Introduction to CAP: Constructive category theory and applications

Sebastian Gutsche and Sebastian Posur

University of Siegen

August 28, 2018

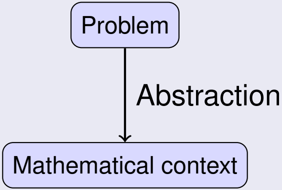
Part I

Constructive category theory

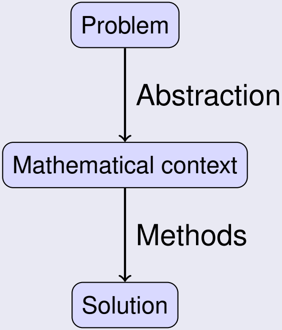
Mathematics

Problem

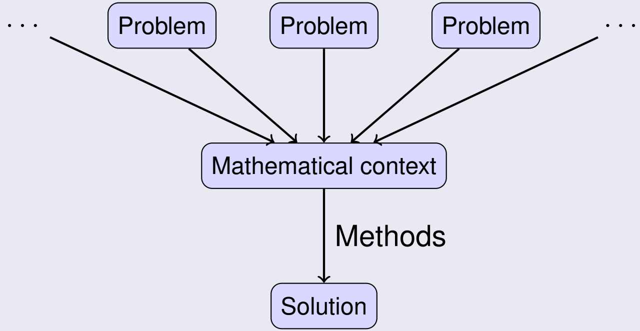
Mathematics



Mathematics



Mathematics



Mathematics

$$\begin{tikzcd}
\cdots \arrow[drrrr] & \simeq & \boxed{\text{Problem}} \arrow[drr] & \simeq & \boxed{\text{Problem}} \arrow[d] & \simeq & \boxed{\text{Problem}} \arrow[dll] & \simeq & \cdots \arrow[dllll] \\
 & & & & \boxed{\text{Mathematical context}} \arrow[d, "\text{Methods}"] & & & & \\
 & & & & \boxed{\text{Solution}} & & & &
\end{tikzcd}$$

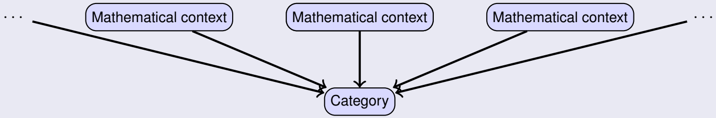
Constructive category theory

Mathematical context

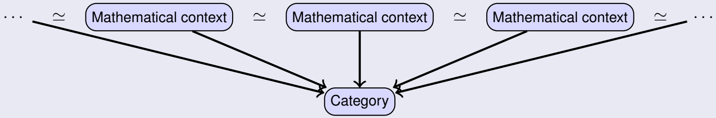
Constructive category theory

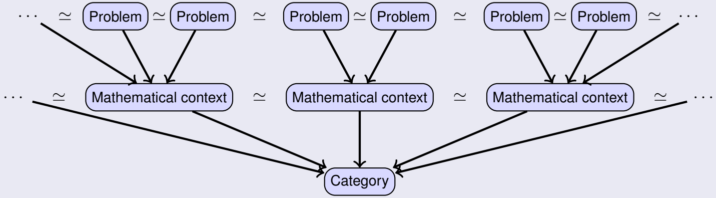
$$\begin{tikzcd} \text{Mathematical context} \arrow[d, "\text{Abstraction}"] \\ \text{Category} \end{tikzcd}$$

Constructive category theory

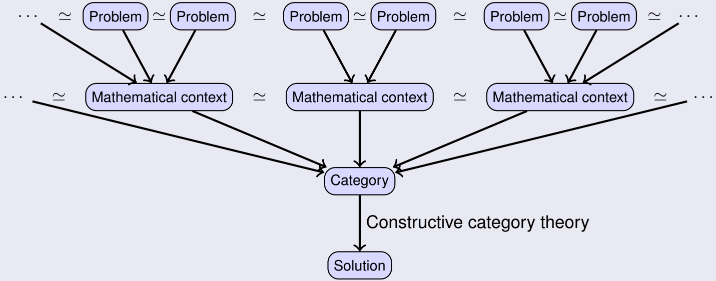


Constructive category theory




· · · ≃ Problem ≃ Problem ≃ Problem ≃ Problem ≃ Problem ≃ Problem ≃ · · ·
· · · ≃ Mathematical context ≃ Mathematical context ≃ Mathematical context ≃ · · ·
Category

Constructive category theory



Abstraction of language

Abstraction of language

Addition of two numbers:

Data type: `int`

Data type: `float`

Abstraction of language

Addition of two numbers: Assembly

Data type: `int`

Data type: `float`

Abstraction of language

Addition of two numbers: Assembly

Data type: `int`

```
addi:
movl   %edi, -4(%rsp)
movl   %esi, -8(%rsp)
movl   -4(%rsp), %esi
addl   -8(%rsp), %esi
movl   %esi, %eax
ret
```

Data type: `float`

Abstraction of language

Addition of two numbers: Assembly

Data type: `int`

```
addi:
movl  %edi, -4(%rsp)
movl  %esi, -8(%rsp)
movl  -4(%rsp), %esi
addl  -8(%rsp), %esi
movl  %esi, %eax
ret
```

Data type: `float`

```
addf:
movss %xmm0, -4(%rsp)
movss %xmm1, -8(%rsp)
movss -4(%rsp), %xmm0
addss -8(%rsp), %xmm0
ret
```

Abstraction of language

Addition of two numbers: C

Data type: `int`

Data type: `float`

Abstraction of language

Addition of two numbers: C

Data type: `int`

```
int addi( int a,
          int b )
{
  return a + b;
}
```

Data type: `float`

Abstraction of language

Addition of two numbers: C

Data type: `int`

```
int addi( int a,
          int b )
{
  return a + b;
}
```

Data type: `float`

```
float addf( float a,
            float b )
{
  return a + b;
}
```

Abstraction of language

Addition of two numbers: GAP or Julia

Data type: `int`

Data type: `float`

Abstraction of language

Addition of two numbers: GAP or Julia

Data type: `int`

```
function( a, b )
    return a + b;
end;
```

Data type: `float`

Abstraction of language

Addition of two numbers: GAP or Julia

Data type: `int`

```
function( a, b )
    return a + b;
end;
```

Data type: `float`

```
function( a, b )
    return a + b;
end;
```

Abstraction of language

Addition of two numbers: GAP or Julia

Data type: `int, float`

```
function( a, b )
    return a + b;
end;
```

Abstraction of language

Addition of two numbers: GAP or Julia

Data type: `int, float`

```
function( a, b )
    return a + b;
end;
```

High language leads to generic code!

Abstraction of language

Computing the intersection of two subobjects

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2 \rangle, \langle w_1, w_2 \rangle \leq V$:

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2 \rangle, \langle w_1, w_2 \rangle \leq V$:
Solution of

$$\begin{aligned} x_1 v_1 &+ x_2 v_2 \\ = y_1 w_1 &+ y_2 w_2 \end{aligned}$$

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2 \rangle, \langle w_1, w_2 \rangle \leq V$:
Solution of

$$\begin{aligned} x_1 v_1 &+ x_2 v_2 \\ = y_1 w_1 &+ y_2 w_2 \end{aligned}$$

Ideals of $\mathbb{Z}$

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2 \rangle, \langle w_1, w_2 \rangle \leq V$:
Solution of

$$\begin{aligned} & x_1 v_1 + x_2 v_2 \\ = & y_1 w_1 + y_2 w_2 \end{aligned}$$

Ideals of $\mathbb{Z}$

$\langle x \rangle, \langle y \rangle \leq \mathbb{Z}$:

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2\rangle, \langle w_1, w_2\rangle \leq V$:
Solution of

$$\begin{aligned} &x_1 v_1 + x_2 v_2 \\ =&y_1 w_1 + y_2 w_2 \end{aligned}$$

Ideals of $\mathbb{Z}$

$\langle x\rangle, \langle y\rangle \leq \mathbb{Z}$:
Euclidean algorithm:

$$\langle \operatorname{lcm}(x, y)\rangle$$

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2\rangle, \langle w_1, w_2\rangle \leq V$:
Solution of

$$\begin{aligned} & x_1 v_1 + x_2 v_2 \\ = & y_1 w_1 + y_2 w_2 \end{aligned}$$

Ideals of $\mathbb{Z}$

$\langle x\rangle, \langle y\rangle \leq \mathbb{Z}$:
Euclidean algorithm:

$$\langle \operatorname{lcm}(x, y)\rangle$$

Generic algorithm for both cases?

Abstraction of language

Computing the intersection of two subobjects

Vector spaces

$\langle v_1, v_2 \rangle, \langle w_1, w_2 \rangle \leq V$:
Solution of

$$x_1 v_1 + x_2 v_2$$
$$= y_1 w_1 + y_2 w_2$$

Ideals of $\mathbb{Z}$

$\langle x \rangle, \langle y \rangle \leq \mathbb{Z}$:
Euclidean algorithm:

$$\langle \mathrm{lcm}\,(x, y) \rangle$$

Generic algorithm for both cases? **Category theory!**

Category theory as programming language

Category theory

Category theory as programming language

Category theory

- abstracts mathematical structures

Category theory as programming language

Category theory

- abstracts mathematical structures
- defines a *language* to formulate theorems and algorithms for different structures *at the same time*

Category theory as programming language

Category theory

- abstracts mathematical structures
- defines a *language* to formulate theorems and algorithms for different structures *at the same time*

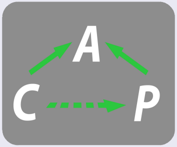
CAP - Categories, Algorithms, Programming

Category theory as programming language

Category theory

- abstracts mathematical structures
- defines a *language* to formulate theorems and algorithms for different structures *at the same time*

CAP - Categories, Algorithms, Programming



CAP implements a
categorical programming language

Categories

Definition

A category $\mathcal{A}$ contains the following data:

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$

$A \qquad\qquad B \qquad\qquad C$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$

$$A \qquad\qquad B \qquad\qquad C$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$

$$A \longrightarrow B \qquad\qquad C$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$

$$A \longrightarrow B \longrightarrow C$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$
- $\circ : \mathrm{Hom}_{\mathcal{A}}(B, C) \times \mathrm{Hom}_{\mathcal{A}}(A, B) \to \mathrm{Hom}_{\mathcal{A}}(A, C)$ (assoc.)

$$A \longrightarrow B \longrightarrow C$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$
- $\circ : \mathrm{Hom}_{\mathcal{A}}(B, C) \times \mathrm{Hom}_{\mathcal{A}}(A, B) \to \mathrm{Hom}_{\mathcal{A}}(A, C)$ (assoc.)

$$\begin{tikzcd} A \arrow[r] \arrow[rr, bend left] & B \arrow[r] & C \end{tikzcd}$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$
- $\circ : \mathrm{Hom}_{\mathcal{A}}(B, C) \times \mathrm{Hom}_{\mathcal{A}}(A, B) \to \mathrm{Hom}_{\mathcal{A}}(A, C)$ (assoc.)
- Neutral elements: $\mathrm{id}_A \in \mathrm{Hom}_{\mathcal{A}}(A, A)$

$$\begin{tikzcd} A \arrow[r] \arrow[rr, bend left] & B \arrow[r] & C \end{tikzcd}$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$
- $\circ : \mathrm{Hom}_{\mathcal{A}}(B, C) \times \mathrm{Hom}_{\mathcal{A}}(A, B) \to \mathrm{Hom}_{\mathcal{A}}(A, C)$ (assoc.)
- Neutral elements: $\mathrm{id}_A \in \mathrm{Hom}_{\mathcal{A}}(A, A)$

$$\begin{tikzcd}
A \arrow[r] \arrow[rr, bend left] \arrow[loop below] & B \arrow[r] \arrow[loop below] & C \arrow[loop below]
\end{tikzcd}$$

Categories

Definition

A category $\mathcal{A}$ contains the following data:

- $\mathrm{Obj}_{\mathcal{A}}$
- $\mathrm{Hom}_{\mathcal{A}}(A, B)$
- $\circ : \mathrm{Hom}_{\mathcal{A}}(B, C) \times \mathrm{Hom}_{\mathcal{A}}(A, B) \to \mathrm{Hom}_{\mathcal{A}}(A, C)$ (assoc.)
- Neutral elements: $\mathrm{id}_A \in \mathrm{Hom}_{\mathcal{A}}(A, A)$

$$
\begin{tikzcd}
A \arrow[r] \arrow[rr, bend left] \arrow[loop below] & B \arrow[r] \arrow[loop below] & C \arrow[loop below]
\end{tikzcd}
$$

Finite dimensional vector spaces

Let k be a field.

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- $\mathrm{Obj} :=$ finite dimensional k-vector spaces

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- $\mathrm{Obj} :=$ finite dimensional k-vector spaces
- $\mathrm{Hom}(V, W) :=$ k-linear maps $V \to W$

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- $\mathrm{Obj} :=$ finite dimensional k-vector spaces
- $\mathrm{Hom}(V, W) :=$ k-linear maps $V \to W$

Example: matrices

- $\mathrm{Obj} := \mathbb{N}_0$

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- $\mathrm{Obj} :=$ finite dimensional k-vector spaces
- $\mathrm{Hom}(V, W) :=$ k-linear maps $V \to W$

Example: matrices

- $\mathrm{Obj} := \mathbb{N}_0$
- $\mathrm{Hom}(n, m) := k^{n \times m}$

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- $\mathrm{Obj} :=$ finite dimensional k-vector spaces
- $\mathrm{Hom}(V, W) :=$ k-linear maps $V \to W$

$$\simeq$$

Example: matrices

- $\mathrm{Obj} := \mathbb{N}_0$
- $\mathrm{Hom}(n, m) := k^{n \times m}$

Finite dimensional vector spaces

Let k be a field.

