An algorithm for computing the first homology groups of CDGAs with linear differential.

V. Álvarez [1,2,3], J.A. Armario [1,2], M.D. Frau [1], P. Real [1,2] and B. Silva [1,2]

Departamento de Matemática Aplicada I, Universidad de Sevilla, Spain.
E-mails: valvarez@cica.es, armario@cica.es, lfrau@euler.fie.us.es, real@cica.es

Abstract

We design here a primary platform for computing the basic homological information of Commutative Differential Graded Algebras (briefly, CDGAs), endowed with linear differential. All the algorithms have been implemented in the framework settled by *Mathematica*, so that we can take advantage of the use of symbolic computation and many other powerful tools this system provides.

Working in the context of CDGAs, Homological Perturbation Theory ([8], [9],[17]) supplies immediately a general algorithm computing the 1-homology of these objects at graded module level. But this process, already sketched by Lambe in [12], often bears high computational charges and actually restricts its application to the low dimensional homological calculus.

Although we already know the work of Lambe on **Axiom** (earlier known as "Scratchpad", [11],[13]), our goal is to implement new functions on *Mathematica*, starting from the theoretical approach settled in [2].

This paper is devoted to design and analyze a particular *Mathematica* "package" for computing the basic homological information of CDGAs endowed with linear differential.

What we do here, indeed, is to compute the differential operator and not the homology itself, actually. The full homological information may be reached by only applying Veblen's algorithm (see [19]). In this sense, it may be an

[1] Partially supported by PAICYT n. FQM-0143, Spain
[2] Partially supported by project DGYCI n. PB97-1025-C02-02, Spain
[3] Supported by a grant from the Junta de Andalucía

interesting effort to implement a routine for computing the Smith normal form of a given matrix, which was not our goal here.

Nevertheless, homological information is available by knowing the differential operator.

All the functions and formulas below may be implemented in different notebooks, so that each of them can be independently executed.

The paper is organized as follows. In Section 1, we give some preliminaries of Differential Homological Algebra. This section is just necessary for getting a basic knowledge about what a homological model of a CDGA is, and how to compute its homology groups.

Sections 2 to 8 are devoted to construct the seven elemental functions which lead to define the differential operator on the homological model. From this data, we may define the function computing that differential operator, in the same way it is done in [1].

Last section includes the full code of the programme.

1 Preliminaries

Our setting is that of Differential Homological Algebra ([4],[15]).

First of all, it seems quite natural to briefly explain what we understand by *homology* of a CDGA.

Given a differential graded module (M, d_M), the notion of *homology* of M refers to the graded module defined by the successive quotient modules

$$H_n(M) = \text{ Ker } (d_{M_n}) \ / \ \text{Im } (d_{M_{n+1}}).$$

This make sense since the operator d_M is assumed to be nilpotent of second order by definition.

It is well known that the notion of homology of a CDGA refers to the homology of its associated *Bar construction*, as it was already characterized by Mac Lane (see [15]).

We next recall what the reduced Bar construction $\bar{B}(A)$ of an algebra A is.

Given a DG-algebra A, we define a new differential graded module, $\bar{B}(A)$,

such that its elements are considered as lists of different length (covering the non negative integers) with entries elements of A or scalars of the ground ring Λ.

Indeed, the Bar construction $\bar{B}(A)$ is nothing more that the module obtained by considering the direct sum of the successive tensor products of copies of A,

$$\Lambda \oplus A \oplus (A \otimes A) \oplus (A \otimes A \otimes A) \oplus \cdots$$

The element of $\bar{B}_0(A)$ corresponding to the identity element of Λ is denoted simply by [ ] (the empty list). The element $a_1 \otimes \cdots \otimes a_n$ of $\bar{B}(A)$ is denoted by $[a_1|\cdots|a_n]$. The tensor and simplicial degrees of an element $[a_1|\cdots|a_n]$ are $|[a_1|\cdots|a_n]|_t = \sum |a_i|$ and $|[a_1|\cdots|a_n]|_s = n$, respectively. The total degree is the sum of both the tensor and simplicial degrees.

The differential structure lies on the addition of two morphisms, the tensor (d_t) and simplicial (d_s) differentials:

$$d_t([a_1|\cdots|a_n]) \;=\; -\textstyle\sum_i (-1)^{e_i-1}[a_1|\cdots|d_A(a_i)|\cdots|a_n],$$

and

$$d_s([a_1|\cdots|a_n]) \;=\; \textstyle\sum_i (-1)^{e_i}[a_1|\cdots|\mu_A(a_i \otimes a_{i+1}|\cdots|a_n]$$

where

$$e_i = i + |a_1| + \cdots + |a_i|.$$

A product $*$ (called *shuffle product*) can be defined on the Bar construction $\bar{B}(A)$ associated to an associative algebra, such that the reduced bar construction has a Hopf algebra structure.

The shuffle product obeys the following rule:

$$[a_1|\cdots|a_m] * [b_1|\cdots|b_n] = \sum_{(m,n)} (-1)^{sg(\pi)}[c_{\pi_1}|\cdots|c_{\pi_{m+n}}],$$

where π covers all the permutations over the set of integers

$$\{1, \ldots, m+n\}$$

which give rise to a natural shuffle of the elements

$$(c_1, \ldots, c_{m+n}) = (a_1, \ldots, a_m, b_1, \ldots, b_n);$$

natural in the sense that both of $1 \leq i < j \leq m$ or $m+1 \leq i < j \leq m+n$ implies that $\pi(i) < \pi(j)$. That is, the internal ordering among each of the lists $(a_1, \ldots, a_m)$ and $(b_1, \ldots, b_n)$ is to be preserved.

The sign $sg(\pi)$ is defined as follows:

$$sg(\pi) = \sum_{\pi(i)>\pi(j)} (|a_i|+1)\cdot(|b_j|+1).$$

In [2], it is explained that the homology of a CDGA may be computed in terms of a *homological model*, by means of *contractions*.

We now deeper explain the notion of *contraction* (see [5], [10]), which refers to a special kind of homotopy equivalence. A contraction is a data set

$$c : \{N, M, f, g, \phi\},$$

briefly denoted by $N \Rightarrow M$, where $f : N \to M$ and $g : M \to N$ are morphisms of DGA-modules (called, respectively, *projection* and *inclusion*) and $\phi : N \to N$ is a morphism of graded modules of degree $+1$ (called *homotopy operator*).

These data are required to satisfy the usual rules (see [14]): **(c1)** $fg = 1_M$, **(c2)** $\phi d_N + d_N \phi = 1 - gf$; and the side conditions **(c3)** $f\phi = 0$, **(c4)** $\phi g = 0$ and **(c5)** $\phi\phi = 0$.

