КАЗАНСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

САМОСТОЯТЕЛЬНЫЕ РАБОТЫ ПО СПЕЦИАЛЬНОМУ КУРСУ «Математические модели теории упругости»

МЕТОДИЧЕСКОЕ ПОСОБИЕ

КАЗАНЬ – 2012

УДК 519.6

ББК 32.81

Бахтиева Л.У.

Самостоятельные работы по специальному курсу «Математические модели теории упругости»: методическое пособие / Л. У. Бахтиева – Казань: 2012. – 22 с.

Введение

В последнее время многие технические расчеты выполняются с помощью системы Matlab. Исключением не являются и задачи теории упругости, которые изучаются в рамках специального курса лекций «Математические модели теории упругости». Для закрепления материала лекций на практических занятиях студентам предлагается выполнить 12 самостоятельных работ. Прежде, чем приступить к их выполнению, рекомендуется изучить предлагаемую в конце данного методического пособия литературу, а также материалы лекций по изучаемой теме.

Самостоятельная работа 1
Знакомство с основными возможностями системы Matlab

Тема 1. Работа в режиме прямых вычислений

Примеры: 1.1. Вычислить произведение полиномов $P(x) = x^5+x^3+1$ и $Q(x) = x^2+2x+3$.

```
>> P=[1 0 1 0 0 1];          %вектор коэффициентов полинома P(x)
>> Q=[1 2 3];                %вектор коэффициентов полинома Q(x)
>> pr=conv(P,Q)              %произведение
pr = 1  2   4  2  3  1  2  3
```

1.2. Найти минимум функции $y = x^2-3x+2$ на отрезке [-5, 5].

```
>> y=inline('x.^2-3*x+2');                %функция пользователя
>> x=-5:0.1:5;                            %вектор значений x
>> x1=fminbnd(y,-5,5);          %определение абсциссы x1 точки минимума
>> %форматированный вывод на экран значений x1 и y(x1)
>> fprintf(' ymin = %3.1f ',y(x1)); fprintf(' при x = %2.1f ',x1);
 ymin = -0.3  при  x = 1.5
```

1.3. Найти частичную сумму S_n ряда $\sum(-1)^{k+1}/k^2$. Сравнить ответ с точным значением суммы $S = \pi^2/12$.

```
>> n=input(' n=');                        %интерактивный ввод параметра n
 n=100
>> for k=1:n  a(k)= (-1)^(k+1)*k^(-2);  end      %вычисление слагаемых ряда
>> s=sum(a)                               %суммирование
s = 0.8224
>> pi^2/12
ans = 0.8225
```

1.4. Определить, находится ли точка M(1.8; 1.8) внутри области D, заданной неравенствами: $x^2+y^2 \geq 4$, $|x| \leq 2$, $|y| \leq 2$.

```
>> x=1.8;  y=1.8;
>> z=and(x^2+y^2>=4,and(abs(x)<=2,abs(y)<=2));
>> if  z = =1  disp('да');   else  disp('нет');  end
да
```

Задания: 1. Вычислить гамма-функцию Г(x) для x = `0.1,0.2,...,10.`

2. Вычислить все значения корня 6-й степени из комплексного числа z= – 64, используя формулу: $z^{1/n} = r^{1/n} [\cos((\varphi+2k\pi)/n) + i \sin((\varphi+2k\pi)/n)]$, где $k = 0, ..., n-1$; r, φ - модуль и аргумент числа z. Сравнить полученный результат с результатом выполнения команды x=z^(1/n).

Тема 2. Программирование, работа с файлами

Примеры:

2.1. Вычислить среднее значение и дисперсию для элементов вектора x.

Первый шаг – создание функции stat:

```
function [mx,dx]=stat(x)                  %объявление имени функции
n=length(x);                              %вычисление длины вектора x
mx=sr(x,n);          %вычисление среднего значения с помощью подфункции sr
dx=sum((x-sr(x,n)).^2)/n;                 %вычисление дисперсии
function s=sr(x,n)                        %подфункция в теле основной функции
```

```
s=sum(x)/n;
```

Второй шаг – сохранение функции в виде m-файла под именем stat.m.

Третий шаг - использование функции stat в командной строке:

```
>> x=[-1,2.2,3.1,-4,5];
>> [mx,dx]=stat(x)
mx = 1.060
dx = 10.167
```

2.2. Построить таблицу значений функции y на отрезке [x1; x2] с шагом dx, записать ее в файл tab.txt и вывести таблицу на экран. Использовать программу для вычисления значений функции y = ln x на отрезке [1; 2] с шагом 0.1.

1 шаг - создание файла – функции:

```
function []=table(func,x1,x2,dx)
f=inline(func);         %функция пользователя, строка func задается в командном окне
x=x1:dx:x2;                                   %вектор значений аргумента
y=f(x);                                              %вектор значений функции
fid=fopen('tab.txt','wt');                     %открытие файла tab.txt для записи
if fid ~ = -1
  fprintf(fid,'-------------------------\n')       %линия (можно также начертить линию с...
                                                  помощью функций blanks и strrep)
  fprintf(fid,'| %9s | %9s |\n','x','y');                    %вывод названий столбцов
  fprintf(fid,'-------------------------\n');                                  %линия
  fprintf(fid,'| %1.5f  | %1.5f  |\n',[x y]);                          % вывод чисел
  fprintf(fid,'-------------------------\n');                                %линия
  fclose(fid);                                                    %закрытие файла
  fid= fopen('tab.txt','rt');                    % открытие файла tab.txt для чтения
  while~feof(fid)                                                 % до конца файла
     stroka=fgetl(fid);                              %считывание строк таблицы
     disp(stroka) %вывод  на экран
  end
  fclose(fid);                                                    %закрытие файла
else  fprintf('Ошибка при открытии файла!')
end
```

2 шаг – сохранение файла под именем table.m.

3 шаг - построение таблицы для функции y=ln x:

```
>> x1=1; x2=2; dx=.1;
>> func='log(x)';
>> table(func,x1,x2,dx)
```

```
--------------------------
|    x     |    y     |
--------------------------
| 1.00000  | 0.00000  |
| 1.10000  | 0.09531  |
| 1.20000  | 0.18232  |
| 1.30000  | 0.26236  |
| 1.40000  | 0.33647  |
| 1.50000  | 0.40547  |
| 1.60000  | 0.47000  |
| 1.70000  | 0.53063  |
| 1.80000  | 0.58779  |
| 1.90000  | 0.64185  |
| 2.00000  | 0.69315  |
--------------------------
```

2.3. Подсчитать количество положительных чисел в произвольном числовом массиве, вывести положительные числа на экран.

