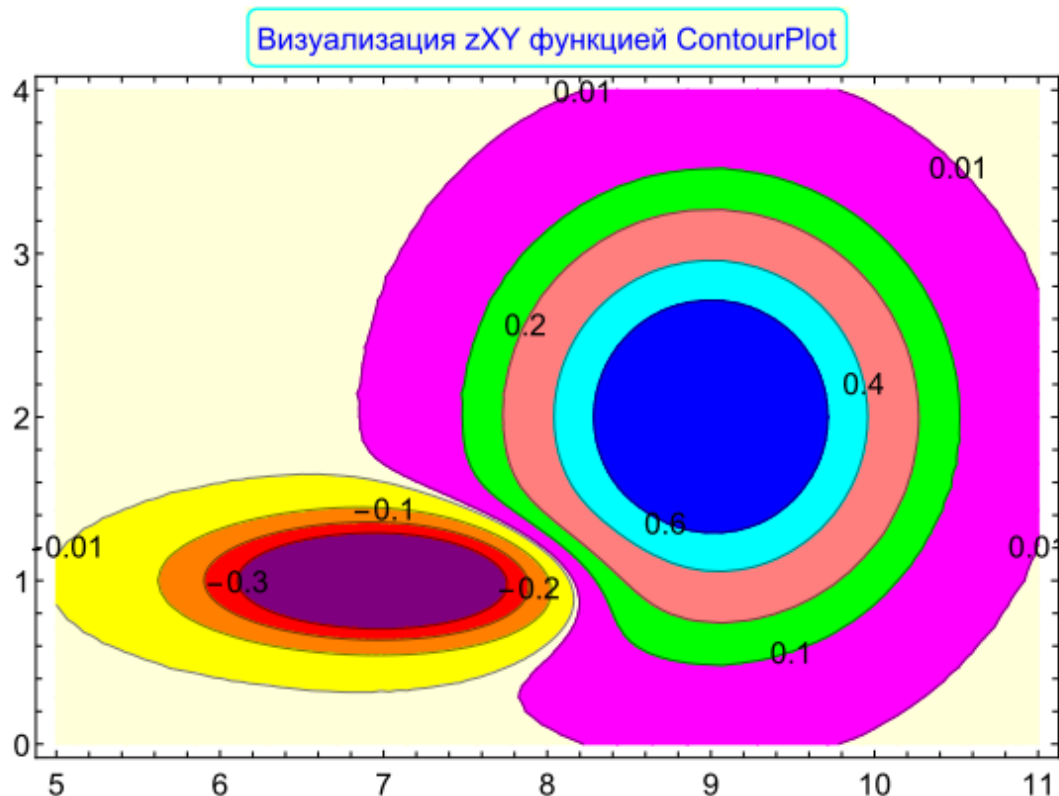


Рис. И.1. Изолинии поверхности, определенной выражением $zXY(x,y)$

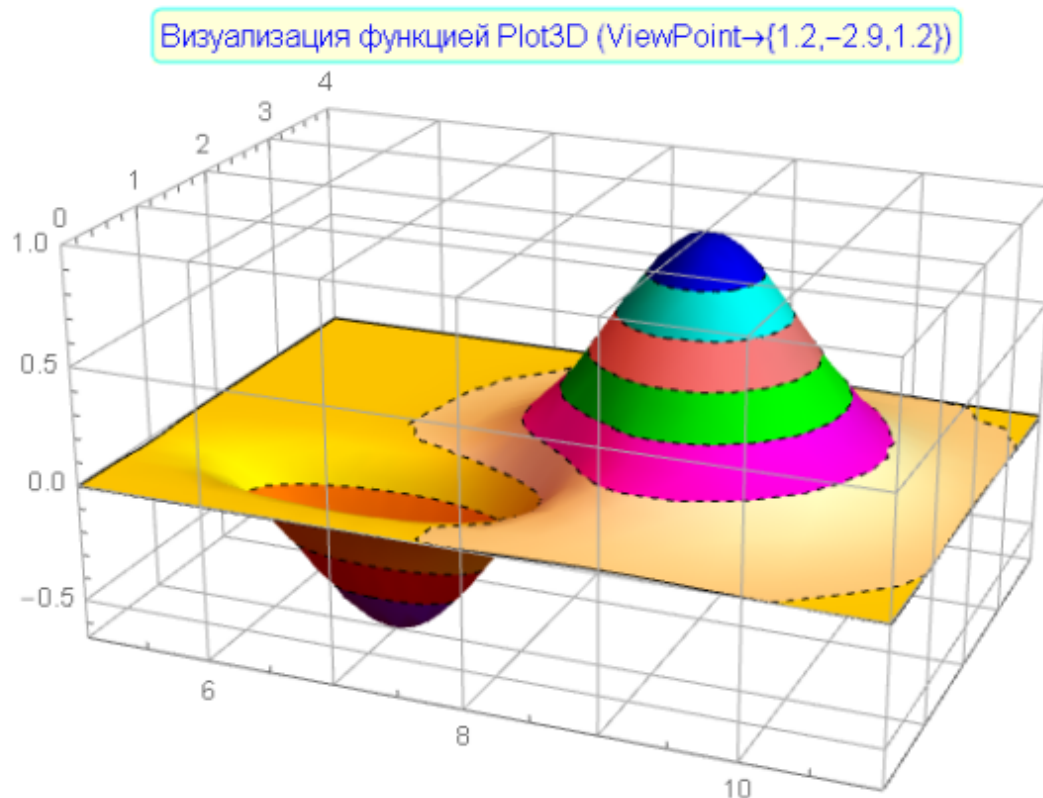


Рис. И.2. Вид поверхности, с цветовой окраской, как в карте изолиний

Функция ContourPlot. Основные опции

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 [2] 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 2D, поясняются и иллюстрируются ниже.

Изолинии. Вывод по умолчанию, масштаб по осям

Как отмечено выше, функция системы *Mathematica* *ContourPlot[f,{x,xmin,xmax},{y,ymin,ymax}]* строит контурный график поверхности $f(x,y)$. Ниже приведены код и результаты его выполнения, иллюстрирующие вывод с установками по умолчанию в варианте прямоугольной области определения с границами $x_{Min}=5$, $x_{Max}=11$, $y_{Min}=0$; $y_{Max}=4$.

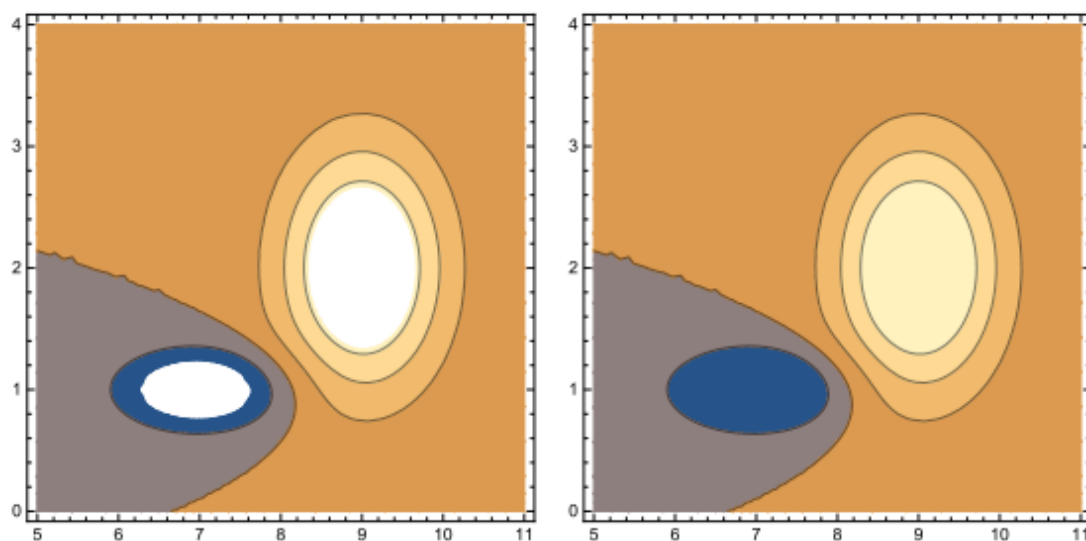
Следует обратить внимание, что изолинии в вариантах *grIzl0*, *grIzl0d*, *grIzl0h* отображаются в квадрат (обеспечивает опция *AspectRatio*, установка в которой по умолчанию определяется значением 1), а не в прямоугольник, также частично утрачены зоны минимальных и максимальных значений в распределении на площади в левом нижнем и в правом верхнем квадрантах на рисунке *grIzl0*. На правом графике *grIzl0d* эта потеря отменена опцией *ClippingStyle* (стиль отсечения), вариант использования *PlotRange*→*Full* приведен в следующей теме. Для пояснения “частично утрачены зоны минимальных и максимальных значений” график *rlzl0h* иллюстрирует, какие части на карте в начальном варианте отсечены (назначены заливки красным и зеленым):

```
xMin := 5; xMax := 11;
yMin := 0; yMax := 4;

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

grIzl0 = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ImageSize → 275];
grIzl0d =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic,
    ImageSize → 275];

Row[{grIzl0, grIzl0d}, Spacer[10]]
```



```
grIzl0h =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → {Red, Green},
    ImageSize → 275];
Row[{grIzl0d, grIzl0h}, Spacer[10]]
```

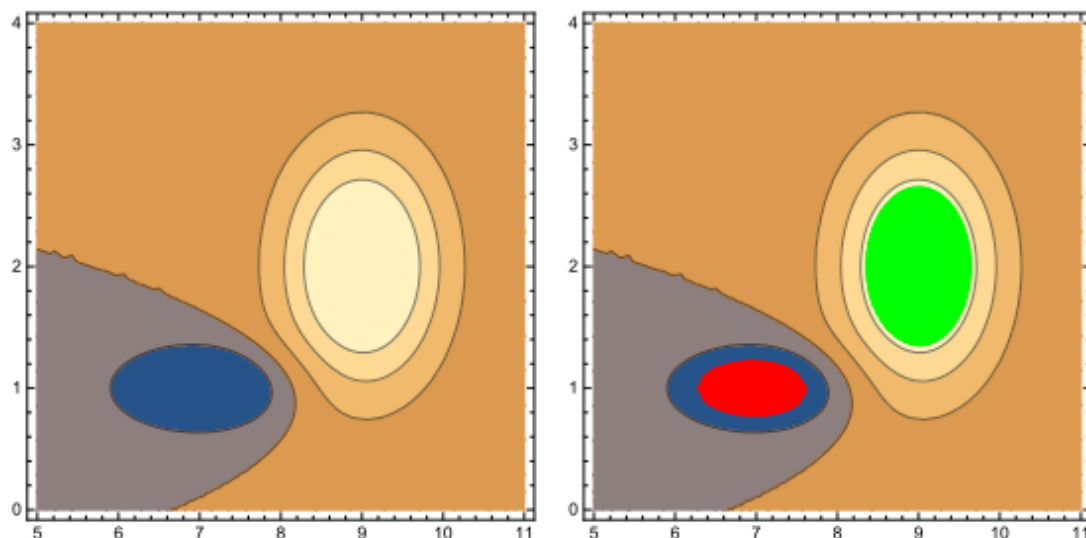
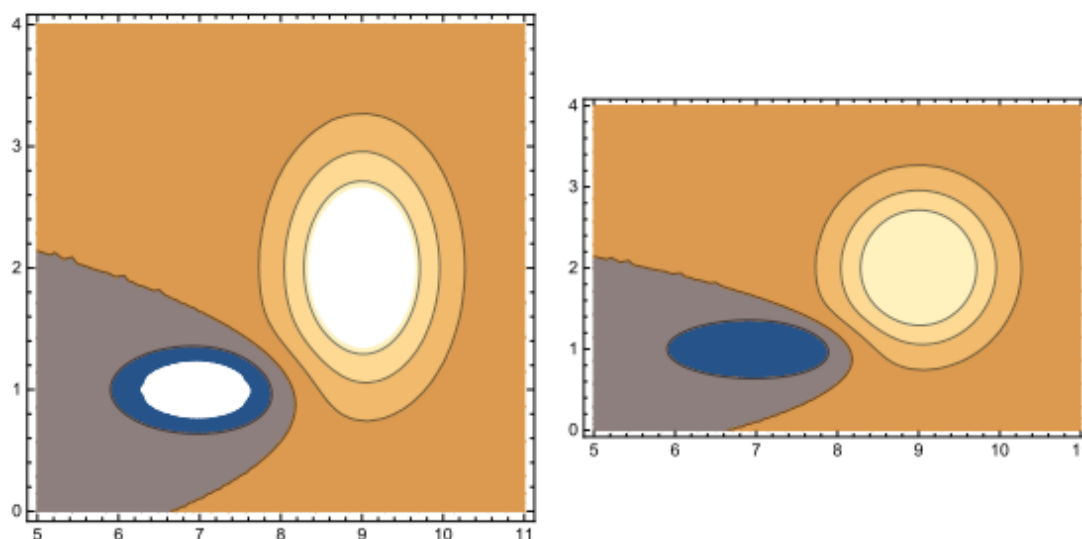


График grIzl1 иллюстрирует эффект применения опции AspectRatio, когда получаем пропорциональное соотношение размеров; соответственно, видим, что в правом верхнем квадранте изолинии круги, т.к. в используемом выражении задано $-(x-9)^2 - (y-2)^2$:

```
aspRat = (yMax - yMin) / (xMax - xMin);
grIzl1 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic,
    AspectRatio → aspRat, ImageSize → 275];
Row[{grIzl0, grIzl1}, Spacer[10]]
```



Функция ContourPlot. Примеры задания атрибутов изолиний

Дополнительно отметим, что возможности графической функции ContourPlot, используемые опции можно вывести командой Options[ContourPlot]. Ниже приведены кратко поясняемые примеры часто используемых опций, при этом, для сопоставления графиков даются предыдущий (или эталонный) слева, а уточненный справа.

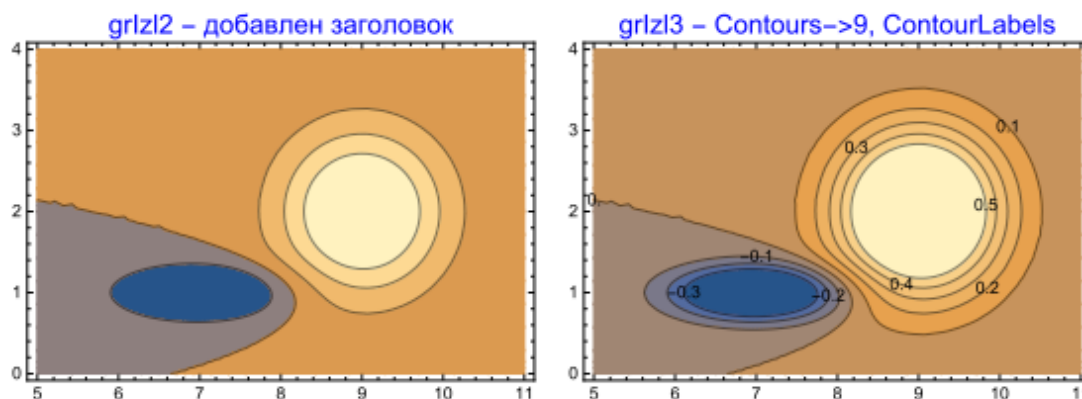
Иллюстрации сопровождаются заголовками, в которых приводится ключевая часть кода.

Пример grIzl3 – иллюстрация применения опций: Contours (задает число контурных линий) и ContourLabels (выводит подписи значений уровней):

```
grIzl2 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grIzl2 – добавлен заголовок", 14, Blue],
    ImageSize → 275];

grIzl3 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grIzl3 – Contours→9, ContourLabels", 14, Blue],
    Contours → 9, ContourLabels → All,
    ImageSize → 275];

Row[{grIzl2, grIzl3}, Spacer[10]]
```

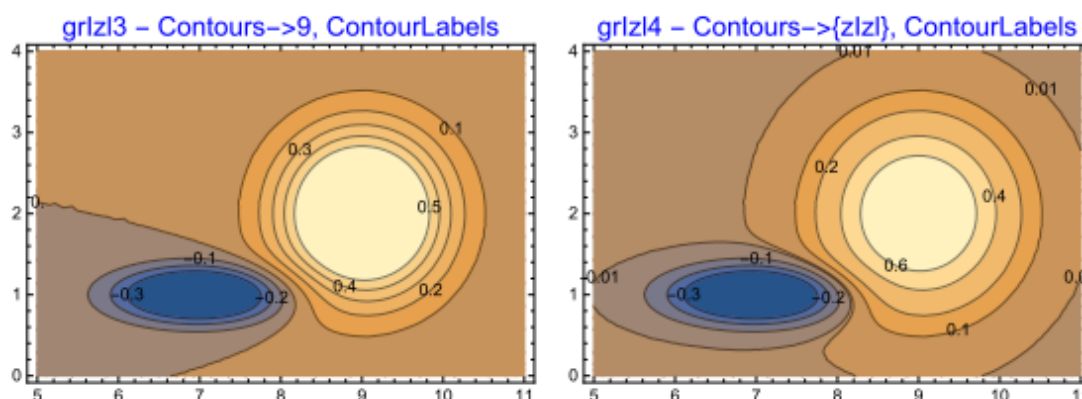



Конкретные значения подлежащих выводу изолиний можно задавать перечислением уровней в списке (Contours $\rightarrow \{z_1, z_2, \dots\}$). В примере grIzl4 в списке zIzl перечислены конкретные значения (допустимо в любом порядке):

```
zIzl = {-0.3, -0.2, -0.1, -0.01, 0.01, 0.1, 0.2, 0.4, 0.6};

grIzl4 = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle  $\rightarrow$  Automatic, AspectRatio  $\rightarrow$  aspRat,
  PlotLabel  $\rightarrow$  Style
    ["grIzl4 – Contours->{zIzl}, ContourLabels", 14, Blue],
  Contours  $\rightarrow$  zIzl, ContourLabels  $\rightarrow$  All, ImageSize  $\rightarrow$  275];

Row[{grIzl3, grIzl4}, Spacer[10]]
```

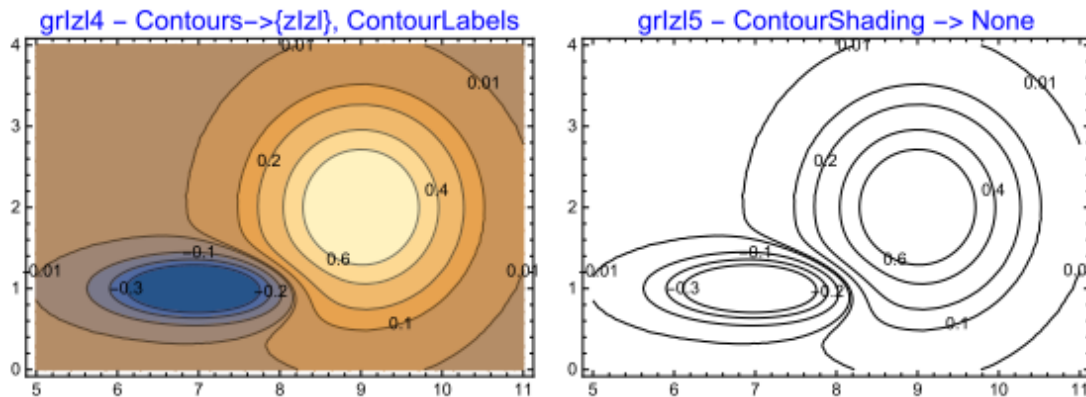


Достаточно часто рассматриваемая функция применяется при выводе иллюстраций нескольких разных распределений, тогда следует для одной из карт изолиний отключать заливку цветом между ними, что обеспечивает опция ContourShading (указывает, должны ли закрашиваться области между контурными линиями) с установкой None (можно и False):

```
grIzl5 = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle  $\rightarrow$  Automatic, AspectRatio  $\rightarrow$  aspRat,

  (* Продолжение кода на следующей странице *)
```

