A Computer Program to Determine Projective Planes over Galois Fields

Z. Akça, A. Bayar and M. M. Uzun

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

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 $\lambda$ in such a way that the last non-zero coordinate is equal to 1. After all, if $x_3 \neq 0$, we have (with $\lambda = \frac{1}{x_3}$)

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

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 \( GF(n) = \{0, 1, 2, \ldots, n-1\} \), where \( n \) is a prime number, and then we construct 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 \( a, a \in \{1, 2, \ldots, 26\} \).

**Step 2:** Finding multiplication matrix \( M = [m_{ij}] \)

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

\[
m_{ij} = x_1y_1 + x_2y_2 + x_3y_3 \mod(n), \ n \in GF(n)
\]

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:** $n=3$

```matlab
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
cons_cell=reshape(confs_cell,(n)ˆ3, 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
cons_size_all=size(points_all,1);
```
points_n = confs_cell; lines_n = confs_cell;
output_1 = points_all

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>001</td>
<td>6</td>
<td>020</td>
<td>11</td>
<td>102</td>
</tr>
<tr>
<td>2</td>
<td>002</td>
<td>7</td>
<td>021</td>
<td>12</td>
<td>110</td>
</tr>
<tr>
<td>3</td>
<td>010</td>
<td>8</td>
<td>022</td>
<td>13</td>
<td>111</td>
</tr>
<tr>
<td>4</td>
<td>011</td>
<td>9</td>
<td>100</td>
<td>14</td>
<td>112</td>
</tr>
<tr>
<td>5</td>
<td>012</td>
<td>10</td>
<td>101</td>
<td>15</td>
<td>120</td>
</tr>
<tr>
<td>6</td>
<td>013</td>
<td>11</td>
<td>102</td>
<td>16</td>
<td>210</td>
</tr>
<tr>
<td>7</td>
<td>014</td>
<td>12</td>
<td>103</td>
<td>17</td>
<td>211</td>
</tr>
<tr>
<td>8</td>
<td>020</td>
<td>13</td>
<td>110</td>
<td>18</td>
<td>200</td>
</tr>
<tr>
<td>9</td>
<td>021</td>
<td>14</td>
<td>111</td>
<td>19</td>
<td>201</td>
</tr>
<tr>
<td>10</td>
<td>022</td>
<td>15</td>
<td>112</td>
<td>20</td>
<td>202</td>
</tr>
</tbody>
</table>

**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{n}{1}+... lines_n{d}{2)*points_n{n}{2}+lines_n{d}{3})*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)ˆ3-1
    output_3b(r,:)=crossings_cell{r};
end
output_3c=characters_cell’;
output_3 = [output_3a output_3b output_3c]

Step 4:

userID=fopen(‘full_matrix.txt’,’w’);
for row=2:size(multiplication_matrix,1);
    fprintf(userID,’%3.0f ‘,multiplication_matrix(row,1));
    fprintf(userID,’%6.0f ‘,characters_cell(row-1));
    fprintf(userID,’(%1.0f%1.0f%1.0f) ‘,points_n{row}(1),points_n{row}(2),points_n{row}(3));
    fprintf(userID,’ ‘);
    for col=2:size(multiplication_matrix,1);
        fprintf(userID,’%1.0f ‘,multiplication_matrix(row,col));
    end
    fprintf(userID,’\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
Step 5:

```matlab
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:size(reduced,1)
    for col=reduced;
        fprintf(fileID,'%1.0f ',multiplication_matrix(reduced(i),col));
    end
    fprintf(fileID,\n');end
```

```
1  (001)  1012120120120120
3  (010)  0111011112220
4  (011)  1120121202010
5  (012)  2012212212210
10  (101)  1012201201202
11  (102)  2021021021022
12  (110)  01111122222222
13  (111)  1120202010122
14  (112)  2102022100212
15  (120)  2221100022222
16  (121)  12012001222222
17  (122)  2210002121022
18  (200)  00002222222221
```
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;
endif
end
end

fileID=fopen('final_matrix.txt','w');
for i=1:size(S,1);
fprintf(fileID,'%1.0f ',S(i,col));
end
fprintf(fileID,'\n');
end

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

References


http://dx.doi.org/10.1501/commua1_0000000319
http://dx.doi.org/10.1016/j.ins.2007.10.003

http://dx.doi.org/10.1016/j.chaos.2007.10.002


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