1 шаг - создание файла – сценария:

```
x=input('Исходный массив чисел:');      %ввод исходного числового массива
y=x(:)>0;                                % логический массив
if any(y)
      s=sum(y);
      string=['В массиве содержится  ', num2str(s), '  положительных чисел'];
      disp(string)
      x(y')
 else
      string=['В массиве нет положительных чисел'];
      disp(string)
 end
```

2 шаг – сохранение файла под именем prim2_3.m.

3 шаг - проверка работы программы:

```
>> prim2_3
Исходный массив чисел: [-1,2.2,3.1,-4,5,8,9.3]
В массиве содержится  5 положительных чисел
ans = 2.2  3.1  5  8  9.3
```

Задания: 1. Написать программу, считывающую из некоторого текстового файла заданное число строк и выводящую эти данные в командное окно.
2. Написать программу - сценарий, преобразующую массив чисел из десятичной системы счисления в двоичную (использовать функцию dec2bin( )).
3. Написать программу - функцию, вычисляющую направляющие косинусы заданного трехмерного вектора.

Самостоятельная работа 2
Вычисление корней полинома и нулей функции

Примеры:

1. Вычислить все корни полинома $P(x) = x^7 + 3.2\,x^5 - 5.2\,x^4 + 0.5\,x^2 + x - 3$.
Для одного из корней сделать проверку.

```
>> p=[1 0 3.2 –5.2 0 0.5 1 -3];                  % коэффициенты полинома
>> r=roots(p)                                     % вычисление корней
r=
    -0.5668 + 2.0698i
    -0.5668 - 2.0698i
    -0.6305 + 0.5534i
    -0.6305 - 0.5534i
     1.2149
     0.5898 + 0.6435i
     0.5898 - 0.6435i
>> polyval(p,1.2149)           % вычисление значения полинома при x=1.2149
  ans =
       0.0000
```

2. Найти нули функции $y = 0.25x + \sin x - 1$ на отрезке [0; 10].

1 шаг – создание функции:

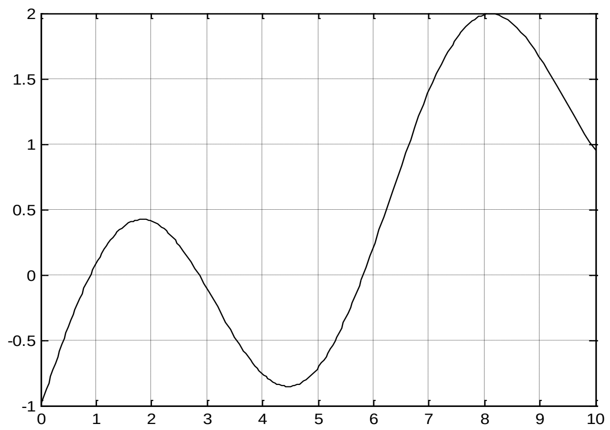
```
function f=fun1(x)
f=0.25*x+sin(x)-1;
```

2 шаг – сохранение файла fun1.m

3 шаг – работа в командном окне:

>> fplot(@fun1,[0,10]);   %построение графика функции fun1 на отрезке [0; 10]... для приближенного определения нулей функции

>> grid on;   % координатная сетка



>> x1=fzero(@fun1,[0 1])   %вычисление нуля функции на отрезке [0; 1]

x1 =

   0.8905

>> x2=fzero(@fun1,[2 3])   %вычисление нуля функции на отрезке [2; 3]

x2 =

   2.8500

>> x3=fzero(@fun1,5,0.001)   %вычисление нуля функции при x ≥ 5,... точность=0.0001

x3 =

   5.8128

3. Решить уравнение: $\sin(x^2 - 0.6) = 0$ на отрезке [0; 3].

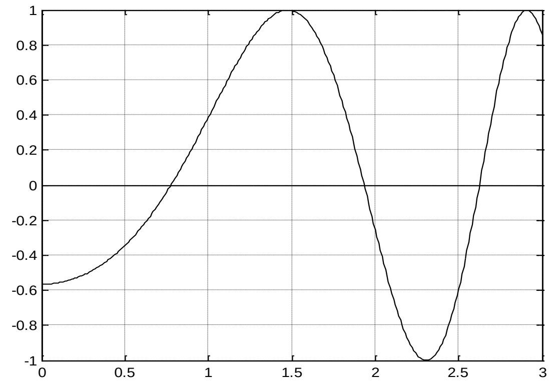
1. Графический способ:

>> x=0:.01:3;   %значения аргумента

>> f=sin(x.^2-0.6);   %значения функции

>> plot(x,[f;0*f])   % графики функций y=f и y=0

>> grid on;   %координатная сетка



>> x1=ginput   %интерактивный вывод на экран координат точки (щелчок... мыши по нужной точке и нажатие клавиши enter)

x1 =

      0.7746   -0.0012   %второе число соответствует координате y1≈0

>> g2=ginput

x2 =

      1.9343   0.0023

>> x3= fzero(f,[2 3])

x3 =

      2.6326

2. Аналитический способ:

```
>> x=0:.01:3;                                      %значения аргумента
>> n=length(x);                                    %вычисление длины вектора x
>> ind=1:n-1;                                      %вектор индексов
>> f=sin(x.^2-0.6);                                % значения функции
>> korni=x(f(ind).*f(ind+1)<=0)         %сравниваются знаки функции на концах...
                                        отрезка, вектор korni попадают только те...
                                        значения x_i , для которых f(x_i) и f(x_i+1)...
                                        имеют разные знаки

 korni =
          0.7746   1.9300   2.6200
```

Задание: Составить программу для решения алгебраического уравнения f(x) = 0 методом Ньютона (итерационная процедура по формуле: $x_{k+1} = x_k - f(x_k) / f'(x_k)$, начальное значение x_0,функция f и ее производная f ' задаются в командном окне).
Применить программу к решению уравнения из примера 3.

