МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ПУТЕЙ СООБЩЕНИЯ (МИИТ)

Кафедра
«Управление и информатика в технических системах»

ПАРАМЕТРИЧЕСКИЙ СИНТЕЗ САУ С ПОМОЩЬЮ ПАКЕТОВ ПРИКЛАДНЫХ ПРОГРАММ

Рекомендовано редакционно-издательским советом университета в качестве методических указаний

для студентов специальности
“УПРАВЛЕНИЕ И ИНФОРМАТИКА В ТЕХНИЧЕСКИХ СИСТЕМАХ”

МОСКВА – 2010

УДК: 681.3.001.2:621.396.6

М17

Монахов О. И., Сафронов А. И., Ковалев М. В.. Рындина Е. Ю. Параметрический синтез САУ с помощью пакетов прикладных программ. Методические указания к курсовому проектированию по дисциплине «Автоматизация проектирования систем и средств управления». – М.: МИИТ. 2010. – 138 с.

Данные методические указания предназначены для изучения основ проектирования в рамках курса «Автоматизация проектирования систем и средств управления», а также могут быть использованы при выполнении лабораторных работ и в дипломном проектировании. Методические указания составлены в виде описания последовательности действий пользователя при работе с пакетами MBTY, MATLAB, LABVIEW с подробными комментариями. Изучать принципы работы пакетов рекомендуется в процессе выполнения заданий, приведенных в конце каждого раздела.

Предисловие

Курсовая работа – это, в принципе, самостоятельная работа студента, при выполнении которой возникает много вопросов, ответы на которые можно получить, в частности, на консультациях с преподавателем. Чаще всего это вопросы, связанные с выполнением тех или иных разделов, алгоритмами работы программ, оформлением пояснительной записки и т.п. Однако конкретные вопросы, связанные с методикой работы на ПЭВМ с использованием сложных программных продуктов, заданием численных значений параметров алгоритмов, последовательностью действий студента при формировании тех или иных графических образов и т.п. иногда остаются не очень понятными.

Данный материал, предназначенный для студентов 5-го курса специальности «Управление и Информатика в Технических Системах», озабоченных выполнением курсового проекта по дисциплине «Автоматизация проектирования систем и средств управления». В этих методических указаниях подробно описываются действия студента при решении конкретной задачи (устойчивость, качество, оптимизация параметров САУ) при использовании таких программных продуктов как LabView, MathLab, МВТУ. Даются рекомендации по разработке индивидуальной программы для решения подобных задач с использованием языка Delphi.

Нижеприведённые разделы написаны:

1. MathLab – Сафронов А.И.
2. МВТУ – Ковалёв М.В.
3. LabView – Рындина Е.Ю.
4. Delphi – Сафронов А.И., Рындина Е.Ю.

Цель курсового проектирования

Целью курсового проекта является параметрический синтез корректирующего устройства, обеспечивающего устойчивость заданной следящей системы автоматического управления (САУ) и выполнение поставленных требований к динамике САУ [1].

Введение

На рис.1 изображена структурная схема исследуемой в данном курсовом проекте следящей САУ.

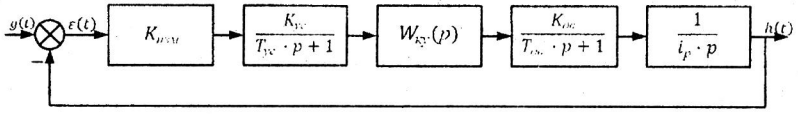

Рисунок 1 – Структурная схема следящей САУ

Рассматриваемая САУ состоит из следующих звеньев:

- Сельсинная пара (СД-СП), работающая в трансформаторном режиме и имеющая статический коэффициент усиления $K_{изм}$.
- Усилитель с передаточной функцией:

$$W_{ус}(p)=\frac{K_{ус}}{T_{ус}\cdot p+1} \qquad (1)$$

где $K_{ус}$ – коэффициент усиления усилителя;

$T_{ус}$ – постоянная времени усилителя.

- Синтезируемое корректирующее устройство с передаточной функцией $W_{ку}(p)$.
- Двигатель с передаточной функцией:

$$W_{дв}(p)=\frac{K_{дв}}{(T_{дв}\cdot p+1)\cdot p} \qquad (2)$$

где $K_{дв}$ – коэффициент усиления усилителя;

$T_{дв}$ – постоянная времени усилителя.

- Редуктор с передаточной функцией:

$$W_p(p) = \frac{1}{i_p} \tag{3}$$

где i_p – передаточное число редуктора.

На вход системы поступает воздействие типа единичная ступень $g(t) = 1(t)$. На выходе получаем $h(t)$ – переходную функцию системы. Следящая система отрабатывает рассогласование между входом и выходом $\varepsilon(t)$.

Под синтезом корректирующего устройства САУ понимается определение структуры и параметров корректирующего устройства в соответствии с заданным критерием качества функционирования. В данном курсовом проекте предъявляются требования по выполнению заданного времени регулирования переходного процесса t_p и перерегулирования σ.

В приложении 1 табл.1. представлены исходные данные для выполнения курсовой работы.

Коэффициент усиления всей системы (общий коэффициент усиления), который будет обеспечивать заданную кинетическую ошибку ε при входном воздействии типа $\omega t \cdot 1(t)$, вычисляется по формуле:

$$K_{общ} = \frac{\omega}{\varepsilon} \tag{4}$$

где ω – скорость изменения входного воздействия;

ε – заданная кинетическая ошибка.

Коэффициент предварительного безынерционного усилителя рассчитывается по формуле:

$$K_{пр.ус} = \frac{K_{общ}}{\left(\frac{K_{изм} \cdot K_{ус} \cdot K_{ов}}{i_p} \right)} \tag{5}$$

Замечание 1

Перед решением задачи параметрического синтеза средствами рассматриваемых далее пакетов прикладных программ *МВТУ*, *Matlab*,

Labview необходимо произвести расчеты «вручную». Для этого надлежит построить логарифмические амплитудно-частотные характеристики (ЛАЧХ) нескорректированной системы $L_{нск}(\omega)$ и желаемой системы $L_{ж}(\omega)$ [2]. При последовательной коррекции ЛАЧХ искомого корректирующего устройства $L_{ку}(\omega)$ находится как разница между ЛАЧХ желаемой и ЛАЧХ нескорректированной системы, т.е. $L_{ку}(\omega) = L_{ж}(\omega) - L_{нск}(\omega)$. При параллельной коррекции для нахождения ЛАЧХ корректирующего устройства $L_{ку}(\omega)$ необходимо из ЛАЧХ нескорректированной системы вычесть ЛАЧХ желаемой системы и ЛАЧХ звена, охватываемого обратной связью с КУ, т.е. $L_{ку}(\omega) = L_{нск}(\omega) - L_{ж}(\omega) - L_{охв}(\omega)$. По точкам излома полученной $L_{ку}(\omega)$ определяется структура и значения параметров корректирующего устройства.

Полученную при «ручном» расчете структуру и параметры корректирующего устройства следует рассматривать как исходные, и использовать в качестве начальной информации при решении поставленной задачи проектирования с помощью программных средств *MBTY*, *Matlab*, *Labview*.

