Compositional Modeling with Stock and Flow Diagrams

John Baez, Xiaoyan Li, Sophie Libkind, Nathaniel Osgood, and Evan Patterson

July 11, 2022

Abstract

Stock and flow diagrams are widely used in epidemiology to model the dynamics of populations. Although tools already exist for building these diagrams and simulating the systems they describe, we have created a new package called StockFlow, part of the AlgebraicJulia ecosystem, which uses ideas from category theory to overcome notable limitations of existing software. Compositionality is provided by the theory of decorated cospans: stock and flow diagrams can be composed to form larger ones in an intuitive way formalized by the operad of undirected wiring diagrams. Our approach also cleanly separates the syntax of stock and flow diagrams from the semantics they can be assigned. We consider semantics in ordinary differential equations, although others are possible. As an example, we explain code in StockFlow that implements a simplified version of a COVID-19 model used in Canada.

1 Introduction

The theoretical advantages of compositionality and functorial semantics are widely recognized among applied category theorists. *Compositionality* means, at the very least, that systems can be described one piece at a time, with a clear formalism for composing these pieces. This formalism can appear in various styles: composing morphisms in a category, tensoring objects in a monoidal category, composing operations in an operad, etc. *Functorial semantics* then means that the map from system descriptions ("syntax") to their behavior ("semantics") preserves all the relevant forms of composition.

While these principles are elegant, in many fields it is still a challenge to produce useful software that takes advantage of them and is embraced by the intended users. This is one of the main challenges of applied category theory. Here we focus on developing software suited to one particular field: epidemiological modeling. At present this software is additionally capable of modeling a wide class of systems studied in the System Dynamics modeling discipline [12, 29].

The AlgebraicJulia ecosystem of software implements compositionality and functorial semantics in a thorough-going way [1]. Decorated and structured cospans are broad mathematical frameworks for turning "closed" system descriptions into "open" ones that can be composed along their boundaries [10, 3, 4]. One part of AlgebraicJulia, called Catlab [22], provides a generic interface for working with such cospans, among other categorical abstractions. With the help of Catlab, a tool called AlgebraicPetri was developed to work with one approach to epidemiological modeling based on Petri nets [18]. Here we explain a new tool, StockFlow, which handles a more flexible and more widely used formalism for epidemiological modeling: stock and flow diagrams.

In Section 2 we review how stock and flow diagrams are used in epidemiological modeling, and discuss some shortcomings of existing software for working with these diagrams. In Section 3 we first use decorated cospans to formalize a simple class of open stock and flow diagrams and their differential equation semantics, and then sketch how to extend this class to the full-fledged diagrams actually used in our software. In Section 4 we describe the software package, StockFlow, that we have developed to work with stock-flow diagrams compositionally and implement a functorial semantics for them.[1]

[1] The link to the StockFlow repository on GitHub is `https://github.com/AlgebraicJulia/StockFlow.jl`.

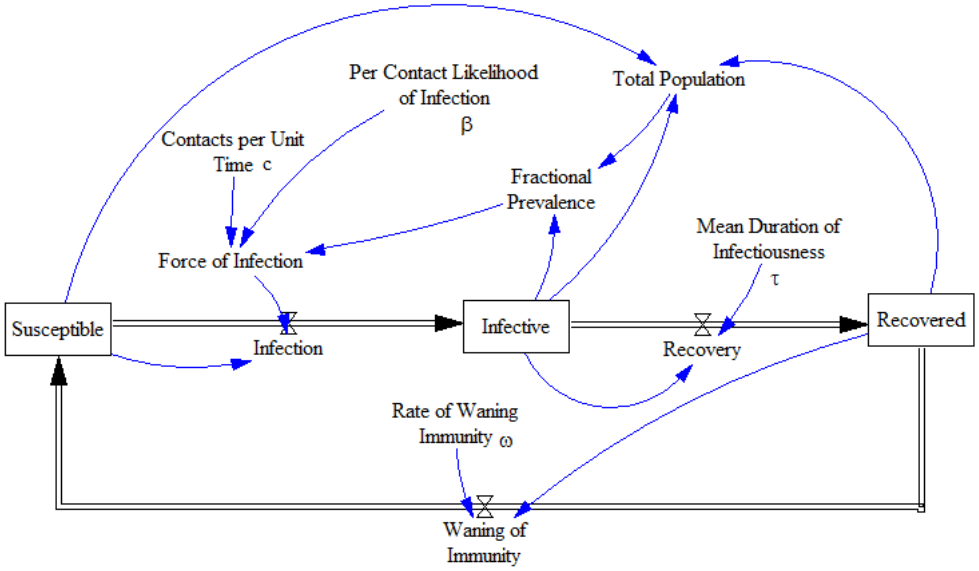


Figure 1: An example stock-flow diagram. This stock-flow diagram has three stocks labeled "Susceptible", "Infective", and "Recovered"; three flows labeled "Infection", "Recovery", and "Waning of Immunity"; many auxiliary variables including "Force of Infection" and "Total Population"; and links depicted by blue arrows.

2 Epidemiological modeling with stock and flow diagrams

Effective decision-making regarding prevention, control, and service delivery to address the health needs of the population involves reasoning about diverse complexities: policy resistance, feedbacks, heterogeneities, multi-condition interactions, and nonlinearities that collectively give rise to counterintuitive results [28, 30]. For over a century, researchers and practitioners have used epidemiology models to address such challenges. Since dynamic epidemiological modeling was first applied to communicable diseases [24, 25, 17], it has both deepened its reach in that area [2, 8] and spread to many other subdomains of epidemiology, including chronic, behavioural, environmental, occupational, and social epidemiology, as well as spheres such as mental health and addictions. Reflecting a world in which growing global interconnection is juxtaposed with increasing ecosystem encroachment and climate stresses, the rise of the "One Health" perspective [7, 21] has motivated such modeling to increasingly incorporate dynamics from domains such as ecology, veterinary and agricultural health, and social dynamics and inequities. Such efforts have come to define the field of mathematical and computational epidemiology.

The earliest and still most common epidemiological models are sets of ordinary differential equations, typically used to characterize epidemiological dynamics in an aggregate fashion [2]. Delay and partial differential equations have also been widely applied. Recent decades have witnessed a rapid growth in use of agent-based models. Although the techniques explored here may be more widely applicable, we focus on aggregate models described using differential equations.

Contemporary aggregate-level modeling involves widespread informal use of diagrams, with the most prevalent type of such diagrams being transition diagrams and their richer and more formal cousins, "stock and flow diagrams" [29], as depicted in Figure 1. Transition diagrams are a minimalist box-and-arrow formalism which draws state variables as boxes and transitions as arrows. Traditionally, most mathematical epidemiologists have focused directly on the underlying differential equations, regarding such diagrams only as an informal presentation of the equations. Thus, diagrams are commonly treated either as ephemeral artifacts useful for thinking out structures and then discarded, or as an expedient aid for communication.

Amongst the notable minority of health modelers who employ stock and flow diagrams (also called "Forrester diagrams", and termed here "stock-flow diagrams" for brevity), these diagrams

play roles at different stages of the modeling process. Stock-flow diagrams depict state variables as stocks (rectangles), changes to those stocks as flows (thick arrows also termed "material connections"), constants and auxiliary variables (also called "dynamic variables"), and links (arrows sometimes called "informational connections" or simply "connections") characterizing instantaneous dependencies.

