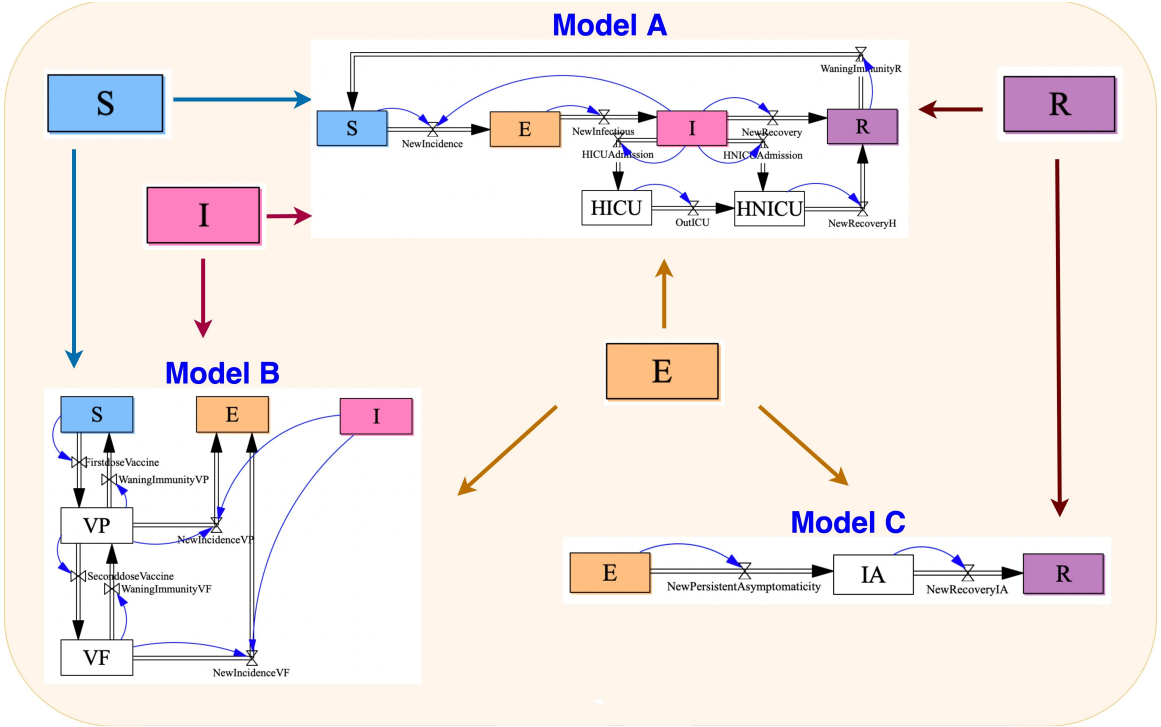
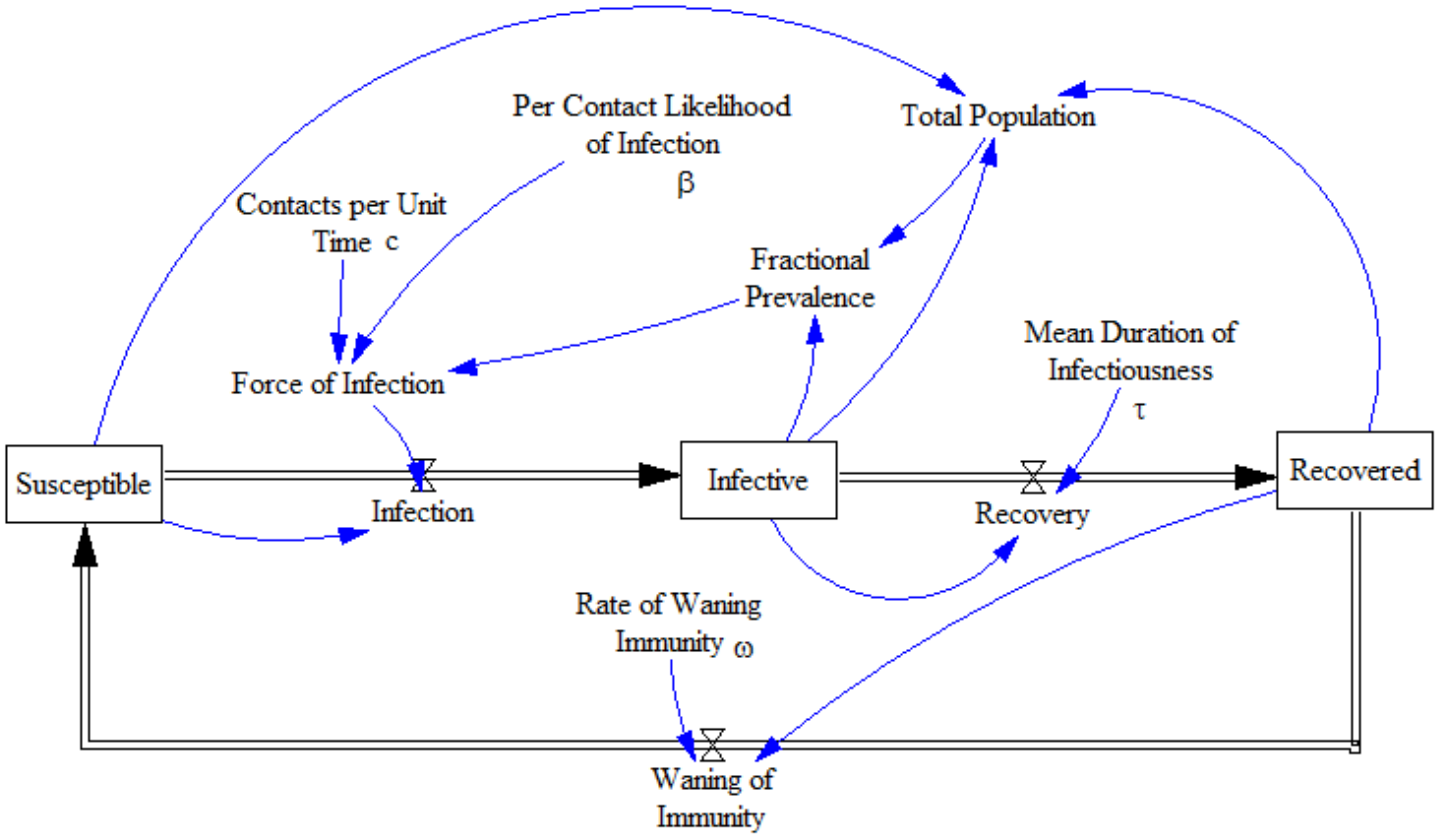
CATEGORY THEORY IN EPIDEMIOLOGY



