



Peer Reviewed

Title:

Surface diagrams for gray-categories

Author:

[Hummon, Benjamin Taylor](#)

Acceptance Date:

2012

Series:

[UC San Diego Electronic Theses and Dissertations](#)

Degree:

Ph. D., [MathematicsUC San Diego](#)

Permalink:

<http://escholarship.org/uc/item/5b24s9cc>

Local Identifier(s):

Abstract:

We exhibit a calculus of dot, string, and surface diagrams for describing computadic compositions. A surface diagram is a cube equipped with a stratification of regions, walls, seams, and nodes. We label the strata with morphisms in a strict 3-category C by codimension. Away from nodes, horizontal slice stratified squares are string diagrams representing compositions of 2-morphisms. In particular, a surface diagram has source and target string diagrams on its bottom and top faces, respectively. We may evaluate a surface diagram to give a 3-morphism in C . The domain and codomain are given by the evaluation of the source and target string diagrams. We build a 3-category $Sd(C)$ of surface diagrams labeled by C in which composition is given by gluing along common faces. More generally, there is a 3-category $Sd(G)$ of surface diagrams for any 3-computad G of generators. We characterize $Sd(G)$ as the free 3-category on G . In $Sd(G)$, diagrams are taken up to an isotopy-like relation called evolution. This relation destroys the braiding that we expect to have in a tricategory. We wish to capture braiding without working in the fully weak setting of a tricategory. We instead choose to work with in the semi-strict setting of Gray- categories. We define Gray surface diagrams to be those with certain good projection properties. Working up to an appropriate form of evolution, we construct the free Gray- category $SdGray(G)$ of Gray-surface diagrams on a Gray- category. Last, we demonstrate the utility of surface diagrams by studying coherence relations for equivalence in a Gray-category. We show that the data encoding any incoherent equivalence may be recast as a coherent equivalence

Copyright Information:

All rights reserved unless otherwise indicated. Contact the author or original publisher for any necessary permissions. eScholarship is not the copyright owner for deposited works. Learn more at http://www.escholarship.org/help_copyright.html#reuse



UNIVERSITY OF CALIFORNIA, SAN DIEGO

Surface Diagrams for Gray-Categories

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Mathematics

by

Benjamin Taylor Hummon

Committee in charge:

Professor Justin Roberts, Chair
Professor Ken Intriligator
Professor Sorin Lerner
Professor Dan Rogalski
Professor Hans Wenzl

2012

Copyright
Benjamin Taylor Hummon, 2012
All rights reserved.

The dissertation of Benjamin Taylor Hummon is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2012

TABLE OF CONTENTS

Signature Page		iii
Table of Contents		iv
Acknowledgements		vi
Vita and Publications		vii
Abstract of the Dissertation		viii
Chapter 1	Introduction	1
Chapter 2	Intuition for Dot Diagrams	4
Chapter 3	Intuition for String Diagrams	9
	3.1 2-Morphisms as String Diagrams	9
	3.2 Degrees and Dimensions	10
	3.3 Slicing and Projecting	12
	3.4 2-Computads and the Free 2-Category	15
	3.5 Monoidal Categories	21
	3.6 Bicatagories	23
Chapter 4	Intuition for Surface Diagrams	26
	4.1 3-Categories	27
	4.2 Tricategories	31
	4.3 Gray-Categories	34
Chapter 5	Topological Background	37
	5.1 Manifolds with Corners	37
	5.2 Stratified Spaces	48
Chapter 6	Dot Diagrams	53
	6.1 1-Computads	53
	6.2 Dot Diagrams	54
	6.3 Dot Evolutions	55
	6.4 Gluing and Normalization	57
	6.5 The Category $\text{dd}(G)$	59
	6.6 Evaluation	60
	6.7 Characterizing $\text{dd}(G)$ as the Free Category	63
Chapter 7	String Diagrams	65
	7.1 2-Computads	65
	7.2 String Diagrams	66
	7.3 String Evolutions	69
	7.4 Gluing and Normalization	71
	7.5 The 2-Category $\text{sd}(G)$	73

	7.6	Cone Diagrams	75
	7.7	Evaluation	76
	7.8	Characterizing $\text{sd}(G)$ as the Free 2-Category	80
Chapter 8		Surface Diagrams	82
	8.1	3-Computads	82
	8.2	Surface Diagrams	83
	8.3	Surface Evolutions	84
	8.4	The 3-Category $\text{Sd}(G)$	85
	8.5	Cone Diagrams	87
	8.6	Evaluation	88
	8.7	Characterizing $\text{Sd}(G)$ as the Free 3-Category	89
Chapter 9		Surface Diagrams for Gray-Categories	90
	9.1	The Free Sesquicategory of Ordered String Diagrams	90
	9.2	Gray Surface Diagrams	92
Chapter 10		Equivalence in 2-Categories	96
	10.1	About Coherence for Equivalence	96
	10.2	Coherification	97
	10.3	Application: Morita Equivalence of Rings	100
Chapter 11		Equivalence in Gray-Categories	103
	11.1	Semi-Coherification	103
	11.2	Morse-Cancellation	106
	11.3	Swallowtail	107
	11.4	Cusp-Flip	108
	11.5	Coherification	111
Chapter 12		Future Work	114
Appendix A		Smooth Functions	116
Appendix B		Adjoint Functors	118
Appendix C		Sesquicategories and Gray-Categories	122
Bibliography		130

ACKNOWLEDGEMENTS

I'd like to thank Justin Roberts for all of his guidance as my graduate advisor. He has been very generous with his time sharing both his expertise and enthusiasm for maths. And he continues to be an exceptional source of encouragement for which I can't thank him enough. I've also been very lucky to have Michael Shulman as a mentor. He, too, seems to have unlimited energy for discussing mathematics. His categorical expertise was critical to help me find a focus for my research. This dissertation owes much to his guidance and inspiration. I must also thank Theresa Jones and John Foley for being great sources of support. And for sending me all of the cute animal photos to cheer me up, in Theresa's case, and for sharing those buckets of sangria in the park, in John's. Last, I thank Noel Dwyer for the endless amount of time sketching diagrams and "talking math" with me.

VITA

- 2006 Bachelor of Arts with *Honors in Mathematics* and *General Honors*,
Vassar College, Poughkeepsie
- 2006-2011 Teaching Assistant, Department of Mathematics, University of Cali-
fornia, San Diego
- 2011, 2012 Associate Instructor, Department of Mathematics, University of Cal-
ifornia, San Diego
- 2012 Doctor of Philosophy, in Mathematics, University of California, San
Diego

ABSTRACT OF THE DISSERTATION

Surface Diagrams for Gray-Categories

by

Benjamin Taylor Hummon

Doctor of Philosophy in Mathematics

University of California, San Diego, 2012

Professor Justin Roberts, Chair

We exhibit a calculus of dot, string, and surface diagrams for describing computadic compositions. A surface diagram is a cube equipped with a stratification of regions, walls, seams, and nodes. We label the strata with morphisms in a strict 3-category C by codimension. Away from nodes, horizontal slice stratified squares are string diagrams representing compositions of 2-morphisms. In particular, a surface diagram has source and target string diagrams on its bottom and top faces, respectively. We may evaluate a surface diagram to give a 3-morphism in C . The domain and codomain are given by the evaluation of the source and target string diagrams. We build a 3-category $\text{Sd}(C)$ of surface diagrams labeled by C in which composition is given by gluing along common faces. More generally, there is a 3-category $\text{Sd}(G)$ of surface diagrams for any 3-computad G of generators. We characterize $\text{Sd}(G)$ as the free 3-category on G .

In $\text{Sd}(G)$, diagrams are taken up to an isotopy-like relation called evolution. This

relation destroys the braiding that we expect to have in a tricategory. We wish to capture braiding without working in the fully weak setting of a tricategory. We instead choose to work with in the semi-strict setting of Gray-categories. We define Gray surface diagrams to be those with certain good projection properties. Working up to an appropriate form of evolution, we construct the free Gray-category $\text{Sd}^{\text{Gray}}(G)$ of Gray-surface diagrams on a Gray-category.

Last, we demonstrate the utility of surface diagrams by studying coherence relations for equivalence in a Gray-category. We show that the data encoding any incoherent equivalence may be recast as a coherent equivalence.

Chapter 1

Introduction

Context

Category theory was discovered in the early 1940s as a language for expressing relationships in algebraic topology. In their work on Čech cohomology, Eilenberg and MacLane first wrote about categories, functors, and natural transformations in order to understand certain limits. In *Categories for the Working Mathematician* [Mac71], MacLane notes the remarkable fact that the arrow notation $f: X \rightarrow Y$ for functions only predates their paper by about two years. From the beginning, people thought of categories as a tool for writing and organizing mathematics. We'll take this "category theory as language" principle quite seriously. Indeed, in some sense, the central results of this dissertation are all of the form, "calculations in a higher category may be written using diagrams." It's also worth noticing that higher categories already make an appearance in Eilenberg and MacLane's first paper: natural transformations should be thought of as the 2-morphisms in a 2-category of categories.

Our discussion now jumps ahead several decades to the early 1990s. In a series of papers [JS91a] [JS91b] [JS93], André Joyal and Ross Street give theories of string diagrams for various types of tensor categories: monoidal categories, braided monoidal categories, symmetric monoidal categories, balanced, autonomous, pivotal, etc. Of particular interest to us, they also gave a theory of string diagrams for 2-categories. While these diagrams were used in the experts' private calculations, Joyal and Street gave the categorical diagrams a foundation by building topological models. While one may read Joyal and Street's theorems as using topology to build a categorical structure, these theorems may equally well be read as giving categorical characterizations of topological structure. In [Shu94], Mei Chee Shum

took this another step further by studying tortile tensor categories. These are monoidal categories equipped with a twist and duals. She describes the free tortile tensor category using framed tangles. Work of Peter Freyd and David Yetter, as well as, J. Scott Carter, Joachim Rieger, and Masahico Saito have drawn further connections between knot theory and higher categories. So not only is category theory a good language for topology, they are actually fundamentally linked in this way.

We may see recent work on extended topological field theories as a close cousin to our surface diagrams. In his 2009 dissertation [SP09], Christopher Schommer-Pries gives a classification of 2D extended TFTs. An extended TFT may be thought of as a tensor functor from a bicategory of bordisms. He characterizes the bordism bicategory as a free symmetric monoidal bicategory with certain duals generated by the point. In parallel, Jacob Lurie developed a sweeping account [Lur09] of TFTs using much fancier technology. Using a theory of symmetric monoidal (∞, n) -categories, he pulls off a delicate induction which similarly characterizes a category of n -dimensional bordisms as being generated by a fully-dualizable point. This result characterizes the n -dimensional extended TFTs.

An extended TFT is different from the theory of surface diagrams we propose in three important ways. First, bordisms are not allowed to have the singular cones that are at the center of our theory. Instead, we may think of a bordism as a manifold with corners with various faces declared as input or output. Second, bordisms are *stabilized* in that we may think of them sitting in a very large dimensional ambient space. This means that the lowest levels of composition just give a symmetric tensor product. We also demand a high level of duality to capture the various ways that the bordism may sit in its ambient space. Third, we only need a single label for a bordism in order to define a TFT. That is, we may think of a TFT as having a single non-region generator. All of the various bordisms are determined as data recording various duals and adjoints for the generating point. From this perspective, the coherence data for an equivalence very much resembles the 2-dimensional bordism category. We hope to explore deeper connections in this direction in future work.

Overview

This dissertation is divided into three main parts. The first part is comprised of Chapters 2 – 4. Here, we give an informal account of dot, string and surface diagrams. We aim to explain how to use these diagrams and motivate their definition. We opt to draw pictures of the diagrams in place of giving precise mathematical definitions and proof.

In the second part of the dissertation, Chapters 5 – 9, we give a formal account of dot, string, and surface diagrams. We begin by giving a topological foundation for the diagrams as manifolds with corners given stratifications. We define dot diagrams for making calculations in 1-categories, string diagrams for calculations in 2-categories, and surface diagrams for calculations in 3-categories. We also define variants of these diagrams for calculating in sesquicategories and Gray-categories. The Gray surface diagrams are more interesting than their strict 3-categorical counterpart. We wish to capture the interesting phenomena of tricategories while avoiding their often overwhelming complexity.

Chapters 10 and 11 make up the third main part of the dissertation. This contains a stand-alone application of surface diagrams. We study adjoint equivalences first in the context of 2-categories and then in Gray-categories. These results are well known to experts in the 2-categorical case. Others, including Gurski in [Gur11a] and Lack in [Lac11], have given similar results in the context of tricategories. Our result demonstrates the utility of surface diagrams. What would be an essentially incomprehensible series of calculations as algebra, becomes a much less formidable manipulation of surfaces.

Last, in Chapter 12 we give a brief overview of material that did not make it into the thesis, but which the author hopes to explore in the future.

Chapter 2

Intuition for Dot Diagrams

A dot diagram is used to display a composition of morphisms in a category. It is an alternative to the popular diagrams of arrows, and in fact, the two are closely related. Here is an example of a dot diagram and the corresponding arrow diagram:

$$\begin{array}{ccccccc} X & f & Y & h & X & k & Z \\ \hline & \bullet & & \bullet & & \bullet & \end{array} \qquad \begin{array}{ccccccc} X & f & Y & h & X & k & Z \\ \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet & \longrightarrow & \bullet \end{array}$$

Notice that, unlike an arrow diagram, the dimensions of the cells in a dot diagram do not agree with the degree of the morphism! That is, the (degree 0) objects X, Y , and Z label (1-D) *regions* and the (degree 1) morphisms f, h and k label (0-D) *dots*. Also, notice that the dot diagram begins and ends with regions instead of dots. The regions adjacent to a dot give the source and target objects for that morphism. It should be clear how one obtains an arrow diagram from a dot diagram: we replace each region with a vertex and each dot with an arrow. We refer to this process as *geometric dualization* of diagrams. We're using the adjective "geometric" to distinguish it from the categorical idea of dualization which "reverses all arrows."

Often, as working mathematicians, we wish to draw commutative diagrams of arrows with several paths of head-to-tail arrows. However, dot diagrams only correspond to arrow diagrams made of a single path of head-to-tail arrows. What value does such a diagram have if we can't even express a commutativity square? The traditional definition of category has a two-to-one composition operation and a zero-to-one unit operation. But in fact, the associative and unit laws ensure that composition is well-defined on a path of head-to-tail morphisms of any length. In short, dot diagrams allow us to record the data going into this many-to-one form of composition. Incidentally, we're going to use an outlandish convention for composition in diagrams: $f \circ g$ will refer to the composite made by doing f first and

then doing g .

Just as individual morphisms have source and target objects, we may think of a dot diagram as having both a source and a target. If the target of a dot diagram matches the source of another dot diagram, then we may form a composite diagram by gluing.

$$\underline{X \overset{f}{\bullet} Y \overset{g}{\bullet} Z} \circ \underline{Z \overset{j}{\bullet} X \overset{f}{\bullet} Y} = \underline{X \overset{f}{\bullet} Y \overset{g}{\bullet} Z \overset{j}{\bullet} X \overset{f}{\bullet} Y}$$

So not only do dot diagrams encode compositions in a category, they themselves may be composed! Thus for a given category C , there is a category $\text{dd}(C)$ of dot diagrams labeled in C . The category $\text{dd}(C)$ has the same objects as C but the morphisms are dot diagrams labeled in C . We won't care about the absolute position of dots, only their ordering.

We will, however, distinguish between a dot diagram with two dots labeled by f and g and the dot diagram for the composite morphism.

$$\underline{X \overset{f}{\bullet} Y \overset{g}{\bullet} Z} \neq \underline{X \overset{fg}{\bullet} Z}$$

(as diagrams)

Of course, these diagrams evaluate to the same morphism, but as formal objects, they are distinct. Similarly, one must be careful when defining the unit for composition of dot diagrams. It is tempting to think that the identity dot diagram for an object Y is the dot diagram with a dot labeled by 1_Y . However, according to the previous point, composing with this dot diagram gives a new, distinct diagram. Instead, the identity dot diagram is just a diagram consisting of a single region labeled by Y . Indeed, gluing this diagram with any other has essentially no effect.

$$\underline{X \overset{f}{\bullet} Y} \circ \underline{Y \overset{1_Y}{\bullet} Y} = \underline{X \overset{f}{\bullet} Y \overset{1_Y}{\bullet} Y}$$

$$\underline{X \overset{f}{\bullet} Y} \circ \underline{Y} = \underline{X \overset{f}{\bullet} Y}$$

So far, we've been thinking of a category of dot diagrams labeled in a fixed category C . One might guess that the resulting category $\text{dd}(C)$ is a fattened, but equivalent, category. After all, any dot diagram labeled in C may be evaluated to give a morphism in C , and this gives us a canonical functor $\text{dd}(C) \rightarrow C$. This idea falls apart, though, when we look for an inverse functor $C \rightarrow \text{dd}(C)$. The obvious candidate is the inclusion map which takes each morphism to a diagram with a single dot labeled by that morphism. However, this map is not a functor! Indeed, this map only hits dot diagrams with a single dot, but composing such gives diagrams with multiple dots.

So if the inclusion is not a functor, what sort of map is it? It preserves domains and codomains, but it does not satisfy the functoriality relations. With this motivation, we define a 1-computad to be a structure which records domains and codomains just like in a category, but doesn't record, and hence won't demand respect for, any compositions or units. We'll call a 1-computad's objects *degree 0 generators* and its (non-composable) morphisms *degree 1 generators*. A 1-computad can be thought of as a directed graph in which the 0-generators are vertices and the 1-generators are arrows. Just as a graph is given the special name, *quiver*, when we expect it to generate a path algebra, we call a graph a *1-computad* when we expect it to generate a (1-)category.

While we were initially inclined to label dot diagrams with objects and morphisms from a fixed category, C , we don't actually need the composition that C afforded us to define dot diagrams. In fact, we may label dot diagrams with generators from any 1-computad, G . And while we cannot compose the generators, we may glue the dot diagrams labeled by generators, and so $\text{dd}(G)$ forms a category. Of course, we cannot evaluate the dot diagrams labeled by G because we don't know how to compose 1-generators. However, while moving to 1-computads loses evaluation, we gain the fact that the inclusion $G \rightarrow \text{dd}(G)$ is a map of 1-computads.

To summarize, any 1-computad G gives a category $\text{dd}(G)$ of dot diagrams, and conversely, given any category C , we may forget its composition and regard it as a computad, $\text{forget}(C)$. These constructions give us functors

$$\begin{aligned} \text{dd}: 1\text{-Cmp} &\longrightarrow \text{Cat} \\ \text{forget}: \text{Cat} &\longrightarrow 1\text{-Cmp} \end{aligned}$$

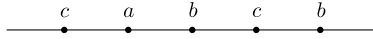
These functors are not inverse, but instead, they are adjoint. That is, to define a functor $\text{dd}(G) \rightarrow C$, it is enough to say where each of the generating objects and morphisms go in C . This amounts to defining a map of 1-computads $G \rightarrow \text{forget}(C)$. So the dot diagrams functor dd gives the the free category on a 1-computad. The counit and unit

$$\begin{aligned} \text{dd}(\text{forget}(C)) &\longrightarrow C && (\text{in Cat}) \\ G &\longrightarrow \text{forget}(\text{dd}(G)) && (\text{in 1-Cmp}) \end{aligned}$$

of the adjunction are the evaluation and inclusion maps.

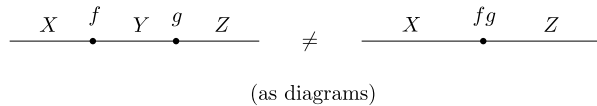
Let's analyze this adjunction in the special case of a 1-computad G with a single 0-generator, X , and three 1-generators, a, b, c . Because there is only one 0-generator, the source and target of each 1-generator must be X . Every dot diagram consists of dots labeled

by a, b, c and regions labeled by X . Actually, there is no need to label the regions in this case, and so we won't.



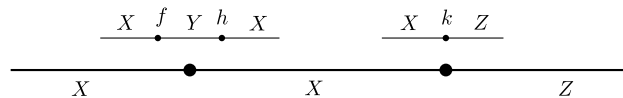
We might write the previous example more concisely as a word, $cabc b$. It is clear that dot diagrams are in one-to-one correspondence with words in the alphabet, $G = \{a, b, c\}$. Moreover, gluing dot diagrams corresponds to concatenating words and the blank dot diagram corresponds to the empty word. Thus, we see that $\text{dd}(G)$ is the free monoid on the alphabet set G . For monoids, the inclusion map tells us to think of each letter as a word of length one. The counit tells us how to evaluate a word if the letters are elements of a monoid. If, on the other hand, we regard the letters in a word $cabc b$ as mere formal symbols from an alphabet set, we lose our ability to evaluate words. Indeed, we need to provide an *interpretation* for the letters $\{a, b, c\} \rightarrow \text{forget}(M)$ in order to evaluate $cabc b$ in some monoid M .

Let's revisit our dubiously distinct dot diagrams with our newly gained appreciation of 1-computads.



This example shows that we shouldn't expect to have unique representation of composites when we take our labels in a category. Notice that in order to write down the second diagram, we need to be able to compose f and g . This is fine if f and g are morphisms in a category, but we are not allowed to compose 1-generators in a 1-computad. This suggests the correct uniqueness statement: dot diagrams uniquely represent composites for a 1-computad. Distinct dot diagrams may evaluate to the same morphism under *some* interpretation in *some* category C . However, if two dot diagrams agree on evaluations for *every* interpretation in *every* category, then those diagrams must be the same.

We can say a bit more about the dd/forget adjunction by using a clever trick. Let's consider dot diagrams whose dots are labeled by dot diagrams.



Assuming f, h , and k are morphisms in some category C , we may evaluate this diagram in two ways. On the one hand, we could evaluate each of the labels to produce a dot diagram labeled in C , which itself can be evaluated. On the other, we could insert the inner

dot diagrams into the outer to produce a dot diagram labeled in C , which we could then evaluate. Unsurprisingly, these two approaches will always give the same result.

The previous observation may be thought of as a fancy way to describe the associative law. Roughly speaking, it says that we may group parts of a dot diagram and evaluate those parts early without changing the ultimate evaluation. Thus, the diagram

$$\begin{array}{c} X \quad f \quad Y \quad g \quad Z \\ \hline X \quad \bullet \quad Z \end{array}$$

explains why the previously discussed formally distinct diagrams must evaluate to the same morphism.

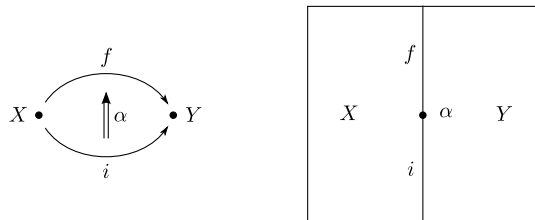
Chapter 3

Intuition for String Diagrams

In this chapter we analyze the diagrams, called *string diagrams*, that we'll use to describe composites in a 2-category. In a 2-category, we may compose 2-morphisms over both 0-morphisms and 1-morphisms. These, following tradition, are called *horizontal* and *vertical* composition. We require these compositions to satisfy an interchange law which, roughly speaking, says that horizontal and vertical compositions commute. It's not a surprise, then, that a diagram capturing compositions of 2-morphisms should be 2-dimensional. This extra dimension makes the diagrams much more interesting but also requires a fair amount of care. We're going to give a freeness result for string diagrams very much like the one for dot diagrams. But there's plenty new to say about string diagrams along the way.

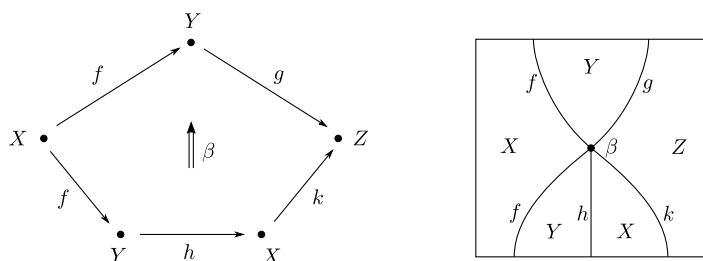
3.1 2-Morphisms as String Diagrams

String diagrams are meant to capture the possible compositions of 2-morphisms in a 2-category. Before we think about all the ways that we might compose, let's first see how we can draw a single 2-morphism as a string diagram. Consider a 2-morphism $\alpha: i \rightarrow f$ going between 1-morphisms $i, f: X \rightarrow Y$. We can draw α as an arrow diagram or as a string diagram as follows.



Unlike an arrow diagram, it is important that we draw a string diagram in a square. Our 2-morphism, α , is drawn as a node between two strings. The string below α is labeled with α 's domain and the string above has α 's codomain. We label the regions to the left and right of α with the objects X and Y . So just as was the case for dot diagrams, a string diagram is the geometric dualization of an arrow diagram. We will follow the conventions of reading a string diagram left-to-right and bottom-to-top to be consistent with standard conventions for coordinate axes.

The previous string diagram of a 2-morphism, while not incorrect, is actually overly simple in an important way. When we actually do mathematics, we'll often want to think of a 2-morphism not just going from a 1-morphism to a 1-morphism, but rather going from a composite of 1-morphisms to a composite of 1-morphisms. String diagrams are very well suited to capturing this idea. Let's see such an example of an arrow diagram and the corresponding string diagram.



Once again, we draw our 2-morphism as a node where strings meet. In this example, the node β has three strings approaching from below and two approaching from above. We read the strings below β to see that its source is the composite of 1-morphisms $f \circ h \circ k$. Similarly, β 's target is $f \circ g$. The extra strings in this diagram determine extra regions, as well. We use these regions to record the various objects being composed over.

3.2 Degrees and Dimensions

In an arrow diagram, the dimension of the cell used to draw an arrow equals the degree of that morphism. It's therefore easy to conflate these two ideas and, say, refer to the dimension of a morphism. As we've seen, in dot and string diagrams the degree of a morphism does not usually match the dimension of the stratum it labels. Thus we must be much more careful with these terms. In this section, we'll write down a couple rules to help us keep degrees and dimensions straight.

In addition to degree and dimension, we'll be using the ideas of codegree and codi-

mension. The complement for degree is meant to be taken in the categorical world and the dimensional complement is to be taken in the geometric world. That is,

$$\begin{aligned} \text{codegree} &= \text{category number} - \text{degree of morphism} \\ \text{codimension} &= \text{dimension of diagram} - \text{dimension of stratum} \end{aligned}$$

Both dot and string diagrams have been defined as geometric duals of arrow diagrams. The same will be true when we consider surface diagrams. Thus, we have the following rule for degrees and dimensions.

Rule. Suppose a morphism labels a stratum in a dot, string, or surface diagram. Then both of the following statements hold.

$$\begin{aligned} \text{degree of the morphism} &= \text{codimension of the stratum} \\ \text{codegree of the morphism} &= \text{dimension of the stratum} \end{aligned}$$

This rule tells us how to calculate degrees of morphisms appearing in a diagram. It does not, however, tell us the degree of the morphism obtained by evaluating a diagram. This important, if subtle, distinction was already present for dot diagrams. In particular, consider the example of a dot diagram consisting of a single region labeled by X . In this diagram, we just have a (degree 0) object, X , but the diagram evaluates to a (degree 1) morphism, 1_X .

$$\text{eval}\left(\begin{array}{c} \text{---} X \text{---} \\ \text{---} \end{array}\right) = 1_X$$

In a 2-category, we need to be able to describe composites of 1-morphisms as well as composites of 2-morphisms. Dot diagrams work perfectly well to describe composites of 1-morphisms, and so we'll use these. In general, we have the following rule about evaluation.

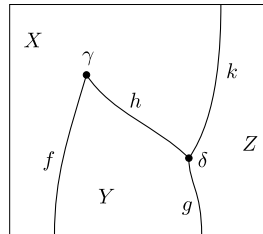
Rule. Suppose a dot, string, or surface diagram D evaluates to a morphism $\text{eval}(D)$. Then

$$\text{deg}(\text{eval}(D)) = \text{the dimension of the cube that } D \text{ is drawn in.}$$

In other words, a dot diagram is drawn in a 1-cube, an interval, and so evaluates to a 1-morphism. A string diagram is drawn in a 2-cube, a square, and so evaluates to a 2-morphism. And a surface diagram will be drawn in a (3-)cube and so will evaluate to a 3-morphism. Notice that dot, string, and surface diagrams are named after the shape of their codimension 1 strata. For example, a string diagram consists of (codimension 1) strings living in a 2-D square. This naming convention will make somewhat more sense once we have considered these diagrams for monoidal categories.

3.3 Slicing and Projecting

In this section, we'll study two fundamental operations that we can perform on string diagrams: slicing and projecting. We'll add to our degree and dimension rules from last section with rules for each of these operations. We'll begin, though, by thinking about the source and targets of string diagrams. Consider the following example of a string diagram with two nodes.



Near the bottom of this diagram, we see the strings f and g . Near the top, there is a single string, k . It's tempting to think that this diagram has the composite $f \circ g$ as its source and the 1-morphism k as its target. It turns out to be better, though, not to regard the source and target as 1-morphisms, but instead as dot diagrams. The string diagram induces the structure of a dot diagram on the square's bottom edge. We declare this dot diagram as the string diagram's source.

$$\begin{array}{c} X \quad f \quad Y \quad g \quad Z \\ \hline \bullet \quad \quad \quad \bullet \end{array}$$

Of course, this dot diagram evaluates to $f \circ g$, but it also contains more information. That is, it tells us how the composite was made, not just the resulting 1-morphism. Similarly, we regard the top edge of a string diagram's square as its target dot diagram.

More generally, we may intersect a string diagram with a horizontal line at most any height to get a dot diagram. This process will be referred to as *slicing* the string diagram. We must only avoid slicing at heights containing a node. The previous example of a string diagram essentially only has three slices.

$$\begin{array}{c} X \quad \quad \quad k \quad Z \\ \hline \bullet \end{array}$$

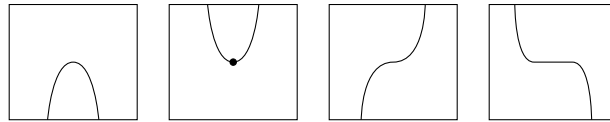
$$\begin{array}{c} X \quad f \quad Y \quad h \quad X \quad k \quad Z \\ \hline \bullet \quad \bullet \quad \bullet \end{array}$$

$$\begin{array}{c} X \quad f \quad \quad Y \quad g \quad Z \\ \hline \bullet \quad \quad \quad \bullet \end{array}$$

By reinterpreting height as time, we may regard a string diagram as a movie of dot diagrams that begins with the source and ends with the target dot diagram. Away from the instant

that a 2-morphism node takes place, the movie shows a dot diagram evolving. That is, dots wiggle left and right but maintain their overall order. Around the time of a node, several adjacent 1-morphism dots fuse together and are replaced by several new dots dispersing.

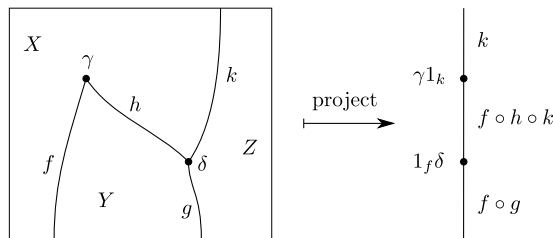
Motivated by the movie interpretation of a string diagram, we will insist that strings never travel horizontally in a string diagram. We call this the *progressivity* rule for strings. In particular, strings are not allowed to “turn back” in the shape of a cup or cap. We will insist on progressivity even as strings approach a node. So all of the following strings are not considered progressive and so won’t be allowed as string diagrams.



As promised, we have a rule for slice degrees and dimensions. We must be careful when counting these in the slice of a string diagram. Since we don’t slice at any height containing a node, only strings and regions appear in a slice. Slicing reduces the dimension of these strata by one. This is better understood working in codimension.

Rule. Slicing preserves codimension and hence preserves degree.

The horizontal and vertical directions play rather different roles in a string diagram and, as such, it does not make sense to take vertical slices. There is, however, an analogous vertical operation: *projection*. Let’s see the projection of our example string diagram.

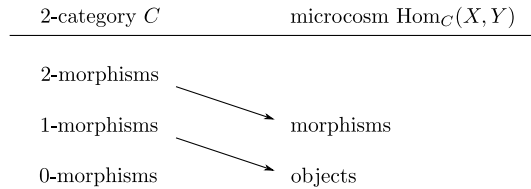


We may think of projection as flattening the string diagram by removing its width. Of course, we normally draw dot diagrams horizontally, but it makes much more sense to draw a projection as a vertical dot diagram. Notice that in a string diagram, the regions are the only strata to have any horizontal thickness. We are effectively removing these strata when building the projection. As the lower dimensional strata get squashed together, their labels become composites. Much like slicing, projection preserves certain statistics of diagrams.

Rule. Projection preserves dimension and hence preserves codegree.

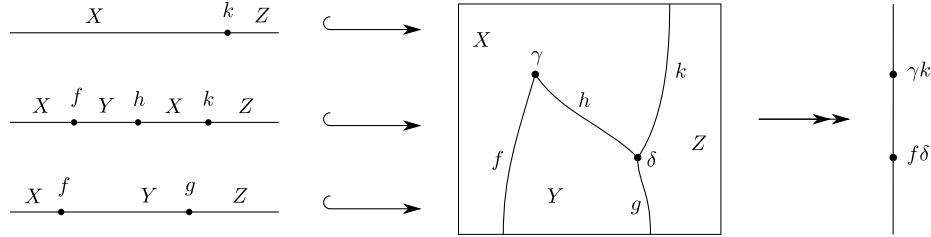
What does it mean for projection to preserve the codegree of morphisms? Let's first observe that projection does not respect codimension. In particular, the codimension 2 strata (labeled by γ or δ) become codimension 1 in the projection, and the codimension 1 strata (f, g, h, k) become codimension 0. Since we expect degree to equal codimension, it would make sense that degree is not preserved either. So why don't we consider γ or δ degree 2 in the projection?

In order to answer these questions, we need the idea of a *microcosm category*. For any two objects, X and Y , in a 2-category C , the collection of 1-morphisms $\text{Hom}_C(X, Y)$ has more structure than just being a set. Indeed, we can think of $\text{Hom}_C(X, Y)$ as a category, the microcosm, whose objects are the 1-morphisms of C from X to Y and whose morphisms are the 2-morphisms of C between those 1-morphisms.



Notice that moving to a microcosm ignores the 0-morphisms of C and shifts the degree of the other morphisms down by one. This is entirely analogous to what happens to codimension when we project. Thus we'll think of a projection as a dot diagram for a microcosm category. The microcosm category is determined by the left-most and right-most objects from the string diagram. In our example string diagram, the projection describes a composite of morphisms in $\text{Hom}_C(X, Z)$. In this microcosm, we'll regard γ and δ as degree 1 morphisms. So projection does not preserve the degree of these morphisms, but their codegree, 0, is preserved!

We'll end this section by observing an important connection between slicing and projection. In our description of projections, the projection dot diagram was labeled with composites of morphisms. While this works if our label morphisms are taken from a 2-category, it doesn't make as much sense if our labels are just formal symbols. Labels on dots in the projection are a composite of both 1-morphisms and 2-morphisms from C . Notice, however, that the regions in the projection dot diagram are made exclusively from C 's 1-morphisms. We may, thus, replace these labels with dot diagrams.



This picture suggests we might use the language of fiber bundles to describe slicing and projection. Over any region in the projection, the fiber is a slice dot diagram, unique up to evolution. Unlike in a fiber bundle, the fiber is allowed to change in a string diagram, but only at heights in the string diagram containing nodes. We'll sometimes ask that nodes appear at distinct heights in the string diagram, moving them slightly if necessary. In this case, each node in the string diagram corresponds to a dot in the projection. Moving past a dot downstairs, then, changes the fiber upstairs in a way controlled by the corresponding node. We may thus think of a string diagram as being a dot diagram's worth of dot diagrams.

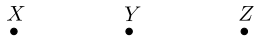
3.4 2-Computads and the Free 2-Category

In this section, we'll sketch a definition of the 2-category $\text{sd}(G)$ for a 2-computad G . A *2-computad* is a collection of generating morphisms of degrees 0, 1, and 2. In some sense, $\text{sd}(G)$ is the minimal 2-category containing the generators of G . We build $\text{sd}(G)$ by including the fewest possible objects, 1-morphisms, 2-morphisms, *and* relations. Much like we did for dot diagrams, we'll characterize $\text{sd}(G)$ as the free 2-category by seeing sd as the left adjoint to a forgetful functor.

