What can abstract mathematics tell us about programming climate models?

Dominic Orchard

UNIVERSITY OF CAMBRIDGE

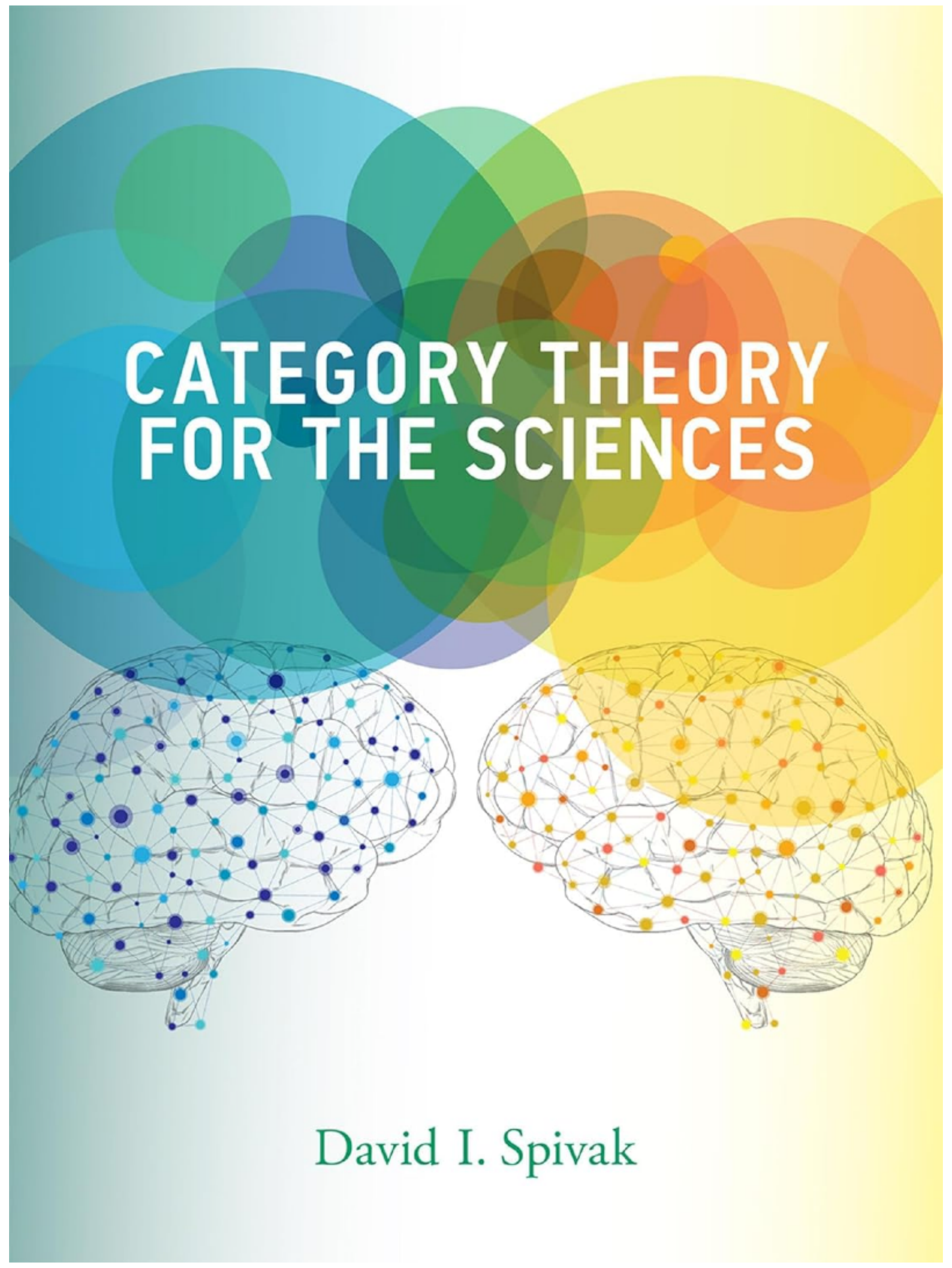
Institute of Computing for Climate Science

