International Mathematical Forum, Vol. 11, 2016, no. 1, 1 - 9
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/imf.2016.5970

A Computer Program to Determine Projective Planes over Galois Fields[1]

Z. Akça, A. Bayar and M. M. Uzun

Eskişehir Osmangazi University, Faculty of Science and Arts
Department of Mathematics-Computer, Eskişehir, Turkey

Copyright © 2015 Z. Akça, A. Bayar and M. M. Uzun. This article is distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

In this paper, an algorithm and codes to construct a projective plane over Galois Field are introduced by using Matlab. Then an example of projective plane of order 3 is given.

Mathematics Subject Classification: 51E12, 51E15, 51E30

Keywords: Projective plane, Galois field

1 Introduction

A finite projective plane is a set of points, say $P = \{p_1, p_2, .., p_n\}$, and a set of lines, say $L = [l_1, l_2, ..., l_n]$, where

1) Any distinct two points are incident with just one line.
2) Any two lines are incident with at least one point.
3) There exists four points of which no three are collinear.

[1]This work is supported by the Eskişehir Osmangazi University under the project number 2013-269.

It is clear that every line contains the same number of points and every point lies on the same number of lines.

It is well known that every projective plane has also an algebraic structure obtained by coordinatization. Conversely, certain algebraic structures can be used to construct projective planes.

Given any field $F = GF(n)$ we can construct the set $PG(2, n) = (N, D, I)$ in the usual way: it is the set of equivalence classes $S/\sim GF(n) \setminus \{(0, 0, 0)\}$, where two non-zero vectors $x = (x_1, x_2, x_3)$ and $y = (y_1, y_2, y_3)$ of $GF(n)$ are called equivalent, denoted $x \sim y$, if there is a scalar $\lambda \in GF(n)^*$ such that $y_i = \lambda x_i$ for all $i = 1, 2, 3$. The "points" of projective plane $PG(2, n)$ are equivalence classes, but to identify them we need to use coordinates. To that end we need a set of representatives - one from each equivalence class. A popular way of achieving that is to select λ in such a way that the last non-zero coordinate is equal to 1. After all, if $x_3 \neq 0$, we have (with $\lambda = \frac{1}{x_3}$)

$$(x_1, x_2, x_3) \sim (\frac{x_1}{x_3}, \frac{x_2}{x_3}, 1).$$

If $GF(n)$ is finite field, say $|GF(n)| = n$, this means that there are $|N| = n^2 + n + 1$ points in $PG(2, n)$ as follows:

n^2 equivalence classes with representatives $(x, y, 1)$, $x, y \in GF(n)$ arbitrary.

n equivalence classes with representatives $(x, 1, 0)$, $x \in GF(n)$ arbitrary.

A single equivalence class with representative $(1, 0, 0)$.

Therefore each line contains $n+1$ points. An easy counting argument shows that each point lies on $n+1$ lines, and the number of points and lines is $|N| = n^2 + n + 1 = |D|$ In this case, one calls the pair (N, D) a projective plane of order n.

So in a way $PG(2, n)$ is the union of a "usual" (affine) plane, a line and a point. If $GF(3) = \{0, 1, 2\}$ then the $9 + 3 + 1 = 13$ elements are (the classes of) $P_1 = (0, 0, 1)$, $P_2 = (0, 1, 0)$, $P_3 = (0, 1, 1)$, $P_4 = (0, 1, 2)$, $P_5 = (1, 0, 1)$, $P_6 = (1, 0, 2)$, $P_7 = (1, 1, 0)$, $P_8 = (1, 1, 1)$, $P_9 = (1, 1, 2)$, $P_{10} = (1, 2, 0)$, $P_{11} = (1, 2, 1)$ $P_{12} = (1, 2, 2)$ and $P_{13} = (2, 0, 0)$. The equivalence relation means that we equate, for example, the point $(2, 1, 0)$ with the point $2(2, 1, 0) = (1, 2, 0) = P_{10}$ and the point $(2, 1, 1)$ with $2(2, 1, 1) = (1, 2, 2) = P_{12}$. In the field of 3 elements there are 2 non-zero constants, so in general 2 points of $GF(3)$ form a single equivalence class.

2 A Numerical Computation of Projective Planes of Order n

It is relatively easy to show that there exists a projective plane of order n where n is a prime number or a prime multiplied by itself a number of times. It is not known if there are projective planes of order 12, 15, 18, 20, and so on.

We want to provide a current list of known projective planes of order n by using Matlab. Firstly, we give an algorithm and then codes to find points, lines and incidence tables of projective planes of order n.

Algorithm

Our algorithm had been staged into 6 steps.

