

Analyzing the topological structure of composite dynamical systems

Michael Robinson Michael L. Szulczewski
 James T. Thorson

September 2025

Abstract

This chapter explores dynamical structural equation models (DSEM s) and their nonlinear generalizations into sheaves of dynamical systems. It demonstrates these two disciplines on part of the food web in the Bering Sea. The translation from DSEM s to sheaves passes through a formal construction borrowed from electronics called a netlist that specifies how data route through a system. A sheaf can be considered a formal hypothesis about how variables interact, that then specifies how observations can be tested for consistency, how missing data can be inferred, and how uncertainty about the observations can be quantified. Sheaf modeling provides a coherent mathematical framework for studying the interaction of various dynamical subsystems that together determine a larger system.

Contents

1	Introduction	2
1.1	Related work	3
1.2	Contributions	4
1.3	Chapter outline	5
2	Dynamical modeling of ecosystems	5
2.1	DSEM background and motivation	5
2.2	Ecological background and the DSEM system for the Bering Sea	7

*Approved for Public Release by The MITRE Corporation; Distribution Unlimited. Public Release Case Number 25-2751. The author's affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the author. ©2025 The MITRE Corporation. ALL RIGHTS RESERVED.

3 Sheaf encodings of composite systems	8
3.1 Netlists	11
3.2 Sheaves and cosheaves	14
3.3 The netlist sheaf	18
3.4 Sheaves modeling autoregressive timeseries	25
4 Sheaf encoding of the Bering Sea	28
5 The topology of subsystems	33
5.1 Dynamical systems	34
5.2 The cosheaf endomorphism of invariant sets	35
5.3 Subsystem decomposition sheaf	37
6 Subsystems of the Bering Sea system	49
7 Conclusion	50

1 Introduction

Ecologists often study systems on spatial and temporal scales that cannot be experimentally manipulated (ecosystem processes are distributed across continents, and arise from evolutionary dynamics over millennia), and for which extrapolating the results of experiments at fine space-time scales is challenging [48]. These systems are also challenging to study because observational data can be noisy and sporadic. A third challenge is the presence of complex, causal relationships between system variables that can change over time.

Understanding the dynamics of these kind of large composite models is much easier reductively. Roughly speaking, a *subsystem* is a collection of state variables that makes sense as an independent dynamical system (Definition 20). Subsystems can be isolated for a variety of reasons, in addition to spatial or temporal separation. Regardless of the reason for the isolation, there is a canonical way to write a dynamical system in terms of its subsystems. This subsystem decomposition is a convenient way to explore dynamical summaries of the original model (Section 5).

This chapter explores dynamical structural equation models (DSEMs) and their nonlinear generalizations via a topologically motivated translation into *sheaves of dynamical systems* (Sections 3 and 5). Sheaves are a strict generalization of DSEMs into nonlinear models, which they losslessly represent (Theorem 6). The translation of DSEMs into sheaves follows a clear graphical recipe, which allows handling observations in three ways: (1) as individual observations, (2) as individual timeseries, and (3) as collections of dynamically related timeseries.

The translation from DSEMs to sheaves passes through a formal construction borrowed from electronics called a *netlist* that specifies how data route through a system. Because the netlist and sheaf methodology is explicit and graphical, we include several illustrative examples (Figures 3 and 5). One real-world example

involves part of the food web in the Bering Sea (Figure 1; Sections 2.2, 4, and 6).

Sheaves provide many advantages to a modeler. They enable exploring the impact of uncertainty in various ways. They support inference of missing or erroneous data, including system parameters and coefficients (Section 3). They also enable forecasts and retrocasts through the same “interface,” namely *consistency radius optimization* (Section 4).

Sheaves also highlight the importance of the original DSEM in model summarization. Using the sheaf of subsystems, Corollary 21 shows that the subsystems of a DSEM can be “read off” its associated graph. This is applied to the Bering Sea ecosystem model in Section 6.

1.1 Related work

The challenges in modeling ecological systems have motivated interest in structural causal models (SCMs) [31]. SCMs can be fit to observational data in space and time, and can decompose the total effect of one variable on another via a combination of direct and indirect effects [16, 5]. Recently, SCMs have been adapted to the analysis of ecological time series via DSEMs [47].

The key idea behind SCMs is that systems can be understood by decomposing them into coherent subsystems. The idea of reducing systems into subsystems has a long history, with general mathematical descriptions of composite systems given by the field of cybernetics, for which Heylighen and Joslyn [17] and Ashby [6] are good introductions. Beyond cybernetics, the study of subsystems of dynamical models [50] has occurred in many fields, including manufacturing and operations research [49, 45, 21], design [2], statistical physics [51], mathematical systems [9], biology [26], and chemistry [18].

Although algorithmic and systematic decomposition of systems into subsystems have become common since the dawn of cybernetics, it remains challenging. Maier et al. [27] laments, “Even though abstraction is frequently mentioned with regards to modeling and simulation, formal definitions are harder to find.” One challenge is that decompositions are often not unique: for example, one may choose to group state variables based on constraints rather than functional units [8, 24]. These choices are important because they drive the usefulness of the decomposition [27]. For example, overlapping, rather than disjoint, subsystem decompositions are useful for analyzing stability of an entire system [40, 4].

We argue that a properly general and formal definition of a subsystem decomposition must support overlappingness, non-uniqueness, and ambiguous granularity. Because the collection of all subsystems forms a *mathematical sheaf* (Definition 21), this implies that seeking disjoint, unambiguous subsystems (as is often done) is fraught.

Aspects of the formalism we introduce in this chapter are not entirely novel. For instance, Hiroto et al. [18] defines a *CRN morphism* that is a special case of our Definition 20. Additionally, the sheaf of subsystems is based upon a clear graphical representation, which is well known in the analysis of software

[29, 1]. Moreover, Abadi and Lamport [1] uses the term *refinement mapping*, which evokes the analogous term from sheaves (Definition 7).

Roughly dual to the notion of a subsystem is that of an *invariant set* of a dynamical system (our Definition 20 makes this a *true* duality). Invariant sets are widely used in dynamical systems [44], where they generalize equilibrium sets and attractors. For linear systems, duality between invariant sets and subsystems is immediate and useful. For instance, the design structure matrix [43] yields invariant sets, giving a clear duality to subsystems.

Finally, we note that the discipline of modeling a system’s state via a decomposition into subsystems of state equations is explained in detail in Robinson [34, Sec. 5], and is specialized to subsystem graphs in Kearney et al. [22]. In Kearney et al. [22], the dynamics are specified locally and are much easier to specify due to the fact that the system is given a graph structure.

1.2 Contributions

This chapter provides an introduction to the discipline of modeling and analyzing a composite system using the language and tools of topology, centered around *sheaves*. Sheaf modeling provides a coherent mathematical framework for studying the complicated interaction of various dynamical subsystems that together determine a larger system. The guiding principles of sheaf modeling are that

- a *sheaf* represents a hypothesis about how variables will interact (Definition 10),
- a non-global *assignment* represents the observations collected on the variables in its support (Definition 8),
- *minimizing consistency radius* estimates values of the variables and parameters that were not observed (Definition 11), and
- the *minimal consistency radius* is a measure of the consistency between the observations and the hypothesis.

This chapter shows that when a dynamical system is described by a linear system, there are three sheaves that provide increasingly granular data about the interactions between variables:

1. the *sheaf of subsystems* (Definition 21),
2. the *netlist sheaf* with timeseries as stalks (Definition 13), and
3. the *netlist sheaf* with additional stalks for individual observations (Definition 14).

1.3 Chapter outline

Section 2 describes a model of a food web in the Bering Sea, which we use to illustrate the use of sheaves. This system is large enough to exhibit interesting structures, and corresponding observational data [47] are available. Additionally, we present a graphical causal modeling discipline called *dynamical structural equation modeling* that serves as an entry point into the more sophisticated (but admittedly less familiar) topological sheaf models. As is later shown in Section 3, sheaves are a strict generalization of DSEM. Sheaves can be nonlinear, whereas DSEM are linear.

Section 3 constructs sheaves that model composite systems, and develops the main inferential tool, *consistency radius minimization*. Section 3 is self-contained, as all of the mathematical background necessary to understand the constructions is introduced as it is needed. Small concrete examples of the construction and use of sheaf models are presented to build intuition as well.

In Section 4, we revisit the ecological model from Section 2 using the sheaf tools from Section 3. The interface between observational data, sheaves, and their inference tools is explored in detail. Moreover, we compare differences between the DSEM and sheaf approaches in detail.

Section 5 introduces the idea of a general topological *dynamical system*, and shows that every dynamical system induces a *sheaf of subsystems* and a *cosheaf of invariant sets*, which form a dual pair. We prove that under appropriate conditions, the subsystems of a DSEM can be “read off” rather directly (Corollary 21). This provides theoretical justification for why DSEM are a useful way to describe a composite linear system by way of its subsystems.

Section 6 revisits the ecological model from Section 2 once again. Because the model satisfies the hypothesis of Corollary 21, we are able to present a clear representation of all the subsystems present in the model.

Finally, Section 7 concludes the chapter with practical advice for modelers and a brief discussion of future research work.

2 Dynamical modeling of ecosystems

This section begins with a brief recount of modeling linear dynamical systems according to an underlying graph structure, and then presents a representative ecosystem model that will be revisited several times in the chapter.

2.1 DSEM background and motivation

Definition 1. Given a set of *variables* $X = \{x_1, \dots, x_J\}$, and a set $Y = \{t_1 < \dots < t_T\}$ of real valued *time lags*, a *dynamic structural equation model (DSEM)* consists of an edge-labeled directed graph G with vertices $X \times Y$ and edges E such that

Causality The presence of an edge $(x_{j_1}, t_{k_1}) \rightarrow (x_{j_2}, t_{k_2})$ implies that $t_{k_1} \leq t_{k_2}$, and

Linearity Each edge $(x_{j_1}, t_{k_1}) \rightarrow (x_{j_2}, t_{k_2})$ is labeled with a real number $\gamma_{j_1, k_1, j_2, k_2}$ called the *path coefficient* for that edge.

The absence of an edge in the graph is assumed to be equivalent to assigning a path coefficient of 0. For brevity, we write a vertex (x_j, t_k) simply as $x_{j,k}$.

The variables in a DSEM are to be interpreted as $C^1(\mathbb{R})$ functions, which are continuous timeseries. A directed edge $x_{i,j} \rightarrow x_{i',j'}$ is to be interpreted as specifying that a change in x_i causes a proportional (linear) change in $x_{i'}$ after a lag of $(t_{j'} - t_j)$, with magnitude controlled by the associated path coefficient $\gamma_{i,j,i',j'}$. Under this interpretation, a DSEM implies that a first order system of linear differential equations governs the values of the variables:

$$\frac{dx_k(\tau - t_\ell)}{d\tau} = \sum_{i=1}^J \sum_{j=1}^T \gamma_{k,\ell,i,j} x_i(\tau - t_j). \quad (1)$$

In what follows, we will refer to solutions of Equation 1 as *solutions to the DSEM*.

In the use of Equation (1) with observational data, there are two kinds of errors that need to be considered: *exogenous* errors and *measurement* errors. Exogenous errors accumulate, which means that an error in the value of a variable x_k at given time τ impacts the value of x_k at all later times. As a result, there is a dependence between the exogenous errors of x_k at different times. In contrast, measurement errors at different times are assumed to be independent.

Exogenous errors will be represented by an additive term, $\epsilon_{k,\ell}$, resulting in

$$\frac{dx_k(\tau - t_\ell)}{d\tau} = \sum_{i=1}^J \sum_{j=1}^T \gamma_{k,\ell,i,j} x_i(\tau - t_j) + \epsilon_{k,\ell}(\tau). \quad (2)$$

We can approximate the solution to Equation (2) using the one-step backwards Euler method with time step h ,

$$\frac{dx_k(\tau - t_\ell)}{d\tau} \approx \frac{1}{h} (x_k(\tau - t_\ell) - x_k(\tau - t_\ell - h)),$$

so that Equation (2) becomes a system of $M = TJ$ linear algebraic equations,

$$x_k(\tau - t_\ell) \approx x_k(\tau - t_\ell - h) + h \sum_{i=1}^J \sum_{j=1}^T \gamma_{k,\ell,i,j} x_i(\tau - t_j) + h \epsilon_{k,\ell}(\tau). \quad (3)$$

If we fix a value of τ and organize the set of values $\{x_k(\tau - t_\ell)\}$ into a vector \mathbf{X} of length M , Equation (3) can be compactly written in matrix form as

$$\mathbf{X} \approx \mathbf{P}\mathbf{X} + \mathbf{E}, \quad (4)$$

where the entries of the $M \times M$ *path coefficient matrix* \mathbf{P} contain both the path coefficients from the DSEM (scaled by h) and the additional nonzero entries due

the $x_k(\tau - t_\ell - h)$ terms. In what follows, we will take $h = 1$, so that the path coefficients in the DSEM appear unchanged as elements of the matrix \mathbf{P} .

To obtain the path coefficient matrix \mathbf{P} from observations of \mathbf{X} , we assume the exogenous errors follow a multivariate normal distribution with variance \mathbf{V} , namely

$$\mathbf{E} \sim \text{MVN}(\mathbf{0}, \mathbf{V}),$$

where \mathbf{E} is the length M vector containing errors ϵ_{tj} .

Equation (4) can then be re-arranged to yield a Gaussian Markov random field,

$$\mathbf{X} \sim \text{MVN}(\mathbf{0}, \mathbf{Q}^{-1}) \tag{5}$$

$$\mathbf{Q} = (\text{id} - \mathbf{P}^T)\mathbf{V}^{-1}(\text{id} - \mathbf{P}), \tag{6}$$

where id is the identity matrix. The path coefficient matrix \mathbf{P} can be obtained from the Cholesky decomposition of \mathbf{Q} . The necessary calculations can be efficiently evaluated using sparse libraries, such as `Eigen` and `CHOLMOD` [11], and we use Template Model Builder [25] to incorporate automatic differentiation and implement the Laplace approximation [39] to marginalize across random effects.

Now we address measurement errors. Assume the distribution of measurement errors of the variable x_k is given by a distribution f_j parameterized by θ_j at time t_j . (If one does not wish to model measurement errors explicitly, so that measurement errors are entirely captured by the exogenous error term, this is obtained by choosing f_j so that it has probability 1 at $x_{k,j}$.) Let us write $y_{k,j}$ for the observation of the variable $x_{k,j}$. We therefore can express the mean of the distribution of $y_{k,j}$ through a link function g_j , via

$$y_{k,j} \sim f_j(g_j^{-1}(\mu_j + x_{k,j}), \theta_j),$$

where μ_j is the true mean.

The clearest way to obtain the required sparsity in solving for \mathbf{P} is to assume additionally that the measurement errors for a given variable do not depend on time t_j . Let \mathbf{G} be the $J \times J$ matrix that is diagonal, and whose diagonal terms are given by the link functions g_j . With this in hand, \mathbf{V} takes the form

$$\mathbf{V} = \text{id}_{T \times T} \otimes \mathbf{G}\mathbf{G}^T, \tag{7}$$

where \otimes is the Kronecker product. This implies that \mathbf{V} is block diagonal, and is thereby efficient to invert.

2.2 Ecological background and the DSEM system for the Bering Sea

To demonstrate the use of sheaves for dynamical systems, we make a sheaf from a DSEM for ecological mechanisms linking regional oceanography (winter sea ice extent) to first-winter survival of juvenile Alaska pollock (*Gadus chalcogrammus*) in the eastern and northern Bering Sea [47]. The model starts

by specifying that abundance of age-0 pollock R_t (termed “age-0 recruitment”) can be predicted from the biomass of spawning females S_t in a given year t :

$$R_t = S_t e^{\alpha - \beta S_t + \epsilon_t} \quad (8)$$

where e^α is the maximum expected recruits per spawning biomass, β is the expected density-dependent decrease in recruits per spawning biomass as biomass increases, and ϵ_t is additional process error representing unmodeled variation in recruitment. This “Ricker stock-recruit model” [33] has been used for over 70 years to represent density-dependent changes in juvenile survival, and as the basis for defining biological reference points that are used worldwide to identify sustainable levels of fishing mortality [42]. The Ricker model is expected to arise for species where adult abundance directly impacts juvenile survival—for example, due to cannibalism or interference competition [15]. Alaska pollock are cannibalistic, so the Ricker model has theoretical justification. Usefully, the Ricker model can be linearized as:

$$\log\left(\frac{R_t}{S_t}\right) = \alpha - \beta S_t + \epsilon_t \quad (9)$$

and a DSEM can be used to elaborate the mechanisms that contribute to process errors ϵ_t based on prior ecological hypotheses.

The DSEM we translate into a sheaf was previously developed by Thorson et al. [47]. It specifies that variable winter sea ice formation (*SeaIce*) drives residual variation in log-recruits per spawning biomass (*Survival*) via two paths, mediated by sea-ice impacts on either copepod abundance (*Copepod*) or krill abundance (*Krill*), and resulting consumption by juvenile pollock. See Table 1 and 2 for more details on the variables and mechanisms in the model. The DSEM includes a first-order autoregressive term for each variable, to allow the model to correct for bias that can arise when correlating variables that follow an autoregressive process (summarized in [28]). This first-order autoregression can also be interpreted to represent Gompertz density-dependence and therefore has some scientific interest [23], although it is not further discussed here.

