The periodic table of $n$-categories

Eugenia Cheng

University of Sheffield
7 January 2009
Plan

1. Degeneracy
Plan

1. Degeneracy

2. The Eckmann-Hilton argument
Plan

1. Degeneracy
2. The Eckmann-Hilton argument
3. Inconvenient elements
Plan

1. Degeneracy
2. The Eckmann-Hilton argument
3. Inconvenient elements
4. Higher maps
Plan

1. Degeneracy
2. The Eckmann-Hilton argument
3. Inconvenient elements
4. Higher maps
5. Stabilisation
Plan

1. Degeneracy
2. The Eckmann-Hilton argument
3. Inconvenient elements
4. Higher maps
5. Stabilisation
6. Other reasons to care
References

- J. Baez and J. Dolan. Higher-dimensional algebra and topological quantum field theory.
References

References


References


References


• E. Cheng and N. Gurski. The periodic table of $n$-categories for low dimensions II: degenerate tricategories.


1. Degeneracy

Definition
1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

• only one 0-cell
• only one 1-cell
• only one 2-cell
... 
• only one $(k - 1)$-cell

So the first non-trivial dimension is $k$. 
1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell

1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell
- only one 1-cell
1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell
- only one 1-cell
- only one 2-cell
- only one $(k-1)$-cell

So the first non-trivial dimension is $k$. 
Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell
- only one 1-cell
- only one 2-cell
- ... only one $(k-1)$-cell

So the first non-trivial dimension is $k$. 
1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell
- only one 1-cell
- only one 2-cell
- \ldots
- only one $(k - 1)$-cell
1. Degeneracy

Definition

A $k$-degenerate $n$-category is an $n$-category with:

- only one 0-cell
- only one 1-cell
- only one 2-cell
- ... only one $(k-1)$-cell

So the first non-trivial dimension is $k$. 
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

```
"old" $n$-category  \longrightarrow  "new" $(n - k)$-category
```

5.
Dimension-shift for $k$-degenerate $n$-categories

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```

```
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

```
"old"  "new"
n-category  (n − k)-category

0-cells
1-cells
```
1. Degeneracy

Dimension-shift for \( k \)-degenerate \( n \)-categories

"old"
\( n \)-category

0-cells
1-cells
\vdots
\( (k - 1) \)-cells

\( \Rightarrow \)

"new"
\( (n - k) \)-category
1. Degeneracy

Dimension-shift for \( k \)-degenerate \( n \)-categories

```
<table>
<thead>
<tr>
<th>“old”</th>
<th>“new”</th>
</tr>
</thead>
<tbody>
<tr>
<td>( n )-category</td>
<td>( (n - k) )-category</td>
</tr>
<tr>
<td>0-cells</td>
<td></td>
</tr>
<tr>
<td>1-cells</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>( (k - 1) ) -cells</td>
<td>trivial</td>
</tr>
</tbody>
</table>
```
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

```
“old”
n-category

0-cells
1-cells

\vdots

(k - 1)-cells

\vdots

k-cells

\mapsto

“new”
(n - k)-category

0-cells
```
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

```
"old"  "new"
n-category  (n - k)-category

0-cells
1-cells

\vdots

(k - 1)-cells

\{\text{trivial}\}

k-cells  0-cells

(k + 1)-cells  1-cells
```
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

\[
\begin{array}{c}
\text{“old”} \\
\text{$n$-category} \\
\text{0-cells} \\
\text{1-cells} \\
\vdots \\
\text{$(k-1)$-cells} \\
\text{$k$-cells} \\
\text{$(k+1)$-cells} \\
\vdots
\end{array} \quad \xrightarrow{\text{trivial}} \quad \begin{array}{c}
\text{“new”} \\
\text{$(n-k)$-category} \\
\text{0-cells} \\
\text{1-cells} \\
\vdots
\end{array}
\]
1. Degeneracy

Dimension-shift for $k$-degenerate $n$-categories

```
"old"  "new"
n-category (n - k)-category

0-cells   0-cells
1-cells   1-cells

\vdots

(k - 1)-cells \quad \vdots

k-cells   \quad \vdots

(k + 1)-cells

\vdots

n-cells   \quad \vdots

\downarrow

\text{trivial}
```
1. Degeneracy

Degenerate categories

A category with only one object is a monoid.
1. Degeneracy

Degenerate categories

A category with only one object is a monoid.
1. Degeneracy

Degenerate categories

A category with only one object is a monoid.

“old” category “new” monoid
1. Degeneracy

Degenerate categories
A category with only one object is a monoid.

“old” category          “new” monoid
objects – trivial
1. Degeneracy

Degenerate categories

A category with only one object is a monoid.

“old” category                          “new” monoid

objects – trivial

morfisms                                      objects
Degenerate categories

A category with only one object is a monoid.

“old” category

“new” monoid

objects – trivial

morphisms \[\rightarrow\] objects

composition \[\rightarrow\] multiplication
1. Degeneracy

Degenerate categories
A category with only one object is a monoid.

```
"old" category         "new" monoid

objects - trivial
morfisms → objects
composition → multiplication
identity → unit
```
1. Degeneracy

Degenerate bicategories
Degenerate bicategories

A bicategory with only one object is a monoidal category.
1. Degeneracy

“old” bicategory \rightarrow “new” monoidal category
1. Degeneracy

\begin{itemize}
  \item “old” bicategory
  \item monoidal category
  \item “new” monoidal category
  \item 0-cells – trivial
\end{itemize}
1. Degeneracy

“old” bicategory

0-cells – trivial

1-cells

“new” monoidal category

objects
1. Degeneracy

“old” bicategory

0-cells – trivial

1-cells

2-cells

“new” monoidal category

→ objects

→ morphisms
1. Degeneracy

“old” bicategory

0-cells – trivial
1-cells
2-cells
composition

“new” monoidal category

composition of 1-cells
composition of objects
composition of morphisms
1. Degeneracy

“old” bicategory

0-cells – trivial
1-cells
2-cells
composition of 1-cells

“new” monoidal category

→ objects
→ morphisms
→ □ of objects
1. Degeneracy

<table>
<thead>
<tr>
<th>“old” bicategory</th>
<th>“new” monoidal category</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-cells – trivial</td>
<td>objects</td>
</tr>
<tr>
<td>1-cells</td>
<td>morphisms</td>
</tr>
<tr>
<td>2-cells</td>
<td>⊗ of objects</td>
</tr>
<tr>
<td>composition</td>
<td>⊗ of morphisms</td>
</tr>
<tr>
<td>of 1-cells</td>
<td></td>
</tr>
<tr>
<td>of 2-cells</td>
<td></td>
</tr>
</tbody>
</table>
1. Degeneracy

“old” bicategory \[\leadsto\] “new” monoidal category

0-cells – trivial
1-cells \[\leadsto\] objects
2-cells \[\leadsto\] morphisms

composition
- of 1-cells \[\leadsto\] \(\otimes\) of objects
- of 2-cells \(\begin{array}{c}\circ\bullet\circ\end{array}\) \[\leadsto\] \(\otimes\) of morphisms
- of 2-cells \(\begin{array}{c}\circ\end{array}\) \[\leadsto\] composition of morphisms
1. Degeneracy

In a $k$-degenerate $n$-category:
1. Degeneracy

In a $k$-degenerate $n$-category:

composition of $k$-cells $\quad \rightarrow \quad \otimes$
1. Degeneracy

In a $k$-degenerate $n$-category:

composition of $k$-cells $\rightarrow$ $\otimes$

$k$-different types
1. Degeneracy

In a $k$-degenerate $n$-category:

composition of $k$-cells $\xrightarrow{\otimes}$$k$-different types

$k$-different types
1. Degeneracy

Example: 2-degenerate
1. Degeneracy

Example: 2-degenerate

“old” \[\rightarrow\] “new”
1. Degeneracy

Example: 2-degenerate

"old" \[\rightarrow\] "new"

\[
\begin{align*}
0\text{-cells} & \quad \text{trivial} \\
1\text{-cells} & \quad \text{trivial}
\end{align*}
\]
1. Degeneracy

Example: 2-degenerate

```
“old” → “new”

0-cells \{ trivial

1-cells

2-cells \[ \rightarrow \]

⊕

→

0-cells
```
1. Degeneracy

Example: 2-degenerate

```
"old"  \rightarrow  "new"
```

```
{0-cells, 1-cells} \rightarrow \text{trivial}
```

```
2-cells  \rightarrow  0-cells
```

composition
1. Degeneracy

Example: 2-degenerate

