主専攻実習（定理証明班）

第五回課題レポート（再提出）

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 16 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 １３] ------------------------------------------------------------;
% 〈メネラウス(Menelaus)の定理の証明〉
% △ABC の３辺 BC, CA, AB またはその延長が頂点を通らない１直線と交わる点を、
% それぞれ P, Q, R とする。 ※xy 座標上で考え、 １直線は x 軸と一致していると想定する
% A(u1, u2) B(u3, u4) C(u5, u6)
% P(x1, 0) Q(x2, 0) R(x3, 0)
%----------------------------------------------------------------------;

% 関数定義読み込み） ---------------------------------------------------;
load_package groebner;
on comp,gcd,ezgcd;
off allfac,pwrds;

%----------------------------------------------------------------------;
% 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 x3, x2, x1, u6, u5, u4, u3, u2, u1;
factor x3, x2, x1;


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

% A, B, C から直線におろした垂線の足を Ad(u1, 0), Bd(u3, 0), Cd(u5, 0)とすると、
% AAd // BBd // CCd だから、 BP:PC=BBd:CCd, CQ:QA=CCd:AAd, AR:RB=AAd:BBd
% これは、両辺を二乗しても同様のことが言える。
```

```
bp2:=squared_euclid(u3, u4, x1, 0);
pc2:=squared_euclid(x1, 0, u5, u6);
cq2:=squared_euclid(u5, u6, x2, 0);
qa2:=squared_euclid(x2, 0, u1, u2);
ar2:=squared_euclid(u1, u2, x3, 0);
rb2:=squared_euclid(x3, 0, u3, u4);

aad2:=squared_euclid(u1, u2, u1, 0);
bbd2:=squared_euclid(u3, u4, u3, 0);
ccd2:=squared_euclid(u5, u6, u5, 0);


% BP:PC=BBd:CCd より
h1:=bp2*ccd2-bbd2*pc2;

% CQ:QA=CCd:AAd より
h2:=cq2*aad2-ccd2*qa2;

% AR:RB=AAd:BBd より
h3:=ar2*bbd2-aad2*rb2;


%-----------------------------------------------------------------------;
% 結論 ------------------------------------------------------------------;

%  BP      CQ      AR
% ----  ・  ----  ・  ---- = 1
%  PC      QA      RB           となる

conclusion:=bp2*cq2*ar2-pc2*qa2*rb2;


%-----------------------------------------------------------------------;
% 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({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};

%---------------------------------------------------------;
% 「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 １３] ----------------------------------------------------------;
% 〈メネラウス(Menelaus)の定理の証明〉
% △ABC の３辺 BC, CA, AB またはその延長が頂点を通らない１直線と交わる点を、
% それぞれ P, Q, R とする。 ※xy 座標上で考え、１直線は x 軸と一致していると想定する
% A(u1, u2) B(u3, u4) C(u5, u6)
% P(x1, 0) Q(x2, 0) R(x3, 0)
%----------------------------------------------------------------------;

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


on comp,gcd,ezgcd;


off allfac,pwrds;



%----------------------------------------------------------------------;
% 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 x3, x2, x1, u6, u5, u4, u3, u2, u1;


factor x3, x2, x1;




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

% A, B, C から直線におろした垂線の足を Ad(u1, 0), Bd(u3, 0), Cd(u5, 0)とすると、
% AAd // BBd // CCd だから、 BP:PC=BBd:CCd, CQ:QA=CCd:AAd, AR:RB=AAd:BBd
% これは、両辺を二乗しても同様のことが言える。

bp2:=squared_euclid(u3, u4, x1, 0);


         2                2     2
bp2 := x1  - 2*x1*u3 + u4  + u3

pc2:=squared_euclid(x1, 0, u5, u6);


         2                2     2
pc2 := x1  - 2*x1*u5 + u6  + u5

cq2:=squared_euclid(u5, u6, x2, 0);


         2                2     2
cq2 := x2  - 2*x2*u5 + u6  + u5

qa2:=squared_euclid(x2, 0, u1, u2);


         2                2     2
```

```
qa2 := x2  - 2*x2*u1 + u2  + u1

ar2:=squared_euclid(u1, u2, x3, 0);

         2               2     2
ar2 := x3  - 2*x3*u1 + u2  + u1

rb2:=squared_euclid(x3, 0, u3, u4);

         2               2     2
rb2 := x3  - 2*x3*u3 + u4  + u3

aad2:=squared_euclid(u1, u2, u1, 0);

          2
aad2 := u2

bbd2:=squared_euclid(u3, u4, u3, 0);

          2
bbd2 := u4

ccd2:=squared_euclid(u5, u6, u5, 0);

          2
ccd2 := u6


% BP:PC=BBd:CCd より
h1:=bp2*ccd2-bbd2*pc2;

        2    2     2               2             2      2   2     2   2
h1 := x1 *(u6  - u4 ) + x1*( - 2*u6 *u3 + 2*u5*u4 ) + u6 *u3  - u5 *u4

% CQ:QA=CCd:AAd より
h2:=cq2*aad2-ccd2*qa2;

        2       2     2            2             2      2   2     2   2
h2 := x2 *( - u6  + u2 ) + x2*(2*u6 *u1 - 2*u5*u2 ) - u6 *u1  + u5 *u2

% AR:RB=AAd:BBd より
h3:=ar2*bbd2-aad2*rb2;

        2    2     2               2             2      2   2     2   2
h3 := x3 *(u4  - u2 ) + x3*( - 2*u4 *u1 + 2*u3*u2 ) + u4 *u1  - u3 *u2


