Two-dimensional algebra

Alistair Savage
University of Ottawa

Slides available online: alistairsavage.ca/talks
Goal: Introduce a two-dimensional approach to algebra by playing with blocks.

Overview:

1. Traditional (one-dimensional) algebra
2. Two-dimensional algebra
3. Being precise: monoidal categories
4. Going a bit further
Monoids

A **monoid** is a set $M$ with a binary operation

$$M \times M \rightarrow M, \quad (a, b) \mapsto ab,$$

that

- is associative, so $(ab)c = a(bc)$ for all $a, b, c \in M$;
- has an identity element $1$, so $1a = a = a1$ for all $a \in M$.

Associativity means we can omit parentheses. E.g. we can write:

$$ab^2ca1ba^3$$

Think of lining up boxes:

```
 a b b c a 1 b a a a
```
Games with blocks

We have relations; some sequences of boxes are equal.

Some relations are forced by the monoid axioms:

1 \ a = a = a \ 1

This relation is local; we can use it inside larger sequences of boxes:

\[ \begin{array}{cccccccc}
  a & b & b & c & a & 1 & b & a & a & a \\
\end{array} = \begin{array}{cccccccc}
  a & b & b & c & a & b & a & a & a & a \\
\end{array} \]

Particular monoids may have other relations. E.g. if a monoid is commutative, we have the local relation:

\[ \begin{array}{cc}
  a & b \\
\end{array} = \begin{array}{cc}
  b & a \\
\end{array} \quad \text{for all } a, b \]

Note

The axiom of associativity is built into our framework—we just omit parentheses.
More games with blocks

**Presentations**

We can define a monoid by giving a presentation:

- a set $S$ of generators (the blocks we can use),
- a set $R$ of (local) relations.

**Example**

Consider the monoid with

- one generator $a$,
- one relation $a^2 = 1$.

With blocks, we have $a \ a = 1$.

This monoid is the cyclic group with two elements.

Recall: A group is a monoid where every element $a$ has an inverse $a^{-1}$, so $aa^{-1} = 1 = a^{-1}a$. This is just a type of relation.
More games with blocks

Example

Consider the monoid with

- one generator \( a \),
- one relation \( a^n = 1 \).

With blocks, we have:

\[
\underbrace{a \ a \ \cdots \ a}_{n \text{ blocks}} = 1
\]

This monoid is the **cyclic group with** \( n \) **elements**.
More games with blocks

**Example**

Fix $n \geq 3$ and consider the monoid with two generators $r, s$ and relations:

\[
\underbrace{r \cdots r} = 1, \quad s \cdot s = 1, \quad r \cdot s \cdot r \cdot s = 1
\]

This is the **dihedral group** (group of symmetries of a regular $n$-gon).
More games with blocks

Example

Fix \( n \geq 1 \) and consider the monoid with generators

\[ s_1, \ldots, s_{n-1}, \]

and relations

\[
\begin{align*}
  s_i s_i &= 1, \quad 1 \leq i \leq n - 1, \\
  s_i s_{i+1} s_i &= s_{i+1} s_i s_{i+1}, \quad 1 \leq i \leq n - 2, \\
  s_i s_j &= s_j s_i, \quad 1 \leq i, j \leq n - 1, \ |i - j| > 1.
\end{align*}
\]

This is the symmetric group \( S_n \).
From blocks to strings

In the symmetric group $S_3$, let’s replace our blocks by string diagrams:

\[
\begin{align*}
    s_1 &= \begin{array}{c}
    \begin{array}{c}
    \text{String Diagram 1}
    \end{array}
    \end{array}, \\
    s_2 &= \begin{array}{c}
    \begin{array}{c}
    \text{String Diagram 2}
    \end{array}
    \end{array}, \\
    1 &= \begin{array}{c}
    \begin{array}{c}
    \text{Identity Diagram}
    \end{array}
    \end{array}
\end{align*}
\]

Lining up blocks becomes concatenation of string diagrams:

As for relations:

\[
\begin{align*}
    s_1 s_1 &= 1 \\
    s_2 s_2 &= 1 \\
    s_1 s_2 s_1 &= s_2 s_1 s_2
\end{align*}
\]

Exercise:
By definition, the symmetric group $S_n$ consists of bijections

$$\{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}.$$

The monoid operation is composition. Every bijection is invertible, so $S_n$ is a group.

We could also consider the monoid of all set maps

$$\{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}.$$

This is no longer a group, but it still a monoid. We can still use blocks:

$$\begin{array}{c|c}
  f & g \\
\end{array} = f \circ g$$

**Exercise:** Come up with a natural way of drawing string diagrams for this monoid.
Composition problems

What if we want to consider maps between different sets?