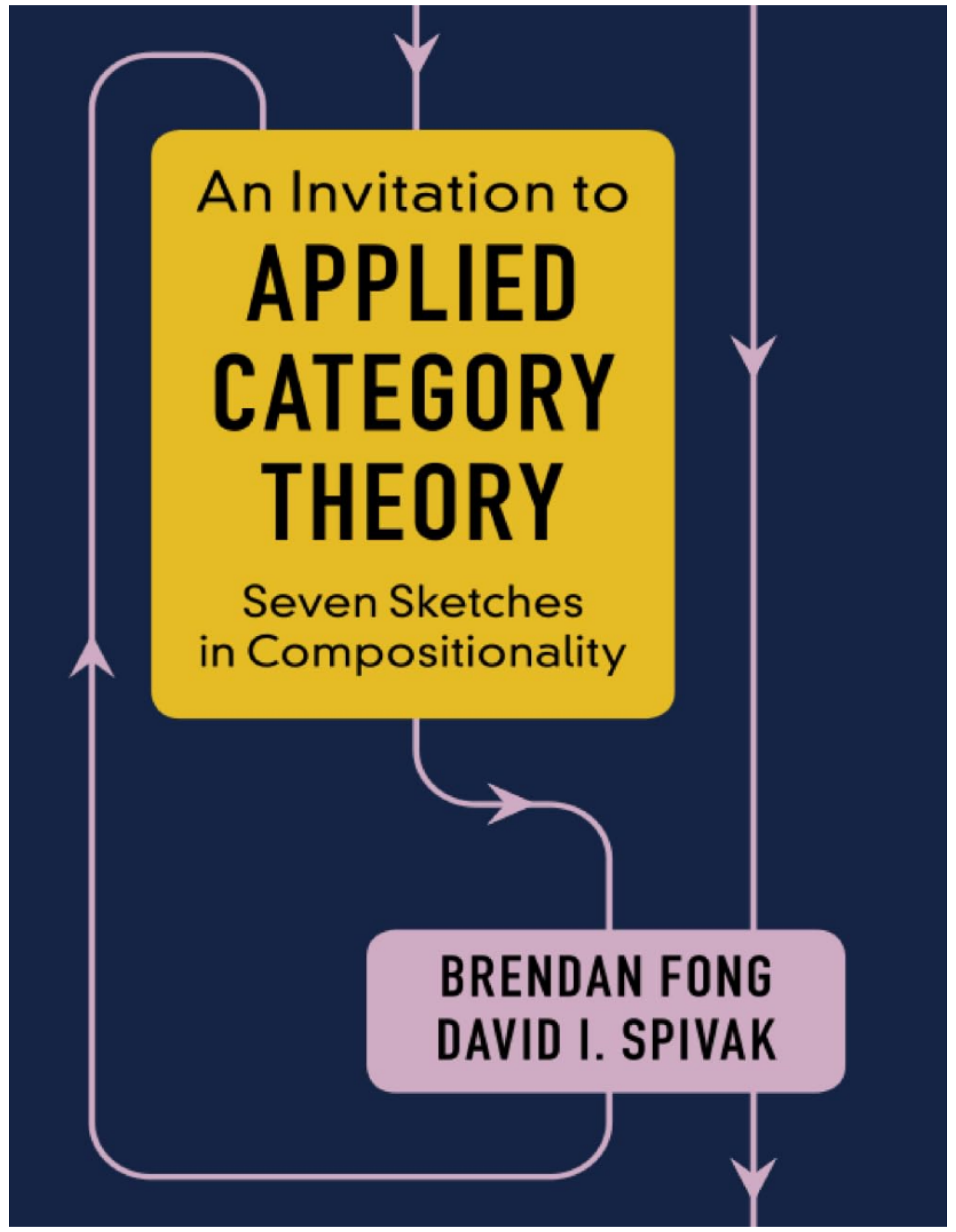
ICCS Summer School 2024, Friday 12th July

