



Mathematica для математиков.

Представлен параграф 2.4 «Графические иллюстрации к решению прикладных задач» нашего пособия «Mathematica для математиков». Здесь вы можете глубже познакомиться с применением графических функций системы при иллюстрации решений разнообразных прикладных задач.

Оглавление

2.4 Графические иллюстрации к решению прикладных задач.....	2
2.4.1 Несколько геометрических примеров.....	2
2.4.2 Моделирование механических движений	5
2.4.3 Движение жидкости в трубах	12
2.4.4 Двумерные волновые движения	16
2.4.5 Одномерные волновые колебания.....	22
2.4.6 Задачи сопротивления материалов и теории упругости	35
2.4.7 Задачи электростатики.....	44
Литература.	49

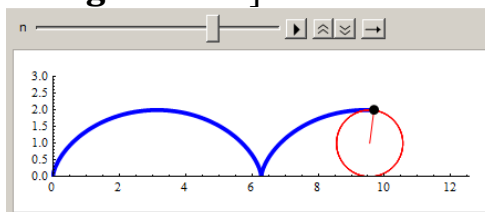
2.4 Графические иллюстрации к решению прикладных задач

В этом разделе собраны примеры, демонстрирующие возможности графических функций системы *Mathematica* для представления решений прикладных задач. При этом мы не даем подробных постановок задач и, как правило, приводим только решение, графическое представление которого нас интересует. Если «физическая» постановка задачи вам неясна, то вы можете пропустить пример.

2.4.1 Несколько геометрических примеров

Пример 1.1. Циклоида. Траектория фиксированной точки окружности, которая катится по прямой, называется циклоидой. Если радиус окружности равен 1 и скорость движения ее центра $v=1$, то параметрическое уравнение циклоиды будет иметь вид $x(t)=t-\sin t$, $y(t)=1-\cos t$. Следующий код строит анимацию качения окружности, движение фиксированной точки на окружности, и заметаемую точкой траекторию (циклоиду).

```
Animate[
  p1 = ParametricPlot[{t - Sin[t], 1 - Cos[t]}, {t, 0, n}
    PlotRange -> {{0, 4π}, {0, 3}}, PlotStyle -> {Blue, Thickness[0.01]};
  p2 = Graphics[{Red, Line[{n - Sin[n], 1 - Cos[n]}, {n, 1}]}];
  p3 = ParametricPlot[{n - Sin[n * t], 1 - Cos[n * t]}, {t, 0, 2π},
    PlotStyle -> {Red}];
  p4 = Graphics[Disk[{n - Sin[n], 1 - Cos[n]}, 0.15]];
  Show[p1, p2, p3, p4],
  {n, 1, 4π}, AnimationRunning -> False]
```



Пример 1.2. Лента Мебиуса – простейшая неориентируемая поверхность. Она получается движением и вращением отрезка прямой вдоль замкнутой пространственной кривой. Пусть эта кривая будет окружностью радиуса R , а n обозначает количество полуоборотов отрезка при обходе кривой. Тогда уравнение поверхности можно записать в параметрическом виде

$$x(u, v) = \left(R + v \cos\left(\frac{nu}{2}\right) \right) \cos u$$

$$y(u, v) = \left(R + v \cos\left(\frac{nu}{2}\right) \right) \sin u$$

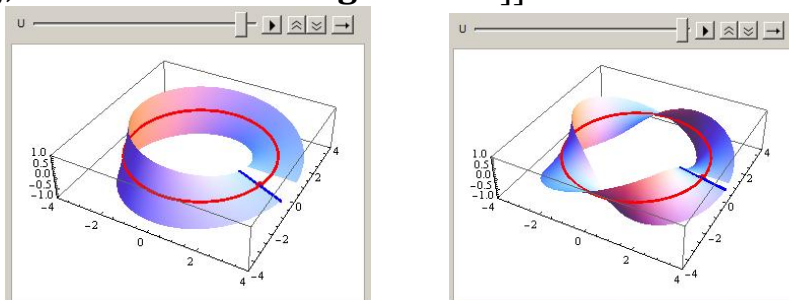
$$z(u, v) = v \sin\left(\frac{nu}{2}\right)$$

Вдоль окружности движется центр отрезка. Следующий код строит анимацию его движения и заметаемую им поверхность (лента Мебиуса).

```

DynamicModule[{R = 3, H = 2, n = 1, x, y, z, po, pt, pc, ps},
  x[u_, v_] = (R + vCos[ $\frac{nu}{2}$ ])Cos[u];
  y[u_, v_] = (R + vCos[ $\frac{nu}{2}$ ])Sin[u];
  z[u_, v_] = vSin[ $\frac{nu}{2}$ ];
  po = ParametricPlot3D[{x[u, 0], y[u, 0], z[u, 0]}, {u, 0, 2 $\pi$ },
    PlotStyle → {Red, Thickness[0.01]},
    PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}];
  Animate[
    pt = Graphics3D[
      {Red, PointSize[0.03], Point[{x[U, 0], y[U, 0], z[U, 0]}]};
    pc = ParametricPlot3D[{x[U, v], y[U, v], z[U, v]}, {v, -H/2, H/2},
      PlotStyle → {Blue, Thickness[0.01]},
      PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}];
    ps = ParametricPlot3D[{x[u, v], y[u, v], z[u, v]}, {u, 0, U}, {v, -H/2, H/2},
      AspectRatio → Automatic, Mesh → None, PlotPoints → {50, 15},
      PlotRange → {{-4, 4}, {-4, 4}, {-1, 1}}];
    Show[pc, ps, po, pt],
    {U, 0.01, 2 $\pi$ }, AnimationRunning → False]]

```



Здесь H обозначает ширину ленты (длину отрезка). При нечетном n получается неориентированная поверхность. На предыдущем рисунке слева показана лента Мебиуса, построенная при $n=1$, а справа – при $n=3$.

Заметим, что картинка ленты Мебиуса имеется в наборе примеров геометрических поверхностей, которую можно построить командой

```
ExampleData[{"Geometry3D", "MoebiusStrip"}]
```

□

Пример 1.3. Бутылка Клейна из лоскутов поверхностей.

Составим поверхность бутылки Клейна из 4-х лоскутов/поверхностей, параметрические уравнения которых достаточно просты.

Строим первый лоскут (следующий рисунок *a*)

$a = 2.5; b = 1.5;$

$x1[u_, v_] = (a + bCos[u])Cos[v];$

$y1[u_, v_] = (a + bCos[u])Sin[v];$

$z1[u_, v_] = -aSin[u];$

$p1 = ParametricPlot3D\{x1[u, v], y1[u, v], z1[u, v]\}, \{u, 0, \pi\}, \{v, 0, 2\pi\},$

PlotStyle → Opacity[0.5], Mesh → None]

Строим второй лоскут (следующий рисунок *b*)

```

x2[u_, v_] = (a + bCos[u])Cos[v];
y2[u_, v_] = (a + bCos[u])Sin[v];
z2[u_, v_] = 3u;
p2 = ParametricPlot3D[{x2[u, v], y2[u, v], z2[u, v]}, {u, 0,  $\pi$ }, {v, 0,  $2\pi$ },
PlotStyle -> Opacity[0.5], Mesh -> None]

```

Строим третий лоскут (следующий рисунок *c*)

```

x3[u_, v_] = 2 - 2Cos[u] + Sin[v];
y3[u_, v_] = Cos[v];
z3[u_, v_] = 3u;
p3 = ParametricPlot3D[{x3[u, v], y3[u, v], z3[u, v]}, {u, 0,  $\pi$ }, {v, 0,  $2\pi$ },
PlotStyle -> Opacity[0.5], Mesh -> None]

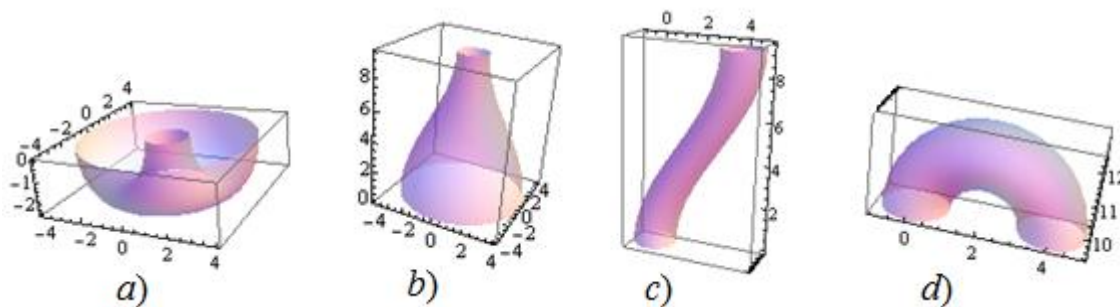
```

Строим четвертый лоскут (следующий рисунок *d*)

```

x4[u_, v_] = 2 + (2 + Cos[v])Cos[u];
y4[u_, v_] = Sin[v];
z4[u_, v_] = 3 $\pi$  + (2 + Cos[v])Sin[u];
p4 = ParametricPlot3D[{x4[u, v], y4[u, v], z4[u, v]}, {u, 0,  $\pi$ }, {v, 0,  $2\pi$ },
PlotStyle -> Opacity[0.5], Mesh -> None]

```



Теперь рисуем все сегменты/куски поверхности вместе.

```

Show[p1, p2, p3, p4, PlotRange -> All, Boxed -> False, Axes -> None]

```



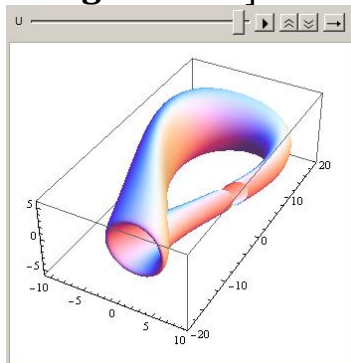
Пример 1.4. Бутылка Клейна. Известны примеры параметрических уравнений поверхностей, которые представляют бутылку Клейна. Здесь мы используем одно из таких уравнений.

```

Animate[ParametricPlot3D[
{6Cos[u](1 + Sin[u]) + 4(1 -  $\frac{\text{Cos}[u]}{2}$ )Cos[ $\frac{u}{2} + \frac{\pi}{4}$ ]Cos[v],
16Sin[u] + 4(1 -  $\frac{\text{Cos}[u]}{2}$ )Sin[ $\frac{u}{2} + \frac{\pi}{4}$ ]Cos[v], 4(1 -  $\frac{\text{Cos}[u]}{2}$ )Sin[v]}

```

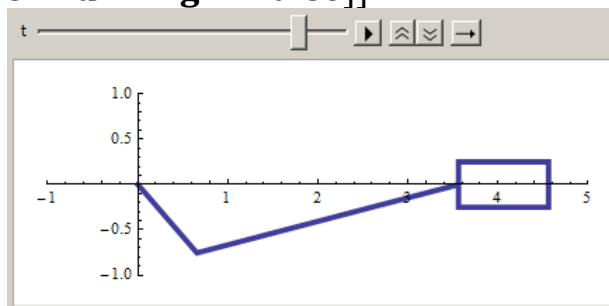
```
{u, 0, U}, {v, 0, 2π}, PlotRange → {{-11, 10}, {-20, 20}, {-6, 6}},
PlotPoints → 30, Mesh → None, ImageSize → 280],
{U, 0.01, 2π}, AnimationRunning → False]
```



2.4.2 Моделирование механических движений

Пример 2.1. Кривошипно – шатунный механизм, является устройством, которое вращательное движение преобразует в поступательное или наоборот. Смоделируем схематически движение этого механизма, изобразив его в виде ломаной.

```
Clear[xl, yl, pt]; w = 1;
xll = {0, Cos[wt], Sqrt[9 - Sin[wt]^2], 0, 1, 0, -1, 0};
yll = {0, Sin[wt], -Sin[wt], -1/4, 0, 1/2, 0, -1/4};
xl = Accumulate[xll];
yl = Accumulate[yll];
DynamicModule[{pt},
  pt[t_] = Transpose[{xl, yl}];
  Animate[ListLinePlot[pt[t],
    AspectRatio → Automatic, PlotRange → {{-1, 5}, {-1, 1}},
    PlotStyle → Thickness[0.01]],
    {t, 0, 2π}, AnimationRunning → False]]
```



Здесь списки **xll** и **yll** содержат длины проекций отрезков ломаной, представляющей механизм, на оси X и Y соответственно. Функция **Accumulate[List]** возвращает список накопительных сумм элементов списка **List**. Например,

```
Accumulate[{a, b, c, d}]
{a, a + b, a + b + c, a + b + c + d}
```

В результате списки **xl** и **yl** содержат x и y координаты узлов ломаной. Из них создается список **pt[t]** пар координат $\{\dots, \{x_i, y_i\}, \dots\}$ узлов, которые используются функцией `ListLinePlot` для рисования ломаной.

Пример 2.2. Поперечные колебания невесомой струны с грузами.

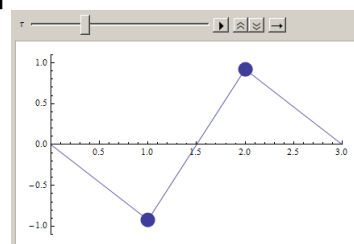
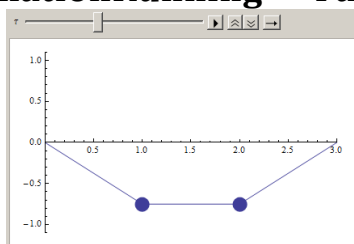
Рассмотрим колебания механической системы, которая состоит из двух частиц, каждая массой M , закрепленных на невесомой упругой струне. Массы разбивают струну на три участка одинаковой длины. Через y_1 обозначим вертикальное смещение 1-й массы и через y_2 – смещение 2-й массы. Известно, что колебание системы может быть представлено в виде [1]

$$\begin{aligned} y_1(t) &= A_1 \cos(\omega_1 t + \varphi_1) + A_2 \cos(\omega_2 t + \varphi_2), \\ y_2(t) &= A_1 \cos(\omega_1 t + \varphi_1) - A_2 \cos(\omega_2 t + \varphi_2), \end{aligned}$$

где значения постоянных определяется из начальных условий и механических характеристик системы.

Рассмотрим колебание системы без начальной скорости (начальные скорости частиц равны нулю). Длина струны равна 3. Колебание, при котором массы двигаются синхронно, называется первой модой. Следующий код моделирует первую моду колебаний (следующий рисунок слева).

```
DynamicModule[{w1 = 1, y1, y2, ptpt, pt},
  y1[t_] = Cos[w1 t];
  y2[t_] = Cos[w1 t];
  Animate[
    ptpt = {{1, y1[t]}, {2, y2[t]}};
    pt = {{0, 0}, {1, y1[t]}, {2, y2[t]}, {3, 0}};
    p1 = ListLinePlot[pt/. t -> \tau, PlotRange -> {{0, 3}, {-1.1, 1.1}}];
    p2 = ListPlot[ptpt/. t -> \tau, PlotStyle -> PointSize[0.05]];
    Show[p1, p2],
    {\tau, 0, 6}, AnimationRunning -> False]]
```



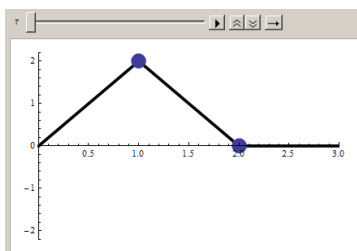
Вторая мода колебаний представляет движение частиц в противофазе. Чтобы построить такое движение в предыдущем коде мы должны изменить знак у функции y_2 и, возможно, изменить угловую частоту. Первые три строки предыдущего кода примут новый вид

```
DynamicModule[{w2 = \sqrt{3}, y1, y2, ptpt, pt},
  y1[t_] = Cos[w2 t];
  y2[t_] = -Cos[w2 t];
  ...
]
```

Остальные строки кода остаются без изменений. В результате получается движение, представленное в панели анимации предыдущего рисунка справа.

Смешанное колебание состоит из суперпозиции обеих мод, вклад каждой из которых определяется начальным положением масс системы. Построим колебание, начальное положение масс которого показано на следующем рисунке. Для этого изменим первые три строки кода следующим образом

```
DynamicModule{w1 = 1, w2 =  $\sqrt{3}$ , y1, y2, ptpt, pt},  
  y1[t_] = Cos[w1t] + Cos[w2t];  
  y2[t_] = Cos[w1t] - Cos[w2t];  
  ...  
]
```



Пример 2.3. Математический бильярд в прямоугольнике.

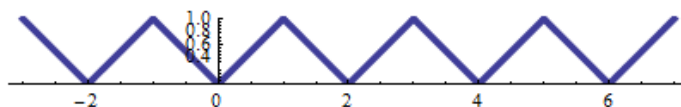
Имеется горизонтальный бильярдный стол произвольной формы без луз (плоская область). По этому столу без трения движется точечный шар, упруго отражаясь от бортов. Требуется определить траекторию этого шара/точки. Такая механическая система – точечный шар в области Q , ограниченной бортом Γ (границей области Q) – называется математическим бильярдом. Траектория точки в математическом бильярде определяется ее начальным положением и начальным вектором скорости.

Введем вспомогательную функцию $stc(x, w)$, которая будет обозначать пилообразную непрерывную (saw tooth continuous) кусочно-линейную функцию, определяемую, например, уравнением

$$stc(x, w) = \left| x - w \left[\frac{x}{w} + \frac{1}{2} \right] \right| \quad \text{или} \quad stc(x, w) = \frac{w}{2\pi} \arccos \cos \frac{2\pi x}{w},$$

где $[z]$ (квадратные скобки) является функцией взятия наибольшего целого, не превосходящего z .

```
Stc[x_, w_] = Abs[x - w Floor[x/w + 1/2]];  
Plot[Stc[x, 2], {x, -3, 7}, , AspectRatio → Automatic]
```



Приведем без доказательства некоторые факты, связанные с преобразованием отражения кривых.

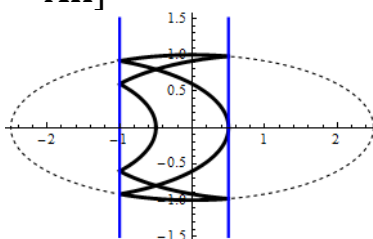
Пусть кривая C задана параметрическими уравнениями $x = x(t)$, $y = y(t)$. Уравнение кривой C_t , полученной многократным внутренним отражением от пары вертикальных прямых $x = x_l$, $x = x_r$ ($x_l < x_r$), имеет вид

$x_t = x_l + stc(x(t) - x_l, 2(x_r - x_l))$, $y_t = y(t)$. Вот пример кривой, полученной отражением эллипса от двух вертикальных прямых.

```

xl = -1; xr = 1/2;
x[t_] = 2.5Cos[t];
y[t_] = Sin[t];
xt[t_] = xl + Stc[x[t] - xl, 2(xr - xl)];
yt[t_] = y[t];
pt = ParametricPlot[{xt[t], yt[t]}, {t, 0, 2π}];
pe = ParametricPlot[{x[t], y[t]}, {t, 0, 2π}];
pl = Graphics[{Blue, Thick,
    {Line[{xl, -1.5}, {xl, 1.5}], Line[{xr, -1.5}, {xr, 1.5}]}];
Show[pe, pt, pl, PlotRange → All]

```



Аналогично, уравнение кривой C_t , полученной многократным внутренним отражением C от пары горизонтальных прямых $y = y_d$, $y = y_u$ ($y_d < y_u$) имеет вид $x_t = x(t)$, $y_t = y_d + stc(y(t) - y_d, 2(y_u - y_d))$.

