A Computational approach of $A_\infty$-(co)algebras

Ainhoa Berciano-Alcaraz∗

Dpto. Matemática Aplicada, Estadística e Investigación Operativa
Euskal Herriko Unibertsitatea, Spain

(201.2 released August 2007)

We present a computer program to compute the explicit structural components of an $A_\infty$-(co)algebra deduced from a contraction, which is a special type of homotopy equivalence. The input is a contraction from a dg-(co)algebra to a simple dg-module and the output is a functional object that defines the operations $\{\mu_i\}_{i\geq 1}$ (resp. $\{\Delta_i\}_{i\geq 1}$) in the deduced $A_\infty$-structure on the simple dg-module. We conclude with some concrete applications.

Keywords: Homological perturbation theory, contraction, Basic Perturbation Lemma, Symbolic Computation, Software.

1. Introduction

Many concepts in algebraic topology involve complicated formulas. Consequently, applications often require sophisticated computations and explicit results are often difficult to obtain. Computer programs that create an environment for experimentation can assist the researcher in making conjectures. And this process can be very fruitful.

Two important tools in homological algebra are (1) the notion of contraction, which is a special type of homotopy equivalence between differential graded modules and (2) the Homological Perturbation Lemma (see [2] or [14]). Given the right conditions, these tools give an explicit algorithm for computing the structural components of an $A_\infty$-(co)algebra [9, 6, 7, 8]. In this paper we present a computer program that computes the explicit structural components of an $A_\infty$-(co)algebra deduced from a contraction, which is a special type of homotopy equivalence. The input is a contraction from a dg-(co)algebra to a simple dg-module and the output is a functional object that defines the operations $\{\mu_i\}_{i\geq 1}$ (resp. $\{\Delta_i\}_{i\geq 1}$) and gives the $A_\infty$-structure on the simple dg-module. We use the program to examine some interesting applications and determine which of these operations are non-vanishing; we give explicit formulas for computing these operations in some cases.

Throughout this exposition we use the generic term “$A_\infty$-structure” to refer to either an $A_\infty$-coalgebra and $A_\infty$-algebra. For simplicity, examples and definitions will be given only for $A_\infty$-coalgebras, with the analogous statements for

∗Work partially supported by a project of University of the Basque Country “EHU06/05” and by PAICYT research project FQM-296. Email: ainhoa.berciano@ehu.es
\[ A_{\infty}-\text{algebras following by duality.} \]

The paper is organized as follows: In Section 2 we review the preliminaries and establish our notation. Theoretical results underlying our computer program as well as some examples demonstrating the program are presented in Section 3. We conclude with some discussion and comments on future work.

2. Preliminaries and notation

Throughout this paper \( \Lambda \) denotes a commutative ring; in the examples, we set \( \Lambda = \mathbb{Z} \), the integers. We begin with a review of the basic definitions (for more detail see [10], [16]). A differential graded (dg-) module is a graded module \( M \) endowed with an endomorphism \( d \) of degree -1 such that \( dd = 0 \). The homology of \( M \) is the graded module \( H_n(M) = \ker d_n / \text{Im} d_{n+1} \)

Given a dg-module \( (M, d_M) \), the suspension of \( M \) is the DGM \( (sM, d_{sM}) \), where \( (sM)_n = M_{n-1} \) and \( d_{sM} = -d_M \). Dually, the desuspension of \( M \) is the DGM \( (s^{-1}M, d_{s^{-1}M}) \) given by \( (s^{-1}M)_n = M_{n+1} \) with differential \( -d_M \). A morphism of graded modules \( f : M \rightarrow N \) induces a morphism of suspensions \( sf : sM \rightarrow sN \) and dually of desuspensions \( s^{-1}f : s^{-1}M \rightarrow s^{-1}N \). A dg-algebra is a dg-module \( (A, d_A) \) endowed with an associative product \( \mu_a \) and a unit. Dually, a dg-coalgebra is a dg-module \( (A, d_a) \) endowed with an associative coproduct \( \Delta_a \) and a counit. The tensor module of \( M \) is the dg-module