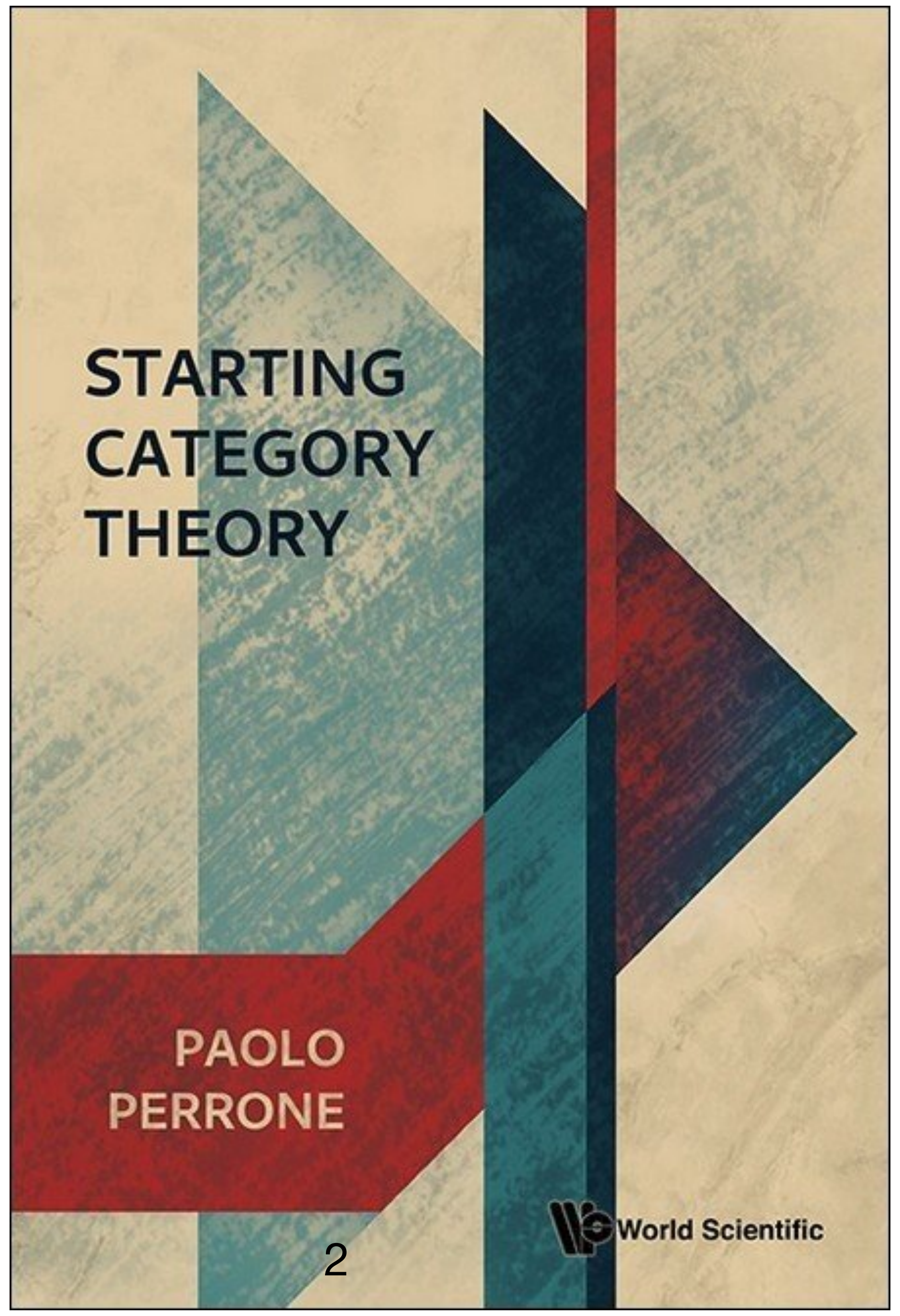
Category theory

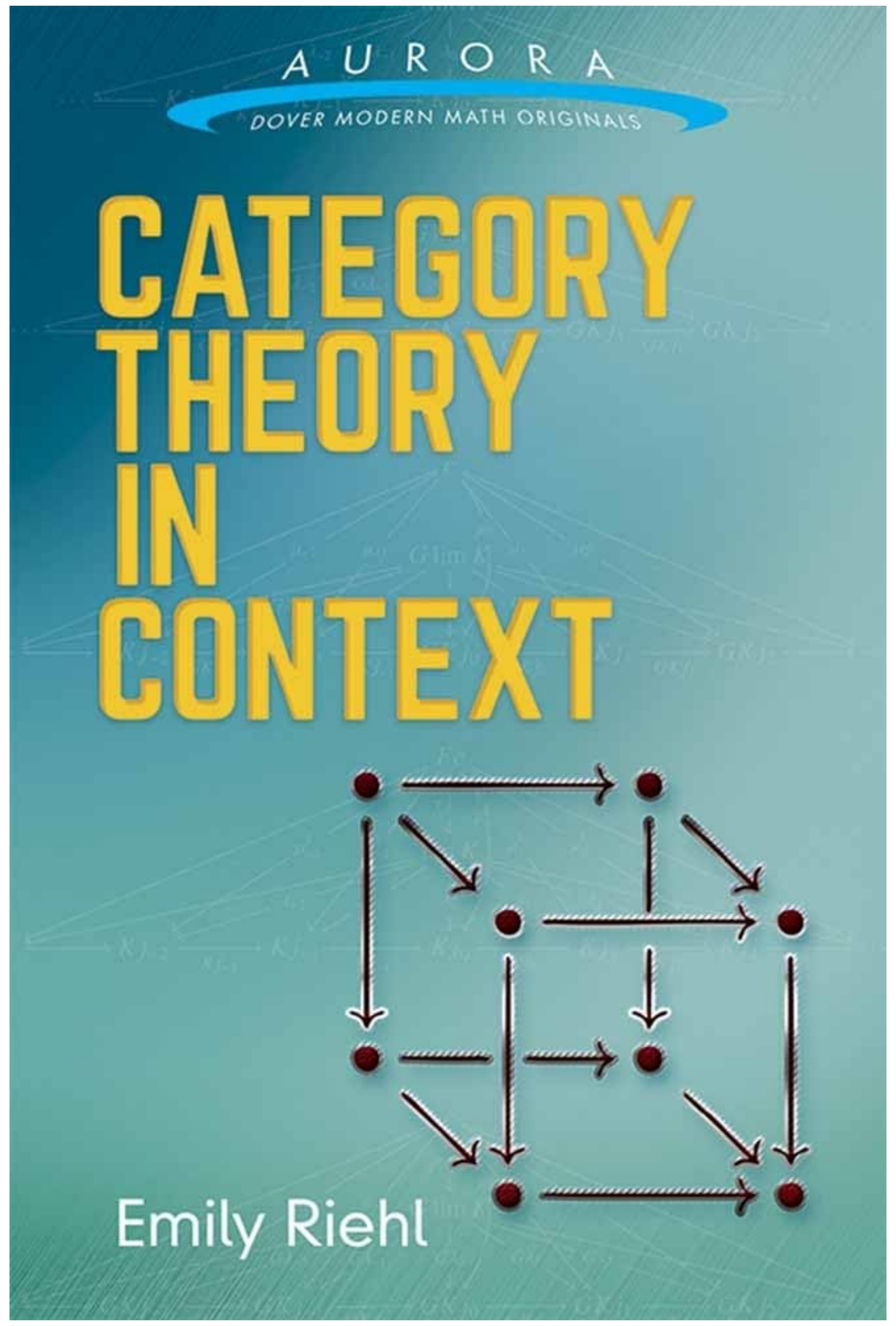
A branch of abstract mathematics

Provides organising principles exposing mathematical structure and structure within mathematics









Today's objectives

View some common ideas in numerical programming through the category theory lens

- Understand what is a category, functor, and natural transformation
- Understand their relationship to particular programming patterns
- Use these basic categorical concepts to inform **testing** and **optimisation** strategies
- Understand these basics as a launch pad

Categories

- Collection of objects obj($\mathscr{C}$)
    - often denoted $A, B, C, \ldots \in$ obj($\mathscr{C}$)
- Collection of "morphisms" (relationships) between pairs of objects morph($\mathscr{C}$)
    - often denoted $f : A \to B, g : B \to C, \ldots \in$ morph($\mathscr{C}$)
- Composition operation