Stock-flow diagrams serve as the central formalism in the modeling tradition of System Dynamics, initiated by Forrester in the 1950s [12, 29]. Since the 1980s, System Dynamics software has provided refined, visually accessible, declarative user interfaces for interactively building, and browsing stock-flow diagrams [23]. While such packages are designed to ensure transparency of model structure to modelers and stakeholders [23], they also serve as simulation tools. For that purpose, the System Dynamics tradition universally interprets stock-flow diagrams as characterizing ordinary differential equations. Stocks represent the state variables; their formulation requires specifying an initial value. Flows represent the differentials associated with stocks, and are each associated with a modeler-specified mathematical expression specifying the flow rate (quantity per unit time) as a function of other variables. Each constant variable is associated with a real scalar. Auxiliary variables generally reflect quantities of domain significance that depend instantaneously on other model quantities. Each such auxiliary variable is associated with a modeler-specified expression characterizing the value of that auxiliary as a function of the current value of other variables (stocks, flows, constants and other auxiliary variables).

Reflecting the strong emphasis that System Dynamics practice places on stakeholder engagement and participatory model building, models built in the stock-flow paradigm are routinely shown to stakeholders without modeling background—be they domain experts from a modeling team, stakeholders, or community members—to elicit critiques and suggestions [23, 31, 15]. System Dynamics has also long sought to recognize, codify, and exploit widespread use of modeling idioms. Thus, researchers and practitioners have formalized dozens of simple stock-flow diagrams called "molecules" for reuse in modeling [14]. Some simulation packages provide molecules as prespecified templates defined by the software, and mechanisms for for directly incorporating built-in templates for such molecules into models. When a molecule is added to a model, the elements of the molecule — such as stocks and flows — are simply added as elements of the surrounding diagram, rather than being reused as higher level abstractions. Moreover, because such libraries of molecules are fixed, such molecules cannot be created or packaged up by the user.

Software packages for stock-flow diagrams have as a central feature the simulation, via numerical integration, of the system of ordinary differential equations described by these diagrams. Many such packages also offer additional forms of model analysis, including identification of feedback loops, performing tests of dimensional homogeneity based on modeler unit annotations, sensitivity analysis and calibration. Some tools support more sophisticated forms of analysis, such as those involving Markov Chain Monte Carlo and extended Kalman filtering. But while existing stock-flow modeling tools offer refined interfaces for building, exploring and simulating models, their support for modern modeling practice is hampered by significant rigidity and several additional shortcomings. The present paper focuses on addressing two limitations of contemporary tools.

First, and most notable from a categorical perspective, existing tools *lack support for composition* of models, despite there being several natural ways in which models might be composed. Instead, each model is currently treated in isolation. If models are composed at all, it is by either outputting data files from one and importing such data into another, or by creating, via an ad hoc process, a third model that contains both of the original models.

Second, existing stock-flow modeling tools *privilege a single semantics* associated with stock-flow diagrams: the interpretation of these diagrams as ordinary differential equations. While alternative interpretations can sometimes be force-fit—for example, a difference equation interpretation by using Euler integration, or a stochastic differential equation interpretation using formulas for flows drawing from suitable probability distributions—they are commonly awkward, obscure, and error-prone. Although particular packages allow for select additional analyses—for example, identification of feedback loops—such features are hard-coded, and many analysis tools demonstrated as valuable by research [26, 13, 16] have not been incorporated in extant software packages.

We turn next to a mathematical framework that provides a remedy for these deficiencies: an explicitly compositional framework where "open" stock-flow diagrams become morphisms in a category and where semantics is described as a functor from this category to some other category.

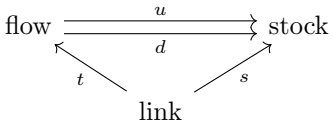
For reasons of space we only describe one choice of semantics, but the clear separation of syntax and semantics permits swapping out this choice for others.

3 The mathematics of stock-flow diagrams

Stock-flow diagrams come in many variants. To illustrate our methodology we begin with a very simple kind. In our code we have implemented a more sophisticated variant with additional features, but the ideas are easier to explain without those features. Our main goal is to study *open* stock-flow diagrams—that is, stock-flow diagrams in which various stocks are specified as "interfaces." We can treat open stock-flow diagrams with two interfaces as morphisms of a category. Composing these morphisms then lets us build larger diagrams from smaller ones. Alternatively, we can compose stock-flow diagrams with any number of interfaces using an operad. Both approaches let us describe the differential equation semantics for open stock-flow diagrams following a paradigm already explored for open Petri nets with rates [4, 5]. We describe that paradigm here.

3.1 A category of stock-flow diagrams

As a first step, we define "primitive" stock-flow diagrams with stocks, flows, and links but not the all-important functions that describe the rate of each flow. For this, we consider a category H freely generated by these objects and morphisms:



We call a functor $F\colon \mathsf{H} \to \mathsf{FinSet}$ a **primitive stock-flow diagram**. It amounts to the following:

- a finite set of stocks $F(\text{stock})$,
- a finite set of flows $F(\text{flow})$,
- functions $F(u), F(d)\colon F(\text{flow}) \to F(\text{stock})$ assigning to each flow the stock **upstream** from it, and the stock **downstream** from it,
- a finite set of links $F(\text{link})$,
- functions $F(s)\colon F(\text{link}) \to F(\text{stock}), F(t)\colon F(\text{link}) \to F(\text{flow})$ assigning to each link its **source**, which is a stock, and its **target**, which is a flow.

Given $f \in F(\text{flow})$, we say f **flows from** the upstream stock $F(u)(f)$ and **flows to** the downstream stock $F(d)(f)$. We say that a link $\ell \in F(\text{link})$ **points from** its source $F(s)(\ell)$ and **points to** its target $F(t)(\ell)$. There is a category of primitive stock-flow diagrams, $\mathsf{FinSet}^{\mathsf{H}}$, where the objects are functors from H to FinSet and a morphism from $F\colon \mathsf{H} \to \mathsf{FinSet}$ to $G\colon \mathsf{H} \to \mathsf{FinSet}$ is a natural transformation.

Primitive stock-flow diagrams are useful for *qualitative* aspects of modeling, since they clearly show which flows depend on which stocks; as such, they can be seen as a restricted form of *system structure diagrams* [20, 9] used in System Dynamics practice. But they become useful for *quantitative* modeling and simulation only when we equip them with functions saying how the rate of each flow depends on the value of each stock. Thus, we define a **stock-flow diagram** to be a pair (F, ϕ) consisting of an object $F \in \mathsf{FinSet}^{\mathsf{H}}$ and a continuous function called a **flow function**

$$\phi_f\colon \mathbb{R}^{F(t)^{-1}(f)} \to \mathbb{R}$$

for each flow $f \in F(\text{flow})$, where $F(t)^{-1}(f)$ is the set of links with target f:

$$F(t)^{-1}(f) = \{\ell \in F(\text{link}) \mid F(t)(\ell) = f\}.$$

The idea is that the flow function ϕ_f says how the rate of the flow f depends on the values of all the stocks with links pointing to it. We make this precise in Section 3.4 when we introduce a

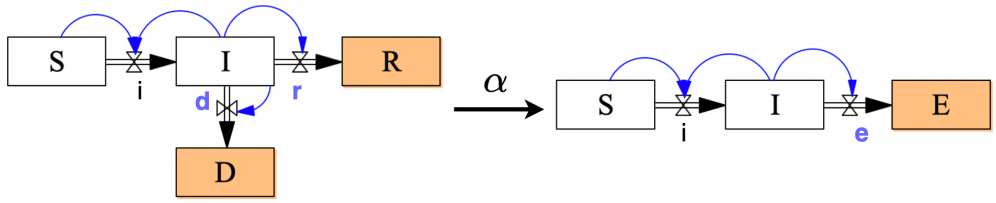


Figure 2: At left is a stock-flow diagram (F, ϕ) with stocks S, I, R, D corresponding to susceptible, infected, recovered and deceased populations and flows i, r, d corresponding to infection, recovery and death. Links are shown in blue. At right we see a simpler stock-flow diagram (G, ψ) where recovered and deceased populations are lumped into a single "removed" stock E, and recovery and death are lumped into a single "removal" flow e. There is an evident morphism $\alpha\colon (F, \phi) \to (G, \psi)$ sending R and D to E and sending r and d to e.

semantics that maps each stock-flow diagram to a first-order differential equation. But rates and values play no formal role in this section.