Параметрическое уравнение кривой C_t , полученной многократным внутренним отражением C от границ прямоугольника $x_l \leq x \leq x_r$, $y_d \leq y \leq y_u$, имеет вид

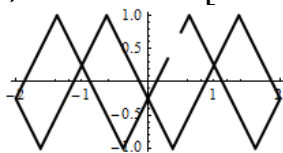
$$\begin{aligned} x_t(t) &= x_l + stc(x(t) - x_l, 2(x_r - x_l)) \\ y_t(t) &= y_d + stc(y(t) - y_d, 2(y_u - y_d)) \end{aligned} \quad (1)$$

Вот пример уравнений ломаной, полученной многократным внутренним отражением прямой/луча от границ прямоугольника $-2 \leq x \leq 2$, $-1 \leq y \leq 1$. Луч стартует в точке $(0.5, 0.75)$ и имеет направляющий вектор $(1, 2)$.

```

xl = -2; xr = 2; yd = -1; yu = 1;
x[t_] = 0.5 + t;
y[t_] = 0.75 + 2t;
xt[t_] = xl + Stc[x[t] - xl, 2(xr - xl)];
yt[t_] = yd + Stc[y[t] - yd, 2(yu - yd)];
ParametricPlot[{xt[t], yt[t]}, {t, 0, 7.8},
    PlotStyle → {Black, Thickness[0.01]}]

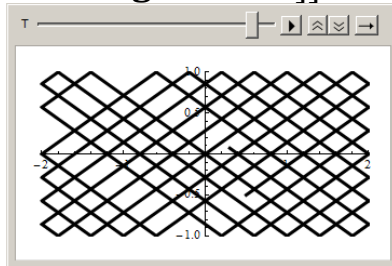
```



Траектория точки в прямоугольном бильярде является ломаной, получаемой многократным отражением луча от «бортов» прямоугольника. Пусть из точки (x_0, y_0) вылетает бесконечно малый шар в направлении вектора $(\cos \alpha, \sin \alpha)$.

Трение отсутствует. Если бы бортов области не было, то он двигался бы вдоль прямой с уравнением $x = x_0 + ct \cos \alpha$, $y = y_0 + ct \sin \alpha$, где t – время, c – скорость, α – угол наклона луча. Уравнение бильярдной траектории будет иметь вид (1), где в качестве функций $x(t)$, $y(t)$ следует использовать уравнение этой прямой. Следующий код моделирует движение точки в прямоугольном бильярде.

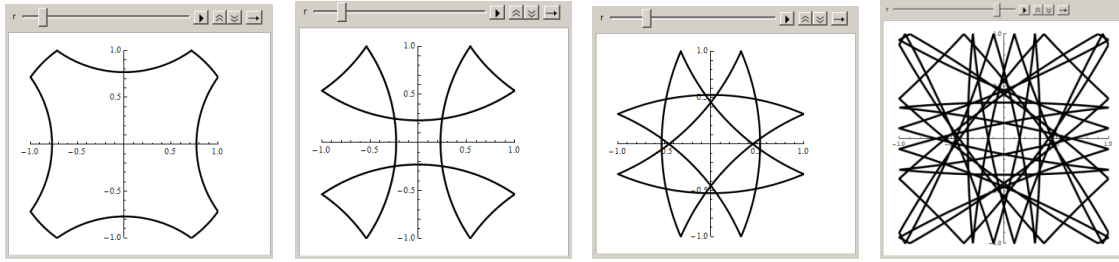
```
DynamicModule[{xl = -2, xr = 2, yd = -1, yu = 1,
  x0 = 0.5, y0 = -0.5,  $\alpha$  = ArcTan[5/7], x, y, xt, yt},
  x[t_] = x0 + t Cos[ $\alpha$ ];
  y[t_] = y0 + t Sin[ $\alpha$ ];
  xt[t_] = xl + Stc[x[t] - xl, 2(xr - xl)];
  yt[t_] = yd + Stc[y[t] - yd, 2(yu - yd)];
  Animate[ParametricPlot[{xt[t], yt[t]}, {t, 0, T},
    PlotStyle → {Black, Thickness[0.01]},
    PlotRange → {{xl, xr}, {yd, yu}}, PlotPoints → 200],
  {T, 0.01, 68}, AnimationRunning → False]
```



Пример 2.4. Отражение кругового волнового фронта.

Формулы отражения (1) кривой $x = x(t)$, $y = y(t)$ от «бортов» прямоугольника можно использовать для кривой любой формы (не только прямой). Например, уравнение кругового волнового фронта распространяющегося из некоторой точки и отражающегося от «берегов» прямоугольника, тоже будут преобразовываться в соответствии с формулами (1). В следующем примере мы строим анимацию распространения и отражения кругового волнового фронта в квадратной области. Начальная точка расположена в центре.

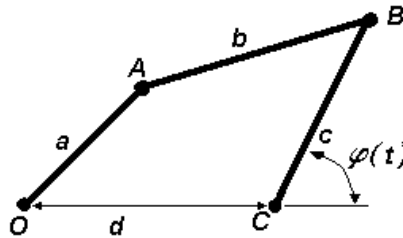
```
DynamicModule[{xl = -1, xr = 1, yd = -1, yu = 1, x, y, xt, yt},
  x[r_] = r Cos[a];
  y[r_] = r Sin[a];
  xt[r_] = xl + Stc[x[r] - xl, 2(xr - xl)];
  yt[r_] = yd + Stc[y[r] - yd, 2(yu - yd)];
  Animate[ParametricPlot[{xt[r], yt[r]}, {a, 0, 2 $\pi$ },
    PlotStyle → {Black, Thickness[0.01]},
    PlotRange → {{xl, xr}, {yd, yu}}, PlotPoints → 200],
  {r, 0.01, 6}, AnimationRunning → False]
```



Несколько кадров волнового фронта в последовательные моменты времени представлены на предыдущем рисунке.

Пример 2.5. Моделирование кривых с помощью шарнирных механизмов

Смоделируем движение шарнирного механизма, состоящего из трех звеньев. Ведущее звено OA вращается вокруг точки O с постоянной угловой скоростью, звенья OA , AB и BC шарнирно соединены в точках A и B . Звено BC также может совершать свободное вращение вокруг точки C . Обозначим длины звеньев через a , b , c так, как показано на рисунке и расстояние по горизонтали между точками O и C обозначим через d .



Чтобы точка A могла пройти горизонтальное положение слева от точки O необходимо, чтобы выполнялось соотношение $a + d \leq b + c$.

Очевидно, что точка B будет двигаться по дуге окружности вокруг точки C , однако закон этого движения нам неизвестен. Обозначим угол, который образует отрезок CB с осью X через $\varphi(t)$. Тогда

$$\begin{cases} x_A = a \cos t \\ y_A = a \sin t \end{cases} \quad \begin{cases} x_B = d + c \cos \varphi(t) \\ y_B = c \sin \varphi(t) \end{cases}$$

Длина отрезка AB равна $(x_A - x_B)^2 + (y_A - y_B)^2 = b^2$. Подставим сюда выражения координат точек A и B из предыдущих соотношений и получим уравнение для определения неизвестной функции $\varphi(t)$.

$$(d + c \cos \varphi(t) - a \cos t)^2 + (c \sin \varphi(t) - a \sin t)^2 = b^2.$$

После преобразований получаем $\alpha \cos \varphi(t) - \beta \sin \varphi(t) = \gamma$, где

$$\alpha = d - a \cos t, \quad \beta = a \sin t, \quad \gamma = \frac{b^2 - a^2 - c^2 - d^2 + 2ad \cos t}{2c}$$

Разделим обе части последнего уравнения на $\sqrt{\alpha^2 + \beta^2}$ и обозначим

$$\sin \delta = \frac{\alpha}{\sqrt{\alpha^2 + \beta^2}}, \quad \cos \delta = \frac{\beta}{\sqrt{\alpha^2 + \beta^2}}$$

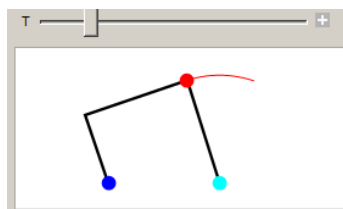
Тогда уравнение относительно $\varphi(t)$ может быть записано в виде $\sin(\delta - \varphi(t)) = \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}$ или $\varphi(t) = \delta(t) - \arcsin \frac{\gamma}{\sqrt{\alpha^2 + \beta^2}}$. Учитывая, что имеет место равенство $\alpha^2 + \beta^2 = d^2 - 2ad \cos t + a^2$, после преобразований получаем $\varphi(t) = \delta(t) - \zeta(t)$,

где

$$\delta(t) = \arccos \frac{a \sin t}{\sqrt{d^2 - 2ad \cos t + a^2}} \quad \text{и} \quad \zeta(t) = \arcsin \frac{b^2 - a^2 - c^2 - d^2 + 2ad \cos t}{2c \sqrt{d^2 - 2ad \cos t + a^2}}.$$

Следующий код моделирует движение трехзвенного механизма

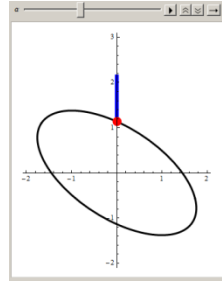
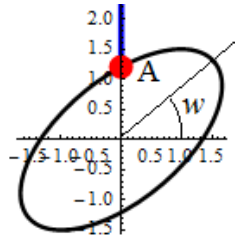
```
DynamicModule[{a = 2, b = 3, c = 3, d = 3.1, δ, ζ, φ, ln, pw, pc, pts},
  δ[t_] = ArcCos[ $\frac{a \sin[t]}{\sqrt{a^2 + d^2 - 2ad \cos[t]}}$ ];
  ζ[t_] = ArcSin[ $\frac{b^2 + 2ad \cos[t] - a^2 - c^2 - d^2}{2c \sqrt{a^2 + d^2 - 2ad \cos[t]}}$ ];
  φ[t_] = δ[t] - ζ[t];
  Manipulate[
    ln = Line[{{0, 0}, {a Cos[t], a Sin[t]}, {d + c Cos[φ[t]], c Sin[φ[t]], {d, 0}}];
    pts = Graphics[{{Blue, Disk[{0, 0}, 0.2]},
      {Red, Disk[{d + c Cos[φ[t]], c Sin[φ[t]], 0.2]},
      {Cyan, Disk[{d, 0}, 0.2]}}];
    pw = Graphics[{Thickness[0.01], Black, ln /. t → T}];
    pc = ParametricPlot[{d + c Cos[φ[t]], c Sin[φ[t]]},
      {t, π/4 + 0.001, T}, PlotStyle → Red];
    Show[pw, pc, pts, PlotRange → {{-a, c + d}, {-c - 0.2, c + 0.2}},
      {T, π/4, 2π + π/4}]]
```



Здесь мы использовали панель манипулятора, а не панель анимации, как в предыдущих примерах.

Пример 2.6. Кулачковый механизм. Эксцентрик в форме эллипса совершает вращательное движение и взаимодействует с кулачком в форме прямолинейного стержня. Смоделируем движение такого механизма, используя графические функции системы.

Пусть полуоси эллипса равны a и b , а длина кулачка/стержня L . В следующем коде функции $xe(t, w)$, $ye(t, w)$ при фиксированном w представляют параметрическое уравнение повернутого на угол w эллипса. Функция $y_A(w)$ представляет y -координату толкателя (точка А на следующем рисунке слева).



```
DynamicModule[{a = 2, b = 1, L = 1, xe, ye, yA, ys},
  xe[t_, w_] = a Cos[t] Cos[w] - b Sin[t] Sin[w];
  ye[t_, w_] = a Cos[t] Sin[w] + b Sin[t] Cos[w];
  yA[w_] =  $\frac{a b}{\sqrt{a^2 \cos^2[w] + b^2 \sin^2[w]}}$ ;
  ys[t_, w_] = yA[w] + t;
  Animate[
    p1 = ParametricPlot[{xe[t,  $\alpha$ ], ye[t,  $\alpha$ ]}, {t, 0, 2  $\pi$ },
      PlotStyle  $\rightarrow$  {Black, Thickness[0.01]};
    p2 = ParametricPlot[{0, ys[t,  $\alpha$ ]}, {t, 0, L},
      PlotStyle  $\rightarrow$  {Blue, Thickness[0.02]};
    pt = Graphics[{Red, Disk[{0, yA[ $\alpha$ ]}, 0.1]}];
    Show[p1, p2, pt, PlotRange  $\rightarrow$  {{-a, a}, {-a, a + 1}},
      { $\alpha$ , 0, 2  $\pi$ }, AnimationRunning  $\rightarrow$  False]]
```

Панель анимации со схематическим изображением кулачкового механизма показана на предыдущем рисунке справа.

Параметрическое уравнение повернутого эллипса $xe(t, w)$, $ye(t, w)$ получается умножением вектора $(a \cos t, b \sin t)$ на матрицу поворота на угол w . Для определения y координаты точки A мы определяем значение параметра t в точке A на повернутом эллипсе. Для этого приравняем нулю функцию $xe(t, w)$ и выражаем t через w (точнее выражаем $\tan t$). Получаем $\tan t = \frac{a \cos w}{b \sin w}$.

Подстановка этого значения в выражение $ye(t, w)$ после упрощений дает функцию $y_A(w)$.

2.4.3 Движение жидкости в трубах

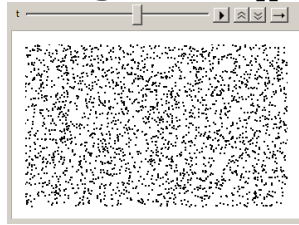
Смоделируем поле скоростей жидкости в цилиндрической трубе с эллиптическим сечением. Скорость жидкости w направлена вдоль оси трубы (ось X), одинакова во всех поперечных сечениях, и в точке сечения с координатами (y, z) определяется формулой

$$w(y, z) = \frac{Ca^2b^2}{a^2 + b^2} \left(1 - \frac{y^2}{a^2} - \frac{z^2}{b^2} \right),$$

где a и b являются длинами полуосей эллипса/сечения, C – некоторая постоянная, зависящая от разности давлений на концах трубы и вязкости жидкости.

Пример 3.1. Смоделируем движение жидкости как движение частиц в продольном сечении $z=0$ трубы. Ось X направлена вдоль оси трубы вправо, ось Y – вертикально вверх. Положим $a=1$, $b=1$, $C=2$. Сгенерируем большое количество случайно расположенных в продольном сечении точек и, в зависимости от их вертикальной координаты y , зададим им скорость движения $w(y)=1-y^2$.

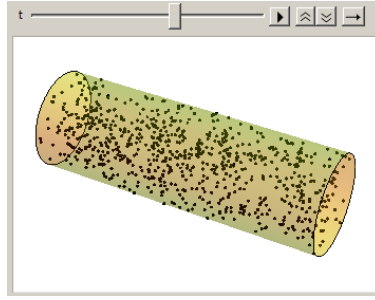
```
DynamicModule[{n = 5000, h = 6, w, xl, yl, pt},
  w[y_] = 1 - y^2;
  xl = RandomReal[{-h, h}, n];
  yl = RandomReal[{-1, 1}, n];
  Animate[
    pt = Transpose[{xl + w[yl]t, yl}];
    ListPlot[pt, PlotStyle -> {Black, PointSize[0.005]},
      PlotRange -> {{0, h}, {-1, 1}}, Axes -> None],
    {t, 0, h, 0.01}, AnimationRunning -> False]]
```



Отметим, что приведенное здесь статическое изображение, не передает процесс движения в отличие от динамического движения, воспроизводимого в панели анимации.

Пример 3.2. Смоделируем пространственное движение жидкости как движение частиц в круглой трубе. Ось X направлена вдоль оси трубы вправо. Положим $a=1$, $b=1$, $C=2$. Сгенерируем большое количество случайно расположенных в трубе точек и, в зависимости от их поперечных координат y и z , зададим им скорость движения $w(y, z)=1-y^2-z^2$.

```
DynamicModule[{n = 2000, h = 6, w, xl, yl, zl, pt},
  w[y_, z_] = 1 - y^2 - z^2;
  xl = RandomReal[{-h, h}, n];
  yl = RandomReal[{-1, 1}, n];
  zl = RandomReal[{-1, 1}, n];
  Animate[
    pt = Transpose[{xl + w[yl, zl] t, yl, zl}];
    pl = ListPointPlot3D[pt, PlotStyle -> {Black, PointSize[0.005]},
      PlotRange -> {{0, h}, {-1, 1}, {-1, 1}},
      RegionFunction -> Function[{x, y, z}, y^2 + z^2 < 1];
    pr = RegionPlot3D[y^2 + z^2 < 1, {x, 0, h}, {y, -1, 1}, {z, -1, 1},
      PlotStyle -> Directive[Yellow, Opacity[0.3]], Mesh -> None];
    Show[pl, pr, BoxRatios -> Automatic, Boxed -> False, Axes -> None],
    {t, 0, h, 0.001}, AnimationRunning -> False]]
```



Особенностью этого примера является создание полупрозрачной трехмерной области **pr**, представляющей участок трубы, и использование у функции `ListPointPlot3D` опции `RegionFunction`, отсекающей изображения точек, выходящие за пределы кругового сечения трубы.

□

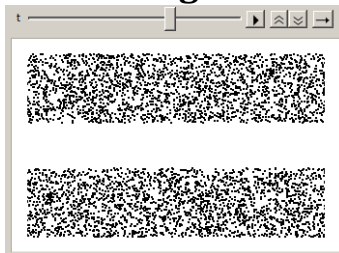
Поле скоростей жидкости в области между двумя коаксиальными круглыми цилиндрами радиусов r_1 и r_2 ($r_2 > r_1$) определяется по формуле

$$w(r) = C \left(r_1^2 - r^2 + \frac{(r_2^2 - r_1^2) \ln(r/r_1)}{\ln(r_2/r_1)} \right),$$

где r – расстояние точки сечения до оси и C – некоторая постоянная, зависящая от вязкости жидкости и разности давлений на входе и выходе трубы.

Пример 3.3. Смоделируем движение жидкости в продольном сечении $z=0$ такой трубы. Ось X направим вдоль оси трубы вправо и положим $r_1 = 0.5$, $r_2 = 2$, $C = 1$. Сгенерируем большое количество случайно расположенных в продольном сечении точек и, в зависимости от их расстояния r до оси, зададим им скорость движения $w(r)$.

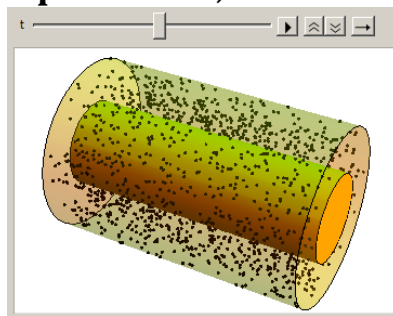
```
DynamicModule[{n = 5000, h = 6, r1 = 0.5, r2 = 2, w, xl, yl1, yl2, pt},
  w[r_] = r12 - r2 + (r22 - r12)Log[r/r1]/Log[r2/r1];
  xl = RandomReal[-h, h, n];
  yl1 = RandomReal[r1, r2, n];
  yl2 = RandomReal[-r2, -r1, n];
  Animate[
    pt = Join[Transpose[{xl + w[Abs[yl1]]t, yl1}],
      Transpose[{xl + w[Abs[yl2]]t, yl2}]];
    ListPlot[pt, PlotStyle -> {Black, PointSize[0.005]},
      PlotRange -> {{0, h}, {-2, 2}}, Axes -> None,
    {t, 0, h - 1, 0.01}, AnimationRunning -> False]
```



В этом примере мы создаем два случайных числовых множеств $yl1$ и $yl2$, которые затем используются при создании единого списка pt пар координат точек.