$$\frac{f : A \to B \qquad g : B \to C}{f; g : A \to C}$$

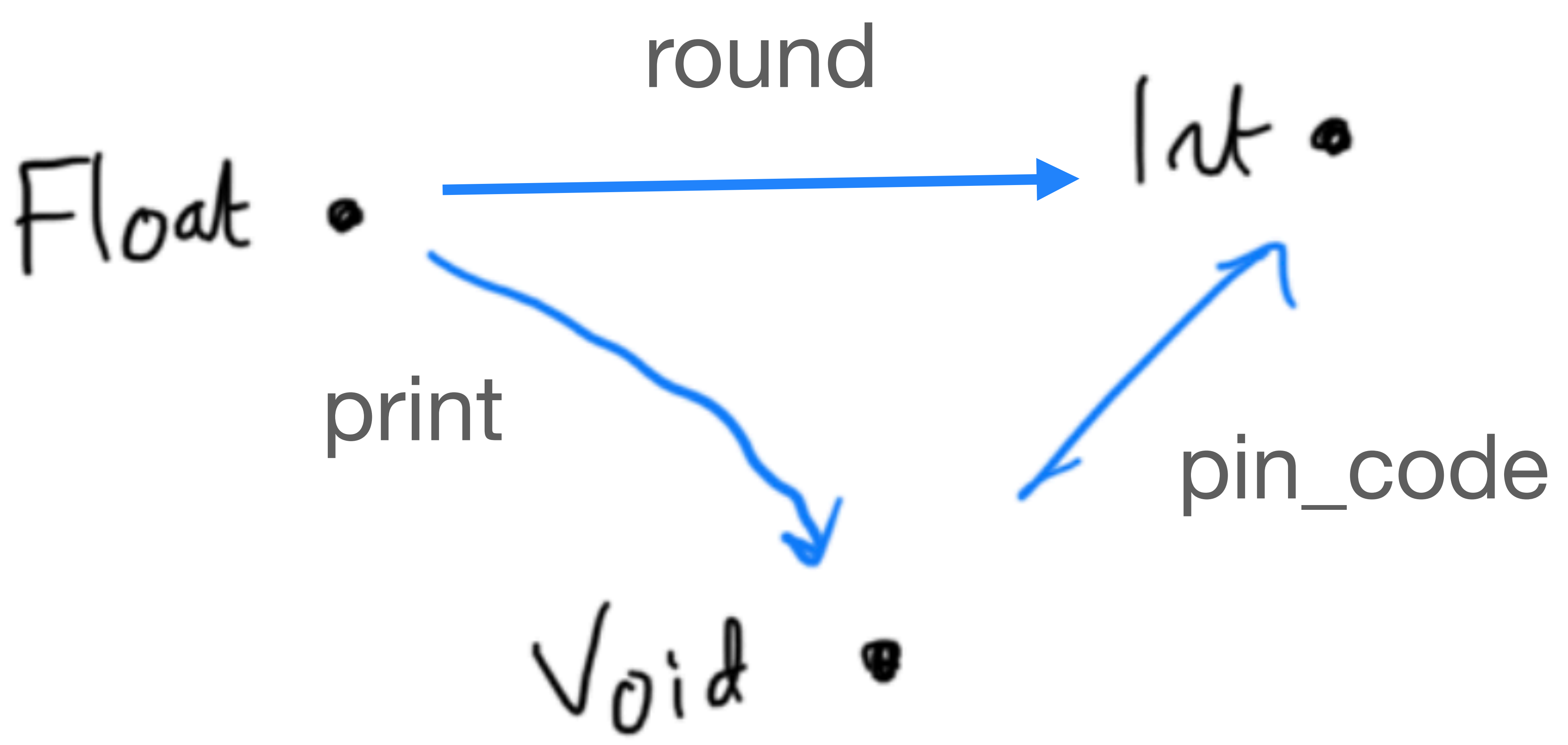
- Identity morphisms

$$\forall A \in \mathrm{obj}(\mathscr{C}) \,.\, id_A : A \to A$$

- Axioms: $f; id = id; f = f$ $\quad$ $f; (g; h) = (f; g); h$

Categories in programming

Types as objects, functions as morphisms



category.py

Functors

A morphism <u>between</u> categories

$\mathsf{F} : \mathscr{C} \to \mathscr{D}$

Has two components:

(1) mapping on objects: $\forall A \in \mathrm{obj}(\mathscr{C}) \implies \mathsf{F}(A) \in \mathrm{obj}(\mathscr{D})$

(2) mapping on morphisms:

$\forall (f : A \to B) \in \mathrm{morph}(\mathscr{C}) \implies \mathsf{F}(f) : \mathsf{F}(A) \to \mathsf{F}(B) \in \mathrm{morph}(\mathscr{D})$

Has two axioms:

$$\mathsf{F}(\mathrm{id}_A) = \mathrm{id}_{\mathsf{FA}}$$

$$\mathsf{F}(f; g) = \mathsf{F}(f); \mathsf{F}(g)$$

Functors in programming

Object mapping is a type constructor

```
list : Type -> Type
```

Morphism mapping lifts a function to apply over the type constructor

$$\frac{A \to B}{\text{list}(A) \to \text{list}(B)}$$

Natural transformations

A morphism <u>between</u> functors

$\mathsf{F} : \mathscr{C} \to \mathscr{D}$ $\qquad\qquad$ $\mathsf{G} : \mathscr{C} \to \mathscr{D}$

A natural transformation $\alpha : F \to G$ is a ***family of morphisms***

meaning $\forall A \in \mathrm{obj}(\mathscr{C})\,.\ \alpha_A : \mathsf{F}A \to \mathsf{G}A \in \mathrm{morph}(\mathscr{D})$

along with the **naturality property**

$$\begin{tikzcd}
\mathsf{F}A \arrow[r, "\alpha_A"] \arrow[d, "\mathsf{F}f"'] & \mathsf{G}A \arrow[d, "\mathsf{G}f"] \\
\mathsf{F}B \arrow[r, "\alpha_B"'] & \mathsf{G}B
\end{tikzcd}$$

Natural transformations in programming

nat_trans.py

Connecting back to climate modelling…

- Functors represent pointwise traversals (common transformation)
- Structural transformations are natural transformations
- Stencil computations are a generalisation to functors….

Comonads (*briefly and incompletely*)

An endofunctor $\mathsf{D} : \mathscr{C} \to \mathscr{C}$ **with:**

- Natural transformation $\varepsilon_A : DA \to A$
- For every morphism, a lifting $(-)^\dagger$

$$\frac{f : DA \to B}{f^\dagger : DA \to DB}$$

cf.

