Universality and Examples in the Context of Functorial Semi-Norms

Dissertation zur Erlangung des Doktorgrades
der Naturwissenschaften (Dr. rer. nat.)
der Fakultät für Mathematik

der Universität Regensburg

vorgelegt von

Johannes Witzig,
geboren Prem,

aus Regensburg

im Jahr 2022

Promotionsgesuch eingereicht am: 21.12.2022

Die Arbeit wurde angeleitet von: Prof. Dr. Clara Löh

Prüfungsausschuss:	Vorsitzender:	Prof. Dr. Moritz Kerz
	1. Gutachter:	Prof. Dr. Clara Löh
	2. Gutachter:	PD Dr. Georgios Raptis
	weiterer Prüfer:	Prof. Stefan Friedl, PhD

Introduction

Deeply rooted in virtually all aspects of geometry, is the desire to assign and measure lengths. One mathematical incarnation in the context of linear structures are *(semi-) norms*, whose direct or indirect use underlies nearly all mathematics.

A much more recent successful concept in mathematics is the language of categories, functors and natural transformations, commonly subsumed under the term *category theory*.

An interesting symbiosis happened, when Gromov introduced the notion of *functorial semi-norms* on singular homology [Grv1][Grv2, paragraph 5.34]: For a fixed $n \in \mathbb{N}$, one assigns to each topological space X a semi-norm on the $\mathbb{R}$-vector space $H_n(X;\mathbb{R})$, such that for every continuous map $X \to Y$, the induced homomorphism $H_n(X;\mathbb{R}) \to H_n(Y;\mathbb{R})$ is norm non-increasing.

Originally, Gromov used the ℓ^1-norm on the singular chain complex with real coefficients, associated to the canonical basis consisting of singular simplices, in order to define the *simplicial volume* of closed manifolds. For an oriented, closed manifold M, the latter is given by

$$\inf\Big\{\sum\nolimits_j |a_j| \;\Big|\; \sum\nolimits_j a_j \cdot \sigma_j \text{ represents the } \mathbb{R}\text{-fundamental class of } M\Big\},$$

and thus measures the complexity of the fundamental class of M. Though this might be surprising for such a topological invariant, Gromov's paper "Volume and bounded cohomology" [Grv1] successfully demonstrates a variety of connections between the simplicial volume and Riemannian geometry.

A useful tool to study simplicial volume is so-called ℓ^1*-homology* of topological spaces: it arises as the homology of the singular chain complex *after* completion with respect to the ℓ^1-norm in terms of functional analysis. Although there is no direct analogue to the classical universal coefficient theorem, ℓ^1-homology is in some sense dual to bounded cohomology [Lö1, Ch. 3].

In the present thesis, we investigate the following aspects of functorial semi-norms in the context of topology, for which we give more detailed summaries in the further sections of this introduction:

Following Gromov's idea, one can also try to refine other functors to vector spaces (or Abelian groups) with semi-norms, leading to the abstract definition of functorial semi-norms [Lö3]. For a fixed functor, we define a relation among all its functorial semi-norms, whose minimal elements we call *universal*. We are then interested in the existence of such universal functorial semi-norms.

Crowley and Löh established a bidirectional correspondence between functorial semi-norms on singular homology and so-called *inflexible manifolds* [CL]. The construction

of such manifolds is based on the construction and study of certain differential graded algebras, which are purely algebraic objects. As such they are very amenable to computations, not only by humans but also via computer programs. We present fragments of a software that facilitates such computations, some new examples that we found in this way, and two results about algorithmic decidability.

It is a well-known fact, that ℓ^1-homology does *not* satisfy the excision axiom in the sense of Eilenberg and Steenrod. On the other hand, the fact that singular homology satisfies the excision axiom is already visible at the level of the singular chain complex functor, namely the latter is *excisive* in the sense of Goodwillie calculus [Go1][Go2][Go3] [Lu2, Ch. 6]. The latter, however, also provides the framework for constructing a universal (or best) excisive approximation to a given functor. We apply this theory to the ℓ^1-chain complex functor and show that its excisive approximation vanishes.

(Universal) Functorial semi-norms

Fix a functor $F\colon C \to \mathsf{Vect}_K$ to vector spaces over a normed field K, for example $\mathbb{Q}$ or $\mathbb{R}$ with the absolute value, and denote by $\mathsf{sn}\,\mathsf{Vect}_K$ the category of semi-normed K-vector spaces. A *functorial semi-norm on* F is a lift of F to a functor $C \to \mathsf{sn}\,\mathsf{Vect}_K$ (Definition 1.2, in slightly more generality). A variant of this, with targets in Abelian groups, has been considered by Löh [Lö3].

The classic examples, studied by Gromov [Grv1], are the ℓ^1-semi-norm on singular homology and the ℓ^∞-semi-norm on singular and bounded cohomology, both with real coefficients. Later, he also gave other examples of functorial semi-norms on singular homology [Grv2, Sec. 5.G$_+$ and 5.H$_+$]. An interesting application of functorial semi-norms are degree theorems, i.e. results that constrain the value of the mapping degree of continuous maps between manifolds [CL, Rem. 2.6].

While there always exists a functorial semi-norm on F, by virtue of the zero semi-norm, it is a non-trivial task to construct interesting functorial semi-norms on a given functor, even on singular homology. In case of the latter, Crowley and Löh [CL, Sec. 4] used an approach of representing homology classes by linear combinations of images of fundamental classes of manifolds, in order to obtain examples with specific properties (see page vi). Briefly, their idea works as follows: To a given function $\nu\colon \mathrm{Mfd}_d \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ on the class of oriented closed connected manifolds of dimension d, they associate a functorial semi-norm that is given on a homology class $a \in H_d(X;\mathbb{R})$ by

$$\inf\Big\{\sum_{j=1}^{N} |a_j| \cdot \nu(M_j) \;\Big|\; \alpha = \sum_{j=1}^{N} a_j \cdot H_d(f_j;\mathbb{R})[M_j]_{\mathbb{R}} \quad \text{with} \\ a_j \in \mathbb{R},\ M_j \in \mathrm{Mfd}_d,\ f_j\colon M_j \to X\Big\}.$$

For functions ν that are compatible with mapping degrees, they can control the behavior of the semi-norm on fundamental classes of manifolds [CL, Thm. 4.2].

In particular, this can be used to prove non-vanishing results. Changing the perspec-

tive a bit, one can systematically study how functorial semi-norms relate in terms of vanishing. This leads to the following definition: a functorial semi-norm is *universal* if it vanishes on as few classes as possible among all finite functorial semi-norms on a given functor (Definition 2.5). It is known [FL, Thm. 1.2] that the ℓ^1-semi-norm is *not* universal on singular homology in most dimensions, thus raising the question whether universal functorial semi-norms exist on singular homology [FL, Q. 4.2].

Towards an answer to this question, we prove the following result (Theorem 2.1) in Chapter 2, in which $\mathsf{Vect}_K^{\mathrm{fin}}$ and Vect_K^{ω} denote the categories of K-vector spaces of finite and countable dimension, respectively:

Theorem (existence of universal functorial semi-norms [LW]). Let C be a category that admits a skeleton with at most countably many objects. Let K be a normed field and let $F\colon C \to \mathsf{Vect}_K$ be a functor.

(i) If K is countable and if F maps to Vect_K^{ω}, then F admits a universal finite functorial semi-norm.

(ii) If F maps to $\mathsf{Vect}_K^{\mathrm{fin}}$, then F admits a universal finite functorial semi-norm.

Restricting attention to finite CW-complexes, this theorem allows us to derive the following result (Corollary 2.2) regarding the singular homology functor:

Corollary. Let $d \in \mathbb{N}$ and let K be a normed field. Then the singular homology functor $H_d(\bullet; K)\colon \mathsf{CW}^{\mathrm{fin}} \to \mathsf{Vect}_K$ admits a universal functorial semi-norm, where $\mathsf{CW}^{\mathrm{fin}}$ denotes the full subcategory of Top on those spaces that are homotopy equivalent to a finite CW-complex.

As ingredients for the proof of the theorem in Section 2.5, we

- generalize the method of Crowley and Löh for constructing functorial semi-norms to arbitrary functors F as above (Section 1.3);
- introduce, as a tool, the so-called *vanishing locus* of a functorial semi-norm (Section 2.3); and
- use both of the above, together with a suitable diagonalization argument, in order to construct a functorial semi-norm that is universal with respect to countably many given ones (Section 2.4).

Additionally, we show that the existence of a universal functorial semi-norm on a functor is compatible with equivalences of categories (Section 2.2), and we include an example of a functor that does *not* admit a universal functorial semi-norm (Section 2.6).

This part is based on joint work with Clara Löh and follows very closely the respective article by Löh and the author of this thesis [LW].

Inflexibility from a computational perspective

It follows from Gromov's work [Grv1], that the ℓ^1-semi-norm vanishes on all classes of non-zero degree of simply-connected topological spaces. Later, he further suggested [Grv2, Remark (b) in paragraph 5.35]: "It is probable that every natural norm vanishes on $H_*(X)$ for all simply connected X". Restated as an open question [CL, Q. 1.1], Crowley and Löh showed [CL, Thm. 1.2], that there indeed exist functorial semi-norms on singular homology (of certain degrees) that are positive and finite on some homology classes of simply-connected spaces. For their proof, they use

- their general method for the construction of functorial semi-norms that we already sketched (page iv). We review their choice of the function ν in Section 3.1. Furthermore, they need
- as additional input simply-connected, *inflexible manifolds* (Section 3.2), that is manifolds with the property that any self-map has mapping degree 0, 1, or -1.

Inflexible simply-connected manifolds, however, are not easily found in nature. Arkowitz and Lupton [AL] gave the first examples, which Crowley and Löh used as a starting point to construct more. In Examples 3.9, we list what is known on the subject by now.

All of the known examples of inflexible, simply-connected manifolds are produced by means of rational homotopy theory. Briefly, this works as follows:

- Find inflexible minimal Sullivan algebras, a certain class of differential graded algebras (Section 3.3).
- Realize such an algebra as the minimal model of a certain type of CW-complex.
- Show that the latter admits rationally equivalent manifolds.

The key observation is, that once this machinery is set up, one can do all experiments and calculations on the purely algebraic side.

As minimal Sullivan algebras are determined by their generators and the restriction of the differential to the latter, it is easy to represent and analyse their properties in computer programs. In the first two section of Chapter 4, we present such a software and showcase some of its features. We include an example of a minimal Sullivan algebra that induces a new simply-connected, inflexible manifold of dimension 38 (Example 4.6). Among such manifolds, this is the least dimensional example known so far.

One important property of minimal Sullivan algebras is *ellipticity*, as it implies [FHT, Prop. 38.3], that the cohomology algebra is a Poincaré algebra. In Section 4.3, we prove the following result (Theorem 4.7):

Theorem (decidable ellipticity). Let (A, δ) be a connected cochain algebra with the property that all generators of even degree are cocycles. Then (A, δ) is elliptic if and only if all generators of even degree are cohomologically nilpotent. In particular, it is algorithmically decidable whether (A, δ) is elliptic if (A, δ) is a minimal Sullivan algebra.

We note, that as per our conventions (Definition 4.2), the underlying graded algebra of a cochain algebra is always a finitely generated, free graded-commutative algebra.

The other property of minimal Sullivan algebras that we are obviously interested in, is inflexibility. In order to treat the latter in our software, we introduce the notion of a *generic morphism* between cochain algebras in Section 4.4. With this tool, we then prove (Theorem 4.9):

Theorem (decidable (in)flexibility). Let k be countable and let $\overline{k}$ be an algebraic closure of k. Let (A, δ) be a connected, elliptic Sullivan algebra.

(i) Let x be a generator of A, such that some power of x represents a fundamental class for (A, δ). Then it is algorithmically decidable, whether (A, δ) is inflexible over $\overline{k}$.

(ii) Let $d \in \mathbb{N}$ be the formal dimension of (A, δ) and let $m \in M(A)_d$ be a representative of a fundamental class for (A, δ). Let $a \in \overline{k}$. Then it is algorithmically decidable, whether (A, δ) over $\overline{k}$ admits a cochain algebra endomorphism of mapping degree a.

Here, $M(A)_d$ denotes the set of all products of generators of A of total degree d (Definition 4.8).

Excisive approximation of ℓ^1-homology

Taking the norm-completion of the singular chain complex $C(\bullet, \bullet; \mathbb{R})$ with respect to the ℓ^1-norm yields the *ℓ^1-chain complex* $C^{\ell^1}(X, A)$ (Sections 1.1 and 1.4) [Lö2, Sec. 3.1][Lö1, Introduction and Ch. 2]. Its homology is the *ℓ^1-homology of* (X, A), the topological dual of (either $C(X, A; \mathbb{R})$ or) $C^{\ell^1}(X, A)$ is the *bounded cochain complex* $C_b(X, A)$ *of* (X, A), and cohomology of the latter is the *bounded cohomology of* (X, A).

Bounded cohomology was used extensively in Gromov's work [Grv1], and since then various connections to other mathematical areas were established, some of which are summarized in Monod's nice "invitation to bounded cohomology" [Mo]. Though its link to ℓ^1-homology is not as tight as in the case of singular (co)homology, there is a helpful duality between ℓ^1-homology and bounded cohomology [Lö1, Ch. 3]. Both notions enable the formulation and proof of structural statements about simplicial volume.

However, there is one "problem" with ℓ^1-homology: it is notoriously difficult to compute. This is mainly due to the fact, that the excision theorem fails for ℓ^1-homology (as well as for bounded cohomology), making the usual divide and conquer approach impossible. This manifests itself even in the simplest examples, such as the following [Lö2, Ex. (2.8)]: we have

$$H_k^{\ell^1}(S^1) \cong 0 \text{ for } k \in \mathbb{N}_{\geq 1}, \text{ but } H_2^{\ell^1}(S^1 \vee S^1) \ncong 0.$$

Thus, a natural question is, whether we can somehow force the theory to become excisive.

Looking at singular homology, one can observe that the excision property is already present at the level of the singular chain complex functor (Theorem A.1). This fact was noted before, e.g. by Lurie [Lu2, Sec. 1.4.2], but the author of this thesis was unable to

find a rigorous proof for it in the literature. Moreover, we explain in Appendix B, how this theorem relates to the classical excision property.

Theorem. The singular chain complex functor $\mathsf{Top} \to \mathsf{Ch}$ is excisive in the sense of Goodwillie calculus.

Better yet, under some mild assumptions on C and D, Goodwillie calculus provides us with a tool that associates to a functor $F\colon C \to D$ its best excisive approximation in the form of a universal arrow $F \Rightarrow P_1 F$, where $P_1 F$ is excisive (Theorem 6.4 and Remark 6.5). In Section 6.2, we apply this to the ℓ^1-chain complex functor on Top_* and obtain (Theorem 6.7):

Theorem (excisive approximation). There exists a best excisive approximation of $C^{\ell^1}\colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ from the right, i.e. an excisive functor $P\colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ and a natural transformation $\eta\colon C^{\ell^1} \to P$, such that for every other excisive functor $P'\colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ and natural transformation $\eta'\colon C^{\ell^1} \to P'$ there exists a natural transformation $\theta\colon P \to P'$ (unique up to homotopy) that makes the triangle

$$\begin{array}{ccc} C^{\ell^1} & \xrightarrow{\eta} & P \\ & \searrow^{\eta'} & \downarrow{\scriptstyle \theta} \\ & & P' \end{array}$$

commutative. This best approximation is trivial, i.e. $P(X) \simeq 0$ for all pointed spaces X.

At this point, we should emphasize, that the framework of Goodwillie calculus is inherently of a homotopical flavor. The natural language for this setting is provided by ∞-categories, and this is also how Lurie [Lu2, Sec. 6.1] phrased his abstraction of Goodwillie's original article [Go3]. Since we use Lurie's machinery, the reader should be aware, that the content of the previous theorem is entirely ∞-categorical. As a consequence, for example, a triangle being commutative only means: commutative up to homotopy. As we are aware of the fact, that not every reader might be fluent in the language of ∞-categories, we include sort of an independent "on the fly introduction" in Chapter 5. In lieu of proofs, we provide an ample amount of references, which should be able to satisfy anyone's thirst for knowledge.

As a further application of Goodwillie calculus, we consider the more general notion of n-excisiveness (Definition 6.10) and prove a vanishing theorem for n-excisive approximations of functors with domain Top or Top_* (Theorem 6.12). A consequence of the latter is the following:

Corollary (n-excisive approximation). Let $n \in \mathbb{N}$. A best n-excisive approximation of $C^{\ell^1}\colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ from the right exists and is trivial.

Except for the last paragraph, all results in this section already appeared in a separate article [Wi], which the respective parts of the present thesis follow very closely.

Outline

This thesis is structured as follows:

Chapter 1 starts with an introduction to the main concepts used in the rest of the present work: the ℓ^1-norm on the singular chain complex; the ℓ^1-semi-norm on singular homology; the general notion of functorial semi-norms; a basic principle for the construction of functorial semi-norms, generalizing previous work by Crowley and Löh; and the definition of ℓ^1-homology.

In Chapter 2, we discuss universal functorial semi-norms and prove the existence theorem and its topological corollary. We first recall the "carrying" relation on functorial semi-norms, which we then use to define universality. Then we investigate the interaction of (universal) functorial semi-norms with natural isomorphisms and their behavior under "weak retractions" and equivalences of categories. In the next two sections, we define vanishing loci as a tool to reason about families of functorial semi-norms and use their language to prove the main ingredient of the existence theorem. We then prove the latter, and finally show by a (counter-)example that its countability assumptions cannot be dropped in general.

Chapter 3 sets the stage for the subsequent chapter and surveys different aspects of inflexible manifolds. More precisely, we recall the definitions of the domination semi-norm and of inflexible manifolds by Crowley and Löh; we summarize their main results and we assemble a list of known constructions of inflexible manifolds; then we recall the notion of differential graded algebras; and finally, we link the latter to manifold topology by means of rational homotopy theory.

Chapter 4 consists of two parts: In Sections 4.1 and 4.2, we present our software for handling cochain algebras, computing ellipticity and inflexibility. In Sections 4.3 and 4.4, we tend to purely mathematical questions and prove the decidable ellipticity and decidable (in)flexibility theorems, respectively.

In Chapter 5, we provide a self-contained, user-oriented primer on the theory of ∞-categories. We first work towards a precise definition of ∞-category and then explain how they generalize ordinary categories. Afterwards, we highlight some analogies and differences between ordinary and ∞-categories, mainly centered around the notion of diagrams in ∞-categories. A series of remarks then sketches how interesting ∞-categories may be obtained from 1-categorical input data. In the final section, we note a few caveats regarding the nomenclature around the term "∞-category".

In Chapter 6, we define the excision property in the sense of Goodwillie calculus in the setting of ∞-categories, recall necessary results by Lurie, and then prove the excisive approximation theorem about the ℓ^1-chain complex functor. Following this, we summarize prerequisites for and give the definition of n-excisiveness. Finally, we prove the n-excisive approximation corollary.

Appendix A contains our proof of the fact that the singular chain complex functor is excisive in the aforementioned sense. As we use the model category theoretic perspective for the latter, we begin with a survey on homotopy pushouts and pullbacks in Top. We then prove the claim in two steps: a model categorical analogue is shown first, which is then translated into the ∞-categorical statement.

Appendix B explains how the abstract excision property in ∞-categories relates to classical excision in the form of Mayer-Vietoris sequences. In the final section, this is applied to the singular chain complex functor, in order to recover the traditional sequence for singular homology.

Appendix C contains the full API documentation for the software package that we present in Chapter 4.

Notation and conventions

In the present thesis, we use the following notations and conventions:

- *Foundations.* We use set theory as a foundation. Most parts do not depend on the exact theory chosen, as long as we can speak about classes as, for instance, in NBG-style set theory. However, in the ∞-categorical chapters, we occasionally need "more layers", for which we use Grothendieck universes (see Remark 5.7).

- *Setup: categories.* The categories of (small) sets, topological spaces and categories will be denoted by Set, Top and Cat, respectively.

  Rings are always unital. For a commutative ring k, we denote the category of k-modules by $\mathsf{Mod}(k)$ and the category of chain complexes over k by Ch_k. We use the abbreviation Ch for $\mathsf{Ch}_{\mathbb{Z}}$. If k is a field, we also write Vect_k for the category of k-vector spaces.

  For a category A and objects x, y of A, we let $\mathrm{Mor}_A(x, y)$ denote the set of morphisms $x \to y$; more generally, Mor_A denotes the corresponding functor $A^{\mathsf{op}} \times A \to \mathsf{Set}$, where A^{op} denotes the opposite category of A.

- *Setup: ∞-categories.* In Chapter 6 and Appendices A and B, we additionally use Top, Ch and $\mathsf{Ch}_{\mathbb{R}}$ as ∞-categories; more precisely, we localize (Remark 5.23) their 1-categorical counterpart with respect to weak homotopy equivalencs and quasi-isomorphisms, respectively. Note, that we do *not* employ the common practice of using the nerve functor implicitly.

Acknowledgments

I would like to thank everyone, who contributed to the success of this thesis. The people who should feel addressed are so numerous, that I am not capable of listing all of them here. Nevertheless, I want to seize this opportunity to express my gratitude to some of them explicitly.

Throughout my whole journey at the university, I received the support of Clara Löh. It is quite fitting that her presence marks both, the beginning of my undergraduate studies, as well as the ending of my doctoral studies under her supervision. I am very grateful for all that she has done for me, for her never-ending belief in my work and my ability to finish this thesis, and for everything I learned from her.

For answering tons of questions, I am grateful to Markus Land, Hoang Kim Nguyen, George Raptis and, once again, Clara Löh. For further useful discussions and comments, I would like to thank Uli Bunke, Denis-Charles Cisinski, Georg Biedermann, and an anonymous referee. I also owe my gratitude to Denis-Charles for his textbook *Higher categories and homotopical algebra*, which was an invaluable reference.

Thanks to my former and current office colleagues Daniel Fauser and Matthias Uschold for the nice time in our office, and for tolerating my somewhat volatile presence in the latter. Thanks for always being supportive when I asked for a favor.

During my time as a doctoral student, I repeatedly had pleasant chats, mathematical and non-mathematical, with the following people: Luigi Caputi, Daniel Fauser, Jonathan Glöckle, Michał Marcinkowski, Johanna Meumertzheim, Marco Moraschini, José Pedro Quintanilha, Julian Seipel, Enrico Toffoli, and certainly others – I apologize, in case I failed to list someone.

I am also thankful to a number of people whose acquaintanceship during my undergraduate studies steered me towards the idea of starting a PhD myself. This includes Michael Völkl, Peter Arndt, Alex Engel, Matthias Nagel, Klaus Kröncke, Michael Gößwein, and Matthias Blank.

I further acknowledge the support that I received from the *SFB 1085 Higher Invariants*.

After all these academically related thanks, it is about time that I express my deep gratitude towards my family and parents, for always being there. This thesis would not have been possible without them.

Similarly, my friends were indispensable to me: acting as my emotional backbone, they never ceased to encourage and console me when it was necessary. I am deeply indebted to them, and I want to sincerely thank all of them for their help.

Finally, I would like to thank a number of people, without whom this thesis would also not exist: everyone who took care of our children, be it regularly or sporadically. Foremost, I thank the employees at our daycare center for the great job they are doing. But also the help of several others was invaluable, including my mother, Barbara W., Judith T., Sophie S., Katharina and Thomas K., and Katrin K.

Contents

Introduction iii

Acknowledgments xi

1. Functorial semi-norms 1
- 1.1. The ℓ^1-semi-norm on singular homology . . . 1
- 1.2. Semi-norms on functors . . . 2
- 1.3. A general way to construct functorial semi-norms . . . 3
- 1.4. ℓ^1-Homology . . . 5

2. Universal finite functorial semi-norms 7
- 2.1. Carrying relation and universality . . . 7
- 2.2. Universality under equivalence of categories . . . 8
- 2.3. Vanishing loci . . . 10
- 2.4. Carrying a sequence of semi-norms . . . 11
- 2.5. Existence proofs . . . 13
- 2.6. Artificial counter-example . . . 15

3. Inflexible manifolds and differential graded algebras 17
- 3.1. Domination semi-norm . . . 17
- 3.2. Inflexible manifolds . . . 18
- 3.3. Differential graded algebras . . . 21
- 3.4. Rational homotopy theory . . . 26

4. Computational aspects of differential graded algebras and inflexible manifolds 29
- 4.1. Introduction to the software . . . 30
- 4.2. Examples . . . 34
- 4.3. Decidable ellipticity . . . 40
- 4.4. Decidable (in)flexibility . . . 42

5. A brief glimpse of infinity 45
- 5.1. From 1 to ∞ in a nutshell . . . 45
- 5.2. Working in ∞-categories . . . 49
- 5.3. How to produce ∞-categories . . . 53
- 5.4. Other names and other models . . . 55

6. Excisive approximation 57
- 6.1. Abstract excision . . . 57

6.2. Excisive approximation of the ℓ^1-chain complex functor 60
6.3. Generalization to n-excisive approximation 62

A. Excision for singular homology **65**
A.1. Model category theoretical bits 65
A.2. Proof of the model categorical statement 68
A.3. Proof of the ∞-categorical statement 70

B. Relation between abstract and classical excision **73**
B.1. Obtaining a fiber sequence 73
B.2. Applying the mapping space functor 74
B.3. The Mayer-Vietoris sequence 76
B.4. Example: singular homology 76

C. Software API documentation **79**

Bibliography **97**

Chapter 1:

Functorial semi-norms

In this chapter, we introduce the main players of this thesis. We start by reviewing the ℓ^1-norm on the singular chain complex and its induced semi-norm on singular homology (Section 1.1). This special example was not only Gromov's prime object of study when he introduced the notion of functorial semi-norms on singular homology [Grv1], but also reappears multiple times in this thesis. We then recall a generalization of Gromov's idea to what we now simply call *functorial semi-norms*, and discuss a general way of constructing such objects (Sections 1.2 and 1.3). In Section 1.4, we recall the definition of the ℓ^1-chain complex and ℓ^1-homology.

1.1. The ℓ^1-semi-norm on singular homology

One of the foundational objects in algebraic topology is the *singular chain complex* $C(X, A; R)$ of a pair of topological spaces (X, A) with R-coefficients for some ring with unit R. Taking its homology, or dualizing and taking cohomology, leads to the classical theory of singular (co)homology of (pairs of) spaces. One way of refining the latter (in the case $R = \mathbb{R}$) is by considering

- the *ℓ^1-norm on $C_n(X; \mathbb{R})$*, namely

$$\left|\sum_{j=1}^{N} a_j \, \sigma_j\right|_1 := \sum_{j=1}^{N} |a_j| \quad \in \mathbb{R}_{\geq 0}$$

  for a reduced chain in $C_n(X)$ with $N \in \mathbb{N}$, singular simplices $\sigma_j \colon \Delta^n \to X$, and coefficients $a_j \in \mathbb{R}$.

- the induced norms(!) $|\bullet|_1$ on the quotient $C(X, A; \mathbb{R}) = C(X; \mathbb{R})/C(A; \mathbb{R})$, and

- the induced semi-norm $\|\bullet\|_1$ on homology, i.e.

$$\|\alpha\|_1 = \inf\big\{|c|_1 \;\big|\; c \in C_n(X, A; \mathbb{R}) \text{ is a cycle with } [c] = \alpha\big\}$$

  for a homology class $\alpha \in H_n(X, A; \mathbb{R})$.

Roughly, the ℓ^1-semi-norm measures how many singular simplices with real coefficients are necessary to realize a particular (relative) homology class. Applying this idea to fundamental classes of oriented manifolds yields the *simplicial volume*, a homotopy invariant introduced by Gromov with many connections to Riemannian geometry [Grv1]

[Lö5]. Other than vanishing results, the exact value of the ℓ^1-semi-norm is typically very hard to compute. One exception is the formula

$$\text{simplicial volume of } S_g = 4 \cdot (g-1)$$

for an oriented closed connected surface of genus $g \in \mathbb{N}_{\geq 2}$, which can be obtained as a special case of one of Gromov's results stating that the simplicial volume of an oriented closed connected *hyperbolic* manifold is proportional to its Riemannian volume [Grv1, Sec. 2.2][Thu, Thm. 6.2]. (However, the inequality "$\leq$" in the above formula can also be obtained by more elementary means, using finite coverings of S_g [Grv1, p. 9][BP, Prop. C.4.7].)

The ℓ^1-semi-norm satisfies a certain functoriality with respect to continuous maps of spaces [Grv1][Rat, Lem. 1 of §11.6]: If $f\colon X \to Y$ is continuous and $\alpha \in H_n(X;\mathbb{R})$, then

$$\|H_n(f)(\alpha)\|_1 \leq \|\alpha\|_1.$$

Other semi-norms on singular homology that satisfy this property are introduced in Gromov's book [Grv2, Sec. 5.G$_+$ and 5.H$_+$]. We introduce a further abstraction of this idea in the following section.

1.2. Semi-norms on functors

We use the following terminology:

Definition 1.1 (semi-norm)**.** Let K be a normed field (e.g. $\mathbb{Q}$ or $\mathbb{R}$ with the standard absolute value).

(i) A *semi-norm* on a K-vector space V is a function $|\bullet|\colon V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ that satisfies
   - $|0| = 0$, the
   - triangle-inequality, i.e., for all $x, y \in V$ we have $|x+y| \leq |x| + |y|$, and
   - homogeneity, i.e., for all $a \in K \setminus \{0\}$ and all $x \in V$ we have $|a \cdot x| = |a| \cdot |x|$

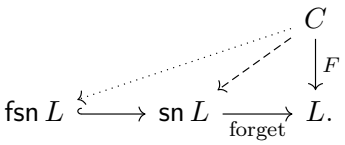
   (with the usual conventions regarding ∞). A semi-norm is *finite* if ∞ is not attained.

(ii) For a subcategory L of Vect_K, we denote the corresponding category of semi-normed K-vector spaces with norm non-increasing homomorphisms by $\mathsf{sn}\, L$; explicitly, the
   - objects of $\mathsf{sn}\, L$ are pairs $(V, |\bullet|)$ with $V \in \mathrm{Ob}(L)$ and $|\bullet|$ a semi-norm on V, and
   - morphisms $(V, |\bullet|) \to (W, |\bullet|')$ are those homomorphisms $f \in \mathrm{Mor}_L(V,W)$ that satisfy $|f(v)|' \leq |v|$ for all $v \in V$.

