

Contents

1	Resolutions of the ground ring	3
2	Resolutions of modules	4
3	Induced equivariant chain maps	5
4	Functors	6
5	Chain complexes	7
6	Homology and cohomology groups	8
7	Poincare series	9
8	Cohomology ring structure	10
9	Commutator and nonabelian tensor computations	11
10	Lie commutators and nonabelian Lie tensors	12
11	Generators and relators of groups	13
12	Orbit polytopes and fundamental domains	14
13	Cocycles	15
14	Words in free ZG-modules	16
15	FpG-modules	17
16	Meataxe modules	18
17	G-Outer Groups	19
18	Cat-1-groups	20
19	Coxeter diagrams and graphs of groups	21
20	Some functions for accessing basic data	22

	2
21 Parallel Computation - Core Functions	23
22 Parallel Computation - Extra Functions	24
23 Topological Data Analysis	25
24 Pseudo lists	26
25 Miscellaneous	27

Chapter 1

Resolutions of the ground ring

ResolutionAbelianGroup(L, n) ResolutionAbelianGroup(G, n) Inputs a list $L := [m_1, m_2, \dots, m_d]$ of nonneg
ResolutionAlmostCrystalGroup(G, n) Inputs a positive integer n and an almost crystallographic pcp group G . I
ResolutionAlmostCrystalQuotient(G, n, c) ResolutionAlmostCrystalQuotient($G, n, c, false$) An alm
ResolutionArtinGroup(D, n) Inputs a Coxeter diagram D and an integer $n > 1$. It returns n terms of a free ZG -re
ResolutionAsphericalPresentation(F, R, n) Inputs a free group F , a set R of words in F which constitute an a
ResolutionBieberbachGroup(G)
ResolutionBieberbachGroup(G, v) Inputs a Bieberbach group G (represented using AffineCrystGroupOnR
ResolutionDirectProduct(R, S) ResolutionDirectProduct($R, S, "internal"$) Inputs a ZG -resolution R
ResolutionExtension(g, R, S) ResolutionExtension($g, R, S, "TestFiniteness"$) ResolutionExter
ResolutionFiniteDirectProduct(R, S) ResolutionFiniteDirectProduct($R, S, "internal"$) Inputs a
ResolutionFiniteExtension($gensE, gensG, R, n$) ResolutionFiniteExtension($gensE, gensG, R, n, true$
ResolutionFiniteGroup($gens, n$) ResolutionFiniteGroup($gens, n, true$) ResolutionFiniteGroup(g
ResolutionFiniteSubgroup(R, K) ResolutionFiniteSubgroup($R, gensG, gensK$) Inputs a ZG -resolution fo
ResolutionGraphOfGroups(D, n) ResolutionGraphOfGroups(D, n, L) Inputs a graph of groups D and a p
ResolutionNilpotentGroup(G, n) ResolutionNilpotentGroup($G, n, "TestFiniteness"$) Inputs a nilpot
ResolutionNormalSeries(L, n) ResolutionNormalSeries($L, n, true$) ResolutionNormalSeries(L, n
ResolutionPrimePowerGroup(P, n) ResolutionPrimePowerGroup(G, n, p) Inputs a p -group P and integer
ResolutionSmallFpGroup(G, n) ResolutionSmallFpGroup(G, n, p) Inputs a small finitely presented group
ResolutionSubgroup(R, K) Inputs a ZG -resolution for an (infinite) group G and a subgroup K of finite index $|G : K|$
ResolutionSubnormalSeries(L, n) Inputs a positive integer n and a list $L = [L_1, \dots, L_k]$ of subgroups L_i of a fin
TwistedTensorProduct($R, S, EhomG, GmapE, NhomE, NEhomN, EltsE, Mult, InvE$) Inputs a ZG -resolution R , a ZN

Chapter 2

Resolutions of modules

| `ResolutionFpGModule (M, n)` Inputs an FpG -module M and a positive integer n . It returns n terms of a minimal fr

Chapter 3

Induced equivariant chain maps

| `EquivariantChainMap(R, S, f)` Inputs a ZG -resolution R , a ZG' -resolution S , and a group homomorphism $f : G \rightarrow G'$

Chapter 4

Functors

•

`HomToIntegers(X)` Inputs either a ZG -resolution $X = R$, or an equivariant chain map $X = (F : R \rightarrow S)$. It returns the chain complex $\text{Hom}(R, \mathbb{Z})$.

`HomToIntegersModP(R)` Inputs a ZG -resolution R and returns the cochain complex obtained by applying $\text{Hom}(R, \mathbb{Z}/p\mathbb{Z})$ to the resolution.

