RingsForHomalg

Rings in External Computer Algebra Systems for the GAP-Package homalg

Version 2009.11.09

November 2009

Mohamed Barakat

Simon Goertzen
Markus Kirschmer
Markus Lange-Hegermann
Max Neunhöffer
Daniel Robertz

(*this manual is still under construction*)

This manual is best viewed as an HTML document. The latest version is available ONLINE at:

http://homalg.math.rwth-aachen.de/~barakat/RingsForHomalg/chap0.html

An OFFLINE version should be included in the documentation subfolder of the package. This package is part of the homalg-project:

http://homalg.math.rwth-aachen.de/index.php/core-packages/ringsforhomalg

Mohamed Barakat
— Email: barakat@mathematik.uni-kl.de
— Homepage: http://wwwb.math.rwth-aachen.de/~barakat/

— Address: Department of Mathematics,
University of Kaiserslautern,
67653 Kaiserslautern,
Germany

Simon Goertzen — Email: simon.goertzen@rwth-aachen.de
— Homepage: http://wwwb.math.rwth-aachen.de/~simon
— Address: Lehrstuhl B für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

Markus Kirschmer — Email: markus.kirschmer@math.rwth-aachen.de
— Homepage: http://www.math.rwth-aachen.de/~Markus.Kirschmer
— Address: Lehrstuhl D für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

Markus Lange-Hegermann — Email: markus.lange-hegermann@rwth-aachen.de
— Homepage: http://wwwb.math.rwth-aachen.de/~markus
— Address: Lehrstuhl B für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

Max Neunhöffer — Email: daniel@momo.math.rwth-aachen.de
— Homepage: http://wwwb.math.rwth-aachen.de/~daniel
— Address: St Andrews University, School of Mathematics and Statistics, Mathematical Institute, North Haugh, St Andrews, Fife KY16 9SS, Scotland, UK

Daniel Robertz — Email: daniel@momo.math.rwth-aachen.de
— Homepage: http://wwwb.math.rwth-aachen.de/~daniel
— Address: Lehrstuhl B für Mathematik, RWTH Aachen, Templergraben 64, 52056 Aachen, Germany

Copyright

Contents

Chapter 1

Introduction

This package is part of the homalg project [ht09]. The role of the package is described in the manual of the homalg package.

1.1 Ring Constructions for Supported External Computer Algebra Systems

Here are some of the supported ring constructions:

1.1.1 external GAP

Example

```
gap> ZZ := HomalgRingOfIntegersInExternalGAP( );
<A homalg external ring residing in the CAS GAP>
 gap> Display( ZZ );
 Z
gap> F2 := HomalgRingOfIntegersInExternalGAP( 2, ZZ );
<A homalg external ring residing in the CAS GAP>
 gap> Display( F2 );
 GF(2)
```

`F2 := HomalgRingOfIntegersInExternalGAP( 2 )` would launch another GAP.

Example

```
gap> Z4 := HomalgRingOfIntegersInExternalGAP( 4, ZZ );
<A homalg external ring residing in the CAS GAP>
 gap> Display( Z4 );
 Z/4Z
gap> Z_4 := HomalgRingOfIntegersInExternalGAP( ZZ ) / 4;
<A homalg residue class ring>
gap> Display( Z_4 );
Z/( 4 )
gap> Q := HomalgFieldOfRationalsInExternalGAP( ZZ );
<A homalg external ring residing in the CAS GAP>
 gap> Display( Q );
 Q
```

1.1.2 Singular

Example
```
gap> F2 := HomalgRingOfIntegersInSingular( 2 );
<A homalg external ring residing in the CAS Singular>
 gap> Display( F2 );
 GF(2)
gap> Q := HomalgFieldOfRationalsInSingular( F2 );
<A homalg external ring residing in the CAS Singular>
 gap> Display( Q );
 Q
```

`Q := HomalgFieldOfRationalsInSingular( )` would launch another Singular.