3 Sheaf encodings of composite systems

In this section, we explain how to construct a *netlist sheaf* whose *global sections* correspond bijectively to the solutions of a DSEM. This is performed in two main steps: (1) the DSEM is translated into a *netlist*, and (2) the netlist is translated into the *netlist sheaf*. Since the machinery of sheaves is not in wide usage, Section 3.2 provides the necessary background.

With the machinery and the translation in place, Theorem 6 establishes that the two representations, the DSEM and the netlist sheaf, are equivalent. The *global sections* of the netlist sheaf are in bijective correspondence with solutions to the DSEM. Moreover, a process called *consistency radius minimization* in the sheaf finds approximate solutions to the DSEM, and this process is robust to perturbations.

Table 1: Variables that describe Alaska pollock recruitment used in the DSEM and sheaf. All except *Spawners* are transformed by the natural logarithm and then centered (i.e., subtracted by their mean) prior to analysis. Timeseries of the variables are taken from [47].

Name	Description
<i>SeaIce</i>	Average spatial extent (km^2) of sea ice in the Bering Sea from Oct.15 to Dec.15 the preceding year, from the National Snow and Ice Center’s Sea Ice Index, Version 3 [14]
<i>ColdPool</i>	Spatial extent (km^2) of waters with temperatures $\leq 2^\circ\text{C}$ near the seafloor, interpolated from measurements by the eastern Bering Sea bottom trawl survey and compiled in R-package “coldpool” [37]
<i>Spawners</i>	Female spawning biomass (in units of 10^6 kg) for Alaska pollock in the eastern and northern Bering Sea, estimated by the age-structured stock assessment model used for management [20]
<i>Survival</i>	Age-0 recruits per spawning biomass (10^3 count/kg), calculated as age-1 abundance the following year (10^9 count) estimated by the age-structured stock assessment model [20] divided by <i>Spawners</i>
<i>Copepods</i>	Density of $\geq 2 \text{ mm}$ copepods (count/m^3) from the Bering Sea middle shelf [38], averaged across samples obtained during the fall mooring cruise along the 70 isobath from Sept. to early Oct. [12] (calculated by Dave Kimmel, pers. comm.)
<i>Krill</i>	Index of euphausiid abundance (count/m^3) [32] obtained from backscatter measured during a summer acoustic-trawl survey in the eastern Bering Sea and converted to abundance using a target-strength model [41]
<i>DietCopepods</i>	Biomass of copepods divided by total prey biomass in juvenile stomach samples (kg/kg), calculated from a fall surface-trawl survey in the eastern Bering Sea [30]. For each surface trawl, total catch of juvenile pollock is weighed, individual pollock are subsampled, and stomach contents for subsampled individuals are identified to species and weighed. The diet index is calculated as the average across subsampled stomachs, weighted by the catch of juvenile pollock in the associated surface trawl sample (calculated by Alex Andrews, pers. comm.).
<i>DietKrill</i>	Same as <i>DietCopepods</i> , but for euphausiids (krill)

Table 2: List of path coefficients connecting variables (defined in Table 1), supporting ecological hypotheses, and hypothesized sign for the path used in the DSEM case study. We also include a first-order autoregressive term for each variable (i.e., 8 AR1 coefficients, not shown here) for reasons discussed in Section 2.2.

Path	Ecological hypothesis and evidence	Sign
$SeaIce \rightarrow ColdPool$	Sea ice formation ($SeaIce$) causes variation in summer cold-pool extent ($ColdPool$)	+
$ColdPool \rightarrow Copepods$	Warmer water temperatures ($ColdPool$) result in higher copepod metabolism and therefore earlier onset of winter diapause, resulting in a decrease in fall copepod abundance ($Copepods$) [10]	+
$ColdPool \rightarrow Krill$	Water temperatures ($ColdPool$) might affect krill overwinter survival, affecting summer krill abundance ($Krill$)	?
$Copepods \rightarrow DietCopepods$	Increased copepod abundance will result in them being a higher proportion of age-0 fall stomach contents ($DietCopepods$), due to pollock being hypothesized to be a relative non-selective predator	+
$Krill \rightarrow DietKrill$	Same as $Copepods \rightarrow DietCopepods$ but for krill	+
$DietCopepods \rightarrow Survival$	Increased fraction of fall diet from copepods ($Copepods$) will increase energy reserves and subsequent survival of age-0 over their first winter ($Survival$) [19]	+
$DietKrill \rightarrow Survival$	Same as $DietCopepods \rightarrow Survival$, but for krill	+
$Spawners \rightarrow Survival$	Increased spawning biomass ($Spawners$) will cause a density-dependent decrease in survival ($Survival$) [15]	-

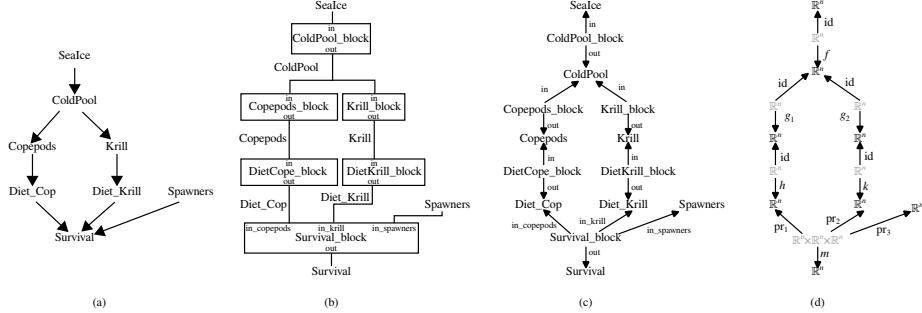


Figure 1: (a) The DSEM model for part of a food web in the Bering Sea [46], (b) its wiring hypergraph, (c) its netlist graph, and (d) its sheaf diagram. The arrows in each subfigure have different meanings: in (a) they denote causal, linear relationships (Sec. 2.1); in (c), they point from netlist parts to nets (Sec. 3.1); and in (d), they denote restriction functions (Sec. 3.2). While the DSEM also estimates a first-order autoregressive term for each variable (not shown in (a) to simplify presentation), there is no autoregressive structure assumed in the sheaf model. This remedied in Section 3.4.

Throughout this section, we refer to Figure 1 for intuition. Figure 1(a) shows the DSEM for part of the food web in the Bering Sea. The DSEM-to-netlist translation, described in Section 3.1, results in Figure 1(b). Figure 1(c) shows a different representation of the netlist that is more expedient for the construction of the netlist sheaf. Proposition 3 establishes that the two representations of netlists (Figures 1(b)–(c)) determine each other, so we may use whichever is more convenient. Finally, the netlist-to-sheaf translation, described in Section 3, results in Figure 1(d). Section 3.4 shows how to encode autoregressive timeseries models as netlist sheaves, which ultimately makes handling missing data both transparent and automatic within the netlist sheaf.

3.1 Netlists

The term “netlist” appears to have entered the technical lexicon in the early days of computing, when IBM started to automate the wiring of mainframe back planes [3]. Since that time, the term “netlist” has been in wide usage but often without a precise definition. In order to formalize the concept, we say that a *netlist* describes a system of *parts* interconnected with *nets*, which carry *time-varying signals* (briefly, *variables*).

Each *variable* consists of the specification of a set of possible *values* for a net. In this chapter, the values for a variable in a net are initially assumed to be continuous timeseries, usually of the form $C^1(\mathbb{R})$. We will also consider sampled timeseries of the form \mathbb{R}^n , where n is the length of the timeseries. In Section 3.4, we show how to handle missing values in such a timeseries.

Each part has a number of *ports*, to which connections can be made. Each *port* is either an *output*, which means that it determines the value of the variable

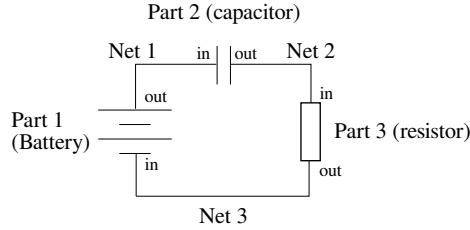


Figure 2: A netlist for an electric circuit, described in Example 1.

of a net connected to it, or an *input*, which means that it does not determine the value of a net connected to it.

Each *net* specifies that a collection of distinct ports on a pair of parts (which need not be distinct) are connected, with the requirement that not more than one of these ports be an output. Finally, each part specifies an *input-output function* for each output port. The domain of an input-output function is from the product of the set of its input variables, and its codomain (range) is the set of output variables at the output port.

This formulation leaves open the possibility of nets that are not attached to any output ports, which are called *external inputs*, and nets which are not attached to any input ports, which are called *external outputs*. Clearly each external output must attach to exactly one port, which must be an output port.

Example 1. Figure 2 shows an electrical circuit with three parts: a battery, a capacitor, and a resistor. These parts are connected to each other by three nets:

1. Connecting the positive (output) port of the battery to the input port of the capacitor,
2. Connecting the output port of the capacitor to the input port of the resistor, and
3. Connecting the output port of the resistor to the input port of the battery.

The values of the variables on the nets specify electrical currents flowing along them. We note that the labeling ports as “input” and “output” in this kind of circuit is arbitrary, since the electrical current can flow in either direction along a net. The input-output functions simply recount classical Ohm’s law for each of the parts in the circuit. This circuit contains no external inputs nor external outputs.

A DSEM graph can be translated into a netlist via the following construction.

Definition 2. Given a DSEM, its corresponding netlist is given by the following recipe:

- each DSEM variable (node) becomes a net,

- each DSEM variable with more than one input becomes a part,
- each net is connected to input ports via its out-neighbors,
- each net is connected to output ports via matching the name of the net to the part with the same name (if any exist), and
- the part’s input-output function is collected from the matrix block in Equation (4) corresponding to the input and output variables.

There are two combinatorial structures associated to a netlist, the *wiring hypergraph* and the *netlist graph*.

Definition 3. The *wiring hypergraph* of a netlist is a vertex- and edge-labeled partition-directed multi-hypergraph that has a vertex for each part and an hyperedge for each net.

The label on each vertex is simply the name of the part corresponding to that vertex.

The vertices within a hyperedge correspond to the parts connected to the corresponding net. The label on each hyperedge is an ordered triple, consisting of the inputs port of the net (if any), the output port of the net (if any), and the variable name of the net. The partition direction of each hyperedge separates the output port from the input ports; either of these may be empty.

Because the labeling on the wiring hypergraph is complicated, we represent it with a standard visual grammar borrowed from electronics. Each part is represented by a rectangle with its label in the center of the rectangle. Each net is drawn as a path (with right-angle bends as needed) to connect the corresponding parts. If a net has more than two ports, the path is drawn as a tree structure. The label of the variable of the net is shown next to the path, but the name of the net’s input and output ports are shown inside the connected parts’ rectangles, around the edge of the rectangle. The input-output functions are not shown explicitly.

Figure 1(b) shows the wiring hypergraph for the netlist constructed using Definition 2 for the Bering Sea DSEM. Notice that the net *ColdPool* corresponds to a hyperedge of size 3 in the wiring hypergraph, because it is connected to one output port and two input ports.

Proposition 1. *The solutions to a DSEM are in bijective correspondence with labelings of the nets with values of variables that are consistent with the netlist’s input-output functions.*

Proof. The solutions to the DSEM are characterized by Equation (4), which is a matrix block assembly of everything that is needed to construct the netlist.

Assume we have a set of variables for all nets that are consistent with the input-output functions. As noted above, each variable takes values in a set of the form $C^1(\mathbb{R})$. On the other hand, each input-output function was constructed from a matrix block in Equation (4). Because all of the DSEM variables appear as nets in the netlist, all such matrix blocks appear as input-output functions

somewhere in the netlist. This means that Equation (4) is satisfied by construction.

Assume that we have a solution to Equation (4). Definition 2 constructed the input-output function from the subblock of Equation (4), so there is nothing further to prove. \square

The wiring hypergraph is closely related to the DSEM, but for constructing the netlist sheaf in Section 3, it is more convenient to use another combinatorial representation.

Definition 4. The *netlist graph* is a vertex- and edge-labeled directed graph that has a vertex for each part, a vertex for each variable, and two edges for each net. The label on a vertex is simply the name of the corresponding part or variable. The two edges for each net are defined as follows. The first edge is labeled with the input port of the net, and leads from that corresponding part to the net. The second edge is labeled with the output port of the net, and leads from that corresponding part to the net.

Figure 1(c) shows the netlist graph for the Bering Sea example.

Corollary 2. *The netlist graph is a directed acyclic graph, and induces a pre-order on the set of parts and variables. In the preorder, each variable is above the parts to which it is connected.*

Proposition 3. *The netlist graph is the incidence bipartite graph of the wiring hypergraph, whose edges are labeled by projecting out the first and second components of the labels of the hyperedges. Consequently, the netlist graph and the wiring hypergraph determine each other fully.*

As we will see, the correspondence between the wiring hypergraph and the netlist graph is convenient. Although Proposition 1 showed that the wiring hypergraph is most closely related to the DSEM, we will later show that the netlist graph is most closely related to the netlist sheaf (Theorem 6).

3.2 Sheaves and cosheaves

Sheaves and cosheaves are topological constructions that allow one to study the local consistency structure of a model. In the case of a DSEM, locality is useful because variables that are near one another in the graph are likely to be related. This nearness can be most easily formalized by using the netlist graph defined in the previous section.

Since the netlist graph is a directed acyclic graph, it naturally induces a pre-ordered set on the vertices. That is, if $a \rightarrow b$ in a directed graph, we define $a \leq b$. When the graph is directed and acyclic, generalizing \leq to paths within the graph results in a relation \leq that is reflexive and transitive. Pre-ordered sets have a natural notion of neighborhoods, hence a natural topology.

A *topological space* is a mathematical formalism that captures the notion of “neighborhoods.”

Definition 5. A *topology* on an arbitrary set X is a collection \mathcal{T} of subsets of X satisfying the following four axioms:

Empty set The empty set \emptyset is an element of \mathcal{T} ,

Whole set The set X is an element of \mathcal{T} ,

Finite intersection If U and V are elements of \mathcal{T} , then $U \cap V$ is an element of \mathcal{T} , and

Arbitrary union If $\mathcal{U} \subseteq \mathcal{T}$ then $\cup \mathcal{U}$ is an element of \mathcal{T} .

The ordered pair (X, \mathcal{T}) is called a *topological space*.

Often, rather than specifying \mathcal{T} directly, we specify a collection of subsets \mathcal{U} of X that *generate* the topology, which is the smallest topology (in the sense of inclusion) that contains \mathcal{U} .

The following are elementary examples of topological spaces,

Discrete topology For any set X , let \mathcal{T} be the power set of X ,

Trivial topology For any set X , let $\mathcal{T} = \{\emptyset, X\}$,

Euclidean topology For $X = \mathbb{R}$, the usual topology \mathcal{T} is generated by the set of open intervals (a, b) for $a < b \in \mathbb{R}$.

Additionally, there is a powerful combinatorial theory of topological spaces (X, \mathcal{T}) in which the topology \mathcal{T} is a finite set [7]. For our purposes, the most interesting of these *finite topological spaces* are those that arise naturally from a pre-ordered set, given by the definition below.

Definition 6. Suppose that (P, \leq) is a pre-ordered set, which is to say that \leq is a reflexive and transitive relation. The *Alexandrov topology* $\text{Alex}(P, \leq)$ on (P, \leq) is the topology generated by all subsets of P of the form $U_x = \{y \in P : y \leq x\}$.

The idea of sheaves and cosheaves is that each *open set*—an element of the topology—is associated with a set of values, called the stalk (for sheaves) or costalk (for cosheaves).

Definition 7. Suppose (X, \mathcal{T}) is a topological space. A *presheaf* \mathcal{S} of sets on (X, \mathcal{T}) consists of the following specification:

1. For each open set $U \in \mathcal{T}$, a set $\mathcal{S}(U)$, called the *stalk at U* ,
2. For each pair of open sets $U \subseteq V$, there is a function $\mathcal{S}(U \subseteq V) : \mathcal{S}(V) \rightarrow \mathcal{S}(U)$, called a *restriction function* (or just a *restriction*), such that
3. For each triple $U \subseteq V \subseteq W$ of open sets, $\mathcal{S}(U \subseteq W) = \mathcal{S}(U \subseteq V) \circ \mathcal{S}(V \subseteq W)$ and
4. $\mathcal{S}(U \subseteq U)$ is the identity function.

Dually, a *precosheaf* \mathfrak{C} of sets on (X, \mathcal{T}) consists of the opposite specification:

1. For each open set $U \in \mathcal{T}$, a set $\mathfrak{C}(U)$, called the *costalk at U*,
2. For each pair of open sets $U \subseteq V$, there is a function $\mathfrak{C}(U \subseteq V) : \mathfrak{C}(U) \rightarrow \mathfrak{C}(V)$, called an *extension function* (or just a *extension*), such that
3. For each triple $U \subseteq V \subseteq W$ of open sets, $\mathfrak{C}(U \subseteq V \subseteq W) = \mathfrak{C}(V \subseteq W) \circ \mathfrak{C}(U \subseteq V)$ and
4. $\mathfrak{C}(U \subseteq U)$ is the identity function.