Самостоятельная работа 3
Работа с матрицами, системы алгебраических уравнений

Примеры:

1. Вычислить алгебраические дополнения к элементам матрицы A и обратную к этой матрице (A – квадратная матрица третьего порядка). Результат сравнить с функциями inv(A) и A^(-1).

```
function [C,C1,C2]=obr(A)      % объявление функции, сохраняемой по именем obr.m
d=det(A)                                       %вычисление определителя матрицы
if d==0 disp('матрица вырожденная');
else
        A11=det(A([2,3],[2,3]));
        A12=-det(A([2,3],[1,3]));
        A13=det(A([2,3],[1,2]));
        A21=-det(A([1,3],[2,3]));
        A22=det(A([1,3],[1,3]));
        A23=-det(A([1,3],[1,2]));
        A31=det(A([1,2],[2,3]));
        A32=-det(A([1,2],[1,3]));
        A33=det(A([1,2],[1,2]));
        B=[A11,A12,A13;A21,A22,A23;A31,A32,A33];
        C=B'/d;
        C1=inv(A);
        C2=A^(-1);
end                          %конец оператора if, далее сохраняем функцию в файле obr.m
```

Работа в командном окне:

```
>> A=[1 2 1;2 11 1;2 1 2]                              %исходная матрица
>> [C,C1,C2]=obr(A)                                     %вызов функции
C =
   -7.0000    1.0000    3.0000
    0.6667    0.0000   -0.3333
    6.6667   -1.0000   -2.3333
C1 =
   -7.0000    1.0000    3.0000
    0.6667    0.0000   -0.3333
    6.6667   -1.0000   -2.3333
C2 = C1
```

2. Решить систему уравнений:

$$
\begin{aligned}
2x + y - 5z + t &= 8,\\
x - 3y - 6t &= 9,\\
2y - z + 2t &= -5,\\
4y - 7z + 6t &= 0.
\end{aligned}
$$

```
>> A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];                  %матрица системы
>> B=[8;9;-5;0];          %вектор – стобец правых частей  (может быть матрицей)
>> A1=[A,B];                                      %расширенная марица системы
>> if  and(rank(A)=rank(A1),rank(A)=4)                  %проверка ранга матрицы
      disp('система имеет единственное решение:');
      x=A\B;              % обратный слэш или деление слева – решение  линейной...
                                                     системы методом Гаусса
      x1=x';
   end
x1 =
   3.0000   -4.0000   -1.0000   1.0000
>> x=A^(-1)*B;  x2=x'                             %второй  вариант  записи A\B
x2 =
  3.0000   -4.0000   -1.0000   1.0000
>> x=inv(A)*B;  x3=x'                             %третий  вариант  записи A\B
x3 =
  3.0000   -4.0000   -1.0000   1.0000
```

3. Решить систему из примера 2 методом наименьших квадратов.

```
>> A=[2 1 -5 1;1 -3 0 -6;0 2 -1 2;1 4 -7 6];                  %матрица системы
>> B=[8;9;-5;0];                                % вектор – стобец правых частей
>> x=lsqr(A,B)            % встроенная  функция  решения   линейной   системы...
                                            (метод  наименьших  квадратов)
x =
   3.0000   -4.0000   -1.0000   1.0000
```

Задания: **1.** Решить систему из примера 2 с помощью формул Крамера.

2. Решить систему нелинейных уравнений:

$$
\begin{aligned}
2x - y^2 + z &= 1,\\
y^2 - z^2 &= 0,\\
x + 2y - 3z &= 1,
\end{aligned}
$$

для решения применить метод последовательных приближений (итерационная процедура $x_{k+1} = B\backslash A(x_k)$).

Самостоятельная работа 4
Численное дифференцирование и интегрирование

Примеры: **1.** Вычислить производную полинома $P(x) = x^5 + x^3 + 1$.

```
>> P=[1 0  1 0 0 1];                          %вектор коэффициентов исходного
полинома
>> P1=polyder(P)                        %вектор коэффициентов  производной P' (x)
P1 =
     5   0   3   0   0
```

2. Вычислить приближенное значение производной функции $y = \sin x$ и убедиться, что $y'(a) \approx \cos a$.

```
>> x=0:.05:10;                                              %вектор аргументов
>> y=sin(x);                                         %вектор значений функции
```

```
>> d=diff(y);       %вектор разностей соседних элементов: d=[y(2)-y(1),...,y(n)-y(n-1)]
>> pr=d/0.05;                                      %вектор значений производной
>> pr(5)                                           %значение производной при x=x(5)=0.2
0.9747
 >> cos(x(5))                                      %сравнение с точным результатом
0.9801
```

3. Найти приближенное значение производной функции y = ln(1-x) в виде полинома 5-й степени. Вычислить y' (0.6).

```
>> h=.01;  x=0:h:0.8;                                  %значения аргумента
>> y=log(1-x);                                         %значения функции
>> y1=diff(y)/h;                                       %значения производной
>> p=polyfit(x(1:length(n)-1,y1,5)   %коэффициенты аппроксимирующего полинома
p =
  -52.0782   75.5962   -43.1952   8.8005   -1.8620   -0.9894
>> c=polyval(p,0.6)                              %значение полинома p при x=0.6
c =
     -2.5209                                     %точное значение  c= - 2.5
```

4. Вычислить интеграл двумя способами: $\int_0^1 (e^{2x} - 1)\, dx$.

1 способ – метод трапеций:

```
>> h=0.001;     x=0:h:1;
>> y=exp(2*x)-1;
>> int=trapz(y)*h
int =
    2.1945
```

2 способ – метод Симпсона:

```
>> int=quad('exp(2*x)-1',0,1,1.0e-5)
int = 2.1945
```

Задания: **1.** Вычислить приближенное значение второй производной функции y = tg x в точке x = $\pi/4$, результат сравнить с точным значением.

2. Вычислить различными способами: $\int_0^{\pi} dx \int_0^{2\pi} (2y\sin x + x\cos y)\, dy$.

а) методом повторного интегрирования;

б) с помощью функции dblquad( );

в) с помощью приближенной формулы $int \approx \sum\sum f(x_i, y_k)\, \Delta x_i\, \Delta y_k$.

`

Самостоятельная работа 5
Использование прикладных пакетов

Примеры: 1. Финансовая задача. Величина кредита составляет $1000. Он должен быть погашен за 12 месяцев тремя платежами. Заемная процентная ставка составляет 10%. Вычислить величину одного платежа. При решении использовать функцию payadv (rate, nper, pv, fv, adv) пакета Financial Toolbox, где rate – величина заемной ставки, nper - количество месяцев, pv - текущая стоимость финансового документа, fv – целевая стоимость, adv – количество платежей.

```
>>  vel= payadv(0.1/12,12, 1000, 0,3);
>>  string=['Вы должны выплачивать по', vel, '   долларов ', adv, '   раза!'];
>> disp(string)
Вы должны выплачивать по  85.9389   долларов  3 раза!
```

Графический интерфейс для работы с этой функцией в диалоговом режиме разработан в примере 10.1.