`HomToIntegralModule(R, f)` Inputs a ZG -resolution R and a group homomorphism $f : G \rightarrow GL_n(\mathbb{Z})$ to the group G . It returns the chain complex $\text{Hom}(R, \mathbb{Z})$ with the differential twisted by f .

`LowerCentralSeriesLieAlgebra(G)` `LowerCentralSeriesLieAlgebra(f)` Inputs a pcp group G . If each $g \in G$ is represented by a matrix M_g , then f is the map $f : G \rightarrow GL_n(\mathbb{Z})$ defined by $f(g) = M_g$. It returns the Lie algebra of the lower central series of G .

`TensorWithIntegers(X)` Inputs either a ZG -resolution $X = R$, or an equivariant chain map $X = (F : R \rightarrow S)$. It returns the chain complex $R \otimes \mathbb{Z}$.

`TensorWithIntegersModP(X, p)` Inputs either a ZG -resolution $X = R$, or an equivariant chain map $X = (F : R \rightarrow S)$. It returns the chain complex $R \otimes \mathbb{Z}/p\mathbb{Z}$.

`TensorWithRationals(R)` Inputs a ZG -resolution R and returns the chain complex obtained by tensoring with \mathbb{Q} .

Chapter 5

Chain complexes

`ChevalleyEilenbergComplex(X, n)` Inputs either a Lie algebra $X = A$ (over the ring of integers Z or over a field K) or a Leibniz algebra $X = A$ (over the ring of integers Z or over a field K)

Chapter 6

Homology and cohomology groups

`Cohomology(X, n)` Inputs either a cochain complex $X = C$ or a cochain map $X = (C \rightarrow D)$ over the integers Z to get the cohomology groups $H^n(X, Z)$.

`CohomologyPrimePart(C, n, p)` Inputs a cochain complex C in characteristic 0, a positive integer n , and a prime p . It returns the p -part of the cohomology group $H^n(C, Z)$.

`GroupCohomology(X, n)` `GroupCohomology(X, n, p)` Inputs a positive integer n and either a finite group $X = G$ or a Coxeter diagram $X = D$ to get the cohomology groups $H^n(G, Z)$ or $H^n(D, Z)$.

`GroupHomology(X, n)` `GroupHomology(X, n, p)` Inputs a positive integer n and either a finite group $X = G$ or a Coxeter diagram $X = D$ to get the homology groups $H_n(G, Z)$ or $H_n(D, Z)$.

`Homology(X, n)` Inputs either a chain complex $X = C$ or a chain map $X = (C \rightarrow D)$. If $X = C$ then the torsion coefficient t_n is the order of the torsion part of $H_n(C, Z)$.

`HomologyPb(C, n)` This is a back-up function which might work in some instances where `Homology(C, n)` fails. It is the same as `Homology(C, n)` but uses a different algorithm.

`HomologyPrimePart(C, n, p)` Inputs a chain complex C in characteristic 0, a positive integer n , and a prime p . It returns the p -part of the homology group $H_n(C, Z)$.

`LeibnizAlgebraHomology(A, n)` Inputs a Lie or Leibniz algebra $X = A$ (over the ring of integers Z or over a field K) and a positive integer n . It returns the homology groups $H_n(A, Z)$ or $H_n(A, K)$.

`LieAlgebraHomology(A, n)` Inputs a Lie algebra A (over the integers or a field) and a positive integer n . It returns the homology groups $H_n(A, Z)$ or $H_n(A, K)$.

`PrimePartDerivedFunctor(G, R, F, n)` Inputs a finite group G , a positive integer n , at least $n + 1$ terms of a ZP -resolution R of Z , and a field F . It returns the p -part of the homology group $H_n(G, F)$.

`RankHomologyPGroup(G, n)` `RankHomologyPGroup(R, n)` `RankHomologyPGroup(G, n, "empirical")` Inputs a finite group G , a positive integer n , and a field F . It returns the rank of the homology group $H_n(G, F)$.

`RankPrimeHomology(G, n)` Inputs a (smallish) p -group G together with a positive integer n . It returns a function `rankPrimeHomology(G, n)` that returns the rank of the homology group $H_n(G, Z)$.

Chapter 7

Poincare series

EfficientNormalSubgroups(G)

EfficientNormalSubgroups(G, k) Inputs a prime-power group G and, optionally, a positive integer k . The default

ExpansionOfRationalFunction(f, n) Inputs a positive integer n and a rational function $f(x) = p(x)/q(x)$ where t

PoincareSeries(G, n) PoincareSeries(R, n)

PoincareSeries(L, n)

PoincareSeries(G) Inputs a finite p -group G and a positive integer n . It returns a quotient of polynomials $f(x) =$

PoincareSeriesPrimePart(G, p, n) Inputs a finite group G , a prime p , and a positive integer n . It returns a quotient

Prank(G) Inputs a p -group G and returns the rank of the largest elementary abelian subgroup.