To define a category of stock-flow diagrams, we need to define morphisms between them. What is a morphism from (F, ϕ) to (G, ψ)? It is a natural transformation $\alpha\colon F \Rightarrow G$ with an extra property. Because α is natural, we get a commutative square

$$\begin{array}{ccc} F(\text{link}) & \xrightarrow{\alpha(\text{link})} & G(\text{link}) \\ {\scriptstyle F(t)}\downarrow & & \downarrow{\scriptstyle G(t)} \\ F(\text{flow}) & \xrightarrow[\alpha(\text{flow})]{} & G(\text{flow}). \end{array}$$

Thus, letting $g = \alpha(\text{flow})(f)$ for $f \in F(\text{flow})$, we get a map

$$\alpha(\text{link})\colon F(t)^{-1}(f) \to G(t)^{-1}(g)$$

and thus a linear map

$$\alpha(\text{link})^*\colon \mathbb{R}^{G(t)^{-1}(g)} \to \mathbb{R}^{F(t)^{-1}(f)}$$

given by precomposition:

$$\alpha(\text{link})^*(x) = x \circ \alpha(\text{link}).$$

We say that α is a **morphism of stock-flow diagrams** from (F, ϕ) to (G, ψ) if

$$\psi_g = \sum_{f \in \alpha(\text{flow})^{-1}(g)} \phi_f \circ \alpha(\text{link})^* \tag{1}$$

for every $g \in G(\text{flow})$. This equation expresses rates of flows in (G, ψ) as sums of rates of flows in (F, ϕ). For example, Figure 2 shows a morphism of stock-flow diagrams in which two flows, "recovery" r and "death" d, are mapped to a single "removal" flow e. The above equation implies that

$$\psi_e = \phi_r \circ \alpha(\text{link})^* + \phi_d \circ \alpha(\text{link})^*.$$

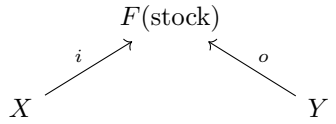
This equation says that the rate of the flow e is the sum of the rates of r and d.

Composition of morphisms between stock-flow diagrams is just composition of their underlying natural transformations; one can show that indeed the composite of two natural transformations obeying Eq. (1) again obeys this equation. We thus obtain a category of stock-flow diagrams, which we call StockFlow.

3.2 Open stock-flow diagrams

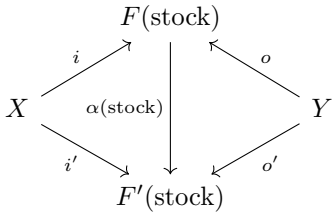
We can build larger stock-flow diagrams by gluing together smaller ones. There are a number of choices of how to formalize this. Here we glue together two stock-flow diagrams by identifying two

collections of stocks to serve as "interfaces." Thus, we define an **open stock-flow diagram** with finite sets X and Y as interfaces to be a stock-flow diagram (F, ϕ) equipped with functions from X and Y to its set of stocks:



We call this an open stock-flow diagram from X to Y and write it tersely as $(F, \phi) \colon X \to Y$, despite the maps i and o being a crucial part of the structure.

We can compose open stock-flow diagrams from X to Y and from Y to Z to obtain one from X to Z. To formalize this composition process we use Fong's theory of decorated cospans [10]. However, to make composition associative and get a category we need to use *isomorphism classes* of open stock-flow diagrams. Two open stock-flow diagrams (F, ϕ) and (F', ϕ') from X to Y are **isomorphic** if there is an isomorphism of stock-flow diagrams $\alpha \colon (F, \phi) \to (F', \phi')$ such that this diagram commutes:



Using the theory of decorated cospans, we obtain:

Theorem 3.1. *There is a category* $\mathsf{Open}(\mathsf{StockFlow})$ *such that:*

- *An object is a finite set* X*.*
- *A morphism from* X *to* Y *is an isomorphism class of open stock-flow diagrams from* X *to* Y*.*

This is a symmetric monoidal category, indeed a hypergraph category.

Proof Sketch. This follows from the theory of decorated cospans [10] once we check the following facts. For each finite set there is a category $\mathsf{C}(S)$ whose

- objects are stock-flow diagrams with S as their set of stocks, and
- morphisms are morphisms α of stock-flow diagrams where $\alpha(\mathrm{stock})$ is the identity on S.

Let $C(S)$ be the set of isomorphism classes of objects in this category. A map of finite sets $f \colon S \to S'$ functorially determines a map from $C(S)$ to $C(S')$, and the resulting functor C is symmetric lax monoidal from $(\mathsf{FinSet}, +)$ to $(\mathsf{Set}, \times)$, where the laxator

$$\gamma \colon C(S) \times C(S') \to C(S + S')$$

maps a pair of stock-flow diagrams to their "disjoint union." It follows from [10, Prop. 3.2] that we get the desired symmetric monoidal category $\mathsf{Open}(\mathsf{StockFlow})$, and from [10, Thm. 3.4] that this is a hypergraph category. □

The point of making open stock-flow diagrams into the morphisms of a hypergraph category is that it gives ways of composing these diagrams that are more flexible than just composing them "end-to-end" (ordinary composition of morphisms) and "side-by-side" (a parallel arrangement expressed by tensoring). Indeed, hypergraph categories are algebras of an operad, sometimes called the operad of undirected wiring diagrams, that encapsulates a wide range of composition strategies [11]. We use this approach in our code, and instead of working with cospans we actually use multicospans [19, 27], a mild generalization that allows for open stock-flow diagrams with any number of interfaces, not just two.

We conclude with a technical remark. In fact, there is a symmetric lax monoidal functor $\mathsf{C} \colon (\mathsf{FinSet}, +) \to (\mathsf{Cat}, \times)$ that sends each finite set S to a *category* of stock-flow diagrams with

S as their set of stocks. Theorem 2.2 of [4] thus gives a symmetric monoidal double category $\mathbb{O}\mathbf{pen}(\mathsf{StockFlow})$ where objects are finite sets and horizontal 1-cells are actual open stock-flow diagrams—not mere isomorphism classes of these.

This double category allows us to work with maps *between* open stock-flow diagrams. This should be useful for mapping several stocks to a single stock in a simplified stock-flow diagram, as in Figure 2, or embedding a stock-flow diagram in a more complicated one. However, StockFlow currently does not attempt to support maps between open stock-flow diagrams, so Theorem 3.1 suffices for us. Indeed, when working with a mere *category* of open stock-flow diagrams, as opposed to a double category, we can define an isomorphic category using structured rather than decorated cospans: for open stock-flow diagrams, the difference only becomes visible at the double category level. Thus, our treatment using decorated cospans looks forward to a future where we work with maps between open stock-flow diagrams.

3.3 Open dynamical systems

Our next goal is to define a semantics for stock-flow diagrams mapping each such diagram to a dynamical system: a system of differential equations that describes the continuous-time evolution of the value of each stock. This semantics is implicit in the usual applications of stock-flow diagrams; indeed, the stock-flow diagram is sometimes regarded merely as a convenient notation for a dynamical system. While we illustrate the choice of a semantics for stock-flow diagrams using the continuous dynamical system interpretation, this semantics holds no privileged status, and there are several other semantics of practical value that could be employed instead.