(* Продолжение. Начало кода на предыдущей странице *)
 PlotLabel → Style
 ["grIzl5 - ContourShading → None", 14, Blue],
 Contours → zIzl, ContourLabels → All, ContourShading → None];
 Row[{grIzl4, grIzl5}, Spacer[10], ImageSize → 275]



Примеры ниже иллюстрируют варианты задания атрибутов изолиний (цвет, толщина, тип полилинии) — обеспечивает опция *ContourStyle*. Используя эту опцию, можно устанавливать стиль рисуемых линий для контурных графиков (по умолчанию *Automatic*). Вариант *grIzl6a*:

grIzl6a =
 ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
 ClippingStyle → Automatic, AspectRatio → aspRat,
 PlotLabel → Style
 ["grIzl6a - ContourStyle → {список}", 14, Blue],
 Contours → zIzl, ContourLabels → All, ContourShading → None,
 ContourStyle → {Red, Green, Thick, Dotted, Dashed},
 ImageSize → 275];
 Row[{grIzl5, grIzl6a}, Spacer[10]]

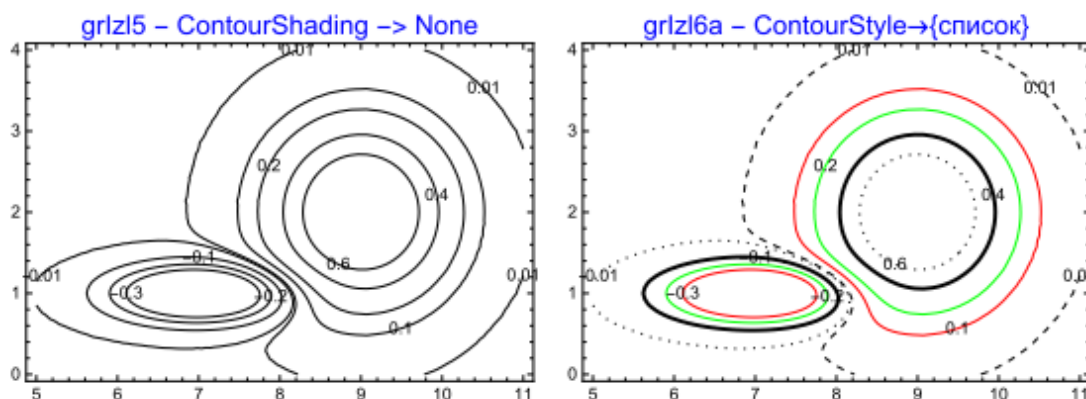
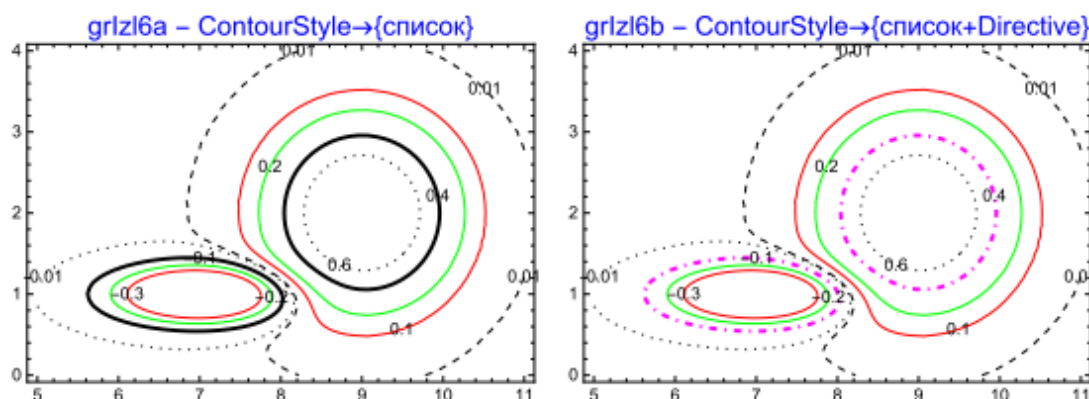


График *grIzl6a* следует пояснить. Записанные в списке атрибуты применяются (вместо установок по умолчанию) в порядке, как они перечислены; причем, в случае, если их число меньше числа изолиний, то применяются повторно в том же порядке — на рисунке видим, что после

пунктирной полилинии (уровень 0.) повторно идут красная, зеленая,

Конструкции кода, когда нужна детализация вида конкретных изолиний приведены на графиках grIzl6b, grIzl6c, grIzl6d. На графике grIzl6b для третьих изолиний (Thick, уровни -0.1 и 0.2), используя Directive, заданы цвет (Magenta), толщина (Thick), тип линии (DotDashed):

```
grIzl6b = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style
    ["grIzl6b - ContourStyle→{список+Directive}", 14, Blue],
  Contours → zIzl, ContourLabels → All, ContourShading → None,
  ContourStyle →
    {Red, Green,
     Directive[{Magenta, Thick, DotDashed}],
     Dotted, Dashed},
  ImageSize → 275];
Row[{grIzl6a, grIzl6b}, Spacer[10]]
```

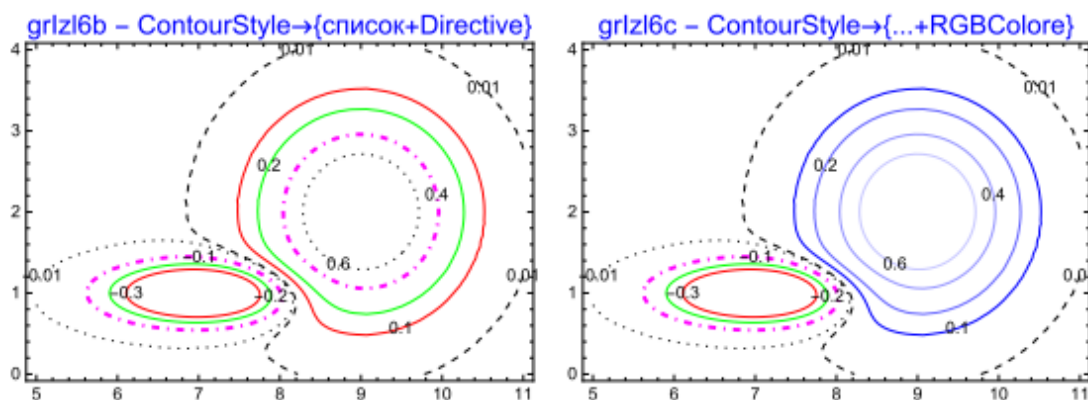


Заметим, что перечислять подобные, как на графике grIzl6b, установки можно в любом порядке. Относительно задания цвета линии – подробно описано в материалах лекции и практического занятия 11 [1]. Напомним, что наиболее часто цвета назначают именами, например: Black (черный), White (белый), Red (красный), Green (зеленый), Blue (синий), Gray (серый), Cyan (сине-зелёный, цвет морской волны), Magenta (сиреневый), Yellow (желтый), Brown (коричневый), Orange (оранжевый), Pink (розовый), Purple (фиолетовый), LightRed (ярко-красный), LightGreen (светло-зеленый), LightBlue (ярко-синий), LightGray (светло-серый), LightCyan (светло-голубой), LightMagenta (малиновый), LightYellow (светло-желтый), LightBrown (светло-коричневый), LightOrange (светло-оранжевый). Возможно использование производных цветов: Lighter, Darker (делают цвет-аргумент белее светлым, темным); Blend – обеспечивает смешивание цветов в указанных пропорциях; ColorNegate – инвертированный цвет (негатив). Также возможно задание графическими директивами в основных цветовых и аппаратно независимых моделях: RGBColor (аддитивная схема

формирования цвета), CMYKColor (субтрактивная схема формирования цвета), Hue (интуитивные цветовые модели HSB, HLS, HSV), GrayLevel (схема оттенков серого), LABColor (цвет в координатах Lab), LCHColor, LUVColor.

В примере grIzl6c список задаваемых атрибутов увеличен, цвета задаются с использованием цветовой модели RGBColor (оттенки синего):

```
grIzl6c = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style
    ["grIzl6c – ContourStyle→{...+RGBColore}", 14, Blue],
  Contours → zIzl, ContourLabels → All,
  ContourShading → None,
  ContourStyle → {Red, Green,
    Directive[{Magenta, Thick, DotDashed}], Dotted, Dashed,
    RGBColor[0, 0, 1], RGBColor[100 / 255, 100 / 255, 1],
    RGBColor[150 / 255, 150 / 255, 1],
    RGBColor[200 / 255, 200 / 255, 1],
    RGBColor[225 / 255, 225 / 255, 1]},
  ImageSize → 275];
Row[{grIzl6b, grIzl6c}, Spacer[10]]
```

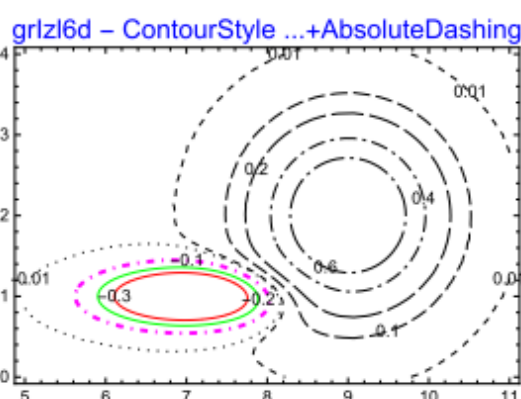
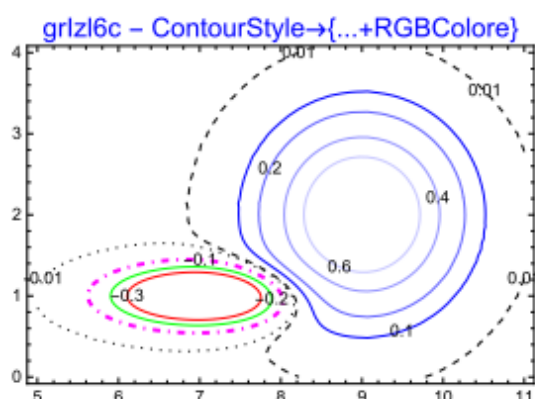


Вариант grIzl6d иллюстрирует способы задания типов линий. В дополнение к основным типам (Dotted, DotDashed, Dashing) задаются линии с “штрихами пользователя” – в этом примере выводимые линии визуально напоминают длинный пунктир (варианты {9, 3}, {17, 3}), двойной (варианты {9,3,2,3}, {17,3,2,3}) и тройной штрих-пунктир:

```
grIzl6d = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style
    ["grIzl6d – ContourStyle ...+AbsoluteDashing", 14, Blue],
  (*Продолжение кода на следующей странице*)
```



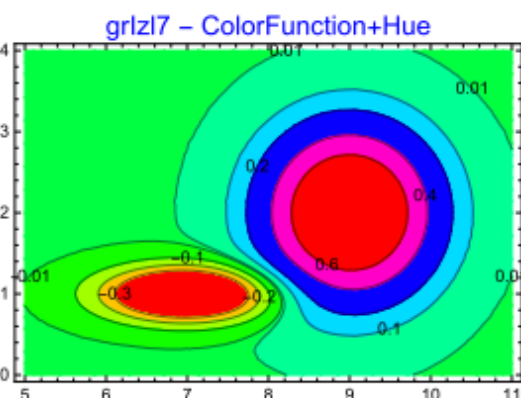
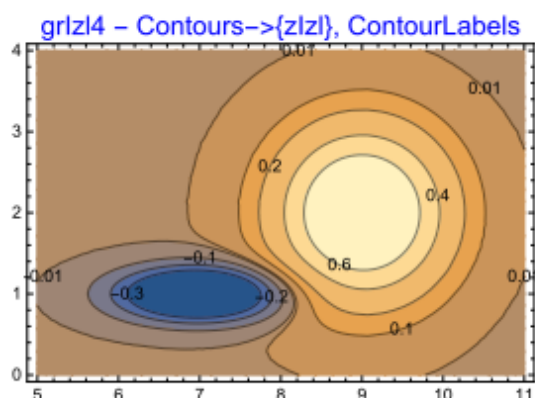
```
(*Продолжение.Начало кода на предыдущей странице*)
Contours → zIz1, ContourLabels → All, ContourShading → None,
ContourStyle → {Red, Green,
  Directive[{Magenta, Thick, DotDashed}], Dotted, Dashed,
  AbsoluteDashing[{9, 3}], AbsoluteDashing[{17, 3}],
  AbsoluteDashing[{9, 3, 2, 3}], AbsoluteDashing[{17, 3, 2, 3}],
  AbsoluteDashing[{9, 3, 2, 3, 2, 3}]}],
ImageSize → 275];
Row[{grIz16c, grIz16d}, Spacer[10]]
```



ContourPlot. Варианты режима цветного вывода

Вариант grIz17 иллюстрирует включение режима цветного вывода – обеспечивает опция ColorFunction, в приведенном варианте используется цветовая схема Hue:

```
grIz17 = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style["grIz17 – ColorFunction+Hue", 14, Blue],
  Contours → zIz1, ContourLabels → All,
  (*ContourShading→None,*)
  ColorFunction → Hue, ImageSize → 275];
Row[{grIz14, grIz17}, Spacer[10]]
```



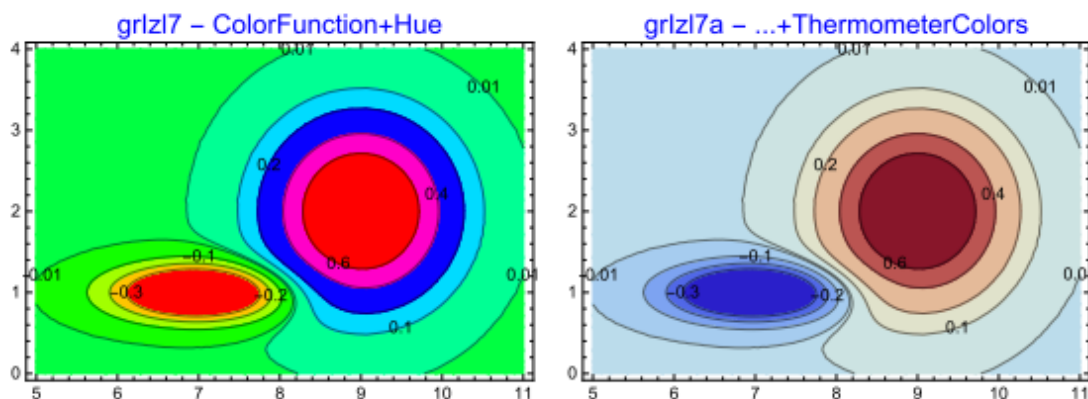
Обратите внимание, что во всех вариантах вывода с заливкой цветом обязательно отключение установки `ContourShading→None`.

Детально возможности и инструменты задания цветов рассмотрены в следующем разделе на примерах для функции `DensityPlot`. Здесь отметим только два примера. Для варианта `grIzl7a` напомним, что в материалах лекции и практического занятия 10 [1] были перечислены разные доступные в системе *Mathematica* цветовые схемы (Color Schemes), которые могут использоваться/подключаться в функциях графики и значительно улучшать зрелищность визуализации. В частности, можно использовать любую градиентную цветовую гамму из следующего списка: `AlpineColors`, `ArmyColors`, `AtlanticColors`, `AuroraColors`, `AvocadoColors`, `BeachColors`, `CandyColors`, `CMYKColors`, `DeepSeaColors`, `FallColors`, `FruitPunchColors`, `IslandColors`, `BrassTones`, `BrownCyanTones`, `CherryTones`, `CoffeeTones`, `FuchsiaTones`, `GrayTones`, `GrayYellowTones`, `DarkTerrain`, `GreenBrownTerrain`, `BrightBands`, `Aquamarine`, `BlueGreenYellow`, `DarkRainbow`, `LightTemperatureMap`, `LakeColors`, `MintColors`, `NeonColors`, `PearlColors`, `PlumColors`, `RoseColors`, `SolarColors`, `SouthwestColors`, `StarryNightColors`, `SunsetColors`, `ThermometerColors`, `WatermelonColors`, `GreenPinkTones`, `PigeonTones`, `RedBlueTones`, `RustTones`, `SiennaTones`, `ValentineTones`, `LightTerrain`, `SandyTerrain`, `DarkBands`, `Pastel`, `Rainbow`, `TemperatureMap`.