Матрица по которой вычисляются определители Гурвица составляется следующим образом:

1. На главной диагонали записываются все коэффициенты с a_1 до a_n;

2. В каждом столбце выше диагональных элементов записываются коэффициенты с последовательно возрастающими индексами, а ниже – с последовательно убывающими индексами;

3. На место коэффициентов с индексами больше n или меньше нуля проставляются нули.

$$\Delta = \begin{vmatrix} a_1 & a_3 & a_5 & a_7 & a_9 & \dots & 0 \\ a_0 & a_2 & a_4 & a_6 & a_8 & \dots & 0 \\ 0 & a_1 & a_3 & a_5 & a_7 & \dots & 0 \\ 0 & a_0 & a_2 & a_4 & a_6 & \dots & 0 \\ 0 & 0 & a_1 & a_3 & a_5 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots & a_n \end{vmatrix}.$$

4.3.3. Частотный критерий устойчивости Найквиста

Теоретическое описание метода:

Устойчивость ЗАМКНУТОЙ системы гарантирована в том случае, когда все корни характеристического уравнения РАЗОМКНУТОЙ системы лежат в левой полуплоскости комплексной плоскости корней (действительные - вдоль отрицательной части действительной оси, комплексные – попарно симметричные относительно отрицательной части действительной оси), и при этом годограф системы не охватывает точку Найквиста (-1; 0j). Охваты считаются левее точки Найквиста и классифицируются, как показано на рис. 4.6.

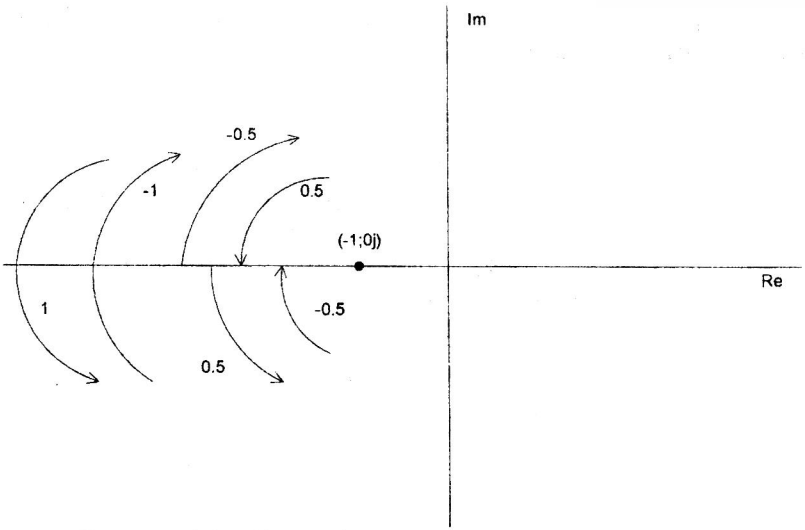

Рисунок 4.6. – Классификация охватов точки Найквиста

Для устойчивости системы с охватами точки Найквиста необходимо наличие у системы корней характеристического уравнения в правой полуплоскости комплексной плоскости корней. Число этих корней должно быть столько, чтобы выполнялось равенство:

$$r = \frac{s}{2}$$

Где r - число охватов точки Найквиста, а s - число правых корней характеристического уравнения.

Для астатических систем, в структуру которых, принципиально входит интегрирующее звено с передаточной функцией вида $W(p) = \frac{K}{p}$ при расчёте устойчивости необходимо дополнять годограф дугой бесконечного большого радиуса, которая будет проходить ν-квадрантов, где ν - степень астатизма системы, равная количеству интегрирующих звеньев.

Так, например, для системы, состоящей из двух инерционных и ν интегрирующих звеньев, годограф Найквиста имеет следующий вид:

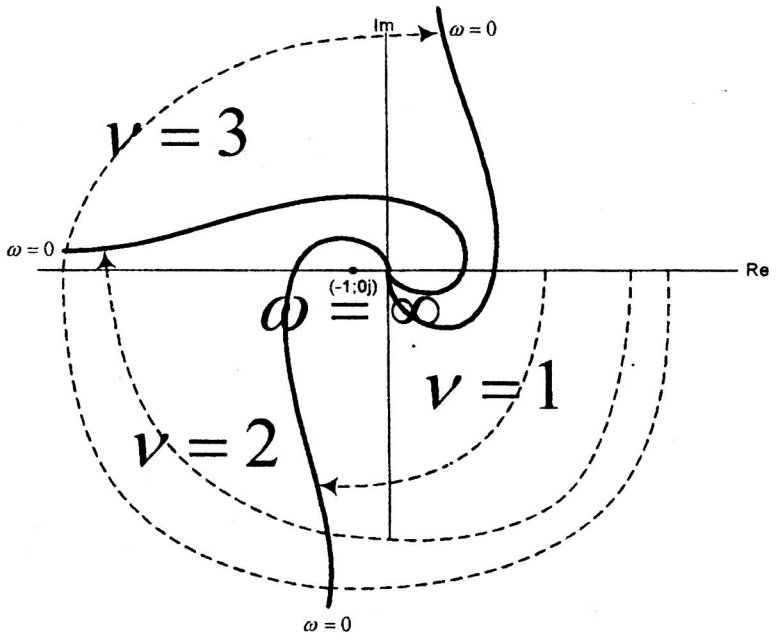

Рисунок 4.7. – Примеры годографов Найквиста с различной степенью астатизма

4.3.4. Критерий устойчивости Михайлова

Теоретическое описание метода:

Устойчивость ЗАМКНУТОЙ системы гарантирована тогда и только тогда, если годограф системы начинается на действительной оси комплексной плоскости и при изменении частоты от нуля до бесконечности последовательно проходит против часовой стрелки n квадрантов, где n – степень характеристического полинома.

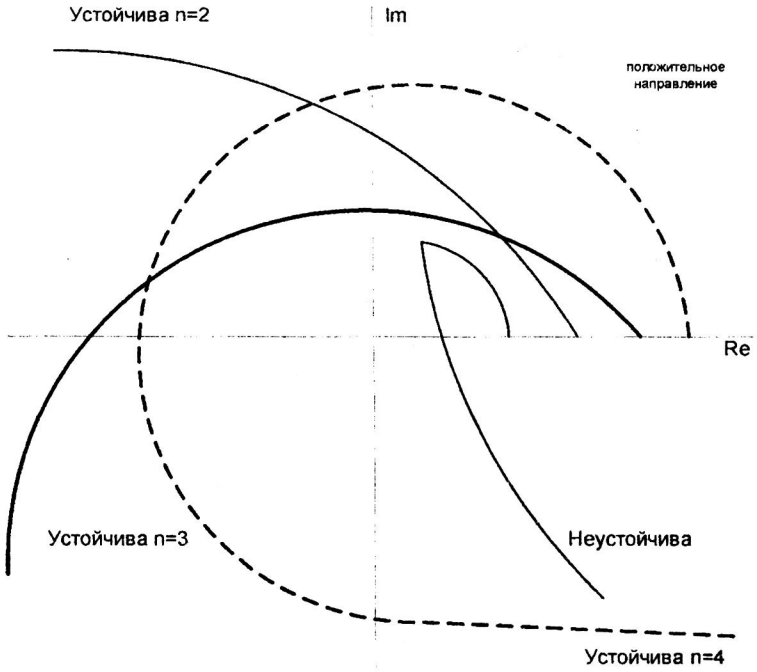

Рисунок 4.8. - Примеры годографов Михайлова с различной степенью характеристического полинома