Example
```
gap> F2xyz := F2 * "x,y,z";
<A homalg external ring residing in the CAS Singular>
 gap> Display( F2xyz );
 GF(2)[x,y,z]
gap> Qxyz := Q * "x,y,z";
<A homalg external ring residing in the CAS Singular>
 gap> Display( Qxyz );
 Q[x,y,z]
```

1.1.3 MAGMA

Example
```
gap> ZZ := HomalgRingOfIntegersInMAGMA( );
<A homalg external ring residing in the CAS MAGMA>
 gap> Display( ZZ );
 Z
gap> F2 := HomalgRingOfIntegersInMAGMA( 2, ZZ );
<A homalg external ring residing in the CAS MAGMA>
 gap> Display( F2 );
 GF(2)
```

`F2 := HomalgRingOfIntegersInMAGMA( 2 )` would launch another MAGMA.

Example
```
gap> Z_4 := HomalgRingOfIntegersInMAGMA( ZZ ) / 4;
<A homalg residue class ring>
gap> Display( Z_4 );
Z/( 4 )
gap> Q := HomalgFieldOfRationalsInMAGMA( ZZ );
<A homalg external ring residing in the CAS MAGMA>
 gap> Display( Q );
 Q
gap> F2xyz := F2 * "x,y,z";
<A homalg external ring residing in the CAS MAGMA>
 gap> Display( F2xyz );
 GF(2)[x,y,z]
gap> Qxyz := Q * "x,y,z";
<A homalg external ring residing in the CAS MAGMA>
 gap> Display( Qxyz );
 Q[x,y,z]
```

1.1.4 Macaulay2

Example
```
gap> ZZ := HomalgRingOfIntegersInMacaulay2( );
<A homalg external ring residing in the CAS Macaulay2>
 gap> Display( ZZ );
 Z
gap> F2 := HomalgRingOfIntegersInMacaulay2( 2, ZZ );
<A homalg external ring residing in the CAS Macaulay2>
 gap> Display( F2 );
 GF(2)
```

`F2 := HomalgRingOfIntegersInMacaulay2( 2 )` would launch another Macaulay2.

Example
```
gap> Z_4 := HomalgRingOfIntegersInMacaulay2( ZZ ) / 4;
<A homalg residue class ring>
gap> Display( Z_4 );
Z/( 4 )
gap> Q := HomalgFieldOfRationalsInMacaulay2( ZZ );
<A homalg external ring residing in the CAS Macaulay2>
 gap> Display( Q );
 Q
gap> F2xyz := F2 * "x,y,z";
<A homalg external ring residing in the CAS Macaulay2>
 gap> Display( F2xyz );
 GF(2)[x,y,z]
gap> Qxyz := Q * "x,y,z";
<A homalg external ring residing in the CAS Macaulay2>
 gap> Display( Qxyz );
 Q[x,y,z]
```

1.1.5 Sage

Example
```
gap> ZZ := HomalgRingOfIntegersInSage( );
<A homalg external ring residing in the CAS Sage>
 gap> Display( ZZ );
 Z
gap> F2 := HomalgRingOfIntegersInSage( 2, ZZ );
<A homalg external ring residing in the CAS Sage>
 gap> Display( F2 );
 GF(2)
```

`F2 := HomalgRingOfIntegersInSage( 2 )` would launch another Sage.

Example
```
gap> Z_4 := HomalgRingOfIntegersInSage( ZZ ) / 4;
<A homalg residue class ring>
gap> Display( Z_4 );
Z/( 4 )
gap> Q := HomalgFieldOfRationalsInSage( ZZ );
<A homalg external ring residing in the CAS Sage>
 gap> Display( Q );
 Q
```

```
gap> F2x := F2 * "x";
<A homalg external ring residing in the CAS Sage>
 gap> Display( F2x );
 GF(2)[x]
gap> Qx := Q * "x";
<A homalg external ring residing in the CAS Sage>
 gap> Display( Qx );
 Q[x]
```

1.1.6 Maple

Example

```
gap> ZZ := HomalgRingOfIntegersInMaple( );
<A homalg external ring residing in the CAS Maple>
 gap> Display( ZZ );
 Z
gap> F2 := HomalgRingOfIntegersInMaple( 2, ZZ );
<A homalg external ring residing in the CAS Maple>
 gap> Display( F2 );
 GF(2)
```

`F2 := HomalgRingOfIntegersInMaple( 2 )` would launch another Maple.