Вариант `grIzl7a` иллюстрирует включение режима цветного вывода с использованием цветовой схемы `ThermometerColors`:

```
grIzl7a = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style
    ["grIzl7a - ...+ThermometerColors", 14, Blue],
  Contours → zIzl, ContourLabels → All,
  ColorFunction → "ThermometerColors",
  ImageSize → 275];

Row[{grIzl7, grIzl7a}, Spacer[10]]
```

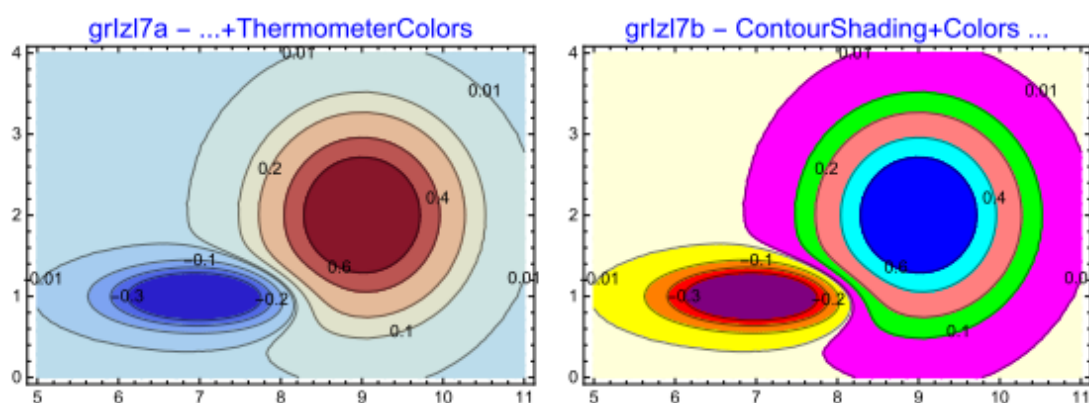


Вариант `grIzl7b` – включение режима цветного вывода с использованием задаваемых пользователем цветов (задаются именами),

реализованы цвета и уровни изолиний, как в иллюстрации рис. И.1:

```
grIzl7b = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  ClippingStyle → Automatic, AspectRatio → aspRat,
  PlotLabel → Style
    ["grIzl7b - ContourShading+Colors ...", 14, Blue],
  Contours → zIzl, ContourLabels → All,
  ContourShading → {Purple, Red, Orange, Yellow,
    LightYellow, Magenta, Green, Pink, Cyan, Blue},
  ImageSize → 275];

Row[{grIzl7a, grIzl7b}, Spacer[10]]
```



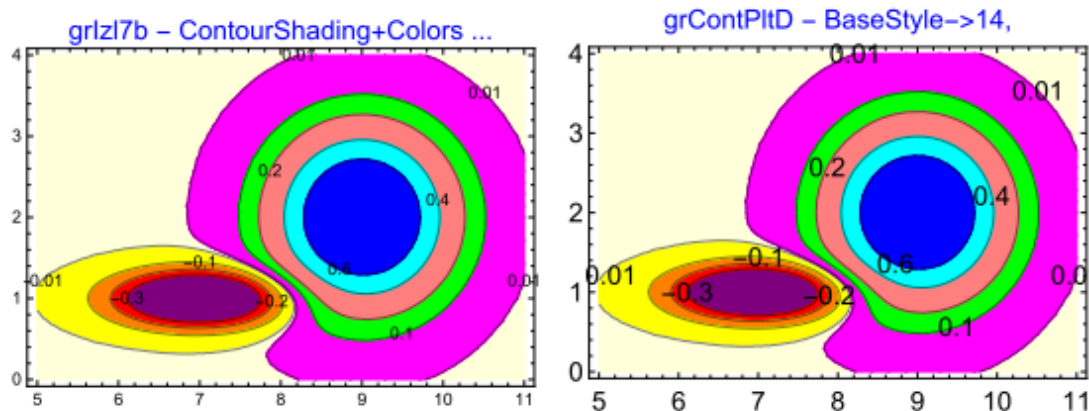
Оформление подписей уровней, рамки окна вывода

В вариантах далее для сокращения кода список цветов заменен идентификатором `rContShad`. Графики `grContPltD` – `grContPltD3` иллюстрируют возможности изменения базового стиля (маркеров изолиний, подписей рамки)

```
rContShad = {Purple, Red, Orange, Yellow,
  LightYellow, Magenta, Green, Pink, Cyan, Blue};

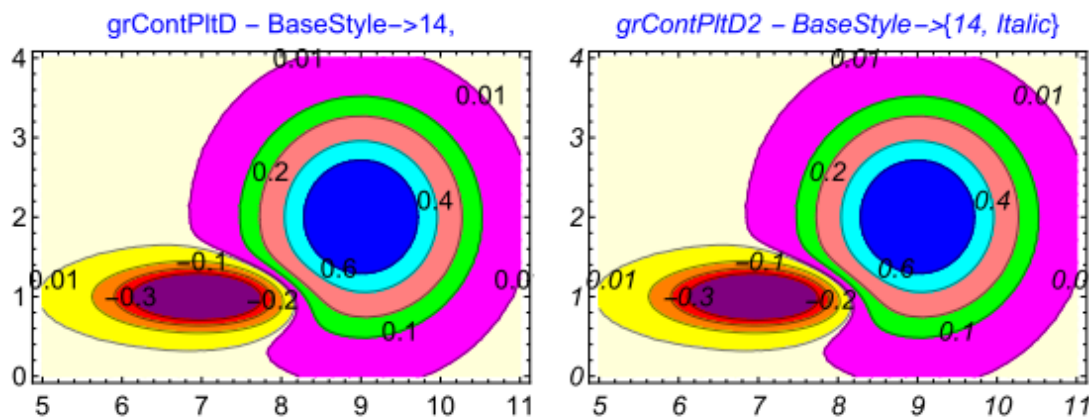
grContPltD =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltD - BaseStyle->14,", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 14,
    ImageSize → 275];

Row[{grIzl7b, grContPltD}, Spacer[10]]
```



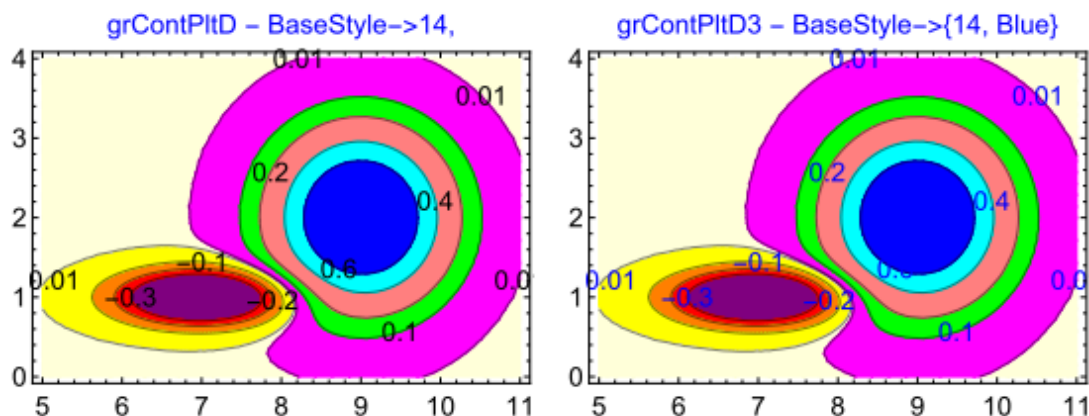
```
grContPltD2 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle -> Automatic, AspectRatio -> aspRat,
    PlotLabel -> Style
      ["grContPltD2 - BaseStyle->{14, Italic}", 14, Blue],
    Contours -> zIzI, ContourLabels -> All,
    ContourShading -> rContShad, BaseStyle -> {14, Italic},
    ImageSize -> 275];
```

```
Row[{grContPltD, grContPltD2}, Spacer[10]]
```



```
grContPltD3 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle -> Automatic, AspectRatio -> aspRat,
    PlotLabel -> Style
      ["grContPltD3 - BaseStyle->{14, Blue}", 14, Blue],
    Contours -> zIzI, ContourLabels -> All,
    ContourShading -> rContShad, BaseStyle -> {14, Blue},
    ImageSize -> 275];
```

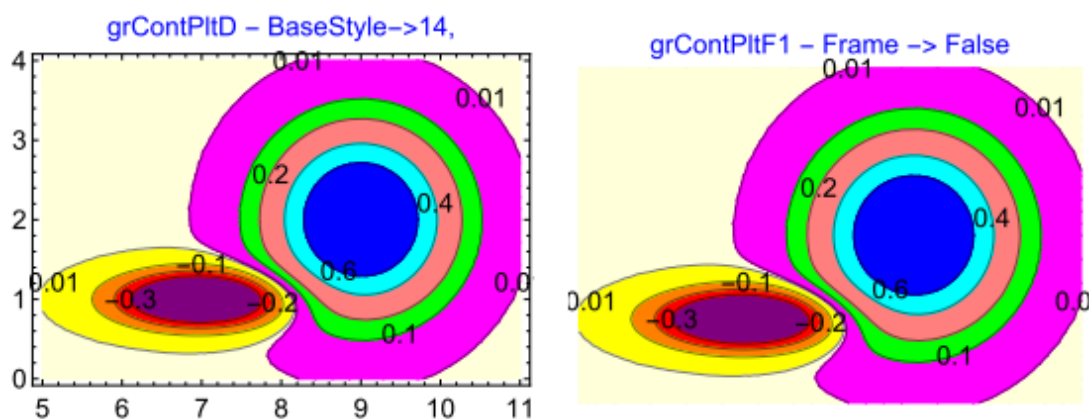
```
Row[{grContPltD, grContPltD3}, Spacer[10]]
```



Графики `grContPltF` – `grContPltF3` иллюстрируют возможности изменения вида, стиля подписей рамки. Основные опции этой категории: `Frame` – задает, выполнять ли прорисовку рамки вокруг графика (при `True`) и ее отсутствие (при `False`); `FrameLabel` – позволяет выводить надписи на гранях рамки (`FrameLabel`→{"Text1","Text2","Text3","Text4"}, построение идет по часовой стрелке, начиная с нижней надписи); `FrameStyle` – позволяет установить индивидуальный стиль граней рамки; `FrameTicks` – задает прорисовку штриховых меток для граней рамки. Пример `grContPltF1` – отмена вывода рамки:

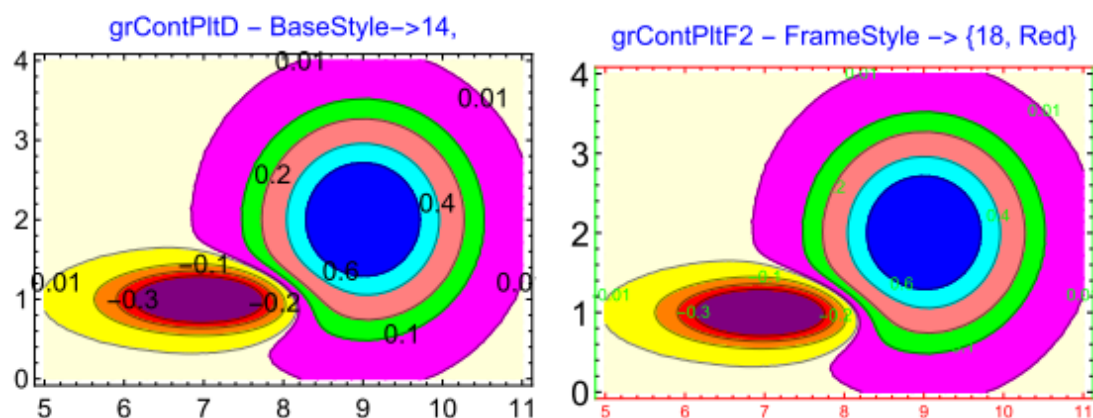
```
grContPltF1 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle -> Automatic, AspectRatio -> aspRat,
    PlotLabel -> Style
      ["grContPltF1 - Frame -> False", 14, Blue],
    Contours -> zIz1, ContourLabels -> All,
    ContourShading -> rContShad, BaseStyle -> 14,
    Frame -> False, ImageSize -> 275];

Row[{grContPltD, grContPltF1}, Spacer[10]]
```



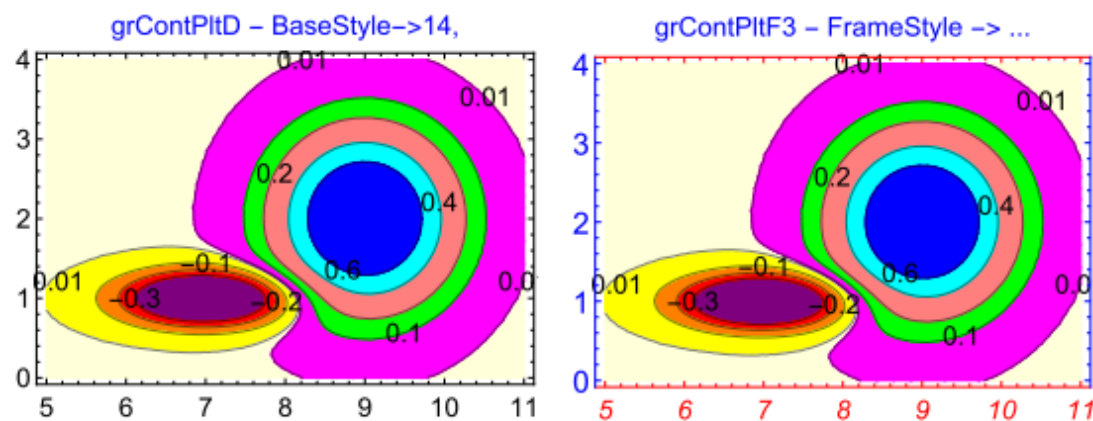
Графики `grContPltF2` – `grContPltF3` иллюстрируют возможности изменения вида, стиля надписей рамки. Обратите внимание, как выводятся подписи уровней, линии и подписи рамки в `grContPltF2`:


```
grContPltF2 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltF2 - FrameStyle -> {18, Red}", 14, Blue],
    Contours → zIzl,
    ContourLabels → All, ContourShading → rContShad,
    BaseStyle → Green, FrameStyle → {18, Red}, ImageSize → 275];
Row[{grContPltD, grContPltF2}, Spacer[10]]
```

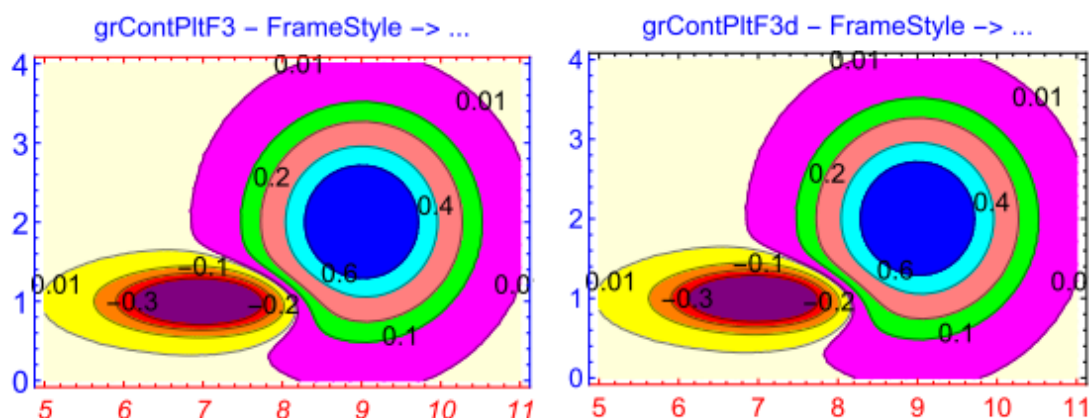


Графики grContPltF3, grContPltF3d иллюстрируют эффекты, как выполняется оформление при записи с использованием Directive и без:

```
grContPltF3 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltF3 - FrameStyle -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 14,
    FrameStyle → {Directive[{Blue, 16}],
      Directive[{Red, Italic}]}, ImageSize → 275];
Row[{grContPltD, grContPltF3}, Spacer[10]]
```




```
grContPltF3d =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltF3d - FrameStyle -> ...", 14, Blue],
    Contours → zIz1, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 14,
    FrameStyle → {{Blue, 16}, {Red, Italic}},
    ImageSize → 275];
Row[{grContPltF3, grContPltF3d}, Spacer[10]]
```

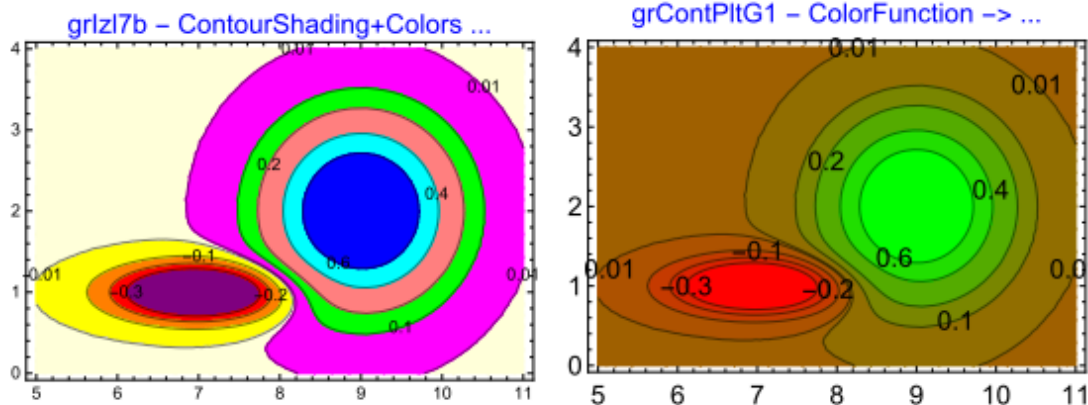


Оформление контурной графики индивидуальной цветовой схемой

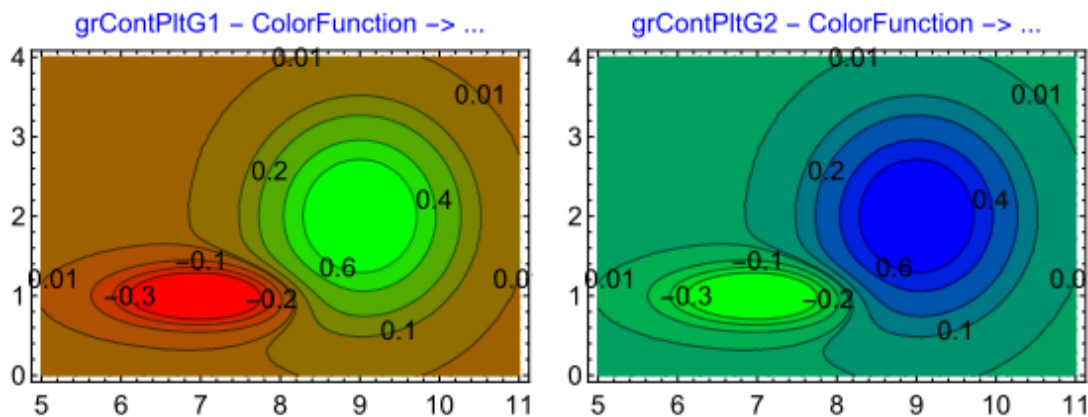
Графики `grContPltG1` – `grContPltGD4` иллюстрируют возможности изменения цветовой схемы окраски (заливки) между контурными линиями. Окраска обеспечивается опцией `ColorFunction`, которая определяет цветовую схему, используемую для функциональной окраски. `ColorFunction` задается в виде функции и должна иметь определенное количество аргументов, которое зависит от графической функции, должна возвращать строго определенный объект: `RGBColor`, `Hue` или `GrayLevel`. Например, функция `RGBColor[r,g,b]` принимает три числовых аргумента в диапазоне от 0 до 1, каждый из которых определяет относительное содержание красной (r), зеленой (g) и синей (b) составляющих цвета; она создает объект, который должна возвращать любая функция, представляющая значение опции `ColorFunction`. На поведение `ColorFunction` влияет опция `ColorFunctionScaling`, которая может принимать значения `True` или `False`. Если `ColorFunctionScaling` → `True` (значение по умолчанию), то все аргументы функции `ColorFunction` автоматически приводятся к диапазону [0, 1]. Если `ColorFunctionScaling` → `False`, то масштабирование аргументов не выполняется.