In fact, our semantics is more general than suggested above: we describe a map from *open* stock-flow diagrams to *open* dynamical systems. Our strategy for defining this semantics closely follows the strategy already used for open Petri nets with rates [3, 4, 5] and implemented for epidemiological models using AlgebraicJulia [18].

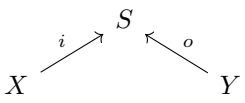
For Petri nets with rates, the dynamics is typically described by the "law of mass action," which only produces dynamical systems that are polynomial-coefficient vector fields on $\mathbb{R}^n$. In stock-flow diagrams this restriction is dropped, but the rate of any flow out of its upstream stock equals the rate of flow into its downstream stock, so the total value of all stocks is conserved. However, the more general stock-flow diagrams of Section 3.5 no longer obey this conservation law, since they allow "inflows" and "outflows" to the diagram as a whole. With these generalizations, stock-flow diagrams become strictly more general than Petri nets with rates—at least in terms of the dynamical systems they can describe.

We begin by defining a **dynamical system** on a finite set S to be a continuous vector field $v\colon \mathbb{R}^S \to \mathbb{R}^S$. In our applications, S will be the set of stocks of some stock-flow model, and the vector field v is used to write down a differential equation describing the dynamics:

$$\frac{dx(t)}{dt} = v(x(t))$$

where at each time t, the vector $x(t) \in \mathbb{R}^S$ describes the value of each stock at time t. Since the vector field is continuous, the Peano existence theorem implies that, for any initial value $x(0) \in \mathbb{R}^S$, the above equation has a solution for all t in some interval $(-\epsilon, \epsilon)$. However, the solution may not be unique unless we require that v be nicer. The theory we develop now can be modified to add extra restrictions, simply by replacing continuous functions with functions of a suitably nicer sort.

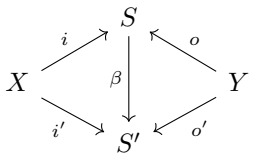
We define an **open** dynamical system from the finite set X to the finite set Y to be a pair (S, v), consisting of a finite set S and a dynamical system v on S, together with functions from X and Y into S. We depict this as follows:



$$v \in D(S)$$

where $D(S)$ is the set of all dynamical systems on S. Two open dynamical systems (S, v) and (S', v') from X to Y are **isomorphic** if there is a bijection $\beta\colon S \to S'$ such that the following

diagram commutes:



$v \in D(S)$

$v' \in D(S')$

and $\beta_* \circ v \circ \beta^* = v'$, where $\beta_* : \mathbb{R}^S \to \mathbb{R}^{S'}$ is the pushforward map defined by

$$\beta_*(x)(\sigma') = \sum_{\sigma \in \beta^{-1}(\sigma')} x(\sigma) \qquad \forall x \in \mathbb{R}^S,\ \sigma' \in S'.$$

We can then construct a category where objects are finite sets and morphisms from X to Y are isomorphism classes of open dynamical systems from X to Y. This was done in [5, Theorem 17] by applying Fong's theory of decorated cospans to a functor $D\colon \mathsf{FinSet} \to \mathsf{Set}$ sending any finite set S to the set $D(S)$ of dynamical systems on S:

Theorem 3.2 (**Baez–Pollard**). *There is a category* $\mathsf{Open}(\mathsf{Dynam})$ *such that:*

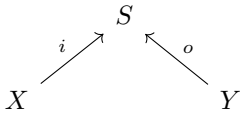
- *An object is a finite set X.*
- *A morphism from X to Y is an isomorphism class of open dynamical systems from X to Y.*

This is a symmetric monoidal category, indeed a hypergraph category.

In fact, there is a symmetric lax monoidal functor $\mathsf{D}\colon (\mathsf{FinSet}, +) \to (\mathsf{Cat}, \times)$ that maps any set S to the discrete category on the set $D(S)$ described above. The theory of decorated cospans then gives a symmetric monoidal *double* category $\mathbb{O}\mathbf{pen}(\mathsf{Dynam})$ where objects are finite sets and horizontal 1-cells are open dynamical systems. This is discussed in [4, Sec. 6.4].

3.4 Open dynamical systems from open stock-flow diagrams

Next we describe a functor sending any open stock-flow diagram to an open dynamical system. Suppose we have an open stock-flow diagram $(F, \phi)\colon X \to Y$, equipped with the cospan



where $S = F(\mathrm{stock})$. Then there is an open dynamical system $v(F, \phi)$ on S given by

$$v(F,\phi)(x)(\sigma) = \sum_{f \in F(d)^{-1}(\sigma)} \phi_f(x \circ F(s)) \ - \sum_{f \in F(u)^{-1}(\sigma)} \phi_f(x \circ F(s)) \qquad \forall x \in \mathbb{R}^S, \sigma \in S. \tag{2}$$

This formula looks a bit cryptic, so let us explain it. Taking the expression $\mathbb{R}^S$ seriously, we can think of $x \in \mathbb{R}^S$ as a real-valued function on the set S of stocks. Each flow $f \in F(\mathrm{flow})$ has a set $F(t)^{-1}(f)$ of links with f as target, so there is an inclusion of sets $F(t)^{-1}(f) \hookrightarrow F(\mathrm{link})$, and we can thus form the composite

$$F(t)^{-1}(f) \hookrightarrow F(\mathrm{link}) \xrightarrow{F(s)} F(\mathrm{stock}) \xrightarrow{x} \mathbb{R}$$

which for short we call simply

$$x \circ F(s) \in \mathbb{R}^{F(t)^{-1}(f)}.$$

For each link ℓ with f as its target, this composite gives the value of the stock that is ℓ's source. Applying the function $\phi_f\colon \mathbb{R}^{F(t)^{-1}(f)} \to \mathbb{R}$, we obtain the rate of the flow f:

$$\phi_f(x \circ F(s)) \in \mathbb{R}.$$

This quantity has the effect of increasing the stock $d(f)$ and also decreasing the stock $u(f)$. Thus the rate of change of any stock $\sigma \in S$ is

$$\sum_{f \in F(d)^{-1}(\sigma)} \phi_f(x \circ F(s)) - \sum_{f \in F(u)^{-1}(\sigma)} \phi_f(x \circ F(s)).$$

This gives our formula for $v(F, \phi)$ in Equation (2).

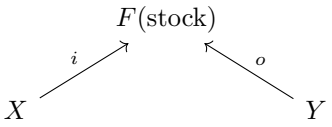
Now, recall that in Theorem 3.1 the category of open stock-flow diagrams was defined as a decorated cospan category using the functor $C \colon \mathsf{FinSet} \to \mathsf{Set}$, while in Theorem 3.2 the category of open dynamical systems was defined in a similar way using the functor $D \colon \mathsf{FinSet} \to \mathsf{Set}$. According to the theory [10], to obtain a semantics mapping open stock-flow diagrams to open dynamical systems, we need to define a natural transformation $\theta \colon C \Rightarrow D$. We do this as follows: for each finite set S, define $\theta(S)$ to map the isomorphism class (F, ϕ) in $C(S)$ to the isomorphism class of $v(F, \phi)$ in $D(S)$.