\[ T(M) = \bigoplus_{n \geq 0} T^n(M) = \bigoplus_{n \geq 0} M_{\otimes n}, \]

where \( M_{\otimes 0} = \Lambda \), and with tensor differential \( d_t = d_{sM}^\otimes \) given by linear extension. A morphism \( f : M \rightarrow N \) of dg-modules induces a morphism \( T(f) : T(M) \rightarrow T(N) \) via \( T(f)|_{M_{\otimes n}} = f^\otimes n \). The tensor algebra of \( M \), denoted by \( T^a(M) \), is the dg-module \( (T(M), d_t) \) together with the product \( \mu \) given by concatenation, i.e.,

\[ \mu((a_1 \otimes \cdots \otimes a_n) \otimes (a_{n+1} \otimes \cdots \otimes a_{n+p})) = a_1 \otimes \cdots \otimes a_{n+p}. \]

The tensor coalgebra of \( M \), denoted by \( T^c(M) \), is the dg-module \( (T(M), d_t) \) together with coproduct \( \Delta \) given by

\[ \Delta(a_1 \otimes \cdots \otimes a_n) = \sum_{i=0}^n (a_1 \otimes \cdots \otimes a_i) \otimes (a_{i+1} \otimes \cdots \otimes a_n). \]

Given a connected dg-algebra \( A \), the reduced bar construction of \( A \) is the dg-coalgebra \( \overline{B}(A) = T \mathcal{s}(A) \) with cofree coproduct

\[ \Delta_a([a_1] \cdots [a_r]) = \sum_{i=1}^{r-1} [a_1] \cdots [a_i] \otimes [a_{i+1}] \cdots [a_r] \]

and differential \( d_a = d_t + d_s \), where \( d_t \) is as above and

\[ d_s = \sum_{i=1}^{r-1} 1^\otimes i-1 \otimes \mu_A \otimes 1^\otimes r-i-1. \]
is the simplicial differential. Dually, given a simply connected dg-coalgebra $C$, the reduced cobar construction of $C$ is the dg-algebra $\Omega(A) = Ts^{-1}(C)$ with the concatenation product and differential $d_\Omega = d_t + d_c$, where $d_t$ is as above and

$$d_c = \sum_{i=1}^{n} (-1)^{i-1} i^{i-1} \Delta \otimes 1^{\otimes n-i}$$

The following connected commutative dg-coalgebras with null differential are important in our applications:

- The polynomial algebra $P(v, 2n), n \geq 1$, generated by $v$ of degree $2n$, product $v^i v^j = v^{i+j}$ and coproduct $\Delta(v^n) = \sum_{i+j=n} \frac{(i+j)!}{i!j!} v^i \otimes v^j$;
- The truncated polynomial algebra $Q_p(v, 2n) = P(v, 2n) / (v^p)$;
- The exterior algebra $E(u, 2n + 1), n \geq 0$, generated by $u$ of degree $2n + 1$, trivial product $u^2 = 0$ and trivial coproduct $\Delta(u) = u \otimes 1 + 1 \otimes u$;
- The divided power algebra $\Gamma(w, 2n), n \geq 1$, generated by $\gamma_1(w) = w$, product $\gamma_i \gamma_j = \frac{(i+j)!}{i!j!} \gamma_{i+j}$ and coproduct $\Delta(\gamma_k(u)) = \sum_{i+j=k} \gamma_i(u) \otimes \gamma_j(u)$.

The notion of $A_\infty$-algebra, introduced by J. D. Stasheff [15] in the sixties, generalizes the notion of an algebra that is “associative up to homotopy.” A dg-module $(M, d)$ together with a non-associative morphism $\mu_2 : M \otimes M \to M$ of degree zero that is compatible with $d$ is associative up to homotopy if there exists a morphism $\mu_3 : M^{\otimes 3} \to M$ of degree $+1$ such that $\mu_3 d + d \mu_3 = \mu_2(\mu_2 \otimes 1) - \mu_2(1 \otimes \mu_2)$. An $A_\infty$-coalgebra (resp. $A_\infty$-algebra) is a graded module $M$ together with a family of maps $\Delta_i : M \to M^{\otimes i}$ (resp. $\mu_i : M^{\otimes i} \to M$) of degree $i - 2$ such that for all $i \geq 1$

$$\sum_{n=1}^{i} \sum_{k=0}^{i-n} (-1)^{n+k+nk} (1^{\otimes n-k} \otimes \Delta_n \otimes 1^k) \Delta_{i-n+1} = 0.$$

