homalg: First steps to an abstract package for homological algebra

Mohamed Barakat Daniel Robertz

Abstract

Homological algebra is a natural extension of the theory of modules over rings. The category of modules and their homomorphisms is replaced by the category of chain complexes and their chain maps. A module is represented by any of its resolutions. The module is then recovered as the only non-trivial homology of the resolution. The notions of derived functors and their homologies, connecting homomorphism and the resulting long exact homology sequences play a central role in homological algebra.

The MAPLE-package homalg [1, 2] provides a way to deal with these powerful notions. The package is abstract in the sense that it is independent of any specific ring arithmetic. If one specifies a ring, not necessarily commutative, in which one can solve the ideal membership problem and compute syzygies, the above homological algebra constructions over that ring become accessible using homalg.

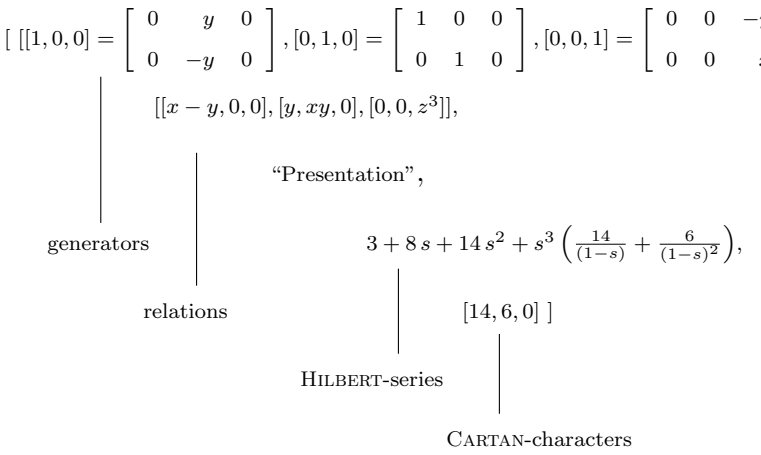
As the name of this package suggests, our intention has been to make as much as possible of the basic homological machinery available in a computer algebra system without the need to specify the ring of operators from the beginning.

Introduction

The basic objects of homalg are finitely presented left modules over rings in which the ideal membership problem is algorithmically solvable and syzygies are effectively computable. We call such rings *computable*. homalg implements the homological constructions for modules over such rings, whereas the ring arithmetic has be to provided by a ring-specific package. The following ring-specific packages have successfully been used with homalg: Involutive and Janet [3], JanetOre, OreModules [4]. PIR is one more tiny package, that makes MAPLE's builtin facilities for dealing with principal ideal rings such as the ring of integers, univariate polynomial rings over fields and all their residue class rings available to homalg. PIR uses the Smith normal form to bring modules to a standard form.

The central objects in homalg are functors. Functors map on the one hand objects of a source category to objects of a target category, and on the other hand morphisms between two objects in the source category to morphisms between their images in the target category in a compatible way. The two most important functors are the Hom-functor and the tensor product functor $\otimes$ and their derived functors, the definition of which will be reproduced below.

Figure 1: A module of homomorphisms between two modules over $D = \mathbb{Q}[x, y, z]$ with INVOLUTIVE



A major effort in the implementation was to find the suitable scheme for realizing the functor part on objects in order to have a unified way in extracting the part of the functor on morphisms. Composition and derivation of functors in `homalg` rely exclusively on this and define again functors. I.e. extracting the morphism part of composed or derived functors is done in the same unified way as for all functors. Hence, using the two basic operations of composing and deriving functors, the user can without effort add new functors to those already existing in `homalg`.

Given a (covariant) functor F the i-th left derivation of F is as usual denoted by L_iF. A short exact sequence $0 \to M' \to M \to M'' \to 0$ of modules then gives rise to a long exact sequence connecting $L_iF(M') \to L_iF(M) \to L_iF(M'')$ and $L_{i+1}F(M') \to L_{i+1}F(M) \to L_{i+1}F(M'')$ for all $i \geq 0$. These so-called connecting homomorphisms are implemented in `homalg`.

1 Finitely presented modules

`homalg` can only deal with finitely presented modules. A finitely presented D-module M is a quotient of a free module of finite rank $D^{1\times l_0}$ by a finitely generated submodule $D^{1\times l_1}A = \mathrm{im}(.A)$, where $A \in D^{l_1\times l_0}$:

$$M = D^{1\times l_0}/D^{1\times l_1}A = \mathrm{coker}(.A).$$

As usual a presentation is given by generators and relations. A presentation of a module in `homalg` is a list containing as first entry the list of generators and as second entry the list of relations. The third entry is a string delimiter to optically indicate the end of the presentation. This string, unless changed by the user, defaults to `"Presentation"`. The

remaining entries provide extra information about the presented module, e.g. its HILBERT-series. This extra information can only be provided by the ring-specific package.

In the list of generators the concrete generators are numbered by abstract generators being the l_0 standard basis vectors of the underlying free module $D^{1\times l_0}$. The list of relations simply contains the rows of the matrix A. An example is given in Figure 1.