2. Вычислить в символьном виде, используя функции пакета Symbolic Toobox:

а) производную от функции y = x^3

```
>> sym x;
>> diff('x^3')
ans =
       3*x^2
```

б) первообразную от функции z = 2sin y

```
>> sym y;
>> int('2*sin(y)')
ans =
       -2*cos(y)
```

в) решения уравнения x^2=b

```
>> syms x b
>> solve(x^2-b)
ans =
      [ b^1/2]
      [-b^1/2]
```

г) решения квадратного уравнения $ax^2 + bx + c = 0$

```
>> syms a b c x
>> x=solve ('a*x^2+b*x+c=0')
x =
    [1/2*a*(-b+(b^2-4*a*c)^(1/2))]
    [1/2*a*(-b- (b^2-4*a*c)^(1/2))]
```

Задания: 1. С помощью функции nchoosek(n,k)=C_n^k построить разложение бинома $(x+y)^n$ для n=10. Ответ записать в символьном виде.
2. Даны два числовых массива x и y. Построить график интерполирующей функции y(x), используя функции пакета Spline Toolbox .

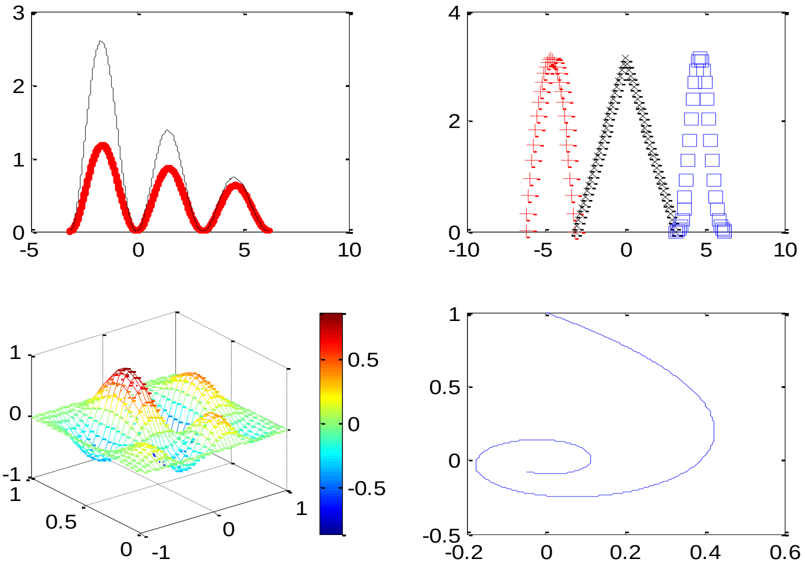
Самостоятельная работа 6
Использование прикладных пакетов

Изучить по справочной системе Matlab приемы работы с пакетом программ для решения систем дифференциальных уравнений в частных производных Partial Differential Toolbox, выполнить самостоятельное задание: решить задачу Дирихле в заданной области (область и условия на границе выбрать по своему усмотрению).

Самостоятельная работа 7
Графика в системе Matlab

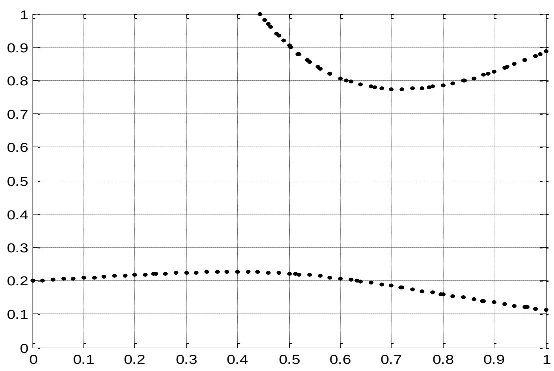
Примеры: 1. Разбить текущее графическое окно на четыре подокна и в каждом из них построить график: 1 – два графика функции одной переменной в одном графическом пространстве, 2 – график кусочно - заданной функции, 3 – график поверхности, 4 – график с эффектом анимации.

```
>> subplot(2,2,1)              %открытие первого подокна  (всего подокон... четыре - две  строки и два столбца)
>> x=-pi:0.01:2*pi;
>> f=exp(-0.1*x).*sin(x).^2;
>> y=-2*pi:0.01:pi;
>> g=exp(-0.2*y).*sin(y).^2;
>> plot(x,f,'r',x,g,'k-')

>> subplot(2,2,2);
>> x1=-2*pi:pi/30:-pi;
>> y1=pi*sin(x1);
>> x2=-pi:pi/30:pi;
>> y2=pi-abs(x2);
>> x3=pi:pi/30:2*pi;
>> y3=pi*sin(x1).^3;
>> x=[x1 x2 x3];
>> y=[y1 y2 y3];
>> plot(x1,y1,'r+',x2,y2,'kx',x3,y3,'bs')

>> subplot(2,2,3);
>> [X,Y]=meshgrid(-1:0.05:1,0:0.05:1);          %проекции узловых точек... поверхности
>> Z=4*sin(2*pi*X).*cos(1.5*pi*Y).*(1-X.^2).*Y.*(1-Y);
>> mesh(X,Y,Z);
>> colorbar                                      %шкала цветов

>> subplot(2,2,4);
>> t=[0:0.001:10];
>> x=sin(t)./(t+1);
>> y=cos(t)./(t+1);
>> comet(x,y);                                   %эффект кометы
```



2. Построить график функции, заданной неявно: $x^3y - 2xy^2 + y - 0.2 = 0$ в квадрате $|x| \le 1$, $|y| \le 1$.

```
>> h=0.02;  x=0:h:1;
>> [X,Y]=meshgrid(x);
>> f=X.^3.*Y-2*X.*Y.^2.+Y-.2;
>> v=[0 0]; contour(x,x,f,v); grid            %линия уровня поверхности... z=f(x,y) при z=0
```



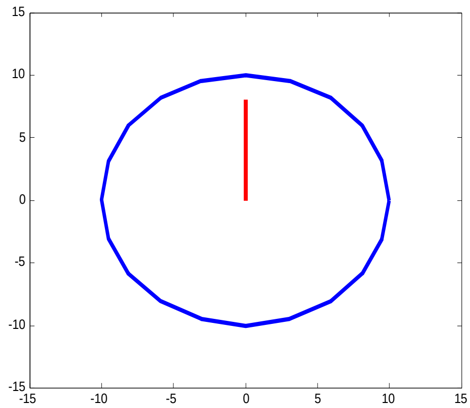
3. Построить график поверхности z = f(x,y) (f(x,y) – левая часть уравнения из примера 2) с нанесенным на нем рисунком из файла uzor.jpg. В соседнем подокне изобразить исходный (заранее подготовленный) рисунок.

