|  |
| --- |
| 主専攻実習（定理証明班） |
| 第三回課題レポート |
| 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10月22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

# 演習01\_1：「重心」の証明プログラムを実行せよ

* 入力ファイル

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

% 三角形ABCの各頂点と、向かい合う辺の中点を結ぶ3本の直線は1点Gで交わる

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

% A(0, 0) B(u1, 0) C(u2, u3)

% D(x1, x2) E(x3, x4) F(x5, x6) G(x7, x8)

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

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

order x8, x7, x6, x5, x4, x3, x2, x1, u3, u2, u1;

factor x8, x7, x6, x5, x4, x3, x2, x1;

% 関数定義読み込み（※ローカル環境へコピーしてきたもの） --------------------------;

in cal\_sys\_relations$

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

% midpoint(x,a1,a2,b1,b2,c1,c2);

% if x=1 then f:=2\*b1-a1-c1

% else f:=2\*b2-a2-c2;

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

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

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

% D は AB の中点 ( A-D-B と考える ) --------------------------------------;

h1:=midpoint(1, 0, 0, x1, x2, u1, 0);

h2:=midpoint(2, 0, 0, x1, x2, u1, 0);

% E は BC の中点 ( B-E-C と考える ) --------------------------------------;

h3:=midpoint(1, u1, 0, x3, x4, u2, u3);

h4:=midpoint(2, u1, 0, x3, x4, u2, u3);

% F は CA の中点 ( C-F-A と考える ) --------------------------------------;

h5:=midpoint(1, u2, u3, x5, x6, 0, 0);

h6:=midpoint(2, u2, u3, x5, x6, 0, 0);

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

% collinear(a1,a2,b1,b2,c1,c2)$

% scalar a;

% a:=(a1-b1)\*(b2-c2)-(a2-b2)\*(b1-c1);

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

% A-G-E は一直線 ------------------------------------------------------;

h7:=collinear(0, 0, x7, x8, x3, x4);

% B-G-F は一直線 ------------------------------------------------------;

h8:=collinear(u1, 0, x7, x8, x5, x6);

% 結論

% C-G-D は一直線 ------------------------------------------------------;

g:=collinear(u2, u3, x7, x8, x1, x2);

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({x8, x7, 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, h8};

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

% 「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(g, gb);

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

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

% first(list), second(list)はそれぞれリスト内の1つ目、2つ目の要素を返す

% solve(f, v) は, 関数f の変数をv の方程式としてみて解く

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

% G座標も求めてみる ------------------------------------------------------;

solve(first(gb), x8);

solve(second(gb), x7);

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

showtime;

;end;

* 出力ファイル

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

% 三角形ABCの各頂点と、向かい合う辺の中点を結ぶ3本の直線は1点Gで交わる

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

% A(0, 0) B(u1, 0) C(u2, u3)

% D(x1, x2) E(x3, x4) F(x5, x6) G(x7, x8)

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

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

order x8, x7, x6, x5, x4, x3, x2, x1, u3, u2, u1;

factor x8, x7, x6, x5, x4, x3, x2, x1;

% 関数定義読み込み（※ローカル環境へコピーしてきたもの） --------------------------;

in cal\_sys\_relations$

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

% midpoint(x,a1,a2,b1,b2,c1,c2);

% if x=1 then f:=2\*b1-a1-c1

% else f:=2\*b2-a2-c2;

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

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

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

% D は AB の中点 ( A-D-B と考える ) --------------------------------------;

h1:=midpoint(1, 0, 0, x1, x2, u1, 0);

h1 := 2\*x1 - u1

h2:=midpoint(2, 0, 0, x1, x2, u1, 0);

h2 := 2\*x2

% E は BC の中点 ( B-E-C と考える ) --------------------------------------;

h3:=midpoint(1, u1, 0, x3, x4, u2, u3);

h3 := 2\*x3 - u2 - u1

h4:=midpoint(2, u1, 0, x3, x4, u2, u3);

h4 := 2\*x4 - u3

% F は CA の中点 ( C-F-A と考える ) --------------------------------------;

h5:=midpoint(1, u2, u3, x5, x6, 0, 0);

h5 := 2\*x5 - u2

h6:=midpoint(2, u2, u3, x5, x6, 0, 0);

h6 := 2\*x6 - u3

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

% collinear(a1,a2,b1,b2,c1,c2)$

% scalar a;