   By restricting to finite semi-norms, we obtain a full subcategory $\mathsf{fsn}\, L$ of $\mathsf{sn}\, L$.

Now we can say what a functorial semi-norm on a given functor is:

Definition 1.2 ((finite) functorial semi-norm)**.** Let C be a category, let L be a subcategory of Vect_K for a normed field K, and let $F\colon C \to L$ be a functor. A *functorial semi-norm on* F is a factorization of F over the forgetful functor $\mathsf{sn}\, L \to L$; it is *finite* if it further factors over the inclusion $\mathsf{fsn}\, L \hookrightarrow \mathsf{sn}\, L$:



Remark 1.3 ((finite) functorial semi-norm, explicitly)**.** In the situation of Definition 1.2, it is clear that a functorial semi-norm σ on F precisely consists of

- a choice of semi-norm $|\bullet|_X$ for all objects $X \in \mathrm{Ob}(C)$, such that
- for all morphisms $f\colon X \to Y$ of C and all $\alpha \in F(X)$, we have $|F(f)(\alpha)|_Y \leq |\alpha|_X$,

and that the functorial semi-norm is finite if and only if $|\bullet|_X$ is finite for all $X \in \mathrm{Ob}(C)$. We usually leave X implicit and write $|\bullet|_\sigma$ for all of the $|\bullet|_X$.

Example 1.4 (trivial functorial semi-norm)**.** Every functor as above admits the *trivial functorial semi-norm*, i.e., the semi-norm that vanishes on every input.

Example 1.5 (functoriality of the ℓ^1-semi-norm)**.** Let $n \in \mathbb{N}$. Then the ℓ^1-semi-norm (Section 1.1) is a finite functorial semi-norm on $H_n(\bullet;\mathbb{R})\colon \mathsf{Top} \to \mathsf{Vect}_{\mathbb{R}}$ and on $H_n(\bullet,\bullet;\mathbb{R})\colon \mathsf{Top}^2 \to \mathsf{Vect}_{\mathbb{R}}$.

1.3. A general way to construct functorial semi-norms

In the following, we describe a very general way of generating functorial semi-norms on a given functor, which generalizes a construction of Crowley and Löh [CL, Sec. 4]. We consider:

Setup 1.6. Let C be a category, let L be a subcategory of Vect_K for a normed field K, and let $F\colon C \to L$ be a functor.

Definition 1.7 (F-element)**.** In the situation of Setup 1.6, an *F-element* is a pair (X,α) where $X \in \mathrm{Ob}(C)$ and $\alpha \in F(X)$. We often suppress X in the notation and simply say that *α is an F-element*.

Definition 1.8 (generated semi-norm)**.** In the situation of Setup 1.6, let S be a class of F-elements and let $v\colon S \to \mathbb{R}_{\geq 0} \cup \{\infty\}$.

– An *S-representation* of an F-element (X, α) is a representation of the form

$$\alpha = \sum_{j=1}^{N} b_j \cdot F(f_j)(\alpha_j)$$

with $N \in \mathbb{N}$, coefficients $b_1, \dots, b_N \in K$, F-elements $(X_1, \alpha_1), \dots, (X_N, \alpha_N) \in S$, and morphisms $f_1 \colon X_1 \to X, \dots, f_N \colon X_N \to X$ in C.

– The *semi-norm $|\bullet|_v$ on F generated by v* is defined by: For all F-elements (X, α), we set

$$|\alpha|_v \coloneqq \inf\Big\{ \sum_{j=1}^{N} |b_j| \cdot v(X_j, \alpha_j) \;\Big|\; \sum_{j=1}^{N} b_j \cdot F(f_j)(\alpha_j) \text{ is an } S\text{-representation of } \alpha \Big\},$$

with $\inf \emptyset := \infty$.

Proposition 1.9 (generating functorial semi-norms via functions)**.** In the situation of Definition 1.8, we have:

(i) The semi-norm $|\bullet|_v$ generated by v is a functorial semi-norm on F.

(ii) For all F-elements α in S, we have $|\alpha|_v \leq v(\alpha)$.

(iii) If $v' \colon S \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ is a function with $v' \geq v$ (pointwise), then $|\alpha|_{v'} \geq |\alpha|_v$ for all F-elements α.

(iv) If S contains all F-elements given by a skeleton of C and v does not attain ∞, then $|\bullet|_v$ is finite.

(v) Let σ be a functorial semi-norm on F and let $v \geq |\bullet|_\sigma$ on S. Then, for all F-elements α, we have $|\alpha|_v \geq |\alpha|_\sigma$.

Proof. Using functoriality of F, it is easy to see that $|\bullet|_v$ is a functorial semi-norm. Also (iii) follows immediately from the definition. For an F-class (X, α), the identity morphism $X \to X$ shows (ii). Property (iv) is a direct consequence of (ii) and the fact that a *functorial* semi-norm is uniquely determined by its restriction to a skeleton. We now prove (v): Let $\sum_{j=1}^{N} b_j \cdot F(f_j)(\alpha_j) = \alpha$ be an S-representation of α. Then

$$|\alpha|_\sigma \leq \sum_{j=1}^{N} |b_j| \cdot |F(f_j)(\alpha_j)|_\sigma \leq \sum_{j=1}^{N} |b_j| \cdot |\alpha_j|_\sigma \leq \sum_{j=1}^{N} |b_j| \cdot v(\alpha_j)$$

by the triangle inequality, functoriality of σ, and assumption on v. Taking the infimum over all S-representations of α, we obtain $|\alpha|_v \geq |\alpha|_\sigma$.

■

Remark 1.10 (finiteness of generated semi-norms)**.** Proposition 1.9 (iv) only provides a sufficient criterion for $|\bullet|_v$ to be finite. For example, let $d \in \mathbb{N}$ and let us consider the case $F = H_d(\bullet;\mathbb{R})\colon \mathsf{Top} \to \mathsf{Vect}_{\mathbb{R}}$. Then, $|\bullet|_v$ is finite whenever S contains and v is finite on enough fundamental classes of manifolds, because rational homology classes can (up to multiplicity) be realized as the push-forward of fundamental classes by a classical result by Thom [Tho] [CL, Corollary 3.2]. Notably, it is already enough to take one of appropriate oriented closed connected (aspherical) d-manifolds, so-called *URC-manifolds* [Ga, p. 1747], together with all its finite coverings [CL, Example 7.10].

1.4. ℓ^1-Homology

Let (X, A) be a pair of topological spaces. Clearly, the boundary operators of the singular chain complex $C(X, A;\mathbb{R})$ are bounded operators with respect to the ℓ^1-norm (Section 1.1) in the sense of functional analysis; and the same is true in each degree for the induced chain map of a continuous map of pairs. This simple observation enables us to make the following:

Definition 1.11 (ℓ^1-chain complex)**.** The *ℓ^1-chain complex of* (X, A) is the degreewise completion of the singular chain complex $C(X, A;\mathbb{R})$ with respect to the ℓ^1-norm, together with the induced boundary operators:

$$C^{\ell^1}(X, A) \coloneqq \overline{C(X, A;\mathbb{R})}^{\ell^1}.$$

For a continuous map of pairs of spaces $f\colon (X, A) \to (Y, B)$, let

$$C^{\ell^1}(f)\colon C^{\ell^1}(X, A) \to C^{\ell^1}(Y, B)$$

denote the degreewise extension of $C(f;\mathbb{R})\colon C(X, A;\mathbb{R}) \to C(Y, B;\mathbb{R})$. ⌟

This functorial constructions turns the *normed chain complex* $C(X, A;\mathbb{R})$ into the *Banach chain complex* $C^{\ell^1}(X, A)$. More systematic definitions of these notions, along with their counterparts for cochain complexes and the duality between those can be found in Löh's doctoral thesis [Lö1, Ch. 1]. This new chain complex gives rise to:

Definition 1.12 (ℓ^1-homology)**.** The homology of $C^{\ell^1}(X, A)$ is the *ℓ^1-homology of* (X, A):

$$H^{\ell^1}(X, A) \coloneqq H_*\big(C^{\ell^1}(X, A)\big).$$

Note, that we apply the usual algebraic homology functor here, i.e. "kernel/image" (*without* taking the closure of the image).

In Chapter 6, we investigate the excisive approximation of the ℓ^1-chain complex functor on pointed spaces $\mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$, i.e. the composition of the functor C^{ℓ^1} from above with the inclusion of Top_* into the category of pairs of topological spaces.

Chapter 2:

Universal finite functorial semi-norms

Informally, a functorial semi-norm on a functor is called universal if it vanishes on as few classes as possible among all functorial semi-norms on the same functor; see Definition 2.5 for the precise statement. In this chapter, we prove the following existence results for universal functorial semi-norms (Section 2.5):

Theorem 2.1. Let C be a category that admits a skeleton with at most countably many objects. Let K be a normed field and let $F\colon C \to \mathsf{Vect}_K$ be a functor.

(i) If K is countable and if F maps to Vect_K^{ω}, then F admits a universal finite functorial semi-norm.

(ii) If F maps to $\mathsf{Vect}_K^{\mathrm{fin}}$, then F admits a universal finite functorial semi-norm.

Here, $\mathsf{Vect}_K^{\mathrm{fin}}$ and Vect_K^{ω} denote the categories of K-vector spaces of finite and countable dimension, respectively.

From the second part of Theorem 2.1 the following consequence for singular homology can be derived:

Corollary 2.2. Let $d \in \mathbb{N}$ and let K be a normed field. Then the singular homology functor $H_d(\bullet; K)\colon \mathsf{CW}^{\mathrm{fin}} \to \mathsf{Vect}_K$ admits a universal functorial semi-norm, where $\mathsf{CW}^{\mathrm{fin}}$ denotes the full subcategory of Top on those spaces that are homotopy equivalent to a finite CW-complex.

This chapter is based on joint work with Clara Löh [LW].

2.1. Carrying relation and universality

Setup 2.3. Let C be a category, let K be a normed field, and let $F\colon C \to \mathsf{Vect}_K$ be a functor.

Definition 2.4 (carries)**.** In the situation of Setup 2.3, let σ and τ be functorial semi-norms on F. Then σ *carries* τ if for all F-elements α, we have

$$|\alpha|_\sigma = 0 \implies |\alpha|_\tau = 0.$$

Definition 2.5 (universal finite functorial semi-norm)**.** In the situation of Setup 2.3, a *universal finite functorial semi-norm on* F is a finite functorial semi-norm on F that carries all other finite functorial semi-norms on F.

Remark 2.6. Definition 2.5 is not interesting for the non-finite case, because the functorial semi-norm that is ∞ everywhere, except at 0, is always universal.

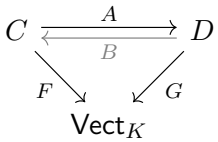
Example 2.7 (non-universality of the ℓ^1-semi-norm)**.** For each $d \in \{3\} \cup \mathbb{N}_{\geq 5}$ there exists a finite functorial semi-norm on $H_d(\bullet;\mathbb{R})$ that is *not* carried by the ℓ^1-semi-norm [FL, Theorem 1.2]. However, all finite functorial semi-norms that are multiplicative under finite coverings *are* carried by the ℓ^1-semi-norm [CL, Proposition 7.11].

Example 2.8 (representable and countably additive functors)**.** In the situation of Setup 2.3, if $K \in \{\mathbb{Q}, \mathbb{R}\}$ and if the functor F is representable or countably additive, then the trivial functorial semi-norm on F is universal [Lö3, Corollaries 4.1 and 4.5].

2.2. Universality under equivalence of categories

Universal finite functorial semi-norms are compatible with equivalences of categories (Corollary 2.11). Indeed, a stronger result holds: In Proposition 2.10, we show that universal functorial semi-norms can be transferred along "weak retractions" of categories.

Setup 2.9. Let C and D be categories, let K be a normed field and let $F\colon C \to \mathsf{Vect}_K$ and $G\colon D \to \mathsf{Vect}_K$ be functors. Let $A\colon C \to D$ be a functor such that $G \circ A$ is naturally isomorphic to F.



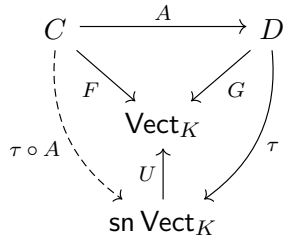
Proposition 2.10. In the situation of Setup 2.9, let $B\colon D \to C$ be a right-inverse of A, i.e., we assume that $A \circ B$ is naturally isomorphic to the identity on D. Then, if F admits a universal functorial semi-norm, so does G.

As an immediate consequence, we obtain:

Corollary 2.11. In the situation of Setup 2.9, assume that $A\colon C \to D$ is an equivalence of categories. Then F admits a universal finite functorial semi-norm if and only if G does.

Before we give the proof of Proposition 2.10, we make a few remarks about the interplay between functorial semi-norms and natural isomorphisms:

Remark 2.12 (non-strict functorial semi-norms)**.** In the situation of Setup 2.9 and given a functorial semi-norm τ on G, one would like to precompose τ with A to get a functorial semi-norm on F. However, as $G \circ A$ is not necessarily *equal* to F, also $\tau \circ A$ will not necessarily be a strict lift of F, but only up to natural isomorphism. In other words: if $U\colon \mathsf{sn}\,\mathsf{Vect}_K \to \mathsf{Vect}_K$ denotes the forgetful functor, the right triangle in the diagram



commutes on the nose while the other two only commute up to natural isomorphism.

One possible way to proceed would be to relax the definition of functorial semi-norm: Instead of $U \circ \tau = G$ we only require $U \circ \tau \cong G$, and then the functorial semi-norm consists of τ together with such a natural isomorphism.

This sounds like the correct setting to pursue the categorical view on functorial semi-norms (or formalization in a proof assistant [Lö4, Ch. 4.1.2]). On the other hand, this setting does not actually increase the pool of functorial semi-norms: Indeed, if $\eta\colon G \Rightarrow U \circ \tau$ is a natural isomorphism, the technique from Remark 2.13 shows how to construct a (strict) functorial semi-norm on G "with the same semi-norms".

Remark 2.13 (pull-back along natural transformation)**.** Let C be a category, let K be a normed field, let $\eta\colon F \Rightarrow F'$ be a natural transformation of functors $C \to \mathsf{Vect}_K$, and let σ be a functorial semi-norm on F'. Then, by naturality of η,

$$C \to \mathsf{sn}\,\mathsf{Vect}_K, \quad \begin{cases} X \mapsto \big(F(X), (\eta_X)^* |\bullet|_\sigma\big) & \text{on objects} \\ f \mapsto F(f) & \text{on morphisms} \end{cases}$$

defines a functorial semi-norm $\eta^*\sigma$ on F.

Proof of Proposition 2.10. First, we fix some notation: Let σ be a universal finite functorial semi-norm on F. Let $\lambda\colon \mathrm{Id}_D \Rightarrow A \circ B$ and $\psi\colon F \Rightarrow G \circ A$ be natural isomorphisms. Then $\phi \coloneqq \psi^{-1} \circ G(\lambda)$ is a natural isomorphism $G \Rightarrow F \circ B$. We consider the induced functorial semi-norm $\tilde{\sigma} \coloneqq \phi^*(\sigma \circ B)$ on G (Remark 2.13).

We show that $\tilde{\sigma}$ is universal for G: Let τ be a finite functorial semi-norm on G. The idea is straightforward: We go to the side of F, compare the result with the universal σ on F, and then derive universality of $\tilde{\sigma}$ on G. However, this involves a round-trip from D over C back to D, and thus we have to take λ into account. More precisely, we proceed as follows:

(i) Identifying the goal: Let $(Y, \tilde{\beta})$ be a G-element with $|\tilde{\beta}|_{\tilde{\sigma}} = 0$. We need to show that we also have $|\tilde{\beta}|_\tau = 0$.

(ii) Twisting τ to prepare for the round-trip: We factor in λ by considering the finite functorial semi-norm $\tau_\lambda \coloneqq G(\lambda)^*(\tau \circ A \circ B)$ on G.

(iii) Going to C: We then use the finite functorial semi-norm $\tilde{\tau} \coloneqq \psi^*(\tau_\lambda \circ A)$ on F.

Let $\beta \coloneqq \phi_Y(\tilde{\beta}) \in F(B(Y))$ be the element corresponding to $\tilde{\beta}$. By construction, we have

$$|\beta|_\sigma = \big|\phi_Y(\tilde{\beta})\big|_\sigma = \big|\phi_Y(\tilde{\beta})\big|_{\sigma \circ B} = |\tilde{\beta}|_{\phi^*(\sigma \circ B)} = |\tilde{\beta}|_{\tilde{\sigma}} = 0;$$

in the second step, we reinterpreted $\phi_Y(\tilde{\beta})$ as element of $F \circ B(Y)$, so that instead of σ on $B(Y)$ we can equivalently apply $\sigma \circ B$ on Y.

From universality of σ, we hence obtain $|\beta|_{\tilde{\tau}} = 0$.

(iv) Translating the result back to D: To keep the notation light, we will not explicitly annotate the objects to which the natural transformations are applied. We compute

$$\begin{aligned} 0 = |\beta|_{\tilde{\tau}} = |\beta|_{\psi^*(\tau_\lambda \circ A)} = \big|\psi(\beta)\big|_{\tau_\lambda \circ A} = \big|\psi(\beta)\big|_{\tau_\lambda} = \big|\psi(\beta)\big|_{G(\lambda)^*(\tau \circ A \circ B)} \\ = \big|G(\lambda)(\psi(\beta))\big|_{\tau \circ A \circ B} = \big|G(\lambda)(\psi(\beta))\big|_\tau. \end{aligned}$$

For every object Z of D, the map $G(\lambda_Z)$ is an isometry with respect to $|\bullet|_\tau$ because λ_Z is an isomorphism in D and τ is a functorial semi-norm on G. Therefore, we can continue with

$$\begin{aligned} \big|G(\lambda)(\psi(\beta))\big|_\tau = \big|\psi(\beta)\big|_\tau = \big|\psi(\phi(\tilde{\beta}))\big|_\tau = \big|\psi \circ \psi^{-1} \circ G(\lambda)(\tilde{\beta})\big|_\tau = \big|G(\lambda)(\tilde{\beta})\big|_\tau \\ = |\tilde{\beta}|_\tau. \end{aligned}$$

We conclude that $|\tilde{\beta}|_\tau = 0$, as claimed.

■

2.3. Vanishing loci

In this section, we reformulate the "carries" relation (Definition 2.4) in terms of vanishing loci (Definition 2.15, Remark 2.16).

The vanishing loci provide a convenient language to reason about families of functorial semi-norms and their relations: In Section 2.4, we use a diagonalization construction on the associated functions to construct a functorial semi-norm that carries countably many given functorial semi-norms (Proposition 2.17 and Corollary 2.18).

Setup 2.14. Let C be a category, let K be a normed field, let $F\colon C \to \mathsf{Vect}_K$ be a functor, and let S be a class of F-elements.

Definition 2.15 (vanishing locus)**.** We assume Setup 2.14; let $X \in \mathrm{Ob}(C)$.

– For a functorial semi-norm σ on F, we define the *vanishing locus* of σ on X by

$$N_\sigma(X) \coloneqq \big\{\alpha \in F(X) \;\big|\; |\alpha|_\sigma = 0\big\}.$$

– If C is small, we write $\mathsf{Fsn}(F)$ for the class of all finite functorial semi-norms on F and set

$$N(X) \coloneqq \bigcap_{\sigma \in \mathsf{Fsn}(F)} N_\sigma(X).$$

– For a function $v\colon S \to \mathbb{R}_{\geq 0}$, we write $N_v(X)$ for $N_{|\bullet|_v}(X)$, where $|\bullet|_v$ is the functorial semi-norm generated by v (Proposition 1.9).

In the situation of the definition, $N_\sigma(X)$ and $N(X)$ are K-subspaces of $F(X)$ and $N(X) \subseteq N_\sigma(X)$. Furthermore, if we regard $\mathsf{Fsn}(F)$ as the preorder category with respect to the "carries" relation, an initial object of this category is precisely a universal finite functorial semi-norm on F, while the trivial functorial semi-norm is always a terminal one.

Remark 2.16. Given Setup 2.14, let σ and τ be functorial semi-norms on F.

(i) Then σ carries τ if and only if

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_\sigma(X) \subseteq N_\tau(X).$$

(ii) If C is small, σ is universal on F if and only if it is finite and fulfills

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_\sigma(X) \subseteq N(X).$$

(iii) By Proposition 1.9 (v), the functorial semi-norm generated by $S \to \mathbb{R}_{\geq 0}$, $\alpha \mapsto |\alpha|_\sigma$ carries σ, i.e.,

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_{\alpha \mapsto |\alpha|_\sigma}(X) \subseteq N_\sigma(X).$$

2.4. Carrying a sequence of semi-norms

The main ingredient for the proof of Theorem 2.1 is that we can carry sequences of finite functorial semi-norms generated on a countable class of elements:

Proposition 2.17. In the situation of Setup 2.14, let S be countable and let $(v_n)_{n\in\mathbb{N}}$ be a sequence of functions $S \to \mathbb{R}_{\geq 0}$. Then there exists a function $v\colon S \to \mathbb{R}_{\geq 0}$ such that $|\bullet|_v$ carries all $(|\bullet|_{v_n})_{n\in\mathbb{N}}$, i.e., with

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_v(X) \subseteq \bigcap_{n\in\mathbb{N}} N_{v_n}(X).$$

In particular: If C is small, if every F-element admits an S-representation, and if

$$\forall\, X \in \mathrm{Ob}(C)\colon\ \bigcap_{n\in\mathbb{N}} N_{v_n}(X) \subseteq N(X),$$

then $|\bullet|_v$ is universal for F.

Proof. The second part follows from the first part and the characterization of universality from Remark 2.16 (ii).

We now prove the first part. As indicated by Proposition 1.9 (iii), we would like to set "$v \coloneqq \sup_n v_n$", but of course this might not produce a *finite valued* function. So instead, we choose

$$v\colon S \to \mathbb{R}_{\geq 0},\ \alpha_n \mapsto \max\bigl\{v_j(\alpha_n) \bigm| j \in \{-1,\dots,n\}\bigr\},$$

where we fix and implicitly use an enumeration $(X_n, \alpha_n)_{n\in\mathbb{N}}$ of S and where $v_{-1} \coloneqq 1$.

In order to show that v has the claimed property, we let $m \in \mathbb{N}$ and show that $|\bullet|_v$ carries $|\bullet|_{v_m}$: We introduce the following constants: Let $q_{-1} \coloneqq 1$, let

$$q_k \coloneqq \begin{cases} v(\alpha_k)\cdot |\alpha_k|_{v_m}^{-1} & \text{if } |\alpha_k|_{v_m} > 0, \\ 1 & \text{if } |\alpha_k|_{v_m} = 0 \end{cases}$$

for all $k \in \{0,\dots,m\}$, and let

$$Q \coloneqq \min\bigl\{q_k \bigm| k \in \{-1,\dots,m\}\bigr\}.$$

By construction, we have that $Q \in (0,1]$. For every F-element α and every S-representation $\alpha = \sum_{j=1}^N b_j \cdot F(f_j)(\alpha_{k_j})$, we can estimate

$$\begin{aligned}
\sum_{j=1}^N |b_j|\cdot v(\alpha_{k_j}) &\geq \sum_{\substack{j\in\{1,\dots,N\}\\ k_j<m}} |b_j|\cdot q_{k_j}\cdot |\alpha_{k_j}|_{v_m} + \sum_{\substack{j\in\{1,\dots,N\}\\ k_j\geq m}} |b_j|\cdot v_m(\alpha_{k_j}) && \text{(definition of } q_{k_j} \text{ and } v) \\
&\geq Q\cdot \sum_{\substack{j\in\{1,\dots,N\}\\ k_j<m}} |b_j|\cdot |\alpha_{k_j}|_{v_m} + \sum_{\substack{j\in\{1,\dots,N\}\\ k_j\geq m}} |b_j|\cdot |\alpha_{k_j}|_{v_m} && \text{(def. of } Q \text{ and P. 1.9 (ii))} \\
&\geq Q\cdot \sum_{j=1}^N |b_j|\cdot |\alpha_{k_j}|_{v_m} && \text{(because } Q\leq 1) \\
&\geq Q\cdot |\alpha|_{v_m},
\end{aligned}$$

where the last step follows from applying $|\bullet|_{v_m}$ to the given S-representation of α. By taking the infimum over all such S-representations, we obtain $|\alpha|_v \geq Q\cdot|\alpha|_{v_m}$. As $Q > 0$, we see that $|\bullet|_v$ carries $|\bullet|_{v_m}$ as desired.

■

Corollary 2.18. In the situation of Setup 2.14, let C be small, let S be countable, and let $T \subseteq \mathsf{Fsn}(F)$ be countable. Then there exists a functorial semi-norm σ on F such that σ carries all of T, i.e., with

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_\sigma(X) \subseteq \bigcap_{\tau\in T} N_\tau(X).$$

In particular: If every F-element admits an S-representation and if

$$\forall\, X \in \mathrm{Ob}(C)\colon \bigcap_{\tau \in T} N_\tau(X) \subseteq N(X),$$

then σ is universal for F.

Proof. Again, the second part follows from the first one and Remark 2.16 (ii).

We prove the first part of the claim: By Remark 2.16 (iii), for each $\tau \in T$, we find a function $v_\tau \colon S \to \mathbb{R}_{\geq 0}$ with

$$\forall\, X \in \mathrm{Ob}(C)\colon\ N_{v_\tau}(X) \subseteq N_\tau(X).$$

We then choose an enumeration of $\{v_\tau \mid \tau \in T\}$ and apply Proposition 2.17.

■

2.5. Existence proofs

In this section, we prove Theorem 2.1. We first treat the case of countable fields where a direct enumeration argument applies. Then we consider functors with range in finite dimensional vector spaces over general normed fields.

In both cases, we use the following observation:

Remark 2.19. By definition, the inclusion functor of a skeleton into the ambient category is an equivalence. Invoking Corollary 2.11, we may equivalently assume that the category itself has only countably many objects.

Proof of Theorem 2.1 (i). We may assume that C itself has only countably many objects (Remark 2.19). Furthermore, by assumption, K and $\dim_K F(X)$ are countable for all objects X of C. Together, we obtain that the class S of *all* F-elements is a countable set. Trivially, all F-elements admit an S-representation.

Let $S' \coloneqq \{(X, \alpha) \in S \mid \alpha \notin N(X)\}$ and for each $(X, \alpha) \in S'$ let σ_α be a finite functorial semi-norm on F with $\alpha \notin N_{\sigma_\alpha}(X)$.

By construction, for every object Y of C, we have

$$F(Y) \setminus N(Y) \subseteq \bigcup_{(X,\alpha) \in S'} F(Y) \setminus N_{\sigma_\alpha}(Y).$$

Hence, by De Morgan's laws and Corollary 2.18, there exists a universal functorial semi-norm on F.

■

Remark 2.20. In general, it would *not* be enough to have a countable set S with the property that every F-element admits an S-representation. Without the countability assumption on $\mathrm{Ob}(C)$, it might not be possible to control the vanishing locus on all

objects by only countably many functorial semi-norms, and thus, the second part of Corollary 2.18 does not apply. A concrete example is given in Section 2.6.

As a preparation for the proof of Theorem 2.1 (ii), we show that we can achieve universality on a single object:

Lemma 2.21. Let C be a small category, let K be a normed field, and let $F\colon C \to \mathsf{Vect}_K$ be a functor. Let $X \in \mathrm{Ob}(C)$ with $\dim_K F(X) < \infty$. Then there exists a finite functorial semi-norm σ on F with $N_\sigma(X) = N(X)$.

Proof. We proceed inductively, using the following observation:

$$\begin{gathered}\text{If } \sigma \in \mathsf{Fsn}(F) \text{ with } N_\sigma(X) \neq N(X), \text{ then there} \\ \text{exists a } \sigma' \in \mathsf{Fsn}(F) \text{ with } \dim_K N_{\sigma'}(X) < \dim_K N_\sigma(X).\end{gathered}$$

Indeed, if $N_\sigma(X) \neq N(X)$, there exists an $\alpha \in N_\sigma(X) \setminus N(X)$. Hence, there is a finite functorial semi-norm τ on F with $|\alpha|_\tau \neq 0$. Then also $\sigma' \coloneqq \sigma + \tau \in \mathsf{Fsn}(F)$ and α witnesses that

$$N_{\sigma'}(X) \subseteq N_\sigma(X) \cap N_\tau(X) \subsetneq N_\sigma(X).$$

Because of $\dim_K N_\sigma(X) \leq \dim_K F(X) < \infty$, we obtain $\dim_K N_{\sigma'}(X) < \dim_K N_\sigma(X)$.

For the actual induction, we start with the trivial functorial semi-norm $\sigma \coloneqq 0$ on F, which satisfies $N_\sigma(X) = F(X)$. We then iteratedly apply the observation above. Because $\dim_K F(X)$ is finite, this will terminate and lead to a finite functorial semi-norm σ on F with $N_\sigma(X) = N(X)$. ∎

Proof of Theorem 2.1 (ii). By Remark 2.19, we may assume without loss of generality, that $\mathrm{Ob}(C)$ is countable. For each $X \in \mathrm{Ob}(C)$, let $(\alpha_i)_{i \in I_X}$ be a finite generating set of the finite-dimensional K-vector space $F(X)$. Then $S \coloneqq \{(X, \alpha_i) \mid X \in \mathrm{Ob}(C),\ i \in I_X\}$ is countable and every F-element admits an S-representation.