Starting from the Bar construction and the notion of contraction as basis tools, Samuel Eilenberg and Saunders Mac Lane gave in the early fifties some contractions for computing the homology of two relevant types of CDGAs (see [5]), which are the exterior and polynomial algebras $E(u, 2n+1)$ and $P(v, 2n)$.

Here, $E(u, 2n+1)$ denotes the exterior algebra of generators 1 (of degree 0) and u (of degree $2n+1$) with trivial differential ($d \equiv 0$) and product ($1 \cdot u = u$ and $u \cdot u = 0$).

By the other hand, $P(v, 2n)$ denotes the polynomial algebra of generators 1 (of degree 0) and v^i (of degree $2ni$), with trivial differential ($d \equiv 0$) and product the usual one of monomials ($v^i \cdot v^j = v^{i+j}$).

There is a third relevant type of CDGA to be used later, which consists of modified polynomial algebras , $(v, 2n)$.

The expression , $(v, 2n)$ denotes the modified polynomial algebra of generators 1 (of degree 0) and $v^{(i)}$ (of degree $2ni$), with trivial differential ($d \equiv 0$) and product the natural one of monomials perturbed with a binomial coefficient ($1 \cdot v^{(i)} = v^{(i)}$ and $v^{(i)} \cdot v^{(j)} = C_{i+j,i} \cdot v^{(i+j)}$).

Eilenberg and Mac Lane stated in [5] some "homological models" for exterior and polynomial algebras. That is, some CDGAs which are free and of finite

type as DG-modules, such that one can easily reach their homology groups by applying suitable algorithms (as Veblen's, see [19]).

Furthermore, they gave the explicit formulas for the following contractions C_{BE} and C_{BP}:

$$C_{\bar{B}E} : \{\bar{B}(E(u, 2n+1)),\, , \, (\sigma(u), 2n+2), f_{\bar{B}E}, g_{\bar{B}E}, 0\}, \tag{1}$$

$$f_{\bar{B}E}([u| \overset{k}{\cdots} |u]) = \sigma(u)^{(k)},$$
$$g_{\bar{B}E}(\sigma(u)^{(k)}) = [u| \overset{k}{.} . |u];$$

$$C_{\bar{B}P} : \{\bar{B}(P(v, 2n)) \ , \ E(\sigma(v), 2n+1) \ , \ f_{\bar{B}P} \ , \ g_{\bar{B}P}, \phi_{\bar{B}P}\}, \tag{2}$$

$$f_{\bar{B}P}[v^r] \ = \ \begin{cases} 0, & \text{if } r \neq 1, \\ \sigma(v), & \text{if } r = 1, \end{cases}$$
$$f_{\bar{B}P}[v^{r_1}|v^{r_2}|\cdots|v^{r_m}] \ = \ 0,$$

$$g_{\bar{B}P}(\sigma(v)) = [v],$$
$$\phi_{\bar{B}P}[v^{r_1}|\cdots|v^{r_m}] = [v|v^{r_1-1}|v^{r_2}|\cdots|v^{r_m}].$$

Our goal is to compute the homology of general CDGAs, not only that of exterior, polynomial and modified polynomial algebras.

The main tool in achieving this purpose concerns to the "bar contraction", which connects the Bar construction of the tensor product of two algebras with the tensor product of the corresponding Bar constructions.

Theorem 1 *[5] Let A and A' be two commutative algebras. The "bar contraction" consists in the following explicit contraction:*

$$C_{\otimes} : \{\bar{B}(A \otimes A'),\, \bar{B}(A) \otimes \bar{B}(A'),\, f_{\otimes}, g_{\otimes}, \phi_{\otimes}\}.$$

For elements a_i in A and a'_i in A',

$$f_{\otimes}[a_1 \otimes a'_1|\ldots|a_m \otimes a'_m] = \sum_{i=0}^{m} \nu_i [a_1|\cdots|a_i] \otimes [a'_{i+1}|\cdots|a'_m].$$

For $i = 0$, the term $[a_1|\cdots|a_i]$ is understood to be the identity of $\bar{B}(A)$; similarly, for $i = m$, $[a'_{i+1}|\cdots|a'_m]$ designates the identity of $\bar{B}(A')$.

The scalar ν_i depends on the scalar character of the input data a_i and a'_j.

We may identify $\bar{B}(A)$ and $\bar{B}(A')$ as sub-DG-algebras of $\bar{B}(A \otimes A')$ by means of the morphisms

$$[a_1|\cdots|a_m] \mapsto [a_1 \otimes 0'|\cdots|a_m \otimes 0'],$$

$$[a'_1|\cdots|a'_m] \mapsto [0 \otimes a'_1|\cdots|0 \otimes a'_m];$$

where 0 and $0'$ are the identities of the algebras A and A', respectively.

With this identification at hand, we define $g_\otimes$ by the formula

$$g_\otimes(u \otimes u') = u * u', \qquad u \in \bar{B}(A),\, u' \in \bar{B}(A');$$

where $*$ means the shuffle product of $\bar{B}(A \otimes A')$.

Let $[a_1 \otimes a'_1|\cdots|a_m \otimes a'_m]$ denotes a homogeneous element of $\bar{B}(A \otimes A')$. Then, $\phi_\otimes = \zeta^{-1} SHI\, \zeta$, where ζ is the isomorphism of DG-modules defined by

$$\zeta[a_1 \otimes a'_1|\ldots|a_m \otimes a'_m] = (-1)^{\sum_{j>k} |a_j||a'_k|}[a_1|\ldots|a_m] \times [a'_1|\ldots|a'_m];$$

and SHI is the homotopy operator of the Eilenberg-Zilber contraction (see [6],[5],[18]),

$$\begin{aligned} SHI([a_1|\cdots|a_n] \times [a'_1|\cdots|a'_n]) \;\; &= \\ = \;\; -\sum (-1)^{m+sg(\alpha,\beta)} &(s_{\beta_q+m}\cdots s_{\beta_1+m} s_{m-1} \partial_{n-q+1}\cdots\partial_n[a_1|\cdots|a_n]\times \\ &\times s_{\alpha_{p+1}+m}\cdots s_{\alpha_1+m}\partial_m\cdots\partial_{m+p-1}[a'_1|\cdots|a'_n]); \end{aligned}$$

where $m = n-p-q$, $sg(\alpha,\beta) = \sum_{i=1}^{p}(\alpha_i - (i-1))$, and the sum is taken over the indexes $0 \le q \le n-1$, $0 \le p \le n-q-1$ and $(\alpha,\beta) \in \{(p+1,q)\text{-shuffles}\}$.

The morphisms ∂_* and s_* constitute the simplicial morphisms of the Bar construction (see [16]).

Homologies of exterior and polynomial algebras and CDGAs are related by the notion of *free CDGAs*.

Definition 2 *([3])* A *free CDGA* consists in a twisted tensor product of exterior and polynomial algebras.