Пример 1.3:

```
{Проверка устойчивости системы в соответствии с заданным критерием. Критерий задаётся путём установки соответствующей методу опции из rgUst}
procedure TfrmUst.btnUstClick(Sender: TObject);
const
  m0=15;
{Метки безусловного перехода по расчёту метода Рауса}
label
  m1, m2, m3, m4, m5, m6, m7;
var
  {Переменные для Рауса и Гурвица}
  a, r: array [1..m0] of real;
  f: array [1..m0, 1..m0] of real;
  k, n: integer; {Параметры цикла}
  u: integer;    {Число правых корней}
  {Переменные для Найквиста и Михайлова}
  omg, Re, Im, omg_st, ohvat, omg_end: real;
```

```
  step, omg_cr: real;
  signR, signI, chetv, intgr, perehod: integer;
  flgposition, flgintegr, flgpositive: boolean;
  {Общие переменные}
  i, j, l, ll: integer;
begin
  case rgust.ItemIndex of
    0: {Критерий устойчивости Рауса}
      begin

        {Считывание порядка характеристического полинома
         ЗАМКНУТОЙ системы}
        n:=StrToInt(frmPeremnozhenie.eOrd.Text);
        {Считывание порядка характеристического полинома
         ЗАМКНУТОЙ системы}
        {Считывание коэффициентов характеристического полинома
         ЗАМКНУТОЙ системы}
        a[1]:=StrToFloat(frmPeremnozhenie.eh1.Text);
        a[2]:=StrToFloat(frmPeremnozhenie.eh2.Text);
        a[3]:=StrToFloat(frmPeremnozhenie.eh3.Text);
        a[4]:=StrToFloat(frmPeremnozhenie.eh4.Text);
        a[5]:=StrToFloat(frmPeremnozhenie.eh5.Text);
        a[6]:=StrToFloat(frmPeremnozhenie.eh6.Text);
        a[7]:=StrToFloat(frmPeremnozhenie.eh7.Text);
        {Считывание коэффициентов характеристического полинома
         ЗАМКНУТОЙ системы}
        if a[1]<=0 then
          for i:=1 to n do
            a[i]:=-a[i];
        k:=0;
        for i:=1 to n+1 do
          begin
            f[1,i]:=a[i+k];
            k:=k+1;
            if (i+k)>(n+1) then
              goto m1;
          end;
        m1:
        k:=0;
        for i:=1 to n+1 do
          begin
            f[2,i]:=a[i+k+1];
            k:=k+1;
            if (i+k+1)>(n+1) then
              goto m2;
          end;
        m2:
        if n>2 then
          goto m3
        else
          for i:=1 to n+1 do
            if a[i]<0 then goto m5;
        if n<3 then goto m4;
```

```
    m3:
    for i:=3 to n+2 do
      begin
        if f[i-1,1]=0 then
          goto m5
      else
        begin
          r[i-2]:=f[i-2,1]/f[i-1,1];
          for j:=1 to n do
            f[i,j]:=f[i-2,j+1]-r[i-2]*f[i-1,j+1];
        end;
    end;
    u:=0;
    for i:=1 to n do
      if r[i]<0 then
        u:=u+1;
      if u>0 then
    goto m7
    else
    m4:
    eUst.Text:='----------';
    chkUst.Checked:=True;
    btnOpt.Enabled:=False;
    btnPerehod.Enabled:=True;
    btnReturn.Enabled:=False;
    goto m6;
    m7:
    if u = 1 then
      eUst.Text:=IntToStr(u) + ' правый корень'
    else if (u > 1) and (u < 5) then
      eUst.Text:=IntToStr(u) + ' правых корня'
        else
          eUst.Text:=IntToStr(u) + ' правых корней';
    chkUst.Checked:=False;
    btnOpt.Enabled:=True;
    btnReturn.Enabled:=True;
    btnPerehod.Enabled:=False;
    goto m6;
    m5:
    eUst.Text:='----------';
    chkUst.Checked:=False;
    btnOpt.Enabled:=True;
    btnReturn.Enabled:=True;
    btnPerehod.Enabled:=False;
    m6:
  end;
1: {Критерий устойчивости Найквиста}
  begin
    {Считывание порядка характеристического полинома
     РАЗОМКНУТОЙ системы}
    n:=StrToInt(frmPeremnozhenie.eOrd.Text);
    {Считывание порядка характеристического полинома
     РАЗОМКНУТОЙ системы}
```

```
{Считывание коэффициентов характеристического полинома
 РАЗОМКНУТОЙ системы}
a[1]:=StrToFloat(frmPeremnozhenie.e1.Text);
a[2]:=StrToFloat(frmPeremnozhenie.e2.Text);
a[3]:=StrToFloat(frmPeremnozhenie.e3.Text);
a[4]:=StrToFloat(frmPeremnozhenie.e4.Text);
a[5]:=StrToFloat(frmPeremnozhenie.e5.Text);
a[6]:=StrToFloat(frmPeremnozhenie.e6.Text);
a[7]:=StrToFloat(frmPeremnozhenie.e7.Text);
{Считывание коэффициентов характеристического полинома
 РАЗОМКНУТОЙ системы}
step:=0.1;        {Расчётный шаг частоты}
omg:=step;        {Ненулевое начальное значение частоты}
omg_end:=1000;    {Правая граница частотного диапазона}
chetv:=0;         {Координатная четверть не определена}
intgr:=0;         {Интегрирующие звенья отсутствуют}
ohvat:=0;         {Охватов годографом точки Найквиста нет}
flgposition:=false; {Начальная координатная четверть не
                     определена}
flgintegr:=false;   {Флаг счёта интегрирующих звеньев}
flgpositive:=false; {Флаг положительности коэффициентов}
{Критерий Стодолы}
j:=0;
for i:=1 to n+1 do
  if a[i]<0 then
    j:=j+1;
if j=0 then
  flgpositive:=true;
{Критерий Стодолы}

if not flgpositive then
  begin
   {Критерий Рауса для подсчёта правых корней}
    if a[1]<=0 then
      for i:=1 to n do
        a[i]:=-a[i];
    k:=0;
    for i:=1 to n+1 do
      begin
        f[1,i]:=a[i+k];
        k:=k+1;
        if (i+k)>(n+1) then
          begin
            k:=0;
            for j:=1 to n+1 do
              begin
                f[2,j]:=a[j+k+1];
                k:=k+1;
                if (j+k+1)>(n+1) then
                  begin
                    if n>2 then
                      begin
                        for l:=3 to n+2 do
```

```
                        begin
                          if not (f[l-1,1] = 0) then
                            begin
                              r[l-2]:=f[l-2,1]/
                                          f[l-1,1];
                              for l1:=1 to n do
                                f[l,l1]:=f[l-2,l1+1]-
                                    r[l-2]*f[l-1,l1+1];
                            end;
                        end;
                    end;
                end;
            end;
        end;
      end;
    u:=0;
    for i:=1 to n do
    if r[i]<0 then
      u:=u+1;
   {Критерий Рауса для подсчёта правых корней}
  end
else
  u:=0;
{Считывание коэффициентов характеристического полинома
 РАЗОМКНУТОЙ системы}
a[1]:=StrToFloat(frmPeremnozhenie.e1.Text);
a[2]:=StrToFloat(frmPeremnozhenie.e2.Text);
a[3]:=StrToFloat(frmPeremnozhenie.e3.Text);
a[4]:=StrToFloat(frmPeremnozhenie.e4.Text);
a[5]:=StrToFloat(frmPeremnozhenie.e5.Text);
a[6]:=StrToFloat(frmPeremnozhenie.e6.Text);
a[7]:=StrToFloat(frmPeremnozhenie.e7.Text);
{Считывание коэффициентов характеристического полинома
 РАЗОМКНУТОЙ системы}
{Поиск интегрирующих звеньев}
for i:=1 to n+1 do
  begin
    if (a[i] = 0) and (not flgintegr) then
      intgr:=intgr+1
    else
      flgintegr:=true;
    if a[i] > 1 then
      intgr:=intgr+1;
  end;
{Поиск интегрирующих звеньев}
{Просмотр поведения годографа в частотном диапазоне omg
 - omg_end}
while omg <= omg_end do
  begin
    Im:=0;     {Сброс значения мнимой части}
    Re:=0;     {Сброс значения действительной части}
    signI:=1; {Сброс знака знакопеременного ряда мнимой
               части}
```

```
signR:=1; {Сброс знака знакопеременного ряда
           действительной части}
for i:=1 to n+1 do
  if i mod 2 = 0 then
  {Расчёт мнимой части}
    begin
      {Возведение частоты в нужную степень}
      omg_st:=1;
      for j:=1 to i-1 do
        omg_st:=omg_st*omg;
      {Возведение частоты в нужную степень}
      Im:=Im+signI*a[i]*omg_st;
      signI:=-1*signI;
    end
  {Расчёт мнимой части}
  else
  {Расчёт действительной части}
    begin
      {Возведение частоты в нужную степень}
      omg_st:=1;
      for j:=1 to i-1 do
        omg_st:=omg_st*omg;
      {Возведение частоты в нужную степень}