Графики `grContPltG1` и `grContPlt2` иллюстрируют интерполяцию между двумя указанными цветами (красным и зеленым, зеленым и синим), задаваемыми функций `RGBColor`:

```
grContPltG1 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG1 - ColorFunction -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (RGBColor[1 - #, #, 0] &),
    ImageSize → 275];
Row[{grIzl7b, grContPltG1}, Spacer[10]]
```

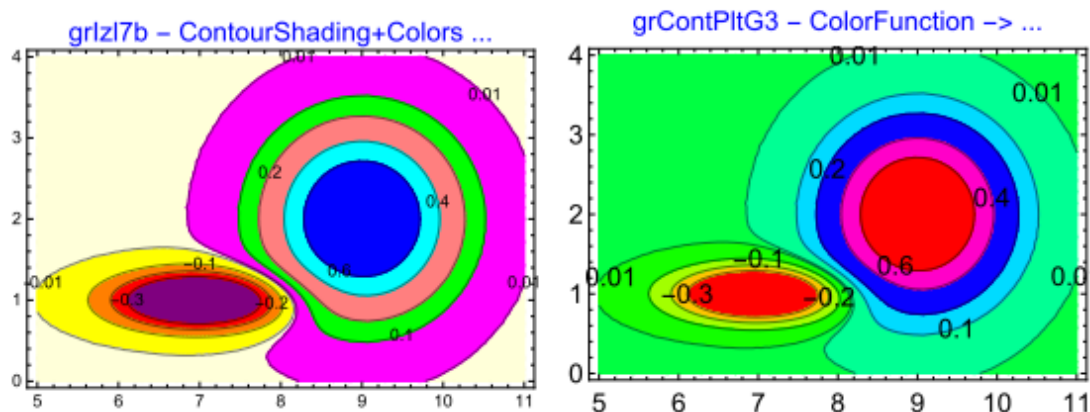


```
grContPltG2 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG2 - ColorFunction -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (RGBColor[0, 1 - #, #] &),
    ImageSize → 275];
Row[{grContPltG1, grContPltG2}, Spacer[10]]
```

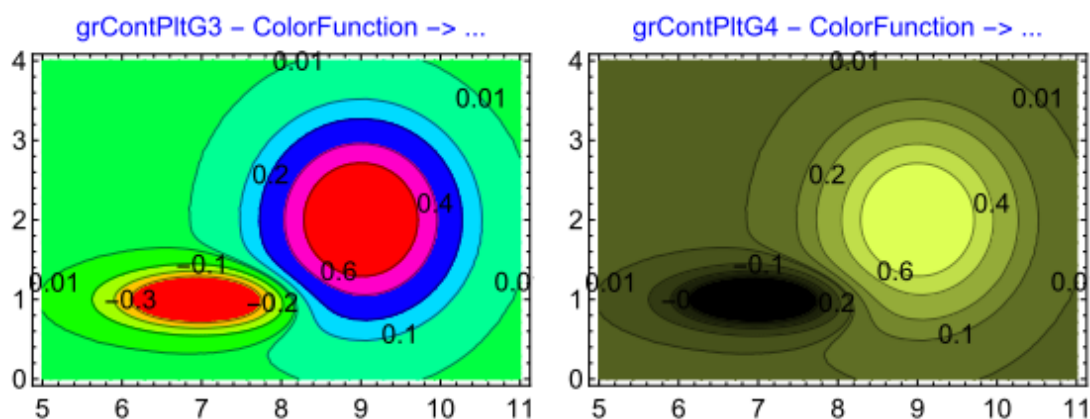


Графики grContPltG3 – grContPlt6 иллюстрируют использование цветовой модели Hue:

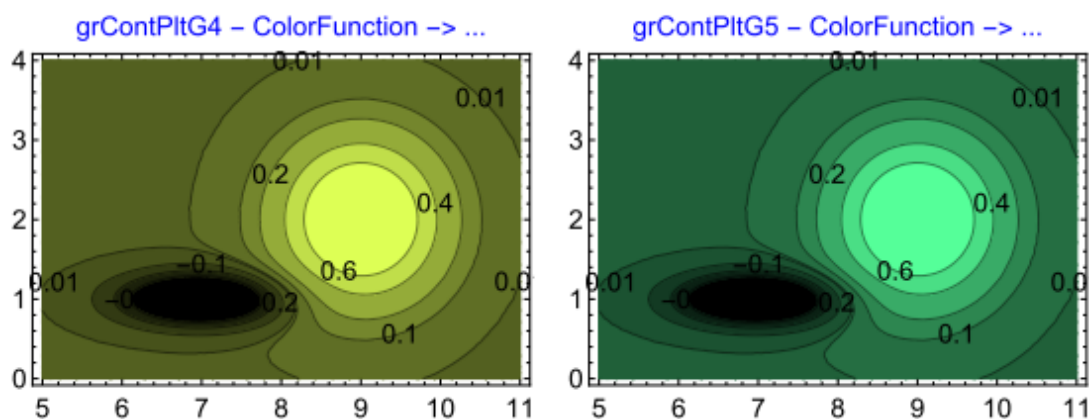
```
grContPltG3 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG3 – ColorFunction -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → Hue,
    ImageSize → 275];
Row[{grIzl7b, grContPltG3}, Spacer[10]]
```



```
grContPltG4 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG4 – ColorFunction -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (Hue[1 / 5, 2 / 3, #] &),
    ImageSize → 275];
Row[{grContPltG3, grContPltG4}, Spacer[10]]
```

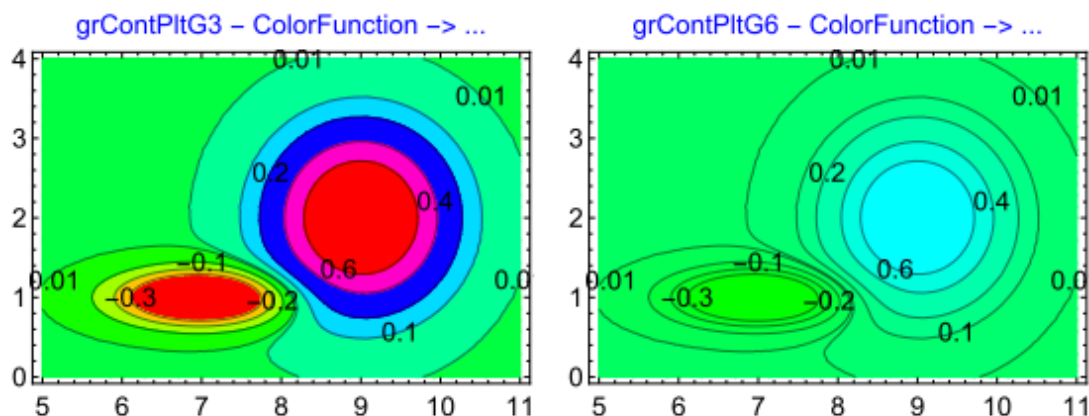


```
grContPltG5 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG5 - ColorFunction -> ...", 14, Blue],
    Contours → zIz1, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (Hue[2 / 5, 2 / 3, #] &),
    ImageSize → 275];
Row[{grContPltG4, grContPltG5}, Spacer[10]]
```



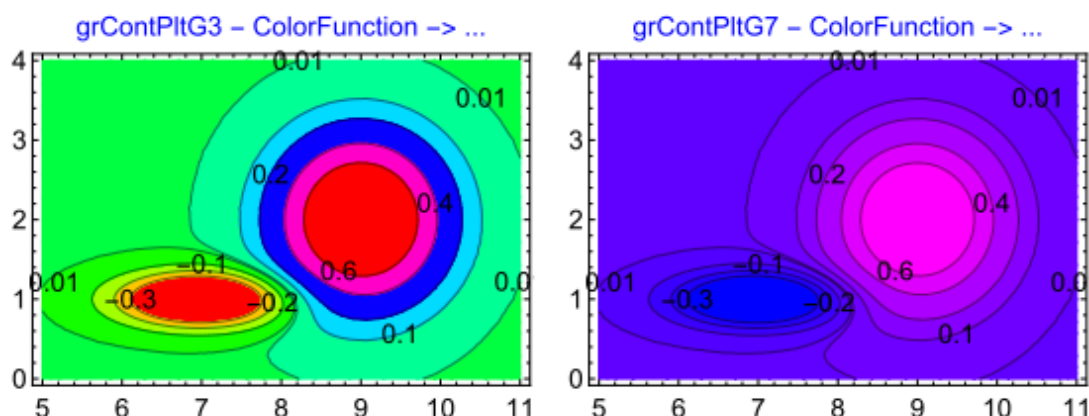
Графики grContPltG6, grContPlt7 иллюстрируют использование цветовой модели Hue для визуализации цветового диапазона между задаваемыми цветами (Green-Cyan, Blue-Magenta):

```
grContPltG6 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG6 - ColorFunction -> ...", 14, Blue],
    Contours → zIz1, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (Hue[1 / 3 + (1 / 6) * #1] &),
    ImageSize → 275];
Row[{grContPltG3, grContPltG6}, Spacer[10]]
```




```
grContPltG7 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltG7 – ColorFunction -> ...", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    BaseStyle → 14,
    ColorFunction → (Hue[2 / 3 + (1 / 6) * #1] &),
    ImageSize → 275];

Row[{grContPltG3, grContPltG7}, Spacer[10]]
```



Изолинии, варианты вывода легенды (принятой шкалы цветов)

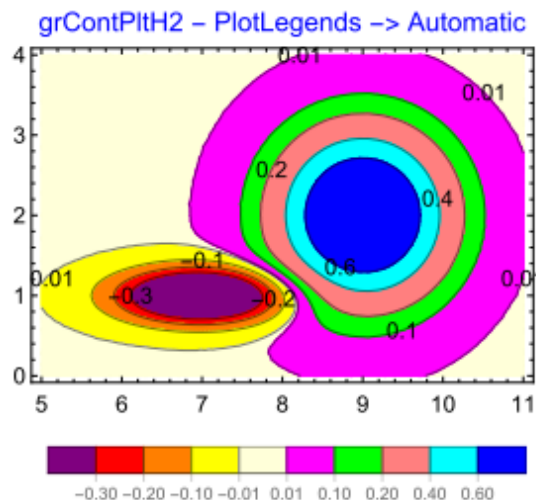
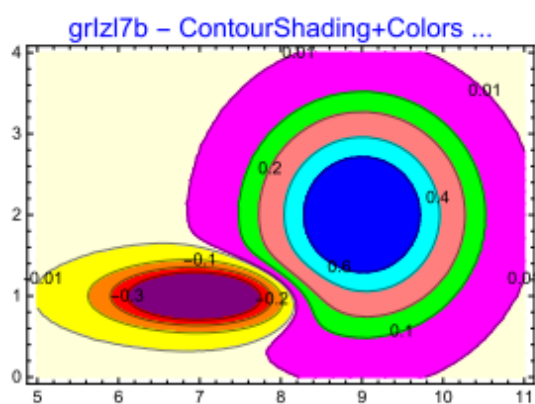
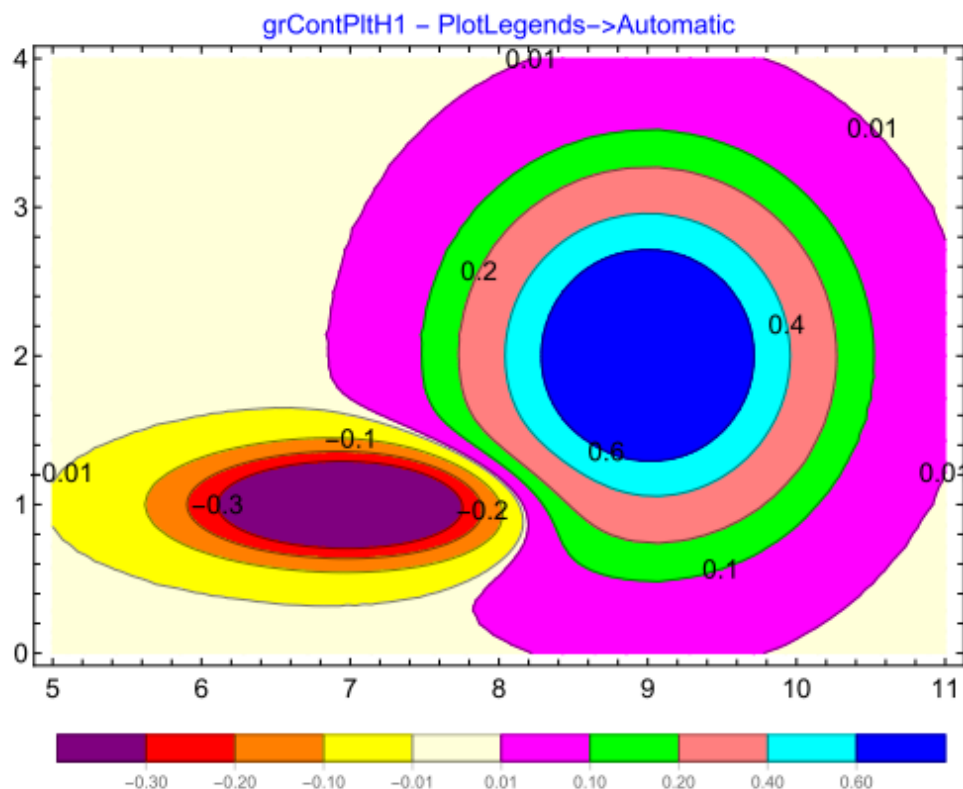
Графики grContPltH1 – grContPltH4 иллюстрируют возможности оформления карт изолиний легендой принятой цветовой палитры фона, шкалы изолиний (показывает параметры цветной карты).

График grContPltH1 построен с использованием PlotLegends->Automatic, которая строит контрольную цветовую полосу снизу от графика. Графики grContPltH2, grContPltH3 иллюстрируют возможности уточнения стиля подписей в легенде (обратите внимание, что на графике grContPltH2 несколько подписей в левой части “заслоняют” предыдущие маркеры):

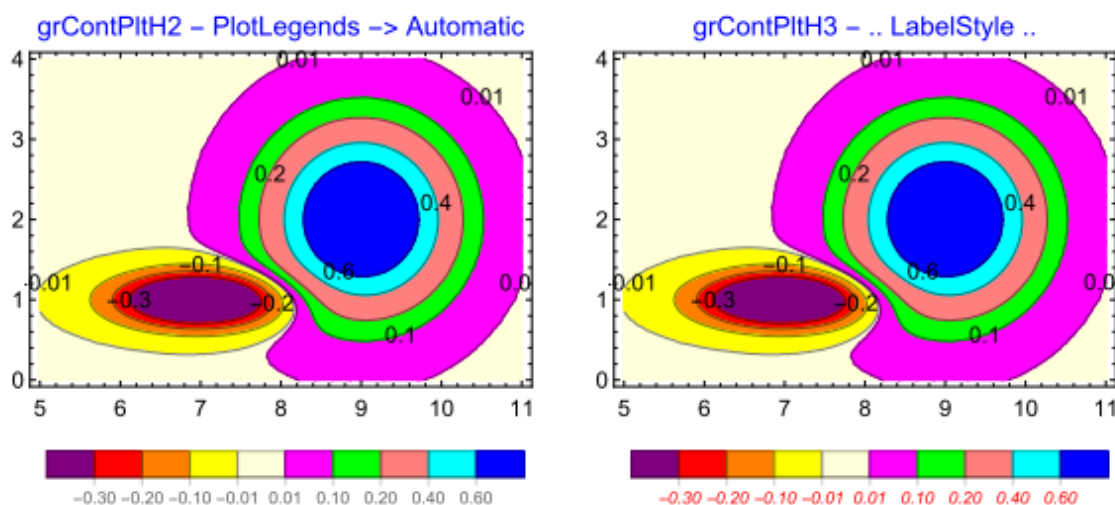
```
grContPltH1 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltH1 – PlotLegends->Automatic", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 14,
    PlotLegends → Automatic,
    ImageSize → 550]
(* Продолжение кода на следующей странице *)
```



```
(* Продолжение. Начало кода на предыдущей странице *)
grContPltH2 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle -> Automatic, AspectRatio -> aspRat,
    PlotLabel -> Style
      ["grContPltH2 - PlotLegends -> Automatic", 14, Blue],
    Contours -> zIz1, ContourLabels -> All,
    ContourShading -> rContShad, BaseStyle -> 12,
    PlotLegends -> Automatic, ImageSize -> 275];
Row[{grIz17b, grContPltH2}, Spacer[10]]
```



```
grContPltH3 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltH3 - .. LabelStyle ..", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 12,
    PlotLegends → Placed[BarLegend[Automatic,
      LabelStyle → {FontSize → 9, Red, Italic}], Bottom],
    ImageSize → 275];
Row[{grContPltH2, grContPltH3}, Spacer[5]]
```



Графики grContPltH4, grContPltH5 иллюстрируют возможности позиционирования с нужной стороны карты (основные варианты: Center, Top, Left, Right, Bottom), включения заголовка, обрамления рамкой задаваемого вида, уточнения стиля подписей в легенде :

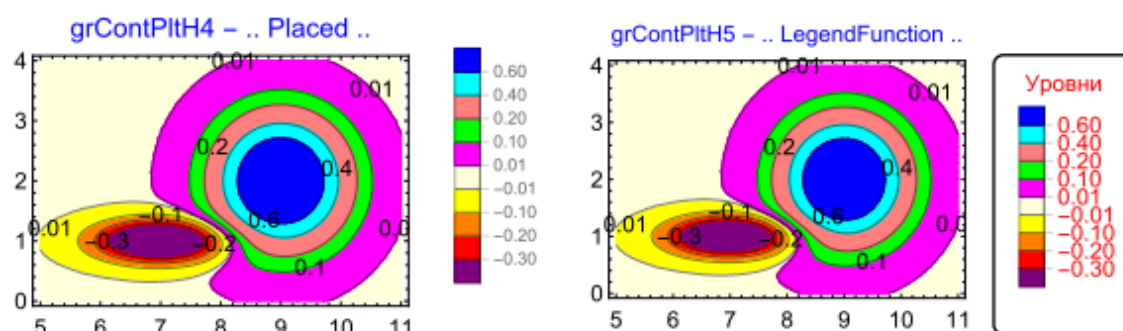
```
grContPltH4 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    PlotLabel → Style
      ["grContPltH4 - .. Placed ..", 14, Blue],
    Contours → zIzl, ContourLabels → All,
    ContourShading → rContShad, BaseStyle → 12,
    PlotLegends → Placed[BarLegend[Automatic], Right],
    ImageSize → 210];

grContPltH5 =
  ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    ClippingStyle → Automatic, AspectRatio → aspRat,
    (* Продолжение кода на следующей странице *)
```

(* Продолжение. Начало кода на предыдущей странице *)

```
PlotLabel → Style
["grContPltH5 - .. LegendFunction ..", 12, Blue],
Contours → zIzl, ContourLabels → All,
ContourShading → rContShad, BaseStyle → 12,
PlotLegends → Placed[BarLegend[Automatic,
  LegendMarkerSize → 110, LabelStyle → {FontSize → 12, Red},
  LegendFunction → (Framed[#, RoundingRadius → 5] &),
  LegendLabel → "Уровни"], {After, Top}],
ImageSize → 200];
```

```
grContPltH5s = Row[{grContPltH4, grContPltH5}, Spacer[5]]
```



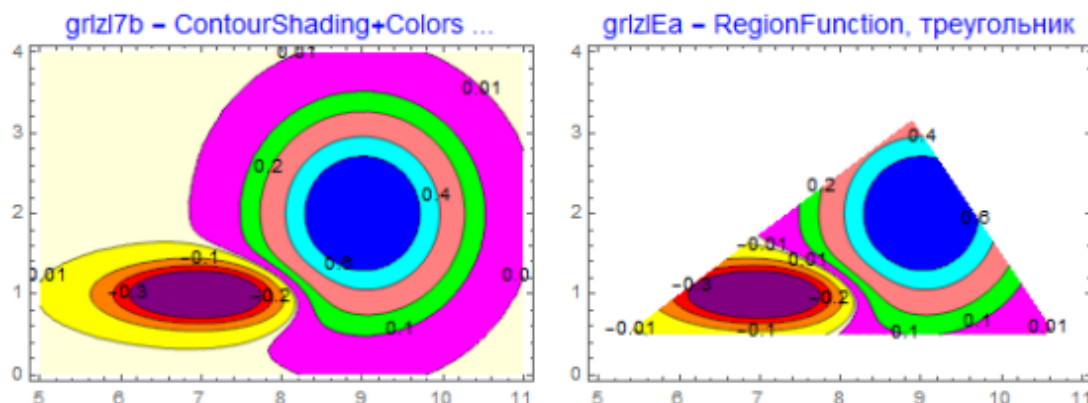
Карты изолиний, примеры специальных настроек

Графики grIzlEa, grIzlEb, grIzlEc, grIzlEd иллюстрируют возможности вывода не всей карты изолиний, а указываемого фрагмента. Такое нужно при формировании сводных изображений, когда вставки делаются с увеличением (детализация участка) или, наоборот, когда размещается миниатюра.