A 2-computad is a significantly more complicated structure than its one dimensional counterpart. In a 1-computad, 1-generators have 0-generators for their source and target. This is also true for 1-generators in a 2-computad, but we must be careful for 2-generators. It would be overly limiting to demand that 2-generators have 1-generators for source and target. In a diagram, this would mean that every node had exactly one string coming from below and one string from above. Instead, we'll allow 2-generators to have source and targets which are built up from the 1-generators. That is, the source and target of a 2-generator are dot diagrams labeled in the lower degree generators.

In the sections on dot diagrams and string diagrams, I've been consistent in my usage of which labels $X, Y, f, g, \alpha, \beta$ are the source and targets of other labels. In other words, I've been drawing all diagrams by using the following 2-computad.

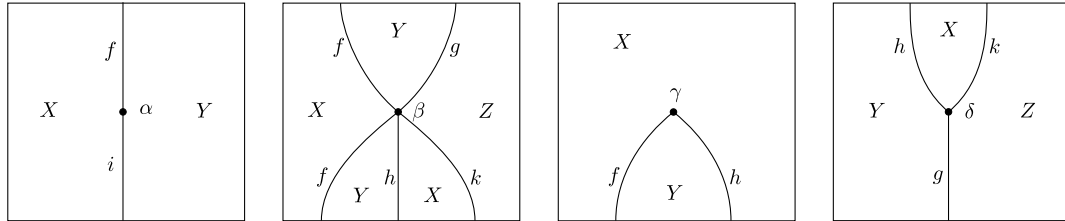
0-generators:



1-generators:

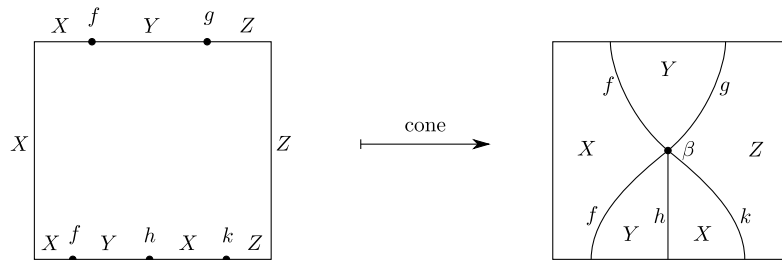


2-generators:



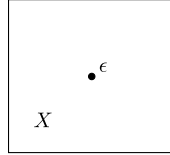
Notice that we draw degree 1 generators in 1-dimensional cubes and degree 2 generators in 2-cubes. Each direction of a cube will give a direction of composition. Thus, it makes sense to draw 0-generators, which aren't composable, as 0-cubes.

Because we allow 2-generators to have dot diagrams for source and target, there is much more variety in the shapes we draw for them. What then are the allowable shapes for 2-generators? Let's consider the 2-generator β . Beta has a source dot diagram for $f \circ h \circ k$ and a target dot diagram for $f \circ g$. These dot diagrams, themselves, must agree on source object (X) and target object (Z). We may label the sides of a square with this information. The string diagram for β is then made by *coning* the boundary of this square to a central node which we label as β .

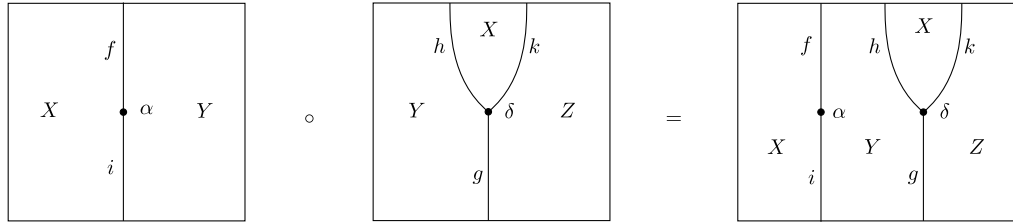


Thus, the shapes used for 2-generators are just cones on the boundary of a square. We place dot diagrams on the top and bottom sides and make sure their left-most objects agree and their right-most objects agree. The 2-generators α and δ are drawn using cones very much like β . The 2-generator γ looks a bit different because its target dot diagram has no dots. This is perfectly fine and we just get no strings from above in the γ 's cone. We may think of γ as going from the composition $f \circ h$ to X , or if we really prefer 1-morphisms, from the

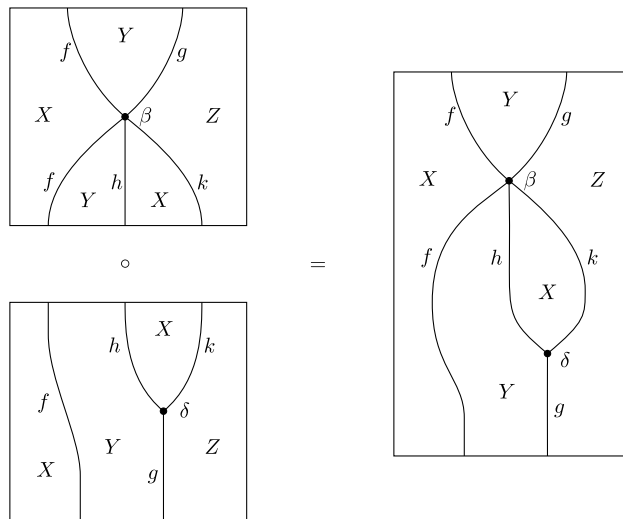
composite $f \circ h$ to 1_X . It also is allowable to have a 2-generator in which both the source and target dot diagrams have no dots. In this case the 2-generator resulting from coning is just a “free floating” node in a single region.



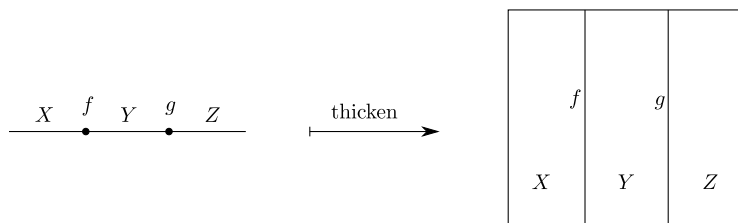
For a fixed 2-computad G of generators, we get string diagrams labeled in G by stringing together combinations of 2-generators. We’ll understand this process in terms of gluing. To get a 2-category of string diagrams, we’ll need to be able to glue diagrams both horizontally and vertically. In order to glue two diagrams horizontally, we just need to check that they agree on an intermediate object. We get the composite by placing the string diagrams side by side. Gluing string diagrams over an object Y also has the effect of gluing their sources over Y and gluing their targets over Y .



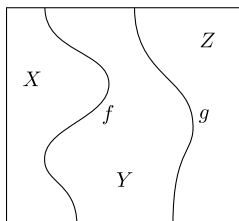
When we compose two string diagrams vertically, we need to check that they agree on an intermediate dot diagram. If they do, we glue the string diagrams along that dot diagram.



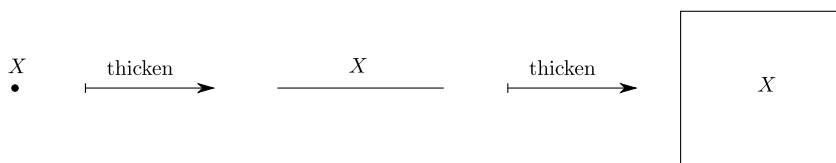
What do our units for vertical composition look like? Given any dot diagram, we should be able to produce a string diagram so that vertically composing with that string diagram has no effect. We get such a string diagram by thickening a dot diagram in height.



As a movie, this string diagram shows the dots f and g staying put. In fact, any string diagram which has no nodes can be thought of as a movie in which very little happens. Indeed, the dots in the movie must maintain their order. We'll call such a node-less string diagram an *evolution of dot diagrams*.



How do we get units for horizontal composition? Well, any object can be regarded as a dot diagram without any dots. By the thickening this vertically, we get the string diagram for the identity on the identity. In sum, we can regard this as thickening a 0-cube both horizontally and vertically.

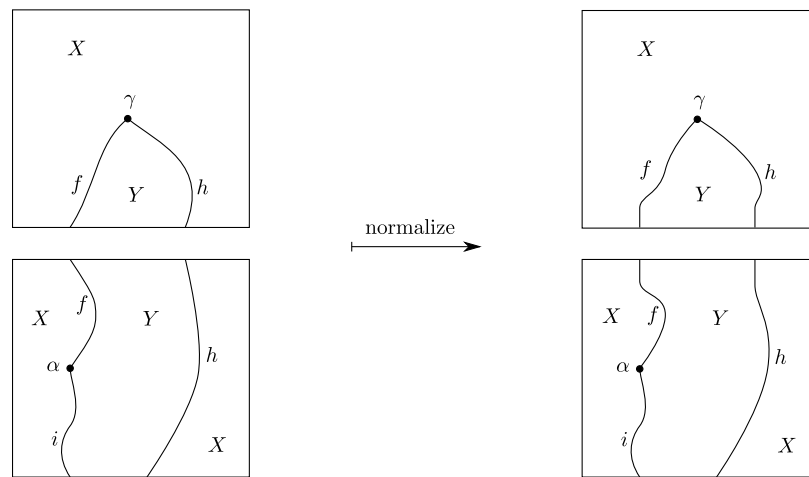


Let us summarize our various compositions and units by making a definition. For any 2-computad G , there is a 2-category $\text{sd}(G)$ of *string diagrams labeled in G* , defined roughly as follows.

- A 0-morphism of $\text{sd}(G)$ is a 0-cube labeled by a degree 0 generator of G .
- A 1-morphism of $\text{sd}(G)$ is a dot diagram labeled by degree 0 and 1 generators of G .
- A 2-morphism of $\text{sd}(G)$ is a string diagram labeled by G .

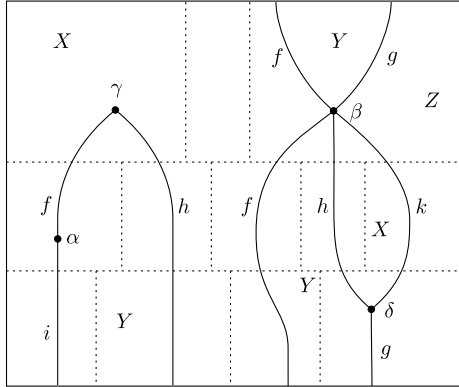
We'll consider two dot diagrams to be equal in $\text{sd}(G)$ if they are related by some evolution. We'll similarly define an evolution of strings diagrams to be a movie of string diagrams in which the strings and nodes may move around over time, but no strata are fundamentally changed. String diagrams in $\text{sd}(G)$ are then also only taken up to string evolution.

Composition in $\text{sd}(G)$ is done by the horizontal and vertical gluing we've outlined so far. Working smoothly and up to evolutions, there are some technical issues to sort out. For example, two string diagrams may agree on an intermediate dot diagram but the strings don't meet up in smooth manner. We'll show any string diagram may be straightened using a string evolution so that the strings meet the top and bottom faces in a perpendicular manner.



We might also wish to glue together string diagrams whose dot diagrams only agree up to a dot evolution. In this case, we may regard the dot evolution as a string diagram in between the other string diagrams. Now, the string diagrams agree on intermediate dot diagrams, and after normalizing, we may achieve the desired gluing.

We may build quite large complex string diagrams by gluing together smaller diagrams. Conversely, any large string diagram may be decomposed by splitting it into simple pieces. Any string diagram may be given a *brick decomposition*.



In such a decomposition, each brick is a string diagram with a simple stratification. If the string diagram is labeled in a 2-category, we may evaluate the diagram by evaluating each brick. We think of the diagram as a vertical composite of horizontal composites. Brick decompositions have the nice property that any two have a common refinement made by intersecting the bricks. This will allow us to see that evaluation is actually independent of the brick decomposition used.

We use evaluation to build an adjunction, much as we had for dot diagrams. Indeed, evaluation of diagrams gives us a counit

$$\text{sd}(\text{forget}(C)) \longrightarrow C,$$

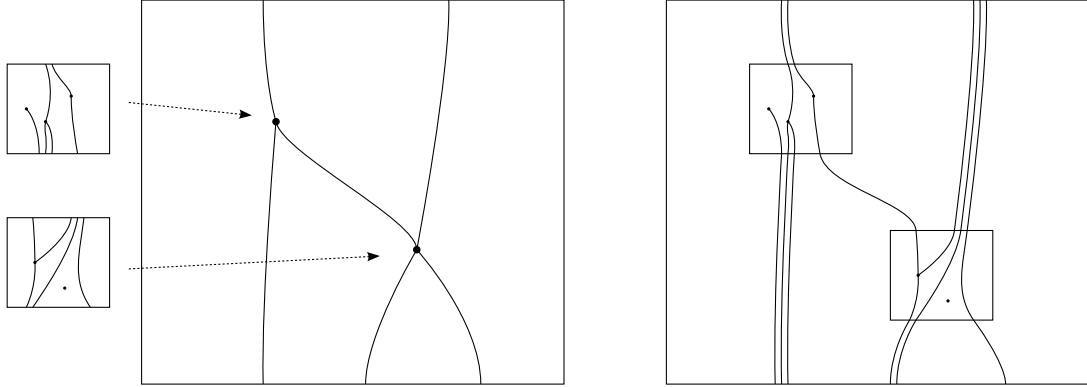
and inclusion of generators as singleton diagrams gives a unit

$$G \longrightarrow \text{forget}(\text{sd}(G)).$$

These give us our main theorem for string diagrams.

Theorem. Suppose G is a 2-computad. Then the 2-category of string diagrams $\text{sd}(G)$ is the free 2-category on the generators in G .

We were able to express a bracketing rule for dot diagrams by labeling the dots of a dot diagrams with dot diagrams. We may similarly take a string diagram and label its strings with dot diagrams and its nodes with string diagrams. Rather than use labels explicitly, it's easier to replace the nodes with the string diagrams that should label them. We'll also replace the strings labeled by dot diagrams with multi-strand strings. We call the result a *rope diagram*.



A general associative law for 2-morphisms then would state that evaluating inner string diagrams and ropes first has no effect on what the diagram ultimately evaluates to.

3.5 Monoidal Categories

Just as dot diagrams with unlabeled regions allow us to write words in monoids, string diagrams with unlabeled regions will be good for describing compositions in a familiar mathematical object: monoidal categories. Roughly speaking, a *monoidal category* is a category with an extra operation \otimes , called *tensor*. For any two objects X and Y , there is an object $X \otimes Y$, their tensor product, picked out. We may think of this as a classical tensor product or just as some abstract operation. We ask that \otimes be associative and that there be an object which functions as an identity for the tensor product. Further, given any morphisms $f: W \rightarrow X$ and $g: Y \rightarrow Z$, we may tensor to get a morphism $f \otimes g: W \otimes Y \rightarrow X \otimes Z$. More precisely, what we are describing here is actually a strict monoidal category. We will later discuss a more general structure, and hence more commonly found in nature, the weak monoidal category.

Any strict monoidal category C defines a 2-category C^\uparrow as follows. There is only one object, $*$, in C^\uparrow . Every object of C is a 1-morphism in C^\uparrow , thought of as an endomorphism of $*$. Every morphism of C is regarded as a 2-morphism of C^\uparrow . The horizontal composition for C^\uparrow is given by C 's tensor product, \otimes , and vertical composition is taken to be C 's usual composition of morphisms. By moving from a monoidal category to a 2-category, we view morphisms as being one degree higher. Going the other way, any 2-category D with a single object, $*$, may be thought of as a monoidal category D^\downarrow by ignoring the object and viewing morphisms as one degree lower. More generally, for any 2-category D with a chosen favorite object $*$, we may define D^\downarrow as the microcosm category $\text{Hom}_D(*, *)$. We get the tensor product for D^\downarrow by taking the tensor product to be D 's horizontal composition.

What does this mean for diagrams in a monoidal category C ? We should be able to describe compositions in C by using dot and string diagrams for C^\uparrow . Spelling this out, we'll have dot diagrams which describe the tensor products of objects of C . There is no need for labeling the regions of these dot diagrams because they all would read $*$. A string diagram will have strings labeled by objects of C and nodes labeled by morphisms of C . Regions do not need to be labeled. Notice that by not labeling regions, there is nothing to be checked for horizontal composition. Indeed, any two objects or any two morphisms may be tensored in a monoidal category.

There are variants of monoidal categories whose compositions are better captured not by strings in a 2-D square, but by strings in higher dimensional space. Compositions in a braided monoidal category may be described using strings in 3-space. There is much more room for strings to pass each other in 3-space than 2, but we can account for this by allowing braidings. If we consider strings in 4-space, we are looking at compositions for a symmetric monoidal category. Here, there's so much room that "over" and "under" crossing information is no longer meaningful. Also, there is no need to move to any higher dimensional ambient space — all tangling of strings has already disappeared by dimension four.

From this perspective, the plain (non-braided) monoidal category is quite special. Unlike in 3 or 4-space, strings separate regions in the square. If this dissertation were focused on high-dimensional ambient spaces and, say, symmetric monoidal categories, then the term "string diagram" would probably be used refer to a 4-cube with strata of dimension at most 1. Instead, we will be interested in the case in which the cube has many regions, separated by strata. As such, we use "string diagram" to refer to stratified 2-cubes. Similarly, "dot diagrams" are named by their codimension 1 strata, dots. Even if a stratified 2-cube has no 1-strata, we'll still consider it a string diagram, albeit a degenerate one. This point is especially confusing when we consider a stratified 3-cube without 2-strata. While one might be tempted to call this a "string diagram," we will consider it a degenerate surface diagram.

In practice, we don't actually find many examples of strict monoidal categories. Consider the category of vector spaces over \mathbb{C} and linear maps. This nearly forms a monoidal category, except the tensor product of vector spaces is not strictly associative. That is,

$$(U \otimes V) \otimes W \quad \text{and} \quad U \otimes (V \otimes W)$$

are not equal "on the nose," but rather there is a canonical isomorphism which describes "how they are the same." We call this invertible map

$$(U \otimes V) \otimes W \longrightarrow U \otimes (V \otimes W)$$

the *associator* for U, V , and W . Similarly, the field \mathbb{C} , thought of as a vector space over itself, functions as a unit for the tensor product. But once again, we don't have an equality, but rather just a canonical isomorphism

$$V \otimes \mathbb{C} \longrightarrow V$$

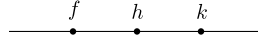
which we'll call the *right unitor* for V . A *weak monoidal category*, then, is a category with a monoidal product \otimes which is associative and unital in this weaker sense. Further, we don't just demand that these data exist, but we demand that the category come equipped with associators and unitors for each choice of objects. We'll also demand that this data satisfy certain *coherence laws* which make sure we don't get into trouble when making calculations.

In many ways, the weak monoidal category is the more natural definition. We do not, as a general principle, want to require equalities when isomorphisms will do. After all, an isomorphism is a more structured statement that tells how two things are the same. On the other hand, strict monoidal categories have much nice formal properties. Indeed, we can just “do algebra” with tensor products of objects in the strict realm. That is, the \otimes is really an honest multiplication making the object set into a monoid. It turns out that every weak monoidal category is equivalent, in some precise sense, to a strict monoidal category. The proof of this fact constructs a 2-category from a bicategory by using the Yoneda embedding into a large presheaf 2-category. So at least in principle, we can replace any weak monoidal category that we find in nature with a strict one. Or maybe more to the point, we may work with a weak monoidal category as if it were strict.

The string diagrams we've considered so far are used to describe compositions in a strict monoidal category. The previous discussion, though, suggests we should be able to modify our theory of string diagrams in order to encode compositions for a weak monoidal category. More generally, a bicategory is the weak analog of a 2-category. By studying string diagrams for bicategories, then, we'll actually be including the case of weak monoidal categories. Indeed, any weak monoidal category may be thought of as a bicategory with a single object. Therefore a diagram for a weak monoidal category will just be a string diagram for a bicategory in which we have unlabeled regions.

3.6 Bicategories

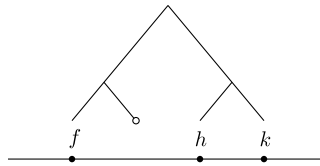
In order to evaluate a dot diagram as a 1-morphism in a bicategory, we need to record more information than we did for a 2-category. For example, the 3-fold composite of 1-morphisms is no longer uniquely defined.



Indeed, we might choose to evaluate this as $(fh)k$ or as $f(hk)$. While the results are guaranteed to be isomorphic, they are not equal. This suggests we might need a binary tree in order to decide such orders of compositions.



In fact, there are other ways we could choose to evaluate this diagram. We are also allowed to take as many units as we like from any of the regions. For example, we would evaluate the following dot diagram



as $(f \circ 1_Y) \circ (h \circ k)$. By decorating our dot diagrams in this manner, we may uniquely evaluate such a diagram in a bicategory.

While we may think of a decoration as a list of instructions on how to disassemble a dot diagram, we may also regard it as giving a record of how the diagram was built up. We think of decorated dot diagrams as being built from two basic types of building block:

- The dot diagram with a single region and decorated with a single region leaf.
- The dot diagram with a single dot and decorated with a single dot leaf.

When we glue together two decorated dot diagrams, we get a decoration for the composite by joining the decorations for the factors. We will therefore call such a decoration a *history*. Notice that composition of decorated dot diagrams cannot be associative. Indeed, having a history prevents the two candidates from being equal. Similarly, we'll require every history to have at least one leaf, and so it is impossible for any decorated dot diagram to be an identity.

So if a history allows us to evaluate dot diagrams in a bicategory, what do we need to evaluate a string diagram? Certainly, the source and target dot diagrams should be equipped with histories. One might think that intermediate slices should also be equipped with histories so that we have a preferred way to evaluate them, as well. In fact, the

coherence relations in a bicategory ensure that we don't need histories on intermediate slices. We therefore define a decorated string diagram to be one that has histories just for its source and target dot diagrams.

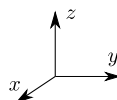
Chapter 4

Intuition for Surface Diagrams

Surface diagrams are a tool used to describe composites of 3-morphisms between composites of 2-morphisms. We begin by interpreting surface diagrams in the strict world of 3-categories. It's relatively easy to write down a theory of such diagrams. Following the example of 2-categories and bicategories, one might expect that we would get a theory of surface diagrams for tricategories by putting decorations on the surface diagrams we made for 3-categories. In fact, this does not work. Indeed, while every bicategory is (bi-)equivalent to a 2-category, the analogous result does not hold for tricategories. We will thus attempt to define a more sophisticated theory of surface diagrams for tricategories, but this proves to be too arduous. In the end, we settle on a theory of surface diagrams for Gray-categories. The definition of a Gray-category is a compromise between the generality of tricategories and the strictness of 3-categories. We'll see that the topology of surface diagrams, then, helps explain why this compromise is a good one.

In order to speak about surface diagrams, we'll need to work in three dimensions. For the categorical constructions we'll want, the following conventions work well:

In \mathbb{R}^3 , the x -coordinate will go from back to front, the y -coordinate will go from left to right, and the z -coordinate will go from bottom to top.



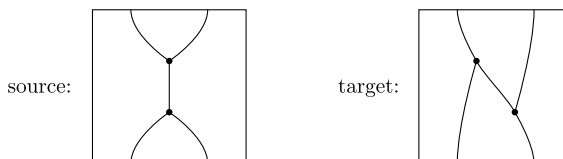
Also, we'll allow ourselves to omit labels for ease of reading.

4.1 3-Categories

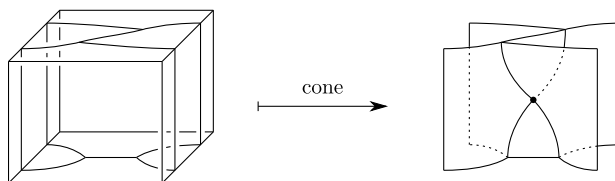
In many respects, surface diagrams are a straightforward generalization of string diagrams, one dimension up. We may perform many of the same operations: coning, thickening, gluing, slicing, projecting, etc. We'll take a tour of these operations in order to get a feel for the diagrams. We draw a surface diagram as a stratification of the cube. As usual, we label the strata with morphisms according to codimension.

- We label the 3-strata, called *regions*, with 0-morphisms.
- We label the 2-strata, called *walls*, with 1-morphisms.
- We label the 1-strata, called *seams*, with 2-morphisms.
- We label the 0-strata, called *nodes*, with 3-morphisms.

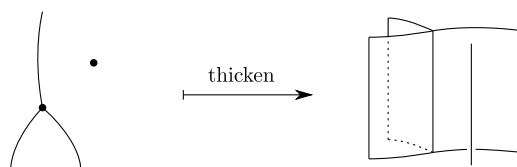
We get basic examples of surface diagrams by coning. Suppose we have a 3-morphism Γ which we want to display as a surface diagram. First, we must choose representations for its source and target 2-morphisms as string diagrams. These string diagrams must agree on source dot diagrams and on target dot diagrams.



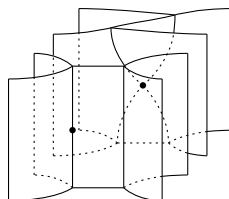
We place these string diagrams on the bottom and top faces of a cube, paying careful attention to our direction conventions. We draw dot evolutions on the left and right sides of the cube to connect up the corresponding strings. And we place a single region on the back face of the cube and a single region on the front face. Coning from the faces to a center point gives a surface diagram. We may think of Γ as labeling the node at the center.



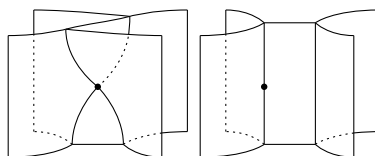
We also get examples of surface diagrams by thickening any string diagram in the z -direction. Note that while the x and y -directions are forward and right in a surface diagram, we'll usually continue to draw them as right and up, respectively, in a string diagram.



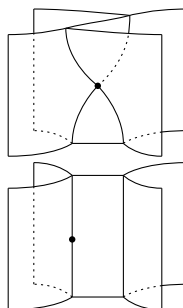
In a surface diagram, we only allow regions to intersect the front and back faces of the cube. If the object labeling the front region of one surface diagram agrees with the object labeling the back region of another, then we may glue these diagrams in the x -direction.



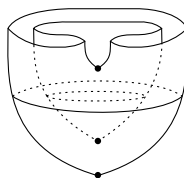
On the left and right faces of a surface diagram, we only allow dot evolutions. Up to string evolution, these are just dot diagrams. When these sufficiently match up, we may glue in the y -direction. We may think of this as a fancy version of composing over 1-morphisms. We say, “fancy,” because we are really composing over a composite of 1-morphisms.



The top and bottom faces of a surface diagram are string diagrams. We may compose over these in the z -direction.



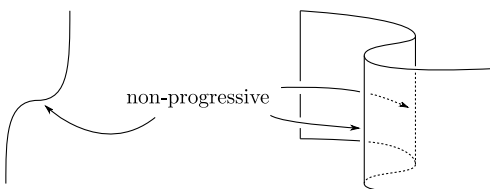
Away from z -heights containing a node, we may take horizontal slices of a surface diagram to get string diagrams.



These slices describe compositions of 2-morphisms in a 3-category. We may think of a surface diagram as being a vertical dot-diagram's worth of these horizontal slices. In order to ensure that these slices make sense, we insist that surface diagrams satisfy two progressivity rules.

- Seams must be transverse to every horizontal plane.
- Walls must be transverse to every line parallel to the x -axis.

These rules ensure that both seams and walls always travel some amount in the z -direction. Further, the progressivity rule for walls ensures that each horizontal slice of a string diagram is progressive in the y -direction. Here are two examples of strata that fail progressivity.

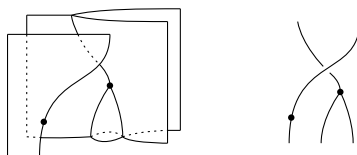


Notice that in the drawing of the non-progressive wall, we drew lines to indicate the wall turning back. These *fold lines* are not actual seams in the surface, but just an artifact of the perspective drawing. We chose the x -direction to be drawn back-to-front in surface diagrams so that fold lines in walls would correspond to places of non-progressivity.

While we may think of a surface diagram as a dot diagram's worth of string diagrams, it is also worth thinking of a surface diagram as a string diagram's worth of dot diagrams.

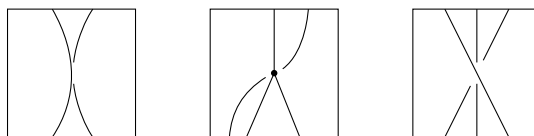
We may project a surface diagram to get a string diagram on the x -plane which has slice x -direction dot diagrams.

This projection takes nodes to nodes, seams to strings, walls to regions, and it ignores the regions of the surface diagram.



We'll include crossing information when seams cross in the projection. In the string diagram, these should just be regarded as a special type of node. We may think of these projection string diagrams as describing composites in the *microcosm* 2-category. Indeed, fixing two objects X and Y in a 3-category, $\text{Hom}(X, Y)$ has the structure of a 2-category.

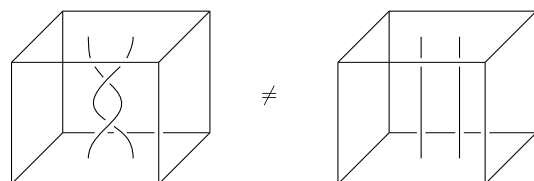
We can run across a few problems when we try to project a surface diagram. For instance, in the projection two seams might kiss, a node might land on a seam that it's not incident to, or three seams might create a triple point.



We'll call a surface diagram *photogenic* if it does not have any of these sorts of configurations of strata in projection. We'll also consider evolutions of surface diagrams which at each time are either photogenic or are mid-move corresponding to one of these three projections.

The string diagrams for the microcosm 2-category have crossing information that looks very much like the sort of thing we'd use for braids or knots. Although the word "seam" suggests that it is the meeting place of walls, we allow surface diagrams to have seams with no adjacent walls. Thus a diagram of strings in a cube can be thought of a special degenerate case of a surface diagram. At least informally, we'll allow ourselves to use "string" to refer to a seam lacking adjacent walls.

Recall that we were able to think of monoidal categories as bicategories with a single unlabeled 0-morphism. We had to shift degrees by one, labeling our strings with objects and nodes with morphisms. Similarly, we make think of a *braided monoidal category* as a tricategory with a single unlabeled 0-morphism and a single unlabeled 1-morphism, namely the identity for the 0-morphism. Here, we should shift degrees by two. We should therefore expect the diagrams of a braided monoidal category to consist of strings in a cube that are allowed to meet at nodes. We'll label the strings with objects and the nodes with morphisms. The single region will remain unlabeled. Roughly speaking, a braided monoidal category is a monoidal category equipped with isomorphisms, called braidings, which half-twist two objects past each other. These braidings do not need to satisfy symmetry relations: two half-twists need not be equal to the identity!



Recall, though, that when we defined our dot, string, and surface diagrams we always identify diagrams that are related by an evolution. We cannot therefore distinguish between the above two diagrams. Indeed, we do not anchor nodes in the source and target string diagrams, and so we may just pull one of the strings around the other in the top face. Our problem is that a braided monoidal category is a doubly-generate tricategory, not a doubly-degenerate 3-category. Doubly-degenerate 3-categories are automatically symmetric

monoidal. If we're going to see this braiding, can't quotient out by evolution willy-nilly. This also means that we're going to be forced to leave the strict world of 3-categories.

4.2 Tricategories

As we've seen, 3-categories are so strict that the braiding of 1-morphisms cannot be detected. Tricategories do not suffer from this deficiency. A tricategory comes equipped with many structural morphisms. These include 3-morphisms which observe the associativity and units for composition of 2-morphisms. Such 3-morphisms, themselves, satisfy coherence relations. There are also structural 2-morphisms that record the associativity and units for composition of 1-morphisms. As we are being fully weak, these structural 2-morphisms don't satisfy coherence relations, but rather there exist coherence 3-morphisms observing their coherence and themselves satisfy some coherence relations.

We expect the 2-morphisms of a 2-category (or bicategory) to satisfy an interchange relation:

$$(\alpha \circ \beta) \otimes (\gamma \circ \delta) = (\alpha \otimes \gamma) \circ (\beta \otimes \delta)$$

In a tricategory such a relation is too strict. Instead, we record an invertible 3-morphism, the *interchanger*, to observe an interchange for any four 2-morphisms. These interchangers satisfy coherence rules meaning that they "play nice" with the associators and unitors.

Consider a pair of 2-morphisms $\alpha, \beta: 1_X \rightarrow 1_X: X \rightarrow X$ in a tricategory whose sources and targets are a 0-morphism, X . We can get the braiding for α and β by composing unitors and interchangers in various directions:

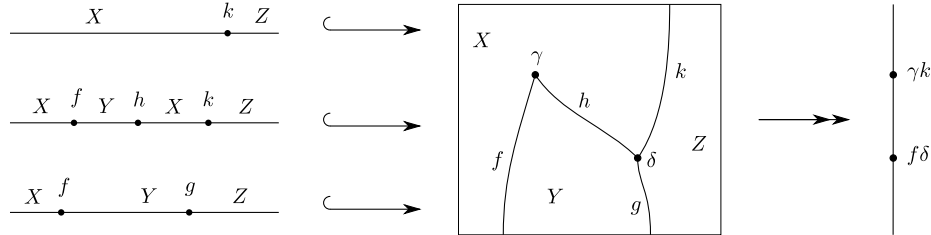
$$\begin{aligned} \alpha \circ \beta &\rightarrow (1_X \otimes \alpha) \circ (\beta \otimes 1_X) \rightarrow (1_X \circ \beta) \otimes (\alpha \circ 1_X) \rightarrow \beta \otimes \alpha \\ &\rightarrow (\beta \circ 1_X) \otimes (1_X \circ \alpha) \rightarrow (\beta \otimes 1_X) \circ (1_X \otimes \alpha) \rightarrow \beta \circ \alpha \\ &\rightarrow (1_X \otimes \beta) \circ (\alpha \otimes 1_X) \rightarrow (1_X \circ \alpha) \otimes (\beta \circ 1_X) \rightarrow \alpha \otimes \beta \\ &\rightarrow (\alpha \circ 1_X) \otimes (1_X \circ \beta) \rightarrow (\alpha \otimes 1_X) \circ (1_X \otimes \beta) \rightarrow \alpha \circ \beta \end{aligned}$$

This is great, except it's a bit of a pain to write out all of these unit and interchange rules explicitly. If we're going to use surface diagrams to describe compositions in a tricategory, we'll need to use many decorations to keep of all this straight. This actually proves to be quite a challenge.

When it came to decorating string diagrams to use in a bicategory, we only needed decorations on the source and target dot diagrams. Coherence ensured that any intermediate

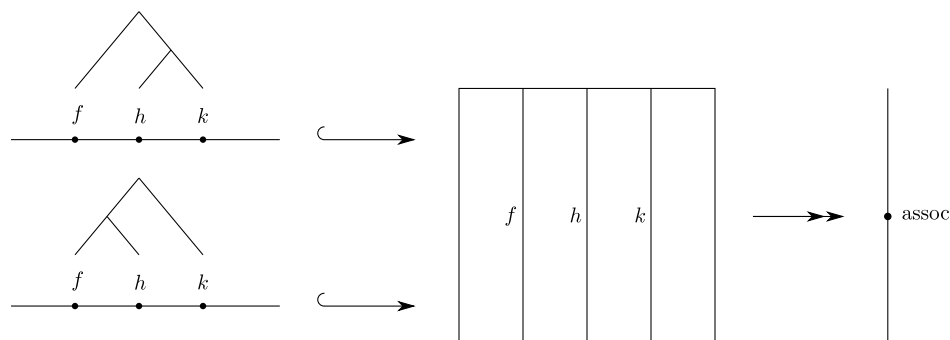
bracketings of slices would not affect which 2-morphism the diagram would evaluate to. Similarly, for surface diagrams, we shouldn't need to decorate all of the slices of a surface diagram, but rather just the source and target string diagrams. That said, what kind of decorations would we need to uniquely evaluate a string diagram in a tricategory?

Recall the picture of a string diagram as a dot diagram's worth of dot diagrams:

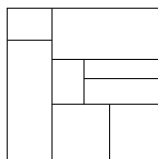


We'll need to keep track of units and bracketing for each of the slices and for the projection. Let's begin by thinking about the associators and unitors for the projection. These control the composition in the y -direction, that is, the composition of 2-morphisms over 1-morphisms. We'll think of these as node-less surface diagrams from a string diagram with one history to the same string diagram with another history. None of this is too much trouble because these associators and unitors are 3-morphisms and so their coherence comes in the form of equalities.

