ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 12

Тема: Решение дифференциальных уравнений в частных производных в MathCad

Цель: Освоить способы решения дифференциальных уравнений (или систем уравнений) в частных производных в MathCad с помощью функций: pdesolve, numol, multigrid и relax.

Дифференциальных уравнений в частных производных требуют нахождения функции не одной, а нескольких переменных. **MathCad** имеет очень ограниченные возможности для решения таких уравнений, ведь для решения каждого вида уравнений в частных производных требуется свой метод решения.

Уравнения в частных производных можно разделить на три типа:

- параболические, содержащие первую производную по одной переменной и вторую – по другой, причем все производные входят в уравнение с одинаковым знаком;
- гиперболические, содержащие первую производную по одной переменной и вторую – по другой, входящие в уравнение с разными знаками;
- эллиптические, содержащие только вторые производные, причем одного знака.

1. Функции решения параболических и гиперболических уравнений

MathCad включает в себя две функции для решения параболических и гиперболических уравнений: pdesolve и numol. Функция pdesolve используется в составе вычислительного блока Given-pdesolve для решения гиперболических или параболических уравнений (или систем уравнений) в частных производных, имеющих в качестве аргументов, как правило, время t и пространственную координату x

Обращение к этой функции:

$$pdesolve(u, x, xrange, t, trange, [xpts], [tpts])$$

возвращает скалярную (для одного уравнения) или векторную (для системы уравнений) функцию, являющуюся решением уравнения (или системы уравнений). Здесь u - явно заданный вектор имен функций (без указания имен аргументов), подлежащих вычислению. Эти функции, а также граничные условия должны быть определены внутри вычислительного блока Given-pdesolve; x - пространственная координата; $xrange$ - вектор значений аргумента x для граничных условий. Он должен состоять из двух чисел,

представляющих две границы расчетного интервала; *t* - время (имя второго аргумента неизвестной функции); *trange* - вектор значений аргумента *t* для граничных условий, состоящий из двух чисел, представляющих две границы расчетного интервала; *xpts* - количество пространственных точек дискретизации (может не указываться); *tpts* - количество временных слоев (также может не указываться).

Пример использования функции pdesolve приведен на рис. 1. Запись вычислительного блока с функцией pdesolve аналогична записи блока с функцией Odesolve. Результаты расчета показаны на рис. 2.

Решение одномерного волнового уравнения
с помощью функции *pdesolve*

Для идентификации частных производных следует использовать нижний индекс, например, $u_{xx}(x,t)$ - вторая производная функции u по пространственной координатах.

Решение одномерного волнового уравнения

$$\frac{d^2}{dt^2}w(x,t) = a^2\cdot\frac{d^2}{dx^2}w(x,t)$$

Здесь w - перемещение, v - скорость перемещения

где $\frac{\partial}{\partial t}w(x,t) = v(x,t)$ тогда $\frac{\partial}{\partial t}v(x,t) = a^2\cdot\frac{d^2}{dx^2}w(x,t)$

