УДК 514.75

Н.В. Малаховский

(Российский государственный университет им. Иммануила. Канта)

Компьютерное моделирование исследования дифференцируемых многообразий и ассоциированных связностей

Разработана компьютерная программа продолжений и охватов полей геометрических объектов на n-мерном дифференцируемом многообразии и ассоциированных связностей, порождённых оснащениями многообразия.

Интенсивное развитие компьютерной техники и программирования позволяет не только осуществлять компьютерное моделирование различных прикладных задач, но и использовать компьютеры в исследованиях в области “чистой” математики. В частности, в [1, 2] разработана программа исследования пфаффовых систем дифференциальных уравнений в частных производных. В современной дифференциальной геометрии при исследовании многообразий важную роль играет метод продолжений и охватов полей геометрических объектов, сопряжённый (при многократном продолжении систем дифференциальных уравнений) с громоздкими вычислениями.

В данной работе на основе пакета программ Maple **9** представлена компьютерная программа продолжений и охватов, позволяющая быстро осуществлять многократные продолжения полей геометрических объектов и структурных форм гладкого многообразия M_n, его подмногообразий и объектов

ассоциированных с ним связностей, а также находить охваты геометрических объектов.

Программа состоит из вводной части и двух основных программ: программы продолжений и программы охватов. Вводная часть содержит список процедур и ряд других команд, необходимых для работы основных программ.

Для использования программы продолжений необходимо задать список дифференциальных уравнений исследуемого многообразия **m_eq**; список **s_eq** структурных уравнений рассматриваемого пространства, формул инфинитезимального перемещения подвижного репера и уравнений структуры дифференцируемого многообразия; список **c_eq** структурных уравнений для форм связности; список **f_eq** конечных соотношений и тождеств на формы Пфаффа, функции и константы; список **d_eq** дифференциалов объектов, необходимых для продолжения системы; список **equations[1]** дифференциальных уравнений исследуемого многообразия, требующих продолжения. Базисные формы рассматриваемого многообразия указываются в списке **b_forms**, структурные формы группы и гладкого многообразия, – в списке **s_forms**, формы связности, – в списке **c_forms**, функции (компоненты объектов) – в списке **scalars**, вершины репера или его базисные векторы – в списке **frame**, тензоры, допускающие взаимное обращение – в списке **recipr**, константы – в списке **constants**. Пустые списки обозначаются символом **[ ]**. Список индексов суммирования **indces** составляется следующим образом. Буква, задающая индекс суммирования приравнивается множеству из букв, задающих индексы суммирования, пробегающих в совокупности значение этого индекса. При этом индексы суммирования из списка **indces** должны соответствовать индексам, используемым в уравнениях структуры. В качестве символа Кронекера необходимо использовать букву **δ**. Требуемое число продолжений исходной системы задаётся числом

n_n. После этого запустить вводную часть и программу продолжений. Результатом работы программы является продолжение соответствующих дифференциальных уравнений рассматриваемого многообразия с приведением в них подобных членов при одинаковых базисных формах, а также получение чистых замыканий продолженных уравнений.

Для использования программы охватов необходимо задать списки **m_eq**, **d_eq**, **f_eq**, **b_forms**, **s_forms**, **scalars**, **constants**, **recipr**, **indces**. Кроме того, необходимо задать список **equations** формул охватываемых объектов, и список **scope** формул охватов, причём левые части формул списка **scope** должны состоять из одного слагаемого – выражения охватываемого объекта. После этого запустить вводную часть и программу охватов. Для нахождения охватов альтернированных (симметрированных) выражений необходимо задавать их в списках **equations** и **scope** в проальтернированном (просимметрированном) виде.

Формы Пфаффа, функции (компоненты объектов), вершины подвижного репера и символы Кронекера задаются со списками индексов суммирования. Непустые списки индексов суммирования задаются следующим образом. Справа от соответствующей буквы в квадратных скобках составляются два списка, каждый из которых в свою очередь заключается или в квадратные скобки, если индексы суммирования списка не являются симметричными, или в фигурные скобки, если индексы суммирования списка являются симметричны. Некоторые из индексов суммирования каждого из этих списков в случае их симметричности заключаются в фигурные скобки. Индексы суммирования первого списка являются нижними. Индексы суммирования второго списка верхние. Индексы суммирования одной группы обозначаются одной буквой с подстрочными индексами – натуральными числами. Оператор внешнего умножения "$\wedge$" задаётся символом "` &^ `", справа

от которого в круглых скобках в соответствующем порядке через запятую записываются формы Пфаффа. Например, выражение $\Gamma^{\alpha_1\alpha_2}_{a_1a_2a_3a_4}\omega^{a_1}\wedge\omega^{a_4}$, в котором группы индексов суммирования a_2,a_3 и α_1,α_2 симметричны, записывается так: Γ [[a_1,{a_2, a_3} , a_4], { α_1,α_2 }] * ` &^ ` (ω [[ ],[a_1]], ω [[ ],[a_4]]). При этом индексы суммирования a_1,a_2,a_3,a_4 и α_1,α_2 принадлежат разным группам. Не допускается использование ненумерованных индексов суммирования, использования знака "_", прописных букв I и строчной буквы d в любых обозначениях. В качестве индексов суммирования допускается использование цифр и констант, указанных в соответствующем списке **constants**. Например, ω^0_0 (ω [[0], [0]]), ω^n_0 (ω [[0], [n]]). Пустые списки индексов суммирования обозначаются символом [ ].

Данная компьютерная программа апробирована на продолжениях систем дифференциальных уравнений в работе [2]. Например, для последовательных продолжений системы дифференциальных уравнений *r*-мерного многообразия ($1\le r\le m(n-m)+n$, $a_1,a_2,...=\overline{1,m},\alpha_1,\alpha_2,...=\overline{m+1,n}$) центрированных плоскостей в *n*-мерном проективном пространстве в параметрической форме:

$$\omega^{a_1}=\Lambda^{a_1}_{i_1}\theta^{i_1},\quad \omega^{\alpha_1}=\Lambda^{\alpha_1}_{i_1}\theta^{i_1},\quad \omega^{\alpha_1}_{a_1}=\Lambda^{\alpha_1}_{a_1i_1}\theta^{i_1}.$$

(см.[2, с. 12]) она имеет вид:

```
>restart:
>b_forms:=[theta[[],[i[1]]]]:c_forms:=[];
>s_forms:=[omega[[J[1]],[]],omega[[J[1]],[J[2]]],omega[
[],[J[1]]],theta[[],[i[1]]]]:
>scalars:=[Lambda[[i[1]],[a[1]]],Lambda[[i[1]],
[alpha[1]]], >Lambda[[a[1],i[1]],[alpha[1]]]]:
>recipr:=[]:frame:=[]:constants:=[]:indces:=[J =
{a,alpha},i={i}]:
```

```
>m_eq:=[omega[[],[a[1]]] = Lambda[[i[1]],[a[1]]]*theta
[[],[i[1]]], omega[[],[alpha[1]]] = Lambda[[i[1]],[alpha
[1]]]*theta[[],[i[1]]], omega[[a[1]],[alpha[1]]] =
Lambda[[a[1], i[1]],[alpha[1]]]*theta[[],[i[1]]]]:
>c_eq:=[]:
>s_eq:=[d(omega[[],[J[1]]])=omega[[],[J[2]]]&^omega[[J[
2]],[J[1]]],d(omega[[J[1]],[]])=omega[[J[1]],[J[2]]]&^o
mega[[J[2]],[]],d(omega[[J[1]],[J[2]]])=omega[[J[1]],[J
[3]]]&^omega[[J[3]],[J[2]]]+delta[[J[1]],[J[2]]]*omega[
[J[3]],[]]&^omega[[],[J[3]]]+`&^`(omega[[J[1]],[]],omeg
a[[],[J[2]]]),d(theta[[],[i[1]]]) = `&^`(theta[[],[i[2]
]], theta[[i[2]],[i[1]]])]:
>d_eq:=[]:f_eq:=[]:n_n:=1:

>equations[1]:=[omega[[],[a[1]]] = Lambda[[i[1]],[a[1]]]
*theta[[],[i[1]]], omega[[],[alpha[1]]] = Lambda[[i[1]]
,[alpha[1]]]*theta[[],[i[1]]], omega[[a[1]],[alpha[1]]]
= Lambda[[a[1], i[1]],[alpha[1]]]*theta[[],[i[1]]],
d(theta[[],[i[1]]]) = `&^`(theta[[],[i[2]]],theta[[i[2]]
,[i[1]]])];
```