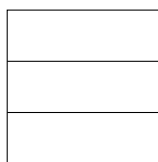
The above slice dot diagrams also need histories to describe their evaluation. We'll need to be able to describe the 2-morphism associators and unitors for these slices. These don't show up in a string diagram. Perhaps, though, we should none-the-less indicate the existence of such an associator or unitor as a node in the projection dot diagram. That is, nodes in the projection dot diagram should indicate changes in slice dot diagrams. This may be an interesting change due to a node in the string diagram, or it may be a less interesting change due to a rebracketing of a slice. We'll refer to a node of the latter type as *virtual*. We also need to be able to display the coherence 3-morphisms for the associator and unitor 2-morphisms. These would be node-less surface diagrams that allow us to manipulate the virtual nodes. We won't spend time going into these detail, though, because we have larger problems with this approach.



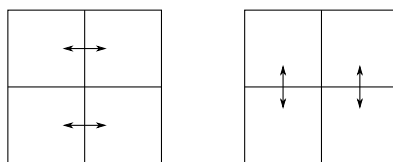
Our idea of a string diagram is as a composite in the y -direction of composites in the x -direction. That's all well and good in 2-categories or even in bicategories because any composition is equal to one in this form. But in a tricategory, we also need to be able to describe an x -direction composite of y -direction composites. Indeed, one of these may be unequal to any of the previous form. Really, we need to allow multiple iterations back and forth of such compositions. We'll briefly explore what this might look like for non-unital associativity. So we really need to be using Mondrians to encode the horizontal and vertical composition orders.



We still need bracket trees in order to make up for the lack of strict associativity or units.



The interchanger surface diagram allows us to reinterpret a vertical composition of horizontal compositions as a horizontal composition of vertical compositions.



We conclude that it should be possible to give an elaborate theory of decorations in order to evaluate string diagrams in tricategories. That said, trying to make a calculation in such a theory would involve many tedious manipulations. These diagrams would be “fully general” but at the expense of making them too cumbersome to be much use to anyone.

4.3 Gray-Categories

Let's have a quick introduction to Gray-categories. A Gray-category is defined to be a category enriched over the monoidal category, Gray. We won't spend time here defining Gray, but rather we discuss how one can think about a Gray-category. A Gray-category C has morphisms of degree 0, 1, 2, and 3. We require a rather strict condition for the lowest levels: the objects and 1-morphisms form a (1-)category. Further, for any pair of objects, (X, Y) , there is a 2-category $\text{Hom}_C(X, Y)$ of C 's 1-morphisms, 2-morphisms, and 3-morphisms. So far, this description sounds a lot like a 3-category. There is an important difference:

In a Gray-category, we may not compose 2-morphisms over objects.

At first glance, it might seem that we lose a lot by excluding horizontal composition of 2-morphisms. We do, however, allow ourselves *whiskering*: we may horizontally compose 2-morphisms with 1-morphisms. So given a pair of 2-morphisms,

$$\begin{aligned}\alpha &: f \rightarrow f' \\ \beta &: g \rightarrow g'\end{aligned}$$

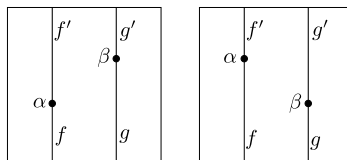
we may not form the horizontal composite

$$\alpha \otimes \beta: f \otimes g \longrightarrow f' \otimes g'$$

but we may form the vertical composites

$$\begin{aligned}(\alpha \otimes g) \circ (f' \otimes \beta) &: f \otimes g \longrightarrow f' \otimes g' \\ (f \otimes \beta) \circ (\alpha \otimes g') &: f \otimes g \longrightarrow f' \otimes g' .\end{aligned}$$

In a 3-category, these vertical composites would equal the horizontal composite, and in a tricategory, they would be coherently isomorphic. So we should regard both vertical composites as viable replacements for the horizontal composite. Indeed, drawing the string diagrams, we see that the vertical composites may be obtained by slightly nudging the 2-morphisms of the horizontal away from the same height.



As part of the data defining a Gray-category, we pick out interchange isomorphisms that relate the candidate horizontal composites:

$$(\alpha \otimes g) \circ (f' \otimes \beta) \longrightarrow (f \otimes \beta) \circ (\alpha \otimes g')$$

Of course, these interchangers must satisfy properties that ensure they are well-behaved and coherent.

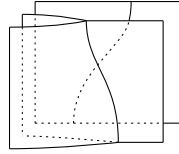
What sort of structure do we get if we truncate a Gray-category by forgetting its 3-morphisms? We may compose 1-morphisms over objects and 2-morphisms over 1-morphisms. These operations are strictly associative and unital. We lose the interchangers because they are 3-morphisms. This sort of structure is called a *sesquicategory*. Composition of 2-morphisms in a sesquicategory can be described by ordered string diagrams. A string diagram is ordered if we insist that its nodes occur at distinct y -heights. Evolution of ordered string diagrams, then, preserves the ordering. Note that we cannot horizontally glue ordered string diagrams and expect the result to be ordered: two nodes might accidentally end up at the same height after gluing. We won't worry about this, though, since we'll only expect there to be a sesquicategory of ordered string diagrams.

Given a Gray-category C (or more generally, a Gray-computad), we will define a Gray-category $\text{sd}(C)$ of surface diagrams labeled in C as follows:

- An object in $\text{sd}(C)$ is a dot labeled by an object of C .
- A 1-morphism in $\text{sd}(C)$ is a dot diagram, up to evolution.
- A 2-morphism in $\text{sd}(C)$ is an ordered string diagram, up to ordered evolution.
- A 3-morphism in $\text{sd}(C)$ is a (Gray) surface diagram, up to (Gray) evolution.

We need to be careful to spell out what we mean by a surface diagram between ordered string diagrams. As we are only looking to get a Gray-category of surface diagrams, we don't need to worry composing string or surface diagrams in the x -direction. This is a good thing, because our Gray-category C doesn't understand the corresponding compositions.

More interesting is how we represent interchangers as surface diagrams. We may slice a surface diagram with a horizontal plane at any height, except those containing nodes, to get a string diagram. Actually, we'll require a bit more: at all but finitely many heights, the slice of a surface diagram is an ordered string diagram. Exceptional heights then may contain a node or have two seams cross in x -projection. Here is an example of such a seam crossing.



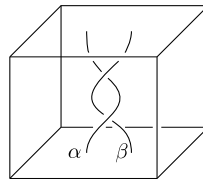
Let's revisit the case of two strings braiding. The braiding for

$$\alpha, \beta: 1_X \rightarrow 1_X: X \rightarrow X$$

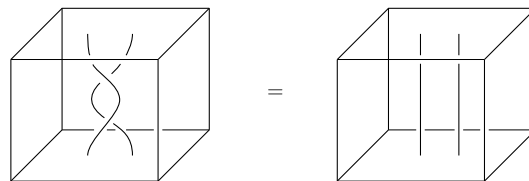
is easier to write down for a Gray-category than it was for a tricategory. In fact, it is just the composite of two interchangers.

$$\begin{aligned} \alpha \circ \beta &= (1_X \otimes \alpha) \circ (\beta \otimes 1_X) \longrightarrow (\beta \otimes 1_X) \circ (1_X \otimes \alpha) = \beta \circ \alpha \\ &= (1_X \otimes \beta) \circ (\alpha \otimes 1_X) \longrightarrow (\alpha \otimes 1_X) \circ (1_X \otimes \beta) = \alpha \circ \beta \end{aligned}$$

We can draw this as the following string diagram.



We require that evolution of surface diagrams restrict to evolution of ordered string diagrams on the bottom and top faces. Thus, we cannot untwist the previous diagram by because we cannot disturb the y -order of the nodes in the source and target string diagrams. On the other hand, we do allow full evolution on the interior of surface diagrams. We may undo a counter-clockwise half twist by following it with a clockwise half twist.



The Gray-interchangers allows us to braid strings in surface diagrams!

Chapter 5

Topological Background

5.1 Manifolds with Corners

In this dissertation, we'll be defining dot, string, and surface diagrams by putting stratifications on intervals, squares, and cubes. One could therefore probably get away with only defining stratifications on these particular spaces. We'll find it useful, however, to work in a bit more generality. In particular, spheres will feature prominently in our recursive definition of stratified spaces. We therefore wish to work in the context of manifolds with corners. We expect the reader to be more or less acquainted with the basic theory of smooth manifolds with and without boundary. Lee's introductory text [Lee03] has the requisite background. We'll have no need for topological manifolds, so all manifolds will be smooth without further mention.

Let's begin by recalling a few facts about manifolds and about manifolds with boundary. A manifold with boundary M is a space for which each point p may be locally identified with the origin in either Euclidean space \mathbb{R}^m or in the half-space $\mathbb{R}_{\geq 0} \times \mathbb{R}^{m-1}$. We accordingly classify p as an interior point or boundary point. We may think of a manifold, then, as a manifold with boundary but which just happens to lack boundary points. Going the other direction, "taking interior" turns a manifold with boundary M into a manifold $\text{int}(M)$ by discarding the boundary points. We have a second way to turn manifolds with boundary into manifolds: "taking boundary" turns an m -dimensional manifold with boundary into an $(m - 1)$ -dimensional manifold, $\partial(M)$. We therefore regard these operations, int and ∂ , as well-behaved for the class of manifolds with boundary. The topological operation of "taking products," however, does not take manifolds with boundary to manifolds with boundary. Suppose $p \in \partial(M)$ and $q \in \partial(N)$ are boundary points of M and N . Then a neighborhood

of (p, q) in $M \times N$ locally looks like

$$(\mathbb{R}_{\geq 0} \times \mathbb{R}^{m-1}) \times (\mathbb{R}_{\geq 0} \times \mathbb{R}^{n-1}) \cong \mathbb{R}_{\geq 0}^2 \times \mathbb{R}^{m+n-2}.$$

This is not a half-space but rather a corner.

We will call the spaces $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ for $a \geq 2$ the *standard corners*. These include, for example, the quadrant $\mathbb{R}_{\geq 0}^2$ in the plane, the octant $\mathbb{R}_{\geq 0}^3$ in Euclidean 3-space, and also the quarter space $\mathbb{R}_{\geq 0}^2 \times \mathbb{R}$ in Euclidean 3-space. An m -dimensional manifold with corners will be defined as a space locally modeled on the various $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ for $0 \leq a \leq m$. Any manifold with boundary is therefore an example of a manifold with corners which just lacks corner points. We may get a manifold by “taking the interior” of any manifold with corners. And it’s clear, at least locally, that the product of manifolds with corners is a manifold with corners. Things are a bit more complicated, however, when we wish to “take the boundary” of a manifold with corners. One should not expect the boundary of a manifold with corners to be a manifold with corners. Instead, we will see that the boundary decomposes into faces of various dimensions.

Definition. Suppose M is a topological space that we intend to give the structure of a manifold with corners. An m -dimensional chart for M is a homeomorphism

$$\Phi: U \rightarrow \Phi(U),$$

where $U \subset M$ and $\Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ are open subsets and $0 \leq a \leq m$.

Two m -dimensional charts

$$(\Phi, U \subset M, \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \quad \text{and} \quad (\Psi, V \subset M, \Psi(V) \subset \mathbb{R}_{\geq 0}^{a'} \times \mathbb{R}^{m-a'})$$

for a space M are *compatible* if either

- U and V are disjoint or
- the *transition functions* made by composing Φ and Ψ restricted to the overlap

$$\Phi(U \cap V) \xrightarrow{\Phi^{-1}} U \cap V \xrightarrow{\Psi} \Psi(U \cap V) \quad \text{and} \quad \Psi(U \cap V) \xrightarrow{\Psi^{-1}} U \cap V \xrightarrow{\Phi} \Phi(U \cap V)$$

are smooth.

Note that while $\Phi(U \cap V)$ is an open subset of $\mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}^{m-a_1}$, it need not be open in \mathbb{R}^m . Therefore, to say a transition function is smooth at a point of the boundary in \mathbb{R}^m , we’ll need to check that it has a smooth extension to some neighborhood in \mathbb{R}^m . See Appendix A for details.

Definition. An m -dimensional atlas for a space M is a collection of m -dimensional charts. We require

- the charts are pairwise compatible and
- every point of M is in at least one chart.

We'll say that a chart Φ is *compatible with an atlas* A if Φ is compatible with every chart in A .

Proposition. Suppose M is a topological space and A is an m -dimensional atlas for M . Then the set of charts,

$$\{\Phi : \Phi \text{ is compatible with } A\},$$

is an atlas and, moreover, is the largest atlas containing A .

Definition. An m -dimensional manifold with corners is a Hausdorff second countable topological space M equipped with a maximal m -dimensional atlas.

When using the word “manifold” informally in this dissertation, we will mean it to include the possibility of having boundary or corners. Also, we no longer have need to talk about charts for arbitrary topological spaces and so from here on a “chart” will always refer to one taken from a manifold’s atlas.

Proposition. Every chart Φ for a manifold with corners M is a diffeomorphism.

Proof. We will always equip $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ with the atlas generated by the single chart,

$$1: \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \longrightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}.$$

Then Φ may therefore be locally represented by the identity map,

$$1_{\Phi(U)}: \Phi(U) \rightarrow \Phi(U).$$

□

Proposition. Suppose M and N are manifolds with corners. Then the product $M \times N$ is a manifold with corners and $\dim(M \times N) = \dim(M) + \dim(N)$.

Proof. We just note that the map reordering factors

$$(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{\dim(M)-a}) \times (\mathbb{R}_{\geq 0}^{a'} \times \mathbb{R}^{\dim(N)-a'}) \longrightarrow \mathbb{R}_{\geq 0}^{a+a'} \times \mathbb{R}^{(\dim(M)+\dim(N)-(a+a'))}$$

is a diffeomorphism. □

Proposition. Suppose p is a point in a manifold with corners M . Then there exists a chart Φ centered at p in the atlas for M . That is, $\Phi(p) = 0$.

Proof. Our definition of atlas ensures that p is in some chart

$$(\Psi, U \subset M, \Psi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}).$$

We'll use the maximality of M 's atlas to show that there is another chart, Φ , which is centered at p . To this end, let's divide the indices $i \in \{1, \dots, m\}$ for x_i into three camps:

1. i where $1 \leq i \leq a$ and $x_i(p) = 0$;
2. i where $1 \leq i \leq a$ and $x_i(p) \neq 0$;
3. i where $a + 1 \leq i \leq m$.

The camp 1 and camp 2 coordinates may appear in any order in $\mathbb{R}_{\geq 0}^a$ for Ψ . Choose a diffeomorphism $\Gamma: \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \rightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ which permutes the factors so that the coordinates of $\Gamma \circ \Psi$ appear in order according to camp.

Let a_1 denote the number of indices in camp 1. Note that $a_1 \leq a$ and so

$$\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} = \mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}_{\geq 0}^{a-a_1} \times \mathbb{R}^{m-a} \subset \mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}^{m-a_1}.$$

This shows that $\Gamma(\Psi(U))$ is a subset of $\mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}^{m-a_1}$, however it may not be open. In any case, we may choose a smaller neighborhood $V \subset U$ of p so that $\Gamma(\Psi(V))$ is open as a subset of $\mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}^{m-a_1}$. For instance, we may take

$$V = \Psi^{-1}(\Psi(U) \cap \text{the open ball of radius } \varepsilon \text{ centered at } q),$$

where $\varepsilon = \min\{x_i(p) : i \text{ is in camp 2}\}$. By definition, $\Gamma(\Psi(p))$ has the first a_1 coordinates all zero. We may therefore define $\Phi: V \rightarrow \mathbb{R}_{\geq 0}^{a_1} \times \mathbb{R}^{m-a_1}$ by

$$\Phi(q) = \Gamma(\Psi(q)) - \Gamma(\Psi(p)).$$

Restrict Φ to its image. We claim that Φ is a chart in M 's atlas. Since the domain of Φ is contained in the domain of Ψ , it suffices to check Φ 's compatibility with Ψ . But this is clear because the transition function is a reordering of coordinates followed by a translation. \square

Definition. Suppose Φ is a chart centered at a point p with $\Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$. We'll say that (for Φ),

- p is an *interior point* of M if $a = 0$;

- p is on the boundary of M if $a \geq 1$;
- p is a corner point of M if $a \geq 2$.

We'll show later in this section that these definitions don't depend on the choice of chart Φ .

Our next task is to define the tangent space for a manifold with corners. Consider the set $C^\infty(M)$ of smooth \mathbb{R} -valued functions on M . This is an \mathbb{R} -algebra with the operations of pointwise addition and pointwise multiplication. Smooth functions $f: M \rightarrow N$ induce algebra homomorphisms

$$C^\infty(N) \longrightarrow C^\infty(M).$$

Definition. A linear map $D: C^\infty(M) \rightarrow \mathbb{R}$ is a *derivation at a point* $p \in M$ if it satisfies

$$D(fg) = g(p)D(f) + f(p)D(g).$$

Example. Suppose $\gamma: \mathbb{R} \rightarrow M$ is a curve taking 0 to $p \in M$. We get a derivation D_γ at p defined by taking a single-variable derivative as follows:

$$D_\gamma \left(M \xrightarrow{\alpha} \mathbb{R} \right) = \left. \frac{d}{dx} (\alpha \circ \gamma) \right|_{x=0}.$$

Note that γ need not be defined on all of \mathbb{R} . Rather, any open interval $I \subset \mathbb{R}$ neighborhood of the origin would suffice.

Definition. A derivation $D \in T_p(M)$ is *locally represented by a curve* $\gamma: I \rightarrow M$ with $\gamma(0) = p$ if it satisfies

$$D(\alpha) = \left. \frac{d}{dx} (\alpha \circ \gamma) \right|_{x=0}.$$

for all $\alpha \in C^\infty(M)$.

The *tangent space of M at p* , written $T_p(M)$, is the vector space of all derivations at p . We'll let $R_p(M) \subset T_p(M)$ denote the *set of representable derivations at p* . Given a smooth map $f: M \rightarrow N$, we may push-forward tangent vectors giving a linear map

$$T_p(M) \xrightarrow{f_*} T_{f(p)}(N).$$

Representable derivations are taken to representable derivations, so we may also see the push-forward as a map

$$R_p(M) \xrightarrow{f_*} R_{f(p)}(N).$$

Proposition. The tangent space T_p and the set of representable derivations R_p are local invariants:

$$T_p(U) = T_p(M) \quad \text{and} \quad R_p(U) = R_p(M)$$

for any neighborhood U of p .

Proposition. We may understand T_p and R_p on our standard models as follows.

1. The inclusion $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \hookrightarrow \mathbb{R}^m$ induces an isomorphism

$$T_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \cong T_0(\mathbb{R}^m)$$

on tangent spaces.

2. The inclusion $\mathbb{R}^{m-a} \hookrightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ induces a bijection

$$R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \cong R_0(\mathbb{R}^{m-a})$$

on representable sets.

3. In Euclidean space, every derivation at the origin is representable:

$$R_0(\mathbb{R}^k) = T_0(\mathbb{R}^k).$$

Proof Sketch.

1. We may show this isomorphism by investigating the kernel and image. In both cases, the key fact is that any smooth map $\alpha: \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \rightarrow \mathbb{R}$ may be extended to a smooth function $\tilde{\alpha}: \mathbb{R}^m \rightarrow \mathbb{R}$.
2. The standard projection $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \rightarrow \mathbb{R}^{m-a}$ is an inverse at the level of representable sets. Any derivation represented by a curve $\gamma: I \rightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ may be equally well represented by the composite curve $\gamma: I \rightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a} \rightarrow \mathbb{R}^{m-a}$.
3. It is a standard fact that every tangent vector is representable as a curve in a manifold without boundary.

□

Proposition. $R_p(M)$ is a subspace of $T_p(M)$. Further,

$$a + \dim(R_p(M)) = m$$

where a is determined by any chart $(\Phi, U \subset M, \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$ centered at p .

Proof. First, we'll show that $R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$ is a subspace of $T_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$. Note that the inclusion $f: \mathbb{R}^{m-a} \rightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ induces a linear map

$$T_0(\mathbb{R}^{m-a}) \rightarrow T_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \quad \text{and a bijection} \quad R_0(\mathbb{R}^{m-a}) \rightarrow R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}).$$

Since $R_0(\mathbb{R}^{m-a}) = T_0(\mathbb{R}^{m-a})$ is a vector space, we may use the bijection to equip the set $R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$ with this structure. The linear map ensures that this structure agrees with $T_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$.

Next, we'll show that $R_p(M)$ is a subspace of $T_p(M)$. Suppose Φ is a chart centered at p . This gives us a linear isomorphism

$$T_p(M) \rightarrow T_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \quad \text{and a bijection} \quad R_p(M) \rightarrow R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}).$$

We make $R_p(M)$ a vector space by identifying it with $R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$ and the linear isomorphism ensures that this makes $R_p(M)$ a subspace of $T_p(M)$. For the dimension result, we just note that we have isomorphisms

$$R_p(M) \cong R_p(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}) \cong R_p(\mathbb{R}^{m-a}) = T_p(\mathbb{R}^{m-a}).$$

□

The previous proposition tells us that we may associate a number to each point of a manifold with corners in a chart-independent way. We'll call $a = m - \dim(R_p(M))$ the *index of p* . We use this number to define the faces of a manifold.

Definition. Suppose M is a manifold with corners. A *face of M of codimension a* is a connected component of the subspace

$$\{p \in M : p \text{ has index } a\} \subset M.$$

We may decompose a manifold with corners into faces of various dimensions. The faces of codimension at least 1 form the boundary and the faces of codimension at least 2 consist of the corner points. We may also refer to the connected components of the interior as faces of codimension 0. We'll show that each face is a manifold without boundary. In fact, the closure of a face is very nearly a manifold with corners in its own right.

Example. Let's understand the faces of $\mathbb{R}_{\geq 0}^m$. Consider functions

$$\varphi: \{1, 2, \dots, m\} \rightarrow \{\{0\}, (0, \infty)\}.$$

We may decompose

$$\mathbb{R}_{\geq 0}^m = \prod_{\varphi} \prod_{1 \leq i \leq m} \varphi(i).$$

If we let $|\varphi|$ denote the number of times φ chooses $\{0\}$, then for a fixed index k , we may write

$$\{p \in \mathbb{R}_{\geq 0}^m : p \text{ has index } k\} = \prod_{|\varphi|=k} \prod_{1 \leq i \leq m} \varphi(i).$$

So $\mathbb{R}_{\geq 0}^m$ has $\binom{m}{k}$ many faces of codimension k .

Example. Faces of $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ are in bijection with the faces of $\mathbb{R}_{\geq 0}^a$ in a codimension preserving manner. In particular, each face of $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ is of the form

$$\left(\prod_{1 \leq i \leq a} \varphi(i) \right) \times \mathbb{R}^{m-a},$$

for some function

$$\varphi: \{1, 2, \dots, a\} \longrightarrow \{\{0\}, (0, \infty)\}.$$

Thus there are $\binom{a}{k}$ faces of $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ of codimension k corresponding to the ways of choosing $\{0\}$ exactly k times out of the first a coordinates.

Lemma. Suppose V is an open subset of a manifold with corners M and F is a face of M . Then $V \cap F$ decomposes as the disjoint union of the faces of V which intersect F .

Proof. The index of each point $p \in M$ is determined locally and so the index for p in V agrees with the index for p in M . Suppose p is a point of index a which is contained in a face F of M and a face F' of V . The points of F' are all index a and so

$$F' \subset \{q \in M : q \text{ has index } a\}.$$

Since F' is connected, it must sit in one connected component. We know F is the connected component containing p and so $F' \subset F$. As $F' \subset V$, we have $F' \subset V \cap F$. \square

Proposition. Suppose F is a face of a manifold with corners M and

$$(\Phi, U \subset M, \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$$

is a chart centered at $p \in F$. Then there is a neighborhood $V \subset U$ of p which satisfies:

- The face of V containing p is $V \cap F$.
- $\Phi(V \cap F) = \Phi(V) \cap \{0\}^a \times \mathbb{R}^{m-a}$.

Proof. Note that $\{0\}^a \times \mathbb{R}^{m-a}$ is the only face of $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ of codimension a . Since Φ is a diffeomorphism, it preserves index and thus

$$\Phi(U \cap F) \subset \Phi(U) \cap (\{0\}^a \times \mathbb{R}^{m-a}).$$

Choose a smaller neighborhood $V \subset U$ of p so that $\Phi(V) \cap \{0\}^a \times \mathbb{R}^{m-a}$ is connected. This ensures that $\Phi(V) \cap \{0\}^a \times \mathbb{R}^{m-a}$ is the only face of $\Phi(V)$ of codimension a . Therefore V also has only one face of codimension a . By our lemma, $V \cap F$ is the union of codimension a faces for V . But as there is only one such, we conclude $V \cap F$ is it. So all index a points of V are in $V \cap F$, and hence,

$$\Phi(V \cap F) = \Phi(V) \cap \{0\}^a \times \mathbb{R}^{m-a}.$$

□

Corollary. Each face F of a manifold with corners M is a manifold without boundary. If p is a point in F , then the index of p is equal to $\text{codim}(F)$. Further, as subspaces of $T_p(M)$,

$$T_p(F) = R_p(M).$$

Proof. Recall that the standard inclusion $\mathbb{R}^{m-a} \hookrightarrow \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ induces an isomorphism

$$T_0(\mathbb{R}^{m-a}) = R_0(\mathbb{R}^{m-a}) \cong R_0(\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}).$$

Since $\Phi: V \rightarrow \Phi(V)$ locally identifies $F \subset M$ with $\{0\}^a \times \mathbb{R}^{m-a} \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$, we may use naturality to conclude

$$T_p(F) = R_p(F) \cong R_p(M).$$

□

Corollary. The interior of a manifold with corners M is a manifold without boundary.

In our standard models $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$ the closure of a face is a manifold with corners. The following example shows that this need not always be the case.

Example (Teardrop). Consider the 2-dimensional manifold with corners M made with one 0-face, one 1-face, and one 2-face. One may think of the 1-face as a curve whose ends meet at a corner. The closure of the 1-face is not a 1-manifold with corners. That is, the boundary of $\mathbb{R}_{\geq 0}^2$ cannot be locally modeled by \mathbb{R} or $\mathbb{R}_{\geq 0}$ at the origin. We see this as a problem of taking the closure in M : what should be two distinct points (the endpoints) of the 1-face have been glued together in M .

Proposition. Suppose F is a face of a manifold with corners M . Then there is a manifold with corners G of the same dimension, an embedding $F \hookrightarrow G$, and a quotient map to the closure of F ,

$$G \twoheadrightarrow \overline{F} \subset M.$$

Proof Sketch. This dissertation will not use this result, so we only provide a brief outline of proof. Suppose p is a point in \overline{F} and

$$(\Phi, U \subset M, \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$$

is a chart centered at p . It is possible for F to approach p in multiple faces of $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$. Roughly speaking, we'll produce G by taking the closure locally in each of these faces separately. Consider the following property (*) for a chart Φ :

$$\Phi^{-1} \left(\{0\}^a \times \mathbb{R}^{\dim(F)} \times \{0\}^{m-a-\dim(F)} \right) \subset F.$$

We'll say that two such charts Φ and Ψ centered at p *agree* if the transition function $\Psi \circ \Phi^{-1}$ takes $\{0\}^a \times \mathbb{R}^{\dim(F)} \times \{0\}^{m-a-\dim(F)}$ to itself. Agreement is an equivalence relation. We define

$$G = \{(p, [\Phi]) : p \in \overline{F} \text{ and } \Phi \text{ satisfying } (*)\}.$$

We give G a topology and smooth structure using the charts satisfying (*). If $p \in F$, all special charts agree and so $F \hookrightarrow G$. \square

Definition. Suppose M is an m -dimensional manifold with corners. We define a *submanifold* $X \subset M$ of dimension $m' \leq m$ as follows. For each point p in X , we ask that there be a *submanifold chart*

$$(\Phi, U \subset M, \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a})$$

centered at p which identifies $U \cap X$ with

$$\Phi(U) \cap \left(\mathbb{R}_{\geq 0}^{a'} \times \{0\}^{a-a'} \right) \times \left(\mathbb{R}^{m'-a'} \times \{0\}^{(m-a)-(m'-a')} \right),$$

for some $a' \leq a$.

Example. Each face F of a manifold with corners M is a submanifold. We found a submanifold chart for F with $a' = 0$ and $m' = m - a$:

$$\{0\}^a \times \mathbb{R}^{m-a} \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}.$$

Definition. Suppose X and Y are submanifolds of a manifold with corners M . We'll say that X and Y *intersect transversely* if at every point $p \in X \cap Y$,

$$T_p(M) = T_p(X) + T_p(Y).$$

Proposition. The intersection of transverse submanifolds $X, Y \subset M$ is a submanifold of both X and Y whose dimension satisfies the following relation:

$$\dim(X \cap Y) + \dim(M) = \dim(X) + \dim(Y).$$

Definition. A submanifold X is *properly embedded* in a manifold with corners M if X intersects each face of M transversely.

Lemma. Suppose X is a properly embedded submanifold of a manifold with corners M . The index of $p \in X$ as a point in X agrees with the index of p as a point in M .

Proof. Suppose

$$\Phi: U \longrightarrow \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$$

is a submanifold chart for X centered at p :

$$\Phi(U \cap X) = \Phi(U) \cap \left(\mathbb{R}_{\geq 0}^{a'} \times \{0\}^{a-a'} \right) \times \left(\mathbb{R}^{m'-a'} \times \{0\}^{(m-a)-(m'-a')} \right).$$

We wish to show $a' = a$.

Suppose F is the face of M containing p . Let $V \subset U$ be a neighborhood of p which makes

$$\Phi(V \cap F) = \Phi(V) \cap \{0\}^a \times \mathbb{R}^{m-a}$$

the face of $\Phi(V)$ containing the origin. Since X is properly embedded, it is transverse to F and hence $\Phi(V \cap X)$ is transverse to $\Phi(V \cap F)$. But this means in $\mathbb{R}_{\geq 0}^a \times \mathbb{R}^{m-a}$,

$$\left(\mathbb{R}_{\geq 0}^{a'} \times \{0\}^{a-a'} \right) \times \left(\mathbb{R}^{m'-a'} \times \{0\}^{(m-a)-(m'-a')} \right)$$

is transverse to $\{0\}^a \times \mathbb{R}^{m-a}$, and hence $a' = a$. □

Proposition. Suppose X is a properly embedded submanifold of a manifold with corners M and F is a face of M . Then $X \cap F$ decomposes as the disjoint union of the faces of X which intersect F .

Proof. The proof of the special-case lemma, where $X = V$ is an open subset, goes through since index is preserved in X . □

5.2 Stratified Spaces

In this section, we'll give a definition of stratified spaces that will be sufficient for our theory of string and surface diagrams. A stratified space is a manifold equipped with distinguished submanifolds, called strata, which fit together in nice ways. Our definition of stratified spaces will be inductive to allow complex strata to be built out of less complex ones. We will first, however, make a definition of a pre-stratified space. This definition may be regarded as a lawless precursor to the correct definition. That is, a stratified space will be defined as a pre-stratified space that satisfies extra rules.

Definition. Suppose M is a manifold with corners. A *pre-stratification for M* consists of a set of properly embedded submanifolds \mathcal{S}_M called *strata*. We require the following.

- Each stratum $S \in \mathcal{S}_M$ is connected, and in particular, is non-empty.
- Every point p of M is in some stratum: $p \in S \in \mathcal{S}_M$.
- The strata are disjoint: for $S, T \in \mathcal{S}_M$,

$$S \cap T \neq \emptyset \implies S = T.$$

- The closure of a stratum may be decomposed into strata:

$$S \cap \bar{T} \neq \emptyset \implies S \subset \bar{T}.$$

The strata are given a pre-order as follows:

$$S \prec T := S \subset \bar{T}.$$

A *pre-stratified space* refers to a space equipped with a pre-stratification. We'll sometimes refer to M as the *ambient manifold*.

Definition. A *map of pre-stratified spaces* is a smooth map $f: M \rightarrow N$ and a map of pre-orders $\varphi: \mathcal{S}_M \rightarrow \mathcal{S}_N$ which satisfy:

$$p \in S \in \mathcal{S}_M \implies f(p) \in \varphi(S).$$

An isomorphism of pre-stratified spaces is therefore a diffeomorphism $M \cong N$ which restricts to diffeomorphisms of strata, $S \cong \varphi(S)$.

Example. We may equip any manifold with a pre-stratification by taking the strata to be its connected components.

Example. Any pre-stratified space M can be thickened with a line to a pre-stratified space $M \times \mathbb{R}$. The strata of $M \times \mathbb{R}$ are of the form

$$S \times \mathbb{R}$$

where $S \in \mathcal{S}_M$.

Example. Similarly, any pre-stratified space may be thickened with a half-line to a pre-stratified space $M \times \mathbb{R}_{\geq 0}$.

Example. Suppose $\mathcal{S}_{\mathbb{S}^{n-1}}$ is a pre-stratification for the $(n-1)$ -sphere. Then we may give \mathbb{R}^n a *cone pre-stratification* $\text{cone}(\mathbb{S}^{n-1})$ as follows.

- The origin $\{0\} \in \mathcal{S}_{\mathbb{R}^n}$ is the unique 0-stratum for \mathbb{R}^n .
- Each stratum $S \in \mathcal{S}_{\mathbb{S}^{n-1}}$ contributes a stratum for \mathbb{R}^n of 1 dimension greater:

$$\{rx \in \mathbb{R}^n : x \in S \text{ and } r \in (0, \infty)\} \in \mathcal{S}_{\mathbb{R}^n}.$$

Example. Suppose U is an open subset of a pre-stratified space M . Then U inherits a pre-stratification by intersecting each stratum $S \in \mathcal{S}_M$ with U and taking connected components.

Definition. We'll define *stratified space* recursively by complexity.

- A *stratified space M of complexity 0* is a pre-stratified space in which the strata are the connected components of M .
- A *stratified space M of complexity $k > 0$* is a pre-stratified space in which all strata have codimension less than or equal to k . We require the following.

Suppose p is a point in M . We demand that there are:

- a triple of natural numbers (a, b, c) satisfying $a + b + c = \dim(M)$,
- a neighborhood $U \subset M$ of p ,
- a stratification of the $(c-1)$ -sphere \mathbb{S}^{c-1} of complexity $k-1$,
- an open subset

$$\Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}(\mathbb{S}^{c-1})),$$

- and an isomorphism of pre-stratified spaces,

$$\Phi: U \xrightarrow{\sim} \Phi(U),$$

taking p to 0.

A few remarks about the definition are in order.

- In the case $c = 0$, we do not have a $(c - 1)$ -sphere. Instead, we give $\mathbb{R}^c = \text{pt}$ its unique stratification.
- A *stratified space* M (of arbitrary complexity) is one where the complexity equals $\dim(M)$.
- A *map of stratified spaces* is just a map of the underlying pre-stratified spaces. Thus, we have already required Φ to be an isomorphism of stratified spaces.
- We'll call the Φ in the stratified spaces definition a *stratification chart* to distinguish it from the charts of M which only preserve the manifold with corners structure.
- The isomorphism Φ preserves the codimension of strata and the codimension of faces. Thus if p sits in a stratum S and in a face F , then $a = \text{codim}(F)$ and $c = \text{codim}(S)$.

Each of our constructions for building new pre-stratified spaces out of old take stratified spaces to stratified spaces.

Proposition. Suppose M is a stratified space. Then

- $M \times \mathbb{R}$ is a stratified space.
- $M \times \mathbb{R}_{\geq 0}$ is a stratified space.
- If $U \subset M$ is an open subset, the pre-stratification that U inherits makes U a stratified space.

Proposition. Suppose \mathbb{S}^{n-1} is equipped with a stratification. Then \mathbb{R}^n equipped with $\text{cone}(\mathbb{S}^{n-1})$ is a stratified space.

Proof. Certainly, the origin in \mathbb{R}^n looks locally like a cone. Let p be any other point in \mathbb{R}^n and $S' \in \mathcal{S}_{\mathbb{R}^n}$ be the stratum containing p . But S' must be a radial thickening of a stratum $S \in \mathcal{S}_{\mathbb{S}^{n-1}}$. Therefore, near p , $S' \cong \mathbb{R} \times S$. We're done since \mathbb{S}^{n-1} has lower complexity. \square

Proposition. Each face F of a stratified space M inherits a stratification so that the inclusion $F \hookrightarrow M$ is a map of stratified spaces.