Taking into account the previous contractions, we can reach a homological model of any free CDGA by means of twisted tensor products of exterior and modified polynomial algebras, by simply tensoring and composing the three contractions above in the appropriate way. Furthermore, we can carry on with general CDGAs, not necessarily commutative.

Indeed, there is a powerful result which makes possible to generalize the computation of the homology of free CDGAs to the computation of the homology

of any CDGA.

Theorem 3 *([7]) Let A be any CDGA. There exist a free CDGA A' and a morphism $\phi : A \to A'$ such that ϕ induces a homology isomorphism.*

Taking this result at hand, then it is possible to reduce the computation of the homology of any CDGA to the computation of the homology of a free CDGA, in terms of the three contractions explained before.

We are now ready to afford the purpose of our work. In this paper, we are concerned with the design of a *Mathematica* "package" for computing the homology of a CDGA with linear differential.

What we do, indeed, is to compute the differential operator of the homological model and not the homology groups, actually. The full homological information may be reached by only applying Veblen's algorithm (see [19]). In this sense, it may be an interesting effort to implement a routine for computing the Smith normal form of a given matrix.

Nevertheless, homological information is available by knowing the differential operator. We devote the rest of the paper to this purpose.

Let suppose that $A = A_1 \otimes \cdots \otimes A_n$ is a CDGA with $A_i \in \{E, , \}$, endowed with a linear differential δ (defined upon the generator of each algebra A_i). We now take into account the morphisms defined in formulas 1, 2 and 1 concerning to homological models for exterior and polynomial algebras and the bar contraction, respectively. Assuming the notation developed there, we can rewrite the differential operator obtained in [2] in the following way:

$$d = ((f_1 \otimes \cdots \otimes f_t)(1^{t-2} \otimes f_\otimes) \cdots f_\otimes \ \delta \circ$$

$$\circ \sum_{i \geq 0} (-1)^i [([\phi_\otimes + \sum_{j=1}^{t-2} (1^{j-1} \otimes g_\otimes)(1^j \otimes \phi_\otimes)(1^{j-1} \otimes f_\otimes)] +$$

$$+ [g_\otimes \cdots (1^{t-2} \otimes g_\otimes) \sum_{j=0}^{t-1} (g_1 f_1 \otimes \cdots \otimes g_j f_j \otimes \phi_{j+1} \otimes 1^{t-j+1})(1^{t-2} \otimes f_\otimes) \cdots f_\otimes])\delta]^i \circ$$

$$\circ [g_\otimes \cdots (1^{t-2} \otimes g_\otimes)(g_1 \otimes \cdots \otimes g_t)].$$

The aim of what follows is trying to translate the formula above into a *Mathematica* notebook. We will consider several steps, according to the single functions that appear above: f_n, g_n, ϕ_n, $g_\otimes$, $f_\otimes$, $\phi_\otimes$ and δ.

2 Introducing the initial data.

We next include the code of the programme for introducing the perturbation datum δ. We provide some verification routines.

```
Label[Inicio];
n1=Input["How many factors do compose the product algebra A?"];
dim1={};
Print["Introduce {a,g} for each factor of A."];
Print["a=0 means Polynomial algebra."];
Print["a=1 means Exterior algebra."];
Print["g denotes the degree of the generator of the algebra."];
Do[
  Print["Please, introduce factor ",i," on A."];
  Label[Retorno];
  a=Input[];
  If[
    ((a[[1]]==1)&&(Mod[a[[2]],2]==0))||
    ((a[[1]]==0)&&(Mod[a[[2]],2]==1)),
      Print["This is not a valid pair. Try again"];
      Goto[Retorno],
     dim1=Append[dim1,a]],
{i,n1}];
dim1=Transpose[dim1];
posalgext=Select[Table[i,{i,n1}],dim1[[1,#1]]==1&];
(*
The number of factors on A is n1. Characters and
generator degrees are located on the list dim1.
*)
Print["The ",n1," generators of A are denoted by u[i]."];
Print["The perturbation datum delta is determined by
  its images upon u[i], which are assumed
  to be lists of length ",n1+1,""];
Print["The first element in the list corresponds
  to the scalar coefficient."];
Print["The following elements represent the exponent of
  the associated power of the generator of
  the correspondent algebra factor."];
delta={};
Do[
  Print["Please, introduce delta(u[",i,"])
   as a list of lists of length ",n1+1];
  delta=Append[delta,Input[]],
{i,n1}];
```

```
coincieen[l1_,l2_]:=Module[{i},
  i=0;
  Do[
    If[
      Rest[l1[[i1]]]==Rest[l2],
    i=i1],
  {i1,Length[l1]}];
i];
simplmodelo[l1_List,l2_List]:=Module[{k,j},
  If[
    l2[[1]]==0,k=l1,
    j=coincieen[l1,l2];
    If[
      j==0,k=Append[l1,l2],
      ReplacePart[l1,Prepend[Rest[l2],
        l1[[j,1]]+l2[[1]]],j]]];
k];
apdelta1[l_]:=Fold[simplmodelo,{Table[0,{i1,n1+1}]},
  Flatten[Map[apdelta2,l],1]];
apdelta2[l_]:=Module[{pos,k},
  k={Table[0,{i1,n1+1}]};
  pos=Select[Table[1+i1,{i1,n1}],l[[#1]]=!=0&];
  k=Join[k,Flatten[Table[apdelta3[l,pos[[i1]]-1],
    {i1,Length[pos]}],1]]];
posalg[l_]:=Module[{i1},
  Do[
    If[l[[1+posalgext[[i1]]]]>1,Throw[False]],
  {i1,Length[posalgext]}];
Throw[True]];
simplalgebra[l_]:=Module[{k},
  If[Catch[posalg[l]],
     k=l,
    k=Table[0,{i1,n1+1}]];
k];
apdelta3[l_,i1_]:=Table[simplalgebra[Prepend[(Rest[l]-Insert[
  Table[0,{i2,n1-1}],1,i1])+Rest[delta[[i1,i3]]],
  l[[1]]*delta[[i1,i3,1]]*l[[i1+1]]*((-1)^Apply[
  Plus,Take[l,{2,i1}]])]],{i3,Length[delta[[i1]]]}];
Print["Checking that delta is well defined as a
  perturbation datum."];
Print["Wait a moment, please."];
comprobar1[l1_,i1_]:=If[
  (Apply[Plus,Rest[l1]*dim1[[2]]]==dim1[[2,i1]]-1)||
  (l1[[1]]==0),
    True,
```

```
    False];
comprobar2[l1_,i1_]:=Module[{},
  Do[
    If[
      comprobar1[l1[[i2]],i1],,
      Throw[False]],
  {i2,Length[l1]}];
Throw[True]];
Do[
  If[
    Catch[comprobar2[delta[[i]],i]],,
    Print["The given data do not define a
    perturbation datum, since delta(u[",i,"])
    is not of the appropiate degree."];
  Goto[Inicio]];
  If[apdelta1[delta[[i]]]=={Table[0,{j,n1+1}]},,
    Print["delta does not define a perturbation
    datum: delta^2(u[",i,"]) is not zero."];
  Goto[Inicio]],
{i,n1}];
Print["OK to proceed with the computation."];
```

We are now ready to implement the single functions.