If for every $U \in \mathcal{T}$ there is a pseudometric d_U on the (co)stalk at U , and each restriction (or extension) is continuous with respect to the corresponding pseudometrics, we call the entire collection of data a *pre(co)sheaf of pseudometric spaces*.

As Definition 7 makes clear, pre(co)sheaves on a topological space are only sensitive to the poset of open sets, and *not* to the points in those open sets. In our context, the set of values should be interpreted as the set of values that a collection of variables in a DSEM can take.

Definition 8. Suppose \mathcal{S} is a presheaf on a topological space (X, \mathcal{T}) . An *assignment* a supported on $\mathcal{U} \subseteq \mathcal{T}$ is an element of the direct product, $\prod_{U \in \mathcal{U}} \mathcal{S}(U)$.

The direct product is in general *not* the direct sum, since the topology may be infinite! For this reason, dually, if \mathfrak{C} is a precosheaf on (X, \mathcal{T}) , then a *coassignment supported on $\mathcal{U} \subseteq \mathcal{T}$* is an element of

$$\left(\bigsqcup_{U \in \mathcal{U}} \mathfrak{C}(U) \right).$$

If $\mathcal{U} = \mathcal{T}$, we usually say that the (co)assignment is *global*.

(Co)assignments may or may not be consistent with their pre(co)sheaf structure. When they are fully consistent, we highlight this fact by calling them (co)sections.

Definition 9. A *global section* of a presheaf \mathcal{S} on a topological space (X, \mathcal{T}) is a global assignment s such that for all open $V \subseteq U$ then $\mathcal{S}(V \subseteq U)(s(U)) = s(V)$.

Dually, a *global cosection* of a precosheaf \mathfrak{C} on a topological space is a global coassignment c of the disjoint union under an equivalence,

$$\mathfrak{C}(X) = \left(\bigsqcup_{U \text{ open}} \mathfrak{C}(U) \right) / \sim,$$

where \sim is the equivalence relation generated by $c_1 \sim c_2$ whenever $c_1 \in \mathfrak{C}(U_1)$, $c_2 \in \mathfrak{C}(U_2)$, with $U_1 \subseteq U_2$, and $(\mathfrak{C}(U_1 \subseteq U_2))(c_1) = c_2$.

Local (co)sections are defined similarly, but refers to some collection \mathcal{U} of open sets.

Intuitively, a (co)section corresponds to data that is fully consistent with the hypothesis posed by a (co)sheaf.

The set of global sections of a presheaf on a topological space may be quite different from $\mathcal{S}(X)$. It is for this reason that when studying presheaves over topological spaces, an additional *gluing axiom* is included to remove this distinction. A similar axiom applies for cosheaves.

Definition 10. Let \mathcal{P} be a presheaf on the topological space (X, \mathcal{T}) . We call \mathcal{P} a *sheaf on* (X, \mathcal{T}) if for every open set $U \in \mathcal{T}$ and every collection of open sets $\mathcal{U} \subseteq \mathcal{T}$ with $U = \bigcup \mathcal{U}$, then $\mathcal{P}(U)$ is isomorphic to the space of sections over the set of elements \mathcal{U} .

Dually, a precosheaf \mathfrak{C} is a *cosheaf on* (X, \mathcal{T}) if for every open set $U \in \mathcal{T}$ and every collection of open sets $\mathcal{U} \subseteq \mathcal{T}$ with $U = \bigcup \mathcal{U}$, then $\mathfrak{C}(U)$ is isomorphic to the space of cosections over the set of elements \mathcal{U} .

For the time being, we will focus on sheaves. Cosheaves will reappear in Section 5.

Given that most assignments are not sections, it is useful to be able to measure how far away an assignment is from being a section. When we have pseudometrics on the stalks, one useful estimate of that distance is the consistency radius.

Definition 11. If \mathcal{S} is a presheaf of pseudometric spaces on a topological space (X, \mathcal{T}) and a is a global assignment, the *p-norm consistency radius* of a is the quantity

$$c_{\mathcal{S}}(a) := \left(\sum_{U \in \mathcal{T}} \sum_{V \in \mathcal{T}: V \subseteq U} (d_V(a(V), \mathcal{S}(V \subseteq U)a(U)))^p \right)^{1/p}, \quad (10)$$

where $p \geq 1$.

In all of our examples, $p = 2$ is used. A subtle point is that the relative weight of each of the different terms in Equation (10) is implicitly carried by the pseudometrics d_V . For instance, if $x, y \in \mathbb{R}^n$, a weighted form of the Euclidean pseudometric could be written

$$d_V(x, y) = \alpha_V \left(\sum_{k=1}^n |x_k - y_k|^p \right)^{1/p},$$

where $\alpha_V > 0$ is a constant that weighs the importance of the value in the stalk on V in the overall consistency radius. In some cases, for instance if different units of measure are involved, the correct choice of α_V is clear. In others, the α_V is a nuisance parameter that needs to be explored by the modeler.

Corollary 4. If s is a global section of a presheaf \mathcal{S} of pseudometric spaces, then $c_{\mathcal{S}}(s) = 0$.

Consistency radius is stable under perturbations, which means that it can be reliably estimated.

Theorem 5. [35, Thm. 1] *Consistency radius is a continuous real-valued function of the assignment.*

We will often need to consider local assignments as well. A natural definition is to define the consistency radius of a local assignment to be the consistency radius of the “best” extension of the local assignment to a global one.

Definition 12. [35, Def. 16] If \mathcal{S} is a presheaf of pseudometric spaces on a topological space (X, \mathcal{T}) and a is an assignment supported on $\mathcal{U} \subseteq \mathcal{T}$, then its consistency radius is

$$c_{\mathcal{S}}(a; \mathcal{U}) := \min \left\{ c_{\mathcal{S}}(b) : b \in \prod_{U \in \mathcal{T}} \mathcal{S}(U) \text{ such that } b(U) = a(U) \text{ if } U \in \mathcal{U} \right\}.$$

We will use the phrase *minimizing the consistency radius of a* as a shorthand for finding the global assignment

$$b_* := \operatorname{argmin} \left\{ c_{\mathcal{S}}(b) : b \in \prod_{U \in \mathcal{T}} \mathcal{S}(U) \text{ such that } b(U) = a(U) \text{ if } U \in \mathcal{U} \right\}.$$

As the rest of this chapter shows, minimizing the consistency radius of a given local assignment is the primary tool for sheaf-based inference.

3.3 The netlist sheaf

The key result of this section is that inference for a DSEM corresponds to consistency radius minimization. In general, it is enabled by Definition 2 that translates a DSEM into a netlist, and Definition 13 that translates a netlist into a sheaf, in such a way that solutions correspond to global sections (Theorem 6).

In order to motivate the construction, and to explain some of its subtleties, we delay the formal construction (Definition 13) until after we have discussed two examples. The first example represents a classic linear regression problem first as a SEM (which is not dynamical), then as a netlist, and finally as a sheaf. This progression is summarized in Figure 3.

Before delving into the details, let us consider the meaning of the arrows shown in Figure 3. The arrows in each of the frames of Figure 3 mean different things. In the SEM the arrows have a causal interpretation: the value of x determines that of y . This interpretation carries over into the netlist, where ports are either inputs or outputs.

In the sheaf diagram the arrows are functions between the stalks. Since the stalks represent the set of possible values for each variable, the functions represented by the arrows will be used to extract data stored on the ports and place them on the nets regardless of whether they are inputs or outputs. There is no intuitive issue with the outputs. An output variable is determined by the

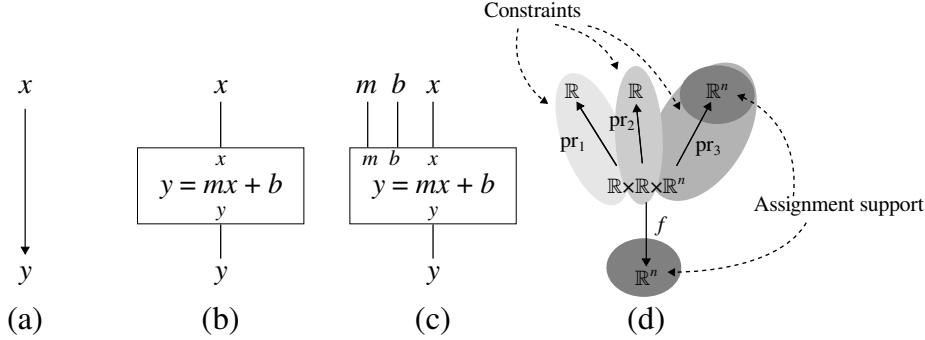


Figure 3: A linear regression problem as (a) a SEM, (b) a netlist with hardcoded coefficients, (c) a netlist with coefficients exposed as inputs, and (d) a sheaf. To solve the linear regression problem, the partial assignment supported on the darkest shaded region is supplied by the observations, and then the assignment is extended to the remaining stalks. Finally, the copies of m , b , and x that should be constrained so that they are identical are shown by the three lighter shadings.

data within the part it is attached to. However, for an input, the only thing the arrow does is extract the corresponding port’s value unmodified. This seems paradoxical! The point is that when two parts are connected to each other on a net, they both have a claim on what the value of the variable should be. If the values correspond to a global section of the sheaf, this is the assertion that both claims on that variable agree, namely the variable produced by the output of one port is the same as the variable that reaches the input port attached to the same net.

Beginning the example in earnest, suppose that $(x_1, y_1), \dots, (x_n, y_n)$ are n points in the plane \mathbb{R}^2 . As a modeling choice, we suppose that the x values can be used to predict the y values, or alternatively that x is an explanatory variable and y is a response variable. If we assert that the model should be linear, we are assuming

$$y \approx b + mx,$$

where b and m are parameters to be found. To express this modeling assumption graphically, we write an arrow $x \rightarrow y$, yielding the SEM graph in Figure 3(a).

The netlist for the problem represents the same information as in the SEM. As shown in Figure 3(b), the netlist consists of two variables (x and y), and one part (the linear equation that predicts y from x).

The prediction process depends on the two parameters b and m , which can also be considered as inputs. This change results in a netlist with four variables (x , y , b , and m) and the same part as before, shown in Figure 3(c).

The sheaf representation of the same system is shown in Figure 3(d). It is considerably more explicit about variable type information. The stalk over m and b is \mathbb{R} , since each of these parameters takes a real value. On the other hand,

the stalk over x and y is \mathbb{R}^n , since they are each a sequence of n real values. The stalk over the single part is the set of its inputs, namely $\mathbb{R} \times \mathbb{R} \times \mathbb{R}^n$, corresponding to m , b , and x , respectively. The restriction maps from the part to the inputs are all projection maps, which select the different inputs. Explicitly,

$$\text{pr}_1(m, b, (x_1, \dots, x_n)) = m,$$

$$\text{pr}_2(m, b, (x_1, \dots, x_n)) = b,$$

and

$$\text{pr}_3(m, b, (x_1, \dots, x_n)) = (x_1, \dots, x_n).$$

The remaining restriction map f shown in Figure 3(d) performs the prediction process, and is given by

$$(y_1, \dots, y_n) = f(m, b, (x_1, \dots, x_n)) = (mx_1 + b, \dots, mx_n + b). \quad (11)$$

The function f applies the common coefficients (b and m) to each of the input values x_k to yield the corresponding output values y_k .

The space of global assignments for the sheaf shown in Figure 3(d) is given by the product of all of the stalks. This means there are *two copies* of m , b , and x in the space of global assignments, one for the value of the variable and one as a component of the part. A typical global assignment a is of the form

$$a := \left(m, b, (x_1, \dots, x_n), (y_1, \dots, y_n), \left(\tilde{m}, \tilde{b}, (\tilde{x}_1, \dots, \tilde{x}_n) \right) \right), \quad (12)$$

where we have listed the four variables first followed by the part. The consistency radius of this assignment is

$$c(a) = \left(|\tilde{m} - m|^p + |\tilde{b} - b|^p + \sum_{k=1}^n |\tilde{x}_k - x_k|^p + \sum_{k=1}^n |b + m\tilde{x}_k - y_k|^p \right)^{1/p} \quad (13)$$

for a given p . In what follows, we will take $p = 2$, so as to agree with classical linear regression.

The problem of classical linear regression seeks real numbers m and b minimizing the last term in Equation (13). Therefore, minimizing consistency radius subject to the constraint that each pair of copies of m , b , and x is equal, and that only m and b are allowed to vary will recover linear regression from the sheaf. These copies are identified in the lighter shaded regions in Figure 3(d).

To follow the paradigm of consistency radius minimization, we specify a local assignment to the variables x and y , and then extend the assignment to a global one. The support of the local assignment is expressed by the darkest shaded region in Figure 3(d). Notice that the nets have no higher elements in the partial order shown in Figure 3, so the support of this assignment is $\mathcal{U} = \{\{x\}, \{y\}\}$. Explicitly, we start with a non-global assignment supported on \mathcal{U} ,

$$(-, -, (x_1, \dots, x_n), (y_1, \dots, y_n), -), \quad (14)$$

where the dashes indicate stalks outside the support of the assignment. If we seek a global assignment g^* such that

$$g^* = \operatorname{argmin} \{c(b) : g(U) = a(U) \text{ for } U \in \mathcal{U}\},$$

this means that we wish to find the entries in the assignment in Equation (12) that are marked with the dashes in Equation (14), namely

$$\tilde{m}, \tilde{b}, m, b, \text{ and } (\widetilde{x_1}, \dots, \widetilde{x_n}).$$

Minimizing consistency radius is therefore given by the problem

$$\operatorname{argmin}_{\tilde{m}, \tilde{b}, m, b, (x_1, \dots, x_n)} \left(|\tilde{m} - m|^2 + |\tilde{b} - b|^2 + \sum_{k=1}^n |\widetilde{x_k} - x_k|^2 + \sum_{k=1}^n |b + mx_k - y_k|^2 \right)^{1/2}.$$

But since both \tilde{m} and m , and \tilde{b} and b are being minimized, the consistency radius reduces to

$$\operatorname{argmin}_{m, b, (x_1, \dots, x_n)} \left(\sum_{k=1}^n |\widetilde{x_k} - x_k|^2 + \sum_{k=1}^n |b + mx_k - y_k|^2 \right)^{1/2}.$$

This permits the values of the variables x and y to differ from their copies, subject to a penalty. Instead of least squares regression, this problem is what is usually called *total* least squares; see Figure 4. After minimization, the differences between each of the copies

$$|\widetilde{x_k} - x_k|$$

expresses the uncertainty of their values if the model is to be taken as a given.

To obtain classical least squares regression, we must constrain $\widetilde{x_k} = x_k$ for all k . The global assignment we seek is of the form

$$g^* = (m, b, (x_1, \dots, x_n), (y_1, \dots, y_n), (m, b, (x_1, \dots, x_n))),$$

so that the consistency radius minimization problem subject to this constraint becomes

$$\operatorname{argmin}_{m, b} \left(\sum_{k=1}^n |b + mx_k - y_k|^2 \right)^{1/2}.$$

Consistency radius minimization unifies several different inference tasks in Figure 3, depending on the support of the initial assignment:

Forward prediction Choose an assignment supported on x , b , and m , of the form

$$(m, b, (x_1, \dots, x_n), -, -).$$

Consistency radius minimization will infer the values for y . Because the above assignment extends to a global section, namely,

$$(m, b, (x_1, \dots, x_n), (b + mx_1, \dots, b + mx_n), (m, b, (x_1, \dots, x_n))),$$

consistency radius minimization does not require constraints in this case.

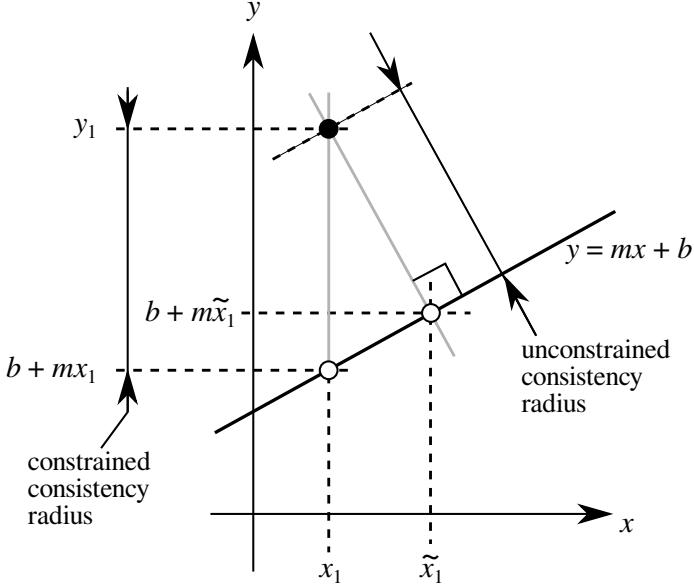


Figure 4: Geometric meanings of the terms contributing to consistency radius in Equation 13.

Backward prediction Choose an assignment supported on y and b , and m , of the form

$$(m, b, -, (y_1, \dots, y_n), -).$$

Consistency radius minimization will infer the values for x . If $m \neq 0$, this always results in a global section,

$$(m, b, ((y_1 - b)/m, \dots, (y_n - b)/m), (y_1, \dots, y_n), (m, b, ((y_1 - b)/m, \dots, (y_n - b)/m)),$$

so consistency radius minimization does not require constraints. If $m = 0$ then the minimizers of consistency radius all have the same consistency radius, and are assignments of the form

$$(0, b, (x_1, \dots, x_n), (y_1, \dots, y_n), (0, b, (x_1, \dots, x_n))).$$

Noting that the two copies of the x variable are always identical, applying constraints does not change the result.