Proof. Suppose p is a point in F and suppose we have a stratified chart for M

$$\Phi: U \longrightarrow \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}(\mathbb{S}^{c-1}, \mathcal{S}_{\mathbb{S}^{c-1}}))$$

centered at p . Let $V \subset U$ be a neighborhood of p for which $V \cap F$ the face of V containing p and

$$\Phi(V \cap F) = \Phi(V) \cap \{0\}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}).$$

Restricting Φ to $V \cap F$ and its image gives a stratification chart for F . \square

Proposition. The product of two stratified spaces is a stratified space.

Proof. Suppose $(\mathbb{R}^c, (\text{cone}(\mathbb{S}^{c-1})))$ and $(\mathbb{R}^{c'}, (\text{cone}(\mathbb{S}^{c'-1})))$ are two cone stratifications. Note that $\mathbb{S}^{c-1} \times \mathbb{S}^{c'-1}$ is not a sphere nor is it even the correct dimension to help in showing that $\mathbb{R}^{c+c'}$ has a cone stratification. Instead, we take the strata for $\mathbb{R}^{c+c'}$ to be of the form $S \times S'$ where $S \in \mathcal{S}_{\mathbb{R}^c}$ and $S' \in \mathcal{S}_{\mathbb{R}^{c'}}$. This is a cone stratification: $\{0\}$ is the unique 0-stratum and each of the other strata are radially consistent. So $\mathbb{S}^{c+c'-1}$ is transverse to the stratification for $\mathbb{R}^{c+c'}$ and thereby inherits its stratification. Taking the cone recovers the stratification for $\mathbb{R}^{c+c'}$. \square

Proposition. Suppose S and T are strata in a stratified space M .

- If $S \subset \bar{T}$ and $T \subset \bar{S}$, then $S = T$.
- If $S \subset \bar{T}$ and $S \neq T$, then $\dim(S) < \dim(T)$.

Proof. These statements follow easily by induction on the complexity of stratified spaces. \square

Proposition. Suppose X is a properly embedded submanifold of a stratified space M which intersects each of the strata $S \in \mathcal{S}_M$ transversely. Then X inherits a stratification by taking the connected components of the $X \cap S$.

Proof. Suppose $p \in X$ lies in a stratum S of M . Let

$$\Phi: U \longrightarrow \Phi(U) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}(\mathbb{S}^{c-1}))$$

be a stratification chart centered at p . Since X is transverse to S , we know that $\Phi(U \cap X)$ is transverse to

$$\Phi(U \cap S) = \Phi(U) \cap \mathbb{R}_{\geq 0}^a \times \mathbb{R}^b \times \{0\}^c.$$

We'll assume that U was chosen small enough so that $U \cap F$ is the face of U containing p . Since X is properly embedded, we also know that $\Phi(U \cap X)$ is transverse to the face

$$\Phi(U) \cap \{0\}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}).$$

Let $k = \text{codim}(X)$. Each map

$$\varphi: \{1, \dots, m - k - a - c\} \longrightarrow \{\{0\}, \mathbb{R}\}$$

with $|\varphi| = k$ determines a submanifold of codimension k as follows:

$$\mathbb{R}_{\geq 0}^a \times \left(\prod_i \varphi(i) \right) \times (\mathbb{R}^c, \text{cone}) \subset \mathbb{R}_{\geq 0}^a \times \mathbb{R}^b \times (\mathbb{R}^c, \text{cone}).$$

For at least one of the φ , taking a projection and section will set up an isomorphism near p :

$$X \cong \mathbb{R}_{\geq 0}^a \times \left(\prod_i \varphi(i) \right) \times (\mathbb{R}^c, \text{cone}).$$

□

Chapter 6

Dot Diagrams

6.1 1-Computads

Definition. A *1-computad* G consists of two sets, G_0 and G_1 , along with a pair of maps

$$s, t: G_1 \longrightarrow G_0$$

called *source* and *target*. We will refer to the elements of G_i as *degree i generators*. A map $f: G \longrightarrow H$ of 1-computads consists of maps

$$f_0: G_0 \longrightarrow H_0$$

$$f_1: G_1 \longrightarrow H_1$$

which preserve the source and target maps: ($g \in G_1$)

$$s(f_1(g)) = f_0(s(g)) \qquad t(f_1(g)) = f_0(t(g)).$$

We'll use $\mathbf{1-Cmp}$ to denote the category of 1-computads.

One might hope that there were a 2-category of 1-computads analogous to the 2-category of small 1-categories. But since 1-computads lack composition, there is no good way to define a “natural transformation of maps of computads.”

Example. Any directed graph may be thought of as a 1-computad in which the 0-generators are the vertices and the 1-generators are the arrows.

Example. Any (small) category may be thought of as a 1-computad. We take the 0-generators to be the objects and the 1-generators to be the morphisms. This amounts to forgetting the category's composition and units.

Example. The *terminal* 1-computad T has a single 0-generator, $*$, and a single 1-generator, \star . We define $s(\star) = *$ and $t(\star) = *$. For any 1-computad G , there is a unique map of 1-computads $G \rightarrow T$.

6.2 Dot Diagrams

In order to define dot diagrams, we will first define dot pre-diagrams. A dot pre-diagram is a stratified space which becomes a dot diagram when it is given labels. One might be tempted to call such spaces “unlabeled dot diagrams.” However, we will reserve this term for dot diagrams labeled by the terminal 1-computad.

Definition. A *dot pre-diagram* d consists of a closed interval $[x_0, x_1]$, with $x_0 < x_1$, and equipped with a stratification that has finitely many strata. In a dot pre-diagram, we will call the codimension 0 strata *regions* and the codimension 1 strata *dots*.

Proposition. Each dot a in a dot pre-diagram d is locally modeled on

$$(\mathbb{R}, \{(-\infty, 0), \{0\}, (0, \infty)\}).$$

We’ll use the ordering on $[x_0, x_1]$ to call the regions incident to a “left” and “right.”

Proof. Consider $\pm\partial/\partial x|_a \in T_a$. Each tangent vector sits in some region of T_a , which may be thought of as a region of d . □

A dot pre-diagram is determined by the location of its dots. Our rules of stratified spaces require the dots to appear in the interior of the diagram. Between any two consecutive dots there is a region consisting of an open interval. There is also a region starting at, and including, x_0 and ending at the left-most dot. Similarly, there is a region from the right-most dot to, and including, x_1 .

Definition. Any dot pre-diagram d has an *underlying 1-computad* $\text{comp}(d)$ which takes the regions as 0-generators and the dots as 1-generators. We declare the source of a dot to be the region incident on the left and the target of a dot to be the region incident on the right.

Definition. A *dot diagram* consists of a dot pre-diagram d on an interval $[x_0, x_1]$, a 1-computad G of *labels*, and a map of 1-computads $\text{comp}(d) \rightarrow G$. The *source* of d is the 0-generator $s_x(d)$ which labels the region containing x_0 . Similarly, the *target* of d is the 0-generator $t_x(d)$ which labels the region containing x_1 . We may write $d: X \rightarrow Y$ to indicate $s_x(d) = X$ and $t_x(d) = Y$. Note, however, if we take X and Y as objects in a category C , this notation is not meant to imply that d is a morphism in C .

Example (cone). Suppose a is a 1-generator in a 1-computad G . We define the *cone dot diagram* $\text{cone}_G(a)$ on $[-1, 1]$ as follows. We give $[-1, 1]$ three strata, $[-1, 0] \amalg \{0\} \amalg (0, 1]$. We label $[-1, 0)$ with the source of a , $(0, 1]$ with the target of a , and $\{0\}$ with a , itself.

We may think about dot diagrams being built up by gluing together cone dot diagrams. This allows us to construct dot diagrams with any positive number of dots. However, there is one other important kind of dot diagram.

Example (dot-less). Suppose X is a 0-generator in a 1-computad G . There is a *dot-less dot diagram* 1_X on $[x_0, x_1]$. The whole space is a single region, $[x_0, x_1]$, and we label it with X . We may also call 1_X a *degenerate* dot diagram for its lack of dots.

6.3 Dot Evolutions

Dot evolutions are stratified spaces which will be playing the role of ambient isotopy for dot diagrams. In fact, one can show that any ambient isotopy can be thought of as a dot evolution. A dot evolution does not track where individual points of a region travel through time, but just tracks the region as a whole. This difference makes gluing dot evolutions considerably easier. We will see in the next chapter that dot evolutions are examples of string diagrams.

Definition. A *pre-evolution of dot diagrams*, or *dot pre-evolution*, δ consists of a stratified square $[x_0, x_1] \times [t_0, t_1]$ satisfying certain rules. We'll still call the strata of codimensions 0 and 1, *regions* and *dots*, despite their being thickened in a time direction.

- There are no strata of codimension 2.
- The left and right faces, $\{x_0\} \times [t_0, t_1]$ and $\{x_1\} \times [t_0, t_1]$, only intersect regions.
- At any point p , the tangent vectors $\pm\partial/\partial x|_p$ are not contained in any dot.

A few remarks are in order.

- In this and all future definitions of diagrams and evolutions we will assume that our closed intervals have non-trivial length: $x_0 < x_1$ and $t_0 < t_1$.
- We will use the words “square” and “cube” to refer to products of closed intervals even when those intervals have different lengths.

- According to our work on manifolds with corners, $\{x_0\} \times [t_0, t_1]$ is not a face of the square, but rather it consists of three faces of the square: $\{x_0\} \times \{t_0\}$, $\{x_0\} \times (t_0, t_1)$, and $\{x_0\} \times \{t_1\}$. That said, we'll allow ourself to use "face" informally to refer to the closure of a face.

Proposition. Suppose δ is a dot pre-evolution on $[x_0, x_1] \times [t_0, t_2]$. Each slice $[x_0, x_1] \times \{t_1\}$ for a fixed $t_1 \in [t_0, t_2]$ inherits a stratification making it a dot pre-diagram, $\delta|_{t=t_1}$.

Proof. We have defined dot pre-evolutions in such a way that each slice $[x_0, x_1] \times \{t_1\}$ be transverse to all strata in δ . So the slice inherits a stratification. By progressivity, the slice may only intersect each dot once. So the slice has finitely many dots, hence finitely many regions as well. \square

Proposition. The computadic structure of a slice dot pre-diagram is preserved across time in a pre-evolution.

Proof. Suppose δ is a dot pre-evolution on $[x_0, x_1] \times [t_0, t_1]$ and suppose S is a dot. We may choose $\varepsilon > 0$ small enough so that

$$A = \{(x, t) : p < x < p + \varepsilon \text{ with } (p, t) \in S\} \subset [x_0, x_1] \times [t_0, t_1]$$

does not intersect any dots. Since A is connected, it must sit in a single region to the right of S . We may similarly choose a small $\varepsilon' > 0$ so that

$$\{(x, t) : p - \varepsilon' < x < p \text{ with } (p, t) \in S\} \subset [x_0, x_1] \times [t_0, t_1]$$

picks out the region to the left of S . \square

Definition. A dot pre-evolution δ has an underlying 1-computad $\text{comp}(\delta)$ in which the 0-generators are the regions of δ and the 1-generators are the dots. We may take the source and target of a dot in any of the slice dot pre-diagrams.

Definition. A *dot evolution* consists of a dot pre-evolution δ , a 1-computad G , and a map of 1-computads, $\text{comp}(\delta) \rightarrow G$. We define:

- $s_x(\delta) = \delta|_{x=x_0}$ is the dot-less dot diagram on the left face.
- $t_x(\delta) = \delta|_{x=x_1}$ is the dot-less dot diagram on the right face.
- $s_t(\delta) = \delta|_{t=t_0}$ is the dot diagram on the initial face.

- $t_t(\delta) = \delta|_{t=t_1}$ is the dot diagram on the final face.

We'll write $\delta: d \rightarrow e$ to mean that $s_t(\delta) = d$ and $t_t(\delta) = e$. In this case, we may also say that d is *evolution-related* to e by δ .

Example. Each dot diagram d on $[x_0, x_1]$ has a constant dot evolution 1_d on $[x_0, x_1] \times [t_0, t_1]$.

Example. Any dot evolution $\delta: d \rightarrow e$ has a reverse evolution $\delta^{\text{rev}}: e \rightarrow d$.

We're interested in dot evolutions to give us an equivalence relation on dot diagrams. We may regard the previous examples of evolutions as demonstrations of reflexivity and symmetry for our relation. We'll also get a transitivity statement for evolutions by gluing in the time direction. This is a bit trickier to define, so it will wait until the next section.

Proposition. Suppose δ is a dot evolution on $[x_0, x_1] \times [t_0, t_2]$. Each slice $\delta|_{t=t_1}$ inherits labels making it a dot diagram.

Proof. Slicing preserves the codimension of strata and it preserves the x -ordering. So the inclusion $[x_0, x_1] \times \{t_1\}$ induces a map of underlying 1-computads,

$$\text{comp}(\delta|_{t=t_1}) \longrightarrow \text{comp}(\delta),$$

which allows us to pull back the labeling from δ . □

Proposition. Suppose d and e are dot diagrams on $[x_0, x_1]$. If d and e have the same labels in the same order, then d and e are related by a dot evolution.

Proof. Locate d at $[x_0, x_1] \times \{-1\}$ and locate e at $[x_0, x_1] \times \{1\}$. We give $[x_0, x_1] \times [-1, 1]$ a stratification by taking the strata of codimension 1 as lines from the k th dot of d to the k th dot of e . The regions are determined as the connected components of the complement of the union of the lines. □

6.4 Gluing and Normalization

Suppose d is a dot diagram on $[x_0, x_2]$ and $x_1 \in [x_0, x_2]$ is a point in a region of d . Then we may *split* d into two dot diagrams, $d|_{[x_0, x_1]}$ and $d|_{[x_1, x_2]}$. These dot diagrams inherit their stratifications and labelings from d . Going the other direction, we'll say that d can be made by *gluing* $d|_{[x_0, x_1]}$ and $d|_{[x_1, x_2]}$. To emphasize this perspective, we'll write

$$d = d|_{[x_0, x_1]} \cup_x d|_{[x_1, x_2]}.$$

Proposition. Suppose d on $[x_0, x_1]$ and e on $[x_1, x_2]$ are dot diagrams, and suppose

$$t_x(d) = s_x(e).$$

Then we may glue d and e . That is, we may form $d \cup_x e$.

Proof. Suppose T is the region in d containing x_1 and S is the region in e containing x_1 . We take $T \cup S$ as a stratum in $d \cup_x e$. Near x_1 , $T \cup S$ looks like the stratified space $(\mathbb{R}, \{\mathbb{R}\})$. We take all other strata from d and e as strata for $d \cup_x e$. We label the strata for $d \cup_x e$ according to their labels in d and e . There is no ambiguity for the label on $T \cup S$ because the label on T is $t_x(d)$ and the label on S is $s_x(e)$. \square

Definition. Suppose δ is a dot evolution on $[x_0, x_2] \times [t_0, t_2]$.

- Suppose $x_1 \in (x_0, x_2)$ is a point for which $[x_0, x_1] \times [t_0, t_2]$ and $[x_1, x_2] \times [t_0, t_2]$ inherit the structure of a dot evolution from δ . Then we'll say that δ can be made by *horizontally gluing*:

$$\delta = \delta|_{[x_0, x_1] \times [t_0, t_2]} \cup_x \delta|_{[x_1, x_2] \times [t_0, t_2]}.$$

- Suppose $t_1 \in (t_0, t_2)$ is any point. Then $[x_0, x_2] \times [t_0, t_1]$ and $[x_0, x_2] \times [t_1, t_2]$ inherit the structure of a dot evolution from δ . We'll say that δ can be made by *gluing in time*:

$$\delta = \delta|_{[x_0, x_2] \times [t_0, t_1]} \cup_t \delta|_{[x_0, x_2] \times [t_1, t_2]}.$$

Proposition. Suppose δ and ϵ are dot evolutions and $t_x(\delta) = s_x(\epsilon)$. Then we may horizontally glue δ and ϵ .

Proof. Suppose δ is located on $[x_0, x_1] \times [t_0, t_1]$ and ϵ is located on $[x_1, x_2] \times [t_0, t_1]$. The face $\{x_1\} \times [t_0, t_1]$ intersects a single region T of δ and a single region S of ϵ . Gluing these, we get a stratum $S \cup T$ for $\delta \cup_x \epsilon$. Indeed, near points (x_1, t) , $S \cup T$ looks like \mathbb{R}^2 for $t \in (t_0, t_1)$ and $S \cup T$ looks like $\mathbb{R} \times \mathbb{R}_{\geq 0}$ for $t \in \{t_0, t_1\}$. All other strata of δ and ϵ become distinct strata for $\delta \cup_x \epsilon$. We label according to the labels from δ and ϵ . \square

It is harder to glue dot evolutions in time. Indeed, suppose δ and ϵ are dot evolutions which satisfy $t_t(\delta) = s_t(\epsilon)$. We may not be able to glue δ and ϵ because their dots do not line up in a smooth manner. By working with dot evolutions that have their dots approach the boundary in a perpendicular manner, we may work around this issue.

Definition. Suppose δ is a dot evolution defined on $[x_0, x_1] \times [t_0, t_1]$. We will say

- δ is *normal near s_t* if δ agrees with the constant evolution $1_{s_t(\delta)}$ on some neighborhood of the initial face $[x_0, x_1] \times \{t_0\}$.
- δ is *normal near t_t* if δ agrees with the constant evolution $1_{t_t(\delta)}$ on some neighborhood of the final face $[x_0, x_1] \times \{t_1\}$.

Proposition (normal replacement). Suppose d and e are dot diagrams that are related by a dot evolution. Then there is a dot evolution $\delta: d \rightarrow e$ which is normal near s_t , or near t_t , or both.

Proof. Suppose $\epsilon: d \rightarrow e$ is a dot evolution on $[x_0, x_1] \times [t_0, t_1]$. For ease of argument, we'll assume that $t_0 = 0$ and $t_1 = 5$. We define a normal dot evolution $\delta: d \rightarrow e$ on $[x_0, x_1] \times [0, 5]$ as follows. Let $f: [0, 5] \rightarrow [0, 5]$ be an increasing smooth function that is constantly 0 on $[0, 1]$, the identity on $[2, 3]$, and constantly 5 on $[4, 5]$. Each stratum S of ϵ contributes a stratum

$$S' = \{(x, t) : (x, f(t)) \in S\}.$$

to δ . In other words, each slice $\delta_{t=k}$ of δ agrees with each slice $\epsilon_{t=f(k)}$ of ϵ . For $t \in [0, 1]$, each slice of δ is $\epsilon_{t=0}$, and for $t \in [4, 5]$, each slice of δ is $\epsilon_{t=5}$. \square

Definition. Suppose $\delta: d \rightarrow e$ and $\epsilon: e \rightarrow f$ are dot evolutions. Then we may *compose δ and ϵ in time* by taking normal replacements and then gluing. That is, take δ' normal near t_t and take ϵ' normal near s_t . We define

$$\delta \circ_t \epsilon = \delta' \cup_t \epsilon'$$

as a dot evolution $d \rightarrow f$.

6.5 The Category $\text{dd}(G)$

Given any dot diagram d on $[x_0, x_1]$, we may produce many other geometrically similar dot diagrams:

- For any $k \in \mathbb{R}$, we get a dot diagram on $[x_0 + k, x_1 + k]$ by translating each of the strata in d by k to the right.
- For any positive number c , we get a dot diagram on $[cx_0, cx_1]$ by dilating each of the strata in d by a factor of c .

Both of these geometric operations induce isomorphisms on the underlying 1-computads. They also preserve the source and target of a dot diagram. We may also think of horizontal translation and dilation acting on dot evolutions. So we may check whether two dot diagrams are related by an evolution before or after translating or dilating the diagrams. We will quotient the collection of dot diagrams by translation, dilation, and dot evolution to get our category of dot diagrams.

Definition. For any 1-computad G there is a *category* $\text{dd}(G)$ of dot diagrams labeled by G .

$$\text{Obj}_{\text{dd}(G)} = G_0.$$

$$\text{Hom}_{\text{dd}(G)}(X, Y) = \frac{\{\text{dot diagrams } d: X \rightarrow Y \text{ labeled by } G\}}{\text{translation, dilation, dot evolution}}.$$

We compose dot diagrams by gluing. We have also seen that dot evolutions may also be horizontally glued, which ensures that gluing descends to the evolution-related equivalence classes of dot diagrams. Also, dot evolution preserves the source s_x and target t_x of a dot diagram, and so these make sense on equivalence classes. We get units for objects by taking dot-less dot diagrams.

Proposition. Suppose Y is a 0-generator. Recall that 1_Y denotes the dot-less dot diagram labeled by Y . In $\text{dd}(G)$, 1_Y is the identity on Y .

Proof. Suppose $d: X \rightarrow Y$ is a dot diagram on $[0, 1]$. Let's view 1_Y on $[1, 2]$ so that the gluing $d \cup_x 1_Y$ is on $[0, 2]$. There is a dot evolution linearly interpolating $1_Y \cup_x d$ to the $\cdot 2$ dilation of d . This shows that 1_Y cancels on the right. Similarly, if we take 1_Y on $[-2, -1]$ and any $e: Y \rightarrow Z$ on $[-1, 0]$, we see that 1_Y also cancels on the left. \square

6.6 Evaluation

In this section, we're interested in evaluating dot diagrams that are labeled in a fixed category C . Our procedure will be to break a dot diagram d into bite-sized pieces, convert each of these into morphisms in C , and then compose them up. There is some ambiguity in exactly what manner we choose our pieces. We'll record this as extra structure for a dot diagram, use it to define evaluation, and then show that evaluation doesn't actually depend on the structure.

Definition. A dot diagram is a *brick* if it has 0 or 1 dots.

Definition. Suppose d is a dot diagram on $[x_0, x_1]$. A set M of mortar points for d is a finite subset of $[x_0, x_1]$. Each point of M is located in a region of d . We require that M include both $\{x_0\}$ and $\{x_1\}$. Between any two successive points $p < q \in M$ we require the dot diagram $d|_{[p,q]}$ to be a brick.

Definition. Suppose d is a dot diagram labeled in a category C and M is a mortar set for d . We'll define the morphism

$$s(d) \xrightarrow{\text{eval}_C^M(d)} t(d)$$

in C as follows.

- Suppose M is empty and d is dot-less. Let X be the object labeling the unique region of d . We define,

$$\text{eval}_C^M(d) = 1_X.$$

- Suppose M is empty and d is a cone. Let a be the label on the single dot of d . We define,

$$\text{eval}_C^M(d) = a.$$

- Suppose $p \in M$ is a splitting point for d . We define,

$$\text{eval}_C^M(d) = \text{eval}_C^{M \cap [x_0,p]}(d|_{[x_0,p]}) \circ \text{eval}_C^{M \cap [p,x_1]}(d|_{[p,x_1]}).$$

To be very precise, we should decide on a convention for which splitting point $p \in M$ we use to evaluate. We could, for example, always take the splitting point with smallest x -coordinate. The associativity for composition in C ensures that a different choice of convention will not change our definition in any way.

Lemma. Suppose $M \subset M'$ are mortar sets for M . Then, $\text{eval}_C^M(d) = \text{eval}_C^{M'}(d)$.

Proof. We'll consider the case $M' = M \cup \{p\}$. Let B be the brick of M containing p .

- Suppose B is a dot-less brick labeled by X . Then for M , we evaluate B as 1_X . For M' , we evaluate B as $1_X \circ 1_X$.
- Suppose B is a cone bricked labeled by a . For M , we evaluate B as a . For M' , we evaluate B as either $1 \circ a$ or $a \circ 1$ depending on which region p is located in.

We get the result for arbitrary $M' \supset M$ by repeatedly adjoining points. \square

Proposition. Evaluation is independent of a dot diagram's mortar set.

Proof. Suppose M and M' are mortar sets for a dot diagram d . By the previous lemma,

$$\text{eval}_C^M(d) = \text{eval}_C^{M \cup M'}(d) = \text{eval}_C^{M'}(d).$$

□

Last, we wish to show that evaluation is independent of evolution of dot diagrams. That is, any two dot diagrams related by a dot evolution should evaluate to the same morphism. Notice that given enough time in a dot evolution, a dot may intersect a mortar point for the initial dot diagram. However, at least for a short period of time, the mortar point will remain valid. This will be enough for us.

Definition. A dot evolution is a *brick* if it contains 0 or 1 dots.

Lemma. Suppose δ is a dot evolution on $[x_0, x_1] \times [t_0, t_4]$. For any $t_2 \in (t_0, t_4)$, there are t_1, t_3 satisfying:

- $t_0 \leq t_1 < t_2 < t_3 \leq t_4$,
- $\delta|_{[x_0, x_1] \times [t_1, t_3]}$ may be written as a *layer of bricks*:

$$\delta|_{[x_0, x_1] \times [t_1, t_3]} = B_1 \cup_x \cdots \cup_x B_k.$$

Proof. We choose a mortar set for the slice dot diagram $\delta|_{[x_0, x_1] \times \{t\}}$. We note that $|\partial x / \partial t|$ is bounded for each of the dots in a dot evolution and so there is some amount of time before a dot intersects a mortar point. □

We may also find a layer of bricks near $t = t_0$ and near $t = t_4$. A *t-mortar set* for a dot evolution δ is a finite set of points $M_t \subset [t_0, t_4]$, including both t_0 and t_4 . Between successive points, we require δ to decompose as a layer of bricks. Each layer of bricks gets its own *x-mortar set* describing the placement of the bricks.

Proposition. Suppose d and e are dot diagrams that are related by a dot evolution. Then,

$$\text{eval}_C(d) = \text{eval}_C(e).$$

Proof. Let $\delta: d \rightarrow e$ be a dot evolution on $[x_0, x_1] \times [t_0, t_1]$. We use compactness of $[t_0, t_1]$ and the previous lemma to produce a *t-mortar set* for δ . It suffices to show that for each layer of bricks, the evaluation of the source agrees with the evaluation of the target. But this is clear since for any brick B , we have $\text{eval}_C(s(B)) = \text{eval}_C(t(B))$. □

6.7 Characterizing $\text{dd}(G)$ as the Free Category

We now have the tools that we need to characterize $\text{dd}(G)$ as the free category on the 1-computad G . Recall that we defined 1-Cmp to be the category of 1-computads. We'll let Cat denote the category of (small) categories. Notice that by viewing each category as a 1-computad, we get a functor

$$\text{forget}: \text{Cat} \longrightarrow \text{1-Cmp}.$$

Our construction of a category of dot diagrams gives us a functor going in the other direction,

$$\text{dd}: \text{1-Cmp} \longrightarrow \text{Cat}.$$

Indeed, any map of 1-computads $f: G \rightarrow H$ induces a functor $\text{dd}(G) \rightarrow \text{dd}(H)$ by relabeling,

$$\text{comp}(d) \longrightarrow G \xrightarrow{f} H.$$

Definition. For any 1-computad G , there is a natural map of 1-computads,

$$\text{cone}_G: G \longrightarrow \text{forget}(\text{dd}(G)),$$

called *conical inclusion*. Degree 0 generators may be thought of as objects in $\text{dd}(G)$. Each 1-generator is sent to its cone dot diagram.

Definition. For any category C , we may view evaluation as a functor

$$\text{eval}_C: \text{dd}(\text{forget}(C)) \longrightarrow C.$$

This functor takes objects of C to themselves and evaluates dot diagrams as described in the previous section. We note that this functor is natural in C .

Lemma (Zig-Zag). The composite of the conical inclusion and the evaluation maps

$$\text{forget}(C) \xrightarrow{\text{cone}_{\text{forget}(C)}} \text{forget}(\text{dd}(\text{forget}(C))) \xrightarrow{\text{forget}(\text{eval}_C)} \text{forget}(C)$$

equals the identity on $\text{forget}(C)$.

Proof. Let's analyze the composite. The first half asks us to view any morphism $a: X \rightarrow Y$ as a dot diagram by conical inclusion. That is, we label the center of a cone on \mathbb{S}^0 with a . The second half asks us to evaluate this dot diagram. According to our definition of evaluation, we get a back. \square

Lemma (Zig-Zag). The composite of the conical inclusion and the evaluation maps

$$\mathrm{dd}(G) \xrightarrow{\mathrm{dd}(\mathrm{cone}_G)} \mathrm{dd}(\mathrm{forget}(\mathrm{dd}(G))) \xrightarrow{\mathrm{eval}_{\mathrm{dd}(G)}} \mathrm{dd}(G)$$

equals the identity on $\mathrm{dd}(G)$.

Proof. The above composition takes a dot diagram d to the following dot diagram e , built as follows:

- At each dot of d , replace the 1-generator a of G labeling the dot with the conical dot diagram, $\mathrm{cone}(a)$.
- Create a dot diagram e by gluing together the various $\mathrm{cone}(a)$ according to d .

Since e has the same dots in the same order as d , these dot diagrams are related by a dot evolution and hence equal in $\mathrm{dd}(G)$. \square

Theorem. The category $\mathrm{dd}(G)$ is the free category on G . More precisely, there is a bijection

$$\mathrm{Hom}_{1\text{-Comp}}(G, \mathrm{forget}(C)) \cong \mathrm{Hom}_{\mathrm{Cat}}(\mathrm{dd}(G), C),$$

which is natural in 1-computads G and categories C .

Proof. This is a straightforward formal consequence of our two Zig-Zag lemmas. See Appendix B for details. We'll sketch the proof in any case.

It's easy to see that a functor $f: \mathrm{dd}(G) \rightarrow C$ determines a map of 1-computads $G \rightarrow \mathrm{forget}(C)$. Consider f as a map of 1-computads $\mathrm{forget}(\mathrm{dd}(G)) \rightarrow \mathrm{forget}(C)$. We get the desired map by pre-composing with conical inclusion, $G \rightarrow \mathrm{forget}(\mathrm{dd}(G))$.

If, on the other hand, we are given an interpretation $f: G \rightarrow \mathrm{forget}(C)$ of a 1-computad G , we should be able to extend this to a functor $\mathrm{sd}(G) \rightarrow C$ used to evaluate dot diagrams. We regard f on dot diagrams, $\mathrm{dd}(G) \rightarrow \mathrm{dd}(\mathrm{forget}(C))$, and then post compose with evaluation $\mathrm{dd}(\mathrm{forget}(C)) \rightarrow C$. So, given a dot diagram d and a labeling $\mathrm{comp}(d) \rightarrow G$, we use our interpretation $G \rightarrow \mathrm{forget}(C)$ to label d in C . We may therefore think of d in $\mathrm{dd}(\mathrm{forget}(C))$, and hence we may evaluate to get a morphism in C . \square

Chapter 7

String Diagrams

7.1 2-Computads

Definition. A 2-computad G consists of three sets of generators G_0, G_1 , and G_2 . We equip G with source and target maps

$$s, t: G_1 \longrightarrow G_0$$

which allow us to generate the free category, $\text{dd}(G)$. Note that the set $\text{dd}(G)_1$ of dot diagrams taken up to dot evolution comes with source and target maps $\text{dd}(G)_1 \longrightarrow G_0$ extending the above maps $G_1 \longrightarrow G_0$. We also equip G with source and target maps

$$s, t: G_2 \longrightarrow \text{dd}(G)_1.$$

We require all 2-generators $\alpha \in G_2$ be globular:

$$s(s(\alpha)) = s(t(\alpha)) \quad \text{and} \quad t(s(\alpha)) = t(t(\alpha)).$$

So not only does a 2-generator α have source and target 1-generators, globularity allows us to talk about α having source and target 0-generators.

A morphism of 2-computads $f: G \longrightarrow H$ is a triple of maps $f_i: G_i \longrightarrow H_i$ for $i = 0, 1, 2$. We require these maps to commute with the source and target maps.

$$\begin{aligned} s(f_1(a)) &= f_0(s(a)) \in H_0 & t(f_1(a)) &= f_0(t(a)) \in H_0, \\ s(f_2(\alpha)) &= f_1(s(\alpha)) \in \text{dd}(H)_1 & t(f_2(\alpha)) &= f_1(t(\alpha)) \in \text{dd}(H)_1. \end{aligned}$$

Note that relation $s(f_2(\alpha)) = f_1(s(\alpha))$ is only enforced “up to evolution” in $\text{dd}(H)_1$.

Example. Any 1-computad may be thought of as a 2-computad with no 2-generators.

Example. There is a 2-computad T with a single 0-generator, $*$, a single 1-generator \star , and a single 2-generator for every pair of dot diagrams. This is the *terminal* 2-computad. For any 2-computad G , there is a unique map $G \rightarrow T$.

Example. Any 2-category C may be thought of a 2-computad $\text{forget}(C)$ in which every morphism is a generator. More precisely, we take

$$\begin{aligned} \text{forget}(C)_0 &= \text{Obj}_C, \\ \text{forget}(C)_1 &= \coprod_{X, Y \in \text{Obj}_C} \text{Hom}_C(X, Y), \\ \text{forget}(C)_2 &= \coprod_{a, b \in \text{dd}(\text{forget}(C))_1} \{(a, b, \alpha) : \alpha \in 2\text{-Hom}_C(\text{eval}(a), \text{eval}(b))\}. \end{aligned}$$

Notice that each 2-morphism α contributes many 2-generators to $\text{forget}(C)$: one for each pair of dot diagrams that evaluate to the domain and codomain. This evaluation happens in the 1-category made by truncating C .

7.2 String Diagrams

Definition. A *string pre-diagram* δ is a stratified square $[x_0, x_1] \times [y_0, y_1]$ with finitely many strata. We call the codimension 0 strata *regions*, the codimension 1 strata *strings*, and the codimension 2 strata *nodes*. We require the following boundary and progressivity rules.

- The left and right faces, $\{x_0\} \times [y_0, y_1]$ and $\{x_1\} \times [y_0, y_1]$, only intersect regions.
- At any point p , the tangent vectors $\pm \partial / \partial x|_p$ are not contained in any string.

We only need to enforce the progressivity rule at points p located in strings or nodes since it holds automatically in regions for dimension reasons. We define

- $s_x(\delta) = \delta|_{x=x_0}$ and $t_x(\delta) = \delta|_{x=x_1}$ are the dot pre-diagrams on the left and right faces.
- $s_y(\delta) = \delta|_{y=y_0}$ and $t_y(\delta) = \delta|_{y=y_1}$ are the dot pre-diagrams on the bottom and top faces.

Definition. We'll say that a string pre-diagram ϵ is *focused on a stratum* S if all other strata in ϵ are incident to S . In particular, S is the stratum of ϵ with the highest codimension.

Lemma. Suppose S is a stratum in a string pre-diagram δ and suppose p is a point in S . Then there is a string pre-diagram ϵ which contains p and is focused on S . We'll say ϵ is made by *cropping* δ about p . If p is in the interior of δ , we may take ϵ so that p is also in the interior of ϵ .

Proof. Suppose δ is located on $[x_0, x_4] \times [y_0, y_4]$ and p has coordinates (x_2, y_2) . We claim that there are x_1, x_3, y_1 , and y_3 making $\epsilon = \delta|_{[x_1, x_3] \times [y_1, y_3]}$ a string diagram focused on S about p . We must be careful to satisfy the boundary rule for a string pre-diagram: the left and right faces may only intersect regions. We choose $[x_1, x_3] \ni x_2$ so that $\delta|_{[x_1, x_3] \times \{y_2\}}$ is a dot pre-diagram focused on S . We note that since strings may not travel horizontally, there is some small interval $[y_1, y_3] \ni y_2$ for which $\{x_1, x_3\} \times [y_1, y_3]$ only intersects regions. Thus $\delta|_{[x_1, x_3] \times [y_1, y_3]}$ is a string pre-diagram focused on S . \square

Definition. Suppose δ is a string pre-diagram and d is a slice dot pre-diagram focused on a string S of δ . We'll define the *source of S* to be $s^d(S) = s_x(d)$ and the *target of S* to be $t^d(S) = t_x(d)$, both thought of as regions of δ .