Functorial mapping

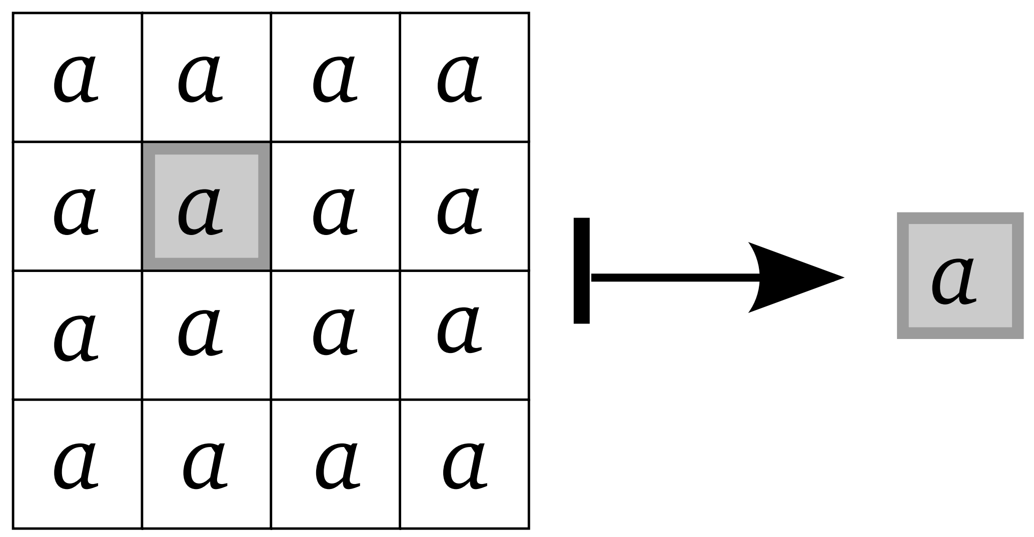
$$\frac{f : A \to B}{\mathsf{F}f : DA \to DB}$$