Theorem 3.3. *There is a functor*

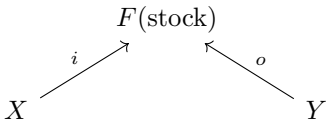
$$v \colon \mathsf{Open}(\mathsf{StockFlow}) \to \mathsf{Open}(\mathsf{Dynam})$$

sending

- *any finite set to itself,*
- *the isomorphism class of the stock-flow diagram (F, ϕ) made open as follows:*



to the isomorphism class of the open dynamical system



$v(F, \phi) \in D(S)$.

This is a symmetric monoidal functor, indeed a hypergraph functor.

Proof Sketch. By [10, Thm. 4.1] it suffices to check that $\theta \colon C \Rightarrow D$ is indeed a natural transformation and furthermore a *monoidal* natural transformation. □

With more work one can extend the natural transformation v to a monoidal natural transformation between the 2-functors $\mathsf{C} \colon \mathsf{FinSet} \to \mathsf{Cat}$ and $\mathsf{D} \colon \mathsf{FinSet} \to \mathsf{Cat}$. By [4, Thm. 2.5], this gives a symmetric monoidal double functor from $\mathbb{O}\mathbf{pen}(\mathsf{StockFlow})$ to $\mathbb{O}\mathbf{pen}(\mathsf{Dynam})$. However, we do not need this yet in our code.

3.5 Full-fledged stock-flow diagrams

In Section 3.1 we defined a simple category of stock-flow diagrams, called $\mathsf{StockFlow}$. Stock-flow diagrams of this type capture two main features of the diagrams used by practitioners: (1) flows between stocks and (2) links that represent the dependency of flow rates on the values of particular stocks. However, the stock-flow diagrams used in epidemiological modeling have additional useful features. Our "full-fledged" stock-flow diagrams include auxiliary variables, sum variables, and partial flows.

Auxiliary variables are quantities on which flow functions can depend. An auxiliary variable is linked to stocks and other auxiliary variables and is equipped with an arbitrary continuous function of the values of stocks and variables to which it is linked. In Figure 1, "Fractional Prevalence", "Force of Infection", and "Infection" are all examples of auxiliary variables. Auxiliary variables

are important to practitioners for several reasons. First, they simplify model specification because they are reusable: instead of computing each flow rate directly as a function of stocks, we can often compute them more simply with the help of auxiliary variables. Many flow rates can depend on a single auxiliary variable. Second, they often represent quantities that are of interest to stakeholders; representing these quantities explicitly make them easier to track throughout a simulation, such as for comparison with empirical data. Third, they are practical for the communication and revision of models. Auxiliary variables give a meaningful decomposition of the flow functions, and changing a single auxiliary variable automatically revises all the flow functions that depend on it, which eliminates the need to revise all these flow functions separately.

While not explicitly distinguished in current stock-flow modeling packages, flow functions often rely on a special case of auxiliary variables called *sum variables*. Such a variable equals the sum of the values of some subset of the stocks. In epidemiology, this frequently corresponds to the size of a population or sub-population. For example, "Total Population" in Figure 1 is a sum variable. In general, a sum variable may link to only a subset of stocks. Sum variables can be seen as a particular type of auxiliary variable in which the function merely sums the values of the stocks to which it is linked—and thanks to this fact, we do not need to label sum variables with functions.

Finally, in the simple stock-flow diagrams described earlier, each flow must have an upstream stock and a downstream stock. However, practitioners often use diagrams including *partial flows*, which may have only an upstream stock or only a downstream stock. These represent the creation or the destruction of some resource, and are commonly used to represent open populations.

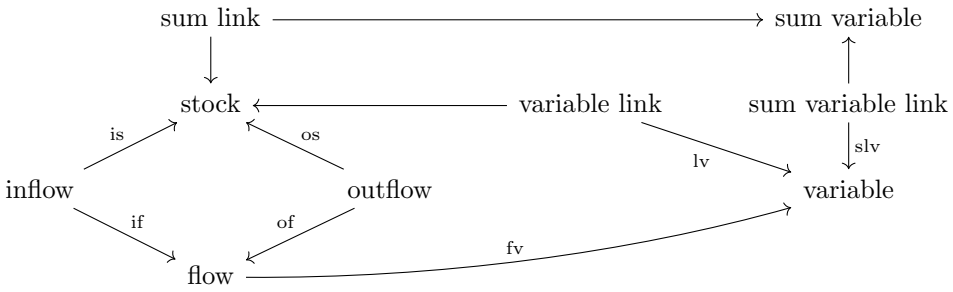


Figure 3: The free category on this diagram, called H_f, is used to define full-fledged stock-flow diagrams. We have named only some of the arrows here.

Figure 3 presents the category H_f used to define full-fledged stock-flow diagrams. A **full-fledged stock-flow diagram** is a pair (F, ϕ) consisting of:

- a functor $F\colon \mathsf{H}_f \to \mathsf{FinSet}$ such that the functions $F(\mathrm{if})$ and $F(\mathrm{of})$ are injective;
- a continuous function $\phi_v\colon \mathbb{R}^{F(\mathrm{lv})^{-1}(v)} \times \mathbb{R}^{F(\mathrm{slv})^{-1}(v)} \to \mathbb{R}$ for each $v \in F(\mathrm{variable})$, called an **auxiliary function**.

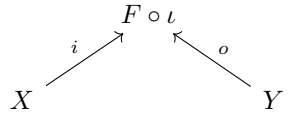
The elements of $F(\mathrm{variable})$ are called **auxiliary variables**. The idea is that in a full-fledged stock-flow diagram each flow has its rate equal to some auxiliary variable. Each auxiliary variable can depend on any finite multiset of sum variables and stocks, and each sum variable can depend on any finite multiset of stocks.

Given an inflow $f \in F(\mathrm{inflow})$, we say that the stock $F(\mathrm{is})(f)$ is the **upstream** stock of the flow $F(\mathrm{if})(f)$. Similarly, given an outflow $f \in F(\mathrm{outflow})$, the stock $F(\mathrm{os})(f)$ is the **downstream** stock of the flow $F(\mathrm{of})(f)$. The injectivity of $F(\mathrm{if})$ and $F(\mathrm{of})$ ensure that each flow has at most one upstream stock and at most one downstream stock. Flows having an upstream stock but not a downstream stock or vice versa are called **partial flows**.

Following the ideas of Section 3.1, we can define a category $\mathsf{StockFlow}_f$ of full-fledged stock-flow diagrams. It is useful to glue together such diagrams not only along stocks but also along sum variables and sum links, for example to keep track of the total population in an epidemiological model. We can still do this using decorated cospans if we introduce the category $\mathsf{FinSet}^{\mathsf{G}}$, where G is the free category on this diagram:

$$\text{stock} \longleftarrow \text{sum link} \longrightarrow \text{sum variable}.$$

There is an evident inclusion functor $\iota\colon \mathsf{G} \to \mathsf{H}_f$, so any functor $F \in \mathsf{FinSet}^{\mathsf{H}_f}$ restricts to a functor $F \circ \iota \in \mathsf{FinSet}^{\mathsf{G}}$, and we define an **open** full-fledged stock-flow diagram to be a full-fledged stock-flow diagram (F, ϕ) equipped with a cospan



where $X, Y \in \mathsf{FinSet}^{\mathsf{G}}$. With this adjustment we can define a category $\mathsf{Open}(\mathsf{StockFlow}_f)$ of open full-fledged stock-flow diagrams following the ideas in Section 3.2. Most importantly, in analogy to Theorem 3.3, there is a functor

$$v\colon \mathsf{Open}(\mathsf{StockFlow}_f) \to \mathsf{Open}(\mathsf{Dynam})$$

providing a semantics for open full-fledged stock-flow diagrams. We have implemented full-fledged stock-flow diagrams and this semantics in our Julia package StockFlow (below) —but for simplicity, Section 4 only discusses the simpler stock-flow diagrams treated in Sections 3.1–3.3.