Пример 3.4. Смоделируем пространственное движение жидкости в области между двумя коаксиальными круглыми цилиндрами. Ось X направим вдоль оси цилиндров и положим $r_1 = 1$, $r_2 = 2$, $C = 1$. Сгенерируем большое количество случайно расположенных в трубе точек и, в зависимости от их расстояния r до оси, зададим им скорость движения $w(r)$.

```
DynamicModule[{n = 3000, h = 6, r1 = 1, r2 = 2, w, rl, ϕl, xl, yl, zl, pt},
  w[r_] = r12 - r2 + (r22 - r12)Log[r/r1]/Log[r2/r1];
  xl = RandomReal[{-h, h}, n];
  rl = RandomReal[{r1, r2}, n];
  ϕl = RandomReal[{0, 2π}, n];
  yl = rl Cos[ϕl];
  zl = rl Sin[ϕl];
  Animate[
    pt = Transpose[{xl + w[rl] t, yl, zl}];
    pl = ListPointPlot3D[pt, PlotStyle -> {Black, PointSize[0.005]},
      PlotRange -> {{0, h}, {-r2, r2}, {-r2, r2}},
      RegionFunction -> Function[{x, y, z},
        y2 + z2 > r12 && y2 + z2 < r22]];
    pr1 = RegionPlot3D[y2 + z2 < r22, {x, 0, h}, {y, -r2, r2}, {z, -r2, r2},
      PlotStyle -> Directive[Yellow, Opacity[0.3]], Mesh -> None];
    pr2 = RegionPlot3D[y2 + z2 < r12, {x, 0, h}, {y, -r2, r2}, {z, -r2, r2},
      PlotStyle -> Directive[Yellow, Opacity[1]], Mesh -> None];
    Show[pl, pr1, pr2, BoxRatios -> Automatic, Boxed -> False,
      Axes -> None],
    {t, 0, h, 0.1}, AnimationRunning -> False,
    DisplayAllSteps -> True, AnimationRate -> 2]]
```



В этом примере мы создаем полупрозрачный внешний цилиндр и непрозрачный внутренний. Координаты случайных точек создаются путем генерирования множества rl случайных радиусов в диапазоне от $r1$ до $r2$ и множества $ϕl$ случайных полярных углов, которые затем используются для создания списка pt координат $\{\dots, \{x_i, y_i, z_i\}, \dots\}$ точек.

2.4.4 Двумерные волновые движения

Пример 4.1. Поверхностные волны Релея. Английский физик Релей показал, что в упругом полупространстве могут существовать специальные волны, которые он назвал поверхностными. У таких волн амплитуда колебаний быстро уменьшается с удалением от поверхности. Одно из таких плоских волновых движений может быть описано с помощью следующих функций [9]

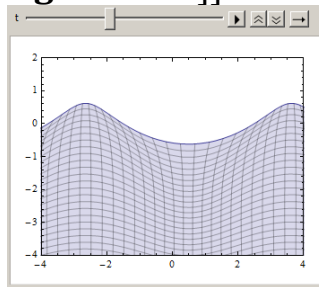
$$\begin{aligned} U(x, y, t) &= C(-e^{-\alpha y} + c e^{-\beta y}) \sin(\xi x + pt) \\ V(x, y, t) &= C a(-e^{-\alpha y} - d e^{-\beta y}) \cos(\xi x + pt) \end{aligned} \quad (1)$$

где U и V представляют продольное и поперечное смещение точек среды (x, y) относительно положения равновесия. Здесь $\alpha = a\xi$, $\beta = b\xi$ и a, b, c, d некоторые постоянные, зависящие от свойств среды, C – произвольная постоянная. Ось Y направлена в глубь среды.

Направим ось Y вертикально вверх (у системы *Mathematica* нет простого способа менять направления осей). В силу этого в примере в формулах Релея (1) мы изменим знак перед переменной y и перед функцией V . Полупространство в состоянии покоя будем моделировать в виде области, заданной в параметрическом виде $x(u, v) = u$, $y(u, v) = v$, где $-\infty < u < \infty$, $v \leq 0$. Тогда смещенные точки области в момент времени t будут иметь координаты $X = x(u, v) + U(x(u, v), y(u, v), t)$, $Y = y(u, v) + V(x(u, v), y(u, v), t)$, где $v \leq 0$.

Следующий код моделирует движение среды – поверхностную волну Релея.

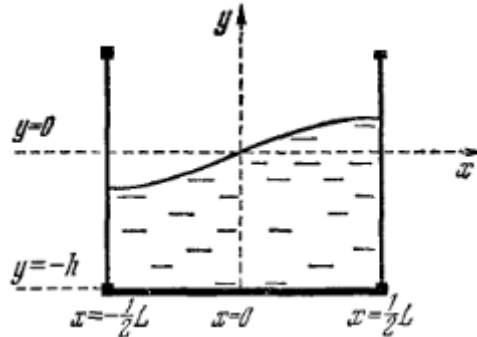
```
DynamicModule[{a = 0.8475, b = 0.3933,
  c = 0.5753, d = 1.732, p = 1, Cc = 1, U, V, X, Y},
  U[x_, y_, t_] = Cc(-Exp[-a y] + c Exp[-b y]) Sin[x + p t];
  V[x_, y_, t_] = Cc a (Exp[-a y] - d Exp[-b y]) Cos[x + p t];
  X[u_, v_, t_] = u + U[u, -v, t];
  Y[u_, v_, t_] = v - V[u, -v, t];
  Animate[ParametricPlot[
    {{X[u, v, t], Y[u, v, t]}}, {u, -5, 5}, {v, -5, 2}, Mesh -> {31, 26},
    RegionFunction -> Function[{x, y, u, v}, v < 0],
    PlotRange -> {{-4, 4}, {-4, 2}}, Axes -> None],
    {t, 0, 2π}, AnimationRunning -> False]]
```



Использование параметрического уравнения области в этом примере позволяет показать не только форму поверхности, но и образ декартовой координатной сетки внутри среды.

Пример 4.2. Установившаяся синусоидальная волна в прямоугольном аквариуме постоянной глубины [1].

Рассмотрим прямоугольный аквариум шириной L с жидкостью налитой до уровня h . Направим ось Y вертикально вверх с началом на свободной поверхности жидкости, а ось X направим вправо (см. следующий рисунок).



Одно из возможных установившихся волновых движений жидкости может быть представлено в виде

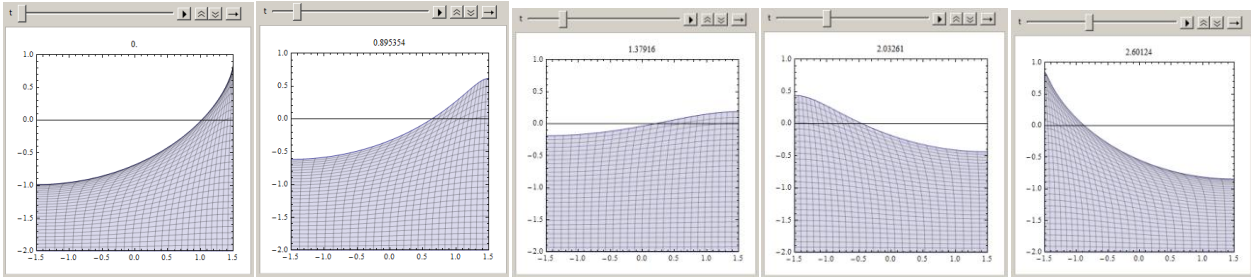
$$\begin{aligned} U &= a \cos(\omega t) \cos(kx) (e^{ky} + e^{-2kh} e^{-ky}) \\ V &= a \cos(\omega t) \sin(kx) (e^{ky} - e^{-2kh} e^{-ky}) \end{aligned} \quad (2)$$

где $k = \pi/L$ и a – произвольная постоянная. Функции $U(x, y, t)$ и $V(x, y, t)$ дают мгновенные значения смещений частицы жидкости с равновесными координатами x, y в направлениях осей X и Y .

Жидкость в состоянии покоя будем моделировать в виде плоской области, заданной в параметрическом виде $x(u, v) = u$, $y(u, v) = v$, где $-L/2 \leq u \leq L/2$, $-h \leq v \leq 0$. Точки области в момент времени t будут иметь координаты $X = x(u, v) + U(x(u, v), y(u, v), t)$, $Y = y(u, v) + V(x(u, v), y(u, v), t)$, где надо положить $x = u$, $y = v$ и $-L/2 \leq u \leq L/2$, $-h \leq v \leq 0$.

```
DynamicModule[{w = 1, L = 3, a = 1, h = 2, k, X, Y}, k =  $\pi/L$ ;  
  X[u_, v_, t_] = u + aCos[wt]Cos[ku](Exp[kv] + Exp[-2kh]Exp[-kv]);  
  Y[u_, v_, t_] = v + aCos[wt]Sin[ku](Exp[kv] - Exp[-2kh]Exp[-kv]);  
  Animate[  
    ParametricPlot[{{X[u, v, t], Y[u, v, t]}}, {u, -L/2, L/2}, {v, -h, 0},  
      Mesh → 26, PlotRange → {{-L/2, L/2}, {-h, 1}},  
      PlotLabel → t, Axes → {True, False}],  
    {t, 0, 2 $\pi$ }, AnimationRunning → False]]
```

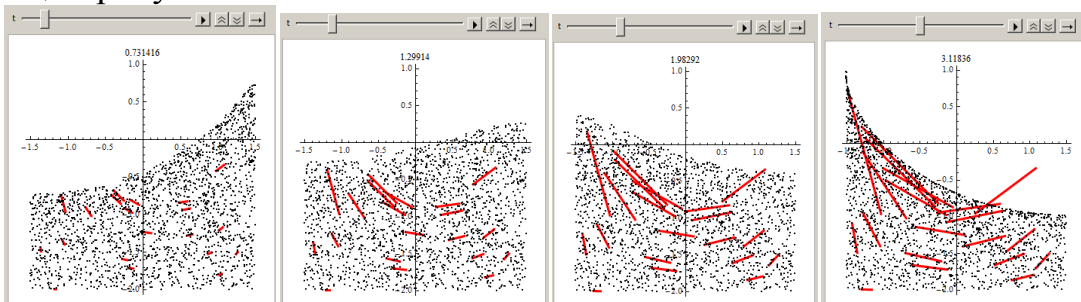
Область, занимаемая жидкостью, в начальный и некоторые последующие моменты времени показана на следующем рисунке.



Наблюдая анимацию, можно заметить, что у стенок и дна движение происходит вдоль границ. Из уравнений (2) видно, что движение любой фиксированной точки (x, y) состоит из гармонического колебания по некоторому прямолинейному отрезку в плоскости XY . Этот факт можно продемонстрировать графически, если среду смоделировать в виде множества точек и нарисовать траектории некоторых из них.

```
DynamicModule[{n = 2000, L = 3, h = 2, a = 1, w = 1, np = 20,
                k, ul, vl, X, Y, nl, up, vp, Xp, Yp, Xl, Yl, pt, pp, pl},
  ul = RandomReal[{-L/2, L/2}, n];
  vl = RandomReal[{-h, 0}, n];
  k =  $\pi/L$ ;
  X[u_, v_, t_] = u + aCos[wt]Cos[ku](Exp[kv] + Exp[-2kh]Exp[-kv]);
  Y[u_, v_, t_] = v + aCos[wt]Sin[ku](Exp[kv] - Exp[-2kh]Exp[-kv]);
  SeedRandom[1234]; (* инициализация генератора случайных чисел *)
  nl = RandomInteger[{1, n}, np];
  up = ul[[nl]]; vp = vl[[nl]];
  Xp[ $\tau$ ] = X[up, vp,  $\tau$ ]; Yp[ $\tau$ ] = Y[up, vp,  $\tau$ ];
  Animate[
    Xl = X[ul, vl, t];
    Yl = Y[ul, vl, t];
    pt = Transpose[{Xl, Yl}];
    pp = ParametricPlot[Transpose[{Xp[ $\tau$ ], Yp[ $\tau$ ]}, { $\tau$ , 0, t},
                        PlotStyle -> {Red, Thickness[0.01]}];
    pl = ListPlot[pt, PlotStyle -> {Black, PointSize[0.01]},
                  PlotRange -> {{-L/2, L/2}, {-h, 1.}}];
    Show[pp, pl, PlotRange -> {{-L/2, L/2}, {-h, 1.}},
          AxesOrigin -> Automatic],
    {t, 0.01, 2 $\pi$ }, AnimationRunning -> False]]
```

Положение точек жидкости в последовательные моменты времени показано на следующем рисунке.



В этом коде из n случайных точек выбираются np . Для них строятся траектории движения, которые, как видно из рисунков, являются отрезками.

Пример 4.3. Установившееся волновое движение жидкости в резервуаре конечной ширины и бесконечной глубины [1].

Рассмотрим сосуд шириной L бесконечной глубины. Ось X направлена вправо, ось Y вверх. Начало координат расположено в середине свободной поверхности жидкости. Одно из возможных установившихся волновых движений жидкости может быть получено из формул (2) при $h = \infty$.

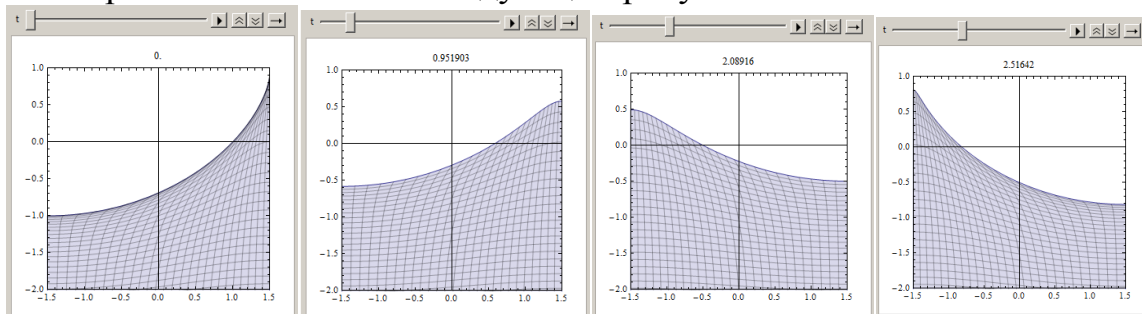
$$\begin{aligned} X &= x + a \cos(wt) \cos(kx) e^{ky} & (-L/2 < u < L/2, v \leq 0), \\ Y &= y + a \cos(wt) \sin(kx) e^{ky} \end{aligned} \quad (3)$$

где (x, y) координаты точек среды в состоянии покоя, X и Y координаты тех же точек в момент времени t , и a, w, k – некоторые постоянные.

Для создания анимации аналогично предыдущему примеру, жидкость в состоянии покоя будем моделировать в виде плоской области, заданной в параметрическом виде $x(u, v) = u, y(u, v) = v$, где $-L/2 < u < L/2, v \leq 0$. Положение точек области в движении будут определяться формулами (3).

```
DynamicModule[{w = 1, L = 3, k =  $\pi/L$ , a = 1, X, Y},
  X[u_, v_, t_] = u + a Cos[wt] Cos[ku] Exp[kv];
  Y[u_, v_, t_] = v + a Cos[wt] Sin[ku] Exp[kv];
  Animate[ParametricPlot[
    {{X[u, v, t], Y[u, v, t]}}, {u, -L/2, L/2}, {v, -2.5, 0}, Mesh → 26,
    PlotRange → {{-L/2, L/2}, {-2, 1}}, PlotLabel → t],
    {t, 0, 2 $\pi$ }, AnimationRunning → False]]
```

Область, занимаемая жидкостью, в начальный и некоторые последующие моменты времени показана на следующем рисунке.



Здесь, как и в предыдущем примере, частицы жидкости двигаются по прямолинейным отрезкам.

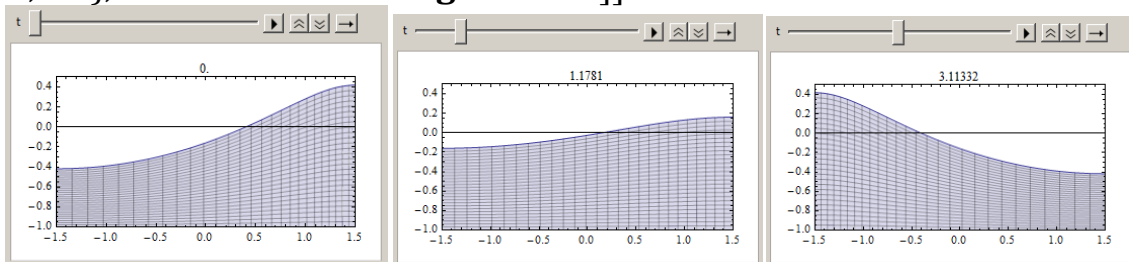
Пример 4.4. Стоячие волны в неглубокой воде [1]. Под ними понимают волны, которые возникают в сосуде, равновесная глубина h которого мала по сравнению с длиной волн $\lambda = 2\pi/k$. В этом случае мы можем аппроксимировать зависимости U и V от u , оставив в разложении в ряд Тейлора только первые члены. В результате уравнения (2) примут вид

$$U = 2a \cos(\omega t) \cos(kx)$$

$$V = 2a \cos(\omega t) \sin(kx) (k(y+h))$$

Для такой волны горизонтальное смещение частиц не зависит от y , а вертикальное смещение меняется линейно с глубиной, достигая нуля на дне и максимума на поверхности.

```
DynamicModule[{w = 1, L = 3, k =  $\pi/L$ , a = 0.2, h = 1, X, Y},
  X[u_, v_, t_] = u + 2a Cos[wt] Cos[ku];
  Y[u_, v_, t_] = v + 2a Cos[wt] Sin[ku] k(v + h);
  Animate[
    ParametricPlot[{{X[u, v, t], Y[u, v, t]}}, {u, -L/2, L/2}, {v, -h, 0},
      Mesh → 26, PlotRange → {{-L/2, L/2}, {-h, 0.5}},
      PlotLabel → t, Axes → {True, False}],
    {t, 0, 2 $\pi$ }, AnimationRunning → False]]
```



Заметим, что в нашем примере $\lambda = 2\pi/k = 2\pi/(\pi/L) = 2L = 6$, $h=1$ и соотношение $h \ll \lambda$ для волн на неглубокой воде выполняется (приблизительно).