Example: k-vec

- Obj := finite dimensional k-vector spaces
- $\mathrm{Hom}(V, W) :=$ k-linear maps $V \to W$

$$\simeq$$

Example: matrices (computerfriendly model)

- $\mathrm{Obj} := \mathbb{N}_0$.
- $\mathrm{Hom}(n, m) := k^{n \times m}$.

Computable categories

Computable categories

A category becomes computable through

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

1 2 1

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

1 2 1

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

$$1 \xrightarrow[\begin{pmatrix} 1 & 2 \end{pmatrix}]{} 2 \xrightarrow[\begin{pmatrix} 3 \\ 4 \end{pmatrix}]{} 1$$

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

$$1 \xrightarrow[\begin{pmatrix} 1 & 2 \end{pmatrix}]{} 2 \xrightarrow[\begin{pmatrix} 3 \\ 4 \end{pmatrix}]{} 1$$

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

$$\begin{pmatrix} 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = (11)$$

$$\begin{tikzcd}
1 \arrow[r, "\begin{pmatrix} 1 & 2 \end{pmatrix}"'] \arrow[rr, bend left] & 2 \arrow[r, "\begin{pmatrix} 3 \\ 4 \end{pmatrix}"'] & 1
\end{tikzcd}$$

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

$$\begin{pmatrix} 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = (11)$$

$$\begin{tikzcd} 1 \arrow[r, "\begin{pmatrix} 1 & 2 \end{pmatrix}"'] \arrow[rr, bend left] & 2 \arrow[r, "\begin{pmatrix} 3 \\ 4 \end{pmatrix}"'] & 1 \end{tikzcd}$$

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}\text{-vec}$

$$\begin{pmatrix} 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = (11)$$

$$\begin{tikzcd}
1 \arrow[r, "\begin{pmatrix} 1 & 2 \end{pmatrix}"'] \arrow[rr, bend left] \arrow[loop below, "(1)"] & 2 \arrow[r, "\begin{pmatrix} 3 \\ 4 \end{pmatrix}"'] \arrow[loop below, "\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}"] & 1 \arrow[loop below, "(1)"]
\end{tikzcd}$$

Computable categories

A category becomes computable through

- Data structures for *objects* and *morphisms*
- Algorithms to compute the *composition* of morphisms and *identity morphisms* of objects

Example: $\mathbb{Q}$-vec

$$\begin{pmatrix} 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 3 \\ 4 \end{pmatrix} = (11)$$

$$\begin{tikzcd}
1 \arrow[r, "\begin{pmatrix} 1 & 2 \end{pmatrix}"'] \arrow[rr, bend left] \arrow[loop below, "(1)"] & 2 \arrow[r, "\begin{pmatrix} 3 \\ 4 \end{pmatrix}"'] \arrow[loop below, "\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}"] & 1 \arrow[loop below, "(1)"]
\end{tikzcd}$$

Equivalences

Example

$$\mathrm{Rep}_k(G) \xrightarrow{\quad\quad\quad\sim\quad\quad\quad} \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec}$$

Equivalences

Example

$$\mathrm{Rep}_k(G) \xrightarrow{\quad\quad\quad\sim\quad\quad\quad} \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec}$$

$$V$$

Equivalences

Example

$$\mathrm{Rep}_k(G) \xrightarrow{\quad\quad\quad\sim\quad\quad\quad} \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec}$$

$$V \simeq \bigoplus_{i \in \mathrm{Irr}(G)} a_i V^i$$

Equivalences

Example

$$\mathrm{Rep}_k(G) \xrightarrow{\quad\quad\quad\sim\quad\quad\quad} \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec}$$

$$V \simeq \bigoplus_{i \in \mathrm{Irr}(G)} a_i V^i$$

Equivalences

Example

$$\mathrm{Rep}_k(G) \xrightarrow{\quad\quad\quad\quad\sim\quad\quad\quad\quad} \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec}$$

$$V \simeq \bigoplus_{i \in \mathrm{Irr}(G)} a_i V^i \qquad \longmapsto \qquad (a_i)_i$$

Equivalences

Example

$$\begin{array}{ccc} \mathrm{Rep}_k(G) & \xrightarrow{\quad\sim\quad} & \bigoplus_{i \in \mathrm{Irr}(G)} k\text{-vec} \\ V \simeq \bigoplus_{i \in \mathrm{Irr}(G)} a_i V^i & \longmapsto & (a_i)_i \end{array}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[rr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rr, "{\mathrm{Diag}(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2)}"'] & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}
V \arrow[rrr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & & V \arrow[d, "\wr"'] \\
{\color{green}V^{\mathbb{1}}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rrr, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & & & {\color{green}V^{\mathbb{1}}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}
V \arrow[rr, "\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}"] \arrow[d, "\wr"'] & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rr, "\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)"'] & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}
V \arrow[rr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rr, "{\mathrm{Diag}(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2)}"'] & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[r, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}[column sep=huge]
V \arrow[r, "\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "\mathrm{Diag}(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2)"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}
V \arrow[rrr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rrr, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[r, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "{\mathrm{Diag}(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2)}"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}[column sep=huge]
V \arrow[r, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "{\mathrm{Diag}(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2)}"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}, V^{\mathrm{sgn}}, V^{\chi}$

$$\begin{tikzcd}
V \arrow[r, "\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[rrr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rrr, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

$$\mathrm{Rep}_{\mathbb{Q}}(S_3) \simeq \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec}$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}[column sep=huge]
V \arrow[r, "\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

$$\mathrm{Rep}_{\mathbb{Q}}(S_3) \simeq \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec}$$

$$\oplus \longleftrightarrow \oplus$$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[r, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

$$\mathrm{Rep}_{\mathbb{Q}}(S_3) \simeq \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec}$$

$\oplus \longleftrightarrow \oplus$ $\qquad$ ker $\longleftrightarrow$ ker

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$\begin{tikzcd}
V \arrow[rrr, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & & & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[rrr, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & & & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}$$

$$\mathrm{Rep}_{\mathbb{Q}}(S_3) \simeq \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec}$$

$\oplus \longleftrightarrow \oplus$ $\qquad$ $\ker \longleftrightarrow \ker$ $\qquad$ $\simeq \longleftrightarrow \simeq$

Equivalences: $\mathrm{Rep}_k(G)$

Example: S_3, irreducible representations: $V^{\mathbb{1}}$, V^{sgn}, V^{χ}

$$
\begin{tikzcd}[column sep=huge, row sep=large]
V \arrow[r, "{\begin{pmatrix} -102 & 5824 & -96 & 20 & 1444 & 584 \\ 58 & -2366 & 60 & 8 & -590 & -240 \\ 83 & -5366 & 75 & -28 & -1328 & -536 \\ -25 & 1354 & -24 & 3 & 336 & 136 \\ -377 & 17200 & -384 & -28 & 4279 & 1736 \\ 351 & -18877 & 348 & -12 & -4682 & -1893 \end{pmatrix}}"] \arrow[d, "\wr"'] & V \arrow[d, "\wr"'] \\
V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi} \arrow[r, "{\mathrm{Diag}\left(-1, 3, \begin{pmatrix} -2 & 4 \\ 1 & -1 \end{pmatrix} \otimes I_2\right)}"'] & V^{\mathbb{1}} \oplus V^{\mathrm{sgn}} \oplus V^{\chi} \oplus V^{\chi}
\end{tikzcd}
$$

$$\mathrm{Rep}_{\mathbb{Q}}(S_3) \simeq \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec} \oplus \mathbb{Q}\text{-vec}$$

$\oplus \longleftrightarrow \oplus$ $\qquad$ $\ker \longleftrightarrow \ker$ $\qquad$ $\simeq \longleftrightarrow =$

The language of category theory

Some categorical operations in abelian categories

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \rightarrow \mathrm{Obj}$

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \to \mathrm{Obj}$
- $\circ : \mathrm{Hom}(B, C) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \to \mathrm{Obj}$
- $\circ : \mathrm{Hom}(B, C) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$
- $+, - : \mathrm{Hom}(A, B) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, B)$

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \to \mathrm{Obj}$
- $\circ : \mathrm{Hom}(B, C) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$
- $+, - : \mathrm{Hom}(A, B) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, B)$
- $\ker : \mathrm{Hom}(A, B) \to \mathrm{Obj}$

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \to \mathrm{Obj}$
- $\circ : \mathrm{Hom}(B, C) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$
- $+, - : \mathrm{Hom}(A, B) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, B)$
- $\ker : \mathrm{Hom}(A, B) \to \mathrm{Obj}$
- ...

The language of category theory

Some categorical operations in abelian categories

- $\oplus : \mathrm{Obj} \times \mathrm{Obj} \to \mathrm{Obj}$
- $\circ : \mathrm{Hom}(B, C) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, C)$
- $+, - : \mathrm{Hom}(A, B) \times \mathrm{Hom}(A, B) \to \mathrm{Hom}(A, B)$
- $\ker : \mathrm{Hom}(A, B) \to \mathrm{Obj}$
- ...

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$.

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$.

$$A \xrightarrow{\varphi} B$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

$$A \xrightarrow{\varphi} B$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

... one needs an object $\ker \varphi$,

$$\begin{tikzcd} \ker \varphi & & \\ & A \arrow[r, "\varphi"] & B \end{tikzcd}$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

... one needs an object $\ker\varphi$,
its embedding $\kappa = \mathrm{KernelEmbedding}(\varphi)$,

$$\begin{tikzcd}
\ker\varphi \arrow[dr, hook, "\kappa"] & & \\
& A \arrow[r, "\varphi"] & B
\end{tikzcd}$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

... one needs an object $\ker\varphi$,
its embedding $\kappa = \mathrm{KernelEmbedding}(\varphi)$,
and for every test morphism τ

$$
\begin{tikzcd}
\ker\varphi \arrow[dr, hook, "\kappa"] \arrow[drr, bend left, "0"] & & \\
& A \arrow[r, "\varphi"] & B \\
T \arrow[ur, "\tau"] \arrow[urr, bend right, "0"'] & &
\end{tikzcd}
$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

... one needs an object $\ker \varphi$,
its embedding $\kappa = \mathrm{KernelEmbedding}(\varphi)$,
and for every test morphism τ
a *unique* morphism $\lambda = \mathrm{KernelLift}(\varphi, \tau)$

$$\begin{tikzcd}
\ker \varphi \arrow[dr, hook, "\kappa"] \arrow[drr, bend left, "0"] & & \\
& A \arrow[r, "\varphi"] & B \\
T \arrow[uu, dashed, "\lambda"] \arrow[ur, "\tau"] \arrow[urr, bend right, "0"'] & &
\end{tikzcd}$$

Implementation of the kernel

Let $\varphi \in \mathrm{Hom}(A, B)$. To fully describe the kernel of φ ...

... one needs an object $\ker \varphi$,
its embedding $\kappa = \mathrm{KernelEmbedding}(\varphi)$,
and for every test morphism τ
a *unique* morphism $\lambda = \mathrm{KernelLift}(\varphi, \tau)$, such that

$$
\begin{tikzcd}
\ker\varphi \arrow[dr, hook, "\kappa"] \arrow[drr, bend left, "0"] & & \\
\arrow[r, phantom, "\circlearrowleft"] & A \arrow[r, "\varphi"] & B \\
T \arrow[uu, dashed, "\lambda"] \arrow[ur, "\tau"] \arrow[urr, bend right, "0"'] & &
\end{tikzcd}
$$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m, n) := \mathbb{Q}^{m \times n}$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m, n) := \mathbb{Q}^{m \times n}$

$$A \xrightarrow{\varphi} B$$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}(m, n) := \mathbb{Q}^{m \times n}$

$\ker \varphi$

$$A \xrightarrow{\varphi} B$$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m, n) := \mathbb{Q}^{m \times n}$

$\ker \varphi$

$$A \xrightarrow{\varphi} B$$

Compute

- $\ker \varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m,n) := \mathbb{Q}^{m \times n}$

$$\begin{tikzcd} \ker\varphi \arrow[dr, hook, "\kappa"] & & \\ & A \arrow[r, "\varphi"] & B \end{tikzcd}$$

Compute

- $\ker\varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, Hom $(m, n) := \mathbb{Q}^{m \times n}$

$$\begin{tikzcd} \ker\varphi \arrow[dr, hook, "\kappa"] & & \\ & A \arrow[r, "\varphi"] & B \end{tikzcd}$$

Compute

- $\ker\varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$
- κ by solving $X \cdot \varphi = 0$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m, n) := \mathbb{Q}^{m \times n}$

$$
\begin{tikzcd}
\ker\varphi \arrow[dr, hook, "\kappa"] & & \\
& A \arrow[r, "\varphi"] & B \\
T \arrow[ur, "\tau"] & &
\end{tikzcd}
$$

Compute

- $\ker\varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$
- κ by solving $X \cdot \varphi = 0$

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, $\mathrm{Hom}\,(m, n) := \mathbb{Q}^{m \times n}$

$$
\begin{tikzcd}
\ker\varphi \arrow[dr, hook, "\kappa"] & & \\
& A \arrow[r, "\varphi"] & B \\
T \arrow[uu, dashed, "\lambda"] \arrow[ur, "\tau"'] & &
\end{tikzcd}
$$

Compute

- $\ker\varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$
- κ by solving $X \cdot \varphi = 0$

Implementation of the kernel: $\mathbb{Q}$-vec

$\mathrm{Obj} := \mathbb{Z}_{\geq 0}$, Hom $(m, n) := \mathbb{Q}^{m \times n}$

$$\begin{tikzcd}
\color{red}{\ker \varphi} \arrow[dr, hook, red, "\kappa"] & & \\
& A \arrow[r, "\varphi"] & B \\
T \arrow[uu, dashed, red, "\lambda"] \arrow[ur, "\tau"] & &
\end{tikzcd}$$