Вводная часть

```
> with(difforms):with(combinat):
> c_f:={seq(op(0,op(p_p,c_forms)),p_p=1..nops(
c_forms))}:
> f_s:={seq(op(0,op(p_p,s_forms)),p_p=1..nops(
s_forms))}:
> s_s:={seq(op(0,op(p_p,[op(scalars),op(frame)])),
> p_p=1..nops([op(scalars),op(frame)]))}: c_s:=constants
:b_f:=b_forms:
> proc_1:=proc(p_1) local i1,i2,w1,w2,w3,w4;global G1_1,
G1_2;G1_1:=[]:w4:=[]:
> w1:=convert(subs(`&^`=`*`,p_1),list):w3:=a_i union
c_i:
> for i1 while i1<=nops(w3) do
>   if not type(op(i1,w3),const) then w2:=numboccur(w1,op
    (i1,w3)):i2:=1:
>     while w2<>0 do if numboccur(w1,op(i1,w3)[i2])<>0
      then
>       w1:=subs(op(i1,w3)[i2]=NULL,w1):w2:=
        numboccur(w1,op(i1,w3)):
```

```
>   end if:i2:=i2+1:end do:
>   G1_1:=[op(G1_1),op(i1,w3)[i2]]:
>   else w4:={op(w4),op(i1,w3)} end if:
> end do:G1_2:=[op(w4),op(G1_1)]: end proc:
> proc_2:=proc(p_1,p_2) local i1,i2,w1,w2;global G2_1;
G2_1:=[]:
> for i1 while i1<=nops(a_i) do
>   w1:=select(has,p_1,op(i1,a_i)):w2:=select(has,
    p_2,op(i1,a_i)):
>   G2_1:=[op(G2_1),seq(op(i2,w1)=op(0,op(w2))[op(op
    (w2))+i2-1],i2=1..nops(w1))]:
> end do: end proc:
> proc_3:=proc(p_1,p_2) local i1,i2,i3,w1;global G3_1;
G3_1:=p_1:
> for i1 while i1<=nops(G3_1) do w1:=op(i1,G3_1):
>   for i2 while i2<=nops(a_i) do select(has,p_2,op(i2,
    a_i)):
>      seq([op(i3,%)=op(i2,a_i)[i3],op(i2,a_i)[i3]=op
       (i3,%)],i3=1..nops(%)):
>    w1:=subs(%,w1):end do:proc_6(w1):
>   for i2 while i2<=nops(a_i) do select(has,G6_3,op(i2,
    a_i)):
>      w1:=subs([seq(op(i3,%)=op(i2,a_i)[i3],i3=1..
       nops(%))],w1):
> end do:G3_1:=subs(op(i1,G3_1)=w1,G3_1):
> end do: end proc:
> proc_4:=proc(p_1,p_2,p_3) local i1,i2,i3,i4,i5,i6,i7,
i8,i9,i10,
> w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14;
> global G4_1,G4_2,G4_3;G4_1:=[]:G4_2:=false:w1[1]:=p_1:
w1[2]:=p_2:
> for i1 while i1<=2 do proc_6(w1[i1]):w2[i1]:=remove(has,
G6_3,p_3) end do:
> for i1 while i1<=nops(a_i) do
>   w3[i1]:=select(has,w2[1],op(i1,a_i)):w4[i1]:=select
    (has,w2[2],op(i1,a_i)):
>   if nops(w3[i1])<>nops(w4[i1]) then i1:=1:break end
    if:
> end do:
> for i2 while i2<=i1-1 do
>   w5:=seq(seq(op(i6,w3[i7])=op(i7,a_i),i6=1..nops
    (w3[i7])),i7=i2+1..i1-1):
```

```
>   w6:=seq(seq(op(i6,w4[i7])=op(i7,a_i),i6=1..nops
    (w4[i7
    ])),i7=i2+1..i1-1):
>   w11:=w3[i2];w12:=w4[i2]:i5:=1:
>   for i3 while i3<=nops(w12) do
>      for i4 while i4<=nops(w11) do
>         w7[i5]:=op(i4,w11)=op(i3,w12):w8:=remove(has
          ,w11,op(i4,w11)):
>         w13:=subs([w5,w6,seq(w7[i6],i6=1..i5),seq(w10
          [i6],i6=1..i2-1),
>         seq(op(i6,w8)=op(i2,a_i),i6=1..nops(w8))],w1
          [1]):
>         w14:=subs([w5,w6,seq(op(i6,w12)=op(i2,a_i),
          i6=i3+1..nops(w12))],w1[2]):
>         if comparray([w13], [w14], dontprint)=true then
>            w11:=remove(has,w11,op(i4,w11)):i5:=i5+1:
            break
>   end if:end do:end do:w10[i2]:=seq(w7[i6],i6=1..i5-1
    ):
>   if nops(w11)<>0 then break end if:
> end do:subs([seq(op([w10[i6]]),i6=1..i2-1)],w1[1]):
> if comparray([%],[w1[2]], dontprint)=true then
>   G4_1:=[seq(op([w10[i6]]),i6=1..i1-1)]:G4_2:=true
> end if: end proc:
> proc_5:=proc(p_1) local i1,i2,i3,i4,w1,w2,w3,w4,w5,w6
;global G5_1;G5_1:={}:
> for i1 in p_1 do expand(lhs(i1)-rhs(i1)):
>   if type(%,`+`) then w1:=convert(%,list) else w1:=[%]
    end if:
>   w2:=factor(convert(select(has,w1,d),`+`)):
>   if numboccur(w2,d)=1 then
>      if type(w2,`*`) then w3:=convert(w2,list) else
       w3:=[w2] end if:
>      w6:=select(has,w3,'d'):proc_6(w6):
>      convert(select(has,remove(has,w3,w6),G6_3),
       `*`)*op(w6):
>      w4:=solve({convert(w1,`+`)},{%}):G5_1:={op(G5_1)
       ,op(w4)}:
>      if has(s_eq,p_1) then proc_6(lhs(op(w4))):
>         if nops(G6_3)<>0 then
>            for i2 in a_i do select(has,G6_3,i2):
```

```
>              w5:=[seq(op(choose(%,p_p)),p_p=1..
               nops(%))]:
>              for i3 in w5 do
>                 if nops(i3)>1 then
>                    G5_1:={op(G5_1),subs([seq(op(i4,i3)
                    =op(1,i3),
>                    i4=2..nops(i3))],op(w4))}:
>      end if:end do:end do:end if:end if else
>      for i2 in w1 do if type(i2,`*`) then convert(i2
       ,list) else [i2] end if:
>         w3:=convert(select(has,%,[op(f_s),op(c_f),
          op(s_s),delta]),`*`):
>         if has(c_eq,p_1) then
>            if numboccur({w3},[op(f_s),op(c_f),op(s_s)
             ])=1 then
>               w4:=solve({convert(w1,`+`)},{w3}):G5_1:=
                {op(G5_1),op(w4)}:
>            end if else
>            if numboccur({w3},[op(f_s),op(c_f),op(s_s)
             ])<>0 then
>               w4:=solve({convert(w1,`+`)},{w3}):G5_1:=
                {op(G5_1),op(w4)}:break
>      end if:end if:end do:end if: end do: end proc:
> proc_6:=proc(p_1) local i1,i2,i3,w1,w2,w3;
> global G6_1,G6_2,G6_3,G6_4;G6_1:=[]:G6_2:=[]:G6_3:=[]:
w2:=[]:
> w1:=convert(subs(`&^`=`*`,p_1),list):w3:=a_i union c_i:
> for i1 while i1<=nops(w3) do
>    if not type(op(i1,w3),const) then i3:=numboccur(w1,
     op(i1,w3)):i2:=1:
>       while i3<>0 do
>          if numboccur(w1,op(i1,w3)[i2])=1 then
>             G6_1:=[op(G6_1),op(i1,w3)[i2]] end if:
>          if numboccur(w1,op(i1,w3)[i2])>1 then
>             G6_2:=[op(G6_2),op(i1,w3)[i2]] end if:
>          if numboccur(w1,op(i1,w3)[i2])>0 then
>             G6_3:=[op(G6_3),op(i1,w3)[i2]] end if:
>             w1:=subs(op(i1,w3)[i2]=NULL,w1):
>          i3:=numboccur(w1,op(i1,w3)):i2:=i2+1:
>       end do:else w2:=[op(w2),seq(op(i1,w3),i4=1..
        numboccur(p_1,op(i1,w3)))]
```

```
> end if: end do:G6_4:=[op(w2),op(G6_3)]: end proc:
> proc_7:=proc(p_2) local i1,i2,i3,i4,i5,i6,
> w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12,w13,w14,w15,
w16;global G7_1;G7_1:=[]:
> w1:=map(proc(x,y) if has(y,lhs(x)) then rhs(x) end if end
proc,i_s,p_2):
> if nops(w1)>1 then w2:=`intersect`(op(w1)) else w2:={}
end if:
> if nops(w2)=0 then
>   for i1 in s_eq do expand(lhs(i1)-rhs(i1)):
>      if type(%,`+`) then w1:=convert(%,set) else w1:={%}
       end if:
>      w7:=`intersect`(op(map(proc(x) proc_6(x):{op(G6_3
       )} end proc,w1))):
>      map(proc(x,y) if has(y,lhs(x)) then op(rhs(x)) end
       if end proc,i_s,w7):
>      map(proc(x,y) if has(y,lhs(x)) then op(rhs(x)) end
       if end proc,i_s,p_2):
>      if (%) subset (%%) then w2:=map(proc(x) proc_6(x):
       op(G6_2) end proc,w1):
>         for i2 while i2<=nops(w2) do w3:=op(map(proc
          (x,y)
>            if has(y,lhs(x)) then y=rhs(x) end if end
            proc,i_s,op(i2,w2))):w5[i2]:=w3:
>            for i3 in p_2 do
>                 w4:=op(map(proc(x,y) if has(y,lhs(x))
                  then x end if end proc,i_s,i3)):
>                 if rhs(w4) subset rhs(w3) then
>                    w3:=lhs(w3)=(rhs(w3) minus rhs(w4))
                    union {lhs(w4)}:w5[i2]:=w3
>         end if:end do:end do:w6:=[seq(w5[i5],i5=1..i2-1
          )]:
>         w8:=[seq(op(map(proc(x) if has(p_p,lhs(x)) then
>         lhs(x) end if end proc,i_s)),p_p=p_2)]:w9:=
          permute(w8,nops(w7)):
>         for i2 while i2<=nops(w9) do w10:=w1:
>            for i3 while i3<=nops(op(i2,w9)) do
            proc_1(w10):
>             w10:=subs(op(i3,w7)=op(select(has,G1_2
              ,op(i3,op(i2,w9)))),w10):
```

```
>         end do:w11[i2]:=w10:end do:w12:=[seq(w11[i4],i4=
          1..i2-1)]:
>         for i2 while i2<=nops(w12) do
>            for i3 while i3<=nops(op(i2,w12)) do w13[i3]
             :=[op(i3,op(i2,w12))]:
>               for i4 while i4<=nops(w6) do
>                  for i5 while i5<=nops(w13[i3]) do
>                     if has(op(i5,w13[i3]),lhs(op(i4
                      ,w6))) then proc_1(op(i5,w13[i3])):
>                        for i6 while i6<=nops(rhs(op(i4,w6
                        ))) do
>                           select(has,G1_2,op(i6,rhs(op
                           (i4,w6)))):
>                           w14[i6]:=subs(lhs(op(i4,w6))=
                           op(%),op(i5,w13[i3])):
>                        end do:w15[i5]:=seq(w14[i7],
                        i7=1..i6-1)
>                        else w15[i5]:=op(i5,w13[i3]):
>                     end if:end do:w13[i3]:=[seq(w15[i7
                      ],i7=1..i5-1)]:
>            end do:end do:proc_11([seq(op(w13[i7]
             ),i7=1..i3-1)]):
>            w16:=select(has,map(proc(x) if x<>0 then x
             end if end proc,G11_1),d):
>            G7_1:=[op(G7_1),op(solve({convert(G11_1,`+`
             )},op(w16)))]:
> end do:end if:end do: end if: end proc:
> proc_8:=proc(p_1,p_2) local i1,i2,i3,w1,w2,w3,w4,w5,w6
,w7,w8,w9;
> global G8_1;G8_1:=table(p_1):
> for i1 while i1<=nops(op(op(G8_1))) do
>    if G8_1[i1]<>0 then subs(`&^`=`*`,numer(G8_1[i1
     ])*denom(G8_1[i1])):
>       if type(%,`*`) then w1:=convert(%,list) else
        w1:=[%] end if:
>       if nops(w1)>1 then w2:=choose(w1,2):
>          for i2 while i2<=nops(w2)
>          do w3:=[op(1,op(i2,w2)),op(2,op(i2,w2))]:
>             if numboccur(w3,delta)<>0 and type(op(1,op
              (i2,w2)),`+`)=false and
>                type(op(2,op(i2,w2)),`+`)=false then
```

```
>              w4:=op(1,select(has,w3,delta)):w5:=op
               (remove(has,w3,w4)):
>              proc_6(w4):w7:=op(G6_3):proc_6(w5):w8:=
               op(G6_3):
>              w9:={w7} intersect {w8}:
>              if nops(w9)<>0 and nops(select(has,[w5],
               [op(p_2),delta
               ]))<>0
>                 then remove(has,[w7],op(1,w9)):subs
                  (op(1,w9)=op(%),w5):
>                 G8_1[i1]:=subs([w4=1,w5=%],G8_1[i1])
                  :i1:=i1-1:break
>      end if:end if:end do:end if else G8_1[i1]:=NULL:
>      G8_1:=table([seq(G8_1[i3],i3=1..nops(op(op(G8_1)
       )))]):i1:=i1-1
> end if: end do:G8_1:=[seq(G8_1[i3],i3=1..nops(op(op(
G8_1))))]: end proc:
> proc_9:=proc(p_1) local i1,i2,w1,w2;global G9_1;
> if nops(p_1)<>0 then
>    for i1 while i1<=nops(p_1) do proc_6(op(i1,p_1)):w1
     [i1]:=G6_1 end do:
> G9_1:=op({seq(w1[i2],i2=1..i1-1)}) else G9_1:=[] end if:
end proc:
> proc_10:=proc(p_1,p_2,p_3)
> local i1,i2,w1,w2,w3,w4;global G10_1;w4:=p_2:G10_1:=
p_3:
> for i1 while i1<=nops(p_1) do w3:=choose(w4,nops(lhs(op
(i1,p_1))*[])-1):
>    for i2 while i2<=nops(w3) do
>       proc_4(lhs(op(i1,p_1)),convert(op(i2,w3),`*`),
        []):
>       if G4_2=true then proc_6(lhs(op(i1,p_1))):w1:=
        G6_3:
>          proc_6(rhs(op(i1,p_1))):w2:=G6_3:
>          proc_1(G10_1):proc_2(remove(has,w2,w1),G1_1):
>          G10_1:=algsubs(subs([op(G4_1),op(G2_1)],op
           (i1,p_1)),G10_1):
>          w4:=subs([seq(op(i3,op(i2,w3))=NULL,i3=1..
           nops(op(i2,w3)))],w4):
>          i1:=0:break end if:
> end do:if nops(w4)=0 then break end if: end do: end proc:
```

```
> proc_11:=proc(p_1) local i1,i2,i3,w1,w2,w3; global G11_1
;G11_1:=p_1:
> for i1 while i1<=nops(G11_1) do
>    subs(`&^`=`*`,op(i1,G11_1)):simplify(%):w1:=numer
     (%)*denom(%):
>    if type(w1,`*`) then w2:=convert(w1,list) else w2:=
     [w1] end if:
>    w3:=select(has,w2,delta):
>    for i2 in w3 do
>       for i3 in d_i do
>          if (has(op(1,i2),op(1,i3)) and has(op(2,i2),op
           (2,i3))) or
>          (has(op(2,i2),op(1,i3)) and has(op(1,i2),op
           (2,i3)))
>             then defform(i2=const):break end if:
>       end do:if not type(i2,const) then assign(i2=0) end
       if:
>       if type(op(1,i2),const) and type(op(2,i2),const)
       then assign(i2=1) end if:
> end do: end do:G11_1:=map(proc(x) if x<>0 then x end if
end proc,G11_1):end proc:
> proc_12:=proc(p_2,p_3,p_5)
> local i1,i2,i3,i4,i5,w1,w2,w3,w4,w5,w6,w7,w8,w9,w10;
global G12_1,G12_2;
> G12_1:=table(p_2):G12_2:=table(p_3):
> w4:=G12_1[nops(op(op(G12_1)))]:w5:=G12_2[nops(op(op(
G12_2)))]:
> for i1 while i1<=nops(w5) do
>    for i2 while i2<=2 do proc_6(op(i2,op(i1,w5))):
>       for i3 while i3<=nops(G6_3) do w1:=op(i3,G6_3):
       proc_1(w4):
>          w2:=op(select(has,G1_1,op(0,w1))):
>          if numboccur(p_5,w1)=1 then
>             if i2=1 then w4:=delta[[w1],[w2]]*subs(w1=w2
              ,w4) else
>                w4:=delta[[w2],[w1]]*subs(w1=w2,w4)
>       end if:w5:=subs(w1=w2,w5):i2:=i2-1:break
>    end if:end do:end do: end do:
> G12_1[nops(op(op(G12_1)))]:=w4:G12_2[nops(op(op(G12_2
)))]:=w5:
> proc_4(convert(w5,`*`),convert(G12_2[1],`*`),[]):
subs(`&^`=`*`,w4):
```

```
> if type(%,`*`) then convert(%,list) else [%] end if:
> w6:=subs([seq(op(i4,w5)=NULL,i4=1..nops(w5))],%):
> for i1 while i1<=nops(G4_1) do proc_6(w6):
>   if numboccur(G6_2,rhs(op(i1,G4_1)))=1 then
>      proc_1(w6):proc_2([rhs(op(i1,G4_1))],G1_1):w6:=
       subs(G2_1,w6):
>      G12_1[nops(op(op(G12_1)))]:=subs(G2_1,G12_1[nops
       (op(op(G12_1)))]):
> end if: end do: G12_1[nops(op(op(G12_1)))]:=subs(G4_1,
G12_1[nops(op(op(G12_1)))]):
> G12_2[nops(op(op(G12_2)))]:=subs(G4_1,G12_2[nops(op(
op(G12_2)))]):
> G12_1:=[seq(G12_1[p_p],p_p=1..nops(op(op(G12_1))))]:
> G12_2:=[seq(G12_2[p_p],p_p=1..nops(op(op(G12_2))))]:
end proc:
> map(proc(x) defform(x=0) end proc,s_s): map(proc(x)
defform(x=1) end proc,f_s):
> map(proc(x) defform(x=1) end proc,c_f): map(proc(x)
defform(x=const) end proc,c_s):
> i_s:={op(indces)} union {seq(op(map(proc(x) x={x} end
> proc,rhs(op(p_p,indces)))),p_p=1..nops(indces))}:
> i_i:=map(proc(x) op(rhs(x)) end proc,i_s):
> c_i:=map(proc(x) if type(x,const) then x end if end
proc,{op(i_i),op(c_s)}):
> a_i:=map(proc(x) lhs(x),op(rhs(x)) end proc,i_s) minus
c_i:n_i:={}:
> for i_1 in i_s do w_1:=rhs(i_1):
>   for i_2 in remove(has,i_s,i_1) do
>      if rhs(i_2) subset rhs(i_1) then w_1:=`union`(w_1
       ,rhs(i_1),{lhs(i_2)})
>   end if:end do:n_i:={op(n_i),lhs(i_1)=w_1}: end do:d_
i:={}:
> for i_1 in n_i do w_1:=cartprod([[lhs(i_1)],rhs(i_1)]):
>   while not w_1[finished] do d_i:={op(d_i),w_1
[nextvalue]()} end do:
> end do:
>
```

