Mathematical & Computational Applications, Vol. 3, No. 1, pp. 59-65, 1998

PEIFFER COMMUTATORS BY USING GAP PACKAGE

Ali MUTLU
University of Celal Bayar, Faculty of Science,
Department of Mathematics, Muradiye Campus,
45030, Manisa, TURKEY

Abstract- We describe Peiffer commutators within the Moore complex **NG** of a simplicial group **G**. The calculation of the Peiffer commutators is made by using **GAP** program.

1. INTRODUCTION

Simplicial groups play an important role in homological groups, homotopy theory, algebraic K-theory and geometry. In each sector they have played a signification part in developments over quite a lengthy period of time and there is an extensive literature on their homotopy theory. In homotopy theory itself, they model all connected homotopy types and allow analysis of features of such homotopy types by a combination of group theoretic methods and tools from combinatorial homotopy theory. Simplicial groups have the natural structure of Kan complexes and so are potentially models for weak infinity categories. The present article intends to study n-types of simplicial groups.

In an interesting recent work, Mutlu and Porter [7] proved the following theorem: Let **G** be a simplicial group with Moore complex **NG** and for $n \geq 0$ let D_n be a subgroup generated by the degenerate elements in dimensions n. Then

$$NG_n \cap D_n = N_n \qquad n \geq 1,$$

where N_n is normal subgroup in G_n generated by an explicitly given set of elements. Mutlu and Porter also in [8] generalised a result of Brown-Loday in [5] and a give for n=3 and 4 a decomposition of the group, $d_n NG_n$ of boundaries of a simplicial group **G** as a product of commutator subgroups. Partial results are given for higher dimension.

The main points of the paper are thus:

(i) to define the Peiffer commutators in higher dimensions,

(ii) to give calculations of Peiffer commutators by computer group packages.

2. DEFINITION AND NOTATION

A simplicial group **G** consists of a family of groups G_n together with face and degeneracy maps $d_i = d_i^n : G_n \to G_{n-1}$, $0 \leq i \leq n$, $n \neq 0$ and $s_i = s_i^n : G_n \to G_{n+1}$ $0 \leq i \leq n$, satisfying the usual simplicial identities given in [6] and also [3]. Another essential reference from our point of view is Carrasco's thesis, [4], where many of the basic techniques used here were developed systematically for the first time and the notion of hypercrossed complex was defined.

2.1 The surjective maps

The following notation and terminology is derived from [4] and the published version, [3], of the analogous group theoretic case.

For the ordered set [n] = { 0 < 1 < ... < n}, let $\alpha_i^n\colon[n+1]\to[n]$ be the increasing surjective map given by

$$\alpha_i^n(j)=\begin{cases} j & if \quad j\le i \\ j-1 & if \quad j>i. \end{cases}$$

Let $S(n,n-l)$ be the set of all monotone increasing surjective maps from **[n]** to **[n-*l*]**. This can be generated from the various α_i^n by composition. The composition of these generating maps satisfies the rule $\alpha_j\alpha_i=\alpha_{i-1}\alpha_j$ with j<i. This implies that every element $\alpha\in S(n,n-l)$ has a unique expression as $\alpha=\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_l}$ with $0\le i_1\le i_2\le\ldots\le i_l\le n$, where the indices i_k are the elements of **[n]** at which $\alpha(i)=\alpha(i+1)$. We thus can identify $S(n,n-l)$ with the set $\{(i_l,\ldots,i_1) : 0\le i_1\le i_2\le\ldots\le i_l\le n-1\}$. In particular the single element of $S(n,n)$, defined by the identity map on **[n]**, corresponds to the empty 0-tuple () denoted by $\varnothing_n$. Similarly the only element of $S(n,0)$ is (n-1, n-2,...0). For all $n\ge 0$, let

$$S(n)=\bigcup_{0\le l\le n}S(n,n-l).$$

We say that $\alpha=(i_l,\ldots,i_1)<\beta=(j_m,\ldots,j_1)$ in S(n)

$$\begin{array}{lll} if \quad i_1=j_1,\ldots,i_k=j_k & but \quad i_{k+1}>j_{k+1} & (k>0) \quad or \\ if \quad i_l=j_l,\ldots,i_1=j_1 & and \quad l<m. & \end{array}$$

This makes S(n) an ordered set. For instance, the orders of S(2) and S(3) and S(4) are respectively:

S(2) = { $\varnothing_2$ < (1) < (0) <(1 , 0) },
S(3) = {$\varnothing_3$ < (2) < (1) < (2 , 1) < (0) < (2 , 0) < (1 , 0) < (2 , 1 , 0) },
S(4) = { $\varnothing_4$ < (3) < (2) < (3 , 2) < (1) < (3 , 1) < (2 , 1) < (3 , 2 , 1) < (0) < (3 , 0) < (2 , 0) < (3 , 2 , 0) < (1 , 0) < (3 , 1 , 0) < (2 , 1 , 0) < (3 , 2 , 1 , 0) }.