Compute

- $\ker \varphi$ as $\dim(A) - \mathrm{rank}(\varphi)$
- κ by solving $X \cdot \varphi = 0$
- λ by solving $X \cdot \kappa = \tau$

The language of category theory

The language of category theory

Given a diagram of abelian groups:

The language of category theory

Given a diagram of abelian groups:

$$\begin{tikzcd}
{\color{blue}\ker} \arrow[r, hook] & A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
{\color{green}\ker} \arrow[r, hook] & A \arrow[r, green] & B
\end{tikzcd}$$

The language of category theory

Given a diagram of abelian groups:

$$\begin{tikzcd}
\color{blue}{\ker} \arrow[r, hook] \arrow[d, dashed] & A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\color{green}{\ker} \arrow[r, hook] & A \arrow[r, green] & B
\end{tikzcd}$$

The language of category theory

Given a diagram of abelian groups:

$$
\begin{tikzcd}
x \in \ker \arrow[r, hook] \arrow[d, dashed] & A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\ker \arrow[r, hook] & A \arrow[r] & B
\end{tikzcd}
$$

The language of category theory

Given a diagram of abelian groups:

$$\begin{tikzcd}
x \in \textcolor{blue}{\ker} \arrow[r, hook] \arrow[d, dashed] & x \in A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\textcolor{green}{\ker} \arrow[r, hook] & A \arrow[r, green] & B
\end{tikzcd}$$

The language of category theory

Given a diagram of abelian groups:

$$\begin{tikzcd}
x \in \color{blue}{\ker} \arrow[r, hook] \arrow[d, dashed] & x \in A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\color{green}{\ker} \arrow[r, hook] & \alpha(x) \in A \arrow[r, green] & B
\end{tikzcd}$$

The language of category theory

Given a diagram of abelian groups:

$$
\begin{tikzcd}
x \in {\color{blue}\mathrm{ker}} \arrow[r, hook] \arrow[d, dashed] & x \in A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
{\color{green}\mathrm{ker}} \arrow[r, hook] & \alpha(x) \in A \arrow[r, green] & 0 \in B
\end{tikzcd}
$$

The language of category theory

Given a diagram of abelian groups:

$$\begin{tikzcd}
x \in \mathrm{ker} \arrow[r, hook] \arrow[d, dashed] & x \in A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\alpha(x) \in \mathrm{ker} \arrow[r, hook] & \alpha(x) \in A \arrow[r] & 0 \in B
\end{tikzcd}$$

Given a diagram of abelian groups:

$$\begin{tikzcd}
x \in \mathrm{ker} \arrow[r, hook] \arrow[d, dashed] & x \in A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\alpha(x) \in \mathrm{ker} \arrow[r, hook] & \alpha(x) \in A \arrow[r] & 0 \in B
\end{tikzcd}$$

The language of category theory

The same example in the language of category theory:

$$\begin{tikzcd}
\mathrm{ker} \arrow[r, "\kappa'"] \arrow[d, dashed] & A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\mathrm{ker} \arrow[r] & A \arrow[r, "\varphi"] & B
\end{tikzcd}$$

The language of category theory

The same example in the language of category theory:

$$\begin{tikzcd}
\textcolor{blue}{\mathrm{ker}} \arrow[r, "\kappa'"] \arrow[d, dashed] & A' \arrow[r, blue] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\textcolor{green}{\mathrm{ker}} \arrow[r] & A \arrow[r, green, "\textcolor{green}{\varphi}"] & B
\end{tikzcd}$$

The same example in the language of category theory:

$$
\begin{tikzcd}
\mathrm{ker} \arrow[r, "\kappa'"] \arrow[d, dashed] & A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\mathrm{ker} \arrow[r] & A \arrow[r, "\varphi"] & B
\end{tikzcd}
$$

$$\dashrightarrow \; =$$

The language of category theory

The same example in the language of category theory:

$$
\begin{tikzcd}
\mathrm{ker} \arrow[r, "\kappa'"] \arrow[d, dashed] \arrow[dr] & A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\mathrm{ker} \arrow[r] & A \arrow[r, "\varphi"] & B
\end{tikzcd}
$$

$$\downarrow \; = \qquad\qquad \alpha \circ \kappa'$$

The same example in the language of category theory:

$$\begin{tikzcd}
\mathrm{ker} \arrow[r, "\kappa'"] \arrow[d, dashed] \arrow[dr] & A' \arrow[r] \arrow[d, "\alpha"'] & B' \arrow[d] \\
\mathrm{ker} \arrow[r] & A \arrow[r, "\varphi"] & B
\end{tikzcd}$$

$$\dashrightarrow = \mathrm{KernelLift}(\varphi, \alpha \circ \kappa')$$

Features of CAP

CAP - Categories, Algorithms, Programming

Features of CAP

CAP - Categories, Algorithms, Programming

CAP is a framework to implement computable categories and provides

Features of CAP

CAP - Categories, Algorithms, Programming

CAP is a framework to implement computable categories and provides

- specifications of categorical operations,

Features of CAP

CAP - Categories, Algorithms, Programming

CAP is a framework to implement computable categories and provides

- specifications of categorical operations,
- generic algorithms based on basic categorical operations,

Features of CAP

CAP - Categories, Algorithms, Programming

CAP is a framework to implement computable categories and provides

- specifications of categorical operations,
- generic algorithms based on basic categorical operations,
- a categorical programming language having categorical operations as syntax elements

Computing the intersection

Let $M_1 \subseteq N$ and $M_2 \subseteq N$ subobjects in an abelian category.

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
M_1 \arrow[dr, hook, "\iota_1"] & \\
& N \\
M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 & & & N \\
 & & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 \arrow[ur, "\pi_1"] & & N \\
 & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[dr, "\pi_2"'] & & N \\
 & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category. Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[dr, "\pi_2"'] & & N \\
 & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category. Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[rr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] \arrow[dr, "\pi_2"'] & & N \\
 & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category. Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \oplus M_2 \arrow[ur, two heads, "\pi_1"] \arrow[dr, two heads, "\pi_2"'] \arrow[rr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] & & N \\
 & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$
- $\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \cap M_2 \arrow[r, hook, "\kappa"] & M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[dr, "\pi_2"'] \arrow[rr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] & & N \\
 & & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$
- $\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category.
Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \cap M_2 \arrow[r, hook, "\kappa"] & M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[dr, "\pi_2"'] \arrow[rr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] & & N \\
 & & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$
- $\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$
- $\kappa := \mathrm{KernelEmbedding}(\varphi)$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category. Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$\begin{tikzcd}
 & & & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \cap M_2 \arrow[r, hook, "\kappa"] & M_1 \oplus M_2 \arrow[urr, "\pi_1"] \arrow[rrr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] \arrow[drr, "\pi_2"'] & & & N \\
 & & & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$
- $\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$
- $\kappa := \mathrm{KernelEmbedding}(\varphi)$

Computing the intersection

Let $M_1 \hookrightarrow N$ and $M_2 \hookrightarrow N$ subobjects in an abelian category. Compute their intersection $\gamma : M_1 \cap M_2 \hookrightarrow N$.

$$
\begin{tikzcd}
 & & M_1 \arrow[dr, hook, "\iota_1"] & \\
M_1 \cap M_2 \arrow[r, hook, "\kappa"] & M_1 \oplus M_2 \arrow[ur, "\pi_1"] \arrow[rr, "\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2"] \arrow[dr, "\pi_2"'] & & N \\
 & & M_2 \arrow[ur, hook, "\iota_2"'] &
\end{tikzcd}
$$

- $\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left((M_1, M_2), i\right)$, $i = 1, 2$
- $\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$
- $\kappa := \mathrm{KernelEmbedding}(\varphi)$
- $\gamma := \iota_1 \circ \pi_1 \circ \kappa$

Translation to CAP

$\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left(\left(M_1, M_2\right), i\right)$, $i = 1, 2$

$\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

$\kappa := \mathrm{KernelEmbedding}\left(\varphi\right)$

$\gamma := \iota_1 \circ \pi_1 \circ \kappa$

Translation to CAP

$\pi_i := \text{ProjectionInFactorOfDirectSum}\left(\left(M_1, M_2\right), i\right),\ i = 1, 2$

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
```

$\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

$\kappa := \text{KernelEmbedding}\left(\varphi\right)$

$\gamma := \iota_1 \circ \pi_1 \circ \kappa$

Translation to CAP

$\pi_i := \mathrm{ProjectionInFactorOfDirectSum}\left(\left(M_1, M_2\right), i\right),\ i = 1, 2$

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
```

$\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

```
lambda := PostCompose( iota1, pi1 );
phi := lambda - PostCompose( iota2, pi2 );
```

$\kappa := \mathrm{KernelEmbedding}\left(\varphi\right)$

$\gamma := \iota_1 \circ \pi_1 \circ \kappa$

Translation to CAP

$\pi_i := \text{ProjectionInFactorOfDirectSum}\left(\left(M_1, M_2 \right), i \right)$, $i = 1, 2$

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
```

$\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

```
lambda := PostCompose( iota1, pi1 );
phi := lambda - PostCompose( iota2, pi2 );
```

$\kappa := \text{KernelEmbedding}\left(\varphi \right)$

```
kappa := KernelEmbedding( phi );
```

$\gamma := \iota_1 \circ \pi_1 \circ \kappa$

Translation to CAP

$\pi_i :=$ ProjectionInFactorOfDirectSum$((M_1, M_2), i)$, $i = 1, 2$

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
```

$\varphi := \iota_1 \circ \pi_1 - \iota_2 \circ \pi_2$

```
lambda := PostCompose( iota1, pi1 );
phi := lambda - PostCompose( iota2, pi2 );
```

$\kappa :=$ KernelEmbedding(φ)

```
kappa := KernelEmbedding( phi );
```

$\gamma := \iota_1 \circ \pi_1 \circ \kappa$

```
gamma := PostCompose( lambda, kappa );
```

Translation to CAP

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );

lambda := PostCompose( iota1, pi1 );
phi := lambda - PostCompose( iota2, pi2 );

kappa := KernelEmbedding( phi );

gamma := PostCompose( lambda, kappa );
```

Translation to CAP

```
pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );

lambda := PostCompose( iota1, pi1 );
phi := lambda - PostCompose( iota2, pi2 );

kappa := KernelEmbedding( phi );

gamma := PostCompose( lambda, kappa );
```

Translation to CAP

```
IntersectionOfSubobject := function( iota1, iota2 )

  pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
  pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
  lambda := PostCompose( iota1, pi1 );
  phi := lambda - PostCompose( iota2, pi2 );
  kappa := KernelEmbedding( phi );
  gamma := PostCompose( lambda, kappa );
```

Translation to CAP

```
IntersectionOfSubobject := function( iota1, iota2 )

  M1 := Source( iota1 );
  M2 := Source( iota2 );

  pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
  pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );

  lambda := PostCompose( iota1, pi1 );
  phi := lambda - PostCompose( iota2, pi2 );

  kappa := KernelEmbedding( phi );

  gamma := PostCompose( lambda, kappa );
```

Translation to CAP

```
IntersectionOfSubobject := function( iota1, iota2 )

  M1 := Source( iota1 );
  M2 := Source( iota2 );
  pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
  pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
  lambda := PostCompose( iota1, pi1 );
  phi := lambda - PostCompose( iota2, pi2 );
  kappa := KernelEmbedding( phi );
  gamma := PostCompose( lambda, kappa );
  return gamma;
end;
```

Translation to CAP

```
IntersectionOfSubobject := function( iota1, iota2 )
  local M1, M2, pi1, pi2, lambda, phi, kappa, gamma;
  M1 := Source( iota1 );
  M2 := Source( iota2 );
  pi1 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 1 );
  pi2 := ProjectionInFactorOfDirectSum( [ M1, M2 ], 2 );
  lambda := PostCompose( iota1, pi1 );
  phi := lambda - PostCompose( iota2, pi2 );
  kappa := KernelEmbedding( phi );
  gamma := PostCompose( lambda, kappa );
  return gamma;
end;
```

Computing the intersection: $\mathbb{Q}$-vec

Compute the intersection of

$$
\begin{tikzcd}
M_1 \arrow[rr, hook, "\iota_1 := \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}"] \arrow[d, phantom, "\cong"] & & N \arrow[d, phantom, "\cong"] & & M_2 \arrow[ll, hook', "\iota_2 := \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}"'] \arrow[d, phantom, "\cong"] \\
2 & & 3 & & 2
\end{tikzcd}
$$

Computing the intersection: $\mathbb{Q}$-vec

Compute the intersection of

$$
\begin{tikzcd}[column sep=huge]
M_1 \arrow[r, hook, "\iota_1 := \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}"] \arrow[d, phantom, sloped, "\cong"] & N \arrow[d, phantom, sloped, "\cong"] & M_2 \arrow[l, hook', "\iota_2 := \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}"'] \arrow[d, phantom, sloped, "\cong"] \\
2 & 3 & 2
\end{tikzcd}
$$