John Baez

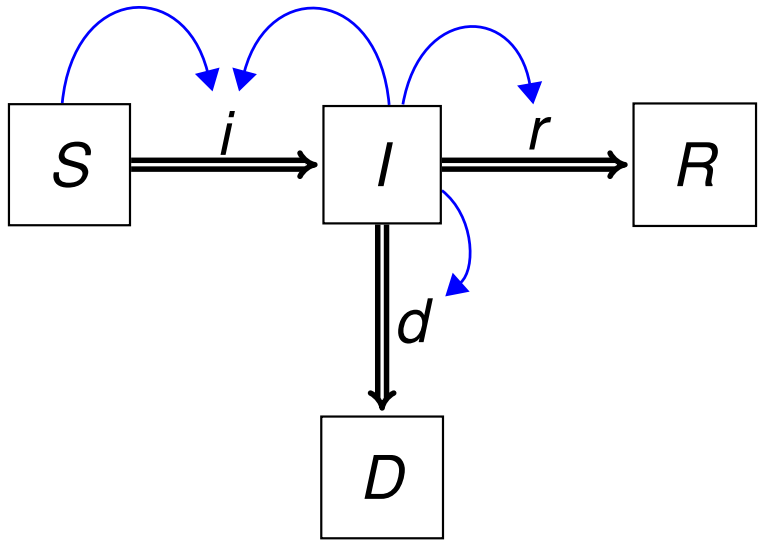
Edinburgh Mathematical Society
2023 December 8

In "System Dynamics", dynamical systems are modeled using "stock-flow diagrams":



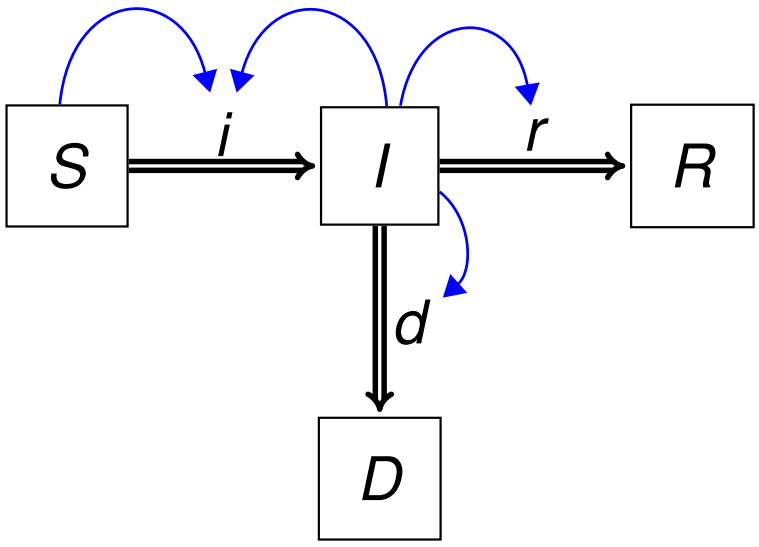
These diagrams are now widely used in economics, population biology, epidemiology, etc.

Here is a simple example of a stock-flow diagram:



The boxes are **stocks**, the double arrows are **flows**, and the blue arrows are **links** from stocks to flows.

If we equip each flow with a **flow function**, a stock-flow diagram gives differential equations describing how the stocks change with time.



$$\frac{dS}{dt} = -\phi_i(S, I)$$

$$\frac{dI}{dt} = \phi_i(S, I) - \phi_r(I) - \phi_d(I)$$

$$\frac{dR}{dt} = \phi_r(I)$$

$$\frac{dD}{dt} = \phi_d(I)$$

Why don’t we just write down the differential equations?

Why don't we just write down the differential equations?

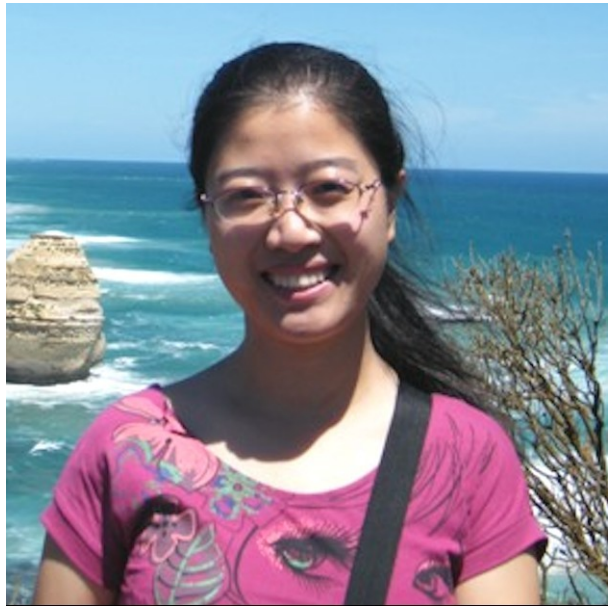
Shockingly, most people find it easier to understand diagrams than differential equations!