If α and β are in S(n), we define $\alpha\cap\beta$ to be the set of indices which belong to both α and β .

2.2 The Moore complex

The Moore complex **NG** of a simplicial group **G** is defined to be the normal chain complex (**NG**,∂) with **NG**$_n$ $=\bigcap_{i=0}^{n-1} Kerd_i$ and with differential $\partial_n : NG_n \to NG_{n-1}$ induced from d_n by restriction.

The Moore complex has the advantage of being smaller than the simplicial group itself and being a chain complex is of a better known form for manipulation. However being non-abelian in general, some new techniques one needs to develope need developing for its study. Its homology gives the homotopy groups of the simplicial group and thus in specific cases, e.g. a truncated free simplicial resolution of a group, gives valuable higher dimensional information on elements.

The Moore complex, **NG**, carries a hypercrossed complex structure (see [3] and [4]) which allows the original **G** to be rebuilt.

2.3 Peiffer commutators generate

In the following we define Peiffer commutators of G_n. First of all we introduce a method to get the construction of a useful family of pairings. We describe a set P(n) consisting of pairs of elements (α,β) from S(n) with $\alpha \cap \beta = \emptyset$ and $\alpha \prec \beta$, where $\alpha = (i_l,\ldots,i_1)$, $\beta = (j_m \ldots j_1) \in S(n)$. The Peiffer commutators

$$F_{\alpha,\beta} : NG_{n-\bullet\alpha} \, x \, NG_{n-\bullet\beta} \to NG_n : (\alpha,\beta) \in P(n) \;\; n \geq 0$$

that we need are given as composites by the diagram where

$$\begin{array}{ccc} NG_{n-\bullet\alpha} \, x \, NG_{n-\bullet\beta} & \to & NG_n \\ \downarrow & & \uparrow \\ G_n \, x \, G_n & \to & G_n \end{array}$$

where

$$s_\alpha = s_{i_l} \ldots s_{i_1} : NG_{n-\bullet\alpha} \to G_n, \quad s_\beta = s_{j_m} \ldots s_{j_1} : NG_{n-\bullet\beta} \to G_n,$$

$p: G_n \to NG_n$ is defined by the composite projections $p(z) = p_{n-1} \ldots p_0(z)$, where $p_j(z) = zs_jd_j(z)^{-1}$ $j = 0,1,\ldots, n-1$ and $\mu : G_n x G_n \to G_n$ is given by the commutator map. Thus

$$\begin{aligned} F_{\alpha,\beta}\left(x_\alpha, y_\beta\right) &= p\mu(s_\alpha x s_\beta)(x_\alpha, y_\beta) \\ &= p[s_\alpha(x_\alpha), s_\beta(y_\beta)] \end{aligned}$$

We now describe the Peiffer commutators of G_n as generated by elements of the form

$$F_{\alpha,\beta}(x_\alpha, y_\beta)$$

where $x_\alpha \in NG_{n-\bullet\alpha}$ and $y_\beta \in NG_{n-\bullet\beta}$ where $\bullet\alpha$ is the length of the string α. We illustrate the Peiffer commutator for n = 2 and 3, to show what it looks like.

Example: For n = 2, suppose α = (1), β = (0) and $x, y \in NG_1 = Kerd_0$. It follows that

$$F_{\alpha,\beta}(x,y) = p_1 p_0 [s_0(x), s_1(y)]$$

$$= p_1[s_0(x), s_1(y)]$$

$$= [s_0(x), s_1(y)]\,[s_1(y), s_1(x)]$$

which is corresponding to crossed module but we omit it this paper. In this example, the Peiffer commutator is generated by elements of the form

$$F_{\alpha,\beta}(x,y) = [s_0(x), s_1(y)]\,[s_1(y), s_1(x)].$$

The imagine of $F_{(0)(1)}(x,y)$ is the Peiffer element determined by x and y, we will call the $F_{\alpha,\beta}(x_\alpha, y_\beta)$ in higher dimensions *higher dimensional Peiffer commutators.*

For n=3, the pairings are as following $F_{(1,0)(2)}$, $F_{(2,0)(1)}$, $F_{(0)(2,1)}$, $F_{(0)(2)}$, $F_{(1)(2)}$, $F_{(0)(1)}$. For $x \in NG_1$, $y \in NG_2$ the corresponding generators of the Peiffer commutators are:

$$F_{(1,0)(2)}(x,y) = [s_1 s_0(x), s_2(y)]\,[s_2(y), s_2 s_0(x)],$$

$$F_{(2,0)(1)}(x,y) = [s_2 s_0(x), s_1(y)]\,[s_1(y), s_2 s_0(x)]\,[s_2 s_1(x), s_2(y)]\,[s_2(y), s_2 s_0(x)]$$

and all $x \in NG_2$, $y \in NG_1$

$$F_{(0)(2,1)}(x,y) = [s_0(x), s_2 s_1(y)]\,[s_2 s_1(y), s_1(x)]\,[s_2(x), s_2 s_1(y)],$$

whilst for all $x, y \in NG_2$,

$$F_{(0)(1)}(x,y) = [s_0(x), s_1(y)]\,[s_1(y), s_1(x)]\,[s_2(x), s_2(y)],$$

$$F_{(0)(2)}(x,y) = [s_0(x), s_2(y)],$$

$$F_{(1)(2)}(x,y) = [s_1(x), s_2(y)]\,[s_2(y), s_2(x)]$$

which is corresponding to 2-crossed modules, but it is omitted here.