```
gap> gamma := IntersectionOfSubobject( iota1, iota2 );
<A morphism in the category of matrices over Q>
```

Computing the intersection: $\mathbb{Q}$-vec

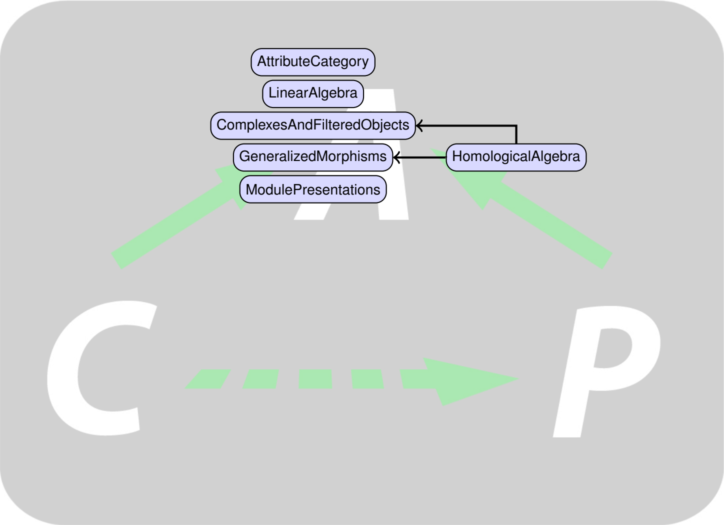
Compute the intersection of

$$\begin{tikzcd} M_1 \arrow[r, hook, "\iota_1 := \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}"] \arrow[d, phantom, "\sqcap"] & N \arrow[d, phantom, "\sqcap"] & M_2 \arrow[l, hook', "\iota_2 := \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}"'] \arrow[d, phantom, "\sqcap"] \\ 2 & 3 & 2 \end{tikzcd}$$

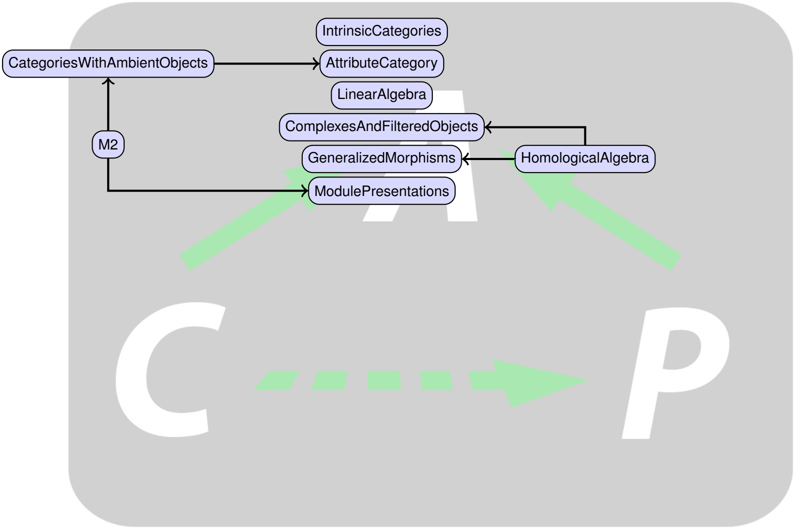
```
gap> gamma := IntersectionOfSubobject( iota1, iota2 );
<A morphism in the category of matrices over Q>
gap> Display( gamma );
[ [  1,  1,  0 ] ]