4 Implementing stock-flow diagrams in AlgebraicJulia

Existing tools for building stock-flow diagrams and simulating the systems they represent suffer from several limitations. In Section 2, we singled out two: an absence of compositionality, which it makes it difficult to build complex models in an intelligible manner, and a blurring of the distinction between syntax and semantics, which inhibits the reusability and interoperability of stock-flow diagrams in different contexts. In Section 3, we addressed both these problems at the mathematical level, the first by constructing a category of open stock-flow diagrams, and the second by constructing a functor from this category into a category of open dynamical systems, whose morphisms describe systems of differential equations. We now describe our implementation of these mathematical structures as new software for System Dynamics modeling.

Our software, available at `https://github.com/AlgebraicJulia/StockFlow.jl` as the open source package StockFlow, is implemented using AlgebraicJulia [1], a family of packages for applied category theory written in the Julia programming language [6]. The AlgebraicJulia ecosystem consists of Catlab, which implements many standard abstractions in category theory, and a collection of packages which apply these abstractions to specific domains of science and engineering. Most relevant to this article are AlgebraicDynamics [19], implementing open dynamical systems based on ordinary and delay differential equations, and AlgebraicPetri [18], implementing Petri nets with rates and their ODE semantics.

Existing capabilities within AlgebraicJulia, based on general category-theoretic abstractions, enable us to give an economical implementation of stock-flow diagrams. The combinatorial essence of stock-flow diagrams—what we called primitive stock-flow diagrams in Section 3.1—are set-valued functors on a certain category H, or H-sets. Such structures are encompassed by the paradigm of categorical databases, for which Catlab has extensive support [22]. Subject to one caveat, stock-flow diagrams—including the flow functions—can also be implemented as categorical databases. In this way, stock-flow diagrams become combinatorial data structures that can be manipulated algorithmically through high-level operations such as limits and colimits. Currently, Catlab has better support for structured cospans than decorated cospans. The latter have some theoretical advantages, as mentioned in Section 3.2, but luckily the two formalisms are equivalent for the tasks carried out here [4]. Thus, at present we use structured cospans to implement open stock-flow diagrams in StockFlow. We elaborate on this in the following subsections.

4.1 Stock-flow diagrams as categorical databases

In Section 3.1 we defined a primitive stock-flow diagram to be a finite H-set for a certain category H, called the **schema** for these diagrams. In Catlab, we present this schema as:

```
@present SchPrimitiveStockFlow(FreeSchema) begin
  (Stock, Flow, Link)::Ob
  (up, down)::Hom(Flow, Stock)
  src::Hom(Link, Stock)
  tgt::Hom(Link, Flow)
end
```

To define stock-flow diagrams, we need to add a data attribute for the flow functions. In general, data attributes [22] are a practical necessity and the main feature that distinguishes categorical databases from the standard mathematical notion of a C-set, meaning a functor from C to Set. In this case, we extend the schema with an attribute type and a data attribute:

```
@present SchStockFlow <: SchPrimitiveStockFlow begin
  FlowFunc::AttrType
  flow::Attr(Flow, FlowFunc)
end
```

Having defined the schema, we can generate a Julia data type for stock-flow diagrams with this single line of code:

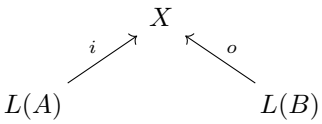
```
@acset_type StockFlow(SchStockFlow, index=[:up, :down, :src, :tgt])
```

where the indices are generated for morphisms in the schema to enable fast traversal of stock-flow diagrams. A stock-flow diagram will then have the Julia type `StockFlow{Function}`, where `Function` is the built-in type for functions in Julia. We see that there is a gap between the mathematical definition of stock-flow diagrams and the present implementation: the domains of the flow functions should be constrained by the links, but this constraint is not yet expressible in the data model supported by Catlab. In practice this is not a major obstacle to using stock-flow diagrams, but it could motivate future work toward increasing the expressivity of database schemas and instances in Catlab.

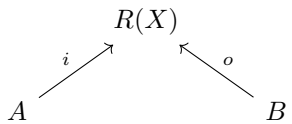
The full-fledged stock flow diagrams described in Section 3.5 are implemented similarly.

4.2 Composition using structured cospans

While the mathematics described in Section 3 uses decorated cospans, we can also describe open stock-flow diagrams using structured cospans [3, 4]. A structured cospan is a diagram of the form

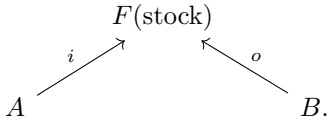


in some category X, where A, B are objects in some other category A, and $L\colon \mathsf{A} \to \mathsf{X}$ is a functor. When L is a left adjoint, we can equivalently think of a structured cospan as a diagram



where R is the right adjoint of L.

For example, we can take $\mathsf{A} = \mathsf{FinSet}$, take $\mathsf{X} = \mathsf{FinSet}^{\mathsf{H}}$, and take $R\colon \mathsf{FinSet}^{\mathsf{H}} \to \mathsf{FinSet}$ to be the functor sending any primitive stock-flow diagram to its set of stocks. In this case, a structured cospan amounts to an **open primitive stock-flow diagram**, that is a primitive stock-flow diagram $F\colon \mathsf{H} \to \mathsf{FinSet}$ together with functions



With more work we can define a structured cospan category equivalent to Open(StockFlow). The advantage of this change in viewpoint is that Catlab already provides a generic framework for working with structured cospans and multicospans—and it implements the composition operations available in both the hypergraph category of structured cospans and the operad algebra of structured multicospans. In addition, Catlab includes a concrete instantiation of structured cospans for systems defined by attributed C-sets. This makes it possible, for a broad class of systems, to use structured cospans in just a few lines of code. This is the approach taken in StockFlow and also its companion package AlgebraicPetri. In order to support the implementation of full-fledged stock-flow diagrams, we expanded the implementation of structured multicospans in Catlab so that the feet in a multicospan can be arbitrary C-sets as opposed to merely finite sets.