```
“old”        →        “new”
```

```
0-cells \{ trivial
1-cells \}
```

```
2-cells ◯ 0-cells
```

```
composition
```

```
⊗1 0
```

Example: 2-degenerate

```
“old” → “new”
```

- 0-cells
- 1-cells
- 2-cells

Composition

```
⊗0
⊗1
```
1. Degeneracy

Example: 2-degenerate

"old" \rightarrow "new"

0-cells \{ trivial
1-cells \}

2-cells \rightarrow 0-cells

composition

⊗_0
⊗_1

?
1. Degeneracy

Example: 3-degenerate

\[\text{3-cells} \rightarrow \text{0-cells}\]
1. Degeneracy

Example: 3-degenerate

3-cells \rightarrow 0-cells

composition
1. Degeneracy

Example: 3-degenerate

3-cells \rightarrow 0-cells

composition

\otimes_0
1. Degeneracy

Example: 3-degenerate

3-cells \( \Rightarrow \) \( \rightarrow \) 0-cells

composition

\( \otimes_0 \)

\( \otimes_1 \)
1. Degeneracy

Example: 3-degenerate

3-cells \rightleftharpoons \rightarrow 0\text{-cells}

composition

0 \circ 0 \rightarrow 0_0

0 \circ 1 \rightarrow 0_1

0 \circ 2 \rightarrow 0_2
1. Degeneracy

**Example: 3-degenerate**

3-cells $\Rightarrow$ 0-cells

**composition**

$\otimes_0$ $\otimes_1$ $\otimes_2$
1. Degeneracy

We have coherence cells for interchange:

\[(a \otimes_0 b) \otimes_1 (c \otimes_0 d) \sim (a \otimes_1 c) \otimes_0 (b \otimes_1 d).\]
1. Degeneracy

We have coherence cells for interchange:

\[
\begin{array}{c}
\Downarrow \quad \Downarrow \\
\Rightarrow \quad \Rightarrow \\
\Downarrow \quad \Downarrow \\
\end{array} \quad \Rightarrow \\
\begin{array}{c}
\Downarrow \quad \Downarrow \\
\Rightarrow \quad \Rightarrow \\
\Downarrow \quad \Downarrow \\
\end{array} \quad \Rightarrow \\
\begin{array}{c}
\Downarrow \quad \Downarrow \\
\Rightarrow \quad \Rightarrow \\
\Downarrow \quad \Downarrow \\
\end{array}
\]

becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) \sim (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]

We have analogous coherence cells for all dimensions, with axioms.

This is called \(k\)-tuply monoidal.
1. Degeneracy

We have coherence cells for interchange:

We have analogous coherence cells for all dimensions, with axioms. This is called \( k \)-tuply monoidal.
1. Degeneracy

We have coherence cells for interchange:

\[
\begin{array}{c}
\rotatebox{90}{$\Downarrow$} \quad \rotatebox{90}{$\Downarrow$} \\
\rightarrow \quad \rightarrow \\
\rotatebox{90}{$\Downarrow$} \quad \rotatebox{90}{$\Downarrow$}
\end{array}
\quad \cong 
\begin{array}{c}
\rotatebox{90}{$\Downarrow$} \quad \rotatebox{90}{$\Downarrow$} \\
\rightarrow \quad \rightarrow \\
\rotatebox{90}{$\Downarrow$} \quad \rotatebox{90}{$\Downarrow$}
\end{array}
\]

becomes

\[(a \otimes_0 b) \otimes_1 (c \otimes_0 d) \cong (a \otimes_1 c) \otimes_0 (b \otimes_1 d).\]
1. Degeneracy

We have coherence cells for interchange:

\[
\begin{array}{c}
\cdot \quad \cdot \quad \cdot \quad \cdot \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\end{array}
\quad \cong \quad
\begin{array}{c}
\cdot \quad \cdot \quad \cdot \quad \cdot \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\downarrow \quad \rightarrow \quad \downarrow \quad \rightarrow \\
\end{array}
\]

becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) \cong (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]

We have analogous coherence cells for all dimensions, with axioms.
1. Degeneracy

We have coherence cells for interchange:

\[
\begin{array}{c}
\begin{array}{c}
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\end{array}
\quad \Rightarrow \\
\begin{array}{c}
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\downarrow \\
\cdot \\
\end{array}
\end{array}
\]

becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) \cong (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]

We have analogous coherence cells for all dimensions, with axioms.

This is called \textit{k-tuply monoidal}.
1. Degeneracy

Slogan:
1. Degeneracy

Slogan:

$k$-degenerate $n$-categories

“are”

$k$-tuply monoidal $(n - k)$-categories.
1. Degeneracy

**Slogan:**

\[ k \text{-degenerate } n \text{-categories} \]

“are”

\[ k \text{-tuply monoidal } (n - k) \text{-categories}. \]

—but this is only part of the point.
1. Degeneracy
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
</tr>
</thead>
<tbody>
<tr>
<td>only one object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>mon.</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>only one object</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
</tr>
</thead>
<tbody>
<tr>
<td>only one object</td>
<td></td>
</tr>
</tbody>
</table>

**monoid**

≡ category with only one object
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>category with only one object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Degeneracy: set
- Only one object
- Category with only one object
- 2-category
- Symmetric monoid
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
<th>symmetric mon.</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>\equiv category with</td>
<td>only one object</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>symmetric mon.</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>2-category with only one object</td>
<td>symmetric mon.</td>
</tr>
</tbody>
</table>

- A monoid is a category with only one object.
- A monoidal category is a 2-category with only one object.
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>category with only one object</td>
<td>monoidal cat.</td>
</tr>
<tr>
<td>≡</td>
<td></td>
<td>≡</td>
</tr>
</tbody>
</table>

A monoid is equivalent to a category with only one object.

A monoidal category is equivalent to a 2-category with only one object.
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>only one object</td>
<td>only one object</td>
<td></td>
</tr>
<tr>
<td>symmetric mon.</td>
<td>symmetric mon.</td>
<td></td>
</tr>
<tr>
<td>2-category</td>
<td>≡ category with only one object</td>
<td></td>
</tr>
<tr>
<td>monoid</td>
<td>≡ category with only one object</td>
<td></td>
</tr>
<tr>
<td>≡ 2-category with only one object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>commutative monoid</td>
<td>≡ 2-cat. with only one 1-cell</td>
<td></td>
</tr>
</tbody>
</table>

### Notes
- Degeneracy refers to a situation where a mathematical structure is simplified or reduced, often by removing dimensions or properties.
- In this context, we see how monoids (with only one object) degenerate into monoidal categories (with only one 1-cell) in the 2-category setting.
- Commutative monoids, which are monoids with an additional commutativity condition, also degenerate appropriately in these contexts.

---

14.
# 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\equiv) category with only one object</td>
<td>(\equiv) 2-category with only one object</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>commutative monoid</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\equiv) 2-cat. with only one 1-cell</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**commutative monoid**

≡ 2-cat. with only one 1-cell
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid (\equiv) category with only one object</td>
<td>monoidal cat. (\equiv) 2-category with only one object</td>
<td>monoidal 2-cat. (\equiv) 3-category with only one object</td>
<td></td>
</tr>
</tbody>
</table>

**Commutative monoid**
\(\equiv\) 2-cat. with only one 1-cell
# 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>$\equiv$ category with only one object</td>
<td>$\equiv$ 2-category with only one object</td>
<td>$\equiv$ 3-category with only one object</td>
<td></td>
</tr>
<tr>
<td>commutative monoid</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\equiv$ 2-cat. with only one 1-cell</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>monoid</strong></td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>3-category with only one object</td>
</tr>
<tr>
<td>$\equiv$ category with only one object</td>
<td>$\equiv$ 2-category with only one object</td>
<td>$\equiv$ 3-category with only one object</td>
<td></td>
</tr>
<tr>
<td><strong>commutative monoid</strong></td>
<td>braided mon. category</td>
<td>2-cat. with only one 1-cell</td>
<td>3-cat. with only one 1-cell</td>
</tr>
</tbody>
</table>

---
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 2-cat.</td>
</tr>
<tr>
<td>$\equiv$ category with</td>
<td>$\equiv$ 2-category with</td>
<td>$\equiv$ 3-category with</td>
<td></td>
</tr>
<tr>
<td>only one object</td>
<td>only one object</td>
<td>only one object</td>
<td></td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\equiv$ 2-cat. with</td>
<td>$\equiv$ 3-cat. with</td>