For $n \geq 0$, let

$$X_n = \{1, 2, \ldots, n\}.$$ 

By convention, $X_0 = \emptyset$.

Let

$$\mathcal{C}(m, n) = \{f : X_m \to X_n\} \quad \text{and} \quad \mathcal{C} = \bigcup_{m,n=0}^{\infty} \mathcal{C}(m, n).$$

**Question:** Does the operation of composition make this a monoid?

**NO!** Composition is not always defined. E.g. if

$$f : X_2 \to X_3 \quad \text{and} \quad g : X_5 \to X_4$$

then $f \circ g$ is not defined.
Colourful boxes

So $\mathcal{C}$ is not a monoid. So our boxes description breaks down.

How can we fix it? Colour the sides of boxes!

$\begin{array}{cccccc} a & b & c & d & e & f \end{array}$

Now we can only compose boxes where the sides match:

$\begin{array}{cccccc} a & d & b & b & c & a \quad d & b & e & f \end{array}$

The identity element 1 in a monoid is replaced by an identity for each colour:

$\begin{array}{cccc} 1 & 1 & 1 & 1 \end{array}$

These satisfy

$\begin{array}{cccc} 1a = a = a1, \quad 1d = d = d1 \end{array}$, etc.
Recall

\[ C(m, n) = \{ f: X_m \to X_n \} \quad \text{and} \quad C = \bigcup_{m,n=0}^{\infty} C(m, n). \]

Now we depict \( f: X_m \to X_n \) as a box with labelled/coloured sides

\[ n \quad \boxed{f} \quad m \]

We can only compose boxes (i.e. maps) when the labels match.

If we use string diagrams for set maps (\( \circ = \text{composition} \)):

\[ \text{not defined} \]
Making things precise

Category

A (small) category $\mathcal{C}$ consists of:

- a set $\text{Ob} \mathcal{C}$ of objects ("colours"),
- for all $X, Y \in \text{Ob} \mathcal{C}$, a set of morphisms $\mathcal{C}(X, Y)$ ("boxes with edges coloured $X$ and $Y$"),

together with composition maps

$$\circ: \mathcal{C}(Y, Z) \times \mathcal{C}(X, Y) \to \mathcal{C}(X, Z)$$

such that

- composition is associative (don’t worry about parentheses),
- each $X \in \text{Ob} \mathcal{C}$ has a identity $1_X$ such that

$$1_Y \circ f = f = f \circ 1_X \quad \text{for all } f \in \mathcal{C}(X, Y).$$
Recall our example

\[
C(m, n) = \{ f : X_m \to X_n \} \quad \text{and} \quad C = \bigcup_{m,n=0}^{\infty} C(m, n).
\]

This is a category with

- **Objects:** \(X_n, \ n \geq 0\)
- For \(X_n, X_m \in \text{Ob} \ C\), the set of **morphisms** from \(X_m\) to \(X_n\) is \(C(m, n)\).

Composition is usual composition of set maps.

- Composition is associative.
- The identity \(1_{X_n} : X_n \to X_n\) is the identity map.
Categories

Example (Sets)
- **Objects**: sets
- **Morphisms**: set maps

Example (Vector spaces)
- **Objects**: real vector spaces
- **Morphisms**: linear maps

Example (Groups)
- **Objects**: groups
- **Morphisms**: group homomorphisms
Categories

Example (Rings)
- **Objects**: rings
- **Morphisms**: ring homomorphisms

Example (Topological spaces)
- **Objects**: topological spaces
- **Morphisms**: continuous maps

Other examples
- modules over a fixed ring
- smooth manifolds
- algebraic varieties
- ...
Stuck in one dimension

So far, everything is one-dimensional. We compose blocks in a line:

```
  a  d  b  b  c  a  d  b  e  f
```

Or string diagrams:

```

```

From now on, we’ll rotate these pictures clockwise 90°:

```

Relations:

```

```

But we’re still stuck in one dimension—only have vertical composition \( \circ \).
Moving to two dimensions

Let’s introduce a new horizontal associative composition $\otimes$!

$$
\begin{array}{ccc}
\begin{array}{c}
e \\
b \\
d \\
a
\end{array} & \otimes & 
\begin{array}{c}
f \\
e \\
d \\
a
\end{array} & = & 
\begin{array}{c}
e \\
b \\
d \\
a
\end{array} & \otimes & 
\begin{array}{c}
f \\
e \\
d \\
a
\end{array}
\end{array}
$$

(1)

How should we interpret the right-hand side? We have objects blue, red, green, yellow, etc.

However, (1) seems to be a morphism

blue-red $\rightarrow$ green-yellow.

So we need a way of “combining colours”.

Precisely, we want an associative map

$$\otimes : \text{Ob} \mathcal{C} \times \text{Ob} \mathcal{C} \rightarrow \text{Ob} \mathcal{C}.$$
Moving to two dimensions

We want the associative map

$$\otimes : \text{Ob} \mathcal{C} \times \text{Ob} \mathcal{C} \to \text{Ob} \mathcal{C}$$

to have an identity. Precisely, we want an identity object $1$ such that

$$1 \otimes X = X = X \otimes 1 \quad \text{for all } X \in \text{Ob} \mathcal{C}.$$ 

Let’s denote this identity object by a dashed edge, e.g.

$$\begin{bmatrix} a \end{bmatrix} : 1 \to \text{red}$$

The identity object also has an identity morphism $1_1 : 1 \to 1$:

$$\begin{bmatrix} 1_1 \end{bmatrix} = \begin{bmatrix} a \end{bmatrix}, \quad \begin{bmatrix} \overline{1_1} \end{bmatrix} = \begin{bmatrix} \overline{b} \end{bmatrix}$$
Moving to two dimensions

We want $1_1$ to also be an identity for horizontal composition:

$$
\begin{array}{ccc}
1_1 & a & = & a & = & a & 1_1
\end{array}
$$

For technical reasons, we also want the interchange relation:

$$
\begin{array}{ccc}
a & 1 & = & a & b & = & 1 & b
\end{array}
$$

You can think of identities a bit like an empty slot in a slide puzzle:
We can now impose two-dimensional relations:

\[
\begin{array}{c}
\frac{a}{c} \quad \frac{b}{d} = \frac{e}{g} \quad \frac{f}{h},
\end{array}
\]

\[
\begin{array}{c}
\frac{a}{c} \quad \frac{b}{d} = 1,
\end{array}
\]

etc.

We can use these relations anywhere in a wall of blocks:
### Making things precise

#### Strict monoidal category

A **strict monoidal category** is a category $C$ equipped with

- a **bifunctor** (the **tensor product**) $\otimes: C \times C \to C$, and
- a **unit object** $1$,

such that, for all objects $A$, $B$, $C$,

- $(A \otimes B) \otimes C = A \otimes (B \otimes C)$ for all objects $A$, $B$, $C$,
- $1 \otimes A = A = A \otimes 1$ for all objects $A$,

and, for all morphisms $a$, $b$, $c$,

- $(a \otimes b) \otimes c = a \otimes (b \otimes c)$,
- $1_1 \otimes a = a = 1_1 \otimes a$.

- $\circ$: **vertical composition**
- $\otimes$: **horizontal composition**
String diagrams

We will denote a morphism $f : A \rightarrow B$ by:

![String diagram for $f : A \rightarrow B$]

The identity map $1_A : A \rightarrow A$ is a string with no label:

![String diagram for the identity map $1_A$]

We sometimes omit the object labels when they are clear or unimportant.
String diagrams

Composition is *vertical stacking* and tensor product is *horizontal juxtaposition*:

\[
\begin{align*}
\circ & = f \circ g \\
\otimes & = f \otimes g
\end{align*}
\]

The *interchange law* then becomes:

\[
\begin{align*}
\circ & = f \circ g \\
\otimes & = f \otimes g
\end{align*}
\]

A morphism \( f: A_1 \otimes A_2 \to B_1 \otimes B_2 \) can be depicted:
Presentations of strict monoidal categories

One can give presentations of strict monoidal categories, just as for monoids, groups, etc.

**Objects:** If the objects are generated by some collection \( A_i, i \in I \), then we have all possible tensor products of these objects:

\[
1, \quad A_i, \quad A_i \otimes A_j \otimes A_k \otimes A_\ell, \quad \text{etc.}
\]

**Morphisms:** If the morphisms are generated by some collection \( f_j, j \in J \), then we have all possible compositions and tensor products of these morphisms (whenever these make sense):

\[
1_{A_i}, \quad f_j \otimes (f_i f_k) \otimes (f_\ell), \quad \text{etc.}
\]

We then often impose some relations on these morphism spaces.

**String diagrams:** We can build complex diagrams out of our simple generating diagrams.
Recall the “one-dimensional” symmetric group $\mathcal{S}_n$

We needed $n - 1$ generators $s_i = \uparrow \cdots \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \uparrow \cdots \uparrow$,

the $n - 1$ quadratic relations

$$\uparrow \cdots \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \uparrow \cdots \uparrow = \uparrow \cdots \uparrow,$$

the $n - 2$ braid relations

