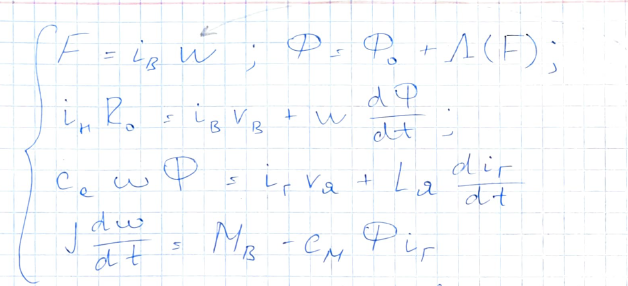
Титульник

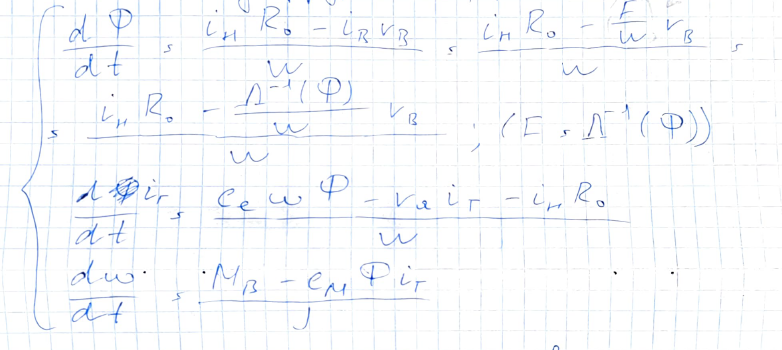
*Цель работы:*преобразовать исходную систему уравнений в СНЛАУ, описывающyю статические режимы; используя пакет MATLAB, решить полученную СНЛАУ и рассчитать статические характеристики динамической системы.

В исходных уравнениях объекта выделить переменные состояния и входные переменные. Исключить промежуточные переменные и записать СНЛАУ в нормированном виде.

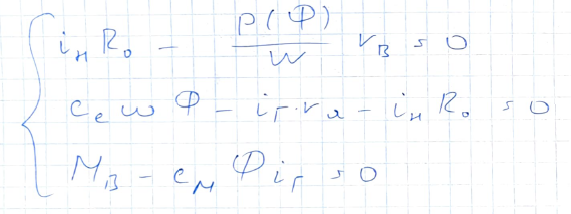
Исходные уравнения:



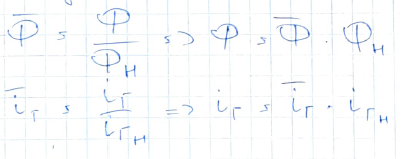
Система диффуров в форме Коши

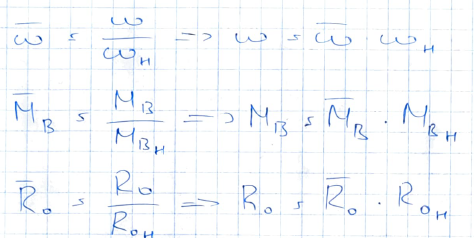


Исключим переменные, не являющиеся переменными состояния

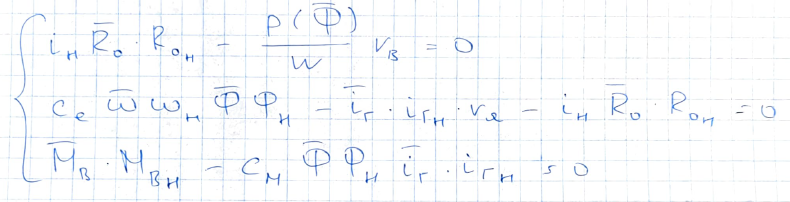


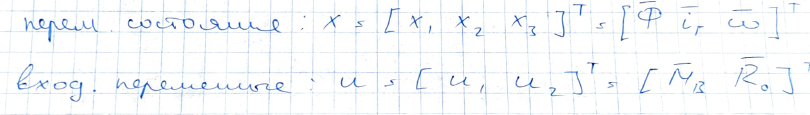
Нормируем все переменные входа и выхода

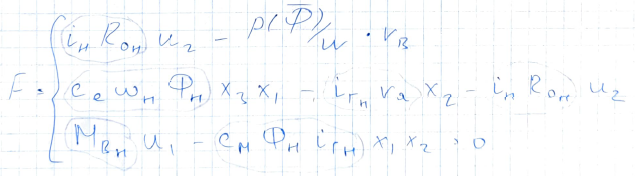




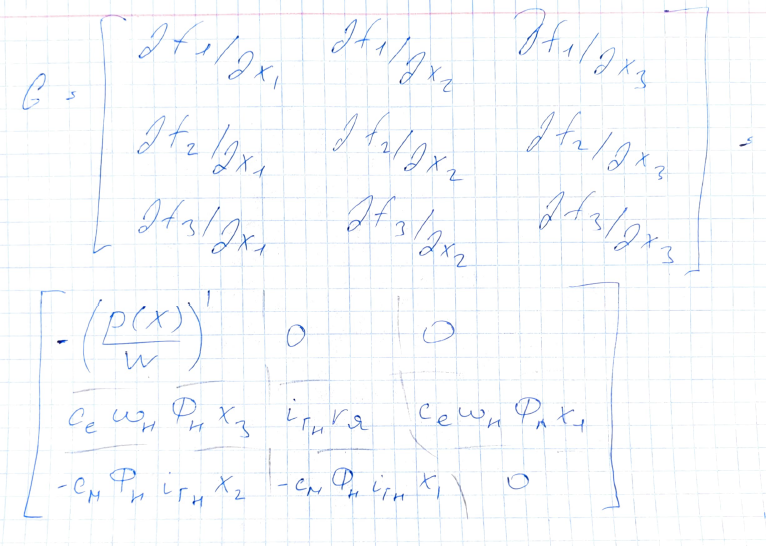
Перепишем систему с учетом нормирования переменных







Запишем матрицу частных производных



Напишем код для построения семейства статических характеристик

fun\_F.m

|  |
| --- |
| % вычисление значения функции f(x, u)  function f = fun\_F(x,u)      global r\_V;      global r\_YA;      global w;      global L\_YA;      global R\_0;      global c\_e;      global c\_M;      global J;      global Fi\_N;      global omega\_N;      global i\_N;      global M\_VN;      global U\_VN;      global i\_VN;      global i\_GN;      global p;      f1 = i\_N \* R\_0 \* u(2) - polyval(p./w, x(1));      f2 = c\_e \* omega\_N \* Fi\_N \* x(3) \* x(1) - i\_GN \* r\_YA \* x(2) - i\_N \* R\_0 \* u(2);      f3 = M\_VN \* u(1) - c\_M \* Fi\_N \* i\_GN \* x(1) \* x(2);      f = [f1; f2; f3];      end |

fun\_G.m

|  |
| --- |
| % вычисление значений матрицы частных производных G(x, u)  function g = fun\_G(x,u)      global r\_V;      global r\_YA;      global w;      global L\_YA;      global R\_0;      global c\_e;      global c\_M;      global J;      global Fi\_N;      global omega\_N;      global i\_N;      global M\_VN;      global U\_VN;      global i\_VN;      global i\_GN;      global p;      g1 = [ -polyval(polyder(p./w),x(1)), 0, 0]; % polyder возвращает производную полинома 'p'      g2 = [c\_e \* omega\_N \* Fi\_N \* x(3), i\_GN \* r\_YA, c\_e \* omega\_N \* Fi\_N\*x(1)];      g3 = [-c\_M \* Fi\_N \* i\_GN \* x(2), -c\_M \* Fi\_N \* i\_GN \* x(1), 0];      g = [g1; g2; g3];      end |

newton.m

|  |
| --- |
| function [x] = newton(F,G,x0,u,e)  % solve nonlinear algebraic equation system by Newton method  %  %         f(x,u)=0,  %                  -1  %         x   =x -g  (x ,u)\*f(x,u )   , g(x,u)=grad(f(x,u))  %          k+1  k      k     k  %  %         x =x0;   |f(x   ,u)| < e  %          0           k+1  %  %         F - name of function which calculates y=f(x,u) (should be written - 'F')  %             function y=F(x,u) have following input variables:  %         G - name of function which calculates gradient f by x (written as 'G')  %             function g=G(x,u) have following input variables:  %         x - argument (vector column);  %         u - parameter's vector (vector column)  %        x0 - initial condition;  %         e - precision (number)      t=0;      y=feval(F,x0,u);      x=x0;      while(norm(y)>e)          gr=feval(G,x,u);          x=x-inv(gr)\*y;          y=feval(F,x,u);          clc,disp(y)      end      end |