In "community-based modeling", diagrams help experts work with community members to build models of the problems they face.

Sociologists would say the power of diagrams is that they're "boundary objects":

> *A **boundary object** is any object that is part of multiple social worlds and facilitates communication between them; it has a different identity in each social world that it inhabits.*

There is a community of epidemiologists who use stock-flow diagrams to model the spread of disease. This includes my coauthors Nate Osgood and Xiaoyan Li, who did COVID modeling for the government of Canada.



Most stock-flow modeling is done using software called AnyLogic. It's powerful, but it has several big problems:

- It has no support for *"composing"* models: that is, taking several smaller models and putting them together to form a larger model.

Most stock-flow modeling is done using software called AnyLogic. It's powerful, but it has several big problems:

- It has no support for *"composing"* models: that is, taking several smaller models and putting them together to form a larger model.
- It thus has no support for *collaboratively* building models.

Most stock-flow modeling is done using software called AnyLogic. It's powerful, but it has several big problems:

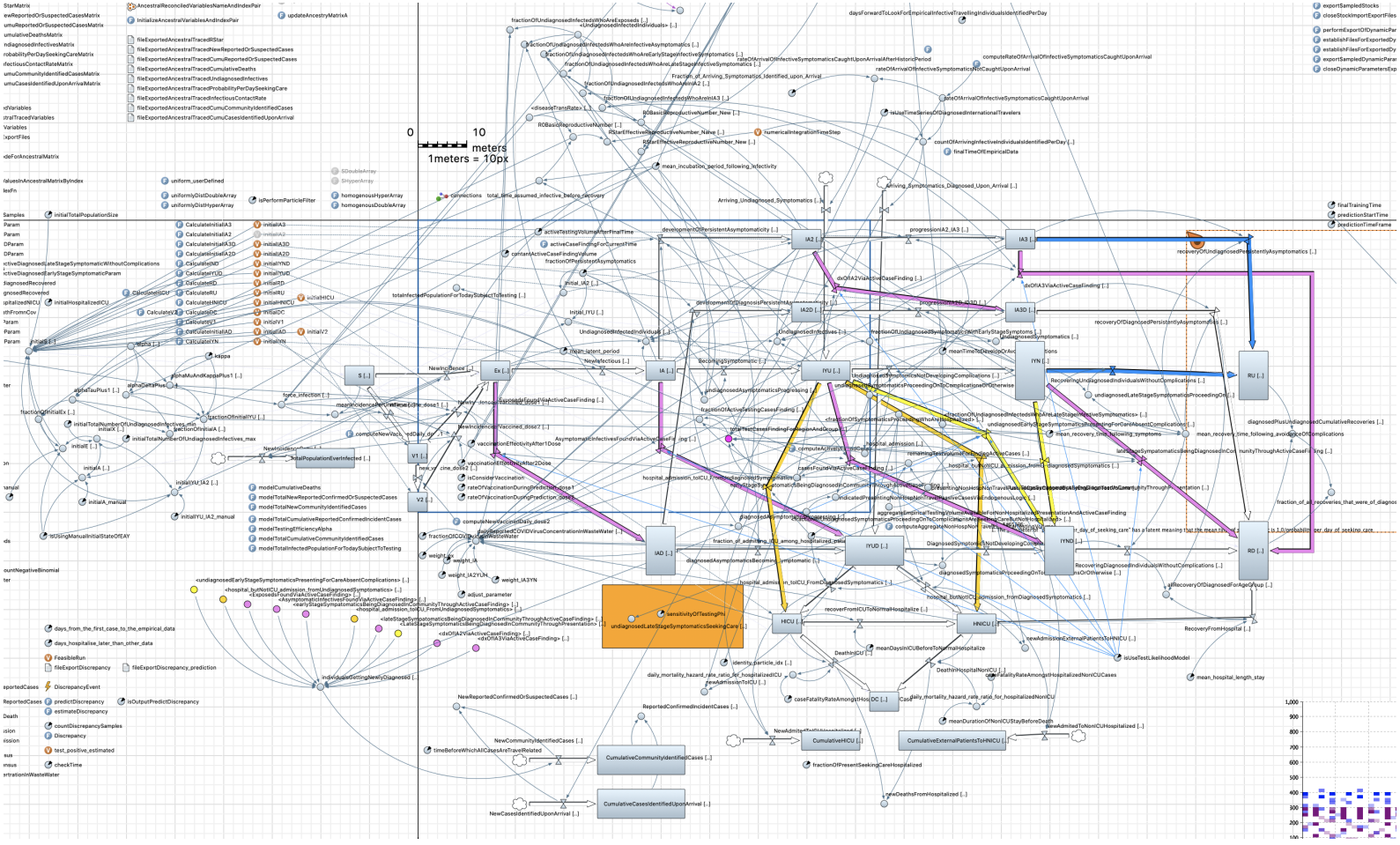
- It has no support for *"composing"* models: that is, taking several smaller models and putting them together to form a larger model.
- It thus has no support for *collaboratively* building models.
- It is not *free* and not *open-source!*

Most stock-flow modeling is done using software called AnyLogic. It's powerful, but it has several big problems:

- It has no support for *"composing"* models: that is, taking several smaller models and putting them together to form a larger model.
- It thus has no support for *collaboratively* building models.
- It is not *free* and not *open-source!*

Our new work aims to fix these problems.