Пример 4.5. Бегущие волны в водоеме конечной глубины, неограниченном в горизонтальном направлении, можно смоделировать уравнениями [1]

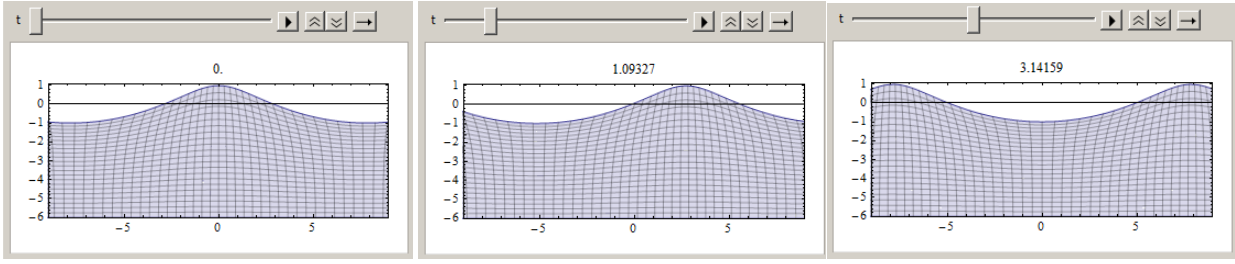
$$U = a \sin(\omega t - kx) (e^{ky} + e^{-2kh} e^{-ky})$$

$$V = a \cos(\omega t - kx) (e^{ky} - e^{-2kh} e^{-ky}) \quad (4)$$

где U и V мгновенные значения смещений частицы жидкости с равновесными координатами x, y в направлениях осей X и Y . Следующий код моделирует движение жидкости в соответствии с уравнениями (4).

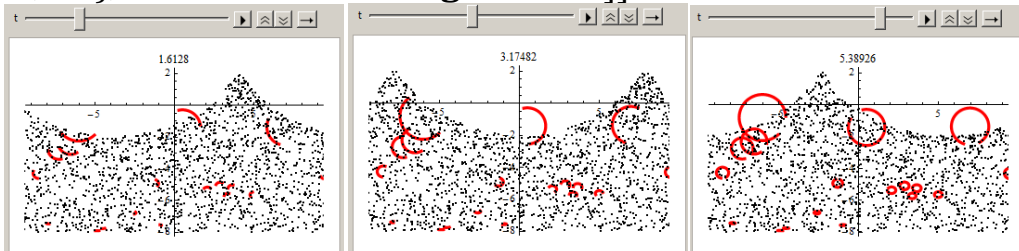
```
DynamicModule[{w = 1, L = 6, a = 1, h = 6, k = 0.4, X, Y},
  X[u_, v_, t_] = u + a Sin[wt - ku] (Exp[kv] + Exp[-2kh] Exp[-kv]);
  Y[u_, v_, t_] = v + a Cos[wt - ku] (Exp[kv] - Exp[-2kh] Exp[-kv]);
  Animate[
    ParametricPlot[{{X[u, v, t], Y[u, v, t]}}, {u, -2L, 2L}, {v, -h, 0},
      Mesh → {49, 21}, PlotRange → {{-1.5L, 1.5L}, {-h, 1.1}},
      PlotLabel → t, Axes → {True, False}],
    {t, 0, 2 $\pi$ }, AnimationRunning → False]]
```

На следующем рисунке представлена двумерная область, имитирующая бегущую волну в последовательные моменты времени.



Из формул (4) следует, что частицы жидкости двигаются по эллипсам $\frac{(X-x)^2}{(e^{ky} + e^{-2kh}e^{-ky})^2} + \frac{(Y-y)^2}{(e^{ky} - e^{-2kh}e^{-ky})^2} = 1$, которые с увеличением глубины постепенно «сплюсчиваются», а на дне вырождаются в горизонтальные отрезки. Этот факт можно продемонстрировать графически, если среду смоделировать в виде множества точек и нарисовать траектории некоторых из них.

```
DynamicModule[{n = 2000, L = 6, h = 8, a = 2, w = 1, np = 20, k = 0.4,
               ul, vl, X, Y, nl, up, vp, Xp, Yp, Xl, Yl, pt, pp, pl},
  ul = RandomReal[{-1.8L, 1.8L}, n];
  vl = RandomReal[{-h, 0}, n];
  X[u_, v_, t_] = u + aSin[wt - ku](Exp[kv] + Exp[-2kh]Exp[-kv]);
  Y[u_, v_, t_] = v + aCos[wt - ku](Exp[kv] - Exp[-2kh]Exp[-kv]);
  SeedRandom[Floor[SessionTime[ ]]];
  nl = RandomInteger[{1, n}, np];
  up = ul[[nl]]; vp = vl[[nl]];
  Xp[τ_] = X[up, vp, τ];
  Yp[τ_] = Y[up, vp, τ];
  Animate[
    Xl = X[ul, vl, t];
    Yl = Y[ul, vl, t];
    pt = Transpose[{Xl, Yl}];
    pp = ParametricPlot[Transpose[{Xp[τ], Yp[τ]}], {τ, 0, t},
      PlotStyle → {Red, Thickness[0.01]}];
    pl = ListPlot[pt, PlotStyle → {Black, PointSize[0.005]},
      PlotRange → {{-1.5L, 1.5L}, {-h, 0}}];
    Show[pp, pl, PlotRange → {{-1.5L, 1.5L}, {-h, 2.1}},
      AxesOrigin → Automatic, PlotLabel → t],
    {t, 0.01, 2π}, AnimationRunning → False]]
```



Если рассмотреть водоем бесконечной глубины, то траектории точек будут окружностями, радиус которых зависит от глубины y . Смоделируйте самостоятельно такое движение.

2.4.5 Одномерные волновые колебания

Поперечные колебания струны, продольные колебания упругого стержня и многие другие явления описываются одномерным волновым уравнением

$$\frac{\partial^2 u(x, t)}{\partial t^2} = a^2 \frac{\partial^2 u(x, t)}{\partial x^2} \quad (1)$$

Для однозначного определения решения необходимо задать начальные условия

$$u(x, 0) = \varphi(x), \quad u'_t(x, 0) = \psi(x) \quad (2)$$

и граничные условия, если у колеблющегося объекта имеются границы. Обычно функция $u(x, t)$ представляет смещение из положения равновесия точки x одномерной среды в момент времени t .

Известно общее решение Даламбера задачи (1), (2) для неограниченной прямой [2]

$$u(x, t) = \frac{\varphi(x + at) + \varphi(x - at)}{2} + \frac{1}{2a} \int_{x-at}^{x+at} \psi(\alpha) d\alpha \quad (-\infty < x < \infty, t \geq 0). \quad (3)$$

Формула (3) при любых функциях $\varphi(x)$ и $\psi(x)$ дает решение уравнения (1). В данном параграфе во всех задачах начальная скорость колебаний равна нулю $\psi(x) = 0$ и поэтому решение будет иметь вид

$$u(x, t) = \frac{\Phi(x + at) + \Phi(x - at)}{2}, \quad (4)$$

а функция $\Phi(x)$ будет подбираться так, чтобы удовлетворить начальному условию $u(x, 0) = \varphi(x)$ и дополнительным условиям на границе. Во всех задачах также будем считать, что параметр $a=1$.

Пример 5.1. Колебание конечной струны с синусоидальным начальным профилем и нулевой начальной скоростью.

Пусть струна имеет конечную длину $L=1$ и ось X направлена вправо вдоль струны. Если начальные условия имеют вид $u(x, 0) = \sin \pi x$, $u'_t(x, 0) = 0$ и струна закреплена на концах, т.е. $u(0, t) = u(L, t) = 0$, то решение уравнения (1), удовлетворяющее указанным условиям, может быть получено из (4) при $\Phi(x) = \sin \pi x$

$$u(x, t) = \frac{\sin(\pi(x + t)) + \sin(\pi(x - t))}{2}$$

В следующем коде мы моделируем движение струны в соответствии с этим решением.

```
DynamicModule[{u, f},
  f[x_] = Sin[π x];
  u[x_, t_] = (f[x + t] + f[x - t])/2;
```

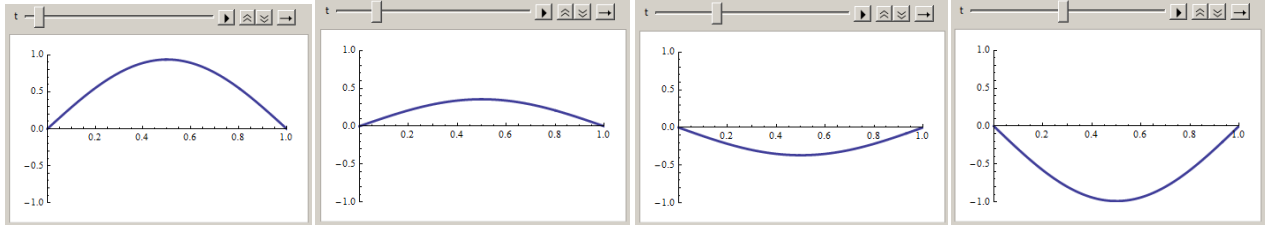


```

Animate[Plot[Evaluate[u[x, t]], {x, 0, 1},
PlotStyle → Thickness[0.01], PlotPoints → 1000,
PlotRange → {{0, 1}, {-1, 1}},
{t, 0, 2}, AnimationRunning → False, DisplayAllSteps → True]]

```

На следующем рисунке показана панель анимации в начальный и несколько последующих моментов времени.



Пример 5.2. *Колебание полубесконечной струны, закрепленной на конце.*

Рассмотрим задачу о распространении начального возмущения $u(x, 0) = \varphi(x)$ по полубесконечной струне с закрепленным концом $x = 0$. Начальная скорость равна нулю $\psi(t) = 0$ и граничное условие имеет вид $u(0, t) = 0$. Можно показать, что, если начальную функцию $\varphi(x)$ продолжить с нуля $x \geq 0$ нечетно на всю вещественную ось и подставить в (4), то получим решение поставленной задачи [2]. Это значит, что полученная таким образом функция $u(x, t)$, рассматриваемая только для $x \geq 0$, будет удовлетворять уравнению, граничному и обоим начальным условиям.

Нечетное продолжение $\Phi(x)$ любой функции $\varphi(x)$, заданной только для положительных x , на всю вещественную ось можно получить по формуле $\Phi(x) = \text{sign}(x) \varphi(|x|)$. Тогда из (4) получаем

$$u(x, t) = \frac{\varphi(x + at) + \text{sign}(x - at)\varphi(|x - at|)}{2} \quad (5)$$

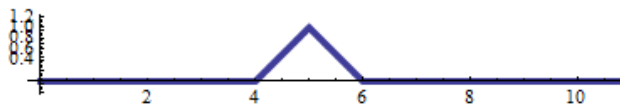
Пусть начальное отклонение струны равно нулю везде, кроме интервала (4, 6), в котором функция $\varphi(x)$ изображается равнобедренным треугольником единичной высоты. На следующем рисунке показана форма начального профиля струны.

$$\varphi[x_] = \frac{1}{2} \text{Abs}[x - 4] - \text{Abs}[x - 5] + \frac{1}{2} \text{Abs}[x - 6];$$

```

Plot[φ[x], {x, 0, 12}, PlotStyle → Thickness[0.01],
PlotPoints → 1000, PlotRange → All, AspectRatio → Automatic]

```



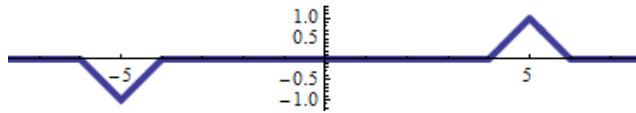
Ниже показан график функции $\Phi(x)$, являющейся нечетным продолжением функции $\varphi(x)$.

$$f[x_] = \text{Sign}[x] \varphi[\text{Abs}[x]]; \quad \text{Plot}[f[x], \{x, -8, 8\}, \text{PlotStyle} \rightarrow \text{Thickness}[0.01], \text{PlotPoints} \rightarrow 1000,$$

```

PlotRange → All, AspectRatio → Automatic]

```

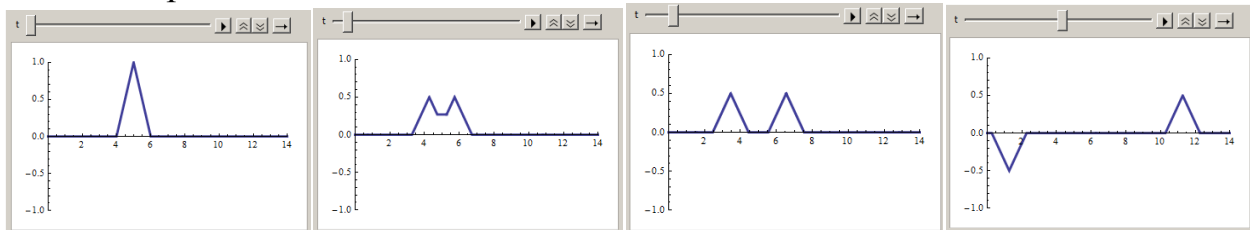


Следующий код моделирует движение струны, используя формулу (5).

```
DynamicModule[{u},
  u[x_, t_] = 
$$\frac{f[x + t] + f[x - t]}{2};$$

  Animate[
    Plot[Evaluate[u[x, t]], {x, 0, 14},
      PlotStyle → Thickness[0.01], PlotPoints → 1000,
      PlotRange → {{0, 14}, {-1, 1}},
    {t, 0, 12}, AnimationRunning → False, DisplayAllSteps → True]]
```

Ниже показана панель анимации в начальный и несколько последующих моментов времени.



Заметим, что если начальная функция $\varphi(x)$ не дифференцируема в некоторых точках, то полученное решение тоже не дифференцируемо в некоторых точках. Такое решение называют обобщенным.

Пример 5.3. *Колебание полубесконечной струны со свободным левым концом.*

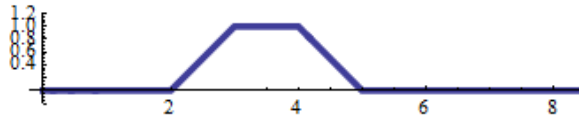
Рассмотрим задачу о распространении начального возмущения $u(x, 0) = \varphi(x)$ по полубесконечной струне со свободным левым концом $x = 0$. Начальная скорость равна нулю $\psi(t) = 0$ и граничное условие имеет вид $u'_x(0, t) = 0$. Можно показать, что, если начальную функцию $\varphi(x)$ продолжить с левых $x \geq 0$ четно на всю вещественную ось и подставить в (4), то получим решение поставленной задачи [2]. Функцию $u(x, t)$ следует рассматриваемая только для $x \geq 0$ и она будет удовлетворять уравнению (1), граничному и обоим начальным условиям.

Четное продолжение $\Phi(x)$ на вещественную ось любой функции $\varphi(x)$, заданной только для $x \geq 0$, можно получить по формуле $\Phi(x) = \varphi(|x|)$. Тогда из (4) получим

$$u(x, t) = \frac{\varphi(x + at) + \varphi(|x - at|)}{2} \quad (6)$$

Пусть начальное отклонение струны равно нулю везде, кроме интервала (2, 5) в котором функция $\varphi(x)$ изображается равнобедренной трапецией высотой 1. На следующем рисунке показан график начального профиля струны.

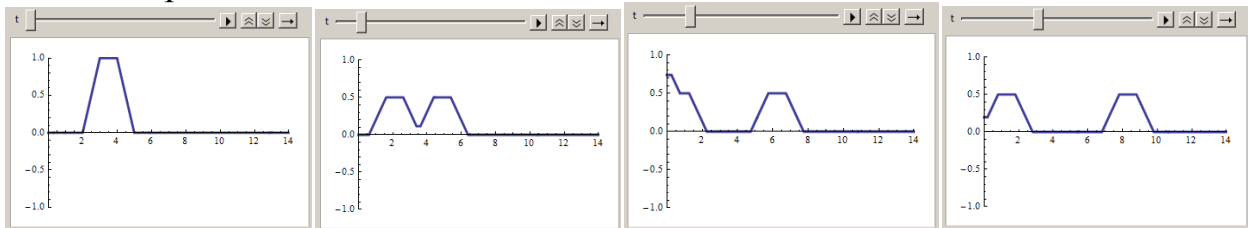
$\varphi[x_] = \frac{1}{2} \text{Abs}[x - 2] - \frac{1}{2} \text{Abs}[x - 3] - \frac{1}{2} \text{Abs}[x - 4] + \frac{1}{2} \text{Abs}[x - 5];$
Plot[$\varphi[x]$, { x , 0, 10}, **PlotStyle** → **Thickness**[0.01], **PlotPoints** → 1000,
PlotRange → All, **AspectRatio** → Automatic]



Следующий код моделирует движение струны, используя формулу (6).

DynamicModule[{ u, φ, f },
 $\varphi[x_] = \frac{1}{2} \text{Abs}[x - 2] - \frac{1}{2} \text{Abs}[x - 3] - \frac{1}{2} \text{Abs}[x - 4] + \frac{1}{2} \text{Abs}[x - 5];$
 $f[x_] = \varphi[\text{Abs}[x]];$
 $u[x_, t_] = \frac{f[x + t] + f[x - t]}{2};$
Animate[
Plot[**Evaluate**[$u[x, t]$], { x , 0, 14}, **PlotStyle** → **Thickness**[0.01],
PlotPoints → 1000, **PlotRange** → {{0, 14}, {-1, 1}},
{ t , 0, 12}, **AnimationRunning** → **False**, **DisplayAllSteps** → **True**]]

Ниже показана панель анимации в начальный и несколько последующих моментов времени.



Пример 5.4. *Колебание конечной струны с закрепленными концами.*

Рассмотрим задачу о колебании конечной струны длины L с закрепленными концами, начальным смещением $u(x, 0) = \varphi(x)$ и нулевой начальной скоростью $\psi(t) = 0$. Граничные условия имеют вид $u(0, t) = 0$, $u(L, t) = 0$. Если начальную функцию $\varphi(x)$, обращающуюся в ноль на концах отрезка $[0, L]$, продолжить на всю вещественную ось нечетно и периодически (с отрезка $[0, L]$ на отрезок $[-L, 0]$ нечетно, а затем с отрезка $[-L, L]$ периодически на всю ось) и, созданную функцию $\Phi(x)$ подставить в (4), то получим решение рассматриваемой задачи [3]. Функция $u(x, t)$, рассматриваемая только для $0 \leq x \leq L$, будет удовлетворять обоим начальным и обоим граничным условиям.

Нечетное периодическое продолжение $\Phi(x)$ любой функции $\varphi(x)$, заданной на отрезке $0 \leq x \leq L$, на всю вещественную ось можно получить по формуле [3]

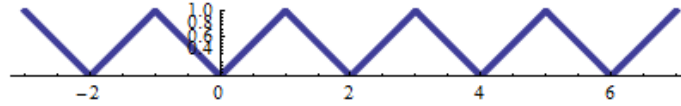
$$\Phi(x) = (-1)^{\left[\frac{x}{L}\right]} \varphi(\text{stc}(x, 2L)), \quad (6)$$

где $[x]$ (квадратные скобки) представляет функцию взятия наибольшего целого, не превосходящего x , а $\text{stc}(x, w)$ является периодической пилообразной непрерывной кусочно-линейной функцией, определяемой формулой

$stc(x, w) = \left| x - w \left[\frac{x}{w} + \frac{1}{2} \right] \right|$ или $stc(x, w) = \frac{w}{2\pi} \arccos \left(\cos \frac{2\pi x}{w} \right)$, где w является периодом этой функции.