<td></td>
<td></td>
</tr>
<tr>
<td>only one 1-cell</td>
<td>only one 1-cell</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td></td>
</tr>
</tbody>
</table>

**commutative monoid**

<table>
<thead>
<tr>
<th></th>
<th>braided mon. category</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2-cat. with only one 1-cell</td>
<td>3-cat. with only one 1-cell</td>
<td>3-cat. with only one 2-cell</td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>$\equiv$ category with only one object</td>
<td>$\equiv$ 2-category with only one object</td>
<td>$\equiv$ 3-category with only one object</td>
<td></td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\equiv$ 2-cat. with only one 1-cell</td>
<td>$\equiv$ 3-cat. with only one 1-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>$\equiv$ 3-cat. with only one 2-cell</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
<td>&quot;&quot;</td>
</tr>
<tr>
<td>$\equiv$ 4-cat. with only one 3-cell</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td></td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td></td>
</tr>
<tr>
<td><strong>commutative monoid</strong></td>
<td>braided mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>**</td>
<td>symmetric mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td>≡ 3-cat. with only one 2-cell</td>
<td>≡ 4-cat. with only one 2-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>**</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>≡ 4-cat. with only one 3-cell</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>**</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td>≡ 4-category with only one object</td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td></td>
</tr>
<tr>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td>≡ 4-cat. with only one 1-cell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>symmetric mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>≡ 3-cat. with only one 2-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>≡ 4-cat. with only one 3-cell</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ \vdots \]
## 1. Degeneracy

<table>
<thead>
<tr>
<th>Set</th>
<th>Category</th>
<th>2-Category</th>
<th>3-Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td>≡ 4-category with only one object</td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td>braided mon. 3-category</td>
</tr>
<tr>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td>≡ 4-cat. with only one 1-cell</td>
<td></td>
</tr>
<tr>
<td></td>
<td>symmetric mon. category</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>≡ 3-cat. with only one 2-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>≡ 4-cat. with only one 3-cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>(\equiv) category with only one object</td>
<td>(\equiv) 2-category with only one object</td>
<td>(\equiv) 3-category with only one object</td>
<td>(\equiv) 4-category with only one object</td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td>braided mon. 3-category</td>
</tr>
<tr>
<td>(\equiv) 2-cat. with only one 1-cell</td>
<td>(\equiv) 3-cat. with only one 1-cell</td>
<td>(\equiv) 4-cat. with only one 1-cell</td>
<td>(\equiv) 5-cat. with only one 1-cell</td>
</tr>
<tr>
<td>(\equiv) 3-cat. with only one 2-cell</td>
<td>(\equiv) 4-cat. with only one 2-cell</td>
<td>(\equiv) 5-cat. with only one 2-cell</td>
<td>(\equiv) 6-cat. with only one 2-cell</td>
</tr>
</tbody>
</table>

...
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td>\equiv category with only one object</td>
<td>\equiv 2-category with only one object</td>
<td>\equiv 3-category with only one object</td>
<td>\equiv 4-category with only one object</td>
</tr>
</tbody>
</table>

**commutative monoid**

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>\equiv 2-cat. with only one 1-cell</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td>braided mon. 3-category</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>\equiv 3-cat. with only one 1-cell</td>
<td>symmetric mon. category</td>
<td>symmetric mon. 2-category</td>
<td>symmetric mon. 3-category</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>\equiv 4-cat. with only one 2-cell</td>
<td>sylleptic mon. 2-category</td>
<td>sylleptic mon. 3-category</td>
<td>sylleptic mon. 4-category</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>\equiv 5-cat. with only one 2-cell</td>
<td>\equiv 5-cat. with only one 3-cell</td>
<td>\equiv 5-cat. with only one 3-cell</td>
<td>\equiv 5-cat. with only one 3-cell</td>
</tr>
</tbody>
</table>
## 1. Degeneracy

<table>
<thead>
<tr>
<th>Set</th>
<th>Category</th>
<th>2-Category</th>
<th>3-Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monoid</td>
<td>Monoidal Cat.</td>
<td>Monoidal 2-Cat.</td>
<td>Monoidal 3-Cat.</td>
</tr>
<tr>
<td>≡ Category with only one object</td>
<td>≡ 2-Category with only one object</td>
<td>≡ 3-Category with only one object</td>
<td>≡ 4-Category with only one object</td>
</tr>
<tr>
<td>Commutative Monoid</td>
<td>Braided Mon. Category</td>
<td>Braided Mon. 2-Category</td>
<td>Braided Mon. 3-Category</td>
</tr>
<tr>
<td>≡ 2-Cat. with only one 1-cell</td>
<td>≡ 3-Cat. with only one 1-cell</td>
<td>≡ 4-Cat. with only one 1-cell</td>
<td>≡ 5-Cat. with only one 1-cell</td>
</tr>
<tr>
<td></td>
<td>Symmetric Mon. Category</td>
<td>Sylleptic Mon. 2-Category</td>
<td>Symmetric Mon. 2-Category</td>
</tr>
<tr>
<td></td>
<td>≡ 3-Cat. with only one 2-cell</td>
<td>≡ 4-Cat. with only one 2-cell</td>
<td>≡ 5-Cat. with only one 2-cell</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
1. Degeneracy

<table>
<thead>
<tr>
<th>Set</th>
<th>Category</th>
<th>2-Category</th>
<th>3-Category</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Monoid</strong></td>
<td>≡ Category with only one object</td>
<td>≡ 2-Category with only one object</td>
<td>≡ 3-Category with only one object</td>
</tr>
<tr>
<td><strong>Commutative Monoid</strong></td>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td>≡ 4-cat. with only one 1-cell</td>
</tr>
<tr>
<td><strong>Symmetric Monoid</strong></td>
<td>≡ 3-cat. with only one 2-cell</td>
<td>≡ 4-cat. with only one 2-cell</td>
<td>≡ 5-cat. with only one 2-cell</td>
</tr>
<tr>
<td><strong>Sylleptic Monoid</strong></td>
<td>≡ 4-cat. with only one 3-cell</td>
<td>≡ 5-cat. with only one 3-cell</td>
<td>≡ 6-cat. with only one 3-cell</td>
</tr>
</tbody>
</table>

...
## 1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
</tr>
<tr>
<td><code>≡</code> category with</td>
<td><code>≡</code> 2-category with</td>
<td><code>≡</code> 3-category with</td>
<td><code>≡</code> 4-category with</td>
</tr>
<tr>
<td>only one object</td>
<td>only one object</td>
<td>only one object</td>
<td>only one object</td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td>braided mon. 3-category</td>
</tr>
<tr>
<td><code>≡</code> 2-cat. with</td>
<td><code>≡</code> 3-cat. with</td>
<td><code>≡</code> 4-cat. with</td>
<td><code>≡</code> 5-cat. with</td>
</tr>
<tr>
<td>only one 1-cell</td>
<td>only one 1-cell</td>
<td>only one 1-cell</td>
<td>only one 1-cell</td>
</tr>
<tr>
<td><code>≡</code> 3-cat. with</td>
<td><code>≡</code> 4-cat. with</td>
<td><code>≡</code> 5-cat. with</td>
<td><code>≡</code> 6-cat. with</td>
</tr>
<tr>
<td>only one 2-cell</td>
<td>only one 2-cell</td>
<td>only one 2-cell</td>
<td>only one 2-cell</td>
</tr>
<tr>
<td><code>≡</code> 4-cat. with</td>
<td><code>≡</code> 5-cat. with</td>
<td><code>≡</code> 6-cat. with</td>
<td><code>≡</code> 7-cat. with</td>
</tr>
<tr>
<td>only one 3-cell</td>
<td>only one 3-cell</td>
<td>only one 3-cell</td>
<td>only one 3-cell</td>
</tr>
<tr>
<td><code>≡</code> 5-cat. with</td>
<td><code>≡</code> 6-cat. with</td>
<td><code>≡</code> 7-cat. with</td>
<td></td>
</tr>
<tr>
<td>only one 2-cell</td>
<td>only one 3-cell</td>
<td>only one 3-cell</td>
<td></td>
</tr>
<tr>
<td><code>≡</code> 6-cat. with</td>
<td><code>≡</code> 7-cat. with</td>
<td></td>
<td></td>
</tr>
<tr>
<td>only one 3-cell</td>
<td>only one 3-cell</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*...*
1. Degeneracy

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
<th>⋮</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
<td>⋮</td>
</tr>
<tr>
<td>≡ category with</td>
<td>≡ 2-category with</td>
<td>≡ 3-category with</td>
<td>≡ 4-category with</td>
<td>⋮</td>
</tr>
<tr>
<td>only one object</td>
<td>only one object</td>
<td>only one object</td>
<td>only one object</td>
<td>⋮</td>
</tr>
</tbody>
</table>

| commutative monoid           | braided mon. category             | braided mon. 2-category         | braided mon. 3-category         | ⋮ |