Chapter 8

Cohomology ring structure

`IntegralCupProduct (R, u, v, p, q)`
`IntegralCupProduct (R, u, v, p, q, P, Q, N)` (Various functions used to construct the cup product are also available)
`IntegralRingGenerators (R, n)` Inputs at least $n + 1$ terms of a ZG -resolution and integer $n > 0$. It returns a minimal resolution.
`ModPCohomologyGenerators (G, n)`
`ModPCohomologyGenerators (R)` Inputs either a p -group G and positive integer n , or else n terms of a minimal resolution.
`ModPCohomologyRing (G, n)`
`ModPCohomologyRing (G, n, level)`
`ModPCohomologyRing (R)`
`ModPCohomologyRing (R, level)` Inputs either a p -group G and positive integer n , or else n terms of a minimal resolution.
`ModPRingGenerators (A)` Inputs a mod p cohomology ring A (created using the preceding function). It returns a minimal resolution.

Chapter 9

Commutator and nonabelian tensor computations

•

BaerInvariant(G, c) Inputs a nilpotent group G and integer $c > 0$. It returns the Baer invariant $M^{(c)}(G)$ defined as

Coclass(G) Inputs a group G of prime-power order p^n and nilpotency class c say. It returns the integer $r = n - c$

EpiCentre(G, N)

EpiCentre(G) Inputs a finite group G and normal subgroup N and returns a subgroup $Z^*(G, N)$ of the centre of N .

NonabelianExteriorProduct(G, N) Inputs a finite group G and normal subgroup N . It returns a record E with the following fields:

NonabelianSymmetricKernel(G)

NonabelianSymmetricKernel(G, m) Inputs a finite or nilpotent infinite group G and returns the abelian invariant $\gamma_m(G)$

NonabelianSymmetricSquare(G)

NonabelianSymmetricSquare(G, m) Inputs a finite or nilpotent infinite group G and returns a record T with the following fields:

NonabelianTensorProduct(G, N) Inputs a finite group G and normal subgroup N . It returns a record E with the following fields:

NonabelianTensorSquare(G)

NonabelianTensorSquare(G, m) Inputs a finite or nilpotent infinite group G and returns a record T with the following fields:

RelativeSchurMultiplier(G, N) Inputs a finite group G and normal subgroup N . It returns the homology group $H_2(G/N, N)$

TensorCentre(G) Inputs a group G and returns the largest central subgroup N such that the induced homomorphism $G/N \rightarrow G/N$ is trivial

ThirdHomotopyGroupOfSuspensionB(G)

ThirdHomotopyGroupOfSuspensionB(G, m) Inputs a finite or nilpotent infinite group G and returns the abelian invariant $\gamma_m(G)$

UpperEpicentralSeries(G, c) Inputs a nilpotent group G and an integer c . It returns the c -th term of the upper central series of G

Chapter 10

Lie commutators and nonabelian Lie tensors

•
Functions on this page are joint work with Hamid Mohammadzadeh, and implemented by him.

`LieCoveringHomomorphism(L)` Inputs a finite dimensional Lie algebra L over a field, and returns a surjective Lie

`LeibnizQuasiCoveringHomomorphism(L)` Inputs a finite dimensional Lie algebra L over a field, and returns a sur

`LieEpiCentre(L)` Inputs a finite dimensional Lie algebra L over a field, and returns an ideal $Z^*(L)$ of the centre of

`LieExteriorSquare(L)` Inputs a finite dimensional Lie algebra L over a field. It returns a record E with the follow

`LieTensorSquare(L)` Inputs a finite dimensional Lie algebra L over a field and returns a record T with the follow

`LieTensorCentre(L)` Inputs a finite dimensional Lie algebra L over a field and returns the largest ideal N such th

Chapter 11

Generators and relators of groups

•
CayleyGraphDisplay(G, X)

CayleyGraphDisplay($G, X, \text{"mozilla"}$) Inputs a finite group G together with a subset X of G . It displays the co

IdentityAmongRelatorsDisplay(R, n) IdentityAmongRelatorsDisplay($R, n, \text{"mozilla"}$) Inputs a free

IsAspherical(F, R) Inputs a free group F and a set R of words in F . It performs a test on the 2-dimensional CW

PresentationOfResolution(R) Inputs at least two terms of a reduced ZG -resolution R and returns a record P w

TorsionGeneratorsAbelianGroup(G) Inputs an abelian group G and returns a generating set $[x_1, \dots, x_n]$ where