Example

```
gap> Z4 := HomalgRingOfIntegersInMaple( 4, ZZ );
<A homalg external ring residing in the CAS Maple>
 gap> Display( Z4 );
 Z/4Z
gap> Z_4 := HomalgRingOfIntegersInMaple( ZZ ) / 4;
<A homalg residue class ring>
gap> Display( Z_4 );
Z/( 4 )
gap> Q := HomalgFieldOfRationalsInMaple( ZZ );
<A homalg external ring residing in the CAS Maple>
 gap> Display( Q );
 Q
gap> F2xyz := F2 * "x,y,z";
<A homalg external ring residing in the CAS Maple>
 gap> Display( F2xyz );
 GF(2)[x,y,z]
gap> Qxyz := Q * "x,y,z";
<A homalg external ring residing in the CAS Maple>
 gap> Display( Qxyz );
 Q[x,y,z]
```

Chapter 2

Installation of the RingsForHomalg Package

To install this package just extract the package's archive file to the GAP `pkg` directory.

By default the RingsForHomalg package is not automatically loaded by GAP when it is installed. You must load the package with

```
LoadPackage( "RingsForHomalg" );
```

before its functions become available.

Please, send us an e-mail if you have any questions, remarks, suggestions, etc. concerning this package. Also, we would be pleased to hear about applications of this package.

The authors.

Chapter 3

The Ring Table

3.1 An Example for a Ring Table - Singular

todo: introductory text, mention: transposed matrices, the macros, refer to the philosophy

3.1.1 BasisOfRowModule (in the homalg table for Singular)

◊ BasisOfRowModule(*M*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro BasisOfRowModule (3.1.2) inside the computer algebra system.

Code

```
BasisOfRowModule :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      "unknown_number_of_rows",
      NrColumns( M ),
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = BasisOfRowModule(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.BasisOfModule
    );

    return N;

  end,
```

3.1.2 BasisOfRowModule (Singular macro)

◊ BasisOfRowModule(*M*) (function)

Returns:

Code
```
    BasisOfRowModule := "\n\
proc BasisOfRowModule (matrix M)\n\
{\n\
  return(std(M));\n\
}\n\n",
```

3.1.3 BasisOfColumnModule (in the homalg table for Singular)

◇ `BasisOfColumnModule(M)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `BasisOfColumnModule` (3.1.4) inside the computer algebra system.

Code
```
BasisOfColumnModule :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      NrRows( M ),
      "unknown_number_of_columns",
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = BasisOfColumnModule(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.BasisOfModule
    );

    return N;

  end,
```

3.1.4 BasisOfColumnModule (Singular macro)

◇ `BasisOfColumnModule(M)` (function)

Returns:

Code
```
    BasisOfColumnModule := "\n\
proc BasisOfColumnModule (matrix M)\n\
{\n\
  return(Involution(BasisOfRowModule(Involution(M))));\n\
}\n\n",
```

3.1.5 DecideZeroRows (in the homalg table for Singular)

◇ `DecideZeroRows(A, B)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `DecideZeroRows` (3.1.6) inside the computer algebra system.

Code
```
DecideZeroRows :=
  function( A, B )
    local N;

    N := HomalgVoidMatrix(
      NrRows( A ),
      NrColumns( A ),
      HomalgRing( A )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = DecideZeroRows(", A, B, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.DecideZero
    );

    return N;

  end,
```

3.1.6 DecideZeroRows (Singular macro)

◊ `DecideZeroRows(A, B)` (function)

Returns:

Code
```
    DecideZeroRows := "\n\
proc DecideZeroRows (matrix A, matrix B)\n\
{\n\
  return(reduce(A,B));\n\
}\n\n",
```

3.1.7 DecideZeroColumns (in the homalg table for Singular)

◊ `DecideZeroColumns(A, B)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `DecideZeroColumns` (3.1.8) inside the computer algebra system.