Программа продолжений

```
> proc_5(m_eq):m_eq:=G5_1;proc_5(c_eq):c_eq:=G5_1:
> proc_5(d_eq):d_eq:=G5_1;proc_5(s_eq):s_eq:=G5_1:
```

```
> proc_5(f_eq):f_eq:={op(f_eq),op(G5_1)}:b_eq:={}:
>for i_1 while i_1<=nops(b_f) do i_7:=1:i_8:=1:i_9:=1:
>   w_1[1][1]:=op(i_1,b_f)=op(i_1,b_f):
>   for i_2 in [1,2] do proc_6(op(i_2,op(i_1,b_f)))
    :w_2:=G6_4:
>        for i_3 in w_2 do
>           for i_4 in n_i do
>              if not has(i_i,lhs(i_4)) and has(i_3,lhs
               (i_4)) then w_3:=permute(rhs(i_4),1):
>                 for i_6 while i_6<=i_7 do
>                    for i_5 in w_3 do
>                       proc_1(w_1[i_9][i_6]):w_4:=op
                        (select(has,G1_2,op(i_5))):
>                       if i_2=1 then w_1[i_9+1][i_8]:=
                        subs(i_3=w_4,lhs(w_1[i_9][i_6]))
                        =
>                          delta[[w_4],[i_3]]*
                           rhs(w_1[i_9][i_6]) else
>                          w_1[i_9+1][i_8]:=subs(i_3=w_4,
                           lhs(w_1[i_9][i_6]))=
>                          delta[[i_3],[w_4]]*rhs
                           (w_1[i_9][i_6]):
>                       end if:proc_11(w_1[i_9+1][i_8]
                        ):i_8:=i_8+1:
>                 end do:end do:i_7:=i_8-1:i_8:=1:i_9:=
                  i_9+1:break
>   end if:end do:end do:end
do:b_eq:={op(b_eq),seq(w_1[i_9][p_p],p_p=1..i_7)}:
> end do:b_eq:=map(proc(x) if lhs(x)<>rhs(x) then x end if
end proc,b_eq):
> seq(unassign(cat(w_,p_p)),p_p=1..4):
> for I_1 while I_1<= n_n do I_3:=0:
>   for I_2 while I_2<=nops(equations[I_1]) do
>      expand(rhs(op(I_2,equations[I_1]))-lhs(op(I_2,
       equations[I_1]))):
>      if type(%,`+`) then w_1:=convert(%,list) else
       w_1:=[%] end if:
>      proc_9(w_1):proc_3(w_1,G9_1):proc_11(G3_1):e_q
       :=G11_1:
>      if nops([G9_1])<>1 then print(`В уравнении`):print
       (op(I_2,equations[I_1])):
>        print(`нарушена система индексов.`):break
```

```
> end if:proc_9(e_q):b_i:=G9_1:expand(d(convert
  (e_q,`+`))):
> if type(%,`+`) then e_q:=table(convert(%,list))
  else e_q:=table([%]) end if:
> for i_1 while i_1<=6 do
>    for i_2 while i_2<=nops(op(op(e_q))) do
     numer(e_q[i_2])*denom(e_q[i_2]):
>       if type(%,`*`) then w_1:=convert(%,list) else
        w_1:=[%] end if:
>       subs(`&^`=`*`,formpart(e_q[i_2])):
>       if type(%,`*`) then w_2:=convert(%,list) else
        w_2:=[%] end if:
>       if i_1=1 then w_3:=select(has,w_2,d):w_5:=
        d_eq:
>          for i_3 in w_3 do w_4:=[]:proc_6(i_3):w_4:=
           [op(w_4),op(G6_4)]:
>       proc_7(w_4):w_5:={op(w_5),op(G7_1)}:end
        do:end if:
>       if i_1=2 then w_3:=w_2:w_5:=c_eq end if:
>       if i_1=3 then w_3:=[]:if numboccur(w_2,c_f)=1
        then
>         w_3:=[select(has,w_2,c_f)]:w_5:=c_eq end
          if:end if:
>       if i_1=4 then w_3:=w_2:w_5:=m_eq end if:
>       if i_1=5 then w_3:={op(w_1),op(w_2)}:w_5:=
        f_eq end if:
>       if i_1=6 then w_3:={op(w_1),op(w_2)}:w_5:=
        b_eq end if:w_6[i_2]:=e_q[i_2]:
>       proc_10(w_5,w_3,w_6[i_2]):w_6[i_2]:=
        expand(simpform(G10_1)):
>    end do:expand(convert([seq(w_6[p_p],p_p=1..
     i_2-1)],`+`)):
>    if type(%,`+`) then convert(%,list) else [%] end
     if:
>    proc_11(%):e_q:=table(G11_1)
> end do:[seq(e_q[p_p],p_p=1..nops(op(op(e_q))))]:
> if nops(b_eq)=0 then proc_8(%,[op(f_s),op(s_s),
  op(c_f)]) else
>    proc_8(%,s_s) end if:
> e_q:=table([op(select(has,G8_1,d)),op(remove
  (has,G8_1,d))]):
```

```
>      w_d:=map(proc(x) wdegree(x) end proc,{op(G8_1)}):
>      if nops(w_d)<>1 then print(`Уравнение`):print(op
       (I_2,equations[I_1])):
>         print(`не однородно относительно форм Пфаффа.`)
          :break
>      end if:seq(unassign(cat(w_,p_p)),p_p=1..9):
>      for i_1 while i_1<=nops(op(op(e_q))) do
>         for i_2 from i_1+1 while i_2<=nops(op(op(e_q
          )))do
>            [select(has,subs(`&^`=`*`,e_q[i_1])*[],[
           op(c_f),op(f_s),op(s_s),delta])]:
>            [select(has,subs(`&^`=`*`,e_q[i_2])*[],[
           op(c_f),op(f_s),op(s_s),delta])]:
>            proc_4(%%,%,b_i):
>            if G4_2=true then simpform(subs(G4_1,e_q
           [i_1])+e_q[i_2]):
>               if%=0 then e_q[i_1]:=NULL:e_q[i_2]:=NULL:
>                  e_q:=table([seq(e_q[p_p],p_p=1..nops
                   (op(op(e_q))))]):
>                  i_1:=i_1-1:break else
                   e_q[i_1]:=factor(%):e_q[i_2]:=NULL:
>                  e_q:=table([seq(e_q[p_p],p_p=1..
                   nops(op(op(e_q))))]):i_2:=i_2-1:
>      end if:end if:end do:end do:
>      w_1:=choose([op(b_f),op(b_f)],op(w_d)):t_s:=[]:
>      seq([assign(w_5[p_p]=[],w_6[p_p]=[])],p_p=1..
       nops(w_1)):
>      for i_1 while i_1<=nops(op(op(e_q))) do subs(
       `&^`=`*`,formpart(e_q[i_1])):
>         if type(%,`*`) then w_2:=convert(%,list) else
          w_2:=[%] end if:
>         w_3:=permute(w_2,op(w_d)):
>         for i_2 while i_2<=nops(w_1) do
>            for i_3 while i_3<=nops(w_3) do
>               for i_4 while i_4<=op(w_d) do
>                  proc_4(op(i_4,op(i_2,w_1)),op(i_4,
                   op(i_3,w_3)),[]):
>               if G4_2=false then break end if:end do:
>               if i_4>op(w_d) then w_5[i_2]:=[op(w_5[i_2
                ]),e_q[i_1]]:
>                  w_6[i_2]:=[op(w_6[i_2]),op(i_3,w_3)]:
>                  proc_12(w_5[i_2],w_6[i_2],b_i):
```

```
>                 w_5[i_2]:=G12_1:w_6[i_2]:=G12_2
                  :e_q[i_1]:=NULL:
>                 e_q:=table([seq(e_q[p_p],p_p=1..
                  nops(op(op(e_q))))]):
>                 i_1:=i_1-1:i_2:=nops(w_1):break end
                  if:end do:end do:
>     for i_1 while i_1<=nops(w_1) do
>       proc_11([seq(op(p_p,w_5[i_1]),p_p=1..
        nops(w_5[i_1]))]):proc_8(G11_1,s_s):
>       simpform(convert(G8_1,`+`)):proc_3([%],b_i):
        w_7[i_1]:=op(G3_1):
>     end do:t_s:=[seq(w_7[p_p],p_p=1..nops(w_1))]:
>     proc_3([seq(e_q[p_p],p_p=1..nops(op(op(e_q
      ))))],b_i):
>     e_q:=table(map(proc(x) if x<>0 then x end if end
      proc,G3_1)):
>     seq(unassign(cat(w_,p_p)),p_p=1..10):
>     seq([assign(w_1[p_p]=[],w_2[p_p]=[])],p_p=1..
      nops(b_f)):i_4:=0:
>     for i_1 while i_1<=nops(op(op(e_q))) do
      subs(`&^`=`*`,e_q[i_1]):
>       if type(%,`*`) then w_4:=convert(%,list) else
        w_4:=[%] end if:
>       for i_2 while i_2<=nops(b_f) do
>         for i_3 while i_3<=nops(w_4) do
>           proc_4(op(i_2,b_f),op(i_3,w_4),[]):
>           if G4_2=true then i_4:=i_4+1:w_5
            [i_4]:=i_2:
>             w_1[i_2]:=[op(w_1[i_2]),e_q[i_1]]:
>             w_2[i_2]:=[op(w_2[i_2]),[op(i_3,w_4)
              ]]:
>             proc_12(w_1[i_2],w_2[i_2],b_i):
>             w_1[i_2]:=G12_1:w_2[i_2]:=G12_2:
              e_q[i_1]:=NULL:
>             e_q:=table([seq(e_q[p_p],p_p=1..
              nops(op(op(e_q))))]):
>       i_2:=nops(b_f):i_1:=i_1-1:break end if:end
        do:end do:
>     end do:w_4:={seq(w_5[p_p],p_p=1..i_4)}:
>     for i_1 in w_4 do proc_11([seq(op(p_p,w_1[i_1]
      ),p_p=1..nops(w_1[i_1]))]):
```

```
>         proc_8(G11_1,s_s):w_1[i_1]:=G8_1:end do:
          i_10:=0:
>       for i_1 in w_4 do i_10:=i_10+1:
>         w_5:=op(0,op(op(1,w_2[i_1]))):proc_11
          (w_1[i_1]):convert(G11_1,`+`):
>         subs(op(op(1,w_2[i_1]))=w_5,%):expand(d(%)):
>         select(has,%,d(w_5)):subs(d(w_5)=1,%):
          expand(simplify(%)):
>         if type(%,`+`) then convert(%,list) else [%] end
          if:
>         proc_8(%,[op(f_s),op(s_s),op(c_f)]):w_8
          :=G8_1:
>         if numboccur(w_8,d)=1 and type(w_5,form)=
          true then
>           w_9:=op(select(has,w_8,'d')):
>           if numboccur(w_9,`&^`)=0 then I_3:=I_3+1:
>             w_9:=op(select(has,w_8,'d')):
              w_7:=formpart(w_9):
>             w_11:=op(formpart(w_9)):w_14:=w_11:
>             proc_6(w_9):w_17:=G6_1:proc_1(w_9):