      if i = 1 then
        Re:=Re+signR*a[i]
      else
        Re:=Re+signR*a[i]*omg_st;
      signR:=-1*signR;
  {Расчёт действительной части}
    end;
{Если начальная координатная четверть не определена,
 то определяем}
if not flgposition then
  if (Re > 0) and (Im > 0) then
    chetv:=1
  else if (Re < 0) and (Im > 0) then
         chetv:=2
       else if (Re < 0 ) and (Im < 0) then
              chetv:=3
            else if (Re > 0) and (Im <0) then
                   chetv:=4;
{Если начальная координатная четверть не определена,
 то определяем}
{Если предыдущая точка была в...}
case chetv of
  1: {...первой координатной четверти...}
    begin
      {...и действительнпя часть стала
       отрицательной, то произошёл
       переход во вторую четверть}
      if (Re < 0) then
        begin
          chetv:=2;
```

```
      end;
    {...и мнимая часть стала отрицательной, то
     Произошёл переход в четвёртую четверть}
    if (Im < 0) then
      begin
        chetv:=4;
      end;
    if (omg+step > omg_end) and
       (not (intgr = 0)) then
      case intgr of
        2: ohvat:=ohvat-0.5;
        3,4: ohvat:=ohvat-1;
      end;
  end;
2: {...второй координатной четверти...}
  begin
    {...и действительная часть стала
     положительной, то произошёл
     переход в первую четверть}
    if (Re > 0) then
      begin
        chetv:=1;
      end;
    {...и мнимая часть стала отрицательной, то
     произошёл переход в третью четверть...}
    if (Im < 0) then
      begin
        {...если в этот момент действительная
         часть меньше -1, то считаем охват точки
         Найквиста +1}
        if Re < -1 then
          ohvat:=ohvat+1;
        chetv:=3;
      end;
    if (omg+step > omg_end)
        and (not (intgr = 0)) then
      case intgr of
        1: ohvat:=ohvat-0.5;
        2,3,4: ohvat:=ohvat-1;
      end;
  end;
3: {...третьей координатной четверти...}
  begin
    {...и действительная часть стала
     положительной, то произошёл
     переход в четвёртую четверть}
    if (Re > 0) then
      begin
        chetv:=4;
      end;
    {...и мнимая часть стала положительной, то
     произошёл переход во вторую четверть...}
    if (Im > 0) then
```

```
            begin
              {...если в этот момент действительная
               часть меньше -1, то считаем охват точки
               Найквиста -1}
              if Re < -1 then
                ohvat:=ohvat-1;
              chetv:=2;
            end;
          if (omg+step > omg_end)
              and (not (intgr = 0)) then
            case intgr of
              4: ohvat:=ohvat-0.5;
            end;
        end;
      4: {...четвёртой координатной четверти...}
        begin
          {...и действительная часть стала
           отрицательной, то произошёл
           переход в третью четверть}
          if (Re < 0) then
            begin
              chetv:=3;
            end;
          {...и мнимая часть стала положительной, то
           Произошёл переход в первую четверть}
          if (Im > 0) then
            begin
              chetv:=1;
            end;
          if (omg+step > omg_end)
              and (not (intgr = 0)) then
            case intgr of
              3: ohvat:=ohvat-0.5;
              4: ohvat:=ohvat-1;
            end;
        end;
    end;
    {Снимаем флаг неопределённого начального квадранта}
    if not flgposition then
      flgposition:=true;
    {Снимаем флаг неопределённого начального квадранта}
    omg:=omg+step; {Переходим к следующему значению
                    частоты}
  end;
 {Просмотр поведения годографа в частотном диапазоне omg
  - omg_end}

{Принятие решения об устойчивости системы}
if not (ohvat = u/2) then
  begin
    chkUst.Checked:=False;
    btnOpt.Enabled:=True;
    btnPerehod.Enabled:=False;
```

```
      btnReturn.Enabled:=True;
      eUst.Text:=FloatToStrF(ohvat, ffFixed, 7, 2) +
                 ' охватов точки H., правых корней: ' +
                  IntToStr(u);
    end
  else
    begin
      chkUst.Checked:=True;
      btnOpt.Enabled:=False;
      btnPerehod.Enabled:=True;
      btnReturn.Enabled:=False;
      eUst.Text:=FloatToStrF(ohvat, ffFixed, 7, 2) +
                 ' охватов точки H., правых корней: ' +
                 IntToStr(u);
    end;
  {Принятие решения об устойчивости системы}
 end;
2: {Критерий устойчивости Гурвица}
  begin

    {Считывание порядка характеристического полинома
     ЗАМКНУТОЙ системы}
    n:=StrToInt(frmPeremnozhenie.eOrd.Text);
    {Считывание порядка характеристического полинома
     ЗАМКНУТОЙ системы}

    {Считывание коэффициентов характеристического полинома
     ЗАМКНУТОЙ системы}
    a[1]:=StrToFloat(frmPeremnozhenie.eh1.Text);
    a[2]:=StrToFloat(frmPeremnozhenie.eh2.Text);
    a[3]:=StrToFloat(frmPeremnozhenie.eh3.Text);
    a[4]:=StrToFloat(frmPeremnozhenie.eh4.Text);
    a[5]:=StrToFloat(frmPeremnozhenie.eh5.Text);
    a[6]:=StrToFloat(frmPeremnozhenie.eh6.Text);
    a[7]:=StrToFloat(frmPeremnozhenie.eh7.Text);
    {Считывание коэффициентов характеристического полинома
     ЗАМКНУТОЙ системы}
    {Критерий Стодолы}
    j:=0;
    for i:=1 to n+1 do
      if a[i]<0 then
        j:=j+1;
    if j<>0 then
      begin
        chkUst.Checked:=False;
        btnReturn.Enabled:=True;
        btnPerehod.Enabled:=False;
        eUst.Text:='----------';
      end
    else if j=0 then
      chkUst.Checked:=True;
    {Критерий Стодолы}
    {Если критерий Стодолы выполняется, то проверяем по
```

```
 матрице Гурвица}
if chkUst.Checked = True then
  begin
    {В зависимости от порядка системы}
    case n of
      1: {1-го порядка}
        j:=0; {всегда устойчива}
      2: {2-го порядка}
        j:=0; {всегда устойчива}
      3: {3-го порядка}
        if (a[2]*a[3]-a[1]*a[4]) > 0 then
          j:=0 {если определитель матрицы положителен -
                устойчивость}
        else
          j:=1; {если определитель матрицы отрицателен
                 или равен 0 (граница устойчивости) -
                 неустойчивость}
      4: {4-го порядка}
        if (a[4]*(a[2]*a[3]-a[1]*a[4])-a[5]*a[2]*a[2])
            > 0 then
          j:=0 {если определитель матрицы положителен -
                устойчивость}
        else
          j:=1; {если определитель матрицы отрицателен
                 или равен 0 (граница устойчивости) -
                 неустойчивость}
      5: {5-го порядка}
        if ((a[2]*a[3]-a[1]*a[4]) > 0)
            and (((a[2]*a[3]-a[1]*a[4])*
           (a[4]*a[5]-a[3]*a[6])-(a[2]*a[5]-a[1]*a[6])*
           (a[2]*a[5]-a[1]*a[6])) > 0) then
          j:=0  {если определитель матрицы положителен -
                 устойчивость}
        else
          j:=1; {если определитель матрицы отрицателен
                 или равен 0 (граница устойчивости) -
                 неустойчивость}
      6: {6-го порядка}
        if ((a[5]*a[6]-a[4]*a[7])*
            (a[2]*a[3]*a[4]-a[2]*a[2]*a[5]-
            a[4]*a[4]*a[1]+a[6]*a[1]*a[2])+
           (a[3]*a[6]-a[2]*a[7])*
           (a[2]*a[2]*a[7]+a[1]*a[4]*a[6]-
            a[2]*a[3]*a[6])+a[1]*a[6]*
           (a[2]*a[5]*a[6]-a[2]*a[4]*a[7]-
            a[6]*a[6]*a[1])) > 0 then
          j:=0  {если определитель матрицы положителен -
                 устойчивость}
        else
          j:=1; {если определитель матрицы отрицателен
                 или равен 0 (граница устойчивости) -
                 неустойчивость}
    end;
```

```
        {Принятие решения об устойчивости системы}