Code
```
DecideZeroColumns :=
  function( A, B )
    local N;

    N := HomalgVoidMatrix(
      NrRows( A ),
      NrColumns( A ),
      HomalgRing( A )
    );
```

```
    homalgSendBlocking(
      [ "matrix ", N, " = DecideZeroColumns(", A, B, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.DecideZero
    );

    return N;

  end,
```

3.1.8 DecideZeroColumns (Singular macro)

◊ DecideZeroColumns(*A*, *B*) (function)

Returns:

```
    DecideZeroColumns := "\n\
proc DecideZeroColumns (matrix A, matrix B)\n\
{\n\
  return(Involution(reduce(Involution(A),Involution(B))));\n\
}\n\n",
```

3.1.9 SyzygiesGeneratorsOfRows (in the homalg table for Singular)

◊ SyzygiesGeneratorsOfRows(*M*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro SyzygiesGeneratorsOfRows (3.1.10) inside the computer algebra system.

```
SyzygiesGeneratorsOfRows :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      "unknown_number_of_rows",
      NrRows( M ),
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = SyzygiesGeneratorsOfRows(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.10 SyzygiesGeneratorsOfRows (Singular macro)

◇ SyzygiesGeneratorsOfRows(*M*) (function)

Returns:

```
Code
    SyzygiesGeneratorsOfRows := "\n\
proc SyzygiesGeneratorsOfRows (matrix M)\n\
{\n\
  return(SyzForHomalg(M));\n\
}\n\n",
```

3.1.11 SyzygiesGeneratorsOfColumns (in the homalg table for Singular)

◇ SyzygiesGeneratorsOfColumns(*M*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro SyzygiesGeneratorsOfColumns (3.1.12) inside the computer algebra system.

```
Code
SyzygiesGeneratorsOfColumns :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      NrColumns( M ),
      "unknown_number_of_columns",
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = SyzygiesGeneratorsOfColumns(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.12 SyzygiesGeneratorsOfColumns (Singular macro)

◇ SyzygiesGeneratorsOfColumns(*M*) (function)

Returns:

```
Code
    SyzygiesGeneratorsOfColumns := "\n\
proc SyzygiesGeneratorsOfColumns (matrix M)\n\
{\n\
  return(Involution(SyzForHomalg(Involution(M))));\n\
}\n\n",
```

3.1.13 BasisOfRowsCoeff (in the homalg table for Singular)

◊ `BasisOfRowsCoeff(M, T)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `BasisOfRowsCoeff` (3.1.14) inside the computer algebra system.

```
BasisOfRowsCoeff :=
  function( M, T )
    local v, N;

    v := homalgStream( HomalgRing( M ) )!.variable_name;

    N := HomalgVoidMatrix(
      "unknown_number_of_rows",
      NrColumns( M ),
      HomalgRing( M )
    );

    homalgSendBlocking(
      [
        "list ", v, "l=BasisOfRowsCoeff(", M, "); ",
        "matrix ", N, " = ", v, "l[1]; ",
        "matrix ", T, " = ", v, "l[2]"
      ],
      "need_command",
      HOMALG_IO.Pictograms.BasisCoeff
    );

    return N;

  end,
```

3.1.14 BasisOfRowsCoeff (Singular macro)

◊ `BasisOfRowsCoeff(M, T)` (function)

Returns:

```
    BasisOfRowsCoeff := "\n\
proc BasisOfRowsCoeff (matrix M)\n\
{\n\
  matrix B = std(M);\n\
  matrix T = lift(M,B);\n\
  list l = B,T;\n\
  return(l)\n\
}\n\n",
```

3.1.15 BasisOfColumnsCoeff (in the homalg table for Singular)

◊ `BasisOfColumnsCoeff(M, T)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `BasisOfColumnsCoeff` (3.1.16) inside the computer algebra system.

```
BasisOfColumnsCoeff :=
  function( M, T )
    local v, N;

    v := homalgStream( HomalgRing( M ) )!.variable_name;

    N := HomalgVoidMatrix(
      NrRows( M ),
      "unknown_number_of_columns",
      HomalgRing( M )
    );

    homalgSendBlocking(
      [
        "list ", v, "l=BasisOfColumnsCoeff(", M, "); ",
        "matrix ", N, " = ", v, "l[1]; ",
        "matrix ", T, " = ", v, "l[2]"
      ],
      "need_command",
      HOMALG_IO.Pictograms.BasisCoeff
    );

    return N;

  end,
```