Представим первое уравнение как
систему двух уравнений первого порядка

Given $v_t(x,t) = a^2\cdot w_{xx}(x,t)$ $w_t(x,t) = v(x,t)$

граничные условия

$w(x,0) = \sin\left(\frac{\pi\cdot x}{L}\right)$ $v(x,0) = 0$

$w(0,t) = 0$ $w(L,t) = 0$

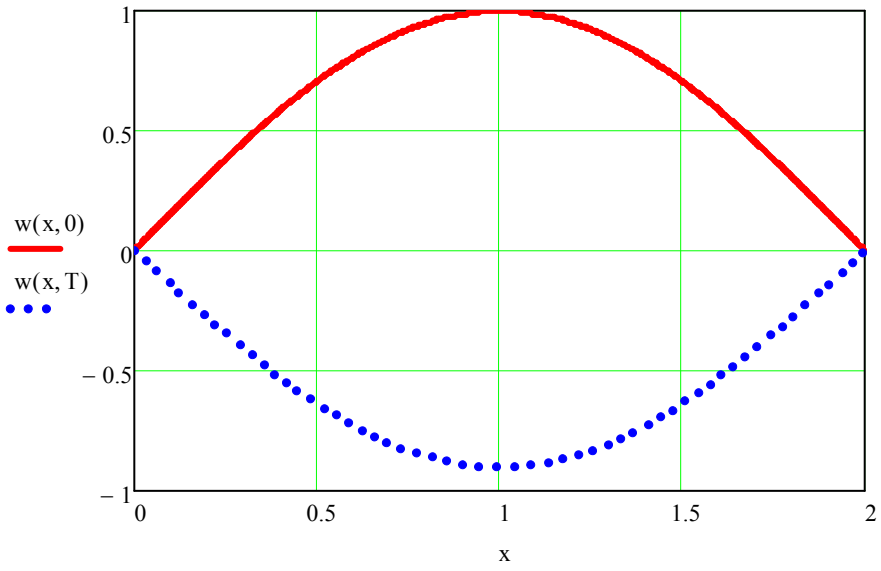
$\begin{pmatrix} w \\ v \end{pmatrix} := \text{Pdesolve}\left[\begin{pmatrix} w \\ v \end{pmatrix}, x, \begin{pmatrix} 0 \\ L \end{pmatrix}, t, \begin{pmatrix} 0 \\ T \end{pmatrix}\right]$

$a \equiv 1$
$L \equiv 2$
$T \equiv 2\cdot\pi$

Глобальное присвоение параметров

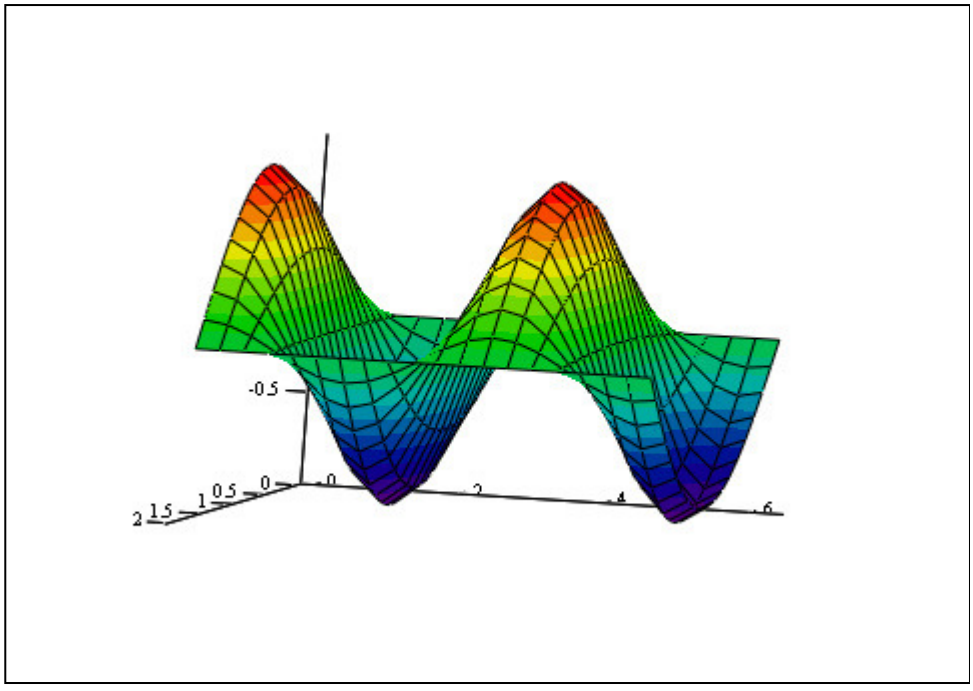
Рис. 1. Пример использования функции pdesolve

Единичное решение волнового уравнения



Сетка решений волнового уравнения на временном и пространственном интервалах

M := CreateMesh (w , 0, L, 0, T)



M

Рис. 2. Результаты решения волнового уравнения.

Обратите внимание на то, что уравнения внутри вычислительного блока должны записываться с аргументами. Для идентификации частных производных следует использовать нижний индекс, например, $u_{xx}(x,t)$ - вторая производная функции u по пространственной координате x.

Недостатком функции pdesolve (как и функции Odesolve) является невозможность ее использования в составе выражения – программы для многократного решения дифференциального уравнения. При необходимости многократного решения обыкновенных дифференциальных уравнений в состав программного модуля можно включать функции Rkadapt и Bulstoer.

При необходимости многократного решения дифференциальных уравнений в частных производных в состав программного модуля можно включать функцию numol.