>             if nops(op(op(e_q)))=0 then w_19:=
              op(1,w_2[i_1]):
>               proc_6(w_19):w_18:=G6_3:proc_2
                (w_18,G1_1):
>               w_20:=subs(G2_1,op(w_19)):w_21[1]:=
                []:w_21[2]:=[]:
>               for i_3 while i_3<=2 do
                w_22:=op(op(-i_3,w_20)):
>                 if type(w_22,const)=false then
>                   w_23:=select(has,op(i_3,w_11),
                    w_18):proc_6(w_23):
>                   if nops({op(G6_3),w_22})>1 and
                    type(op(i_3,w_11),set)=false then
>                     w_21[i_3]:=subs(op(w_23)={op(
                      G6_3),w_22},op(i_3,w_11)) else
>                     w_21[i_3]:=subs(op(w_23)=(op(
                      G6_3),w_22),op(i_3,w_11)):
>                   end if else w_21[i_3]:=op(i_3,
                    w_11)
>                 end if:
>               end do:w_25[1]:=`&^`(w_20,op(0,w_11)
                [seq(w_21[p_p],p_p=1..2)])
```

```
>                 else w_19:=b_f:
>                 for i_2 while i_2<=nops(w_19) do
>                    proc_6(op(i_2,w_19)):proc_2
                     (G6_3,G1_1):
>                    w_20[i_2]:=subs(G2_1,op(i_2,w_19))
                     :w_21[1]:=[]:w_21[2]:=[]:
>                       for i_3 while i_3<=2 do
                        w_22:=op(op(-i_3,w_20[i_2])):
>                          if type(w_22,const)=false then
>                            w_21[i_3]:=subs(op(op(i_3,
                             w_11))=
>                             (op(op(i_3,w_11)),w_22)
                              ,op(i_3,w_11)):
>                             else w_21[i_3]:=op(i_3,w_11)
>                       end if:end do:w_25[i_2]:=`&^`
                        (w_20[i_2],op(0,w_11)
>                       [seq(w_21[p_p],p_p=1..2)]):end
                        do:end if:w_27:=[w_9]:
>                 for i_3 while i_3<=nops(w_8) do
                  subs(`&^`=`*`,op(i_3,w_8)):
>                    if type(%,`*`) then convert(%,list)
                     else [%] end if:w_15:=choose(%):
>                    for i_4 while i_4<=nops(w_15) do
>                       proc_4(w_14,convert(op(i_4,w_15)
                        ,`*`),[]):
>                    if G4_2=true then w_27:=[op(w_27),op
                     (i_3,w_8)]:break end if:end do:
>                 end do:w_28:=table(remove(
                  has,w_8,w_27)):i_2:=1:w_29[0]:=[]:
>                 seq([assign(w_30[p_p]=[],w_31[p_p]=
                  [])],p_p=1..nops(op(op(w_28)))):
>                 for i_3 while i_3<=nops(op(op(w_28))) do
>                    w_32:=subs(`&^`=`*`,formpart
                     (w_28[i_3])):
>                    if type(w_32,`*`) then
                     w_33:=convert(w_32,list) else
>                    w_33:=[w_32] end if:w_29[i_2]
                     :=[op(w_29[i_2-1]),w_32]:
>                    for i_4 while i_4<=nops(w_29[i_2]) do
>                       proc_4(op(i_4,w_29[i_2]),w_32,
                        []):
```

```
>                     if G4_2=true then w_34[i_2]:=i_4:
>                        w_30[i_4]:=[op(w_30[i_4]),w_28
                         [i_3]]:
>                        w_31[i_4]:=[op(w_31[i_4]),
                         w_33]:
>                        proc_12(w_30[i_4],w_31[i_4],
                         w_17):
>                        w_30[i_4]:=G12_1:w_31
                         [i_4]:=G12_2:w_28[i_3]:=NULL:
>                        w_28:=table([seq(w_28[p_p],
                         p_p=1..nops(op(op(w_28))))]):
>                  i_2:=i_2+1:i_3:=i_3-1:break end if:end
                   do:end do:
>                  w_35:={seq(w_34[p_p],p_p=1..i_2-1)}:
>                  for i_3 in w_35 do
>                     proc_11([seq(op(p_p,w_30[i_3]
                     ),p_p=1..nops(w_30[i_3]))]):
>                     if nops(G11_1)=1 or numboccur
                     (G11_1,s_s)=0 then
>                        proc_8(G11_1,[op(f_s),op(s_s)
                        ,op(c_f)]) else proc_8(G11_1,s_s)
>                     end if:w_36[i_3]:=factor(convert
                     (G8_1,`+`)):
>                  end do:simpform(convert([op(w_27),
                   seq(w_36[p_p],p_p=w_35)],`+`)):
>                  if type(%,`+`) then convert(%,list) else
                   [%] end if:proc_8(%,f_s):
>                  w_39[i_10]:=`&^`(op(op(1,w_2[i_1]
                   )),{convert(G8_1,`+`)}):
>                  proc_3(sign(w_9)*G8_1,b_i):
                   w_37:=convert(G3_1,`+`):
>                  proc_3([seq(w_25[p_p],p_p=1..nops
                   (w_19))],b_i):w_38:=convert(G3_1,`+`):
>                  e_1[I_3]:=w_37=w_38:proc_5([e_1[I_3
                   ]]):
>                  d_eq:=[op(d_eq),op(G5_1)]:expand
                   (`&^`(op(op(1,w_2[i_1])),w_38)):
>                  if type(%,`+`) then w_60[i_10]:=convert
                   (%,list) else
>                  w_60[i_10]:=[%] end if:
>                  print(nprintf("Продолженное
                   уравнение:")):print(e_1[I_3]) else
```

```
>              w_39[i_10]:=`&^`(op(op(1,w_2[i_1])),
               {convert(w_8,`+`)}):w_60[i_10]:=[]:
>           end if else
>         w_39[i_10]:=`&^`(op(op(1,w_2[i_1]
          )),{convert(w_8,`+`)}):w_60[i_10]:=[]:
>       end if:unassign('w_12','w_13','w_32','w_29'
       ,'w_33','w_30','w_31','w_36'):
>    end do:print(nprintf("Чистое замыкание%d- го
    уравнения:",I_2)):
>    [seq(e_q[p_p],p_p=1..nops(op(op(e_q)))),
>    seq(w_39[p_p],p_p=1..i_10),convert(t_s,`+`)]:
    simpform(convert(%,`+`)):
>    if type(%,`+`) then convert(%,list) else [%] end if:
>    proc_3(%,b_i):print(convert(G3_1,`+`)=0):
>    expand(convert([seq(e_q[p_p],p_p=1..nops(op
    (op(e_q)))),
>    seq(op(w_60[p_p]),p_p=1..i_10),op(t_s)],`+`)):
>    if type(%,`+`) then convert(%,list) else [%] end
    if:w_52:=table(%):
>    if nops(c_eq)<>0 and op(op(w_52))<>NULL then
>       w_54:=choose([op(b_f),op(b_f)],op(w_d)):
>       seq([assign(w_57[p_p]=[],w_58[p_p]=[])],
       p_p=1..nops(w_54)):
>       for i_1 while i_1<=nops(op(op(w_52))) do
       subs(`&^`=`*`,formpart(w_52[i_1])):
>          if type(%,`*`) then w_55:=convert(%,list) else
          w_55:=[%] end if:
>             w_56:=permute(w_55,op(w_d)):
>             for i_2 while i_2<=nops(w_54) do
>                for i_3 while i_3<=nops(w_56) do
>                   for i_4 while i_4<=op(w_d) do
>                      proc_4(op(i_4,op(i_2,w_54)),
                      op(i_4,op(i_3,w_56)),[]):
>                   if G4_2=false then break end if:end do:
>                   if i_4>op(w_d) then w_57[i_2]:=[op(w_57
                   [i_2]),w_52[i_1]]:
>                      w_58[i_2]:=[op(w_58[i_2]),
                      op(i_3,w_56)]:
>                      proc_12(w_57[i_2],w_58[i_2],b_i):
>                      w_57[i_2]:=G12_1:w_58[i_2]:=
                      G12_2:w_52[i_1]:=NULL:
```

```
>                   w_52:=table([seq(w_52[p_p],p_p=1..
                    nops(op(op(w_52)))]):
>                   i_1:=i_1-1:i_2:=nops(w_54):break
>      end if:end do:end do:end do:
>      for i_1 while i_1<=nops(w_54) do
>         proc_11([seq(op(p_p,w_57[i_1]),
          p_p=1..nops(w_57[i_1]))]):proc_8(G11_1,s_s):
>         w_59[i_1]:=simpform(convert(G8_1,`+`)):
>      end do:w_53:=seq(w_59[p_p],p_p=1..nops(w_54)):
>      w_52:=seq(w_52[p_p],p_p=1..nops(op(op(w_52)))):
>      print(nprintf("Чистое замыкание%d- го уравнения с
       учётом разрешения по лемме Картана:",I_2)):
       simpform(convert([w_52,w_53],`+`)):
>      if type(%,`+`) then convert(%,list) else [%] end if:
>      proc_8(%,f_s):proc_3(G8_1,b_i):print
       (convert(G3_1,`+`)=0):
>    end if:print(`---------------------------------
     --------------------`):
>    seq(unassign(cat(w_,p_p)),p_p=1..60):
> end do:equations[I_1+1]:=[seq(e_1[p_p],p_p=1..I_3)]
:c_eq:=[]: end do:
>
```

