Алгоритмы как основа автоматизации инженерных расчетов.

Алгоритм – конечная последовательность однозначных предписаний, исполнение которых позволяет с помощью конечного числа шагов получить решение поставленной задачи, однозначно определяемое исходными данными

Основные требования: дискретность, детерминированность, конечность, результативность, массовость .

Итог – единственный конечный результат за конечное число шагов, примененное к допустимым исходным данным.

Восходящее и нисходящее проектирование.

Восходящее проектирование – сначала разрабатываются алгоритмы решения отдельных подзадач, а потом программа собирается в единое целое. Используется для небольших задач.

Нисходящее проектирование – определяется основная задача, разбивается на подэтапы. Каждый из подэтапов также может быть разбит на отдельные элементы. Отладка программы происходит поэтапно. Используется ВСЕГДА.

М-функции: назначение и способ построения

Функция – фрагмент программного кода, к которому можно обращаться.

Состав: имя, вход, выход, тело.

Порядок действий при создании: - Войти в текущую папку - Создать файл функции - В открывшемся окне набрать текст тела функции, входные и выходные данные - Сохранить с именем и расширением .m

Порядок действий при вызове: - Набрать имя функции с указанием имен входных и выходных переменных.

Каждая функция располагается в отдельном файле. Имя файла и имя функции совпадают.

Управляющие конструкции в MatLab

В MATLAB определены следующие управляющие конструкции и команды:

```
for % цикл с заданным числом шагов
while % цикл по условию
if else % условие
elseif % дополнительное условие
else % в противном случае
switch % дерево ветвлений
case % ветка дерева ветвлений
otherwise % в противном случае
break % преждевременный выход из цикла
continue % перейти к началу цикла
```

Степенные ряды и их графическое представление

Степенной ряд - разновидность функционального ряда, члены которого имеют вид

c(n)*x^n (n от 0)

В графичеком виде они как правило имеют вид парабол или гипербол.

Графики параметрически заданных функций

Для построения функций заданных параметрически, следует сперва сгенерировать вектор значений аргумента. Затем необходимо вычислить значения функций и записать их в векторы, которые и надо использовать в качестве аргументов plot.

ф-я построения графика

```
t=0:pi/100:2*pi;
x=3*t;
y=t.^2-4;
plot(x,y)
```

Способы представления алгоритмов

Словесный(естественный), формульно-словесный(псевдо-код), табличный(расписание), графический(блок-схемы).

Переменные. константы, арифметические и логические операции в MatLab

Переменная – выделенная область памяти для хранения данных.

Константа – переменная с постоянным значением.

Арифметические операции в Octave: + (сложение), - (вычитание), * (умножение), / (деление), ^ (возведение в степень).

Операции отношения в Octave: > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), == (равно), ~= (не равно).

Логические операции в Octave: & - и, | - или, ~ - не

Способы построения двумерных графиков

С помощью команд plot(x,y)и comet(x,y) можно построить двумерные графики.

%plot полностью выводит построенный график на экран.

%comet поэтапно строит график, пока не построит полностью.

Построение графиков m-функций

Построение графиков функций в MATLAB можно реализовать разными способами, например, через plot или polar.

plot(y) создает кусочно-линейный график зависимости элементов y от их индексов

plot(x,y) создаст график зависимости y от x. Например, для построения графика sin(x)

Команда hold on позволяет размещать несколько кривых на одном графике. Функция subplot позволяет выводить множество графиков в одном окне.

Числовые ряды и способы их вычисления

Числовые ряды имеют вид

a(n){n=1->a}=a(1)+a(2)+...+a(n) где все a(n) - числа

Числовые ряды могут вычисляться через цикл

```
for i=1:1:n
sum = sum + a(i);
endfor
```

либо через функцию

Полярные и декартовые координаты точек на плоскости. Перевод из одной системы в другую

Декартова система координат на плоскости определяется некоторой ее точкой O и базисом из двух векторов.

Полярная система координат представляет собой точку O и ось исходящую из полюса. Полярные координаты p и j. p- расстояние от Точки O до искомой точки, j - полярный угол - угол между осью и вектором OM (отсчитывается от оси против часовой стрелки)

функция для перевода полярных координат в декартовые

```
function [ x,y ] = polar_to_decar( r,fi )
x = r*cos(fi);
y = r*sin(fi);
end
function [ r,fi ] = decar_to_polar( x,y )
r = sqrt(x^2 + y^2);
fi = atan2(y,x);
end
```

Графики в трехмерном пространстве.