2 An Example

Here we demonstrate on a simple example some main procedures in `homalg`. We start with the principal ideal ring $D := \mathbb{Z}/2^8\mathbb{Z}$ and the short exact sequence of modules

$$(0 \to M' \to M \to M'' \to 0) = (0 \to \mathbb{Z}/2^2\mathbb{Z} \to \mathbb{Z}/2^5\mathbb{Z} \to \mathbb{Z}/2^3\mathbb{Z} \to 0).$$

The functor we want to derive is the composed functor

$$F := \mathrm{Hom}(\mathrm{Hom}(-, K), L) = \mathrm{Hom}(\mathrm{Hom}(-, \mathbb{Z}/2^7\mathbb{Z}), \mathbb{Z}/2^4\mathbb{Z})$$

. We compute the long exact sequence of the left derived functors $\mathrm{L}_i F$:

```
> restart;
> with(PIR): with(homalg):
> RPP:='PIR/homalg':  'homalg/default':=RPP;
```

$$homalg/default := PIR/homalg$$

```
> var:=[[],[2^8]]:  Pvar(var);
```

$$[\text{“Z”}, 256]$$

```
> M:=Cokernel([2^5],var); alpha2:=matrix([[1]]);
> _M:=Cokernel([2^3],var);
```

$$M := [[1 = 1], [32], \text{“Presentation”}, [32], 0]$$

$$\alpha 2 := \begin{bmatrix} 1 \end{bmatrix}$$

$$\_M := [[1 = 1], [8], \text{“Presentation”}, [8], 0]$$

```
> M_:=Kernel(M,alpha2,_M,var); alpha1:=KernelEmb(M,alpha2,_M,var);
```

$$M\_ := [[1 = 8], [4], \text{“Presentation”}, [4], 0]$$

$$\alpha 1 := \begin{bmatrix} 8 \end{bmatrix}$$

```
> IsShortExactSeq(M_,alpha1,M,alpha2,_M,var);
```

$$true$$

```
> K:=Cokernel([2^7],var); L:=Cokernel([2^4],var);
```

$$K := [[1 = 1], [128], \text{“Presentation”}, [128], 0]$$

$$L := [[1 = 1], [16], \text{“Presentation”}, [16], 0]$$

```
> GlobalDim(var); q:=4;
```

$$\infty$$

$$q := 4$$

```
> seqs:=ResolveShortExactSeq(q,M_,alpha1,M,alpha2,_M,var,"TRUNCATE"):
> Seqs:=HomHomOnSeqs(K,L,seqs,var):
> LEHS:=LongExactHomologySeq(Seqs,var):
> map(a->LHomHomMap(a,M_,alpha1,M,K,L,var),[$0..q]);
> map(a->LHomHomMap(a,M,alpha2,_M,K,L,var),[$0..q]);
```

$$[[\ 8\ ],\ [\ 4\ ],\ [\ 4\ ],\ [\ 4\ ],\ [\ 4\ ]]$$

$$[[\ 1\ ],\ [\ 2\ ],\ [\ 2\ ],\ [\ 2\ ],\ [\ 2\ ]]$$

We obtain the long exact sequence of derived functors:

$$0 \leftarrow \mathbb{Z}/8\mathbb{Z} \overset{(7)}{\leftarrow} \mathbb{Z}/16\mathbb{Z} \overset{(8)}{\leftarrow} \mathbb{Z}/4\mathbb{Z} \overset{(2)}{\leftarrow} \mathbb{Z}/8\mathbb{Z} \overset{(6)}{\leftarrow} \mathbb{Z}/8\mathbb{Z} \overset{(4)}{\leftarrow} \mathbb{Z}/4\mathbb{Z} \overset{(2)}{\leftarrow} \mathbb{Z}/8\mathbb{Z} \overset{(6)}{\leftarrow} \mathbb{Z}/8\mathbb{Z} \overset{(4)}{\leftarrow} \cdots \text{periodic}$$

```
> lehs:=LEHS2lehs(LEHS);
```

$lehs :=$ [[[1 = [1]], [8], "Presentation", [8], 0], [7],

[[1 = $\begin{bmatrix} 7 \\ 249 \end{bmatrix}$], [16], "Presentation", [16], 0], [8],

[[1 = [1]], [4], "Presentation", [4], 0], [2], [[1 = [2]], [8], "Presentation", [8], 0],

[6], [[1 = $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$], [8], "Presentation", [8], 0], [4],

[[1 = [4]], [4], "Presentation", [4], 0], [2], [[1 = [1]], [8], "Presentation", [8], 0],

[6], [[1 = $\begin{bmatrix} 6 \\ 250 \end{bmatrix}$], [8], "Presentation", [8], 0], [4],

[[1 = [1]], [4], "Presentation", [4], 0], [2], [[1 = [2]], [8], "Presentation", [8], 0],

[6], [[1 = $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$], [8], "Presentation", [8], 0], [4],

[[1 = [4]], [4], "Presentation", [4], 0], [2], [[1 = [1]], [16], "Presentation", [16], 0],

$\begin{bmatrix} 14 \\ 0 \end{bmatrix}$, [[[1, 0] = $\begin{bmatrix} 254 \\ 2 \end{bmatrix}$, [0, 1] = $\begin{bmatrix} 0 \\ 15 \end{bmatrix}$], [[8, 0], [0, 16]], "Presentation", [8, 16], 0], [0 1],

[[1 = [1]], [16], "Presentation", [16], 0]]

```
> IsExactSeq(lehs,var,"VERBOSE");
```

true

References

[1] M. Barakat, D. Robertz, *Computing invariants of multidimensional linear systems on an abstract homological level*, to appear, proceedings MTNS 2006, Japan.

[2] M. Barakat, D. Robertz, `homalg`*: An abstract package for homological algebra*, in prep.

[3] Y. A. Blinkov, C. F. Cid, V. P. Gerdt, W. Plesken, D. Robertz, *The MAPLE Package "Janet": I. Polynomial Systems. II. Linear Partial Differential Equations.* Proc. 6th Int. Workshop on Computer Algebra in Scientific Computing, Passau, 2003. Cf. also `http://wwwb.math.rwth-aachen.de/Janet`.

[4] F. Chyzak, A. Quadrat, D. Robertz, OreModules project, `http://wwwb.math.rwth-aachen.de/OreModules`.

[5] P. J. Hilton, U. Stammbach, *A Course in Homological Algebra*, second edition, Springer, 1997.