3 The inclusion g_*.

In this section, we are interested in the implementation of the inclusion morphisms g_* from $\bar{B}(E)$ and $\bar{B}(P)$ to , and E, respectively.

Elements in the homological models , and E are codified as pairs of integers $\{\lambda, k\}$. This way, $\{\lambda, k\}$ is understood to be the element λu^k, where u denotes the corresponding generator.

We know that the homogeneous elements in $\bar{B}(E)$ and $\bar{B}(P)$ are lists of the type $[u|\cdots|u]$ and $[u^{i_1}|\cdots|u^{i_s}]$. Thus, we codify these elements as tuples of the form $\{\lambda, \text{exponents}\}$, where the integers in "exponent" refer to the exponents of the powers of the generator u.

We now proceed to implement the inclusions g_*,

$$g_{\bar{B}E}(\sigma(u)^{(k)}) = [u|\overset{k}{\ldots}|u],$$

$$g_{\bar{B}P}(\sigma(v)) = [v].$$

No difficulties arise in order to understand the code.

```
(*
A basis element of simplicial degree t of B(Ai(u))
is codified as a list of length t+1, so that
{lambda,s1,...,st} is in correspondance with the
element lambda[u^s1|...|u^st]. Elements in the
homological model of B(Ai) are codified as pairs
{lambda,exponent} and their linear relations.
*)
(*
g3[n][{lambda,t}] gives the image of lambda*sigma[u]^t
by the inclusion morphism on the homological model
of B(Ai)]), with n=0 if Ai=P and n=1 if Ai=E,
as it is the case.
*)
g3[n][{lambda,t}] gives the image of lambda*sigma[u]^t
by the inclusion morphism on the homological model
of B(Ai)]), with n=0 if Ai=P and n=1 if Ai=E,
as it is the case.
*)
g3[1][l_]:=Module[{k},
  If[l[[2]]==0,
    k={l[[1]],{}},
    k=Prepend[Table[1,{i,l[[2]]}],l[[1]]]];
k];
g3[0][l_]:=Module[{k},
  If[l[[2]]==0,
    k={l[[1]],{}},
    If[l[[2]]==1,
      k={l[[1]],1},
      k={0,{}}]];
k];
```

4 The projection f_*.

We are interested here in the implementation of the projection morphisms f_* from $\bar{B}(E)$ and $\bar{B}(P)$ to , and E, respectively:

$$f_{\bar{B}E}([u|\overset{k}{\cdots}|u]) = \sigma(u)^{(k)},$$

$$f_{\bar{B}P}[v^r] \;=\; \begin{cases} 0, & \text{if } r \neq 1, \\ \sigma(v), & \text{if } r = 1, \end{cases}$$

$$f_{\bar{B}P}[v^{r_1}|v^{r_2}|\cdots|v^{r_m}]. \;=\; 0,$$

Once again, it is easy to translate these functions into a *Mathematica* code.

```
(*
The output of f[n][l] is a list of length 2,
{lambda,t}. The first element (lambda)is
nothing but the corresponding scalar integer.
The second element (t) corresponds to the
exponent of the generator u on the homological
model of B(Ai) with n=0 if Ai=P and n=1 if Ai=E,
as it is the case.
*)
f[1][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={l[[1]],0},
    k={l[[1]],Length[l]-1}];
k];
f[0][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={l[[1]],0},
    If[(Length[l]==2)&&(l[[2]]==1),
      k={l[[1]],1},
      k={0,Length[l]-1}]];
k];
f[dimen_List][l_List]:=unif2[Table[
  f[dimen[[i]]][l[[i]]],{i,Length[dimen]}]];
```

5 The homotopy ϕ_*.

The implementation of the homotopy operators ϕ_* is quite easy,

$$\phi_{\bar{B}E} \equiv 0,$$

$$\phi_{\bar{B}P}[v^{r_1}|\cdots|v^{r_m}] = [v|v^{r_1-1}|v^{r_2}|\cdots|v^{r_m}].$$

We next include the code.

```
(*
The function hofi[n][l] computes the image of an
```

```
element l in B(An) by applying the homotopy operator
of the contraction. Notice that hofi is necessarely
zero in the case Ai=E.
*)
hofi[1][l_]:=Table[0,{i,Length[l]+1}];
hofi[0][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={0,{}},
    If[l[[2]]=!=1,
      k=Join[{l[[1]],1,l[[2]]-1},Drop[l,2]],
      k={0,{}}]];
k];
```

6 The inclusion $g_{\otimes}$.

The formula of $g_{\otimes}$ consists indeed in the shuffle product one,

$$g_{\otimes}([a_1|\cdots|a_n],[g_1|\cdots|g_m]) = [a_1\otimes 0'|\cdots|a_n\otimes 0']*[0\otimes g_1|\cdots|0\otimes g_m].$$

Realize that we are not only interested in the formula of $g_{\otimes}$ but also in formulas of each of the compositions $g_{\otimes}\cdots(1^s\otimes g_{\otimes})$. As $A = A_1\otimes\cdots\otimes A_t$, we will define a function `gbar[i][l1,l2]` which will transform an element of

$$\bar{B}(A_1)\otimes\cdots\otimes\bar{B}(A_i)\otimes\bar{B}(A_{i+1}\otimes\cdots\otimes A_t)$$

onto an element of

$$\bar{B}(A_1)\otimes\cdots\otimes\bar{B}(A_{i-1})\otimes\bar{B}(A_{p_i}\otimes\cdots\otimes A_t).$$

Taking into account that the first $(i-1)$ factors remain unchangeable, we define the function attending only to the last two factors.