В системе MATLAB предусмотрено несколько функций для построения трехмерных графиков. Значения элементов числового массива рассматриваются как z-координаты точек над плоскостью, определяемой координатами x и y. Возможно несколько способов соединения этих точек. Первый из них - это соединение точек в сечении (функция plot3 и comet3), второй -построение сетчатых поверхностей (функции mesh и surf).

Основные программные конструкции: сумма, максимум

Сумма:

```
s=0;
for j=1:1:10
a=input('num');
s+=a;
end;
```

Максимум:

```
max=A(1)
for j=2:length(A)
if (A(j)>max) max=A(j);
end; end;
```

Графики функций многих переменных

```
hold off;
 x=-1:0.1:1;
y=-2:0.1:2;
[X,Y]=meshgrid(x,y);
% Функция [X,Y] = meshgrid(x,y) преобразует векторные (одномерные)
массивы x,y в матричные (двухмерные) массивы X,Y соответственно.
Z = exp(-X.^2 - Y.^2);
surf(X,Y,Z)
```

Понятие интерполяции. Методы интерполяции

Интерполяция (интерполирование) — процесс нахождения промежуточных значений по ряду данных, для восполнения пробелов между точными значениями приближенными. Точные значения так же называют узловыми точками.

Алгоритм Линейной интерполяции строит прямыи линии между узловыми точками.

В Matlab такой способ реализован с помощью команды interp1(x,y, xi, 'linear')

Есть разные интерполяционные полиномы — функции, определяющие как будут изменяться приближенные значения между узловыми точками: Канонический полином Полином Лагранжа Полином Ньютона

```
function yy=lagrange(x,y,xx)
% вычисление интерполяционного полинома в форме Лагранжа
% x - массив координат узлов
% y - массив значений интерполируемой функции
% xx - массив значений аргумента, для которых надо вычислить значения полинома
% yy - массив значений полинома в точках xx
% узнаем число узлов интерполяции
(N=n+1) N=length(x);
% создаем нулевой массив значений интерполяционного полинома
yy=zeros(size(xx));
% в цикле считаем сумму по узлам
for k=1:N
% вычисляем произведения, т.е. функции Psi_k
t=ones(size(xx));
for j=[1:k-1, k+1:N]
t=t.*(xx-x(j))/(x(k)-x(j));
end
% накапливаем сумму
yy = yy + y(k)*t;
end
```

Теперь создайте ещѐ один файл и запишем в него само решение поставленной задачи:

```
% задание узлов интерполяции
x=1:2:9;
y=sin(x);
% задание точек, в которых требуется найти значения интерполяционного полинома
xx=linspace(1,9,1000);
% нахождение значений интерполяционного полинома
yy=lagrange(x,y,xx);
% построение графиков
figure('Color','w')
% вывод графика sin(x)
fplot(@sin,[1 9]) hold on
% вывод графика полинома
plot(xx,yy,'r')
% вывод узлов интерполяции
plot(x,y,'bo')
```

Двумерные массивы в MatLab. Способы задания и операции с ними

Двумерный массив (матрица) – упорядоченный набор элементов, каждый из которых однозначно определяется его двумя индексами

Двумерный массив состоит из:

- Имя массива (переменной)
- Числа строк
- Числа столбцов
- Значений элементов

Элементы могут быть РАЗЛИЧНЫХ ТИПОВ

Для задания диапазона необходимо использовать значок «Переменная диапазон» на панели «Матрица»

Синтаксис :

Существует несколько способов задать двумерный массив в Matlab:

1. Задание элементов массива в виде матрицы. Например:

```
A = [1 2 3; 4 5 6; 7 8 9];
```

2. С помощью функций, таких как zeros, ones или rand. Например:

```
B = zeros(3,4);
```

3. Загрузка данных из файла. Например:

```
C = load('data.txt');
```

Понятие регрессии. Сглаживание

Регрессия – приближение выборки данных {(xj,yj)} непрерывной функцией, минимизирующей ошибки в узлах

В Octave применяется использованием встроенной функции LinearRegression (F, y);

Рекурсивные функции

Под рекурсией понимается метод определения функции через еѐ предыдущие и ранее определенные значения, а так же способ организации вычислений, при котором функция вызывает сама себя с другим аргументом.

Пример рекурсивной функции для вычисления факториала числа:

f(0)=1

f(n)= n*f(n-1)

Последовательная подстановка дает – f(n)=1*2*3*…*n = n!