| ≡ 2-cat. with               | ≡ 3-cat. with                     | ≡ 4-cat. with                   | ≡ 5-cat. with                   | ⋮ |
| only one 1-cell             | only one 1-cell                   | only one 1-cell                 | only one 1-cell                 | ⋮ |

| symmetric mon. category      | symmetric mon. 2-category         | symmetric mon. 3-category       | symmetric mon. 3-category       | ⋮ |
| ≡ 3-cat. with               | ≡ 4-cat. with                     | ≡ 5-cat. with                   | ≡ 6-cat. with                   | ⋮ |
| only one 2-cell             | only one 2-cell                   | only one 2-cell                 | only one 2-cell                 | ⋮ |

| sylleptic mon. category     | sylleptic mon. 2-category         | sylleptic mon. 3-category       | sylleptic mon. 3-category       | ⋮ |
| ≡ 4-cat. with               | ≡ 5-cat. with                     | ≡ 6-cat. with                   | ≡ 7-cat. with                   | ⋮ |
| only one 3-cell             | only one 3-cell                   | only one 3-cell                 | only one 3-cell                 | ⋮ |

| *** mon. category           | *** mon. 2-category               | *** mon. 3-category             | *** mon. 3-category             | ⋮ |
| ≡ 5-cat. with               | ≡ 6-cat. with                     | ≡ 7-cat. with                   | ≡ 8-cat. with                   | ⋮ |
| only one 3-cell             | only one 3-cell                   | only one 3-cell                 | only one 3-cell                 | ⋮ |
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $\ast$ and $\circ$ such that

1. $\ast$ and $\circ$ are unital with the same unit
2. $\ast$ and $\circ$ distribute over each other, i.e.,
   $$\forall a,b,c,d \in A \quad (a \ast b) \circ (c \ast d) = (a \circ c) \ast (b \circ d).$$

Then $\ast$ and $\circ$ are in fact equal and this operation is commutative.
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $\ast$ and $\circ$ such that
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $\ast$ and $\circ$ such that

1. $\ast$ and $\circ$ are unital with the same unit
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $\ast$ and $\circ$ such that

1. $\ast$ and $\circ$ are unital with the same unit
2. $\ast$ and $\circ$ distribute over each other
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $\ast$ and $\circ$ such that

1. $\ast$ and $\circ$ are unital with the same unit

2. $\ast$ and $\circ$ distribute over each other
   
   i.e. $\forall a, b, c, d \in A$
2. The Eckmann-Hilton argument

Let $A$ be a set with two binary operations $*$ and $\circ$ such that

1. $*$ and $\circ$ are unital with the same unit

2. $*$ and $\circ$ distribute over each other
   i.e. $\forall a, b, c, d \in A$
   $$(a * b) \circ (c * d) = (a \circ c) * (b \circ d).$$

Then $*$ and $\circ$ are in fact equal and this operation is commutative.
1. Degeneracy

In a bicategory we have the interchange laws:

\[(a \otimes 0 b) \otimes 1 (c \otimes 0 d) = (a \otimes 1 c) \otimes 0 (b \otimes 1 d)\]

This is exactly the condition needed for the Eckmann-Hilton argument:

\[(a \ast b) \circ (c \ast d) = (a \circ c) \ast (b \circ d)\]
1. Degeneracy

In a bicategory we have the interchange laws:

\[
\begin{array}{c}
\begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array} & \begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array} \\
= \begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array} & \begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array} \\
\begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array} & \begin{array}{cc}
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\Downarrow & \Downarrow \\
\end{array}
\end{array}
\]

This becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) = (a \otimes_1 c) \otimes_0 (b \otimes_1 d).\]

This is exactly the condition needed for the Eckmann-Hilton argument

\[
(a \ast b) \circ (c \ast d) = (a \circ c) \ast (b \circ d).\]
1. Degeneracy

In a bicategory we have the interchange laws:

\[
\begin{array}{c}
\begin{array}{c}
\downarrow \\
\rightarrow \\
\downarrow \\
\rightarrow \\
\end{array}
\end{array}
= \begin{array}{c}
\begin{array}{c}
\downarrow \\
\rightarrow \\
\downarrow \\
\rightarrow \\
\end{array}
\end{array}
\]

becomes

\[
(a \otimes 0 b) \otimes 1 (c \otimes 0 d) = (a \otimes 1 c) \otimes 0 (b \otimes 1 d).
\]

This is exactly the condition needed for the Eckmann-Hilton argument:

\[(a \ast b) \circ (c \ast d) = (a \circ c) \ast (b \circ d).\]
1. Degeneracy

In a bicategory we have the interchange laws:

\[
\begin{array}{c}
\begin{array}{ccc}
& \downarrow & \\
\downarrow & & \downarrow \\
\rightarrow & & \rightarrow \\
\downarrow & & \downarrow \\
& \downarrow & \\
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{ccc}
& \downarrow & \\
\downarrow & & \downarrow \\
\rightarrow & & \rightarrow \\
\downarrow & & \downarrow \\
& \downarrow & \\
\end{array}
\end{array}
= \begin{array}{c}
\begin{array}{ccc}
& \downarrow & \\
\downarrow & & \downarrow \\
\rightarrow & & \rightarrow \\
\downarrow & & \downarrow \\
& \downarrow & \\
\end{array}
\end{array}
\begin{array}{c}
\begin{array}{ccc}
& \downarrow & \\
\downarrow & & \downarrow \\
\rightarrow & & \rightarrow \\
\downarrow & & \downarrow \\
& \downarrow & \\
\end{array}
\end{array}
\]

becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) = (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]
1. Degeneracy

In a bicategory we have the interchange laws:

\[
\begin{array}{c}
\begin{array}{c}
\downarrow \\
\rightarrow \\
\rightarrow \\
\end{array}
\end{array}
\quad = 
\begin{array}{c}
\begin{array}{c}
\downarrow \\
\rightarrow \\
\rightarrow \\
\end{array}
\end{array}
\]

becomes

\[
(a \otimes_0 b) \otimes_1 (c \otimes_0 d) = (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]

This is exactly the condition needed for the Eckmann-Hilton argument.
1. Degeneracy

In a bicategory we have the interchange laws:

\[
\begin{array}{ccc}
\downarrow & \downarrow & \downarrow \\
\downarrow & \downarrow & \downarrow \\
\downarrow & \downarrow & \downarrow \\
\end{array}
\quad = 
\begin{array}{ccc}
\downarrow & \downarrow & \downarrow \\
\downarrow & \downarrow & \downarrow \\
\downarrow & \downarrow & \downarrow \\
\end{array}
\]

becomes

\[(a \otimes_0 b) \otimes_1 (c \otimes_0 d) = (a \otimes_1 c) \otimes_0 (b \otimes_1 d).\]

This is exactly the condition needed for the Eckmann-Hilton argument

\[(a * b) \circ (c * d) = (a \circ c) * (b \circ d).\]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[ \alpha \circ \beta = \beta \circ \alpha \]

\[ (\alpha \circ 1) \circ (1 \circ \beta) = 1 \circ (\beta \circ \alpha) \]

\[ (1 \circ \alpha) \circ (\beta \circ 1) = \alpha \circ (1 \circ \beta) \]

\[ (1 \circ \alpha) \circ (\beta \circ 1) = 1 \circ (\beta \circ \alpha) \]

\[ (\alpha \circ 1) \circ (1 \circ \beta) = \alpha \circ (1 \circ \beta) \]

\[ (1 \circ \alpha) \circ (\beta \circ 1) = 1 \circ (\beta \circ \alpha) \]

\[ (\alpha \circ 1) \circ (1 \circ \beta) = \beta \circ (1 \circ \alpha) \]

\[ (1 \circ \alpha) \circ (\beta \circ 1) = 1 \circ (\beta \circ \alpha) \]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[ \alpha \triangleleft \beta \]

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ \beta \ast \alpha \]

\[ \alpha \triangleleft \beta \]

\[ (1 \circ \beta) \ast (\alpha \circ 1) \]

\[ \alpha \triangleleft \beta \]

\[ (1 \circ \alpha) \ast (\beta \circ 1) \]

\[ \beta \ast \alpha \]

\[ \alpha \triangleleft \beta \]

\[ (1 \circ \beta) \ast (\alpha \circ 1) \]

\[ \alpha \triangleleft \beta \]
2. The Eckmann-Hilton argument

\[
\beta \ast \alpha \to (\beta \circ 1) \ast (1 \circ \alpha)
\]

interchange

\[
(\beta \ast 1) \circ (1 \ast \alpha)
\]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ \beta \ast \alpha \]

\[ (\beta \ast 1) \circ (1 \ast \alpha) \]

\[ \beta \circ \alpha \]

\[ (1 \ast \beta) \circ (\alpha \ast 1) \]

\[ \beta \ast \alpha \]

\[ (1 \ast \beta) \circ (\alpha \ast 1) \]

\[ \beta \circ \alpha \]

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ \beta \ast \alpha \]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ \beta \circ \alpha \]

\[ (1 \ast \beta) \circ (\alpha \ast 1) \]

\[ (1 \circ \alpha) \ast (\beta \circ 1) \]

\[ \alpha \ast \beta \]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[
\begin{align*}
\alpha \ast \beta &\rightarrow (\beta \circ 1) \ast (1 \circ \alpha) \\