Step 1: Finding all possible points of projective plane of order n

This means we start a Galois field $GF(n) = \{0, 1, 2, ..., n-1\}$, where n is a prime number, and then we costruct two sets N and D including all triples (x_1, x_2, x_3), $x_1, x_2, x_3 \in GF(n)$ different from $(0, 0, 0)$

$$N = \{(x_1, x_2, x_3) \ : \ x_1, x_2, x_3 \in GF(n) \text{and } (x_1, x_2 x_3) \neq (0, 0, 0)\}$$

$$D = \left\{ \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} : \ y_1, y_2, y_3 \in GF(n) \text{ and } \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \neq \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right\}$$

and every different triple has labelled by the number a, $a \in \{1, 2, ...26\}$.

Step 2: Finding multiplication matrix $M = [m_{ij}]$

This means we find multiplication matrix $M = [m_{ij}]$, where

$$m_{ij} = x_1 y_1 + x_2 y_2 + x_3 y_3 \bmod(n),\ n \in GF(n)$$

for $i = (x_1, x_2, x_3) \in N$ and $j = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \in D$.

Step 3: Getting elements of M satisfying $m_{ij} = 0$

This means we select elements of M matrix which are equal to 0. These elements form a new set of group which are equivalent to each other in modules. We call new matrix as "E". In order to further reduce E matrix in proceeding step, we label each row of E as a character number which are obtained by multiplying all elements in a row. We call this set as C matrix.

Step 4: Filtering out equivalent rows of another matrix

This means we will base on the information from C matrix, we filter out equivalent rows from E matrix.

Step 5: Getting reduced matrix

This means we will go back to M matrix and select elements according to filtered rows in previous step. We call reduced M matrix as RM.

Step 6: Getting final matrix.

This means we will replace each "0" in RM matrix with "1" and set all other elements to "0". We call final matrix as F. So, we find the incidence matrix F of a finite projective plane of order n. The incidence matrix F of $PG(2,n)$ is the $(n^2+n+1)(n^2+n+1)$-matrix, where the lines are represented by the rows and the points are represented by the columns, such that row i has a 1 in column j if the line corresponding to row i contains the point corresponding to column j, and a 0 otherwise.

Example 1 *We will give codes results to construct a projective plane of order 3 over Galois field* $GF(3)$.

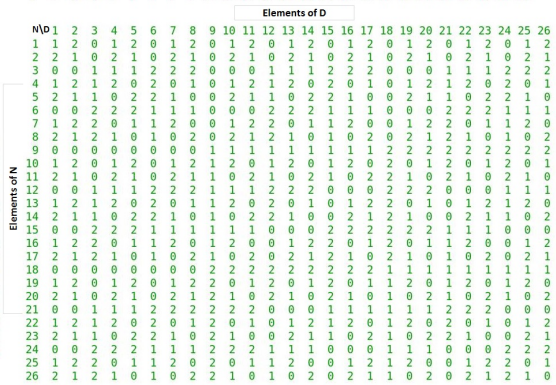
Step 1: n=3

```
x=0:n-1;
N=size(x,2);
c=0
for n1=1:N
for n2=1:N
for n3=1:N
c=c+1;
cellz=n1-1;celly=n2-1;cellx=n3-1;
confs_cell{n1,n2,n3}=[cellx celly cellz];
end
end
end
confs_cell=reshape(confs_cell,(n)^3,1);
for i=size(confs_cell,1)
points_all(i,1)=confs_cell{i}(1);
points_all(i,2)=confs_cell{i}(2);
points_all(i,3)=confs_cell{i}(3);
end
confs_size_all=size(points_all,1);
```

points_n = confs_cell;lines_n =confs_cell;
output_1 = points_all

1	(001)	6	(020)	11	(102)	16	(121)	21	(210)	26	(222)
2	(002)	7	(021)	12	(110)	17	(122)	22	(211)		
3	(010)	8	(022)	13	(111)	18	(200)	23	(212)		
4	(011)	9	(100)	14	(112)	19	(201)	24	(220)		
5	(012)	10	(101)	15	(120)	20	(202)	25	(221)		

Step 2: multiplication_matrix=zeros(size(lines_n,1) size(points_n,1));
for d=1:size(lines_n,1)
for n=1:size(points_n,1)
multiplication_matrix(d,n)=mod((lines_n{d}(1)*points_{n}(1)+...
lines_n{d}(2)*points_n{n}(2)+lines_n{d}*points_n{n}(3)),(n));
end
end
multiplication_matrix(2:size(points_all,1),1)=(1:1:size(points_all,1)-1';
multiplication_matrix(1,2:size(points_all,1))=(1:1:size(points_all,1)-1;output_2 =
multiplication_matrix