(resp. $\sum_{n=1}^{i} \sum_{k=0}^{i-n} (-1)^{n+k+nk} \mu_{i-n+1}(1^k \otimes \mu_n \otimes 1^{\otimes i-n-k}) = 0.$)

Next we review the main facts from homological perturbation theory used in this paper. Let $(N, d_N)$ and $(M, d_M)$ be dg-modules. A contraction $c : \{N, M, f, g, \phi\}$ (see [4,14]) from $(N, d_N)$ to $(M, d_M)$ is a special type of homotopy equivalence given by morphisms $f : N \to M$ and $g : M \to N$ of degree zero and a homotopy operator $\phi : N \to N$ of degree $+1$ that satisfy the following conditions:

$$(c1) \ f g = 1_M, \quad (c2) \ \phi d_N + d_N \phi + gf = 1_N,$$

$$(c3) \ f \phi = 0, \quad (c4) \ \phi g = 0, \quad (c5) \ \phi \phi = 0.$$ 

Of course, the homology groups of $N$ are canonically isomorphic to the homology groups of $M$, but any additional algebraic structure on $N$ (such as a dg-algebra structure, for example) does not transfer to $M$ in general. In fact a contraction from a dg-(co)algebra to a dg-module induces an $A_\infty$-(co)algebra structure on $M$ [6,2] by means of a perturbation process.
A contraction \( c : \{ N, M, f, g, \phi \} \) between dg-modules induces the following contractions of suspensions and tensor modules \([5,6]\):

- The **suspension contraction** of \( c \), \( s \ c \):
  \[
  s \ c : \{ s N, s M, s f, s g, s \phi \}.
  \]

- The **tensor module contraction**, \( T(c) \):
  \[
  T(c) : \{ T(N), T(M), T(f), T(g), T(\phi) \},
  \]
  where
  \[
  T(\phi)|_{T^s(N)} = \phi^{[s_n]} = \sum_{i=0}^{n-1} 1^i \otimes \phi \otimes (gf)^{n-i-1}.
  \]

A fundamental tool in homological perturbation theory is the **Basic Perturbation Lemma** (BPL) \([14],[2]\), which is an algorithm whose input is a contraction \( c : \{ N, M, f, g, \phi \} \) and a perturbation datum \( \delta \) of \( c \) and whose output is a new contraction \( c_\delta \). The only requirement that the composition \( \phi \delta \) be pointwise nilpotence so that the sums involved in the formulas are finite for each \( x \in N \):

\[
\begin{align*}
\text{Input:} & \quad c : (N, d_N) \xrightarrow{f} (M, d_M) + \text{perturbation } \delta \\
\text{Output:} & \quad c_\delta : (N, d_N + \delta) \xrightarrow{f_\delta} (M, d_M + d_\delta)
\end{align*}
\]

where \( f_\delta, g_\delta, \phi_\delta, d_\delta \) are given by the formulas

\[
\begin{align*}
d_\delta &= f \delta \Sigma^\delta_c g; \\
f_\delta &= f (1 - \delta \Sigma^\delta_c \phi); \\
g_\delta &= \Sigma^\delta_c g; \\
\phi_\delta &= \Sigma^\delta_c \phi;
\end{align*}
\]

and \( \Sigma^\delta_c = \sum_{i \geq 0} (-1)^i (\phi \delta)^i \).