4.3 Composing epidemiological models: an example

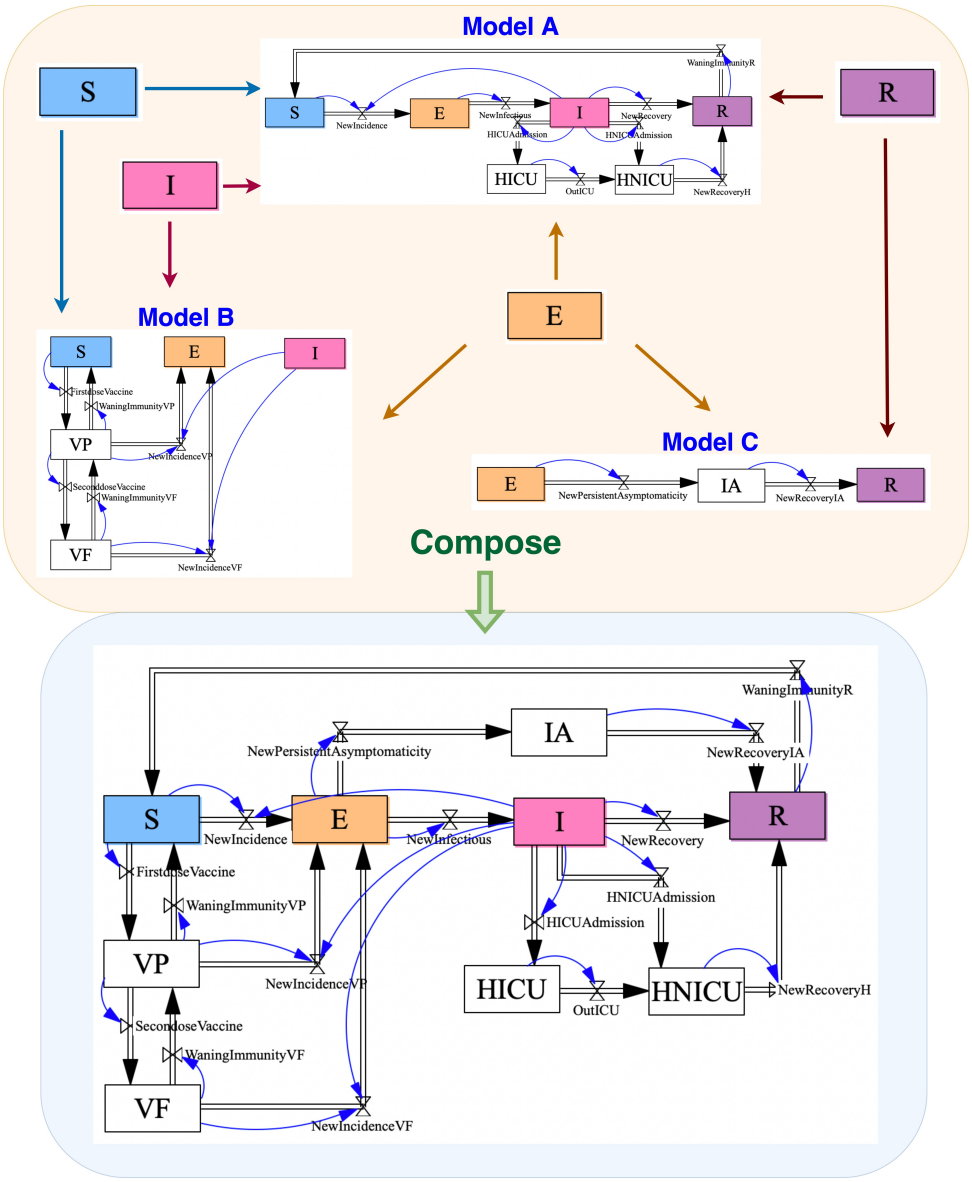


Figure 4: Example of composing a COVID-19 model from three smaller models

The Julia package StockFlow implements both the open stock-flow diagrams of Section 3.2 and the full-fledged open stock-flow diagrams of Section 3.5. We now illustrate the use of this package by constructing a simplified version of a COVID-19 model that has has been employed during

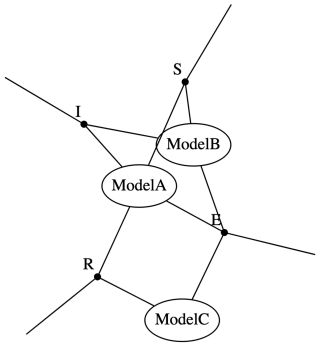


Figure 5: The undirected wiring diagram representing composing structured multicospans

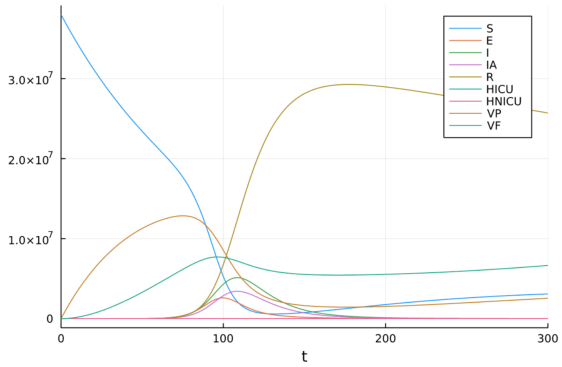


Figure 6: A simulation of the composite COVID-19 model

the pandemic for daily reporting and planning throughout the Province of Saskatchewan, and for weekly reporting by the Public Health Agency of Canada for each of Canada's ten provinces, as well as by First Nations and Inuit Health for reporting to provincial groupings of First Nations Reserves.

We build this simplified model as the composite of three component models: (A) a model of the natural history of infection, pathogen transmission, and hospitalization, (B) a model of vaccination, and (C) a model of the natural history of infection among asymptomatic or oligosymptomatic individuals. To exhibit the ideas with a minimum of complexity, we use the simpler open stock-flow diagrams discussed in Sections 3.1-3.4, not the full-fledged ones.

The top of Figure 4 shows the open stock-flow diagrams for three models. Although the flow functions are omitted from the figure, they are defined in the Jupyter notebook implementing this example.[2] Model (A) is the SEIRH (Susceptible-Exposed-Symptomatic Infectious-Recovered-Hospitalized) model, which simulates the disease transmission from, course of infection amongst, and hospitalization of symptomatically infected individuals. The stocks labelled "HICU" and "HNICU" represent the populations of hospitalized ICU and non-ICU patients, respectively. Model (B) characterizes vaccination-related dynamics. The stock "VP" represents individuals who are partially protected via vaccination, due to having been administered only a first dose or to waning of previously full vaccine-induced immunity. In contrast, the stock "VF" represents individuals who are fully vaccinated by virtue of having received two or more doses of the vaccine. Notably, neither partially or fully vaccinated individuals are considered fully protected from infection. Thus, there are flows from stock "VP" and "VF" to "E" that represent new infection of vaccinated individuals. Model (C) characterizes the natural history of infection in individuals who are persistently asymptomatic. The stock "IA" indicates the infected individuals without any symptoms.

[2]Readers interested in the code for this example can refer to https://github.com/AlgebraicJulia/StockFlow.jl/blob/master/examples/primitive_schema_examples/Covid19_composition_model_in_paper.ipynb on the GitHub repository for StockFlow.

The ODEs finally generated from the composite stock-flow diagram are as follows:

$$\dot{S} = \frac{R}{t_w} + \frac{V_P}{t_w} - \frac{\beta S I}{N} - r_v S \qquad \dot{E} = \frac{\beta S I}{N} + \frac{\beta(1-e_p) I V_P}{N} + \frac{\beta(1-e_f) I V_F}{N} - r_{ia} E - r_i E$$

$$\dot{I} = r_i E - \frac{I}{t_r} \qquad \dot{R} = \frac{(1-f_H) I}{t_r} + \frac{I_A}{t_r} + \frac{H_{\text{NICU}}}{t_H} - \frac{R}{t_w}$$

$$\dot{I_A} = r_{ia} E - \frac{I_A}{t_r} \qquad \dot{V_F} = r_v V_P - \frac{V_F}{t_w} - \frac{\beta(1-e_f) I V_F}{N}$$

$$\dot{H}_{\text{ICU}} = \frac{f_H f_{\text{ICU}} I}{t_r} - \frac{H_{\text{ICU}}}{t_{\text{ICU}}} \qquad \dot{V_P} = r_v S + \frac{V_F}{t_w} - \frac{V_P}{t_w} - r_v V_P - \frac{\beta(1-e_p) I V_P}{N}$$

$$\dot{H}_{\text{NICU}} = \frac{H_{\text{ICU}}}{t_{\text{ICU}}} + \frac{f_H(1-f_{\text{ICU}}) I}{t_r} - \frac{H_{\text{NICU}}}{t_H}$$

where for simplicity we use $1/t_r$ to stand for the rate at which infected individuals proceed to the next stage (stocks R, HICU or HNICU), and assume this is also the rate at which asymptomatic infected individuals go to the next stage (stock R). A plot of a solution of these equations is shown in Figure 6. In our software, the initial values and values of parameters are defined separately from the stock-flow diagram. This design enables the users to flexibly define and explore multiple scenarios involving the same dynamical system, in a manner similar to some existing stock-flow modeling packages. For example, the parameter values used in Figure 6 are from Canada's population. We can efficiently run this model on other populations (e.g., the United States) by changing these parameter values.[3]

This particular COVID-19 model simplifies the structure and assumptions of the model used in practice. Our example omits features such as characterization of active case-finding, diagnosis and reporting, mortality, and transmission by asymptomatic/oligosymptomatic individuals, because the simplified stock-flow diagrams do not support auxiliary variables, sum variables, and partial flows. However, our StockFlow package implements the full-fledged stock-flow diagrams defined in Section 3.5, and hence enables the application of these additional features.