        if j=1 then
          begin
            chkUst.Checked:=False;
            btnOpt.Enabled:=True;
            btnReturn.Enabled:=True;
            btnPerehod.Enabled:=False;
            eUst.Text:='----------';
          end
        else
          begin
            btnPerehod.Enabled:=True;
            btnOpt.Enabled:=False;
            btnReturn.Enabled:=False;
            eUst.Text:='----------';
          end;
        {Принятие решения об устойчивости системы}
      end;
  end;
3: {Критерий устойчивости Михайлова}
  begin
    {Считывание порядка характеристического полинома
     ЗАМКНУТОЙ системы}
    n:=StrToInt(frmPeremnozhenie.eOrd.Text);
    {Считывание порядка характеристического полинома
     ЗАМКНУТОЙ системы}
    {Считывание коэффициентов характеристического полинома
     ЗАМКНУТОЙ системы}
    a[1]:=StrToFloat(frmPeremnozhenie.eh1.Text);
    a[2]:=StrToFloat(frmPeremnozhenie.eh2.Text);
    a[3]:=StrToFloat(frmPeremnozhenie.eh3.Text);
    a[4]:=StrToFloat(frmPeremnozhenie.eh4.Text);
    a[5]:=StrToFloat(frmPeremnozhenie.eh5.Text);
    a[6]:=StrToFloat(frmPeremnozhenie.eh6.Text);
    a[7]:=StrToFloat(frmPeremnozhenie.eh7.Text);
    {Считывание коэффициентов характеристического полинома
     ЗАМКНУТОЙ системы}
    {Начальные уставки работы критерия устойчивости}
    step:=0.1;      {Расчётный шаг частоты}
    omg:=step;      {Ненулевое начальное значение частоты}
    omg_end:=1000; {Правая граница частотного диапазона}
    chetv:=0;       {Координатная четверть не определена}
    perehod:=0;     {Сброс счётчика числа переходов}
    flgposition:=false; {Начальная координатная четверть не
                         определена}
    {Начальные уставки работы критерия устойчивости}
    {Просмотр поведения годографа в частотном диапазоне omg
     - omg_end}
    while omg <= omg_end do
      begin
        Im:=0;     {Сброс значения мнимой части}
        Re:=0;     {Сброс значения действительной части}
        signI:=1; {Сброс знака знакопеременного ряда мнимой
```

```
            части}
signR:=1; {Сброс знака знакопеременного ряда
           действительной части}
for i:=1 to n+1 do
  if i mod 2 = 0 then
  {Расчёт мнимой части}
    begin
      {Возведение частоты в нужную степень}
      omg_st:=1;
      for j:=1 to i-1 do
        omg_st:=omg_st*omg;
      {Возведение частоты в нужную степень}
      Im:=Im+signI*a[i]*omg_st;
      signI:=-1*signI;
    end
  {Расчёт мнимой части}
  else
  {Расчёт действительной части}
    begin
      {Возведение частоты в нужную степень}
      omg_st:=1;
      for j:=1 to i-1 do
        omg_st:=omg_st*omg;
      {Возведение частоты в нужную степень}
      if i = 1 then
        Re:=Re+signR*a[i]
      else
        Re:=Re+signR*a[i]*omg_st;
      signR:=-1*signR;
  {Расчёт действительной части}
    end;
{Если начальная координатная четверть не определена,
 то определяем}
if not flgposition then
  if (Re > 0) and (Im > 0) then
    begin
      {Годограф начинается в первой четверти,
       начинаем учёт переходов}
      chetv:=1;
      perehod:=perehod+1;
    end
  else if (Re < 0) and (Im > 0) then
         chetv:=2
       else if (Re < 0 ) and (Im < 0) then
              chetv:=3
            else if (Re > 0) and (Im <0) then
                   chetv:=4;
{Если начальная координатная четверть не определена,
 то определяем}
{Если предыдущая точка была в...}
case chetv of
  1: {...первой координатной четверти...}
    begin
```

```
    {...и действительнпя часть стала
     отрицательной, то произошёл
     переход во вторую четверть...}
    if (Re < 0) then
      begin
        {...в положительном направлении, переход
         учитываем}
        chetv:=2;
        perehod:=perehod+1;
      end;
    {...и мнимая часть стала отрицательной, то
     произошёл  переход в четвёртую четверть...}
    if (Im < 0) then
      {...в отрицательном направлении, переход не
       учитываем}
      chetv:=4;
  end;
2: {...второй координатной четверти...}
  begin
    {...и действительная часть стала
     положительной, то произошёл
     переход в первую четверть...}
    if (Re > 0) then
      {...в отрицательном направлении, переход не
       учитываем}
      chetv:=1;
    {...и мнимая часть стала отрицательной, то
     произошёл переход в третью четверть}
    if (Im < 0) then
      begin
        {...в положительном направлении, переход
         учитываем}
        chetv:=3;
        perehod:=perehod+1;
      end;
  end;
3: {...третьей координатной четверти...}
  begin
    {...и действительная часть стала
     положительной, то произошёл
     переход в четвёртую четверть...}
    if (Re > 0) then
      begin
        {...в положительном направлении, переход
         учитываем}
        chetv:=4;
        perehod:=perehod+1;
      end;
    {...и мнимая часть стала положительной, то
     произошёл переход во вторую четверть...}
    if (Im > 0) then
      {...в отрицательном направлении, переход не
       учитываем}
```

```
            chetv:=2;
          end;
        4: {...четвёртой координатной четверти...}
          begin
            {...и действительная часть стала
             отрицательной, то произошёл
             переход в третью четверть...}
            if (Re < 0) then
              {...в отрицательном направлении, переход не
               учитываем}
              chetv:=3;
            {...и мнимая часть стала положительной, то
             произошёл переход в первую четверть...}
            if (Im > 0) then
              begin
                {...в положительном направлении, переход
                 учитываем}
                chetv:=1;
                perehod:=perehod+1;
              end;
          end;
      end;
      {Снимаем флаг неопределённого начального квадранта}
      if not flgposition then
        flgposition:=true;
      {Снимаем флаг неопределённого начального квадранта}
      omg:=omg+step; {Переходим к следующему значению
                      частоты}
    end;
  {Просмотр поведения годографа в частотном диапазоне omg
   - omg_end}
  {Принятие решения об устойчивости системы}
  if not (n = perehod) then
    begin
      {Если неустойчива по Михайлову}
      chkUst.Checked:=False;
      btnOpt.Enabled:=True;
      btnReturn.Enabled:=True;
      btnPerehod.Enabled:=False;
      if perehod = 1 then
        eUst.Text:=IntToStr(perehod) +
                   ' квадрант в положит. направ.'
      else if (perehod > 1) and (perehod < 5) then
             eUst.Text:=IntToStr(perehod) +
                        ' квадранта в положит. направ.'
           else
             eUst.Text:=IntToStr(perehod) +
                        ' квадрантов в положит. направ.';
      {Если неустойчива по Михайлову}
    end
  else
    begin
      {Если устойчива по Михайлову}
```

```
            chkUst.Checked:=True;
            btnOpt.Enabled:=False;
            btnPerehod.Enabled:=True;
            btnReturn.Enabled:=False;
            if perehod = 1 then
              eUst.Text:=IntToStr(perehod) +
                         ' квадрант в положит. направ.'
            else if (perehod > 1) and (perehod < 5) then
                   eUst.Text:=IntToStr(perehod) +
                              ' квадранта в положит. направ.'
                 else
                   eUst.Text:=IntToStr(perehod) +
                              ' квадрантов в положит. направ.';
            {Если устойчива по Михайлову}
          end;
        end;
        {Принятие решения об устойчивости системы}
  end;
end;
```