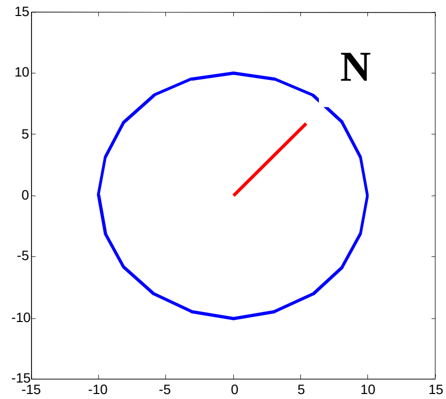
```
>> h=0.02;  x=0:h:1;
>> [X,Y]=meshgrid(x);
>> Z=X.^3.*Y-2*X.*Y.^2.+Y-.2;
>> A=imread(‘uzor’,’jpg’);       %считывание рисунка из файла, занесение...
                                    данных в графический объект A
>> subplot(1,2,1);
>> image(A)                                              %вывод рисунка
>> axes off
>> subplot(1,2,2);
>> surf(X,Y,Z,’CData’,A,’FaceColor’,’texture’)          %график поверхности
   >> axes off
```

4. Изобразить движение стрелки компаса при изменении положения точки N (север). Точку N вывести в диалоговом режиме с помощью функции gtext( ).

```
>> f=figure('Name','Kompas');                    %открытие графического окна
>> %рисование окружности
>> plot(10*cos(0:pi/10:2*pi),10*sin(0:pi/10:2*pi),'LineWidth',3);
>> hold on                                      %режим добавления графиков
>> axis([-15,15,-15,15]);                                   %границы осей
>> x1=[0 0]; y1=[0 8];                                %координаты стрелки
>> h=plot(x1,y1,'LineWidth',3,'Color','r','EraseMode','xor');      %стрелка
>> a=gtext('N');                                          %ввести точку N
>> b=get(a);                    %вывод графических характеристик объекта a
>> v=b.Position;
>> xn=v(1);     yn=v(2);                             %координаты точки N
>> R=sqrt(xn^2+yn^2);                  %вычисление расстояния до точки N
>> alfa=-asin(xn/R);                                        %угол поворота
>> xh=x1*cos(alfa)-y1*sin(alfa);                   %преобразование координат
```

```
>> yh=x1*sin(alfa)+y1*cos(alfa);
>> set(h,'XData',xh,'YData',yh)                %новая стрелка
```



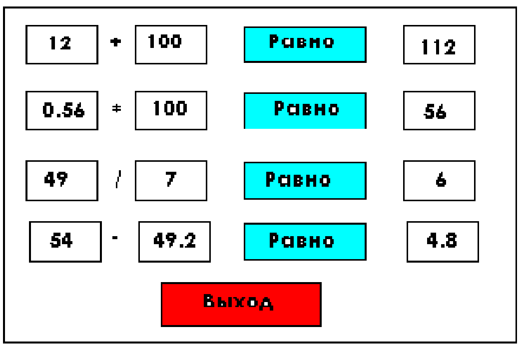


Задания: **1.** Изобразить циферблат часов с движущимися стрелками.

2. Построить график поверхности $z = 1 - x^2 - y^2$ при $0 < x < 1, 0 < y < 1$ с использованием функций shading interp, diffuse, colormap( ) и вычислить объем, заключенный между указанной поверхностью и плоскостью $z = 0$ $(V \approx \sum\sum z(x_i, y_k)\, \Delta x_i \Delta y_k)$.

Самостоятельная работа 8
Графический интерфейс пользователя

Примеры: 1. Разработать графическое окно, реализующее функции калькулятора:



Разработка: 1 шаг – создание бланка (File-GUI-Blank)

для ввода цифр – объекты типа Edit (свойство Tag – a, b или rez);

для надписей + - * / – объекты типа Text ;

для кнопок равно и выход – объекты типа Pushbutton (Tag - ravno и Close).

2 шаг – сохранение бланка в файле под именем calcullyator.fig.

3 шаг – изменение программы (файл calculyator.m):

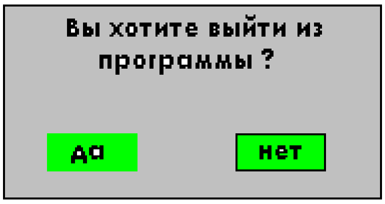
а) изменение кода кнопки равно для + (клавиша f – ravno_Callback):

```
function ravno_Callback(hObject,evendata,handles)
a=get(handles.a,'String');          %считывание первого слагаемого
a=str2double(a);                    %перевод из символьного вида в числовой
b=get(handles.b,'String');          %считывание второго слагаемого
b=str2double(b);                    %перевод из символьного вида в числовой
```

```
rez=a+b;                              %суммирование
set(handles.rez,'String',rez);        %запись результата в объект rez
```

б) изменение кода кнопок равно для *** / - :** по аналогии с пунктом а), но вместо a, b и rez соответственно a1, b1, rez1, a2, b2, rez2, a3,b3, rez3.

4 шаг – создание вспомогательного диалогового окна выход (File-GUI-Blank)



для надписи – объект типа Text ;
для кнопок да и нет – объекты типа Push Button (Tag - da и net).

5 шаг – сохранение вспомогательного окна в файле под именем danet.fig.
6 шаг – изменение программы диалогового окна (файл danet.m) :

а) изменение кода кнопки да (клавиша f – da_Callback):

```
function da_Callback(hObject,evendata,handles)
close;                        %закрытие диалогового окна выход
Close_Callback=close;         %выход из программы
```

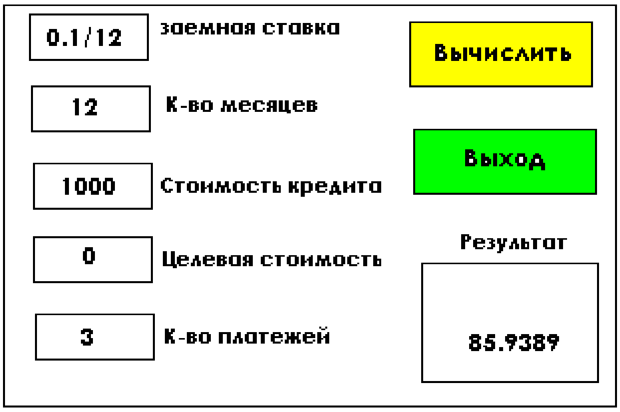
б) изменение кода кнопки нет (клавиша f – net_Callback):

```
function net_Callback(hObject,evendata,handles)
close;                        %закрытие диалогового окна выход
```

7 шаг – изменение кода кнопки выход: (клавиша f – Close_Callback):

```
function Close_Callback(hObject,evendata,handles)
danet;                        %вызов функции danet
```

2. Разработать диалоговое окно для работы с функцией payadv( ) (пример 8.1).