A morphism in the category of matrices over Q
```

CAP packages

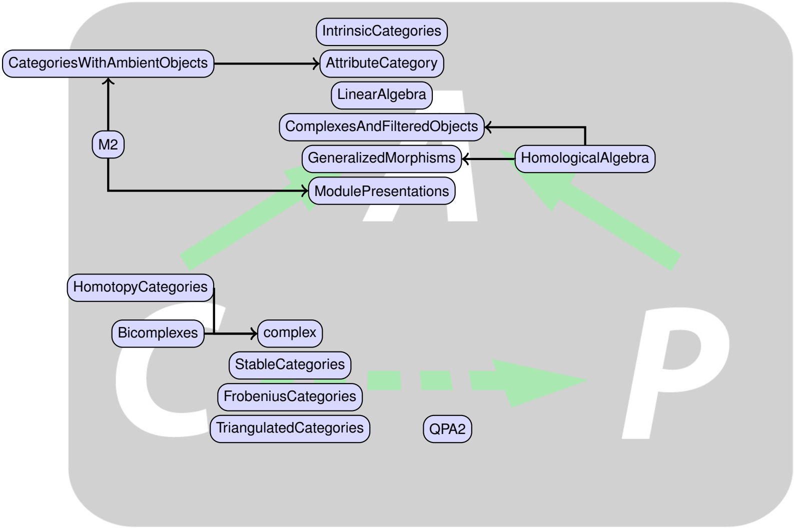
CAP packages



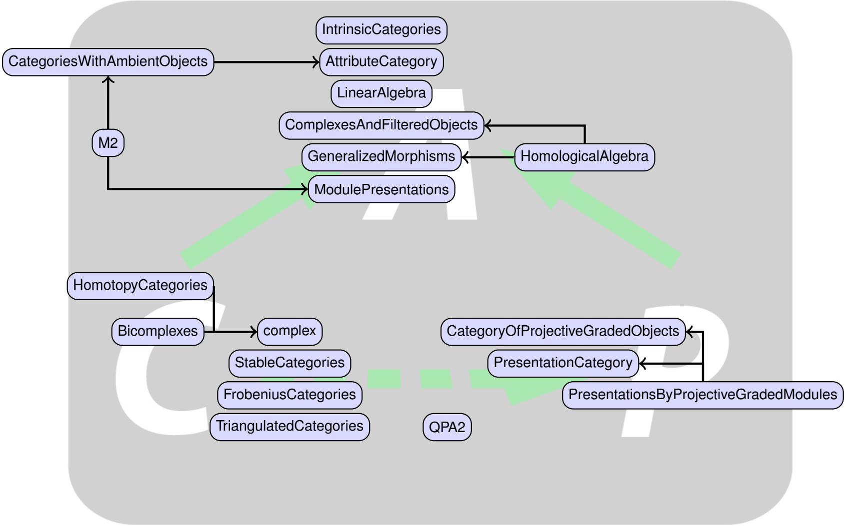
CAP packages



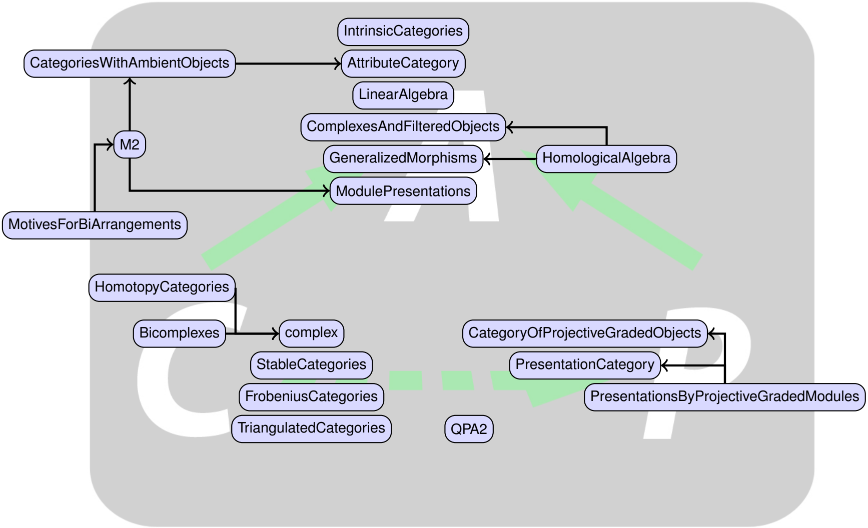
CAP packages



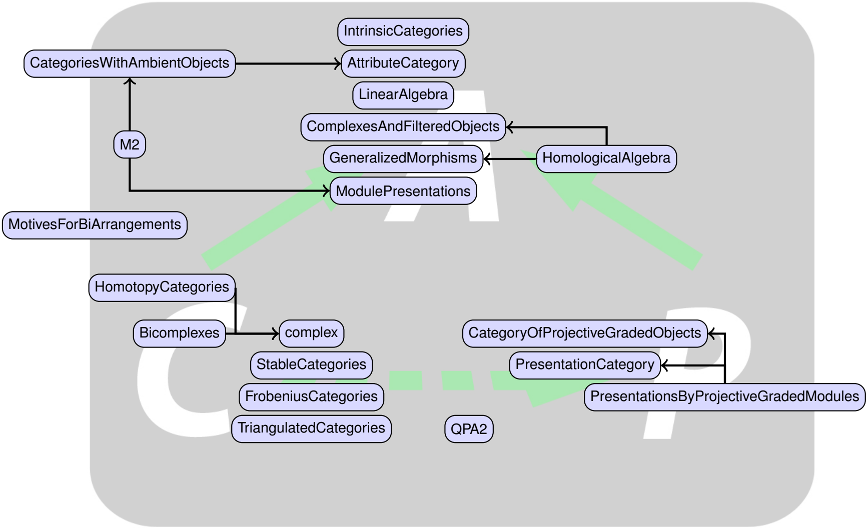
CAP packages



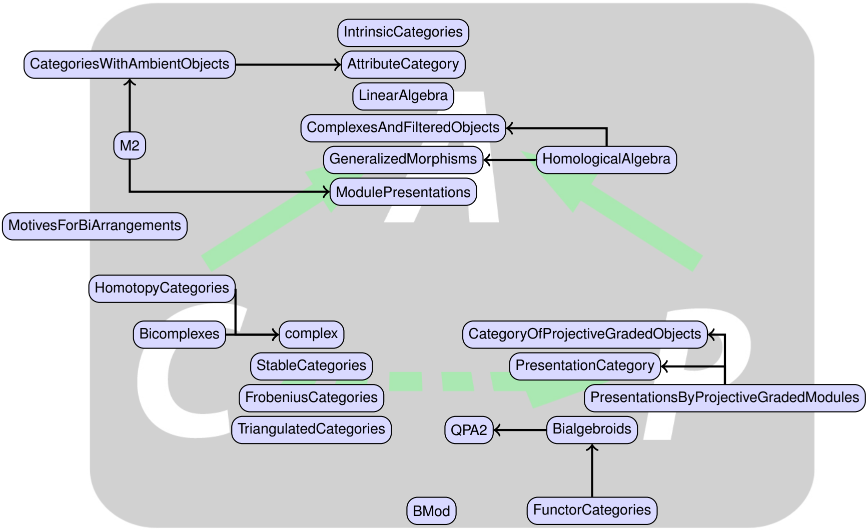
CAP packages



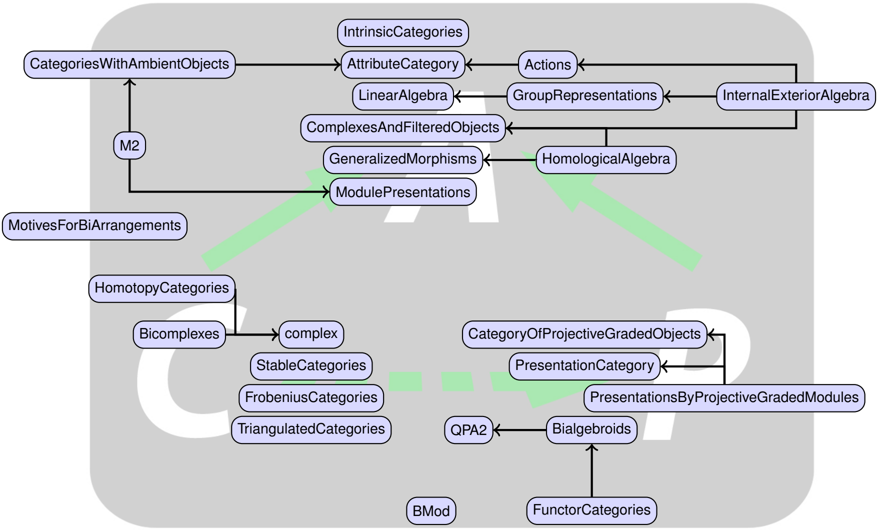
CAP packages



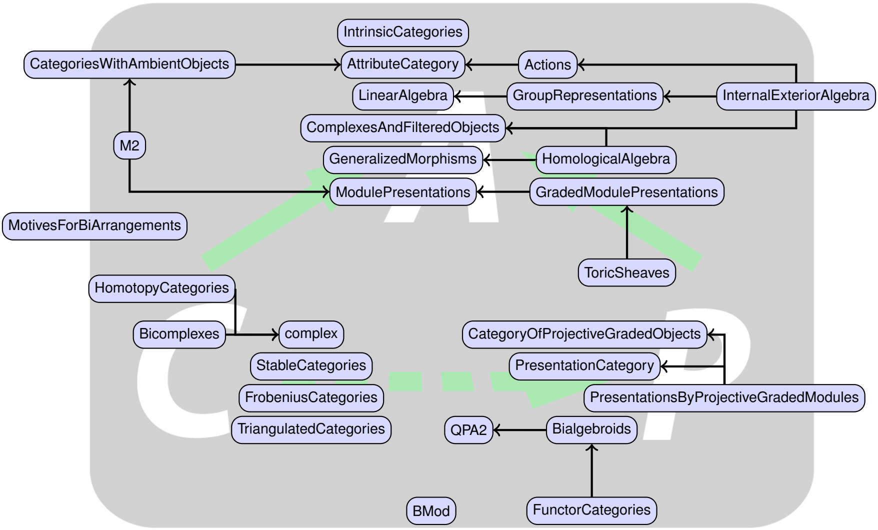
CAP packages



CAP packages



CAP packages



Snake lemma

$$\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \operatorname{coker}(\alpha) & & &
\end{tikzcd}$$

Snake lemma

$$
\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d] \arrow[dddll, bend left, blue] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Part II

Generalized morphisms

1. Classical diagram chases
2. Additive relations
3. Generalized morphisms
4. Applications of generalized morphisms
   - An algorithm for spectral sequences
   - The purity filtration

1. Classical diagram chases
2. Additive relations
3. Generalized morphisms
4. Applications of generalized morphisms
   - An algorithm for spectral sequences
   - The purity filtration

What are diagram chases?

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

1. properties

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

1. properties
2. the existence

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

1. properties
2. the existence

of morphisms

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

1. properties
2. the existence

of morphisms situated in (commutative) diagrams of prescribed shape.

What are diagram chases?

Diagram chases are a tool in homological algebra used for proving

1. properties
2. the existence

of morphisms situated in (commutative) diagrams of prescribed shape.

Connecting homomorphism in the snake lemma

$$
\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Wanted: $\ker(\gamma) \xrightarrow{\partial} \mathrm{coker}(\alpha)$.

Connecting homomorphism in the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

Start: $c \in \ker(\gamma)$.

Connecting homomorphism in the snake lemma

$$
\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \operatorname{coker}(\alpha) & & &
\end{tikzcd}
$$

This lies in C.

Connecting homomorphism in the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & {\color{red}b} \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & B' \arrow[r] & C' & \\
 & \operatorname{coker}(\alpha) & & &
\end{tikzcd}$$

Choose: $b \in \epsilon^{-1}(\{c\})$.

Connecting homomorphism in the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d] & {\color{red}b'} \in B' \arrow[r] & C' & \\
 & \operatorname{coker}(\alpha) & & &
\end{tikzcd}$$

Map: $b \overset{\beta}{\mapsto} b'$.

Connecting homomorphism in the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & {\color{red}a'} \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

Compute: $a' \in \mu^{-1}(b')$.

Connecting homomorphism in the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & {\color{red} a' + \mathrm{im}(\alpha)} \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

Map: $a' \mapsto a' + \mathrm{im}(\alpha)$.

Connecting homomorphism in the snake lemma

$$
\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] \arrow[dddll, blue, rounded corners] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & a' + \mathrm{im}(\alpha) \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Result: $c \overset{\partial}{\mapsto} a' + \mathrm{im}(\alpha)$.

Connecting homomorphism in the snake lemma

$$
\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] \arrow[dddll] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & a' + \mathrm{im}(\alpha) \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Result: $c \overset{\partial}{\mapsto} a' + \mathrm{im}(\alpha)$. **Context**: modules

Classical solutions: embedding theorems

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Any small abelian category $\mathbf{A}$ admits an exact fully faithful covariant embedding

$$F : \mathbf{A} \hookrightarrow R - \mathbf{mod}$$

into the category of R-modules for some ring R.

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Any small abelian category $\mathbf{A}$ admits an exact fully faithful covariant embedding

$$F : \mathbf{A} \hookrightarrow R - \mathbf{mod}$$

into the category of R-modules for some ring R.

Application: existence of morphisms

$$\mathrm{Hom}_{\mathbf{A}}(A, B) \quad \cong \quad \mathrm{Hom}_{R-\mathbf{mod}}(FA, FB)$$

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Any small abelian category $\mathbf{A}$ admits an exact fully faithful covariant embedding

$$F : \mathbf{A} \hookrightarrow R - \mathbf{mod}$$

into the category of R-modules for some ring R.

Application: existence of morphisms

$$\begin{array}{ccc} \mathrm{Hom}_{\mathbf{A}}(A, B) & \cong & \mathrm{Hom}_{R-\mathbf{mod}}(FA, FB) \\ & & \uplus \\ & & \varphi \end{array}$$

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Any small abelian category $\mathbf{A}$ admits an exact fully faithful covariant embedding

$$F : \mathbf{A} \hookrightarrow R - \mathbf{mod}$$

into the category of R-modules for some ring R.

Application: existence of morphisms

$$\begin{array}{ccc} \mathrm{Hom}_{\mathbf{A}}(A, B) & \cong & \mathrm{Hom}_{R-\mathbf{mod}}(FA, FB) \\ \Cup & & \Cup \\ F^{-1}\varphi & \leftrightarrow & \varphi \end{array}$$

Classical solutions: embedding theorems

Freyd-Mitchell embedding theorem

Any small abelian category $\mathbf{A}$ admits an exact fully faithful covariant embedding

$$F : \mathbf{A} \hookrightarrow R - \mathbf{mod}$$

into the category of R-modules for some ring R.

Application: existence of morphisms

$$\begin{array}{ccc} \mathrm{Hom}_{\mathbf{A}}(A, B) & \cong & \mathrm{Hom}_{R-\mathbf{mod}}(FA, FB) \\ \Cup & & \Cup \\ F^{-1}\varphi & \leftrightarrow & \varphi \end{array}$$

Problem: this isomorphism between Hom-sets is **not constructive.**

Back to the snake lemma

$$\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] \arrow[dddll, blue, bend left] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & b \in B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & a' + \mathrm{im}(\alpha) \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

Result: $c \overset{\partial}{\mapsto} a' + \mathrm{im}(\alpha)$.

Back to the snake lemma

$$
\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] \arrow[dddll, blue] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & {\color{red} b \in B} \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & a' + \mathrm{im}(\alpha) \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Crucial step: the **uncanonical** choice $b \in \epsilon^{-1}(\{c\})$.

Back to the snake lemma

$$
\begin{tikzcd}
 & & & c \in \ker(\gamma) \arrow[d] \arrow[dddll, blue, rounded corners] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & {\color{red} b \in B} \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & c \in C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & a' \in A' \arrow[r, "\mu"'] \arrow[d] & b' \in B' \arrow[r] & C' & \\
 & a' + \mathrm{im}(\alpha) \in \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

Make this step canonical: **relations** instead of maps: $c \mapsto \epsilon^{-1}(\{c\})$

Relations

Let A, B be abelian groups.

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from A to B.**

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from** A **to** B.

Example

Let $\epsilon : A \to B$ be a homomorphism of abelian groups.

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from A to B**.

Example

Let $\epsilon : A \to B$ be a homomorphism of abelian groups.

$$\Gamma(\epsilon) := \{(a, b) \in A \oplus B \mid \epsilon(a) = b\}$$

is a relation from A to B

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from A to B**.

Example

Let $\epsilon : A \to B$ be a homomorphism of abelian groups.

$$\Gamma(\epsilon) := \{(a, b) \in A \oplus B \mid \epsilon(a) = b\}$$

is a relation from A to B, called **graph of ϵ**

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from A to B**.

Example

Let $\epsilon : A \to B$ be a homomorphism of abelian groups.

$$\Gamma(\epsilon) := \{(a, b) \in A \oplus B \mid \epsilon(a) = b\}$$

is a relation from A to B, called **graph of ϵ**, and

$$\epsilon^{-1} := \{(b, a) \in B \oplus A \mid \epsilon(a) = b\}$$

is a relation from B to A

Relations

Let A, B be abelian groups.

Definition

A subgroup $f \subseteq A \oplus B$ is called a **relation from A to B**.

Example

Let $\epsilon : A \to B$ be a homomorphism of abelian groups.

$$\Gamma(\epsilon) := \{(a, b) \in A \oplus B \mid \epsilon(a) = b\}$$

is a relation from A to B, called **graph of** ϵ, and

$$\epsilon^{-1} := \{(b, a) \in B \oplus A \mid \epsilon(a) = b\}$$

is a relation from B to A, called **pseudo-inverse of** ϵ.

Relations

Composition of relations

Relations

Composition of relations

Given $f \subseteq A \oplus B$ and $g \subseteq B \oplus C$, define

Relations

Composition of relations

Given $f \subseteq A \oplus B$ and $g \subseteq B \oplus C$, define

$$g \circ f := \{(a, c) \in A \oplus C \mid \exists b \in B : (a, b) \in f, (b, c) \in g\}$$

Relations

Composition of relations

Given $f \subseteq A \oplus B$ and $g \subseteq B \oplus C$, define

$$g \circ f := \{(a, c) \in A \oplus C \mid \exists b \in B : (a, b) \in f, (b, c) \in g\}$$

If f and g correspond to maps, this describes their usual composition.

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$\mathrm{dom}(f) := \left\{ a \in A \mid \exists b \in B : (a, b) \in f \right\}$

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$$\mathrm{dom}(f) := \left\{ a \in A \mid \exists b \in B : (a, b) \in f \right\}$$

Defect

$$\mathrm{def}(f) := \left\{ b \in B \mid (0, b) \in f \right\}$$

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$\mathrm{dom}(f) := \left\{ a \in A \mid \exists b \in B : (a, b) \in f \right\}$

Defect

$\mathrm{def}(f) := \left\{ b \in B \mid (0, b) \in f \right\}$

A: When it has a full domain

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$\mathrm{dom}(f) := \left\{ a \in A \mid \exists b \in B : (a, b) \in f \right\} = A$

Defect

$\mathrm{def}(f) := \left\{ b \in B \mid (0, b) \in f \right\}$

A: When it has a full domain

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$\mathrm{dom}(f) := \{a \in A \mid \exists b \in B : (a,b) \in f\} = A$

Defect

$\mathrm{def}(f) := \{b \in B \mid (0,b) \in f\}$

A: When it has a full domain and 0 defect.

Relations

Q: When does an additive relation $f \subseteq A \oplus B$ defines an honest map (a group homomorphism)?

Domain

$\mathrm{dom}(f) := \left\{ a \in A \mid \exists b \in B : (a, b) \in f \right\} = A$

Defect

$\mathrm{def}(f) := \left\{ b \in B \mid (0, b) \in f \right\} = 0$

A: When it has a full domain and 0 defect.

The snake lemma for a last time

$$\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

Wanted: $\ker(\gamma) \xrightarrow{\partial} \mathrm{coker}(\alpha)$.

The snake lemma for a last time

$$
\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] & C' & \\
 & \operatorname{coker}(\alpha) & & &
\end{tikzcd}
$$

ι

The snake lemma for a last time

$$
\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] \arrow[l, bend right, "\epsilon^{-1}"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

$$\epsilon^{-1} \circ \iota$$

The snake lemma for a last time

$$\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] \arrow[l, bend right, "\epsilon^{-1}"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

$$\beta \circ \epsilon^{-1} \circ \iota$$

The snake lemma for a last time

$$\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] & \\
 & A \arrow[r] \arrow[d, "\alpha"] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"] & C \arrow[r] \arrow[d, "\gamma"] \arrow[l, bend right, "\epsilon^{-1}"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"] & B' \arrow[r] \arrow[l, bend left, "\mu^{-1}"] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}$$

$$\mu^{-1} \circ \beta \circ \epsilon^{-1} \circ \iota$$

The snake lemma for a last time

$$
\begin{tikzcd}
& & & \ker(\gamma) \arrow[d, "\iota"] & \\
& A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] \arrow[l, bend right, "\epsilon^{-1}"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] \arrow[l, bend left, "\mu^{-1}"] & C' & \\
& \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

$$\pi \circ \mu^{-1} \circ \beta \circ \epsilon^{-1} \circ \iota$$

The snake lemma for a last time

$$
\begin{tikzcd}
 & & & \ker(\gamma) \arrow[d, "\iota"] \arrow[dddll, rounded corners] & \\
 & A \arrow[r] \arrow[d, "\alpha"'] & B \arrow[r, "\epsilon"] \arrow[d, "\beta"'] & C \arrow[r] \arrow[d, "\gamma"'] \arrow[l, bend right, "\epsilon^{-1}"'] & 0 \\