Stc[x_, w_] = Abs[x - w Floor[x/w + 1/2]];

Plot[Stc[x, 2], {x, -3, 7}, , AspectRatio → Automatic]



Из (4) имеем

$$u(x, t) = \frac{(-1)^{\left\lfloor \frac{x+at}{L} \right\rfloor} \varphi(stc(x + at, 2L)) + (-1)^{\left\lfloor \frac{x-at}{L} \right\rfloor} \varphi(stc(x - at, 2L))}{2} \quad (7)$$

Пусть $L=1$ и начальная функция $\varphi(x)$ имеет треугольную форму, как показано на следующем рисунке слева. Следующий код моделирует движение точек струны, смещение $u(x, t)$ которых задается формулой (7).

DynamicModule[{L = 1, φ , F, u},

$\varphi[x_] = 1/2 - \text{Abs}[x - 1/2];$

$F[x_] = (-1)^{\text{Floor}[x/L]} \varphi[\text{Stc}[x, 2L]];$

$u[x_, t_] = \frac{F[x + t] + F[x - t]}{2};$

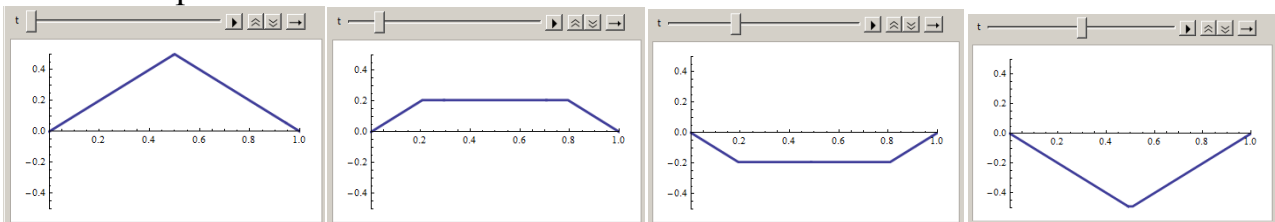
Animate[

Plot[Evaluate[u[x, t]], {x, 0, L}, PlotStyle → Thickness[0.01],

PlotPoints → 1000, PlotRange → {{0, L}, {-0.5, 0.5}},

{t, 0, 2}, AnimationRunning → False, DisplayAllSteps → True]]

Ниже показана панель анимации в начальный и несколько последующих моментов времени.



Формула (7) дает решение задачи о колебании конечной струны длины L , закрепленной на концах, имеющей нулевую начальную скорость и начальное смещение $u(x, 0) = \varphi(x)$. Приведем несколько примеров колебаний при различных функциях $\varphi(x)$. Во всех примерах левые панели анимации показаны для момента времени $t=0$, т.е. на них показаны начальные профили струн.

DynamicModule[{L = 4, f, F, u},

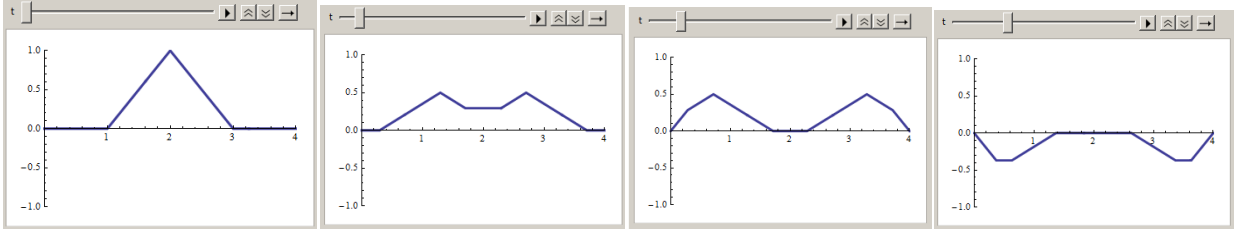
$f[x_] = \frac{1}{2} \text{Abs}[x - 1] - \text{Abs}[x - 2] + \frac{1}{2} \text{Abs}[x - 3];$

$F[x_] = (-1)^{\text{Floor}[x/L]} f[\text{Stc}[x, 2L]];$

$u[x_, t_] = \frac{F[x + t] + F[x - t]}{2};$

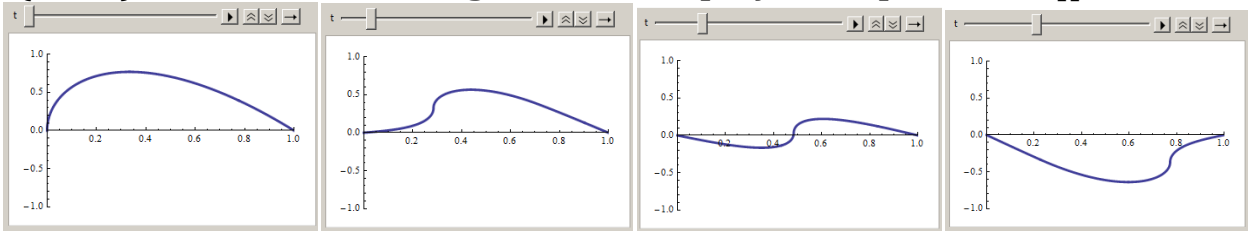
Animate[

**Plot[Evaluate[u[x, t]], {x, 0, L}, PlotStyle → Thickness[0.01],
 PlotPoints → 1000, PlotRange → {{0, L}, {-1, 1}},
 {t, 0, 8}, AnimationRunning → False, DisplayAllSteps → True]]**



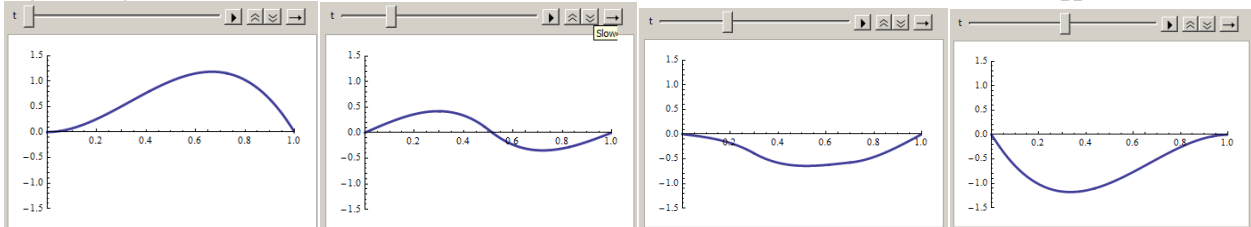
**DynamicModule[{L = 1, f, F, u},
 f[x_] = $2\sqrt{x}(1 - x)$;
 F[x_] = $(-1)^{\text{Floor}[x/L]}f[\text{Stc}[x, 2L]]$;
 u[x_, t_] = $\frac{F[x + t] + F[x - t]}{2}$;**

**Animate[
 Plot[Evaluate[u[x, t]], {x, 0, L}, PlotStyle → Thickness[0.01],
 PlotPoints → 1000, PlotRange → {{0, L}, {-1, 1}},
 {t, 0, 2}, AnimationRunning → False, DisplayAllSteps → True]]**



**DynamicModule[{L = 1, f, F, u},
 f[x_] = $8x^2(1 - x)$;
 F[x_] = $(-1)^{\text{Floor}[x/L]}f[\text{Stc}[x, 2L]]$;
 u[x_, t_] = $\frac{F[x + t] + F[x - t]}{2}$;**

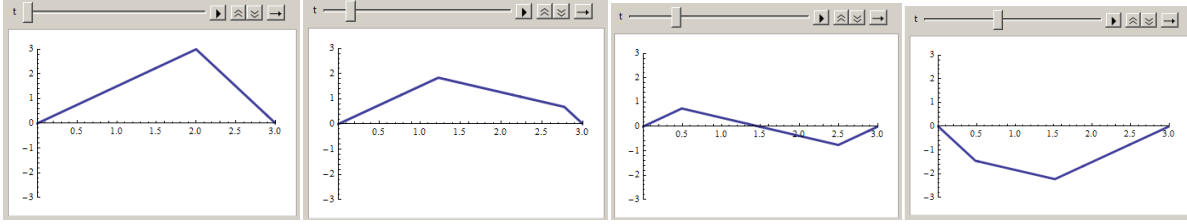
**Animate[
 Plot[Evaluate[u[x, t]], {x, 0, L}, PlotStyle → Thickness[0.01],
 PlotPoints → 1000, PlotRange → {{0, L}, {-1.5, 1.5}},
 {t, 0, 2}, AnimationRunning → False, DisplayAllSteps → True]]**



**DynamicModule[{L = 3, f, F, u},
 f[x_] = $-\frac{3}{4}x + \frac{9}{2} - \frac{9}{4}\text{Abs}[x - 2]$;
 F[x_] = $(-1)^{\text{Floor}[x/L]}f[\text{Stc}[x, 2L]]$;
 u[x_, t_] = $\frac{F[x + t] + F[x - t]}{2}$;**

Animate[

**Plot[Evaluate[u[x, t]], {x, 0, L}, PlotStyle → Thickness[0.01],
 PlotPoints → 1000, PlotRange → {{0, L}, {-3, 3}},
 {t, 0, 6}, AnimationRunning → False, DisplayAllSteps → True]]**



Заметим, что во всех приведенных выше примерах используется определение функции $stc(x, w)$, приведенное нами ранее.

Пример 5.5. Решим задачу о колебание конечной струны с закрепленным левым концом и свободным правым. Для этого надо решить уравнение колебаний (1) с начальными условиями (2) и граничными условиями

$$u(0, t) = 0, u'_x(L, t) = 0 \quad (8)$$

Известно, если начальные данные в задаче о распространении колебаний на неограниченной прямой являются нечетными функциями относительно некоторой точки x_0 , то соответствующее решение в этой точке равно нулю.

Если начальные данные в задаче о распространении колебаний на неограниченной прямой являются четными функциями относительно некоторой точки x_0 , то производная по x соответствующего решения в этой точке равна нулю.

Рассмотрим функцию $\varphi(x)$ заданную на отрезке $[0, L]$ такую, что $\varphi(0) = 0$. Продолжим ее четно относительно точки $x = L$ на отрезок $[L, 2L]$, т.е. построим функцию $\varphi_1(x)$, $x \in [0, 2L]$ такую, что для $x \in [0, L]$, $\varphi_1(x) = \varphi(x)$, а для $x \in [L, 2L]$ $\varphi_1(x) = \varphi(2L - x)$. Функцию $\varphi_1(x)$ с отрезка $[0, 2L]$ продолжим нечетно на отрезок $[-2L, 0]$, т.е. построим функцию $\varphi_2(x)$, $x \in [-2L, 2L]$ такую, что для $x \in [0, 2L]$, $\varphi_2(x) = \varphi_1(x)$, а для $x \in [-2L, 0]$, $\varphi_2(x) = -\varphi_1(-x)$. Полученную функцию $\varphi_2(x)$ с отрезка $[-2L, 2L]$ продолжим периодически на всю ось с периодом $4L$. В результате получим периодическую функцию $\Phi(x)$ которая на отрезке $x \in [0, L]$ совпадает с $\varphi(x)$, является четной относительно точки $x = L$ и нечетной относительно точки $x = 0$. Назовем описанное продолжение функции $\varphi(x)$ четно-нечетным периодическим продолжением. В силу сказанного выше, подстановка такой функции $\Phi(x)$ в формулу (4) даст решение поставленной задачи (1), (2), (8).

Известно [4], что для любой функции $\varphi(x)$ заданной на отрезке $[0, L]$ ($L > 0$), такой что $\varphi(0) = 0$, формула четно-нечетного периодического продолжения имеет следующий вид

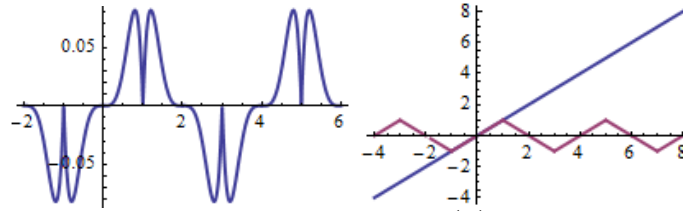
$$\Phi(x) = (-1)^{\left[\frac{x}{2L}\right]} \varphi(stc(x, 2L)) \quad (9)$$

Вот пример четно-нечетного продолжения функции $f(x) = x^4(1-x)$ с отрезка $[0,1]$ на всю вещественную ось (следующий рисунок слева)

$L = 1; f[x_] = x^4(L - x);$

$F[x_] = (-1)^{\text{Floor}[x/(2L)]}f[\text{Stc}[x, 2L]];$

$\text{Plot}[F[x], \{x, -2, 6\}, \text{PlotRange} \rightarrow \text{All}, \text{PlotStyle} \rightarrow \text{Thickness}[0.01]]$



Четно – нечетное продолжение функции $f(x) = x$ с отрезка $[0,1]$ на всю вещественную ось показано на предыдущем рисунке справа, где кроме продолженной функции, для сравнения мы изобразили и исходную.

Таким образом, решением задачи (1), (2), (8) при $\psi(x) = 0$ будет функция [4]

$$u(x, t) = \frac{(-1)^{\left\lfloor \frac{x+at}{2L} \right\rfloor} \varphi(\text{stc}(x + at, 2L)) + (-1)^{\left\lfloor \frac{x-at}{2L} \right\rfloor} \varphi(\text{stc}(x - at, 2L))}{2} \quad (10)$$

Пусть $L=4$ и начальная функция $\varphi(x) = x$. На следующем рисунке слева показана начальная форма струны. Код, приведенный ниже, моделирует движение точек струны, смещение которых $u(x, t)$ задается формулой (10).

DynamicModule[$\{L = 4, f, F, u\}$,

$f[x_] = x;$

$F[x_] = (-1)^{\text{Floor}[x/(2L)]}f[\text{Stc}[x, 2L]];$

$u[x_, t_] = \frac{F[x + t] + F[x - t]}{2};$

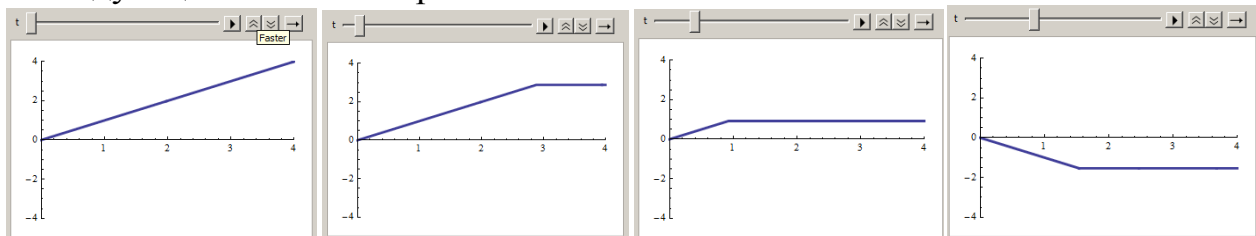
Animate[

$\text{Plot}[\text{Evaluate}[u[x, t]], \{x, 0, L\}, \text{PlotStyle} \rightarrow \text{Thickness}[0.01],$

$\text{PlotPoints} \rightarrow 1000, \text{PlotRange} \rightarrow \{\{0, L\}, \{-L, L\}\},$

$\{t, 0, 16\}, \text{AnimationRunning} \rightarrow \text{False}, \text{DisplayAllSteps} \rightarrow \text{True}]]$

На следующих рисунках показана панель анимации в начальный и несколько последующих моментов времени.



Заметим, что в коде используется определение функции $\text{stc}(x, w)$, приведенное нами ранее.

Пример 5.6. Продольные колебания стержня постоянного поперечного сечения описываются уравнением (1) с начальными условиями (2) и граничными условиями, зависящими от способа закрепления концов [5].

Пусть круглый стержень длиной L , закрепленный на одном конце и свободный на другом, подвержен линейному растяжению. Начало координат расположим в центре левого (фиксированного) сечения, и ось X направим вправо вдоль оси стержня. Начальное смещение будет иметь вид $u(x,0) = \varphi(x) = kx$, где k некоторая постоянная. В начальный момент времени растянутый конец освобождается и стержень начинает сжиматься, а затем удлиняться – начинаются продольные колебания. Такие колебания будут описываться уравнением (1), где $u(x,t)$ является продольным смещением в момент времени t сечения стержня, имеющего в нулевой момент времени координату x . Начальные условия будут иметь вид

$$u(x,0) = kx, \quad u_t(x,0) = 0, \quad (11)$$

а граничные условия будут иметь вид (8). Решение этой задачи нами построено в предыдущем примере. Здесь мы смоделируем движение круглого стержня, сечения которого смещаются в соответствии с решением $U(x,t)$ задачи (1), (8), (11). Для функции решения мы используем обозначение U , чтобы не путать его с параметром u в уравнениях поверхности.

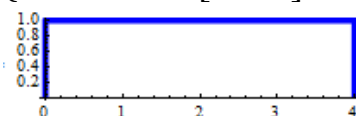
Пусть параметрические уравнения поверхности стержня в состоянии покоя (нулевой момент времени) имеют вид $x = x(u,v)$, $y = y(u,v)$, $z = z(u,v)$. Все точки одного и того же поперечного сечения стержня смещаются в момент t на одну и ту же величину $U(x,t)$. Поэтому в момент t они будут иметь продольную координату $X = x(u,v) + U(x(u,v),t)$. В плоскости сечения смещения точек нет, поэтому y и z координаты точек сечения не меняются. В результате в момент времени t точки стержня, имевшие при $t=0$ координаты $x = x(u,v)$, $y = y(u,v)$, $z = z(u,v)$, будут иметь координаты

$$X = x(u,v) + U(x(u,v),t), \quad Y = y(u,v), \quad Z = z(u,v). \quad (12)$$

В следующем коде мы моделируем продольные колебания круглого стержня радиуса r , точки поверхности которого в момент t имеют координаты (12).