4.4 Future work

Three lines of work are underway to extend the work described here: extending the Julia application programming interface (API), constructing a graphical user interface, and training modelers to use StockFlow.

For the first, key priorities include supporting within-diagram constants in the diagrams and allowing auxiliary variables to depend on other auxiliary variables in an acyclic fashion. Approaches are also being explored to allow hierarchical composition of diagrams and ensure consistency of the functions governing flows, in the sense of dimensional analysis.

Second, we aim to help modelers use the software without needing to know category theory. Thus, building atop the API, we are currently constructing a declarative, real-time graphical user interface (GUI) for collaboratively constructing, manipulating, composing and packaging stock-flow diagrams.

Third, we are training both students and professional modelers in the use of Stockflow, and will ramp up these efforts soon, once the GUI achieves sufficient functionality to offer practical utility. This will both build a user base and provide useful feedback as to how epidemiological modelers interact with the software.

Acknowledgments Author Osgood wishes to express his appreciation to SHA, PHAC and FNIH for support of varying elements of this work, to SYK, and to NSERC for support via the Discovery Grants program (RGPIN 2017-04647). Patterson acknowledges support from AFOSR (Award FA9550-20-1-0348). Baez and Libkind thank the Topos Institute for their support.

[3]Readers interested in the code for this example can refer to https://github.com/AlgebraicJulia/StockFlow.jl/blob/master/examples/primitive_schema_examples/Covid19_composition_model_in_paper.ipynb on the GitHub repository for StockFlow.

References

[1] AlgebraicJulia team. AlgebraicJulia: Bringing compositionality to technical computing. Available at https://www.algebraicjulia.org.

[2] R. M. Anderson and R. M. May. *Infectious Diseases of Humans: Dynamics and Control*. Oxford University Press, 1992.

[3] J. C. Baez and K. Courser. Structured cospans. *Theor. Appl. Categ.*, 35:1771–1822, 2020. Available at arXiv:1911.04630.

[4] J. C. Baez, K. Courser, and C. Vasilakopoulou. Structured versus decorated cospans. *Compositionality*, 4(3), 2022. Available at arXiv:2101.09363.

[5] J. C. Baez and B. S. Pollard. A compositional framework for reaction networks. *Reviews in Mathematical Physics*, 29(09):1750028, 2017. Available at arXiv:1704.02051.

[6] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah. Julia: A fresh approach to numerical computing. *SIAM Review*, 59(1):65–98, 2017.

[7] D. Destoumieux-Garzón, P. Mavingui, G. Boetsch, J. Boissier, F. Darriet, et al. The One Health concept: 10 years old and a long road ahead. *Frontiers in Veterinary Science*, page 14, 2018.

[8] O. Diekmann and J. A. P. Heesterbeek. *Mathematical Epidemiology of Infectious Diseases: Model Building, Analysis and Interpretation*, volume 5. John Wiley & Sons, 2000.

[9] V. Faghihi, A. R. Hessami, and D. N. Ford. Sustainable campus improvement program design using energy efficiency and conservation. *Journal of Cleaner Production*, 107:400–409, 2015.

[10] B. Fong. Decorated cospans. *Theor. Appl. Categ.*, 30(33):1096–1120, 2015. Available at arXiv:1502.00872.

[11] B. Fong and D. I. Spivak. Hypergraph categories, 2018. Available at arXiv:1305.0297.

[12] J. W. Forrester. *Industrial Dynamics*. Pegasus Communications, 1961.

[13] B. Guneralp, J. D. Sterman, N. P. Repenning, R.S. Langer, J. I. Rowe, and J.M. Yanni. Progress in eigenvalue elasticity analysis as a coherent loop dominance analysis tool. In *The 23rd International Conference of The System Dynamics Society*. System Dynamics Society, 2005.

[14] J. Hines. Molecules of structure: building blocks for System Dynamics models, version 2.02. Available at https://vensim.com/modeling-with-molecules-2-02/.

[15] P. S. Hovmand. *Community Based System Dynamics*. Springer, 2014.

[16] C. E. Kampmann. Feedback loop gains and system behavior (1996). *System Dynamics Review*, 28(4):370–395, 2012.

[17] W. O. Kermack and A. G. McKendrick. A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London. Series A*, 115(772):700–721, 1927.

[18] S. Libkind, A. Baas, M. Halter, E. Patterson, and J. Fairbanks. An algebraic framework for structured epidemic modeling, 2022. Available at arXiv:2203.16345.

[19] S. Libkind, A. Baas, E. Patterson, and J. Fairbanks. Operadic modeling of dynamical systems: mathematics and computation. Available at arXiv:2105.12282.

[20] K. H. Lich and J. Kuhlberg. Engaging stakeholders in mapping and modeling complex systems structure to inform population health research and action. In Y. Apostolopoulos, K. H. Lich, and M. K. Lemke, editors, *Complex Systems and Population Health*, page 119. Oxford University Press, 2020.

[21] J. S. Mackenzie and M. Jeggo. The One Health approach—why is it so important? *Tropical Medicine and Infectious Disease*, 4(2):88, 2019.

[22] E. Patterson, O. Lynch, and J. Fairbanks. Categorical data structures for technical computing, 2021. Available at arXiv:2106.04703.

[23] B. Richmond. Stella: Software for bringing System Dynamics to the other 98%. In *Proceedings of the 1985 International Conference of the System Dynamics Society: 1985 International System Dynamics Conference*, pages 706–718. System Dynamics Society, 1985.

[24] R. Ross. An application of the theory of probabilities to the study of a priori pathometry—Part I. *Proceedings of the Royal Society of London. Series A*, 92(638):204–230, 1916.

[25] R. Ross and H. P. Hudson. An application of the theory of probabilities to the study of a priori pathometry—Part II. *Proceedings of the Royal Society of London. Series A*, 93(650):212–225, 1917.

[26] M. Saleh, P. Davidsen, and K. Bayoumi. A comprehensive eigenvalue analysis of system dynamics models. In *Proceedings of the International System Dynamics Conference, The System Dynamics Society, Boston*, page 130. System Dynamics Society, 2005.

[27] D. I. Spivak. The operad of wiring diagrams: formalizing a graphical language for databases, recursion, and plug-and-play circuits, 2013. Available at arXiv:1305.0297.

[28] J. D. Sterman. Learning in and about complex systems. *System Dynamics Review*, 10(2-3):291–330, 1994.

[29] J. D. Sterman. *Business Dynamics*. McGraw-Hill, Inc., 2000.

[30] M. A. Trecker, D. J. Hogan, C. L. Waldner, J. R. Dillon, and N. D. Osgood. Revised simulation model does not predict rebound in gonorrhoea prevalence where core groups are treated in the presence of antimicrobial resistance. *Sexually Transmitted Infections*, 91(4):300–302, 2015.

[31] J. A. M. Vennix. *Group Model Building*. Chichester, 1996.