% a:=(a1-b1)\*(b2-c2)-(a2-b2)\*(b1-c1);

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

% A-G-E は一直線 ------------------------------------------------------;

h7:=collinear(0, 0, x7, x8, x3, x4);

h7 := - x8\*x3 + x7\*x4

% B-G-F は一直線 ------------------------------------------------------;

h8:=collinear(u1, 0, x7, x8, x5, x6);

h8 := - x8\*x5 + x8\*u1 + x7\*x6 - x6\*u1

% 結論

% C-G-D は一直線 ------------------------------------------------------;

g:=collinear(u2, u3, x7, x8, x1, x2);

g := - x8\*x1 + x8\*u2 + x7\*x2 - x7\*u3 - x2\*u2 + x1\*u3

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({x8, x7, 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, h8};

gb := { - 3\*x8 + u3,

3\*x7 - (u1 + u2),

2\*x6 - u3,

2\*x5 - u2,

2\*x4 - u3,

2\*x3 - (u1 + u2),

x2,

2\*x1 - u1}

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

% 「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;

{u2 + u1, - u2 + 2\*u1}

%%% gbを法としてgを簡約 --------------------------------------------------;

preduce(g, gb);

0

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

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

% first(list), second(list)はそれぞれリスト内の1つ目、2つ目の要素を返す

% solve(f, v) は, 関数f の変数をv の方程式としてみて解く

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

% G座標も求めてみる ------------------------------------------------------;

solve(first(gb), x8);

u3

{x8=----}

3

solve(second(gb), x7);

u2 + u1

{x7=---------}

3

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

showtime;

Time: 20 ms plus GC time: 40 ms

;

end;

# 演習01\_2：重心の証明プログラムを修正し、「垂心」の証明プログラムを実行せよ

* 入力ファイル

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

% 三角形ABCの各頂点から、向かい合う辺に下ろした3本の垂線は1点V(垂心)で交わる

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

% 三角形ABCを考える -----------------------------------------------------;

% A(0, 0) B(u1, 0) C(u2, u3)

% D(x1, x2) E(x3, x4) F(x5, x6)

% V(x7, x8)

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

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

% 出力時の変数の順序を定める宣言的記述 -------------------------------------;

order x8, x7, x6, x5, x4, x3, x2, x1, u3, u2, u1;

% 引数の順序には何の影響も与えないが、区切って出力させる宣言的記述 ----------------;

% その後の様々な操作をする際の前提としてこの宣言が必要になる場合がある --------------;

factor x8, x7, x6, x5, x4, x3, x2, x1;

% 関数定義読み込み（※ローカル環境へコピーした） -------------------------------;

in cal\_sys\_relations$

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

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

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

% 垂線: 直線ABと直線CDが垂直に交わるときの関数f

% vertically(a1,a2,b1,b2,c1,c2,d1,d2);

% f:=(a1-b1)\*(c1-d1)+(a2-b2)\*(c2-d2);

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

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

% 共線: 直線AB上に点Cがあるときの関数f

% collinear(a1,a2,b1,b2,c1,c2)$

% f:=(a1-b1)\*(b2-c2)-(a2-b2)\*(b1-c1);

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

% D は 頂点Cから対辺ABに下ろした垂線の足 ( A-D-B と考える ) -------------------;

% A(0, 0) B(u1, 0) C(u2, u3) D(x1, x2)

% このとき、「AB ⊥ CD」 かつ 「A, D, Bは共線 collinear」である ------------------;

s1:=vertically(0, 0, u1, 0, u2, u3, x1, x2);

s2:=collinear(0, 0, x1, x2, u1, 0);