By Lemma 2.21, for each $X \in \mathrm{Ob}(C)$, we find a functorial semi-norm σ_X on F with $N_{\sigma_X}(X) = N(X)$. Therefore, for all $Y \in \mathrm{Ob}(C)$, we have

$$\bigcap_{X \in \mathrm{Ob}(C)} N_{\sigma_X}(Y) \subseteq N(Y).$$

Applying Corollary 2.18 to the countable set $\{\sigma_X \mid X \in \mathrm{Ob}(C)\}$ thus shows that there exists a universal functorial semi-norm on F. ∎

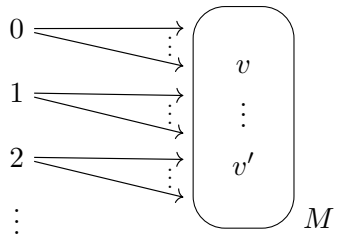
Proof of Corollary 2.2. The key observation is that functorial semi-norms are compatible with homotopy equivalences and thus the existence of a universal functorial semi-norm can be answered on the level of the homotopy category of $\mathsf{CW}^{\mathrm{fin}}$. The latter, however, admits a skeleton with countably many objects, to which Theorem 2.1 applies [LW, Sec. 5.2]. ∎

2.6. Artificial counter-example

If one drops the countability assumption on the skeleton that we imposed in Theorem 2.1, universal functorial semi-norms may fail to exist. In this section, we include an example of a functor that does *not* admit a universal functorial semi-norm.

Example 2.22 (a functor that does *not* admit a universal functorial semi-norm)**.** We define a category C by:

- We set $M \coloneqq (\mathbb{R}_{\geq 1})^{\mathbb{N}}$ and $\mathrm{Ob}(C) \coloneqq \mathbb{N} \sqcup M$.
- The only non-identity morphisms in C are the morphisms $f_{m,v} \colon m \to v$ with $m \in \mathbb{N}$ and $v \in M$.



We define a functor $F \colon C \to \mathsf{Vect}_{\mathbb{Q}}^{\mathrm{fin}}$ as follows:

- For all objects $X \in \mathrm{Ob}(C)$, we set $F(X) \coloneqq \mathbb{Q}$.
- For $m \in \mathbb{N}$ and $v \in M$, we set

$$F(f_{m,v}) \coloneqq \lceil v(m) \rceil \cdot \mathrm{id}_{\mathbb{Q}} \,. \qquad \lrcorner$$

Generating a functorial semi-norm via an appropriate function (Proposition 1.9), it is then not difficult to derive a contradiction from the assumption that the functor F from Example 2.22 admits a universal functorial semi-norm [LW, Prop. 6.5].

However, it seems unclear whether the phenomenon of this example also applies to the homology functor on the category of all topological spaces.

Chapter 3:

Inflexible manifolds and differential graded algebras

There exists a tight interaction between so-called *(strongly) inflexible manifolds* and (finite) functorial semi-norms on singular homology [CL]. Here, we recapitulate this correspondence in Section 3.2 and introduce some preliminaries in Section 3.1.

In order to obtain exotic examples of functorial semi-norms on singular homology (Theorem 3.6), one needs simply-connected inflexible manifolds. The only known method for constructing such manifolds, passes through Sullivan's approach of rational homotopy theory [Su][FHT], whose algebraic side is based on the notion of *differential graded algebras*. As these algebras are also at the core of Chapter 4, we introduce these objects in a hands-on but self-contained manner in Section 3.3.

In the last section of this chapter, we point the reader to the literature for the passage from (inflexible) differential graded algebras to (inflexible) manifolds.

3.1. Domination semi-norm

Given two oriented closed connected manifolds M, N of the same dimension, N is said to *dominate* M if there is a continuous map $N \to M$ of non-zero mapping degree. Fixing M and taking into account the largest such degree, one obtains a way to measure how often a given manifold can be "wrapped around M". Making this idea rigorous results in the following construction by Crowley and Löh [CL, Def. 7.1 and 7.3]:

Definition 3.1 (domination semi-norm)**.** Let $d \in \mathbb{N}_{\geq 1}$ and let Mfd_d denote the class of all oriented closed connected manifolds of dimension d.

(i) For manifolds $M, N \in \mathrm{Mfd}_d$, let

$$\deg(N, M) \coloneqq \{\deg(f) \mid f\colon N \to M \text{ continuous}\} \subseteq \mathbb{Z},$$

where $\deg(f)$ denotes the mapping degree of a continuous map f.

(ii) Let $M \in \mathrm{Mfd}_d$. The *domination semi-norm* $|\bullet|_M$ *on singular homology in degree* d

(with respect to M) is the semi-norm generated by the function

$$\begin{aligned}\big\{(N,[N]_{\mathbb{R}}) \;\big|\; N \in \mathrm{Mfd}_d\big\} &\to \mathbb{R}_{\geq 0} \cup \{\infty\} \\ (N,[N]_{\mathbb{R}}) &\mapsto v_M(N) \coloneqq \sup\big\{|k| \;\big|\; k \in \deg(N,M)\big\}\end{aligned}$$

in the sense of Definition 1.8; here $\sup \emptyset = 0$, and $[N]_{\mathbb{R}}$ denotes the real fundamental class of the manifold N.

Proposition 3.2. Let $M \in \mathrm{Mfd}_d$. Then the domination semi-norm on $H_d(\bullet;\mathbb{R})$ with respect to M has the following properties:

(i) The domination semi-norm $|\bullet|_M$ is a functorial semi-norm in the sense of Definition 1.2.

(ii) For all $N \in \mathrm{Mfd}_d$, we have $\big|[N]_{\mathbb{R}}\big|_M = v_M(N)$.

Proof. The first part follows directly from Proposition 1.9 (i). The inequality $|\bullet|_M \leq v_M$ generally holds on Mfd_d (Proposition 1.9 (ii)), and the other inequality is a consequence of the multiplicativity of the mapping degree: the latter implies that for all continuous maps $f\colon N' \to N$ with $N', N \in \mathrm{Mfd}_d$, we have

$$|\deg(f)| \cdot v_M(N') \leq v_M(N).$$

From this, the desired inequality can easily be derived [CL, Thm. 4.2 (1)]. ∎

Now if we have information about the place of a d-manifold within the "dominated by" relation, the previous proposition enables us to construct a functorial semi-norm on singular homology in degree d with control over its values on fundamental classes of oriented closed connected manifolds. Via this technique, Crowley and Löh [CL] and also Fauser and Löh [FL] have constructed functorial semi-norms that exhibit different exotic behaviors. A key to those constructions are so-called *inflexible* manifolds, which we review in the following section.

3.2. Inflexible manifolds

In order to obtain interesting examples of functorial semi-norms by means of the domination semi-norm, we need manifolds that do not admit high-degree maps onto them. This notion is captured by the following:

Definition 3.3 ((strongly) inflexible manifold)**.** Let $d \in \mathbb{N}_{\geq 1}$ and let $M \in \mathrm{Mfd}_d$, i.e. M is an oriented closed connected manifold of dimension d. Then M is

– *inflexible* if $\deg(M,M)$ is finite,

– *strongly inflexible* if $\deg(N,M)$ is finite for all $N \in \mathrm{Mfd}_d$,

– *flexible* if it is not inflexible

– *weakly flexible* if it is not strongly inflexible.

Remark 3.4. The following observations are immediate from Definition 3.3 by multiplicativity of the mapping degree:

(i) We have $\deg(M,M) \subseteq \{-1,0,1\}$ if and only if M is inflexible.

(ii) If M is dominated by some weakly flexible manifold, it must be weakly flexible itself.

Examples 3.5. In the following examples, all manifolds are assumed to be oriented, closed, connected and of non-zero dimension.

(i) Spheres are flexible.

(ii) Any product manifold with a flexible factor is flexible.

(iii) In particular, tori are flexible.

(iv) By functoriality, any manifold of non-zero simplicial volume is strongly inflexible [CL, cf. Ex. 6.15]. For example, this includes all hyperbolic manifolds, and more generally, Riemannian manifolds with sectional curvature bounded from above by a negative constant [IY][Grv1, (B) in Sec. 1.2].

(v) Generalizing the previous item: If there exists a [finite] functorial semi-norm such that its value on the real fundamental class of M is finite and positive, the manifold M must be [strongly] inflexible.

Conversely to (v), a [strongly] inflexible d-manifold M yields a [finite] functorial semi-norm on $H_d(\bullet;\mathbb{R})$ by Proposition 3.2, such that $|[M]_{\mathbb{R}}|_M = 1$. The following results were shown in this way:

Theorem 3.6 (exotic functorial semi-norms [FL][CL])**.**

(i) There exist finite functorial semi-norms on singular homology that are *not* carried (Definition 2.4) by the ℓ^1-semi-norm [FL, Thm. 1.2].

(ii) There exist functorial semi-norms on singular homology that are positive and finite on certain homology classes of simply-connected spaces [CL, Thm. 1.2].

The first part clarifies the role of the ℓ^1-semi-norm among all finite functorial semi-norms on singular homology in the sense that not even non-vanishing of homology classes is determined by the ℓ^1-semi-norm. The second part is interesting because it answers the question raised by Gromov whether functorial semi-norms always vanish on simply-connected spaces [Grv2, Remark (b) in paragraph 5.35]. At the time, none of the functorial semi-norms constructed by Crowley and Löh were known to be finite [CL, Rem. 7.5],

but by now Costoya, Muñoz and Viruel showed that they are *not* finite [CMuV]. Thus, the following is still open:

Question 3.7 ([CL, Q. 1.1(2)])**.** Does every *finite* functorial semi-norm on singular homology (in degree greater than zero) vanish on simply-connected spaces?

The existence of a strongly inflexible, simply-connected manifold would answer this question in the negative, so it is certainly worthwhile to obtain a better understanding of the class of (strongly) inflexible manifolds. If it exists, such a manifold must be at least 7-dimensional, because Crowley and Löh proved that all finite functorial semi-norms on homology of degree 1 to 6 vanish on simply-connected spaces [CL, Thm. 1.3].

On the other hand, the existence of a finite functorial semi-norm on singular homology of degree at least 7 that does *not* vanish on some simply connected space (not necessarily manifold!) would also settle the existence of a strongly inflexible, simply-connected manifold by the following result:

Proposition 3.8 ([CL, Prop. 7.6])**.** For $d \in \mathbb{N}_{\geq 4}$, the existence of a finite functorial semi-norm on $H_d(\bullet;\mathbb{R})$ that does *not* vanish on all simply-connected spaces is equivalent to the existence of a strongly inflexible, simply-connected d-dimensional manifold.

In order to get closer to *strongly* inflexible, simply-connected manifolds, one can try to study and find more inflexible ones. We give an overview of known constructions:

Examples 3.9 (inflexible, simply-connected manifolds)**.** Historically, the first examples of inflexible, simply-connected closed manifolds were given by Arkowitz and Lupton [AL, Ex. 5.1 and 5.2], which were of dimensions 208 and 228. Since then,

- Crowley and Löh derived a "design pattern" and two new examples [CL, Sec. I.1] in dimensions 64 and 108;
- they also proved that iterated self-products can be used to obtain examples in all dimensions $d \cdot k$ with $d \in \{108, 208, 228\}$ and $k \in \mathbb{N}_{\geq 1}$ [CL, Cor. II.7].
- Amann found a countably infinite family of examples in *odd* dimensions $231 + 4 \cdot k$ with $k \in \mathbb{N}$ [Am, Sec. 3];
- and proved, also via products, that those give rise to examples in *all* dimensions at least 921 [Am, Cor. 3.5];
- he further twisted the above example of dimension 64 to obtain one in dimension 66 [Am, Ex. 3.8].
- Costoya and Viruel obtained an example manifold of dimension $415 + 160 \cdot k$ for each finite, connected graph on k vertices with $k \in \mathbb{N}_{\geq 2}$ [CV, Thm. 2.6, Lem. 3.2, and proof of Thm. 1.5 (p. 60)].

– Together with Méndez, they later gave two refinements of their approach, resulting in constructions of highly connected examples, which are, however, of even (much) higher dimension [CMV1, Sec. 4.1][CMV2, Sec. 3].

– The author of this thesis found a new example in dimension 38 (cf. Example 4.6), based on Crowley and Löh's design pattern and verified by computer calculations from the software that is presented in Chapter 4.

All of these known examples of inflexible, simple-connected manifolds have in common that they are constructed through rational homotopy theory. More precisely, this means that for all of the above examples, one actually constructs an inflexible differential graded algebra (Section 3.3), and then uses methods from rational homotopy theory to realize this algebra as the minimal Sullivan model of some manifold (Section 3.4), which, as a consequence, will be inflexible as well. While this approach has proven to be fruitful, it would certainly be interesting to have more different sources of examples:

Question 3.10. How can inflexible, simply-connected manifolds be constructed
– without passing through rational homotopy theory?
– in more geometric or direct ways?

3.3. Differential graded algebras

As an algebraic foundation, rational homotopy theory relies on a setting with gradings and differentials, which we introduce in this section. Building upon the notion of graded vector spaces, differential graded algebras of two different types are the main players in Quillen's [Qu] and Sullivan's [Su] work: The former employs a Lie bracket as additional structure, whereas an associative multiplication is utilized in the latter. As we will not need Lie type objects in this thesis, we will not introduce any details about them and focus on Sullivan algebras instead.

If one is familiar with the tensor product of chain complexes [Ei, Sec. 17.3] and monoid objects in monoidal categories [Mʟ2, Sec. VII.3], there is a very succinct definition of differential graded(-commutative) algebra, which we state in the following remark before giving a more hands-on definition.

Remark 3.11 (abstract definition of differential graded algebra)**.** Fix a commutative ring k. The category of *differential graded k-algebras* is the category of monoid objects in the (symmetric) monoidal category $(\mathsf{Ch}_k, \otimes, k)$ of chain complexes over k with respect to the tensor product. Analogously, the category of *differential graded-commutative k-algebras* is given by the category of commutative monoid objects.

While this categorical description yields a nice conceptual approach to the subject, it requires quite some familiarity with abstract category theory. Given the fact that we are interested in the algebraic framework from a computational perspective (Chapter 4) we will not have the opportunity to exploit such a high level point of view. A comprehensive

reference for the categorical features of the framework is an article by Anel and Joyal [AJ, Sec. 1.1 and 1.2].

In the rest of this section, we will introduce differential graded algebras in more elementary terms and the reader who is happy with Remark 3.11 might want to skip it. For a lot of the following material, generalizations in several directions exist, but we deliberately restrict the exposition to the necessary pieces that are used in this thesis. More comprehensive treatments can be found in the literature [ML1, Ch. VI][GM, Ch. 10] [FHT, Ch. 3].

For the rest of the section, fix a commutative ring k. In the applications of rational homotopy theory, k will mostly be a field of characteristic 0, specifically $k = \mathbb{Q}$.

Definition 3.12 (graded module)**.** (i) A *graded k-module* M is a family $(M_n)_{n\in\mathbb{Z}}$ of k-modules.

(ii) For a graded k-module M, an *element x of M* is an element $x \in M_n$ for some $n \in \mathbb{Z}$; in this case, $\deg x := n$ is the *degree of x*.

(iii) For graded k-modules M and N, a *graded k-homomorphism* $f\colon M \to N$ is a family of k-homomorphisms $(f_n\colon M_n \to N_{n+d})_{n\in\mathbb{Z}}$ for some $d \in \mathbb{Z}$; in this case, $\deg f := d$ is the *degree of f*. A *morphism* $M \to N$ is a graded k-homomorphism of degree 0. For an element x of M, we set $f(x) := f_{\deg x}(x)$.

(iv) Together with component-wise composition, graded k-modules and morphisms form a category $\mathsf{gMod}(k)$.

Example 3.13 (trivial grading)**.** Every k-module M_0 can be promoted to a graded k-module M by setting $M_n := 0$ for all $n \in \mathbb{Z}\setminus\{0\}$. In this case, we say that M is *trivially graded*.

Applying the same idea also to morphisms, we obtain a full and faithful functor $\mathsf{Mod}(k) \to \mathsf{gMod}(k)$. ⌟

Example 3.14 (graded module of graded homomorphisms)**.** Let M and N be graded k-modules. Then the graded k-homomorphisms $M \to N$ can be organized into a graded k-module itself: For $n \in \mathbb{Z}$, we let $\mathrm{Hom}(M,N)_n$ be given by the set of graded k-homomorphisms $M \to N$ of degree n, equipped with pointwise scalar multiplication and addition.

Remark 3.15 (degree-wise concepts)**.** We extend the notions of direct sums, submodules, and quotients in a degree-wise manner. For example, for graded k-modules A and B, we have $(A \oplus B)_n = A_n \oplus B_n$.

Furthermore, by a *free* graded k-module, we mean that it is free in every degree, and a *basis* consists of the (disjoint) union of respective bases. ⌟

In the sense of modules as trivially graded modules (Example 3.13), the tensor product of k-modules extends to graded k-modules:

Definition 3.16 (tensor product)**.** Let M and N be graded k-modules. Their *tensor product* $M \otimes N$ is defined by

$$(M \otimes N)_n \coloneqq \bigoplus_{p \in \mathbb{Z}} M_p \otimes N_{n-p}$$

for all $n \in \mathbb{Z}$. For graded k-homomorphisms $f\colon M \to M'$ and $g\colon N \to N'$, we define the graded k-homomorphism $f \otimes g\colon M \otimes N \to M' \otimes N'$ as follows: for $n \in \mathbb{Z}$, we set

$$\begin{aligned}(f \otimes g)_n\colon (M \otimes N)_n &\to (M' \otimes N')_{n+\deg f+\deg g} \\ x \otimes y &\mapsto (-1)^{\deg x \cdot \deg g} \cdot f(x) \otimes g(y)\end{aligned} \tag{3.1}$$

on elementary tensors of direct summands.

As per our convention in Definition 3.12 (iii), formula (3.1) is a concise equivalent of

$$M_p \otimes N_{n-p} \ni x \otimes y \mapsto (-1)^{p \cdot \deg g} \cdot f_p(x) \otimes g_{n-p}(y).$$

Remark 3.17 (signs)**.** In the graded setting, signs as in (3.1) are unavoidable. Our sign conventions follow the "Koszul sign rule": whenever two "objects" a, b with degrees need to pass each other, a sign

$$(-1)^{\deg a \cdot \deg b}$$

is introduced. For example, this explains the sign in (3.1): to evaluate $(f \otimes g)(x \otimes y)$, we have to move x past g.

Definition 3.18 (differential graded algebra)**.** A *differential graded k-algebra* consists of

- a graded k-module A,
- a morphism $m\colon A \otimes A \to A$, called *multiplication* and written $ab \coloneqq a \cdot b \coloneqq m(a \otimes b)$,
- an element $1 \in A_0$, called *unit*, and
- a graded k-homomorphism $\delta\colon A \to A$ of degree -1, called *differential*,

subject to the following conditions:

- associativity of multiplication: $\forall\, a, b, c \in A\colon\ a \cdot (b \cdot c) = (a \cdot b) \cdot c$,
- unitality: $\forall\, a \in A\colon\ 1 \cdot a = a = a \cdot 1$,
- complex: $\forall\, n \in \mathbb{Z}\colon\ \delta_n \circ \delta_{n+1} = 0$,
- graded Leibniz rule: $\forall\, a, b \in A\colon\ \delta(a \cdot b) = \delta(a) \cdot b + (-1)^{\deg a} \cdot a \cdot \delta(b)$.

It is *graded-commutative* if it additionally satisfies

$$\forall\, a, b \in A\colon\ a \cdot b = (-1)^{\deg a \cdot \deg b} \cdot b \cdot a.$$

We denote a differential graded k-algebra by (A, δ), leaving multiplication and unit implicit.

A morphism of differential graded k-algebras is a morphism of graded k-modules that is compatible with multiplication, preserves the unit, and commutes with taking differentials.

We denote the category of differential graded k-algebras by $\mathsf{dgAlg}(k)$ and its full subcategory of graded-commutative objects by $\mathsf{dgcAlg}(k)$.

Dropping differential and graded Leibniz rule, we obtain the notion of a *graded k-algebra A*. ⌟

Example 3.19 (tensor product of algebras)**.** Let A and B be graded k-algebras with units 1_A and 1_B, respectively. The tensor product of graded k-modules $A \otimes B$ becomes a graded k-algebra via the multiplication (given on elementary tensors by)

$$(a \otimes b) \cdot (a' \otimes b') \coloneqq (-1)^{\deg a' \cdot \deg b} \cdot (aa') \otimes (bb')$$

and the unit $1_A \otimes 1_B$. If, additionally, (A, δ_A) and (B, δ_B) are differential graded k-algebras, their tensor product $(A, \delta_A) \otimes (B, \delta_B)$ carries the differential

$$\delta_A \otimes \mathrm{id}_B + \mathrm{id}_A \otimes \delta_B,$$

which, by Definition 3.16, is explicitly given on an elementary tensor $a \otimes b$ by the formula $\delta_A(a) \otimes b + (-1)^{\deg a} \cdot a \otimes \delta_B(b)$.

Forgetting multiplication and unit of a differential graded algebra, we evidently obtain a ($\mathbb{Z}$-indexed) chain complex. Thus, it makes sense to consider its homology:

Definition 3.20 (homology of a differential graded algebra)**.** The *homology functor* $H \colon \mathsf{dgAlg}(k) \to \mathsf{gMod}(k)$ is the composition of the forgetful functor $\mathsf{dgAlg}(k) \to \mathsf{Ch}_k$ with the usual homology functor on chain complexes. Explicitly, for a differential graded k-algebra (A, δ), we have

$$H(A, \delta) = \big(\ker(\delta_n) / \operatorname{im}(\delta_{n+1})\big)_{n \in \mathbb{Z}}$$

and the k-module $H(A, \delta)_n$ is also denoted by $H_n(A, \delta)$.

Most often, one is not primarily interested in a differential graded k-algebra itself, but in its homology. In such a case, the following notion is of great interest:

Definition 3.21 (quasi-isomorphism)**.** A morphism of chain complexes over k or of differential graded k-algebras is a *quasi-isomorphism* if its image under the homology functor is an isomorphism.

We now introduce some general examples of graded algebras, which play an essential role in the theory of Sullivan algebras.

Definition 3.22 (tensor and symmetric algebra)**.** Let M be a graded k-module.

(i) The *tensor algebra of* M is the graded k-algebra

$$T(M) \coloneqq \bigoplus_{d\in\mathbb{N}} M^{\otimes d}$$

with multiplication given by "concatenation", i.e.

$$(x_1 \otimes \cdots \otimes x_r) \cdot (y_1 \otimes \cdots \otimes y_s) \coloneqq x_1 \otimes \cdots \otimes x_r \otimes y_1 \otimes \cdots \otimes y_s,$$

and with unit $1 \in k = M^{\otimes 0}$. We view M as a graded submodule of $T(M)$ via the direct summand $M^{\otimes 1}$.

(ii) The *symmetric algebra of* M is the quotient graded k-algebra

$$S(M) \coloneqq T(M)/I,$$

where the two-sided ideal $I \subseteq T(M)$ is generated by

$$\big\{xy + (-1)^{\deg x \cdot \deg y} yx \;\big|\; x, y \in M\big\}.$$

As the ideal contains no elements of $M \subseteq T(M)$, we can still view M as a graded submodule of $S(M)$.

The tensor and symmetric algebras satisfy universal mapping properties [Mʟ1, Prop. VI.3.1 and Exerc. VI.4.2], which readily imply that for M free with basis X, the tensor algebra of M is the free graded k-algebra on X and similarly, the symmetric algebra of M is the free graded-commutative k-algebra on X. (Though it will not be important to us in the following, it should be noted that caution is necessary in the case that 2 is *not* a unit in k. For example, Mac Lane distinguishes between "commutative" and "strictly commutative" [Mʟ1, Sec. VI.3, p. 178 f.].)

Despite the nomenclature, the symmetric algebra of a graded module provides a joint generalization of the notions of symmetric and exterior algebra from the non-graded setting. With this in mind, the following notation might seem a bit odd, but unfortunately it is the standard one used in rational homotopy theory, which we therefore adopt:

Proposition 3.23 (free graded-commutative algebra)**.** Assume that 2 is a unit in k and let M be a free graded k-module with basis $(x_1, \ldots, x_r)$ for some $r \in \mathbb{N}$. Then $\wedge(x_1, \ldots, x_r) \coloneqq \wedge(M) \coloneqq S(M)$ is called the *free graded-commutative algebra on* M and has the following properties:

(i) A basis for the underlying graded k-module of $\wedge(M)$ is given by all products of the form

$$x_1^{\ell_1} \cdots x_r^{\ell_r} \quad \text{with} \begin{cases} \ell_j \in \mathbb{N} & \text{if } \deg x_j \text{ is even and} \\ \ell_j \in \{0, 1\} & \text{otherwise,} \end{cases}$$

and in particular, for all $n \in \mathbb{Z}$, a basis of the k-module $\wedge(M)_n$ is given by all such products with $n = \sum_{j=1}^{r} \ell_j \cdot \deg x_j$.

(ii) Every differential δ, making $(\wedge(M), \delta)$ a differential graded algebra, is uniquely determined by its restriction to the morphism $\delta|_M \colon M \to \wedge(M)$ of graded k-modules.

Proof. The first part is Corollary A2.3 in Eisenbud's book [Ei] and the second part follows immediately from the graded Leibniz rule.

■

Definition 3.24 (semi-free)**.** A differential graded-commutative k-algebra (A, δ) is *semi-free*, if A is a free graded-commutative algebra as in Proposition 3.23.

The prefix "semi-" is needed, because it is only the underlying graded-commutative algebra that is free, not the object (A, δ) in $\mathsf{dgcAlg}(k)$.

Example 3.25 (polynomial differential forms)**.** Let $d \in \mathbb{N}$. Let $M_0 \coloneqq M_{-1} \coloneqq \mathbb{Q}^d$ and let $M_n \coloneqq 0$ for all $n \in \mathbb{Z} \setminus \{0, -1\}$. Furthermore, let $\delta \colon \wedge(M) \to \wedge(M)$ be the unique differential with $\delta_0 = \mathrm{id}_{\mathbb{Q}^d}$. The differential graded $\mathbb{Q}$-algebra $(\wedge(M), \delta)$ is (isomorphic to) the *algebra of polynomial differential forms*, which is the foundational object of Sullivan's theory [Su][FHT, §10 (c)].

3.4. Rational homotopy theory

A continuous map $f \colon X \to Y$ between topological spaces is a *rational (homology) equivalence* if

$$H_*(f; \mathbb{Q}) \colon H_*(X; \mathbb{Q}) \to H_*(Y; \mathbb{Q})$$

is an isomorphism of graded $\mathbb{Q}$-vector spaces. For simply-connected X and Y, this condition is equivalent to f being a rational *homotopy* equivalence, i.e. to

$$\pi_*(f) \otimes_{\mathbb{Z}} \mathbb{Q} \colon \pi_*(X) \otimes_{\mathbb{Z}} \mathbb{Q} \to \pi_*(Y) \otimes_{\mathbb{Z}} \mathbb{Q}$$

being an isomorphism of graded $\mathbb{Q}$-vector spaces [FHT, §9 (c)]. *Rational homotopy theory* is the study of topological spaces up to rational equivalences.

Sullivan's theory [Su] establishes a bijective correspondence between rational homotopy types and (isomorphism classes of) minimal Sullivan algebras over $\mathbb{Q}$, both under suitable finiteness conditions. We refer the reader to the textbook by Félix, Halperin and Thomas [FHT, Part II] for the general picture and to Section 6.1 of Crowley and Löh's article [CL] for a concise recap of the relevant notions in the context of inflexible manifolds. Furthermore, in Section 6.2 they explain precisely under what conditions the algebras can be realized by simply-connected manifolds, as to obtain the examples from Section 3.2.

Remark 3.26 (cohomological grading)**.** A certain change in perspective is customary in rational homotopy theory, mainly in order to avoid negative degrees: As the most

frequent graded algebras are non-positively graded, one tends to use "upper grading" and redefines the term "degree" accordingly.

More explicitly: Given a graded module M, one uses the notation $M^n := M_{-n}$ and says that elements of M^n have *(upper) degree* n. As this convention is most often used, when $M_n = 0$ for all $n \in \mathbb{Z}_{>0}$, there is only a small risk of confusion between the two meanings of degree. ⌟

Chapter 4:

Computational aspects of differential graded algebras and inflexible manifolds

One big advantage of Sullivan algebras is their manifest simple algebraic nature, which makes them very amenable to computations. Another one is, at least for finite type algebras, that they are easily described by a finite amount of data. Taken together, this makes it quite feasible to implement computer software handling such structures.

In order to understand inflexible manifolds better – and maybe at some point in the future even strongly inflexible ones – the author worked on such a computer program. One goal was to free the researcher from most of those tedious computations that have to be done whenever one tries new differential graded algebras. Another idea was the possibility of random search for inflexible differential graded algebras, which was unfortunately not implemented so far.

Basic features of the software are introduced in Section 4.1, more advanced functions are demonstrated by examples in Section 4.2, and Appendix C contains the full API documentation. A new example (Example 4.6) of an inflexible, minimal Sullivan algebra in dimension 38 is contained in Section 4.2.

In the course of implementing the software, we also obtained results about algorithmic decidability of some properties of minimal Sullivan algebras. We state and prove these in the second part of this chapter, which consists of Sections 4.3 and 4.4.

We make the following conventions for this chapter:

Setup 4.1. Where not otherwise noted, we work over a ground field k of characteristic 0. To simplify notation, we will mostly leave k implicit.

Definition 4.2 (cochain algebra)**.** A *cochain algebra* is a semi-free differential graded-commutative algebra (A, δ) with $A_n = 0$ for all $n \in \mathbb{Z}_{>0}$. It is *connected*, if additionally $A_0 \cong k$ holds. By *generators of* A we always refer to the elements $x_1, \ldots, x_r$ such that $A = \wedge(x_1, \ldots, x_r)$.

Note, that this terminology is non-standard insofar as semi-freeness is usually not assumed, and some authors also drop "-commutative".

Furthermore, we make use of cohomological grading (Remark 3.26), i.e. "degree" will refer to upper degrees in this chapter. In particular, differentials *raise* the degree by 1.