4.4. Расчёт переходного процесса:

Студентам, для выполнения курсового проекта, предлагается использовать программу расчёта переходного процесса «PEREHOD.BAS», составленную Сеславиным А.И. на языке Basic.

Применение программы заключается, как и в случае с перемножением многочленов, не в непосредственном её использовании, а в изучении алгоритма, который должен быть внедрён в общую структуру программы, составляемой на языке Pascal или Delphi. Такой метод перевода программ с одного языка на другой получил название «трансляция». Таким образом, студентам предлагается транслировать программу «PEREHOD.BAS» для дальнейшего её использования, а также для отработки навыков работы с различными языками программирования.

Замечание 1.3: неправильный выбор шага интегрирования зачастую приводит к расхождению процесса численного интегрирования, посему авторами рекомендуется выбирать шаг в десять раз меньше самой маленькой постоянной времени

4.4.1. Этап приведения к дифференциальным уравнениям нормальной формы Коши по схеме Горнера

Схема Горнера – алгоритм вычисления значения многочлена, записанного в виде суммы вида $a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + ... + a_0$ при заданном значении переменной. Схема Горнера позволяет найти корни многочлена, а также вычислить производные полинома в заданной точке.

Пусть задан многочлен $W(p)$:

$$W(p) = a_n \cdot p^n + a_{n-1} \cdot p^{n-1} + ... + a_0, a_i \in R$$

Требуется вычислить значение данного многочлена при фиксированном $p = p_0$. Преобразуем многочлен $W(p)$ к виду:

$$W(p) = ((...(a_n \cdot p + a_{n-1}) \cdot p ... + a_2) \cdot p + a_1) \cdot p + a_0$$

Определим следующую последовательность:

$$b_n = a_n;$$
$$b_{n-1} = a_{n-1} + b_n \cdot p;$$
$$...$$
$$b_i = a_i + b_{i+1} \cdot p;$$
$$...$$
$$b_0 = a_0 + b_1 \cdot p.$$

Искомое значение $W(p_0) = b_0$. Покажем, что это так:

В полученную форму записи $W(p)$ подставим $p = p_0$ и будем вычислять значение выражения, начиная со внутренних скобок. Для этого будем заменять подвыражения через b_i.

$$W(p_0) = ((...(a_n \cdot p_0 + a_{n-1}) \cdot p_0 ... + a_2) \cdot p_0 + a_1) \cdot p_0 + a_0;$$
$$W(p_0) = ((...(b_n \cdot p_0 + a_{n-1}) \cdot p_0 ... + a_2) \cdot p_0 + a_1) \cdot p_0 + a_0;$$
$$W(p_0) = ((...(b_{n-1}) \cdot p_0 ... + a_2) \cdot p_0 + a_1) \cdot p_0 + a_0;$$
$$...$$
$$W(p_0) = (b_1) \cdot p_0 + a_0;$$
$$W(p_0) = b_0.$$

4.4.2. Численный метод Эйлера для решения дифференциальных уравнений

Метод Эйлера (также известен под названием «метод ломаных», так как участки кривой функции заменяются отрезками прямых) является простейшим приближённым способом решения дифференциальных уравнений. Рассмотрим его геометрическую интерпретацию на примере одного уравнения:

$$\frac{dy}{dx} = f(x, y)$$

Требуется проинтегрировать уравнение на интервале $[a, b]$, то есть найти профиль изменения функции y для заданного диапазона значений аргумента x, как показано на рис. 4.9.