Step 3:
kcell1; for kr=2:size(multiplication_matrix,1);
nn=1;
for kc=2:size(multiplication_matrix,1);
if multiplication_matrix(kr,kc)==0
intersections(nn)=kc-1;
nn=nn+1;
end
end

```
crossings_cell{kcell}=[intersections];
product=1
for mc=1:size(crossings_cell){kcell},2);
product=product*crossings_cell{kcell}(mc);
end
characters_cell(kcell)=mod(product,100001);
kcell=kcell+1;
intersections= 0;
end
output_3a=(1:size(crossings_cell,2))';
for r=1:((n)^3-1)
output_3b(r,:)=crossings_cell{r};
end
output_3c=characters_cell';
output_3 = [output_3a output_3b output_3c]
```

Step 4:

```
fileID=fopen('full_matrix.txt','w');
    for row=2:size(multiplication_matrix,1);
    fprintf(fileID,'%3.0f ',multiplication_matrix(row,1));
    fprintf(fileID,'%6.0f ',characters_cell(row-1));
    fprintf(fileID,'(%1.0f%1.0f%1.0f) ',points_n{row}(1),points_n{row}(2),points_n{row}(3));
    fprintf(fileID,' ');
    for col=2:size(multiplication_matrix,1);
        fprintf(fileID,'%1.0f ',multiplication_matrix(row,col));
    end
    fprintf(fileID,'\n');
    end
    ! sort -nk 2,2 full_matrix.txt > filtered.txt
    ! cat filtered.txt | awk '{print $2}' | uniq > temp1.dat
    ! > temp2.dat ;
    ! for i in 'cat temp1.dat'; do grep $i filtered.txt | head -1 >> temp2.dat;
done
    ! cat temp2.dat | sort -nk 1,1 > temp3.dat
    ! cat temp3.dat | awk '{print $1}' > temp4.dat
    reduced=load('temp4.dat');
    ! rm -f temp1.dat temp2.dat temp3.dat temp4. filtered.txt output_4=reduced
```

```
 1  3  6   9  12  15  18  21  24    36875
 2  3  6   9  12  15  18  21  24    36875
 3  1  2   9  10  11  18  19  20    43065
 4  5  7   9  14  16  18  23  25    88698
 5  4  8   9  13  17  18  22  26    13255
 6  1  2   9  10  11  18  19  20    43065
 7  4  8   9  13  17  18  22  26    13255
 8  5  7   9  14  16  18  23  25    88698
 9  1  2   3   4   5   6   7   8    40320
10  3  6  11  14  17  19  22  25    40876
11  3  6  10  13  16  20  23  26    77923
12  1  2  15  16  17  21  22  23     7293
13  5  7  11  13  15  19  21  26    20262
14  4  8  10  14  15  20  21  25    92945
15  1  2  12  13  14  24  25  26    40119
16  4  8  11  12  16  19  23  24    13904
17  5  7  10  12  17  20  22  24    76461
18  1  2   3   4   5   6   7   8    40320
19  3  6  10  13  16  20  23  26    77923
20  3  6  11  14  17  19  22  25    40876
21  1  2  12  13  14  24  25  26    40119
22  5  7  10  12  17  20  22  24    76461
23  4  8  11  12  16  19  23  24    13904
24  1  2  15  16  17  21  22  23     7293
25  4  8  10  14  15  20  21  25    92945
26  5  7  11  13  15  19  21  26    20262
```

Step 5:

```
fileID=fopen('reduced_matrix.txt','w');
reduced=reduced+1;
for i=1:size(reduced,1);
fprintf(fileID,'%3.0f ',multiplication_matrix(reduced(i),1));
fprintf(fileID,'%6.0f ',characters_cell(reduced(i)-1));
fprintf(fileID,'(%1.0f%1.0f%1.0f) ',points_n{reduced(i)}(1),...
points_n{reduced(i)}(2),points_n{reduced(i)}(3));
fprintf(fileID,' ');
for col=reduced;
fprintf(fileID,'%1.0f
',multiplication_matrix(reduced(i),col));
end
fprintf(fileID,'\n');
end
fileID=fopen('tmp.txt','w');
for i=1:seize(reduced,1)
for col=reduced;
fprintf(fileID,'%1.0f
',multiplication_matrix(reduced(i),col));
end
fprintf(fileID,'\n'); end
```

```
 1   (001)    1 0 1 2 1 2 0 1 2 0 1 2 0
 3   (010)    0 1 1 1 0 0 1 1 1 2 2 2 0
 4   (011)    1 1 2 0 1 2 1 2 0 2 0 1 0
 5   (012)    2 1 0 2 2 1 1 0 2 2 1 0 0
10   (101)    1 0 1 2 2 0 1 2 0 1 2 0 2
11   (102)    2 0 2 1 0 2 1 0 2 1 0 2 2
12   (110)    0 1 1 1 1 1 2 2 2 0 0 0 2
13   (111)    1 1 2 0 2 0 2 0 1 0 1 2 2
14   (112)    2 1 0 2 0 2 2 1 0 0 2 1 2
15   (120)    0 2 2 2 1 1 0 0 0 2 2 2 2
16   (121)    1 2 0 1 2 0 0 1 2 2 0 1 2
17   (122)    2 2 1 0 0 2 0 2 1 2 1 0 2
18   (200)    0 0 0 0 2 2 2 2 2 2 2 2 1
```

