主専攻実習（定理証明班）

第六回課題レポート

担当：森継 修一

知識情報システム主専攻 201611502 久保川一良

2018 年 11 月 19 日

◇接続環境

- 自分のローカル環境に Reduce をインストールして利用した。
  - 使用した PC のスペックについて： https://bit.ly/2Cg7hqb
- OS や Reduce のオプション設定などについては以下の通りである。

```
username@my_computer:~ [HH:MM:SS]
 $ lsb_release -a

No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 16.04.5 LTS
Release:        16.04
Codename:       xenial

username@my_computer:~ [HH:MM:SS]
 $ reduce                        # alias reduce='redcsl -v -w -k 4000 --nogui'

Codemist Standard Lisp revision 4765 for linux-gnu:x86_64: Sep 19 2018
Created: Wed Sep 19 15:57:15 2018

Reduce (Free CSL version, revision 4765), 19-Sep-18 ...
Memory allocation: 4168 Mbytes
There are 8 processors available
```

◇入力ファイル

```
% [定理 ２５] -----------------------------------------------------------;
% △ABC の頂点 A,B,C から直線 g に降ろした垂線の足を A,E,F とすると、
% 点 D,E,F からそれぞれ BC, CA, AB におろした垂線 DP,EQ,FR は一点で交わる
% ※直線 g を x 軸と一致するものとみなして証明していく
% A(u1, u2) B(u3, u4) C(u5, u6) D(u1, 0) E(u3, 0) F(u5, 0)
% P(x1, x2) Q(x3, x4) R(x5, x6)
%-----------------------------------------------------------------------;

% 関数定義読み込み） ---------------------------------------------------;

load_package groebner;
on comp,gcd,ezgcd;
off allfac,pwrds;

in cal_sys_relations$


% ２点間のユークリッド距離 D^2 -----------------------------------------------;
procedure squared_euclid(a1,a2,b1,b2)$
begin
 scalar d;
 d:=(a1-b1)^2+(a2-b2)^2;
 return d
end$
%-----------------------------------------------------------------------;


% ---------------------------------------------------------------------;
% ＜証明＞ -------------------------------------------------------------;

order x6, x5, x4, x3, x2, x1, u6, u5, u4, u3, u2, u1;
factor x6, x5, x4, x3, x2, x1;

%-----------------------------------------------------------------------;
% 仮定 -----------------------------------------------------------------;

% DP, EQ, FR の共点を言うためには、§６例３により、
```

```
% ĐB^2 - ĐC^2 + EC^2 -EA^2 + FA^2 -FB^2 = 0 を示せれば良い

% ĐP⊥BC かつ  B-P-C は共線
h1:=vertically(u1, 0, x1, x2, u3, u4, u5, u6);
h2:=collinear(u3, u4, x1, x2, u5, u6);
% EQ⊥CA かつ  C-Q-A は共線
h3:=vertically(u3, 0, x3, x4, u5, u6, u1, u2);
h4:=collinear(u5, u6, x3, x4, u1, u2);
% FR⊥AB かつ  A-R-B は共線
h5:=vertically(u5, 0, x5, x6, u1, u2, u3, u4);
h6:=collinear(u1, u2, x5, x6, u3, u4);

% EQ と FR が P で交わる
% E-P-Q  は共線となっている

h7:=collinear(u3, 0, x1, x2, x3, x4);

%----------------------------------------------------------------------;
% 結論 ------------------------------------------------------------------;
% F-P-R は共線となっている  →  2 つの共線の共通の点、は共点である（？

conclusion:=collinear(u5, 0, x1, x2, x5, x6);



showtime;

%----------------------------------------------------------------------;
% Groebner Basis: 結果が 1 となったら、仮定が誤っている可能性が高い ---------------;

%%% 変数を定義し、lex 形式で並べる ------------------------------------------;
torder({x6, x5, x4, x3, x2, x1}, lex)$

%%% 仮定において定義した式から Groebner Basis を求める -------------------------;
gb:=groebner{h1, h2, h3, h4, h5, h6, h7};

%---------------------------------------------------------;
% 「glterms」が出力するのは、グレブナー基底の計算過程で〈ゼロにはならない〉
% と仮定された式のリストである。
%---------------------------------------------------------;

%%% u に関する制約条件 -------------------------------------------------;
glterms;


%%% gb を法として g を簡約 --------------------------------------------------;
preduce(conclusion, gb);

% ==> 0 になっていれば、定理は成立 ------------------------------------------;

showtime;
;end;
```