main.m

|  |
| --- |
| % очистка экрана и рабочего пространства  clc;  clear;  close;  global r\_V; r\_V = 97.2;  global r\_YA; r\_YA = 0.05;  global w; w = 3000;  global L\_YA; L\_YA = 0.03;  global R\_0; R\_0 = 4;  global c\_e; c\_e = 130;  global c\_M; c\_M = 120;  global J; J = 1.2;  global Fi\_N; Fi\_N = 0.01;  global omega\_N; omega\_N = 100;  global i\_N; i\_N = 100;  global M\_VN; M\_VN = 300;  global U\_VN; U\_VN = 400; % лр №1  global i\_VN; i\_VN = U\_VN / r\_V;  global i\_GN; i\_GN = i\_N + i\_VN  global p; p = [-0.0014 0.1628 -0.4707 0.4731 -0.0316 0];  % u1 = var, u2 = const  u1 = 0.2:0.2:1.2;  u\_1 = [u1; ones(size(u1)); ones(size(u1))];  x0 = [1; 1; 1];  xx = [];  for i = 1:length(u1)      x = newton('fun\_F', 'fun\_G', x0, u\_1(:,i), 0.001);      xx = [xx x];      x0 = x;  end  % уменьшим на 20% u1  u1\_minus20 = 0.2 \* 0.8:0.2 \* 0.8:1.2 \* 0.8;  u\_1\_minus20 = [u1\_minus20; ones(size(u1\_minus20)); ones(size(u1\_minus20))];  x0\_minus20 = [1; 1; 1];  xx\_minus20 = [];  for i = 1:length(u1\_minus20)      x\_minus20 = newton('fun\_F', 'fun\_G', x0\_minus20, u\_1(:,i), 0.001);      xx\_minus20 = [xx\_minus20 x\_minus20];      x0\_minus20 = x\_minus20;  end  %построение графиков х1(u1), х2(u1), х3(u1)  figure(1);  plot(u1, xx(1,:), 'b-', u1\_minus20, xx\_minus20(1,:), 'r--');  legend('Ф\_н(M\_В)', 'Ф\_н(M\_В\*0.8)');  xlabel('M\_В');  ylabel('Ф\_н');  grid on;  figure(2);  plot(u1, xx(2,:), 'b-', u1\_minus20, xx\_minus20(2,:), 'r--');  legend('i\_гн(M\_В)', 'i\_гн(M\_В\*0.8)');  xlabel('M\_В');  ylabel('i\_гн');  grid on;  figure(3);  plot(u1, xx(3,:), 'b-', u1\_minus20, xx\_minus20(3,:), 'r--');  legend('w\_н(M\_В)', 'w\_н(M\_В\*0.8)');  xlabel('M\_В'),  ylabel('w\_н');  grid on;  % u1 = const, u2 = var  u2 = 0.2:0.2:1.2;  u\_2 = [ones(size(u2)); u2; ones(size(u2))];  x0 = [1; 1; 1];  xx = [];  for i = 1:length(u2)      x = newton('fun\_F', 'fun\_G', x0, u\_2(:,i), 0.001);      xx = [xx x];      x0 = x;  end  % уменьшим на 20% u2  u2\_minus20 = 0.2 \* 0.8:0.2 \* 0.8:1.2 \* 0.8;  u\_2\_minus20 = [ones(size(u2\_minus20)); u2\_minus20; ones(size(u2\_minus20))];  x0\_minus20 = [1; 1; 1];  xx\_minus20 = [];  for i = 1:length(u2\_minus20)      x\_minus20 = newton('fun\_F', 'fun\_G', x0\_minus20, u\_2\_minus20(:,i), 0.001);      xx\_minus20 = [xx\_minus20 x\_minus20];      x0\_minus20 = x\_minus20;  end  %построение графиков х1(u2), х2(u2), х3(u2)  figure(4);  plot(u2, xx(1,:), 'b-', u2\_minus20, xx\_minus20(1,:), 'r--');  legend('Ф\_н(R\_0)', 'Ф\_н(R\_0\*0.8)');  xlabel('R\_0');  ylabel('Ф\_н');  grid on;  figure(5);  plot(u2, xx(2,:), 'b-', u2\_minus20, xx\_minus20(2,:), 'r--');  legend('i\_гн(R\_0)', 'i\_гн(R\_0\*0.8)');  xlabel('R\_0');  ylabel('i\_гн');  grid on;  figure(6);  plot(u2, xx(3,:), 'b-', u2\_minus20, xx\_minus20(3,:), 'r--');  legend('w\_н(R\_0)', 'w\_н(R\_0\*0.8)');  xlabel('R\_0');  ylabel('w\_н');  grid on; |

Графики













По идее вывод по характере полученных характеристик (близки к линейным и тд и тп…)

вывод