% E は 頂点Aから対辺BCに下ろした垂線の足 ( B-E-C と考える ) -------------------;

% A(0, 0) E(x3, x4) B(u1, 0) C(u2, u3)

% このとき、「AE ⊥ BC」 かつ 「C, F, Aは共線 collinear」である -------------------;

s3:=vertically(0, 0, x3, x4, u1, 0, u2, u3);

s4:=collinear(u1, 0, x3, x4, u2, u3);

% F は 頂点Bから対辺CAに下ろした垂線の足 ( C-F-A と考える ) -------------------;

% C(u2, u3) A(0, 0) B(u1, 0) F(x5, x6)

% このとき、「CA ⊥ BF」 かつ 「C, F, Aは共線 collinear」である -------------------;

s5:=vertically(u2, u3, 0, 0, u1, 0, x5, x6);

s6:=collinear(u2, u3, x5, x6, u2, u3);

% C-V-D は一直線 -------------------------------------------------------;

% C(u2, u3) V(x7, x8) D(x1, x2)

s7:=collinear(u2, u3, x7, x8, x1, x2);

% A-V-E は一直線 -------------------------------------------------------;

% A(0, 0) V(x7 ,x8) E(x3, x4)

s8:=collinear(0, 0, x7, x8, x3, x4);

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

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

% B-V-F は一直線 -------------------------------------------------------;

% B(u1, 0) V(x7, x8) F(x5, x6)

v\_conclusion:=collinear(u1, 0, x7, x8, x5, x6);

showtime;

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

% Groebner Basis: -----------------------------------------------------;

%%% 変数を定義し、lex形式で並べる ------------------------------------------;

torder({x8, x7, 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{s1, s2, s3, s4, s5, s6, s7, s8};

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

% 「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v\_conclusionを簡約 --------------------------------------------------;

preduce(v\_conclusion, gb);

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

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

% first(list), second(list)はそれぞれリスト内の1つ目、2つ目の要素を返す

% solve(f, v) は, 関数f の変数をv の方程式としてみて解く

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

% G座標も求めてみる ------------------------------------------------------;

solve(first(gb), x8);

solve(second(gb), x7);

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

showtime;

;end;

* 出力ファイル

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

% 三角形ABCの各頂点から、向かい合う辺に下ろした3本の垂線は1点V(垂心)で交わる

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

% 三角形ABCを考える -----------------------------------------------------;

% A(0, 0) B(u1, 0) C(u2, u3)

% D(x1, x2) E(x3, x4) F(x5, x6)

% V(x7, x8)

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

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

% 出力時の変数の順序を定める宣言的記述 -------------------------------------;

order x8, x7, x6, x5, x4, x3, x2, x1, u3, u2, u1;

% 引数の順序には何の影響も与えないが、区切って出力させる宣言的記述 ----------------;

% その後の様々な操作をする際の前提としてこの宣言が必要になる場合がある --------------;

factor x8, x7, x6, x5, x4, x3, x2, x1;

% 関数定義読み込み（※ローカル環境へコピーした） -------------------------------;

in cal\_sys\_relations$

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

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

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

% 垂線: 直線ABと直線CDが垂直に交わるときの関数f

% vertically(a1,a2,b1,b2,c1,c2,d1,d2);

% f:=(a1-b1)\*(c1-d1)+(a2-b2)\*(c2-d2);

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

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

% 共線: 直線AB上に点Cがあるときの関数f

% collinear(a1,a2,b1,b2,c1,c2)$

% f:=(a1-b1)\*(b2-c2)-(a2-b2)\*(b1-c1);

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

% D は 頂点Cから対辺ABに下ろした垂線の足 ( A-D-B と考える ) -------------------;

% A(0, 0) B(u1, 0) C(u2, u3) D(x1, x2)