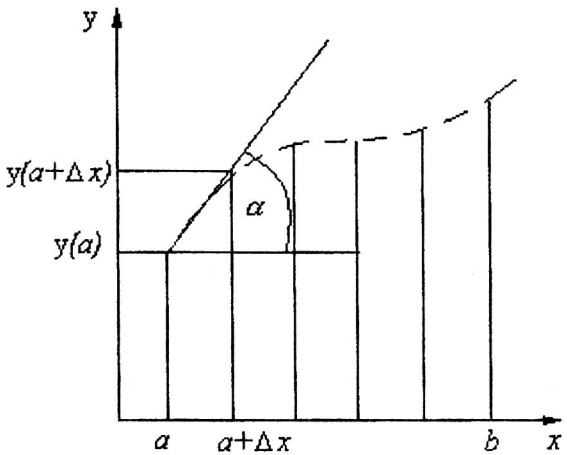

Рисунок 4.9. – Иллюстрация метода Эйлера

Разделим интервал $[a, b]$ на равные малые отрезки Δx, называемые шагом интегрирования. На рис. 4.9. пунктиром обозначена неизвестная функция y, которую требуется найти. Мысленно проведем в точке касательную к функции. Приближенным значением функции y в точке с координатой $a + \Delta x$, то есть на конце первого

отрезка будем считать точку пересечения касательной и перпендикуляра, восстановленного в этой точке. Тогда

$$y(a+\Delta x) = y(a) + \Delta x \cdot tg\alpha$$

Угол α – это угол наклона касательной, проведенной в точке a. Из геометрического смысла первой производной следует, что тангенс угла наклона касательной, проведенной в некоторой точке, численно равен первой производной, вычисленной в этой точке, то есть:

$$tg\alpha = \left.\frac{dy}{dx}\right|_{x=a}$$

Формула для приближенного значения функции на конце отрезка принимает вид:

$$y(a+\Delta x) = y_{i-1} + \Delta x \cdot \left.\frac{dy}{dx}\right|_{x=x_{i-1}}$$

Вычислив значение функции после первого шага интегрирования, применим выведенную формулу для второго шага и т.д. Для произвольного шага i формула Эйлера принимает вид:

$$y_i = y_{i-1} + \Delta x \left.\frac{dy}{dx}\right|_{x=x_{i-1}}$$

Таким образом, приближенное значение функции на конце текущего отрезка интегрирования равно сумме значения функции в начале отрезка и произведения шага интегрирования на величину производной, вычисленной в начале отрезка. Для первого шага должно быть известно значение функции в начале всего интервала интегрирования (в точке a), это значение называют начальным условием.

Для решения системы из n дифференциальных уравнений формулу Эйлера на каждом шаге интегрирования применяют для каждого из n уравнений.

Метод Эйлера достаточно прост для реализации, поэтому его часто программируют непосредственно в прикладной программе.

4.4.3. Численный метод Рунге-Кутты (4 точки) для решения дифференциальных уравнений

Система обыкновенных диференциальных уравнений высшего порядка путем введения новых неизвестных может быть сведена к системе уравнений первого порядка. Рассмотрим такую систему:

$$y_i' = F_i(x, y),$$

где $i = 1,...,n$;

$y = (y_1, y_2, ..., y_n)$;

$y(x_0) = y_0 = (y_{01}, ..., y_{0n})$.

В методе Рунге-Кутты $y_{k,i}$ вычисляет по формуле:

$$y_{k+1,i} = y_{k,i} + \frac{(k_{i,1} + 2 \cdot k_{i,2} + 2 \cdot k_{i,3} + k_{i,4})}{6},$$

где

$k_{i,1} = F_i(x_k, y_k) \cdot h$;

$k_{i,2} = F_i(x_k + \frac{h}{2}, y_k + \frac{k_{i,1}}{2}) \cdot h$;

$k_{i,3} = F_i(x_k + \frac{h}{2}, y_k + \frac{k_{i,2}}{2}) \cdot h$;

$k_{i,4} = F_i(x_k + h, y_k + k_{i,3}) \cdot h$;

$h = \frac{x_n - x_0}{m}$;

m - количество шагов интегрирования.

Процедура использует набор функций $F(i,x,y)$, которые соответствуют функциям $F_i(x,y)$ описанным выше.

Пример 1.4:

```
procedure TfrmPerehod.btnRunClick(Sender: TObject);
const
  m0=15;
const
  u=1; {входной сигнал - единичная ступенька}

var
  step: real;           {Шаг интегрирования}
  t_intgr, t_reg: real;{Времена: интегрирования и регулирования}
  y_max, ymin, y_const: real; {Важные значения переходной
                  характеристики: максимальное и установившееся}
  y, t, tmax, tmin: extended;
  k1, k2, k3, k4: real;     {Расчётные коэффициенты 4-х точечного
                             метода Рунге-Кутты}
  a: array [1..m0] of real;     {Массив коэффициентов числителя}
  b: array [1..m0] of real;   {Массив коэффициентов знаменателя}
  g: array [1..m0] of extended;     {Массив функций при p при
                  расписывании диф. уравнений по схеме Горнера}
  F: array [1..m0] of extended; {Массив функций для вычисления
 правых частей системы диф. уравнений по схеме Горнера}
  i, j, n, l: longint;
  flgforce, flgsettle: boolean;
  ft: TextFile;
begin
  flgsettle:=false;
  AssignFile(ft, 'temp.1');
  ReWrite(ft);
  n:=StrToInt(frmPeremnozhenie.eOrd.Text);   {Порядок   полинома
знаменателя}
  a[1]:=StrToFloat(frmPeremnozhenie.ekobsh.Text)*
        StrToFloat(frmPeremnozhenie.ea1.Text);
  a[2]:=StrToFloat(frmPeremnozhenie.ekobsh.Text)*
        StrToFloat(frmPeremnozhenie.ea2.Text);
  a[3]:=StrToFloat(frmPeremnozhenie.ekobsh.Text)*
        StrToFloat(frmPeremnozhenie.ea3.Text);
  b[1]:=StrToFloat(frmPeremnozhenie.eh1.Text);
  b[2]:=StrToFloat(frmPeremnozhenie.eh2.Text);
  b[3]:=StrToFloat(frmPeremnozhenie.eh3.Text);
  b[4]:=StrToFloat(frmPeremnozhenie.eh4.Text);
  b[5]:=StrToFloat(frmPeremnozhenie.eh5.Text);
  b[6]:=StrToFloat(frmPeremnozhenie.eh6.Text);
  b[7]:=StrToFloat(frmPeremnozhenie.eh7.Text);
  step:=StrToFloat(estep.Text);
  t_intgr:=StrToFloat(etintgr.Text);
  y_max:=0;              {Максимальное значение не определено}
  y_const:=a[1]/b[1];    {Установившееся значение}
```

```
t_reg:=0;                {Время регулирования не определено}
flgsettle:=false;
tmin:=1;
t:=0;
y:=0;
writeln(ft, t:0:10, '    ',y:0:10);
for i := 1 to n do
  begin
    g[i]:=0;
    F[i]:=0;
  end;
l:=0;
repeat
    if n=1 then
      F[1]:=a[1]*u-b[1]*(g[n]+a[n+1]*u)/b[n+1]
    else
      begin
        F[1]:=a[1]*u-b[1]*(g[n]+a[n+1]*u)/b[n+1];
        for i := 2 to n do
          F[i]:=g[i-1]+a[i]*u-b[i]*(g[n]+a[n+1]*u)/b[n+1];
      end;
    case rgMethod.ItemIndex  of
      0:
        begin
          for i := 2 to n do
            g[i]:=g[i]+step*F[i];
        end;
      1:
        begin
          for i := 1 to n do
            begin
              k1:=F[i]*step;
              k2:=(F[i]+0.5*k1)*step;
              k3:=(F[i]+0.5*k2)*step;
              k4:=(F[i]+k3)*step;
              g[i]:=g[i]+(k1+2*k2+2*k3+k4)/6;
            end;
        end;
      end;
      y:=(g[n]+a[n+1]*u)/b[n+1];
      t:=t+step;
      writeln(ft, t:0:10, '    ',y:0:10);
      if y{[j]} > y_max then
        begin
          y_max:=y{[j]};
          tmax:=t;
        end;
      if t>tmax+0.005 then
        if y<1 then
          tmin:=t;
      if (t>tmin+0.005) and ((y{[j]} > 0.95*y_const)
         and (y{[j]} < 1.05*y_const)) and
         (not flgsettle) then
```

```
          begin
            l:=l+1;
            if l=1 then
              begin
                t_reg:=t{[j]};
                ymin:=y;
              end;
            if step < 1 then
              if {l >= int(0.5/step)} y>ymin then
                flgsettle:=true
            else
              if l >= 3 then
                flgsettle:=true;
          end
        else
          l:=0;
        j:=j+1;
  until t{[j-1]} >= t_intgr;
  CloseFile(ft);
  y_max:=(y_max-y_const)*100/y_const;
  eovershoot.Text:=FloatToStrF(y_max, ffFixed, 7, 7);
  etreg.Text:=FloatToStrF(t_reg, ffFixed, 7, 7);
  btnRun.Enabled:=False;
  btnPrint.Enabled:=True;
end;
```