График grIzlEa – результат вывода с использованием опции RegionFunction участка, ограниченного сторонами треугольника:

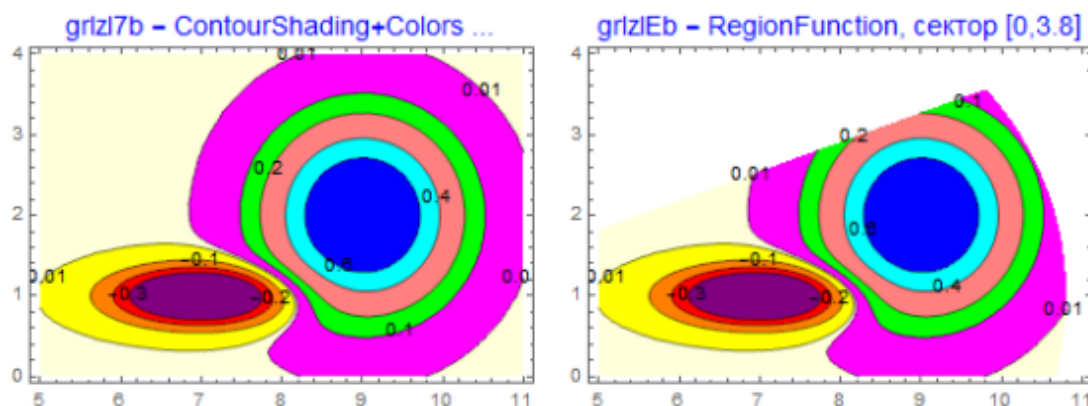
```
grIzlEa = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat,
  PlotLabel → Style[
    "grIzlEa - RegionFunction, треугольник", 14, Blue],
  Contours → zIzl,
  ContourLabels → All, ContourShading → rContShad,
  RegionFunction → Function[
    {x, y}, y ≤ 0.75 * x - 3.5 && y ≤ -1.5 * x + 16.5 && y ≥ 0.5
  ] (*в треугольнике*),
  ImageSize → 275];
```

```
grIzlEas = Row[{grIzl7b, grIzlEa}, Spacer[10]]
```

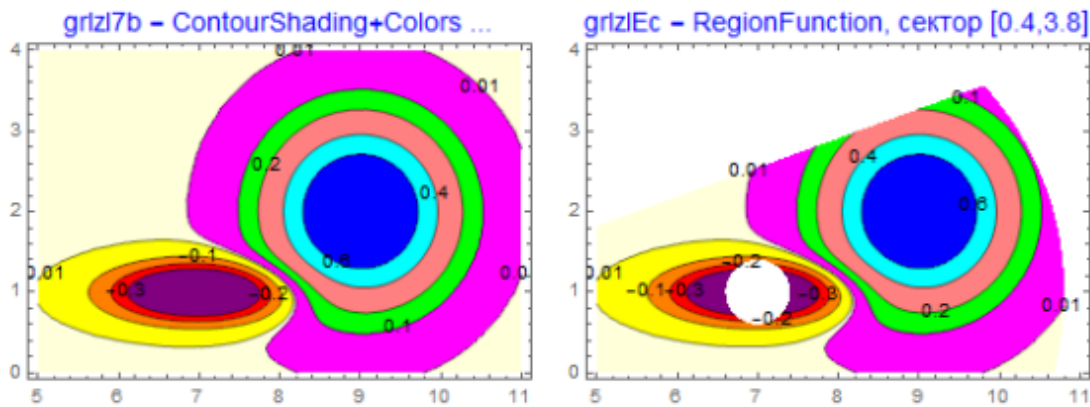


Графики grIzlEb, grIzlEc иллюстрируют варианты вывода сектора, сектора-кольца:

```
grIzlEb = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat,
  PlotLabel →
    Style["grIzlEb - RegionFunction, сектор [0,3.8]", 14, Blue],
  Contours → zIzl, ContourLabels → All,
  ContourShading → rContShad, RegionFunction →
    Function[{x, y}, 0 ≤ Sqrt[(x - 7) ^ 2 + (y - 1) ^ 2] ≤ 3.8 &&
      0 ≤ ArcTan[x, y] ≤ π / 9] (*в секторе*),
  ImageSize → 275];
grIzlEbs = Row[{grIzl7b, grIzlEb}, Spacer[10]]
```

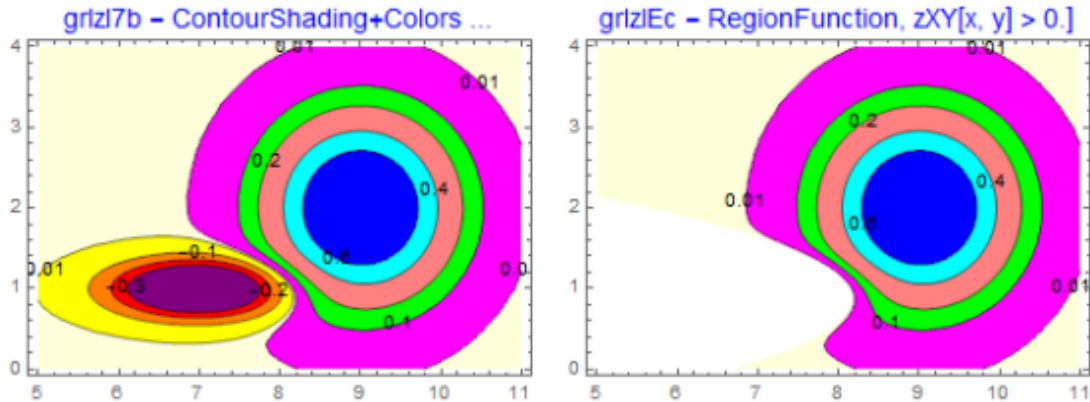


```
grIzlEc = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat,
  PlotLabel → Style[
    "grIzlEc - RegionFunction, сектор [0.4,3.8]", 14, Blue],
  Contours → zIzl, ContourLabels → All,
  ContourShading → rContShad, RegionFunction →
    Function[{x, y}, 0.4 ≤ Sqrt[(x - 7) ^ 2 + (y - 1) ^ 2] ≤ 3.8 &&
      0 ≤ ArcTan[x, y] ≤ π / 9] (*в секторе кольца*),
  ImageSize → 275];
grIzlEcs = Row[{grIzl7b, grIzlEc}, Spacer[10]]
```

Иной способ выделения фрагмента иллюстрирует график `grIzlEd`, который формируется, когда выводится часть изображения по превышению задаваемого уровня:

```
grIzlEd = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat,
  PlotLabel → Style[
    "grIzlEc - RegionFunction, zXY[x, y] > 0.", 14, Blue],
  Contours → zIzl, ContourLabels → All, ContourShading → rContShad,
  RegionFunction → Function[{x, y, z}, zXY[x, y] > 0.],
  ImageSize → 275];
grIzlEds = Row[{grIzl7b, grIzlEd}, Spacer[10]]
```



Графики `grIzlFa`, `grIzlFb` иллюстрируют возможности настройки вывода подписей изолиний. Управление размещением подписей контурных линий важно в случаях, когда автоматически сформированные подписи в отдельных местах затеяют изолинии или имеет место наложение подписей. Вариант вывода подписей в указанной полосе реализован в примере ниже. Достигается требуемое синтезом двух изображений. Базовый график `grIzl7b` незначительно упрощается – отключается вывод меток изолиний (`ContourLabels→None`), соответствующее изображение идет в фон. Задаются уточненные, конкретно интересные пользователю, значения уровней – список `zIzl2`. В изображении `grIzlFa`, выводимом над фоном, полоса вывода задается применением опции `RegionFunction`. Используя функцию `Show`,

перечисленные изображения сливаются в один график grIzlEfs:

```
grIzl7b2 = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat, PlotLabel →
  Style["specify position of ContourLabel", 14, Blue],
  Contours → zIzl, ContourLabels → None,
  ContourShading → rContShad, ImageSize → 275];

zIzl2 = {-0.3, -0.1, 0.1, 0.4, 0.6};

grIzlFa = ContourPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  PlotRange → Full, AspectRatio → aspRat,
  Contours → zIzl2, ContourStyle → None, ContourLabels → All,
  ContourShading → None, RegionFunction →
  Function[{x, y}, 0.5 * x + 1.5 - y > 3.5 && 0.5 * x + 1.5 - y < 4],
  ImageSize → 275];
grIzlEfs = Show[grIzl7b2, grIzlFa];

grIzlEds = Row[{grIzl7b, grIzlEfs}, Spacer[10]]
```

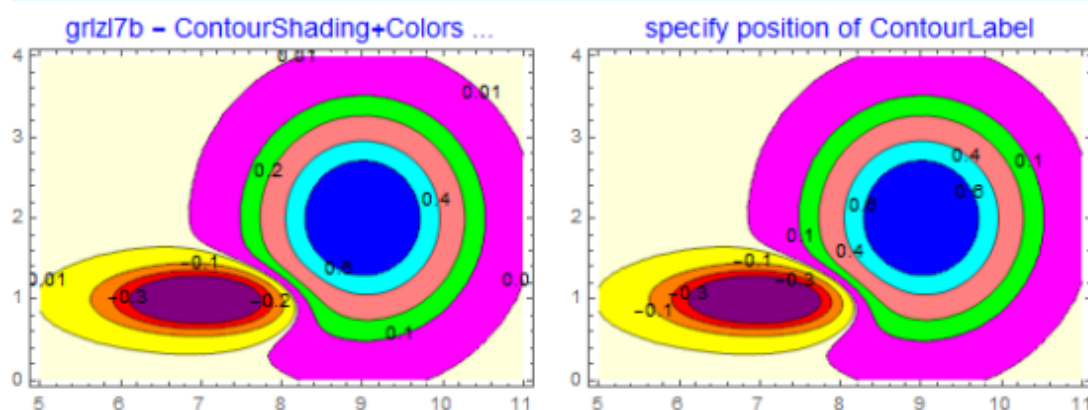
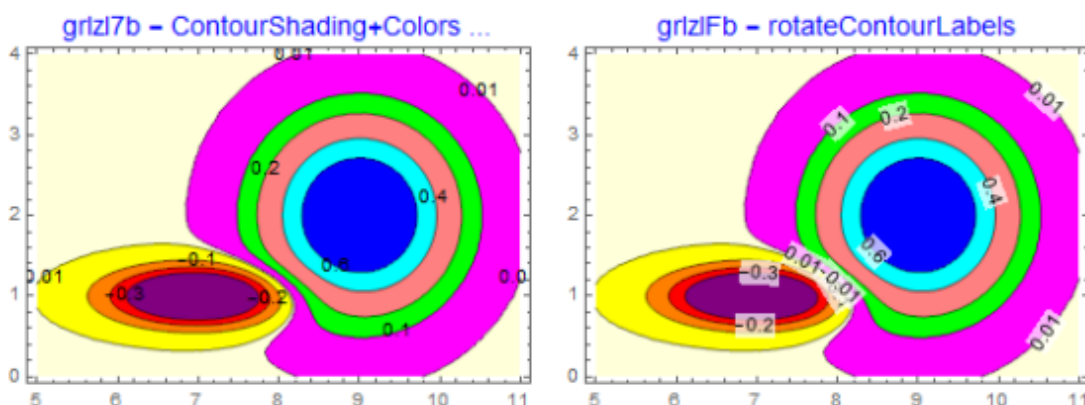


График grIzlFb иллюстрирует возможность реализации, как автоматически размещать метки изолиний в середине контурных линий, причем, с ориентацией подписи приблизительно параллельно изолинии. Код секции использует функцию rotateContourLabels [2]. Функция rotateContourLabels принимает выходные данные ContourPlot, предполагая, что они были созданы с помощью опции ContourLabel->All. График передается через требуемый первый аргумент. Функция поворачивает метки всех контуров так, чтобы они были приблизительно параллельны изолиниям. Необязательная функция Label может указывать пользовательский стиль метки в виде функции f[#1, #2, #3], где {#1, #2} – вектор 2D-местоположения метки, а #3 – значение построенной функции в этом местоположении. Стиль метки по умолчанию задается текстом функции [#3, {#1, #2}]&. Второй параметр, выравнивание, влияет на размещение меток по контуру. Выравнивание по умолчанию Automatic оставляет исходное расположение нетронутым, любое другое значение

приведет к смещению меток в предполагаемое центральное положение по контуру. :

```
grIzlFb0 = rotateContourLabels[grIzl7b,
  Alignment -> Center, LabelingFunction -> (Text[#3, {#1, #2},
    Background -> Directive[Opacity[.7], White]] &)];
grIzlFb = Show[grIzlFb0, PlotLabel -> Style[
  "grIzlFb - rotateContourLabels", 14, Blue]];

grIzlFbs = Row[{grIzl7b, grIzlFb}, Spacer[10]]
```



Тестовые задания для самоконтроля

Упражнение И.1.

Графика системы *Mathematica*. Визуализация двумерных данных.

Примеры задания атрибутов изолиний

На иллюстрации ниже приведены два разных графика. Использован код:

```
fXY[x_, y_] := Sin[1 + 1.2*E^(-(x/2 - 2)^2 - (y/2 - 2)^2)];

grXY1 = XXXX[fXY[x, y], {x, 0, 9}, {y, 0, 9},
  Contours -> {0.85, 0.87, 0.91, 0.95, 0.99}, ImageSize -> 275];

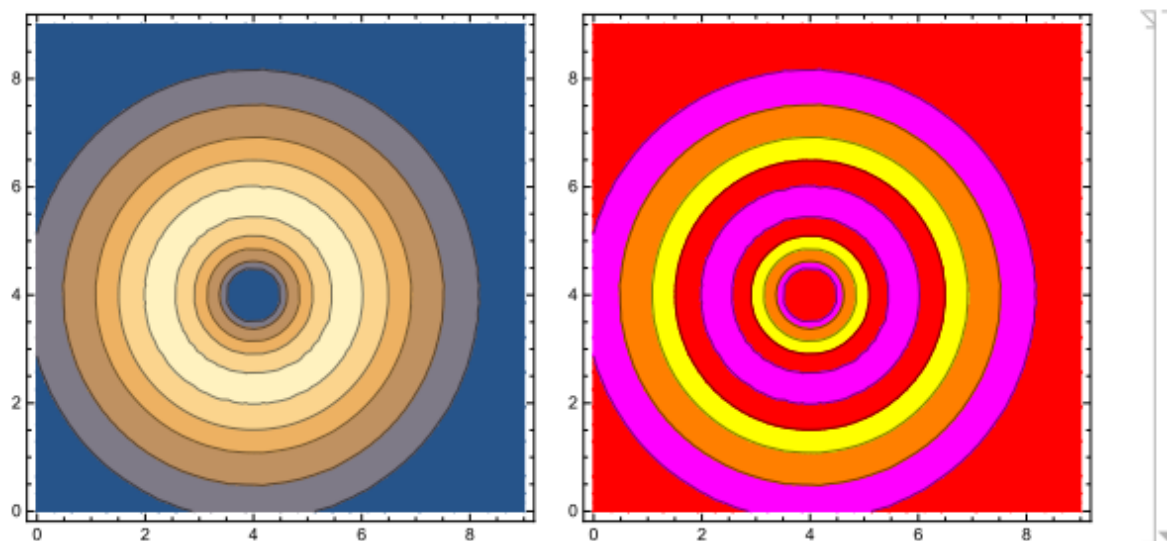
grXY2 = XXXX[fXY[x, y], {x, 0, 9}, {y, 0, 9},
  Contours -> {0.85, 0.87, 0.91, 0.95, 0.99}, Saaaa, ImageSize -> 275];

Row[{grXY1, grXY2}, Spacer[20]]
```

Что следует вписать вместо Saaaa для получения приведенного графика?

Подсказки:

Дополнительную опцию, использованную при формировании графика справа, надо определить и отправить. Символов в ответе - 43. Пробелов в ответе быть не должно, есть adin; цвета задавать именами; правило вписывать двумя символами ->.



Упражнение И.2.

Графика системы *Mathematica*. Визуализация двумерных данных. Изолинии. Цветовые схемы.

На иллюстрации ниже приведены два разных графика. Использован код:

```
rClrF = {"BeachColors", "CandyColors", "CherryTones", "FallColors",  
"LakeColors", "MintColors", "RoseColors", "SolarColors", "SunsetColors"};
```

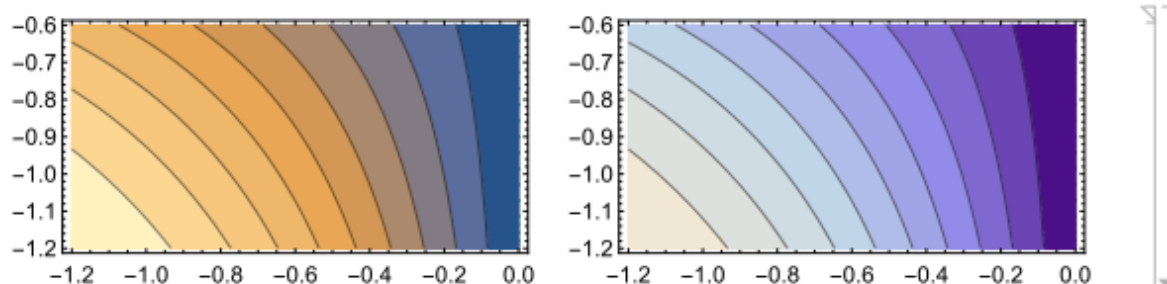
```
grC1 = ContourPlot[Sin[x*y], {x, -1.2, 0}, {y, -1.2, -0.6},  
AspectRatio -> 1/2, BaseStyle -> 12, ImageSize -> 275];
```

```
grC2 = ContourPlot[Sin[x*y], {x, -1.2, 0}, {y, -1.2, -0.6},  
AspectRatio -> 1/2, BaseStyle -> 12, ImageSize -> 275, ????];
```

```
Row[{grC1, grC2}, Spacer[10]]
```

Получить приведенный результат. В поле ответа внести скрытый ???? фрагмент.

Подсказки: Символов в ответе - 25.