% このとき、「AB ⊥ CD」 かつ 「A, D, Bは共線 collinear」である ------------------;

s1:=vertically(0, 0, u1, 0, u2, u3, x1, x2);

s1 := x1\*u1 - u2\*u1

s2:=collinear(0, 0, x1, x2, u1, 0);

s2 := - x2\*u1

% E は 頂点Aから対辺BCに下ろした垂線の足 ( B-E-C と考える ) -------------------;

% A(0, 0) E(x3, x4) B(u1, 0) C(u2, u3)

% このとき、「AE ⊥ BC」 かつ 「C, F, Aは共線 collinear」である -------------------;

s3:=vertically(0, 0, x3, x4, u1, 0, u2, u3);

s3 := x4\*u3 + x3\*(u2 - u1)

s4:=collinear(u1, 0, x3, x4, u2, u3);

s4 := x4\*( - u2 + u1) + x3\*u3 - u3\*u1

% F は 頂点Bから対辺CAに下ろした垂線の足 ( C-F-A と考える ) -------------------;

% C(u2, u3) A(0, 0) B(u1, 0) F(x5, x6)

% このとき、「CA ⊥ BF」 かつ 「C, F, Aは共線 collinear」である -------------------;

s5:=vertically(u2, u3, 0, 0, u1, 0, x5, x6);

s5 := - x6\*u3 - x5\*u2 + u2\*u1

s6:=collinear(u2, u3, x5, x6, u2, u3);

s6 := 0

% C-V-D は一直線 -------------------------------------------------------;

% C(u2, u3) V(x7, x8) D(x1, x2)

s7:=collinear(u2, u3, x7, x8, x1, x2);

s7 := - x8\*x1 + x8\*u2 + x7\*x2 - x7\*u3 - x2\*u2 + x1\*u3

% A-V-E は一直線 -------------------------------------------------------;

% A(0, 0) V(x7 ,x8) E(x3, x4)

s8:=collinear(0, 0, x7, x8, x3, x4);

s8 := - x8\*x3 + x7\*x4

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

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

% B-V-F は一直線 -------------------------------------------------------;

% B(u1, 0) V(x7, x8) F(x5, x6)

v\_conclusion:=collinear(u1, 0, x7, x8, x5, x6);

v\_conclusion := - x8\*x5 + x8\*u1 + x7\*x6 - x6\*u1

showtime;

Time: 0 ms plus GC time: 20 ms

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

% Groebner Basis: -----------------------------------------------------;

%%% 変数を定義し、lex形式で並べる ------------------------------------------;

torder({x8, x7, 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{s1, s2, s3, s4, s5, s6, s7, s8};

2

gb := { - u3\*x8 + (u1\*u2 - u2 ),

x7 - u2,

- u3\*x6 - u2\*x5 + u1\*u2,

2 2 2 2

(u1 - 2\*u1\*u2 + u2 + u3 )\*x4 - (u1 \*u3 - u1\*u2\*u3),

2 2 2 2

(u1 - 2\*u1\*u2 + u2 + u3 )\*x3 - u1\*u3 ,

x2,

x1 - u2}

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

% 「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;

{u1,

u3,

- u2 + u1,

2 2 2

u3 + u2 - 2\*u2\*u1 + u1 }

%%% gbを法としてv\_conclusionを簡約 --------------------------------------------------;

preduce(v\_conclusion, gb);

0

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

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

% first(list), second(list)はそれぞれリスト内の1つ目、2つ目の要素を返す

% solve(f, v) は, 関数f の変数をv の方程式としてみて解く

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

% G座標も求めてみる ------------------------------------------------------;

solve(first(gb), x8);

2

- u2 + u2\*u1

{x8=----------------}

u3

solve(second(gb), x7);

{x7=u2}

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

showtime;

Time: 10 ms

;

end;

# 演習02：第一回授業での配布資におけるNo.4にあるプログラム内「Collinear」「Parallel」「Vertical」「Equal」について、点の配置と式の関係を解明せよ。（システム任せだとブラックボックスになってしまうので(計算内容を)可能な範囲で手で確かめること）