Chapter 12

Orbit polytopes and fundamental domains

•
FundamentalDomainAffineCrystGroupOnRight(v, G) Inputs a crystallographic group G (represented using Affi
OrbitPolytope(G, v, L) Inputs a permutation group or matrix group G of degree n and a rational vector v of leng
PolytopalComplex(G, v)
PolytopalComplex(G, v, n) Inputs a permutation group or matrix group G of degree n and a rational vector v of
PolytopalGenerators(G, v) Inputs a permutation group or matrix group G of degree n and a rational vector v of
VectorStabilizer(G, v) Inputs a permutation group or matrix group G of degree n and a rational vector of degr

Chapter 13

Cocycles

•
CocycleCondition(R, n) Inputs a resolution R and an integer $n > 0$. It returns an integer matrix M with the following properties:
StandardCocycle(R, f, n)
StandardCocycle(R, f, n, q) Inputs a ZG -resolution R (with contracting homotopy), a positive integer n and an integer q . It returns an integer matrix M with the following properties:
Syzygy(R, g) Inputs a ZG -resolution R (with contracting homotopy) and a list $g = [g[1], \dots, g[n]]$ of elements in G .

Chapter 14

Words in free ZG -modules

`AddFreeWords(v, w)` Inputs two words v, w in a free ZG -module and returns their sum $v + w$. If the characteristic of Z is p , the result is $v + w$ if $p \nmid 1$ and v if $p \mid 1$.

`AddFreeWordsModP(v, w, p)` Inputs two words v, w in a free ZG -module and the characteristic p of Z . It returns the sum $v + w$ if $p \nmid 1$ and v if $p \mid 1$.

`AlgebraicReduction(w)` Inputs a word w in a free ZG -module and returns a reduced version of the word in ZG .

`AlgebraicReduction(w, p)` Inputs a word w in a free ZG -module and returns a reduced version of the word in ZG modulo p .

`Multiply Word(n, w)` Inputs a word w and integer n . It returns the scalar multiple $n \cdot w$.

`Negate([i, j])` Inputs a pair $[i, j]$ of integers and returns $[-i, j]$.

`NegateWord(w)` Inputs a word w in a free ZG -module and returns the negated word $-w$.

`PrintZGword(w, elts)` Inputs a word w in a free ZG -module and a (possibly partial but sufficient) listing `elts` of the elements of G . It prints the word w in terms of the elements of G .

`TietzeReduction(S, w)` Inputs a set S of words in a free ZG -module, and a word w in the module. The function returns a reduced version of w modulo S .

Chapter 15

FpG-modules

`DirectSumOfFpGModules (M, N)`
`DirectSumOfFpGModules ([M[1], M[2], ..., M[k]])` Inputs two *FpG*-modules M and N with common group G .
`FpGModule (A, P)`
`FpGModule (A, G, p)` Inputs a p -group P and a matrix A whose rows have length a multiple of the order of G . It returns an *FpG*-module.
`FpGModuleDualBasis (M)` Inputs an *FpG*-module M . It returns a record R with two components: $R.freeModule$ is a free module and $R.dualBasis$ is a dual basis.
`FpGModuleHomomorphism (M, N, A)`
`FpGModuleHomomorphismNC (M, N, A)` Inputs *FpG*-modules M and N over a common p -group G . Also inputs a list A of matrices.
`DesuspensionFpGModule (M, n)`
`DesuspensionFpGModule (R, n)` Inputs a positive integer n and an *FpG*-module M . It returns an *FpG*-module $D^n M$.
`RadicalOfFpGModule (M)` Inputs an *FpG*-module M with G a p -group, and returns the Radical of M as an *FpG*-module.
`GeneratorsOfFpGModule (M)` Inputs an *FpG*-module M and returns a matrix whose rows correspond to a minimal generating set.
`ImageOfFpGModuleHomomorphism (f)` Inputs an *FpG*-module homomorphism $f : M \rightarrow N$ and returns its image.
`IntersectionOfFpGModules (M, N)` Inputs two *FpG*-modules M, N arising as submodules in a common free module.
`IsFpGModuleHomomorphismData (M, N, A)` Inputs *FpG*-modules M and N over a common p -group G . Also inputs a list A of matrices.
`MultipleOfFpGModule (w, M)` Inputs an *FpG*-module M and a list $w := [g_1, \dots, g_t]$ of elements in the group $G = M$.
`ProjectedFpGModule (M, k)` Inputs an *FpG*-module M of ambient dimension $n|G|$, and an integer k between 1 and n .
`RandomHomomorphismOfFpGModules (M, N)` Inputs two *FpG*-modules M and N over a common group G . It returns a random homomorphism.
`Rank (f)` Inputs an *FpG*-module homomorphism $f : M \rightarrow N$ and returns the dimension of the image of f as a vector space.
`SumOfFpGModules (M, N)` Inputs two *FpG*-modules M, N arising as submodules in a common free module $(FG)^n$.
`SumOp (f, g)` Inputs two *FpG*-module homomorphisms $f, g : M \rightarrow N$ with common source and common target. It returns their sum.
`VectorsToFpGModuleWords (M, L)` Inputs an *FpG*-module M and a list $L = [v_1, \dots, v_k]$ of vectors in M . It returns a list of words.