Разработка: 1 шаг – создание бланка (File-GUI-Blank)
для ввода цифр–объекты типа Edit (свойство Tag – rate, nper, pv, fv, adv);
для надписей – объекты типа Text ;
для кнопок вычислить и выход–объекты типа Push Button (Tag-pusk и close)
для вывода результата – объект типа Text (Tag – pmt).

2 шаг – сохранение бланка в файле под именем primer2.fig.
3 шаг – изменение программы (файл primer2.m):
изменение кода кнопки вычислить (клавиша f – pusk_Callback):

```
function pusk_Callback(hObject,evendata,handles)
rate=get(handles.rate,'String');              %считывание первого аргумента
rate=eval(rate);                              %вычисление выражения
nper=get(handles.nper,'String');              %считывание второго аргумента
nper=str2double(nper);              %перевод из символьного вида в числовой
pv=get(handles.pv,'String');                  %считывание третьего аргумента
pv=str2double(pv);                  %перевод из символьного вида в числовой
fv=get(handles.fv,'String';                   %считывание четвертого аргумента
fv=str2double(fv);                  %перевод из символьного вида в числовой
adv=get(handles.adv,'String');                %считывание пятого аргумента
adv=str2double(adv);                %перевод из символьного вида в числовой
pmt=payadv(rate,nper,pv,fv,adv)               %вызов функции
set(handles.pmt,'String',pmt);                %запись результата в объект pmt
```

4 шаг – работа с диалоговым окном выход (см. пример 10.1).

Задание: Разработать GUI для просмотра нескольких графических окон. Указание: создать объект PopupMenu, перечислить несколько характерных примеров (график функции одной переменной, эффект comet, график поверхности и пр.). Для вывода графического окна создать объект типа axes. В программе воспользоваться оператором выбора switch.

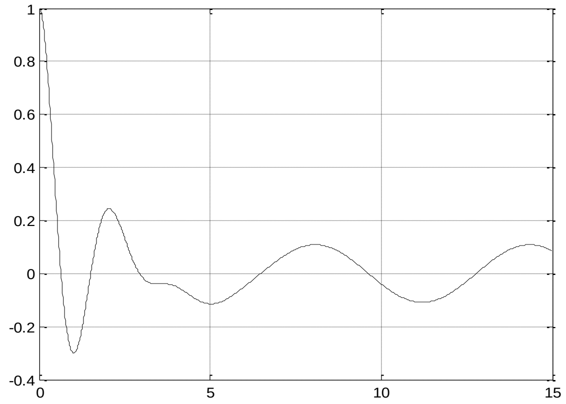
Самостоятельная работа 9
Численные методы решения дифференциальных уравнений

Примеры: 1. Решить задачу Коши: $y'' + 2y' +10y = \sin x$; $y(a) = c$, $y'(a) = d$, $x \in[a,b]$, к решению применить метод Эйлера, построить график функции y(x).
Указание: записать уравнение в виде системы: $y' = z$,
$z' = -2z - 10y + \sin x$,
разбить отрезок [a, b] на n частей $[x_k, x_{k+1}]$ и применить формулу $V_{k+1}=V_k + h*F(x_k, y_k)$, где V - вектор неизвестных, F - вектор правых частей, h - длина интервала, $V_0 = [c; d]$.

```
function y=solvejler(a,b,y0,h)                  % начало файла solvejler.m
xt=a;  yt=y0;   k=1;          %начальные значения текущих переменных xt, yt
while xt<b;
      x(k)=xt;
      y(k)=yt(1);
      k=k+1;
      c=[yt(2);-2*yt(2)-10*yt(1)+sin(xt)];          %вектор правых частей
      yt=yt+h*c;
      xt=xt+h;
end
plot(x,y); grid                        %график решения, сетка;  конец m-файла
```

Работа в командном окне:

```
>> a=0; b=15; y0=[1;0]; h=0.01;
>> solvejler(a,b,y0,h)                                        %вызов функии
```

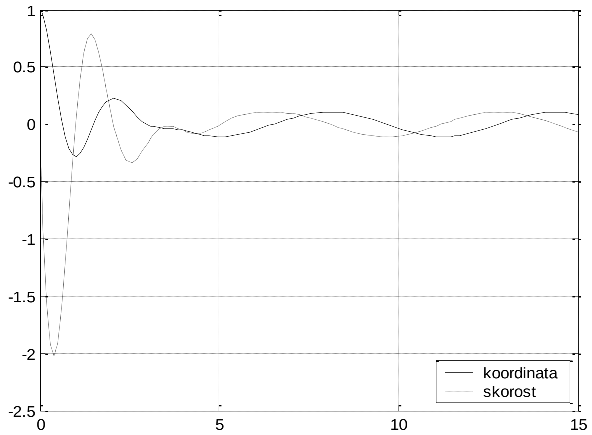


2. Решить задачу примера 1 методом Адамса с помощью стандартной функции ode113( ). Построить графики функций y(x) и y'(x).

```
function   F=primer(x,y)                    %m-файл, задающий вектор правых частей
F=[y(2);-2*y(2)-10*y(1)+sin(x)];                                       %конец m-файла
```

Работа в командном окне:

```
>> y0=[1;0];
>> [X,Y]=ode113(@primer,[0 15],y0);                        %решение задачи Коши
>> plot(X,Y(:,1),'-')                                      %график функции y(x)
>> hold on                         %режим добавления графиков в текущее окно
>> plot(X,Y(:,2),':')                                  %график функции y' (x)
>> grid                                                   %координатная сетка
>> legend('koordinata','skorost',4)                                  %легенда
```

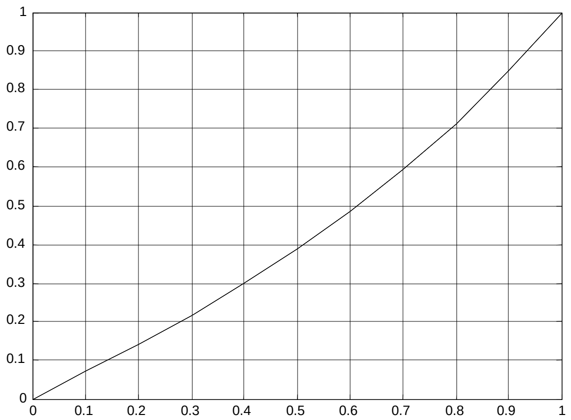


3. Решить краевую задачу: $y'' - y = x$; $y(0) = 0$, $y(1) = 1$.