Одномерные массивы в MatLab. Способы задания и операции с ними

Одномерный массив – упорядоченный набор элементов; каждый элемент однозначно определяется его порядковым номером (от первого до последнего) В MatLab нумерация начинается с 1, а элементы могут быть различных типов

```
A = [1, 2, 3, 4, 5]; % Создание одномерного массива
B = A + 2; % Добавление 2 к каждому элементу массива
C = B .* 2; % Умножение каждого элемента массива на 2
sumC = sum(C); % Сумма элементов массива C
disp(C); % Вывод массива C
```

Основные программные конструкции: счетчик, признак

Счётчик:

```
ch=0
for j=1:1:10
a=input(‘  ‘)
if (a>0) ch+=1;
end; end;
```

Признак:

```
pr =1 %«все четные»
for j=1:1:10
a=input(‘n’);
if (mod(a,z)==1) pr = 0;
break;
end; end
```

Построить m-функцию, вычисляющую сумму слагаемых числового ряда до достижения условия окончания

```
N = input('enter number');
sum = N;
while (N ~= 0)
        N = input('enter number');
        sum = sum+N;
end;
disp(sum);
```

Построить график функции – степенного ряда

```
% Задание значения переменной x и количества членов ряда n
x = linspace(0, 5, 100);
n = 10;
y=0;


for k = 0:n
  y = y + x;
end

% Построение графика функции степенного ряда
figure;
plot(x, y = (x.^n));
xlabel('X');
ylabel('Y');
title('График функции степенного ряда');

% Дополнительные настройки графика
grid on;
```

Построить график функции двух переменных

```
% Задание диапазонов значений переменных x и y
x = linspace(-5, 5, 100);
y = linspace(-5, 5, 100);

% Создание сетки значений переменных x и y
[X, Y] = meshgrid(x, y);

% Определение функции двух переменных
function z = myFunction(X, Y)
   z = sin(Y)+sin(X);
end

% Вычисление значений функции для каждой точки на сетке
Z = myFunction(X, Y);

% Построение 3D-графика
figure;
surf(X, Y, Z);
xlabel('X');
ylabel('Y');
zlabel('Z');
title('График функции двух переменных');

shading interp;  % Сглаживание графика
```

создать м-функцию для вычисления выражения (ф-я 3-х пер-х) f(x, a, p)
построить на одном графике для значений x, a, p при p=4, варьируя a от 1 до 4 с интервалом 1

```
clear;
hold off;
p = 4;
for a = 1:1:4;
x = -5:0.1:5;
for j = 1:1:length(x)
y(j) = ip(x(j),a,p);
end;
plot(x,y);
hold on;
endfor;
```

Построить функцию перевода декартовых координат в полярные и обратно

```
function [d, alf] = c2p(x, y)
   d = sqrt(x^2 + y^2);
   alf = atan2(y, x);
end

function [x, y] = p2c(d, alf)
   x = d * cos(alf);
   y = d * sin(alf);
end

[d, alf] = c2p(x, y);
fprintf('Полярные координаты: d = %f, alf = %f\n', d, alf);

[x_new, y_new] = p2c(d, alf);
fprintf('Декартовы координаты: x = %f, y = %f\n', x_new, y_new);
```

Построить линейную интерполяцию заданного набора точек, сплайн-Интерполяцию

```
% Задание набора точек
x = [0, 1, 2, 3, 4];
y = [0, 1, 4, 9, 16];

% Линейная интерполяция
xi = linspace(min(x), max(x), 100);
yi_linear = interp1(x, y, xi, 'linear');

% Сплайн-интерполяция
yi_spline = spline(x, y, xi);

% Построение графика
figure;
plot(x, y, 'o', xi, yi_linear, 'g', xi, yi_spline, 'b');
%Линейная интерполяция построена зелёной линией, сплайн-интерполяция построена синей линией, исходный точки отмечены ркужочками
xlabel('X');
ylabel('Y');

% Дополнительные настройки графика
grid on;
```

Построить m-функцию, вычисляющую сумму заданного числа слагаемых числового ряда

```
function [] = m3_3 ()
clear;
sum = 0;
 k = 0;
n = input('Введите количсвто чисел'); ## число n показывает сколько всего чисел будет введено
 for j = 1:1:n ## цикл обрабатывает входные числа
a = input('введите числа'); ## вводим число
sum = sum + a; ## Находим сумму всех чисел
if ((a>3) &&(a<10)) ## если число в интервале от 3 до 10,
k = k+1; ## то увеличиваем счетчик на 1 endif
endfor
disp(sum); ## выводим сумму всех чисел
disp(k); ## количество чисел в интервале от 3 до 10
endfunction
```