Обращение к этой функции:

$numol(xrange, xpts, trange, tpts, Npde, Nae, rhs, init, bc)$

возвращает матрицу решения дифференциального уравнения в частных производных в каждой точке по пространственной (по строкам) и временной (по столбцам) координате. Если решается не одно уравнение, а система уравнений, то результатом решения является составная матрица, образованная путем слияния (слева направо) со значениями каждой искомой сеточной функции. Здесь *xrange* - вектор значений аргумента x для граничных условий. Он должен состоять из двух чисел, представляющих две границы расчетного интервала; *trange* - вектор значений аргумента t для граничных условий, состоящий из двух чисел, представляющих две границы расчетного интервала; *xpts* - количество пространственных точек дискретизации (может не указываться); *tpts* - количество временных слоев (также может не указываться); *Npde* - количество дифференциальных уравнений в системе; *Nae* - количество дополнительных алгебраических уравнений, входящих в систему; *rhs* - вектор правых частей уравнений; *init* - векторная функция, определяющая начальные условия для каждой неизвестной функции; *bc* - функциональная матрица граничных условий.

Вектор граничных условий может иметь значения трех типов:

- *rhs* содержит вторые пространственные производные: граничные условия (или Дирихле «D», или Неймана «N») требуются по одному с каждой стороны интервала интегрирования;
- *rhs* содержит первые пространственные производные: граничные условия Дирихле на левой или правой границе интервала, на другой стороне NA;
- нет пространственных производных – граничные условия не требуются.

Функциональная матрица *bc* содержит три столбца, имеющих следующий вид:

- $(init\_left(t)\ init\_right(t)$ «D»$)$ - для граничных условий Дирихле;
- $(init\_left(t)\ init\_right(t)$ «N»$)$ - для граничных условий Неймана.

Пользоваться функцией numol намного сложнее, чем функцией pdesolve (рис. 3 и 4).

Решение одномерного волнового уравнения
с помощью функции *numol*

Для идентификации частных производных следует использовать нижний индекс, например, $u_{xx}(x,t)$ - вторая производная функции u по пространственной координатах.

исходное уравнение то же,
что и при решении функцией **pdesolve**

$$\frac{d^2}{dt^2}w(x,t) = a^2\cdot\frac{d^2}{dx^2}w(x,t)$$

$a \equiv 1$

$L \equiv 2$

$T \equiv 2\cdot\pi$

заменено системой
уравнений

$$\frac{\partial}{\partial t}w(x,t) = v(x,t) \qquad \frac{\partial}{\partial t}v(x,t) = a^2\cdot\frac{d^2}{dx^2}w(x,t)$$

$Npde := 2$ Дифференциальных уравнений в частных производных 2

$Nae := 0$ алгебраических уравнений нет

вектор правых частей уравнений

$$rhs(x,t,u,u_x,u_{xx}) := \begin{pmatrix} u_1 \\ a^2\cdot u_{xx_0} \end{pmatrix}$$

здесь, $u_1 = v$ и $u_0 = w$.

Вектор начальных условий

$$init(x) := \begin{pmatrix} \sin\left(\frac{\pi\cdot x}{L}\right) \\ 0 \end{pmatrix}$$

Граничные условия
в начале и в конце
выбранных интервалов x и t

$$bc\_func(t) := \begin{pmatrix} init(0)_0 & init(L)_0 & "D" \\ "NA" & "NA" & "D" \end{pmatrix}$$

Решение уравнения:

$nx := 30$ $nt := 20$

В этом примере 20 точек по времени на каждую функцию.
Итого получаем матрицу из 40 колонн.

$$sol := numol\left[\begin{pmatrix} 0 \\ L \end{pmatrix}, nx, \begin{pmatrix} 0 \\ T \end{pmatrix}, nt, Npde, Nae, rhs, init, bc\_func\right]$$

$rows(sol) = 30$ $cols(sol) = 40$

Матрица sol содержит два решения: первое для u_0=w и второе для u_1=v

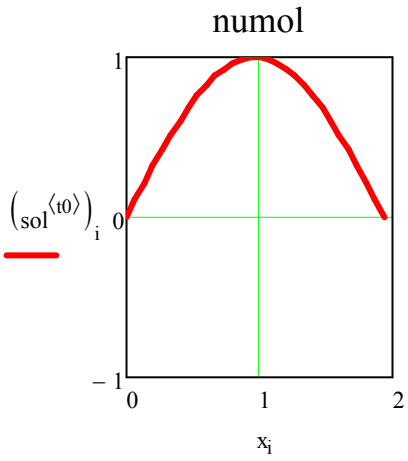
(посмотрите их. Для этого на трехмерном графике замените SOL на sol)