\beta \circ \alpha &\rightarrow (\beta \ast 1) \circ (1 \ast \alpha) \\
(\alpha \ast 1) \circ (1 \ast \beta) &\rightarrow (\alpha \circ 1) \ast (1 \circ \beta) \\
(1 \ast \beta) \circ (\alpha \ast 1) &\rightarrow (1 \circ \alpha) \ast (\beta \circ 1) \\
\end{align*}
\]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

\[ (\alpha \circ 1) \ast (1 \circ \beta) \]

\[ (1 \circ \alpha) \ast (\beta \circ 1) \]

\[ (1 \ast \alpha) \circ (\beta \ast 1) \]

\[ (\beta \ast 1) \circ (1 \ast \alpha) \]

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ (\alpha \ast 1) \circ (\beta \ast 1) \]

\[ (\alpha \ast 1) \circ (\beta \circ 1) \]

\[ (\beta \ast 1) \circ (1 \ast \alpha) \]

\[ (\beta \circ 1) \ast (1 \circ \alpha) \]

\[ (\alpha \circ 1) \ast (1 \circ \beta) \]

\[ (1 \circ \alpha) \ast (\beta \circ 1) \]

\[ (1 \ast \alpha) \circ (\beta \ast 1) \]

\[ (\alpha \ast 1) \circ (\beta \circ 1) \]
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

Increasing dimensions: tricategories

We use a categorified Eckmann-Hilton argument:
2. The Eckmann-Hilton argument

Increasing dimensions: tricategories

We use a categorified Eckmann-Hilton argument:

Let $\mathcal{C}$ be a category with monoidal structures $\otimes_0$ and $\otimes_1$ such that

1. $\otimes_0$ and $\otimes_1$ have the same unit, and
2. there are coherent interchange isomorphisms

$$(a \otimes_0 b) \otimes_1 (c \otimes_1 d) \sim (a \otimes_1 c) \otimes_0 (b \otimes_1 d).$$

Then $\otimes_0$ and $\otimes_1$ are isomorphic and we have coherent isomorphisms

$$a \otimes_0 b \sim (b \otimes_1 a).$$

—a braiding.
2. The Eckmann-Hilton argument

Increasing dimensions: tricategories

We use a categorified Eckmann-Hilton argument:

Let $\mathcal{C}$ be a category with monoidal structures $\otimes_0$ and $\otimes_1$ such that

1. $\otimes_0$ and $\otimes_1$ have the same unit, and
2. there are coherent interchange isomorphisms

$$(a \otimes_0 b) \otimes_1 (c \otimes d) \xrightarrow{\cong} (a \otimes_1 c) \otimes_0 (b \otimes_1 d).$$
2. The Eckmann-Hilton argument

Increasing dimensions: tricategories

We use a categorified Eckmann-Hilton argument:

Let $\mathsf{C}$ be a category with monoidal structures $\otimes_0$ and $\otimes_1$ such that

1. $\otimes_0$ and $\otimes_1$ have the same unit, and
2. there are coherent interchange isomorphisms

\[
(a \otimes_0 b) \otimes_1 (c \otimes d) \xrightarrow{\cong} (a \otimes_1 c) \otimes_0 (b \otimes_1 d).
\]

Then $\otimes_0$ and $\otimes_1$ are isomorphic and we have coherent isomorphisms

\[
a \otimes b \xrightarrow{\cong} b \otimes a.
\]
2. The Eckmann-Hilton argument

Increasing dimensions: tricategories

We use a categorified Eckmann-Hilton argument:

Let $\mathcal{C}$ be a category with monoidal structures $\otimes_0$ and $\otimes_1$ such that

1. $\otimes_0$ and $\otimes_1$ have the same unit, and
2. there are coherent interchange isomorphisms

\[(a \otimes_0 b) \otimes_1 (c \otimes d) \xrightarrow{\cong} (a \otimes_1 c) \otimes_0 (b \otimes_1 d).\]

Then $\otimes_0$ and $\otimes_1$ are isomorphic and we have coherent isomorphisms

\[a \otimes b \xrightarrow{\cong} b \otimes a.\]

—a braiding.
2. The Eckmann-Hilton argument

Slight problem
2. The Eckmann-Hilton argument

Slight problem

Horizontal composition of 2-cells is not strictly unital in a bicategory.

\[ \alpha \circ 1 = ? \]
2. The Eckmann-Hilton argument

Slight problem

Horizontal composition of 2-cells is not strictly unital in a bicategory.

\[ \alpha \circlearrowleft 1 = ? \]

Instead, we have to use the following operation:
2. The Eckmann-Hilton argument

Slight problem

Horizontal composition of 2-cells is not strictly unital in a bicategory.

\[ \alpha \alpha^{-1} = ? \]

Instead, we have to use the following operation:
2. The Eckmann-Hilton argument

Slight problem

Horizontal composition of 2-cells is not strictly unital in a bicategory.

\[
\begin{array}{cc}
\alpha & \beta \\
\hline
1 & ?
\end{array}
\]

Instead, we have to use the following operation:

This issue gets worse as we increase dimensions.
2. The Eckmann-Hilton argument

Doubly degenerate tricategories
2. The Eckmann-Hilton argument

**Doubly degenerate tricategories**

\[
\begin{array}{c}
0\text{-cells} \\ 1\text{-cells} \\ \text{trivial}
\end{array}
\]
2. The Eckmann-Hilton argument

Doubly degenerate tricategories

0-cells \{ 1-cells \} trivial

2-cells \rightarrow objects

Nevertheless, a lengthy calculation shows that a doubly degenerate tricategory is indeed a braided monoidal category.
2. The Eckmann-Hilton argument

Doubly degenerate tricategories

\[
\begin{array}{c}
0\text{-cells} \quad 1\text{-cells} \\
\{ \text{trivial} \}
\end{array}
\]

\[
\begin{array}{c}
2\text{-cells} \\
3\text{-cells}
\end{array} \quad \rightarrow \quad \text{objects}
\]

\[
\begin{array}{c}
2\text{-cells} \\
3\text{-cells}
\end{array} \quad \rightarrow \quad \text{morphisms}
\]
2. The Eckmann-Hilton argument

Doubly degenerate tricategories

\[
\begin{align*}
0\text{-cells} & \quad \bigg\} \quad \text{trivial} \\
1\text{-cells} & \\
2\text{-cells} & \quad \rightarrow \quad \text{objects} \\
3\text{-cells} & \quad \rightarrow \quad \text{morphisms} \\
\text{composition of 2\text{-cells}} & \end{align*}
\]
Doubly degenerate tricategories

0-cells \{ 1-cells \} trivial

2-cells \rightarrow \text{objects}

3-cells \rightarrow \text{morphisms}

composition of 2-cells

1-composition \rightarrow \otimes
2. The Eckmann-Hilton argument

Doubly degenerate tricategories

0-cells \{ 1-cells \} trivial

2-cells \rightarrow objects
3-cells \rightarrow morphisms

composition of 2-cells

1-composition \rightarrow \otimes
0-composition \rightarrow not quite a \otimes

Nevertheless, a lengthy calculation shows that a doubly degenerate tricategory is indeed a braided monoidal category.
2. The Eckmann-Hilton argument

Doubly degenerate tricategories

0-cells \{ 1-cells \} trivial

2-cells \rightarrow objects

3-cells \rightarrow morphisms

composition of 2-cells

1-composition \rightarrow \otimes

0-composition \rightarrow not quite a \otimes

Nevertheless, a lengthy calculation shows that a doubly degenerate tricategory is indeed a braided monoidal category.
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument
2. The Eckmann-Hilton argument

In general

• $(k-1)$-composition becomes a $\otimes$, but
• $i$-composition gets further and further from really being a $\otimes$ as $i$ decreases to 0.

Moral

Our slogan was $k$-degenerate $n$-categories "are" $k$-tuply monoidal $(n-k)$-categories, but it does take some effort.
2. The Eckmann-Hilton argument

In general

- \((k - 1)\)-composition becomes a \(\otimes\), but
2. The Eckmann-Hilton argument

In general

- $(k - 1)$-composition becomes a $\otimes$, but
- $i$-composition gets further and further from really being a $\otimes$ as $i$ decreases to 0.
In general

- $(k - 1)$-composition becomes a $\otimes$, but
- $i$-composition gets further and further from really being a $\otimes$ as $i$ decreases to 0.

Moral
2. The Eckmann-Hilton argument

In general

- $(k - 1)$-composition becomes a $\otimes$, but
- $i$-composition gets further and further from really being a $\otimes$ as $i$ decreases to 0.

Moral

Our slogan was

$k$-degenerate $n$-categories

“are”

$k$-tuply monoidal $(n - k)$-categories.
2. The Eckmann-Hilton argument

In general

- $(k - 1)$-composition becomes a $\otimes$, but
- $i$-composition gets further and further from really being a $\otimes$ as $i$ decreases to 0.

Moral

Our slogan was

\[ k\text{-degenerate } n\text{-categories} \]

\[ \text{“are”} \]

\[ k\text{-tuply monoidal } (n - k)\text{-categories}. \]

but it does take some effort.
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell and one 1-cell is precisely a commutative monoid with a distinguished invertible element. This comes from coherence constraints:

\[
\begin{array}{c|c|c|c}
\text{0-cells} & \text{1-cells} & \text{2-cells} & \text{coherence isomorphisms} \\
\hline
\text{old} & \text{new} & \text{trivial} & \text{invertible elements}
\end{array}
\]
3. Inconvenient elements

Theorem (Leinster).
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid.
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid with a distinguished invertible element.
3. Inconvenient elements

Theorem (Leinster).
A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid 
with a distinguished invertible element.

This comes from coherence constraints:
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid with a distinguished invertible element.

This comes from coherence constraints:

```
“old”           “new”
```

\[ \text{trivial} \]
Theorem (Leinster).

A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid with a distinguished invertible element.

This comes from coherence constraints:

```
“old”  →  “new”
```

```
0-cells
1-cells
```

trivial
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell $x$ and one 1-cell $1_x$ is precisely a commutative monoid with a distinguished invertible element.

This comes from coherence constraints:

```
“old”  ───►  “new”

0-cells  }  trivial
1-cells  

2-cells  ───►  elements
```
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell \( x \) and one 1-cell \( 1_x \) is precisely a commutative monoid with a distinguished invertible element.

This comes from coherence constraints:

```
  “old”   ────>  “new”

0-cells  } trivial

1-cells  }

2-cells  ────>  elements

coherence isomorphisms
```
3. Inconvenient elements

Theorem (Leinster).

A bicategory with only one 0-cell \( x \) and one 1-cell \( 1_x \) is precisely a commutative monoid with a distinguished invertible element.

This comes from coherence constraints:

```
"old"  \[\longrightarrow\]  "new"

0-cells  \[\} \]  trivial

1-cells  \[\} \]  elements

2-cells  \[\longrightarrow\]  coherence isomorphisms  \[\longrightarrow\]  invertible elements
```
3. Inconvenient elements

Coherence constraints giving distinguished invertible elements

We might expect three such elements: $\alpha$, $\iota$, $\eta$. 
Coherence constraints giving distinguished invertible elements

We might expect three such elements: \( a, \ i, \ r. \)

However

\[ 2^2 = 3, \quad \text{so} \quad 2 = 1, \quad \text{and} \quad \text{in any bicategory,} \quad I = I, \quad \text{so we have} \quad 2 = 2. \]

This leaves just one distinguished invertible element:

\[ I. \]
3. Inconvenient elements

Coherence constraints giving distinguished invertible elements

We might expect three such elements: \( a, l, r \).

However

- the associativity pentagon gives us \( a^2 = a^3 \), so \( a = 1 \), and
3. Inconvenient elements

Coherence constraints giving distinguished invertible elements

We might expect three such elements: \( a, l, r \).

However

- the associativity pentagon gives us \( a^2 = a^3 \), so \( a = 1 \),
- and
- in any bicategory, \( l_I = r_I \), so we have \( l = r \).
3. Inconvenient elements

Coherence constraints giving distinguished invertible elements

We might expect three such elements: \( a, l, r \).

However

- the associativity pentagon gives us \( a^2 = a^3 \), so \( a = 1 \), and
- in any bicategory, \( l_I = r_I \), so we have \( l = r \).

This leaves just one distinguished invertible element: \( l \).
3. Inconvenient elements

Coherence constraints giving distinguished invertible elements

We might expect three such elements: $a$, $l$, $r$.

However

- the associativity pentagon gives us $a^2 = a^3$, so $a = 1,$
  and
- in any bicategory, $l_I = r_I$, so we have $l = r$.

This leaves just one distinguished invertible element: $l$.

Can we fix this using higher morphisms?
3. Inconvenient elements

Theorem (Leinster).

A weak functor between doubly degenerate bicategories

\[ F : X \longrightarrow Y \]
3. Inconvenient elements

**Theorem (Leinster).**

A weak functor between doubly degenerate bicategories

\[ F : X \rightarrow Y \]

is precisely a monoid homomorphism.
3. Inconvenient elements

**Theorem (Leinster).**

A weak functor between doubly degenerate bicategories

\[ F : X \longrightarrow Y \]

is precisely a monoid homomorphism

*together with a distinguished invertible element in Y.*
3. Inconvenient elements

Theorem (Leinster).

A weak functor between doubly degenerate bicategories

\[ F : X \rightarrow Y \]

is precisely a monoid homomorphism

*together with a distinguished invertible element in* \( Y \).

We have coherence isomorphisms for weak functoriality
3. Inconvenient elements

Theorem (Leinster).

A weak functor between doubly degenerate bicategories

\[ F : X \rightarrow Y \]

is precisely a monoid homomorphism

*together with a distinguished invertible element in* \( Y \).

We have coherence isomorphisms for weak functoriality

\[ \phi_{II} : FI \circ FI \Rightarrow F(I \circ I) \]
3. Inconvenient elements

**Theorem (Leinster).**

A weak functor between doubly degenerate bicategories

\[ F : X \longrightarrow Y \]

is precisely a monoid homomorphism

*together with a distinguished invertible element in* \( Y \).

We have coherence isomorphisms for weak functoriality

\[ \phi_{II} : FI \circ FI \Rightarrow F(I \circ I) \]
\[ \phi_{x} : I \Rightarrow FI \]
3. Inconvenient elements

Theorem (Leinster).
A weak functor between doubly degenerate bicategories

\[ F : X \longrightarrow Y \]

is precisely a monoid homomorphism

*together with a distinguished invertible element in \( Y \).*

We have coherence isomorphisms for weak functoriality

\[ \phi_{II} : FI \circ FI \Rightarrow F(I \circ I) \]

\[ \phi_x : I \Rightarrow FI \]

The axioms eliminate one of them.
3. Inconvenient elements

A weak transformation $F \Rightarrow G$
3. Inconvenient elements

A weak transformation $F \Rightarrow G$

\[
\begin{array}{c}
\uparrow \downarrow \\
X & F & Y \\
\downarrow & \uparrow \\
& G \\
\end{array}
\]
3. Inconvenient elements

A weak transformation $F \Rightarrow G$

\[
\begin{array}{c}
X \downarrow \quad \downarrow \\
F \quad G \\
Y \quad \uparrow \uparrow
\end{array}
\]

is the assertion $F = G$. 
3. Inconvenient elements

A weak transformation $F \Rightarrow G$

\[ F \xrightarrow{\downarrow} \xrightarrow{\uparrow} \xrightarrow{\downarrow} \xrightarrow{\downarrow} \]

is the assertion $F = G$.

A modification “from the assertion $F = G$ to itself”
3. Inconvenient elements

A weak transformation $F \Rightarrow G$

\[ F \quad \Rightarrow \quad G \]

$X \quad \downarrow \quad Y$

is the assertion $F = G$.

A modification “from the assertion $F = G$ to itself”

\[ F \quad \Rightarrow \quad G \]

$X \quad \Rightarrow \quad Y$
3. Inconvenient elements

A weak transformation $F \Rightarrow G$

\[
\begin{array}{c}
\text{X} \\
\downarrow \\
\text{Y}
\end{array}
\quad \quad \quad
\begin{array}{c}
\Rightarrow \\
\text{F} \\
\text{G}
\end{array}
\quad \quad \quad
\begin{array}{c}
\text{G} \\
\uparrow \\
\text{Y}
\end{array}
\quad \quad \quad
\begin{array}{c}
\text{F} \\
\downarrow \\
\text{X}
\end{array}
\]

is the assertion $F = G$.

A modification “from the assertion $F = G$ to itself”

\[
\begin{array}{c}
\text{X} \\
\Rightarrow \\
\text{Y}
\end{array}
\quad \quad \quad
\begin{array}{c}
\Rightarrow \\
\text{F} \\
\text{G}
\end{array}
\quad \quad \quad
\begin{array}{c}
\text{G} \\
\downarrow \\
\text{X}
\end{array}
\]

is a distinguished element in $Y$. 
3. Inconvenient elements

**Totality of doubly degenerate bicategories**
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

doubly degenerate bicategories
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

doubly degenerate bicategories

weak functors
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

doubly degenerate bicategories
weak functors
weak transformations
3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of doubly degenerate bicategories</th>
<th>Totality of commutative monoids</th>
</tr>
</thead>
<tbody>
<tr>
<td>weak functors</td>
<td></td>
</tr>
<tr>
<td>weak transformations</td>
<td></td>
</tr>
<tr>
<td>modifications</td>
<td></td>
</tr>
</tbody>
</table>

doubly degenerate bicategories
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

doubly degenerate bicategories

weak functors

weak transformations

modifications

commutative monoids
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

doubly degenerate bicategories

weak functors

weak transformations

modifications

commutative monoids

homomorphisms
3. Inconvenient elements

**Totality of doubly degenerate bicategories**

- doubly degenerate bicategories
- weak functors
- weak transformations
- modifications

**Totality of commutative monoids**

- commutative monoids
- homomorphisms
- identities

---

Structure comparison
### 3. Inconvenient elements

**Totality of doubly degenerate bicategories**

<table>
<thead>
<tr>
<th>Weak functors</th>
<th>Commutative monoids</th>
</tr>
</thead>
<tbody>
<tr>
<td>Weak transformations</td>
<td>Homomorphisms</td>
</tr>
<tr>
<td>Modifications</td>
<td>Identities</td>
</tr>
</tbody>
</table>

**Totality of commutative monoids**
3. Inconvenient elements

- Totality of doubly degenerate bicategories
- Weak functors
- Weak transformations
- Modifications

structure comparison

- Totality of commutative monoids
- Commutative monoids
- Homomorphisms
- Identities
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

not equivalence

Totality of commutative monoids

- doubly degenerate bicategories
  - weak functors
  - weak transformations
  - modifications

- commutative monoids
  - homomorphisms
  - identities
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

not equivalence

doubly degenerate bicategories

weak functors

weak transformations

modifications

Totality of commutative monoids

commutative monoids

homomorphisms

identities

identities
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

not equivalence

Totality of commutative monoids

doubly degenerate bicategories

weak functors

weak transformations

modifications

cat.

bicat.

not equivalence

equivalence

commutative monoids

homomorphisms

identities

identities
3. Inconvenient elements

Totality of doubly degenerate bicategories

Structure comparison

Totality of commutative monoids

Not equivalence

Equivalence

Doubly degenerate bicategories
Weak functors
Weak transformations
Modifications

Cat.
Bicat.
Tricat.

Cat.
Bicat.
Bicat.
Cat.

Commutative monoids
Homomorphisms
Identities
Identities
3. Inconvenient elements

Totality of doubly degenerate bicategories

structure comparison

Totality of commutative monoids

not equivalence

\[
\begin{align*}
\text{cat.} & \quad \text{bicat.} & \quad \text{tricat.} & \quad \text{bicat.} & \quad \text{tricat.} \\
\text{doubly degenerate bicategories} & \quad \text{weak functors} & \quad \text{weak transformations} & \quad \text{modifications} & \quad \text{commutative monoids} \\
\text{not equivalence} & \quad \text{equivalence} & \quad \text{not equivalence} & \quad \text{not equivalence} & \quad \text{homomorphisms} \\
\text{not equivalence} & \quad \text{equivalence} & \quad \text{not equivalence} & \quad \text{not equivalence} & \quad \text{identities} \\