Продолженное уравнение

$$d\left(\Lambda_{[i_1],[a_1]}\right)-\Lambda_{[i_2],[a_1]}\theta_{[i_1],[i_2]}+\Lambda_{[i_1],[a_2]}\omega_{[a_2],[a_1]}+\Lambda_{[i_1],[\alpha_1]}\omega_{[\alpha_1],[a_1]}=\Lambda_{[\{i_1,i_2\}],[a_1]}\theta_{[\ ],[i_2]}$$

Чистое замыкание 1-го уравнения

$$\left\{d\left(\Lambda_{[i_1],[a_1]}\right)-\Lambda_{[i_2],[a_1]}\theta_{[i_1],[i_2]}+\Lambda_{[i_1],[a_2]}\omega_{[a_2],[a_1]}+\Lambda_{[i_1],[\alpha_1]}\omega_{[\alpha_1],[a_1]}\right\}\wedge\theta_{[\ ],[i_1]}=0$$

Продолженное уравнение

$$d\left(\Lambda_{[i_1],[\alpha_1]}\right)-\Lambda_{[i_2],[\alpha_1]}\theta_{[i_1],[i_2]}+\Lambda_{[i_1],[\alpha_2]}\omega_{[\alpha_2],[\alpha_1]}=\Lambda_{[\{i_1,i_2\}],[\alpha_1]}\theta_{[\ ],[i_2]}$$

Чистое замыкание 2-го уравнения

$$\left\{d\left(\Lambda_{[i_1],[\alpha_1]}\right)-\Lambda_{[i_2],[\alpha_1]}\theta_{[i_1],[i_2]}+\Lambda_{[i_1],[\alpha_2]}\omega_{[\alpha_2],[\alpha_1]}\right\}\wedge\theta_{[\ ],[i_1]}-$$
$$\Lambda_{[i_1],[a_1]}\Lambda_{[a_1i_2],[\alpha_1]}\left(\theta_{[\ ],[i_1]}\wedge\theta_{[\ ],[i_2]}\right)=0$$

```
   ----------------------------------------
--------------------
```

Продолженное уравнение

$$d\left(\Lambda_{[a_1,i_1],[\alpha_1]}\right)-\Lambda_{[a_1,i_2],[\alpha_1]}\theta_{[i_1],[i_2]}-\Lambda_{[a_2,i_1],[\alpha_1]}\omega_{[a_1],[a_2]}+$$
$$\Lambda_{[a_1,i_1],[\alpha_2]}\omega_{[\alpha_2],[\alpha_1]}-\Lambda_{[i_1],[\alpha_1]}\omega_{[a_1],[\ ]}=\Lambda_{[a_1,\{i_1i_2\}],[\alpha_1]}\theta_{[\ ],[i_2]}$$

Чистое замыкание 3-го уравнения

$$\left\{d\left(\Lambda_{[a_1,i_1],[\alpha_1]}\right)-\Lambda_{[a_1,i_2],[\alpha_1]}\theta_{[i_1],[i_2]}-\Lambda_{[a_2,i_1],[\alpha_1]}\omega_{[a_1],[a_2]}+\right.$$
$$\left.\Lambda_{[a_1,i_1],[\alpha_2]}\omega_{[\alpha_2],[\alpha_1]}-\Lambda_{[i_1],[\alpha_1]}\omega_{[a_1],[\ ]}\right\}\wedge\theta_{[\ ],[i_1]}=0$$

```
   ----------------------------------------------------
```

Продолженное уравнение

$$d\left(\theta_{[i_2],[i_1]}\right)-\left(\theta_{[i_2],[i_3]}\wedge\theta_{[i_3],[i_1]}\right)=\theta_{[\ ],[i_3]}\wedge\theta_{[\{i_3,i_2\}],[i_1]}$$

Чистое замыкание 4-го уравнения

$$\left\{d\left(\theta_{[i_2],[i_1]}\right)-\left(\theta_{[i_2],[i_3]}\wedge\theta_{[i_3],[i_1]}\right)\right\}\wedge\theta_{[\ ],[i_2]}=0$$

```
   ----------------------------------------------------
```