Chapter 16

Meataxe modules

•
DesuspensionMtxModule(M) Inputs a meataxe module M over the field of p elements and returns an FpG-module
FpG_to_MtxModule(M) Inputs an FpG-module M and returns an isomorphic meataxe module.
GeneratorsOfMtxModule(M) Inputs a meataxe module M acting on, say, the vector space V . The function returns

Chapter 17

G-Outer Groups

`GOuterGroup(E, N)`

`GOuterGroup()` **Inputs** a group E and normal subgroup N . It returns N as a G -outer group where $G = E/N$. The fun

`GOuterGroupHomomorphismNC(A, B, phi)`

`GOuterGroupHomomorphismNC()` **Inputs** G -outer groups A and B with common acting group, and a group homomor

`GOuterHomomorphismTester(A, B, phi)` **Inputs** G -outer groups A and B with common acting group, and a group ho

`Centre(A)` **Inputs** G -outer group A and returns the group theoretic centre of `ActedGroup(A)` as a G -outer group.

`DirectProduct(A, B)`

`DirectProduct(Lst)` **Inputs** G -outer groups A and B with common acting group, and returns their group-theoretic

Chapter 18

Cat-1-groups

`AutomorphismGroupAsCatOneGroup(G)` Inputs a group G and returns the Cat-1-group C corresponding to the crossed module (G, G) .

`HomotopyGroup(C, n)` Inputs a cat-1-group C and an integer n . It returns the n th homotopy group of C .

`ModuleAsCatOneGroup(G, alpha, M)` Inputs a group G , an abelian group M and a homomorphism $\alpha: G \rightarrow \text{Aut}(M)$.

`MooreComplex(C)` Inputs a cat-1-group C and returns its Moore complex $[M_1 \rightarrow M_0]$ as a list whose single entry is the Moore complex.

`NormalSubgroupAsCatOneGroup(G, N)` Inputs a group G with normal subgroup N . It returns the Cat-1-group C corresponding to the crossed module (G, N) .

Chapter 19

Coxeter diagrams and graphs of groups

`CoxeterDiagramComponents(D)` Inputs a Coxeter diagram D and returns a list $[D_1, \dots, D_d]$ of the maximal connected components of D .

`CoxeterDiagramDegree(D, v)` Inputs a Coxeter diagram D and vertex v . It returns the degree of v (i.e. the number of edges incident to v).

`CoxeterDiagramDisplay(D)` Inputs a Coxeter diagram D and displays it as a .gif file.

`CoxeterDiagramDisplay(D, "web browser")` Inputs a Coxeter diagram D and displays it as a .gif file. It uses the web browser to display the file.

`CoxeterDiagramFpArtinGroup(D)` Inputs a Coxeter diagram D and returns the corresponding finitely presented Artin group.

`CoxeterDiagramFpCoxeterGroup(D)` Inputs a Coxeter diagram D and returns the corresponding finitely presented Coxeter group.

`CoxeterDiagramIsSpherical(D)` Inputs a Coxeter diagram D and returns "true" if the associated Coxeter group is spherical.

`CoxeterDiagramMatrix(D)` Inputs a Coxeter diagram D and returns a matrix representation of it. The matrix is the Cartan matrix.

`CoxeterSubDiagram(D, V)` Inputs a Coxeter diagram D and a subset V of its vertices. It returns the full sub-diagram induced by V .

`CoxeterDiagramVertices(D)` Inputs a Coxeter diagram D and returns its set of vertices.

`EvenSubgroup(G)` Inputs a group G and returns a subgroup G^+ . The subgroup is that generated by all products xy of adjacent vertices x, y in the Coxeter diagram of G .

`GraphOfGroupsDisplay(D)` Inputs a graph of groups D and displays it as a .gif file.

`GraphOfGroupsDisplay(D, "web browser")` Inputs a graph of groups D and displays it as a .gif file. It uses the web browser to display the file.

`GraphOfGroupsTest(D)` Inputs an object D and tries to test whether it is a Graph of Groups. However, it DOES NOT work.