Указание: разбить отрезок [0; 1] на n равных частей. Производные $y''(x_k)$ заменить конечными разностями $y''_k = (y_{k-1} - 2y_k + y_{k+1}) / h^2$, $h = x_{k+1} - x_k$, $k \geq 2$, решить полученную линейную систему Ay=B. Построить график решения и сравнить его с графиком точного решения y = c1*exp(x) + c2*exp(-x) – x, где $c1 = 2e/(e^2-1)$, c2 = - c1.

```
>> n=11; h=1/(n-1);   x=0:h:1;                               %разбиение отрезка
>> B=[0,h*h*x(2:n-1),1]';                         %вектор правых частей системы
>> k=-2-h^2;
>> A=[1 0 0 0 0 0 0 0 0 0 0;1 k 1 0 0 0 0 0 0 0 0;..        %матрица линейной системы
 0 1 k 1 0 0 0 0 0 0 0 ;0 0 1 k 1 0 0 0 0 0 0;...
 0 0 0 1 k 1 0 0 0 0 0;0 0 0 0 1 k 1 0 0 0 0;...
 0 0 0 0 0 1 k 1 0 0 0;0 0 0 0 0 0 1 k 1 0 0;...
 0 0 0 0 0 0 0 1 k 1 0;0 0 0 0 0 0 0 0 1 k 1;...
 0 0 0 0 0 0 0 0 0 0 1];
 >> y1=A\B;                                             %решение линейной системы
```

```
>> plot(x,y1);  grid                          %график приближенного решения
>> hold on                                    %режим  добавления  графиков
>> c1=2*exp(1)/(exp(1)^2-1);  c2=-c1;         %коэффициенты точного решения
>> y2=c1*exp(x)+c2*exp(-x)-x;                 %точное решение
>> plot(x,y2)            %график точного решения (совпадает с графиком y1)
```



Другой вариант формирования матрицы A:

```
>> v=[k,1,zeros(1,n-2)];
>> A=toeplitz(v);                 %матрица  Тёплица  с одним аргументом  (см. [2])
>> A(1,1)=1;  A(1,2)=0;
>> A(n,n-1)=0;  A(n,n)=1;
```

Задание: Решить краевую задачу из примера 3 двумя способами:

1) с помощью стандартной функции bvp4c( );

2) с помощью метода начальных параметров, т.е. сведением краевой задачи V' (x)= A(x)V(x)+B(x), V=[y(x), y'(x)]; y(a)=y1, y(b)=y2, к решению трех задач Коши. Решение ищется в виде суммы $V(x) = V_0(x) + c1 * V_1(x) + c2*V_2(x)$, где V_0 – решение уравнения V'=AV+B с нулевыми начальными условиями $V_0(0) = [0, 0]$; V_1 и V_2 – решения уравнения V'=AV, $V_1(0) = [1, 0]$, $V_2(0) = [0, 1]$.
Коэффициенты c1, c2 определяются из граничных условий.

Самостоятельная работа 10
Решение задачи изгиба пластины методом коллокаций и методом наименьших квадратов

Изучить по предлагаемому справочнику* [2, т.2] *указанные методы. Методом коллокаций (стр. 161-164) определить приближенное выражение для упругой поверхности жестко заделанной на опорном контуре квадратной изотропной пластины постоянной толщины, нагруженной постоянной по поверхности пластины поперечной нагрузкой. Провести решение той же задачи методом наименьших квадратов (стр.164-166), сравнить результаты.

Самостоятельная работа 11
Расчет пластины на изгиб методом Бубнова-Галеркина

Задание. Решить задачу изгиба тонкой прямоугольной пластины при условии шарнирного опирания краев. При решении опираться на приведенное ниже решение с другими краевыми условиями. Сравнить значения максимального

прогиба в центре пластины в первом и во втором случае. Изобразить изогнутую поверхность пластины средствами системы Matlab.

Для иллюстрации метода Бубнова—Галеркина рассмотрим изгиб защемленной по контуру прямоугольной пластинки, к которой приложена равномерно распределенная нагрузка. Направление координатных осей показано на рис. 1 .

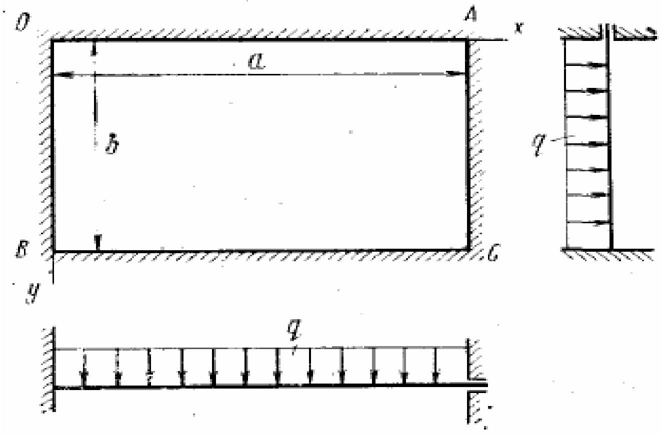


Рис. 1

Из характера закрепления пластинки вытекают следующие граничные условия. На гранях пластинки OB и AC

при $x = 0$ и $x = a$

$$w = \frac{\partial w}{\partial x} = 0. \qquad \text{(а)}$$

На гранях OA и BC

при $y = 0$ и $y = b$

$$w = \frac{\partial w}{\partial y} = 0. \qquad \text{(б)}$$

Чтобы удовлетворить этим условиям, приближенное выражение функции прогибов можно принять в виде ряда

$$w_n = \sum_{k=1}^{n} \sum_{l=1}^{n} a_{kl} \left(1 - \cos\frac{2k\pi x}{a}\right)\left(1 - \cos\frac{2l\pi y}{b}\right), \qquad \text{(в)}$$

где функция

$$\varphi_{kl} = \left(1 - \cos\frac{2k\pi x}{a}\right)\left(1 - \cos\frac{2l\pi y}{b}\right)$$

каждого его члена удовлетворяет всем граничным условиям. Так, на грани OB

$$\cos\frac{2k\pi x}{a}\Big|_{x=0} = \cos 0 = 1$$

и, следовательно, $w_n = 0$. На грани AC

$$\cos\frac{2k\pi x}{a}\Big|_{x=a} = \cos 2k\pi = 1$$

и тоже $w_n = 0$. Точно так же выполняются условия (б) для прогибов на гранях OA и BC.