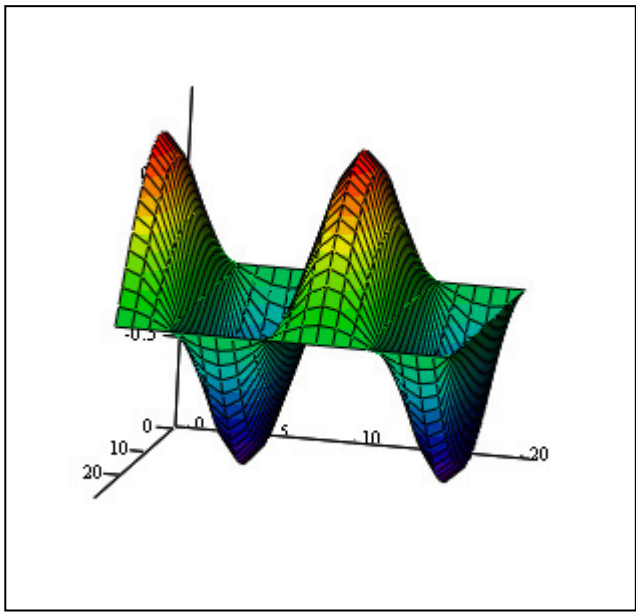
Исключим второе решение u_1 $SOL := submatrix(sol, 0, nx-1, 0, nt)$

$i := 0..30$ $L = 2$ $x_i := \frac{i\cdot L}{30}$ $t0 := 0$

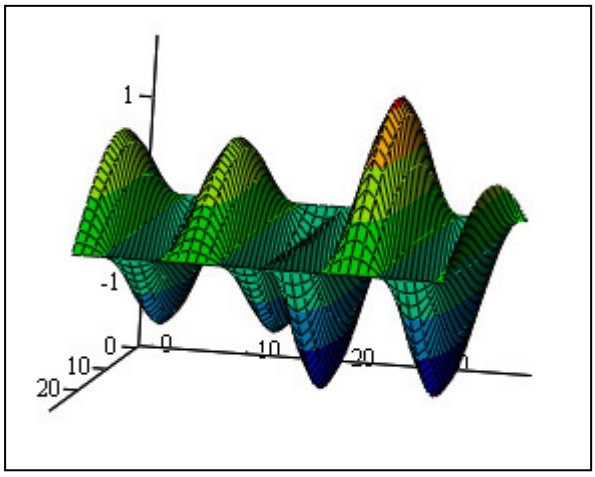
Рис. 3. Решение волнового уравнения с помощью функции numol.

Сетка решений в пространстве и времени





SOL



sol

Рис. 4. Результаты решения волнового уравнения с помощью функции numol.

Процесс изменения решения во времени удобно наблюдать с использованием анимации результатов расчета (рис. 5).