\text{not equivalence} & \quad \text{equivalence} & \quad \text{not equivalence} & \quad \text{not equivalence} & \quad \text{identities} \\
\end{align*}
\]
### 3. Inconvenient elements

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
<td>\ldots</td>
</tr>
<tr>
<td>\equiv category with only one object</td>
<td>\equiv 2-category with only one object</td>
<td>\equiv 3-category with only one object</td>
<td>\equiv 4-category with only one object</td>
<td>\ldots</td>
</tr>
<tr>
<td>commutative monoid</td>
<td>braided mon. category</td>
<td>braided mon. 2-category</td>
<td>braided mon. 3-category</td>
<td>\ldots</td>
</tr>
<tr>
<td>\equiv 2-cat. with only one 1-cell</td>
<td>\equiv 3-cat. with only one 1-cell</td>
<td>\equiv 4-cat. with only one 1-cell</td>
<td>\equiv 5-cat. with only one 1-cell</td>
<td>\ldots</td>
</tr>
<tr>
<td>symmetric mon. category</td>
<td>symmetric mon. 2-category</td>
<td>symmetric mon. 3-category</td>
<td>\equiv 6-cat. with only one 2-cell</td>
<td>\ldots</td>
</tr>
<tr>
<td>\equiv 3-cat. with only one 2-cell</td>
<td>\equiv 4-cat. with only one 2-cell</td>
<td>\equiv 5-cat. with only one 2-cell</td>
<td>\equiv 7-cat. with only one 3-cell</td>
<td>\ldots</td>
</tr>
<tr>
<td>\equiv 4-cat. with only one 3-cell</td>
<td>\equiv 5-cat. with only one 3-cell</td>
<td>\equiv 6-cat. with only one 3-cell</td>
<td>\equiv \ldots</td>
<td>\ldots</td>
</tr>
</tbody>
</table>

...
3. Inconvenient elements

**Totality of triply degenerate tricategories**
3. Inconvenient elements

Totality of triply degenerate tricategories

structure comparison
3. Inconvenient elements

Totality of triply degenerate tricategories \hspace{10cm} \text{structure comparison} \hspace{10cm} \text{Totality of commutative monoids}
3. Inconvenient elements

Totality of triply degenerate tricategories

structure comparison

Totality of commutative monoids

triply degenerate tricategories
3. Inconvenient elements

**Totality of triply degenerate tricategories**

---

structure comparison

---

**Totality of commutative monoids**

---

triply degenerate tricategories

weak functors
3. Inconvenient elements

**Totality of**
- **triply degenerate**
- **tricategories**

**structure comparison**

**Totality of**
- **commutative**
- **monoids**

triply degenerate tricategories

weak functors

tritransformations
3. Inconvenient elements

Totality of triply degenerate tricategories

structure comparison

Totality of commutative monoids

triply degenerate tricategories
weak functors
tritransformations
trimodifications
3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of triply degenerate tricategories</th>
<th>structure comparison</th>
<th>Totality of commutative monoids</th>
</tr>
</thead>
</table>

- triply degenerate tricategories
- weak functors
- tritransformations
- trimodifications
- perturbations
3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of triply degenerate tricategories</th>
<th>structure comparison</th>
<th>Totality of commutative monoids</th>
</tr>
</thead>
<tbody>
<tr>
<td>triply degenerate tricategories</td>
<td>commutative monoids</td>
<td></td>
</tr>
<tr>
<td>weak functors</td>
<td>homomorphisms</td>
<td></td>
</tr>
<tr>
<td>tritransformations</td>
<td>identities</td>
<td></td>
</tr>
<tr>
<td>trimodifications</td>
<td>identities</td>
<td></td>
</tr>
<tr>
<td>perturbations</td>
<td>identities</td>
<td></td>
</tr>
</tbody>
</table>
## 3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of</th>
<th>structure comparison</th>
<th>Totality of</th>
</tr>
</thead>
<tbody>
<tr>
<td>triply degenerate tricategories</td>
<td></td>
<td>commutative monoids</td>
</tr>
</tbody>
</table>

- triply degenerate tricategories
- weak functors
- tritransformations
- trimodifications
- perturbations

- not a category
- homomorphisms
- identities
- identities
- identities
### 3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of triply degenerate tricategories</th>
<th>Structure comparison</th>
<th>Totality of commutative monoids</th>
</tr>
</thead>
<tbody>
<tr>
<td>triply degenerate tricategories</td>
<td>not a bicategory</td>
<td>commutative monoids</td>
</tr>
<tr>
<td>weak functors</td>
<td>homomorphisms</td>
<td>identities</td>
</tr>
<tr>
<td>tritransformations</td>
<td>identities</td>
<td>identities</td>
</tr>
<tr>
<td>trimodifications</td>
<td>identities</td>
<td>identities</td>
</tr>
<tr>
<td>perturbations</td>
<td>identities</td>
<td>identities</td>
</tr>
</tbody>
</table>
3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of</th>
<th>structure comparison</th>
<th>Totality of</th>
</tr>
</thead>
<tbody>
<tr>
<td>triply degenerate tricategories</td>
<td></td>
<td>commutative monoids</td>
</tr>
<tr>
<td>triply degenerate tricategories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weak functors</td>
<td></td>
<td>homomorphisms</td>
</tr>
<tr>
<td>tritransformations</td>
<td></td>
<td>identities</td>
</tr>
<tr>
<td>trimodifications</td>
<td></td>
<td>identities</td>
</tr>
<tr>
<td>perturbations</td>
<td></td>
<td>identities</td>
</tr>
</tbody>
</table>
3. Inconvenient elements

**Totality of triply degenerate tricategories**

- triply degenerate tricategories
- weak functors
- tritransformations
- trimodifications
- perturbations

---

**Totality of commutative monoids**

- commutative monoids
- homomorphisms
- identities
- identities
- identities

---

Structure comparison
3. Inconvenient elements

**Totality of triply degenerate tricategories**

- triply degenerate tricategories
- weak functors
- tritransformations
- trimodifications
- perturbations

**Structure comparison**

**Equivalence**

**Totality of commutative monoids**

- commutative monoids
- homomorphisms
- identities
- identities
- identities
3. Inconvenient elements

**Totality of triply degenerate tricategories**

- Triply degenerate tricategories
- Weak functors
- Tritransformations
- Trimodifications
- Perturbations

**Totality of commutative monoids**

- Commutative monoids
- Homomorphisms
- Identities

---

### Structure Comparison

**equivalence**

**tricategory**

**tetracategory**
3. Inconvenient elements

Totality of triply degenerate tricategories

structure comparison

Totality of commutative monoids

equivalence

triply degenerate tricategories
weak functors
tritransformations
trimodifications
perturbations

tricategory

tetracategory
tetracategory

commutative monoids
homomorphisms
identities
identities
identities
3. Inconvenient elements

<table>
<thead>
<tr>
<th>Totality of triple degenerate tricategories</th>
<th>Structure comparison</th>
<th>Totality of commutative monoids</th>
</tr>
</thead>
<tbody>
<tr>
<td>Totality of commutative monoids</td>
<td></td>
<td></td>
</tr>
<tr>
<td>equivalence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>not equivalence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>triple degenerate tricategories</td>
<td></td>
<td></td>
</tr>
<tr>
<td>weak functors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tritransformations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>trimodifications</td>
<td></td>
<td></td>
</tr>
<tr>
<td>perturbations</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tetracategory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tetracategory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tetracategory</td>
<td></td>
<td></td>
</tr>
<tr>
<td>commutative monoids</td>
<td></td>
<td></td>
</tr>
<tr>
<td>homomorphisms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>identities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>identities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>identities</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3. Inconvenient elements

In general

The issue of distinguished elements affects $k$-degenerate $n$-categories for all $k \geq 2$. The issue goes away for non-algebraic definitions i.e. when coherence constraints are not specified. However, there are still other problems.
3. Inconvenient elements

In general

• The issue of distinguished elements affects $k$-degenerate $n$-categories for all $k \geq 2$. 
3. Inconvenient elements

In general

- The issue of distinguished elements affects $k$-degenerate $n$-categories for all $k \geq 2$.
- The issue goes away for non-algebraic definitions i.e. when coherence constraints are not specified.
3. Inconvenient elements

In general

- The issue of distinguished elements affects $k$-degenerate $n$-categories for all $k \geq 2$.
- The issue goes away for non-algebraic definitions i.e. when coherence constraints are not specified.

However there are still other problems.
4. Higher morphisms

• A bicategory with only one 0-cell is a monoidal category.
• A weak functor between such is a monoidal functor.
• A weak transformation between such is quite different from a monoidal transformation.
Degenerate bicategories
Degenerate bicategories

- A bicategory with only one 0-cell is a monoidal category.
4. Higher morphisms

Degenerate bicategories

- A bicategory with only one 0-cell is a monoidal category.
- A weak functor between such is a monoidal functor.
4. Higher morphisms

Degenerate bicategories

- A bicategory with only one 0-cell is a monoidal category.
- A weak functor between such is a monoidal functor.
- A weak transformation between such is quite different from a monoidal transformation.
4. Higher morphisms

A weak transformation of degenerate bicategories

\[
\begin{array}{ccc}
X & \downarrow \alpha & Y \\
F & \circlearrowleft & G
\end{array}
\]

\[\alpha \in Y\text{ for all }A \in Y\text{ a morphism }\alpha_A : GA \otimes \alpha \to \alpha \otimes FA\text{ satisfying axioms.}\]
4. Higher morphisms

A weak transformation of degenerate bicategories

\[ F \xrightarrow{\alpha} G \]

is an object \( \alpha \in Y \) together with

\[ \alpha : G \circ F \rightarrow F \circ G \]

This is very different from a monoidal transformation, which has for all \( A \in Y \) a morphism

\[ \alpha_A : F(A) \rightarrow G(A) \]
4. Higher morphisms

A weak transformation of degenerate bicategories

\[
\begin{array}{ccc}
X & \xrightarrow{\alpha} & Y \\
\downarrow & & \downarrow \\
\downarrow & & \downarrow \\
\alpha & & \alpha \\
\end{array}
\]

is an object \( \alpha \in Y \) together with

for all \( A \in Y \) a morphism

\[ \alpha_A : GA \otimes \alpha \longrightarrow \alpha \otimes FA \]
4. Higher morphisms

A weak transformation of degenerate bicategories

\[
\begin{array}{ccc}
F & \Downarrow\alpha & G \\
X & \downarrow & Y \\
G & \Downarrow\alpha & F \\
\end{array}
\]