In fact, an \( A_\infty \)-structure is induced on the small dg-module \( M \) of the contraction \( c \) in the following way:

- As initial data, let \( c : \{ A, M, f, g, \phi \} \) be a contraction in which \( A \) is an algebra (or alternatively a coalgebra).
- Then \( c \) induces a contraction \( Ts(c) : \{ Ts(A), Ts(M), Ts(f), Ts(g), Ts(-\phi) \} \) of associated tensor modules.
- Given as perturbation datum of \( Ts(c) \) the simplicial differential \( d_\delta \), apply the Basic Perturbation Lemma and obtain a contraction \( b(c) : \{ \tilde{B}(A), \tilde{B}(M), \tilde{b}(f), \tilde{b}(g), \tilde{b}(\phi) \} \) from the reduced bar construction of \( A \) to the tilde-bar construction of \( M \).
- Finally, extract the differential induced on \( \tilde{B}(M) \), \( d_\delta \), where the \( A_\infty \)-algebra
structure induced on $M$ is given by $\delta$:

\[
\begin{align*}
    \mu_1 &= -d_{st}, \\
    \mu_n &= (-1)^{n+1} f \mu^{(1)} \phi^{[=2]} \mu^{(2)} \cdots \phi^{[=n-1]} \mu^{(n-1)} g^n, \quad n \geq 2
\end{align*}
\]

where

\[
\mu^{(k)} = \sum_{i=0}^{k-1} (-1)^{i+1} 1^0 \otimes \mu_A \otimes 1^i \otimes 1^{k-i-1}.
\]

Our program is organized as a module that enhances the Kenzo program $\delta$, developed by F. Sergeraert and some collaborators. Kenzo is a Common Lisp Object System (CLOS) that implements numerous algebraic structures including dg-algebras, simplicial groups, morphisms, contractions and chain complexes, and uses them to compute various topological invariants such as the homology groups of sophisticated spaces.

3. Algorithms to compute $\psi$-structures: ARAIA and CRAIC

The enhancement to Kenzo introduced here allows us to compute the $\psi$-(co)algebra structures induced on the small module of a contraction when the big dg-module is a (co)algebra. This program component consists of approximately 2000 lines of code and significantly enriches the set of Kenzo classes (see $\delta$, $\delta$), and it is important to mention that as far as we know it is the first computer program available to compute $\psi$-structures given by contractions. The most important problem we faced when creating this software was creating the appropriate data structures to handle the large quantity of data generated by the complicated formulas of the BPL.

3.1. Symbolic Representation

In our computational framework, we encode an $\psi$-coalgebra as a map of degree zero from $M$ to $T^a(M)$. When doing so, we actually ignore the degree of the maps. This allows us to write the code as simply as possible. But apart from that, some difficult problems must be solved:

“Infinit e” loops
The first problem is to translate mathematical categories into computational classes.

- Given coalgebras $C$ and $D$, the tensor product $C \otimes D$ is a coalgebra with co-product $\Delta : C \otimes D \rightarrow (C \otimes D)^{\otimes 2}$ given by $(1 \otimes T \otimes 1)(\Delta_C \otimes \Delta_D)$, where $T$ interchanges the second and third tensor factors.
- In our “computational world” there is a class coalgebra, which must respect tensor products as indicated above. Thus the program defines the morphism $\Delta$ by applying the rule. Realizing that $(C \otimes D)^{\otimes 2}$ must be a coalgebra, the program tries to define $\Delta$ recursively but in doing so falls into an infinite loop.
Coherence

- **Mathematical coherence:** An $A_\infty$-structure is much more complicated than a (co)algebra. Indeed, an $A_\infty$-algebra is an algebra in which associativity is satisfied *up to homotopy*.

- **Computational coherence:** We view the class of $A_\infty$-algebras as a *superclass* of the class of algebras, i.e., as a class of *more general* (“weaker”) objects.

Here is a summary of our solutions to these problems:

**Inheritance**
To avoid “infinite” loops we used a *lazy* programming style: The slot-unbound Lisp generic function allows us to implement a redundant slot *dynamically only when required* and thereby avoid infinite loops.

**Extension of the class family**
Since a dg-algebra induces a chain complex, we adopt the following notation:

- CHCM=chain complex; A=dg-algebra; C=dg-coalgebra; HA=dg-Hopf algebra.
- AA=$A_\infty$-algebra; AC=$A_\infty$-coalgebra.