%-----------------------------------------------------------------;
% 結論 -----------------------------------------------------------;

%  BP     CQ     AR
% ---- ・ ---- ・ ---- = 1
%  PC     QA     RB           となる

conclusion:=bp2*cq2*ar2-pc2*qa2*rb2;

                2   2                      2   2       2     2     2     2
conclusion := x3 *x2 *x1*(2*u5 - 2*u3) + x3 *x2 *( - u6  - u5  + u4  + u3 )

                   2      2                      2
               + x3 *x2*x1 *( - 2*u5 + 2*u1) + x3 *x2*x1*(4*u5*u3 - 4*u5*u1)

                   2         2          2             2          2
```

```
                + x3 *x2*(2*u6 *u1 + 2*u5 *u1 - 2*u5*u4  - 2*u5*u3 )

                    2   2    2     2     2     2
                + x3 *x1 *(u6  + u5  - u2  - u1 )

                    2            2          2             2          2      2
                + x3 *x1*( - 2*u6 *u3 - 2*u5 *u3 + 2*u5*u2  + 2*u5*u1 ) + x3 *(

                    2   2    2   2    2   2    2   2    2   2    2   2
                  u6 *u4  + u6 *u3  - u6 *u2  - u6 *u1  + u5 *u4  + u5 *u3

                       2   2    2   2          2   2
                   - u5 *u2  - u5 *u1 ) + x3*x2 *x1 *(2*u3 - 2*u1)

                       2
                + x3*x2 *x1*( - 4*u5*u3 + 4*u3*u1)

                       2      2          2          2          2
                + x3*x2 *(2*u6 *u3 + 2*u5 *u3 - 2*u4 *u1 - 2*u3 *u1)

                          2
                + x3*x2*x1 *(4*u5*u1 - 4*u3*u1) + x3*x2

                      2             2                2             2
               *( - 4*u6 *u3*u1 - 4*u5 *u3*u1 + 4*u5*u4 *u1 + 4*u5*u3 *u1)

                       2          2          2             2          2
                + x3*x1 *( - 2*u6 *u1 - 2*u5 *u1 + 2*u3*u2  + 2*u3*u1 )

                             2             2                   2              2
                + x3*x1*(4*u6 *u3*u1 + 4*u5 *u3*u1 - 4*u5*u3*u2  - 4*u5*u3*u1 ) +

                           2   2          2   2          2      2       2      2
                x3*( - 2*u6 *u4 *u1 - 2*u6 *u3 *u1 + 2*u6 *u3*u2  + 2*u6 *u3*u1

                         2   2          2   2          2      2       2      2
                   - 2*u5 *u4 *u1 - 2*u5 *u3 *u1 + 2*u5 *u3*u2  + 2*u5 *u3*u1 )

                    2   2        2     2     2     2
                + x2 *x1 *( - u4  - u3  + u2  + u1 )

                    2            2          2           2           2      2
                + x2 *x1*(2*u5*u4  + 2*u5*u3  - 2*u3*u2  - 2*u3*u1 ) + x2 *(

                       2   2    2   2    2   2    2   2    2   2    2   2
                   - u6 *u4  - u6 *u3  - u5 *u4  - u5 *u3  + u4 *u2  + u4 *u1

                       2   2    2   2
                   + u3 *u2  + u3 *u1 )

                       2            2          2       2          2
                + x2*x1 *( - 2*u5*u2  - 2*u5*u1  + 2*u4 *u1 + 2*u3 *u1) + x2*x1

                          2             2                2             2
               *( - 4*u5*u4 *u1 - 4*u5*u3 *u1 + 4*u5*u3*u2  + 4*u5*u3*u1 ) + x2*(

                     2   2          2   2          2   2          2   2
                  2*u6 *u4 *u1 + 2*u6 *u3 *u1 + 2*u5 *u4 *u1 + 2*u5 *u3 *u1

                            2   2          2   2          2   2          2   2
                   - 2*u5*u4 *u2  - 2*u5*u4 *u1  - 2*u5*u3 *u2  - 2*u5*u3 *u1 ) +

                  2    2   2    2   2    2   2    2   2    2   2    2   2
                x1 *(u6 *u2  + u6 *u1  + u5 *u2  + u5 *u1  - u4 *u2  - u4 *u1

                       2   2    2   2                2      2       2      2
                   - u3 *u2  - u3 *u1 ) + x1*( - 2*u6 *u3*u2  - 2*u6 *u3*u1

                         2      2       2      2          2   2          2   2
                   - 2*u5 *u3*u2  - 2*u5 *u3*u1  + 2*u5*u4 *u2  + 2*u5*u4 *u1

                            2   2          2   2
                   + 2*u5*u3 *u2  + 2*u5*u3 *u1 )


%----------------------------------------------------------------------;
```

```
% 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({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};


             2     2    2           2       2             2   2     2   2
gb := { - (u2  - u4 )*x3  - (2*u1*u4  - 2*u2 *u3)*x3 + (u1 *u4  - u2 *u3 ),

          2     2    2           2       2             2   2     2   2
       (u2  - u6 )*x2  + (2*u1*u6  - 2*u2 *u5)*x2 - (u1 *u6  - u2 *u5 ),

             2     2    2           2       2             2   2     2   2
        - (u4  - u6 )*x1  - (2*u3*u6  - 2*u4 *u5)*x1 + (u3 *u6  - u4 *u5 )}


%----------------------------------------------------------;
% 「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,

  - u6 + u4,

 u6 + u2,

  - u6 + u2,

 u4 + u2,

  - u4 + u2}



%%% 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: 30 ms


;

end;
```