3.1.16 BasisOfColumnsCoeff (Singular macro)

◇ `BasisOfColumnsCoeff(M, T)` (function)

Returns:

```
    BasisOfColumnsCoeff := "\n\
proc BasisOfColumnsCoeff (matrix M)\n\
{\n\
  list l = BasisOfRowsCoeff(Involution(M));\n\
  matrix B = l[1];\n\
  matrix T = l[2];\n\
  l = Involution(B),Involution(T);\n\
  return(l);\n\
}\n\n",
```

3.1.17 DecideZeroRowsEffectively (in the homalg table for Singular)

◇ `DecideZeroRowsEffectively(A, B, T)` (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `DecideZeroRowsEffectively` (3.1.18) inside the computer algebra system.

Code
```
DecideZeroRowsEffectively :=
  function( A, B, T )
    local v, N;

    v := homalgStream( HomalgRing( A ) )!.variable_name;

    N := HomalgVoidMatrix(
      NrRows( A ),
      NrColumns( A ),
      HomalgRing( A )
    );

    homalgSendBlocking(
      [
        "list ", v, "l=DecideZeroRowsEffectively(", A, B, "); ",
        "matrix ", N, " = ", v, "l[1]; ",
        "matrix ", T, " = ", v, "l[2]"
      ],
      "need_command",
      HOMALG_IO.Pictograms.DecideZeroEffectively
    );

    return N;

  end,
```

3.1.18 DecideZeroRowsEffectively (Singular macro)

◊ DecideZeroRowsEffectively(*A, B, T*) (function)

Returns:

Code
```
    DecideZeroRowsEffectively := "\n\
proc DecideZeroRowsEffectively (matrix A, matrix B)\n\
{\n\
  matrix M = reduce(A,B);\n\
  matrix T = lift(B,M-A);\n\
  list l = M,T;\n\
  return(l);\n\
}\n\n",
```

3.1.19 DecideZeroColumnsEffectively (in the homalg table for Singular)

◊ DecideZeroColumnsEffectively(*A, B, T*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro `DecideZeroColumnsEffectively` (3.1.20) inside the computer algebra system.

Code
```
DecideZeroColumnsEffectively :=
  function( A, B, T )
    local v, N;
```

```
    v := homalgStream( HomalgRing( A ) )!.variable_name;

    N := HomalgVoidMatrix(
      NrRows( A ),
      NrColumns( A ),
      HomalgRing( A )
    );

    homalgSendBlocking(
      [
        "list ", v, "l=DecideZeroColumnsEffectively(", A, B, "); ",
        "matrix ", N, " = ", v, "l[1]; ",
        "matrix ", T, " = ", v, "l[2]"
      ],
      "need_command",
      HOMALG_IO.Pictograms.DecideZeroEffectively
    );

    return N;

  end,
```

3.1.20 DecideZeroColumnsEffectively (Singular macro)

◊ DecideZeroColumnsEffectively(*A, B, T*) (function)

Returns:

```
    DecideZeroColumnsEffectively := "\n\
  proc DecideZeroColumnsEffectively (matrix A, matrix B)\n\
  {\n\
    list l = DecideZeroRowsEffectively(Involution(A),Involution(B));\n\
    matrix B = l[1];\n\
    matrix T = l[2];\n\
    l = Involution(B),Involution(T);\n\
    return(l);\n\
  }\n\n",
```