The code of this morphism is the following. Notice that functions `*aument*` compute lists of shuffles and functions `gradoshu*` calculate the right signs to consider.

```
(*
Function gbar[i][l1,l2] computes the image of
the inclusion morphism of the bar contraction,
applied to the given elements l1 and l2 in B(Ai)
```

```
and B(Ai+1x...xAn1), respectively.
*)
aument[j6_][k6_]:=Table[Join[k6,{j5}],{j5,Last[k6]+1,j6}];
apaument[j6_,j5_][k6_]:=Flatten[Map[aument[j6-j5+1+Length[k6[[1]]]],k6],1];
shuf[j5_,1]:=Table[{j6},{j6,1,j5}];
shuf[j5_,j6_]:=Nest[apaument[j5,j6],shuf[j5-j6+1,1],j6-1];
insertarlista[k5_,k6_][k4_]:=Module[{j5,k7},
  k7=k6;
  Do[
    k7=Insert[k7,k5[[j5]],k4[[j5]]],
  {j5,Length[k5]}];
k7];
gradshuf2[k1_]:=1+Apply[Plus,k1*Take[dim1[[1]],-Length[k1]]];
gradoshu[1][k2_,k3_]:=1;
gradoshu[0][k2_,k3_]:=Module[{suma},
  suma=0;
  Do[
    suma=suma+Apply[Plus,Map[gradshuf2,Take[
      k2,{2,k3[[i1]]}]]],
  {i1,Length[k3]}]; (-1)^suma];
mezclar[k1_,k2_,i_][k3_]:=Prepend[insertarlista[Rest[k1],
 Rest[k2]][k3],k1[[1]]*gradoshu[dim1[[1,i]]][k2,k3]];
anadir1[l1_][l2_]:=Map[anadir2[l1],l2];
anadir2[l1_][l2_]:=Join[l1,l2];
gbar[i_][l1_,l2_]:=Module[{k,shuffle,m1,m2},
  If[VectorQ[l1[[2]]],
    If[VectorQ[l2[[2]]]&&Length[l2[[2]]]==0,
      k={l1[[1]]*l2[[1]],{}},
      k=Prepend[anadir1[{0}][Map[List,Rest[l2]]],
       l1[[1]]*l2[[1]]]],
    If[VectorQ[l2[[2]]]&&Length[l2[[2]]]==0,
      k=Prepend[Table[Prepend[Table[0,{i1,n1-i}],
       l1[[i2]]],{i2,2,Length[l2]}],l1[[1]]*l2[[1]]],
      m2=l2;
      If[i==n1-1,m2=Prepend[Map[List,Rest[l2]],l2[[1]]]];
      m1=Prepend[Table[Prepend[Table[0,{i2,n1-i}],
       l1[[i1]]],{i1,2,Length[l1]}],l1[[1]]*l2[[1]]];
      m2=Prepend[anadir1[{0}][Rest[m2]],1];
      shuffle=shuf[Length[l1]+Length[l2]-2,Length[l1]-1];
      k=Map[mezclar[m1,m2,i],shuffle]]];
k];
```

7 The projection $f_\otimes$.

Recall the projection morphism of the bar contraction,

$$f_\otimes[a_1 \otimes a'_1|\dots|a_m \otimes a'_m] = \sum_{i=0}^{m} \nu_i[a_1|\cdots|a_i] \otimes [a'_{i+1}|\cdots|a'_m],$$

where ν_i is zero whenever one of $\{a_{i+1}, \dots, a_n, a'_1, \dots, a'_i\}$ **is not** a scalar, and their product otherwise.

An analogous reasoning to the function just implemented leads us to ask for a more complicated function `fbar` which is available for all the products $(1^s \otimes f_\otimes)$.

The main difficulties on the implementation of this morphism are finding the indexes which lead to non zero summands, and determining the right signs of each of these summands. Commands `cota*` and `signo` help in this purpose.

```
(*
Function fbar[i][l] computes the image of an element l
of B(Aix...xAn) by the projection of the bar contraction.
*)
fbar[i_][l_]:=Module[{k,m1,m2,cota1,cota2,cota3,cota4,t},
  If[
    Length[l]==2&&Length[l[[2]]]==0,
      k={{{l[[1]],{}},{1,{}}}},
      m1=Table[First[l[[i1]]],{i1,2,Length[l]}];
      m2=Map[Rest,Rest[l]];
      t=Length[m1];
      cota1=t+1-Position[Append[Reverse[m2],Table[0,
       {i1,n1-i}]],Table[0,{i1,n1-i}],1,1];
      cota2=Position[Append[m1,0],0,1,1]-1;
      cota3=Last[Prepend[Select[Table[i1,
       {i1,t}],m1[[#1]]=!=0&],0]];
      cota4=First[Append[Select[Table[i1,{i1,0,t-1}],
       m2[[#1+1]]=!=Table[0,{i1,n1-i}]&],t]];
      cota1=Max[cota1,cota3];
      cota2=Min[cota2,cota4];
      If[cota1>cota2,
        k={{0,{}},{0,{}}},
        If[i==2,m2=Flatten[m2,1]];
        k=Table[{Join[{l[[1]]},Take[m1,i1]],Join[{1},
         Take[m2,i1-t]]},{i1,cota1,cota2}]]];
k];
```

8 The homotopy operator $\phi_{\otimes}$.

We finally implement the homotopy operator $\phi_{\otimes}$,

$$\phi_{\otimes} = \zeta^{-1} SHI\, \zeta,$$

where ζ is the isomorphism of DG-modules defined by

$$\zeta[a_1 \otimes a'_1|\dots|a_m \otimes a'_m] = (-1)^{\sum_{j>k}|a_j||a'_k|}[a_1|\dots|a_m] \times [a'_1|\dots|a'_m];$$

and SHI is the homotopy operator of the Eilenberg-Zilber contraction (see [6],[5],[18]),

$$\begin{aligned} SHI([a_1|\cdots|a_n] \times [a'_1|\cdots|a'_n]) \;\; &= \\ = \;\; -\sum(-1)^{m+sg(\alpha,\beta)}&(s_{\beta_q+m}\cdots s_{\beta_1+m}s_{m-1}\partial_{n-q+1}\cdots\partial_n[a_1|\cdots|a_n]\times \\ &\times s_{\alpha_{p+1}+m}\cdots s_{\alpha_1+m}\partial_m\cdots\partial_{m+p-1}[a'_1|\cdots|a'_n]); \end{aligned}$$

where $m = n-p-q$, $sg(\alpha,\beta) = \sum_{i=1}^{p}(\alpha_i - (i-1))$, and the sum is taken over the indexes $0 \le q \le n-1$, $0 \le p \le n-q-1$ and $(\alpha,\beta) \in \{(p+1,q)\text{-shuffles}\}$.

The morphisms ∂_* and s_* constitute the simplicial morphisms of the Bar construction (see [16]).

In the context of the differential operator we are trying to define, a theoretical improvement may be done such that the formula above reduces to the following

$$\phi_{\otimes}[a_1 \otimes a'_1|\cdots|a_n \otimes a'_n] = \sum_{q\in\{\nu_i\neq 0\}}\sum_{p=0}^{n-q-1}(-1)^{\text{signo}}.$$

$$[a_1 \otimes a'_1|\cdots|a_{n-p-q-1}\otimes a'_{n-p-q-1}|a'_{n-p-q}\cdots a'_{n-q}|a_{n-p-q}|\cdots|a_{n-q}|a'_{n-q+1}|\cdots|a'_n],$$

where

$$\text{signo} = n-p-q+|a_1|+\cdots|a_{n-p-q-1}| + \sum_{i=n-q+1}^{n}(|a_1|+\cdots+|a_i|)+$$

$$+q(|a_1|+\cdots+|a_{n-q}|)+\sum_{i=n-p-q}^{n-q-1}(|a'_1|+\cdots+|a'_i|)+(p-1)(|a'_1|+\cdots+|a'_{n-q}|)+$$

$$+\sum_{i=n-p-q}^{n-q}|a_i|(|a'_i|+\cdots+|a'_n|)+\sum_{j=n-q+1}^{n}\sum_{k=1}^{j-1}|a_j||a'_k|.$$

This is proved in a work of the authors still in progress.