Lemma. Suppose d and e are slice dot pre-diagrams focused on a string S of a string pre-diagram δ . Suppose further that d and e both intersect δ at a point p . Then

$$s^d(S) = s^e(S) \quad \text{and} \quad t^d(S) = t^e(S).$$

Proof. The intersection $d \cap e$ is a string pre-diagram focused on S . Certainly,

$$s^d(S) = s^{d \cap e}(S) = s^e(S) \quad \text{and} \quad t^d(S) = t^{d \cap e}(S) = t^e(S).$$

\square

Proposition. Suppose S is a string in a string pre-diagram δ and suppose that d and e are any slice dot pre-diagrams focused on S . Then

$$s^d(S) = s^e(S) \quad \text{and} \quad t^d(S) = t^e(S).$$

Proof. Our cropping lemma allows us to cover S with string pre-diagrams focused on S . For any two points p and q in S , we may choose a path from p to q in S . We cover the image of the path with a finite sequence $\epsilon_1, \dots, \epsilon_n$ of string pre-diagrams focused on S . The source and target of S are preserved when taken as slices for any single ϵ_i . We'll index the ϵ_i so that each pair $(\epsilon_i, \epsilon_{i+1})$ has non-trivial overlap. Notice that $\epsilon_i \cap \epsilon_{i+1}$ is also a string pre-diagram focused on S . Let d be a slice dot pre-diagram focused on S in $\epsilon_i \cap \epsilon_{i+1}$. We may see d as part of a slice of ϵ_i and also part of a slice of ϵ_{i+1} . By the previous lemma, we may calculate the source and target of S for both ϵ_i and ϵ_{i+1} as $s^d(S)$ and $t^d(S)$. \square

Definition. Suppose δ is a string pre-diagram. There is a *1-computad* $\text{comp}^1(\delta)$ underlying δ as follows. We take the regions of δ to be 0-generators and the strings of δ to be 1-generators. The source and target of a string are given by $s^d(S)$ and $t^d(S)$ for any slice dot pre-diagram d focused on S .

Definition. Suppose N is a node in a string pre-diagram δ . For any string pre-diagram ϵ focused on N , we'll define the following.

- The *source of N* is a dot diagram $s^\epsilon(N) = s_y(\epsilon)$ of strata approaching N from below.
- The *target of N* is a dot diagram $t^\epsilon(N) = t_y(\epsilon)$ of strata approaching N from above.

We view $s^\epsilon(N)$ and $t^\epsilon(N)$ as being labeled in $\text{comp}^1(\delta)$. We label each stratum in the slice with itself, but thought of in δ .

Proposition. The dot diagrams $s^\epsilon(N)$ and $t^\epsilon(N)$ are independent of the choice of string pre-diagram ϵ focused on N .

Proof. Suppose N is located at (x_2, y_2) and suppose

$$\epsilon = \delta|_{[x_1, x_3] \times [y_1, y_3]} \quad \text{and} \quad \epsilon' = \delta|_{[x'_1, x'_3] \times [y'_1, y'_3]}$$

are string diagrams focused on N . Then the intersection

$$\epsilon \cap \epsilon' = \delta|_{[\max(x_1, x'_1), \min(x_3, x'_3)] \times [\max(y_1, y'_1), \min(y_3, y'_3)]}$$

is also a string diagram focused on N . We claim that in $\text{dd}(\text{comp}^1(\delta))$,

$$s^\epsilon(N) = s^{\epsilon \cap \epsilon'}(N) = s^{\epsilon'}(N),$$

and similarly for targets. This follows from,

$$\begin{aligned} s^\epsilon(N) &= \delta|_{[x_1, x_3] \times \{y_1\}} \\ &= \delta|_{[x_1, x_3] \times \{\max(y_1, y'_1)\}} \\ &= \delta|_{[\max(x_1, x'_1), \min(x_3, x'_3)] \times \{\max(y_1, y'_1)\}} \\ &= s^{\epsilon \cap \epsilon'}(N), \end{aligned}$$

where we use a dot evolution $\delta|_{[x_1, x_3] \times [y_1, \max(y_1, y'_1)]}$ and we horizontally glue dot-less dot diagrams $\delta|_{[x_1, \max(x_1, x'_1)] \times \{\max(y_1, y'_1)\}}$ and $\delta|_{[\min(x_3, x'_3), x_3] \times \{\max(y_1, y'_1)\}}$. \square

Definition. Any string pre-diagram δ has an *underlying 2-computad* $\text{comp}^2(\delta)$. This extends the 1-computad $\text{comp}^1(\delta)$ of region 0-generators and string 1-generators by including the nodes of δ as 2-generators.

Definition. A *string diagram* consists of a string pre-diagram δ on a square $[x_0, x_1] \times [y_0, y_1]$, a 2-computad G of labels, and a map of 2-computads $\text{comp}^2(\delta) \rightarrow G$. The faces $s_x(\delta)$, $t_x(\delta)$, $s_y(\delta)$, and $t_y(\delta)$ each inherit labels making them dot diagrams. Note that $s_x(\delta)$ and $t_x(\delta)$ are dot-less and so each only intersect a single region. We'll write $\delta: d \rightarrow e$ to mean $s_y(\delta) = d$ and $t_y(\delta) = e$.

Example. Every dot evolution δ on $[x_0, x_1] \times [t_0, t_1]$ is a node-less string diagram when we regard time, t , as height, y .

7.3 String Evolutions

Definition. A *string pre-evolution* is a finite stratification of a cube $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$ satisfying certain rules. Despite being thickened in time, we will still refer to the codimension 0 strata as *regions*, the codimension 1 strata as *strings*, and the codimension 2 strata as *nodes*.

- There are no strata of codimension 3.
- The left and right faces, $\{x_0\} \times [y_0, y_1] \times [t_0, t_1]$ and $\{x_1\} \times [y_0, y_1] \times [t_0, t_1]$, only intersect regions.
- The bottom and top faces, $[x_0, x_1] \times \{y_0\} \times [t_0, t_1]$ and $[x_0, x_1] \times \{y_1\} \times [t_0, t_1]$, only intersect regions and strings.
- At any point p , the tangent vectors $\pm\partial/\partial x|_p$ are not contained in any string.
- At any point p , the only tangent vector in $\langle \partial/\partial x|_p, \partial/\partial y|_p \rangle$ allowed to be in a node is the trivial vector, 0.

We define

- $s_x(\Delta) = \Delta|_{x=x_0}$ and $t_x(\Delta) = \Delta|_{x=x_1}$ are the string pre-diagrams on the left and right faces.
- $s_y(\Delta) = \Delta|_{y=y_0}$ and $t_y(\Delta) = \Delta|_{y=y_1}$ are the string pre-diagrams on the bottom and top faces.
- $s_t(\Delta) = \Delta|_{t=t_0}$ and $t_t(\Delta) = \Delta|_{t=t_1}$ are the string pre-diagrams on the initial and final faces.

Proposition. Suppose Δ is a string pre-evolution. For a fixed time $t_1 \in [t_0, t_2]$, the slice $\Delta|_{t=t_1}$ is a string pre-diagram.

Proof. We've chosen our definition of string pre-evolution so that the strata of Δ are transverse to any $t = t_1$ slice. Further, the boundary and progressivity rules of Δ give the appropriate boundary and progressivity rules for the slice. \square

Suppose Δ is a string pre-evolution. Just as with string pre-diagrams, for any stratum S we may crop Δ to produce a string pre-evolution Λ focused on S . That is, Λ only intersects strata incident to S . And we may choose Λ to be about any point p of S . The definitions of source and target of strata in a string pre-evolution are almost identical to those for a string pre-diagram.

Definition. Suppose Δ is a string pre-evolution and S is a string in Δ . Suppose d is a dot pre-diagram focused on S . We define the *source of S* to be $s^d(S) = s_x(S)$ and the *target of S* to be $t^d(S) = t_x(S)$, both thought of as regions in Δ .

Proposition. Suppose d and e are dot pre-diagrams focused on a string S in a string pre-evolution Δ . Then

$$s^d(S) = s^e(S) \quad \text{and} \quad t^d(S) = t^e(S).$$

Proof. The proof for string pre-diagrams also holds for string pre-evolutions after minor adjustments. \square

Definition. Any string pre-evolution Δ has an *underlying 1-computad* $\text{comp}^1(\Delta)$. We take the regions as 0-generators and strings as 1-generators.

Definition. Suppose N is a node in a string pre-evolution Δ . Suppose δ is a slice string pre-diagram focused on N . We define the *source of N* to be $s^\delta(N) = s_y(\delta)$ and the *target of N* to be $t^\delta(N) = t_y(\delta)$, both thought of as dot diagrams labeled by $\text{comp}^1(\Delta)$.

Proposition. Suppose Δ is a string pre-evolution and N is a node in Δ . Suppose δ and ϵ are string pre-diagrams focused on N . Then

$$s^\delta(N) = s^\epsilon(N) \quad \text{and} \quad t^\delta(N) = t^\epsilon(N).$$

Proof. If δ and ϵ happen to both intersect N at a point p , then $\delta \cap \epsilon$ is a string pre-diagram focused on N and we see

$$s_y(\delta) = s_y(\delta \cap \epsilon) = s_y(\epsilon) \quad \text{and} \quad t_y(\delta) = t_y(\delta \cap \epsilon) = t_y(\epsilon).$$

For arbitrary δ and ϵ , we take a path in N from $\delta \cap N$ to $\epsilon \cap N$ and cover it with string pre-evolutions focused on N . The source and target of N are preserved in each pre-evolution

focused on N . If two pre-evolutions focused on N overlap, then we see that source and target is preserved by considering a slice in the overlap. \square

Definition. Any string pre-evolution Δ has an *underlying 2-computad* $\text{comp}^2(\Delta)$ extending $\text{comp}^1(\Delta)$. We take the regions as 0-generators, the strings as 1-generators, and the nodes as 2-generators.

Definition. A *string evolution*, or an *evolution of string diagrams*, consists of a string pre-evolution Δ on a cube $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$, a 2-computad G , and a map of 2-computads $\text{comp}^2(\Delta) \rightarrow G$. We note that $s_x(\Delta)$ and $t_x(\Delta)$ only intersect regions. Also, $s_y(\Delta)$ may be viewed as a dot evolution $s_y(\delta) \rightarrow s_y(\epsilon)$ and $t_y(\Delta)$ may be viewed a dot evolution $t_y(\delta) \rightarrow t_y(\epsilon)$. We will write $\Delta: \delta \rightarrow \epsilon$ to mean $s_t(\Delta) = \delta$ and $t_t(\Delta) = \epsilon$.

Example. Any string diagram δ may be thickened to give a *constant string evolution* 1_δ .

Example. Any string evolution $\Delta: \delta \rightarrow \epsilon$ may be *reversed* to give a string evolution $\Delta^{\text{rev}}: \epsilon \rightarrow \delta$.

Proposition. Suppose Δ is a string evolution. Then for each fixed $t_1 \in [t_0, t_2]$, the slice $\Delta|_{t=t_1}$ inherits labels making it a string diagram.

7.4 Gluing and Normalization

In this section, we'll discuss gluing both for string diagrams and for string evolutions. In fact, it's a bit easier to split a diagram than it is to glue.

Definition. Suppose δ is a string diagram on $[x_0, x_2] \times [y_0, y_2]$.

- Suppose $\{x_1\} \times [y_0, y_2]$ only intersects regions for some point $x_1 \in (x_0, x_2)$. Then we may *horizontally split* δ at $x = x_1$. That is, δ restricts to string diagrams $\epsilon = \delta|_{[x_0, x_1] \times [y_0, y_2]}$ and $\zeta = \delta|_{[x_1, x_2] \times [y_0, y_2]}$. We may also say that δ is made by *horizontally gluing*:

$$\delta = \epsilon \cup_x \zeta.$$

- Suppose $[x_0, x_1] \times \{y_1\}$ only intersects regions and strings for some point $y_1 \in (y_0, y_2)$. Then we may *vertically split* δ at $y = y_1$ by restricting to string diagrams: $\epsilon = \delta|_{[x_0, x_2] \times [y_0, y_1]}$ and $\zeta = \delta|_{[x_0, x_2] \times [y_1, y_2]}$. We may also say that δ is made by *vertically gluing*:

$$\delta = \epsilon \cup_y \zeta.$$

Proposition. Suppose δ and ϵ are string diagrams satisfying $t_x(\delta) = s_x(\epsilon)$. Then we may glue δ and ϵ horizontally. That is, we may form $\delta \cup_x \epsilon$.

Proof. Suppose δ is located on $[x_0, x_1] \times [y_0, y_1]$ and ϵ is located on $[x_1, x_2] \times [y_0, y_1]$. Since $t_x(\delta)$ is dot-less, δ must agree with $[x_0, x_1] \times t_x(\delta)$ on some neighborhood of $\{x_1\} \times [y_0, y_1]$. Similarly, ϵ must agree with $[x_1, x_2] \times s_x(\epsilon)$ on some neighborhood of $\{x_1\} \times [y_0, y_1]$. These glue together in $[x_0, x_2] \times [y_0, y_1]$ to locally look like $\mathbb{R} \times [y_0, y_1]$. \square

Definition. Suppose δ is a string diagram defined on $[x_0, x_1] \times [y_0, y_1]$.

- We will say δ is *normal near s_y* if on some neighborhood of $[x_0, x_1] \times \{y_0\}$, δ agrees with the constant evolution $1_{s_y(\delta)} = s_y(\delta) \times [y_0, y_1]$.
- We will say δ is *normal near t_y* if on some neighborhood of $[x_0, x_1] \times \{y_1\}$, δ agrees with the constant evolution $1_{t_y(\delta)} = t_y(\delta) \times [y_0, y_1]$.

Proposition. Suppose δ and ϵ are string diagrams satisfying $t_y(\delta) = s_x(\epsilon)$. If δ is normal near t_y and ϵ is normal near s_y , then we may vertically glue δ and ϵ .

Proof. Suppose δ and ϵ are located on $[x_0, x_1] \times [y_0, y_1]$ and $[x_0, x_1] \times [y_1, y_2]$, respectively. We give $[x_0, x_1] \times [y_0, y_2]$ a stratification by taking all of the strata in δ and ϵ and gluing strata that intersect $[x_0, x_1] \times \{y_1\}$. The result of gluing two strata along $y = y_1$ is a stratum because near the gluing, the strata agree with a thickened $s(\delta)$. \square

Proposition (string diagram normalization). Suppose $\delta: d \rightarrow e$ is a string diagram. Then there is a string evolution Δ from δ to a normal string diagram $\delta': d \rightarrow e$. Further, we may take Δ so that $s_y(\Delta)$ is the constant evolution 1_d and $t_y(\Delta)$ is the constant evolution 1_e .

Proof. This proposition should be regarded as a strengthening of a similar result for dot evolutions. We build a normal replacement in the same manner, being careful to leave any nodes the range that stays stationary. We may also build a family of string diagrams that interpolate from the δ to the normal replacement by smoothly interpolating from the identity $[0, 5] \rightarrow [0, 5]$ to the normalization map $[0, 5] \rightarrow [0, 5]$. \square

There are three directions, x , y , and t , in which we may glue string evolutions. We define gluing in much the same way that we did for string diagrams. That is, two evolutions glue together if they are made by splitting a larger string evolution.

Definition. Suppose Δ is a string evolution on $[x_0, x_2] \times [y_0, y_2] \times [t_0, t_2]$.

- Suppose $\{x_1\} \times [y_0, y_2] \times [t_0, t_2]$ only intersects regions for some point $x_1 \in (x_0, x_2)$. Then we may *horizontally split* Δ at $x = x_1$ to produce string evolutions $\Lambda = \Delta|_{[x_0, x_1] \times [y_0, y_2] \times [t_0, t_2]}$ and $\Omega = \Delta|_{[x_1, x_2] \times [y_0, y_2] \times [t_0, t_2]}$. We may also say that Δ is made by *horizontally gluing*:

$$\Delta = \Lambda \cup_x \Omega.$$

- Suppose $[x_0, x_2] \times \{y_1\} \times [t_0, t_2]$ only intersects regions and strings for some point $y_1 \in (y_0, y_2)$. Then we may *vertically split* Δ at $y = y_1$ to produce string evolutions $\Lambda = \Delta|_{[x_0, x_2] \times [y_0, y_1] \times [t_0, t_2]}$ and $\Omega = \Delta|_{[x_0, x_2] \times [y_1, y_2] \times [t_0, t_2]}$. We may also say that Δ is made by *vertically gluing*:

$$\Delta = \Lambda \cup_y \Omega.$$

- Suppose t is any point in (t_0, t_2) . Then we may *split* Δ at time $t = t_1$ to produce string evolutions $\Lambda = \Delta|_{[x_0, x_2] \times [y_0, y_2] \times [t_0, t_1]}$ and $\Omega = \Delta|_{[x_0, x_2] \times [y_0, y_2] \times [t_1, t_2]}$. We may also say that Δ is made by *gluing in time*:

$$\Delta = \Lambda \cup_t \Omega.$$

If two string evolutions Δ and Ω agree on a common face, $t_x(\Delta) = s_x(\Omega)$, then we may horizontally glue to get a string evolution $\Delta \cup_x \Omega$. Indeed, since the only strata allowed to intersect these faces are regions, the faces are already normal. On the other hand, to glue string evolutions in other directions we will need to normalize near the s_y , t_y , s_t , and t_t faces.

7.5 The 2-Category $\text{sd}(G)$

Definition. For any 2-computad G , there is a 2-category $\text{sd}(G)$ of string diagrams labeled by G .

$$\begin{aligned} \text{Obj}_{\text{sd}(G)} &= G_0 \\ \text{Hom}_{\text{sd}(G)}(X, Y) &= \frac{\{\text{dot diagrams } d: X \rightarrow Y\}}{\text{translation, dilation, dot evolution}} \\ 2\text{-Hom}_{\text{sd}(G)}(d, e) &= \frac{\{\text{string diagrams } \delta: d \rightarrow e\}}{\text{translation, dilation, string evolution}} \end{aligned}$$

In order for this definition to make sense, we'll need to be able to compose string diagrams δ and ϵ vertically when $t_y(\delta)$ only agrees with $s_y(\epsilon)$ up to a dot evolution. We do not currently have such a definition, so we'll make one.

Definition. Suppose δ and ϵ are string diagrams and $\zeta: t_y(\delta) \rightarrow s_y(\epsilon)$ is a dot evolution. We define the *vertical composite of δ and ϵ over ζ* by taking normal replacements δ' , ζ' , and ϵ' and gluing:

$$\delta' \cup_y \zeta' \cup_y \epsilon'.$$

Proposition. Vertical composites of string diagrams over dot evolutions are unique up to string evolution.

Proof. Suppose δ_0 , ϵ_0 , δ_1 , and ϵ_1 are string diagrams and suppose $\zeta_0: t_y(\delta_0) \rightarrow s_y(\epsilon_0)$ and $\zeta_1: t_y(\delta_1) \rightarrow s_y(\epsilon_1)$ are dot evolutions. Assume

- δ_0 and δ_1 are normal near t_y ;
- ζ_0 and ζ_1 are normal near both s_y and t_y ;
- ϵ_0 and ϵ_1 are normal near s_y .

Suppose $\Delta: \delta_0 \rightarrow \delta_1$ and $\Omega: \epsilon_0 \rightarrow \epsilon_1$ are string evolutions. We'll show that there is a string evolution

$$\delta_0 \cup_y \zeta_0 \cup_y \epsilon_0 \longrightarrow \delta_1 \cup_y \zeta_1 \cup_y \epsilon_1.$$

We begin rechoosing Δ and Ω as Δ' and Ω' so that Δ' is normal near t_y and Ω' is normal near s_y . We may choose any string evolution Ψ with $s_y(\Psi) = t_y(\Delta')$, $t_y(\Psi) = s_y(\Omega')$, $s_t(\Psi) = \zeta_0$, and $t_t(\Psi) = \zeta_1$. We then rechoose Ψ as Ψ' so that it is normal near all of its faces. Then $\Delta' \cup_y \Psi' \cup_y \Omega'$ is the desired string evolution. \square

In the above, we should think of the dot evolution ζ as a degenerate string diagram. We may also define horizontal composition of string diagrams over an intermediate diagram. In this case, we should compose over string diagrams without nodes nor strings. We'll call such string diagrams *doubly degenerate*. Certainly, the compositions for $\text{sd}(G)$ are associative and satisfy the interchange rule. The degenerate string diagrams serve as units for vertical composition and the doubly degenerate string diagrams are units for horizontal composition.

To be very careful, we should check that our definition of composition is independent of dilating one of the factors. This is relatively easy, though, since dilating a normal replacement does not affect the normality. We should also check that if vertically composing is a generalization of gluing.

Proposition. Suppose δ and ϵ are vertically gluable string diagrams. Then in $\text{sd}(G)$,

$$\delta \circ_y \epsilon = \delta \cup_y \epsilon.$$

Proof. Let δ' and ϵ' be normal replacements for δ and ϵ . Using the normalization lemma, we may choose string evolutions $\Delta: \delta \rightarrow \delta'$ and $\Lambda: \epsilon \rightarrow \epsilon'$ that are gluable in the y -direction. So in $\text{sd}(G)$, we have

$$\begin{aligned} \delta \circ_y \epsilon &= \delta' \cup_y 1_{t_y(\delta)} \cup_y \epsilon' \\ &= \delta' \cup_y \epsilon' \\ &= \delta \cup_y \epsilon. \end{aligned}$$

□

7.6 Cone Diagrams

Suppose α is a 2-generator in a 2-computad G . We define the *cone string diagram* $\text{cone}_G(\alpha)$ on the square $[-1, 1] \times [-1, 1]$, as follows. We'll think of $s(\alpha)$ and $t(\alpha)$ as dot diagrams defined on $[-1, 1] \times \{-1\}$ and $[-1, 1] \times \{1\}$, respectively. Radial projection allows us to view the each of the strata in $s(\alpha)$ on the lower quarter of the circle,

$$\left\{ \frac{5}{8} \cdot 2\pi \leq \theta \leq \frac{7}{8} \cdot 2\pi \right\} \subset \mathbb{S}^1.$$

Similarly, we may view the strata of $t(\alpha)$ on the upper quarter of the circle,

$$\left\{ \frac{1}{8} \cdot 2\pi \leq \theta \leq \frac{3}{8} \cdot 2\pi \right\} \subset \mathbb{S}^1.$$

We glue the left-most region of $s(\alpha)$ to the left-most region of $t(\alpha)$ using the left quarter of the circle,

$$\left\{ \frac{3}{8} \cdot 2\pi \leq \theta \leq \frac{5}{8} \cdot 2\pi \right\} \subset \mathbb{S}^1.$$

We also glue the right-most region of $s(\alpha)$ to the right-most region of $t(\alpha)$ using the right quarter of the circle,

$$\left\{ -\frac{1}{8} \cdot 2\pi \leq \theta \leq \frac{1}{8} \cdot 2\pi \right\} \subset \mathbb{S}^1.$$

This gives \mathbb{S}^1 a stratification. We cone to get a stratification of the plane \mathbb{R}^2 , which we then restrict to the desired stratification of the square, $[-1, 1] \times [-1, 1]$. Note that $\text{cone}_G(\alpha)$ extends the stratifications we gave to the faces of the square. We label the strata according to their labels in $s(\alpha)$ and $t(\alpha)$. By globularity of α , the source 0-generator of α labels the left face $\{-1\} \times [-1, 1]$ and the target 0-generator of α labels the right face $\{1\} \times [-1, 1]$.

The origin is the only stratum in $\text{cone}_G(\alpha)$ which doesn't correspond to a stratum in either $s(\alpha)$ or $t(\alpha)$. We label the origin with α . Note that all strata in $\text{cone}_G(\alpha)$ are incident to the origin. In the language of the string pre-diagrams section, may say that $\text{cone}_G(\alpha)$ is focused on the origin. Our construction, therefore, has produced a string diagram in which the source and target dot diagrams of the origin node agree with the source and target of α as a 2-generator in G .

7.7 Evaluation

In this section, we will use brick decompositions to evaluate strings diagrams labeled in a 2-category. As usual, we'll see that every string diagram has a many brick decompositions and that evaluation is independent of which we choose to equip the diagram with.

Definition. A string diagram δ is a *brick* if it is focused on a stratum. That is, all strata in δ are incident to some stratum S .

Definition. Suppose δ is a string diagram on $[x_0, x_1] \times [y_0, y_1]$. A *y-layer of bricks* for δ consists of a finite set $M_x \subset [x_0, x_1]$ of *x-mortar points*. We require the following.

- $x_0, x_1 \in M_x$.
- For each *x-mortar point* $p \in M_x$, the set $\{p\} \times [y_0, y_1]$ only intersects a region of δ .
- Between successive *x-mortar points* $p < q$, the string diagram $\delta|_{[p,q] \times [y_0,y_1]}$ is a brick.

Definition. Suppose δ is a string diagram on $[x_0, x_1] \times [y_0, y_1]$. A *brick wall* or *brick decomposition* for δ consists of a finite set $M_y \subset [y_0, y_1]$ of *y-mortar points*. We require the following.

- $y_0, y_1 \in M_y$.
- For each *y-mortar point* $p \in M_y$, the set $[x_0, x_1] \times \{p\}$ only intersects regions and strings of δ .
- Between successive *y-mortar points* $p < q$, the string diagram $\delta|_{[x_0,x_1] \times [p,q]}$ has a *y-layer of bricks*. We call the mortar set $M_x^{p,q}$.

Given two brick decompositions, we may find a common refinement by intersecting their bricks. Roughly speaking, this amounts to taking unions of their mortar sets.

Definition. Suppose δ as above has two brick decompositions, $D = (M_y, M_x^{*,*})$ and $E = (N_y, N_x^{*,*})$. Then there is a brick decomposition $D \cap E$ defined as follows.

- The y -mortar set for $D \cap E$ is the union $M_y \cup N_y$.
- Suppose $r < s$ are successive y -mortar points in $M_y \cup N_y$. We take the y -mortar points $r_M, s_M \in M_y$ so that r_M and s_M are successive in M_y and $[r, s] \subset [r_M, s_M]$. Similarly take successive $r_N, s_N \in N_y$ about $[r, s]$. Then the x -mortar set for $M_y \cup N_y$ between r and s is $M_x^{r_M, s_M} \cup N_x^{r_N, s_N}$.

Proposition. We may build the refinement $D \cap E$ by starting with D and using the following two operations repeatedly.

- We may split a brick into two by adding an extra x -mortar point.
- We may split a y -layer of bricks into two by adding an extra y -mortar point.

Proof. We begin by splitting each of the rows of bricks D vertically turn the y -mortar set M_y into $M_y \cup N_y$. For each x -mortar point of E , we split bricks of $D \cap E$ horizontally in the appropriate y -layers. \square

Proposition. Every string diagram has a brick decomposition.

Proof. We must be able to separate strings from strings, strings from nodes, and nodes from nodes.

- We may always separate two strings with x -mortar points by choosing enough y -mortar points.
- By choosing y -mortar points just above and just below a node, we may separate it from non-incident strings using x -mortar points.
- If two nodes are at distinct y -heights, we may choose a y -mortar point between them.
- If two nodes share a y -height, we may choose y -mortar points just above and below the nodes and an x -mortar point between them.

\square

Definition. Suppose δ is a string diagram equipped with a brick decomposition D . Suppose further that δ is labeled by a 2-category C . We define the 2-morphism in C

$$\text{eval}_C(s(\delta)) \xrightarrow{\text{eval}_C^D(\delta)} \text{eval}_C(t(\delta))$$

as follows.

- If δ is a brick focused on a region that is labeled by an object X , then we define

$$\text{eval}_C^D(\delta) = 1_{1_X}.$$

- If δ is a brick focused on a string that is labeled by 1-morphism a , then we define

$$\text{eval}_C^D(\delta) = 1_a.$$

- If δ is a brick focused on a node that is labeled by a 2-morphism α , then we define

$$\text{eval}_C^D(\delta) = \alpha.$$

- If δ is a y -layer of bricks B_1, \dots, B_n , we define

$$\text{eval}_C^D(\delta) = \text{eval}_C^D(B_1) \circ_x \dots \circ_x \text{eval}_C^D(B_n).$$

- For general δ made of y -layers $L_1 \dots, L_m$, we define

$$\text{eval}_C^D(\delta) = \text{eval}_C^D(L_1) \circ_y \dots \circ_y \text{eval}_C^D(L_m).$$

Proposition. Evaluation does not depend on the brick decomposition.

Proof. We'll show that refining a brick decomposition does not change what the string diagram evaluates to. Since any two brick decompositions have a common refinement, we'll be done.

splitting a brick horizontally Suppose δ is a brick which is focused on a stratum S .

Splitting the brick horizontally, we get two bricks $\delta = \delta_l \cup_x \delta_r$, one must be focused on S and the other must be focused on a region. Suppose, δ_l is focused on S and δ_r is focused on a region labeled by an object X . Then

$$\text{eval}_C(\delta) = \text{eval}_C(\delta_l) = \text{eval}_C(\delta_l) \circ_x 1_{1_X} = \text{eval}_C(\delta_l) \circ_x \text{eval}_C(\delta_r).$$

splitting a y -layer of bricks vertically We prove this case by induction on the number of bricks in a y -layer.

- Suppose δ is a y -layer of bricks consisting of a single brick. Splitting vertically, we get lower and upper bricks: $\delta = \delta_l \cup_y \delta_u$. One of these bricks, say δ_u , must be a node-less string diagram. So $\text{eval}_C(\delta_u) = 1_{\text{eval}_C(s(\delta_u))}$ and hence

$$\text{eval}_C(\delta) = \text{eval}_C(\delta_l) \circ_y 1_{\text{eval}_C(t(\delta_l))} = \text{eval}_C(\delta_l) \circ_y \text{eval}_C(\delta_u).$$

- Suppose δ is a y -layer of bricks consisting of two bricks ϵ and ζ . Splitting δ gives four bricks. This comes down to the interchange rule in C .

$$\begin{aligned}
& \text{eval}_C(\epsilon) \circ_x \text{eval}_C(\zeta) \\
&= \text{eval}_C(\epsilon_l \cup_y \epsilon_u) \circ_x \text{eval}_C(\zeta_l \cup_y \zeta_u) \\
&= (\text{eval}_C(\epsilon_l) \circ_y \text{eval}_C(\epsilon_u)) \circ_x (\text{eval}_C(\zeta_l) \circ_y \text{eval}_C(\zeta_u)) \\
&= (\text{eval}_C(\epsilon_l) \circ_x \text{eval}_C(\zeta_l)) \circ_y (\text{eval}_C(\epsilon_u) \circ_x \text{eval}_C(\zeta_u)) \\
&= (\text{eval}_C(\epsilon_l \cup_x \zeta_l)) \circ_y (\text{eval}_C(\epsilon_u \cup_x \zeta_u)).
\end{aligned}$$

- For a layer with more than two bricks, we inductively apply the interchange rule.

□

We also wish to show that evaluation is independent of string evolution. To do this, we'll extend our definition of brick decomposition to string evolutions.

Definition. A string evolution Δ is a *brick* if it is focused on a single region.

Definition. Suppose Δ is a string evolution on $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$. A *y -layer of bricks* for Δ consists of a finite set $M_x \subset [x_0, x_1]$ of *x -mortar points*. We require the following.

- $x_0, x_1 \in M_x$.
- For each x -mortar point $p \in M_x$, the set $\{p\} \times [y_0, y_1] \times [t_0, t_1]$ only intersects a region of Δ .
- Between successive x -mortar points $p < q$, the string evolution $\Delta|_{[p,q] \times [y_0,y_1] \times [t_0,t_1]}$ is a brick.

Definition. Suppose Δ is a string evolution on $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$. A *t -layer of bricks* for Δ consists of a finite set $M_y \subset [y_0, y_1]$ of *y -mortar points*. We require the following.

- $y_0, y_1 \in M_y$.
- For each y -mortar point $p \in M_y$, the set $[x_0, x_1] \times \{p\} \times [t_0, t_1]$ only intersects regions and strings of Δ .
- Between successive y -mortar points $p < q$, the string evolution $\Delta|_{[x_0,x_1] \times [p,q] \times [t_0,t_1]}$ has a y -layer of bricks. We call the mortar set $M_x^{p,q}$.

Definition. Suppose Δ is a string evolution on $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$. A *brick decomposition* for Δ consists of a finite set $M_t \subset [t_0, t_1]$ of *t-mortar points*. We require the following.

- $t_0, t_1 \in M_t$.
- Between successive *t-mortar points* $p < q$, the string evolution $\Delta|_{[x_0, x_1] \times [y_0, y_1] \times [p, q]}$ has a *t-layer* of bricks. We call the mortar set $M_y^{p, q}$.

Proposition. Every string evolution has a brick decomposition.

Proof. Suppose Δ is a string evolution on $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_2]$ and let $t_1 \in [t_0, t_2]$ be any time. Then we may equip the slice $\Delta|_{t=t_1}$ with a brick decomposition for a string diagram. We may thicken this to a *t-layer*, at least for small $|t - t_1|$. In this manner, we may cover $[t_0, t_2]$ and, by compactness, we need only finitely such overlapping *t-layers*. By shrinking the *t-layers* in time to remove overlap, we produce a brick decomposition for Δ . \square

Proposition. Suppose δ and ϵ are string diagrams labeled in a 2-category C . Suppose further that δ and ϵ are related by a string evolution Δ . Then

$$\text{eval}_C(\delta) = \text{eval}_C(\epsilon).$$

Proof. Suppose Δ is located on $[x_0, x_1] \times [y_0, y_1] \times [t_0, t_1]$. We choose any brick decomposition for Δ . It is clear that evaluation is preserved across time for each brick. So evaluation is preserved across time for each *y-layer* of bricks. And hence evaluation is preserved across time for each *t-layer* of bricks. \square

7.8 Characterizing $\text{sd}(G)$ as the Free 2-Category

Let's denote the category of 2-computads by 2-Cmp and let's denote the category of small 2-categories and strict 2-functors by 2-Cat . We have made associations

$$\begin{aligned} \text{forget}: 2\text{-Cat} &\longrightarrow 2\text{-Cmp} \\ \text{sd}: 2\text{-Cmp} &\longrightarrow 2\text{-Cat} \end{aligned}$$

which are easily seen to be functors.

Definition. For any 2-computad G , there is a natural map of 2-computads, called *conical inclusion*,

$$\text{cone}_G: G \longrightarrow \text{forget}(\text{sd}(G)).$$

A 0-generator in G can be regarded as an object of $\text{sd}(G)$. A degree 1 generator is sent to its cone dot diagram and a degree 2 generator is sent to its cone string diagram.

Definition. For any 2-category C , there is a natural 2-functor, called *evaluation*,

$$\text{eval}_C: \text{sd}(\text{forget}(C)) \longrightarrow C.$$

This 2-functor was described in the previous section.

The following two lemmas show that inclusion and evaluation are the unit and counit of an adjunction.

Lemma (Zig-Zag). The composite of the inclusion and evaluation maps

$$\text{forget}(C) \xrightarrow{\text{cone}_{\text{forget}(C)}} \text{forget}(\text{sd}(\text{forget}(C))) \xrightarrow{\text{forget}(\text{eval}_C)} \text{forget}(C)$$

equals the identity on $\text{forget}(C)$.

Proof. The first half of this composition takes a 2-morphism of C and views it as a cone string diagram. The second half evaluates this string diagram, recovering the original 2-morphism. \square

Lemma (Zig-Zag). The composite of the inclusion and evaluation maps

$$\text{sd}(G) \xrightarrow{\text{sd}(\text{cone}_G)} \text{sd}(\text{forget}(\text{sd}(G))) \xrightarrow{\text{eval}_{\text{sd}(G)}} \text{sd}(G)$$

equals the identity on $\text{sd}(G)$.

Proof. This composition $n: \text{sd}(G) \longrightarrow \text{sd}(G)$ takes a string diagram δ labeled in G , breaks it up into bricks according to a brick decomposition, and then composes those bricks. Any individual brick is unchanged by n , at least up to evolution. Further, since horizontal composition may be enacted by gluing, a y -layer of bricks is unchanged by n . Last, we showed that vertical composition may be enacted by vertical gluing, at least when the factors are gluable. Each y -layer of bricks is made by splitting δ and so these are certainly gluable. \square

Theorem. Suppose G is a 2-computad. The 2-category $\text{sd}(G)$ of string diagrams is the free 2-category on G .

Proof. This is a formal consequence of the Zig-Zag lemmas. \square

Chapter 8

Surface Diagrams

8.1 3-Computads

Definition. A *3-computad* G consists of four sets of generators, G_0 , G_1 , G_2 , and G_3 . We require G have source and target maps

$$s, t: G_1 \longrightarrow G_0.$$

We may now speak about the category $\text{dd}(G)$ of dot diagrams. In particular, $\text{dd}(G)_1$ is the set of dot diagrams, taken up to dot evolution. Extending the source and target maps as usual, we get maps $\text{dd}(G)_1 \longrightarrow \text{dd}(G)_0 = G_0$. We require that G have globular source and target maps

$$s, t: G_2 \longrightarrow \text{dd}(G)_1.$$

This allows us to now speak about the 2-category $\text{sd}(G)$ of string diagrams. Note that $\text{sd}(G)_2$ is the set of string diagrams, taken up to string evolution. We get source and target maps $\text{sd}(G)_2 \rightarrow \text{sd}(G)_1 = \text{dd}(G)_1$. We require that G have globular source and target maps

$$s, t: G_3 \longrightarrow \text{sd}(G)_2.$$

A *map of 3-computads* $f: G \longrightarrow H$ is a quadruple of maps $f_i: G_i \longrightarrow H_i$ that commute with the source and target maps.