Comonads in programming

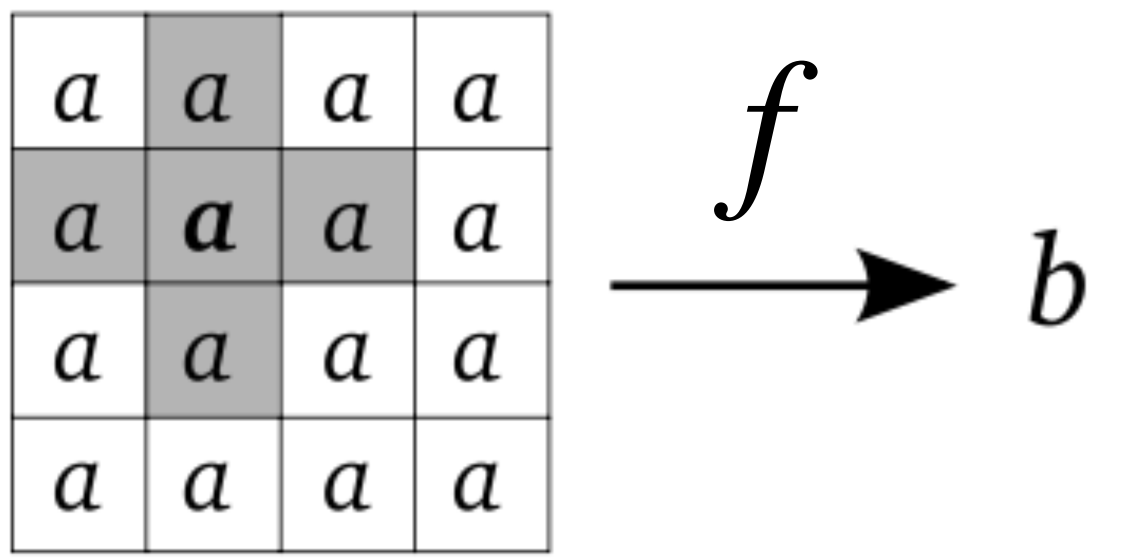
Array comonad

$$DA = \text{Array } I\ A \times I$$

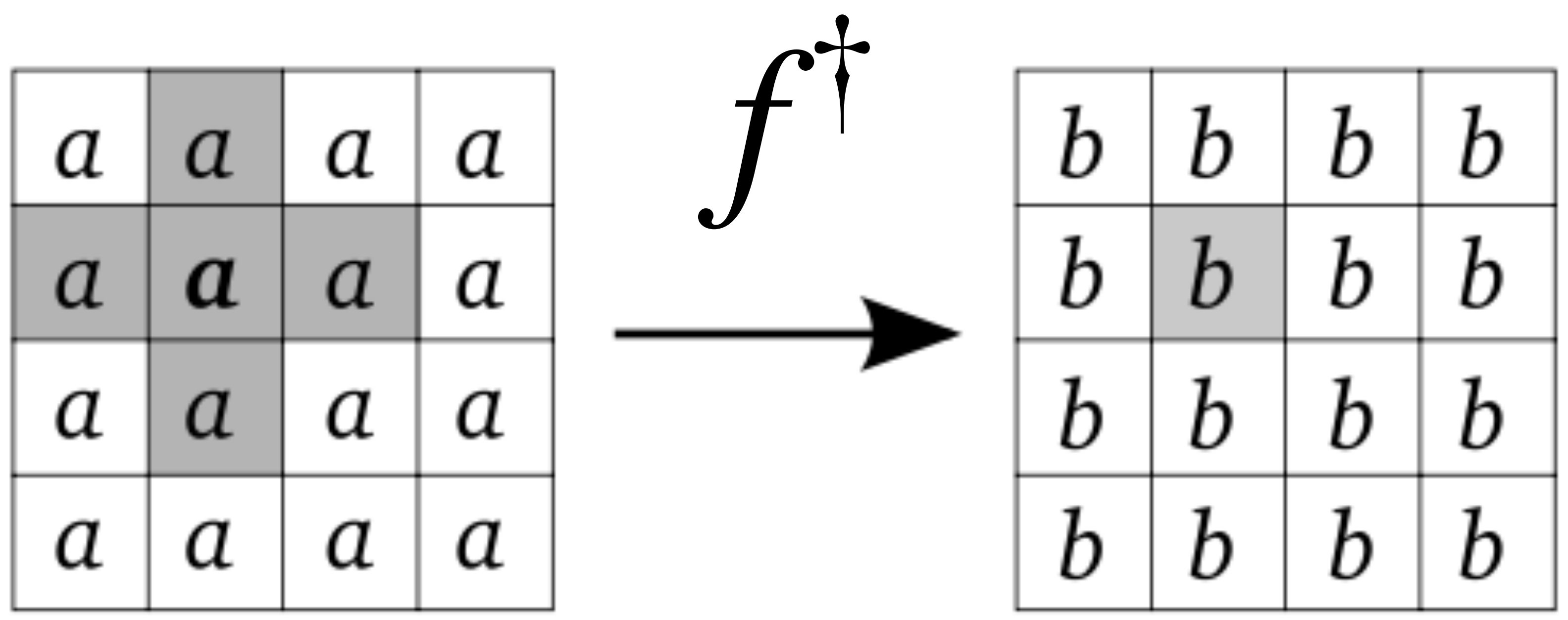
array-data × cursor



$$\varepsilon_A : DA \to A$$



Local computation (neighbourhood)



Global computation

Ypnos: declarative, parallel structured grid programming (Orchard et al. 2010)
A Categorical Outlook on Cellular Automata (Capobianco, Uustalu, 2010)