Regression (model fitting) (Details above, included for completeness here.)

Choose an assignment supported on x and y , of the form

$$(-, -, (x_1, \dots, x_n), (y_1, \dots, y_n), -).$$

Consistency radius minimization will infer the values for b and m . As noted above, without constraints consistency radius minimization solves total least squares, while constraints are necessary to recover classical regression.

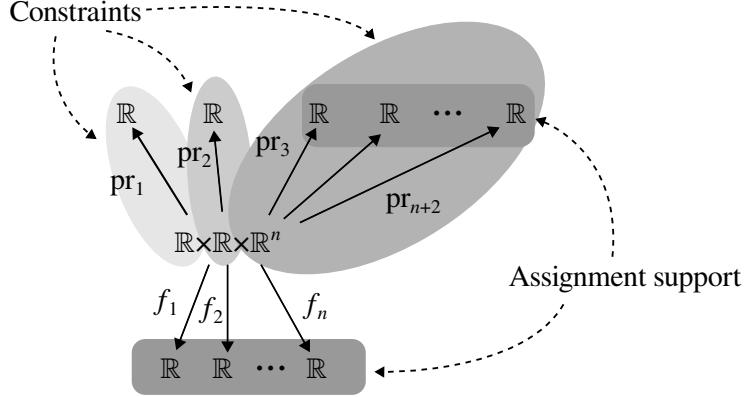


Figure 5: Modification to the sheaf in Figure 3(d) to allow for missing data.

Hybrid versions of the above problems can also be addressed.

Assignments are populated stalk-wise, so the sheaf in Figure 3(d) explicitly requires that we have access to all of the n data points, since the stalks for x and y are each \mathbb{R}^n . If there is missing data, a different sheaf construction is possible, in which each separate component of x and y is given its own stalk. Figure 5 shows the resulting construction.

The f_k restriction maps appearing in Figure 5 are the individual components of the f restriction map in Figure 3(d), namely given Equation (11),

$$y_k = f_k(m, b, (x_1, \dots, x_n)) = mx_k + b.$$

The set of global assignments for the sheaf in Figure 3(d) is the same as that for the sheaf in Figure 5, but its components are delineated differently. A typical global assignment a for the sheaf in Figure 5 is given by

$$a := \left(m, b, x_1, \dots, x_n, y_1, \dots, y_n, (\tilde{m}, \tilde{b}, \tilde{x}_1, \dots, \tilde{x}_n) \right),$$

where the main difference between the above and Equation (12) is in the placement of parentheses. The consistency radius for a global assignment in both sheaves is given by exactly the same formula. As in the previous sheaf, we can express the linear regression problem as a consistency radius minimization problem, in which a local assignment supported on the x_k and y_k variables (shown by the darkest shaded regions in Figure 5) is extended to a global assignment, subject to the constraint that each of the copies of the duplicated variables are identical (shown by the three lighter shaded regions in Figure 5). But now, if there is a missing x_k or y_k value, this can simply be excluded from the support of the initial assignment, leaving the specification of the task as a consistency radius minimization unchanged.

Feedback connections are easily represented in all of the frameworks under consideration. Moreover, depending on the set of variables that are permissible, the resulting sheaf will or will not have global sections (Definition 9).

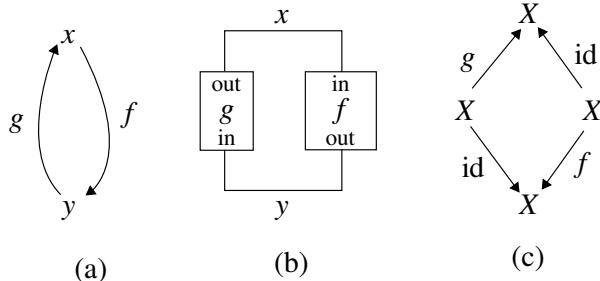


Figure 6: Feedback connections can be handled: (a) a (D)SEM model with feedback, (b) its netlist, (c) its sheaf representation.

Consider the setting shown in Figure 6:

$X = \mathbb{R}$, $f(x) = x$, $g(x) = x$ (Linear SEM) global sections occur whenever the two variables have the same value.

$X = \mathbb{R}$, $f(x) = -x$, $g(x) = x$ (Linear SEM) the only global section is for both variables to be 0.

$X = \mathbb{R}$, $f(x) = 1 - x$, $g(x) = x$ (Affine, nonlinear SEM) The only global section is for both variables to take the value 1/2.

$X = \mathbb{Z}$, $f(x) = 1 - x$, $g(x) = x$ (Discrete values) No global sections exist.

Feedback will play an important role in defining a sheaf to model autoregressive timeseries in Section 3.4.

With the preliminary intuition established by the previous two examples, we are now in a position to discuss the general translation algorithm.

Definition 13. If we have a netlist N , we build the *netlist sheaf* on the Alexandrov topology of the preorder of its netlist graph of N . The stalk on each net is the set of variables for that net. The stalk on each part is the product of its input ports. The restriction from a part to a net along an input port is the projection function for the corresponding variable set. The restriction from a part to a net along an output port is the function that computes the output variable from the set of input variables.

It is often useful to have individual observations on their own stalks, like we did in Figure 5. The following modification to Definition 13 allows for missing data in general.

Definition 14. Starting with a netlist sheaf as defined in Definition 13, add an additional element to the preorder of the netlist graph for each observation of each variable. These elements are located above their respective variables in the preorder. The restriction map from each variable to each observation is the projection that selects the corresponding observation from its parent timeseries.

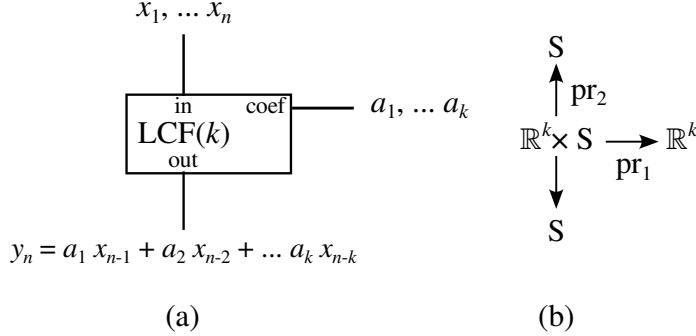


Figure 7: A linear causal filter LCF(k) with a sliding window size k as (a) netlist wiring hypergraph and (b) netlist sheaf.

Theorem 6. *Variable values on the netlist correspond bijectively to DSEM solutions and to global sections.*

Proof. (see also [34][Prop. 6]) There is a direct correspondence between the values of variables on the nets and the nodes in the DSEM. If these are values correspond to a solution, then they directly imply consistency with the restriction maps. \square

Moreover, according to [35, Thm. 1] there is stability in consistency radius when we perturb away from a consistent set of variables. This is classical in the case of the linear regression example, because the linear regression coefficients m and b are stable with respect to perturbations in the data variables x and y .

3.4 Sheaves modeling autoregressive timeseries

Autoregressive timeseries are sequences \dots, x_0, x_1, \dots that obey an equation of the form

$$x_n = a_1 x_{n-1} + a_2 x_{n-2} + \dots + a_k x_{n-k},$$

for some fixed a_1, \dots, a_k . We say that such a sequence is AR(k) autoregressive. Autoregressive timeseries can be modeled using the graphical framework being developed in this chapter by the use of feedback connections.

It is easiest to see how the construction of autoregressive timeseries works by starting with a one-step delayed Linear Causal Filter with sliding window size k (which we write as “LCF(k)” for short in diagrams). Like the linear regression example from the previous section, a variable x is considered an explanatory variable that predicts the values of a response variable y . This prediction is given by

$$y_n = a_1 x_{n-1} + a_2 x_{n-2} + \dots + a_k x_{n-k}$$

where the a_1, \dots, a_k are constants.

We can realize this equation as a netlist with an input for x , an input for a , and an output for y shown in Figure 7(a). Using Definition 13, we obtain the

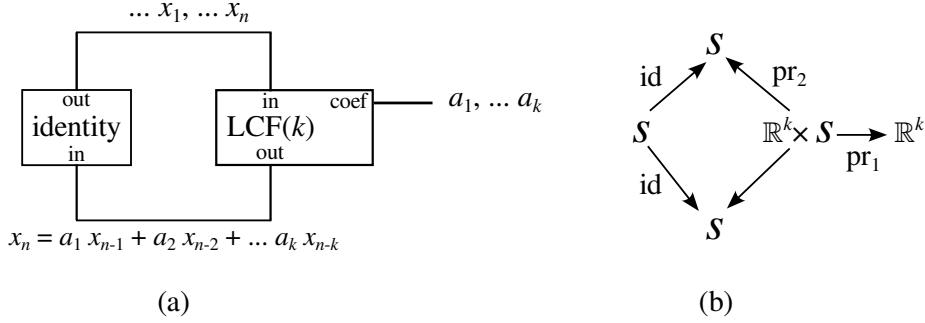


Figure 8: A (a) netlist and (b) a sheaf that encodes an autoregressive timeseries.

netlist sheaf shown in Figure 7(b), where S is the set of infinite sequences of real numbers.

To handle autoregressive timeseries, we merely need to consider the pair of equations

$$\begin{cases} y_n = a_1 x_{n-1} + a_2 x_{n-2} + \dots + a_k x_{n-k}, \\ x_n = y_n. \end{cases}$$

This is implemented as a netlist with two parts and a feedback connection, as shown in Figure 8(a), where again S is the set of infinite sequences of real numbers. The linear causal filter part is the same as before, but the identity part implements the second equation above. Error terms are not explicitly mentioned, because they are accounted for in the consistency radius calculation (Equation (10)).

The associated netlist sheaf is shown in Figure 8(b). Again, consistency radius measures how well the data x fit the model given with coefficients a . Following a theme already present in the linear regression example, there is duplication of data in the sheaf model. Indeed, the values of x are effectively duplicated in *four* places: the x and $y = x$ variables, and in the two parts. Once again, if we consider an assignment supported on the two variables (with the same values on each!), minimizing consistency radius will infer the values of the a coefficients. Once again, if we run an *unconstrained* optimization, this assumes that some uncertainty is permitted in the values of x .

When the timeseries are finite in length, the equation defining an AR(k) sequence cannot represent any of the first k time steps. Therefore, instead of the identity part in Figure 8, the sheaf for an AR(k) sequence of length n must crop off the first k components of the vector in the stalk, resulting in a sequence of length $n - k$. The resulting construction is shown in Figure 9, where we note that a slight abuse of definition occurs in Figure 9(a) because the two outputs are connected to each other. While this means that the netlist is not valid as such, the sheaf constructed in Figure 9(b) correctly represents an autoregressive sequence. Global sections of the sheaf in Figure 9(b) are precisely the AR(k) sequences of length n .

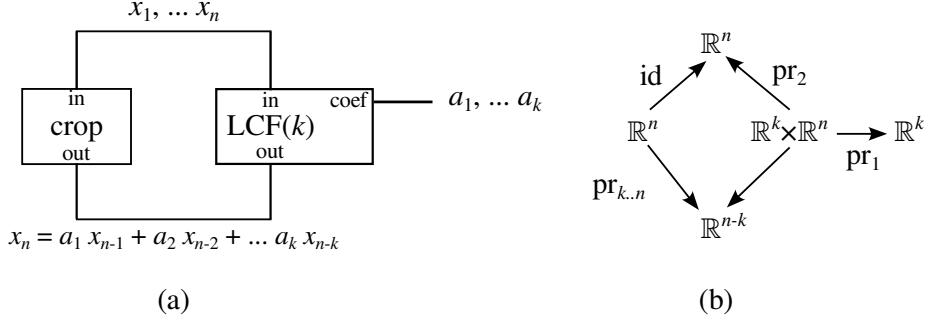


Figure 9: A (a) netlist and (b) a sheaf that encodes an autoregressive timeseries.

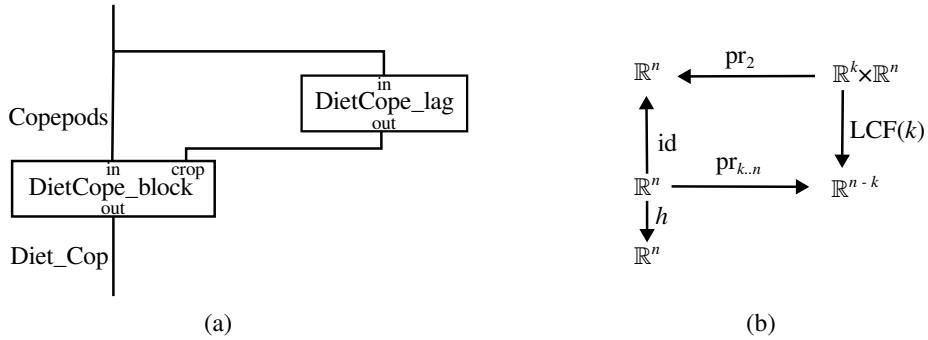


Figure 10: Modification to Figure 1(d) to support autoregressive timeseries, shown for the *Copepods* variable: (a) netlist wiring hypergraph, (b) sheaf diagram. This modification is performed for each variable in Figure 1 resulting in Figure 13.

Autoregressive sequences can be modeled in the sheaf shown in Figure 1(d), our ecological example. All that is needed is a modification to each variable in the netlist to ensure that each variable is an autoregressive sequence. Specifically, each of the input variables for each of the parts in the netlist shown in Figure 1(b) must be duplicated to represent a lagged copy of the variable, and there must be a new part added for each variable to perform the autoregression itself. As in Figure 9, each original variable gets wired to the input of the corresponding LCF part. The duplicated (lagged) input on each preexisting part is cropped to be only the most recent samples (since the timeseries is finite), and then that is what is attached to the output port of the LCF part. The transformation that is required for the *Copepods* variable is shown in Figure 10.

4 Sheaf encoding of the Bering Sea

We now return to the ecological DSEM example introduced in Section 2.2, and refer the reader to Figure 1. The reader is directed to [36] for the software that generates the sheaf results presented in this section.

The DSEM is shown in Figure 1(a), its corresponding netlist wiring hypergraph is shown in Figure 1(b), its netlist graph is shown in Figure 1(c), and its netlist sheaf is shown in Figure 1(d).

The netlist sheaf in Figure 1(d) does not express the path coefficients as variables, as they are instead “hard coded” within each part. Nevertheless, if the path coefficients are known (for instance, they can be taken from [46]), then the sheaf model can be used to predict the values of each of the variables, starting from *SeaIce* and *Spawners*. If we apply the modification to the sheaf to require AR(1) timeseries so that missing data values are interpolated, and use the path coefficients stated in [46] (see Table 3), the resulting timeseries are shown in Figure 11.

The DSEM was constrained to fit the measurements exactly, whereas the sheaf had no such constraints applied. Where the sheaf differs from the measurements, the extent of that difference is a measure of the uncertainty in the value of the variable at the given time. This uncertainty is composed of both the measurement and exogenous errors; the sheaf model does not distinguish between the types of error. Moreover, where there are no measurements available (especially for the earlier measurements), the DSEM reports the expected mean. The sheaf predictions are typically close to these mean values. Nevertheless, there is close agreement throughout. This is not unexpected, because both the sheaf and the DSEM approach are approximations to the same DSEM solution. There are some differences on the behavior of the earlier inferred data, because many of the observations are missing there. In these regions, the sheaf tends to yield somewhat less variable predictions than the DSEM (except in the case of the *Krill* variable).

As noted earlier, we will compute consistency radius using the Euclidean $p = 2$ norm. Lacking other information, we chose to weight the terms in Equation (10) equally. The consistency radius of the assignment after minimization is

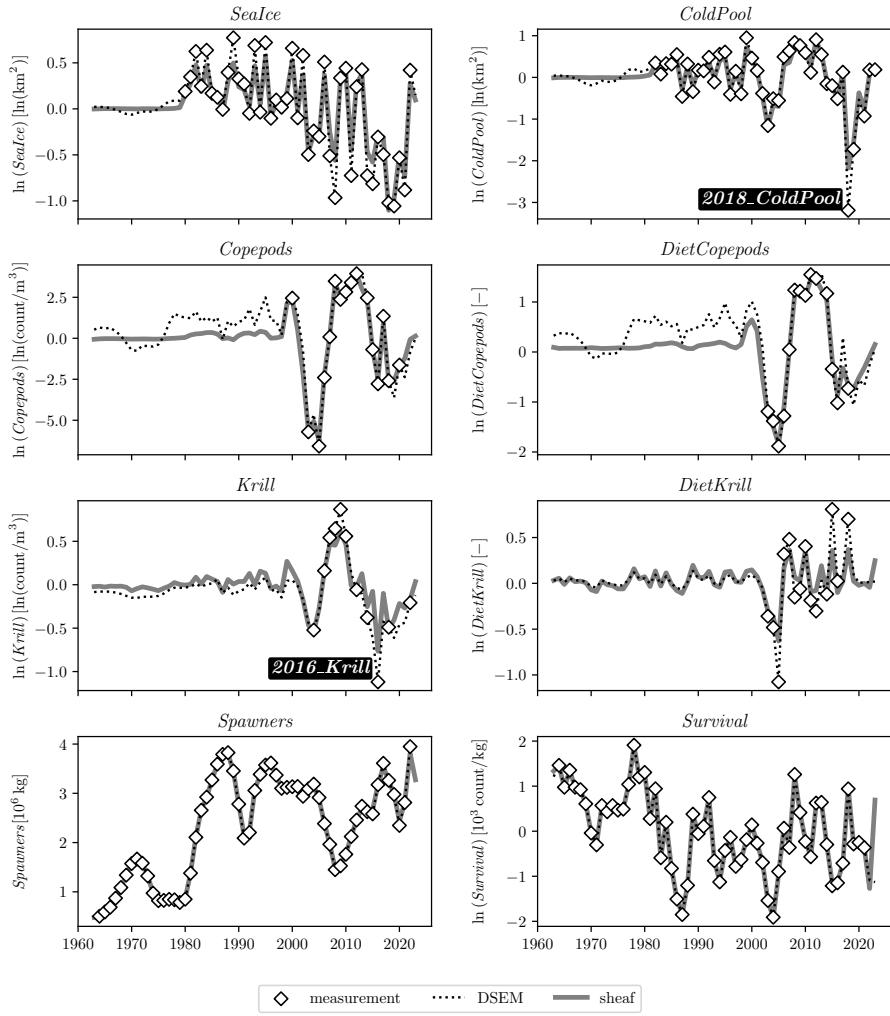


Figure 11: Comparison between the DSEM output and the sheaf with hard-coded path coefficients shown in Figure 1(d) and AR(2) timeseries. The DSEM was constrained to fit the measurements exactly, whereas the sheaf had no such constraints applied.