2.4 The case n=4

With dimension 4, the situation is still manageable. We firstly n = 4 omit here the key pairings $F_{\alpha,\beta}(x_\alpha, y_\beta)$.

3. THE GAP PROGRAM FOR PEIFFER COMMUTATORS

In this section we have used the **Groups, Algorithms and Programming (GAP)** package, which is a computational group package. We refer [2] for details. We give calculations $F_{\alpha,\beta}(x_\alpha, y_\beta)$ for higher values n using **GAP.** We introduce formulae for the number of pairs (α, β).

3.1 The GAP program listing for $F_{\alpha,\beta}$

```
Display := function(LCL);
local lc,nc,C,C2,i,jk,str;
str := ["s0","s1","s3" ];
  nc := Length(LCL);
  for j in [1..nc] do
     Print("[");
     C := LCL[j]; C1 := C[1]; C2 := C[2];
     lc := length(C1);
     for i in [1..lc-1] do
       k := C1[i]+1;
       Print(str[k]);
       od ;
     Print(C1[lc], ",");
     lc := Length(C2);
     for i in [1..lc-1] do
       k := C2[i]+1;
       Print(str[k]);
       od;
     Print(C2[lc], "] ");
     od;
   Print("\n");
 end;

 Sdpass := function(k,L)
   local len, p1,p2;
   p1 := Position(L,k);
   # Print("SDpass(k,L), p1: ", k, ", ", L, ", ", p1, ", ");
   if (p1=false) then
     p2 := Position(L,k-1);
```

```
  # Print("p2= ", p2, ", ");
  if (p2=false) then
    L:=[ ];
  else
    L[p2]:= k;
  fi;
 fi;
 # Print(" - > ", L, "\n");
 return L;
 end;

Projection := function(k,LCL)
 local LCL2, j, C, c1, c2, nc, j, C1, C2, L1, L2   ;
 LCL2 := Copy(LCL);
 nc, := Length(LCL);   # = number of commutators
 for j in Reversed([1..nc]) do
   C := LCL[j];        # reverse the order for the inverse of a product
   C1 := C[2];         # the inverse of a commutator [u,v]
   C2 := C[1];         # is the reversed commutator [v,u]
   L1 := SDpass(k,C1);
   L2 := Sdpass(k,C2);
   if ((L1 <> [ ] ) and (L2 <> [ ])) then
   Add(LCL2, [L1,L2]);
   Print("adding", [L1,L2], "\n");
   fi;
 od;
 return LCL2;
 end;

FIJxy := function(n,I,J)
 local LCL,LI,LJ,i;
 Print ( "\nFIJxy in dimension ", n, " with I,J = ", I, ", ", J, "\n");
 LI := Copy(I);
 Add(LI,"x");
 LJ := Copy(J);
 Add(LJ,"y");
 # create initial list of 'commutators' of lists
 # where a 'commutators' is a two-element list
 LCL := [ [LI, LJ] ];
 Print( "Initial  LCL = \n" );
 Display(LCL);
 for i in [0...(n-1)] do
   Print("Projecting in dimension ", i, "\n");
   LCL := Projection(i,LCL);
 od;
    Print("Final  LCL = \n");
```

Display(LCL);

end;

REFERENCES

[1] A. Mutlu, Peiffer Pairings in the Moore Complex of a Simplicial Group, *Ph.D. Thesis*, University of Wales Bangor, (1997); *Bangor* Preprint 97.11.

[2] Schönert, M. et. al, GAP: Groups, Algorithms, and programming, Lehrstuhl D für Matematik, Rheinisch Westfälische Technische Hoch-schule, Aachen, Germay, third edition, 1993.

[3] P. Carrasco and A. M. Cegarra, Group-theoretic Algebraic Models for Homotopy Types, *Jour. Pure Appl. Algebra,* **75**, 195-235, (1991).

[4] P. Carrasco, Complejos Hipercruzados, Cohomologia y Extensiones, *Ph.D. Thesis*, Universidad de Granada, (1987).

[5] R. Brown and J.-L. Loday, Van Kampen Theorems for Diagram of Spaces, *Topology*, **26**, 311-335, (1987).

[6] E. B. Curtis, Simplicial Homotopy Theory, *Adv. in Math.*, **6**, 107- 209, 1971.

[7] A. Mutlu and T. Porter, Iterated Peiffer pairings in the Moore complex of a simplicial group, *Trans. Amer. Math. Soc.* (summitted).

[8] A. Mutlu and T. Porter, Applications Peiffer pairings in the Moore complex of a simplicial group, *Theor. App. Cat.* (summitted)