Example. Every 2-computad is a 3-computad without 3-generators.

Example. A 3-category C has an underlying 3-computad, $\text{forget}(C)$.

$$\begin{aligned}\text{forget}(C)_0 &= \text{Obj}_C, \\ \text{forget}(C)_1 &= \coprod_{X, Y \in \text{Obj}_C} \text{Hom}_C(X, Y), \\ \text{forget}(C)_2 &= \coprod_{a, b \in \text{dd}(\text{forget}(C))_1} \{(a, b, \alpha) : \alpha \in 2\text{-Hom}_C(\text{eval}(a), \text{eval}(b))\}, \\ \text{forget}(C)_3 &= \coprod_{\alpha, \beta \in \text{sd}(\text{forget}(C))_2} \{(\alpha, \beta, \Gamma) : \Gamma \in 3\text{-Hom}_C(\text{eval}(\alpha), \text{eval}(\beta))\}.\end{aligned}$$

8.2 Surface Diagrams

Definition. A *surface pre-diagram* Δ is a stratified cube $[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1]$ with finitely many strata. We'll call the codimension 0 strata *regions*, the codimension 1 strata *walls*, the codimension 2 strata *seams*, and the codimension 3 strata *nodes*. We require the following rules.

- The x -faces $\{x_0, x_1\} \times [y_0, y_1] \times [z_0, z_1]$ only intersect regions.
- The y -faces $[x_0, x_1] \times \{y_0, y_1\} \times [z_0, z_1]$ only intersect regions and walls.
- At any point p , the tangent vectors $\pm \partial/\partial x|_p$ are not contained in any wall.
- At any point p , the only tangent vector in $\langle \partial/\partial x|_p, \partial/\partial y|_p \rangle$ allowed to be in a seam is the trivial vector.

We define the usual $s_x, t_x, s_y, t_y, s_z, t_z$. We note that the z -faces are string pre-diagrams, the y -faces are degenerate string pre-diagrams, and the x -faces are doubly degenerate string pre-diagrams.

We have defined a surface pre-diagram Δ so that almost every z -slice is a string pre-diagram. Indeed, we need only miss z -coordinates containing nodes in order to ensure that a z -slice is transverse to all of the strata. We define source and targets for a stratum in a surface pre-diagram by using slice pre-diagram focused on the stratum in question. We get uniqueness for these definitions by covering strata with focused surface diagrams made by cropping Δ .

Definition. The source and target of a stratum S in a surface pre-diagram Δ are defined as follows.

- If S is a wall, we take the source s_x and target t_x in any yz -slice dot pre-diagram d focused on S .
- If S is a seam, we take the source s_y and target t_y in any z -slice string pre-diagram δ focused on S .
- If S is a node, we take the source s_z and target t_z in any surface pre-diagram Λ focused on S .

Definition. Any surface pre-diagram Δ has an *underlying 3-computad* $\text{comp}^3(\Delta)$. We take the regions as 0-generators, the walls as 1-generators, the seams as 2-generators, and the nodes as 3-generators.

Definition. A *surface diagram* consists of a surface pre-diagram Δ , a 3-computad of labels G , and a map of 3-computads, $\text{comp}(\Delta) \rightarrow G$. We write $\Delta: \delta \rightarrow \epsilon$ to mean $s_z(\Delta) = \delta$ and $t_z(\Delta) = \epsilon$.

8.3 Surface Evolutions

Definition. A *surface pre-evolution* Υ is a finite stratification of a 4-cube

$$[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1] \times [t_0, t_1]$$

subject to rules. As usual, we retain the naming conventions for strata from surface diagrams despite the strata being thickened in time.

- There are no strata of codimension 4.
- The x -faces, $\{x_0, x_1\} \times [y_0, y_1] \times [z_0, z_1] \times [t_0, t_1]$, only intersect regions.
- The y -faces, $[x_0, x_1] \times \{y_0, y_1\} \times [z_0, z_1] \times [t_0, t_1]$, only intersect regions and walls.
- The z -faces, $[x_0, x_1] \times [y_0, y_1] \times \{z_0, z_1\} \times [t_0, t_1]$, only intersect regions, walls, and seams.
- At any point p , the tangent vectors $\pm\partial/\partial x|_p$ are not contained in any wall.
- At any point p , the only tangent vector in $\langle\partial/\partial x|_p, \partial/\partial y|_p\rangle$ allowed to be in a seam is the trivial vector.
- At any point p , the only tangent vector in $\langle\partial/\partial x|_p, \partial/\partial y|_p, \partial/\partial z|_p\rangle$ allowed to be in a node is the trivial vector.

Suppose \mathcal{T} is a surface pre-evolution on $[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1] \times [t_0, t_2]$. For any fixed $t_1 \in [t_0, t_2]$, we see that the slice $\mathcal{T}|_{t_1}$ is a surface pre-diagram. Further, the computadic structure of the slices is preserved across time. Indeed, any surface pre-diagram focused on a stratum S in a slice may be thickened in time to be a surface pre-evolution focused on S , at least for some small time interval. During this time, the computadic structure is fixed. We may need to use another surface pre-evolution focused on S to continue the argument. This is fine, since any overlapping surface pre-evolution focused on S must agree on the computadic structure for S in the overlap.

Definition. Any surface pre-evolution \mathcal{T} has an *underlying 3-computad* $\text{comp}^3(\mathcal{T})$ in which the regions are 0-generators, the walls are 1-generators, the seams are 2-generators, and the nodes are 3-generators.

Definition. A *surface evolution* consists of a surface pre-evolution \mathcal{T} , a 3-computad G , and a map of 3-computads, $\text{comp}^3(\mathcal{T}) \rightarrow G$. $s_t(\mathcal{T}) = \Delta$ and $t_t(\mathcal{T}) = \Omega$. We note that

- $s_z(\mathcal{T})$ and $t_z(\mathcal{T})$ are degenerate surface diagrams,
- $s_y(\mathcal{T})$ and $t_y(\mathcal{T})$ are doubly degenerate surface diagrams, and
- $s_x(\mathcal{T})$ and $t_x(\mathcal{T})$ are triply degenerate surface diagrams.

Example. Any surface diagram Δ may be thickened in time to give a constant surface evolution, $1_\Delta = \Delta \times [t_0, t_1]$.

Example. Any surface evolution $\mathcal{T}: \Delta \rightarrow \Omega$ has a reverse surface evolution $\mathcal{T}^{\text{rev}}: \Omega \rightarrow \Delta$ made by reversing time.

8.4 The 3-Category $\text{Sd}(G)$

We may glue surface diagrams in the x , y , and z -directions if two diagrams have a face in common. Of course, we typically need to take normal replacements in order to ensure that we may glue strata along the face. The x -faces only intersect regions and so these strata already meet the x -faces in a normal manner. Gluing will allow us to define the composition for a 3-category of surface diagrams.

Definition. For any 3-computad G , there is a 3-category $\text{Sd}(G)$ of surface diagrams labeled by G .

$$\begin{aligned} \text{Obj}_{\text{Sd}(G)} &= G_0 \\ \text{Hom}_{\text{Sd}(G)}(X, Y) &= \frac{\{\text{dot diagrams } d: X \rightarrow Y\}}{\text{translation, dilation, dot evolution}} \\ 2\text{-Hom}_{\text{Sd}(G)}(d, e) &= \frac{\{\text{string diagrams } \delta: d \rightarrow e\}}{\text{translation, dilation, string evolution}} \\ 3\text{-Hom}_{\text{Sd}(G)}(\delta, \epsilon) &= \frac{\{\text{surface diagrams } \Delta: \delta \rightarrow \epsilon\}}{\text{translation, dilation, surface evolution}} \end{aligned}$$

We note that the by ignoring the surface diagrams, we recover the 2-category $\text{sd}(G)$ of string diagrams. It is therefore our job now to give compatible composition for the surface diagrams. Composition in the x -direction is easy. We may compose by gluing, or if we prefer, we may glue over triply degenerate surface diagrams.

We must work harder to show that there are meaningful compositions in the y and z -directions. First we note that while surface diagrams are not globular, they become globular in $\text{Sd}(G)$. That is, for any surface diagram Δ , the dot diagrams $s(s(\Delta))$ and $s(t(\Delta))$ agree up to evolution, and so therefore sit in the same equivalence class. The same goes for $t(s(\Delta))$ and $t(t(\Delta))$.

Definition. Suppose Δ and Ω are surface diagrams and $\Psi: t_z(\Delta) \rightarrow s_z(\Omega)$ is a string evolution. Then we may *compose Δ and Ω over Ψ* by taking normal replacements, Δ' , Λ' and Ω' , and gluing:

$$\Delta' \cup_z \Lambda' \cup_z \Omega'.$$

Proposition. Composing surface diagrams in the z -direction over string evolutions is unique up to surface evolution.

Proof. Suppose $\Delta_0, \Omega_0, \Delta_1$, and Ω_1 are surface diagrams and suppose $\Psi_0: t_z(\Delta_0) \rightarrow s_z(\Omega_0)$ and $\Psi_1: t_z(\Delta_1) \rightarrow s_z(\Omega_1)$ are string evolutions. Assume

- Δ_0 and Δ_1 are normal near t_z ;
- Ψ_0 and Ψ_1 are normal near both s_z and t_z ;
- Ω_0 and Ω_1 are normal near s_z .

Suppose $A: \Delta_0 \rightarrow \Delta_1$ and $B: \Omega_0 \rightarrow \Omega_1$ are surface evolutions. We'll show that there is a surface evolution

$$\Delta_0 \cup_z \Psi_0 \cup_z \Omega_0 \longrightarrow \Delta_1 \cup_z \Psi_1 \cup_z \Omega_1.$$

We rechoose A and B as A' and B' so that A' is normal near t_z and B' is normal near s_z . We may choose a surface evolution D with the following string evolutions as faces: $s_z(D) = t_z(A)$, $t_z(D) = s_z(B)$, $s_t(D) = \Psi_0$, and $t_t(D) = \Psi_1$. We rechoose D as D' so that it is normal near all of its faces. Then $A' \cup_z D' \cup_z B'$ is the desired surface evolution. \square

We make a similar definition for composing in the y -direction.

Definition. Suppose Δ and Ω are surface diagrams and $\Psi: t_y(\Delta) \rightarrow s_y(\Omega)$ is a string evolution. Then we may *compose Δ and Ω over Ψ* by taking normal replacements, Δ' , Λ' and Ω' , and gluing:

$$\Delta' \cup_y \Lambda' \cup_y \Omega'.$$

We note that, if anything, composition in the y -direction is simpler than composition in the z -direction. They y -faces are degenerate string diagrams and so the string evolution Ψ is a doubly degenerate surface diagram. We may show that composition in the y -direction is unique up to surface evolution using the same argument as we used in the z -direction.

8.5 Cone Diagrams

Suppose Γ is a 3-generator in a 3-computad G . We'll define the *cone surface diagram* $\text{cone}_G(\Gamma)$ for Γ . We begin by setting up a bijective correspondence of points between the boundary of the cube $[-1, 1]^3$ and the unite sphere \mathbb{S}^2 by radially projecting. On each of the faces of the cube, this correspondence is smooth. We give the sphere a stratification by placing stratifications on the faces of the cube and observing their image under the projection.

- We locate $s(\Gamma)$ on $[x_0, x_1] \times [y_0, y_1] \times \{z_0\}$.
- We locate $t(\Gamma)$ on $[x_0, x_1] \times [y_0, y_1] \times \{z_1\}$.
- We required $s(s(\Gamma))$ and $s(t(\Gamma))$ be related by a dot evolution. We choose a dot evolution for $[x_0, x_1] \times \{y_0\} \times [z_0, z_1]$ which will glue with the adjacent strata on the top and bottom faces in projection.
- We required $t(s(\Gamma))$ and $t(t(\Gamma))$ be related by a dot evolution. We choose a dot evolution for $[x_0, x_1] \times \{y_1\} \times [z_0, z_1]$ which will glue with the adjacent strata on the top and bottom faces in projection.
- We give $\{x_0\} \times [y_0, y_1] \times [z_0, z_1]$ a single region and no other strata.

- We give $\{x_1\} \times [y_0, y_1] \times [z_0, z_1]$ a single region and no other strata.

Notice that we must carefully choose the dot evolutions for the left and right faces. Otherwise, the 1-strata for the top and bottom faces would not meet the 1-strata for the left and right faces in a smooth manner on the sphere. We give \mathbb{R}^3 the cone stratification for our stratified \mathbb{S}^2 . When we restrict the stratification for $\text{cone}_G(\Gamma)$ by restricting this to the cube, $[-1, 1]^3$. We give the strata labels according the labels on $s(\Gamma)$ and $t(\Gamma)$. We label the node located at the origin with Γ . This makes $\text{cone}_G(\Gamma)$ a surface diagram focused on the origin node.

8.6 Evaluation

We evaluate a surface diagram labeled in a 3-category by equipping the diagram with a brick decomposition.

Definition. A surface diagram Δ is a *brick* if it is focused on a stratum.

Definition. Suppose Δ is a surface diagram on $[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1]$. A *y-layer of bricks* for Δ consists of a finite set $M_x \subset [x_0, x_1]$ of *x-mortar points*. We require:

- $x_0, x_1 \in M_x$.
- For each *x-mortar point* $p \in M_x$, the set $\{p\} \times [y_0, y_1] \times [z_0, z_1]$ only intersects a region of Δ .
- Between successive *x-mortar points* $p < q$, the string diagram $\Delta|_{[p,q] \times [y_0, y_1] \times [z_0, z_1]}$ is a brick.

We define a *z-layer of bricks* for a surface diagram to consist of a set of *y-mortar points*. Between any two successive *y-mortar points* is a *y-layer of bricks*. Further, a *brick decomposition* for a surface diagram is a choice of *z-mortar points*. Between successive *z-mortar points* is a *z-layer of bricks*. Every surface diagram may be given a brick decomposition. By choosing enough *z-mortar points*, we may ensure that no seam or wall moves too much in the *x* or *y*-directions in any one *z*-layer. In each *z*-layer, we choose enough *y-mortar points* so that seams do not move too much in the *x*-direction. This allows us to choose *x-mortar points* giving bricks.

We evaluate a surface diagram labeled in a 3-category by using a brick decomposition. That is, we always evaluate a diagram as a *z*-composite of *y*-composites of *x*-composites.

We note that any two brick decompositions have a common refinement and that evaluation is fixed under refinement. We may also extend brick decompositions to evolution of surface diagrams by having t -layers of bricks. This allows one to see that evaluation of surface diagrams is fixed under evolution, as well.

8.7 Characterizing $\text{Sd}(G)$ as the Free 3-Category

We've defined

$$\begin{aligned} \text{forget} &: \mathbf{3}\text{-Cat} \longrightarrow \mathbf{3}\text{-Cmp} \\ \text{Sd} &: \mathbf{3}\text{-Cmp} \longrightarrow \mathbf{3}\text{-Cat} \end{aligned}$$

which are easily seen to be functors. We also have a natural conical inclusion map,

$$\text{cone}_G: G \longrightarrow \text{forget}(\text{Sd}(G)),$$

and a natural evaluation functor,

$$\text{eval}_C: \text{Sd}(\text{forget}(C)) \longrightarrow C.$$

Lemma (Zig-Zag). The composite of the inclusion and evaluation maps

$$\text{forget}(C) \xrightarrow{\text{cone}_{\text{forget}(C)}} \text{forget}(\text{Sd}(\text{forget}(C))) \xrightarrow{\text{forget}(\text{eval}_C)} \text{forget}(C)$$

equals the identity on $\text{forget}(C)$.

Proof. For any 3-morphism Γ in C ,

$$\text{eval}_C(\text{cone}_{\text{forget}(C)}(G)) = G.$$

□

Lemma (Zig-Zag). The composition of the inclusion and evaluation maps

$$\text{Sd}(G) \xrightarrow{\text{Sd}(\text{cone}_G)} \text{Sd}(\text{forget}(\text{Sd}(G))) \xrightarrow{\text{eval}_{\text{Sd}(G)}} \text{Sd}(G)$$

equals the identity on $\text{Sd}(G)$.

Proof. This is clear on bricks. We get the result for y -layers of bricks, z -layers of bricks, and then composites for a full brick decomposition by noting that each composition may be enacted by the appropriate direction of gluing in $\text{Sd}(G)$. □

Theorem. Suppose G is a 3-computad. Then the 3-category of surface diagrams $\text{Sd}(G)$ is the free 3-category on G .

Proof. This follows from the Zig-Zag lemmas. □

Chapter 9

Surface Diagrams for Gray-Categories

9.1 The Free Sesquicategory of Ordered String Diagrams

We're interested in defining surface diagrams for Gray-categories. If we truncate a Gray-category, leaving the objects, 1-morphisms, and 2-morphisms, we do not get a 2-category. Instead, these morphisms form a sesquicategory. See Appendix C for definitions of sesquicategory and Gray-category. In this section, we'll show that ordered string diagrams are the correct diagrams for describing compositions in a sesquicategory. This will be an important tool for our next section on surface diagrams for Gray-categories.

Definition. An *ordered string diagram* δ is a string diagram in which each node is located at a distinct y -height.

We think about the set of nodes getting an order according to their y -height. When we vertically glue two ordered string diagrams, the result is also an ordered string diagram. However, the same cannot be said for horizontal gluing. A node in one of the factors may be at the same y -height as a node in the other. Instead, we'll be satisfied with whiskering an ordered string diagram with a dot diagram.

Definition. Suppose $\delta: d \rightarrow e$ is an ordered string diagram and suppose f is a dot diagram whose source 0-generator matches the target 0-generator of δ . We define the *whiskering of δ by f on the right* as

$$\delta \circ_x f = \delta \cup 1_f,$$

where 1_f is the constant dot evolution on f .

We similarly define whiskering by dot diagrams on the left. Notice that since 1_f lacks nodes, $\delta \circ_x f$ is an ordered string diagram. We expect ordered string diagrams to preserve their ordering of nodes across evolution.

Definition. An *ordered string evolution* is a string evolution in which the slice string diagrams are ordered at every time.

We may compose ordered string evolutions in the y -direction and in time. We may also whisker ordered string evolutions with dot diagrams. This amounts to gluing on a doubly degenerate string evolution in the x -direction. It's easy to check that the collection of ordered string diagrams form a sesquicategory.

Definition. For any 2-computad G , there is a sesquicategory $\text{sd}^{\text{ord}}(G)$ of ordered string diagrams labeled by G .

$$\begin{aligned} \text{Obj} &= G_0 \\ \text{Hom}(X, Y) &= \frac{\{\text{dot diagrams } d: X \rightarrow Y\}}{\text{translation, dilation, dot evolution}} \\ \text{2-Hom}(d, e) &= \frac{\{\text{ordered string diagrams } \delta: d \rightarrow e\}}{\text{translation, dilation, ordered string evolution}} \end{aligned}$$

We wish to show that $\text{sd}^{\text{ord}}(G)$ is the free sesquicategory on the 2-computad G . We'll follow our argument for 2-categories. We may use the same conical inclusion map since cone string diagrams are ordered. We also know that evaluation is well-defined for ordered string diagrams labeled in a 2-category. We need something slightly stronger though: we need evaluation to work when the ordered string diagram is labeled in a sesquicategory.

Definition. Suppose δ is an ordered string diagram labeled in a sesquicategory C . An *ordered brick decomposition* is a brick decomposition which satisfies one extra rule:

- Each layer of bricks contains at most one node.

We may choose an ordered brick decomposition for any ordered string diagram δ . In fact, any brick decomposition may be refined to become ordered for δ by including extra y -mortar points, as necessary. Our evaluation functor may now be thought of, not as a vertical composite of horizontal composites, but rather, as a vertical composite of whiskerings. We may therefore evaluate ordered string diagrams in a sesquicategory.

Just as we had in the 2-categorical case, we have functors,

$$\begin{aligned} \text{forget}: \text{Sesqui-Cat} &\longrightarrow \text{2-Cmp}, \\ \text{sd}^{\text{ord}}: \text{2-Cmp} &\longrightarrow \text{Sesqui-Cat}. \end{aligned}$$

We also have a natural conical inclusion map

$$\text{cone}_G: G \longrightarrow \text{forget}(\text{sd}^{\text{ord}}(G))$$

and a natural evaluation functor

$$\text{eval}_C: \text{sd}^{\text{ord}}(\text{forget}(C)) \longrightarrow C.$$

These satisfy the Zig-Zag lemmas just as their 2-categorical counterparts did. Thus, we may characterize $\text{sd}^{\text{ord}}(G)$ as the free sesquicategory on the 2-computad G .

9.2 Gray Surface Diagrams

Definition. A *Gray-computad* G consists of four sets of generators G_0 , G_1 , G_2 , and G_3 . We have source and target maps

$$s, t: G_1 \longrightarrow G_0$$

which let us form the free category $\text{dd}(G)$ of dot diagrams. We also have globular source and target maps

$$s, t: G_2 \longrightarrow \text{dd}(G)_1$$

which allow us to generate the free sesquicategory $\text{sd}^{\text{ord}}(G)$ of ordered string diagrams. We take globular source and target maps

$$s, t: G_3 \longrightarrow \text{sd}^{\text{ord}}(G)_2.$$

We note that $\text{sd}^{\text{ord}}(G)_1 = \text{dd}(G)_1$ and $\text{sd}^{\text{ord}}(G)_0 = G_0$.

Consider a surface pre-diagram Δ on $[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1]$. We may *x-project* the foams, seams, and nodes of Δ to get subspaces of $[y_0, y_1] \times [z_0, z_1]$. Gray surface diagrams are surface diagrams that have certain nice properties in the *x*-projection.

Definition. A *Gray surface pre-diagram* is a surface pre-diagram Δ on a cube

$$[x_0, x_1] \times [y_0, y_1] \times [z_0, z_1]$$

which satisfies the following rules.

- The *z*-faces

$$[x_0, x_1] \times [y_0, y_1] \times \{z_0\} \quad \text{and} \quad [x_0, x_1] \times [y_0, y_1] \times \{z_1\}$$

are ordered string pre-diagrams.

- All but finitely many z -slices are ordered string pre-diagrams.

A Gray surface pre-diagram will be called *photogenic* if the following also hold.

- If a line segment $[x_0, x_1] \times \{j\} \times \{k\}$ intersects a node of Δ , then it intersects no other nodes nor any seams.
- Each line segment $[x_0, x_1] \times \{j\} \times \{k\}$ intersects at most two seams S_1, S_2 of Δ . In this case, we require that the x -projections $\pi_x(S_1), \pi_x(S_2) \subset [y_0, y_1] \times [z_0, z_1]$ intersect transversely. We'll call the point $(j, k) \in [y_0, y_1] \times [z_0, z_1]$ a *seam crossing*.

Remark. A Gray surface pre-diagram is, in particular, a surface diagram and so has an underlying 3-computad. It is also true that any Gray-computad has an underlying 3-computad. It would be a mistake, however, to label a Gray surface pre-diagram with a map between these 3-computads.

There are only finitely many z -coordinates in which seams need not be ordered. As such, near a node, the seams approaching must have a defined y -ordering on some neighborhood. When we crop a Gray-surface diagram in order to find a Gray-surface diagram focused on a node, we'll choose it small enough so that the seams maintain their order. This allows us to make the following definition.

Definition. Suppose Δ is a Gray surface pre-diagram. Then Δ has an *underlying Gray-computad* $\text{comp}^{\text{Gray}}(\Delta)$.

Definition. A *Gray surface diagram* consists of a Gray surface pre-diagram Δ , a Gray-computad G , and a map of Gray-computads $\text{comp}^{\text{Gray}}(\Delta) \rightarrow G$.

Example. Suppose Γ is a 3-generator in a Gray-computad G . The cone $\text{cone}_G(\Gamma)$ is a Gray surface diagram focused on the origin node. Indeed, s_z and t_z are ordered and so all slices are therefore ordered.

We may compose Gray surface diagram in the y and z -directions, at least after normal replacements. We won't define gluing Gray surface diagrams in the x -direction. That said, we may whisker a Gray surface diagram with a dot diagram by thickening the dot diagram twice and gluing.

Definition. A *Gray surface pre-evolution* is a surface pre-evolution \mathbb{T} in which for each $l \in [t_0, t_1]$, the time slice $\mathbb{T}|_{t=l}$ is a Gray surface pre-diagram.

Just as a Gray surface diagram has an underlying Gray-computad, so too does any Gray surface pre-evolution \mathcal{T} . Indeed, computadic structure is preserved across time in a pre-evolution by standard arguments.

Definition. A *Gray surface evolution* is a Gray surface pre-evolution \mathcal{T} labeled by a Gray-computad G . We'll write $\mathcal{T}: \Delta \rightarrow \Lambda$ to indicate $s_t(\mathcal{T}) = \Delta$ and $t_t(\mathcal{T}) = \Lambda$. We note that each time slice $\mathcal{T}|_{t=l}$ is a Gray surface diagram.

We'll say that a Gray surface evolution \mathcal{T} is *photogenic* if all but finitely many time-slices are photogenic Gray surface diagrams. Further, we require that submanifolds of $[y_0, y_1] \times [z_0, z_1] \times [t_0, t_1]$ made by x -projecting the seams and nodes of \mathcal{T} intersect transversely. In x -projection, seams of evolutions are codimension 1 and nodes of evolutions are codimension 2. So at most three seams may intersect in x -projection and nodes may not intersect. In any case, a node may intersect a seam in x -projection.

We may compose Gray surface evolutions in the y , z , and t -directions by normalizing near the appropriate face and then gluing. We may also whisker Gray surface evolutions by gluing triply thickened dot diagrams in the x -direction. Note also that each of these operations take photogenic evolutions to photogenic evolutions.

Definition. For any Gray-computad G , there is a Gray-category $\text{Sd}^{\text{Gray}}(G)$ of *Gray surface diagrams labeled by G* .

$$\begin{aligned} \text{Obj}_{\text{Sd}(G)} &= G_0 \\ \text{Hom}_{\text{Sd}(G)}(X, Y) &= \frac{\{\text{dot diagrams } d: X \rightarrow Y\}}{\text{translation, dilation, dot evolution}} \\ 2\text{-Hom}_{\text{Sd}(G)}(d, d') &= \frac{\{\text{ordered string diagrams } \delta: d \rightarrow d'\}}{\text{translation, dilation, ordered string evolution}} \\ 3\text{-Hom}_{\text{Sd}(G)}(\delta, \delta') &= \frac{\{\text{photogenic Gray surface diagrams } \Delta: \delta \rightarrow \delta'\}}{\text{translation, dilation, photogenic evolution}} \end{aligned}$$

We evaluate a photogenic Gray surface diagram by equipping it with a brick decomposition. We ask that brick decompositions be of a particularly nice form.

Definition. A *brick decomposition for a photogenic Gray surface diagram* is one in which each y -layer has either a single seam crossing, a single node, or neither. Also, each z -mortar point is taken at a height k so that the $z = k$ slice string diagram is ordered.

To evaluate a photogenic Gray surface diagram, we send each seam crossing to its Gray interchanger. More precisely, consider a y -layer of bricks Δ in which two seams

S_1 and S_2 sit above a seam crossing $(i, j) \in [y_0, y_1] \times [z_0, z_1]$. We'll assume, at least to start, that the first brick is focused on S_1 and the last brick is focused on S_2 . We record of the walls and regions between S_1 and S_2 with a dot diagram δ . Evaluating this dot diagram gives a 1-morphism $\text{eval}_C(\delta)$. We then define $\text{eval}_C(\Delta)$ to be the Gray interchanger $\Theta_{\lambda(S_1), \text{eval}_C(\delta), \lambda(S_2)}$, where $\lambda(S_1)$ and $\lambda(S_2)$ are the labels on the seams. Of course, if there were other bricks before S_1 or after S_2 , we would evaluate these as usual and whisker the Gray interchanger.

We note that evaluation is independent of refining brick decomposition for photogenic Gray surface diagrams. We should also check that evaluation is independent of photogenic evolution. For any photogenic Gray surface evolution $\mathbb{1}$, we may choose a brick decomposition in which each brick $\mathbb{1}$ is of one of the following four forms:

- (I) Every time slice of $\mathbb{1}$ is a photogenic Gray surface diagram.
- (II) There is a single time slice of $\mathbb{1}$ which is not photogenic. In the x -projection of this time slice, three seams cross.
- (III) There is a single time slice of $\mathbb{1}$ which is not photogenic. In the x -projection of this time slice, two seams intersect in a non-transverse manner.
- (IV) There is a single time slice of $\mathbb{1}$ which is not photogenic. In the x -projection of this time slice, a node intersects a seam.

If a brick is of type (I), then it is clear that evaluation is preserved:

$$\text{eval}_C(s_t(\mathbb{1})) = \text{eval}_C(t_t(\mathbb{1})).$$

We claim that evaluation is preserved for each of the other types of brick. For type (II), we note that $\text{eval}_C(s_t(\mathbb{1}))$ and $\text{eval}_C(t_t(\mathbb{1}))$ are related by a braid relation in the Gray-category C . Evaluation for type (III) bricks is preserved by canceling a Gray interchange with its inverse. Last, type (IV) bricks respect evaluation by naturality of the Gray interchange.

This discussion allows us to define evaluation,

$$\text{eval}_C: \text{Sd}^{\text{Gray}}(\text{forget}(C)) \longrightarrow C.$$

The same Zig-Zag formalism goes through for Gray surface diagrams. So we get our main result.

Theorem. Suppose G is a Gray-computad. Then the Gray-category $\text{Sd}^{\text{Gray}}(G)$ of Gray surface diagrams is the free Gray-category on G .

Chapter 10

Equivalence in 2-Categories

10.1 About Coherence for Equivalence

In this last part of the dissertation, we'll be investigating coherence relations for equivalence. One may see this as an advertisement for the value of string and surface diagrams. The first chapter reviews a proof that any incoherent equivalence in a 2-category may be improved to a coherent equivalence. This argument is not new. However, it functions as a nice illustration of string diagrams. We will use the ideas in this chapter as a starting off point to investigate equivalence in a Gray-category. Indeed, the same sort of techniques will work with surface diagrams to *coherify* any incoherent equivalence in a Gray-category. There are, however, both more data and many more relations to keep track of in this context. That said, all arguments come down to “topologically reasonable” manipulations of surfaces.

The diagrams in this part of the dissertation are not, strictly speaking, examples of string and surface diagrams as we've defined them. In particular, these diagrams are drawn in a non-progressive manner. So, for example, when we draw a string turning around, we actually intend to splice in a node at the extremum. We similarly splice in nodes to seams that turn around, and we splice seams into walls that turn around. Further, we must deform the diagram slightly so as to make the strata approaching other strata progressive. This gives our diagrams a “pointy” appearance. In this way, we may interpret each of the non-progressive strings and surfaces as progressive diagrams.



10.2 Coherification

Definition. In a 2-category, an (*incoherent*) *equivalence* of objects

$$X \text{ and } Y$$

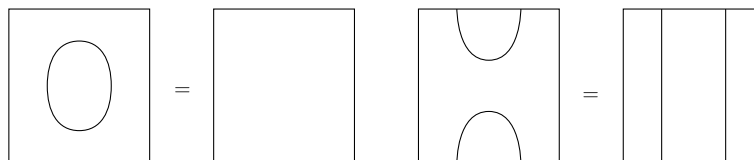
consists of 1-morphisms

$$\frac{f}{X \rightarrow Y} \quad \frac{g}{Y \rightarrow X}$$

as well as 2-morphisms

$$\begin{array}{cccc} Y & \alpha & X & Y \\ f \curvearrowright & & f \curvearrowright & g \curvearrowright \\ X & & Y & X \\ g & & \alpha^{-1} & \beta \end{array} \quad \begin{array}{cccc} X & \beta^{-1} & Y & X \\ g \curvearrowright & & g \curvearrowright & f \curvearrowright \\ Y & & X & Y \\ f & & \beta & \end{array}$$

which satisfy inverse relations:



We'll think of f and g as allowing us to compare X and Y . To say that f and g were inverse, we'd demand the relations $f \circ g = 1_X$ and $g \circ f = 1_Y$. It's not really fair to demand such relations of 1-morphisms in a 2-category, so we'll instead just ask for invertible 2-morphisms to make these comparisons. That is, we demand the existence of α and α^{-1} in order to compare $f \circ g$ with 1_X , and β and β^{-1} to compare $g \circ f$ with 1_Y . Our relations ensure that α and α^{-1} really are inverse to each other, as are β and β^{-1} .

Consider the statement,

$$f \circ g \circ f \cong f.$$

This is a statement of existence: there is an isomorphism from $f \circ g \circ f$ to f . When working with equalities, we don't typically need to know why two things are the same, just that

they are. But in fact, there are two perfectly good reasons that $f \circ g \circ f$ is isomorphic to f . We might cancel g using the f on its left or the f on its right. Put another way, we could build an automorphism $f \cong f \circ g \circ f \cong f$ of f by, say, uncanceled on the left followed by canceling on the right. While we might expect this automorphism to be 1_f , there's nothing in our definition that requires this. As a result, we must be somewhat careful when making calculations with f and g . Carelessly making cancellations might lead us to think we have equalities where we don't actually. This explains the modifier, "incoherent," in our definition.

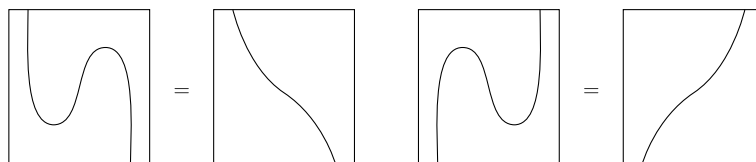
Definition. In a 2-category, a pair of 1-morphisms

$$\frac{f}{X \rightarrow Y} \quad \frac{g}{Y \rightarrow X}$$

are *adjoint* if there are 2-morphisms

$$\begin{array}{c} Y \\ \alpha \\ \text{---} \\ f \quad \text{---} \quad g \\ X \end{array} \quad \begin{array}{c} Y \\ g \quad \text{---} \quad f \\ \beta \\ X \end{array}$$

satisfying *zig-zag* relations



We may say that (g, f) are an adjoint pair with *unit* β and *counit* α . Note that this definition is not symmetric in f and g . One should look to the unit for the left adjoint to be on the left and the right adjoint to be on the right. This definition is meant to generalize the definition of adjoint functors in the 2-category of categories, functors, and natural transformations.

Lemma. In an adjunction, the unit and counit determine each other.

Proof. Suppose both α and α' satisfy the zig-zag relations with β . Then

$$\begin{array}{c} \alpha \\ \text{---} \end{array} = \begin{array}{c} \alpha \quad \alpha' \\ \text{---} \quad \beta \\ \text{---} \end{array} = \begin{array}{c} \alpha' \\ \text{---} \end{array}$$

□

Definition. The previous lemma allows us to define an exponent \sim which swaps between corresponding units and counits.

$$\alpha^\sim = \beta \quad \text{and} \quad \beta^\sim = \alpha.$$

This should be thought of as playing a similar a role to the exponent -1 for inversion.

Definition. A *coherent equivalence* is an equivalence in which the 2-morphisms satisfy all four possible zig-zag relations:

- β and α demonstrate (g, f) as an adjoint pair;
- α^{-1} and β^{-1} demonstrate (f, g) as an adjoint pair.

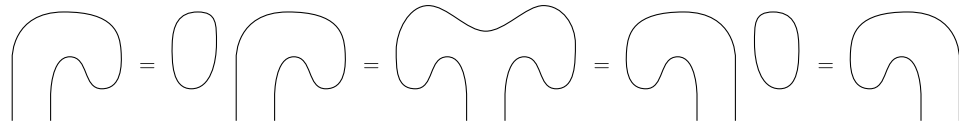
Theorem (equivalence coherification). Given an incoherent equivalence as defined above, we may rechoose β and β^{-1} so as to get a coherent equivalence. Moreover, there is a unique such choice.