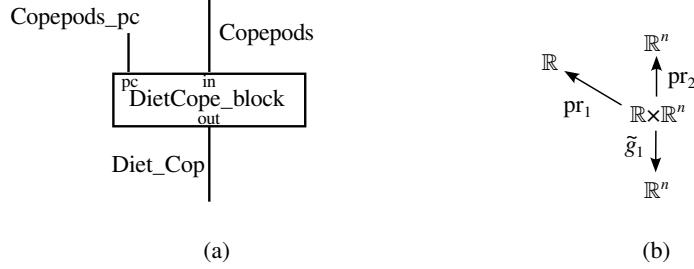


Figure 12: Modification to the netlist to include path coefficients and constants as an input.

11.9. Since this is not zero, this means that the fit between the data and the model is not perfect. While the DSEM fits the data for maximum likelihood, the sheaf fits for minimum inconsistency. This difference in optimization task results in the observed differences between the sheaf and the DSEM.

Taking a cue from Figure 3 in the previous section, we can break out path coefficients as separate variables so that they can be adjusted or estimated. Figure 12 shows how one of the parts in the netlist shown in Figure 1(b) can be modified so that its path coefficients are inputs. To handle missing data, we apply Definition 14 to the netlist sheaf, which results in Figure 13.

Using the sheaf shown in Figure 13, we can infer the path coefficients and autoregressive coefficients by consistency radius minimization. Specifically, we construct an assignment supported only on the values of the variables that correspond to observations present in the data. Then, when we minimize consistency radius, the values of the path coefficients, autoregressive coefficients, *and any missing observations* will be inferred. The resulting global assignment has a complete timeseries—no missing observations—for each variable as well as path coefficients and autoregressive coefficients. Because the approach explained in Section 2.1 uses a different strategy for approximating solutions to the problem posed by the DSEM, the inferred path coefficients and missing observations will be somewhat different from those inferred by the sheaf.

There are some differences between the sheaf and the measurement data. The contributions to consistency radius are not uniformly distributed over the sheaf. Some of the inconsistency is due to disagreements between the measurements and the DSEM graph model, and some of the inconsistency is due to the fact that the measurements are not AR(1) timeseries. This is visually apparent in Figure 13, where it is shown that the two largest contributors to the consistency radius are

1. the autoregression cell for *Copepods* (labeled *Copepods_lagvar*), and
2. the year 2018 observations of *ColdPool* (labeled *2018_ColdPool*).

The second of these is easier to interpret. We should suspect that the 2018 observation of *ColdPool* is an outlier (in the L^2 sense) from what was expected

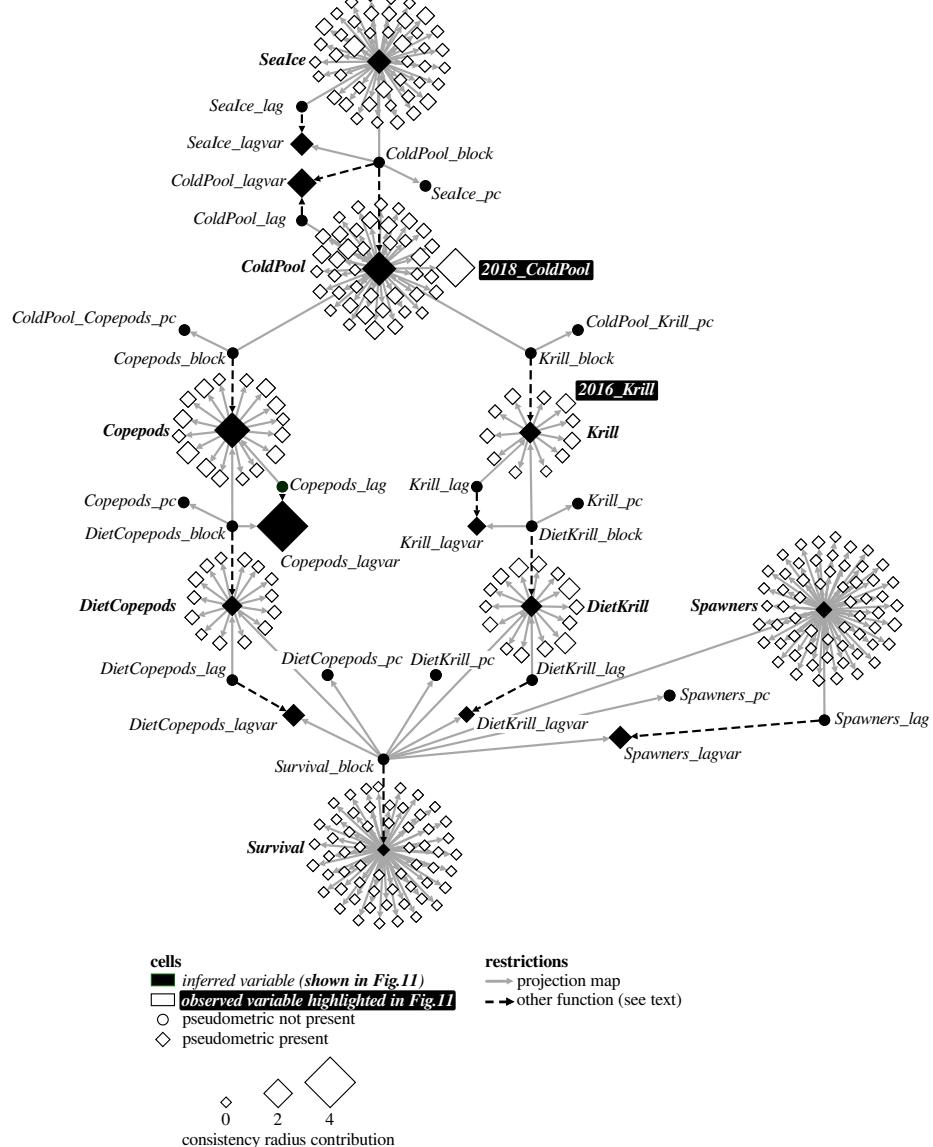


Figure 13: The full sheaf for the DSEM described in Section 2.2. Its structure reflects the hexagonal backbone shown in the diagrams in Fig. 1. The black cells represent inferred variables, with the variable names shown in italics. Variable names that are also bold correspond to variables plotted in Fig. 11. White cells represent variables that are observed. All observed variables except for two are not labeled for clarity. The two that are labeled have their names in white italics with black backgrounds. These variables exhibit relatively large contributions to the consistency radius and are highlighted in Fig. 11.

Source	Target	DSEM [46] AR(1)	Sheaf			
			none	AR(1)	AR(2)	AR(10)
<i>SeaIce</i>	<i>ColdPool</i>	0.6	1.68	1.81	1.78	1.74
<i>ColdPool</i>	<i>Copepods</i>	1.79	4.45	4.38	4.47	4.17
<i>ColdPool</i>	<i>Krill</i>	0.18	0.44	0.38	0.41	0.39
<i>Copepods</i>	<i>DietCopepods</i>	0.29	0.32	0.35	0.36	0.34
<i>Krill</i>	<i>DietKrill</i>	0.06	0.52	0.70	0.65	0.56
<i>DietCopepods</i>	<i>Survival</i>	0.15	-0.50	-0.12	-0.05	-0.32
<i>DietKrill</i>	<i>Survival</i>	0.13	7.56	5.29	7.19	5.63
<i>Spawners</i>	<i>Survival</i>	-0.59	-0.82	-0.65	-0.55	-0.74
Consistency radius		11.9	6.60	9.48	9.03	7.93
Runtime (s)		2	2848	2637	2679	2907

Table 3: Comparison between path coefficients estimated from the DSEM and the sheaf

from the model, and that these differences may have propagated into other parts of the model. This probably explains why the 2018 observations of *Krill* and *DietKrill* are substantially different from the sheaf predictions in Figure 11.

We should interpret the largest contributor to consistency radius as suggesting that the *Copepods* variable is not well represented by an AR(1) timeseries. Notice that the *Copepods* observations contribute equally to consistency radius, since the small white diamonds encircling the *Copepods* variable are about the same size. This suggests that it is simply that the assumption of Copepods being represented by an AR(1) timeseries is faulty, rather than any particularly bad observation.

Table 3 shows the path coefficients inferred by the DSEM (using maximum likelihood as explained in Section 2.2) and by the sheaf (using minimum consistency radius). Table 4 shows the autoregressive coefficients estimated by the sheaf for the AR(1) and AR(2) cases. (The AR(10) case is not shown for space considerations.) The DSEM-derived path coefficients were obtained using the assumption of AR(1) timeseries. Several different sheaves were constructed with autoregressive sequences of different window sizes. As a consequence of the construction of consistency radius, minimizing consistency radius infers the following information: (1) missing observations in any variable, (2) all path coefficients, and (3) autoregressive coefficients for each variable.

There is broad agreement about the values of the path coefficients between the sheaves with different autoregressive window sizes, and some agreement between the DSEM and the sheaves. Since the DSEM does not natively imply a consistency radius, the consistency radius shown for the DSEM is that for the sheaf using AR(1) timeseries and the hard-coded path coefficients as shown. Because the consistency radius minimization process on that sheaf cannot adjust the path coefficients—it can only adjust the missing observation values and the autoregressive coefficients—the consistency radius is notably higher in this case.

Some caution in comparing consistency radius across the columns of Table

Variable	AR(1) lag 1	AR(2)	
		lag 1	lag 2
<i>ColdPool</i>	0.582	0.480	0.202
<i>SeaIce</i>	0.361	0.287	0.190
<i>Copepods</i>	0.828	1.16	-0.442
<i>Krill</i>	0.692	0.308	0.411
<i>Spawners</i>	1.01	1.78	-0.768
<i>DietCopepods</i>	0.886	1.68	-0.924
<i>DietKrill</i>	0.060	0.0596	0.0445

Table 4: Autoregressive coefficients estimated by the sheaf for AR(1) and AR(2) models.

3 is needed. The number of terms in the consistency radius is the same for each of the sheaves in all but the non-autoregressive case (the fourth column from the left). This is because the autoregressive coefficients and timeseries are bundled as shown in Figure 9. Naturally enough, the non-autoregressive sheaf’s consistency radius contains no terms pertaining to the autoregressive coefficients, and so is expected to be smaller than the others. The sheaf column listed as “none” means that no autoregressive timeseries assumptions were applied. Because with no autoregressive assumptions in play, the resulting sheaf diagram is smaller, consequently the consistency radius is smaller. Interestingly, the consistency radius is smallest for the AR(10) case, which suggests that more flexibility in the autoregressive coefficients leads to somewhat better prediction accuracy in the measurement data.

Runtimes shown in Table 3 are representative when run on an Intel Core Ultra 7 155U at 1.4 GHz with 32 GB RAM. The process was not memory limited and consumes less than 500 MB RAM. The sheaf runs roughly 1500 times slower than the DSEM. This is because the DSEM solves a sparse linear problem, while the sheaf methodology supports fully nonlinear, non-convex problems. The sheaf software does not attempt to detect whether the problem is linear, so the consistency radius minimization is always performed as a nonlinear, non-convex optimization problem.

5 The topology of subsystems

Classically, dynamical systems have been studied using the structure of *invariant sets*. These are subsets of the space of variable values that are preserved by the action of the dynamical system. This section shows that invariant sets are one half of a duality pair. We can take two different perspectives of a multi-scale dynamical system: invariant sets (which lead to cosheaves) versus subsystems (which lead to sheaves).

We will establish that a dynamical system induces a *cosheaf of invariant sets*. The cosheaf of invariant sets breaks the global state of the system into different regimes of behavior, which are parameterized by the open sets of the

base space topology. Conversely, there is also a *sheaf of subsystems* that splits the variables into nested collections that each act independently.

We will formalize the *topology of subsystems* as a finite topological space, by using the Alexandrov topology for a specific preorder (Definition 6). Each subsystem corresponds to a preorder element, with composite subsystems “hooked together” according to the preorder. The preorder relation decomposes composite subsystems into their component pieces. Intuitively, moving “up” in the preorder yields more abstracted “high-level” systems. This is not entirely compatible with *all* system decompositions in the literature, so caution is advised! (The intuition of the presentation here is compatible with Kearney et al. [22], where the system is modeled as a graph. In Kearney et al. [22], vertices are the loci of state variables, and are “above” edges in the preorder constructed in that paper. Our presentation is also compatible with Steward [43], after transitive closure.)

5.1 Dynamical systems

Definition 15. A *dynamical system* is a continuous bijection $f : S \rightarrow S$. The set S in this case is called the set of *states* of the dynamical system.

It is a classical fact that for a fixed timestep, the solutions to a smooth first order differential equation of the form (1) induce a dynamical system [44]. As a consequence, the DSEM, netlist, and sheaf models of the previous sections represent dynamical systems.

Definition 16. For a dynamical system $f : S \rightarrow S$, a subset $V \subseteq S$ is called an *invariant set* if

$$f(V) \subseteq V.$$

Corollary 7. If V is an invariant set of $f : S \rightarrow S$, then f restricts to a function $f : V \rightarrow V$.

Definition 17. Suppose that $A \subseteq B$. The *inclusion* is the function $i : A \rightarrow B$ is a function such that $i(x) = x$ for every $x \in A$. Notice that $(i|A) \circ i = i$.

Dually, a *projection* is a function $p : B \rightarrow A$ such that $p \circ p = p$ and $p|A = \text{id}_A$.

Proposition 8. Suppose that U and V are two invariant sets for a dynamical system $f : S \rightarrow S$ and that $U \subseteq V$. Then the following diagram

$$\begin{array}{ccc} U & \xrightarrow{f} & U \\ i \downarrow & & \downarrow i' \\ V & \xrightarrow{f} & V \end{array}$$

commutes, where i and i' are appropriate inclusion maps, which is to say that

$$f \circ i = i' \circ f.$$

Proof. Suppose that $x \in U$. Since U is an invariant set, $f(U) \subseteq U$. However, since $U \subseteq V$, $x \in V$. Therefore, $f(x) \subseteq V$ because V is also an invariant set. \square

Definition 18. The category **Dyn** of dynamical systems has as its objects dynamical systems. Each morphism of **Dyn** is a commutative diagram of the form

$$\begin{array}{ccc} S_1 & \xrightarrow{f_1} & S_1 \\ g \downarrow & & \downarrow g \\ S_2 & \xrightarrow{f_2} & S_2 \end{array}$$

Composition of morphisms is given by composing the g functions.

Proposition 9. *Isomorphisms in **Dyn** are conjugacy classes of dynamical systems.*

5.2 The cosheaf endomorphism of invariant sets

The state space of a dynamical system can be decomposed as the (non-disjoint) union of all its invariant sets. This collection of invariant sets of a dynamical system is also partially ordered by subset inclusion, which means that the *collection* of invariant sets can be given an Alexandrov topology. A cosheaf can be defined to capture the relationship between an invariant set and the invariant sets that contain it. To this end, the cosheaf identifies duplicate points within these invariant sets with each other.

We begin by observing that the invariance of a collection of subsets with respect to a dynamical system is not necessary to define a cosheaf; it can be constructed generally.

Lemma 10. *Suppose that $\mathcal{U} \subseteq 2^X$ is an arbitrary collection of subsets of a set X . Consider the inclusion partial order on \mathcal{U} , given by $U \leq V$ whenever $U \subseteq V$. Define the following precosheaf $\mathfrak{C}_{\mathcal{U}}$ on the Alexandrov topology of the inclusion partial order (\mathcal{U}, \leq) :*

1. $\mathfrak{C}_{\mathcal{U}}(U) = U$
2. $\mathfrak{C}_{\mathcal{U}}(U \leq V) = \mathfrak{C}_{\mathcal{U}}(U \subseteq V) : U \rightarrow V$ via the inclusion map.