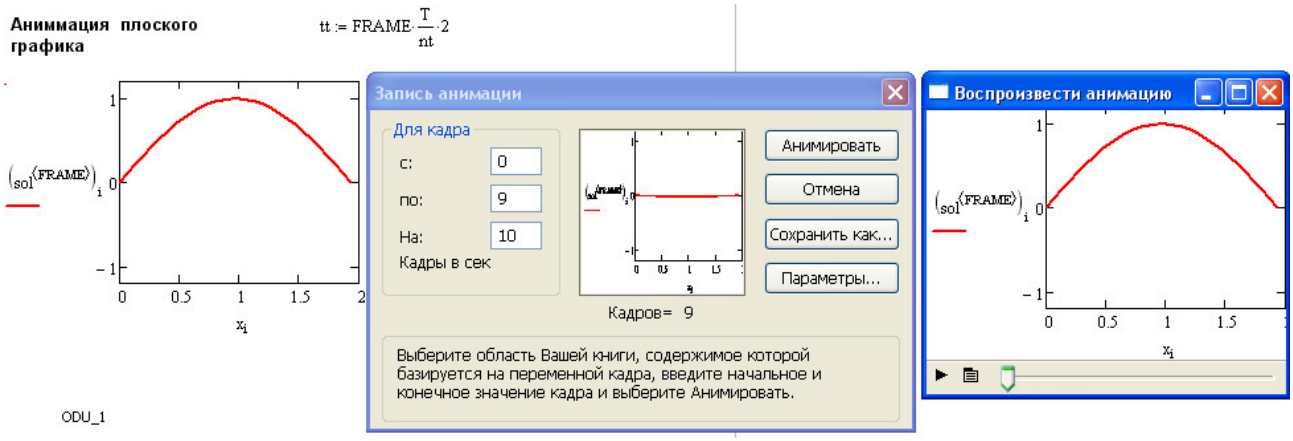


Рис. 5. Анимация решения волнового уравнения.

2. Решение эллиптических уравнений (Лапласа и Пуассона).

С помощью двух встроенных функций, multigrid и relax, можно решить простейшие случаи уравнения Пуассона. Это уравнение очень часто используется в технике, например, для описания полей напряжений и деформаций в плоской задаче теории упругости, в задачах теплопроводности, гидродинамики, аэродинамики, электростатики.

В MathCad для численного решения уравнения Пуассона используется метод конечных разностей.

Уравнение Пуассона имеет вид

$$\frac{\partial^2 F(x,y)}{\partial x^2} + \frac{\partial^2 F(x,y)}{\partial y^2} = -\rho(x,y).$$

Если правая часть уравнения равна нулю, такое уравнение называется уравнением Лапласа

$$\frac{\partial^2 F(x,y)}{\partial x^2} + \frac{\partial^2 F(x,y)}{\partial y^2} = 0.$$

На квадратной области уравнение Пуассона представляется в виде:

$$au_{i+1,j} + bu_{i-1,j} + cu_{i,j-1} + du_{i,j+1} + eu_{i,j} = p_{i,j}.$$

Численное решение ищется в MathCad только на квадратной области, состоящей из $(n+1)\times(n+1)$ точек. Поэтому граничные условия должны быть определены пользователем для всех четырех сторон квадрата. Самый простой (и наиболее используемый) вариант – это нулевые граничные условия (уравнение Лапласа). В таком случае можно использовать функцию multigrid.

Обращение к функции:

$$multigrid\,(M\,,Ncycle),$$

где M - квадратная матрица размером $1\times 2n$, которая содержит значения правой части уравнения Пувссона в соответствующей точке квадратной области;

$Ncycle$ - число циклов на каждом уровне итерации функции multigrid. Значение $Ncycle = 2$ обычно дает хорошую аппроксимацию решения.

Пример использования функции multigrid приведен на рис. 6.

Решение уравнения Пуассона с помощью функции *multigrid*

$M := 0$ обнуление предыдущих значений M

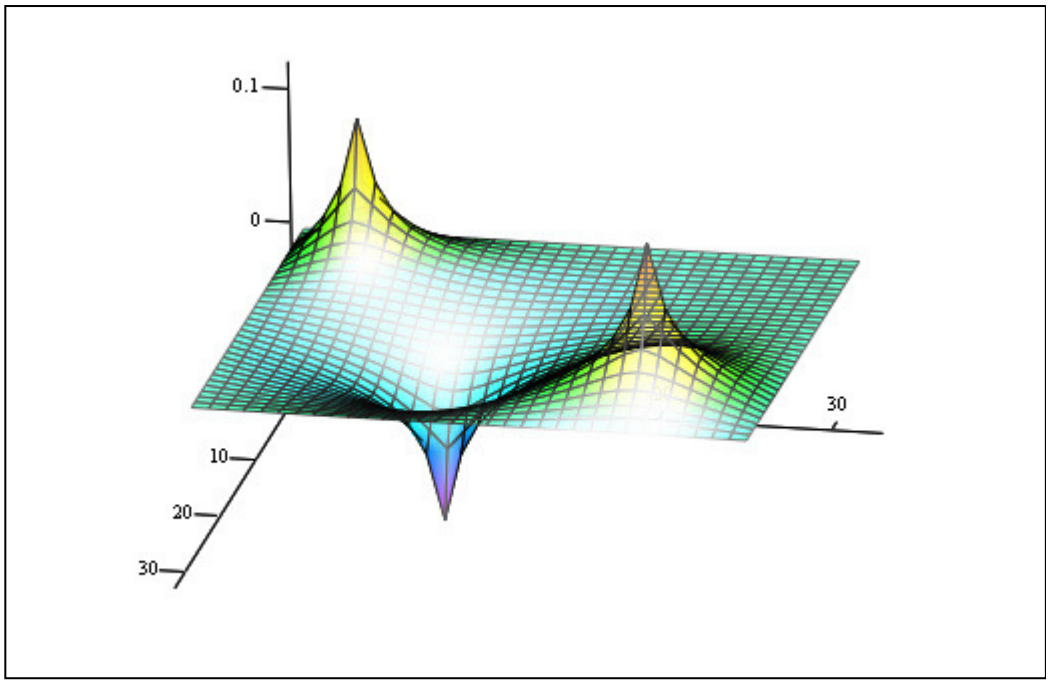
$n := 32$ $M_{n,n} := 0$ Обнуление матрицы правых частей уравнения