Proof. The reader is encouraged at this point to ignore the following and discover a proof for herself. To start, one may solve

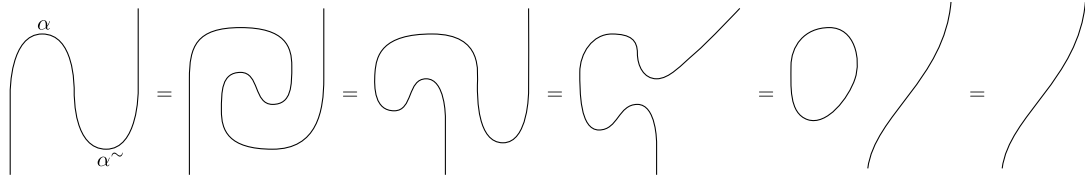
for α^\sim using only the inverse relations. We choose

We will demonstrate $(\alpha, \alpha^{-1}, \alpha^\sim, \alpha^{-\sim})$ give a coherent equivalence. It is straightforward to see that α^\sim and $\alpha^{-\sim}$ are inverse. We must show that our data satisfy the four zig-zag relations. One of the four may be seen by the following:

Next, notice



which allows us to show a second zig-zag relation:



The remaining two zig-zag relations can be proved by using arguments whose diagrams are top-bottom reflections of the given ones. Uniqueness follows from uniqueness of inverses and the previous unit/counit uniqueness lemma. \square

To make labels unnecessary, we can put orientations on the data of a coherent equivalence:



Then we just make a convention, say, X is to the right and Y is to the left as we travel along the string.

Once we have a coherent equivalence, it is easy to work with the 2-morphisms defining such an equivalence. One can prove, in a certain Morse-theoretic sense, that the zig-zag relations generate all topological relations of strings in the plane. So one is free to manipulate any composition of coherent equivalence data using topological relations without changing the 2-morphism being described.

10.3 Application: Morita Equivalence of Rings

Let's take a quick digression to discuss the definitions and results of the previous section for the bicategory, \mathbf{Rng} , whose 0-morphisms are rings, 1-morphisms are bimodules, and 2-morphisms are linear maps. Horizontal composition of bimodules is given by taking tensor product. In this category, incoherent equivalence is given the name *Morita equivalence*. The definition of adjoint pair gives the usual handed definition of dual bimodules. Of course, we would need to decorate our string diagrams with histories in order to evaluate

them in this bicategory. Any choice of histories will do and this choice has no real effect on the theory.

Example. We may think of column vectors $\text{Mat}_{n,1}(R)$ as forming an (R, R) -bimodule by using left and right scalar multiplication. We can similarly get a bimodule of row vectors, $\text{Mat}_{1,n}(R)$. We can recognize $(\text{Mat}_{1,n}(R), \text{Mat}_{n,1}(R))$ as a dual pair with the maps

$$\begin{aligned} R &\longrightarrow \text{Mat}_{1,n} \otimes_R \text{Mat}_{n,1} \\ r &\mapsto \sum_i r e^i \otimes e_i \\ \text{Mat}_{n,1} \otimes_R \text{Mat}_{1,n} &\longrightarrow R \\ &\text{trace of the product.} \end{aligned}$$

We can also recognize $(\text{Mat}_{n,1}(R), \text{Mat}_{1,n}(R))$ as a dual pair with the maps

$$\begin{aligned} R &\longrightarrow \text{Mat}_{n,1} \otimes_R \text{Mat}_{1,n} \\ r &\mapsto \sum_i r e_i \otimes e^i \\ \text{Mat}_{1,n} \otimes_R \text{Mat}_{n,1} &\longrightarrow R \\ &\text{dot product.} \end{aligned}$$

It's worth noting, though, that these maps don't satisfy the inverse relations of a coherent equivalence: when we take the trace of the $n \times n$ identity matrix, we get n , not 1. So for $R = \mathbb{Z}$, or even worse, $\mathbb{Z}/n\mathbb{Z}$, this ambidextrous adjunction is quite far from a coherent equivalence.

Example. We may also think of column vectors $\text{Mat}_{n,1}(R)$ as forming a $(\text{Mat}_{n,n}(R), R)$ -bimodule by using left and right matrix multiplication. Similarly, row vectors $\text{Mat}_{1,n}(R)$ form an $(R, \text{Mat}_{n,n}(R))$ -bimodule. The matrix multiplication maps,

$$\begin{aligned} \text{Mat}_{1,n} \otimes_{\text{Mat}_{n,n}} \text{Mat}_{n,1} &\longrightarrow R \\ \text{Mat}_{n,1} \otimes_R \text{Mat}_{1,n} &\longrightarrow \text{Mat}_{n,n} \end{aligned}$$

are both invertible. To invert the first, we rely heavily on the $\text{Mat}_{n,n}$ -middle linearity. The inverse of the second can be thought of as, "regard a matrix as a sum of rank 1 matrices."

These maps give us the data for a coherent equivalence which demonstrates that any ring R is Morita equivalent to its ring of square matrices, $\text{Mat}_{n,n}(R)$. It follows that the rings of $m \times m$ and $n \times n$ matrices are Morita equivalent. In fact, one can directly demonstrate this equivalence by considering the bimodule $\text{Mat}_{m,n}(R)$ and its dual, $\text{Mat}_{n,m}(R)$.

In the previous example, our choice of maps

$$\begin{aligned}\alpha: \text{Mat}_{1,n} \otimes_{\text{Mat}_{n,n}} \text{Mat}_{n,1} &\longrightarrow R \\ \beta^{-1}: \text{Mat}_{n,1} \otimes_R \text{Mat}_{1,n} &\longrightarrow \text{Mat}_{n,n}\end{aligned}$$

as matrix multiplication gave a coherent equivalence of rings. We can fiddle with α as follows: given $r, s \in R$ invertible, we define $\alpha'(x) = r(\alpha(x))s$. This α' is invertible. It will, however, usually only demonstrate an incoherent equivalence with β . We can similarly fiddle with β by left and right multiplying by invertible matrices. Coherification improves these less-than-wonderful choices of α' and β' by normalizing one using the other.

Chapter 11

Equivalence in Gray-Categories

11.1 Semi-Coherification

Definition. An *incoherent equivalence* in a Gray-category between 0-morphisms

$$X \quad \text{and} \quad Y$$

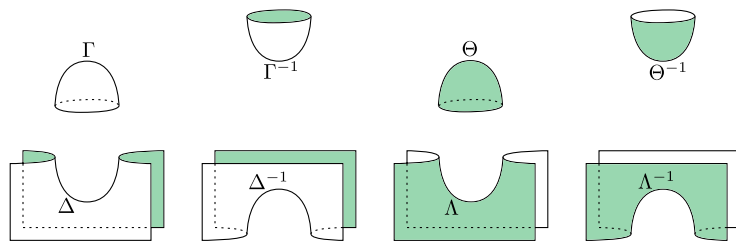
consists of 1-morphisms

$$\begin{array}{c} f/X \\ \hline Y \end{array} \quad \begin{array}{c} g/Y \\ \hline X \end{array}$$

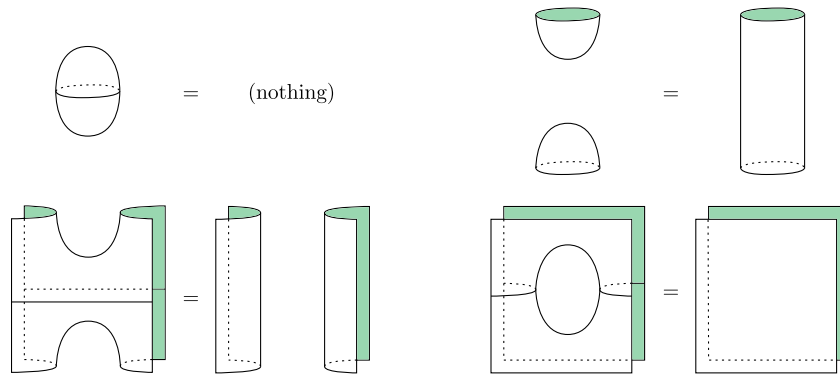
2-morphisms

$$\begin{array}{c} f \\ \xrightarrow{\quad} \\ g \end{array} \alpha \quad \begin{array}{c} g \\ \xleftarrow{\quad} \\ f \end{array} \beta \quad \begin{array}{c} f \\ \xleftarrow{\quad} \\ g \end{array} \gamma \quad \begin{array}{c} g \\ \xrightarrow{\quad} \\ f \end{array} \delta$$

and 3-morphisms

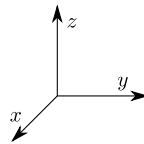


We require the 3-morphisms to satisfy invertibility relations:



Several remarks on the diagrams are in order.

- In our surface diagrams, we have composition across 0-morphisms in the x -direction, across 1-morphisms in the y -direction, and across 2-morphisms in the z -direction. We will choose our axes as follows.

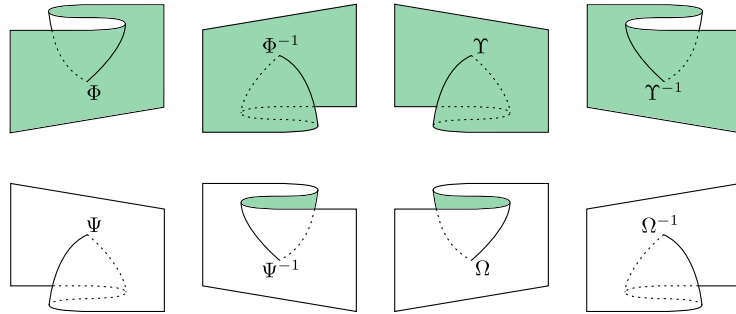


- We color one side of each surface. This is meant to be interpreted according to the following convention: the region on the non-colored side of a surface corresponds to the object X and the region on the colored side corresponds to Y .
- One could write the 2-morphisms on the fold lines of the surfaces. (Fold lines are boundaries of the projection which aren't boundary of the surface.) We have chosen not to do this, however, as it makes the diagrams busier and is not necessary given the coloring convention.
- As an example of our conventions, we note that the saddle Δ is a map from $1_{f \circ_x g}$ to $\alpha \circ_y \gamma$.
- We will need to reference various symmetries of the cube that act on the diagrams. R_x refers to reflection in the x -direction, and similarly for R_y and R_z . Further, we'll refer to swapping the colored side of a surface as R_c . We allow ourselves to refer to R_c as a reflection for convenience.

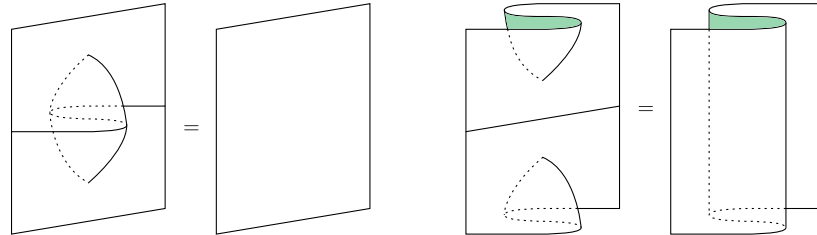
- When we write a relation in the above definition, we actually mean to impose all relations of that form. To get the other relations, we act by all compositions of R_x, R_y, R_z and R_c .

Next, we define a variant of equivalence in which we demand that the morphisms at levels 0-2 form an up-to-isomorphism adjoint equivalence. This amounts to taking the coherence relations for a 2-categorical adjoint equivalence as data.

Definition. A *semi-coherent equivalence* in a Gray-category is an incoherent equivalence with extra 3-morphism data:



We require this extra data to satisfy invertibility relations:

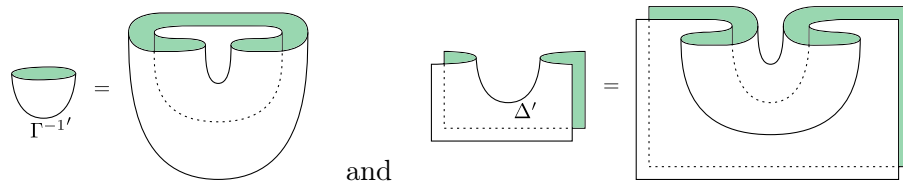


Theorem (semi-coherification). Any equivalence can be semi-coherified.

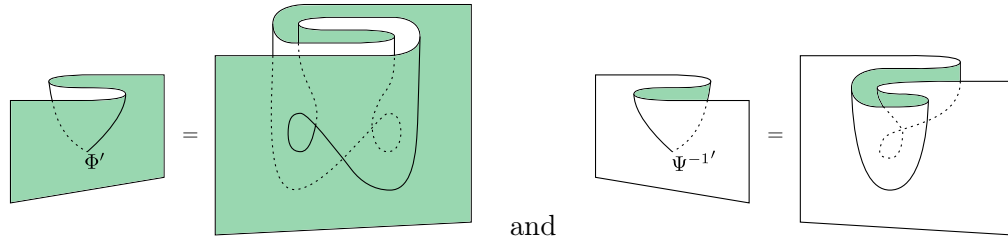
Proof. We define γ' and δ' in terms of $\alpha, \beta, \gamma,$ and δ as follows:

$$\gamma' \leftarrow = \text{cup} \quad \text{and} \quad \rightarrow \delta' = \text{cap}$$

Proofs from the 2-categorical case now become data that we record. The cups, caps, and saddles for γ' and δ' can be defined as



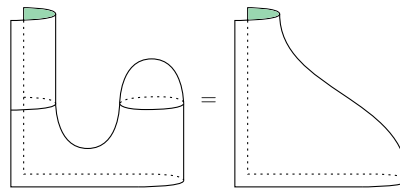
or various reflections thereof. We define cusps



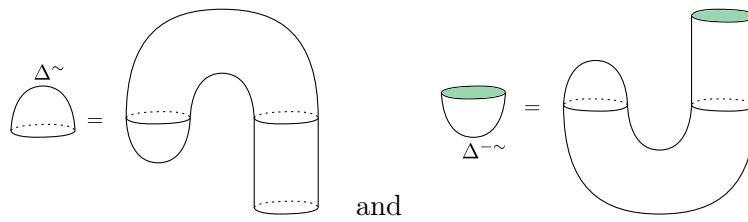
and get the other cusps as R_x, R_z , and $R_x R_z$ reflections of these. We automatically get the desired invertibility relations. Indeed, the R_z reflection of a diagram gives the inverse morphism. \square

11.2 Morse-Cancellation

We will now consider a coherence relation on the 3-data: the Morse-cancellation.



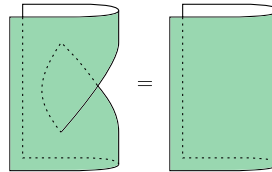
This relation has a R_x symmetry, so there are $|\langle R_y, R_z, R_c \rangle| = 8$ Morse-cancellation relations. We can regard these relations as zig-zag relations. Indeed, in $\text{Hom}(X, X)$, we have $\Gamma, \Gamma^{-1}, \Delta$, and Δ^{-1} , which form an incoherent 2-equivalence. This is straightforward to see: just flatten out the x -direction of a surface diagram to get a two dimensional diagram. In other words, the fold lines of a surface diagram give 1-morphisms in the microcosm 2-category. According to the 2-categorical coherence argument, we can rechoose Γ and Γ^{-1} as



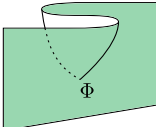
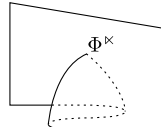
to give an adjoint equivalence with the Δ, Δ^{-1} saddles. One could equally well rechoose the saddles instead of the cup and cap. The reader may find it amusing to draw these, as well.

11.3 Swallowtail

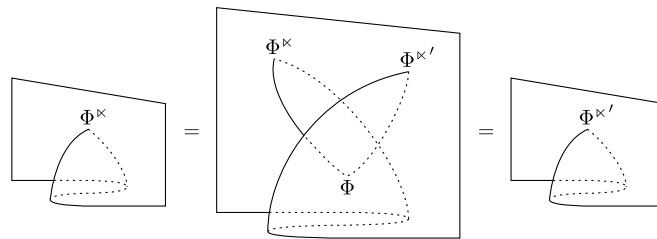
We consider another coherence relation on 3-morphisms: the swallowtail.



The swallowtail has a $R_x R_z$ symmetry (rotation about the y -axis) and so there are 8 variants. It can be thought of as a twisted zig-zag relation. Following this assertion, we will recreate the arguments for 2-categorical coherification, but in a twisted manner.

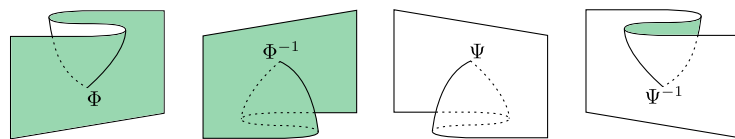
Definition. For any cusp,  there is at most one cusp  satisfying the swallowtail relations, which we'll call the *swallowtail partner*.

Proof. Consider making the right and left cancellations:



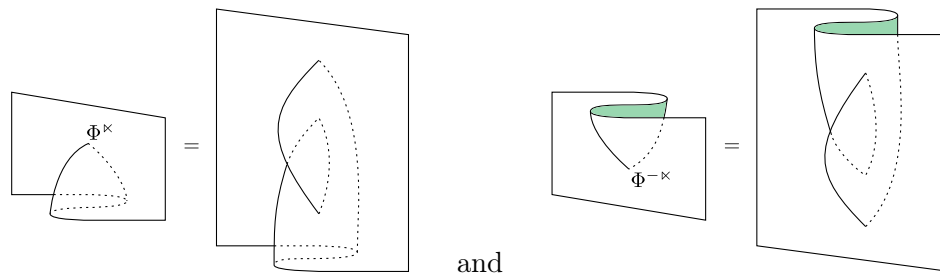
□

Lemma (existence of swallowtail partners). Given cusps,

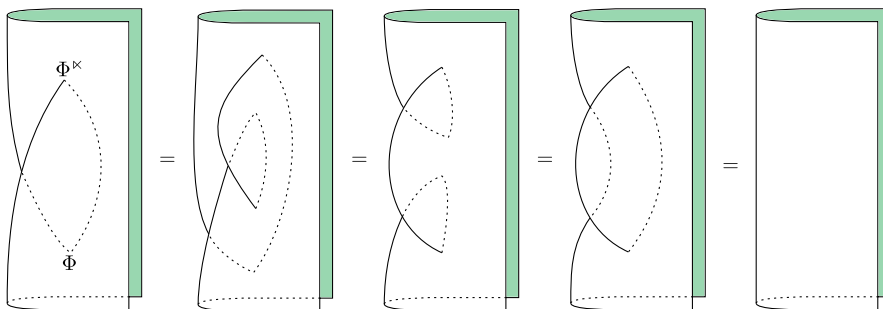


we may rechoose Ψ and Ψ^{-1} so that the cusps satisfy the swallowtail relations.

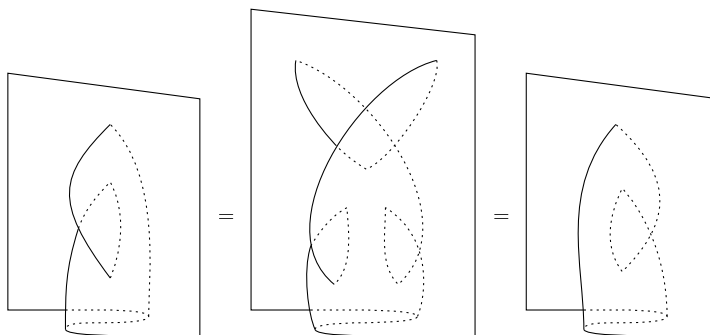
Proof. Let's define Φ^\times and $\Phi^{-\times}$ based on $\Phi, \Phi^{-1}, \Psi,$ and Ψ^{-1} as follows:



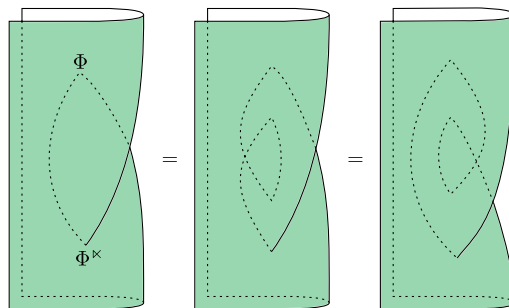
We now claim that Φ and Φ^\times satisfy their two swallowtail relations, as do Φ^{-1} and $\Phi^{-\times}$. We'll show the relations for Φ and Φ^\times and just note that the relations for the inverses may be proved as R_z reflections. We get one swallowtail relation as follows:



Now consider the following



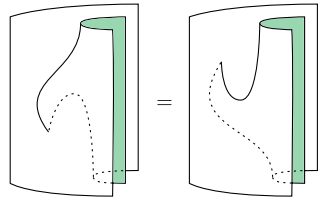
which lets us make the following reduction:



We can simplify the right-most diagram by using a $R_x R_y R_c$ reflection of the argument for the other swallowtail relation. □

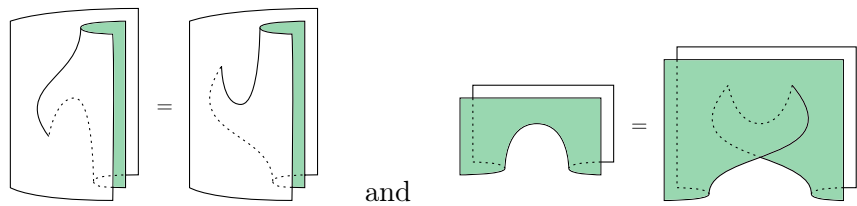
11.4 Cusp-Flip

In this section, we analyze the cusp-flip relations. We will assume that we have a semi-coherent equivalence with the Morse-cancellation and swallowtail relations at our disposal. The cusp-flip relation is given by the following diagram.

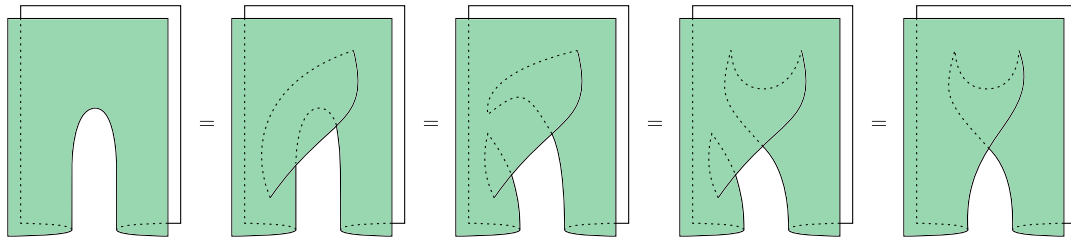


We can think of this rule as asserting that the the down and up deformations of a horizontal cusp must describe the same morphism. The cusp-flip relation has a $R_x R_z R_c$ symmetry, so there are 8 variants.

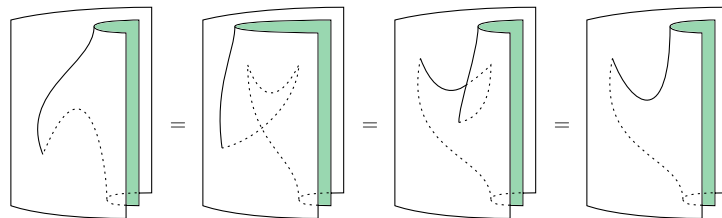
Lemma (solving for saddle). The following relations imply each other:



Proof. For the forward direction, consider

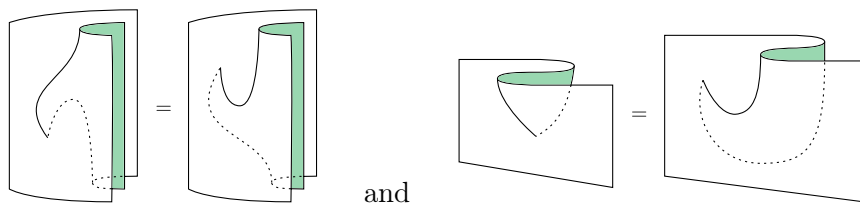


The reverse direction follows from

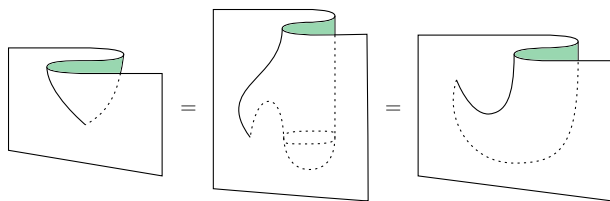


□

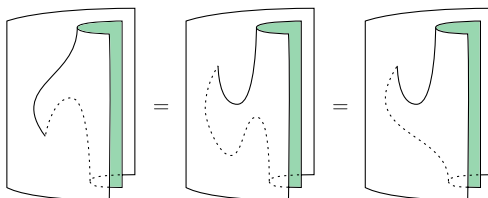
Lemma (solving for cusp). The following relations imply each other:



Proof. For the forward direction, consider



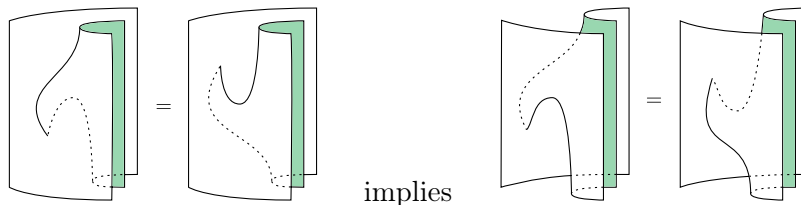
The reverse direction follows from



□

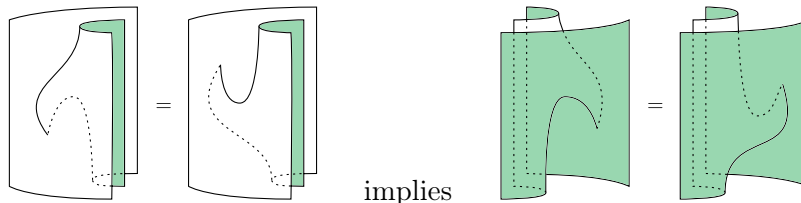
The last three lemmas of this section each say that a cusp-flip relation implies a reflected cusp-flip relation. Of course, each converse immediately holds, so we may regard each lemma as giving a logical equivalence.

Lemma (R_z). A cusp-flip relation implies its R_z reflection cusp-flip.

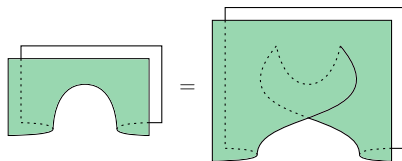


Proof. The data in the relation on the right are the inverses of the data in the relation on the left. The proof only involves vertical composition, so it's easy enough to read in algebraic notation: if $A = B$, then $A^{-1}AB^{-1} = A^{-1}BB^{-1}$ and so $B^{-1} = A^{-1}$. In other words, inverses are unique. □

Lemma (R_xR_y). A cusp-flip relation implies its R_xR_y reflection cusp-flip.

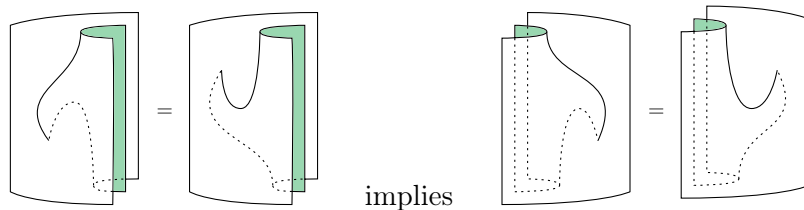


Proof. We showed in a previous lemma that the relation on the left is equivalent to

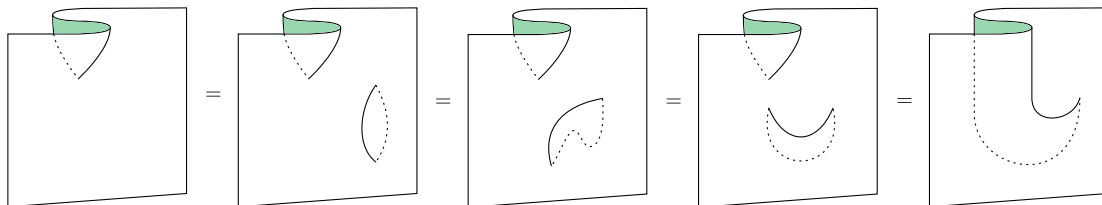


Further, an $R_x R_y$ reflection of that same lemma shows that the relation on the right is also logically equivalent to this saddle relation. \square

Lemma (R_y). A cusp-flip relation implies its R_y reflection cusp-flip.



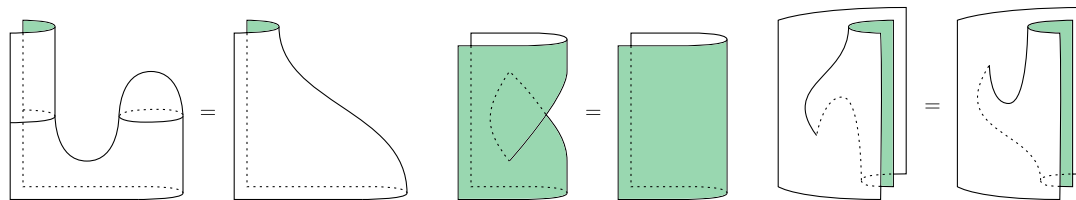
Proof. We'll prove the result using the alternative relations obtained by solving each for a cusp. Then



gives the result. \square

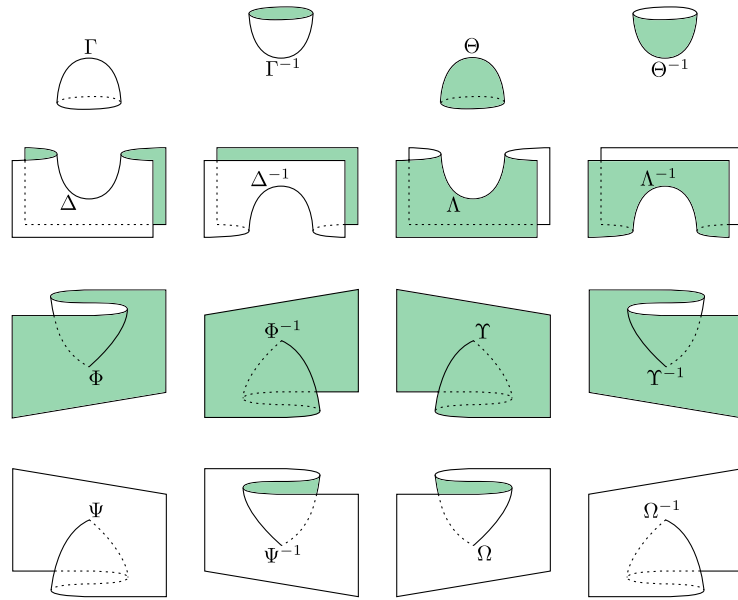
11.5 Coherification

Definition. A *fully coherent equivalence* in a Gray-category is a semi-coherent equivalence that further satisfies Morse-cancellations, swallowtails, and cusp-flips.

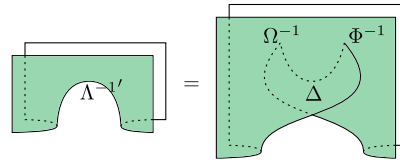


Theorem (coherification). Any incoherent equivalence can be fully coherified.

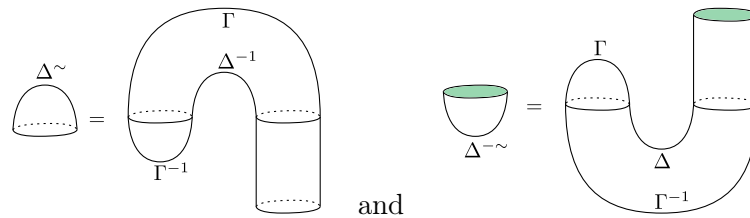
Proof. First, semi-coherify so that we have data:



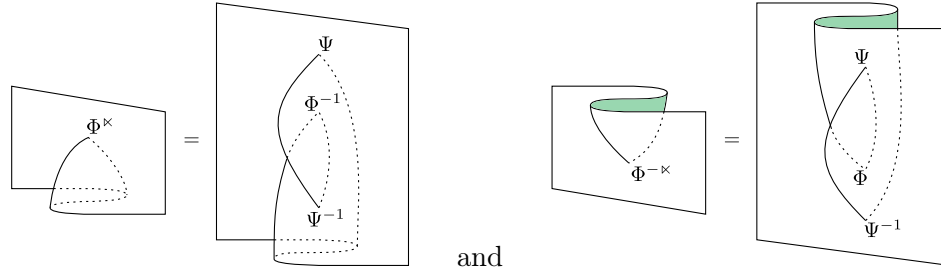
Rechoose saddle Λ^{-1} as $\Lambda^{-1'}$ in terms of Δ , Ω^{-1} , and Φ^{-1} as follows:



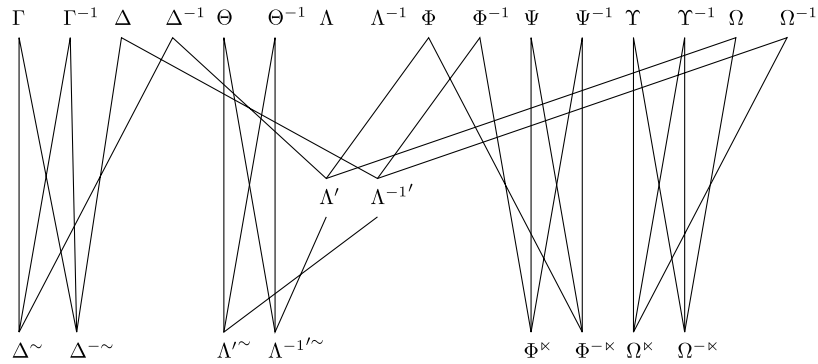
Similarly, rechoose Λ as Λ' in terms of Δ^{-1} , Ω , and Φ . These choices have ensured that at least one of the eight cusp-flip relations are satisfied. Next, rechoose the Γ and Γ^{-1} as Δ^{\sim} and $\Delta^{-\sim}$:



Similarly, rechoose Θ , and Θ^{-1} as Λ^{\sim} and $\Lambda^{-1'\sim}$. Rechoose Ψ and Ψ^{-1} as Φ^{\times} and $\Phi^{-\times}$:



Also, rechoose Υ and Υ^{-1} as Ω^\times and $\Omega^{-\times}$. We summarize our choices of data as follows.



At the top, is the semi-coherified data. If we were feeling more adventurous, we could extend this tree back to the incoherent equivalence data. If we rechoose a piece of data, we place that directly below and note which other data went into that choice. We claim that the following data forms a fully coherent equivalence:

$$\Delta^\sim, \Delta^{-\sim}, \Delta, \Delta^{-1}, \Lambda'^\sim, \Lambda^{-1'\sim}, \Lambda', \Lambda^{-1'}, \Phi, \Phi^{-1}, \Phi^\times, \Phi^{-\times}, \Omega^\times, \Omega^{-\times}, \Omega, \Omega^{-1}$$

We've ensured that all Morse-cancellations and swallowtails hold and one of the eight cusp-flip holds. By the lemmas of the cusp-flip section, all eight cusp-flip relations now hold, automatically. Indeed, $R_z, R_x R_y$, and R_y generate all cusp-flips given the $R_x R_z R_c$ symmetry of the cusp-flip relation. □

Chapter 12

Future Work

There are a number of things that I'd like to explore that didn't make it into the main part of the dissertation. Here is a list of a few of them.

- I'd like to prove that the $\text{Sd} / \text{forget}$ adjunctions are monadic.
- We should be able to describe relations for a free n -category of diagrams as a quotient of a $(n + 1)$ -category of diagrams, where the $(n + 1)$ -morphisms are used to encode the relations.
- There is a free bicategory of string diagrams equipped with histories. In fact, most of this has already been written, but a few technical points remain to be settled concerning evaluation.
- The free bicategory of string diagrams with histories is biequivalent to the free 2-category of string diagrams. This would carefully encode the idea that decorations for a string diagram are not interesting.
- Suppose C and D are 2-categories and suppose we have a string diagram δ labeled in C . We would like to describe how to overlay δ with string diagrams in order to describe functors $C \rightarrow D$, pseudonatural transformations, and modifications acting on δ .

I also have a few questions that I'm still working on for the coherence of equivalence part of the dissertation.

How unique is coherification in a Gray-category?

The coherification result for equivalences in Gray-categories may be regarded as a statement of existence. That is, given any incoherent equivalence, we have give a procedure to build a coherent equivalence. I'm working on a uniqueness result: a coherent equivalence in a Gray-category should be determined by a small subset of its defining data.