Then $\mathfrak{C}_{\mathcal{U}}$ is a cosheaf of sets on the Alexandrov topology of the inclusion partial order (\mathcal{U}, \leq) .

Proof. Suppose that $V \in \mathcal{U}$, and that $\mathcal{V} \subseteq \mathcal{U}$ is a collection of subsets with $V = \cup \mathcal{V}$. We need to establish that the space of global cosections on \mathcal{V} is identical to $\mathfrak{C}_{\mathcal{U}}(V) = V$. The space of global cosections on \mathcal{V} is

$$\left(\bigsqcup_{W \in \mathcal{V}} \mathfrak{C}_{\mathcal{U}}(W) \right) / \sim = \left(\bigsqcup_{W \in \mathcal{V}} W \right) / \sim = \bigcup_{W \in \mathcal{V}} W = \cup \mathcal{V} = V,$$

since the equivalence \sim identifies points that agree on overlaps. \square

The above cosheaf construction is functorial, which means that it is compatible with transformations of the underlying sets. In order to establish functoriality, we need to formalize these transformations by defining the class of morphisms for sheaves and cosheaves.

Definition 19. Suppose that \mathcal{R} is a sheaf on (X, \mathcal{T}_X) , \mathcal{S} is a sheaf on (Y, \mathcal{T}_Y) , and that $f : (X, \mathcal{T}_X) \rightarrow (Y, \mathcal{T}_Y)$ is a continuous function. A *sheaf morphism* $m : \mathcal{R} \rightarrow \mathcal{S}$ is a collection of maps $m_U : \mathcal{R}(f^{-1}(U)) \rightarrow \mathcal{S}(U)$ for each $U \in \mathcal{T}_Y$ such that the following diagram commutes whenever $U, V \in \mathcal{T}_Y$ and $U \subseteq V$,

$$\begin{array}{ccc} \mathcal{R}(f^{-1}(V)) & \xrightarrow{m_V} & \mathcal{S}(V) \\ \mathcal{R}(f^{-1}(U) \subseteq f^{-1}(V)) \downarrow & & \downarrow \mathcal{S}(U \subseteq V) \\ \mathcal{R}(f^{-1}(U)) & \xrightarrow{m_U} & \mathcal{S}(U) \end{array}$$

Dually, if \mathfrak{R} is a cosheaf on (X, \mathcal{T}_X) , and \mathfrak{S} is a cosheaf on (Y, \mathcal{T}_Y) , a *cosheaf morphism* $m : \mathfrak{R} \rightarrow \mathfrak{S}$ is a collection of maps $m_U : \mathfrak{R}(f^{-1}(U)) \rightarrow \mathfrak{S}(U)$ such that the following diagram commutes whenever $U, V \in \mathcal{T}_Y$ and $U \subseteq V$,

$$\begin{array}{ccc} \mathfrak{R}(f^{-1}(V)) & \xrightarrow{m_V} & \mathfrak{S}(V) \\ \mathfrak{R}(f^{-1}(U) \subseteq f^{-1}(V)) \uparrow & & \uparrow \mathfrak{S}(U \subseteq V) \\ \mathfrak{R}(f^{-1}(U)) & \xrightarrow{m_U} & \mathfrak{S}(U) \end{array}$$

With the definition of morphisms in hand, we can now establish that the cosheaf construction in Lemma 10 is functorial.

Lemma 11. *There is a functor $\mathbf{Top} \rightarrow \mathbf{CoShv}$ that takes a topological space (X, \mathcal{T}) to a cosheaf $\mathfrak{C}_{(X, \mathcal{T})}$ of sets on (X, \mathcal{T}) via $\mathfrak{C}_{(X, \mathcal{T})}(U) := U$ and $\mathfrak{C}_{(X, \mathcal{T})}(U \subseteq V)$ is the inclusion $U \hookrightarrow V$.*

Proof. First, we observe that Lemma 10 establishes that $\mathfrak{C}_{(X, \mathcal{T})}$ is a well-defined cosheaf on (X, \mathcal{T}) .

Suppose that $f : (X, \mathcal{T}_X) \rightarrow (Y, \mathcal{T}_Y)$ is a continuous map. This lifts to a cosheaf morphism $F : \mathfrak{C}_{(X, \mathcal{T}_X)} \rightarrow \mathfrak{C}_{(Y, \mathcal{T}_Y)}$. Suppose that $U \subseteq V$ are two open sets in Y . Then we have that $f^{-1}(U) \subseteq f^{-1}(V)$ are two open sets in X . Therefore, the following diagram commutes

$$\begin{array}{ccc} \mathfrak{C}_{(X, \mathcal{T}_X)}(f^{-1}(U)) = f^{-1}(U) & \xrightarrow{F_U := f|U} & \mathfrak{C}_{(Y, \mathcal{T}_Y)}(U) = U \\ \mathfrak{C}_{(X, \mathcal{T}_X)}(f^{-1}(U) \subseteq f^{-1}(V)) \downarrow & & \downarrow \mathfrak{C}_{(Y, \mathcal{T}_Y)}(U \subseteq V) \\ \mathfrak{C}_{(X, \mathcal{T}_X)}(f^{-1}(V)) = f^{-1}(V) & \xrightarrow{F_V := f|V} & \mathfrak{C}_{(Y, \mathcal{T}_Y)}(V) = V \end{array}$$

which establishes definitions for the component maps of F , and therefore that F is a cosheaf morphism.

Now suppose that we have two continuous maps $f : (X, \mathcal{T}_X) \rightarrow (Y, \mathcal{T}_Y)$ and $g : (Y, \mathcal{T}_Y) \rightarrow (Z, \mathcal{T}_Z)$. We must show that the corresponding composition of cosheaf morphisms $G \circ F$ is equal to the one induced by $(g \circ f)$. This follows immediately because the components maps of the cosheaf morphism $G \circ F$ are simply restrictions of the composition $(g \circ f)$. \square

Suppose that $f : S \rightarrow S$ is a dynamical system. The invariant sets of f are indeed a collection of subsets, which are partially ordered by inclusion. Therefore, Lemma 10 establishes that there is a well-defined cosheaf \mathfrak{S} of invariant sets of f .

Proposition 12. *A dynamical system $f : S \rightarrow S$ induces a morphism $m : \mathfrak{S} \rightarrow \mathfrak{S}$ on the cosheaf of invariant sets, and for which the induced map on global cosections is $m_S = f$.*

Proof. Suppose that U is an invariant set of f . Let $m_U : U \rightarrow U$ be the restriction of f to U . If $U \subseteq V$ are two invariant sets, then Proposition 8 implies that

$$\begin{array}{ccc} U & \xrightarrow{m_U=f} & U \\ i \downarrow & & \downarrow i \\ V & \xrightarrow{m_V=f} & V \end{array}$$

commutes, where i is the inclusion map. It is immediate that this is exactly the condition that the m maps are the components of a cosheaf morphism. Moreover, since S is itself an invariant set, the proof is complete. \square

5.3 Subsystem decomposition sheaf

Rather than carving up the state space into different regimes of behavior, we can instead carve it into non-interacting collections of variables. In this way, we arrive at the *subsystem sheaf* instead of the invariant set cosheaf. The global sections combine variables together into vectors, whereas global cosections paste subsets of values together.

Dualizing the condition for an invariant set yields the condition for a subsystem. Suppose that $f : S \rightarrow S$ is a bijection and that $U \subseteq S$ is an invariant set for f . If $i : U \rightarrow S$ is the inclusion map, then the diagram at left below commutes:

$$\begin{array}{ccc} S & \xrightarrow{f} & S \\ i \uparrow & & \uparrow i \\ U & \xrightarrow{f|_U} & U \end{array} \quad \begin{array}{ccc} S & \xrightarrow{f} & S \\ p \downarrow & & \downarrow p \\ B & \xrightarrow{g} & B \end{array}$$

Dually, the diagram at right above captures the situation where B is a *subsystem* of f .

Definition 20. If $f : S \rightarrow S$ is a dynamical system, a *subsystem* is a pair (g, p) consisting of a dynamical system $g : B \rightarrow B$ and a surjection $p : S \rightarrow B$ such that $p \circ f = g \circ p$. We will call p the *subsystem projection*. When p is clear from context, we will often say g is a *subsystem of f* .

We can think of the function g as a dynamical system in its own right.

The idea of a subsystem is neatly compatible with the DSEM construction. As will be shown later in Corollary 21, when the DSEM graph is acyclic, the subsystems can be “read off” directly. For the moment, a few examples will build the necessary intuition.

Example 2. Consider the DSEM with two variables A and B , given by the graph with one edge $A \rightarrow B$. The variable A is a subsystem on its own, whereas B cannot be a subsystem on its own because its value cannot be predicted from B alone. As a result, there are two nested subsystems: $\{A\}$ and $\{A \rightarrow B\}$.

To see this explicitly, suppose that the values of A are given by the timeseries $\{a_n\}$ and the values of B are given by the timeseries $\{b_n\}$, with the prediction of B from A given by the formula

$$b_{n+1} = \beta(a_n, a_{n-1}, \dots).$$

The dynamical system implied by this DSEM is represented by shifting the timeseries by one timestep. Specifically, the dynamical system is given by the function $f : A \times B \rightarrow A \times B$ given by

$$\begin{aligned} f(\dots, a_n, a_{n-1}, \dots, \dots, b_n, b_{n-1}, \dots) \\ = (\dots, a_{n+1}, a_n, \dots, \dots, \beta(a_n, a_{n-1}, \dots), \beta(a_{n-1}, a_{n-2}, \dots), \dots). \end{aligned}$$

Because of this formula, it should be clear that $\{B\}$ cannot be a subsystem because the values of the $\{b_n\}$ timeseries depend on the values of $\{a_n\}$. Under a projection that removes the $\{a_n\}$ from the domain, the values of $\{b_n\}$ cannot be determined.

The subsystem $\{A\}$ arises using the subsystem projection $p : A \times B \rightarrow A$, namely

$$p(\dots, a_n, a_{n-1}, \dots, \dots, b_n, b_{n-1}, \dots) = (\dots, a_{n+1}, a_n, \dots).$$

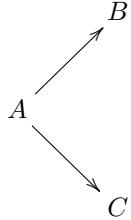
The subsystem dynamical map $g : A \rightarrow A$ is simply

$$g(\dots, a_n, a_{n-1}, \dots) = (\dots, a_{n+1}, a_n, \dots).$$

Verification that (g, p) is a subsystem is then simply a calculation,

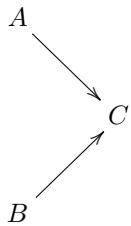
$$\begin{aligned} (p \circ f)(\dots, a_n, a_{n-1}, \dots, \dots, b_n, b_{n-1}, \dots) \\ = p(\dots, a_{n+1}, a_n, \dots, \dots, \beta(a_n, a_{n-1}, \dots), \beta(a_{n-1}, a_{n-2}, \dots), \dots) \\ = (\dots, a_{n+1}, a_n, \dots) \\ = g(\dots, a_n, a_{n-1}, \dots) \\ = (g \circ p)(\dots, a_n, a_{n-1}, \dots, \dots, b_n, b_{n-1}, \dots). \end{aligned}$$

Example 3.



Following the logic of Example 2, the subsystems are $\{A\}$, $\{A \rightarrow B\}$, $\{A \rightarrow C\}$, and the original system.

Example 4. Consider the DSEM with three variables A , B , and C given by the graph



Following the logic of Example 2, the subsystems are $\{A\}$, $\{B\}$, and the original system. Notice that $\{C\}$ cannot be a subsystem on its own because its values are determined by both A and B .

When a dynamical system is described by a DSEM with feedback, there are often fewer subsystems because the values of the variables cannot be determined in isolation.

Example 5. Consider the DSEM on variables A and B given by the graph



(See also Figure 6 for the sheaf model.) In this case, the only subsystem is the entire system, because the values of A cannot be determined without knowing B , and conversely the values of B cannot be determined without knowing A .

Linear systems are special because invariant sets and subsystems reduce to the same thing, as the next example shows.

Example 6. Let V be a finite dimensional vector space and $f : V \rightarrow V$ be a linear isomorphism. If we use the usual Euclidean norm on V , f is continuous, so it is also a dynamical system. Subsystems and invariant subspaces of f are in bijective correspondence.

To see this, suppose that $v \in V$ is an eigenvector for f , namely

$$f(v) = \lambda v$$

for some λ . Then the subspace spanned by v is an invariant set. Conversely, every invariant set of f is a linear subspace, spanned by a set of eigenvectors (possibly with complex eigenvalues).

Since V was assumed to be finite dimensional, every subspace $W \subseteq V$ also has an associated orthogonal projection $\text{pr}_W : V \rightarrow W$. If W is an invariant set for f , then $(f|W, \text{pr}_W)$ is a subsystem. To see this, suppose that $v \in V$, which can be written as the decomposition $u + w$, where $w \in W$ and $\text{pr}_W(u) = 0$. Because f is a linear isomorphism, the assumption on u means that $\text{pr}_W(f(u)) = 0$. All that remains is to verify that the definition of subsystem holds,

$$\begin{aligned} (\text{pr}_W \circ f)(v) &= \text{pr}_W(f(u + w)) \\ &= \text{pr}_W(f(u) + f(w)) \\ &= \text{pr}_W(f(u)) + f(w) \\ &= f(w) \\ &= (f|W)(w) \\ &= (f|W)(\text{pr}_W(u + w)) \\ &= (f|W \circ \text{pr}_W)(v). \end{aligned}$$

Lemma 13. *The relation “is a subsystem of” is a preorder, or in other words a reflexive, transitive relation.*

Proof. Suppose that $f : S \rightarrow S$ is a dynamical system. Reflexivity follows immediately by taking (f, id_S) as a subsystem. For transitivity, suppose that (g_2, p_2) is a subsystem of f , and that (g_1, p_1) is a subsystem of g_2 . That is, we have the commutative diagram

$$\begin{array}{ccccc} S & \xrightarrow{f} & S \\ \downarrow p_2 & & \downarrow p_2 \\ B_2 & \xrightarrow{g_2} & B_2 \\ \downarrow p_1 & & \downarrow p_1 \\ B_1 & \xrightarrow{g_1} & B_1 \end{array}$$

so that $(g_1, (p_1 \circ p_2))$ is a subsystem of f . □

Intuitively, the preorder specifies how data can flow from one subsystem to the next. If (g_1, p_1) is a subsystem of (g_2, p_2) , then each variable in (g_2, p_2) is also a variable of (g_1, p_1) . As a result, the state of g_1 can influence the state of g_2 .

Example 7. Consider the dynamical system $f : \mathbb{Z}^3 \rightarrow \mathbb{Z}^3$ given by

$$f(x, y, z) := ((1 - x), y(1 - x) + zx, z(1 - x) + yx).$$

This has a nontrivial subsystem $\text{pr}_1 : \mathbb{Z}^3 \rightarrow \mathbb{Z}$, since the map

$$g(x) := 1 - x$$

makes the following diagram commute

$$\begin{array}{ccc} \mathbb{Z}^3 & \xrightarrow{f} & \mathbb{Z}^3 \\ \text{pr}_1 \downarrow & & \downarrow \text{pr}_1 \\ \mathbb{Z} & \xrightarrow{g} & \mathbb{Z} \end{array}$$

In this case, the x variable in the subsystem acts as an input to the overall system, even though its behavior is isolated from the rest of the system.

It is not necessarily the case that subsystems are invariant sets.

Example 8. Consider the dynamical system $f : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, given by $f(x, y) := (x, y+1)$. Consider the subset $B = \{(x, 0) : x \in \mathbb{R}\}$. This set yields a subsystem, since the following diagram commutes

$$\begin{array}{ccc} \mathbb{R}^2 & \xrightarrow{f} & \mathbb{R}^2 \\ p \downarrow & & \downarrow p \\ B & \xrightarrow{\text{id}} & B \end{array}$$

where $p(x, y) = (x, 0)$, even though the set B is not an invariant set.

However, conversely, invariant sets of subsystems do determine invariant sets of their parent system.

Lemma 14. Suppose that $f : S \rightarrow S$ is a dynamical system with $g : B \rightarrow B$ is a subsystem with subsystem projection $p : S \rightarrow B$. If $V \subseteq B$ is an invariant set of g , then $p^{-1}(V)$ is an invariant set of f .

Proof. The hypotheses posit a commutative diagram of the form

$$\begin{array}{ccc} S & \xrightarrow{f} & S \\ p \downarrow & & \downarrow p \\ B & \xrightarrow{g} & B \end{array}$$

Suppose that $x \in p^{-1}(V) \subseteq S$. We have that $p(f(x)) = g(p(x))$ via the commutative diagram above. Noting that $p(x) \in V$ by construction, and that V is an invariant set of g , this means that $g(p(x)) \in V$. Thus, $p(f(x)) \in V$, so $f(x) \in p^{-1}(V)$, which establishes that $p^{-1}(V)$ is an invariant set of f . \square

Lemma 15. Suppose that $f : S \rightarrow S$ is a dynamical system and that $Y \subseteq S$ is an invariant set for f . If $g : B \rightarrow B$ is a subsystem of f with subsystem projection p , then g is also a subsystem of $f|Y$.