4.5. Оптимизация параметров методами НЛП

Задача обеспечения требуемого качества переходных процессов в проектируемой САУ формулируется как задача нелинейного программирования (НЛП), то есть при заданном математическом описании объекта в виде передаточных функций либо дифференциальных уравнений определить параметры x_i корректирующего устройства заданной структуры, обеспечивающих минимум критерия оптимальности, который может быть сформирован, например, в виде:

$$Q(x_1,...,x_n) = \alpha_1(\sigma^* - \sigma(x_1,...,x_n))^2 + \alpha_2(t_p^* - t_p(x_1,...,x_n))^2,$$

где α_1, α_2 - весовые коэффициенты; σ^*, t_p^* - требуемые показатели качества динамики САУ; $\sigma(x_1,...,x_n), t_p(x_1,...,x_n)$ - значение тех же показателей, получаемых в результате вариации параметров корректирующего устройства x_i.

4.5.1. Случайный метод с возвратом при неудачном шаге. Метод наилучшей пробы

Теоретическое описание метода:

Рассматривается следующая многомерная задача локальной безусловной оптимизации: найти минимум критерия оптимальности $Q(x)$ определенного в n-мерном евклидовом пространстве R^n.

$$\min_{x \in R^n} Q(x) = Q(x^*) = Q^*$$

При решении такой задачи методом с возвратом при неудачном шаге (одношаговый метод оптимизации) используется итерационная формула:

$$x^{r+1} = x^r + h^r \cdot E^r$$

где h^r - величина шага на r-ой итерации, E^r - реализация n-мерного случайного вектора. Обычно в качестве координат вектора E^r используют независимые случайные величины, равномерно распределенные в интервале $[-1;1]$.

Схема метода с возвратом при неудачном шаге:

1. Задаем начальную точку x_0, начальную длину шага h_0 и полагаем счетчик числа итераций $r = 0$.
2. Задаем начальное значение счетчика числа неудачных попыток $k = 1$.
3. Получаем реализацию случайных чисел - $e_1, e_2, ..., e_n$ компонент вектора E^r и по формуле $x^{r+1} = x^r + h^r \cdot E^r$ находим пробную точку x^{r+1}.
4. Вычисляем значение $Q(x^{r+1})$ функции $Q(x)$ в точке x^{r+1}.
5. Если $Q(x^{r+1}) < Q(x^r)$, то полагаем $r = r + 1$ и переходим к пункту 3. Иначе – переходим к пункту 6.
6. Полагаем $k = k + 1$. Если $k < K$, то переходим к пункту 3. Иначе – переходим к пункту 7. Здесь K – предельное

количество неудачных попыток (свободный параметр метода). Рекомендуется $K = 3 \cdot n$.

7. Проверяем условие окончания поиска ($\left\| x^{r+1} - x^r \right\| = h^r \le \varepsilon_x$ или $\left| Q(x^{r+1}) - Q(x^r) \right| \le \varepsilon$). Если условие окончания поиска выполнено, то полагаем $x^* \approx x^{r+1}$ и завершаем итерации. Иначе – полагаем $r = r + 1$, $h^r = \lambda \cdot h^r$ и переходим к пункту 2. Здесь $h^r \in (0;1)$ - коэффициент уменьшения шага (свободный параметр метода).

В качестве условия окончания поиска можно использоваться одно из стандартных условий окончания итераций:

$$\left\| x^{r+1} - x^r \right\| = h^r \le \varepsilon_x$$

где ε_x - константа, определяющая требуемую точность решения по x;

$$\left| Q(x^{r+1}) - Q(x^r) \right| \le \varepsilon$$

где ε - константа, определяющая требуемую точность решения по $Q(x)$.

Заключение

В данных методических указаниях авторы, руководствуясь собственным опытом решения задач, приведённых в курсовой работе по дисциплине «Автоматизация проектирования систем и средств управления», постарались достаточно подробно описать действия обучающихся по выполнению тех или иных разделов. Особенно это касалось методики использования таких программных комплексов как LabView, MathLab, МВТУ, предназначенных для решения задач параметрического синтеза систем автоматического управления (САУ) с применением методов нелинейного программирования (НЛП). Подробные объяснения с примерами использования той или иной программы позволят, по мнению авторов, избежать типичных ошибок при выполнении курсовой работы.

Для удобства пользования студентами дневной очной и вечерней форм обучения, данные методические указания размещены на студенческом сайте кафедры «Управление и информатика в технических системах» МИИТа, где представлены и другие материалы преподавателей кафедры и студентов, которые должны, по мнению авторов, помочь в освоении таких дисциплин, как «Теория Автоматического Управления», «Локальные системы», «Идентификация и диагностика систем управления» и др.

Авторы будут благодарны замечаниям, дополнениям, высказанным читателями данных методических указаний. Ваша критика обязательно будет учтена в дальнейшей работе.

Список литературы

1. Монахов О.И. Анализ и синтез САУ с применением ЭВМ. Методические указания к курсовому проекту. – М.: МИИТ. 2004. – 28 с.
2. Урдин В.И., Олексеевич В.П. Методические указания к курсовому проектированию. – М.: МИИТ. 1988. – 28 с.
3. Монахов О.И., Мигулёва М.А., Тырнова О.В. Проектирование систем управления средствами программного комплекса МВТУ 3.5. Методические указания к курсовому проекту. – М.: МИИТ. 2006. – 37 с.
4. Монахов О.И., Александров Е.В. Проектирование систем и средств управления средствами инструментальной системы MATLAB 6.5. Методические указания к курсовому проекту. – М.: МИИТ. 2005. – 28 с.
5. Монахов О.И., Сергеев С.С. Проектирование систем и средств управления средствами Labview. Методические указания к курсовому проекту. – М.: МИИТ. 2005. – 67 с.

Учебно-методическое издание

Монахов Олег Иванович,
Сафронов Антон Игоревич,
Ковалев Максим Владимирович,
Рындина Екатерина Юрьевна

ПАРАМЕТРИЧЕСКИЙ СИНТЕЗ САУ С ПОМОЩЬЮ ПАКЕТОВ ПРИКЛАДНЫХ ПРОГРАММ

Методические указания к курсовому проектированию

Подписано в печать 01.04.10. Формат 60×84/16 Тираж 100 экз.

Усл. печ. л. - 8,75. Заказ № 203. Изд. № 67-10

127994, Москва, ул. Образцова, д.9, стр.9
Типография МИИТа.