Двумерная графика системы *Mathematica*. Визуализация функций

Таранчук Валерий Борисович

БГУ,

факультет прикладной математики и информатики

Учебные материалы, инструкции и рекомендации пользователям системы
компьютерной алгебры *Mathematica*, обучающие примеры и упражнения
(оригинал документа предоставляется студентам в формате PDF)

▼ План лекции, лабораторного занятия “Карты плотности” >>>>

- ✓ Визуализация функции двух переменных средствами *DensityPlot*
- ✓ *DensityPlot* – оформление, детали, цветовые диапазоны
- ✓ Тестовые задания для самоконтроля

Визуализация функции двух переменных средствами *DensityPlot*

Следуя изложению и принятой в [1] классификации типов графических объектов системы *Mathematica* к категории “Средства 2D графики аналитически задаваемых функций” отнесены и обозначены первыми:

- *ContourPlot* – контурный график на плоскости, который показывает линии равного уровня (изолинии); контурные графики (или графики в горизонталях) наиболее часто используются в топографии для представления на плоскости объемного рельефа местности;
- *DensityPlot* – плотностный график (карта зон), значение функции в каждой точке отображается при помощи окрашивания в определённый цвет, можно задавать интервалы и соответствующие им цвета, можно применять плавные цветовые переходы (градиентная заливка);

В материалах предыдущей темы “Изолинии” рассмотрены возможности и примеры визуализации поверхностей средствами двумерной графики при помощи изолиний (линий равного уровня) – *ContourPlot*. При таком способе иллюстрирования поверхностей реализуется техника, в которой выполняются следующие действия: поверхность, заданную функцией двух переменных $z = f(x, y)$, пересекают плоскостью $z = \text{const}$; получаемую в сечении плоскую кривую (кривые), проектируют на плоскость XY ; в результате многократного пересечения с разными значениями z и отображения проекций всех получаемых кривых на плоскость XY формируется семейство кривых, представляющих изображение поверхности 3D набором линий уровней в 2D. При использовании *ContourPlot* с установками по умолчанию площади между изолиниями закрашиваются оттенками цветов принятой схемы раскраски, но заполнение можно отключать.

Часто для двумерного иллюстрирования функций двух переменных используется функция Wolfram *Mathematica* *DensityPlot*. При ее выполнении

область изменения каждого из аргументов разбивается на определяемое алгоритмом в зависимости от характера поведения визуализируемой функции количество дискретных ячеек. Значение функции в каждой ячейке отображается при помощи окрашивания этой ячейки в определённый цвет. Правила окрашивания подобны, как и для ContourPlot, но есть много особенностей. Рассмотрим основные опции и специфику работы с DensityPlot.

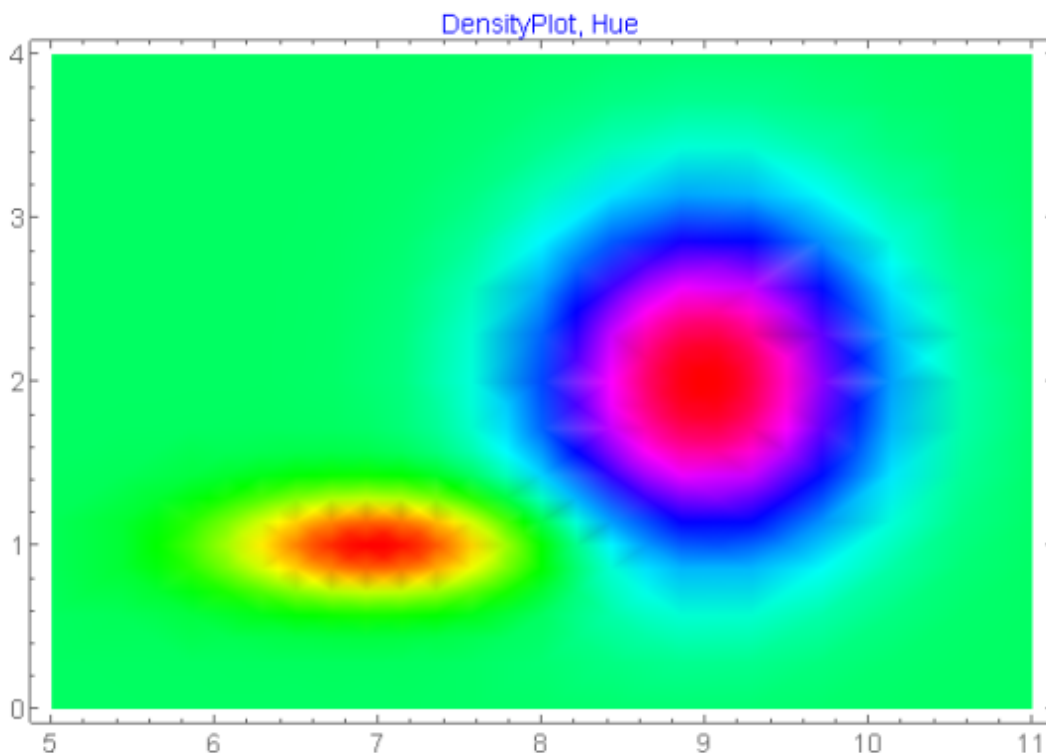
В изложении материалов текущей темы, как и предыдущей, используется выражение $z_{XY}(x,y)$, определяемое аналитически:

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

Результат визуализации с использованием DensityPlot при принятой традиционной схеме раскраски ColorFunction→Hue, минимуме дополнений в настройках, поясненных при визуализации grIzll уточнениях вывода (PlotRange→Full) показан на графике grDens1:

```
xMin := 5; xMax := 11; yMin := 0; yMax := 4;
aspRat = (yMax - yMin) / (xMax - xMin);
zXY[x_, y_] :=
  E^(-(x - 9)^2 - (y - 2)^2) - (2 / 3) * E^(-(x - 7)^2 - (3 * y - 3)^2);

grDens1 = DensityPlot[zXY[x, y],
  {x, xMin, xMax}, {y, yMin, yMax}, AspectRatio → aspRat,
  BaseStyle → 14, PlotRange → Full, ColorFunction → Hue,
  PlotLabel → Style["DensityPlot, Hue", 14, Blue],
  ImageSize → 550]
```

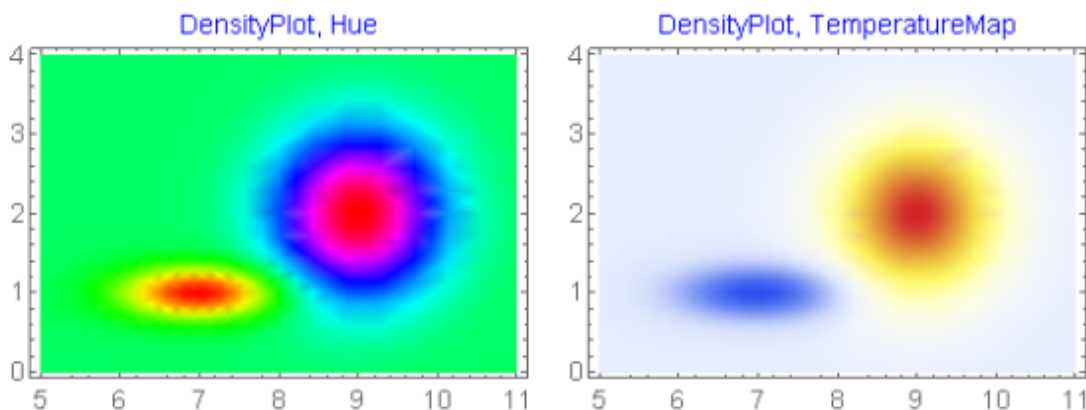


Следует обратить внимание, на искажения однородности раскраски (как их устранить пояснено ниже), которые могут быть не так заметны при использовании других цветовых схем; в частности проверены, можно считать искажения мало заметными: LakeColors, MintColors, NeonColors, PearlColors, PlumColors, RoseColors, SolarColors, SouthwestColors, StarryNightColors, SunsetColors, ThermometerColors, WatermelonColors, GreenPinkTones, PigeonTones, RedBlueTones, Pastel, Rainbow, TemperatureMap. Пример варианта с TemperatureMap иллюстрирует график grDens2:

```
grDens1d = Show[grDens1, ImageSize → 275];

grDens2 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → "TemperatureMap",
  PlotLabel → Style["DensityPlot, TemperatureMap", 14, Blue],
  ImageSize → 275];

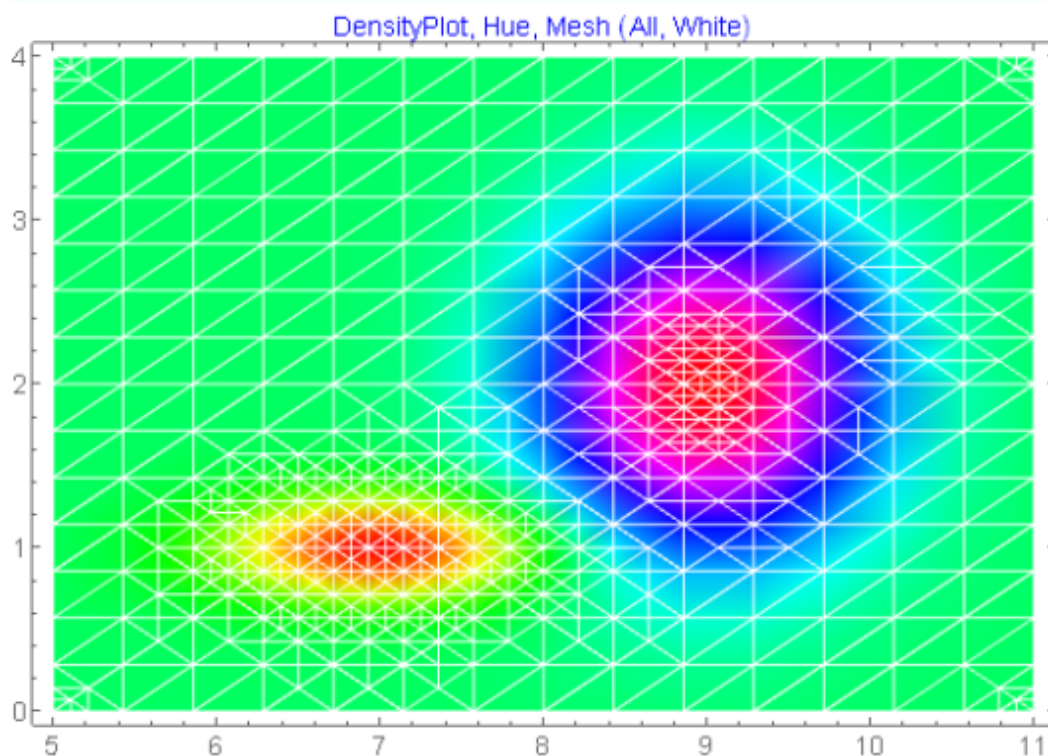
grDens2s = Row[{grDens1d, grDens2}, Spacer[10]]
```



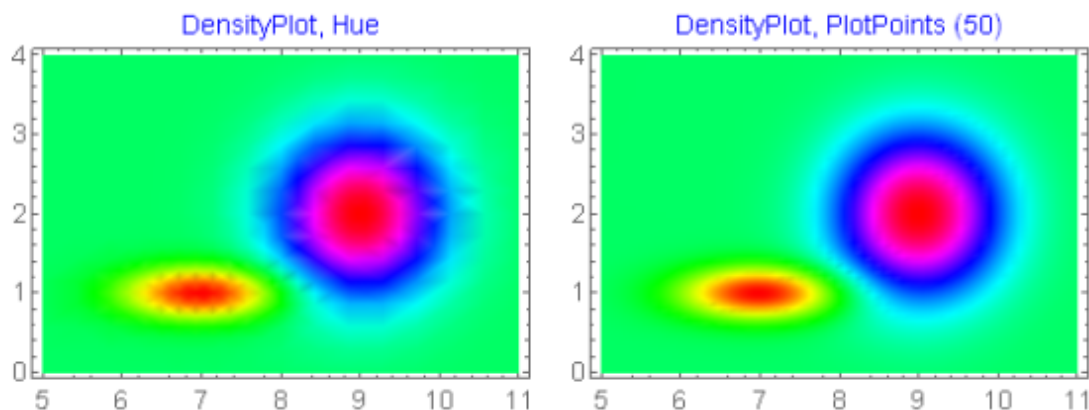
Также заметим, что отмеченные искажения присутствуют, хотя при формировании карты плотности в местах быстрого изменения функции имеет место детализация – система автоматически сгущает сетку. Это подтверждает график grDens3, на котором в дополнение к grDens1, чтобы отобразить принятую в расчете сетку, используется опция Mesh→All.

Для повышения качества графика можно использовать опции PlotPoints, MaxRecursion. Заметим, что их применение предполагает контрольные расчеты, так как неоправданное увеличение соответствующих значений приводит к существенному увеличению времени формирования графика. В приведенном примере grDens4 при числе точек меньше 45, искажения в отдельных местах имеются, увеличение более 45 заметного улучшения картинки не дает, но время расчета, начиная с 50, заметно больше:

```
grDens3 = DensityPlot[zXY[x, y],
  {x, xMin, xMax}, {y, yMin, yMax}, AspectRatio → aspRat,
  BaseStyle → 14, PlotRange → Full, ColorFunction → Hue,
  Mesh → All, MeshStyle → White, PlotLabel →
  Style["DensityPlot, Hue, Mesh (All, White)", 14, Blue],
  ImageSize → 550]
```



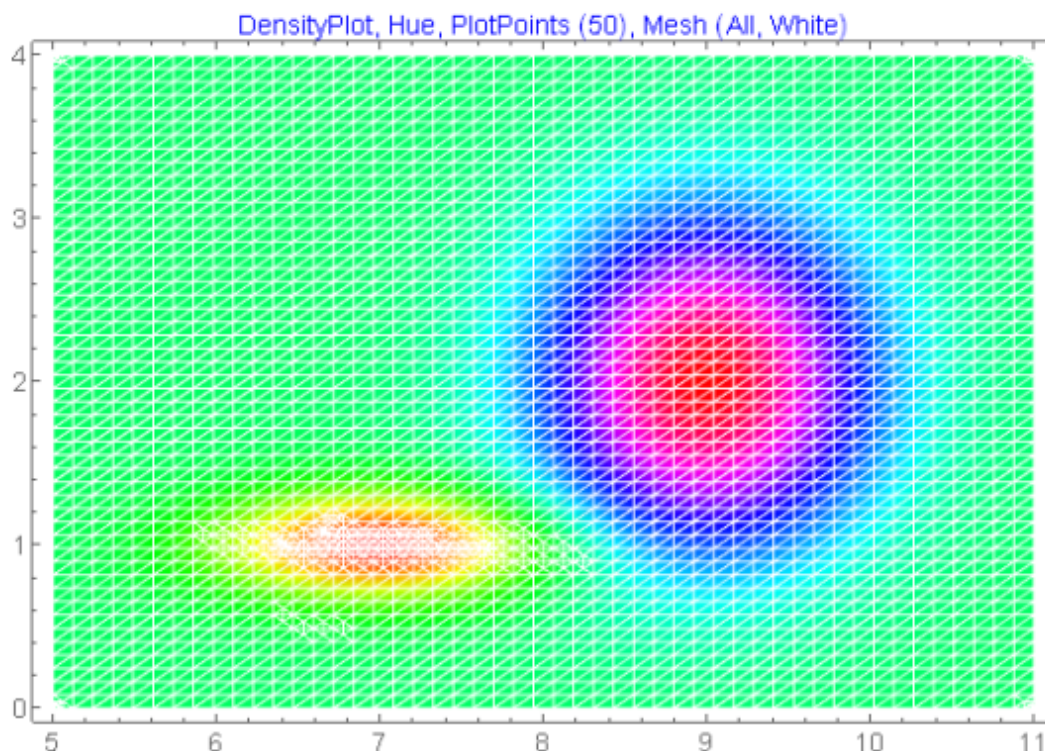
```
grDens4 = DensityPlot[zXY[x, y],
  {x, xMin, xMax}, {y, yMin, yMax}, AspectRatio → aspRat,
  BaseStyle → 14, PlotRange → Full, ColorFunction → Hue,
  PlotLabel → Style["DensityPlot, PlotPoints (50)", 14, Blue],
  PlotPoints → 50,
  ImageSize → 275];
grDens4s = Row[{grDens1d, grDens4}, Spacer[10]]
```



Приведенный ниже график grDens4d иллюстрирует эффект, как

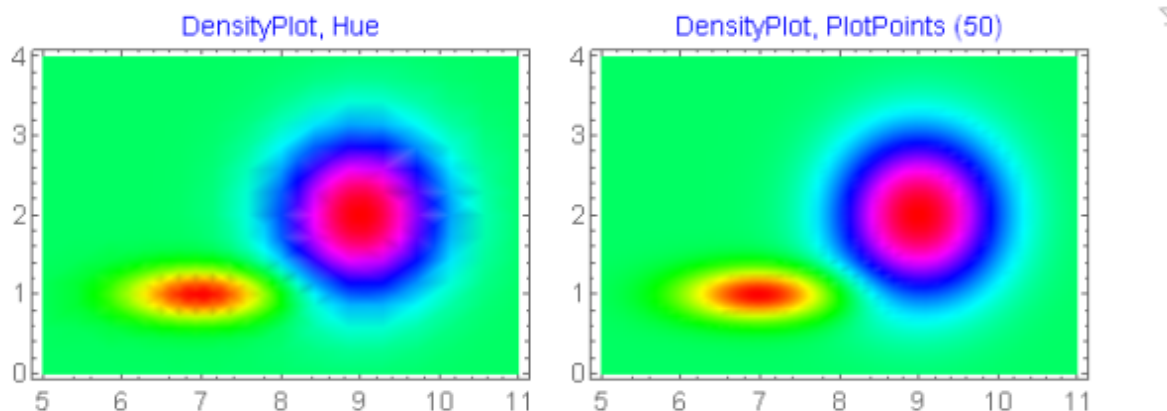
строятся ячейки при задании числа точек пользователем; причем, и в таком варианте система автоматически сгущает сетку (почему – поясняется ниже):

```
grDens4d = DensityPlot[zXY[x, y], {x, xMin, xMax},
  {y, yMin, yMax}, AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → Hue, Mesh → All,
  MeshStyle → White, PlotPoints → 50, PlotLabel →
  Style["DensityPlot, Hue, PlotPoints (50), Mesh (All, White)",
    14, Blue],
  ImageSize → 550]
```



Относительно эффекта задания `MaxRecursion` в рассматриваемом примере можно констатировать, что в вариантах с `PlotPoints` по умолчанию, качество картинки не повышается даже при `MaxRecursion`, равном 15, а большее значение система не допускает. Задание `MaxRecursion → 0` обеспечивает отключение сгущения ячеек. Следует одновременно задавать `PlotPoints` и `MaxRecursion`, но пробные расчеты показывают, что для рассматриваемого распределения $zXY(x, y)$ существенного влияния `MaxRecursion` нет.

```
grDens5 = DensityPlot[zXY[x, y],
  {x, xMin, xMax}, {y, yMin, yMax}, AspectRatio → aspRat,
  BaseStyle → 14, PlotRange → Full, ColorFunction → Hue,
  PlotLabel → Style["DensityPlot, PlotPoints (50)", 14, Blue],
  PlotPoints → 45, MaxRecursion → 9,
  ImageSize → 275];
grDens5s = Row[{grDens1d, grDens5}, Spacer[10]]
```