Proof. Suppose that $i : Y \rightarrow S$ is the inclusion map. The hypotheses state that the diagram of solid arrows below commutes:

$$\begin{array}{ccccc}
Y & \xrightarrow{(f|Y)} & Y \\
\searrow i & & \swarrow i \\
& S & \xrightarrow{f} & S \\
\downarrow p & & & \downarrow p \\
B & \xrightarrow{g} & B
\end{array}$$

The conclusion follows by completing the diagram's dashed arrows with the composition $p \circ i$ as the subsystem projection for g as a subsystem of $f|Y$. \square

A related statement to Lemma 15 could consider the conditions under which a subsystem of an invariant set lifts to a subsystem of the entire system. Diagrammatically, this consists of a situation where the subsystem projections defined by the dashed arrows in the diagram below could be constructed:

$$\begin{array}{ccccc}
Y & \xrightarrow{(f|Y)} & Y \\
\searrow i & & \swarrow i \\
& S & \xrightarrow{f} & S \\
\downarrow & & & \downarrow \\
B & \xrightarrow{g} & B
\end{array}$$

Therefore, when studying a dynamical system, one will often encounter problems of the following form.

Question 1. When do lifts to the dashed arrows in the diagram above exist?

Answers to this question relate closely to the expected behavior of systems when they are rewritten with new variables. This routinely happens with compiled software, as the next example shows.

Example 9. Suppose that X represents the state space of a computer, perhaps a Turing machine. The design of the computer and physical laws yield a dynamical system $f : X \rightarrow X$. For this example, f is not bijective.

The way that the computer is used is that the user loads an executable and then runs it. The initial state of the executable is a point within a subset $U \subseteq X$. The user does not have control over the entire state of the machine,

but rather can constrain it to a smaller portion of the state space. It makes sense to require that U is an invariant set, which means that not only the initial state is included, but all possible future states as well. Therefore, the execution of the executable is completely determined by the commutative diagram

$$\begin{array}{ccc} U & \xrightarrow{f|_U} & U \\ \ell \downarrow & & \downarrow \ell \\ X & \xrightarrow{f} & X \end{array}$$

As an example in PDP-11 assembly, we could have

$$U = \{\text{PC} \in \{0, 1\}, \text{memory} = \{0 : \text{ADD R1, R2}, 1 : \text{HALT}\}\},$$

where all values of the unspecified parts of the machine state (other registers, the rest the memory) are included in U . If the program counter PC is initialized to 0, the program will execute the instructions at 0 and 1, and then will halt. Evidently, if $\text{PC} = 1$, then the program halts immediately. No modifications to memory can occur given an initialization with U , and PC cannot be moved outside of those two instructions. This ensures that $f(U) \subseteq U$ is indeed an invariant set.

We might instead imagine that the executable specified by U was the result of a compiled, high-level program. Such a program would necessarily be of the form $g : Y \rightarrow Y$, where Y holds the values of the two registers R1 and R2. For a PDP-11, this means $Y = (\{0, 1\}^{16})^2$, and

$$g(x, y) := (x, x + y),$$

which is to say that R1 is unchanged by the program, and R2 takes the sum of R1 and R2.

The compilation process essentially ensures that we have the following commutative diagram

$$\begin{array}{ccc} U & \xrightarrow{f|_U} & U \\ q \downarrow & & \downarrow q \\ Y & \xrightarrow{g} & Y \end{array}$$

where the q maps select the two registers R1 and R2 from the entirety of the machine state.

Notice that we may write $q = p \circ \ell$, where ℓ is the inclusion of $U \hookrightarrow X$, and p still selects the two registers R1 and R2 from the entirety of the machine state. Since the machine state is very large in comparison to U , the following diagram does *not* commute:

$$\begin{array}{ccc} X & \xrightarrow{f} & U \\ p \downarrow & & \downarrow p \\ Y & \xrightarrow{g} & Y \end{array}$$

Values of X for which the commutativity fails egregiously are instances of *weird machine states* [13].

However, when the operating system loads an executable, there are conventions about initialization. This helps to avoid weird machine states. We can formalize this idea by way of an initialization function $i : Y \rightarrow U$ that is a right inverse to q , namely $q \circ i = (p \circ \ell) \circ i = \text{id}_Y$. This means that we have the following commutative diagrams

$$\begin{array}{ccc} U & \xrightarrow{f|U} & U \\ i \uparrow & & \downarrow q \\ Y & \xrightarrow{g} & Y \end{array} \quad \begin{array}{ccc} X & \xrightarrow{f} & X \\ \ell \circ i \uparrow & & \downarrow p \\ Y & \xrightarrow{g} & Y \end{array}$$

For instance, in the example PDP-11 program, we could use

$$\begin{aligned} i(x, y) := \{ & \text{PC} = 0, \\ & \text{R1} = x, \\ & \text{R2} = y, \\ & \text{R[3-6]} = 0, \\ & \text{memory} = \{0 : \text{ADD R1,R2}, 1 : \text{HALT}, [2-] : 0\} \}, \end{aligned}$$

Notice that since i does not have the ability to change the program counter PC, the following diagram does *not* commute

$$\begin{array}{ccc} U & \xrightarrow{f|U} & U \\ i \uparrow & & \uparrow i \\ Y & \xrightarrow{g} & Y \end{array}$$

Inspired by Example 9, suppose that we have a commutative diagram

$$\begin{array}{ccc} X & \xrightarrow{f} & X \\ i \uparrow & & \downarrow p \\ Y & \xrightarrow{g} & Y \end{array}$$

where i is injective, p is surjective, and f, g are bijective.

This leads to another question that is often of interest when studying system behaviors.

Question 2. Under what conditions does

$$\begin{array}{ccc} X & \xrightarrow{f} & X \\ p \downarrow & & \downarrow p \\ Y & \xrightarrow{g} & Y \end{array}$$

commute? Clearly if g is bijective, then a sufficient condition is that $p = g^{-1} \circ p \circ f$. It is probably the case that $p \circ i = \text{id}_Y$ in most applications, but it is unlikely to be the case that $i \circ p = \text{id}_X$.

Lemma 16. *The subsystem preorder is a meet-semilattice. That is, if we have two subsystems $f_i : S_i \rightarrow S_i$ for $i = 1, 2$ of a dynamical system $f : S \rightarrow S$, there is a common subsystem $f_3 : S_3 \rightarrow S_3$ of both of them (which might be trivial) that satisfies the following universal property. If $f_4 : S_4 \rightarrow S_4$ is another common subsystem of f_1 and f_2 , then f_4 is a subsystem of f_3 .*

Proof. We start with two subsystems of a common dynamical system $f : S \rightarrow S$, so that we have a commutative diagram

$$\begin{array}{ccc} S_1 & \xrightarrow{f_1} & S_1 \\ p_1 \uparrow & & \uparrow p_1 \\ S & \xrightarrow{f} & S \\ p_2 \downarrow & & \downarrow p_2 \\ S_2 & \xrightarrow{f_2} & S_2 \end{array}$$

We want to construct a subsystem of all three of these $f_3 : S_3 \rightarrow S_3$, that is as large as possible. Realize that what is needed to satisfy the universal property is a definition for the dashed arrows in

$$\begin{array}{ccc} S & \xrightarrow{p_1} & S_1 \\ p_2 \downarrow & & \downarrow p'_3 \\ S_2 & \xrightarrow[p''_3]{\quad} & S_3 \end{array}$$

such that this diagram is a colimit.

Since each of the S_i are sets, there is a standard colimit construction, namely $S_3 = (S_1 \sqcup S_2)/\sim$ where $x \sim y$ if $x \in S_1, y \in S_2$ such that there is a $z \in S$ with $p_1(z) = x$ and $p_2(z) = y$. The colimit condition implies that when we apply this construction twice, there is a unique f_3 completing the diagram below

$$\begin{array}{ccccc} S & \xrightarrow{p_1} & S_1 & & \\ p_2 \downarrow & & \downarrow p'_3 & \searrow f_1 & \\ S_2 & \xrightarrow[p''_3]{\quad} & S_3 & \dashrightarrow & S_1 \\ & \searrow f_2 & & \swarrow f_3 & \downarrow p'_3 \\ & & S_2 & \xrightarrow[p''_3]{\quad} & S_3 \end{array}$$

□

Proposition 17. *Restrict attention to $f : S \rightarrow S$ being a (not necessarily linear) bijection on a vector space S , and require that the subsystem projection $p : S \rightarrow B$ for each subsystem (g, p) of f is a linear surjection. In this case, the relation “is a subsystem of” is also antisymmetric up to conjugacy by linear isomorphisms.*

As a result, data feedback loops are confined to happen within a given subsystem.

Proof. Suppose that (g_2, p_2) is a subsystem of $g_1 : B_1 \rightarrow B_1$, and that (g_1, p_1) is a subsystem of $g_2 : B_2 \rightarrow B_2$, so that we have the commutative diagram

$$\begin{array}{ccc} B_1 & \xrightarrow{g_1} & B_1 \\ p_2 \downarrow & & \downarrow p_2 \\ B_2 & \xrightarrow{g_2} & B_2 \\ p_1 \downarrow & & \downarrow p_1 \\ B_1 & \xrightarrow{g_1} & B_1 \end{array}$$

Since p_1 and p_2 are surjective linear maps, this means that $(p_1 \circ p_2) : B_1 \rightarrow B_1$ is a linear surjection. Since it also evidently preserves dimension, it must be a linear isomorphism. Because both p_1 and p_2 are surjective, this implies that both must also be injective. Hence both p_1 and p_2 must also be linear isomorphisms, which establishes that $g_2 = p_2 \circ g_1 \circ p_2^{-1}$ and $g_1 = p_1 \circ g_2 \circ p_1^{-1}$ as claimed. \square

Example 10. There is no function h that will make the diagram below commute

$$\begin{array}{ccc} \mathbb{Z}^2 & \xrightarrow{f} & \mathbb{Z}^2 \\ \text{id} \downarrow & & \downarrow \text{id} \\ \mathbb{Z}^2 & \xrightarrow{h} & \mathbb{Z}^2 \\ \text{id} \uparrow & & \uparrow \text{id} \\ \mathbb{Z}^2 & \xrightarrow{g} & \mathbb{Z}^2 \end{array}$$

where

$$f(x, y) = (x, 1 - x),$$

and

$$g(x, y) = (y, y).$$

There is also no function h that will make the diagram below commute

$$\begin{array}{ccc}
\mathbb{Z}^2 & \xrightarrow{f} & \mathbb{Z} \\
\text{pr}_1 \downarrow & & \downarrow \text{id} \\
\mathbb{Z} & \xrightarrow{h} & \mathbb{Z} \\
\text{pr}_2 \uparrow & & \uparrow \text{id} \\
\mathbb{Z}^2 & \xrightarrow{g} & \mathbb{Z}
\end{array}$$

where

$$f(x, y) = 1 - x,$$

and

$$g(x, y) = y.$$

Suppose that $f : S \rightarrow S$ is a dynamical system in which S is a vector space and the subsystem projections are all linear surjections, as required by Proposition 17. Let (\mathcal{B}, \leq) be the collection of all subsystems of f , with the partial order established by Lemma 13 and Proposition 17. Each element of \mathcal{B} is a pair (g_B, p_B) where $g_B : B \rightarrow B$ is a bijection and $p_B : S \rightarrow B$. For brevity, if g_1 is a subsystem of g_2 , which is to say that there is a $p_{1,2} : B_2 \rightarrow B_1$ such that $p_1 = p_{1,2} \circ p_2$, we write $(g_1, p_1) \leq (g_2, p_2)$.

Definition 21. Define the *sheaf \mathcal{F}_f of subsystems of f* according to the following recipe:

Stalks $\mathcal{F}_f((g_B, p_B)) := B$, and

Restrictions $\mathcal{F}_f((g_1, p_1) \leq (g_2, p_2)) := p_{1,2}$.

Even if the subsystem projections are not linear surjections, the Alexandrov topology on the subsystem preorder bundles together all collections of subsystems that participate in cycles. Without the conclusion of Proposition 17, the stalks of \mathcal{F}_f are not necessarily well defined, since there is no guarantee that the subsystems of a given cycle have the same state spaces.

Lemma 18. *For a dynamical system $f : S \rightarrow S$, the space of global sections of \mathcal{F}_f is precisely S .*

Proof. First of all, notice that $\text{id}_S : S \rightarrow S$ meets the criteria for a subsystem. We merely need to verify that the definition of global sections for \mathcal{F}_f doesn't conflict with this. The space of assignments for \mathcal{F}_f is

$$\bigoplus_{p: S \rightarrow B \text{ subsystem}} \mathcal{F}_f(p) = \bigoplus_{p: S \rightarrow B \text{ subsystem}} B.$$

Suppose that we have a global section s . On the other hand, if $(g_B, p_B) \leq (f, \text{id}_S)$, then

$$(\mathcal{F}_f((g_B, p_B) \leq (f, \text{id}_S))) (s(S)) = p_B(s(S)) = s(B).$$

Therefore, the value of s on the subsystem $\text{id}_S : S \rightarrow S$ determines the values of s on every other subsystem. \square

Proposition 19. *A dynamical system $f : S \rightarrow S$ induces an endomorphism on the sheaf of all subsystems, and for which the induced map on global sections is f .*

Proof. This follows immediately from the definition, as soon as we notice that for a subsystem $p : S \rightarrow B$, the g map guaranteed by the definition is the corresponding component map for the sheaf morphism. \square

In short, a multi-scale discrete dynamical system can be encoded as component dynamical systems on some (or all) of the stalks of a sheaf \mathcal{S} via self maps $f_x : \mathcal{S}(x) \rightarrow \mathcal{S}(x)$. One may also consider the action of different semigroups on stalks to model continuous dynamical systems.

We are now ready to establish the main result of this section, which relates the sheaf of subsystems of a DSEM to its graph representation. As we have seen in Example 5, feedback loops in the DSEM graph must be confined to being entirely within a subsystem. Because we can collapse all feedback loops in an arbitrary directed graph to obtain an acyclic graph, we will assume that the DSEM graph is acyclic without loss of generality.

The key insight is that if we select a given variable in the DSEM, any subsystem containing that variable must also contain every variable that can impact its value. Any variable with a directed path leading to our variable of interest will therefore need to be included in the subsystem.

Definition 22. In a directed graph $G = (V, E)$ an *in-closed* subset $I \subseteq V$ is a set of vertices such that if $v \in I$, then if $e = (w, v) \in E$, then $w \in I$.

Lemma 20. *If a dynamical system is defined by a DSEM, every in-closed subset of variables is a subsystem.*

Proof. Suppose that I is a in-closed subset of variables in a DSEM on a directed graph G . If $v \in I$ then all of the dependencies of v are also in I , so the next timestep of v can be predicted from the variables in I . Therefore, projecting out just the variables in I from the set of all variables will result in a new dynamical update map when restricted to I . \square

As a consequence of Lemma 20, we have the following result that explains why modeling with DSEM is a good idea.

Corollary 21. *If a dynamical system is defined by a DSEM on a partially ordered set, then the Alexandrov topology of the dual order is a subspace of the base space topology of its subsystem sheaf.*

Corollary 21 does not establish that the Alexandrov topology of the dual order of the DSEM *is* the subsystem sheaf. This is because if the original variables in the DSEM are chosen coarsely, there may be additional subsystems that are “hidden” within them. These hidden subsystems will be present in the subsystem sheaf, but will not correspond to distinct in-closed subsets of the DSEM graph.

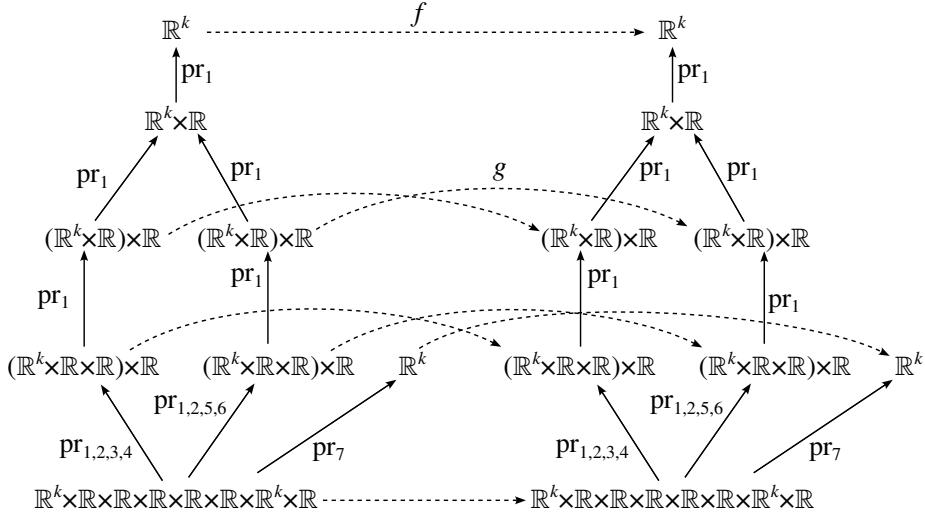


Figure 14: Sheaf of subsystems for the Bering Sea example. Solid arrows are the subsystem projection maps; dashed arrows are the dynamical system state update maps. Maps f and g are explained in the text.

6 Subsystems of the Bering Sea system

Figure 14 shows the sheaf of subsystems for the Bering Sea example, with the stalks organized in the same way as shown in Figure 13.