$i := 1..n$ $j := 1..n$ $M_{i,j} := -2$ Включите выражения в рамке

$M_{4,4} := -200$ $M_{20,12} := 200$ $M_{25,25} := -200$ Три точечных источника

Значения правой части уравнения Пуассона постоянны во всех точках

$F := multigrid(M, 2)$



F

Рис. 6. Решения уравнения Лапласа с помощью функции multigrid.

Если граничные условия со стороны квадрата ненулевые, необходимо использовать функцию relax.

Обращение к функции:

$$relax(a,b,c,d,e,f,u,rjac),$$

где a,b,c,d,e - квадратные матрицы одинакового размера, содержащие коэффициенты аппроксимирующего уравнения; f - квадратная матрица, содержащая значения правой части уравнения в каждой точке области, где ищется решение; u - квадратная матрица, содержащая граничные значения решения на границах квадратной области и начальное приближение для решения внутри области; $rjac$ - спектральный радиус итераций Якоби. Это число между 0 и 1, которое управляет сходимостью процесса релаксации.

Использование этой функции требует глубокого анализа метода конечных разностей для составления указанных матриц. Пример использования функции relax приведен на рис 7 ,8 и 9.

Решение уравнения Пуассона с помощью функции *relax*

Если граничные условия по сторонам квадрата ненулевые необходимо использовать функцию *relax*.

Определим размеры сетки

$n := 32$ $i := 0..n$ $j := 0..n$

$a_{i,j} := 1$ $b := a$ $c := a$ $d := a$ $e := -4 \cdot a$

Определим положение и интенсивность источника тепла
Пусть во всех узлах интенсивность постоянная

Граничные условия $S_{i,j} := 0.05$ $S_{i,j} := 0.01$

на верхней границе $f_{0,j} := 0$

на нижней границе $f_{n,j} := 2 \cdot \cos\left(4 \cdot \pi \cdot \frac{j}{n}\right)$

по бокам $f_{i,0} := 2 \cdot \frac{i}{n}$ $f_{i,n} := 2 \cdot \frac{i}{n}$

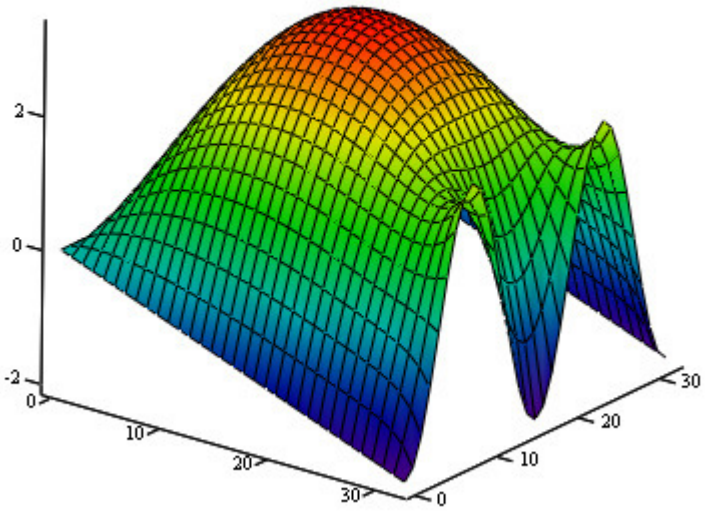
Поменяйте условия. Включите выражения в рамке.