Step 6:

```
I=load('tmp.txt'); ! rm tmp.txt [m,n]=size(I); S=zeros(m,n); for i=1:m
for j=1:n
if I(i,j) == 0
S(i,j)=1;
elseif I(i,j) ~= 0
S(i,j)=0;
end
end
end
fileID=fopen('final_matrix.txt','w');
for i=1:size(S,1);
fprintf(fileID,' ');
for col=1:size(S,2);
fprintf(fileID,'%1.0f ',S(i,col));
end
fprintf(fileID,'\n');
end
```

1	(001)	1 0 1 2 1 2 0 1 2 0 1 2 0	0 1 0 0 0 0 1 0 0 1 0 0 1
3	(010)	0 1 1 1 0 0 1 1 1 2 2 2 0	1 0 0 0 1 1 0 0 0 0 0 0 1
4	(011)	1 1 2 0 1 2 1 2 0 2 0 1 0	0 0 0 1 0 0 0 0 1 0 1 0 1
5	(012)	2 1 0 2 2 1 1 0 2 2 1 0 0	0 0 1 0 0 0 0 1 0 0 0 1 1
10	(101)	1 0 1 2 2 0 1 2 0 1 2 0 2	0 1 0 0 0 1 0 0 1 0 0 1 0
11	(102)	2 0 2 1 0 2 1 0 2 1 0 2 2	0 1 0 0 1 0 0 1 0 0 1 0 0
12	(110)	0 1 1 1 1 1 2 2 2 0 0 0 2	1 0 0 0 0 0 0 0 0 1 1 1 0
13	(111)	1 1 2 0 2 0 2 0 1 0 1 2 2	0 0 0 1 0 1 0 1 0 1 0 0 0
14	(112)	2 1 0 2 0 2 2 1 0 0 2 1 2	0 0 1 0 1 0 0 0 1 1 0 0 0
15	(120)	0 2 2 2 1 1 0 0 0 2 2 2 2	1 0 0 0 0 0 1 1 1 0 0 0 0
16	(121)	1 2 0 1 2 0 0 1 2 2 0 1 2	0 0 1 0 0 1 1 0 0 0 1 0 0
17	(122)	2 2 1 0 0 2 0 2 1 2 1 0 2	0 0 0 1 1 0 1 0 0 0 0 1 0
18	(200)	0 0 0 0 2 2 2 2 2 2 2 2 1	1 1 1 1 0 0 0 0 0 0 0 0 0

In forthcoming papers, we will develop algoritms and codes for some subjects of fiber and fuzzy projective geometries in $[1-5]$ by using Matlab.

References

[1] Z. Akça, A. Bayar, S. Ekmekçi, H. Van Maldeghem, Fuzzy Projective Spreads of Fuzzy Projective Spaces, *Fuzzy Sets and Systems*, **157** (2006), no. 24, 3237-3247. http://dx.doi.org/10.1016/j.fss.2006.08.005

[2] Z. Akça, A. Bayar, S. Ekmekçi, On the Classification of Fuzzy Projective Lines of Fuzzy 3-Dimensional Projective Space, *Communications, Faculty Of Science, University of Ankara Series A1Mathematics and Statistics*, **55** (2006), no. 2, 17-23.
http://dx.doi.org/10.1501/commua1_0000000319

[3] A. Bayar, S. Ekmekçi, Z. Akça, A Note on Fibered Projective Plane Geometry, *Information Sciences*, **178** (2008), 1257-1262.
http://dx.doi.org/10.1016/j.ins.2007.10.003

[4] S. Ekmekçi, Z. Akça, A. Bayar, On the Classification of Fuzzy Projective Planes of Fuzzy 3-Dimensional Projective Space, *Chaos, Solitons and Fractals*, **40** (2009), 2146-2151.
http://dx.doi.org/10.1016/j.chaos.2007.10.002

[5] S. Ekmekçi, A. Bayar, Z. Akça, Fuzzy Ellipsoidal Projective Plane Model, *International Journal of Physical Sciences*, **6** (2011), no. 7, 1808-1813.

[6] J.W.P. Hirschfeld, *Projective Geometries over Finite Fields*, second edition, Oxford University Press, Oxford, 1998.

Received: September 18, 2015; Published: November 12, 2015