The code of this last formula is as follows.

```
(*
Function phibar[i][l] computes the homotopy operator of
the bar contraction acting on l. Notice that the
contraction consists in B(Aix...xAn)-->B(Ai)xB(Ai+1x...xAn).
*)
phibar[i_][l_]:=Module[{k,m0,m1,m2,m3,m4,n,x},
  n=Length[l]-1;
  x=First[Append[Select[Table[i1,{i1,n}],
   First[l[[n+2-#1]]]=!=0&],n]];
  gr[l1_]:=Apply[Plus,l1*Take[dim1[[2]],-Length[l1]]];
  m1[i1_,i2_]:=Take[l,{2,n-i1-i2}];
  m2[i1_,i2_]:={Prepend[Apply[Plus,Table[
   Rest[l[[i3]]],{i3,n+1-i2-i1,n+1-i1}]],0]};
  m3[i1_,i2_]:=Table[Join[{First[l[[i3]]]},Table[0,
   {i4,Length[l[[2]]]-1}]],{i3,1+n-i2-i1,1+n-i1}];
  m4[i1_,i2_]:=Table[Prepend[Rest[l[[i3]]],0],
   {i3,n+2-i1,n+1}];
  m0[i1_,i2_]:=Module[{signo},
    signo=n-i1-i2;
    signo=signo+Apply[Plus,Table[Apply[Plus,
     Table[gr[Rest[l[[j2]]]],{j2,2,j1+1}]],
     {j1,n-i1-i2,n-i1-1}]];
    signo=signo+(i2+1)*Apply[Plus,Table[gr[
     Rest[l[[j1]]]],{j1,2,n+1-i1}]];
    If[dim1[[1,i]]==1,
      signo=signo+Apply[Plus,Table[Apply[
       Plus,Table[First[l[[j2]]],
       {j2,2,j1+1}]],{j1,n-i1+1,n}]];
      signo=signo+Apply[Plus,Table[
       First[l[[j1]]],{j1,2,n-i1-i2}]];
      signo=signo+i1*Apply[Plus,Table[
       First[l[[j1]]],{j1,2,n-i1+1}]];
      signo=signo+Apply[Plus,Table[
       First[l[[j1]]]*Apply[Plus,Table[
       gr[Rest[l[[j2]]]],{j2,j1,n}]],
       {j1,n+1-i1-i2,n+1-i1}]];
      signo=signo+Apply[Plus,Table[
       Apply[Plus,Table[First[l[[j1]]]*
       gr[Rest[l[[j2]]]],{j2,2,j1}]],
       {j1,n-i1+2,n+1}]]];
    signo=(-1)^signo;
  signo];
  k=Flatten[Table[Table[Prepend[Join[m1[i1,i2],
   m2[i1,i2],m3[i1,i2],m4[i1,i2]],l[[1]]*m0[i1,i2]],
   {i2,0,n-i1-1}],{i1,0,x-1}],1];
```

```
k];
```