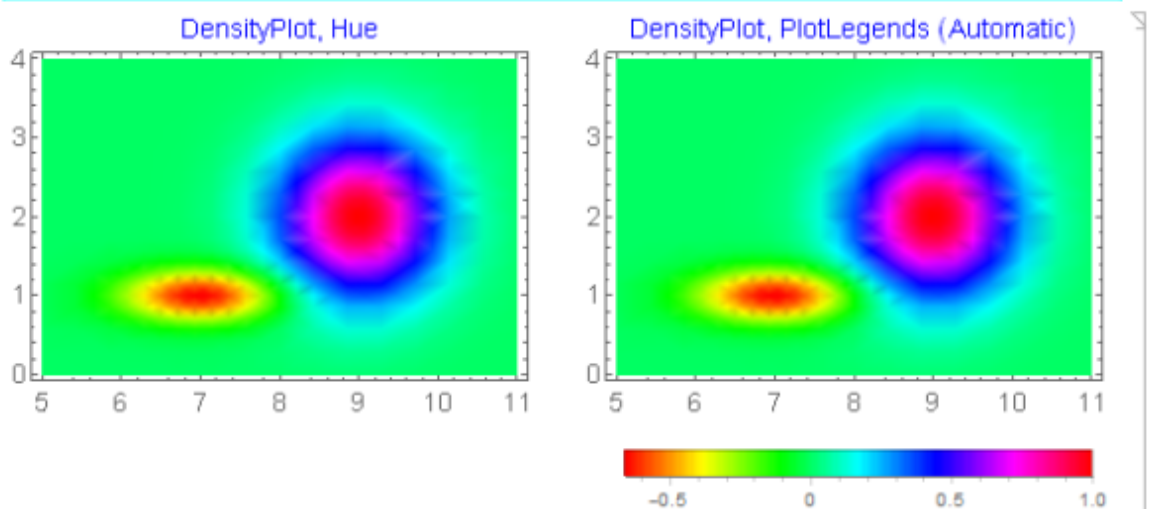



DensityPlot – оформление, детали, цветовые диапазоны

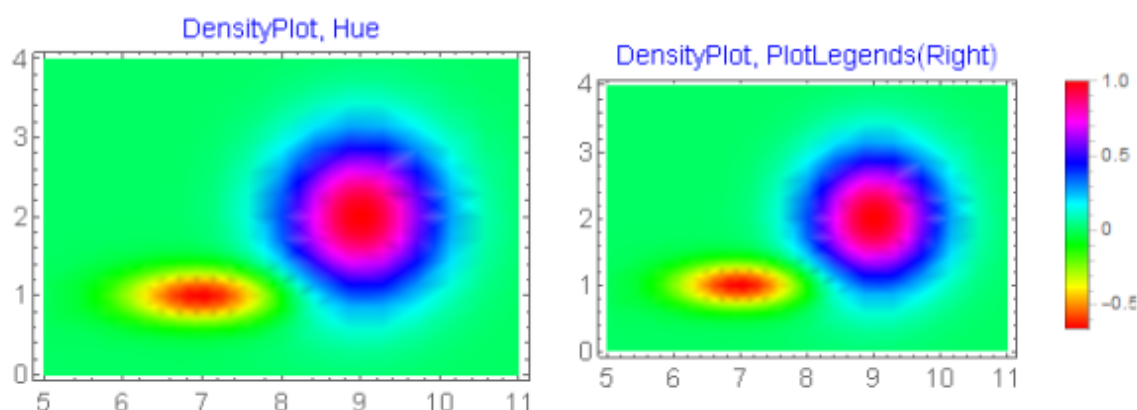
Одним из обязательных атрибутов карты плотности является легенда, ее вывод обеспечивает опция `PlotLegends` – особенности ее настройки, возможности позиционирования с нужной стороны карты приведены в материалах темы “Изолинии”. В частности, отмечены основные варианты размещения окна легенды: `Center`, `Top`, `Left`, `Right`, `Bottom`, поясняли примерами варианты включения заголовка, обрамления рамкой задаваемого вида, уточнения стиля подписей в легенде.

Примеры с вариантами позиционирования приведены на графиках `grDensL1` и `grDensL2` (положение легенды обеспечивает директива `Placed`):

```
grDensL1 =
  DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    AspectRatio → aspRat, BaseStyle → 14,
    PlotRange → Full, ColorFunction → Hue, PlotLabel →
      Style["DensityPlot, PlotLegends (Automatic)", 14, Blue],
    PlotLegends → Automatic, ImageSize → 275];
grDensL1s = Row[{grDensL1d, grDensL1}, Spacer[10]]
```



```
grDensL2 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → Hue,
  PlotLabel → Style["DensityPlot, PlotLegends(Right)", 14, Blue],
  PlotLegends → Placed[Automatic, Right], ImageSize → 235];
grDensL2s = Row[{grDens1d, grDensL2}, Spacer[10]]
```



Разные возможности дизайна пользователями цветовых схем предоставляет функция `ColorData`, которая включает множество настраиваемых наборов цветов, подключение которых можно реализовать непосредственно с опцией `ColorFunction`. Приведем несколько примеров. График `grDensC1` иллюстрирует способ инвертирования цветов на примере цветовой схемы `Hue`:

```
grDensC1 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → (Hue[1 - #1] &),
  PlotLabel → Style["ColorFunction(инвертирование Hue)",
    14, Blue], ImageSize → 275];
grDensC1s = Row[{grDens1d, grDensC1}, Spacer[10]]
```

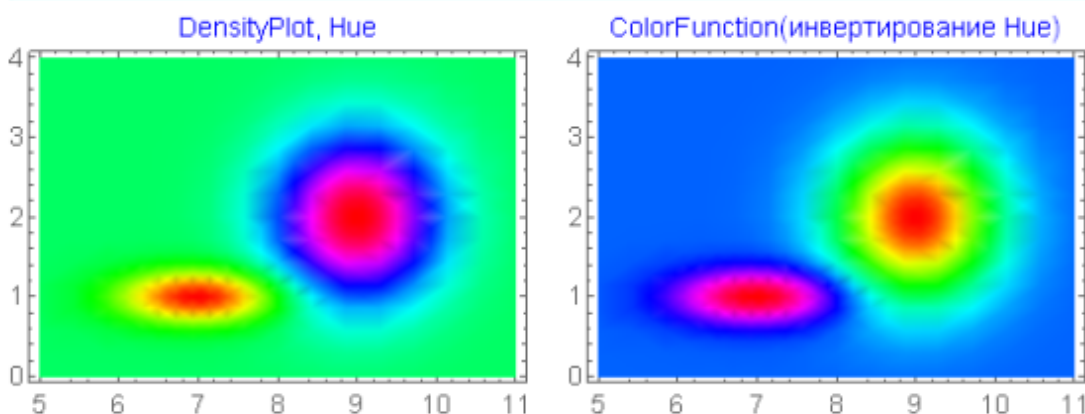


График `grDensC2` иллюстрирует способ, как используя приведенную выше функцию, меняя аргументы, можно задавать цветовые диапазоны

полного спектра радуги, стартуя с любого цвета (в примере – с зеленого):

```
grDensC2 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → (Hue[120 / 360 - #1] &),
  PlotLabel → Style["ColorFunction(Hue[120/360 - #1]",
    14, Blue], ImageSize → 275];
grDensC2s = Row[{grDens1d, grDensC2}, Spacer[10]]
```

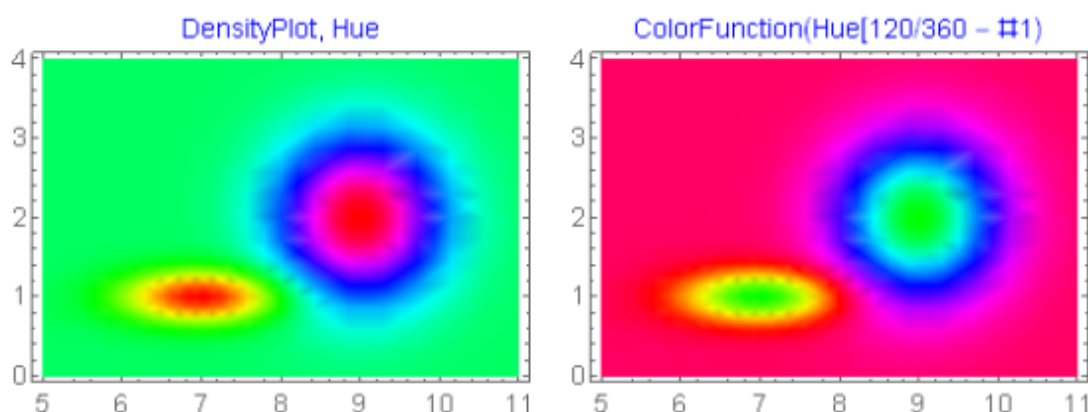


График grDensC3 иллюстрирует способ, как можно, стартуя с любого цвета (в примере – с зеленого), выводить 1/2 цветового диапазона:

```
grDensC3 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14, PlotRange → Full,
  ColorFunction → (Hue[120 / 360 - #1 / 2] &),
  PlotLabel → Style
    ["ColorFunction(Hue[120/360 - #1/2]", 14, Blue],
  ImageSize → 275];
grDensC3s = Row[{grDensC2, grDensC3}, Spacer[10]]
```

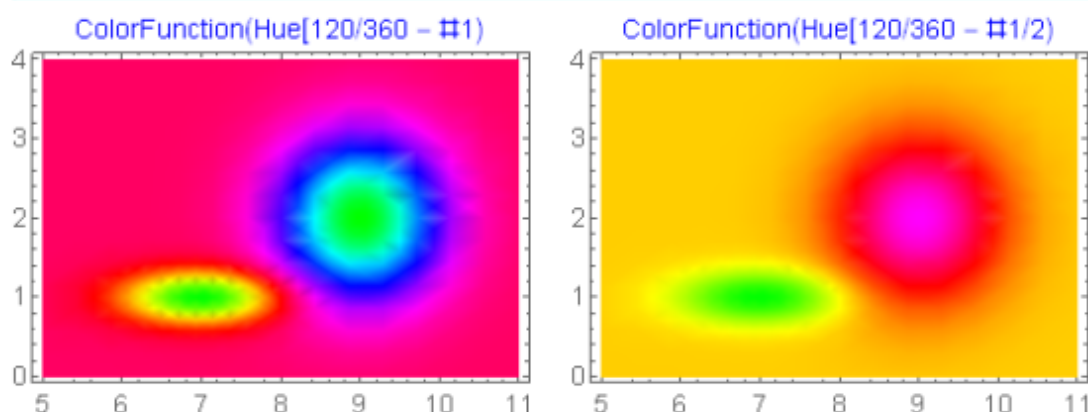
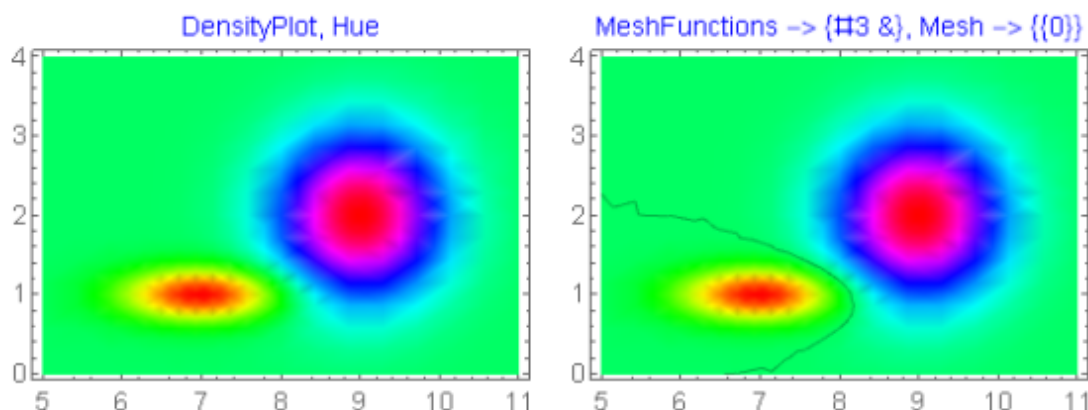


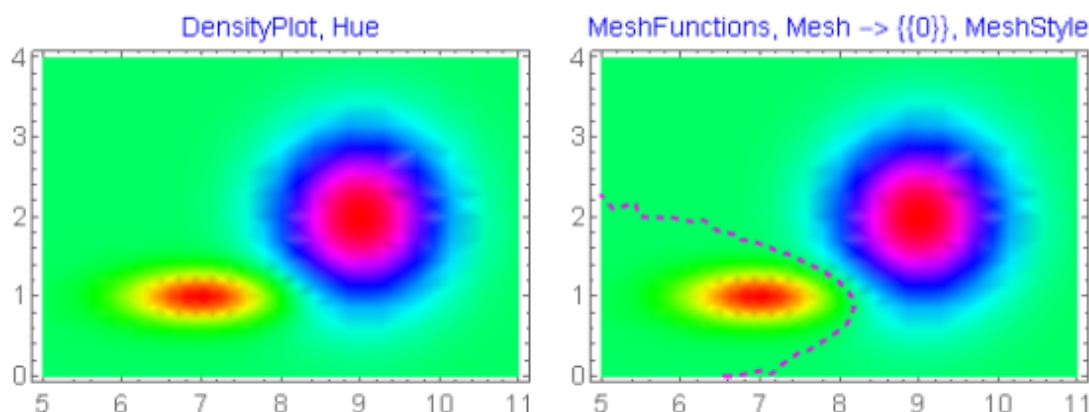
График grDensG1 иллюстрирует способ, как можно на карте плотности обозначить линию пересечения поверхности с плоскостью $z=0$:

```
grDensG1 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → Hue,
  MeshFunctions → {#3 &}, Mesh → {{0}},
  PlotLabel → Style
    ["MeshFunctions -> {#3 &}, Mesh -> {{0}}", 14, Blue],
  ImageSize → 275];
grDensG1s = Row[{grDens1d, grDensG1}, Spacer[10]]
```



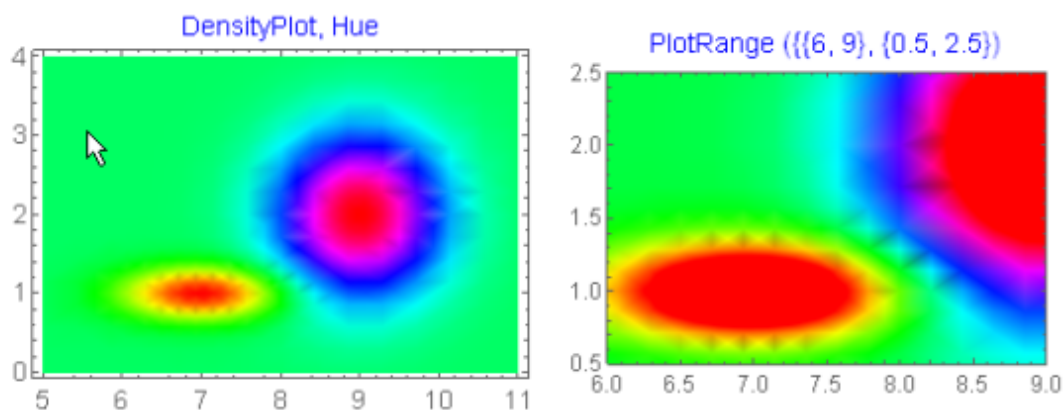
Пример уточнения стиля выводимой на карте плотности изолинии $z=0$ иллюстрирует график grDensG2:

```
grDensG2 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  AspectRatio → aspRat, BaseStyle → 14,
  PlotRange → Full, ColorFunction → Hue,
  MeshFunctions → {#3 &}, MeshStyle → {Magenta, Dashed, Thick},
  Mesh → {{0}},
  PlotLabel → Style["MeshFunctions, Mesh -> {{0}}, MeshStyle",
    14, Blue], ImageSize → 275];
grDensG2s = Row[{grDens1d, grDensG2}, Spacer[10]]
```



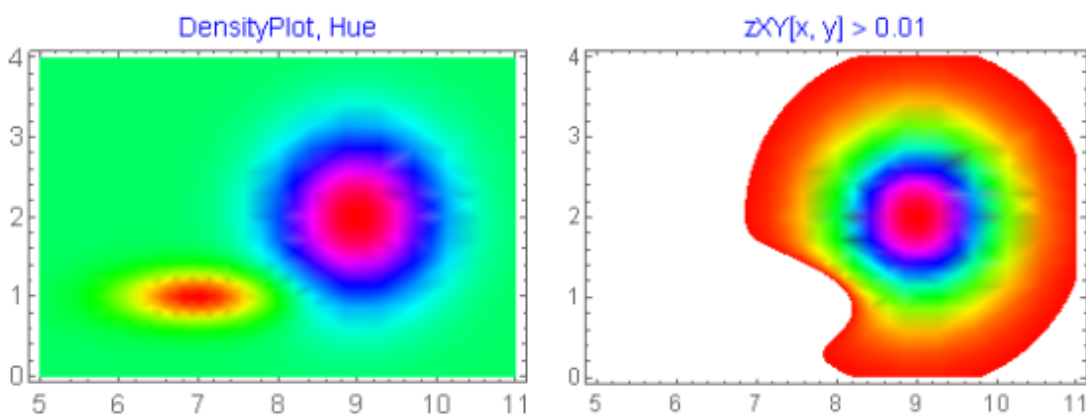
Как, используя опцию `PlotRange`, детализировать распределение на конкретном участке площади (чтобы сосредоточиться на представляющих интерес фрагментах карты плотности) иллюстрирует график grDensG3 :


```
grDensG3 = DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
  BaseStyle → 12, ColorFunction → Hue,
  PlotLabel →
    Style["PlotRange ({{6, 9}}, {0.5, 2.5})", 14, Blue],
  AspectRatio → (2.5 - 0.5) / (9 - 6),
  PlotRange → {{6, 9}, {0.5, 2.5}},
  PlotRange → Full, ImageSize → 260];
grDensG3s = Row[{grDens1d, grDensG3}, Spacer[10]]
```