Alg Algebra basics
Alg.Classes Eq, Ord, Show instances for algebras
Alg.Cohomology Cohomological aspects
Alg.GenericMor Generic algebra morphisms
Cache Caching of recurring computations
Extern
Extern.Maple Rudimentary interface to Maple
Extern.Sage Interface to Sage functions
FracClear Abstract interface for clearing fractions
HFMext Extension of the HaskellForMaths package
Lens Lens for caches (and algebras)
Types Type definitions and synonyms
Util Generic utility functions

Figure 4.1.: Modules in the Haskell package `inflexible`, as displayed in the Haddock generated package documentation

4.1. Introduction to the software

We chose to implement the software in Haskell [Ma]: "An advanced, purely functional programming language"[1]. From the perspective of the mathematician, it has the huge advantage over imperative languages, that its functions behave like mathematical definitions: established objects are immutable; a function transforms input to output, which can then be given a new name, if necessary. Furthermore, it has builtin support for exact arithmetics. In particular, computations over $\mathbb{Q}$ in the following are *not* approximated by fixed size decimal numbers.

We make no attempt of explaining more details about Haskell or the concept of purely functional programming in this thesis. We refer the reader to the existing literature and ask the uninitiated reader for forgiveness. The official Haskell web site contains *lots* of resources[2].

Our Haskell package is called `inflexible` and consists of several modules, see Figure 4.1. The main modules are `Types`, `Alg`, `Alg.Cohomology`, and `Alg.GenericMor`, each of which contains documentation in form of source code annotations. Full, standalone API documentation can be generated by Haddock[3], and we include a compiled version of its LaTeX output in Appendix C. In this chapter, we will only present some distinguished fragments of the whole package. The subdirectory `interactive/` of the latter contains examples and some useful utilities for playing around in GHCi.

[1] https://www.haskell.org/
[2] https://www.haskell.org/documentation/
[3] https://hackage.haskell.org/package/haddock

How to obtain the package

The source code for `inflexible` can be obtained from the author's github repository

`https://github.com/J0J0/inflexible` (the username is: J-zero-J-zero)

or, in the unlikely event that this is not available anymore, upon email request to proof@fantasymail.de. The git commit hash of the package at time of publication of the present thesis is `9d641736f82bccdddafebdfcc5b4dba63772b1cd`.

Noteworthy third party libraries that we build on

The package `HaskellForMaths-excerpt`[4] serves as a basis for the implementation of cochain algebras, which is – as the name indicates – an excerpt from the package `HaskellForMaths`[5] by David Amos. The latter already comes with facilities for free modules over a ring, symmetric algebras, exterior algebras, and tensor product of algebras. It also contains support for multivariate polynomial rings and Gröbner bases with respect to the most commonly used monomial orderings.

However, as the implementation of Buchberger's algorithm for computing Gröbner bases in the `HaskellForMaths` package only performs well in simple examples, our module `Extern.Sage` taps into SageMath [Sm], in order to use its module for "Ideals in multivariate polynomial rings". The latter, in turn, largely delegates the work to the computer algebra system Singular [DGPS]. As SageMath is based on the Python programming language[6], we utilize a patched version of the Haskell package `cpython`[7] to bridge the gap.

The full list of Haskell packages that `inflexible` depends on, can be found under the "build-depends" field in the cabal file `inflexible.cabal`.

Representing cochain algebras

Usually, examples of cochain algebras are given as follows [FHT][AL][CL]:

- One specifies a family of "symbols" and respective degrees that are to be the generators of the algebra. In the context of Proposition 3.23, the symbols would denote a basis of the free module M, but most often, the latter is never mentioned explicitly.
- The action of the differential on these generators is given (which determines it uniquely, Proposition 3.23 (ii)), sometimes
- accompanied by a justification, why the differential has the complex property.

Remark 4.3. For instance, Example 3.25 would be specified as follows in this way: The algebra of polynomial differential forms is the cochain algebra $(\wedge(t_1, \ldots, t_d, y_1, \ldots, y_d), \delta)$ where the t_j and y_j have degrees 0 and 1, respectively, and the differential δ is defined by $\delta(t_j) \coloneqq y_j$ and $\delta(y_j) \coloneqq 0$.

[4] https://github.com/J0J0/HaskellForMaths-fork/tree/tensoralg-excerpt
[5] https://hackage.haskell.org/package/HaskellForMaths
[6] https://www.python.org
[7] https://hackage.haskell.org/package/cpython

```
-- | A 'DgaSpec' k a contains the specification of a dga over k,
-- with generators from type a.
data DgaSpec k a = DgaSpec { _gens :: Generators a
                           , _diff :: Differential k a }

-- | Type synonym for degree in the sense of gradings.
type Deg = Int
-- | A collection of algebra generators is represented as a 'Map'
-- whose keys are the generators with value the respective degree.
type Generators a = M.Map a Deg
-- | A differential is represented by an actual Haskell function.
type Differential k a = (Lam k a -> Lam k a)

-- | 'Lam' k a types the elements of algebras
-- specified by instances of 'DgaSpec' k a.
-- The type Vect k b from the HaskellForMaths package
-- types elements of the free vector space over k with
-- basis elements from b.
type Lam k a = Vect k (FGCA a)
-- | Following the HaskellForMaths convention,
-- the type synonym 'FGCA', which is short for FreeGradedCommutativeAlgebra,
-- types the /basis elements/ of the free module structure of such an algebra.
type FGCA a  = Tensor (SymmetricAlgebra a) (ExteriorAlgebra a)
```

Listing 1: Types for representing cochain algebras [Types.hs]

```
data G2 = T1 | T2 | Y1 | Y2  deriving (Eq, Ord, Show)

([t1,t2,y1,y2], apl2) =
    mkDgaWithGenerators [ (T1,0, y1)
                        , (T2,0, y2)
                        , (Y1,1, 0)
                        , (Y2,1, 0) ]


data G = T Int | Y Int  deriving (Eq, Ord, Show)

(t, y) = (injectFGCA_even . T, injectFGCA_odd . Y)

apl :: Int -> DgaSpec Q G
apl m = mkDga $ ts ++ ys
  where
    ts = map (\ j -> (T j, 0, y j)) [1..m]
    ys = map (\ j -> (Y j, 1, 0))   [1..m]
```

Listing 2: Algebra of polynomial differential forms [polynomial-differential-forms.hs]

In `inflexible`, we mimic this way of representing cochain algebras. See Listing 1 for the relevant definitions from the `Types` module. Note, that here and henceforth, the listings often deviate slightly from the actual code in the package. In particular, we leave out fragments that are irrelevant to the general discussion in this thesis (such as most `deriving` clauses) and strip or modify comments. The listings' captions will point to the original source code file, which can be found under the subdirectories `src/` or `interactive/` of the package.

The module `Alg` provides facilities for the construction of `DgaSpec` objects. In Listing 2 we showcase how Example 3.25 can be defined in `inflexible`: the first part realizes the cochain algebra of polynomial differential forms in a concrete dimension as `apl2` ($d = 2$ in Example 3.25), followed by a parametric version `apl`. There are a few things to note here:

- We define the custom data types `G2` and `G` for the generators, whereas in principle, one could also use `String` or any other predefined type with an `Ord` instance. Besides that fact, that our approach seems cleaner, it also has the advantage, that we could provide custom `Show` instances.
- The definition of `apl2` uses the same ingredients as Remark 4.3: The argument to `mkDgaWithGenerators` is a list of triples, each consisting of a generator, its degree, and what its image under the differential should be. Laziness of Haskell breaks the apparently circular use of `y1` and `y2` in the latter.
- The definition of `apl` demonstrates, that not all inhabitants of the type of generators must be used. In this case, given $m \in \mathbb{N}$, the cochain algebra `apl m` uses only `T 1`,...,`T m` and `Y 1`,...,`Y m`.
- As its type signature is fixed, `apl m` represents a cochain algebra over $\mathbb{Q}$ (simply `Q` in `HaskellForMaths`), with generators from the type `G`.

  In contrast, Haskell's polymorphism (with GHC's `NoMonomorphismRestriction`) allows `apl2` to be used via `apl2 :: DgaSpec k G2` as cochain algebra over any ring k.
- All cochain algebra construction functions from the `Alg` module check, that the specified differential is valid. Changing the last line in the definition of `apl2` to

  ```
  , (Y2,1, y1*y2) ]
  ```

  would result in an error message (upon usage of `apl2`).

Gradings

There is one detail that we did not mention yet: In contrast to our treatment in Section 3.3, using *external grading*, graded objects are *not* handled as families of modules in `inflexible`. This is due to practical reasons and the fact that `HaskellForMaths` does not provide the graded objects that we use. Instead, we use the (ungraded) algebras from the latter and treat them as *internally graded*, i.e. as the direct sum over a family of submodules. However, the two approaches are equivalent and we refer to Mac Lane's book [ML1, Ch. VI] for further information about the distinction. It should also be noted,

```
-- (setup for generators as before ...)

a1 = mkDga [ (X1, 2,  0)
           , (X2, 4,  0)
           , (Y1, 9,  x1^3*x2  )
           , (Y2,11,  x1^2*x2^2)
           , (Y3,13,  x1*x2^3  )
           , ( Z,35,  x2^4*y1*y2-x1*x2^3*y1*y3+x1^2*x2^2*y2*y3+x1^18+x2^9)
           ]

w = x2^2*y1*y2 - x1*x2*y1*y3 + x1^2*y2*y3
```

Listing 3: Definition of the cochain algebra A_1 [ex-A1-crowley-loeh.hs.hs]

that this is often not made explicitly in the literature: especially, when graded objects are defined externally, they are frequently used as their internally graded equivalent without further notice.

4.2. Examples

In this section, we show how `inflexible` aids the researcher with the treatment of cochain algebras that are needed as as input for the machinery of Chapter 3, i.e. to produce inflexible, simply-connected manifolds. We start by going through some of the computations that Crowley and Löh needed to make in their article [CL], in order to show that their cochain algebras have the desired properties. Afterwards, we show how `inflexible` can be used to obtain new examples by varying their design pattern (Example 4.6).

First of all, we recall:

Example 4.4 (A_1[CL, Ex. I.1])**.** The generators' degrees and the differential of the minimal Sullivan algebra

$$A_1 \coloneqq \big(\wedge(x_1, x_2, y_1, y_2, y_3, z), \delta\big)$$

shall be given according to the table

generator g	x_1	x_2	y_1	y_2	y_3	z
degree $\deg g$	2	4	9	11	13	35
differential $\delta(g)$	0	0	$x_1^3x_2$	$x_1^2x_2^2$	$x_1x_2^3$	$x_2^2w + x_1^{18} + x_2^9$,

where $w \coloneqq x_2^2y_1y_2 - x_1x_2y_1y_3 + x_1^2y_2y_3$.

Once we made the according definition, see Listing 3, we can use `inflexible`'s functions on it. In the following, we use references of the form "CL.Proposition x.y" to denote Proposition x.y from Crowley and Löh's article [CL].

```
hs> a1
The dga specified by generators (with degrees)
X1  X2  Y1  Y2  Y3   Z
 2   4   9  11  13  35
and differential (given on generators):
X1 |--> 0
X2 |--> 0
Y1 |--> X1^3X2^1
Y2 |--> X1^2X2^2
Y3 |--> X1^1X2^3
Z  |--> X1^2X2^2Y2^1Y3^1-X1^1X2^3Y1^1Y3^1+X2^4Y1^1Y2^1+X2^9+X1^18
hs>
hs> x2^4*y1*y2-x1*x2^3*y1*y3+x1^2*x2^2*y2*y3+x1^18+x2^9 == x2^2*w+x1^18+x2^9
True
hs>
hs> d = _diff a1
hs> x1*x2*w == d (y1*y2*y3)
True
```

Listing 4: Basic computations with cochain algebras [GHCi session]

Basic computations with algebra elements

As simple as it may seem, already the computation of a differential or the verification of equalities can be tremendously helpful. For instance, Listing 4 shows a GHCi session where two "claims" from CL.Example I.1 are checked.

Cohomological computations

In CL.Proposition I.5, Crowley and Löh prove that A_1 is elliptic. This is done by showing that the generators x_1 and x_2 are cohomologically nilpotent. In CL.Proposition I.6, they prove that $[x_2^{16}]$ is a fundamental class for A_1. We show how to check these results in Listing 5, which we now comment on:

- The introduction of `a1Q` might seem peculiar at first, but it is necessary to use the functions from the `Alg.Cohomology` module, which require the base ring to be known. Alternatively, we could have fixed `a1`'s type by adding `a1 :: DgaSpec Q G` at the beginning of Listing 3.

- The function

  ```
  isCoboundary' a1Q :: Lam Q G -> Bool
  ```

  returns `True` if and only if the (homogeneous) input element is a coboundary in `a1Q`. With this, we not only obtain that $[x_1]$ and $[x_2]$ are nilpotent in $H(A_1)$, but also that 19 and 17 are the least positive powers showing this. The function

  ```
  fillCochain' a1Q :: Lam Q G -> Lam Q G
  ```

  provides us with a preimage of x_2^{17} under the differential δ.

```
hs> a1Q = a1 :: DgaSpec Q G
hs>
hs> isCoboundary' a1Q (x1^19)
True
hs> isCoboundary' a1Q (x1^18)
False
hs>
hs> isCoboundary' a1Q (x2^18)
True
hs> isCoboundary' a1Q (x2^17)
True
hs> isCoboundary' a1Q (x2^16)
False
hs>
hs> degree a1Q (x2^16)
Just 64
hs> formalDimension a1Q
64
hs>
hs> Just preim = fillCochain' a1Q (x2^17)
hs> preim
-X1^2X2^1Y2^1Y3^1Z^1+X1^1X2^2Y1^1Y3^1Z^1-X2^3Y1^1Y2^1Z^1+X2^8Z^1+X1^17Y1^1Y2^1Y3^1-X1^17X2^5Y3^1
hs> _diff a1Q preim == x2^17
True
hs>
hs> isElliptic' a1Q
Just True
```

Listing 5: Checking ellipticity and fundamental class of A_1 [GHCi session]

```
f  = genericAlgMor a1 a1 :: Lam (GrevlexPoly Q T) G -> Lam (GrevlexPoly Q T) G
cs = genericDgaEndoConstraints a1 :: [GrevlexPoly Q T]

vol = x2^16
[bx2] = basisElems x2
t_for_x2 = coeff bx2 (f x2)

elim_vars = delete t_for_x2 $ nub $ concat $ map Poly.vars cs

elim_ideal_gens = Poly.eliminate elim_vars cs
```

Listing 6: Computing inflexibility of A_1 [ex-A1-crowley-loeh.hs.hs]

```
hs> mapM_ (\x->putStrLn $ show x <>" |--> "<> show (f x)) (algGenerators' a1)
X1^1 |--> τ1X1^1
X2^1 |--> τ2X2^1+τ3X1^2
Y1^1 |--> τ4Y1^1
Y2^1 |--> τ5Y2^1+τ6X1^1Y1^1
Y3^1 |--> τ7Y3^1+τ9X1^1Y2^1+τ8X2^1Y1^1+τ10X1^2Y1^1
Z^1 |--> τ11Z^1+τ13X1^1Y1^1Y2^1Y3^1+τ14X1^1X2^5Y3^1+τ12X2^6Y2^1+...[many summands]...
hs>
hs> cs
[-τ1^3τ2+τ4,-τ1^3τ3,-τ1^2τ2^2+τ5,-2τ1^2τ2τ3+τ6,-τ1^2τ3^2,...[many elements]...
hs>
hs> elim_ideal_gens
[τ2^24-τ2^15]
```

Listing 7: Showcase the objects from Listing 6 [GHCi session]

- Whereas the manual analysis of $[x_2^{16}] \in H(A_1)$ in CL.Proposition I.6 is tedious and lengthy, our function `isCoboundary'` quickly tells us, that it is non-zero. Together with the fact that it has the correct degree, this implies that $H^{64}(A_1)$ is generated by $[x_2^{16}]$.
- In fact, checking of ellipticity can be automated for cochain algebras whose differential vanishes on generators of even degree, see Theorem 4.7. Here, this is applied via `isElliptic' a1Q`.

Computing (in)flexibility

CL.Proposition I.10 contains the proof that A_1 is inflexible. For this, the form of a generic morphism of graded algebras $A_1 \to A_1$ is analysed. In `inflexible`, a generic such morphism can be built via the function `genericAlgMor` from the module `Alg.GenericMor`. The function extends the base ring of the algebra to a polynomial ring and inserts a variable of the latter in place of the coefficients that are named $\alpha_1, \alpha_2, \alpha_{2,1}, \beta_1, \dots$ in the proof of CL.Proposition I.10. The requirement that this generic morphism is a morphism of cochain algebras, translates into a system of polynomial constraints. Instead of treating this system manually, one can hope that this can also be automated. In certain cases, this is indeed possible and we refer the reader to section Section 4.4 for the exact statements. Here, we only illustrate in Listings 6 and 7 how this can be implemented:

- We assign to `f` and `cs` the generic morphism and constraints, respectively.
- In the interactive session, we show `f` on the generators of A_1. Here, `τ1`, `τ2`, ... denote the variables of the polynomial ring. The computation of `cs` yields the generic constraints in the form of polynomials in the `τ...`, with the meaning that substituting field values for the variables in `f` yields a morphism of cochain algebras if and only if all the polynomials in `cs` evaluate to zero.
- The rather clumsy code following the definition of `f` and `cs` computes that A_1 is indeed inflexible: the fact that `elim_ideal_gens` contains a polynomial in `τ2` implies

that `τ2` can only assume finitely many values, which means that cochain morphisms $A_1 \to A_1$ can only have finitely many mapping degrees.

Ideally, this functionality would be included in `inflexible` itself. Unfortunately, it was not implemented yet and is repeated multiple times in the examples in `interactive/`.

New examples of lower dimensions

Among all inflexible minimal Sullivan algebras previously known, yielding the manifolds in Examples 3.9, the cochain algebra A_1 by Crowley and Löh (Example 4.4) has the least (formal) dimension $\dim A_1 = 64$. It is thus a natural question to ask:

Question 4.5. Do inflexible minimal Sullivan algebras exist in dimensions below 64? What is the lowest dimension that admits such algebras?

In experimenting with the design pattern from Crowley and Löh [CL, Sec. I.1], we found the following partial answer to this question: There exists an elliptic, inflexible, minimal Sullivan algebra

- of dimension 60, `interactive/ex-A1mod.hs`,
- of dimension 38, `interactive/ex-A0.hs`, and
- of dimension 36, `interactive/ex-A0mod.hs`.

We only present the cochain algebra A_0 of dimension 38 in detail here:

Example 4.6 (A_0)**.** The generators' degrees and the differential of the minimal Sullivan algebra

$$A_0 \coloneqq \big(\wedge(x_1, x_2, y_1, y_2, y_3, z), \delta\big)$$

shall be given according to the table

generator g	x_1	x_2	y_1	y_2	y_3	z
degree $\deg g$	2	2	7	7	7	19
differential $\delta(g)$	0	0	$x_1^3 x_2$	$x_1^2 x_2^2$	$x_1 x_2^3$	$x_1 w + x_1^{10} + x_2^{10}$,

where $w \coloneqq x_2^2 y_1 y_2 - x_1 x_2 y_1 y_3 + x_1^2 y_2 y_3$ (this is the same w as in Example 4.4).

While our example in `interactive/ex-A0mod.hs` is in some sense the "minimal" example that fits the design pattern by Crowley and Löh, A_0 has the advantage that its formal dimension is not divisible by 4, hence we do not have to check its intersection form in order to know that it is realizable by a manifold.

Listing 8 contains the full example file that does the calculations for A_0 and in Listing 9 we display the results in a GHCi session. As this is fairly similar to the pieces that we discussed in detail for A_1, we only comment on the new parts:

```
{−# LANGUAGE NoMonomorphismRestriction #−}
import Prelude hiding ( (*>), (<*) )
import ImportAll
import qualified ImportPoly     as Poly
import qualified ImportPolySage as Poly
import Var (T(T))

data G = X1 | X2 | Y1 | Y2 | Y3 | Z  deriving (Eq, Ord, Show, Enum, Bounded)
[x1,x2, y1,y2,y3, z] =
    (injectFGCA_even <$> [X1,X2]) ++ (injectFGCA_odd <$> [Y1,Y2,Y3,Z])

−− A0. A variant of A1 to A4 from Crowley/Löh
a0 = mkDga [ (X1, 2,  0), (X2, 2,  0)
           , (Y1, 7,  x1^3*x2), (Y2, 7,  x1^2*x2^2), (Y3, 7,  x1*x2^3)
           , ( Z,19,  x1*w + x1^10+x2^10)]
  where w = x2^2*y1*y2 - x1*x2*y1*y3 + x1^2*y2*y3

(vol_gen, vol_gen_power) = (x1, 19)
vol = vol_gen^vol_gen_power

a0Q = a0 :: DgaSpec Q G
(my_dga, my_dgaQ) = (a0, a0Q)

is_elliptic = maybe False id $ isElliptic' my_dgaQ
have_fundamental_class = dropCache $ do
    let q0 = degree my_dgaQ vol == Just (formalDimension my_dgaQ)
    q1 <- isCocycle my_dgaQ vol
    q2 <- not <$> isCoboundary my_dgaQ vol
    return $ q0 && q1 && q2
basic_info = print my_dga >> do
    putStrLn $ "Of dimension: " <> show (formalDimension my_dga)
    putStrLn $ "Elliptic: " <> show is_elliptic
    putStrLn $ "Fundamental class: " <>
            if have_fundamental_class then show vol else "None"

f  = genericAlgMor @(GrevlexPoly Q T) my_dga my_dga
cs = genericDgaEndoConstraints @(GrevlexPoly Q T) my_dga

[bvol] = basisElems vol_gen
t_for_vol_gen = coeff bvol (f vol_gen)
elim_vars = delete t_for_vol_gen $ nub $ concat $ map Poly.vars cs

elim_ideal_gens = Poly.eliminate elim_vars cs
elim_ideal_gens_via_sage = Poly.sageEliminate elim_vars cs

elim_ideal_finite = Poly.sageHasFiniteProjection cs t_for_vol_gen
```

Listing 8: Complete example treating A_0 [ex-A0.hs]

```
hs> basic_info
The dga specified by generators (with degrees)
X1  X2  Y1  Y2  Y3   Z
 2   2   7   7   7  19
and differential (given on generators):
X1 |--> 0
X2 |--> 0
Y1 |--> X1^3X2^1
Y2 |--> X1^2X2^2
Y3 |--> X1^1X2^3
Z  |--> X1^3Y2^1Y3^1-X1^2X2^1Y1^1Y3^1+X1^1X2^2Y1^1Y2^1+X1^10+X2^10
Of dimension: 38
Elliptic: True
Fundamental class: X1^19
hs>
hs> -- elim_ideal_gens -- may take some time
hs> elim_ideal_gens_via_sage
[τ2^20-τ2^18]
hs> elim_ideal_finite
True
```

Listing 9: Results of the computations from Listing 8 [GHCi session]

– We do not evaluate `elim_ideal_gens` this time, because it is calculated by the `HaskellForMaths` library. As previously mentioned (page 31), its pure Haskell implementation for Gröbner basis calculation is easily overchallenged. In this case, it still computes in a reasonable amount of time – about one minute at the author's machine – but it could not handle the polynomials that appear in `ex-A0mod.hs` – at least not in a couple of hours on the author's machine.

– Instead, we look at `elim_ideal_gens_via_sage`, which returns its result almost immediately.

– Alternatively, `elim_ideal_finite` can be used to check directly, that the polynomial constraints allow only for finitely many mapping degrees.

4.3. Decidable ellipticity

In this section, we prove that certain cochain algebras allow for an algorithmic detection of ellipticity:

Theorem 4.7 (decidable ellipticity)**.** Let (A, δ) be a connected cochain algebra with the property that all generators of even degree are cocycles. Then (A, δ) is elliptic if and only if all generators of even degree are cohomologically nilpotent. In particular, it is algorithmically decidable whether (A, δ) is elliptic if (A, δ) is a minimal Sullivan algebra.

The main ingredients for the proof are already contained in the first paragraph of the proof of CL.Proposition I.5, which we generalize and expand on:

Proof. By definition, our cochain algebras are finitely generated. Hence, we only have to treat $H(A, \delta)$.

On the one hand, if the latter is finite dimensional, then all classes of degree at least 1 are nilpotent for degree reasons. On the other hand, the converse also holds: it clearly suffices to show, that $H(A, \delta)$ is finitely generated as an algebra. Indeed, this can be shown by the following trick (cf. proof of Proposition 32.1 in the book by Félix/Halperin/Thomas [FHT]): Consider the subalgebra A_{even} of A that is generated by the even degree generators of A. Because the odd degree generators square to zero, there are only finitely many products of such generators (as in Proposition 3.23 (i)). Thus, the algebra multiplication turns A into a finitely generated A_{even}-module, and the assumption on the even degree generators implies that $\ker(\delta)$ is an A_{even}-submodule of A. It follows, that $\ker(\delta)$ is also a finitely generated A_{even}-module, say by elements $a_1, \dots, a_N$, because A_{even} is Noetherian as a polynomial ring over a field. But then $H(A, \delta)$ is evidently generated as an algebra, by the classes of all even degree generators of A, together with the classes $[a_1], \dots, [a_N]$.

To prove the first part of the claim, it now suffices to show: all cohomology classes of degree at least 1 are nilpotent if and only if the classes of the even degree generators are nilpotent. One implication is trivial. For the other one, assume the nilpotency of the generator classes. Let $a \in \ker(\delta)$ with $\deg a \geq 1$ and let $\ell \in \mathbb{N}$. By the argument in the previous step, $a^\ell \in \ker(\delta)$ can be written as a finite linear combination of the a_j with coefficients in A_{even}. But since the degrees of the a_j are fixed, the degrees of the coefficients must become large for increasing values of ℓ. As all elements of A_{even} of high degree are cohomologically trivial by assumption, we have $[a]^\ell = [a^\ell] = 0$ for large enough ℓ.

The second part now follows from the first one: Let X and Y be the set of odd and even degree generators of A, respectively, and let

$$d \coloneqq \sum_{x \in X} \deg x - \sum_{y \in Y} (\deg y - 1) \ \in \mathbb{N}.$$

Well known algorithms from linear algebra can then be used to compute, for all $x \in X$, whether $x^{1+\lfloor d/\deg x \rfloor}$ is a coboundary, i.e. whether it vanishes in cohomology. If all such generator powers *are* coboundaries, then (A, δ) must be elliptic by the first part. If at least one of them is *not* a coboundary, (A, δ) cannot be elliptic: if it was, it would have formal dimension d [FHT, Prop. 38.3], hence could not have non-trivial cohomology above degree d.

■

The decision criterion from the last part of the proof is implemented in the function `isElliptic` in the module `Alg.Cohomology` of `inflexible`.

4.4. Decidable (in)flexibility

In this section, we introduce the notion of a generic morphism between free graded-commutative algebras. We then use this tool to prove that special forms of fundamental classes of elliptic Sullivan algebras imply algorithmic detectability of (in)flexibility and existence of a prescribed mapping degree (Theorem 4.9).

Definition 4.8 (generic morphism)**.** Let $A = \wedge(x_1, \dots, x_r)$ and $B = \wedge(y_1, \dots, y_s)$ be free graded-commutative algebras with generators of degree at least 1.

- For $k \in \mathbb{N}_{\geq 1}$, let $M(B)_k$ be the basis of B^k from Proposition 3.23 (i), i.e. consisting of all products in the y_j of total degree k.
- Let
$$P(A,B) \coloneqq k\big[T_{x_j,m} \;\big|\; j \in \{1,\dots,r\} \wedge m \in M(B)_{\deg x_j}\big]$$
be a polynomial ring over k in $\sum_{j=1}^{r} \dim_k B^{\deg x_j} \in \mathbb{N}$ variables.
- We let $\tilde{A} \coloneqq P(A,B) \otimes A$ denote the extension of scalars of A along $k \hookrightarrow P(A,B)$ (where $P(A,B)$ is concentrated in degree 0), and similarly for $\tilde{B}$.
- The *generic morphism from A to B* is the morphism of graded $P(A,B)$-algebras $f\colon \tilde{A} \to \tilde{B}$ that is given by
$$f(x_j) \coloneqq \sum_{m \in M(B)_{\deg x_j}} T_{x_j,m} \cdot m$$
for all $j \in \{1,\dots,r\}$.

The generic endomorphism of a cochain algebra now plays an important role in the proof of the following result:

Theorem 4.9 (decidable (in)flexibility)**.** Let k be countable and let $\overline{k}$ be an algebraic closure of k. Let (A,δ) be a connected, elliptic Sullivan algebra.

(i) Let x be a generator of A, such that some power of x represents a fundamental class for (A,δ). Then it is algorithmically decidable, whether (A,δ) is inflexible over $\overline{k}$.

(ii) Let $d \in \mathbb{N}$ be the formal dimension of (A,δ) and let $m \in M(A)_d$ be a representative of a fundamental class for (A,δ). Let $a \in \overline{k}$. Then it is algorithmically decidable, whether (A,δ) over $\overline{k}$ admits a cochain algebra endomorphism of mapping degree a.

Proof. Let X be the set of generators of A. We consider $f\colon \tilde{A} \to \tilde{A}$, the generic morphism from A to A, and extend δ to $\tilde{A}$. Comparing coefficients according to the bases $M(A)$ among all elements of the set

$$\big\{(f \circ \delta)(x) - (\delta \circ f)(x) \;\big|\; x \in X\big\}$$

yields a finite set $C \subseteq P(A, A)$ of polynomials.

Substitution of values from k for the variables from $P(A, A)$ in f clearly yields an endomorphism of the cochain algebra (A, δ) if and only if all polynomials from C evaluate to 0 on these values.

In the language of varieties, for which we refer to the book by Cox, Little and O'Shea [CLO, Ch. 1], we are looking at the affine variety $\mathbf{V}(C) \subseteq k^N$, where $N \in \mathbb{N}$ is the number of variables of $P(A, A)$. Cochain algebra endomorphisms of (A, δ) are then parametrized by $\mathbf{V}(C)$: for a point $p \in \mathbf{V}(C)$, let f_p denote the corresponding morphism. We now proceed separately for the two parts of the claim.