The ability to *compose* models is crucial because realistic models are complicated and built out of many smaller parts. Here is Osgood and Li's COVID model used by the government of Canada:



For compositional modeling we use an idea going back to Bill Lawvere's 1963 thesis: "functorial semantics".

For compositional modeling we use an idea going back to Bill Lawvere's 1963 thesis: "functorial semantics".

In traditional applications to computer science, this says roughly:

- There is a category C whose objects are data types and morphisms are programs.

For compositional modeling we use an idea going back to Bill Lawvere's 1963 thesis: "functorial semantics".

In traditional applications to computer science, this says roughly:

- There is a category C whose objects are data types and morphisms are programs.
- There is a category D whose objects are sets and whose morphisms are (partially defined) functions.

For compositional modeling we use an idea going back to Bill Lawvere's 1963 thesis: "functorial semantics".

In traditional applications to computer science, this says roughly:

- There is a category C whose objects are data types and morphisms are programs.
- There is a category D whose objects are sets and whose morphisms are (partially defined) functions.
- There is a functor $F : \mathrm{C} \to \mathrm{D}$ sending each data type to the set of data of that type, and sending each program to the function it computes.

We say F maps **syntax** to **semantics**.

In compositional modeling, functorial semantics works a bit differently:

- There is a category C where morphisms are models and *composing* morphisms lets us putting together smaller models to form bigger ones.

In compositional modeling, functorial semantics works a bit differently:

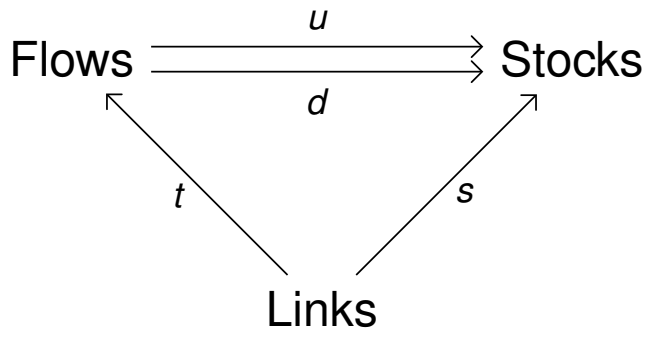
- There is a category C where morphisms are models and *composing* morphisms lets us putting together smaller models to form bigger ones.
- There is a functor D where morphisms are collections of first-order ordinary differential equations.

In compositional modeling, functorial semantics works a bit differently:

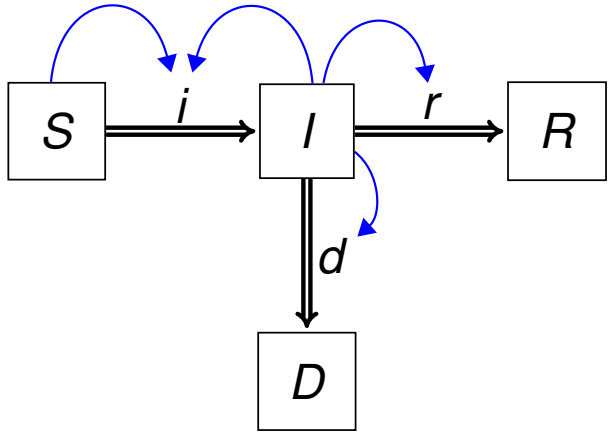
- There is a category C where morphisms are models and *composing* morphisms lets us putting together smaller models to form bigger ones.
- There is a functor D where morphisms are collections of first-order ordinary differential equations.
- There is a functor $F : \mathrm{C} \to \mathrm{D}$ sending each model to the differential equations it describes.

This needs more explanation!

A **stock-flow diagram** consists of finite sets and functions:



together with, for each $f \in$ Flows, a **flow function** $\phi_f : \mathbb{R}^{L(f)} \to \mathbb{R}$ where $L(f)$ is the set of all $\ell \in$ Links with $t(\ell) = f$.