Программа нахождения охватов геометрических объектов

```
> equations:={}: scope:={}:
> proc_5(m_eq):m_eq:=G5_1;proc_5(c_eq):c_eq:=G5_1:
> proc_5(d_eq):d_eq:=G5_1;proc_5(s_eq):s_eq:=G5_1:
> proc_5(f_eq):f_eq:={op(f_eq),op(G5_1)}:
> w_1:=table([seq([op(p_p,scope)],p_p=1..nops(scope
))]):
> f_scope:=[seq(lhs(op(w_1[p_p])),p_p=1..nops(scope))]:
> for i_5 while i_5<=nops(scope) do ex-
pand(rhs(op(w_1[i_5]))):
>  if type(%,`+`) then convert(%,list) else [%] end
   if:proc_9(%):proc_8(%%):
>  w_2:=table(map(proc(x) if x<>0 then x end if end
   proc,G8_1)):
```

```
> for i_7 while i_7<=nops(op(op(w_2))) do
>    for i_8 from i_7+1 while i_8<=nops(op(op(w_2))) do
>       [select(has,w_2[i_7]*[],[op(f_s),op(s_s),
        op(c_f),delta])]:
>       [select(has,w_2[i_8]*[],[op(f_s),op(s_s),
        op(c_f),delta])]:proc_4(%%,%,G9_1):
>       if G4_2=true then simpform(subs(G4_1,w_2[i_7])+
        w_2[i_8]):
>          if %=0 then w_2[i_7]:=NULL:w_2[i_8]:=NULL:
>             w_2:=table([seq(w_2[p_p],p_p=1..
              nops(op(op(w_2))))]):
>             i_7:=i_7-1:break else w_2[i_7]:=factor(%):
              w_2[i_8]:=NULL:
>             w_2:=table([seq(w_2[p_p],p_p=1..
              nops(op(op(w_2))))]):i_8:=i_8-1:
> end if:end if:end do:end do:w_1[i_5]:=[lhs(op
  (w_1[i_5]))=
> convert([seq(w_2[p_p],p_p=1..nops(op(op(w_2)
  )))],`+`)]:
> end do: scope1:=[seq(op(w_1[p_p]),p_p=1..nops
(scope))]:
> seq(unassign(cat(w_,p_p)),p_p=1..2):
> for i_1 in scope1 do lhs(i_1)-rhs(i_1):
> if type(%,`+`) then convert(%,list) else [%] end if:
> proc_8(%,[op(s_s),op(f_s)]):w_1:=G8_1:w_2:={}:
> for i_2 in w_1 do
>    if type(i_2,`*`) then convert(i_2,list) else [i_2]
     end if:
>    w_2:={op(w_2),op(select(has,%,delta))}:
> end do:w_3:=[seq(op(choose(w_2,p_p)),p_p=1..
  nops(w_2))]:
> for i_2 while i_2<=nops(w_3) do w_4:=w_1:
>    for i_3 in op(i_2,w_3) do
     w_4:=expand(w_4*op(0,i_3)[op(2,i_3),op(1,i_3)]):
>       proc_8(w_4,[op(s_s),op(f_s)]):w_5:=table(G8_1):
>       for i_4 while i_4<=nops(op(op(w_5))) do w_5[i_4]:
>          if type(%,`*`) then w_6:=convert(%,list) else
           w_6:=[%] end if:
>          w_7[i_4]:=w_5[i_4]:proc_10(f_eq,w_6,
           w_7[i_4]):w_7[i_4]:=expand(G10_1):
>       end do:expand(convert([seq(w_7[p_p],p_p=1..
        i_4-1)],`+`)):
>       if type(%,`+`) then convert(%,list) else [%] end
        if:proc_11(%):
```

```
>       w_8:=map(proc(x) if x<>0 then x end if end
        proc,G11_1):
>       f_scope:={op(f_scope),op(1,w_8)}:i_6:=0:
>       for i_4 in w_8 do
>          if type(i_4,`*`) then w_9:=convert(i_4,list)
           else w_9:=[i_4] end if:
>          for i_5 in w_9 do proc_6(i_5):
>             if has(i_5,delta) then
>                if nops(G6_3)=1 then proc_1(w_8):proc_2
                (G6_3,G1_1):
>                   w_8:=subs(i_5=subs(G2_1,i_5),
                   expand(w_8/i_5)) else i_6:=i_6+1
>       end if:end if:end do:end do:scope1:=
        {op(scope1),op(1,w_8)=
>       -convert([seq(op(p_p,w_8),p_p=2..nops(w_8))
        ],`+`)}:
> if i_6=0 then break end if:end do:end do:
> end do:seq(unassign(cat(w_,i)),i=1..9):
> for I_2 while I_2<=nops(equations) do I_1[I_2]:=0:
>  w_1:=expand(lhs(op(I_2,equations))-rhs(op(I_2,
equations))):
>  if type(w_1,`+`) then w_2:=convert(w_1,list) else
w_2:=[w_1] end if:
>  proc_9(w_2):proc_3(w_2,G9_1):proc_11(G3_1):
>  e_q:=map(proc(x) if x<>0 then x end if end proc,G11_1):
>  proc_9(e_q):b_i:=G9_1:w_3:=convert(e_q,`*`):
>  if type(w_3,`*`) then w_4:=convert(w_3,list) else
w_4:=[w_3] end if:
>  for i_1 in w_4 do
>     for i_3 in recipr do proc_4(i_1,i_3,[]):
>        if G4_2=true then
>           for i_2 in a_i do
>              if numboccur(i_1,i_2)=2 then
>                 w_5:=subs([seq(op(p_p,G6_3)=i_2[p_p],
                  p_p=[1,2])],i_1):
>                 w_6:=subs([seq(op(p_p,G6_3)=i_2[p_p+1],
                  p_p=[1,2])],i_1):
>                 w_7:=op(0,i_1)[op(1,w_5),op(2,w_5)]*
                  op(0,i_1)[op(2,w_6),op(1,w_6)]:
>                 proc_6(op(1,i_1)):w_8:=G6_3:proc_6(op
                  (2,i_1)):w_9:=G6_3:
>                 if nops(w_8)=0 and nops(w_9)=2 then
>                    f_eq:={op(f_eq),w_7=delta[[i_2[3]],
                     [i_2[1]]]}:
```

```
>               scope1:={op(scope1),delta[[i_2[3]],
                [i_2[1]]]=w_7} end if:
>               if nops(w_8)=2 and nops(w_9)=0 then
>                  f_eq:={op(f_eq),w_7=delta[[i_2[1]],
                   [i_2[3]]]}:
>                  scope1:={op(scope1),delta[[i_2[1]],
                   [i_2[3]]]=w_7}
> end if:end if:end do:end if:end do:end do:seq
  (unassign(cat(w_,i)),i=1..7):
> expand(d(convert(e_q,`+`))):
> if type(%,`+`) then e_q:=table(convert(%,list)) else
  e_q:=table([%]) end if:
> for i_1 while i_1<=3 do
>    for i_2 while i_2<=nops(op(op(e_q))) do numer
     (e_q[i_2])*denom(e_q[i_2]):
>     if type(%,`*`) then w_1:=convert(%,list) else
      w_1:=[%] end if:
>     if i_1=1 then w_3:=select(has,w_1,d):w_5:=d_eq
      end if:
>      if i_1=2 then w_3:=w_1:w_5:=m_eq end if:
>      if i_1=3 then w_3:=w_1:w_5:=f_eq end if:
>      w_6[i_2]:=e_q[i_2]:proc_10(w_5,w_3,w_6[i_2]):
       w_6[i_2]:=G10_1:
>    end do:expand(convert([seq(w_6[p_p],p_p=1..i_2-1)
     ],`+`)):
>    if type(%,`+`) then convert(%,list) else [%] end if:
>    e_q:=table(map(proc(x) if x<>0 then x end if end
     proc,%)):
>    if i_1=2 then i_3:=1:
>       w_10:=select(has,[seq(e_q[p_p],p_p=1..nops(op
        (op(e_q))))],'d'):
>       for i_4 while i_4<=nops(w_10) do
>          convert(scalarpart(op(i_4,w_10))*[],list):
           w_11:=select(has,%,s_s):
>          for i_5 in w_11 do
>             proc_6(op(1,i_5)):w_15:=G6_3:proc_6
              (op(2,i_5)):w_16:=G6_3:
>             if nops(w_15)=2 and nops(w_16)=0 or
              nops(w_15)=0 and nops(w_16)=2 then
>                proc_6(op(i_4,w_10)):w_12:=G6_2:
>                proc_6(i_5):w_13:=select(has,G6_3,
                 w_12):
>                i_5:=subs([op(1,G6_3)=op(2,G6_3),op
                 (2,G6_3)=op(1,G6_3)],i_5):
```

```
>                if nops(w_13)=1 then
>                   proc_1([entries(e_q),seq(w_14[p_p],
                    p_p=1..i_3-1)]):
>                   select(has,G1_1,op(0,op(w_13))):
                    subs(op(w_13)=op(%),i_5):
>                   w_14[i_3]:=op(0,%)[op(2,%),
                    op(1,%)]:i_3:=i_3+1:
>       end if:end if:end do:end do:convert([seq
        (w_14[p_p],p_p=1..i_3-1)],`*`):
>       proc_8(expand([seq(e_q[p_p],p_p=1..
        nops(op(op(e_q))))]*%),[op(f_s),op(s_s)]):
> e_q:=table(G8_1):end if:end do:
> proc_8([seq(e_q[p_p],p_p=1..nops(op(op(e_q))))],
     [op(f_s),op(s_s)]):
> proc_9(G8_1):proc_3(G8_1,G9_1):proc_11(G3_1):
> w_18:=map(proc(x) if x<>0 then x end if end
     proc,G11_1):proc_9(w_18):b_i:=G9_1:
> e_q:=table([op(select(has,w_18,d)),op
     (remove(has,w_18,d))]):
> seq(unassign(cat(w_,i)),i=1..18):
> for i_1 while i_1<=nops(op(op(e_q))) do
>    for i_2 from i_1+1 while i_2<=nops(op(op(e_q))) do
>       [select(has,e_q[i_1]*[],[op(f_s),op(s_s),
        delta])]:
>       [select(has,e_q[i_2]*[],[op(f_s),op(s_s),