Ad (i): Let $\pi\colon k^N \to k$ denote the projection onto the coordinate that corresponds to the variable $T_{x,x}$. By assumption, a power of x represents a fundamental class of (A, δ), hence the mapping degree of f_p is given by the very same power of $\pi(p)$, for all $p \in \mathbf{V}(C)$. In other words, if $\deg((A, \delta), (A, \delta))$ denotes the set mapping degrees of endomorphisms of (A, δ), we have

$$\#\pi\big(\mathbf{V}(C)\big) < \infty \iff \#\deg((A, \delta), (A, \delta)) < \infty.$$

We now apply some basic (algorithmic) algebraic geometry: First, the *Closure Theorem* [CLO, Ch. 4, §4, Thm. 4] implies, that the Zariski closure of $\pi(\mathbf{V}(C))$ is equal to $\mathbf{V}(I)$, both in $\overline{k}^N$, where $I \coloneqq \langle C\rangle \cap \overline{k}[T_{x,x}]$ is a so-called *elimination ideal*. Note the move to $\overline{k}$ here, which is necessary, because the Closure Theorem only applies to algebraically closed fields. Observe, that I is an ideal of a polynomial ring in a single variable, hence a principal ideal.

Second, the *Elimination Theorem* [CLO, Ch. 3, §1, Thm. 2] (and its proof and the discussion of Gröbner bases in Chapter 2 of loc. cit.) imply that we can algorithmically determine a generating set U of I. If I is the trivial ideal, we either get $U = \emptyset$ or $U = \{0\}$, depending on the exact algorithm. Otherwise, we have $U = \{g\}$ for some non-constant polynomial g.

We now use the fact that the Zariski closure operation is easy to describe in single dimensional affine space: either a subset of the field is finite, then its Zariski closure is the set itself [CLO, Ch. 1, §2, Exerc. 6], or it is infinite, in which case its Zariski closure must be the whole field.

Combining all of the previous steps, we obtain: $\deg((A, \delta), (A, \delta)) \subseteq \overline{k}$ is finite, if and only if $U = \{g\}$ for non-constant g, and the latter is algorithmically decidable. This finishes the proof of claim (i).

Ad (ii): We look at $f(m) \in \tilde{A}$ and its coefficient $h \in P(A, A)$ of m. For $p \in \mathbf{V}(C)$, the mapping degree of f_p is then given by $h(p)$.

In geometric terms, we are interested in the intersection

$$\mathbf{V}(C) \cap \mathbf{V}(h - a) = \mathbf{V}\big(C \cup \{h - a\}\big)$$

inside k^N, and this will be non-empty, if and only if (A, δ) admits an endomorphism of mapping degree a. The proof is now completed by an application of the *consistency algorithm* [CLO, Ch. 4, §1], which decides whether $C \cup \{h - a\}$ has a solution in $\overline{k}^N$. ∎

The construction of generic morphisms and the computation of the polynomials C from the proof of Theorem 4.9 are implemented in the functions `genericAlgMor` and `genericDgaEndoConstraints` in the module `Alg.GenericMor` of `inflexible`.

Chapter 5:

A brief glimpse of infinity

For many mathematicians, their first contact points with ∞-categories are rather vague and handwavy. For example, the speaker in a talk might refer to work that happens in the language of ∞-categories, but in order to make the talk accessible to a broad audience, they will try to phrase the material without mentioning the word ∞-category. While this approach often suffices to keep the talk going, it does not bring the audience closer to ∞-category theory, but rather leaves the uninitiated in a state of uncertainty or even indifference. The author was asked more than once by someone with only a nebulous idea of ∞-categories: *What is an ∞-category?*, in the sense that they wanted to see a rigorous definition and how it relates to the concept of *higher morphisms*. This chapter is an attempt to give one possible answer to this question in a brief and self-contained way, without any proofs but with lots of pointers to the literature for further reading. The only prerequisite is some familiarity with basic category theory, e.g. as in the first two chapters of Mac Lane's book [ML2, Sec. I.1–4, I.8, II.1–4].

In this chapter, "category" always means 1-category and we will explicitly use the term "∞-category" as in Definition 5.4.

5.1. From 1 to ∞ in a nutshell

Our first goal is to give a precise definition of ∞-categories in terms of Lurie and to observe that there's a formal way to turn 1-categories and functors into ∞-categorical ones. We refer the reader to the literature [Ci][Ri2][Grth][K][Lu1] for a more comprehensive treatment of the subject.

The underlying foundational objects for this theory are *simplicial sets*, a generalization of (ordered) simplicial complexes, for which we give a rigorous definition in the following but almost no illustration or background. We recommend the beautiful survey article by Friedman [Fr] as an introduction to the subject and the textbook by Goerss and Jardine [GJ] for further reading.

Definition 5.1 (simplicial set)**.** (i) Let $(X, \leq)$ be a preorder, i.e. $\leq$ is a transitive and reflexive relation on X. We define the *preorder category* $\mathrm{Pre}(X, \leq)$ to have X as objects and a unique morphism from x to x' exactly if $x \leq x'$ (and no further morphisms).

(ii) For $n \in \mathbb{N}$ let $[n] \coloneqq \mathrm{Pre}(\{0, \dots, n\}, \leq)$. Let $\triangle$ be the full subcategory of Cat on the objects $\{[n] \mid n \in \mathbb{N}\}$.

(iii) A *simplicial set* is a functor $\triangle^{\mathsf{op}} \to \mathsf{Set}$. The *category* sSet *of simplicial sets* is the functor category $\mathrm{Fun}(\triangle^{\mathsf{op}}, \mathsf{Set})$. Hence, a morphism of simplicial sets is a natural transformation of functors. For a simplicial set X and $n \in \mathbb{N}$ one often uses the notation $X_n \coloneqq X([n])$.

(iv) Let X be a simplicial set. Then a *simplicial subset* of X is a simplicial set U such that for all $n \in \mathbb{N}$ we have $U_n \subseteq X_n$ and for all morphisms $f\colon [m] \to [n]$ in $\triangle$ we have $U(f) = X(f)|_{U_n}$.

Remark 5.2 (operations with simplicial sets)**.** Many operations one normally does with plain sets can be lifted to simplicial sets by just applying them "dimensionwise". For instance, for two simplicial subsets U, V of the same simplicial set, one can define $U \cap V$ by $(U \cap V)_n \coloneqq U_n \cap V_n$ for all $n \in \mathbb{N}$ and restriction on morphisms. In particular the expression "smallest simplicial subset with some property" can be made precise by taking intersections as usual.

The formal reason why this works, is the fact that limits and colimits in a functor category (like sSet) are just computed "pointwise" if they exist in the target category. Taking intersection is a pullback in Set, for example.

With this remark in mind, we can now introduce some of the most important simplicial sets:

Examples 5.3 (standard simplex, horn)**.**

(i) Let $n \in \mathbb{N}$. The simplicial set $\Delta^n \coloneqq \mathrm{Mor}_{\triangle}(\bullet, [n])$ is the *standard n-simplex*. For $n \geq 1$ and $i \in \{0, \dots, n\}$ the unique functor $d_i^{n-1} \in \mathrm{Mor}_{\triangle}([n-1], [n]) = (\Delta^n)_{n-1}$ with $d_i^{n-1}(\{0, \dots, n-1\}) = \{0, \dots, n\} \setminus \{i\}$ is the *i-th coface map*.

(ii) For $n \in \mathbb{N}$ the smallest simplicial subset of Δ^{n+1} that contains $\{d_0^n, \dots, d_{n+1}^n\} \subseteq (\Delta^{n+1})_n$ is the *simplicial n-sphere*.

(iii) Let $n \in \mathbb{N}_{\geq 1}$ and $i \in \{0, \dots, n\}$. The *(n,i)-horn* Λ_i^n is the smallest simplicial subset of the standard n-simplex that contains $\{d_0^{n-1}, \dots, d_n^{n-1}\} \setminus \{d_i^{n-1}\} \subseteq (\Delta^n)_{n-1}$.

Informally speaking, the simplicial $(n-1)$-sphere is obtained from Δ^n by removing the "interior" of the n-simplex; and by further removing the $(n-1)$-dimensional face opposite to the i-th vertex we get the horn Λ_i^n. The latter now plays the main role in the definition of an ∞-category:

Definition 5.4 (∞-category)**.** An *∞-category* is a simplicial set C with the following property: For all $n \in \mathbb{N}_{\geq 1}$ and all $i \in \{1, \dots, n-1\}$ every morphism $\Lambda_i^n \to C$ factors over the inclusion $\Lambda_i^n \hookrightarrow \Delta^n$.

$$\begin{array}{ccc} \Lambda_i^n & \longrightarrow & C \\ \downarrow & \overset{\exists}{\nearrow} & \\ \Delta^n & & \end{array}$$

A usual formulation of this property is the following: In an ∞-category all so-called *inner horns* $(0 < i < n)$ have a *not* necessarily unique(!) "filler". Those fillers are witnesses of (higher dimensional) compositions in an ∞-category and the existence statement then ensures that a composition can always be found. See Remark 5.12 for the case of Λ^2_1; for examples using higher dimensional horns and the existence of their fillers, we refer to the literature [Lu1, Sec. 1.2.3][Ci, proof of Lem. 1.6.2][K, 003U, 0041]. For nomenclature linking the different pieces of an ∞-category to classical categorical terms, see Definition 5.11.

Example 5.5. For all $n \in \mathbb{N}$, the standard n-simplex is an ∞-category. On the other hand, for $n \in \mathbb{N}_{\geq 1}$, the simplicial n-sphere is *not* an ∞-category.

Remark 5.6 (functor ∞-categories)**.** For ∞-categories C and D, a functor $C \to D$ is simply taken to be a morphism of simplicial sets.

A certain construction [K, 0060][GJ, Sec. I.5] turns the category of simplicial sets into a *cartesian closed* category [ML2, Sec. IV.6]: it provides a simplicial set Y^X that extends the set of morphisms between two simplicial sets X and Y, i.e. with $(Y^X)_0 = \mathrm{Mor}_{\mathsf{sSet}}(X, Y)$. This is often called an *internal Hom*, because it is again an object of the ambient category, or in this specific case the *function complex from X to Y*. It can be shown [Ci, Cor. 3.2.10] that Y^X is again an ∞-category whenever Y is an ∞-category. In particular, this shows how functors $C \to D$ for fixed ∞-categories C and D give rise to a functor ∞-category $\mathrm{Fun}(C, D)$. ⌟

Remark 5.7 (size issues)**.** As the reader has probably already noticed, the operation of taking "all functors $C \to D$" might or might not yield a well defined object, depending on the set theory that one chooses as a mathematical foundation. To be more specific: If one considers functors between categories "of the same size" (in terms of the "number" of objects and morphisms) one often obtains something that is "bigger" than that. On the other hand, if one imposes the correct size constraints on C and D, the result can also be kept "small enough".

For example, suppose that we work within NBG (von Neumann, Bernays, Gödel) set theory. Then a category is usually defined to allow for a proper class of objects, but most often only morphism *sets* between any two objects are permitted. If we consider such categories, it is well known that one is usually not permitted to form the functor category between two categories whose classes of objects are both proper classes. Though, if we consider a *small* category C, i.e. its class of objects is a set, and another category D, the functor category $\mathrm{Fun}(C, D)$ is again a well defined category in the sense of this paragraph.

Analogously, such considerations can be made for ∞-categories when appropriate size restrictions are in place [Ci, Cor. 5.7.7]. Even then, however, the "two layers" of NBG set theory are not enough for the arguments in Section 6.2: there we want to look at the functor category $\mathrm{Fun}(\mathsf{Top}_*, \mathsf{Ch}_{\mathbb{R}})$ where the size conditions are *not* met. A usual way to circumvent this problem is the usage of so-called *Grothendieck universes*. Roughly speaking, this means the following:

A set U is called a *universe* if it satisfies certain closure properties like the *power set axiom*:

$$\forall x \in U, \quad \{y \mid y \subseteq x\} \in U.$$

Then an additional axiom, the *universe axiom*, is imposed:

$$\forall x \quad \exists\, U \text{ universe}, \quad x \in U.$$

One can then just choose a universe that is big enough to support all the operations one wants to make, i.e. such that "enough layers exist". (For our purpose, as one may convince oneself, it is actually enough to consider just "one more layer" on top of NBG, giving it a name like "superclasses" or "conglomerates" if one wants to maintain the hierarchy that is already in place.)

For further information, we refer the reader to the literature about size issues and possible solutions: A concise introduction to Grothendieck universes can be found in a paper by Low [Lo]; a gentle explanation of the "conglomerate solution" can be found in the category theory book by Adámek, Herrlich and Strecker [AHS, 2 Foundations]; an expository article by Shulman compares set theoretical foundations for category theory from a modern perspective [Sh]. ⌟

Now that we have defined what an ∞-category is, we would certainly want to "import" all the 1-categories that we already know of into this setting. Put differently, we really want that ∞-categories provide a generalization of 1-categories. This can be done via the *nerve* functor:

Definition 5.8 (nerve of a 1-category)**.** Let $\iota\colon \triangle \hookrightarrow \mathsf{Cat}$ denote the inclusion functor. Then the *nerve functor* $\mathrm{N}\colon \mathsf{Cat} \to \mathsf{sSet}$ arises from the functor

$$\mathrm{Mor}_{\mathsf{Cat}}(\iota^{\mathsf{op}} \bullet, \bullet)\colon \triangle^{\mathsf{op}} \times \mathsf{Cat} \to \mathsf{Set}$$

by *currying*, i.e. $\mathrm{N}(X) = \mathrm{Mor}_{\mathsf{Cat}}(\iota^{\mathsf{op}} \bullet, X)$. ⌟

More formally, this currying construction can be seen as an application of the exponential law in the category of categories: $\mathrm{Fun}(C \times C', D) \cong \mathrm{Fun}(C', \mathrm{Fun}(C, D))$, analogously to the exponential law for functions [Mʟ2, Exerc. 2 of Sec. II.5].

Example 5.9 (standard simplex via nerve)**.** For all $n \in \mathbb{N}$ we have $\Delta^n = \mathrm{N}([n])$.

Proposition 5.10 (nerve is ∞-category [Ci, Prop. 1.4.11, Ex. 1.5.3][Lu1, Prop. 1.1.2.2] [K, 002Z])**.** For every category C, the nerve $\mathrm{N}(C)$ is an ∞-category. Furthermore, the nerve functor N is fully faithful, i.e. for all categories C and D the map $\mathrm{Mor}_{\mathsf{Cat}}(C, D) \ni F \mapsto \mathrm{N}(F) \in \mathrm{Mor}_{\mathsf{sSet}}(\mathrm{N}(C), \mathrm{N}(D))$ is bijective.

This means that whenever one embeds two 1-categories into the ∞-world via the nerve, there will be exactly "the same" functors between them as there were before. So in this

sense, ∞-categories generalize 1-categories.

However, while it is nice that we have 1-categories as ∞-categories now, it's not an enhancement of the original 1-categories. Often, one builds ∞-categories by other means and then they contain some homotopy theoretic information of the original 1-category. In Section 5.3 we will go into a few more details.

5.2. Working in ∞-categories

Now that we know how to model ∞-categories by simplicial sets, we can set up some nomenclature to make *working* in an ∞-category more similar to working in a 1-category.

Definition 5.11 (nomenclature for ∞-categories)**.** Let C be an ∞-category and let us use the notations of Definition 5.1 (iii) and Example 5.3 (i). Then

- *objects of* C are the elements of C_0,
- *morphisms of* C are the elements of C_1,
- *2-morphisms of* C are the elements of C_2,
- and generally for $n \in \mathbb{N}$ the *n-morphisms of* C are the elements of C_n.

For two objects x and y in C, a morphism $x \to y$ is a morphism f of C such that $C(d_1^0)(f) = x$ and $C(d_0^0)(f) = y$. For an object x of C the morphism $C([1] \to [0])(x)$ of C is the *identity morphism of* x, where we simply write $[1] \to [0]$ for the unique such functor. ⌟

Note that "morphism" and "1-morphism" mean exactly the same thing, as do "object" and "0-morphism", though the latter is not used by all authors. With this notation at hand, one can, at least superficially, start to use an ∞-category just as a 1-category. However, there is one big catch:

Remark 5.12 (lack of composition map)**.** In an ∞-category C, there is usually no "composition map $\mathrm{Mor}(y,z) \times \mathrm{Mor}(x,y) \to \mathrm{Mor}(x,z)$" like in 1-categories. Instead, given morphisms $f\colon x \to y$ and $g\colon y \to z$ there could be many possible compositions of f and g – and this is indeed one of the key notions in the generalization from 1-categories to ∞-categories. More precisely, f and g give rise to a map $\Lambda_1^2 \to C$ of simplicial sets, and any extension $\sigma\colon \Delta^2 \to C$ to the standard simplex

determines *a* composition h of f and g, namely:

$$h = C(d_1^1)\big(\sigma_{[2]}(\mathrm{Id}_{[2]})\big) \in C_1.$$

The existence of such an extension is guaranteed by the fact that C is an ∞-category, but there can (and usually will) be many! In the situation above, we will say that σ *witnesses that* h *is a composition of* f *and* g.

A useful analogy is the following: When defining the concatenation of two compatible paths with common domain interval in a topological space, one has to subdivide the interval, which is usually done by splitting it in the middle. This might look most symmetric at first, but as soon as we consider three paths, in fact *no* such subdivision will make the concatenation map associative. One way to enforce this, is passing to the homotopy category, but as mentioned below (Remark 5.21), this might be a bad idea. Instead, we could just declare that *any* concatenation (regardless of the chosen interval subdivision) is a valid composition of the paths, i.e. we allow many different compositions of two paths. ⌟

With this in mind, the next natural question is: How does the notion of *(commutative) diagrams* [K, 005H] enter ∞-category theory? Indeed, since we already know what a functor of ∞-categories is, we could say, analogously to the usual definition for 1-categories, that a diagram in an ∞-category C is a functor $C' \to C$ from some ∞-category C' (the "shape of the diagram") to C. Though this would suffice on a purely technical level [Ci, Thm. 7.3.22][Lu1, Prop. 4.2.3.14] it is more convenient to allow "shapes" that are not ∞-categories themselves:

Definition 5.13 (diagram in an ∞-category)**.** Let C be an ∞-category and let K be a simplicial set. A *diagram in* C *indexed by* K is a morphism of simplicial sets $K \to C$.

Example 5.14 (diagrams indexed by simplices)**.** Let $n \in \mathbb{N}$ and let C be an ∞-category. A diagram $\Delta^n \to C$ is equivalently an n-morphism of C; more precisely: the map

$$\mathrm{Mor}_{\mathsf{sSet}}(\Delta^n, C) \to C_n, \quad \sigma \mapsto \sigma_{[n]}(\mathrm{Id}_{[n]})$$

is a bijection. (This is an instance of the Yoneda lemma [Mac2, Sec. III.2].) It is common to identify n-morphisms of C with diagrams $\Delta^n \to C$ via this bijection whenever it is convenient.

Example 5.15 (commutative squares)**.** Let $K \coloneqq \mathrm{N}([1] \times [1])$ be the nerve of the product category $[1] \times [1]$ (taken in Cat). More explicitly, the latter can be depicted as

$$\begin{array}{ccc} (0,0) & \xrightarrow{f} & (1,0) \\ {\scriptstyle f'}\downarrow & \searrow^{h} & \downarrow{\scriptstyle g} \\ (0,1) & \xrightarrow[g']{} & (1,1) \end{array}$$

and its nerve has the obvious additional 2-simplices σ and τ "filling the two triangles". Then a diagram $p\colon K \to C$ amounts to the data of

- four objects $p((0,0))$, $p((1,0))$, $p((0,1))$ and $p((1,1))$,
- five morphisms $p(f)$, $p(g)$, $p(f')$, $p(g')$ and $p(h)$, and
- two 2-morphisms $p(\sigma)$ and $p(\tau)$

of C, such that

- the morphisms have the correct source and target and
- the 2-morphisms $p(\sigma)$ and $p(\tau)$ witness that $p(h)$ is a composition of $p(f)$ and $p(g)$ as well as of $p(f')$ and $p(g')$. (Here we apply Example 5.14.)

For example, when C is an ∞-category of topological spaces, this would mean that $p(\sigma)$ and $p(\tau)$ are homotopies $p(h) \simeq p(g) \circ p(f)$ and $p(h) \simeq p(g') \circ p(f')$. So in particular, every such diagram in C gives a 1-categorical commutative diagram in the homotopy category of C. But it is crucial to understand that the latter notion is just a *property* of a 1-categorical diagram while an actual diagram $K \to C$ carries the 2-morphisms $p(\sigma)$ and $p(\tau)$ as additional *data*. ⌟

Remark 5.16 (homotopy coherence)**.** The last example already shows that commutative diagrams in ∞-categories are quite a bit more delicate to handle than in 1-categories. Of course, more complex diagrams than commutative squares require even more data to be specified by a diagram. In particular, diagrams indexed by "infinite shapes", e.g. $\mathrm{N}(\mathrm{Pre}(\mathbb{N}, \leq))$, may require chosen n-morphisms in the target ∞-category for *all* $n \in \mathbb{N}$ in a compatible way – one also speaks of a *coherent* choice of higher morphisms. For further reading about homotopy coherence, we recommend notes by Riehl [Ri1], which also contain an example [Ri1, Ex. I.3.4] of a homotopy commutative diagram that cannot be made homotopy coherent.

Remark 5.17 (sloppy notation)**.** As we have seen in Example 5.15 (we will reuse its notation here), a commutative square in an ∞-category needs to specify a "diagonal" morphism. Indeed, if one defines a commutative square in a 1-category C' analogously as a functor $F\colon [1] \times [1] \to C'$, one a priori has to define $F(h)$. But since compositions in 1-categories are unique, one easily sees that keeping this piece of data is *redundant*, i.e. inferable from $(F(f), F(g))$. So for such a 1-categorical commutative square, the restriction of F to the "outer" four morphisms uniquely determines F and this is why we usually don't mention the "diagonal" at all.

However, this is not true anymore for diagrams in ∞-categories! So when speaking of a commutative square

$$\begin{array}{ccc} a & \longrightarrow & b \\ \downarrow & & \downarrow \\ c & \longrightarrow & d \end{array}$$

in an ∞-category C, one really means a diagram $p\colon K \to C$ where $p((0,0)) = a \to d = p((1,1))$ as well as $p(\sigma)$ and $p(\tau)$ are not visible, although they are implicit data associated to the square. ⌟

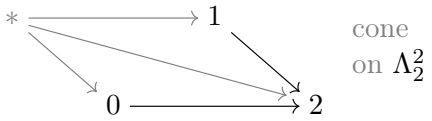
Outlook 5.18 ((co)limits)**.** Having said what a diagram is, one would like to speak about limits and colimits of such. Unfortunately, we cannot fully define this notion here, because it requires a more technical insight into the theory of ∞-categories and simplicial sets. To give an idea of why this is the case, we may try to transplant the usual picture of how one thinks about limits in 1-categories to the ∞-categorical world: traditionally, one starts with the notion of a *cone* on a diagram. While this is easy for 1-categorical, i.e. "1-dimensional" diagrams, one already has to give the question some thought, how to define a cone on an ∞-categorical, i.e "higher dimensional" simplicial set shaped diagram. However, it is indeed not difficult to write down some explicit formulae that define the cone of a simplicial set [K, 0172, 0177], which in turn may be used to formulate what a cone on an ∞-diagram is.

Next, we would like to single out *universal* cones. In 1-category theory, those are then called limits or limit cones of the original diagram, and the universal property roughly says that *every* cone factors through such a limit cone. The latter property, though, cannot immediately be transferred to ∞-cones for several reasons: first, remember that we do not have unique compositions of morphisms (Remark 5.12), and second, one would have to take higher simplices into account.

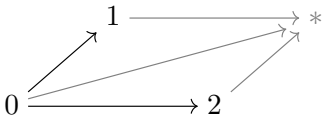
To solve this problem, one arranges for the ∞-cones on a given diagram to be the objects of a certain ∞-category. The latter then includes all the information about the higher morphisms and certain objects of this ∞-category will then be called universal ∞-cones.

All of this applies to cones, yielding *limits*, as well as dually to cocones, yielding *colimits*. In the first case, the cone point is the initial vertex of each simplex, in the second case the final vertex. For an expository account of the matter, skipping most technicalities yet providing all necessary intermediate steps, see Groth's notes [Grth, Sec. 2]. All details can be found in the literature [Ci, Sec. 6.2][Lu1, Sec. 1.2.13]. ⌟

Example 5.19 (pullbacks and pushouts)**.** As an informal example of the previous outlook, we say a bit more about pullback and pushout squares. For this, we consider diagrams indexed by the horns Λ^2_2 and Λ^2_0. The cone on Λ^2_2 and the cocone on Λ^2_0 are both isomorphic to $\mathrm{N}([1] \times [1])$; a fact that is intuitively clear by the following pictures:



cone on Λ^2_2



cocone on Λ^2_0,

i.e. both of them may index commutative squares (Example 5.15) in an ∞-category. Given a commutative square $p\colon \mathrm{N}([1] \times [1]) \to C$, we say that p is

– a *pullback square* if the cone on Λ^2_2 that it determines is universal and it is

– a *pushout square* if the cocone on Λ^2_0 that it determines is universal,

both in the sense of Outlook 5.18.

5.3. How to produce ∞-categories

The following method is, historically and because of its role in Lurie's book [Lu1], one of the most important constructions that convert 1-categorical input into a corresponding ∞-category, taking the homotopical structure into account:

Remark 5.20 (homotopy coherent nerve)**.** Let C be a *simplicially enriched category*: instead of just morphism sets, each $\mathrm{Mor}_C(X,Y)$ is a simplicial set and the usual requirements regarding composition are applied mutatis mutandis [K, 00JQ]. For such a category C, there is a construction analogous to the nerve of a 1-category (Definition 5.8), called the *homotopy coherent nerve* or *simplicial nerve*, that builds a simplicial set $\mathrm{N}^{\mathsf{hc}}(C)$ out of C [Lu1, Def. 1.1.5.5][DS2, Sec. 2.4]. Under suitable conditions, $\mathrm{N}^{\mathsf{hc}}(C)$ is an ∞-category [Lu1, Prop. 1.1.5.10], which is in some sense a "homotopical thickening" of the nerve $\mathrm{N}(C)$ of C.

To illustrate the last comment a bit more, we consider an example: Let C be a convenient category of topological spaces, e.g. compactly generated weak Hausdorff spaces. Here, the simplicial set $\mathrm{Mor}_C(X,Y)$ between two spaces X and Y is given by applying the singular simplicial set functor to the mapping space between X and Y. By easy inspection, one sees that the 2-simplices $\mathrm{N}(C)_2$ of the usual nerve correspond to commuting triangles in C, i.e. triples (f,g,h) of morphisms in C such that $h = g \circ f$. By contrast, $\mathrm{N}^{\mathsf{hc}}(C)_2$ will correspond to triangles commuting up to homotopy together with such a homotopy [K, 00KX], i.e. to quadruples (f,g,h,H) where f,g,h are morphisms in C and H is a homotopy from $g \circ f$ to h. Of course, we still have the strictly commuting triangles, but also a lot of other ones, which allows for more flexibility.

In certain situations, the homotopy coherent nerve can provide ∞-categorical versions of homotopy categories:

Remark 5.21 (homotopy categories)**.** Let C be a *simplicial model category*, i.e. a simplicially enriched category that is also a *model category* in a compatible way. The structure of a model category [Ci, Def. 2.2.1] that is subject to some conditions, of course, is the following: There is a distinguished class of morphisms W of C, called *weak equivalences*, that one is interested in from a homotopical point of view. Furthermore there are two distinguished classes of auxiliary morphisms, called *fibrations* and *cofibrations*.

There is a well-known construction that assigns to a model category C with weak equivalences W its *homotopy category*, which in turn presents itself as a model for the localization $C[W^{-1}]$, i.e. [Ci, Def. 2.2.8] the universal category obtained from C by making all morphisms in W isomorphisms. For example, looking at the category Top of topological spaces with W the class of homotopy equivalences, the corresponding homotopy category can be obtained from Top by modding out the "is homotopic to" equivalence relation on all morphism sets.

While 1-categorical homotopy categories are certainly useful, working with them tends to lose information in the sense that the homotopies that make diagrams commutative are not part of the data. They are merely required to exist for any choice of two parallel

morphisms, but not necessarily in a coherent way, see also Remark 5.16.

In the case of the simplicial model category C, this is remedied by an ∞-categorical enhancement of $C[W^{-1}]$: One takes a certain full subcategory C° of C (which inherits a simplicial enrichment from C) and forms $\mathrm{N}^{\mathsf{hc}}(C^\circ)$. This gives an ∞-category that is tightly linked to the homotopy category of C. (More specifically: one can also define what the homotopy category of an ∞-category is and if one applies this notion to $\mathrm{N}^{\mathsf{hc}}(C^\circ)$ the result will be a 1-category that is equivalent to $C[W^{-1}]$. This can be seen by combining several facts about homotopy categories [K, 00M4][Hi, Prop. 9.5.24 (2)][Hi, Sec. 7.5.6].)

However, while these constructions are certainly useful and applicable in some important cases, they require a lot of structure – which, in applications, is often not naturally available. For example, one might want to consider (model) categories where no extension to a *simplicial* model category exists or is known. And even if there is, the functors one would like to consider can most likely not easily be lifted to respect the simplicial enrichment. Thus, it is useful to have alternative approaches, specifically ones that allow to import a given 1-categorical situation, including functors, more directly. The following notion captures the minimal setting that is necessary to talk about homotopical situations:

Definition 5.22 (relative category)**.** A *relative category* (C, W) consists of

- a category C, together with
- a class of morphisms W of C, called *weak equivalences*.

A *relative functor*, also called a *homotopical functor*, from a relative category (C_1, W_1) to a relative category (C_2, W_2) is a functor $F\colon C_1 \to C_2$ that maps weak equivalences to weak equivalences, i.e. with $F(W_1) \subseteq W_2$. ⌟

Barwick and Kan have shown [BK] that relative categories can be used to model the homotopy theory of ∞-categories (see also Remark 5.25). Also note, that every model category yields an example of a relative category by simply forgetting the fibrations and cofibrations.