Круглый стержень в п.2.2.3 мы строили, как поверхность вращения ломаной, имеющей форму «скобы». Параметрическое уравнение «скобы» имеет вид

```
L = 4; r = 1;
xc[t_] =  $\frac{L}{2} + \frac{1}{2}\text{Abs}[t - r] - \frac{1}{2}\text{Abs}[t - r - L];$ 
zc[t_] =  $\left(r + \frac{L}{2}\right) - \frac{1}{2}\text{Abs}[t - r] - \frac{1}{2}\text{Abs}[t - r - L];$ 
ParametricPlot[{xc[t], zc[t]}, {t, 0, L + 2r},
PlotStyle → {Thickness[0.02], Blue}]
```

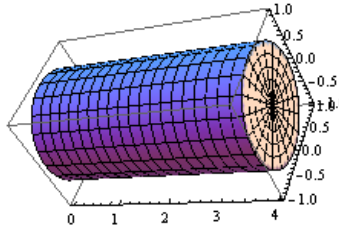


Параметрическое уравнение поверхности вращения кривой $x_c(u)$, $y_c(u)$ вокруг оси X имеет вид $x(u,v) = x_c(u)$, $y(u,v) = y_c(u)\cos v$, $z(u,v) = y_c(u)\sin v$.

```

x[u_, v_] = xc[u];
y[u_, v_] = Cos[v]zc[u];
z[u_, v_] = Sin[v]zc[u];
ParametricPlot3D[{x[u, v], y[u, v], z[u, v]}, {u, 0, L + 2r}, {v, 0, 2π},
  Mesh → Full, PlotPoints → {31, 21}, BoxRatios → {L, 2r, 2r}]

```



Записываем определение координатных функций (12).

```

Stc[x_, w_] = Abs[x - w Floor[x/w + 1/2]];

```

```

f[x_] = x/L;

```

```

F[x_] = (-1)^Floor[x/(2L)] f[Stc[x, 2L]];

```

```

U[x_, t_] = (F[x + t] + F[x - t]) / 2;

```

```

X[u_, v_, t_] = x[u, v] + U[x[u, v], t];

```

```

Y[u_, v_, t_] = y[u, v];

```

```

Z[u_, v_, t_] = z[u, v];

```

Создаем список `tc` изображений стержня в различные моменты времени и анимируем его с помощью функции `ListAnimate`.

```

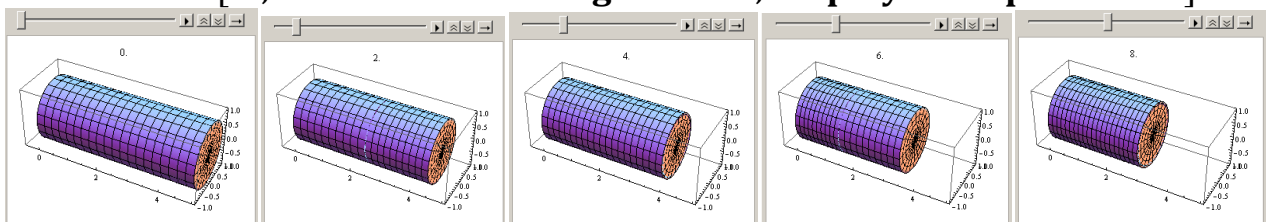
tc = Table[ParametricPlot3D[
  {X[u, v, t], Y[u, v, t], Z[u, v, t]}, {u, 0, L + 2r}, {v, 0, 2π},
  Mesh → Full, PlotPoints → {31, 21},
  PlotRange → {{-0.5, L + 1.1}, {-r, r}, {-r, r}}, PlotLabel → t],
  {t, 0, 4L - 0.25, 0.25}];

```

```

ListAnimate[tc, AnimationRunning → False, DisplayAllSteps → True]

```



Обратите внимание на то, что стержень не просто сжимается/разжимается, но в отдельных частях сжимается/разжимается координатная сетка на его поверхности.

Причина по которой мы использовали функцию `ListAnimate`, а не `Animate`, состоит в том, что построение анимации большого количества 3D графиков занимает у системы много времени. В примере процесс разделен на два шага. Первый состоит в создании списка изображений. Второй шаг состоит воспроизведении этого вписка, что происходит более гладко.

Отметим, что приведенный код не собран нами в динамический модуль для того, чтобы продемонстрировать результаты работы отдельных блоков кода. Однако, вы можете сделать это самостоятельно.

Пример 5.7. Продольные колебания пружины. Уравнение продольных колебаний для стержня и пружины записываются одинаково [2].

Пусть спиральная пружина длиной L , закрепленная на одном конце и свободная на другом, подвержена линейному растяжению. Начало координат расположим в левом краю пружины, и ось X направлена вправо вдоль ее оси. Если к свободному концу пружины приложить растягивающее усилие, то деформация будет линейной. Допустим, что усилие таково, что правый конец сместится на единицу. Тогда начальное смещение будет иметь вид $u(x,0) = x/L$. В начальный момент времени растянутый конец освобождается и пружина начинает сжиматься и разжиматься – начинаются продольные колебания. Такие колебания описываются уравнением (1), где $u(x,t)$ является продольным смещением в момент времени t точек пружины, имевших в нулевой момент времени координату x .

Решение уравнения (1) с начальными условиями (11) и граничными условиями (8) нами уже построено ранее. Здесь мы смоделируем движение спиральной пружины, точки которой смещаются в соответствии с решением $U(x,t)$ задачи (1), (8), (11).

Пусть параметрические уравнения спиральной кривой, представляющей пружину, имеют вид $x = x(u)$, $y = y(u)$, $z = z(u)$. Точка пружины в момент времени t смещается в продольном направлении на величину $U(x,t)$. Поэтому в момент t она будет иметь новую координату $X = x(u) + U(x(u),t)$. Координаты y и z точки не меняются. В результате в момент времени t точка пружины, имевшая при $t=0$ координаты $x = x(u)$, $y = y(u)$, $z = z(u)$, будет иметь координаты

$$X = x(u) + U(x(u),t), \quad Y = y(u), \quad Z = z(u). \quad (13)$$

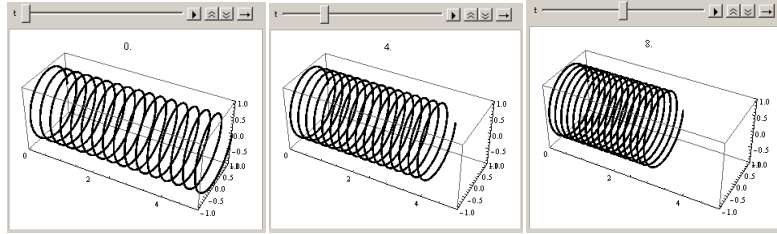
В следующем коде мы моделируем продольные колебания пружины радиуса r , точки которой в момент t имеют координаты (13). Пружину рисуем в виде пространственной круговой спиральной кривой длиной $L=4$ с $n=4$ витками на единице длины.

```
DynamicModule[{L = 4, n = 4, x, y, z, f, F, U, X, Y, Z},
  x[w_] = w;
  y[w_] = Cos[2 * Pi * w * n];
  z[w_] = Sin[2 * Pi * w * n];
  f[x_] =  $\frac{z}{L}$ ;
  F[x_] = (-1)^Floor[x/(2L)] f[Stc[x, 2L]];
  U[x_, t_] =  $\frac{F[x - t] + F[x + t]}{2}$ ;
  X[w_, t_] = x[w] + U[x[w], t];
  Y[w_, t_] = y[w];
  Z[w_, t_] = z[w];
  Animate[
    ParametricPlot3D[Evaluate[{X[w, t], Y[w, t], Z[w, t]}], {w, 0, L},
```

PlotStyle → **Thickness[0.01]**, **AspectRatio** → **Automatic**,
PlotPoints → **1000**, **PlotRange** → **{{0, L + 1}, {-1, 1}, {-1, 1}}**,
PlotLabel → **t**,

{t, 0, 4L}, **AnimationRunning** → **False**, **DisplayAllSteps** → **True**]]

Форма пружины в моменты времени $t = 0, 4, 8$ показана на следующем рисунке.



Пример 5.8. Крутильные колебания кругового стержня.

Рассмотрим круглый вертикальный стержень длиной L и радиуса r , зафиксированный в нижнем сечении и закрученный парой сил в верхнем. Круговые сечения стержня остаются круговыми в процессе закручивания, а диаметр сечений и расстояния между ними не меняется при условии, что угол закручивания мал. Начало координат расположим в центре основания стержня и ось Z направим вдоль его оси вертикально вверх. В начальный момент верхнее сечение освобождается и в стержне возникают крутильные колебания. Известно [5], что для поперечного сечения с координатой z угол поворота $\theta(z, t)$ удовлетворяет волновому уравнению (1).

Начальный угол поворота $\theta(z, 0)$ будет линейно меняться вдоль стержня $\theta(z, 0) = k z$, где постоянная k зависит от свойств материала стержня и пары закручивающих сил. Пусть в параметрических уравнениях поверхности цилиндра $x(u, v)$, $y(u, v)$, $z(u, v)$ параметр v представляет угол. Тогда в уравнениях поверхности закрученного цилиндра этот параметр должен быть увеличен на величину угла θ . Угол поворота сечений является функцией координаты z и момента времени t , т.е. $\theta = U(z, t)$. В результате уравнения поверхности закрученного стержня будут иметь вид

$$X = x(u, v + U(z(u, v), t)), \quad Y = y(u, v + U(z(u, v), t)), \quad Z = z(u, v + U(z(u, v), t)), \quad (14)$$

а функцию $U(z, t)$ следует искать как решения волнового уравнения (1) с начальными условиями $U(z, 0) = k z$, $U'_t(z, 0) = 0$ и граничными условиями $U(0, t) = 0$ (нижнее сечение закреплено, поворота нет), $U'_x(L, t) = 0$ (верхнее сечение свободно) [5]. Решение этой задачи мы уже построили в примере 5.5. Используем его для графического представления крутильных колебаний.

Вначале составляем параметрические уравнения поверхности цилиндрического стержня, как поверхности вращения ломаной в форме «скобы» вокруг оси Z .

$L = 6; r = 2;$

$xc[t_]= (r + \frac{L}{2}) - \frac{1}{2}Abs[t - r] - \frac{1}{2}Abs[t - r - L];$

$zc[t_]= \frac{L}{2} + \frac{1}{2}Abs[t - r] - \frac{1}{2}Abs[t - r - L];$

ParametricPlot[{**xc**[**t**], **zc**[**t**]}, {**t**, 0, **L** + 2**r**},

PlotStyle → {**Thickness**[0.02], **Blue**}] (* рисунок скобы слева *)

x[**u**_, **v**_] = **Cos**[**v**]**xc**[**u**];

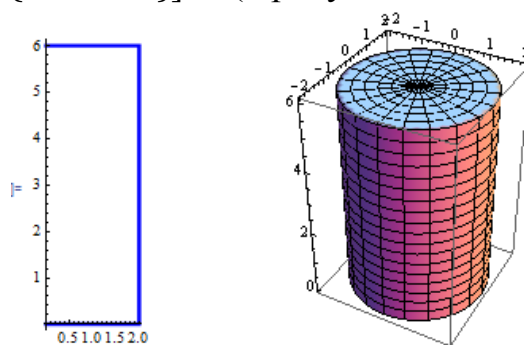
y[**u**_, **v**_] = **Sin**[**v**]**xc**[**u**];

z[**u**_, **v**_] = **zc**[**u**];

ParametricPlot3D[**Evaluate**[{**x**[**u**, **v**], **y**[**u**, **v**], **z**[**u**, **v**]},

{**u**, 0, **L** + 2**r**}, {**v**, 0, 2**π**}, **Mesh** → **Full**, **PlotPoints** → {31, 21},

BoxRatios → {2**r**, 2**r**, **L**}] (* рисунок цилиндра справа *)



Затем строим решение уравнения крутильных колебаний (1) с начальными $U(z,0)=z/L, U'_t(z,0)=0$ и граничными $U(0,t)=0, U'_x(L,t)=0$ условиями.

Stc[**x**_, **w**_] = **Abs**[**x** - **wFloor**[**x**/**w** + 1/2]];

f[**x**_] = **x**/**L**;

F[**x**_] = (-1)^**Floor**[**x**/(2**L**)]**f**[**Stc**[**x**, 2**L**]];

$U[x_ , t_]= \frac{F[x + t] + F[x - t]}{2};$

Составляем параметрические уравнения поверхности закрученного стержня в соответствии с формулами (14)

X[**u**_, **v**_, **t**_] = **x**[**u**, **v** + **U**[**z**[**u**, **v**], **t**]];

Y[**u**_, **v**_, **t**_] = **y**[**u**, **v** + **U**[**z**[**u**, **v**], **t**]];

Z[**u**_, **v**_, **t**_] = **z**[**u**, **v** + **U**[**z**[**u**, **v**], **t**]];

Создаем список **tc** изображений стержня в различные моменты времени и анимируем его с помощью функции **ListAnimate**.

tc = **Table**[**ParametricPlot3D**[

Evaluate[{**X**[**u**, **v**, **t**], **Y**[**u**, **v**, **t**], **Z**[**u**, **v**, **t**]}, {**u**, 0, **L** + 2**r**}, {**v**, 0, 2**π**},

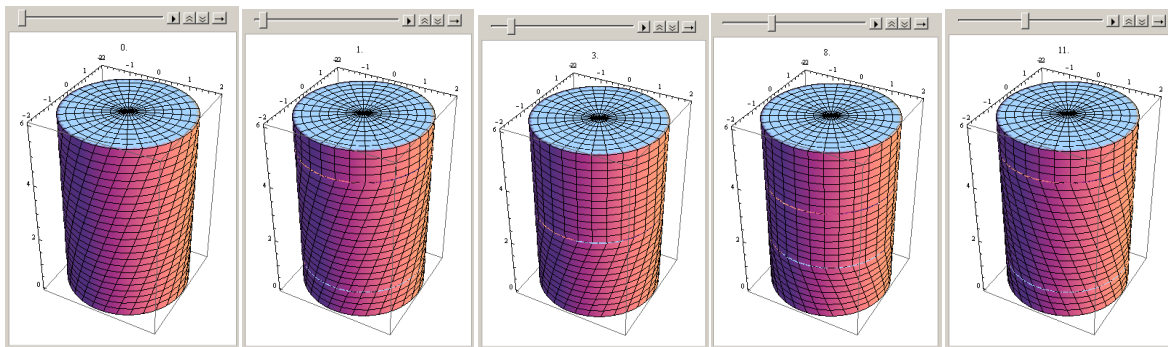
PlotPoints → {41, 31}, **Mesh** → **Full**,

PlotRange → {{-**r**, **r**}, {-**r**, **r**}, {-0.1, **L** + 0.1}}, **PlotLabel** → **t**],

{**t**, 0, 4**L** - 0.25, 0.25}];

ListAnimate[**tc**, **AnimationRunning** → **False**, **DisplayAllSteps** → **True**]

Изображения крутящегося цилиндра в начальный и некоторые последующие моменты времени представлены на следующем рисунке.



Обратите внимание, что геометрическое тело – цилиндр остается цилиндром, однако меняется координатная сеть на его поверхности. Она показывает, как закручены те или иные сечения цилиндра.

2.4.6 Задачи сопротивления материалов и теории упругости

Пример 6.1. Изгиб балки равномерно распределенной нагрузкой.

Пусть начало координат находится на левом конце балки и ось X направим вдоль оси балки вправо, ось Y направим вертикально вверх. Под действием приложенных к балке вертикальных сил она прогибается.

Уравнение изгибающего момента балки имеет вид

$$\frac{d^2 M}{dx^2} = -q, \quad (1)$$

где q внешняя сила, рассчитанная на единицу длины, а $M(x)$ изгибающий момент в балке. Для определения прогиба балки нужно решить уравнение

$$E J \frac{d^2 u(x)}{dx^2} = -M(x), \quad (2)$$

где J – момент инерции поперечного сечения балки, E – модуль упругости материала балки, $u(x)$ – отклонение нейтральной оси балки от положения равновесия. Из (1) и (2) получаем дифференциальное уравнение прогиба

$$\frac{d^2}{dx^2} \left(E J \frac{d^2 u}{dx^2} \right) = q(x) \quad (3)$$

Для свободно опертой на краях балки граничные условия имеют вид

$$u(0) = 0, u(L) = 0, u''(0) = 0, u''(L) = 0 \quad (4)$$

При ином закреплении граничные условия будут другими.

Рассмотрим балку длины L и высотой/толщиной h , свободно опертую на концах. Известно [6], что прогиб такой балки определяется по формуле

$$u(x) = \frac{q}{24 E J} (L^3 x - 2 L x^3 + x^4), \quad (5)$$

и

$$M(x) = \frac{q x}{2} (L - x) \quad (6)$$

Легко проверить, что функция $u(x)$ удовлетворяет дифференциальному уравнению (3) и граничным условиям (4), а функция $M(x)$ получается из (2) подстановкой в нее функции $u(x)$.

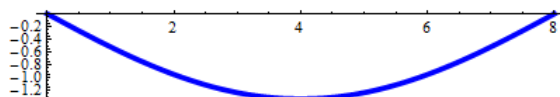
В соответствии с (5) нагруженная балка может быть представлена как изогнутый отрезок.

$x = .; L = 8; h = 2; Em = 10; J = 4; q = -1; (* Em - модуль упругости *)$

$U[x_] = q/(24 * Em * J) * (L^3 * x - 2 * L * x^3 + x^4);$

$Plot[U[x], \{x, 0, L\}, PlotStyle \rightarrow \{Thickness[0.01], Blue\},$

$AspectRatio \rightarrow Automatic]$

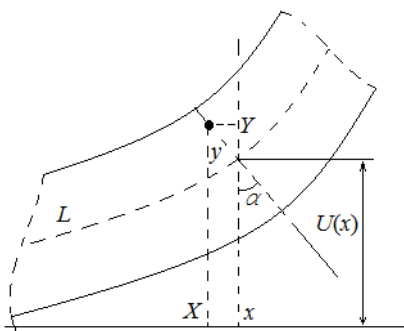


Здесь для модуля упругости мы использовали идентификатор Em , поскольку символ E в системе защищен (он используется для обозначения основания натуральных логарифмов). Нагрузке q мы присвоили отрицательное значение, поскольку она направлена вниз, а ось Y мы направили вверх.

После выполнения одномерного расчета смоделируем поле напряжений внутри балки. Для этого балку рассмотрим как двумерную область, параметрические уравнения которой имеют вид

$$x(u, v) = u, \quad y(u, v) = v \quad (0 \leq u \leq L, -h/2 \leq v \leq h/2).$$

Используя уравнения плоской области $x(u, v)$, $y(u, v)$ недеформированной балки и уравнение $U(x)$ ее нейтральной оси, можно написать параметрическое уравнение области деформированной балки. В соответствии с гипотезой плоских сечений, все поперечные сечения, которые были перпендикулярны нейтральной оси балки до деформирования, остаются перпендикулярными нейтральной оси балки после ее деформирования.



Точки, имевшие до деформирования координаты (x, y) , переходят в точки с координатами (X, Y) , где $X = x - y \sin \alpha$, $Y = U(x) + y \cos \alpha$ (см. рисунок). Но

$\tan \alpha = U'(x)$ и, поэтому, $X = x - \frac{U'_x(x)y}{\sqrt{1 + (U'_x(x))^2}}$ и $Y = U(x) + \frac{y}{\sqrt{1 + (U'_x(x))^2}}$. Тогда

функции $X(u, v)$, $Y(u, v)$, представляющие параметрические уравнения области изогнутой балки, будут иметь вид

$$\begin{aligned} X(u, v) &= x(u, v) - \frac{U'_x(x(u, v))y(u, v)}{\sqrt{1 + (U'_x(x(u, v)))^2}}, \\ Y(u, v) &= U(x(u, v)) + \frac{y(u, v)}{\sqrt{1 + (U'_x(x(u, v)))^2}} \end{aligned} \quad (7)$$

Строим область деформированной балки.

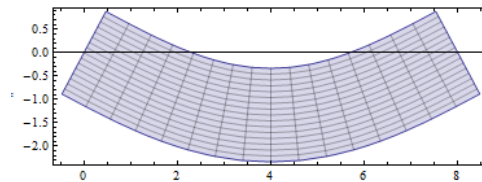
$x[u_, v_] = u; \quad y[u_, v_] = v;$

$U1 = U';$

$X[u_, v_] = x[u, v] - \frac{U1[x[u, v]] y[u, v]}{\sqrt{1 + U1[x[u, v]]^2}};$

$Y[u_, v_] = U[x[u, v]] + \frac{y[u, v]}{\sqrt{1 + U1[x[u, v]]^2}};$

ParametricPlot[[{X[u, v], Y[u, v]}, {u, 0, L}, {v, -h/2, h/2}]



Следующим шагом будет показать в цвете продольные напряжения σ_x в балке.