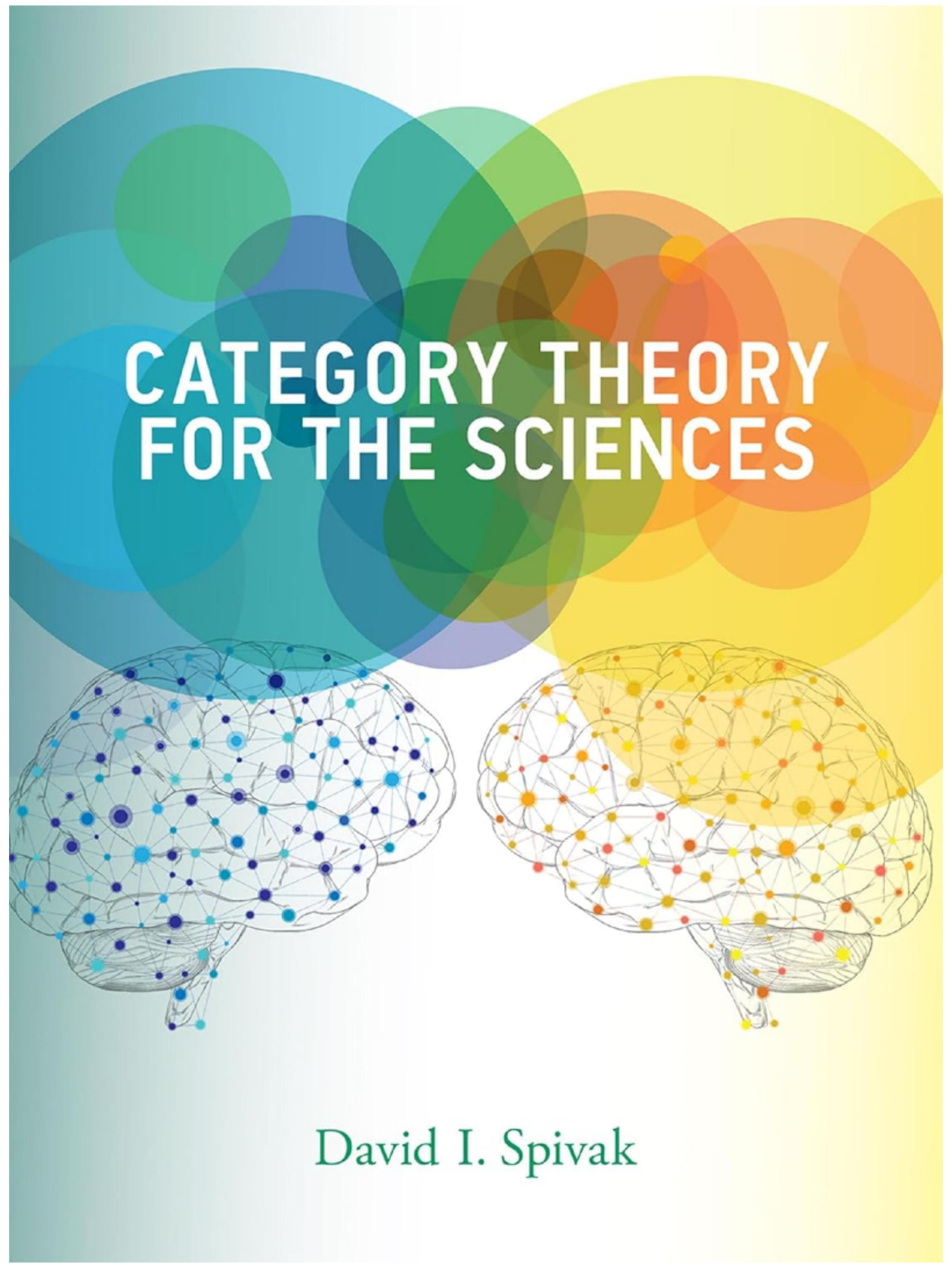
12

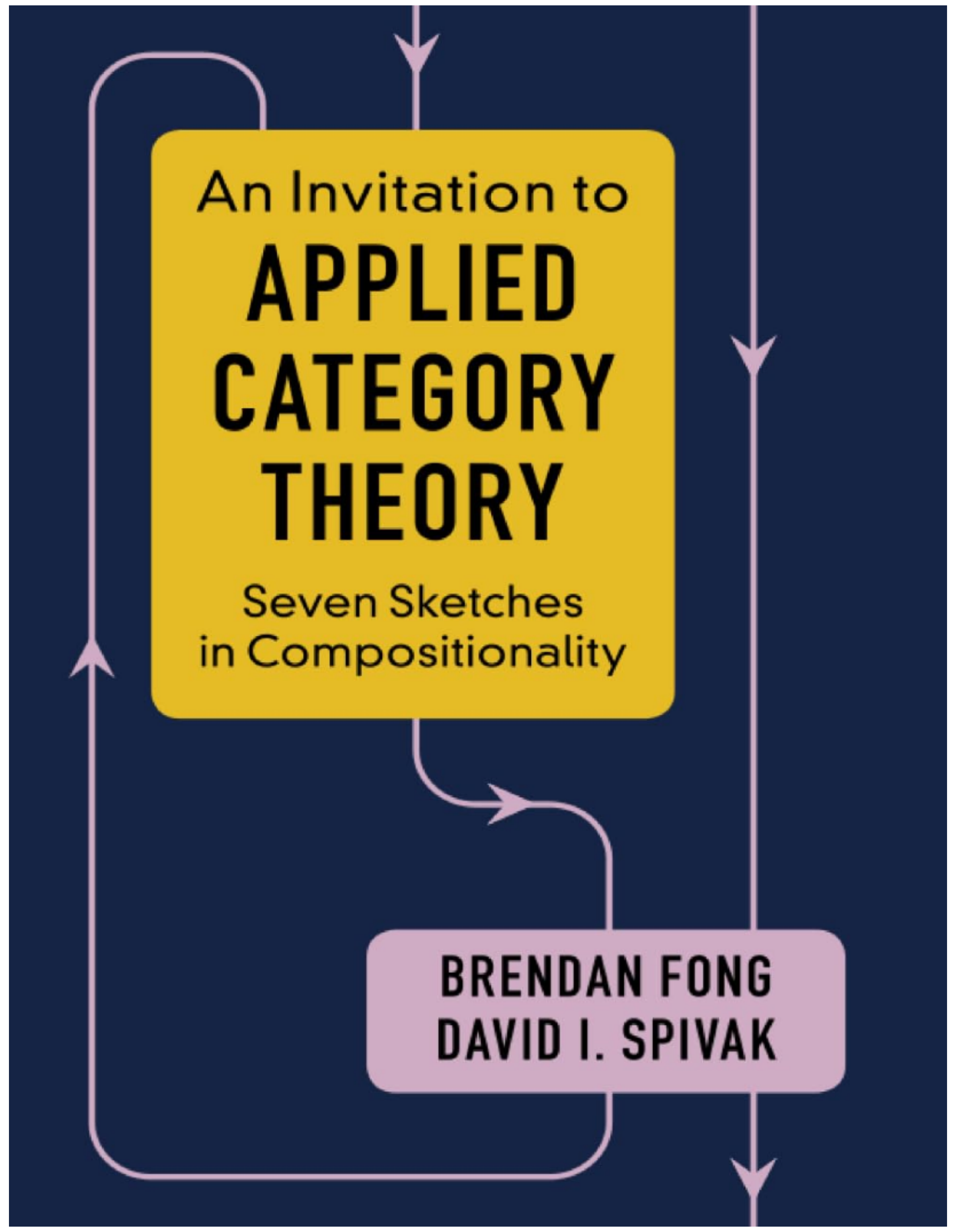
Today's objectives

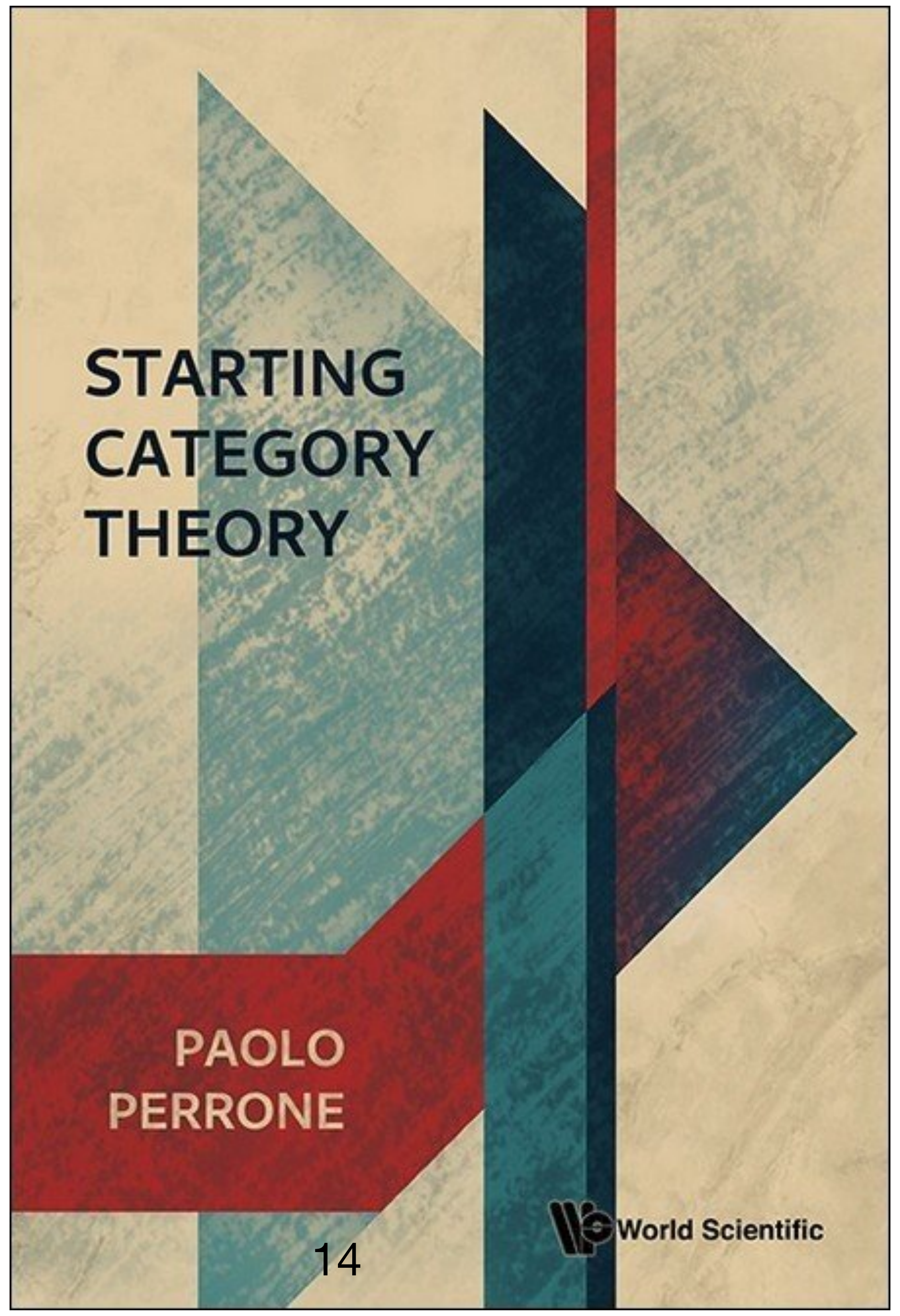
View some common ideas in numerical programming through the category theory lens