Similarly to Remark 5.21, one is usually not only interested in a 1-categorical localization $C[W^{-1}]$ of a relative category (C, W), but rather in a better behaved ∞-categorical version thereof. Classically, this was done by a process called *simplicial localization*, which was introduced and studied in a series of papers by Dwyer and Kan [DK1; DK2; DK3]. From this, one gets a simplicially enriched category – but not of the form to which Remark 5.20 applies directly. Instead of trying to rectify this, one can go straight to localizations of ∞-categories [Ci, Sec. 7.1]:

Remark 5.23 (localization)**.** Let (C, W') be a relative category. In terms of Definition 5.11, we can view W' as a subset of the morphisms of either C itself or its nerve $\mathrm{N}(C)$. Let W denote the smallest simplicial subset of $\mathrm{N}(C)$ that contains W'. One can then form the localization $W^{-1}\mathrm{N}(C)$, which is an ∞-category right away [Ci, Prop. 7.1.3]. (It will have [Ci, Rem. 7.1.6] the same link to $C[W^{-1}]$ as described at the end of Remark 5.21.)

5.4. Other names and other models

As the final act of this chapter, we should emphasize, that we merely presented one facet of the theory of ∞-categories. So as not to confuse the novice reader, we purposely refrained from mentioning this before; but since its complete concealment might lead to frustration when consulting the literature, we deem it necessary to inform the reader of the seemingly bizarre terminology landscape:

Remark 5.24 (other words for ∞-category)**.** In the literature, there are three terms for the *same* object, listed in reversed chronological order of appearance:

- ∞-category (Definition 5.4), as used by Lurie [Lu1],
- *quasi-category*, Joyal's terminology [Jo1][Jo2], and
- *weak Kan complex*, introduced by Boardman and Vogt [BV][Vo].

All of them are still commonly used, although it is customary to annotate the choice of "∞-category" with the phrase "in terms of Lurie" or similar, in order to avoid misunderstandings (see Remark 5.25). ⌟

Remark 5.25 (other things that are called ∞-category)**.** On the other hand, there are *different* objects that are also termed "∞-categories" in the literature:

- topologically enriched categories,
- simplicially enriched categories,
- Segal categories,
- complete Segal spaces,
- ...

This stems mainly from the fact, that during the evolution of higher category theory, the term "∞-category" was used to denote the somewhat vague concept of "category with infinitely many levels of morphisms". To make this notion precise, several people developed different approaches, which resulted (among others) in the objects listed above. Nowadays, it is common to denote the conceptual idea, which they are all a model of, by the term "$(\infty, 1)$-category". While a little longer than "∞-category", this helps in avoiding the ambiguity of the latter term. ⌟

For a more comprehensive account of the history and comparison between the different conceptual aspects and models, we refer to the literature [Be2][Be1][Jo2, Introduction and Sec. 2–5][Lu1, Ch. 1] and the *n*Lab [nL], starting at the articles "higher category theory", "(n,r)-category", and "quasi-category".

Chapter 6:

Excisive approximation

In this chapter, we investigate the (n-)excisive approximation of ℓ^1-homology – or more precisely, of the ℓ^1-chain complex functor. First, we introduce a concept of *excision* whose formulation applies to *any* functor between ∞-categories (Definition 6.1), we quote a theorem by Lurie ensuring the existence of such an approximation (Theorem 6.4), and we explain in what sense such an approximation is universal (Remark 6.5). We then use these notions to show that the excisive approximation of $C^{\ell^1} : \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$, i.e. the ℓ^1-chain complex functor on pointed spaces, is trivial (Theorem 6.7). An explicit formula by Lurie makes the argument short and accessible.

Afterwards, in Section 6.3, we consider n-excisive approximations of C^{ℓ^1}. Looking into the construction of the latter, we are able to prove that all of them vanish (Corollary 6.13).

As this chapter's setup is entirely ∞-categorical, the word "category" will always mean ∞-category in the sense of Lurie [Lu1] (Definition 5.4) and we will use 1-category to refer to categories in the sense of traditional category theory. Likewise, we will deal with other concepts such as functors, limits, etc., i.e. "functor" will mean a functor of ∞-categories, "(co)limit"/"pushout"/"pullback" will mean the ∞-categorical notion, etc.

The following ensures, that the ℓ^1-chain complex functor is well-defined in this setting:

Proposition 6.0. The 1-categorical ℓ^1-chain complex functor induces a functor of ∞-categories via the universal property of the localization (Remark 5.23).

Proof. It suffices to recognize the 1-categorical functor C^{ℓ^1} as a *relative* functor (Definition 5.22), i.e., mapping weak homotopy equivalences of topological spaces to quasi-isomorphisms of chain complexes. This follows from the fact that the bounded cochain complex functor C_b is a relative functor [Iv, 6.4 Cor.] and the translation principle by Löh [Lö2, Cor. 5.1]. Note, that Ivanov's result is only stated for maps between *path-connected* spaces, but his proof of the theorem preceding the corollary can be taken over verbatim if one drops the path-connectedness assumptions and replaces the "i.e."-part by the more general definition of "k-equivalence" involving all base points and π_0; see the literature [tD, Sec. 6.7][Ha, Ch. 4].

■

6.1. Abstract excision

First of all, we introduce the concept of an excisive functor of ∞-categories:

Definition 6.1 (excisive functor)**.** Let C be a category that admits pushouts and let $F\colon C \to D$ be a functor. Then F is *excisive* if it sends pushout squares to pullback squares, i.e., if the following holds: Given a pushout square

$$\begin{array}{ccc} W & \xrightarrow{f} & U \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle g'} \\ V & \xrightarrow{f'} & X \end{array}$$

in C, the diagram

$$\begin{array}{ccc} F(W) & \xrightarrow{F(f)} & F(U) \\ {\scriptstyle F(g)}\downarrow & & \downarrow{\scriptstyle F(g')} \\ F(V) & \xrightarrow{F(f')} & F(X) \end{array}$$

is a pullback square in D. ⌟

Note again, that this must be read in the ∞-categorical setting. In particular, pushout/pullback squares do *not* have to commute "on the nose" but only "up to homotopy" and the latter is part of the data(!), see also Example 5.15 and Remark 5.17.

Example 6.2. The constant functor $\mathsf{Top} \to \mathsf{Top}$ that sends everything to a point is excisive. On the other hand, the identity functor on Top is *not* excisive: for example, there is a pushout square

$$\begin{array}{ccc} S^0 & \xrightarrow{\text{inclusion}} & D^1 \\ \downarrow & & \downarrow{\scriptstyle \substack{\text{quotient} \\ \text{map}}} \\ * & \longrightarrow & D^1/S^0 \end{array}$$

that is *not* a pullback square. We will get back to this in Example A.7.

As it turns out, given a functor between categories with enough limits, it always has a "best approximation" by an excisive functor in a precise sense. The following definition captures the niceness assumptions that we have to impose on the codomain category:

Definition 6.3. Let C be a category. Then C is *differentiable* if it admits finite limits and sequential colimits and if the formation of the latter commutes with the former.

More explicitly, this means: every finite [K, 0130] diagram $K \to C$ admits a limit, every diagram $\mathrm{N}(\mathbb{N}) \to C$ admits a colimit in C, and the functor $\mathrm{colim}\colon \mathrm{Fun}(\mathrm{N}(\mathbb{N}), C) \to C$ commutes with finite limits. Here, $\mathbb{N}$ is viewed as the preorder category (Definition 5.1 (i)) of the usual ordering on the natural numbers and N is the nerve functor (Definition 5.8) from 1-categories to ∞-categories.

Theorem 6.4 ([Lu2, Thm. 6.1.1.10 and Ex. 6.1.1.28])**.** Let C be a category with a terminal object and finite colimits and let D be a differentiable category. Let $\mathrm{Exc}(C, D)$ be the full subcategory of the functor category $\mathrm{Fun}(C, D)$, spanned by the excisive functors. Then the inclusion functor $\mathrm{Exc}(C, D) \hookrightarrow \mathrm{Fun}(C, D)$ has a left adjoint

$$P_1 \colon \mathrm{Fun}(C, D) \to \mathrm{Exc}(C, D).$$

Furthermore, if $F \colon C \to D$ is *reduced*, which means that F maps every terminal object of C to a terminal object of D, and if D has a zero object, the following holds:

$$P_1 F \simeq \mathrm{colim}_{n \in \mathbb{N}} \Omega^n \circ F \circ \Sigma^n$$

where Ω and Σ are the loop and suspension functor on D and C, respectively. ⌟

See Lurie [Lu2, Rem. 1.1.2.6] for the construction of loop and suspension functors and Section B.1 for a short discussion of the former. For any functor $F \colon C \to D$, the functor $P_1 F \colon C \to D$ can be seen as a best approximation of F by an excisive functor from the right, as we shall now explain:

Remark 6.5 (best approximation through adjunction)**.** One way of making the term *best approximation* precise in a categorical setting is by so-called *universal arrows* [Mɪ2, Sec. III.1], i.e. by a certain mapping property:

Let C' be a full subcategory of some category C and let X be an object of C that we want to approximate by an object of C'. For example, C could be a functor category and C' could be its full subcategory of excisive functors. Then a *best approximation of X by an object of C' from the right* consists of an object Y of C' and a morphism $f \colon X \to Y$ in C with the following property: for every other morphism $f' \colon X \to Y'$ to an object of C', there exists an (up to homotopy) unique morphism $g \colon Y \to Y'$ such that

$$\begin{array}{ccc} X & \xrightarrow{f} & Y \\ & \searrow^{f'} & \downarrow{\scriptstyle g} \\ & & Y' \end{array}$$

commutes. Intuitively, we search for the "biggest quotient" of X that has the desired properties.

It is a classical 1-categorical fact [Mɪ2, Thm. IV.1.1] that an adjunction gives rise to universal arrows. We will sketch the argument for our situation: Assume that we have the inclusion functor $i \colon \mathrm{Exc} \hookrightarrow \mathrm{Fun}$ of a full subcategory and that it has a left adjoint $P \colon \mathrm{Fun} \to \mathrm{Exc}$. Using a mapping space functor (see Section B.2), one can define adjunctions in ∞-categories analogously to the 1-categorical case [Ci, Def. 6.1.3], i.e. we have an invertible natural transformation

$$\mathrm{Map}_{\mathrm{Fun}}(\bullet, i(\bullet)) \xrightarrow{c} \mathrm{Map}_{\mathrm{Exc}}(P(\bullet), \bullet).$$

Here, invertibility means, that each component of c is an invertible morphism [Ci, Def. 1.6.10 and 1.5.1], or equivalently, that c is an invertible morphism in the corresponding functor category [Ci, Cor. 3.5.12].

Now let F be an object of Fun, choose an inverse of c and let $\eta \coloneqq c^{-1}_{F,PF}(\mathrm{id}_{PF}) \in \mathrm{Map}_{\mathrm{Fun}}(F, iPF)$. In order to show that this morphism

$$F \xrightarrow{\eta} PF$$

is universal in the above sense, let P' be another object of Exc, let $(\eta' \colon F \to P') \in \mathrm{Map}_{\mathrm{Fun}}(F, iP')$ be a morphism and set $\theta \coloneqq c_{F,P'}(\eta')$. Then the triangle

$$\begin{array}{ccc} F & \xrightarrow{\eta} & PF \\ & \searrow^{\eta'} & \downarrow{\scriptstyle \theta} \\ & & P' \end{array}$$

commutes, which can be seen by chasing η through the following diagram:

$$\begin{array}{ccc} \mathrm{Map}_{\mathrm{Fun}}(F, PF) & \xrightarrow{\mathrm{Map}_{\mathrm{Fun}}(\mathrm{id}_F, \theta)} & \mathrm{Map}_{\mathrm{Fun}}(F, PF) \\ \downarrow{\scriptstyle c_{F,PF}} & & \downarrow{\scriptstyle c_{F,P'}} \\ \mathrm{Map}_{\mathrm{Exc}}(PF, PF) & \xrightarrow{\mathrm{Map}_{\mathrm{Exc}}(\mathrm{id}_{PF}, \theta)} & \mathrm{Map}_{\mathrm{Exc}}(PF, P'). \end{array}$$

⌟

Remark 6.6. As one might already have noticed, we only investigate approximations *from the right*. But in the first part of Remark 6.5, nothing prevents us from looking instead at morphisms *into* the object that we want to approximate. Intuitively, this corresponds then to a "biggest subobject" with the desired properties.

It is shown by Raptis [Rap], that the best approximation of the bounded cochain complex functor *from the right* is exactly the comparison map from the bounded to the singular cochain complex [Rap, Sec. 2.5], and that the arguments dualize to the comparison map in the case of ℓ^1-homology [Rap, Sec. 3.3], thus identifying the best approximation of the ℓ^1-chain complex functor *from the left*. ⌟

6.2. Excisive approximation of the ℓ^1-chain complex functor

We can now apply Theorem 6.4 to ℓ^1-homology, but we have to be a little bit careful: After a bit of thought, it is quite apparent that the classical *excision axiom* for a homology theory in the sense of Eilenberg and Steenrod is *not* a property of a single functor H_n for a fixed n, but of a whole sequence of functors: after all, we know [Sw, cf. Ch. 7, esp. 7.34 and 7.35] that excision can equivalently be recast in the form of a Mayer-Vietoris sequence running through all "dimensions" of a homology theory. (In the usual form of the Eilenberg-Steenrod axioms the excision axiom seems to depend only on a single functor at a time, but this is deceptive because it uses *relative* homology which is tightly linked to its absolute version in different(!) dimensions by the long exact sequence of a pair.)

So the upshot of this discussion is that we should not try to approach each of the functors $H_n^{\ell^1}$ individually, but instead we have to look at the corresponding functor $\mathsf{Top} \to \mathsf{Ch}_{\mathbb{R}}$ to chain complexes. We will see later (Appendix A) that indeed the usual singular chain complex functor *is* excisive in our sense. In order to apply the formula of Theorem 6.4, we need a reduced functor, but $C^{\ell^1}(*)$ is *not* a contractible chain complex. This is why we consider pointed spaces and the functor $C^{\ell^1} \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$. We then obtain:

Theorem 6.7 (excisive approximation of ℓ^1-homology)**.** There exists a best excisive approximation of $C^{\ell^1} \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ from the right, i.e. an excisive functor $P \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ and a natural transformation $\eta \colon C^{\ell^1} \to P$, such that for every other excisive functor $P' \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ and natural transformation $\eta' \colon C^{\ell^1} \to P'$ there exists a natural transformation $\theta \colon P \to P'$ (unique up to homotopy) that makes the triangle

$$\begin{array}{ccc} C^{\ell^1} & \xrightarrow{\eta} & P \\ & \searrow^{\eta'} & \downarrow{\scriptstyle \theta} \\ & & P' \end{array}$$

commutative. This best approximation is trivial, i.e. $P(X) \simeq 0$ for all pointed spaces X.

Proof. As Top_* is complete and cocomplete and $\mathsf{Ch}_{\mathbb{R}}$ is a stable [Lu2, Prop. 1.3.5.9 (with Prop. 1.3.5.15)] and cocomplete category, so in particular differentiable [Lu2, Ex. 6.1.1.7], the prerequisites of Theorem 6.4 are fulfilled. So the functor $P_1 \colon \mathrm{Fun}(\mathsf{Top}_*, \mathsf{Ch}_{\mathbb{R}}) \to \mathrm{Exc}(\mathsf{Top}_*, \mathsf{Ch}_{\mathbb{R}})$ and in particular $P_1(C^{\ell^1}) \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ exist. Since P_1 is left adjoint to the inclusion functor $i \colon \mathrm{Exc}(\mathsf{Top}_*, \mathsf{Ch}_{\mathbb{R}}) \hookrightarrow \mathrm{Fun}(\mathsf{Top}_*, \mathsf{Ch}_{\mathbb{R}})$, we also know (see Remark 6.5) that the unit morphism $F = \mathrm{Id}(F) \to (i \circ P_1)(F)$ of the adjunction provides a universal arrow in the sense described in the claim.

Furthermore, since we consider C^{ℓ^1} relative to the base point, the second part of Theorem 6.4 is applicable. For a pointed space X this gives us

$$P_1(C^{\ell^1})(X) \simeq \mathrm{colim}_n \, \Omega^n\big(C^{\ell^1}(\Sigma^n X)\big),$$

where $\Sigma \colon \mathsf{Top}_* \to \mathsf{Top}_*$ is the (unreduced) suspension functor and $\Omega \colon \mathsf{Ch}_{\mathbb{R}} \to \mathsf{Ch}_{\mathbb{R}}$ is the loop functor on chain complexes. But since $\Sigma^n X$ is simply-connected for $n \in \mathbb{N}_{\geq 2}$ and we know that C^{ℓ^1} vanishes on path-connected spaces with amenable (so in particular trivial) fundamental group [Iv, 8.4 Thm.][Lö2, Cor. 5.1], we get $P_1(C^{\ell^1})(X) \simeq \mathrm{colim}_{n \in \mathbb{N}_{\geq 2}} \Omega^n(0)$. Consider the fact that the loop functor Ω on chain complexes can be realized by just shifting a complex down by one (i.e. $(\Omega C)_n = C_{n+1}$). Using this, it follows that we have

$$P_1(C^{\ell^1})(X) \simeq \mathrm{colim}_{n \in \mathbb{N}_{\geq 2}} 0 \simeq 0.$$

■

6.3. Generalization to n-excisive approximation

The property of being excisive (Definition 6.1) is in fact only the first one of a whole sequence, namely so-called *n-excisive functors*. Whereas (1-)excisive functors act in a nice way on square shaped diagrams, an n-excisive functor satisfies a higher dimensional analogue involving $(n+1)$-dimensional cubical diagrams. In order to give a precise definition, we have to introduce some further notation:

Definition 6.8 (n-cube)**.** Let C be a category, let $n \in \mathbb{N}$, and for a set A let $\mathcal{P}(A)$ denote the preorder category (Definition 5.1 (i)) of the powerset of A with respect to inclusion, i.e.

$$\mathcal{P}(A) \coloneqq \mathrm{Pre}\big(\{Z \mid Z \subseteq A\}, \subseteq\big),$$

and let $\mathcal{P}_*(A)$ be its full subcategory on all non-empty sets.

(i) An *n-cube in C* is a diagram

$$\mathrm{N}\big(\mathcal{P}(\{0,\dots,n-1\})\big) \to C.$$

(ii) Let X be an n-cube in C and let $T, T' \subseteq \{0,\dots,n-1\}$. Then the restriction of X to $T \cap T', T, T', T \cup T'$ determines a square

$$\begin{array}{ccc} X(T \cap T') & \longrightarrow & X(T) \\ \downarrow & & \downarrow \\ X(T') & \longrightarrow & X(T \cup T') \end{array}$$

in C. A square obtained in this way is a *2-dimensional face of X*.

(iii) Let X be an n-cube. Then $\emptyset$ is an initial object in $\mathcal{P}(\{0,\dots,n-1\})$, so we can identify X with a cone on its restriction to $\mathrm{N}(\mathcal{P}_*(\{0,\dots,n-1\}))$. If this cone is a limit cone (Outlook 5.18), X is *Cartesian*.

(iv) An n-cube is *strongly coCartesian* if all of its 2-dimensional faces are pushout squares. ⌟

More details about such cubical diagrams can be found in Lurie's book [Lu2, Sec. 6.1.1]. (He also proves [Lu2, Prop. 6.1.1.15] that his definition of strongly coCartesian cubes is equivalent to the one given above.)

Remark 6.9. Cartesian 2-cubes can precisely be identified with pullback squares and, similarly, strongly coCartesian 2-cubes with pushout squares.

In view of this remark, excisive functors (Definition 6.1) are a special case, for $n = 1$, of the following:

Definition 6.10 (n-excisive functor)**.** Let C be a category with all finite colimits, let $F\colon C \to D$ be a functor and let $n \in \mathbb{N}$. Then F is *n-excisive* if it sends strongly coCartesian $(n+1)$-cubes in C to Cartesian $(n+1)$-cubes in D.

Analogous to the case of (1-)excisive functors, there will be a best n-excisive approximation to a given functor if the categories are nice enough (compare Theorem 6.4 and Remark 6.5):

Theorem 6.11 ([Lu2, Thm. 6.1.1.10])**.** Let C be a category with a terminal object and finite colimits and let D be a differentiable category (Definition 6.3). Let $\mathrm{Exc}^n(C,D)$ be the full subcategory of the functor category $\mathrm{Fun}(C,D)$, spanned by the n-excisive functors. Then the inclusion functor $\mathrm{Exc}^n(C,D) \hookrightarrow \mathrm{Fun}(C,D)$ has a left adjoint $P_n\colon \mathrm{Fun}(C,D) \to \mathrm{Exc}^n(C,D)$.

It should be noted, that being $(n+1)$-excisive is a *weaker* condition on a functor than being n-excisive. More precisely, in the situation of Theorem 6.11, we have $\mathrm{Exc}^n(C,D) \subseteq \mathrm{Exc}^m(C,D)$ for all $m \in \mathbb{N}_{\geq n}$ [Lu2, Cor. 6.1.1.14]. Thus, universality of P_n (Remark 6.5) leads to a diagram of the form

$$\cdots \to P_2F \to P_1F \to P_0F,$$

called the *Taylor tower of F* [Lu2, Sec. 6.1.2]. In general, one might seek to obtain information about F from the more and more accurate approximations in this tower – however, we now show that the Taylor tower does *not* help to study the ℓ^1-chain complex functor. In fact, the tower will always be trivial for a functor that vanishes on all highly connected spaces:

Theorem 6.12 (Taylor tower invisibility)**.** Let D be a differentiable category with terminal object $*$ and let F be a functor $\mathsf{Top}_* \to D$. Let $k \in \mathbb{N}$ and suppose that F vanishes on all k-connected pointed spaces, i.e. for all k-connected pointed spaces X, we have $F(X) \simeq *$. Then F is *Taylor tower invisible*, meaning that

$$(P_nF)(X) \simeq *$$

holds for all $n \in \mathbb{N}$ and all pointed spaces X.

The same statement, removing all "pointed", holds for functors $F\colon \mathsf{Top} \to D$. ⌟

Proof. Since the argument is exactly the same, "space" will either mean pointed or unpointed space in this proof, depending on F.

Fix $n \in \mathbb{N}$. To obtain the result, we will have to unwrap enough details of the construction of the n-excisive approximation functor P_n. The first step in Lurie's construction [Lu2, Constr. 6.1.1.18] is the *S-pointed cone functor* $(X,S) \mapsto C_S(X)$, which takes a space and a finite set and produces again a space. We observe that it is chosen as a section to a particular trivial Kan fibration and that the property [K, 006Y] that provides

such a section additionally allows us to prescribe the functor $S \mapsto C_S(X)$ as $S \mapsto X * S$ for all X; here, $X * S$ is the *join of X and S*, where the finite set S is viewed as a discrete space, and where the join inherits the basepoint of X if necessary.

Next, the functor $T_n \colon \mathrm{Fun}(\mathsf{Top}_{(*)}, D) \to \mathrm{Fun}(\mathsf{Top}_{(*)}, D)$ is defined, where

$$(T_n F)(X) \;=\; \lim F \circ C_\bullet(X)|_{\mathcal{P}_*(\{0,\dots,n\})} \;=\; \lim_{\emptyset \neq S \subseteq \{0,\dots,n\}} F(X * S).$$

The functor P_n is then defined as a sequential colimit over subsequent applications of T_n. As such, it is hardly surprising, that one finds $P_n(F) \simeq P_n(T_n F)$ [Lu2, Lem. 6.1.1.34].

Now let X be a space. In view of the last paragraph, it suffices to show our claim for $T_n^{k+2}F$ instead of F. Applying the above formula multiple times, we see that the value $(T_n^{k+2}F)(X)$ is given by some limits over objects of the form

$$F(X * S_0 * \cdots * S_{k+1})$$

for subsets $S_0, \dots, S_{k+1} \subseteq \{0, \dots, n\}$. But since $S_0, \dots, S_{k+1}$ are all non-empty, the argument of F will be k-connected, and thus F of it vanishes by assumption. Similarly, we have

$$(T_n^\ell F)(X) \simeq * \quad \text{for all } \ell \in \mathbb{N}_{\geq k+2}.$$

Finally, by the above and the definition of P_n, we obtain

$$\begin{aligned}(P_n F)(X) &\simeq (P_n T_n^{k+2} F)(X) \\ &= \mathrm{colim}\big((T_n^{k+2}F)(X) \to (T_n^{k+3}F)(X) \to \dots\big) \\ &\simeq \mathrm{colim}(* \to * \to \dots) \\ &\simeq *.\end{aligned}$$

■

Since the ℓ^1-chain complex functor satisfies the assumptions of the theorem, we obtain the following consequence:

Corollary 6.13 (n-excisive approximation of ℓ^1-homology)**.** Let $n \in \mathbb{N}$. A best n-excisive approximation of $C^{\ell^1} \colon \mathsf{Top}_* \to \mathsf{Ch}_{\mathbb{R}}$ from the right exists and is trivial.

Appendix A:

Excision for singular homology

Let us consider singular homology on Top. It satisfies classical excision, e.g. in the sense of having Mayer-Vietoris sequences. As discussed before (Section 6.2), this should rather be seen as a property of the underlying chain complex functor $C\colon \mathsf{Top} \to \mathsf{Ch}$, and indeed formal arguments then can be used to derive Mayer-Vietoris sequences from this fact, see Appendix B. In this chapter we will show that C really is excisive in the sense of Definition 6.1, i.e. that the latter notion is a generalization of the classical property.

Theorem A.1 (C is excisive)**.** The ∞-categorical singular chain complex functor $C\colon \mathsf{Top} \to \mathsf{Ch}$ is excisive.

As this is often used as an example or motivation for the definition of an excisive functor, it seems to be an easy fact for homotopy theorists, but an explicit proof seems to be hard to find in the literature, so we will include one here.

A.1. Model category theoretical bits

To prove the main theorem of this chapter, we will use the formalism of model categories and we will show the following more explicit version:

Proposition A.2. Homotopy pushout squares in Top are sent to homotopy pullback squares in Ch by the singular chain complex functor.

Here, we consider the *classical* or *Quillen model structure* on Top [Ho, Sec. 2.4] and the *projective model structure* on Ch [Ho, Sec. 2.3]. Familiarity with those is, however, *not* required to follow the logic of this chapter.

We will now give an ad hoc definition of homotopy pushouts in Top, which can be thought of as "thick pushouts", as witnessed by the *double mapping cylinder* construction in the below definition. As we shall see shortly, we will actually concern ourselves only with homotopy pushouts of a certain form, but using that as our definition would hide the essence of this homotopy theoretic notion entirely. Thus we have [Hi, Def. 15.5.8][MV, Def. 3.7.1]:

Definition A.3 (homotopy pushout (square) in Top)**.** A commutative square

$$\begin{array}{ccc} W & \xrightarrow{f} & U \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle p} \\ V & \xrightarrow{q} & X \end{array}$$

in Top is a *homotopy pushout (square)* if the composition

$$\big(U \sqcup (W \times [0,1]) \sqcup V\big)/\!\sim \quad \xrightarrow[(w,t)\mapsto f(w)\text{"}]{\text{"}u\mapsto u,\ v\mapsto v} \quad U \sqcup_{f,g} V \quad \xrightarrow[p\sqcup_{f,g} q]{} \quad X$$

is a weak equivalence; here

- the first space is the *double mapping cylinder* of f and g, where $\sim$ is the equivalence relation generated by $f(w) \sim (w, 0)$ and $(w, 1) \sim g(w)$ for $w \in W$.
- $U \sqcup_{f,g} V$ is the pushout $(U \sqcup V)/\!\approx$ where $\approx$ is generated by $f(w) \approx g(w)$ for $w \in W$.
- $p \sqcup_{f,g} q$ is the map induced by p and q by the universal property of the pushout. ⌟

Similarly, homotopy pullback squares can be defined in Ch. For the following, however, we will not need this explicitly.

Example A.4. There are homotopy pushouts

$$\begin{array}{ccc} \emptyset & \longrightarrow & U \\ \downarrow & & \downarrow \\ V & \longrightarrow & U \sqcup V \end{array} \qquad \text{and} \qquad \begin{array}{ccc} X & \longrightarrow & X \times [0,1] \\ \downarrow & & \downarrow \\ X & \longrightarrow & X \end{array} \qquad \text{and} \qquad \begin{array}{ccc} X & \longrightarrow & \mathrm{Cone}(X) \\ \downarrow & & \downarrow \\ \mathrm{Cone}(X) & \longrightarrow & \Sigma X \end{array}$$

for all topological spaces U, V, X (where $\mathrm{Cone}(X)$ is the *cone on* X and ΣX is the *suspension of* X).

Remark A.5 (philosophy behind homotopy (co)limits)**.** Generally speaking, homotopy (co)limits make up for the fact that ordinary (co)limits do *not* preserve the notion of weak equivalence. As a concrete example in Top, we consider the commutative diagram

$$\begin{array}{ccccc} * & \longleftarrow & S^0 & \xrightarrow[\text{inclusion}]{} & D^1 \\ \downarrow & & {\scriptstyle \mathrm{id}}\downarrow & & \downarrow \\ * & \longleftarrow & S^0 & \longrightarrow & * \end{array}$$

and note, that even though all vertical maps are (weak) homotopy equivalences, the map $D^1/S^0 \to *$ that they induce on the ordinary pushouts of the rows is *not* a (weak) homotopy equivalence.

Moreover, homotopy (co)limits in a model category are strongly connected to ∞-categorical (co)limits in the associated ∞-category, i.e. the localization (Remark 5.23) at

weak equivalences. For example: the latter compute the former [Ci, Rem. 7.9.10], (certain) homotopy pushout squares are sent to pushout squares in the ∞-category by the localization functor [Ci, dual of Cor. 7.5.20], and in the setting of simplicially enriched categories (see Remark 5.20) a theorem by Lurie compares both notions [Lu1, Thm. 4.2.4.1]. ⌟

Remark A.6 (existence and uniqueness of homotopy pushouts)**.** Given the solid part of the diagram

$$\begin{array}{ccc} W & \xrightarrow[f]{} & U \\ {\scriptstyle g}\downarrow & & \vdots \\ V & \dashrightarrow & ?, \end{array}$$

one can always extend it to an ordinary pushout square, e.g. with $? = U \sqcup_{f,g} V$ (in the notation of Definition A.3). However, it may happen, that there is *no* extension to a commutative homotopy pushout square! For example, assume that there is a homotopy pushout

$$\begin{array}{ccc} S^0 & \longrightarrow & * \\ \downarrow & & \downarrow \\ * & \longrightarrow & X \end{array}$$

and denote its double mapping cylinder by Z. Then the weak equivalence $Z \to X$ factors through $*$ by definition, but this is impossible as Z is homeomorphic to S^1.