        delta])]:proc_4(%%,%,b_i):
>       if G4_2=true then simpform(subs(G4_1,e_q[i_1])
        +e_q[i_2]):
>          if%=0 then e_q[i_1]:=NULL:e_q[i_2]:=NULL:
>             e_q:=table([seq(e_q[p_p],p_p=1..
              nops(op(op(e_q))))]):
>             i_1:=i_1-1:break else e_q[i_1]:=%:e_q[i_2]
              :=NULL:
>             e_q:=table([seq(e_q[p_p],p_p=1..
              nops(op(op(e_q))))]):i_2:=i_2-1:
> end if:end if:end do:end do:
> w_1:=table(remove(has,[seq(e_q[p_p],p_p=1..
  nops(op(op(e_q))))],d)):
> w_2:=select(has,[seq(e_q[p_p],p_p=1..
  nops(op(op(e_q))))],d):i_2:=1:w_4[0]:=[]:
> seq([assign(w_5[p_p]=[],w_6[p_p]=[],w_7[p_p]=
  [])],p_p=1..nops(op(op(w_1)))):
> for i_1 while i_1<=nops(op(op(w_1))) do
```

```
>    w_3:=[formpart(w_1[i_1])]:w_4[i_2]:=[op(w_4
     [i_2-1]),%]:
>    for i_3 while i_3<=nops(w_4[i_2]) do proc_4(op(i_3,
     w_4[i_2]),w_3,[]):
>       if G4_2=true then w_7[i_2]:=i_3:w_5[i_3]:=
        [op(w_5[i_3]),w_1[i_1]]:
>          w_6[i_3]:=[op(w_6[i_3]),w_3]:proc_12
           (w_5[i_3],w_6[i_3],b_i):
>          w_5[i_3]:=G12_1:w_6[i_3]:=G12_2:
           w_1[i_1]:=NULL:
>          w_1:=table([seq(w_1[p_p],p_p=1..
           nops(op(op(w_1))))]):
> i_1:=i_1-1:i_2:=i_2+1:break end if:end do:end do:
> w_4:={seq(op(op(p_p,w_4[i_2-1])),p_p=1..
  nops(w_4[i_2-1]))}:
> w_8:={seq(w_7[p_p],p_p=1..i_2-1)}:i_6:=0:
> for i_1 in w_8 do w_10:=[seq(op(p_p,w_5[i_1]),p_p=1..
  nops(w_5[i_1]))]:
>    for i_2 while i_2<=nops(scope1) do expand(rhs(op
     (i_2,scope1))):
>       if type(%,`+`) then w_12:=convert(%,list) else
        w_12:=[%] end if:
>       proc_6(lhs(op(i_2,scope1))):w_9:=G6_2:i_9:=0:
>       if nops(w_10)>=nops(w_12) and op(w_12)<>0 then
        w_13:=table(w_10):
>          for i_3 while i_3<=nops(w_12) do i_10:=0:
           op(i_3,w_12):
>             if type(%,`*`) then w_29:=convert(%,list)
              else w_29:=[%] end if:
>             w_14:=select(has,w_29,[op(s_s),delta]):
>             w_15:=convert(remove(has,w_29,w_14),
              `*`):proc_6(w_14):
>             w_16:=subs({seq(op(p_p,G6_2)=NULL,p_p=1..
              nops(G6_2))},w_14):
>             for i_4 while i_4<=nops(op(op(w_13))) do
              w_13[i_4]:
>                if type(%,`*`) then w_17:=convert
                 (%,list) else w_17:=[%] end if:
>                w_19:=select(has,%,[op(s_s),
                 delta]):proc_6(w_19):
>                if nops(w_19)>=nops(w_14) then
>                   w_18:=subs({seq(op(p_p,G6_2)=NULL,
                    p_p=1..nops(G6_2))},w_19):
```

```
>                  w_20:=choose([seq(p_p,p_p=1..
                   nops(w_19))],nops(w_14)):
>                  for i_5 while i_5<=nops(w_20) do
>                     w_21:=[seq(op(p_p,w_18),
                      p_p=op(i_5,w_20))]:
>                     w_22:=[seq(op(p_p,w_19),
                      p_p=op(i_5,w_20))]:
>                     proc_4(convert(w_14,`*`),
                      convert(w_22,`*`),[]):
>                     if G4_2=true then
>                        if i_9=0 then w_11:=G4_1:
                        i_10:=i_10+1:
>                          w_25:=convert(remove(has,
                          w_17,w_22),`*`)/w_15:
>                          proc_1(w_17):proc_2(w_9,
                          G1_1):
>                          w_27:=subs([op(G2_1),op(w_11
                          )],lhs(op(i_2,scope1)))*w_25:
>                          w_13[i_4]:=NULL:
>                          w_13:=table([seq(w_13[p_p],
                          p_p=1..nops(op(op(w_13))))]):
>                          i_9:=i_9+1:i_4:=nops
                          (op(op(w_13))):break else
>                          w_26:=convert(remove(has
                          ,w_17,w_22),`*`)/w_15:
>                          proc_4(simplify(w_26),
                          simplify(w_25),b_i):
>                          w_23:=convert(subs(w_11,
                          w_16),`*`):
>                          w_24:=convert(w_21,`*`):
>                          if comparray([w_23],[w_24],
                          dontprint)=true and G4_2=true
>                             then i_10:=i_10+1:
                             w_13[i_4]:=NULL:
>                             w_13:=table([seq(w_13[p_p]
                             ,p_p=1..nops(op(op(w_13))))]):
>                             i_9:=i_9+1:i_4:=nops
                             (op(op(w_13))):break:
>              end if:end if:end if:end do:end if:
>           end do:if i_10=0 then break end if:end do:
>           if i_9=nops(w_12) then
>              w_10:=[seq(w_13[p_p],p_p=1..
              nops(op(op(w_13)))),w_27]:i_2:=0 end if:
>       end if:end do:i_6:=i_6+1:w_28[i_6]:=w_10:
```

```
> end do:w_30:=convert([op(w_2),seq(op(w_28[p_p]),
  p_p=1..i_6)],`+`):
> if type(w_30,`+`) then w_31:=convert(w_30,list) else
  w_31:=[w_30] end if:
> proc_8(w_31,[op(s_s),op(f_s)]):proc_3
  (G8_1,b_i):proc_11(G3_1):w_32:=G11_1:
> proc_5([convert(w_32,`+`)=0]):d_eq:=[op(d_eq),
  op(G5_1)]:
> e_q:=collect(convert(w_32,`+`)=0,w_4,factor):i_5:=0:
> for i_1 in w_32 do if type(i_1,`*`) then convert
  (i_1,list) else [i_1] end if:
>    w_33:=select(has,%,f_s):w_34:=select(has,%%,s_s):
>    if has(i_1,d) then i_5:=i_5+1 else
>       for i_2 while i_2<=nops(w_33) do
>          for i_3 while i_3<=nops(b_forms) do
>             proc_4(op(i_2,w_33),op(i_3,b_forms),[]):
>          if G4_2=true then i_5:=i_5+1:break end if:
>          end do:
>          if i_3>nops(b_forms) then
>             if nops(w_34)=0 then i_5:=i_5+1 else
>                for i_3 while i_3<=nops(w_34) do
>                   for i_4 while i_4<=nops(f_scope) do
>                      proc_4(op(i_3,w_34),op(i_4,
                       f_scope),[]):
>                      if G4_2=true then break end if:
>                   end do:if i_4>nops(f_scope) then break
                    end if:
>                end do:if i_3>nops(w_34) then i_5:=i_5+1
                 end if:
> end if:end if:end do:end if:end do:print(e_q):
> if i_5=nops(w_32) then I_1[I_2]:=I_1[I_2]+1 end if:
> end do:print(`Система величин:`):
> print({seq(lhs(op(p_p,scope)),p_p=1..nops(scope))}):
> if sum('I_1[k]','k'=1..I_2-1)=nops(equations) then
> print(`образует охват.`) else print(`не образует
  охвата.`)
> end if:seq(unassign(cat(w_,p_p)),p_p=1..34):
```

Список литературы

Малаховский В.С., Малаховский Н.В. Компьютерные программы приложения метода внешних форм Картана к исследованию систем линейных дифференциальных уравнений в

частных производных и к исследованию дифференцируемых многообразий// Вестник КГУ. Калининград, 2005. Вып.1–2, 2005.-С.46–54.

Бондаренко Е.В. Связности на многообразии центрированных плоскостей в проективном пространстве// Диф. геом. многообр. фигур. Калининград, 2000. № 31. С. 12–13.

Computer modeling of research of differentiable manifolds and associated connections

Computer program of prolongations and scopes of fields of geometrical objects on n-dimensional differentiable manifold and associated with it connections, generated by framing of manifolds, is developed.