- Understand what is a <u>category,</u> <u>functor,</u> and <u>natural transformation</u>
- Understand their relationship to particular programming patterns
- Use these basic categorical concepts to inform **testing** and **optimisation** strategies
- Understand these basics as a launch pad

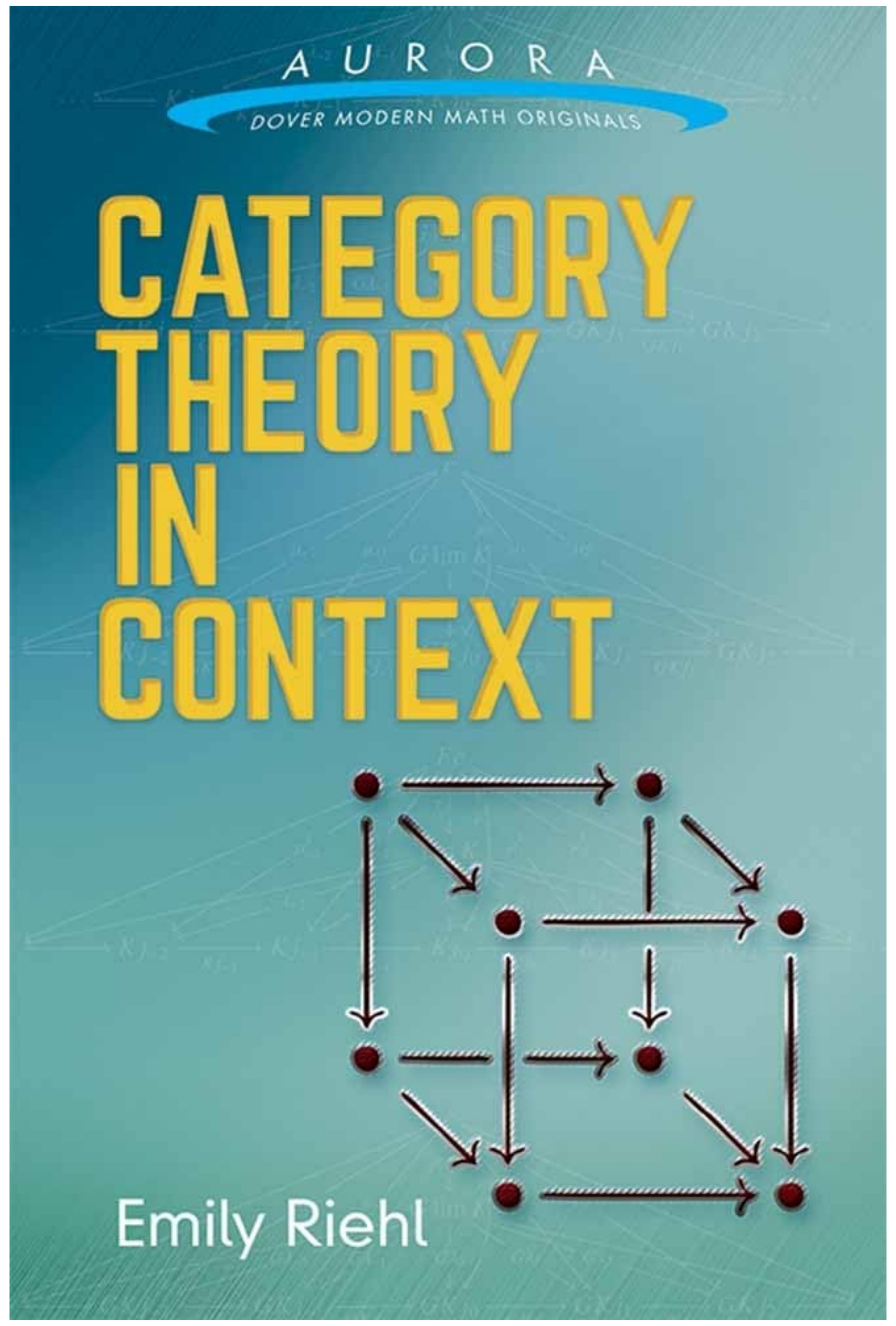
Thanks!


CATEGORY THEORY
FOR THE SCIENCES
David I. Spivak


An Invitation to
APPLIED
CATEGORY
THEORY
Seven Sketches
in Compositionality
BRENDAN FONG
DAVID I. SPIVAK


STARTING
CATEGORY
THEORY
PAOLO
PERRONE
World Scientific


CATEGORY THEORY
FOR PROGRAMMERS
Bartosz Milewski


AURORA
DOVER MODERN MATH ORIGINALS
CATEGORY
THEORY
IN
CONTEXT
Emily Riehl