is an object \( \alpha \in Y \) together with

for all \( A \in Y \) a morphism

\[
\alpha_A : GA \otimes \alpha \longrightarrow \alpha \otimes FA
\]

satisfying axioms.
4. Higher morphisms

A weak transformation of degenerate bicategories

\[
\begin{array}{ccc}
X & \downarrow & Y \\
F & \circlearrowleft & G \\
\downarrow & & \downarrow \\
\alpha & & \alpha
\end{array}
\]

is an object \( \alpha \in Y \) together with

for all \( A \in Y \) a morphism

\[
\alpha_A : GA \otimes \alpha \longrightarrow \alpha \otimes FA
\]

satisfying axioms.

This is very different from a monoidal transformation,
4. Higher morphisms

A weak transformation of degenerate bicategories

\[
\begin{array}{ccc}
X & \overset{F}{\longrightarrow} & Y \\
\downarrow^{\alpha} & & \downarrow^{\alpha} \\
G & \overset{\alpha}{\longleftarrow} & \end{array}
\]

is an object \(\alpha \in Y\) together with for all \(A \in Y\) a morphism

\[
\alpha_A : GA \otimes \alpha \longrightarrow \alpha \otimes FA
\]
satisfying axioms.

This is very different from a monoidal transformation, which has for all \(A \in Y\) a morphism

\[
\alpha_A : FA \longrightarrow GA.
\]
4. Higher morphisms

Can we fix this?
4. Higher morphisms

Can we fix this?

- Modifications don’t help.
4. Higher morphisms

Can we fix this?

• Modifications don’t help.

• Restrict to $\alpha = I$ and lax transformations?
4. Higher morphisms

Can we fix this?

• Modifications don’t help.

• Restrict to $\alpha = I$ and lax transformations? —not closed under composition.
4. Higher morphisms

Can we fix this?

- Modifications don’t help.
- Restrict to $\alpha = I$ and lax transformations? —not closed under composition.
- Construct closure under composition?
Can we fix this?

- Modifications don’t help.

- Restrict to $\alpha = I$ and lax transformations?
  —not closed under composition.

- Construct closure under composition?
  —this doesn’t work (technical).
4. Higher morphisms

Can we fix this?

• Modifications don’t help.

• Restrict to $\alpha = I$ and lax transformations? —not closed under composition.

• Construct closure under composition? —this doesn’t work (technical).

• Icons? (Lack)
Can we fix this?

• Modifications don’t help.

• Restrict to $\alpha = I$ and lax transformations?  
  —not closed under composition.

• Construct closure under composition?  
  —this doesn’t work (technical).

• Icons? (Lack)  
  —this works, but it isn’t a restriction of $\textbf{Bicat}$. 
4. Higher morphisms

We should probably proceed in two separate stages:
4. Higher morphisms

We should probably proceed in two separate stages:

\[ k \text{-degenerate } n \text{-categories} \]
4. Higher morphisms

We should probably proceed in two separate stages:

\[ \text{k-degenerate } n\text{-categories} \]
4. Higher morphisms

We should probably proceed in two separate stages:

\[
\begin{array}{c}
\text{\textit{k-degenerate \textit{n-categories}}}\\
\downarrow
\\
\text{\textit{k-tuply monoidal \textit{n-categories}}}
\end{array}
\]
4. Higher morphisms

We should probably proceed in two separate stages:

\[ k\text{-degenerate } n\text{-categories} \]

\[ \downarrow \]

\[ k\text{-tuply monoidal } n\text{-categories} \]
4. Higher morphisms

We should probably proceed in two separate stages:

$k$-degenerate $n$-categories

\[
\begin{array}{c}
\downarrow \\
\end{array}
\]

$k$-tuply monoidal $n$-categories

\[
\begin{array}{c}
\downarrow \\
\end{array}
\]

Periodic Table

Moral so far: The second step is more precise than the first.
4. Higher morphisms

We should probably proceed in two separate stages:

\[ k\text{-degenerate } n\text{-categories} \]

\[ \Downarrow \]

\[ k\text{-tuply monoidal } n\text{-categories} \]

\[ \Downarrow \]

Periodic Table

Moral so far:

The second step is more precise than the first.
5. Stabilisation
5. Stabilisation

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
<td>\ldots</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td>≡ 4-category with only one object</td>
<td></td>
</tr>
</tbody>
</table>

- **commutative monoid**
  - ≡ 2-cat. with only one 1-cell
  - ≡ 3-cat. with only one 1-cell
  - ≡ 4-cat. with only one 1-cell
  - ≡ 5-cat. with only one 1-cell

- **symmetric mon. category**
  - ≡ 3-cat. with only one 2-cell
  - ≡ 4-cat. with only one 2-cell
  - ≡ 5-cat. with only one 2-cell
  - ≡ 6-cat. with only one 2-cell

- **symmetric mon. 2-category**
  - ≡ 4-cat. with only one 3-cell
  - ≡ 5-cat. with only one 3-cell
  - ≡ 6-cat. with only one 3-cell
  - ≡ 7-cat. with only one 3-cell

| \ldots | \ldots | \ldots | \ldots |
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category:
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n+2$. 
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n + 2$.

adding a monoidal structure
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n + 2$.

adding a monoidal structure
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n + 2$.

adding a monoidal structure

making existing monoidal structure more symmetric
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n + 2$.

adding a monoidal structure

↓

making existing monoidal structure more symmetric

Eventually it becomes maximally symmetric
5. Stabilisation

Idea

There’s a limit to how many monoidal structures we can fit on an $n$-category: $n + 2$.

adding a monoidal structure

\[
\downarrow
\]

making existing monoidal structure more symmetric

Eventually it becomes maximally symmetric
— we get symmetric monoidal $n$-categories.
<table>
<thead>
<tr>
<th>Category</th>
<th>Monoid</th>
<th>Monoidal Cat.</th>
<th>Monoidal 2-Cat.</th>
<th>Monoidal 3-Cat.</th>
<th>Monoidal 4-Cat.</th>
<th>4-Category</th>
<th>5-Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Only One</td>
<td>Only One</td>
<td>Only One</td>
<td>Only One</td>
<td>Only One</td>
<td>Only One</td>
<td>Only One</td>
</tr>
<tr>
<td>3-Category</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4-Category</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5-Category</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. Stabilisation
## 5. Stabilisation

<table>
<thead>
<tr>
<th>set</th>
<th>category</th>
<th>2-category</th>
<th>3-category</th>
<th>4-category</th>
</tr>
</thead>
<tbody>
<tr>
<td>monoid</td>
<td>monoidal cat.</td>
<td>monoidal 2-cat.</td>
<td>monoidal 3-cat.</td>
<td>3-categories</td>
</tr>
<tr>
<td>≡ category with only one object</td>
<td>≡ 2-category with only one object</td>
<td>≡ 3-category with only one object</td>
<td>≡ 4-category with only one object</td>
<td>4-categories</td>
</tr>
<tr>
<td><strong>commutative monoid</strong></td>
<td>braided mon.</td>
<td>braided mon.</td>
<td>braided mon.</td>
<td>5-categories</td>
</tr>
<tr>
<td>≡ 2-cat. with only one 1-cell</td>
<td>≡ 3-cat. with only one 1-cell</td>
<td>≡ 4-cat. with only one 1-cell</td>
<td>≡ 5-cat. with only one 1-cell</td>
<td>6-categories</td>
</tr>
<tr>
<td><strong>symmetric mon. category</strong></td>
<td>sylleptic mon.</td>
<td>sylleptic mon.</td>
<td>sylleptic mon.</td>
<td>7-categories</td>
</tr>
<tr>
<td>≡ 3-cat. with only one 2-cell</td>
<td>≡ 4-cat. with only one 2-cell</td>
<td>≡ 5-cat. with only one 2-cell</td>
<td>≡ 6-cat. with only one 2-cell</td>
<td>8-categories</td>
</tr>
<tr>
<td><strong>symmetric mon. 2-category</strong></td>
<td>*** mon.</td>
<td>*** mon.</td>
<td>*** mon.</td>
<td>9-categories</td>
</tr>
<tr>
<td>≡ 4-cat. with only one 3-cell</td>
<td>≡ 5-cat. with only one 3-cell</td>
<td>≡ 6-cat. with only one 3-cell</td>
<td>≡ 7-cat. with only one 3-cell</td>
<td>10-categories</td>
</tr>
<tr>
<td><strong>symmetric mon. 3-category</strong></td>
<td></td>
<td></td>
<td></td>
<td>11-categories</td>
</tr>
</tbody>
</table>

...
5. Stabilisation

Extended TQFT Hypothesis (Baez-Dolan)
5. Stabilisation

Extended TQFT Hypothesis (Baez-Dolan)

The $n$-category of which $n$-dimensional extended TQFTs are representations is the free stable weak $n$-category with duals on one object.
6. Other reasons to care

• If we start with an $n$-category and restrict to a single 0-cell, we get a degenerate $n$-category.—like a loop space.

• If we restrict to the identity on that 0-cell, we get a 2-degenerate $n$-category.—like a double loop space.

There are many more connections with topology.
6. Other reasons to care

- If we start with an $n$-category and restrict to a single 0-cell, we get a degenerate $n$-category.
6. Other reasons to care

• If we start with an $n$-category and restrict to a single 0-cell, we get a degenerate $n$-category.
  —like a loop space.
6. Other reasons to care

- If we start with an $n$-category and restrict to a single 0-cell, we get a degenerate $n$-category.
  —like a loop space.

- If we restrict to the identity on that 0-cell, we get a 2-degenerate $n$-category.
6. Other reasons to care

- If we start with an $n$-category and restrict to a single 0-cell, we get a degenerate $n$-category.
  —like a loop space.

- If we restrict to the identity on that 0-cell, we get a 2-degenerate $n$-category.
  —like a double loop space.
6. Other reasons to care

- If we start with an \( n \)-category and restrict to a single 0-cell, we get a degenerate \( n \)-category.
  —like a loop space.

- If we restrict to the identity on that 0-cell, we get a 2-degenerate \( n \)-category.
  —like a double loop space.

There are many more connections with topology.
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us

- every weak 2-category is 2-equivalent to a strict 2-category,
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us

- every weak 2-category is 2-equivalent to a strict 2-category, but
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us

- every weak 2-category is 2-equivalent to a strict 2-category, but
- not every weak 3-category is 3-equivalent to a strict 3-category.
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us

- every weak 2-category is 2-equivalent to a strict 2-category, but
- not every weak 3-category is 3-equivalent to a strict 3-category.

The obstruction is braidings.
6. Other reasons to care

Degenerate $n$-categories capture the essence of weak $n$-categories.

Coherence tells us

- every weak 2-category is 2-equivalent to a strict 2-category, but
- not every weak 3-category is 3-equivalent to a strict 3-category.

The obstruction is braidings.

All the difficulties come from having non-trivial morphisms between identity cells.
6. Other reasons to care

**Slogan**

The Periodic Table measures the difference between weak and strict $n$-categories.