**Schedule**
The old (unenhanced) structure of Kenzo expressed graphically is:

![Diagram](image)

To obtain the $A_\infty$-structure induced by a contraction, we take as *input* a contraction, where the top chain complex is a dg-(co)algebra. Applying ARAIA (resp. CRAIC), the output is the same contraction, but the bottom chain complex now has an explicit $A_\infty$-structure. We wish to compare the $A_\infty$-structure induced by a contraction with the trivial one whenever the bottom chain complex has non-trivial $A_\infty$-structure. Thus we must modify the classes and add some new ones. We adopt the following additional notation:

- AAAA=an object with a double $A_\infty$-algebra; analogously AAAC, ACAC and ACAA.
- DAA=an algebra plus an $A_\infty$-algebra; analogously DCA, DAC, DCC, DCHA, DAHA.
Theoretical steps
Given a contraction \( c : N \rightarrow M \) from a dg-(co)algebra \( N \), CRAIC,Coalgebra Reduction A-Infinity Coalgebra (resp. ARAIA, Algebra Reduction A-Infinity Algebra) is a CLOS method which, roughly speaking, follows this plan:

- Take as input a contraction \( c : N \rightarrow M \).
- Check whether or not \( N \) is a dg-coalgebra.
- If so, create a new contraction \( T^{-1}c : T(N) \rightarrow T(M) \).
- Using the appropriate perturbation \( \delta \) that in this case is the cosimplicial differential \( d_{c} \) (see [15] for the details), produce a new contraction \( \Omega c : \Omega N \rightarrow (\Omega M, d_{T} + d_{\delta}) \) from the reduced cobar construction of \( N \), i.e., the tensor algebra of \( N \) with a perturbed differential, to the tilde-cobar construction of \( M \),
- Extract the new differential \( d_{\delta} \) induced on the small complex and transform it into a collection of maps defining the induced \( A_{\infty} \)-coalgebra structure on \( M \).
- Finally CRAIC returns the original contraction \( c : N \rightarrow M \) having modified \( M \) from an object in the class of dg-modules to an object in the class of \( A_{\infty} \)-coalgebras.

More precisely, the object \( M \) has the same components as before plus a new one \texttt{imcprd} (Induced Multi CoPRoDuct), which is a map from \( M \) to \( TM \) given by summing the \( \Delta_{i} \)'s with appropriate signs. The \texttt{imcprd} component is a functional object; if we wish to study the induced \( A_{\infty} \)-coalgebra structure on \( M \), we can apply this functional object to generators or some combination thereof and examine the results. This component is implemented by a Lisp statement formatted as follows:

\[
\text{(imcprd module (degree of element) element)}.
\]

The output is a combination \( \text{(cmbn)} \) of elements of \( M^{\otimes n} \) with \( n \geq 2 \). For example, \( x = x_{1} \otimes x_{2} \otimes x_{3} \in M^{\otimes 3} \) is coded as
Let us consider two examples. The first is trivial: Given a dg-coalgebra, we compute the $A_\infty$-coalgebra structure induced by an automorphism. The second is more interesting and shows the transference from the bar construction of a truncated polynomial algebra (that is a dg-coalgebra) to the tensor product $E \otimes \Gamma$ of an exterior algebra $E$ with a divided power algebra, $\Gamma$.