◇出力ファイル

```
% [定理 ２５] ---------------------------------------------------------;
% △ABC の頂点 A,B,C から直線 g に降ろした垂線の足を A,E,F とすると、
% 点 Đ,E,F からそれぞれ BC, CA, AB におろした垂線 ĐP,EQ,FR は一点で交わる
% ※直線 g を x 軸と一致するものとみなして証明していく
% A(u1, u2) B(u3, u4) C(u5, u6) Đ(u1, 0) E(u3, 0) F(u5, 0)
% P(x1, x2) Q(x3, x4) R(x5, x6)
%----------------------------------------------------------------------;

% 関数定義読み込み） ---------------------------------------------------;

load_package groebner;


on comp,gcd,ezgcd;


off allfac,pwrds;



in cal_sys_relations$




% ２点間のユークリッド距離 Đ^2 -----------------------------------------------;
procedure squared_euclid(a1,a2,b1,b2)$
begin
 scalar d;
 d:=(a1-b1)^2+(a2-b2)^2;
 return d
end$


%----------------------------------------------------------------------;


% --------------------------------------------------------------------;
% ＜証明＞ -------------------------------------------------------------;

order x6, x5, x4, x3, x2, x1, u6, u5, u4, u3, u2, u1;


factor x6, x5, x4, x3, x2, x1;



%----------------------------------------------------------------------;
% 仮定 -----------------------------------------------------------------;

% ĐP, EQ, FR の共点を言うためには、§６例３により、
% ĐB^2 - ĐC^2 + EC^2 -EA^2 + FA^2 -FB^2 = 0 を示せれば良い

% ĐP⊥BC かつ  B-P-C は共線
h1:=vertically(u1, 0, x1, x2, u3, u4, u5, u6);


h1 := x2*(u6 - u4) + x1*(u5 - u3) - u5*u1 + u3*u1

h2:=collinear(u3, u4, x1, x2, u5, u6);


h2 := x2*( - u5 + u3) + x1*(u6 - u4) - u6*u3 + u5*u4

% EQ⊥CA かつ  C-Q-A は共線
h3:=vertically(u3, 0, x3, x4, u5, u6, u1, u2);


h3 := x4*( - u6 + u2) + x3*( - u5 + u1) + u5*u3 - u3*u1

h4:=collinear(u5, u6, x3, x4, u1, u2);
```

```
h4 := x4*(u5 - u1) + x3*( - u6 + u2) + u6*u1 - u5*u2

% FR⊥AB かつ  A-R-B は共線
h5:=vertically(u5, 0, x5, x6, u1, u2, u3, u4);


h5 := x6*(u4 - u2) + x5*(u3 - u1) - u5*u3 + u5*u1

h6:=collinear(u1, u2, x5, x6, u3, u4);


h6 := x6*( - u3 + u1) + x5*(u4 - u2) - u4*u1 + u3*u2


% EQ と FR が P で交わる
% E-P-Q は共線となっている

h7:=collinear(u3, 0, x1, x2, x3, x4);


h7 := x4*x1 - x4*u3 - x3*x2 + x2*u3


%----------------------------------------------------------------------;
% 結論 ----------------------------------------------------------------;
% F-P-R は共線となっている → 2 つの共線の共通の点、は共点である（？

conclusion:=collinear(u5, 0, x1, x2, x5, x6);


conclusion := x6*x1 - x6*u5 - x5*x2 + x2*u5




showtime;


Time: 0 ms


%----------------------------------------------------------------------;
% Groebner Basis: 結果が 1 となったら、仮定が誤っている可能性が高い ---------------;

%%% 変数を定義し、lex 形式で並べる ------------------------------------------;
torder({x6, x5, x4, x3, x2, x1}, lex)$



%%% 仮定において定義した式から Groebner Basis を求める -------------------------;
gb:=groebner{h1, h2, h3, h4, h5, h6, h7};


gb := {1}


%---------------------------------------------------------;
% 「glterms」が出力するのは、グレブナー基底の計算過程で〈ゼロにはならない〉
% と仮定された式のリストである。
%---------------------------------------------------------;

%%% u に関する制約条件 --------------------------------------------------;
glterms;


{ - u6 + u4,

  - u5 + u3,

  - u6 + u2,

  - u5 + u1,

  - u4 + u2,

  - u3 + u1,
```

```
   2                2                2     2
 u6  - 2*u6*u4 + u5  - 2*u5*u3 + u4  + u3 ,

   2                2                2     2
 u6  - 2*u6*u2 + u5  - 2*u5*u1 + u2  + u1 ,

   2                2                2     2
 u4  - 2*u4*u2 + u3  - 2*u3*u1 + u2  + u1 ,

 u3}


%%% gb を法として g を簡約 --------------------------------------------------;
preduce(conclusion, gb);


0


% ==> 0 になっていれば、定理は成立 ------------------------------------------;

showtime;


Time: 20 ms  plus GC time: 21 ms

;

end;
```