Chapter 20

Some functions for accessing basic data

`BoundaryMap(C)` Inputs a resolution, chain complex or cochain complex C and returns the function $C!.boundary$.

`BoundaryMatrix(C,n)` Inputs a chain or cochain complex C and integer $n>0$. It returns the n -th boundary map of C .

`Dimension(C)` Inputs a resolution, chain complex or cochain complex C and returns the function $C!.dimension$.

`Dimension(M)` Inputs a resolution, chain complex or cochain complex C and returns the function $C!.dimension$.

`EvaluateProperty(X,"name")` Inputs a component object X (such as a ZG -resolution or chain map) and a string `name`.

`GroupOfResolution(R)` Inputs a ZG -resolution R and returns the group G .

`Length(R)` Inputs a resolution R and returns its length (i.e. the number of terms of R that HAP has computed).

`Map(f)` Inputs a chain map, or cochain map or equivariant chain map f and returns the mapping function (as opposed to the chain map).

`Source(f)` Inputs a chain map, or cochain map, or equivariant chain map, or FpG -module homomorphism f and returns the source object.

`Target(f)` Inputs a chain map, or cochain map, or equivariant chain map, or FpG -module homomorphism f and returns the target object.

Chapter 21

Parallel Computation - Core Functions

```
ChildProcess()  
ChildProcess("computer.ac.wales") This starts a GAP session as a child process and returns a stream to the ch  
  
- open a shell on thishost  
- cd .ssh  
- ls  
-> if id_dsa, id_rsa etc exists, skip the next two steps!  
- ssh-keygen -t rsa  
- ssh-keygen -t dsa  
- scp *.pub user@remotehost:~/  
- ssh remotehost -l user  
- cat id_rsa.pub >> .ssh/authorized_keys  
- cat id_dsa.pub >> .ssh/authorized_keys  
- rm id_rsa.pub id_dsa.pub  
- exit
```

You should now be able to connect from "thishost" to "remotehost" without a password prompt.)

ChildClose(s) This closes the stream s to a child GAP process.

ChildCommand("cmd;", s) This runs a GAP command "cmd;" on the child process accessed by the stream s. Here

NextAvailableChild(L) Inputs a list L of child processes and returns a child in L which is ready for computation

IsAvailableChild(s) Inputs a child process s and returns true if s is currently available for computations, and fal

ChildPut(A, "B", s) This copies a GAP object A on the parent process to an object B on the child process s. (The

ChildGet("A", s) This functions copies a GAP object A on the child process s and returns it on the parent process

Chapter 22

Parallel Computation - Extra Functions

`ChildFunction("function(arg);", s)` This runs the GAP function "function(arg);" on a child process accessed by `s`.

`ChildRead(s)` This returns, as a string, the output of the last application of `ChildFunction("function(arg);", s)`.

`ChildReadEval(s)` This returns, as an evaluated string, the output of the last application of `ChildFunction("function(arg);", s)`.

`ParallelList(I, fn, L)` Inputs a list I , a function fn such that $fn(x)$ is defined for all x in I , and a list of children L .

Chapter 23

Topological Data Analysis

`MatrixToTopologicalSpace(A, n)` Inputs an integer matrix A and an integer n . It returns a 2-dimensional topological space.

`ReadImageAsTopologicalSpace("file.png", n)` `ReadImageAsTopologicalSpace("file.png", [m, n])` Reads an image file ("file.png", "file.eps", "file.bmp" etc) and returns an integer matrix of size $[m, n]$.

`ReadImageAsMatrix("file.png")` Reads an image file ("file.png", "file.eps", "file.bmp" etc) and returns an integer matrix of size $[m, n]$.

`WriteTopologicalSpaceAsImage(T, "filename", "ext")` Inputs a 2-dimensional topological space T , and a filename and extension.

`ViewTopologicalSpace(T)` `ViewTopologicalSpace(T, "mozilla")` Inputs a topological space T , and optional browser name.

`Bettinnumbers(T, n)` `Bettinnumbers(T)` Inputs a topological space T and a non-negative integer n . It returns the n -th Betti number of T .

`PathComponent(T, n)` Inputs a topological space T and an integer n in the range $0, \dots, \text{Bettinnumbers}(T, 0)$. It returns the n -th path component of T .

`SingularChainComplex(A)` Inputs a topological space T and returns a (usually very large) integral chain complex.

`ContractTopologicalSpace(T)` Inputs a topological space T of dimension d and removes d -dimensional cells from T .

`BoundaryTopologicalSpace(T)` Inputs a topological space T and returns its boundary as a topological space.

`BoundarySingularities(T)` Inputs a topological space T and returns the subspace of points in the boundary where the boundary is singular.