0 \arrow[r] & A' \arrow[r, "\mu"'] \arrow[d, "\pi"'] & B' \arrow[r] \arrow[l, bend left, "\mu^{-1}"] & C' & \\
 & \mathrm{coker}(\alpha) & & &
\end{tikzcd}
$$

∂ is an honest map given by a composition of relations!

From relations to generalized morphisms

- **Wanted**: a categorical framework for relations.

From relations to generalized morphisms

- **Wanted**: a categorical framework for relations.
- **Solution**: generalized morphisms.

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation

$$
\begin{tikzcd}
A \oplus B \\
D \arrow[u, hook]
\end{tikzcd}
$$

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation

$$\begin{tikzcd} A \oplus B \\ D \arrow[u, hook, "\begin{pmatrix} \alpha & \beta \end{pmatrix}"] \end{tikzcd}$$

From relations to generalized morphisms

Let A, B be objects in an abelian category **A**.

Relation

$$\begin{tikzcd} A & & B \\ & D \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation $\rightsquigarrow$ generalized morphism

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul, "\alpha", red] \arrow[ur, "\beta"', blue] & \end{tikzcd}$$

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation $\rightsquigarrow$ generalized morphism (data structure: span)

$$
\begin{tikzcd}
A \arrow[rr, dashed] & & B \\
& D \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] &
\end{tikzcd}
$$

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation $\rightsquigarrow$ generalized morphism (data structure: span)

$$
\begin{tikzcd}
A \arrow[rr, dashed] & & B \\
& D \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] &
\end{tikzcd}
$$

Equality

From relations to generalized morphisms

Let A, B be objects in an abelian category **A**.

Relation $\rightsquigarrow$ generalized morphism (data structure: span)

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

Equality

Two spans (α, β) and (α', β') are **equal as generalized morphisms** if

From relations to generalized morphisms

Let A, B be objects in an abelian category $\mathbf{A}$.

Relation $\rightsquigarrow$ generalized morphism (data structure: span)

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

Equality

Two spans (α, β) and (α', β') are **equal as generalized morphisms** if

$$\operatorname{im}\left((\alpha, \beta) : D \to A \oplus B\right) = \operatorname{im}\left((\alpha', \beta') : D' \to A \oplus B\right).$$

Composition of generalized morphisms

Composition

Composition of generalized morphisms

Composition

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\ & X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Composition of generalized morphisms

Composition

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\ & X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Composition of generalized morphisms

Composition

$$\begin{tikzcd}
A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\
& X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

Composition of generalized morphisms

Composition

$$
\begin{tikzcd}
A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\
& X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \\
& & X \times_B Y \arrow[ul] \arrow[ur] & &
\end{tikzcd}
$$

Composition of generalized morphisms

Composition

$$\begin{tikzcd}
A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\
& X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \\
& & X \times_B Y \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

Composition of generalized morphisms

Composition

$$\begin{tikzcd}
A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\
& X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \\
& & X \times_B Y \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

Composition of generalized morphisms

Composition

$$\begin{tikzcd}
A \arrow[rr, dashed] & & B \arrow[rr, dashed] & & C \\
& X \arrow[ul] \arrow[ur] & & Y \arrow[ul] \arrow[ur] & \\
& & X \times_B Y \arrow[ul] \arrow[ur] & &
\end{tikzcd}$$

$\rightsquigarrow$ Category of generalized morphisms $\mathrm{G}(\mathbf{A})$

Pseudo-inverses

Pseudo-inverses

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Pseudo-inverses

Pseudo-inverses

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul, red] \arrow[ur, blue] & \end{tikzcd}$$

Pseudo-inverses

Pseudo-inverses

$$
\begin{tikzcd}
A \arrow[rr, dashed] & & B \\
& D \arrow[ul, red] \arrow[ur, blue] &
\end{tikzcd}
$$

Pseudo-inverses

Pseudo-inverses

$$\begin{tikzcd} A \arrow[rr, dashed] & & B \\ & D \arrow[ul, red] \arrow[ur, blue] & \end{tikzcd} \quad \rightsquigarrow$$

Pseudo-inverses

Pseudo-inverses

$$
\begin{tikzcd}
A \arrow[rr, dashed] & & B \\
& D \arrow[ul, red] \arrow[ur, blue] \arrow[u, phantom, "\leftrightarrow"] &
\end{tikzcd}
\quad \rightsquigarrow \quad
\begin{tikzcd}
B \arrow[rr, dashed] & & A \\
& D \arrow[ul, blue] \arrow[ur, red] &
\end{tikzcd}
$$

Honest morphisms

Honest morphisms

Honest morphisms

A embeds into G(**A**):

Honest morphisms

Honest morphisms

A embeds into G(**A**):

$$A \longrightarrow B$$

Honest morphisms

Honest morphisms

A embeds into G(**A**):

$$A \xrightarrow{\quad\quad\quad} B \quad \mapsto$$

Honest morphisms

Honest morphisms

A embeds into G(**A**):

$$
\begin{tikzcd}
 & & & A \arrow[rr, dashed] & & B \\
A \arrow[r] & B & \mapsto & & & \\
 & & & & A \arrow[uul, "\mathrm{id}_A"] \arrow[uur] &
\end{tikzcd}
$$

Honest morphisms

Honest morphisms

A embeds into G(**A**):

$$
A \longrightarrow B \quad \mapsto \quad
\begin{tikzcd}
A \arrow[rr, dashed] & & B \\
& A \arrow[ul, "\mathrm{id}_A"] \arrow[ur] &
\end{tikzcd}
$$

Generalized morphisms with such a representation are called **honest**.

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha)$

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha)$

Defect

$\mathrm{def}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \beta(\mathrm{ker}(\alpha))$

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha)$

Defect

$\mathrm{def}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \beta(\mathrm{ker}(\alpha))$

A: When it has a full domain

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha) = A$

Defect

$\mathrm{def}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \beta(\ker(\alpha))$

A: When it has a full domain

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha) = A$

Defect

$\mathrm{def}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \beta(\ker(\alpha))$

A: When it has a full domain and 0 defect.

Honest morphisms

Q: When does $A \xleftarrow{\alpha} D \xrightarrow{\beta} B$ define an honest morphism?

Domain

$\mathrm{dom}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \mathrm{im}(\alpha) = A$

Defect

$\mathrm{def}(A \xleftarrow{\alpha} D \xrightarrow{\beta} B) := \beta(\ker(\alpha)) = 0$

A: When it has a full domain and 0 defect.

Computing representatives

Computing representatives

Given an honest generalized morphism in $G(\mathbf{A})$, compute the corresponding morphism in $\mathbf{A}$.

Computing representatives

Given an honest generalized morphism in G($\mathbf{A}$), compute the corresponding morphism in $\mathbf{A}$.

$$\begin{tikzcd} A & & B \\ & D \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$
\begin{tikzcd}
 & A \sqcup_D B & \\
A \arrow[ur] & & B \arrow[ul] \\
 & D \arrow[ul] \arrow[ur] &
\end{tikzcd}
$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$\begin{tikzcd} & A \sqcup_D B & \\ A \arrow[ur] & & B \arrow[ul] \\ & A \times_{A \sqcup_D B} B \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$\begin{tikzcd}
A & & B \\
& A \times_{A \sqcup_D B} B \arrow[ul, "\sim"'] \arrow[ur] &
\end{tikzcd}$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$
\begin{tikzcd}
A \arrow[dr, "\sim"] & & B \\
& A \times_{A \sqcup_D B} B \arrow[ur] &
\end{tikzcd}
$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$A \longrightarrow B$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$
\begin{tikzcd}
\mathbb{Q} & & \mathbb{Q} \\
& \mathbb{Q}^2 \arrow[ul, "\begin{pmatrix} 1 \\ -1 \end{pmatrix}"] \arrow[ur, "\begin{pmatrix} 2 \\ -2 \end{pmatrix}"'] &
\end{tikzcd}
$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$
\begin{tikzcd}
 & \mathbb{Q} & \\
\mathbb{Q} \arrow[ur, "2"] & & \mathbb{Q} \arrow[ul, "1"'] \\
 & \mathbb{Q}^2 \arrow[ul, "\begin{pmatrix} 1 \\ -1 \end{pmatrix}"] \arrow[ur, "\begin{pmatrix} 2 \\ -2 \end{pmatrix}"'] &
\end{tikzcd}
$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$\begin{tikzcd}
& \mathbb{Q} & \\
\mathbb{Q} \arrow[ur, "2"] & & \mathbb{Q} \arrow[ul, "1"'] \\
& \mathbb{Q} \arrow[ul, "1"] \arrow[ur, "2"'] &
\end{tikzcd}$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$
\begin{tikzcd}
\mathbb{Q} & & \mathbb{Q} \\
& \mathbb{Q} \arrow[ul, "1", "\sim"'] \arrow[ur, "2"'] &
\end{tikzcd}
$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$\begin{tikzcd}
\mathbb{Q} \arrow[dr, "1"', "\sim"] & & \mathbb{Q} \\
& \mathbb{Q} \arrow[ur, "2"'] &
\end{tikzcd}$$

Computing representatives

Given an honest generalized morphism in G(**A**), compute the corresponding morphism in **A**.

$$\mathbb{Q} \xrightarrow{2} \mathbb{Q}$$

Constructive diagram chases

Constructive diagram chases

Strategy for constructive diagram chases

Constructive diagram chases

Strategy for constructive diagram chases

1. Compute in $\mathrm{G}(\mathbf{A})$ using pseudo-inverses and compositions.

Constructive diagram chases

Strategy for constructive diagram chases

1. Compute in $\mathrm{G}(\mathbf{A})$ using pseudo-inverses and compositions.
2. Compute the honest representative of the resulting generalized morphism.

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$.

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$P_{i+1} \xrightarrow{\partial_{i+1}} P_i \xrightarrow{\partial_i} P_{i-1}$$

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$
\begin{tikzcd}
P_{i+1} \arrow[rr, "\partial_{i+1}"] & & P_i \arrow[rr, "\partial_i"] & & P_{i-1} \\
& \operatorname{im}(\partial_{i+1}) \arrow[ur, hook] & & &
\end{tikzcd}
$$

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$\begin{tikzcd}
P_{i+1} \arrow[rr, "\partial_{i+1}"] & & P_i \arrow[rr, "\partial_i"] & & P_{i-1} \\
& \operatorname{im}(\partial_{i+1}) \arrow[ur, hook] & & \ker(\partial_i) \arrow[ul, hook'] &
\end{tikzcd}$$

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$\begin{tikzcd}
P_{i+1} \arrow[rr, "\partial_{i+1}"] & & P_i \arrow[rr, "\partial_i"] & & P_{i-1} \\
& \operatorname{im}(\partial_{i+1}) \arrow[ur, hook] \arrow[rr, hook] & & \ker(\partial_i) \arrow[ul, hook'] &
\end{tikzcd}$$

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$\begin{tikzcd}
P_{i+1} \arrow[rr, "\partial_{i+1}"] & & P_i \arrow[rr, "\partial_i"] & & P_{i-1} \\
& \mathrm{im}\,(\partial_{i+1}) \arrow[ur, hook] \arrow[rr, hook] & & \ker\,(\partial_i) \arrow[ul, hook'] \arrow[r, two heads] & \mathrm{H}_i\,(P_\bullet)
\end{tikzcd}$$

Example: functoriality of homology

Let $(P_\bullet, \partial)$ be a complex in an abelian category $\mathcal{A}$. Then we can compute the generalized embedding of the i-th homology.

$$\begin{tikzcd}
P_{i+1} \arrow[rr, "\partial_{i+1}"] & & P_i \arrow[rr, "\partial_i"] & & P_{i-1} \\
& \mathrm{im}\,(\partial_{i+1}) \arrow[ur, hook] \arrow[rr, hook] & & \ker\,(\partial_i) \arrow[ul, hook, red] \arrow[r, two heads, red] & \mathrm{H}_i\,(P_\bullet) \arrow[ull, hook, dashed]
\end{tikzcd}$$

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism.

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i(P_\bullet) \to \mathrm{H}_i(Q_\bullet)$ can be computed using generalized morphisms:

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i\left(P_\bullet\right) \to \mathrm{H}_i\left(Q_\bullet\right)$ can be computed using generalized morphisms:

$$P_i \xrightarrow{\varepsilon_i} Q_i$$

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i(P_\bullet) \to \mathrm{H}_i(Q_\bullet)$ can be computed using generalized morphisms:

$$\mathrm{H}_i(P_\bullet) \dashrightarrow P_i \xrightarrow{\varepsilon_i} Q_i$$

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i(P_\bullet) \to \mathrm{H}_i(Q_\bullet)$ can be computed using generalized morphisms:

$$\mathrm{H}_i(P_\bullet) \dashrightarrow P_i \xrightarrow{\varepsilon_i} Q_i \dashleftarrow \mathrm{H}_i(Q_\bullet)$$

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i(P_\bullet) \to \mathrm{H}_i(Q_\bullet)$ can be computed using generalized morphisms:

$$\mathrm{H}_i(P_\bullet) \dashrightarrow P_i \xrightarrow{\varepsilon_i} Q_i \dashrightarrow \mathrm{H}_i(Q_\bullet)$$

Example: functoriality of homology

Theorem

Let $\mathcal{A}$ be an abelian category and $\varepsilon : P_\bullet \to Q_\bullet$ a chain morphism. Then the morphism $\mathrm{H}_i(P_\bullet) \to \mathrm{H}_i(Q_\bullet)$ can be computed using generalized morphisms:

$$\begin{tikzcd} \mathrm{H}_i(P_\bullet) \arrow[r, dashed] \arrow[rrr, bend right] & P_i \arrow[r, "\varepsilon_i"] & Q_i \arrow[r, dashed, red] & \mathrm{H}_i(Q_\bullet) \end{tikzcd}$$

1. Classical diagram chases
2. Additive relations
3. Generalized morphisms
4. Applications of generalized morphisms
   - An algorithm for spectral sequences
   - The purity filtration

Spectral sequences via generalized morphisms

Given: an excerpt of a filtered chain complex.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
 & & \uparrow & & \uparrow & & \uparrow & & \\
\cdots & \longrightarrow & A_{i+1} & \longrightarrow & B_{i+1} & \longrightarrow & C_{i+1} & \longrightarrow & \cdots \\
 & & \uparrow & & \uparrow & & \uparrow & & \\
\cdots & \longrightarrow & A_{i} & \longrightarrow & B_{i} & \longrightarrow & C_{i} & \longrightarrow & \cdots \\
 & & \uparrow & & \uparrow & & \uparrow & & \\
\cdots & \longrightarrow & A_{i-1} & \longrightarrow & B_{i-1} & \longrightarrow & C_{i-1} & \longrightarrow & \cdots \\
 & & \uparrow & & \uparrow & & \uparrow & & \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

Spectral sequences via generalized morphisms

We pass to its graded parts.

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots & \\
\cdots \arrow[r] & \frac{A_{i+1}}{A_i} \arrow[r, "\overline{\partial}"] & \frac{B_{i+1}}{B_i} \arrow[r, "\overline{\partial}"] & \frac{C_{i+1}}{C_i} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_i}{A_{i-1}} \arrow[r] & \frac{B_i}{B_{i-1}} \arrow[r] & \frac{C_i}{C_{i-1}} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_{i-1}}{A_{i-2}} \arrow[r] & \frac{B_{i-1}}{B_{i-2}} \arrow[r] & \frac{C_{i-1}}{C_{i-2}} \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots &
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

We can compute the differentials via generalized morphisms.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$\overline{\partial}$:

Spectral sequences via generalized morphisms

We can compute the differentials via generalized morphisms.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$\overline{\partial} : \frac{A_{i+1}}{A_i} \qquad\qquad \frac{B_{i+1}}{B_i}$$

Spectral sequences via generalized morphisms

We can compute the differentials via generalized morphisms.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$\overline{\partial} \; : \; \frac{A_{i+1}}{A_i} \twoheadleftarrow A_{i+1} \qquad\qquad \frac{B_{i+1}}{B_i}$$

Spectral sequences via generalized morphisms

We can compute the differentials via generalized morphisms.

$$\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}$$

$$\overline{\partial} \; : \; \frac{A_{i+1}}{A_i} \twoheadleftarrow A_{i+1} \hookrightarrow A \qquad\qquad \frac{B_{i+1}}{B_i}$$

Spectral sequences via generalized morphisms

This is a generalized **subquotient embedding**.

$$\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}$$

$$\overline{\partial} \; : \; \begin{tikzcd} \frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \end{tikzcd} \qquad \frac{B_{i+1}}{B_i}$$

Spectral sequences via generalized morphisms

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots & \\
\cdots \arrow[r] & \frac{A_{i+1}}{A_i} \arrow[r, "\overline{\partial}"] & \frac{B_{i+1}}{B_i} \arrow[r, "\overline{\partial}"] & \frac{C_{i+1}}{C_i} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_i}{A_{i-1}} \arrow[r] & \frac{B_i}{B_{i-1}} \arrow[r] & \frac{C_i}{C_{i-1}} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_{i-1}}{A_{i-2}} \arrow[r] & \frac{B_{i-1}}{B_{i-2}} \arrow[r] & \frac{C_{i-1}}{C_{i-2}} \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\overline{\partial} : \begin{tikzcd}
\frac{A_{i+1}}{A_i} \arrow[rr, bend left, dashed] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B
\end{tikzcd} \qquad \frac{B_{i+1}}{B_i}
$$

Spectral sequences via generalized morphisms

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i}}{A_{i-1}} & \longrightarrow & \frac{B_{i}}{B_{i-1}} & \longrightarrow & \frac{C_{i}}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$
\overline{\partial} : \quad
\begin{tikzcd}
\frac{A_{i+1}}{A_i} & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[ll, bend right, dashed] \arrow[r, "\partial"] & B & B_{i+1} \arrow[l, hook'] & & \frac{B_{i+1}}{B_i}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$
\overline{\partial} : \begin{tikzcd} \frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i} \end{tikzcd}
$$

Spectral sequences via generalized morphisms

This is a generalized **subquotient projection**.

$$\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots & \\
\cdots \arrow[r] & \frac{A_{i+1}}{A_i} \arrow[r, "\overline{\partial}"] & \frac{B_{i+1}}{B_i} \arrow[r, "\overline{\partial}"] & \frac{C_{i+1}}{C_i} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_i}{A_{i-1}} \arrow[r] & \frac{B_i}{B_{i-1}} \arrow[r] & \frac{C_i}{C_{i-1}} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_{i-1}}{A_{i-2}} \arrow[r] & \frac{B_{i-1}}{B_{i-2}} \arrow[r] & \frac{C_{i-1}}{C_{i-2}} \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots &
\end{tikzcd}$$

$$\overline{\partial} : \begin{tikzcd}
\frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B \arrow[rr, dashed, bend left] & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i}
\end{tikzcd}$$

Spectral sequences via generalized morphisms

We can compose the arrows.

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots & \\
\cdots \arrow[r] & \frac{A_{i+1}}{A_i} \arrow[r, "\overline{\partial}"] & \frac{B_{i+1}}{B_i} \arrow[r, "\overline{\partial}"] & \frac{C_{i+1}}{C_i} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_i}{A_{i-1}} \arrow[r] & \frac{B_i}{B_{i-1}} \arrow[r] & \frac{C_i}{C_{i-1}} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_{i-1}}{A_{i-2}} \arrow[r] & \frac{B_{i-1}}{B_{i-2}} \arrow[r] & \frac{C_{i-1}}{C_{i-2}} \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\begin{tikzcd}
\overline{\partial} \ : \ \frac{A_{i+1}}{A_i} \arrow[rr, bend left, dashed, red] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B \arrow[rr, bend left, dashed] & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

We can compose the arrows.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_i}{A_{i-1}} & \longrightarrow & \frac{B_i}{B_{i-1}} & \longrightarrow & \frac{C_i}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$
\overline{\partial} : \begin{tikzcd} \frac{A_{i+1}}{A_i} \arrow[rr, bend left, dashed] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] \arrow[rr, bend left, dashed] & B & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i} \end{tikzcd}
$$

Spectral sequences via generalized morphisms

We can compose the arrows.

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots & \\
\cdots \arrow[r] & \frac{A_{i+1}}{A_i} \arrow[r, "\overline{\partial}"] & \frac{B_{i+1}}{B_i} \arrow[r, "\overline{\partial}"] & \frac{C_{i+1}}{C_i} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_i}{A_{i-1}} \arrow[r] & \frac{B_i}{B_{i-1}} \arrow[r] & \frac{C_i}{C_{i-1}} \arrow[r] & \cdots \\
\cdots \arrow[r] & \frac{A_{i-1}}{A_{i-2}} \arrow[r] & \frac{B_{i-1}}{B_{i-2}} \arrow[r] & \frac{C_{i-1}}{C_{i-2}} \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\overline{\partial} : \begin{tikzcd}
\frac{A_{i+1}}{A_i} \arrow[rr, bend left, dashed] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] \arrow[rr, bend left, dashed] & B & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

This formula still makes sense if we map 1 step down.

$$
\begin{array}{ccccccccc}
\cdots & \longrightarrow & A & \xrightarrow{\partial} & B & \xrightarrow{\partial} & C & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & & \\
\cdots & \longrightarrow & \frac{A_{i+1}}{A_i} & \xrightarrow{\overline{\partial}} & \frac{B_{i+1}}{B_i} & \xrightarrow{\overline{\partial}} & \frac{C_{i+1}}{C_i} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i}}{A_{i-1}} & \longrightarrow & \frac{B_{i}}{B_{i-1}} & \longrightarrow & \frac{C_{i}}{C_{i-1}} & \longrightarrow & \cdots \\
\cdots & \longrightarrow & \frac{A_{i-1}}{A_{i-2}} & \longrightarrow & \frac{B_{i-1}}{B_{i-2}} & \longrightarrow & \frac{C_{i-1}}{C_{i-2}} & \longrightarrow & \cdots \\
 & & \vdots & & \vdots & & \vdots & &
\end{array}
$$

$$
\overline{\partial} : \begin{tikzcd} \frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B \arrow[rr, dashed, bend left] & B_{i+1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i+1}}{B_i} \end{tikzcd}
$$

Spectral sequences via generalized morphisms

This formula still makes sense if we map 1 step down.

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots & \\
\cdots \arrow[dr, dashed] & \frac{A_{i+1}}{A_i} \arrow[dr, dashed, "\overline{\partial}^1"] & \frac{B_{i+1}}{B_i} \arrow[dr, dashed] & \frac{C_{i+1}}{C_i} \arrow[dr, dashed] & \cdots \\
\cdots \arrow[dr, dashed] & \frac{A_i}{A_{i-1}} \arrow[dr, dashed] & \frac{B_i}{B_{i-1}} \arrow[dr, dashed, "\overline{\partial}^1"] & \frac{C_i}{C_{i-1}} \arrow[dr, dashed] & \cdots \\
\cdots & \frac{A_{i-1}}{A_{i-2}} & \frac{B_{i-1}}{B_{i-2}} & \frac{C_{i-1}}{C_{i-2}} & \cdots \\
 & \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\begin{tikzcd}
\overline{\partial}^1 : \frac{A_{i+1}}{A_i} \arrow[rr, bend left, dashed, red] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, red, "\partial"] & B \arrow[rr, bend left, dashed, red] & B_i \arrow[l, hook'] \arrow[r, two heads] & \frac{B_i}{B_{i-1}}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

One more step . . .

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
& \vdots & \vdots & \vdots & \\
\cdots \arrow[dr, dashed] & \frac{A_{i+1}}{A_i} \arrow[dr, dashed, "\overline{\partial}^1"] & \frac{B_{i+1}}{B_i} \arrow[dr, dashed] & \frac{C_{i+1}}{C_i} \arrow[dr, dashed] & \cdots \\
\cdots \arrow[dr, dashed] & \frac{A_i}{A_{i-1}} \arrow[dr, dashed] & \frac{B_i}{B_{i-1}} \arrow[dr, dashed, "\overline{\partial}^1"] & \frac{C_i}{C_{i-1}} \arrow[dr, dashed] & \cdots \\
\cdots & \frac{A_{i-1}}{A_{i-2}} & \frac{B_{i-1}}{B_{i-2}} & \frac{C_{i-1}}{C_{i-2}} & \cdots \\
& \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\overline{\partial}^1 : \begin{tikzcd}
\frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B \arrow[rr, dashed, bend left] & B_i \arrow[l, hook'] \arrow[r, two heads] & \frac{B_i}{B_{i-1}}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

One more step ...

$$
\begin{tikzcd}
\cdots \arrow[r] & A \arrow[r, "\partial"] & B \arrow[r, "\partial"] & C \arrow[r] & \cdots \\
 & \vdots & \vdots & \vdots & \\
\cdots \arrow[ddr, dashed] & \frac{A_{i+1}}{A_i} \arrow[ddr, dashed, "\overline{\partial}^2"'] & \frac{B_{i+1}}{B_i} \arrow[ddr, dashed] & \frac{C_{i+1}}{C_i} \arrow[ddr, dashed] & \cdots \\
\cdots & \frac{A_i}{A_{i-1}} & \frac{B_i}{B_{i-1}} & \frac{C_i}{C_{i-1}} & \cdots \\
\cdots & \frac{A_{i-1}}{A_{i-2}} & \frac{B_{i-1}}{B_{i-2}} & \frac{C_{i-1}}{C_{i-2}} & \cdots \\
 & \vdots & \vdots & \vdots &
\end{tikzcd}
$$

$$
\overline{\partial}^2 : \begin{tikzcd}
\frac{A_{i+1}}{A_i} \arrow[rr, dashed, bend left] & A_{i+1} \arrow[l, two heads] \arrow[r, hook] & A \arrow[r, "\partial"] & B \arrow[rr, dashed, bend left] & B_{i-1} \arrow[l, hook'] \arrow[r, two heads] & \frac{B_{i-1}}{B_{i-2}}
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$\cdots \dashrightarrow \frac{A_{i+1}}{A_i} \xrightarrow{\overline{\partial_A^r}} \frac{B_{i+1-r}}{B_{i-r}} \xrightarrow{\overline{\partial_B^r}} \frac{C_{i+1-2r}}{C_{i-2r}} \dashrightarrow \cdots$$

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$
\begin{tikzcd}
\cdots \arrow[r, dashed] & \frac{A_{i+1}}{A_i} \arrow[r, dashed, "\overline{\partial^r_A}"] & \frac{B_{i+1-r}}{B_{i-r}} \arrow[r, dashed, "\overline{\partial^r_B}"] & \frac{C_{i+1-2r}}{C_{i-2r}} \arrow[r, dashed] & \cdots \\
& & \frac{\mathrm{dom}(\overline{\partial^r_B})}{\mathrm{def}(\overline{\partial^r_A})} \arrow[u, dashed] & &
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$
\begin{tikzcd}
\cdots \arrow[r, dashed] & \frac{A_{i+1}}{A_i} \arrow[rr, dashed, "\overline{\partial^r_A}"] & & \frac{B_{i+1-r}}{B_{i-r}} \arrow[rr, dashed, "\overline{\partial^r_B}"] & & \frac{C_{i+1-2r}}{C_{i-2r}} \arrow[r, dashed] & \cdots \\
\cdots & \frac{\mathrm{dom}}{\mathrm{def}} \arrow[u, dashed] & & \frac{\mathrm{dom}(\overline{\partial^r_B})}{\mathrm{def}(\overline{\partial^r_A})} \arrow[u, dashed] & & \frac{\mathrm{dom}}{\mathrm{def}} \arrow[u, dashed] & \cdots
\end{tikzcd}
$$

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$\begin{tikzcd}
\cdots \arrow[r, dashed] & \frac{A_{i+1}}{A_i} \arrow[r, dashed, "\overline{\partial^r_A}"] & \frac{B_{i+1-r}}{B_{i-r}} \arrow[r, dashed, "\overline{\partial^r_B}"] & \frac{C_{i+1-2r}}{C_{i-2r}} \arrow[r, dashed] & \cdots \\