Если поперечное сечение балки прямоугольное, то σ_x вычисляются по формуле

$\sigma_x(x, y) = \frac{M(x) \cdot y}{J}$, где $M(x)$ – момент инерции в сечении x балки, y –

вертикальное смещение точки балки относительно нейтральной оси. Чтобы использовать эту функцию в качестве функции цвета, ее нужно привести к диапазону $[0, 1]$. Для этого нужно знать минимальные и максимальные значения напряжения σ_x . В нашей задаче они, очевидно, равны напряжениям в срединном сечении балки в верхней и нижней точках

$\sigma_{x \max/\min} = \pm \frac{M(L/2) \cdot (h/2)}{J}$, где h – высота балки.

Создаем функцию продольных напряжений S_x и вычисляем ее максимальное и минимальное значения S_{\max} и S_{\min} . Они используются для построения шкалы цветов напряжения справа от рисунка. Затем рисуем область изогнутой балки, раскрашенную в соответствии со значениями функции продольных напряжений S_x .

$M = .; M[x_] = \frac{q x(L - x)}{2};$

$Sx[u_, v_] = M[x[u, v]]y[u, v]/J;$

$Smax = Sx[L/2, h/2]$

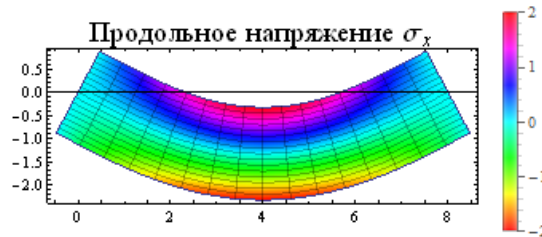
$Smin = Sx[L/2, -h/2]$

2

-2

ParametricPlot[[{X[u, v], Y[u, v]}, {u, 0, L}, {v, -h/2, h/2},

ColorFunction \rightarrow **Function** $\left[\{x, y, u, v\}, \text{Hue}[\text{Rescale}[\text{Sx}[u, v], \{-2, 2\}]]\right]$,
ColorFunctionScaling \rightarrow **False**,
PlotLegends \rightarrow **BarLegend** $\left[\{\text{Hue}, \{-2, 2\}\}\right]$,
PlotLabel \rightarrow **Style** $["\text{Продольное напряжение } \sigma_x, 18]$



Здесь функция `Rescale` используется для приведения диапазона $[-2, 2]$ значений функции S_x к диапазону $[0, 1]$, который требуется для функции `Hue`. Опция `ColorFunctionScaling` \rightarrow `False` нужна для того, чтобы система не пыталась приводить аргументы функции `ColorFunction` к диапазону $[0, 1]$, что она делает по умолчанию.

Пример 6.2. Действие сосредоточенной силы на упругую полуплоскость.

Рассмотрим плоскую упругую среду, ограниченную сверху горизонтальной прямой. В некоторой точке к границе приложена вертикальная сосредоточенная сила P . Ось X направим вправо вдоль границы, а ось Y из точки приложения силы вертикально вверх. Среда расположена в полуплоскости $y < 0$. Вертикальные напряжения Y_y в полуплоскости вычисляются по формулам [8]

$$Y_y = \frac{2P}{\pi} \frac{y^3}{(x^2 + y^2)^2} \quad (8)$$

Напряжение Y_y представляет отнесенную к единице площади силу, направленную вдоль оси Y , приложенную к горизонтальной площадке (перпендикулярной направлению y).

Напряжение Y_y стремится к минус бесконечности в точке приложения силы. Чтобы его отобразить в цвете на полуплоскости диапазон изменения функции Y_y необходимо ограничить. Для этого создадим обрезанную снизу функцию напряжений $TensYy(x, y)$, используя функцию `Piecewise`. Значение обрезания подбираем из «эстетических» соображений.

PP = 1;

$$Yy[x_, y_] = \frac{2PP}{\pi} \frac{y^3}{(x^2 + y^2)^2};$$

TensYy $[x_, y_] = \text{Piecewise}[\{\{Yy[x, y], Yy[x, y] > -1.6\}, -1.6];$

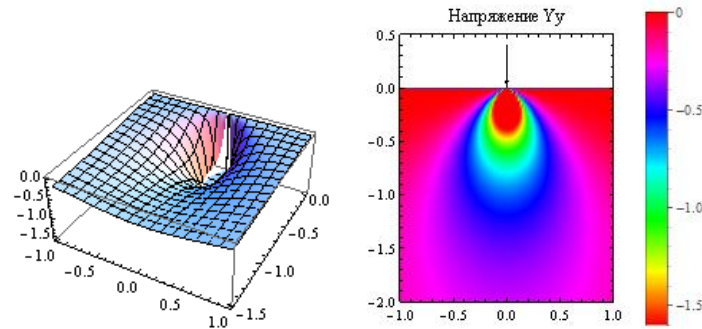
Plot3D $[TensYy[x, y], \{x, -1, 1\}, \{y, 0, -1.5\}]$ (* следующий рисунок слева *)

Теперь строим полуплоскость с окраской, соответствующей значениям функции $TensYy$, предварительно нормируя ее к диапазону $[0, 1]$ (следующий рисунок справа)

```

p = ParametricPlot[{{u, v}}, {u, -1., 1.}, {v, -2, 0.5},
  PlotPoints → 100, Mesh → None,
  RegionFunction → Function[{x, y, u, v}, y < 0],
  PlotRange → {{-1., 1.}, {-2, 0.5}},
  ColorFunction → Function[{x, y, u, v},
    Hue[Rescale[TensYy[u, v], {-1.6, 0}]]],
  ColorFunctionScaling → False,
  PlotLegends → BarLegend[{Hue, {-1.6, 0}}],
  Axes → None, PlotLabel → "Напряжение Yy"];

```



На последнем рисунке ярко красная зона (под точкой приложения нагрузки) соответствует большим отрицательным (сжимающим) напряжениям, а холодный красный (у поверхности вдали от точки приложения нагрузки) соответствует близким к нулю напряжениям. Это показывает шкала цветов, построенная справа от графика.

Известно [7], что любой элемент среды, расположенный на расстоянии r от точки приложения силы, подвергается простому сжатию в радиальном направлении. При этом компоненты напряжения определяются формулами

$$\sigma_r = \frac{2P \cos \theta}{\pi r}, \quad \sigma_\theta = \tau_{r\theta} = 0, \quad \text{где } \theta - \text{угол, образуемый радиус-вектором точки с вертикалью.}$$

Выберем окружность произвольного диаметра d с центром на вертикальной оси и касательную к поверхности полуплоскости. Для любой точки этой окружности имеем $d \cos \theta = r$. Отсюда, согласно выражению для

$$\text{напряжения } \sigma_r, \text{ получаем } \sigma_r = \frac{2P}{\pi d}, \text{ т.е. напряжения } \sigma_r \text{ во всех точках}$$

окружности остаются одинаковыми, за исключением точки приложения нагрузки. Проиллюстрируем этот факт графически. Для этого изобразим

$$\text{напряжения } \sigma_r(x, y) = \frac{2P \cos \theta}{\pi r} = \frac{2P y}{\pi(x^2 + y^2)} \text{ в области полуплоскости в виде}$$

контурного графика

$$Sr[x, y] = \frac{2 P y}{\pi(x^2 + y^2)};$$

```

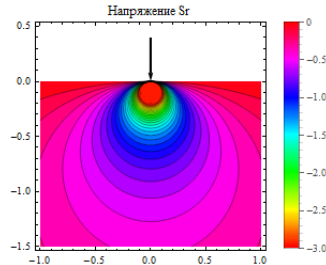
p = ContourPlot[TensSr[x, y], {x, -1, 1}, {y, -1.5, 0.5},
  RegionFunction → Function[{x, y, f}, y < 0],
  Contours → Table[pp, {pp, -3, 0, 0.1}],

```

```

PlotRange → {-3, 0}, ClippingStyle → Automatic,
ColorFunction → Function[{f}, Hue[Rescale[f, {-3, 0}]]],
ColorFunctionScaling → False, Axes → None,
PlotLegends → Automatic];
ar = Graphics[{Thickness[0.01], Arrow[{{0, 0.4}, {0, 0}}]}];
Show[p, ar, PlotLabel → "Напряжение  $\sigma_r$ "]

```



Как видно, линии уровня функции $S_r(x, y)$ (напряжения σ_r) являются окружностями. Ярко красный круг под точкой приложения нагрузки представляет область обрезания, в которой $\sigma_r \leq -3$.

Обратите внимание на следующие особенности этого кода. Опция `PlotRange → {-3, 0}` задает диапазон изменения значений функции $S_r(x, y)$ и, тем самым, определяет область отсекаемых значений. Чтобы отсекаемые зоны изображались как часть поверхности, а не как «дырки», используется опция `ClippingStyle → Automatic`. Диапазон `{-3, 0}`, задаваемый опцией `PlotRange`, также используется опцией `PlotLegends → Automatic`, которая строит контрольную цветовую полосу справа от графика.

Пример 6.3. Действие нескольких сосредоточенной силы на упругую полуплоскость.

В предыдущем примере мы изобразили сжимающие напряжения Y_y в бесконечной полуплоскости, возникающие под действием одной сосредоточенной силы. Поскольку уравнения теории упругости линейные, то напряжения, возникающие от действия нескольких сосредоточенных сил, будут суммой напряжений (8), возникающих от действия каждой из сил.

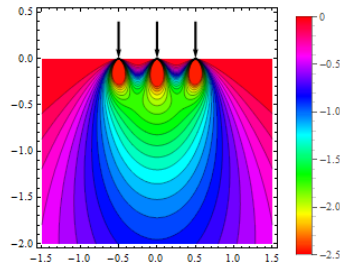
Пусть точки приложения трех сосредоточенных сил имеют координату $x = -0.5, 1.0, 0.5$. Изобразим суммарное Y_y напряжение в полуплоскости в виде контурного графика

```

PP = 1; Yy[x_, y_] =  $\frac{2PP}{\pi} \frac{y^3}{(x^2 + y^2)^2}$ ;
minYy = -2.5; (* уровень обрезания напряжения *)
SYy[x_, y_] = Yy[x, y] + Yy[x - 0.5, y] + Yy[x + 0.5, y];
p = ContourPlot[SYy[x, y], {x, -1.5, 1.5}, {y, -2, 0.5},
  RegionFunction → Function[{x, y, f}, y < 0],
  Contours → Table[pp, {pp, minYy, 0, 0.1}],

```

ColorFunction \rightarrow **Function** [**{f}**, Hue[Rescale[f, {minYy, 0}]]],
ColorFunctionScaling \rightarrow **False**, **Axes** \rightarrow **None**,
PlotRange \rightarrow {minYy, 0}, **ClippingStyle** \rightarrow **Automatic**,
PlotLegends \rightarrow **Automatic**];



Опция `PlotRange->{minYy, 0}` задает диапазон изменения значений функции $SY_y(x, y)$. Чтобы отсекаемые зоны изображались как часть поверхности используется опция `ClippingStyle->Automatic`. Опция `PlotLegends->Automatic` строит справа от графика шкалу цветов с диапазоном {minYy, 0}, который был указан в опции `PlotRange`.

Пример 6.4. Напряжения в бесконечной плоской пластинке при наличии круглого отверстия.

Решим задачу о визуализации напряжений в бесконечной пластинке с отверстием, подверженной однородному растяжению величиной P (усилие, приходящееся на единицу длины). Введем полярную систему координат (r, θ) , начало которой положим в центре отверстия радиуса a , а полярную ось направим вдоль усилия P . Известно [7], что напряжения в такой пластинке определяются по формулам

$$\begin{aligned}\sigma_r &= \frac{P}{2} \left(1 - \frac{a^2}{r^2} \right) + \frac{P}{2} \left(1 + \frac{3a^4}{r^4} - \frac{4a^2}{r^2} \right) \cos 2\theta \\ \sigma_\theta &= \frac{P}{2} \left(1 + \frac{a^2}{r^2} \right) - \frac{P}{2} \left(1 + \frac{3a^4}{r^4} \right) \cos 2\theta \\ \tau_{r\theta} &= -\frac{P}{2} \left(1 - \frac{3a^4}{r^4} + \frac{2a^2}{r^2} \right) \sin 2\theta\end{aligned}\tag{9}$$

Изобразим напряжения σ_r в виде функциональной окраски области с круговым отверстием. Для этого создадим функцию $\sigma_r = S_r(r, \theta)$

$P = 1$; $a = 1$;

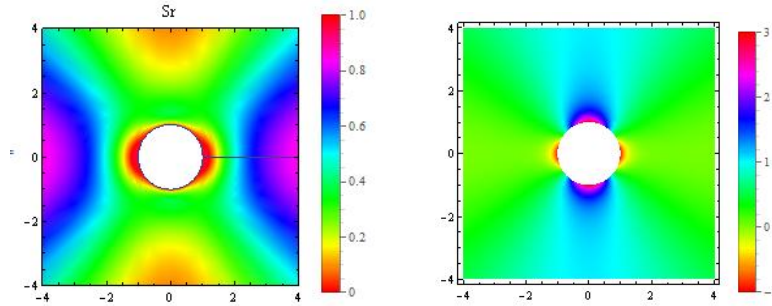
$$\text{Sr}[\mathbf{r_}, \mathbf{v_}] = \frac{P}{2} \left(1 - \frac{a^2}{r^2} \right) + \frac{P}{2} \left(1 + 3 \frac{a^4}{r^4} - 4 \frac{a^2}{r^2} \right) \text{Cos}[2v];$$

Из выражения для напряжения σ_r видно, что его минимальное и максимальное значения равны 0 и P . Эти величины мы используем при нормировке `ColorFunction` и задания диапазона значений для шкалы цветов.

```

ParametricPlot[[{uCos[v], uSin[v]}], {u, 0, 6}, {v, 0, 2π},
Mesh → None, PlotPoints → 50,
RegionFunction → Function[{x, y, u, v}, x2 + y2 > 1],
PlotRange → {{-4, 4}, {-4, 4}},
ColorFunction → Function[{x, y, u, v}, Hue[Rescale[Sr[u, v], {0, 1}]]],
ColorFunctionScaling → False,
PlotLegends → BarLegend[{Hue, {0, 1}}],
Axes → None, PlotLabel → "Sr" (* следующий рисунок слева *)

```



Напомним, что растяжение выполняется в горизонтальном направлении вдали от отверстия.

Напряжение σ_θ графически представим с помощью функции **DensityPlot**, предварительно записав его в декартовой системе координат (а не полярной, как в формулах (9)). Легко видеть, что $\cos 2\theta = \frac{x^2 - y^2}{x^2 + y^2}$. Поэтому

$$\text{StC}[x_y] = \frac{P}{2} \left(1 + \frac{a^2}{x^2 + y^2} \right) - \frac{P}{2} \left(1 + 3 \frac{a^4}{(x^2 + y^2)^2} \right) \frac{x^2 - y^2}{x^2 + y^2};$$

Минимальное и максимальное значение σ_θ равны $-P$ и $3P$. График плотности функции $\sigma_\theta = \text{StC}(x, y)$, изображенный на предыдущем рисунке справа, строится следующим образом

```

DensityPlot[StC[x, y], {x, -4, 4}, {y, -4, 4},
RegionFunction → Function[{x, y, f}, x2 + y2 > 1],
PlotPoints → 30, PlotRange → {-1, 3},
ColorFunction → Hue, PlotLegends → Automatic]

```

Нашей следующей задачей является представление продольных напряжения σ_x в области пластинки. Напряжение σ_x , заданное в декартовой системе координат, выражается через напряжения $\sigma_r, \sigma_\theta, \tau_{r\theta}$ в полярной системе по формуле $\sigma_x = \sigma_r \cos^2 \theta + \sigma_\theta \sin^2 \theta - \tau_{r\theta} \sin 2\theta$, где

$$\sin(2\theta) = \frac{2xy}{x^2 + y^2}, \quad \cos^2(\theta) = \frac{x^2}{x^2 + y^2}, \quad \sin^2(\theta) = \frac{y^2}{x^2 + y^2}, \quad r^2 = x^2 + y^2.$$

Следующий код выполняет все необходимые преобразования и использует функцию $\sigma_\theta = \text{StC}(x, y)$, определенную нами выше.

$P = 1; a = 1;$

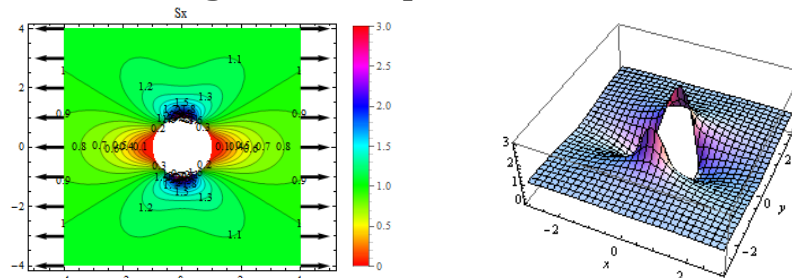
$$\text{SrC}[x_ , y_] = \frac{P}{2} \left(1 - \frac{a^2}{x^2 + y^2} \right) + \frac{P}{2} \left(1 + 3 \frac{a^4}{(x^2 + y^2)^2} - 4 \frac{a^2}{x^2 + y^2} \right) \frac{x^2 - y^2}{x^2 + y^2};$$

$$\text{TrtC}[x_ , y_] = -\frac{P}{2} \left(1 - 3 \frac{a^4}{(x^2 + y^2)^2} + 2 \frac{a^2}{x^2 + y^2} \right) \frac{2xy}{x^2 + y^2};$$

$$\text{SxC}[x_ , y_] = \text{SrC}[x, y] \frac{x^2}{x^2 + y^2} + \text{StC}[x, y] \frac{y^2}{x^2 + y^2} - \text{TrtC}[x, y] \frac{2xy}{x^2 + y^2};$$

Для построения контурного графика и шкалы цветов надо определить минимальное и максимальное значение напряжения σ_x . Они равны 0 и $3P$ соответственно. Тогда

```
pc = ContourPlot[SxC[x, y], {x, -4, 4}, {y, -4, 4},
  RegionFunction -> Function[{x, y, f}, x^2 + y^2 > 1],
  Contours -> Table[pp, {pp, 0, 3, 0.1}], ColorFunction -> Hue,
  ClippingStyle -> Automatic, Axes -> None, PlotLabel -> Sx,
  PlotLegends -> BarLegend[{Hue, {0, 3}}],
  PlotRange -> {0, 3}, ContourLabels -> All]
arR = Graphics[
  Table[{Thickness[0.01], Arrow[{{4, i}, {5, i}}]}, {i, -4, 4, 1}];
arL = Graphics[
  Table[{Thickness[0.01], Arrow[{{-4, i}, {-5, i}}]}, {i, -4, 4, 1}];
Show[pc, arR, arL, PlotRange -> All, AspectRatio -> Automatic]
```



Естественно, что любую функцию двух переменных можно изобразить как поверхность. Продольные напряжения, показанные на предыдущем рисунке справа, строятся как график функции двух переменных следующим кодом

```
Plot3D[SxC[x, y], {x, -3, 3}, {y, -3, 3},
  RegionFunction -> Function[{x, y, f}, x^2 + y^2 > a^2],
  PlotPoints -> 30, Mesh -> Full, PlotRange -> All]
```

Из этих графиков видно, что максимальное значения продольное напряжение σ_x достигается на концах диаметра круга, перпендикулярного к направлению растяжения, и оно в 3 раза больше постоянного напряжения P , приложенного на концах пластинки.