What can we say about Tang_2^1 ?

The calculations in this paper should be regarded as taking place in a free Gray-category on the given generators, quotiented by the given invertibility and topological relations. There's another Gray-category that I'm interested in, Tang_2^1 , which has all the same generating morphisms as a coherent equivalence, but is only subject to the topological relations. I've been looking at understanding maps between representations of Tang_2^1 .

To what degree can we extend "mates" to surfaces?

Suppose a and b are parallel 1-morphisms in a 2-category with right adjoints. Then any map $\tau: a \rightarrow b$ has a corresponding mate $\tau^*: b^* \rightarrow a^*$. If τ happens to be an isomorphism, then a quick diagrammatic check shows the inverse of the mate is the mate of the inverse. The situation is more complicated with surfaces. Indeed, the existence of saddles requires us to be working with the ambidextrous adjunction. But then any τ induces \mathbb{Z} -many maps, half $a \rightarrow b$ and half $b^* \rightarrow a^*$. I'm thinking about a calculus of surfaces with embedded curves to sort this all out.

Appendix A

Smooth Functions

In this Appendix, we review from Calculus our definition of a smooth function between arbitrary subsets of Euclidean spaces.

Definition. A function $f: \mathbb{R} \rightarrow \mathbb{R}$ is of class C^k if it has continuous derivatives up to order k :

$$f', f'', \dots, f^{(k)}.$$

We say f is of class C^∞ or *smooth* if it has derivatives of all orders.

“Being smooth” is a local property of a function. We only need our function to be defined on some neighborhood of a point in order to decide if the function is smooth. So we say that a function is *smooth at a point* p if there is some neighborhood U of p on which it is smooth. The function is smooth, then, if and only if it is smooth at every point in its domain. We may similarly define smoothness locally for a function whose domain is a higher dimensional Euclidean space.

A function $f: \mathbb{R}^m \rightarrow \mathbb{R}$ is *smooth at a point* $p \in \mathbb{R}^m$ if there exist partial derivatives of all orders on some neighborhood U of p .

It is straightforward to define smoothness for functions whose codomains are higher dimensional Euclidean spaces.

A map $g: U \rightarrow \mathbb{R}^n$ is smooth if and only if each of the component functions

$$U \xrightarrow{g} \mathbb{R}^n \xrightarrow{\text{proj}_i} \mathbb{R}, \quad 1 \leq i \leq n,$$

is smooth.

Our most general definition of smooth function works for functions defined on arbitrary (not necessarily open) subsets of Euclidean spaces.

Suppose $A \subset \mathbb{R}^m$ and $B \subset \mathbb{R}^n$ are arbitrary subsets. A map $f: A \rightarrow B$ is smooth at $p \in A$ if there is an open neighborhood $U \subset \mathbb{R}^m$ of p on which f has a smooth extension

$$\tilde{f}: U \rightarrow \mathbb{R}^n.$$

Our last definition is perhaps a bit strange for the following reason: at each point we demand the existence of an extension, \tilde{f} , but we do not keep a record of \tilde{f} for later use. This is reasonable because the values of a smooth function f near the boundary determine the function values at the boundary. Why then do we bother to ask for an extension? There exist functions which are smooth on the interior of their domain $A^\circ \subset \mathbb{R}^m$ but which admit no smooth extension on a neighborhood of the boundary. Such functions do not deserve to be called smooth at their boundary.

Example. The function $y = x^{1/3}$ on $\mathbb{R}^{\geq 0} \rightarrow \mathbb{R}^{\geq 0}$ is smooth at all points in the interior of $\mathbb{R}^{\geq 0}$. It is not smooth at 0, however, because it cannot be smoothly extended on any neighborhood of 0 in \mathbb{R} . Indeed, the one-sided derivative shows that dy/dx must be undefined at $x = 0$.

Appendix B

Adjoint Functors

A pair of functors, $r: C \rightarrow D$ and $l: D \rightarrow C$ are an *adjoint pair* (l, r) if there is a bijection,

$$\text{Hom}_D(X, r(Y)) \cong \text{Hom}_C(l(X), Y),$$

which is natural in X and Y . We'll say that l is the left adjoint of r or that r is the right adjoint of l . The use of "left" and "right" refer to which slot, the domain or codomain, in which we place l and r above.

Adjoint functors are good for describing "free" constructions. A free object is one in which the maps out of it are determined by a choice of where to send the generators. Moreover, the generators should be "independent" in the sense that we are free to send the generators wherever we want. A free functor, then, is a left adjoint to a functor which "forgets" structure.

$$\text{Hom}(X, \text{forget}(Y)) \cong \text{Hom}(\text{free}(X), Y)$$

Every adjoint pair of functors automatically comes with unit and counit natural transformations:

$$\begin{aligned}\alpha: D &\Rightarrow D \xrightarrow{l} C \xrightarrow{r} D \\ \beta: C &\xrightarrow{r} D \xrightarrow{l} C \Rightarrow C\end{aligned}$$

These will satisfy *zig-zag relations*:

$$\begin{aligned}D \xrightarrow{l} C \Rightarrow D \xrightarrow{l} C \xrightarrow{r} D \xrightarrow{l} C \Rightarrow D \xrightarrow{l} C &= D \xrightarrow{l} C \\ C \xrightarrow{r} D \Rightarrow C \xrightarrow{r} D \xrightarrow{l} C \xrightarrow{r} D \Rightarrow C \xrightarrow{r} D &= C \xrightarrow{r} D\end{aligned}$$

In fact, the converse is also true.

Theorem. Functors $r: C \rightarrow D$ and $l: D \rightarrow C$ form an adjoint pair (l, r) if and only if there are natural transformations

$$\begin{aligned}\alpha: D &\Rightarrow D \xrightarrow{l} C \xrightarrow{r} D \\ \beta: C &\xrightarrow{r} D \xrightarrow{l} C \Rightarrow C\end{aligned}$$

satisfying the zig-zag relations.

Proof. We have two directions to show.

adjoints have units and counits Let's give the natural bijection of Hom sets a name:

$$\varphi_{X,Y}: \text{Hom}_D(X, r(Y)) \cong \text{Hom}_C(l(X), Y).$$

We define α and β as follows. Suppose c is an object of C and d is an object of D . Then we define:

$$\begin{aligned}\alpha(d) &= \varphi_{d, l(d)}^{-1}(1_{l(d)}): d \rightarrow r(l(d)) \\ \beta(c) &= \varphi_{r(c), c}(1_{r(c)}): l(r(c)) \rightarrow c\end{aligned}$$

These assignments are natural. For example, given $f: d \rightarrow d'$ in D , we get the naturality square

$$d \xrightarrow{\alpha(d)} r(l(d)) \xrightarrow{r(l(f))} r(l(d')) = d \xrightarrow{f} d' \xrightarrow{\alpha(d')} r(l(d'))$$

because both composites are the image of $l(f)$ under

$$\text{Hom}_C(l(d), l(d')) \xrightarrow{\varphi_{d, l(d')}^{-1}} \text{Hom}_D(d, r(l(d')))$$

according to the naturality squares for $\varphi_{d, l(f)}^{-1}$ and $\varphi_{f, l(d')}^{-1}$. The naturality for β is very similar.

It remains to show that the zig-zag relations hold. We'll show

$$r(c) \xrightarrow{\alpha(r(c))} r(l(r(c))) \xrightarrow{r(\beta(c))} r(c)$$

is the identity on $r(c)$. The naturality square for $\varphi_{r(c), \beta(c)}$ tell us

$$\begin{aligned}\text{Hom}_D(r(c), r(l(r(c)))) &\xrightarrow{\beta(c)_*} \text{Hom}_D(r(c), r(c)) \xrightarrow{\varphi_{r(c), c}} \text{Hom}_C(l(r(c)), c) = \\ \text{Hom}_D(r(c), r(l(r(c)))) &\xrightarrow{\varphi_{r(c), l(r(c))}} \text{Hom}_C(l(r(c)), l(r(c))) \xrightarrow{\beta(c)_*} \text{Hom}_C(l(r(c)), c)\end{aligned}$$

Pushing $\alpha(r(c))$ around this diagram gives the desired result. The other zig-zag relation holds due to naturality of φ with respect to $\alpha(l(d))$.

zig-zags imply adjunction We'll define

$$\varphi_{X,Y}: \text{Hom}_D(X, r(Y)) \longrightarrow \text{Hom}_C(l(X), Y)$$

and its proposed inverse as follows:

$$\begin{aligned} X \xrightarrow{a} r(Y) &\quad \mapsto \quad l(X) \xrightarrow{l(a)} l(r(Y)) \xrightarrow{\beta(Y)} Y \\ l(X) \xrightarrow{b} Y &\quad \mapsto \quad X \xrightarrow{\alpha(X)} r(l(X)) \xrightarrow{r(b)} r(X) \end{aligned}$$

We'll first check that these are, in fact, inverse:

$$\begin{aligned} X \xrightarrow{a} r(Y) &\quad \mapsto \quad X \xrightarrow{\alpha(X)} r(l(X)) \xrightarrow{r(l(a))} r(l(r(Y))) \xrightarrow{r(\beta(Y))} r(Y) \\ &= \quad X \xrightarrow{a} r(Y) \xrightarrow{\alpha(r(Y))} r(l(r(Y))) \xrightarrow{r(\beta(Y))} r(Y) \\ &= \quad X \xrightarrow{a} r(Y) \xrightarrow{1} r(Y) \end{aligned}$$

Here we used naturality of α with respect to a and one of the zig-zag relations. We may use naturality of β with respect to b and the other zig-zag relation to show that the following composite is equal to the identity.

$$l(X) \xrightarrow{b} Y \quad \mapsto \quad l(X) \xrightarrow{l(\alpha(X))} l(r(l(X))) \xrightarrow{l(r(b))} l(r(Y)) \xrightarrow{\beta(Y)} Y$$

We must also check naturality for φ . That is, we'll show that the $\varphi_{X,Y}$ are the components of a natural transformation. Suppose $f: Y \rightarrow Y'$ is a morphism in C . Naturality of φ with respect to f requires that for any $a: X \rightarrow r(Y)$,

$$l(X) \xrightarrow{l(a)} l(r(Y)) \xrightarrow{\beta(Y)} Y \xrightarrow{f} Y' \quad = \quad l(X) \xrightarrow{l(a)} l(r(Y)) \xrightarrow{l(r(f))} l(r(Y')) \xrightarrow{\beta(Y')} Y'$$

This follows by naturality of β with respect to f . Naturality of φ with respect to a morphism $g: X' \rightarrow X$ in D is even easier: in this case g acts on the left and β on the right, so naturality follows from the associativity of composition alone. Since φ is a natural transformation, it follows automatically that by inverting its components, φ^{-1} , we get a natural transformation.

□

This theorem gives us a characterization of adjunctions that works in any bicategory. In particular, we may apply it to monoidal categories, in which case we recover the usual notion of dual objects. We'll introduce “mates” in order to prove uniqueness of adjoints.

Definition. Suppose $\sigma: f \rightarrow g$ and suppose that f and g both have right adjoints, f^* and g^* . Then σ induces a morphism $\sigma^*: g^* \rightarrow f^*$ which we call the σ 's (*right*) *mate*. We define σ^* as follows:

$$g^* \longrightarrow g^* \circ f \circ f^* \xrightarrow{g^* \circ \sigma \circ f^*} g^* \circ g \circ f^* \longrightarrow f^*$$

We may make a similar definition for left adjoints.

Proposition. If a right adjoint exists, then it is essentially unique.

Proof. Suppose $\sigma: f \rightarrow g$ is an isomorphism and suppose that f and g have right adjoints. Then σ induces an isomorphism $g^* \rightarrow f^*$ which respects the adjunction. This is best seen by as a proof-by-diagrams which we leave as an exercise to the reader. \square

Appendix C

Sesquicategories and Gray-Categories

Roughly speaking, a sesquicategory is a 2-category which lacks horizontal composition of 2-morphisms. This lack of composition means that we cannot state, and therefore don't demand, an interchange rule. We retain, however, a horizontal composition of 2-morphisms with 1-morphisms.

Definition. A *sesquicategory*, or $1\frac{1}{2}$ -*category*, C has data as follows.

- a collection Obj_C of *objects*;
- a collection $\text{Hom}_C(X, Y)$ of *1-morphisms* for each pair of objects $X, Y \in \text{Obj}_C$;
- a collection $2\text{-Hom}_C(a, b)$ of *2-morphisms* for each pair of 1-morphisms $a, b \in \text{Hom}_C(X, Y)$.

We also have structure as follows.

- a binary operation

$$\text{Hom}_C(X, Y) \times \text{Hom}_C(Y, Z) \xrightarrow{\circ_x} \text{Hom}_C(X, Z)$$

called *horizontal composition*,

- a binary operation

$$2\text{-Hom}_C(a, b) \times 2\text{-Hom}_C(b, c) \xrightarrow{\circ_y} 2\text{-Hom}_C(a, c)$$

called *vertical composition*,

- a nullary operation

$$* \xrightarrow{1_X} \text{Hom}_C(X, X)$$

called the *identity for objects*.

- a nullary operation

$$* \xrightarrow{1_a} 2\text{-Hom}_C(a, a)$$

called the *identity for 1-morphisms*.

- we have a 2-morphism $a \circ_x \alpha \in 2\text{-Hom}_C(a \circ_x b, a \circ_x c)$ called the *whiskering of α by a on the left* for any

- objects $W, X, Y \in \text{Obj}_C$,
- 1-morphism $a \in \text{Hom}_C(W, X)$,
- 1-morphisms $b, c \in \text{Hom}_C(X, Y)$, and
- 2-morphism $\alpha \in 2\text{-Hom}_C(b, c)$,

- we have a 2-morphism $\alpha \circ_x d \in 2\text{-Hom}_C(b \circ_x d, c \circ_x d)$ called the *whiskering of α by d on the right* for any

- objects $X, Y, Z \in \text{Obj}_C$,
- 1-morphism $d \in \text{Hom}_C(Y, Z)$,
- 1-morphisms $b, c \in \text{Hom}_C(X, Y)$, and
- 2-morphism $\alpha \in 2\text{-Hom}_C(b, c)$.

We require the following rules.

- Horizontal composition is associative and unital.

$$(a \circ_x b) \circ_x c = a \circ_x (b \circ_x c),$$

$$1_X \circ_x a = a,$$

$$a \circ_x 1_Y = a.$$

- Vertical composition is associative and unital.

$$(\alpha \circ_y \beta) \circ_y \gamma = \alpha \circ_y (\beta \circ_y \gamma),$$

$$1_a \circ_y \alpha = \alpha,$$

$$\alpha \circ_y 1_b = \alpha.$$

- Whiskering distributes over vertical composition.

$$\begin{aligned} a \circ_x (\alpha \circ_y \beta) &= (a \circ_x \alpha) \circ_y (a \circ_x \beta), \\ (\alpha \circ_y \beta) \circ_x d &= (\alpha \circ_x d) \circ_y (\beta \circ_x d). \end{aligned}$$

- Whiskering is unital.

$$\begin{aligned} 1_X \circ_x \alpha &= \alpha, \\ \alpha \circ_x 1_Y &= \alpha. \end{aligned}$$

- Whiskering satisfies bimodule relations.

$$\begin{aligned} (a \circ_x b) \circ_x \alpha &= a \circ_x (b \circ_x \alpha), \\ \alpha \circ_x (c \circ_x d) &= (\alpha \circ_x c) \circ_x d, \\ (b \circ_x \alpha) \circ_x c &= b \circ_x (\alpha \circ_x c). \end{aligned}$$

Functors of sesquicategories are straightforward to define. Notice, also, that pseudo-natural transformations do not use horizontal composites. Indeed, they only use vertical composites of whiskered 2-morphisms and so their definition goes through for sesquicategories.

Definition. A *Gray-category* C is a sesquicategory with extra data, structure, and rules. In addition to having objects, 1-morphisms, and 2-morphisms, we have

- a collection $\mathbf{3}\text{-Hom}_C(\alpha, \beta)$ of *3-morphisms* for each pair of 2-morphisms $\alpha, \beta \in \mathbf{2}\text{-Hom}_C(a, b)$.

We have structure as follows.

- a binary operation

$$\mathbf{3}\text{-Hom}_C(\alpha, \beta) \times \mathbf{3}\text{-Hom}_C(\gamma, \delta) \xrightarrow{\circ_y} \mathbf{3}\text{-Hom}_C(\alpha \circ_y \gamma, \beta \circ_y \delta).$$

- a binary operation

$$\mathbf{3}\text{-Hom}_C(\alpha, \beta) \times \mathbf{3}\text{-Hom}_C(\beta, \gamma) \xrightarrow{\circ_z} \mathbf{3}\text{-Hom}_C(\alpha, \gamma).$$

- a nullary operation

$$* \xrightarrow{1_\alpha} \mathbf{3}\text{-Hom}_C(\alpha, \alpha).$$

- We have a 3-morphism $a \circ_x \Gamma \in 3\text{-Hom}_C(a \circ_x \alpha, a \circ_x \beta)$ for any
 - objects $W, X, Y \in \text{Obj}_C$,
 - 1-morphism $a \in \text{Hom}_C(W, X)$,
 - 1-morphisms $b, c \in \text{Hom}_C(X, Y)$,
 - 2-morphisms $\alpha, \beta \in 2\text{-Hom}_C(b, c)$, and
 - 3-morphism $\Gamma \in 3\text{-Hom}_C(\alpha, \beta)$.
- We have a 3-morphism $\Gamma \circ_x d \in 3\text{-Hom}_C(\alpha \circ_x d, \beta \circ_x d)$ for any
 - objects $X, Y, Z \in \text{Obj}_C$,
 - 1-morphism $d \in \text{Hom}_C(Y, Z)$,
 - 1-morphisms $b, c \in \text{Hom}_C(X, Y)$,
 - 2-morphisms $\alpha, \beta \in 2\text{-Hom}_C(b, c)$, and
 - 3-morphism $\Gamma \in 3\text{-Hom}_C(\alpha, \beta)$.
- We have an invertible 3-morphism

$$\Theta_{\alpha, c, \beta} \in 3\text{-Hom}_C((\alpha \circ_x c \circ_x d) \circ_y (b \circ_x c \circ_x \beta), (a \circ_x c \circ_x \beta) \circ_y (\alpha \circ_x c \circ_x e))$$

called the *Gray-interchanger* for any

- objects $W, X, Y, Z \in \text{Obj}_C$,
- 1-morphisms $a, b \in \text{Hom}_C(W, X)$, $c \in \text{Hom}_C(X, Y)$, $d, e \in \text{Hom}_C(Y, Z)$, and
- 2-morphisms $\alpha \in 2\text{-Hom}_C(a, b)$, $\beta \in 2\text{-Hom}_C(d, e)$.

These satisfy the following rules. Note that each of the Gray-interchanger rules has a similar rule for Θ^{-1} that we will also impose.

- The compositions are associative and unital.

$$\begin{aligned} (\Gamma \circ_y \Delta) \circ_y \Lambda &= \Gamma \circ_y (\Delta \circ_y \Lambda), \\ 1_{1_a} \circ_y \Gamma &= \Gamma \quad \text{and} \quad \Gamma \circ_y 1_{1_b} = \Gamma, \\ (\Gamma \circ_z \Delta) \circ_z \Lambda &= \Gamma \circ_z (\Delta \circ_z \Lambda), \\ 1_\alpha \circ_z \Gamma &= \Gamma \quad \text{and} \quad \Gamma \circ_z 1_\beta = \Gamma. \end{aligned}$$

- We have an interchange rule for the y and z -compositions.

$$(\Gamma \circ_y \Delta) \circ_z (\Lambda \circ_y \Omega) = (\Gamma \circ_z \Lambda) \circ_y (\Delta \circ_z \Omega).$$

- Whiskering distributes over the compositions.

$$a \circ_x (\Gamma \circ_y \Delta) = (a \circ_x \Gamma) \circ_y (a \circ_x \Delta),$$

$$(\Gamma \circ_y \Delta) \circ_x d = (\Gamma \circ_x d) \circ_y (\Delta \circ_x d),$$

$$a \circ_x (\Gamma \circ_z \Delta) = (a \circ_x \Gamma) \circ_z (a \circ_x \Delta),$$

$$(\Gamma \circ_z \Delta) \circ_x d = (\Gamma \circ_x d) \circ_z (\Delta \circ_x d).$$

- Whiskering is unital.

$$1_X \circ_x \Gamma = \Gamma,$$

$$\Gamma \circ_x 1_Y = \Gamma.$$

- Whiskering satisfies bimodule relations.

$$(a \circ_x b) \circ_x \Gamma = a \circ_x (b \circ_x \Gamma),$$

$$\Gamma \circ_x (c \circ_x d) = (\Gamma \circ_x c) \circ_x d,$$

$$(b \circ_x \Gamma) \circ_x c = b \circ_x (\Gamma \circ_x c).$$

- The Gray-interchanger is natural.

$$((\Gamma \circ_x c \circ_x d) \circ_y (b \circ_x c \circ_x \gamma)) \circ_z \Theta_{\beta,c,\gamma} = \Theta_{\alpha,c,\gamma} \circ_z ((a \circ_x c \circ_x \gamma) \circ_y (\Gamma \circ_x c \circ_x e))$$

$$((\beta \circ_x c \circ_x d) \circ_y (b \circ_x c \circ_x \Delta)) \circ_z \Theta_{\beta,c,\delta} = \Theta_{\beta,c,\gamma} \circ_z ((a \circ_x c \circ_x \Delta) \circ_y (\beta \circ_x c \circ_x e))$$

for any

- objects W, X, Y, Z ,
- 1-morphisms $a, b \in \text{Hom}_C(W, X)$, $c \in \text{Hom}_C(X, Y)$, $d, e \in \text{Hom}_C(Y, Z)$,
- 2-morphisms $\alpha, \beta \in 2\text{-Hom}_C(a, b)$, $\gamma, \delta \in 2\text{-Hom}_C(d, e)$, and
- 3-morphisms $\Gamma \in 3\text{-Hom}_C(\alpha, \beta)$, $\Delta \in 3\text{-Hom}_C(\gamma, \delta)$.

- The Gray-interchanger satisfies braid relations.

$$\begin{aligned}
& ((\Theta_{\alpha,X,\beta} \circ_x e) \circ_y (b \circ_x d \circ_x \gamma)) \\
& \quad \circ_z ((a \circ_x \beta \circ_x e) \circ_y \Theta_{\alpha,d,\gamma}) \\
& \quad \circ_z ((a \circ_x \Theta_{\beta,Y,\gamma}) \circ_y (\alpha \circ_x d \circ_x f)) \\
& = \\
& ((\alpha \circ_x c \circ_x e) \circ_y (b \circ_x \Theta_{\beta,Y,\gamma})) \\
& \quad \circ_z (\Theta_{\alpha,c,\gamma} \circ_y (b \circ_x \beta \circ_x f)) \\
& \quad \circ_z ((a \circ_x c \circ_x \gamma) \circ_y (\Theta_{\alpha,X,\beta} \circ_x f))
\end{aligned}$$

for any

- objects $W, X, Y, Z \in \text{Obj}_C$,
- 1-morphisms $a, b \in \text{Hom}_C(W, X)$, $c, d \in \text{Hom}_C(X, Y)$, $e, f \in \text{Hom}_C(Y, Z)$, and
- 2-morphisms $\alpha \in 2\text{-Hom}_C(a, b)$, $\beta \in 2\text{-Hom}_C(c, d)$, $\gamma \in 2\text{-Hom}_C(e, f)$.

- The Gray-interchanger satisfies rope relations.

$$\begin{aligned}
\Theta_{(\alpha \circ_y \beta),d,\gamma} &= ((\alpha \circ_x d \circ_x e) \circ_y \Theta_{\beta,d,\gamma}) \circ_z (\Theta_{\alpha,d,\gamma} \circ_y (\beta \circ_x d \circ_x f)) \\
\Theta_{\beta,d,(\gamma \circ_y \delta)} &= (\Theta_{\beta,d,\gamma} \circ_y (c \circ_x d \circ_x \delta)) \circ_z ((b \circ_x d \circ_x \gamma) \circ_y \Theta_{\beta,d,\delta})
\end{aligned}$$

for any

- objects $W, X, Y, Z \in \text{Obj}_C$
- 1-morphisms $a, b, c \in \text{Hom}_C(W, X)$, $d \in \text{Hom}_C(X, Y)$, $e, f, g \in \text{Hom}_C(Y, Z)$, and
- 2-morphisms $\alpha \in 2\text{-Hom}_C(a, b)$, $\beta \in 2\text{-Hom}_C(b, c)$, $\gamma \in 2\text{-Hom}_C(e, f)$,
 $\delta \in 2\text{-Hom}_C(f, g)$.

Our definition of a Gray-category is rather verbose. We may package this up neatly using the theory of categories enriched in a monoidal category. While we won't review enriched categories, we will give a brief introduction to the monoidal category Gray of 2-categories and 2-functors. We do not equip Gray with the standard Cartesian product of categories. Instead, we give Gray a tensor product which is best understood in terms of cubical functors.

Definition. Suppose C_1, C_2, \dots, C_n , and D are 2-categories. A *cubical functor*

$$C_1 \times \cdots \times C_n \xrightarrow{f} D$$

is a weak 2-functor which satisfies the following. For any pair of 1-morphisms of the following form ($1 \leq i \leq n$)

$$a = \left(X_1 \xrightarrow{a_1} Y_1, \dots, X_{i-1} \xrightarrow{a_{i-1}} Y_{i-1}, X_i \xrightarrow{a_i} Y_i, Y_{i+1} \xrightarrow{1} Y_{i+1}, \dots, Y_n \xrightarrow{1} Y_n \right)$$

$$b = \left(Y_1 \xrightarrow{1} Y_1, \dots, Y_{i-1} \xrightarrow{1} Y_{i-1}, Y_i \xrightarrow{b_i} Z_i, Y_{i+1} \xrightarrow{b_{i+1}} Z_{i+1}, \dots, Y_n \xrightarrow{b_n} Z_n \right)$$

we have $f(ab) = f(a)f(b)$ and the structural morphism $f(ab) \cong f(a)f(b)$ is the identity.

Consider a cubical functor of two variables, $C_1 \times C_2 \xrightarrow{\otimes} D$. For any 1-morphism (a_1, a_2) , we may write both

$$a_1 \otimes a_2 = (a_1 \otimes 1) \circ (1 \otimes a_2), \text{ and}$$

$$a_1 \otimes a_2 \cong (1 \otimes a_2) \circ (a_1 \otimes 1).$$

Notice that the first line is an equality, but the second line uses a structural 2-morphism for \otimes . Putting these lines together, the structural 2-morphism may be thought of as a square,

$$(a_1 \otimes 1) \circ (1 \otimes a_2) \cong (1 \otimes a_2) \circ (a_1 \otimes 1).$$

So while we prefer to think of $a_1 \otimes a_2$ as “doing a_1 before doing a_2 ” (LHS), we also have a chosen isomorphism to the other order (RHS).

Now let’s consider a cubical functor of three variables, $C_1 \times C_2 \times C_3 \xrightarrow{\otimes} D$. We note that for fixed objects X_1, X_2 , and X_3 ,

$$\otimes(X_1, -, -), \quad \otimes(-, X_2, -), \quad \text{and} \quad \otimes(-, -, X_3)$$

are cubical functors of 2-variables. Our cubical condition allows us to write an equality for any 1-morphism (a_1, a_2, a_3) ,

$$\otimes(a_1, a_2, a_3) = \otimes(a_1, 1, 1) \circ \otimes(1, a_2, 1) \circ \otimes(1, 1, a_3).$$

We may paste together the structural 2-morphisms to build composite 2-morphisms,

$$\begin{aligned} & \otimes(a_1, 1, 1) \circ \otimes(1, a_2, 1) \circ \otimes(1, 1, a_3) \\ \Rightarrow & \otimes(1, a_2, 1) \circ \otimes(a_1, 1, 1) \circ \otimes(1, 1, a_3) \\ \Rightarrow & \otimes(1, a_2, 1) \circ \otimes(1, 1, a_3) \circ \otimes(a_1, 1, 1) \\ \Rightarrow & \otimes(1, 1, a_3) \circ \otimes(1, a_2, 1) \circ \otimes(a_1, 1, 1) \end{aligned}$$

and

$$\begin{aligned}
& \otimes(a_1, 1, 1) \circ \otimes(1, a_2, 1) \circ \otimes(1, 1, a_3) \\
\Rightarrow & \otimes(a_1, 1, 1) \circ \otimes(1, 1, a_3) \circ \otimes(1, a_2, 1) \\
\Rightarrow & \otimes(1, 1, a_3) \circ \otimes(a_1, 1, 1) \circ \otimes(1, a_2, 1) \\
\Rightarrow & \otimes(1, 1, a_3) \circ \otimes(1, a_2, 1) \circ \otimes(a_1, 1, 1).
\end{aligned}$$

The coherence relations for \otimes , thought of as a weak 2-functor, ensure that these composite 2-morphisms agree. If we like pasting diagrams, we can think of these compositions taking place on the boundary of a cube. We might also recognize this as a braid relation.

Definition. Suppose A and B are 2-categories. We'll let $\text{Ps}(A, B)$ denote the 2-category of (strict) 2-functors $A \rightarrow B$, pseudonatural transformations, and modifications.

There is a tensor product \square that makes $\text{Ps}(B, C)$ the “internal hom” for the 2-category 2-Cat of small 2-categories. We won't define \square , but just note that it is characterized by the following proposition.

Proposition. Suppose A , B , and C are 2-categories. The following are equivalent.

- a cubical functor $A \times B \rightarrow C$, and
- a 2-functor $A \rightarrow \text{Ps}(B, C)$,
- a 2-functor $A \square B \rightarrow C$.

Proof. See Gordon-Power-Street [GPS95]. □

Definition. We'll let Gray denote the monoidal category of 2-categories and (strict) 2-functors. The monoidal product is given by \square .

A Gray-category C , then, is a category with Hom sets enriched in Gray . So for any two objects X and Y , we have that $\text{Hom}_C(X, Y)$ is a 2-category. By the previous proposition, we may think of composition as a cubical functor,

$$\text{Hom}_C(X, Y) \times \text{Hom}_C(Y, Z) \xrightarrow{\circ_x} \text{Hom}_C(X, Z).$$

The structural 2-morphism squares record Gray-interchangers for seams separated by a region. We get our Gray-interchanger “over a 1-morphism” as the structural morphism for the cubical functor in two variables,

$$\circ_x(-, c, -): \text{Hom}_C(W, X) \times \text{Hom}_C(X, Y) \times \text{Hom}_C(Y, Z) \rightarrow \text{Hom}_C(W, Z),$$

for a fixed 1-morphism $c \in \text{Hom}_C(X, Y)$.

Bibliography

- [Bar05] Bruce Bartlett. Categorical aspects of topological quantum field theories. Master's thesis, Utrecht University, September 2005.
- [BD95] John C. Baez and James Dolan. Higher-dimensional algebra and topological quantum field theory. *Journal of Mathematical Physics*, 36(11):6073–6105, 1995.
- [CG07] Eugenia Cheng and Nick Gurski. Towards an n-category of cobordisms. *Theory and Applications of Categories*, 18(10):274–302, 2007.
- [CS99] J. Scott Carter and Masahico Saito. *Knotted Surfaces and Their Diagrams*. American Mathematical Society, 1999.
- [CSR97] J. Scott Carter, Masahico Saito, and Joachim Rieger. A combinatorial description of knotted surfaces and their isotopies. *Advances in Mathematics*, 127, April 1997.
- [FY89] Peter Freyd and David N. Yetter. Braided compact closed categories with applications to low-dimensional topology. *Adv. Math.*, 77(2):156–182, 1989.
- [FY92] Peter Freyd and David N. Yetter. Coherence theorems via knot theory. *J. Pure Appl. Algebra*, 78:49–76, 1992.
- [GM98] Mark Goresky and Robert MacPherson. *Stratified Morse Theory*. Springer, 1998.
- [GPS95] Robert Gordon, A. J. Power, and Ross Street. *Coherence for Tricategories*, volume 117. American Mathematical Society, September 1995.
- [Gra74] John W. Gray. *Formal Category Theory: Adjointness for 2-Categories*. Lecture notes in mathematics. Springer-Verlag, 1974.
- [Gur07] Nick Gurski. *An algebraic theory of tricategories*. PhD thesis, U Chicago, March 2007.
- [Gur11a] Nick Gurski. Biequivalences in tricategories. *CORD Conference Proceedings*, February 2011.
- [Gur11b] Nick Gurski. Loop spaces, and coherence for monoidal and braided monoidal bicategories. *ArXiv e-prints*, February 2011.
- [Hir76] Morris W. Hirsch. *Differential Topology*. Springer, 1976.

- [JS91a] André Joyal and Ross Street. The geometry of tensor calculus I. *Advances in Math.*, 88:55–112, 1991.
- [JS91b] André Joyal and Ross Street. The geometry of tensor calculus II. Available on maths.mq.edu.au/~street/, 1991.
- [JS91c] André Joyal and Ross Street. An introduction to tannaka duality and quantum groups. *Lecture Notes in Math.*, 1488, 1991.
- [JS93] André Joyal and Ross Street. Braided tensor categories. *Advances in Math.*, 102:20–78, 1993.
- [Lac04] Stephen Lack. A quillen model structure for bicategories. *K-Theory*, 33:185–197, 2004.
- [Lac11] Stephen Lack. A quillen model structure for gray-categories. *Journal of K-theory*, 9(2), October 2011.
- [Lau11] Aaron Lauda. An introduction to diagrammatic algebra and categorified quantum $\mathfrak{sl}(2)$. *ArXiv e-prints*, June 2011.
- [Lee03] John Lee. *Introduction to Smooth Manifolds*. Springer, 2003.
- [Lei04] Tom Leinster, editor. *Higher Operads, Higher Categories*, volume 298 of *London Mathematical Society Lecture Note Series*. Cambridge University Press, 2004.
- [Lew72] Geoffrey Lewis. Coherence for a closed functor. *Lecture Notes in Mathematics*, 281:148–195, 1972.
- [Lur09] Jacob Lurie. On the classification of topological field theories. Draft, May 2009.
- [Mac63] Saunders MacLane. Natural associativity and commutativity. *Rice Univ. Studies*, 49(4):28–46, 1963.
- [Mac71] Saunders MacLane. *Categories for the Working Mathematician*. Springer, 1971.
- [Mil63] John Milnor. *Morse Theory*. Princeton University Press, 1963.
- [Mil65] John Milnor. *Lectures on the h-Cobordism Theorem*. Princeton University Press, 1965.
- [MP85] Saunders MacLane and Robert Paré. Coherence for bicategories and indexed categories. *Journal of Pure and Applied Algebra*, 37:59–80, 1985.
- [MSS02] Martin Markl, Steve Shnider, and Jim Stasheff. *Operads in Algebra, Topology, and Physics*. American Mathematical Society, 2002.
- [MW10] Scott Morrison and Kevin Walker. The blob complex. *ArXiv e-prints*, September 2010.
- [MW11] Scott Morrison and Kevin Walker. Higher categories, colimits, and the blob complex. *Proceedings of the National Academy of Sciences*, 108:8139–8145, 2011.

- [MZ08] Mihaly Makkai and Marek Zawadowski. The category of 3-computads is not cartesian closed. *Journal of Pure and Applied Algebra*, June 2008.
- [Pow89] A. J. Power. A general coherence result. *Journal of Pure and Applied Algebra*, 57:165–173, 1989.
- [Pow95] A. J. Power. Why tricategories? *Information and Computation*, 120:251–262, August 1995.
- [Sel09] Peter Selinger. A survey of graphical languages for monoidal categories. *ArXiv e-prints*, August 2009.
- [Shu94] Mei Chee Shum. Tortile tensor categories. *Journal of Pure and Applied Algebra*, 93(1):57–110, 1994.
- [SP09] Christopher Schommer-Pries. *The Classification of Two-Dimensional Extended Topological Field Theories*. PhD thesis, UC Berkeley, May 2009.
- [Str80] Ross Street. Fibrations in bicategories. *Cahiers de Topologie et Geometrie Differentielle*, 101(2):111–160, 1980.
- [Wol73] Harvey Wolff. V-cat and v-graph. *Journal of Pure and Applied Algebra*, 4(2):124–135, 1973.
- [Yet89] David Yetter. Category theoretic representations of knotted graphs in S^3 . *Advances in Math.*, 77:137–155, 1989.