9 Programme code.

We next include the full programme code.

```
Label[Inicio];
n1=Input["How many factors do compose the product algebra A?"];
dim1={};
Print["Introduce {a,g} for each factor of A."];
Print["a=0 means Polynomial algebra."];
Print["a=1 means Exterior algebra."];
Print["g denotes the degree of the generator of the algebra."];
Do[
  Print["Please, introduce factor ",i," on A."];
  Label[Retorno];
  a=Input[];
  If[
    ((a[[1]]==1)&&(Mod[a[[2]],2]==0))||
    ((a[[1]]==0)&&(Mod[a[[2]],2]==1)),
      Print["This is not a valid pair. Try again"];
      Goto[Retorno],
     dim1=Append[dim1,a]],
{i,n1}];
dim1=Transpose[dim1];
posalgext=Select[Table[i,{i,n1}],dim1[[1,#1]]==1&];
(*
The number of factors on A is n1. Characters and
generator degrees are located on the list dim1.
*)
Print["The ",n1," generators of A are denoted by u[i]."];
Print["The perturbation datum delta is determined by
  its images upon u[i], which are assumed
  to be lists of length ",n1+1,""];
Print["The first element in the list corresponds
  to the scalar coefficient."];
Print["The following elements represent the exponent of
  the associated power of the generator of
  the correspondent algebra factor."];
delta={};
Do[
  Print["Please, introduce delta(u[",i,"])
```

```
    as a list of lists of length ",n1+1];
  delta=Append[delta,Input[]],
{i,n1}];
coincieen[l1_,l2_]:=Module[{i},
  i=0;
  Do[
    If[
      Rest[l1[[i1]]]==Rest[l2],
    i=i1],
  {i1,Length[l1]}];
i];
simplmodelo[l1_List,l2_List]:=Module[{k,j},
  If[
    l2[[1]]==0,k=l1,
    j=coincieen[l1,l2];
    If[
      j==0,k=Append[l1,l2],
      ReplacePart[l1,Prepend[Rest[l2],
        l1[[j,1]]+l2[[1]]],j]]];
k];
apdelta1[l_]:=Fold[simplmodelo,{Table[0,{i1,n1+1}]},
  Flatten[Map[apdelta2,l],1]];
apdelta2[l_]:=Module[{pos,k},
  k={Table[0,{i1,n1+1}]};
  pos=Select[Table[1+i1,{i1,n1}],l[[#1]]=!=0&];
  k=Join[k,Flatten[Table[apdelta3[l,pos[[i1]]-1],
    {i1,Length[pos]}],1]]];
posalg[l_]:=Module[{i1},
  Do[
    If[l[[1+posalgext[[i1]]]]>1,Throw[False]],
  {i1,Length[posalgext]}];
Throw[True]];
simplalgebra[l_]:=Module[{k},
  If[Catch[posalg[l]],
     k=l,
    k=Table[0,{i1,n1+1}]];
k];
apdelta3[l_,i1_]:=Table[simplalgebra[Prepend[(Rest[l]-Insert[
  Table[0,{i2,n1-1}],1,i1])+Rest[delta[[i1,i3]]],
  l[[1]]*delta[[i1,i3,1]]*l[[i1+1]]*((-1)^Apply[
  Plus,Take[l,{2,i1}]])]],{i3,Length[delta[[i1]]]}];
Print["Checking that delta is well defined as a
  perturbation datum."];
Print["Wait a moment, please."];
comprobar1[l1_,i1_]:=If[
```

```
  (Apply[Plus,Rest[l1]*dim1[[2]]]==dim1[[2,i1]]-1)||
  (l1[[1]]==0),
    True,
    False];
comprobar2[l1_,i1_]:=Module[{},
  Do[
    If[
      comprobar1[l1[[i2]],i1],,
      Throw[False]],
  {i2,Length[l1]}];
Throw[True]];
Do[
  If[
    Catch[comprobar2[delta[[i]],i]],,
    Print["The given data do not define a
    perturbation datum, since delta(u[",i,"])
    is not of the appropiate degree."];
  Goto[Inicio]];
  If[apdelta1[delta[[i]]]=={Table[0,{j,n1+1}]},,
    Print["delta does not define a perturbation
    datum: delta^2(u[",i,"]) is not zero."];
  Goto[Inicio]],
{i,n1}];
Print["OK to proceed with the computation."];
(*
A basis element of simplicial degree t of B(Ai(u))
is codified as a list of length t+1, so that
{lambda,s1,...,st} is in correspondance with the
element lambda[u^s1|...|u^st]. Elements in the
homological model of B(Ai) are codified as pairs
{lambda,exponent} and their linear relations.
*)
(*
g3[n][{lambda,t}] gives the image of lambda*sigma[u]^t
by the inclusion morphism on the homological model
of B(Ai)]), with n=0 if Ai=P and n=1 if Ai=E,
as it is the case.
*)
g3[1][l_]:=Module[{k},
  If[l[[2]]==0,
    k={l[[1]],{}},
    k=Prepend[Table[1,{i,l[[2]]}],l[[1]]]];
k];
g3[0][l_]:=Module[{k},
  If[l[[2]]==0,
```

```
    k={l[[1]],{}},
    If[l[[2]]==1,
      k={l[[1]],1},
      k={0,{}}]];
k];
(*
The output of f[n][l] is a list of length 2,
{lambda,t}. The first element (lambda)is
nothing but the corresponding scalar integer.
The second element (t) corresponds to the
exponent of the generator u on the homological
model of B(Ai) with n=0 if Ai=P and n=1 if Ai=E,
as it is the case.
*)
f[1][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={l[[1]],0},
    k={l[[1]],Length[l]-1}];
k];
f[0][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={l[[1]],0},
    If[(Length[l]==2)&&(l[[2]]==1),
      k={l[[1]],1},
      k={0,Length[l]-1}]];
k];
f[dimen_List][l_List]:=unif2[Table[
  f[dimen[[i]]][l[[i]]],{i,Length[dimen]}]];
(*
The function hofi[n][l] computes the image of an
element l in B(An) by applying the homotopy operator
of the contraction. Notice that hofi is necessarely
zero in the case Ai=E.
*)
hofi[1][l_]:=Table[0,{i,Length[l]+1}];
hofi[0][l_]:=Module[{k},
  If[VectorQ[l[[2]]],
    k={0,{}},
    If[l[[2]]=!=1,
      k=Join[{l[[1]],1,l[[2]]-1},Drop[l,2]],
      k={0,{}}]];
k];
(*
Function gbar[i][l1,l2] computes the image of
the inclusion morphism of the bar contraction,
```

```
applied to the given elements l1 and l2 in B(Ai)
and B(Ai+1x...xAn1), respectively.
*)
aument[j6_][k6_]:=Table[Join[k6,{j5}],{j5,Last[k6]+1,j6}];
apaument[j6_,j5_][k6_]:=Flatten[Map[aument[j6-j5+1+Length[k6[[1]]]],k6],1];
shuf[j5_,1]:=Table[{j6},{j6,1,j5}];
shuf[j5_,j6_]:=Nest[apaument[j5,j6],shuf[j5-j6+1,1],j6-1];
insertarlista[k5_,k6_][k4_]:=Module[{j5,k7},
  k7=k6;
  Do[
    k7=Insert[k7,k5[[j5]],k4[[j5]]],
  {j5,Length[k5]}];
k7];
gradshuf2[k1_]:=1+Apply[Plus,k1*Take[dim1[[1]],-Length[k1]]];
gradoshu[1][k2_,k3_]:=1;
gradoshu[0][k2_,k3_]:=Module[{suma},
  suma=0;
  Do[
    suma=suma+Apply[Plus,Map[gradshuf2,Take[
      k2,{2,k3[[i1]]}]]],
  {i1,Length[k3]}]; (-1)^suma];
mezclar[k1_,k2_,i_][k3_]:=Prepend[insertarlista[Rest[k1],
 Rest[k2]][k3],k1[[1]]*gradoshu[dim1[[1,i]]][k2,k3]];
anadir1[l1_][l2_]:=Map[anadir2[l1],l2];
anadir2[l1_][l2_]:=Join[l1,l2];
gbar[i_][l1_,l2_]:=Module[{k,shuffle,m1,m2},
  If[VectorQ[l1[[2]]],
    If[VectorQ[l2[[2]]]&&Length[l2[[2]]]==0,
      k={l1[[1]]*l2[[1]],{}},
      k=Prepend[anadir1[{0}][Map[List,Rest[l2]]],
       l1[[1]]*l2[[1]]]],
    If[VectorQ[l2[[2]]]&&Length[l2[[2]]]==0,
      k=Prepend[Table[Prepend[Table[0,{i1,n1-i}],
       l1[[i2]]],{i2,2,Length[l2]}],l1[[1]]*l2[[1]]],
      m2=l2;
      If[i==n1-1,m2=Prepend[Map[List,Rest[l2]],l2[[1]]]];
      m1=Prepend[Table[Prepend[Table[0,{i2,n1-i}],
       l1[[i1]]],{i1,2,Length[l1]}],l1[[1]]*l2[[1]]];
      m2=Prepend[anadir1[{0}][Rest[m2]],1];
      shuffle=shuf[Length[l1]+Length[l2]-2,Length[l1]-1];
      k=Map[mezclar[m1,m2,i],shuffle]]];
k];
(*
Function fbar[i][l] computes the image of an element l
of B(Aix...xAn) by the projection of the bar contraction.
```

```
*)
fbar[i_][l_]:=Module[{k,m1,m2,cota1,cota2,cota3,cota4,t},
  If[
    Length[l]==2&&Length[l[[2]]]==0,
      k={{{l[[1]],{}},{1,{}}}},
      m1=Table[First[l[[i1]]],{i1,2,Length[l]}];
      m2=Map[Rest,Rest[l]];
      t=Length[m1];
      cota1=t+1-Position[Append[Reverse[m2],Table[0,
       {i1,n1-i}]],Table[0,{i1,n1-i}],1,1];
      cota2=Position[Append[m1,0],0,1,1]-1;
      cota3=Last[Prepend[Select[Table[i1,
       {i1,t}],m1[[#1]]=!=0&],0]];
      cota4=First[Append[Select[Table[i1,{i1,0,t-1}],
        m2[[#1+1]]=!=Table[0,{i1,n1-i}]&],t]];
      cota1=Max[cota1,cota3];
      cota2=Min[cota2,cota4];
      If[cota1>cota2,
        k={{0,{}},{0,{}}},
        If[i==2,m2=Flatten[m2,1]];
        k=Table[{Join[{l[[1]]},Take[m1,i1]],Join[{1},
         Take[m2,i1-t]]},{i1,cota1,cota2}]]];
k];
(*
Function phibar[i][l] computes the homotopy operator of
the bar contraction acting on l. Notice that the
contraction consists in B(Aix...xAn)-->B(Ai)xB(Ai+1x...xAn).
*)
phibar[i_][l_]:=Module[{k,m0,m1,m2,m3,m4,n,x},
  n=Length[l]-1;
  x=First[Append[Select[Table[i1,{i1,n}],
   First[l[[n+2-#1]]]=!=0&],n]];
  gr[l1_]:=Apply[Plus,l1*Take[dim1[[2]],-Length[l1]]];
  m1[i1_,i2_]:=Take[l,{2,n-i1-i2}];
  m2[i1_,i2_]:={Prepend[Apply[Plus,Table[
   Rest[l[[i3]]],{i3,n+1-i2-i1,n+1-i1}]],0]};
  m3[i1_,i2_]:=Table[Join[{First[l[[i3]]]},Table[0,
   {i4,Length[l[[2]]]-1}]],{i3,1+n-i2-i1,1+n-i1}];
  m4[i1_,i2_]:=Table[Prepend[Rest[l[[i3]]],0],
   {i3,n+2-i1,n+1}];
  m0[i1_,i2_]:=Module[{signo},
    signo=n-i1-i2;
    signo=signo+Apply[Plus,Table[Apply[Plus,
     Table[gr[Rest[l[[j2]]]],{j2,2,j1+1}]],
     {j1,n-i1-i2,n-i1-1}]];
```

```
    signo=signo+(i2+1)*Apply[Plus,Table[gr[
     Rest[l[[j1]]]],{j1,2,n+1-i1}]];
    If[dim1[[1,i]]==1,
      signo=signo+Apply[Plus,Table[Apply[
       Plus,Table[First[l[[j2]]],
       {j2,2,j1+1}]],{j1,n-i1+1,n}]];
      signo=signo+Apply[Plus,Table[
       First[l[[j1]]],{j1,2,n-i1-i2}]];
      signo=signo+i1*Apply[Plus,Table[
       First[l[[j1]]],{j1,2,n-i1+1}]];
      signo=signo+Apply[Plus,Table[
       First[l[[j1]]]*Apply[Plus,Table[
       gr[Rest[l[[j2]]]],{j2,j1,n}]],
       {j1,n+1-i1-i2,n+1-i1}]];
      signo=signo+Apply[Plus,Table[
       Apply[Plus,Table[First[l[[j1]]]*
       gr[Rest[l[[j2]]]],{j2,2,j1}]],
       {j1,n-i1+2,n+1}]]];
    signo=(-1)^signo;
  signo];
  k=Flatten[Table[Table[Prepend[Join[m1[i1,i2],
   m2[i1,i2],m3[i1,i2],m4[i1,i2]],l[[1]]*m0[i1,i2]],
   {i2,0,n-i1-1}],{i1,0,x-1}],1];
k];
```