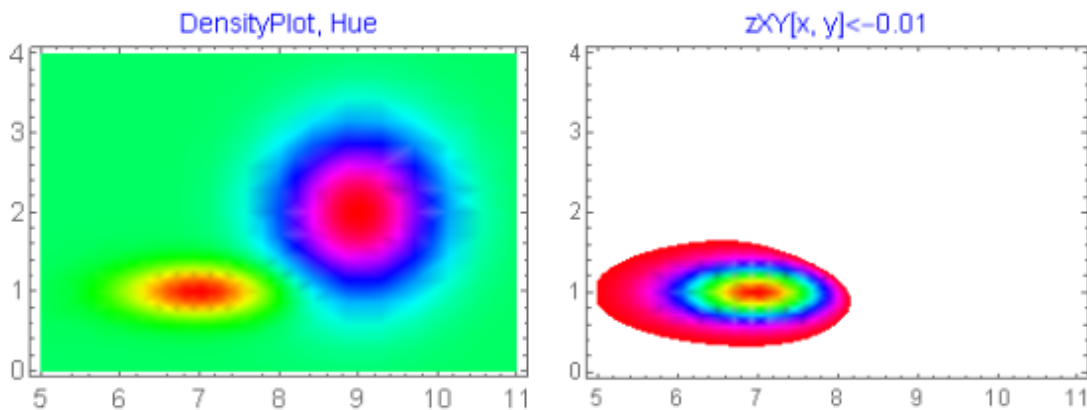
Варианты вывода только контролируемых участков (в примере ниже показаны части поверхности выше и ниже горизонтальной плоскости нулевого уровня) иллюстрируют графики grDensG4a и grDensG4b. Используется функция RegionFunction включением ограничений, задаваемых неравенствами:

```
grDensG4a =
  DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    BaseStyle → 12, ColorFunction → Hue,
    PlotLabel → Style["zXY[x, y] > 0.01", 14, Blue],
    AspectRatio → aspRat,
    RegionFunction → Function[{x, y, z}, zXY[x, y] > 0.01],
    ImageSize → 275];
grDensG4as = Row[{grDens1d, grDensG4a}, Spacer[10]]
```



```
grDensG4b =
  DensityPlot[zXY[x, y], {x, xMin, xMax}, {y, yMin, yMax},
    BaseStyle → 12, ColorFunction → Hue,
    PlotLabel → Style["zXY[x, y] < -0.01", 14, Blue],
    AspectRatio → aspRat,
    RegionFunction → Function[{x, y, z}, zXY[x, y] < -0.01],
    ImageSize → 275];

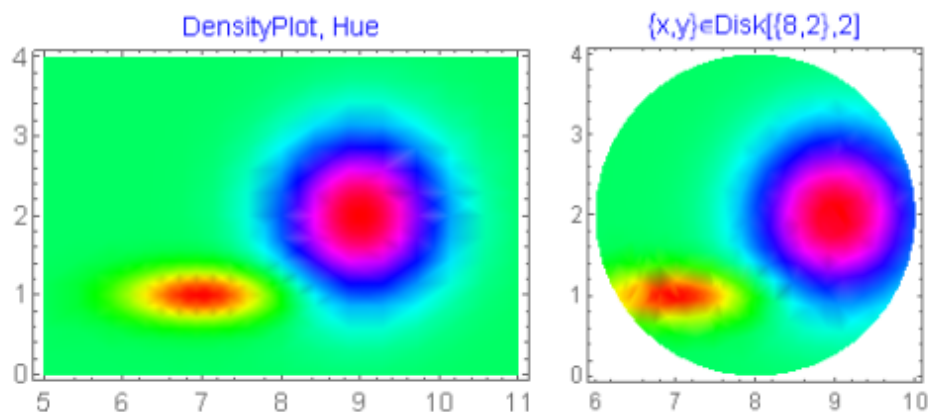
grDensG4bs = Row[{grDens1d, grDensG4b}, Spacer[10]]
```



Также можно представлять изображение в конкретном регионе. В примере grDensG5 таковым является диск (может быть определенный заданием границы на плоскости любой геометрический объект):

```
grDensG5 = DensityPlot[zXY[x, y], {x, y} ∈ Disk[{8, 2}, 2],
  BaseStyle → 12, ColorFunction → Hue,
  PlotLabel → Style["{x,y} ∈ Disk[{8,2},2]", 14, Blue],
  AspectRatio → 1,
  ImageSize → 190];

grDensG5s = Row[{grDens1d, grDensG5}, Spacer[10]]
```



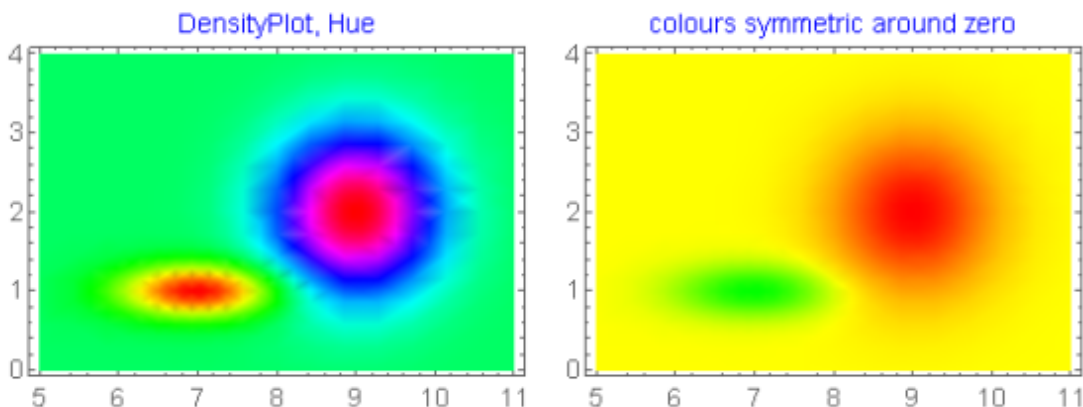
В примере grDensF1 реализован вывод, когда цвета графика плотности симметричны вокруг нуля. Для этого дополнительно вычисляются минимальное и максимальное значения $zXY(x,y)$ в рассматриваемой

области. Используются функции `NMinimize` и `NMaximize`, функция `Blend` дает цвет, получаемый путем смешивания долей первого и второго указанных цветов, `ColorFunctionScaling` (нормировать аргумент функции отсвечивания) определяет, следует ли масштабировать аргументы:

```
zXYmin =
  NMinimize[{zXY[x, y], xMin < x < xMax, yMin < y < yMax}, {x, y}];
zXYmax =
  NMaximize[{zXY[x, y], xMin < x < xMax, yMin < y < yMax}, {x, y}];

grDensF1 = DensityPlot[zXY[x, y],
  {x, xMin, xMax}, {y, yMin, yMax}, AspectRatio → aspRat,
  BaseStyle → 14, PlotRange → Full, ColorFunction →
    (Blend[{zXYmin[[1]], Green}, {0, Yellow},
      {zXYmax[[1]], Red}], #) &,
  ColorFunctionScaling → False, PlotLabel → Style
    ["colours symmetric around zero", 14, Blue],
  ImageSize → 275];

grDensF1s = Row[{grDens1d, grDensF1}, Spacer[10]]
```

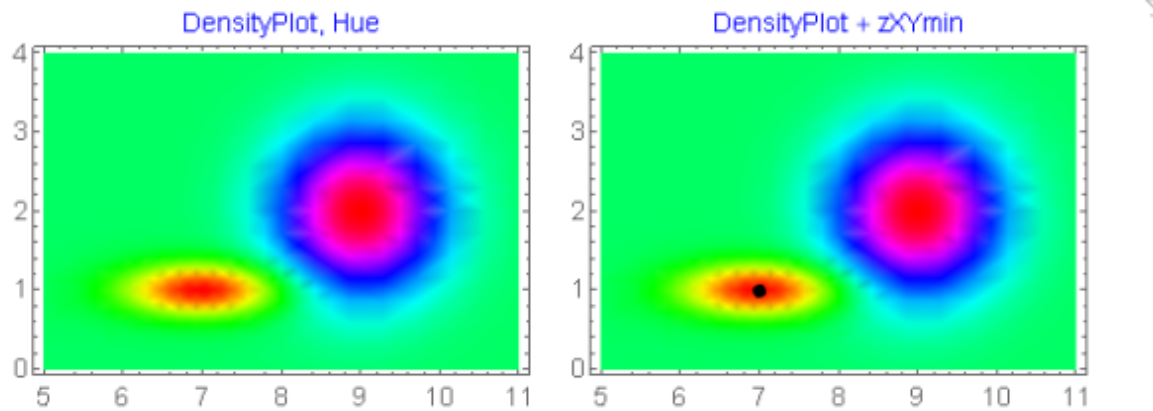


В примере `grDensF2` реализован вывод точки, в которой достигается минимум визуализируемой функции $zXY(x,y)$ в рассматриваемой области. Используются функции `NMinimize`, функция `ReplaceAll` применен вариант краткой записи `/.` дает подстановку (передаются координаты точки минимума), `Point` – используемый геометрический примитив:

```
zXYmin =
  NMinimize[{zXY[x, y], xMin < x < xMax, yMin < y < yMax}, {x, y}];

grDensF2 = Show[grDens1d, Graphics[
  {Black, PointSize[Large], Point[{x, y} /. Last[%]]},
  PlotLabel → Style["DensityPlot + zXYmin", 14, Blue]];

grDensF2s = Row[{grDens1d, grDensF2}, Spacer[10]]
```



Тестовые задания для самоконтроля

Упражнение КП.1.

2D графика системы *Mathematica*. Визуализация функций.

Приведен скриншот графиков.

Внимание. Здесь 2 задания, ответы на которые надо вводить, разделяя их запятыми.

1) Что следует вписать вместо Saaaa для получения приведенного графика слева?

$fGr[x, y] = \sin[x + y^2]$; $xmin = -\pi$; $xmax = \pi$; $ymin = -\pi$; $ymax = \pi$;

$grf1 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 275, \text{PlotLabel} \rightarrow \text{"grf1"}, \text{ColorFunction} \rightarrow \text{Hue}]$;

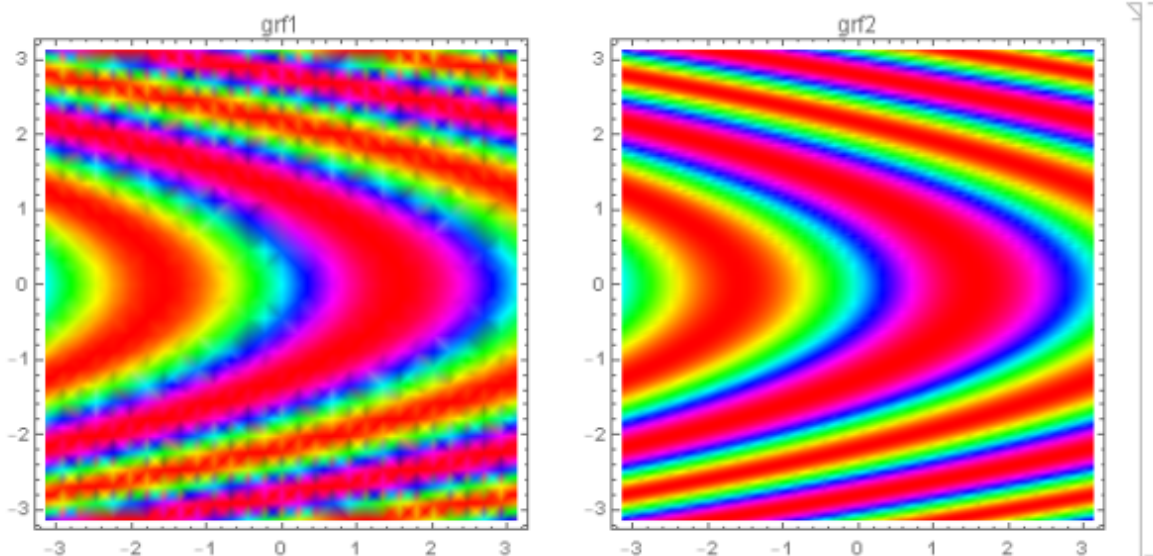
2) Что следует вписать вместо Sbbbbb для получения приведенного графика справа?

$grf2 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 275, \text{PlotLabel} \rightarrow \text{"grf2"}, \text{Sbbbbb} \rightarrow \{50, 99\}, \text{ColorFunction} \rightarrow \text{Hue}]$;

$\text{Row}[\{grf1, grf2\}, \text{Spacer}[20]]$

Подсказки: Ответы надо вводить, разделяя их запятыми. Символов в ответе - 22.

Пробелов в ответе быть не должно. Если ответ не знаете, введите 0



Упражнение КП.2.

2D графика системы *Mathematica*. Визуализация функций.

Приведен скриншот графиков.

Внимание. Здесь 2 задания, ответы на которые надо вводить, разделяя их запятыми.

1) Что следует вписать вместо Saaaa для получения приведенного графика слева?

$fGr[x_, y_] = \sin[x + y^2]$; $xmin = -\pi$; $xmax = \pi$; $ymin = -\pi$; $ymax = \pi$;

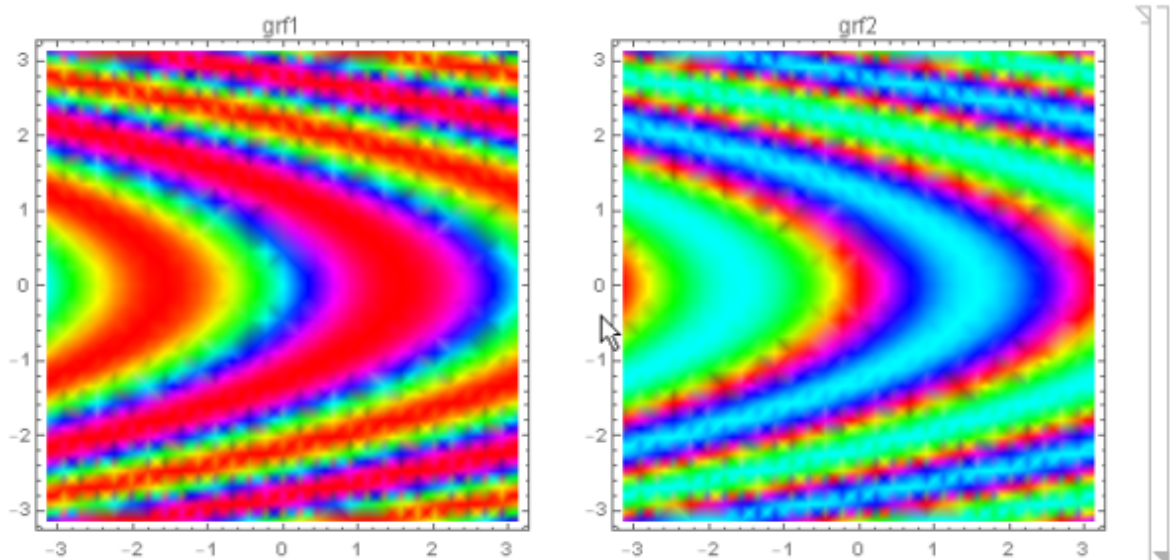
$grf1 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 275, \text{PlotLabel} \rightarrow \text{"grf1"}, \text{ColorFunction} \rightarrow \text{Hue}]$;

2) Что следует вписать вместо Sbbbbb для получения приведенного графика справа?

$grf2 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 275, \text{PlotLabel} \rightarrow \text{"grf2"}, \text{ColorFunction} \rightarrow \text{Sbbbbb}]$;

$\text{Row}\{grf1, grf2\}, \text{Spacer}[20]$

Подсказки: Ответы надо вводить, разделяя их запятыми. Символов в ответе - 25. Пробелов в ответе быть не должно. Если ответ не знаете, введите 0



Упражнение КП.3.

2D графика системы *Mathematica*. Визуализация функций.

Приведен скриншот графиков.

Внимание. Здесь 2 задания, ответы на которые надо вводить, разделяя их запятыми.

1) Что следует вписать вместо Saaaa для получения приведенного графика слева?

$fGr[x_, y_] = \sin[x + y^2]$; $xmin = -\pi$; $xmax = \pi$; $ymin = -\pi$; $ymax = \pi$;

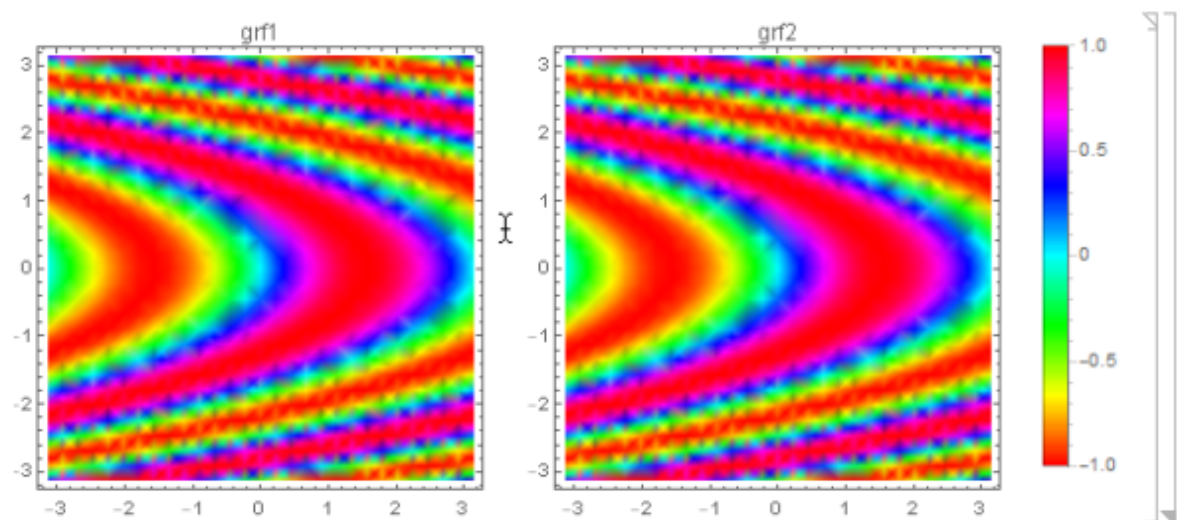
$grf1 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 250, \text{PlotLabel} \rightarrow \text{"grf1"}, \text{ColorFunction} \rightarrow \text{Hue}]$;

2) Что следует вписать вместо Sbbbbb для получения приведенного графика справа?

$grf2 = \text{Saaaa}[fGr[x, y], \{x, xmin, xmax\}, \{y, ymin, ymax\}, \text{ImageSize} \rightarrow 250, \text{PlotLabel} \rightarrow \text{"grf2"}, \text{ColorFunction} \rightarrow \text{Hue}, \text{Sbbbbb} \rightarrow \text{Automatic}]$;

$\text{Row}\{grf1, grf2\}, \text{Spacer}[10]$

Подсказки: Ответы надо вводить, разделяя их запятыми. Символов в ответе - 23. Пробелов в ответе быть не должно. Если ответ не знаете, введите 0



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.
2. How to automatically have ContourLabels in middle of Contour lines if specific levels are not specified? [Электронный ресурс]. URL : <https://mathematica.stackexchange.com/questions/103289/how-to-automatically-have-contourlabels-in-middle-of-contour-lines-if-specific-l?noredirect=1&lq=1>

Содержание

ПРЕДИСЛОВИЕ	3
Материалы темы "Изолинии"	4
Общее о 2D визуализации функций	4
Изолинии. Определения	6
Аналитическая функция, используемая в примерах	8
Функция ContourPlot. Основные опции	9
Изолинии. Вывод по умолчанию, масштаб по осям	10
Функция ContourPlot. Примеры задания атрибутов изолиний	12
ContourPlot. Варианты режима цветного вывода	17
Оформление подписей уровней, рамки окна вывода	19
Оформление контурной графики индивидуальной цветовой схемой	23
Изолинии, варианты вывода легенды (принятой шкалы цветов)	27
Карты изолиний, примеры специальных настроек	30
Тестовые задания для самоконтроля	34
Материалы темы "Карты плотности"	36
Визуализация функции двух переменных средствами DensityPlot	36
DensityPlot – оформление, детали, цветовые диапазоны	41
Тестовые задания для самоконтроля	48
Рекомендуемая литература	51

Учебное издание

Таранчук Валерий Борисович

**ДВУМЕРНАЯ ГРАФИКА
СИСТЕМЫ МАТЕМАТИКА.
ВИЗУАЛИЗАЦИЯ ФУНКЦИЙ**

**Учебные материалы для студентов
факультета прикладной математики
и информатики**

В авторской редакции

Ответственный за выпуск *В. Б. Таранчук*

Подписано в печать 30.12.2022. Формат 60×84/16. Бумага офсетная.
Усл. печ. л. 3,02. Уч.-изд. л. 2,94. 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 с оригинал-макета заказчика
на копировально-множительной технике
факультета прикладной математики и информатики
Белорусского государственного университета.
Пр. Независимости, 4, 220030, Минск.