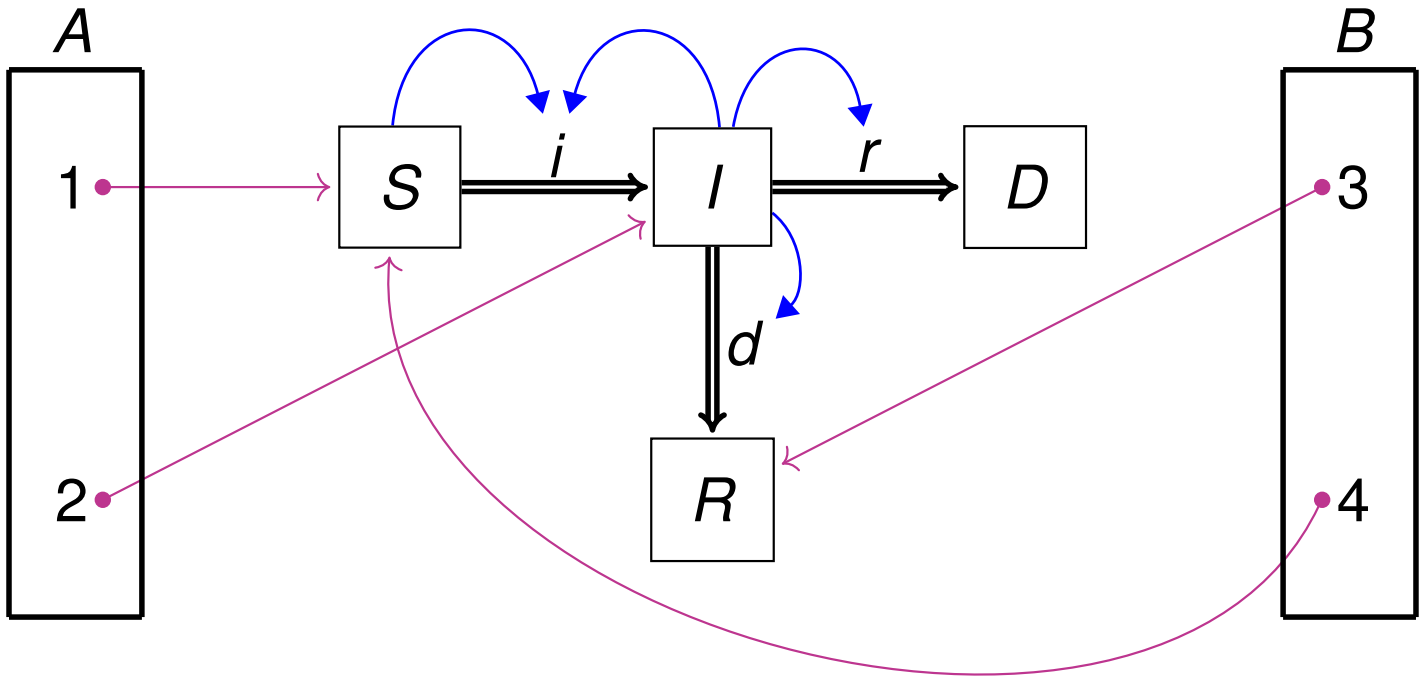


$\phi_i : \mathbb{R}^2 \to \mathbb{R}$ gives $\phi_i(S, I)$

$\phi_r : \mathbb{R} \to \mathbb{R}$ gives $\phi_r(I)$

$\phi_d : \mathbb{R} \to \mathbb{R}$ gives $\phi_d(I)$

An **open stock-flow diagram** is a stock-flow diagram equipped with maps $i\colon A \to \text{Stocks}$, $o\colon B \to \text{Stocks}$ for some finite sets A, B.

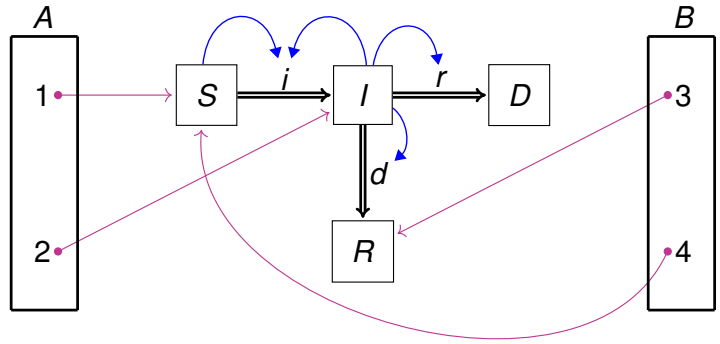


We call this an open stock-flow diagram **from *A* to *B*** and write it as $A \xrightarrow{F} B$.

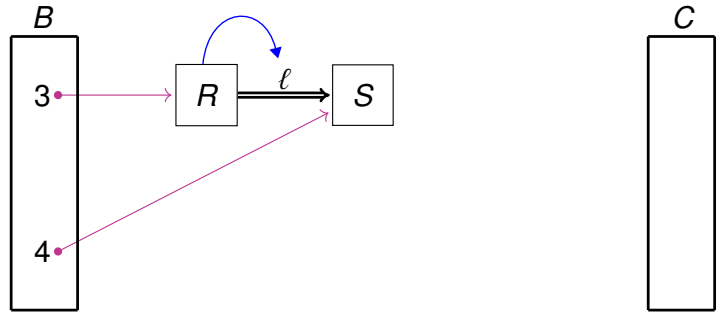
We can **compose** open stock-flow diagrams $A \xrightarrow{F} B$ and $B \xrightarrow{G} C$ by "gluing them together along B".

We get an open stock-flow diagram called $A \xrightarrow{G \circ F} C$.

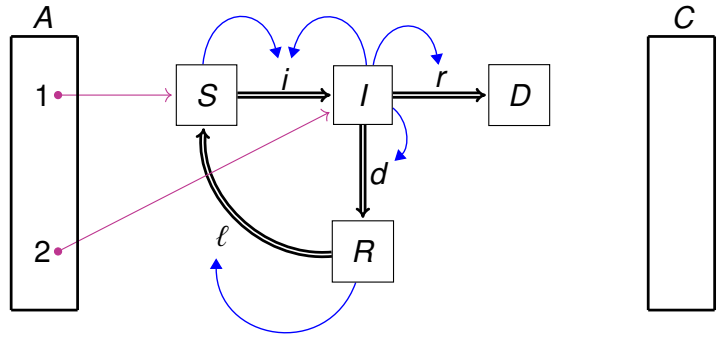
$A \xrightarrow{F} B$



$B \xrightarrow{G} C$



$A \xrightarrow{G \circ F} C$



Thus, we get a category Open(StockFlow) with:

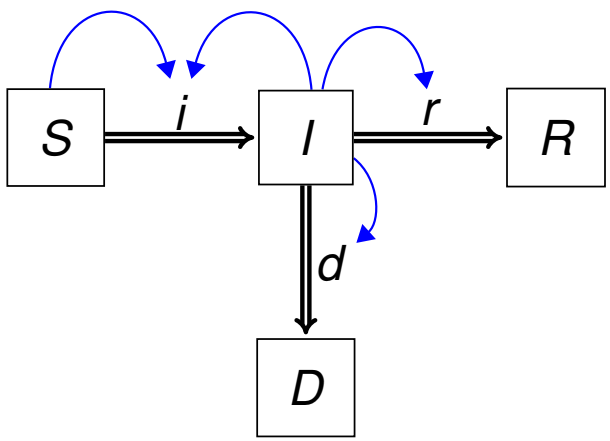
- finite sets as objects,
- open stock-flow diagrams as morphisms.