2.4.7 Задачи электростатики

Пример 7.1. Электростатический потенциал точечного заряда в трехмерном пространстве вычисляется по формуле [2]

$$\varphi(\mathbf{r}, \mathbf{p}_0) = \varphi(x, y, z, x_0, y_0, z_0) = \frac{q}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}}, \quad (1)$$

где $\mathbf{p}_0 = (x_0, y_0, z_0)$ – точка расположения заряда. Потенциал поля n зарядов вследствие суперпозиции силовых полей будет выражаться формулой

$$\varphi = \sum_{i=1}^n \varphi(\mathbf{r}, \mathbf{p}_i). \quad (2)$$

Изобразим потенциал поля двух единичных зарядов противоположного знака, расположенных в точках $(-1, 0, 0)$ и $(1, 0, 0)$, с помощью контурного графика. Для этого создадим функцию, реализующую формулу (2)

$$\varphi[\mathbf{q}, \mathbf{p}, \mathbf{r}] := \text{Sum} \left[\frac{q[[i]]}{\text{Norm}[\mathbf{r} - \mathbf{p}[[i]]]}, \{i, \text{Length}[\mathbf{q}]\} \right];$$

Здесь \mathbf{q} представляет список значений зарядов, \mathbf{p} является списком координат точек расположения этих зарядов, \mathbf{r} списком имен пространственных переменных. Функция `Norm` вычисляет норму вектора/списка

Norm $\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}$

$$\sqrt{\text{Abs}[x]^2 + \text{Abs}[y]^2 + \text{Abs}[z]^2}$$

Или

Norm $\{\mathbf{x}_1, \mathbf{y}_1, \mathbf{z}_1\} - \{\mathbf{x}_2, \mathbf{y}_2, \mathbf{z}_2\}$

$$\sqrt{\text{Abs}[x_1 - x_2]^2 + \text{Abs}[y_1 - y_2]^2 + \text{Abs}[z_1 - z_2]^2}$$

Вот, что возвращает функция $\varphi[\mathbf{q}, \mathbf{p}, \mathbf{r}]$ для двух единичных зарядов противоположного знака, расположенных в точках $(-1, 0, 0)$ и $(1, 0, 0)$.

$$\varphi[\underbrace{\{-1, 1\}}_1, \underbrace{\{\{-1, 0, 0\}, \{1, 0, 0\}\}}_1, \{x, y, z\}] // \text{TraditionalForm}$$

$$\frac{\sqrt{|x-1|^2 + |y|^2 + |z|^2}}{\sqrt{|x+1|^2 + |y|^2 + |z|^2}}$$

Команда построения контурного графика функции φ в трехмерном пространстве может иметь вид (следующий график слева)

ContourPlot3D[

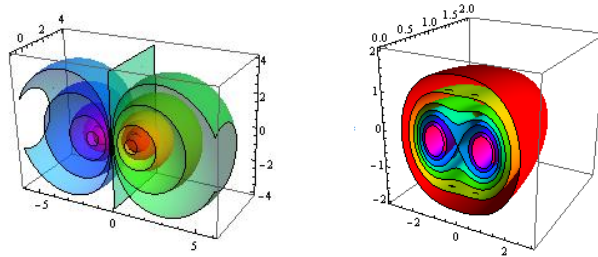
Evaluate $\left[\varphi[\{1, -1\}, \{\{-1, 0, 0\}, \{1, 0, 0\}\}, \{x, y, z\}] \right],$

$\{x, -6, 6\}, \{y, -1, 4\}, \{z, -4, 4\},$

Contours $\rightarrow \{-1, -0.5, -0.25, -0.1, -0.05, 0, 0.05, 0.1, 0.25, 0.5, 1\},$

ContourStyle $\rightarrow \text{Table} \left[\left\{ \text{Opacity}[0.5], \text{Hue} \left[\frac{i}{11} \right] \right\}, \{i, 0, 10\} \right],$

Mesh $\rightarrow \text{None}, \text{BoxRatios} \rightarrow \{12, 5, 8\}$ (* следующий рисунок слева *)



Для двух положительных единичных зарядов контурный график, показанный на предыдущем рисунке справа, построен следующим кодом

```
ContourPlot3D[
  Evaluate[ $\varphi[\{1, 1\}, \{-1, 0, 0\}, \{1, 0, 0\}, \{x, y, z\}]$ ],
  {x, -3, 3}, {y, 0, 2}, {z, -2, 2},
  Contours  $\rightarrow \{1, 1.25, 1.5, 1.75, 2, 2.5, 3\}$ ,
  ContourStyle  $\rightarrow$  Table[Hue[i/7], {i, 0, 6}], Mesh  $\rightarrow$  None]
```

Используя функции $\varphi[\mathbf{q}, \mathbf{p}, \mathbf{r}]$ и ContourPlot3D, можно построить потенциал электростатического поля любого конечного множества точечных зарядов.

Пример 7.2. Рассмотрим распределение зарядов в пространстве, обладающее цилиндрической симметрией, т.е. рассмотрим электростатическое поле заряженной прямой. Электростатический потенциал прямой в трехмерном пространстве, совпадающей с осью Z, задается формулой

$$\varphi = 2q \ln \frac{1}{\rho}, \quad (3)$$

где $\rho = \sqrt{x^2 + y^2}$ и q – линейная плотность заряда вдоль линии. Решение (3) называется логарифмическим потенциалом и обладает круговой симметрией вокруг полюса в точке $\rho = 0$, в которой он обращается в бесконечность. Аналогично, потенциал однородной прямой, перпендикулярной плоскости $z=0$ и проходящей через точку (x_0, y_0) , дает плоское поле и выражается формулой.

$$\varphi(\mathbf{r}, \mathbf{p}_0) = \varphi(x, y, x_0, y_0) = 2q \ln \frac{1}{\sqrt{(x - x_0)^2 + (y - y_0)^2}} \quad (4)$$

Потенциал поля n зарядов вследствие суперпозиции силовых полей будет выражаться формулой (2).

Изобразим потенциал поля двух противоположно заряженных прямых, расположенных в точках $(-1, 0)$ и $(1, 0)$, с помощью контурного графика на плоскости XY. Для этого создадим функцию, реализующую формулу (4)

```
 $\varphi[\mathbf{q}, \mathbf{p}, \mathbf{r}] := \text{Sum}[2q[[i]] \text{Log}[\frac{1}{\text{Norm}[\mathbf{r} - \mathbf{p}[[i]]]}], \{i, \text{Length}[\mathbf{q}]\}];$ 
```

Здесь \mathbf{q} представляет список значений линейных плотностей зарядов, \mathbf{p} является списком координат $\{x_i, y_i\}$ точек пересечения прямых с плоскостью $z=0$, \mathbf{r} списком имен переменных (координат в плоскости), например, $\{x, y\}$.

Вот, что дает функция $\varphi[\mathbf{q}, \mathbf{p}, \mathbf{r}]$ для двух логарифмических потенциалов с плотностями q_1, q_2 , расположенных в точках (x_1, y_1) и (x_2, y_2) .

$\varphi[\{q_1, q_2\}, \{\{x_1, y_1\}, \{x_2, y_2\}\}, \{x, y\}] // \text{TraditionalForm}$

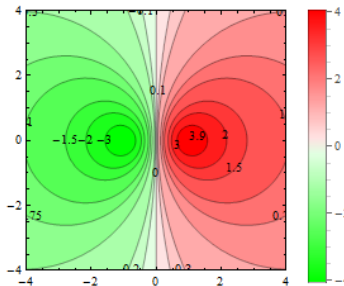
$$2q_1 \log\left(\frac{1}{\sqrt{|x - x_1|^2 + |y - y_1|^2}}\right) + 2q_2 \log\left(\frac{1}{\sqrt{|x - x_2|^2 + |y - y_2|^2}}\right)$$

Создадим список цветов для контурных кривых. Для положительных значений потенциала будем использовать красные оттенки, для отрицательных – зеленые.

```
c = Join[Table[Lighter[Green, i/9], {i, 0, 8}],
         Table[Lighter[Red, i/9], {i, 8, 0, -1}]];
```

Функция `Lighter[Color, f]` создает оттенок цвета `Color`, определяемого числом f ($0 \leq f \leq 1$). Теперь строим контурный график.

```
ContourPlot[Evaluate[\varphi[{-1, 1}, {{-1, 0}, {1, 0}}, {x, y}]],
             {x, -4, 4}, {y, -4, 4}, PlotRangePadding -> None,
             Contours -> {-3, -2, -1.5, -1, -0.75, -0.5, -0.2, -0.1, 0,
                          0.1, 0.2, 0.5, 0.75, 1., 1.5, 2, 3},
             ClippingStyle -> Automatic, ContourShading -> c,
             PlotLegends -> Automatic, ContourLabels -> All]
```



Ниже мы хотим смоделировать в манипуляторе поле нескольких точечных «зарядов». Но вначале мы выполним построения по шагам.

Предварительно скорректируем функцию потенциала φ так, чтобы избавиться от модулей.

```
\varphi[q_, p_, r_] := Simplify[Sum[2q[[i]]Log[
    1
    Norm[r - p[[i]]]], {i, Length[q]}],
    Join[{Element[r, Reals]}, Map[Element[#, Reals] &, p]]];
```

Тогда, например,

$\varphi[\{q_1, q_2\}, \{\{x_1, y_1\}, \{x_2, y_2\}\}, \{x, y\}] // \text{TraditionalForm}$

$$q_1(-\log((x - x_1)^2 + (y - y_1)^2)) - q_2 \log((x - x_2)^2 + (y - y_2)^2)$$

или

$\varphi[\{q_1\}, \{\{x_1, y_1\}\}, \{x, y\}] // \text{TraditionalForm}$

$$q_1(-\log((x - x_1)^2 + (y - y_1)^2))$$

Для вычисления градиента можно использовать функцию дифференцирования `D` в виде

```
Clear[f, x, y]; D[f[x, y], {{x, y}}
```

$$\{f^{(1,0)}[x, y], f^{(0,1)}[x, y]\}$$

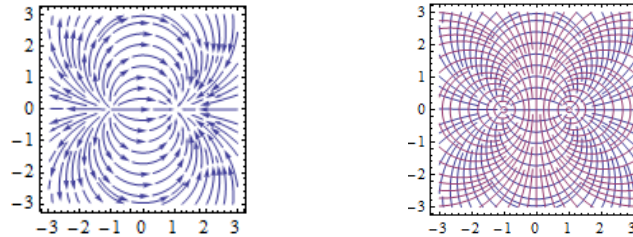
Вторым аргументом такого использования `D` должен быть список списков имен координат. Тогда, например

$$D[\varphi[\{1\}, \{\{0, 0\}\}, \{x, y\}], \{\{x, y\}\}]$$

$$\left\{ -\frac{2x}{x^2 + y^2}, -\frac{2y}{x^2 + y^2} \right\}$$

В *Mathematica* есть функция `StreamPlot`, которая умеет строить линии тока векторного поля (одного или нескольких). Например силовые линии поля двух зарядов -1 и $+1$, расположенных в точках $(-1, 0)$ и $(+1, 0)$, можно построить следующим образом (следующий рисунок слева)

**StreamPlot[Evaluate[D[$\varphi[\{-1, 1\}, \{\{-1, 0\}, \{1, 0\}\}, \{x, y\}], \{\{x, y\}\}]]$,
 $\{x, -3, 3\}, \{y, -3, 3\}, \text{StreamScale} \rightarrow 0.2]$**



Кроме силовых линий мы будем строить линии уровня потенциала. Их можно строить по – разному, например, как контурный график. Мы поступим по – другому. Создадим векторное поле, ортогональное полю градиента, и построим его линии тока. Очевидно, они будут линиями уровня потенциала. Этот способ мы выбираем из – за того, что линии тока обеих векторных полей можно построить одной функцией `StreamLine`. Например для предыдущего поля, создаваемого двумя зарядами, следующий код строит силовые линии поля и линии уровня потенциала (предыдущий рисунок справа)

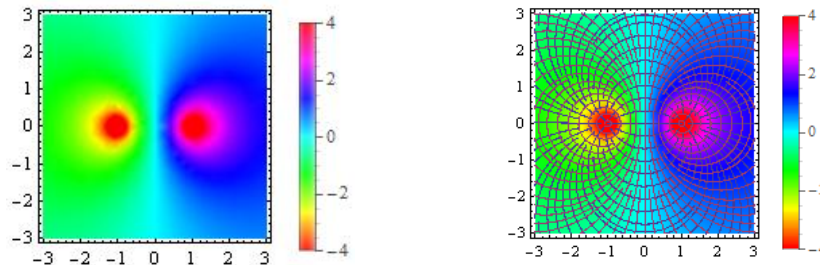
**gd = Simplify[D[$\varphi[\{-1, 1\}, \{\{-1, 0\}, \{1, 0\}\}, \{x, y\}], \{\{x, y\}\}]]$
gdu = Reverse[gd * {1, -1}];
ps = StreamPlot[Evaluate[{gd, gdu}], {x, -3, 3}, {y, -3, 3},
StreamScale → None]**

$$\left\{ \frac{4 - 4x^2 + 4y^2}{(1 - 2x + x^2 + y^2)(1 + 2x + x^2 + y^2)}, -\frac{8xy}{(1 - 2x + x^2 + y^2)(1 + 2x + x^2 + y^2)} \right\}$$

Здесь опция `StreamScale->None` отменяет рисование стрелок на линиях поля. Векторное поле градиента приведено сразу после кода, а векторное поле **gdu** ортогонально к полю градиента и получается перестановкой его координатных функций, перед одной из которых изменен знак.

Теперь строим график плотности потенциала (следующий рисунок слева)

pd=DensityPlot[
Evaluate[$\varphi[\{-1, 1\}, \{\{-1, 0\}, \{1, 0\}\}, \{x, y\}]]$, {x, -3, 3}, {y, -3, 3},
ColorFunction → Hue, PlotLegends → Automatic,
PlotRange → {-4, 4}, ClippingStyle → Automatic]



Оба графика совмещаем с помощью функции `Show`. При этом первым должен рисоваться график плотности **pd**, иначе он закроет график линий тока **ps**.

Show[*pd*, *ps*] (* предыдущий рисунок справа *)

Наконец создадим манипулятор, в котором построим поле трех точечных «зарядов» в виде графика плотности с силовыми линиями и линиями уровня потенциала. Вот результирующий код.

DynamicModule[{*φ*, *z*, *gd*, *gdu*, *m* = 3, *ps*, *pd*},

$\varphi[q_p_r_] := \text{Simplify}[\text{Sum}[2q[[i]]\text{Log}[\frac{1}{\text{Norm}[r - p[[i]]]}], \{i, \text{Length}[q]\}],$

$\text{Join}[\{\text{Element}[r, \text{Reals}]\}, \text{Map}[\text{Element}[\#, \text{Reals}] \&, p]]];$

z[*x*_, *y*_, *q*_, *p*_] := *φ*[*q*, *p*, {*x*, *y*}];

Manipulate[

gd = **Simplify**[*D*[*z*[*x*, *y*, {*q*1, *q*2, *q*3}, *p*], {{*x*, *y*}}];

gdu = **Reverse**[*gd* * {1, -1}];

ps = **StreamPlot**[**Evaluate**[{*gd*, *gdu*}], {*x*, -*m*, *m*}, {*y*, -*m*, *m*},
StreamScale → **None**];

pd = **DensityPlot**[**Evaluate**[*z*[*x*, *y*, {*q*1, *q*2, *q*3}, *p*]],
{*x*, -*m*, *m*}, {*y*, -*m*, *m*}, **ColorFunction** → **Hue**,
PlotLegends → **Automatic**, **PlotRange** → {-6, 6},
ClippingStyle → **Automatic**];

Show[*pd*, *ps*, **PlotRange** → {{-*m*, *m*}, {-*m*, *m*}},
PlotRangePadding → **None**],

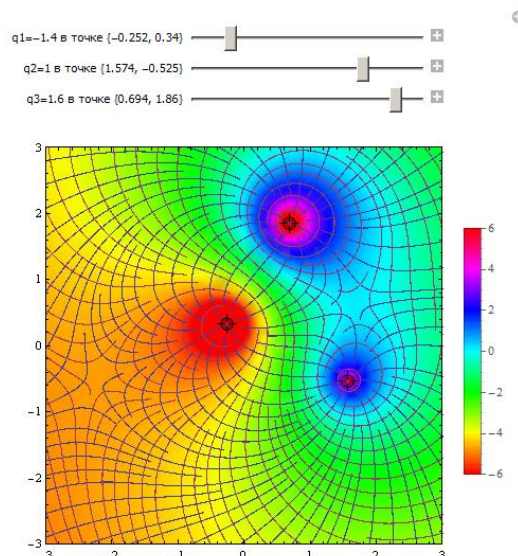
{{*q*1, -1, **Dynamic**["*q*1 = " <> **ToString**[*q*1] <> "в точке" <>
ToString[*p*[[1]]]]}, -2, 2, 0.1},

{{*q*2, 1, **Dynamic**["*q*2 = " <> **ToString**[*q*2] <> "в точке" <>
ToString[*p*[[2]]]]}, -2, 2, 0.1},

{{*q*3, 1, **Dynamic**["*q*3 = " <> **ToString**[*q*3] <> "в точке" <>
ToString[*p*[[3]]]]}, -2, 2, 0.1},

{{*p*, {{-1, 0}, {1, 0}, {0, 1}}}, {-2, -2}, {2, 2}, **Locator**},

Paneled → **False**, **ContinuousAction** → **False**]]



Основные пояснения для этого кода были выполнены выше. Код внутри функции `Manipulate` после функции `Show` выводит текстовую информацию слева от ползунков – значение заряда и его координаты, и создает три «локатора», координаты которых определяют положение зарядов. Опция **`ContinuousAction` → `False`** позволяет перерисовывать область внутри манипулятора только, когда вы отпускаете ползунок. Иначе перерисовка выполняется непрерывно при движении ползунков.

Вы можете менять значение всех трех «зарядов», перемещая ползунки. Перемещая «локаторы», вы меняете расположение «зарядов». Одно из возможных состояний манипулятора показано на предыдущем рисунке.

Литература.

1. Крауфорд Ф. Волны. М., 1974 г., 528 стр.
2. Тихонов А.Н., Самарский А.А. Уравнения математической физики. – М.: Наука, 1977. – 736с.
3. Доля П.Г. Периодическое продолжение функций и решение уравнения колебаний струны в системах символьной математики.// Вестник Харьк. нац. ун-та., - 2006.- № 733. Сер. "Математическое моделирование.
4. Dolya P.G. Solution to the homogeneous boundary value problems of free vibrations of a finite string // Journal of Mathematical Physics, Analysis, Geometry, - 2008, vol.4, No 2 , pp. 237 – 251.
5. Тимошенко С.П. Колебания в инженерном деле. – М.: Гос. изд. физ.-мат. литературы, 1959.
6. Тимошенко С.П. Сопротивление материалов. Т.1 – М.: Наука, 1965.
7. Тимошенко С.П., Гудьер Дж. Теория упругости. – М.: Наука, 1979.
8. Филоненко Бородин М.М. Теория упругости. – М.: Гос. изд. физ.-мат. лит., 1959.
9. Снеддон И.Н., Берри Д.С. Классическая теория упругости. – М.: Гос. изд. физ.-мат. лит., 1961.