◇関数定義ファイル（試作）

```
%----------------------------------------------------------------------;
% Proving Geometry Theorems by Groebner Basis
%----------------------------------------------------------------------;
load_package groebner;
on comp,gcd,ezgcd,combineexpt;
off allfac,pwrds;

%----------------------------------------------------------------------;
% AB and CD cross vertically.
procedure vertically(a1,a2,b1,b2,c1,c2,d1,d2)$
begin
 scalar c;
 c:=(a1-b1)*(c1-d1)+(a2-b2)*(c2-d2);
 return c
end$
%----------------------------------------------------------------------;

% ２点間のユークリッド距離 D^2 -----------------------------------------------;
procedure squared_euclid(a1,a2,b1,b2)$
begin
 scalar d;
 d:=(a1-b1)^2+(a2-b2)^2;
 return d
end$
%----------------------------------------------------------------------;

% tangent theta -------------------------------------------------------;
% n1=m1 のときは定義されないことに注意
procedure tan_theta(n1, n2, m1, m2)$
begin
    scalar f;
    f:=(n2-m2)/(n1-m1);
    return f
end$
%----------------------------------------------------------------------;

% ######################################################################

% caliculate tangent alpha and half-alpha -----------------------------;
procedure calc_tan_alpha(b1, b2, c1, c2)$
begin
    scalar bd_2, gx, gy, mx, my, e1, e2, e3, e4, x_d, y_d;
    bc_2:=squared_euclid(b1, b2, c1, c2);
    gx:=b1+sqrt(bc_2);
    gy:=by;
    mx:=(gx-c1)/2;
    my:=(gy-c2)/2;
    tan_alpha:=tan_theta(b1, b2, c1, c2);
    tan_half_alpha:=tan_theta(b1, b2, mx, my);
    e1:=1/tan_alpha;
    e2:=2*tan_half_alpha/(1+tan_half_alpha);
    tan_list:=list(e1, e2);
    return tan_list
end$
%----------------------------------------------------------------------;

% rotate angle --------------------------------------------------------;
procedure rotate(px, py, b1, b2, c1, c2)$
begin
    scalar e1, e2, e3, e4, px_d, py_d;
    e_list:=calc_tan_alpha(b1, b2, c1, c2);
    e1:=first(e_list);
    e2:=second(e_list);
    e3:=px+py*tan_alpha;
    e4:=px*tan_alpha+py;
    px_d:=(-1)*e1*e2*e3;
    py_d:=(-1)*e1*e2*e4;
    pxy_list:=list(px_d, py_d);
    return pxy_list
end$

% tangent ∠ABC --------------------------------------------------------;
```

```
procedure tan_abc(a1, a2, b1, b2, c1, c2)$ ----------------------------;
begin
    scalar ;
    h1:=vertically(b1, b2, a1, a2, b1, b2, c1, c2);
    if h1=0 then write "this angle is cross vertival."
    else if a1-b1=0 and not c1-b1=0
    then <<
    c_d_list:=rotate(c1, c2, b1, b2, c1, c2);
    c1_d:=first(c_d_list);
    c2_d:=second(c_d_list);
    %%% もしかして：C のぶんの計算は不要？
    a_d_list:=rotate(a1, a2, b1, b2, c1, c2);
    a1_d:=first(a_d_list);
    a2_d:=second(a_d_list)
    tangent_abc:=calc_tan_alpha(b1, b2, a1_d, a2_d)>>

    else if c1-b1=0 and not a1-b1=0
    then <<
    c_d_list:=rotate(c1, c2, b1, b2, a1, a2);
    c1_d:=first(c_d_list);
    c2_d:=second(c_d_list);
    a_d_list:=rotate(a1, a2, b1, b2, a1, a2);
    a1_d:=first(a_d_list);
    a2_d:=second(a_d_list)
    tangent_abc:=calc_tan_alpha(b1, b2, a1_d, a2_d)>>

    else <<
    k1:=(a2-b2)/(a1-b1); % AB の傾き
    k2:=(c2-b2)/(c1-b1); % BC の傾き
% tan(ABC)  <<(k2-k1)/(1+k1*k2)>>
    m1:=(k2-k1)*(a1-b1)*(c1-b1); % 分子
    m2:=(1+k1*k2)*(a1-b1)*(c1-b1); % 分母
    >>

end$
%----------------------------------------------------------------------;
% ∠ABC = ∠ÐEF
procedure tangent(a1,a2,b1,b2,c1,c2,d1,d2,e1,e2,f1,f2)$
begin
        scalar ;

        if a1-b1=0 and not c1-b1=0      % c1-b1=0 or d1-e1=0 or f1-e1=0 は後回し
        then <<
        tan_a:=tan_theta(b1, b2, c1, c2);
        tan_half_a:=tan_theta(b1, b2, m1, m2)
        >>
        else <<
        k1:=(a2-b2)/(a1-b1); % AB の傾き
        k2:=(c2-b2)/(c1-b1); % BC の傾き
        k3:=(d2-e2)/(d1-e1); % ÐE の傾き
        k4:=(f2-e2)/(f1-e1)>>% EF の傾き

% tan(ABC)  <<(k2-k1)/(1+k1*k2)>>
        m1:=(k2-k1)*(a1-b1)*(c1-b1); % 分子
        m2:=(1+k1*k2)*(a1-b1)*(c1-b1); % 分母

% tan(ÐEF)  <<(k4-k3)/(1+k3*k4)>>
        n1:=(k4-k3)*(d1-e1)*(f1-e1);
        n2:=(1+k3*k4)*(d1-e1)*(f1-e1);

        p:=m1*n2-m2*n1;
        return p
end$
%----------------------------------------------------------------------;


order x6, x5, x4, x3, x2, x1, u6, u5, u4, u3, u2, u1;
factor x6, x5, x4, x3, x2, x1;

tangent(x6, x5, x4, x3, x2, x1, u6, u5, u4, u3, u2, u1);

end$
```