On the other hand, if there are two homotopy pushouts

$$\begin{array}{ccc} W & \xrightarrow[f]{} & U \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle p} \\ V & \xrightarrow{q} & X \end{array} \qquad \text{and} \qquad \begin{array}{ccc} W & \xrightarrow[f]{} & U \\ {\scriptstyle g}\downarrow & & \downarrow{\scriptstyle p'} \\ V & \xrightarrow{q'} & X' \end{array}$$

with the same W, U, V, f, g, the spaces X and X' must be in the same equivalence class that is generated by weak equivalences, because the double mapping cylinder of f and g is weakly equivalent to both of them. One may thus call (the weak equivalence class of) the double mapping cylinder the *homotopy pushout of the diagram*

$$V \xleftarrow{g} W \xrightarrow{f} U\,,$$

and this is well-defined, even if there is no extension to a homotopy pushout square. ⌟

Example A.7. In Example 6.2 we claimed that there is an ∞-categorical pushout

$$\begin{array}{ccc} S^0 & \xrightarrow{i} & D^1 \\ \downarrow & & {\scriptstyle q}\downarrow \\ * & \longrightarrow & D^1/S^0 \end{array} \qquad (\square)$$

in Top, where i is the inclusion and q the quotient map, that is *not* a pullback. Using

the language of homotopy pushouts/pullbacks and the references given in Remark A.5, we can now give a proof of this:

The given square (□) is a homotopy pushout and thus sent to an ∞-pushout by the localization functor. However, it cannot be an ∞-pullback: Dually to Remark A.6, there is a well-defined weak equivalence class of spaces that one can call *homotopy pullback of the diagram*

$$* \longrightarrow D^1/S^0 \xleftarrow{q} D^1 .$$

Now if we assume, for a contradiction, that (□) is an ∞-pullback, the assertion that ∞-pullbacks compute homotopy pullbacks just means, that S^0 is in this weak equivalence class. We will now show that also the loop space ΩS^1, at an arbitrary base point $e \in S^1$, is in this equivalence class, which cannot both be true, as we would get

$$2 = \#\pi_0(S^0) = \#\pi_0(\Omega S^1) = \#\pi_1(S^1) = \#\mathbb{Z}$$

as cardinal numbers. To that end, consider the following diagram:

$$\begin{array}{ccccc} * & \longrightarrow & D^1/S^0 & \xleftarrow{q} & D^1 \\ \uparrow & & \uparrow & & \uparrow \\ * & \xrightarrow{e} & S^1 & \xleftarrow{e} & * . \end{array}$$

We can certainly find vertical maps such that they are all weak equivalences and such that both squares commute. Now because the homotopy pullback of diagrams *does* respect such a "weak equivalence of diagrams" [Hi, Prop. 13.3.4] (compare Remark A.5 for ordinary limits), the homotopy pullbacks of the rows yield the same weak equivalence class. Using an explicit definition of homotopy pullback [MV, Def. 3.2.4] it is then easy to see, that the homotopy pullback of the diagram in the bottom row is given by ΩS^1 [MV, Ex. 3.2.10]. ⌟

In the rest of this chapter, we will give a proof of Proposition A.2 and then show how this implies Theorem A.1.

A.2. Proof of the model categorical statement

First of all, we may reduce our attention to homotopy pushouts of a specific type by virtue of the following three facts:

Proposition A.8 ([MV, Prop. 3.7.4 and Ex. 3.7.5])**.** Let S' be a homotopy pushout square in Top. Then there exists a homotopy pushout square

$$S = \begin{pmatrix} W & \longrightarrow & U \\ \downarrow & & \downarrow \\ V & \longrightarrow & X \end{pmatrix}$$

and a map of squares $\eta\colon S \to S'$, such that

- U and V are open subsets of X and $W = U \cap V$,
- all maps in S are inclusions, and
- η is component-wise a weak equivalence.

Furthermore, any square S as above is a homotopy pushout square. ⌟

Proposition A.9 ([Ha, Prop. 4.21])**.** The singular chain complex functor is *homotopical*, i.e. it sends weak equivalences in Top to weak equivalences in Ch.

Proposition A.10. Being a homotopy pullback square in Ch is invariant under weak equivalences of squares in the following sense: Given two squares S and S' in Ch and a map of diagrams $S \to S'$ that is component-wise a weak equivalence, then S is a homotopy pullback if and only if S' is.

Proof. This is Proposition 13.3.13 in Hirschhorn's book [Hi]. Its prerequisites are fulfilled because every object in Ch is fibrant [Ho, Thm. 2.3.11 and subsequent paragraph] and thus Ch is right proper [Hi, Cor. 13.1.3].

■

With this, we are left to show:

Proposition A.11. Let S be as in Proposition A.8. Then the induced square

$$\begin{array}{ccc} C(W) & \longrightarrow & C(U) \\ \downarrow & & \downarrow \\ C(V) & \longrightarrow & C(X) \end{array}$$

is a homotopy pullback in Ch.

Proof. Let $C(S)$ denote the square from the claim and let $\iota\colon C^{U,V}(X) \hookrightarrow C(X)$ be the inclusion of the subcomplex generated by the singular simplices that are contained in U or V. Then it is well known [Ha, Prop. 2.21] that ι is a weak equivalence (this is the core of the proof of classical excision statements). Now we consider the square

$$\begin{pmatrix} C(W) & \xrightarrow{i_{WU}} & C(U) \\ \downarrow{\scriptstyle i_{WV}} & & {\scriptstyle i_{UX}}\downarrow \\ C(V) & \xrightarrow{i_{VX}} & C^{U,V}(X) \end{pmatrix} =: S^{U,V}$$

where all morphisms denote the canonical injections and the component-wise inclusion $S^{U,V} \to C(S)$, which is component-wise a weak equivalence. Hence, Proposition A.10 applies and we can reduce to the problem of showing that $S^{U,V}$ is a homotopy pullback.

It is known, that Ch is a *stable* model category [Ho, Sec. 7.1], so we may equivalently [Ho, Rem. 7.1.12] show that $S^{U,V}$ is a homotopy *pushout*. But in the model structure on Ch, all objects in $S^{U,V}$ are cofibrant [Ho, Lem. 2.3.6] and the morphism i_{WU} is a cofibration by Lemma A.12 (see below), hence it suffices [RV, Ex. 8.8] to show that $S^{U,V}$ is a pushout. It is easily seen that this is the case if (and only if)

$$C(W) \xrightarrow{(i_{WU}, i_{WV})} C(U) \oplus C(V) \xrightarrow{i_{UX} - i_{VX}} C^{U,V}(X) \longrightarrow 0$$

is an exact sequence in Ch. But this is true, basically by construction of $C^{U,V}(X)$, so we are done.

■

Lemma A.12 (C of injection)**.** Let $i\colon A \to X$ be an injective map in Top. Then $C(i)\colon C(A) \to C(X)$ is a cofibration in Ch.

Proof. By definition of the singular chain complex functor, $C(A)$ and $C(X)$ are free modules, the chain map $C(i)$ is dimensionwise injective and maps a basis of $C(A)$ to a subset of a basis of $C(X)$. Thus, $C(i)$ is dimensionwise a split injection with free cokernel, hence a cofibration in Ch [Ho, Prop. 2.3.9 and Lem. 2.3.6].

■

This finishes the proof of Proposition A.2. In the last section we promote this model categorical result to the ∞-categorical setting.

A.3. Proof of the ∞-categorical statement

Proof that Proposition A.2 implies Theorem A.1. To make this proof more transparent, let us fix the following notation: Top denotes the 1-category, Top_∞ the corresponding ∞-category, i.e. the localization (Remark 5.23) at weak homotopy equivalences, similarly, Ch denotes the 1-category, Ch_∞ the ∞-category, i.e. the localization at quasi-isomorphisms, and $C\colon \mathsf{Top} \to \mathsf{Ch}$ is the 1-categorical functor, while $C_\infty\colon \mathsf{Top}_\infty \to \mathsf{Ch}_\infty$ is the induced functor on ∞-categories (which is well-defined since C is homotopical, Proposition A.9).

Both, Ch and Ch_∞ are *stable*, Ch as a model category [Ho, Sec. 7.1] and Ch_∞ as ∞-category [Lu2, Prop. 1.3.5.9 (with Prop. 1.3.5.15)]. In particular, a square in Ch is a homotopy pullback if and only if it is a homotopy pushout [Ho, Rem. 7.1.12], and a square in Ch_∞ is a pullback if and only if it is a pushout [Lu2, Prop. 1.1.3.4].

With this, we see that C_∞ being excisive is equivalent to C_∞ commuting with pushouts. Hence, it suffices to show that C_∞ commutes with *all* finite colimits. This follows from the dual of Proposition 7.5.28 in Cisinski's book [Ci], applied to the composition $\bar{C}\colon \mathrm{N}(\mathsf{Top}) \to \mathsf{Ch}_\infty$ of $\mathrm{N}(C)$ and the localization functor $\mathrm{N}(\mathsf{Ch}) \to \mathsf{Ch}_\infty$; for this, Ch_∞ is made an ∞-category with weak equivalences and cofibrations in a trivial way [Ci, Ex. 7.5.4]. To apply the proposition, we just need to show that $\bar{C}$ is a *right exact* [Ci, Def. 7.5.2] functor:

(i) As $C(\emptyset) \cong 0$, initial objects are preserved by $\bar{C}$.

(ii) All morphisms in Ch_∞ are cofibrations and C is homotopical (Proposition A.9). Thus, $\bar{C}$ preserves cofibrations and trivial cofibrations.

(iii) Lastly, we need to check that pushout squares (in Top) of the form

$$\begin{array}{ccc} A & \xrightarrow[i]{} & X \\ \downarrow & & \downarrow \\ A' & \longrightarrow & X' \end{array}$$

with i a cofibration and A and A' (and thus also X) cofibrant are sent to pushout squares by $\bar{C}$. But these conditions ensure [RV, Ex. 8.8] that such a square is also a homotopy pushout square, which is preserved by C by combining Proposition A.2 and the comment about stability of Ch above. Furthermore, by the first item and Lemma A.12, C also preserves the additional properties of i, A and A', so the square induced by C is sent to a pushout square in Ch_∞ [Ci, dual of Cor. 7.5.20]. ■

Appendix B:

Relation between abstract and classical excision

In this chapter, we relate abstract excision (Definition 6.1) to classical excision in the form of Mayer-Vietoris sequences. To this end, we will explain why and to some extent how an excisive functor always admits Mayer-Vietoris sequences, coinciding with the usual one in the case of singular homology, for example.

If our excisive functor is defined on Top, the first thing to note is that by the last part of Proposition A.8 we may indeed apply the abstract excision property in the setting $X = U \cup V$ with U and V open in X.

This leaves us with the task of explaining how a pullback square in an ∞-category leads to a long exact sequence. To this end, the first step is the following:

B.1. Obtaining a fiber sequence

Let C be an ∞-category with finite limits and a zero object, i.e. an object $*$ of C that is both initial and terminal. Suppose we start with the following pullback in C:

$$\begin{array}{ccc} W & \longrightarrow & U \\ \downarrow & & \downarrow \\ V & \longrightarrow & X. \end{array}$$

Then from this, one can derive a fiber sequence like this:

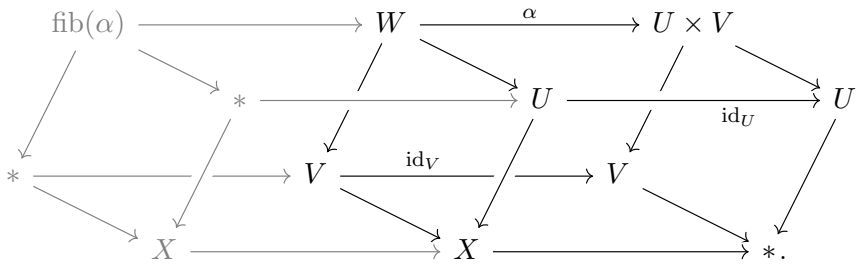
$$\Omega X \to W \to U \times V, \tag{B.1}$$

where Ω denotes the loop functor. Here, being a fiber sequence means (by definition) being a pullback square

$$\begin{array}{ccc} \Omega X & \longrightarrow & W \\ \downarrow & & \downarrow \\ * & \longrightarrow & U \times V \end{array}$$

and the loop functor $\Omega\colon C \to C$ assigns to an object X the pullback of $* \to X \leftarrow *$, or in other words it completes $* \to X$ to a fiber sequence $\Omega X \to * \to X$. Lurie describes in detail how to construct loop and suspension functors [Lu2, paragraph preceeding Rem. 1.1.2.6].

For a precise treatment of the passage from pullback to fiber sequence, we would need more technicalities about limits in ∞-categories. Informally, though, the basic ideas are as follows: We have two pullback squares and a morphism $\alpha\colon W \to U \times V$ induced by the universal property of the product as in the right part of this diagram:



Taking fibers (also a limit!) in the horizontal direction and using the fact that limits commute with limits [Lu1, dual of Lem. 5.5.2.3], as is the case in 1-categories, we obtain that the left square is also a pullback. Hence, by definition of the loop functor, $\mathrm{fib}(\alpha)$ is (equivalent to) ΩX.

B.2. Applying the mapping space functor

Let C be a 1-category. A quite important notion in 1-category theory is the functor

$$\mathrm{Mor}_C\colon C^{\mathsf{op}} \times C \to \mathsf{Set}$$

that takes a pair (x, y) of objects of C to the set of morphisms $\mathrm{Mor}_C(x, y)$. In the literature, this is often called "Hom-functor".

Now let C be an ∞-category. One slogan of ∞-category theory is: the role of Set in 1-category theory is taken by Top in ∞-category theory. This motivates the desire to find, for a pair of objects (x, y) of C, not only a set of morphisms $x \to y$, but instead a *space* of such morphisms – and indeed, this is always possible. The following remark indicates how, but its point is rather to illustrate the analogy between the 1- and ∞-categorical situation than to provide the reader with a better understanding of the space itself. Thus, it can also safely be skipped.

Remark B.1 (construction of mapping spaces)**.** We consider the 1-categorical situation first. Here, the set $\mathrm{Mor}_C(x, y)$ can be found as a pullback of "sets" as in the following pullback square:

$$\begin{array}{ccc} \mathrm{Mor}_C(x,y) & \longrightarrow & \mathrm{Mor}(C) \\ \downarrow & & \downarrow{\scriptstyle (f:a\to b)\mapsto(a,b)} \\ \{0\} & \xrightarrow{0\mapsto(x,y)} & \mathrm{Ob}(C)\times\mathrm{Ob}(C) \end{array}$$

where $\mathrm{Mor}(C)$ and $\mathrm{Ob}(C)$ denote the set (or class) of morphisms and objects of C, respectively.

Now if C is instead an ∞-category, we can form the analogous diagram

$$\begin{array}{ccc} & & C^{\Delta^1} \\ & & \downarrow{\scriptstyle ((d_1^0)^*,(d_0^0)^*)} \\ \Delta^0 & \xrightarrow{*_{(x,y)}} & C^{\Delta^0} \times C^{\Delta^0} \end{array}$$

in sSet and *define* the space of morphisms from x to y as (the geometric realization of) its pullback; here we use the following notation:

- C^{Δ^n} is the *internal Hom* simplicial set as in Remark 5.6,
- $(d_i^0)^* \colon C^{\Delta^1} \to C^{\Delta^0}$ is the morphism induced by the i-th coface map d_i^0, and
- $*_{(x,y)}$ denotes the unique morphism with

  $$(\Delta^0)_0 \ni \mathrm{Id}_{[0]} \mapsto (x,y) \in C_0 \times C_0$$

  under the identification $C_0 \times C_0 \cong \mathrm{Mor}_{\mathsf{sSet}}(\Delta^0, C) \times \mathrm{Mor}_{\mathsf{sSet}}(\Delta^0, C) = \left(C^{\Delta^0} \times C^{\Delta^0}\right)_0$ from Example 5.14.

Dugger and Spivak [DS1] discuss multiple ways to construct mapping spaces and in particular they consider the construction above [DS1, Prop. 1.2]. ⌟

If properly organized [Ci, Sec. 5.8], one can even define a functor of ∞-categories

$$\mathrm{Map}_C \colon C^{\mathsf{op}} \times C \to \mathsf{Top},$$

which we call Map_C to distinguish it from the 1-categorical concept. (Note, however, that $\mathrm{Map}_C(x,y)$ will in general only be weakly equivalent to the space that we constructed in Remark B.1. Also, in Cisinski's book the target of the mapping space functor is "$\mathcal{S}$", so to get one to Top, we have to compose with an equivalence between those two ∞-categories [Ci, Rem. 7.8.10 and Rem. 7.8.11].) For this functor one can show:

Proposition B.2 ([Ci, Cor. 6.3.5])**.** Let C be an ∞-category and let x be an object of C. Then the functor $\mathrm{Map}_C(x, \bullet) \colon C \to \mathsf{Top}$ preserves limits.

In particular, since these are defined via pullbacks, $\mathrm{Map}_C(x, \bullet)$ maps fiber sequences to fiber sequences. This gives us a method to convert the fiber sequence (B.1) from Section B.1 to one in Top: for any test object S of C we obtain

$$\mathrm{Map}_C(S, \Omega X) \to \mathrm{Map}_C(S, W) \to \mathrm{Map}_C(S, U \times V), \tag{B.2}$$

a fiber sequence of spaces, and canonically also of pointed spaces by virtue of zero maps.

B.3. The Mayer-Vietoris sequence

As is well known, a fiber sequence $F \to X \to Y$ of pointed spaces always yields a long exact sequence of homotopy groups:

$$\cdots \to \pi_n(F) \to \pi_n(X) \to \pi_n(Y) \to \pi_{n-1}(F) \to \cdots .$$

Using this on the fiber sequence (B.2) from the last section, again the fact that $\mathrm{Map}_C(S, \bullet)$ preserves limits (Proposition B.2), and well known facts about the homotopy groups, we get the following exact sequence:

$$\begin{aligned} \cdots \to \pi_{n+1} \mathrm{Map}_C(S, X) &\to \pi_n \mathrm{Map}_C(S, W) \\ &\to \pi_n \mathrm{Map}_C(S, U) \times \pi_n \mathrm{Map}_C(S, V) \to \pi_n \mathrm{Map}_C(S, X) \to \cdots . \end{aligned} \tag{B.3}$$

In this sense, one can always extract Mayer-Vietoris type sequences from a pullback square.

B.4. Example: singular homology

We shall now explain, how the long exact sequence (B.3) from Section B.3 about homotopy groups also yields the familiar Mayer-Vietoris sequence of *homology* groups. For this we have to use the "correct" object S plus some additional arguments.

Let us first recapitulate the situation: We have a space $X = U \cup V$ with U and V open in X, $W = U \cap V$, and C the singular chain complex functor. From the pushout formed by $U \cap V$, U, V and X, we get the according pullback of chain complexes. We now apply the previous material, where for the test complex S we choose the complex $\mathbb{Z}[0]$ that has $\mathbb{Z}$ in degree 0 and is trivial elsewhere. Note, that $\mathbb{Z}[0]$ is just one usual notation for this and no connection to the preorder category $[0]$ from Definition 5.1 is intended. Looking at the sequence (B.3), we then get a lot of terms of the form

$$\pi_n \mathrm{Map}_{\mathsf{Ch}}\big(\mathbb{Z}[0], C(Z)\big)$$

for $n \in \mathbb{N}$ and Z one of W, U, V, X, which can be computed via the following:

Proposition B.3. Let c be a chain complex and let $n \in \mathbb{N}$. Then:

$$\pi_n \mathrm{Map}_{\mathsf{Ch}}(\mathbb{Z}[0], c) \cong H_n(c)$$

where the right hand side is the n-th homology of c.

Proof. We can [K, 00PE] view the 1-category of chain complexes as a *DG-category*, which we denote by $\mathsf{Ch}^{\mathsf{dg}}$; this means [K, 00P9] that $\mathsf{Ch}^{\mathsf{dg}}$ is enriched over chain complexes, i.e. $\mathrm{Mor}_{\mathsf{Ch}^{\mathsf{dg}}}(c', c)$ is again a chain complex for all chain complexes c', c. Then the so-called *DG-nerve* [K, 00PK], a gadget similar in spirit to the homotopy coherent nerve from Remark 5.20, provides us with an ∞-category $\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})$, which can be seen as

the localization (Remark 5.23) of chain complexes at chain homotopy equivalences [Lu2, Prop. 1.3.4.5]. Mapping spaces in ∞-categories obtained via N^{dg} are particularly tractable [Lu2, Rem. 1.3.1.12], hence

$$\mathrm{Map}_{\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})}(\mathbb{Z}[0], c) \simeq \big|\mathrm{K}\big(\tau_{\geq 0}\,\mathrm{Mor}_{\mathsf{Ch}^{\mathsf{dg}}}(\mathbb{Z}[0], c)\big)\big|,$$

where

- $|\bullet|$ denotes the geometric realization of a simplicial set (in this case: after forgetting the group structure),
- K sends a non-negative chain complex to its associated simplicial Abelian group under the *Dold-Kan correspondence* [We, Sec. 8.4], and
- $\tau_{\geq 0}$ is the good truncation functor [We, 1.2.7] that essentially forces a chain complex to be non-negative.

Thus, we see

$$\begin{aligned}\pi_n\,\mathrm{Map}_{\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})}(\mathbb{Z}[0], c) &\cong \pi_n\,\mathrm{K}\,\tau_{\geq 0}\,\mathrm{Mor}_{\mathsf{Ch}^{\mathsf{dg}}}(\mathbb{Z}[0], c)\\ &\cong H_n\big(\mathrm{Mor}_{\mathsf{Ch}^{\mathsf{dg}}}(\mathbb{Z}[0], c)\big)\\ &\cong H_n(c),\end{aligned}$$

because taking the homotopy group of a geometric realization is isomorphic to the simplicial homotopy group [GJ, p. 60] and the latter is isomorphic to the homology of the associated chain complex [We, Thm. 8.4.1]; finally, $\mathrm{Mor}_{\mathsf{Ch}^{\mathsf{dg}}}(\mathbb{Z}[0], c) \cong c$ is easily seen by expanding the definitions. This shows the claim for $\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})$ instead of Ch.

To pass from $\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})$ to Ch (i.e. from localization at chain homotopy equivalences to localization at all quasi-isomorphisms), we additionally use the so-called *Hurewicz model structure* on chain complexes, which has chain homotopy equivalences as weak equivalences. It can be combined with the projective model structure to yield a *mixed model structure* on chain complexes. All three model structures are discussed in May and Ponto's book under the names "h-/q-/m-model structure" [MP, Sec. 18.3, 18.4 and 18.6]. Both, projective and mixed, have the quasi-isomorphisms as weak equivalences, but the latter has the advantage of being a right Bousfield localization of the Hurewicz model structure. This can be used to see that the restriction of a mapping space functor in the sense of Cisinski [Ci, (dual/cofibrant version of) 7.10.7] for the Hurewicz model structure yields a mapping space functor for the mixed model structure. Together with the fact that $\mathbb{Z}[0]$ is cofibrant in all three model structures, we get

$$\mathrm{Map}_{\mathrm{N}^{\mathsf{dg}}(\mathsf{Ch}^{\mathsf{dg}})}(\mathbb{Z}[0], c) \simeq \mathrm{Map}_{\mathsf{Ch}}(\mathbb{Z}[0], c).$$

■

If we apply this proposition to the long exact sequence (B.3) that we get from Section B.3 for $S = \mathbb{Z}[0]$, we indeed arrive at the familiar long exact sequence:

$$\cdots \to H_{n+1}(X) \to H_n(U \cap V) \to H_n(U) \times H_n(V) \to H_n(X) \to \cdots.$$

Appendix C:

API documentation for

inflexible-0.3.1: Semi-free differential graded-commutative algebras

A package to reason about semi-free differential graded-commutative algebras. Start reading with the `Alg` module and the `DgaSpec` type in the `Types` module.

Contents

1 Alg **3**
- 1.1 Dga construction . . . 3
- 1.2 Generators of an algebra . . . 5
- 1.3 Degree . . . 6
- 1.4 Maps of algebras . . . 6
- 1.5 Miscellaneous . . . 7

2 Alg.Classes **8**

3 Alg.Cohomology **9**
- 3.1 Utilities . . . 11
- 3.2 Low-level access . . . 12

4 Alg.GenericMor **13**

5 Cache **15**

6 Extern.Maple **16**

7 Extern.Sage **18**
- 7.1 Initialization . . . 19
- 7.2 Exposed Interface . . . 19
- 7.3 Python exceptions . . . 20

8 FracClear **21**

9 HFMext **23**

10 Lens **24**

11 Types **25**
- 11.1 Type synonyms for various types of maps . . . 26
- 11.2 Constraint abbreviations . . . 27
  - 11.2.1 Row echelon reduction . . . 27
- 11.3 Types for the cached operations . . . 28

12 Util **30**

1 Alg

```
module Alg (
    mkAlg, mkDga, mkDgaWithGenerators, mkDgaWithGeneratorsParanoid,
    finiteRepr, injectFGCA', injectFGCA'', algGenerators,
    algGenerators', polymorphic_generators, injectFGCA_even,
    injectFGCA_odd, _diff', homDegree, degree, findBasis_, extendToAlg,
    extendToDiff, isMapOfDeg, isDifferential, formalDimension,
    extendScalars, decompose
  ) where
```

Here we provide basic functionality for constructing semi-free differential graded-commutative algebras, i.e. free graded-commutative algebras with a differential that satisfies the graded Leibniz rule.

As far as this library is concerned, we use the following conventions and terminology:

- Semi-free differential graded-commutative algebras are called *cochain algebras* and we also use the abbreviation *dga*.

  Note, that both terms often denote broader classes of objects in the literature, but since this library does not treat other than the mentioned objects, there should arise no confusion.

- Free graded-commutative algebras (without differential) will simply be called *algebras*. They are represented by a dga with zero differential, if necessary.

- Differentials raise the degree by 1, i.e. we use "upper grading".

- A *monomial* is a product of algebra generators.

1.1 Dga construction

```
mkAlg :: ENO k a => [(a, Deg)] -> DgaSpec k a
```

Make a `DgaSpec` from a `[(generator, degree)]` list, together with the zero differential.

```
mkDga :: ENO k a => [(a, Deg, Lam k a)] -> DgaSpec k a
```

Make a `DgaSpec` from a `[(generator, degree, image under differential)]` list.

This function will throw an error if the given input does not describe a valid differential.

```
mkDgaWithGenerators :: ENO k a => [(a, Deg, Lam k a)] -> ([Lam k a], DgaSpec k a)
```

Like `mkDga`, but additionally, return the algebra generators *in the same order* as the input list.

Thanks to laziness, we can then use the following construct:

```
([t,dt], interval) =
    mkDgaWithGenerators [ ("t",0, dt)
                        , ("dt",1, 0) ]
```

In case of a lot of manually typed generators, `mkDgaWithGeneratorsParanoid` is an even safer variant.

```
mkDgaWithGeneratorsParanoid :: ENO k a =>
                               [(a, Lam k a, Deg, Lam k a)] -> ([Lam k a], DgaSpec k a)
```

Similar to `mkDgaWithGenerators`, but the input list must be of the form `[(generator, generator as an algebra element, degree, image under differential)]` (see also `algGenerators` and `injectFGCA'`).

We can then write

```
([t,dt], interval) =
    mkDgaWithGeneratorsParanoid [ ("t",t, 0, dt)
                                , ("dt",dt, 1, 0) ]
```

and it is checked that `t` really corresponds to the generator `"t"`. If we had accidentally written `([dt,t], interval) = …`, an error would have been thrown.

```
finiteRepr :: ENO k a => Iso' (DgaSpec k a) (Map a (Deg, Lam k a))
```

Isomorphism (in terms of the `optics` package) between `DgaSpec` and a finite representation that can be used e.g. for `Eq` and `Ord` instances.

1.2 Generators of an algebra

```
injectFGCA' :: ENO k a => DgaSpec k a -> a -> Maybe (Lam k a)
```

Given an element of type `a`, return the corresponding algebra generator. Returns `Nothing` if the element is not a generator in the given `DgaSpec`.

Mathematical analogy: If M is a free module on the basis X, this function corresponds to the canonical injection $X \hookrightarrow M \hookrightarrow \wedge(M)$.

```
injectFGCA'' :: ENO k a => DgaSpec k a -> a -> Lam k a
```

The same as `injectFGCA'`, but generates an error instead of returning `Nothing`.

```
algGenerators :: ENO k a => DgaSpec k a -> Map a (Lam k a)
```

Algebra generators of a `DgaSpec` in the form a `Map` with keys the generators as elements of `a` and values the corresponding algebra element.

```
algGenerators' :: ENO k a => DgaSpec k a -> [Lam k a]
```

Algebra generators of a `DgaSpec` as a list.

```
polymorphic_generators :: ENO k a => Generators a -> Map a (Lam k a)
```

The same as `algGenerators` for any `DgaSpec` with the given `Generators`.

```
injectFGCA_even :: ENO k a => a -> Lam k a
```

For defining dgas algorithmically, in particular for speciying the differential in `mkDga`, it is often necessary to have access to the generators as algebra elements in advance.

The functions `injectFGCA_even` and `injectFGCA_odd` make this possible, but it must be known whether the generator will have even or odd degree.

Note, that this is a potentially unsafe operation: the parity used here **must** match the parity of the degree that is later specified in `mkDga` (or otherwise constructed `Generators a`).

```
injectFGCA_odd :: ENO k a => a -> Lam k a
```

See `injectFGCA_even`.

```
_diff' :: ENO k a => DgaSpec k a -> a -> Lam k a
```

Convenience function for applying the differential of a `DgaSpec` directly to an element of type `a`. If the latter is not an algebra generator, an error is generated.