`ThickenedTopologicalSpace(T)` `ThickenedTopologicalSpace(T, n)` Inputs a topological space T and returns a thickened version of T .

`ComplementTopologicalSpace(T)` Inputs a topological space T and returns a topological space S . A euclidean point x is in S if and only if x is not in T .

`ConcatenatedTopologicalSpace(L)` Inputs a list L of topological spaces whose underlying arrays of numbers are all the same length.

Chapter 24

Pseudo lists

`Add(L, x)` Let L be a pseudo list of length n , and x an object compatible with the entries in L . If x is not in L then th

`Append(L, K)` Let L be a pseudo list and K a list whose objects are compatible with those in L . This operation appli

`ListToPseudoList(L)` Inputs a list L and returns the pseudo list representation of L .

Chapter 25

Miscellaneous

`BigStepLCS(G, n)` Inputs a group G and a positive integer n . It returns a subseries $G = L_1 > L_2 > \dots > L_k = 1$ of the 1

`Compose(f, g)` Inputs two FpG -module homomorphisms $f : M \rightarrow N$ and $g : L \rightarrow M$ with $Source(f) = Target(g)$.

`HAPcopyright()` This function provides details of HAP'S GNU public copyright licence.

`IsLieAlgebraHomomorphism(f)` Inputs an object f and returns true if f is a homomorphism $f : A \rightarrow B$ of Lie algebras.

`IsSuperperfect(G)` Inputs a group G and returns "true" if both the first and second integral homology of G is trivial.

`MakeHAPManual()` This function creates the manual for HAP from an XML file.

`PermToMatrixGroup(G, n)` Inputs a permutation group G and its degree n . Returns a bijective homomorphism $f : G \rightarrow GL(n, \mathbb{C})$.

`SolutionsMatDestructive(M, B)` Inputs an $m \times n$ matrix M and a $k \times n$ matrix B over a field. It returns a $k \times m$ matrix C such that $CM = B$.

`TestHap()` This runs a representative sample of HAP functions and checks to see that they produce the correct output.

Index

- Add, 26
- AddFreeWords, 16
- AddFreeWordsModP, 16
- AlgebraicReduction, 16
- Append, 26
- AutomorphismGroupAsCatOneGroup, 20

- BaerInvariant, 11
- Bettinnumbers, 25
- BigStepLCS, 27
- BoundaryMap, 22
- BoundaryMatrix, 22
- BoundarySingularities, 25
- BoundaryTopologicalSpace, 25

- CayleyGraphDisplay, 13
- Centre, 19
- ChevalleyEilenbergComplex, 7
- ChildClose, 23
- ChildCommand, 23
- ChildFunction, 24
- ChildGet, 23
- ChildProcess, 23
- ChildPut, 23
- ChildRead, 24
- ChildReadEval, 24
- Coclass, 11
- CocycleCondition, 15
- Cohomology, 8
- CohomologyPrimePart, 8
- ComplementTopologicalSpace, 25
- Compose(f,g), 27
- ConcatenatedTopologicalSpace, 25
- ContractTopologicalSpace, 25
- CoxeterDiagramComponents, 21
- CoxeterDiagramDegree, 21
- CoxeterDiagramDisplay, 21
- CoxeterDiagramFpArtinGroup, 21
- CoxeterDiagramFpCoxeterGroup, 21
- CoxeterDiagramIsSpherical, 21
- CoxeterDiagramMatrix, 21
- CoxeterDiagramVertices, 21
- CoxeterSubDiagram, 21

- DesuspensionFpGModule, 17
- DesuspensionMtxModule, 18
- Dimension, 22
- DirectProductGog, 19
- DirectSumOfFpGModules, 17

- EpiCentre, 11
- EquivariantChainMap, 5
- EvaluateProperty, 22
- EvenSubgroup, 21
- ExpansionOfRationalFunction, 9

- FpGModule, 17
- FpGModuleDualBasis, 17
- FpGModuleHomomorphism, 17
- FpG_to_MtxModule, 18
- Fundamental domains (HAPcryst), 14

- GeneratorsOfFpGModule, 17
- GeneratorsOfMtxModule, 18
- GOuterGroup, 19
- GOuterGroupHomomorphismNC, 19
- GOuterHomomorphismTester, 19
- GraphOfGroupsDisplay, 21
- GraphOfGroupsTest, 21
- GroupCohomology, 8
- GroupHomology, 8
- GroupOfResolution, 22

- HAPcopyright, 27
- Homology, 8
- HomologyPb, 8
- HomologyPrimePart, 8
- HomotopyGroup, 20
- HomToIntegers, 6