Next we can construct a category Open(Dynam) of "open dynamical systems", and a functor

$$\Phi : \mathrm{Open}(\mathrm{StockFlow}) \to \mathrm{Open}(\mathrm{Dynam})$$

This turns any open stock-flow diagram into an open dynamical system.

We have already seen how this works without the "openness":



$$\frac{dS}{dt} = -\phi_i(S, I)$$

$$\frac{dI}{dt} = \phi_i(S, I) - \phi_r(I) - \phi_d(I)$$

$$\frac{dR}{dt} = \phi_r(I)$$

$$\frac{dD}{dt} = \phi_d(I)$$

A **dynamical system** on some finite set of variables X is a vector field v on $\mathbb{R}^X$. This lets us write down a system of first-order ordinary differential equations.

For example, if $X = \{S, I, D, R\}$ and v is the 4-component vector field (v_S, v_I, v_D, v_R) on $\mathbb{R}^X \cong \mathbb{R}^4$, we get

$$\begin{aligned}
\frac{d}{dt}S(t) &= v_S(S(t), I(t), D(t), R(t)) \\
\frac{d}{dt}I(t) &= v_I(S(t), I(t), D(t), R(t)) \\
\frac{d}{dt}D(t) &= v_D(S(t), I(t), D(t), R(t)) \\
\frac{d}{dt}R(t) &= v_R(S(t), I(t), D(t), R(t))
\end{aligned}$$

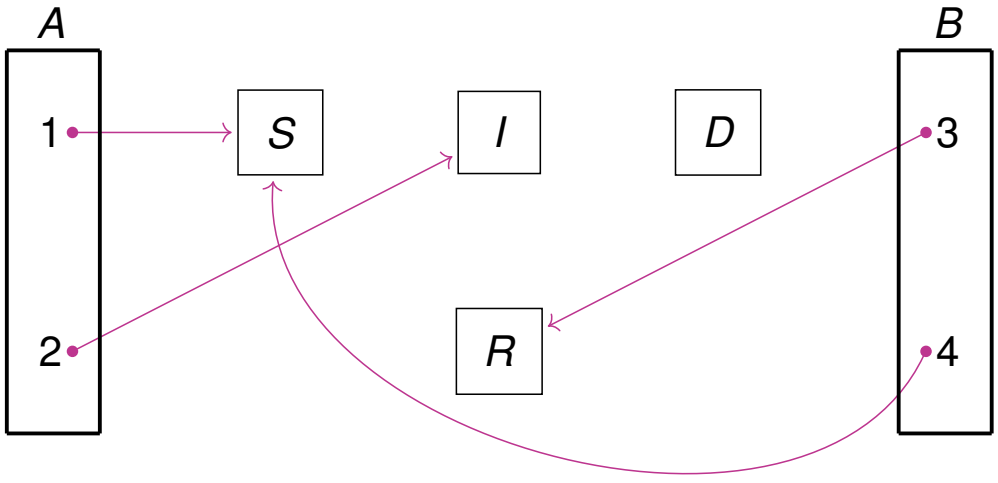
A **dynamical system** on some finite set of variables X is a vector field v on $\mathbb{R}^X$. This lets us write down a system of first-order ordinary differential equations.

For example, if $X = \{S, I, D, R\}$ and v is the 4-component vector field (v_S, v_I, v_D, v_R) on $\mathbb{R}^X \cong \mathbb{R}^4$, we get

$$\begin{aligned} \frac{dS}{dt} &= v_S \\ \frac{dI}{dt} &= v_I \\ \frac{dD}{dt} &= v_D \\ \frac{dR}{dt} &= v_R \end{aligned}$$

An **open dynamical system** $A \xrightarrow{v} B$ is a dynamical system v on some finite set X equipped with maps $i\colon A \to X$, $o\colon B \to X$ for some finite sets A, B.

For example:



$$\frac{dS}{dt} = v_S$$

$$\frac{dI}{dt} = v_I$$

$$\frac{dR}{dt} = v_R$$

$$\frac{dD}{dt} = v_D$$

Here $X = \{S, I, D, R\}$.

Just as we constructed the category Open(StockFlow), we can construct a category Open(Dynam) with:

- finite sets as objects,
- open dynamical systems as morphisms.

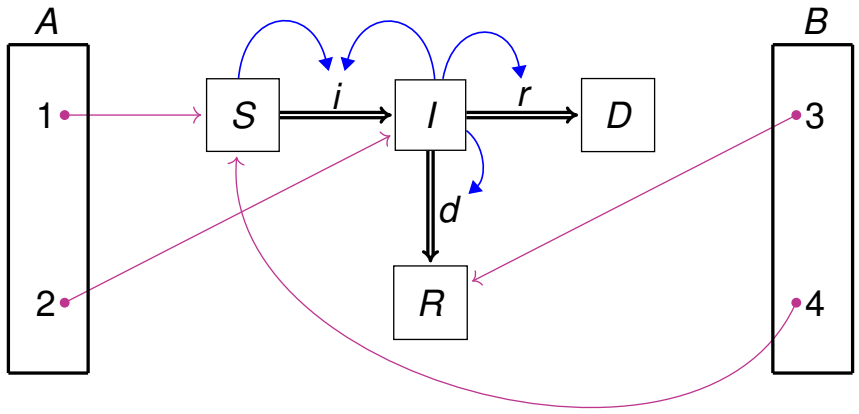
Just as we constructed the category Open(StockFlow), we can construct a category Open(Dynam) with:

- finite sets as objects,
- open dynamical systems as morphisms.