3.1.21 RelativeSyzygiesGeneratorsOfRows (in the homalg table for Singular)

◊ RelativeSyzygiesGeneratorsOfRows(*M, M2*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro RelativeSyzygiesGeneratorsOfRows (3.1.22) inside the computer algebra system.

```
  RelativeSyzygiesGeneratorsOfRows :=
    function( M, M2 )
      local N;

      N := HomalgVoidMatrix(
        "unknown_number_of_rows",
```

```
      NrRows( M ),
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = RelativeSyzygiesGeneratorsOfRows(", M, M2, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.22 RelativeSyzygiesGeneratorsOfRows (Singular macro)

◊ RelativeSyzygiesGeneratorsOfRows(*M*, *M2*) (function)

Returns:

Code
```
    RelativeSyzygiesGeneratorsOfRows := "\n\
proc RelativeSyzygiesGeneratorsOfRows (matrix M1, matrix M2)\n\
{\n\
  return(std(modulo(M1, M2)));\n\
}\n\n",
```

3.1.23 RelativeSyzygiesGeneratorsOfColumns (in the homalg table for Singular)

◊ RelativeSyzygiesGeneratorsOfColumns(*M*, *M2*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro RelativeSyzygiesGeneratorsOfColumns (3.1.24) inside the computer algebra system.

Code
```
RelativeSyzygiesGeneratorsOfColumns :=
  function( M, M2 )
    local N;

    N := HomalgVoidMatrix(
      NrColumns( M ),
      "unknown_number_of_columns",
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = RelativeSyzygiesGeneratorsOfColumns(", M, M2, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.24 RelativeSyzygiesGeneratorsOfColumns (Singular macro)

◊ RelativeSyzygiesGeneratorsOfColumns(*M*, *M2*) (function)

Returns:

```
    RelativeSyzygiesGeneratorsOfColumns := "\n\
proc RelativeSyzygiesGeneratorsOfColumns (matrix M1, matrix M2)\n\
{\n\
  return(Involution(RelativeSyzygiesGeneratorsOfRows(Involution(M1),Involution(M2))));\n\
}\n\n",
```

3.1.25 ReducedSyzygiesGeneratorsOfRows (in the homalg table for Singular)

◊ ReducedSyzygiesGeneratorsOfRows(*M*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro ReducedSyzygiesGeneratorsOfRows (3.1.26) inside the computer algebra system.

```
ReducedSyzygiesGeneratorsOfRows :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      "unknown_number_of_rows",
      NrRows( M ),
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = ReducedSyzygiesGeneratorsOfRows(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.26 ReducedSyzygiesGeneratorsOfRows (Singular macro)

◊ ReducedSyzygiesGeneratorsOfRows(*M*) (function)

Returns:

```
    ReducedSyzForHomalg := "\n\
proc ReducedSyzForHomalg (matrix M)\n\
{\n\
  return(matrix(nres(M,2)[2]));\n\
}\n\n",
```

3.1.27 ReducedSyzygiesGeneratorsOfColumns (in the homalg table for Singular)

◊ ReducedSyzygiesGeneratorsOfColumns(*M*) (function)

Returns:

This is the entry of the homalg table, which calls the corresponding macro ReducedSyzygiesGeneratorsOfColumns (3.1.28) inside the computer algebra system.

```
ReducedSyzygiesGeneratorsOfColumns :=
  function( M )
    local N;

    N := HomalgVoidMatrix(
      NrColumns( M ),
      "unknown_number_of_columns",
      HomalgRing( M )
    );

    homalgSendBlocking(
      [ "matrix ", N, " = ReducedSyzygiesGeneratorsOfColumns(", M, ")" ],
      "need_command",
      HOMALG_IO.Pictograms.SyzygiesGenerators
    );

    return N;

  end,
```

3.1.28 ReducedSyzygiesGeneratorsOfColumns (Singular macro)

◊ ReducedSyzygiesGeneratorsOfColumns(*M*) (function)

Returns:

```
    ReducedSyzygiesGeneratorsOfRows := "\n\
proc ReducedSyzygiesGeneratorsOfRows (matrix M)\n\
{\n\
  return(ReducedSyzForHomalg(M));\n\
}\n\n",
```

References

[ht09] T. homalg team. *The homalg project*, 2003-2009. http://homalg.math.rwth-aachen.de/. 5

Index