[HomToIntegersModP](#), 6
[HomToIntegralModule](#), 6

[IdentityAmongRelatorsDisplay](#), 13
[ImageOfFpGModuleHomomorphism](#), 17
[IntegralCupProduct](#), 10
[IntegralRingGenerators](#), 10
[IntersectionOfFpGModules](#), 17
[IsAspherical](#), 13
[IsAvailableChild](#), 23
[IsFpGModuleHomomorphismData](#), 17
[IsLieAlgebraHomomorphism](#), 27
[IsSuperperfect](#), 27

[LeibnizAlgebraHomology](#), 8
[LeibnizComplex](#), 7
[LeibnizQuasiCoveringHomomorphism](#), 12
[Length](#), 22
[LieAlgebraHomology](#), 8
[LieCoveringHomomorphism](#), 12
[LieEpiCentre](#), 12
[LieExteriorSquare](#), 12
[LieTensorCentre](#), 12
[LieTensorSquare](#), 12
[ListToPseudoList](#), 26
[LowerCentralSeriesLieAlgebra](#), 6

[MakeHAPManual](#), 27
[Map](#), 22
[MatrixToTopologicalSpace](#), 25
[ModPCohomologyGenerators](#), 10
[ModPCohomologyRing](#), 10
[ModPRingGenerators](#), 10
[ModuleAsCatOneGroup](#), 20
[MooreComplex](#), 20
[MultipleOfFpGModule](#), 17
[MultiplyWord](#), 16

[Negate](#), 16
[NegateWord](#), 16
[NextAvailableChild](#), 23
[NonabelianExteriorProduct](#), 11
[NonabelianSymmetricKernel](#), 11
[NonabelianSymmetricSquare](#), 11
[NonabelianTensorProduct](#), 11
[NonabelianTensorSquare](#), 11
[NormalSubgroupAsCatOneGroup](#), 20

[OrbitPolytope](#), 14

[ParallelList](#), 24
[PathComponent](#), 25
[PermToMatrixGroup](#), 27
[PoincareSeries](#), 9
[PoincareSeriesPrimePart](#), 9
[PolytopalComplex](#), 14
[PolytopalGenerators](#), 14
[Prank](#), 9
[PresentationOfResolution](#), 13
[PrimePartDerivedFunctor](#), 8
[PrintZGword](#), 16
[ProjectedFpGModule](#), 17

[RadicalOfFpGModule](#), 17
[RandomHomomorphismOfFpGModules](#), 17
[Rank](#), 17
[RankHomologyPGroup](#), 8
[RankPrimeHomology](#), 8
[ReadImageAsMatrix](#), 25
[ReadImageAsTopologicalSpace](#), 25
[RelativeSchurMultiplier](#), 11
[ResolutionAbelianGroup](#), 3
[ResolutionAlmostCrystalGroup](#), 3
[ResolutionAlmostCrystalQuotient](#), 3
[ResolutionArtinGroup](#), 3
[ResolutionAsphericalPresentation](#), 3
[ResolutionBieberbachGroup \(HAPcryst\)](#), 3
[ResolutionDirectProduct](#), 3
[ResolutionExtension](#), 3
[ResolutionFiniteDirectProduct](#), 3
[ResolutionFiniteExtension](#), 3
[ResolutionFiniteGroup](#), 3
[ResolutionFiniteSubgroup](#), 3
[ResolutionFpGModule](#), 4
[ResolutionGraphOfGroups](#), 3
[ResolutionNilpotentGroup](#), 3
[ResolutionNormalSeries](#), 3
[ResolutionPrimePowerGroup](#), 3
[ResolutionSmallFpGroup](#), 3
[ResolutionSubgroup](#), 3
[ResolutionSubnormalSeries](#), 3

[SingularChainComplex](#), 25
[SolutionsMatDestructive](#), 27
[Source](#), 22
[StandardCocycle](#), 15

SumOfFpGModules, [17](#)
SumOp, [17](#)
Syzygy, [15](#)

Target, [22](#)
TensorCentre, [11](#)
TensorWithIntegers, [6](#)
TensorWithIntegersModP, [6](#)
TensorWithRationals, [6](#)
TestHap, [27](#)
ThickenedTopologicalSpace, [25](#)
ThirdHomotopyGroupOfSuspensionB, [11](#)
TietzeReduction, [16](#)
TorsionGeneratorsAbelianGroup, [13](#)
TwistedTensorProduct, [3](#)

UpperEpicentralSeries, [11](#)

VectorStabilizer, [14](#)
VectorsToFpGModuleWords, [17](#)
ViewTopologicalSpace, [25](#)

WriteTopologicalSpaceAsImage, [25](#)