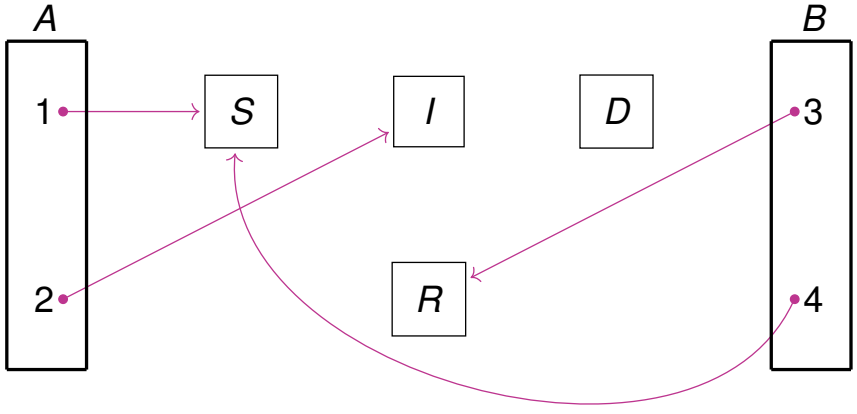
The process we've already seen for converting stock flow diagrams into dynamical systems then gives a functor

$$\Phi\colon \mathrm{Open}(\mathrm{StockFlow}) \to \mathrm{Open}(\mathrm{Dynam})$$

For example, Φ maps this open stock flow diagram:



to this *open* dynamical system:



$$\frac{dS}{dt} = -\phi_i(S, I)$$

$$\frac{dI}{dt} = \phi_i(S, I) - \phi_r(I) - \phi_d(I)$$

$$\frac{dR}{dt} = \phi_r(I)$$

$$\frac{dD}{dt} = \phi_d(I)$$

What does it mean that

$$\Phi : \mathrm{Open}(\mathrm{StockFlow}) \to \mathrm{Open}(\mathrm{Dynam})$$

is a functor?

What does it mean that

$$\Phi\colon \mathrm{Open}(\mathrm{StockFlow}) \to \mathrm{Open}(\mathrm{Dynam})$$

is a functor?

Most importantly, it means that for any open stock-flow diagrams $A \xrightarrow{F} B$ and $B \xrightarrow{G} C$ we have

$$\Phi(G \circ F) = \Phi(G) \circ \Phi(F)$$

We created software to implement these ideas with Evan Patterson and Sophie Libkind at the Topos Institute, who are experts on categories for computing:


- JB, Xiaoyan Li, Sophie Libkind, Nathaniel D. Osgood and Evan Patterson, Compositional modeling with stock and flow diagrams.

We used AlgebraicJulia: a framework for high-performance scientific computing using categories. This was developed by James Fairbanks, Evan, Sophie, and many others.

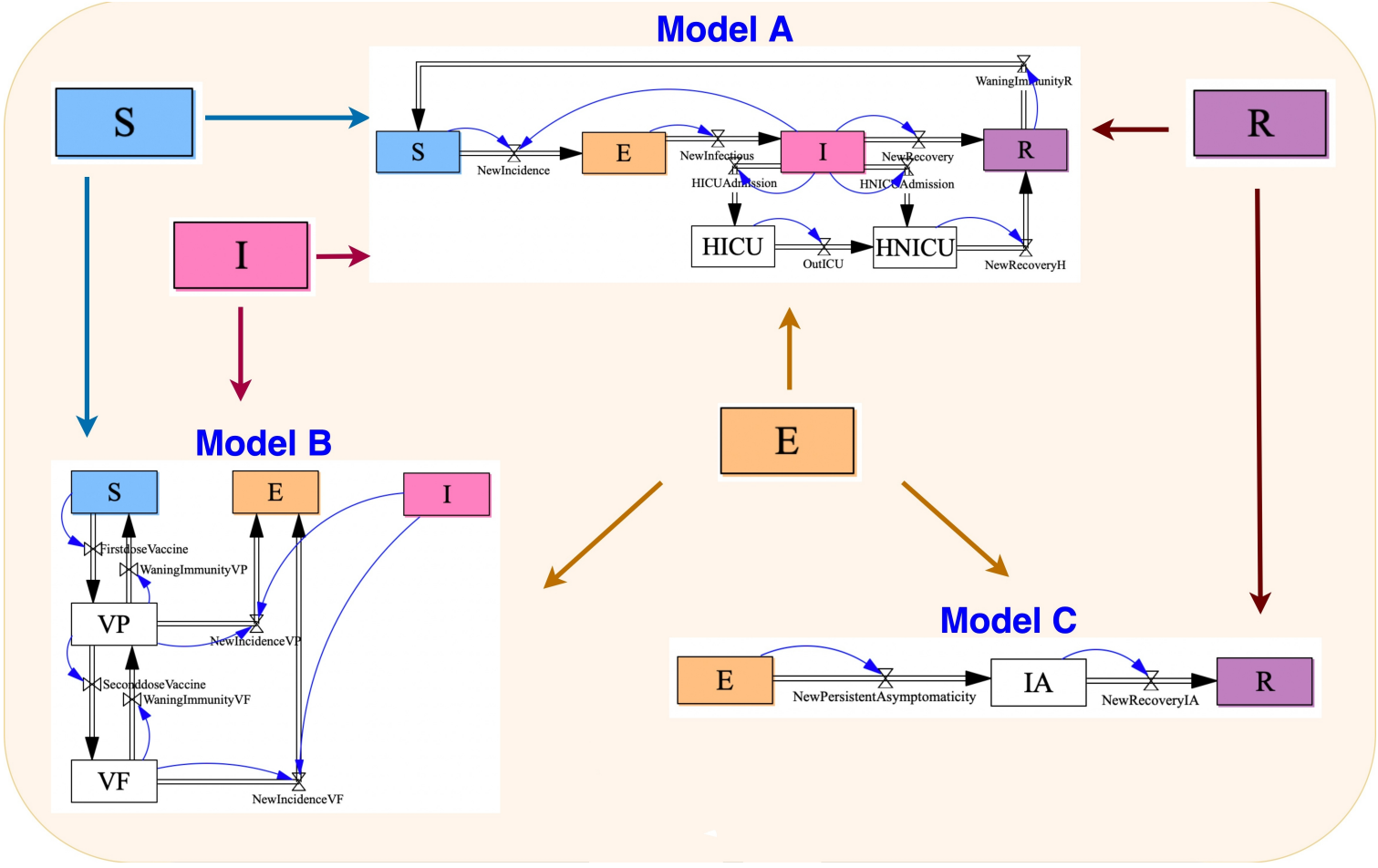
Composition of dynamical systems had already been implemented in AlgebraicJulia:

- Sophie Libkind, Andrew Baas, Evan Patterson and James Fairbanks, Operadic modeling of dynamical systems: mathematics and computation.

Using AlgebraicJulia we created a software package called **StockFlow**, now available on GitHub. This lets you:

- build stock-flow diagrams
- view them
- make them “open”
- apply various functors to interpret stock-flow diagrams, e.g. as systems of differential equations but also other things
- numerically solve the resulting differential equations.

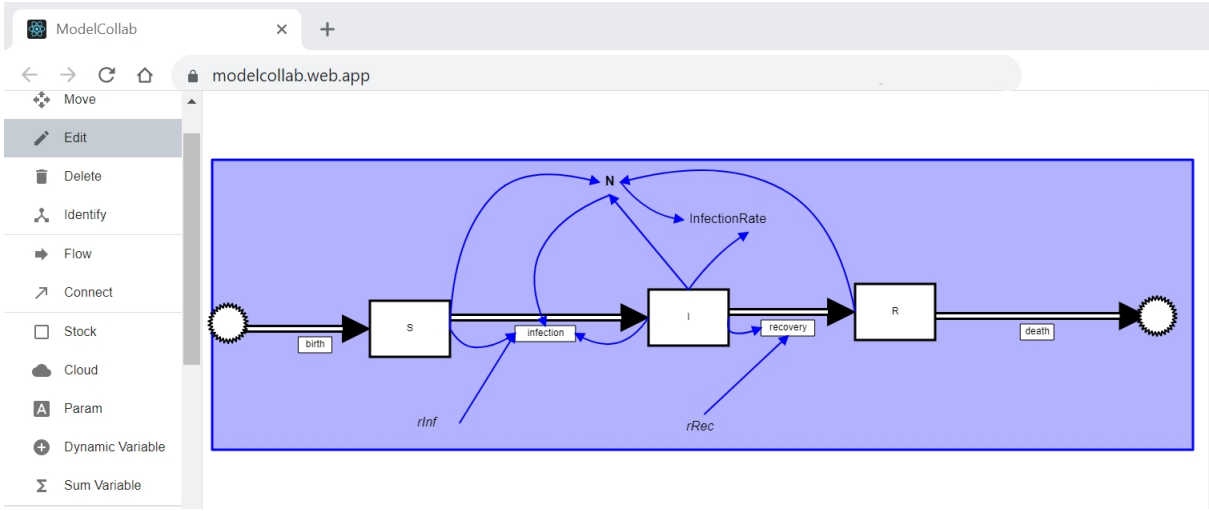
Besides composing open stock-flow diagrams end-to-end, our software also let you compose them in more complex ways, like this:



With Eric Redekopp we have made a graphical user interface for StockFlow, called **ModelCollab** — also available on GitHub.

- John C. Baez, Xiaoyan Li, Sophie Libkind, Nathaniel D. Osgood, Long Pham and Eric Redekopp, A categorical framework for modeling with stock and flow diagrams.

ModelCollab runs in your browser, so teams can collaborate to build stock-flow diagrams.



And you *don't need to know anything about category theory or AlgebraicJulia to use it!*

The fancy stuff is black-boxed.

Next steps:

In the spring of 2024, Nate, Xiaoyan, Evan, Kris Brown (Topos), Sean Wu (Merck) and I will spend 6 weeks at the ICMS here in Edinburgh.

We'll extend our work to "agent-based models". These simulate individual agents rather than treating them *en masse* as mere "stocks".

Together with Patricia Mabry at the HealthPartners Institute for Education and Research, Nate Osgood and I will apply compositional modeling to health problems related to substance abuse.

Finally, a little announcement.

Finally, a little announcement.

Nate Osgood and I have been called in to lead a program called Mathematics for Climate Change at the Fields Institute in Toronto.

It's not mainly about the geophysics of climate change — but rather, the human response to it: that is, *figuring out what we should do!*

A lot of mathematicians want to do something about climate change... but don’t know what.

A lot of mathematicians want to do something about climate change... but don't know what.

I hope we can form working groups where mathematicians learn from experts trying to solve the problems of climate change — and help them with subjects like these:

- Parameter estimation
- Causal discovery and attribution
- Optimization
- Uncertainty quantification
- Agent-based models
- Tipping point theory
- Game theory
- Decision theory