\cdots \arrow[r] & \frac{\mathrm{dom}}{\mathrm{def}} \arrow[r] \arrow[u, dashed] & \frac{\mathrm{dom}(\overline{\partial^r_B})}{\mathrm{def}(\overline{\partial^r_A})} \arrow[r] \arrow[u, dashed] & \frac{\mathrm{dom}}{\mathrm{def}} \arrow[r] \arrow[u, dashed] & \cdots
\end{tikzcd}$$

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$\begin{tikzcd}
\cdots \arrow[r, dashed] & \frac{A_{i+1}}{A_i} \arrow[r, dashed, "\overline{\partial^r_A}"] & \frac{B_{i+1-r}}{B_{i-r}} \arrow[r, dashed, "\overline{\partial^r_B}"] & \frac{C_{i+1-2r}}{C_{i-2r}} \arrow[r, dashed] & \cdots \\
\cdots \arrow[r, red] & \color{red}{\frac{\mathrm{dom}}{\mathrm{def}}} \arrow[u, dashed] \arrow[r, red] & \color{red}{\frac{\mathrm{dom}(\overline{\partial^r_B})}{\mathrm{def}(\overline{\partial^r_A})}} \arrow[u, dashed] \arrow[r, red] & \color{red}{\frac{\mathrm{dom}}{\mathrm{def}}} \arrow[u, dashed] \arrow[r, red] & \cdots
\end{tikzcd}$$

- These are the chain complexes on the r-th page of the associated **spectral sequence.**

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$\begin{tikzcd}
\cdots \arrow[r] & C_{j+1} \arrow[r] & C_j \arrow[r] & C_{j-1} \arrow[r] & \cdots \\
\cdots \arrow[r, dashed] & \frac{F_{i+1}C_{j+1}}{F_iC_{j+1}} \arrow[u, dashed] \arrow[r, dashed] & \frac{F_{i+1-r}C_j}{F_{i-r}C_j} \arrow[u, dashed] \arrow[r, dashed] & \frac{F_{i+1-2r}C_{j-1}}{F_{i-2r}C_{j-1}} \arrow[u, dashed] \arrow[r, dashed] & \cdots \\
\cdots \arrow[r] & E^r_{i+1,j-i} \arrow[u, dashed] \arrow[r] & E^r_{i+1-r,j-i+(r-1)} \arrow[u, dashed] \arrow[r] & E^r_{i+1-2r,j-i+2(r-1)} \arrow[u, dashed] \arrow[r] & \cdots
\end{tikzcd}$$

- These are the chain complexes on the r-th page of the associated **spectral sequence.**

Spectral sequences via generalized morphisms

For all $r \geq 0$, we get so-called generalized chain complexes.

$$
\begin{tikzcd}
\cdots \arrow[r] & C_{j+1} \arrow[r] & C_j \arrow[r] & C_{j-1} \arrow[r] & \cdots \\
\cdots \arrow[r, dashed] & \frac{F_{i+1}C_{j+1}}{F_iC_{j+1}} \arrow[r, dashed] \arrow[u, dashed] & \frac{F_{i+1-r}C_j}{F_{i-r}C_j} \arrow[r, dashed] \arrow[u, dashed] & \frac{F_{i+1-2r}C_{j-1}}{F_{i-2r}C_{j-1}} \arrow[r, dashed] \arrow[u, dashed] & \cdots \\
\cdots \arrow[r] & E^r_{i+1,j-i} \arrow[r] \arrow[u, dashed] & E^r_{i+1-r,j-i+(r-1)} \arrow[r] \arrow[u, dashed] & E^r_{i+1-2r,j-i+2(r-1)} \arrow[r] \arrow[u, dashed] & \cdots
\end{tikzcd}
$$

- These are the chain complexes on the r-th page of the associated **spectral sequence.**
- We just computed them **without a recursive strategy**.

1. Classical diagram chases
2. Additive relations
3. Generalized morphisms
4. Applications of generalized morphisms
   - An algorithm for spectral sequences
   - The purity filtration

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0C_\bullet$ be a finitely filtered complex

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0C_\bullet$ be a finitely filtered complex and for $p, q \in \mathbb{Z}$, $r \geq 0$ let E^r_{pq} be the **computed** objects

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0 C_\bullet$ be a finitely filtered complex and for $p, q \in \mathbb{Z}$, $r \geq 0$ let E^r_{pq} be the **computed** objects with their generalized embeddings

$$E^r_{pq} \hookrightarrow C_{p+q}.$$

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0 C_\bullet$ be a finitely filtered complex and for $p, q \in \mathbb{Z}$, $r \geq 0$ let E^r_{pq} be the **computed** objects with their generalized embeddings

$$E^r_{pq} \hookrightarrow C_{p+q}.$$

Then for all p, q and all $k \geq n+1$ we have

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0 C_\bullet$ be a finitely filtered complex and for $p, q \in \mathbb{Z}$, $r \geq 0$ let E^r_{pq} be the **computed** objects with their generalized embeddings

$$E^r_{pq} \hookrightarrow C_{p+q}.$$

Then for all p, q and all $k \geq n+1$ we have

$$E^k_{pq} \cong E^{n+1}_{pq}$$

Spectral sequences

Convergence

Let $C_\bullet := 0 = F_{-n-1}C_\bullet \leq F_{-n}C_\bullet \leq \cdots \leq F_0 C_\bullet$ be a finitely filtered complex and for $p, q \in \mathbb{Z}$, $r \geq 0$ let E^r_{pq} be the **computed** objects with their generalized embeddings

$$E^r_{pq} \hookrightarrow C_{p+q}.$$

Then for all p, q and all $k \geq n+1$ we have

$$E^k_{pq} \cong E^{n+1}_{pq} =: E^\infty_{pq}.$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology,

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$
\begin{tikzcd}
E^{\infty}_{p,q} \arrow[rr, dashed, bend right] & & \mathrm{H}_{p+q}\left(C_{\bullet}\right)
\end{tikzcd}
$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd} E^{\infty}_{p,q} \arrow[r, hook, dashed] \arrow[rr, bend right, dashed] & C_{p+q} & \mathrm{H}_{p+q}(C_{\bullet}) \end{tikzcd}$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd} E^{\infty}_{p,q} \arrow[r, hook, dashed] \arrow[rr, bend right, dashed] & C_{p+q} \arrow[r, two heads, dashed] & \mathrm{H}_{p+q}(C_\bullet) \end{tikzcd}$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd} E^{\infty}_{p,q} \arrow[r, hook, dashed] \arrow[rr, bend right, dashed] & C_{p+q} \arrow[r, two heads, dashed] & \mathrm{H}_{p+q}(C_{\bullet}) \end{tikzcd}$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd}
E^{\infty}_{p,q} & & \mathrm{H}_{p+q}(C_\bullet) \\
& X \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] &
\end{tikzcd}$$

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd} E^{\infty}_{p,q} & & \mathrm{H}_{p+q}(C_\bullet) \\ & X \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

This induces a filtration on $H := \mathrm{H}_{p+q}(C_\bullet)$ with

Filtration morphisms

Using the above generalized embedding and the generalized projection to the homology, we get a generalized morphism

$$\begin{tikzcd} E^{\infty}_{p,q} & & \mathrm{H}_{p+q}\left(C_\bullet\right) \\ & X \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

This induces a filtration on $H := \mathrm{H}_{p+q}\left(C_\bullet\right)$ with

$$\begin{aligned} F_pH/F_{p-1}H &\cong E^{\infty}_{p,q} \\ F_pH &\cong \mathrm{im}\left(\beta\right). \end{aligned}$$

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension.

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at 0.*

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at 0.*
2. *$\mathrm{H}_0(C_\bullet)$ is constructively isomorphic to M.*

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at 0.*
2. $\mathrm{H}_0\left(C_\bullet\right)$ *is constructively isomorphic to M.*
3. *The induced spectral sequence of $C_\bullet$ is the bidualizing spectral sequence,*

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at 0.*
2. $\mathrm{H}_0\left(C_\bullet\right)$ *is constructively isomorphic to M.*
3. *The induced spectral sequence of $C_\bullet$ is the bidualizing spectral sequence, i.e., we have*

$$E^2_{pq} = \mathrm{Ext}^{-p}(\mathrm{Ext}^{q}(M,S),S)) \Longrightarrow M \quad \textit{for } p+q=0,$$

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at* 0.
2. $\mathrm{H}_0\left(C_\bullet\right)$ *is constructively isomorphic to M.*
3. *The induced spectral sequence of $C_\bullet$ is the bidualizing spectral sequence, i.e., we have*

$$E^2_{pq} = \mathrm{Ext}^{-p}(\mathrm{Ext}^q(M,S),S)) \Longrightarrow M \qquad \textit{for } p+q=0,$$

   which yields the purity filtration of M,

The bidualizing spectral sequence

Theorem

Let M be a finitely presented module over a computable ring S of finite projective dimension. Then one can ***compute*** *a filtered complex $C_\bullet$ with the following properties:*

1. *$C_\bullet$ is exact everywhere except at 0.*
2. *$\mathrm{H}_0\left(C_\bullet\right)$ is constructively isomorphic to M.*
3. *The induced spectral sequence of $C_\bullet$ is the bidualizing spectral sequence, i.e., we have*

$$E^2_{pq} = \mathrm{Ext}^{-p}(\mathrm{Ext}^q(M,S),S)) \Longrightarrow M \quad \textit{for } p+q=0,$$

which yields the purity filtration of M, i.e., a finite filtration where all graded parts $F_{-i}M/F_{-(i+1)}M$ are pure of codimension i.

Filtration morphisms

Using all of the above, we now get the generalized morphism

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$
\begin{tikzcd}
E^{\infty}_{-p,p} \arrow[rrr, bend right, dashed] & & & M
\end{tikzcd}
$$

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$\begin{tikzcd} E^{\infty}_{-p,p} \arrow[r, hook, dashed] \arrow[rr, bend right, dashed] & C_0 & M \end{tikzcd}$$

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$
\begin{tikzcd}
E^{\infty}_{-p,p} \arrow[r, hook, dashed] \arrow[rrr, bend right, dashed] & C_0 \arrow[r, two heads, dashed] & \mathrm{H}_0\left(C_\bullet\right) & M
\end{tikzcd}
$$

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$
\begin{tikzcd}
E^{\infty}_{-p,p} \arrow[r, hook, dashed] \arrow[rrr, bend right, dashed] & C_0 \arrow[r, two heads, dashed] & \mathrm{H}_0\left(C_\bullet\right) \arrow[r, "\simeq"] & M
\end{tikzcd}
$$

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$
\begin{tikzcd}
E^{\infty}_{-p,p} & & M \\
& X \arrow[ul, "\alpha"'] \arrow[ur, "\beta"] &
\end{tikzcd}
$$

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$\begin{tikzcd}
E^{\infty}_{-p,p} & & M \\
& X \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] &
\end{tikzcd}$$

For the purity filtration of M, we have

Filtration morphisms

Using all of the above, we now get the generalized morphism

$$\begin{tikzcd} E^{\infty}_{-p,p} & & M \\ & X \arrow[ul, "\alpha"] \arrow[ur, "\beta"'] & \end{tikzcd}$$

For the purity filtration of M, we have

$$\begin{aligned} F_{-p}M/F_{-p-1}M &\cong E^{\infty}_{-p,p} \\ F_{-p}M &\cong \operatorname{im}(\beta)\,. \end{aligned}$$

Presentations from filtrations

Let $F_{-n}M \leq F_{-n+1}M \leq \cdots \leq F_0M := M$ be a finitely presented filtered module.

Presentations from filtrations

Let $F_{-n}M \leq F_{-n+1}M \leq \cdots \leq F_0M := M$ be a finitely presented filtered module.

If $\mathtt{M}_i$ is a presentation matrix for $F_iM/F_{i-1}M$,

Presentations from filtrations

Let $F_{-n}M \leq F_{-n+1}M \leq \cdots \leq F_0 M := M$ be a finitely presented filtered module.

If $\mathtt{M}_i$ is a presentation matrix for $F_i M / F_{i-1} M$, then M can be presented by an upper block triangular matrix

$$\begin{pmatrix} \mathtt{M}_0 & * & \dots & \dots & * \\ & \mathtt{M}_{-1} & * & \dots & * \\ & & \ddots & \ddots & \vdots \\ & & & \mathtt{M}_{-n+1} & * \\ & & & & \mathtt{M}_{-n} \end{pmatrix}.$$

Example: filtered presentation

Consider the module with relations

$$\begin{pmatrix} 0 & 0 & 0 & 0 & xz & -z^2 \\ 0 & 0 & 0 & 0 & xy & -yz \\ 0 & -x^2z+xyz+xz^2 & y^2z & -xz+yz & x-y & 0 \\ 0 & 0 & 0 & 0 & x^2 & -xz \\ -xy & -x^3+x^2y+x^2z & xy^2 & -x^2+xy & 0 & x-y \\ z & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Example: filtered presentation

Consider the module with relations

$$\begin{pmatrix} 0 & 0 & 0 & 0 & xz & -z^2 \\ 0 & 0 & 0 & 0 & xy & -yz \\ 0 & -x^2z+xyz+xz^2 & y^2z & -xz+yz & x-y & 0 \\ 0 & 0 & 0 & 0 & x^2 & -xz \\ -xy & -x^3+x^2y+x^2z & xy^2 & -x^2+xy & 0 & x-y \\ z & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Computing the purity filtration by using the bidualizing spectral sequence yields

$$\left(\begin{array}{ccccc|c|c} x & -z & 0 & 0 & 0 & 0 & 1 \\ -y & z & y^2z & -yz^2 & -xz+yz & 0 & -1 \\ 0 & x-y & xy^2 & -xyz & -x^2+xy & xy & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & z & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & z \\ 0 & 0 & 0 & 0 & 0 & 0 & y \\ 0 & 0 & 0 & 0 & 0 & 0 & x \end{array}\right)$$

References I

- Mohamed Barakat and Markus Lange-Hegermann, *An axiomatic setup for algorithmic homological algebra and an alternative approach to localization*, J. Algebra Appl. **10** (2011), no. 2, 269–293, (`arXiv:1003.1943`). MR 2795737 (2012f:18022)
- Mohamed Barakat and Markus Lange-Hegermann, *Gabriel morphisms and the computability of Serre quotients with applications to coherent sheaves*, (`arXiv:1409.2028`), 2014.
- Sebastian Gutsche, *Constructive category theory with applications to algebraic geometry*, Ph.D. thesis, University of Siegen, 2017.

References II

Gutsche, Sebastian, Skartsæterhagen, Øystein, and Posur, Sebastian, *The* CAP *project – Categories, Algorithms, and Programming*, (`http://homalg-project.github.io/CAP_project`), 2013–2017.

Peter Hilton, *Correspondences and exact squares*, Proc. Conf. Categorical Algebra (La Jolla, Calif., 1965), Springer, New York, 1966, pp. 254–271. MR 0204487

Peter T. Johnstone, *Sketches of an elephant: a topos theory compendium. Vol. 1*, Oxford Logic Guides, vol. 43, The Clarendon Press, Oxford University Press, New York, 2002. MR 1953060 (2003k:18005)

References III

Sebastian Posur, *Constructive category theory and applications to equivariant sheaves*, Ph.D. thesis, University of Siegen, 2017, `http://dokumentix.ub.uni-siegen.de/opus/volltexte/2017/1179/`.

Dieter Puppe, *Korrespondenzen in abelschen Kategorien*, Math. Ann. **148** (1962), 1–30. MR 0141698