Для проверки граничных условий в отношении углов поворота на контуре пластинки вычисляем производные от функции прогибов в) по x и y:

$$\frac{\partial w_n}{\partial x} = \frac{2\pi}{a} \sum_{k=1}^{n} \sum_{l=1}^{n} a_{kl}\, k \sin\frac{2k\pi x}{a}\left(1-\cos\frac{2l\pi y}{b}\right);$$

$$\frac{\partial w_n}{\partial y} = \frac{2\pi}{b} \sum_{k=1}^{n} \sum_{l=1}^{n} a_{kl}\, l\left(1-\cos\frac{2k\pi x}{a}\right)\sin\frac{2l\pi y}{b}.$$

На грани OB

$$\sin\frac{2k\pi x}{a}\Big|_{x=0} = \sin 0 = 0$$

и, следовательно, производная $\frac{\partial w_n}{\partial x} = 0$. Точно так же на грани AC

$$\sin\frac{2k\pi x}{a}\Big|_{x=a} = \sin 2k\pi = 0$$

и производная $\frac{\partial w_n}{\partial x} = 0$. Аналогично, на гранях OA и BC обращается в нуль производная $\frac{\partial w_n}{\partial y}$. Таким образом, функция прогибов (в) удовлетворяет всем граничным условиям (а) и (б).

Для отыскания неопределенных параметров a_{kl} нужно составить систему уравнений Бубнова—Галеркина. В первом приближении ограничимся одним членом ряда (в):

$$w_1 = a_{11}\left(1-\cos\frac{2\pi x}{a}\right)\left(1-\cos\frac{2\pi y}{b}\right). \qquad \text{(г)}$$

Тогда функция φ_{kl} для этого члена ряда будет

$$\varphi_{11} = \left(1-\cos\frac{2\pi x}{a}\right)\left(1-\cos\frac{2\pi y}{b}\right). \qquad \text{(д)}$$

Подставляя соотношения (г) и (д) в уравнения (7), получаем

$$\int_0^a\int_0^b\Big\{-16\pi^4 a_{11} D\Big[\frac{1}{a^4}\cos\frac{2\pi x}{a}\left(1-\cos\frac{2\pi y}{b}\right)-$$

$$-\frac{2}{a^2b^2}\cos\frac{2\pi x}{a}\cos\frac{2\pi y}{b}+\frac{1}{b^4}\left(1-\cos\frac{2\pi x}{a}\right)\cos\frac{2\pi y}{b}\Big]-$$

$$-q\Big\}\left(1-\cos\frac{2\pi x}{a}\right)\left(1-\cos\frac{2\pi y}{b}\right)dx\,dy = 0.$$

После преобразования приходим к сумме произведений интегралов:

$$-16\pi^4 a_{11} D\Big[\frac{1}{a^4}\int_0^a\left(\cos\frac{2\pi x}{a}-\cos^2\frac{2\pi x}{a}\right)dx\int_0^a\left(1-2\cos\frac{2\pi y}{b}+\right.$$

$$\left.+\cos^2\frac{2\pi y}{b}\right)dy-\frac{2}{a^2b^2}\int_0^a\left(\cos\frac{2\pi x}{a}-\cos^2\frac{2\pi x}{a}\right)dx\int_0^b\left(\cos\frac{2\pi y}{b}-\right.$$

$$-\cos^2\frac{2\pi y}{b}\Big)dy+\frac{1}{b^4}\int_0^a\Big(1-2\cos\frac{2\pi x}{a}+\cos^2\frac{2\pi x}{a}\Big)dx\int_0^b\Big(\cos\frac{2\pi y}{b}-$$

$$-\cos^2\frac{2\pi y}{b}\Big)dy\Big]-q\int_0^a\Big(1-\cos\frac{2\pi x}{a}\Big)dx\int_0^b\Big(1-\cos\frac{2\pi y}{b}\Big)dy=0.$$

Интегрируя, получаем

$$-16\pi^4 a_{11} D\Big[\frac{1}{a^4}\Big(-\frac{a}{2}\Big)\Big(b+\frac{b}{2}\Big)-\frac{2}{a^2b^2}\Big(-\frac{a}{2}\Big)\Big(-\frac{b}{2}\Big)+$$

$$+\frac{1}{b^4}\Big(a+\frac{a}{2}\Big)\Big(-\frac{b}{2}\Big)\Big]-qab=0$$

или после упрощения

$$16\pi^4 a_{11} D\Big(\frac{3b}{4a^3}+\frac{1}{2ab}+\frac{3a}{4b^3}\Big)-qab=0.$$

Отсюда коэффициент

$$a_{11}=\frac{qa^4}{4\pi^4 D}\,\frac{1}{3+2a^2/b^2+3a^4/b^4}.$$

Внося полученное значение a_{11} в формулу (г), находим функцию прогибов в первом приближении:

$$w_1=\frac{qa^4}{4\pi^4 D}\,\frac{\Big(1-\cos\dfrac{2\pi x}{a}\Big)\Big(1-\cos\dfrac{2\pi y}{b}\Big)}{3+2a^2/b^2+3a^4/b^4}.$$

Максимальный прогиб возникает в центре пластинки (при $x=a/2$ и $y=b/2$). Для квадратной пластинки при $a=b$ и коэффициенте Пуассона $\nu=0{,}3$ получаем следующее значение максимального прогиба:

$$\max w_1=qa^4/(8\pi^4D)=qa^4/(8\pi^4Eh^3)\,12\,(1-\nu^2)=$$
$$=0{,}0140qa^4/(Eh^3).$$

Точное значение максимального прогиба квадратной пластинки, защемленной по контуру и находящейся под действием равномерно распределенной нагрузки,

$$\max w=0{,}0138qa^4/(Eh^3).$$

Таким образом, максимальный прогиб, полученный в первом приближении, отличается от точного значения менее чем на 1,5%.

При вычислении изгибающих моментов и поперечных сил ряды сходятся значительно хуже.

Самостоятельная работа 12
Расчет устойчивости пластины под действием динамического сжатия

Рассчитать на динамическую устойчивость прямоугольную пластину под действием всестороннего сжатия. Процесс потери устойчивости изобразить на экране дисплея, методом графического анализа определить момент потери устойчивости. Расчет и графический анализ производить в системе MATLAB с использованием изученных методов. Подробный алгоритм выполнения задания приведен в лекции 16.

Литература

1. Бахтиева Л.У. Научно-технические расчеты в системе Matlab: учебное пособие для студентов и аспирантов естественнонаучных факультетов.-Казань, КГУ, 2007.

2. Палий О.М. Справочник по строительной механике, т.1-2, Л., 1982.

3. Самуль В.И. Основы теории упругости и пластчности: учебное пособие для студентов вузов. – 2-е изд., перераб. – М., Высш. Школа, 1982.

4. Вольмир А.С. Устойчивость упругих систем, М., Наука, 1967.