The function f performs an AR (k) update:

$$f(x_1, \dots, x_k) = \left(x_2, \dots, x_k, \sum_{i=0}^{k-1} a_i x_{k-i} \right),$$

while the function g performs the dynamical update for the subsystem containing the *Krill* variables:

$$g(x_1, \dots, x_k, y, z) = \left(x_2, \dots, x_k, \sum_{i=0}^{k-1} a_i x_{k-i}, y + cx_k, z + dy \right).$$

Notice how f is obtained from g by projecting out the first k components, in accordance with the commutativity of Figure 14.

Although Figures 1(d) (with modifications to support autoregressive time-series), 13, and 14 represent different sheaves, they all represent the same dynamical system. Consequently, the global sections of these three sheaves are different but are in a natural bijective correspondence. The three sheaves offer three distinct perspectives, with increasing granularity,

Definition 21: Figure 14 Stalks are nested collections of dynamically related variables, each represented by sliding windows of timeseries,

Definition 13: Figure 1(d) Each variable is an entire timeseries and appears alone in at least one stalk, and

Definition 14: Figure 13 Each observation (a timestep for a single variable) appears alone in at least one stalk.

With this perspective, the boundaries between subsystems are easily seen in Figure 13: those restriction maps that are identity maps from parts to nets are those that cross subsystem boundaries. The variables at the heads of any identity maps in Figure 13 are those that are removed by the subsystem projections involved. Moreover, the state spaces arise as one time step of the space of local sections over each subsystem, once cut.

7 Conclusion

In this chapter, we have demonstrated how the general framework of sheaf modeling applies to several composite dynamical systems, including an ecological model of the Bering Sea and a dynamical model of low-level computer software. Sheaf modeling provides a coherent mathematical framework for studying the complicated interaction of various dynamical subsystems that together determine a larger system. The guiding principles of sheaf modeling are that

- a sheaf represents a hypothesis about how variables will interact,
- a non-global assignment represents the observations collected on the variables in its support,
- minimizing consistency radius predicts values of the variables that were not observed, and
- the minimal consistency radius is a measure of the consistency between the observations and the hypothesis.

This chapter shows that when a dynamical system is described by a DSEM, there are three sheaves that provide increasingly granular data about the interactions between variables:

1. the sheaf of subsystems (Definition 21),
2. the netlist sheaf with timeseries as stalks (Definition 13), and
3. the netlist sheaf with additional stalks for individual observations (Definition 14).

With these three sheaves in hand, a system modeler can apply the guiding principles above to measure how well their model fits observational data. The sheaf encodings allow the modeler to perform a variety of standard inferences (e.g. forward prediction, backward prediction, regression, and missing-data imputation) using a unified framework. The sheaf modeling framework easily supports

hybrid versions, for instance performing simultaneous forward and backward predictions, or simultaneously performing regression and prediction. Since the sheaf framework measures the fit between observations and the model, the modeler can assess their confidence in these inference tasks.

It remains future work to compare estimates of uncertainty computed by the DSEM (appearing in the \mathbf{V} and \mathbf{E} matrices) to the consistency radius of the corresponding sheaf. In particular, it seems possible to view consistency radius as a test statistic for the distributional model posited by the DSEM. Indeed, Equation (10) is strikingly close to the log likelihood if the distributions of measurement errors are assumed to follow an exponential model. If this is true, then it should be possible to lift the sheaf modeling discipline described here into a standard statistical hypothesis testing framework.

Acknowledgments

The linear regression example in Section 3.3 is due to Donna Dietz.

This article is based upon work supported by the Office of Naval Research (ONR) under Contract Nos. N00014-15-1-2090 and N00014-18-1-2541, the Defense Advanced Research Projects Agency (DARPA) SafeDocs program under contract HR001119C0072, and the MITRE Corporation’s Independent Research and Development (IR&D) Program. Any opinions, findings and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of ONR, DARPA, or MITRE.

References

- [1] Martín Abadi and Leslie Lamport. The existence of refinement mappings. *Theoretical Computer Science*, 82(2):253–284, 1991. ISSN 0304-3975. doi: [https://doi.org/10.1016/0304-3975\(91\)90224-P](https://doi.org/10.1016/0304-3975(91)90224-P). URL <https://www.sciencedirect.com/science/article/pii/030439759190224P>.
- [2] Christopher Alexander. *Notes on the synthesis of form*. Harvard University Press, 1964.
- [3] G. W. Altman, L. A. Decampo, and C R. Warburton. Automation of computer panel wiring. *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, 79(2):118–125, 1960. doi: 10.1109/TCE.1960.6368554.
- [4] James Anderson and Antonis Papachristodoulou. Dynamical system decomposition for efficient, sparse analysis. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6565–6570, 2010. doi: 10.1109/CDC.2010.5717269.
- [5] Suchinta Arif and M. Aaron MacNeil. Applying the structural causal model framework for observational causal inference in ecology. *Ecological*

Monographs, 93(1):e1554, 2023. ISSN 1557-7015. doi: 10.1002/ecm.1554.
URL <https://onlinelibrary.wiley.com/doi/abs/10.1002/ecm.1554>.
_eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/ecm.1554>.

- [6] Ross Ashby. *Introduction to Cybernetics*. Methuen, London, 1956.
- [7] Jonathan Barmak. *Algebraic Topology of Finite Topological Spaces and Applications*. Springer, 2011.
- [8] Noemi Chiriac, Katja Hölttä-Otto, Dusan G. Lysy, and Eun Suk Suh. Three approaches to complex system decomposition. In *13th International Dependency and structure modeling conference (DSM11)*, Cambridge, MA, USA, September 2011.
- [9] A. J. Chorin, O. H. Hald, and R. Kupferman. Optimal prediction with memory. *Physica D: Nonlinear Phenomena*, 166:239–257, 2002.
- [10] K. O. Coyle and G. A. Gibson. Calanus on the Bering Sea shelf: probable cause for population declines during warm years. *Journal of Plankton Research*, 39(2):257–270, March 2017. ISSN 0142-7873. doi: 10.1093/plankt/fbx005. URL <https://doi.org/10.1093/plankt/fbx005>.
- [11] Timothy A Davis and Yifan Hu. The University of Florida sparse matrix collection. *ACM Transactions on Mathematical Software (TOMS)*, 38(1):1–25, 2011. Publisher: ACM.
- [12] Janet T. Duffy-Anderson, Phyllis J. Stabeno, Elizabeth C. Siddon, Alex G. Andrews, Daniel W. Cooper, Lisa B. Eisner, Edward V. Farley, Colleen E. Harpold, Ron A. Heintz, David G. Kimmel, Fletcher F. Sewall, Adam H. Spear, and Ellen C. Yasumishii. Return of warm conditions in the southeastern Bering Sea: Phytoplankton - Fish. *PLOS ONE*, 12(6): e0178955, June 2017. ISSN 1932-6203. doi: 10.1371/journal.pone.0178955. URL <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0178955>. Publisher: Public Library of Science.
- [13] Thomas Dullien. Weird machines, exploitability, and provable unexploitability. *IEEE Transactions on Emerging Topics in Computing*, 8(2):391–403, 2020. doi: 10.1109/TETC.2017.2785299.
- [14] F. Fetterer, K. Knowles, W. N. Meier, M. Savoie, and A. K. Windnagel. Sea ice index, version 3: Regional daily data (updated daily). NSIDC, 2017.
- [15] Andrew P. Foss-Grant, Elise F. Zipkin, James T. Thorson, Olaf P. Jensen, and William F. Fagan. Hierarchical analysis of taxonomic variation in intraspecific competition across fish species. *Ecology*, 97(7):1724–1734, July 2016. ISSN 1939-9170. doi: 10.1890/15-0733.1. URL <http://onlinelibrary.wiley.com/doi/10.1890/15-0733.1/abstract>.

- [16] James B. Grace and Kathryn M. Irvine. Scientist's guide to developing explanatory statistical models using causal analysis principles. *Ecology*, 101(4):e02962, 2020. ISSN 1939-9170. doi: 10.1002/ecy.2962. URL <https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1002/ecy.2962>.
- [17] Francis Heylighen and Cliff Joslyn. Cybernetics and second order cybernetics. In *Encyclopedia of Physical Science and Technology*. Academic Press, 2003. doi: 10.1016/B0-12-227410-5/00161-7.
- [18] Yuji Hirono, Takashi Okada, Hiroyasu Miyazaki, and Yoshimasa Hidaka. Structural reduction of chemical reaction networks based on topology. *Phys. Rev. Res.*, 3:043123, Nov 2021. doi: 10.1103/PhysRevResearch.3.043123. URL <https://link.aps.org/doi/10.1103/PhysRevResearch.3.043123>.
- [19] George L. Hunt, Jr, Kenneth O. Coyle, Lisa B. Eisner, Edward V. Farley, Ron A. Heintz, Franz Mueter, Jeffrey M. Napp, James E. Overland, Patrick H. Ressler, Sigrid Salo, and Phyllis J. Stabeno. Climate impacts on eastern Bering Sea foodwebs: a synthesis of new data and an assessment of the Oscillating Control Hypothesis. *ICES Journal of Marine Science*, 68(6):1230–1243, July 2011. ISSN 1054-3139. doi: 10.1093/icesjms/fsr036. URL <https://doi.org/10.1093/icesjms/fsr036>.
- [20] James N. Ianelli, Sarah Stienessen, Taina Honkalehto, Elizabeth Siddon, and Caitlin Allen-Akselrud. Assessment of the walleye pollock stock in the Eastern Bering Sea. NPFMC Bering Sea and Aleutian Islands SAFE, North Pacific Fishery Management Council, Anchorage, AK, 2022.
- [21] M. E. Johnson and J. Kalvenes. Subsystem decomposition in simulation of a PCB assembly line. *Proceedings of 1993 Winter Simulation Conference - (WSC '93)*, pages 790–795, 1993.
- [22] Griffin M. Kearney, Kevin F. Palmowski, and Michael Robinson. Sheaf-theoretic framework for optimal network control, arxiv: 2012.00120, 2020.
- [23] Jonas Knape and Perry de Valpine. Are patterns of density dependence in the Global Population Dynamics Database driven by uncertainty about population abundance? *Ecology Letters*, 15(1):17–23, January 2012. ISSN 1461-0248. doi: 10.1111/j.1461-0248.2011.01702.x. URL <http://onlinelibrary.wiley.com/doi/10.1111/j.1461-0248.2011.01702.x/abstract>.
- [24] H. Komoto and T. Tomiyama. Multi-disciplinary system decomposition of complex mechatronics systems. *CIRP Annals*, 60(1):191–194, 2011. ISSN 0007-8506. doi: <https://doi.org/10.1016/j.cirp.2011.03.102>. URL <https://www.sciencedirect.com/science/article/pii/S000785061100103X>.

- [25] Kasper Kristensen, Anders Nielsen, Casper W. Berg, Hans Skaug, and Bradley M. Bell. TMB: Automatic differentiation and Laplace approximation. *Journal of Statistical Software*, 70(5):1–21, 2016. doi: 10.18637/jss.v070.i05.
- [26] Arthur Liberzon, Chet Birger, Helga Thorvaldsdóttir, Mahmoud Ghandi, Jill P. Mesirov, and Pablo Tamayo. The molecular signatures database hallmark gene set collection. *Cell systems*, 1:417–425, 2015. URL <https://api.semanticscholar.org/CorpusID:86123173>.
- [27] Jakob F. Maier, Claudia M. Eckert, and P. John Clarkson. Model granularity in engineering design – concepts and framework. *Design Science*, 3, 2017.
- [28] Bennett T. McCallum. Is the spurious regression problem spurious? *Economics Letters*, 107(3):321–323, June 2010. ISSN 0165-1765. doi: 10.1016/j.econlet.2010.02.004. URL <https://www.sciencedirect.com/science/article/pii/S0165176510000674>.
- [29] C. Montes de Oca and D.L. Carver. A visual representation model for software subsystem decomposition. In *Proceedings Fifth Working Conference on Reverse Engineering (Cat. No.98TB100261)*, pages 231–240, 1998. doi: 10.1109/WCRE.1998.723193.
- [30] J. Murphy, S. Garcia, J. Dimond, J. Moss, F. Sewall, W. Strasburger, E. Lee, T. Dann, E. Labunski, and T. Zeller. Northern Bering Sea surface trawl and ecosystem survey cruise report, 2019. *NOAA Technical Memorandum NMFS-AFSC-423*, 2021. URL https://repository.library.noaa.gov/view/noaa/32075/noaa_32075_DS1.pdf.
- [31] Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009. ISSN 1935-7516. doi: 10.1214/09-SS057. URL <http://projecteuclid.org/euclid.ssu/1255440554>.
- [32] Patrick H. Ressler, Alex De Robertis, Joseph D. Warren, Joy N. Smith, and Stan Kotwicki. Developing an acoustic survey of euphausiids to understand trophic interactions in the Bering Sea ecosystem. *Deep Sea Research Part II: Topical Studies in Oceanography*, 65-70:184–195, June 2012. ISSN 0967-0645. doi: 10.1016/j.dsr2.2012.02.015. URL <https://www.sciencedirect.com/science/article/pii/S096706451200029X>.
- [33] W. E. Ricker. Stock and Recruitment. *Journal of the Fisheries Research Board of Canada*, 11(5):559–623, May 1954. ISSN 0015-296X. doi: 10.1139/f54-039. URL <https://cdnsciencepub.com/doi/10.1139/f54-039>. Publisher: NRC Research Press.
- [34] Michael Robinson. Sheaf and duality methods for analyzing multi-model systems. In I. Pesenson, Q.T. Le Gia, A. Mayeli, H. Mhaskar, and D.-X. Zhou, editors, *Novel Methods in Harmonic Analysis*. Birkhäuser, 2017.

- [35] Michael Robinson. Assignments to sheaves of pseudometric spaces. *Compositionality*, 2(2), 2020.
- [36] Michael Robinson. Netlist sheaf Python library <https://github.com/kb1dds/netlist-sheaf>, 2025.
- [37] S. K. Rohan, L. A. K. Barnett, and N. Charriere. Evaluating approaches to estimating mean temperatures and cold pool area from AFSC bottom trawl surveys of the eastern Bering Sea. *NOAA Technical Memorandum NMFS-AFSC*, 456:42, 2022.
- [38] E. Siddon. Ecosystem Status Report 2022: Eastern Bering Sea. Technical report, North Pacific Fishery Management Council, 2022.
- [39] Hans Skaug and Dave Fournier. Automatic approximation of the marginal likelihood in non-Gaussian hierarchical models. *Computational Statistics & Data Analysis*, 51(2):699–709, 2006.
- [40] Christoffer Sloth. On the computation of Lyapunov functions for interconnected systems. In *2016 IEEE Conference on Computer Aided Control System Design (CACSD)*, pages 850–855, 2016. doi: 10.1109/CACSD.2016.7602545.
- [41] Joy N. Smith, Patrick H. Ressler, and Joseph D. Warren. A distorted wave Born approximation target strength model for Bering Sea euphausiids. *ICES Journal of Marine Science*, 70(1):204–214, January 2013. ISSN 1054-3139. doi: 10.1093/icesjms/fss140. URL <https://doi.org/10.1093/icesjms/fss140>.
- [42] T Smith and Andre E. Punt. The gospel of maximum sustainable yield in fisheries management: birth, crucifixion and reincarnation. In John D. Reynolds, Georgina M. Mace, Kent H. Redford, and John G. Robinson, editors, *Conservation of exploited species*, pages 41–66. Cambridge University Press, Cambridge, UK, 2001.
- [43] Donald Steward. The design structure system: a method for managing the design of complex systems. *IEEE Transactions on Engineering Management*, EM-28(3), August 1981.
- [44] Steven Strogatz. *Nonlinear Dynamics And Chaos: With Applications To Physics, Biology, Chemistry, And Engineering*. CRC Press, 2000.
- [45] Eun Suk Suh, Noemi Chiriac, and Katja Hölttä-Otto. Seeing complex system through different lenses: Impact of decomposition perspective on system architecture analysis. *Syst. Eng.*, 18:229–240, 2015.
- [46] James T Thorson, Alexander G Andrews III, Timothy E Essington, and Scott I Large. Dynamic structural equation models synthesize ecosystem dynamics constrained by ecological mechanisms. *Methods in Ecology and Evolution*, 15(4):744–755, 2024.

- [47] James T. Thorson, Alexander G. Andrews III, Timothy E. Essington, and Scott I. Large. Dynamic structural equation models synthesize ecosystem dynamics constrained by ecological mechanisms. *Methods in Ecology and Evolution*, 15(4):744–755, 2024. ISSN 2041-210X. doi: 10.1111/2041-210X.14289. URL <https://onlinelibrary.wiley.com/doi/abs/10.1111/2041-210X.14289>. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/2041-210X.14289>.
- [48] David Tilman and Peter M. Kareiva. *Spatial Ecology: The Role of Space in Population Dynamics and Interspecific Interactions*. Princeton University Press, 1997. ISBN 0-691-01652-6.
- [49] David Wynn and P. John Clarkson. Process models in design and development. *Research in Engineering Design*, 29:161–202, 2018.
- [50] Xunyuan Yin, Kevin Arulmaran, and Jinfeng Liu. Subsystem decomposition for distributed state estimation of nonlinear systems. *2016 American Control Conference (ACC)*, pages 5569–5574, 2016.
- [51] Robert Zwanzig. *Nonequilibrium Statistical Mechanics*. Oxford University Press, New York, 2001.