### 3.2. Example 1: The trivial contraction $P(u, 2) \to P(u, 2)$

Let $P(u, 2)$ be the polynomial algebra on one generator $u$ of degree 2 with coefficients in $\mathbb{Z}$ and consider the automorphism given by the identity on $P$, which induces a trivial contraction $c : \{P(u, 2), P(u, 2), 1_p, 1_p, 0\}$. With the exception of $\Delta_2$, which is the original coproduct, all $A_\infty$-components $\Delta_i$ induced on $P(u, 2^n)$ vanish. Thus the program returns the initial coalgebra. To create this algebra, run:

```
(setf p (plnm_algb 0 2)) ← PoLyNoMial ALGeBra in Z, deg(u)=2.


> (setf r (trivial-rdct p)) ← Construction of the trivial contraction.

[K9 Reduction K1 => K1] ← Applying CRAIC.

> (setf m (bcc r)) ← Ask for the small dg-module.

[K1 double-a-infty-clgb-h-algb] ← The class changes from hopf-alg to hopf-alg+induced $A_\infty$-coalg.

> (imcprd m 2 1) ← $A_\infty$-coalg. over $u$?

----------------------{CMBN 2}

<-1 * <<Mtnpr [0 0] [2 1]>>>
<-1 * <<Mtnpr [2 1] [0 0]>>>

----------------------

> (imcprd m 4 2) ← $A_\infty$-coalg. over $u^2$?

----------------------{CMBN 4}

<-1 * <<Mtnpr [0 0] [4 2]>>>
<-1 * <<Mtnpr [2 1] [2 1]>>>
<-1 * <<Mtnpr [4 2] [0 0]>>>

----------------------

> (imcprd m 6 3) ← $A_\infty$-coalg. over $u^3$?

----------------------{CMBN 6}

<-1 * <<Mtnpr [0 0] [6 3]>>>
<-1 * <<Mtnpr [2 1] [4 2]>>>
<-1 * <<Mtnpr [4 2] [2 1]>>>
<-1 * <<Mtnpr [6 3] [0 0]>>>

----------------------
```

Evidently the machine returns the coproduct of $P(u, 2)$ as the induced $A_\infty$-coalgebra structure.

The computations were performed by a processor AMD 3000, with 512mb of memory and 40gb of hard disc. The time required to compute the maps was inde-
3.3. Example 2: The contraction $\hat{B}(Q_5(u, 2)) \to E(v, 3) \otimes \Gamma(w, 12)$

Let us consider the contraction from the reduced bar construction of a truncated polynomial algebra $Q_5(u, 2n)$ to the tensor product $E \otimes \Gamma$ of an exterior algebra and a divided power algebra. Denoting an element $[u^r \cdots | u^m]$ of $\hat{B}(Q_5(u, 2n))$ by its exponents $[r_1 \cdots | r_m]$, $0 \leq r_i < 5$, the morphisms $f$, $g$ and $\phi$ are given by the following explicit formulas (see [11, 14, 12]):

\[
\begin{align*}
    f[r_1|t_1| \cdots |r_m|t_m] &= \{ \prod_{k=1}^m \delta_{p,r_k+t_k} \} \gamma_m(w), \\
    f[r_1|t_1| \cdots |r_m|t_m] &= \delta_{1,t} \{ \prod_{k=1}^m \delta_{p,r_k+t_k} \} v \otimes \gamma_m(w),
\end{align*}
\]

where $\delta_{i,j}$ is the Kronecker delta, and

\[
g(v) = [1], \quad g(\gamma_k(w)) = [1 | p - 1 \cdots | p - 1].
\]

Monomials in these algebras are encoded as above, i.e., by sequences of exponents corresponding to polynomials of the generator in successive factors of the “bar” tensor product.
It is important to observe that the only non-vanishing structure maps in the induced $A_{\infty}$-coalgebra structure on $E(u,3) \otimes \Gamma(w,12)$ are $\Delta_2$ and $\Delta_5$. Experimentally we obtain the formula

$$\Delta_5(u^i \otimes \gamma_j(w)) = \sum_{k_1 + \cdots + k_p = j-1} u^{i+1} \gamma_{k_1}(w) \otimes \cdots \otimes u^{i+p} \gamma_{k_p}(w), \quad \text{with } u \in \{0,1\}.$$ 

Here is a time-table for computing $\Delta_i$ as a function of the degree $d$ of the generator:

<table>
<thead>
<tr>
<th>Time used (Seconds)</th>
<th>$d = 3$</th>
<th>$d &lt; 48$</th>
<th>$d = 48$</th>
<th>$d = 60$</th>
<th>$d = 63$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta_i's$</td>
<td>0, 15''</td>
<td>&lt; 1''</td>
<td>53''</td>
<td>20h 28' 30''</td>
<td>breaks</td>
</tr>
</tbody>
</table>

Conclusions and future work

This program gives useful hints when studying the delicate properties of $A_{\infty}$-structures obtained from various processes. One could use the program to study the tensor product of $A_{\infty}$-structures, for example. A future and important extension of this work will be to compute the induced structure maps on the small dg-module of a contraction when the big initial dg-module is a dg-Hopf algebra.

References