References

[1] V. Álvarez, J.A. Armario, M.D. Frau and P. Real, An algorithm for computing the first homology groups of some semidirect products of finite groups, *Third International Symposium on Mathematica IMS'99*, Hagenberg (Austria), 1999.

[2] V. Álvarez, J.A. Armario, P. Real and B. Silva, HPT and computability of the homology of commutative DGA-algebras, *Conference in Secondary Calculus and Cohomological Physics*, Moscow, 1997.

[3] D. Burghelea and M. Vigué Poirrier, Cyclic homology of commutative algebras I, *Lecture Notes in Mathematics, Algebraic Topology Rational Homotopy*, **1318** (1986), 51–72.

[4] H. Cartan and S. Eilenberg, Homological Algebra, *Princeton University Press, Princeton* 1956.

[5] S. Eilenberg and S. Mac Lane, On the groups $H(\pi, n)$ II, *Annals of Math.* **66** (1954) 49–139.

[6] S. Eilenberg and J.A. Zilber, On the products of complexes. *Am. J. Math.* **75** (1959) 200–204.

[7] J.A. Guccione and J.J. Guccione, Hochschild homology of commutative algebras in positive characteristic, *Communications in algebra*, **22 (15)** (1994), 6037–6046.

[8] V.K.A.M. Gugenheim and L. Lambe, Perturbation theory in Differential Homological Algebra, I, *Illinois J. Math.* **33** (1989) 556–582.

[9] V.K.A.M. Gugenheim, L. Lambe and J. Stasheff, Perturbation theory in Differential Homological Algebra II, *Illinois J. Math.* **35** n. 3 (1991) 357–373.

[10] J. Huesbschmann and Kadeishvili. Small models for chain algebras, *Math. Z.* **207** (1991) 245–280.

[11] L. A. Lambe, Resolutions via homological perturbation, *J. Symbolic Comp.* **12** (1991) 71–87.

[12] L. A. Lambe, Homological perturbation theory, Hochschild homology and formal groups, *Proc. Conference on Deformation Theory and Quantization with Applications to Physics, Amherst, MA, June 1990, Cont. Math.* **134** A.M.S (1992) 183–218.

[13] L. Lambe, Next generation computer algebra systems AXIOM and the scraptchpad concept: applications to research in algebra, *Collection "Analysis, algebra and computers in mathematical research"*, Lulea (1992) 201–222. *Lect. Notes in Pure and Appl. Math.* **156**, Dekker, New York (1994).

[14] L. Lambe and J. Stasheff, Applications of perturbation theory to iterated fibrations, *Manuscripta Math.* **58** (1987) 367–376.

[15] S. Mac Lane, Homology, *Classics in Mathematics* Springer-Verlag, Berlin, 1995. Reprint of the 1975 edition.

[16] J.P. May, Simplicial objects in Algebraic Topology, *Chicago Lectures in Mathematics* , The University of Chicago Press, Chicago and London, 1992.

[17] P. Real, Homological Perturbation Theory and Associativity, *Preprint of Dpto. Matematica Aplicada I, Sevilla* 1996.

[18] J. Rubio, Homologie effective des espaces de lacets itérés: un logiciel, *Ph. D. in Math. of University J. Fourier*, Grenoble (1991).

[19] O. Veblen, Analysis situs, *A.M.S. Publications*, **v. 5**, 1931.