$$\uparrow \cdots \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \uparrow \cdots \uparrow = \uparrow \cdots \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \uparrow \cdots \uparrow,$$

and the (order $n^2$) distant braid relations

$$\begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \vdash \uparrow \cdots \vdash \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} = \uparrow \vdash \uparrow \begin{array}{c} \text{(i+1)} \\ \text{(i)} \end{array} \uparrow \vdash \uparrow \vdash \uparrow \vdash \uparrow.$$

Recall the “one-dimensional” symmetric group $S_n$

Too many generators!!

Many of the relations are just “shifts” of each other, e.g.

$$
\begin{array}{c}
\uparrow \\
\cdots \\
\uparrow \\
\cdots \\
\uparrow \\
\end{array} \\
\begin{array}{c}
i+1 \\
i \\
i+1 \\
i \\
i+1 \\
i \\
\end{array} = \\
\begin{array}{c}
\uparrow \\
\cdots \\
\uparrow \\
\end{array}.
$$

If we can compose horizontally, we just need one generator

and one relation

and similarly for the other relations.
Monoidally generated symmetric groups

Define a strict monoidal category $S$ with one generating object $X$ and denote

\[ 1_X = \uparrow \]

We have one generating morphism

\[ \otimes : X \otimes X \to X \otimes X. \]

We impose the relations:

\[ \begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\uparrow
\end{array}
\end{array}
\end{array}
\end{array} = \begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\uparrow
\end{array}
\end{array}
\end{array}
\end{array}, \quad \begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\otimes
\end{array}
\end{array}
\end{array}
\end{array} = \begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\begin{array}{c}
\otimes
\end{array}
\end{array}
\end{array}
\end{array}. \end{array} \]

Then

\[ S(X^\otimes n, X^\otimes n) = S_n \]

is the symmetric group on $n$ letters.
Monoidally generated symmetric groups

This monoidal presentation of $S_n$ is very efficient! We only needed

- one generating morphism, and
- two relations,

... to get all the symmetric groups.

Note that the distant braid relation follows for free from the interchange law:
Monoidal categories in the wild

Example (Sets I)
- **Objects**: finite sets
- **Morphisms**: set maps
- **Vertical composition**: composition of set maps
- **Horizontal composition**: cartesian product
- **Unit object** $1$: one element set \{\star\}

Example (Sets II)
- **Objects**: finite sets
- **Morphisms**: set maps
- **Vertical composition**: composition of set maps
- **Horizontal composition**: disjoint union
- **Unit object** $1$: empty set $\emptyset$
Monoidal categories in the wild

Example (Vector spaces)

- **Objects**: $\mathbb{R}$-vector spaces
- **Morphisms**: linear transformations
- **Vertical composition**: composition of linear transformations
- **Horizontal composition**: tensor product $\otimes$
- **Unit object** $\mathbf{1}$: $\mathbb{R}$

Example (Abelian groups)

- **Objects**: abelian groups
- **Morphisms**: group homomorphisms
- **Vertical composition**: composition of group homomorphisms
- **Horizontal composition**: tensor product $\otimes_{\mathbb{Z}}$
- **Unit object** $\mathbf{1}$: $\mathbb{Z}$
Example (Rings)

- **Objects**: rings
- **Morphisms**: ring homomorphisms
- **Vertical composition**: composition of ring homomorphisms
- **Horizontal composition**: tensor product $\otimes\mathbb{Z}$
- **Unit object** $1$: $\mathbb{Z}$

Technicality

The above examples are not really **strict** monoidal categories. E.g. for sets $X$, $Y$, $Z$, we don’t have $(X \times Y) \times Z = X \times (Y \times Z)$.

However, we have a **natural isomorphism**

$$(X \times Y) \times Z \cong X \times (Y \times Z).$$

So the above are **strict** monoidal categories. Every monoidal category is equivalent to a strict one.
More colours please!

Recall our horizontal composition

\[
\begin{array}{ccc}
  e & f & e \\
  b & e & b \\
  d & d & d \\
  a & c & a
\end{array}
\]

Why are only the horizontal edges coloured?

Indeed, we can colour the vertical edges as well:

\[
\begin{array}{ccc}
  a \\
\end{array}
\]

Now we can only compose horizontally when colours match (just like for vertical composition)!
# 2-categories

## 2-category

- objects (vertical edges of boxes),
- 1-morphisms (horizontal edges of boxes),
- 2-morphisms (boxes).

## Example

Bimodules over rings

- **Objects**: rings
- **1-Morphisms**: bimodules
- **2-morphisms**: bimodule homomorphisms.

A monoidal category is a 2-category with one object (one vertical edge colour):

- 1-morphisms of 2-category $\rightsquigarrow$ objects of monoidal category
- 2-morphisms of 2-category $\rightsquigarrow$ morphisms of monoidal category