$f_{0,j} := 0$ $f_{n,j} := 0$ $f_{i,0} := 20$ $f_{i,n} := 20$

Спектральный радиус Якоби $r := 1 - \frac{2 \cdot \pi}{n}$ $r = 0.804$

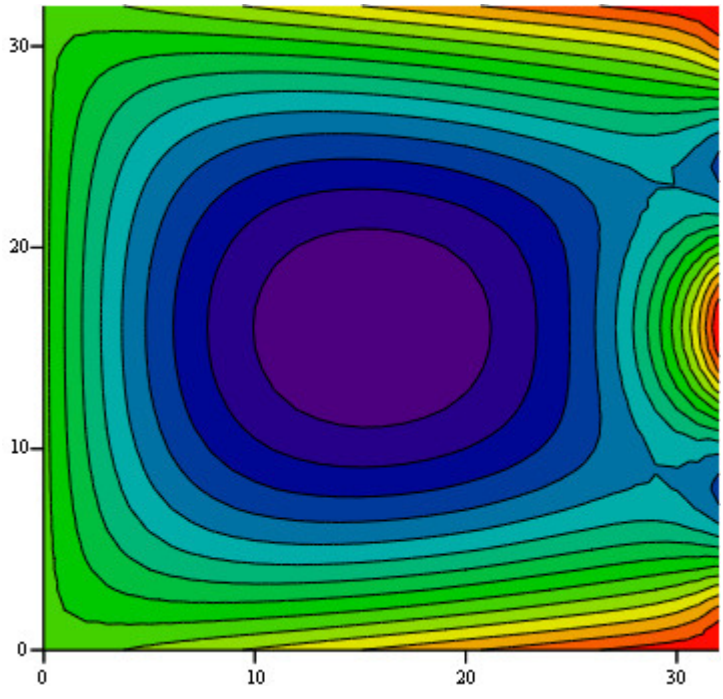
Решение уравнения Пуассона $F := relax(a,b,c,d,e,S,f,r)$

Рис. 7. Решения уравнения Пуассона с помощью функции relax.



–F

Контурный график распределения температуры



F

Рис. 8. Результаты решения уравнения Пуассона с помощью функции relax.

Решение уравнения Пуассона с помощью функции *relax*

Если граничные условия по сторонам квадрата ненулевые необходимо использовать функцию *relax*.

Определим размеры сетки

$n := 32 \qquad i := 0..n \qquad j := 0..n$

$a_{i,j} := 1 \quad b := a \qquad c := a \qquad d := a \qquad e := -4 \cdot a$

Определим положение и интенсивность источника тепла
Пусть во всех узлах интенсивность постоянная

Граничные условия $S_{i,j} := 0.01$

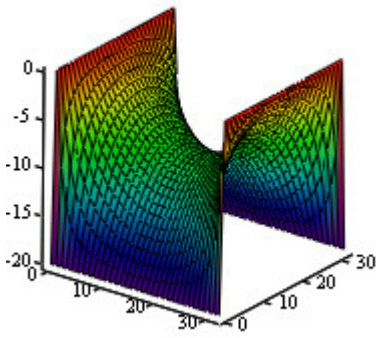
на верхней границе $f_{0,j} := 0$

на нижней границе $f_{n,j} := 0$

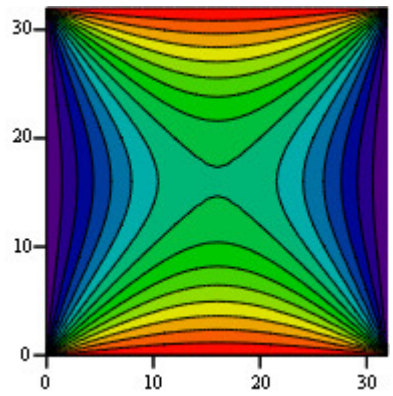
по бокам $f_{i,0} := 20 \qquad f_{i,n} := 20$

Спектральный радиус Якоби $r := 1 - \frac{2 \cdot \pi}{n} \qquad r = 0.804$

Решение уравнения Пуассона $F := relax(a, b, c, d, e, S, f, r)$



Контурный график распределения температуры



–F

F

Рис. 9. Результаты решения уравнения Пуассона с другими граничными условиями с помощью функции relax.