1.3 Degree

`homDegree :: Ord a => DgaSpec k a -> FGCA a -> Deg`

Degree of a monomial: the degree of $x_1 \cdots x_r$ is the sum of the degrees of the x_j.

`degree :: Ord a => DgaSpec k a -> Lam k a -> Maybe Deg`

Degree of an arbitrary algebra element. Returns `Nothing` if the given element is not homogeneous.

`findBasis_ :: ENO k a => DgaSpec k a -> Deg -> Basis k a`

Compute a `k`-basis of monomials for the given *connected* algebra in the given degree.

Connectedness means, that the input algebra **must not** have generators of degree 0.

(The cached variant of this function is `findBasis`.)

1.4 Maps of algebras

`extendToAlg :: forall k a b. ENOO k a b => (a -> Lam k b) -> AlgMor k a b`

Universal property of free graded-commutative algebras: Given a map on generators, it uniquely extends to a morphism of algebras.

Note, that it depends on `DgaSpecs` whether the resulting `AlgMor` respects the grading. The function `isMapOfDeg` can be used to check this.

It is valid to pass a partial function as long as it is defined on all actually used algebra generators.

`extendToDiff :: forall k a. ENO k a => (a -> Lam k a) -> Differential k a`

Extend a map on generators to a differential. By the Leibniz rule, this extension is unique.

Note, that it depends on a `DgaSpec` whether the resulting `Differential` is sound (i.e. whether it respects the grading and composition with itself is zero). The function `isDifferential` can be used to check this.

It is valid to pass a partial function as long as it is defined on all actually used algebra generators.

`isMapOfDeg :: ENOO k a b => DgaSpec k a -> DgaSpec k b -> Deg -> LinHom k a b -> Bool`

Check if a given given `LinHom` is graded of a specified degree.

For example, given `DgaSpecs` `algA` and `algB`, use "`isMapOfDeg algA algB 0 …` " to check if a `LinHom` is of degree 0, i.e. a morphism of graded modules.

`isDifferential :: ENO k a => DgaSpec k a -> LinEndo k a -> Bool`

Check if a given `LinEndo` is a differential for the given algebra, i.e. if it raises degree by 1 and composition with itself is zero.

(The differential of the given `DgaSpec` is ignored.)

1.5 Miscellaneous

`formalDimension :: Ord a => DgaSpec k a -> Int`

The formal dimension of an elliptic Sullivan algebra.

`extendScalars :: (ENOO k a b, Algebra k b, p ~ Vect k b) => DgaSpec k a -> DgaSpec p a`

For a `k`-algebra `p`, extend `DgaSpec k a` to `DgaSpec p a`.

`decompose :: FGCA a -> Maybe (Either a a, FGCA a)`

Decompose a monomial $x_1 \cdot x_2 \cdots x_r$ into a pair, consisting of x_1 and $x_2 \cdots x_r$.

Returns `Nothing` on the empty product, `(Left x_1, ...)` if x_1 is of even degree, and `(Right x_1, ...)` if x_1 is of odd degree.

2 Alg.Classes

```
module Alg.Classes (

  ) where
```

In this module, `Eq`, `Ord`, and `Show` instances for the `DgaSpec` type are implemented.

3 Alg.Cohomology

```
module Alg.Cohomology (
    isCocycle, isCocycle', isCoboundary, isCoboundary',
    fillCochain, fillCochain', isElliptic, isElliptic',
    mappingDegreeAt, mappingDegreeAt', findBasis, bdyBasis,
    bdyMonomials, hasOnlyBdyMonos, llDifferential
  ) where
```

A cochain algebra is, in particular, a cochain complex. Here we provide functionality to reason about the cohomology of the latter. Throughout this module, we assume, that all algebras are connected (that is, they have no generators of degree 0).

Most functions in this module expect *homogeneous* algebra elements as input, and **this is not checked**. It is the caller's responsibity to pass only such elements. When in doubt, see if `degree` yields `Just` some degree.

For brevity, we use the term *cochain* as an abbreviation for homogeneous element. (This coincides with the terminology of cochain complexes in homological algebra where non-homogeneous elements do usually not appear.)

As an additional convenience (mainly useful for interactive sessions), we provide "primed" variants of the main functions of this module, which discard the cache (`dropCache`) after they are done.

```
isCocycle :: ENO k a => DgaSpec k a -> Lam k a -> Cached k a Bool
```

Check whether an algebra element is a cocycle, i.e. whether it is in the kernel of the differential. (This also works on non-homogeneous elements.)

```
isCocycle' :: ENO k a => DgaSpec k a -> Lam k a -> Bool
```

See `isCocycle`.

```
isCoboundary :: (ENO' k a, Elim' k, Eq (UnFrac k)) =>
                DgaSpec k a -> Lam k a -> Cached k a Bool
```

Check whether a cochain is a coboundary, i.e. whether it is in the image of the differential.

```
isCoboundary' :: (ENO' k a, Elim' k, Eq (UnFrac k)) => DgaSpec k a -> Lam k a -> Bool
```

See `isCoboundary`.

`fillCochain`

```
::  (ENO' k a, Elim' k, Eq (UnFrac k))
=>  DgaSpec k a
->  Lam k a                              cochain c to be filled
->  Cached k a (Maybe (Lam k a))         a cochain c' such that d c' = c, or
                                         Nothing if c wasn't a coboundary.
```

Given a cochain, compute `Just` a preimage of it under the differential; or `Nothing` if it isn't a coboundary.

`fillCochain'`

```
::  (ENO' k a, Elim' k, Eq (UnFrac k))
=>  DgaSpec k a
->  Lam k a                              cochain c to be filled
->  Maybe (Lam k a)                      a cochain c' such that d c' = c, or
                                         Nothing if c wasn't a coboundary.
```

See `fillCochain`.

```
isElliptic :: (ENO' k a, Elim' k, Eq (UnFrac k)) => DgaSpec k a -> Cached k a (Maybe Bool)
```

Given a dga, we try to show whether it is *elliptic*; because our cochain algebras are finitely generated by design, ellipticity is equivalent to finite dimensional cohomology.

The return value `Nothing` signifies unknown ellipticity status.

Currently, this function can only determine the latter, if all generators of even degree are cocycles. This includes the case of pure Sullivan algebras.

```
isElliptic' :: (ENO' k a, Elim' k, Eq (UnFrac k)) => DgaSpec k a -> Maybe Bool
```

See `isElliptic`.

`mappingDegreeAt`

```
::  (ENO' k a, Elim' k)
=>  DgaSpec k a
->  Lam k a                 representative of fundamental class
->  DgaEndo k a             endomorphism
->  Cached k a k            mapping degree of the endomorphism
```


Suppose `k` is a field. Let f be an endomorphism of the given dga A and let c be a cochain in A of degree n, such that

- the cohomology $H^n(A)$ in degree n is one-dimensional over `k`, and
- the cochain c represents its non-trivial class, i.e. c is a cocycle but *not* a coboundary.

Then this function computes the *mapping degree of f with respect to c*, i.e. the element λ from `k`, such that $H(f)[c] = \lambda \cdot [c] \in H^n(A)$ holds.

In the case of a Poincaré dga, e.g. an elliptic dga, of dimension n, the cochain c is a representative of the fundamental class and this function simply returns the *mapping degree of f*.

`mappingDegreeAt'`

```
::  (ENO' k a, Elim' k)
=>  DgaSpec k a
->  Lam k a                 representative of fundamental class
->  DgaEndo k a             endomorphism
->  k                       mapping degree of the endomorphism
```

See `mappingDegreeAt`.

3.1 Utilities

```
findBasis :: ENO' k a => DgaSpec k a -> Deg -> Cached k a (Basis k a)
```

Cached version of `findBasis_`.

```
bdyBasis :: (ENO' k a, Elim k) => DgaSpec k a -> Deg -> Cached k a (Basis k a)
```

Compute a basis of the submodule of coboundaries of a given `DgaSpec` in a specified degree.

```
bdyMonomials :: (ENO' k a, Elim k) => DgaSpec k a -> Deg -> Cached k a (Set (FGCA a))
```

Compute the `Set` of all monomials that appear in the submodule of coboundaries in a specified degree.

```
hasOnlyBdyMonos :: (ENO' k a, Elim k) => DgaSpec k a -> Deg -> Lam k a -> Cached k a Bool
```

Check whether an algebra element contains only monomials that appear in the submodule of coboundaries of a given degree.

This is mainly useful as a first (and quick to check) necessary condition on an element to be a coboundary.

3.2 Low-level access

```
llDifferential :: (ENO' k a, Elim k) =>
                  DgaSpec k a -> Deg -> Cached k a (EchelonInfo (Mat k))
```

Reduce a matrix representation of the differential in the specified degree to row echelon form and additionally cache other information that is obtained on the way.

This function is used internally by the rest of this module.

4 Alg.GenericMor

```
module Alg.GenericMor (
    genericAlgMor, genericDgaMorConstraints,
    genericDgaEndoConstraints
  ) where
```

In this module, we provide *generic* morphisms from one algebra to another and deduce polynomial constraints for it being a dga morphism. In more detail, this means the following:

Let (A, d_A) and (B, d_B) be cochain algebras over `k`, generated by $a_1, \dots, a_r$ and $b_1, \dots, b_s$, respectively. A morphism of algebras $A \to B$ must map each a_i to a linear combination of monomials in the b_j of the same degree, and each such choice for mapping the generators yields a morphism (see also `extendToAlg`). Abstracting over the coefficients, we may describe a *generic morphism* f by the rules

$$a_i \mapsto \sum_m T_{i,m} \cdot m$$

with m ranging over all monomials in B that are of the same degree as a_i, where the $T_{i,m}$ are variables of a polynomial ring over `k`.

The equation $d_B \circ f - f \circ d_A = 0$, evaluated on the a_i, determines a finite number of polynomial equations in the $T_{i,m}$. A choice of substitution for the $T_{i,m}$ in f determines a dga morphism $(A, d_A) \to (B, d_B)$ if and only if it satisfies this system of polynomial equations.

```
genericAlgMor :: forall p k a b m v.
                 (ENOO k a b, PolyRing k p m v, Enum v) =>
                 DgaSpec k a -> DgaSpec k b -> AlgMor p a b
```

Generate a generic morphism of algebras. The type of variables `v` must be an `Enum`, from which the $T_{\dots}$-variables are sampled (ascending and starting at "1").

```
genericDgaMorConstraints :: forall p k a b m v.
                            (ENOO k a b, PolyRing k p m v, Enum v) =>
                            DgaSpec k a -> DgaSpec k b -> [p]
```

Based on `genericAlgMor`, this computes the polynomial constraints.

```
genericDgaEndoConstraints :: forall p k m v a.
                             (ENO k a, PolyRing k p m v, Enum v) => DgaSpec k a -> [p]
```

Specialization of `genericDgaMorConstraints` where domain and codomain of the morphism coincide.

5 Cache

```
module Cache (
    cacheResult, cachePureResult, viewCache, setCache, viewCached,
    setCached, viewCachedErr, dropCache, emptyCache
  ) where
```

This module provides a simple cache framework for storing and retrieving results of functions that expect a `DgaSpec` and a particular degree as input. For `Keys` we use the lens from the `Lens` module, which facilitates easy access to the values.

```
cacheResult :: ENO' k a =>
               Key k a y
               -> (DgaSpec k a -> Deg -> Cached k a y)
                  -> DgaSpec k a -> Deg -> Cached k a y

cachePureResult :: ENO' k a =>
                   Key k a y
                   -> (DgaSpec k a -> Deg -> y) -> DgaSpec k a -> Deg -> Cached k a y

viewCache :: ENO' k a => Key k a y -> DgaSpec k a -> Deg -> Cache k a -> Maybe y

setCache :: ENO' k a => Key k a y -> DgaSpec k a -> Deg -> y -> Cache k a -> Cache k a

viewCached :: ENO' k a => Key k a y -> DgaSpec k a -> Deg -> Cached k a (Maybe y)

setCached :: ENO' k a => Key k a y -> DgaSpec k a -> Deg -> y -> Cached k a ()

viewCachedErr :: ENO' k a => String -> Key k a y -> DgaSpec k a -> Deg -> Cached k a y

dropCache :: Cached k a y -> y

emptyCache :: Cache k a
```

6 Extern.Maple

```
module Extern.Maple (
    MapleInterp(MapleInterp, hproc, hout, hin), newMaple,
    closeMaple, putMaple, getMaple
  ) where
```

This module provides a very elementary interface to the `maple` executable.

```
data MapleInterp
```

Constructors

```
=  MapleInterp
       { hin :: Handle
       , hout :: Handle
       , hproc :: ProcessHandle
       }
```

```
newMaple :: IO MapleInterp
```

Create new instance of the maple interpreter. The `maple` executable must be available on the executing system (and must be able to obtain a license, if necessary).

```
closeMaple :: MapleInterp -> IO ()
```

Quits a maple interpreter and waits for its termination. The passed `MapleInterp` should not be used afterwards.

```
putMaple :: MapleInterp -> String -> IO ()
```

Feed input to a maple interpreter.

```
getMaple :: MapleInterp -> IO [String]
```

Read one "chunk" of maple output (split at linebreaks); we consider the end of each chunk to be three empty lines (which are *not* part of the returned list). For example, running


```
do m <- newMaple
   putMaple m "41 + 1; printf(\"\\n\\n\\n\");"
   res <- getMaple m
   return (res == ["42"])
```

in the `IO` monad should yield `True`.

7 Extern.Sage

```
module Extern.Sage (
    initialize, VarNum, Polynomial, IdealGenerators,
    idealDimension, idealDimension', hasFiniteProjection,
    hasFiniteProjection', eliminationIdeal, eliminationIdeal',
    numberOfVariables, onPyException, printPyException
  ) where
```

This module provides an interface to some functionality of SageMath. More specifically, we use the module for "Ideals in multivariate polynomial rings".

An accompanying Python module `pybits/Extern/sageglue.py` is responsible for calling the actual sage functions. We use the `cpython` package to call this Python module directly from Haskell.

Usage notes

Everything in this module happens over the base field $\mathbb{Q}$, i.e. the rational numbers.

For simplicity, this interface imposes the following restrictions:

- A `Polynomial` in the sense of this interface has a fixed variable type of `VarNum = Int` and by convention, we use only non-negative variables.
- No variable gaps are possible: if the variable `2 :: VarNum` is used in some polynomial, we automatically work over a polynomial ring in three variables, namely `0`, `1`, and `2`, whether `0` and `1` are used or not. (See also `numberOfVariables`.)

This means that, in order to use this interface, you have to enumerate your variables by `0,1,…` and convert your polynomials to `Polynomial`s first. In the case of a `newtype` around `Int` as variable type (as used in the `interactive/` examples) the conversion is easily handled by `coerce`.

7.1 Initialization

```
initialize :: IO ()
```

Initialize the Python and Sage bits. **Calling this is mandatory before using the interface below.**

7.2 Exposed Interface

```
type VarNum = Int
```

```
type Polynomial = GrevlexPoly Q VarNum
```

Polynomials as used by this interface, based on the module `Math.CommutativeAlgebra.Polynomial` from the `HaskellForMaths` package.

```
type IdealGenerators = [Polynomial]
```

```
idealDimension
```

```
::  Int                 total number of variables
->  IdealGenerators
->  IO Integer
```

Dimension of the generated ideal within a polynomial ring with specified number of variables.

```
idealDimension' :: IdealGenerators -> IO Integer
```

Convenience function that calls `idealDimension` with the number of variables as computed by `numberOfVariables`.

```
hasFiniteProjection
```

```
::  Int                 total number of variables
->  IdealGenerators
->  VarNum              projection target
->  IO Bool
```

Let `V` be the zero locus of the given ideal (generators) in `n`-dimensional affine space, where `n` is the first argument. Then this function computes whether the projection of `V` onto the affine line that is given by the third argument has a finite image.

```
hasFiniteProjection' :: IdealGenerators -> VarNum -> IO Bool
```

Convenience function that calls `hasFiniteProjection` with the number of variables as computed by `numberOfVariables`.

```
eliminationIdeal
    ::  Int                 total number of variables
    ->  IdealGenerators
    ->  [VarNum]            variables to eliminate
    ->  IO IdealGenerators
```

Consider the polynomial ring in a given number of variables and let `I` be the ideal given by the specified generators. This function computes the elimination ideal resulting from eliminating the given variables from `I`.

```
eliminationIdeal'
    ::  IdealGenerators
    ->  [VarNum]            variables to eliminate
    ->  IO IdealGenerators
```

Convenience function that calls `eliminationIdeal` with the number of variables as computed by `numberOfVariables`.

```
numberOfVariables :: [Polynomial] -> Int
```

Computes the number of variables "used" in the given polynomials. Here, a variable `j` counts as used if there exists a variable `k` in the given polynomials with `j  k`. In other words, it simply returns `1 + maximal variable that actually appears`.

7.3 Python exceptions

```
onPyException :: Exception -> IO ()
```

An exception handler for Python exceptions, printing type, value, and traceback (if available) to standard error.

```
printPyException :: IO a -> IO ()
```

This function is useful for debugging a Python exception: When you get a non-informative

```
*** Exception: <CPython exception>
```

message, feeding the same action to this function will make Python print more details about the exception.

8 FracClear

```
module FracClear (
    FracClear(unfracs, unfrac, fromUnFrac, UnFrac)
  ) where
```

See `FracClear` type class.

Background: This module exists, because exact calculations with `Ratio k` elements can be terribly slow, possibly in contrast to direct computations with elements of `k`. So if, for a particular task, denominators can be "put outside of the calculation", using `FracClear` might give a performance increase.

```
class (Num k, Num (UnFrac k)) => FracClear k where
```

Making a commutative ring `k` an instance of `FracClear` encapsules the abstract concept of "clearing fractions".

Note, that this does *not* require the presence of actual fractions: for example, the integers also fit this concept (with all "denominators" equal to `1`). In this sense, the terminology "fraction", "numerator" and "denominator" is used abstractly here.

Associated Types

```
type UnFrac k
```

`UnFrac k` is the type of numerator and denominator of an element of `k`.

Methods

```
fromUnFrac :: UnFrac k -> k
```

Inclusion of `UnFrac k` into `k`, usually by putting `1` as denominator.

```
unfrac
```

```
::  k
->  (UnFrac k, UnFrac k)
```
(numerator, denominator)

Split an element into numerator and denominator, i.e. we want

```
let (x,y) = unfrac val
 in val * y == x
```

to be `True`.

```
unfracs :: [k] -> ([UnFrac k], UnFrac k)
```

Given a list of elements, clear all denominators simultaneously, i.e. we want

```
let ([xs],y) = unfracs vals
 in map (* y) vals == xs
```

to be `True`.

A default implementation in terms of `unfrac` is provided – however, a better method might be available. For example, for rational numbers, the least common multiple of all present denominators provides a suitable `y`.

```
instance FracClear Q
instance FracClear Integer
instance FracClear Int
instance Integral i => FracClear (Ratio i)
```


9 HFMext

```
module HFMext (
    basisElems, coeffs, changeBaseRing
  ) where
```

This module contains some functionality that is directly related to the `HaskellForMaths(-excerpt)` package, but does not seem to exist there.

```
basisElems :: Vect k b -> [b]
```

```
coeffs :: Vect k b -> [k]
```

```
changeBaseRing :: (Eq k', Num k') => (k -> k') -> Vect k a -> Vect k' a
```

10 Lens

```
module Lens (
    deepSetDef
  ) where
```

This module provides `Lens'` in the sense of the `optics` package family (mainly) for the `Cache` machinery.

Lens are generically derived via the packages `generics-sop` and `optics-sop` for the record fields of the types `Cache`, `GenPropCache`, `DgaPropCache`, and `DgaSpec`; following the usual convention that the lenses' names are the same as the record field names without the initial underscore.

```
deepSetDef :: (Alternative f, Is k A_Setter, Is l A_Setter) =>
              a -> Optic' k is s (f a) -> Optic' l is' a x -> x -> s -> s
```

Given

```
o1 :: Setter' s (f a)
```

and

```
o2 :: Setter' a x
```

where `f` is an `Alternative` and we have some default value for `a`, combine `o1` and `o2` in the following way: if `o1`'s target is empty, replace it by `pure` the given default; then use `o2` to set the inner target.

11 Types

```
module Types (
    DgaSpec(DgaSpec, _diff, _gens), Deg, Generators, Differential,
    Lam, FGCA, PolyRing, AlgMap, LinHom, AlgMor, DgaMor, LinEndo,
    AlgEndo, DgaEndo, ENO, ENOO, ENO', Elim, Elim',
    Cache(Cache, _dgaProp, _genProp), DegMap, Basis, Mat,
    GenPropCache(GenPropCache, _chainsBasis),
    DgaPropCache(DgaPropCache,
                 _differentialFs,
                 _differentialUF,
                 _monomialsBdy,
                 _boundariesBasis),
    Cached, Key(DgaKey, GenKey), PropKey
  ) where
```

This module collects all data types and type synonyms that are used in the main library.

Note, that we use the terminology that we introduce in the beginning of the `Alg` module.

```
data DgaSpec k a
```

A `DgaSpec k a` contains the specification of a dga over `k`, in terms of algebra generators with degrees and a differential.

Here, `k` is the base ring and `a` is the type of the generators. (Note, that *not all* inhabitants of `a` need be generators. For example, `a` might be `String` but only `"x"` and `"y"` be used as generators.)

Constructors

```
=  DgaSpec
       { _gens :: Generators a
       , _diff :: Differential k a
       }
```

```
instance Generic (DgaSpec k a)
instance (Show k, Show a, ENO k a) => Show (DgaSpec k a)
instance Generic (DgaSpec k a)
instance ENO k a => Eq (DgaSpec k a)
instance (ENO k a, Ord k) => Ord (DgaSpec k a)
type instance Rep (DgaSpec k a) = D1 ('MetaData "DgaSpec" "Types" "inflexible-0.3.0-inplace" 'Fal
type instance Code (DgaSpec k a) = GCode (DgaSpec k a)
```

```
type Deg = Int
```

Type synonym for degree in the sense of gradings.

```
type Generators a = Map a Deg
```

A collection of algebra generators is represented as a `Map` whose keys are the generators with value the respective degree.

```
type Differential k a = Lam k a -> Lam k a
```

A differential is represented by an actual Haskell function. Note, that this means that a `DgaSpec` itself is usually not a "finite" object, but see `finiteRepr`.

```
type Lam k a = Vect k (FGCA a)
```

`Lam k a` types the elements of algebras specified by instances of `DgaSpec k a`.

```
type FGCA a = Tensor (SymmetricAlgebra a) (ExteriorAlgebra a)
```

Following the `HaskellForMaths` convention (e.g. in `Math.Algebras.TensorAlgebra`), the type (synonym) `FGCA`, which is short for `FreeGradedCommutativeAlgebra`, types the *basis elements* of the free module structure of such an algebra.

```
type PolyRing k p m v = (MonomialConstructor m, Ord (m v), Algebra k (m v), p ~ Vect k (m v))
```

The type constraint synonym `PolyRing k p m v` says that we want the type `p` to represent a polynomial ring over `k` in variables coming from the type `v`, using the monomial constructor `m` (see `Math.CommutativeAlgebra.Polynomial` of the `HaskellForMaths` package).

11.1 Type synonyms for various types of maps

Of course, the following type synonyms do *not* provide any type safety, e.g. an `AlgMor` is not guaranteed to preserve the algebra structure. However, it makes type signatures self-contained and clearly states the assumptions on the input.

```
type AlgMap k a b = Lam k a -> Lam k b
```

Short for a *set map* of algebras.


```
type LinHom k a b = AlgMap k a b
```

Short for `LinearHomomorphism`, i.e. a graded `k`-homomorphism of underlying graded modules of algebras (possibly of non-zero degree, i.e. grade-shifting)

```
type AlgMor k a b = LinHom k a b
```

Short for `AlgebraMorphism`, i.e. a morphism in the category of graded algebras, i.e. a graded `k`-homomorphism of degree 0 that is compatible with multiplication and unit.

```
type DgaMor k a b = AlgMor k a b
```

Short for a morphism of cochain algebras, i.e. an `AlgMor` that commutes with the differentials

```
type LinEndo k a = LinHom k a a
```

Short for a `k`-linear endomorphism of the underlying graded module of an algebra

```
type AlgEndo k a = AlgMor k a a
```

Short for an endomorphism of a graded algebra

```
type DgaEndo k a = DgaMor k a a
```

Short for an endomorphism of a cochain algebra

11.2 Constraint abbreviations

Most functions need some `Eq`, `Num` and `Ord` instances. The following synonyms are used for convenience.

```
type ENO k a = (Eq k, Num k, Ord a)
```

```
type ENOO k a b = (ENO k a, Ord b)
```

```
type ENO' k a = (ENO k a, Ord k)
```

11.2.1 Row echelon reduction

Additionally, computations in `Alg.Cohomology` use row echelon reduction of matrices, for which we require the base ring `k` to satisfy the following type constraints. ("Elim" as in "Gaussian elimination".)

```
type Elim k = (FracClear k, MatrixReprFor (UnFrac k), MatrixEchelonFor (UnFrac k))
```

```
type Elim' k = (Elim k, Fractional k, MatrixAccessFor (UnFrac k))
```

11.3 Types for the cached operations

```
data Cache k a
```

A `Cache k a` caches computational data for cochain algebras specified by instances of `DgaSpec k a`.

Constructors

```
=  Cache
       { _genProp :: Map (Generators a) (GenPropCache k a)
       , _dgaProp :: Map (DgaSpec k a) (DgaPropCache k a)
       }
```

```
instance Generic (Cache k a)
instance Generic (Cache k a)
type instance Rep (Cache k a) = D1 ('MetaData "Cache" "Types" "inflexible-0.3.0-inplace" 'False)
type instance Code (Cache k a) = GCode (Cache k a)
```

```
type DegMap x = Map Deg x
```

```
type Basis k a = [Lam k a]
```

Here, `Basis` means: basis of some submodule.

```
type Mat k = Matrix (UnFrac k)
```

```
data GenPropCache k a
```

Cached data that only depends on the algebra generators.

Constructors

```
=  GenPropCache
       { _chainsBasis :: DegMap (Basis k a)
       }
```

```
instance Monoid (GenPropCache k a)
instance Semigroup (GenPropCache k a)
instance Generic (GenPropCache k a)
instance Generic (GenPropCache k a)
type instance Rep (GenPropCache k a) = D1 ('MetaData "GenPropCache" "Types" "inflexible-0.3.0-inp
type instance Code (GenPropCache k a) = GCode (GenPropCache k a)
```



```
data DgaPropCache k a
```

Cached data that also depends on the differental.

Constructors

```
=  DgaPropCache
       { _boundariesBasis :: DegMap (Basis k a)
       , _monomialsBdy :: DegMap (Set (FGCA a))
       , _differentialUF :: DegMap (EchelonInfo (Mat k))
       , _differentialFs :: DegMap [UnFrac k]
       }
```

```
instance Monoid (DgaPropCache k a)
instance Semigroup (DgaPropCache k a)
instance Generic (DgaPropCache k a)
instance Generic (DgaPropCache k a)
type instance Rep (DgaPropCache k a) = D1 ('MetaData "DgaPropCache" "Types" "inflexible-0.3.0-inp
type instance Code (DgaPropCache k a) = GCode (DgaPropCache k a)
```

```
type Cached k a x = State (Cache k a) x
```

```
data Key k a y
```

Constructors

```
=  GenKey (PropKey GenPropCache k a y)
|  DgaKey (PropKey DgaPropCache k a y)
```

```
type PropKey t k a y = Lens' (t k a) (DegMap y)
```

12 Util

```
module Util (
    (.:), (<&>), allSatisfy, solveDegreeEquation, multiAt,
    toSupscript, allM
  ) where
```

`(.:) :: (c -> d) -> (a -> b -> c) -> a -> b -> d`

Composition with "two arguments": `(f .: g) x y = f (g x y)`.

`(<&>) :: Functor f => f a -> (a -> b) -> f b`

Flipped `(<$>)`.

`allSatisfy :: Foldable t => t a -> (a -> Bool) -> Bool`

Flipped `all` for infix use.

`solveDegreeEquation`

`::`	`Integral i`	
`=>`	`[i]`	input coefficients
`->`	`i`	target value
`->`	`[[i]]`	solutions

A naive implementation to solve the following Diophantine problem: Given positive integers $d_1, \ldots, d_r$, n, find all solutions $x \in (\mathbb{Z}_{\geq 0})^r$ to the equation

$$\sum_{j=1}^{r} d_j \cdot x_j = n$$

with the additional property that $x_j \in \{0, 1\}$ if d_j is odd.

`multiAt :: Integral i => [a] -> [i] -> [a]`

Like `genericIndex` but for many indices.

`toSupscript :: Char -> Char`

Transform digits to supscript characters. Produces an error on non-digit input. Example:

```
map toSupscript "123" = "¹²³"
```

`allM :: Monad m => (a -> m Bool) -> [a] -> m Bool`

From `GHC.Utils.Monad`: Monad version of `all`, aborts the computation at the first `False` value

Bibliography

[AHS] J. Adámek, H. Herrlich, and G. E. Strecker. *Abstract and Concrete Categories. The Joy of Cats*. Wiley, 1990. Most recent version can be found here: http://katmat.math.uni-bremen.de/acc/. Cited on page 48.

[AJ] M. Anel and A. Joyal. *Sweedler Theory for (co)algebras and the bar-cobar constructions*. 2013. arXiv: 1309.6952 [math.CT]. Cited on page 22.

[AL] M. Arkowitz and G. Lupton. *Rational obstruction theory and rational homotopy sets*. In: Math. Z. 235.3 (2000), pp. 525–539. arXiv: math/0010126 [math.AT]. Cited on pages vi, 20, 31.

[Am] M. Amann. *Degrees of self-maps of simply connected manifolds*. In: Int. Math. Res. Not. 18 (2015), pp. 8545–8589. The earlier version arXiv:1109.0960v1 contains mostly the same mathematical content, but deviates in exposition and numbering(!). Cited on page 20.

[Be1] J. E. Bergner. *A survey of* $(\infty, 1)$*-categories*. In: *Towards Higher Categories*. Springer, 2010, pp. 69–83. arXiv: math/0610239 [math.AT]. Cited on page 55.

[Be2] J. E. Bergner. *The Homotopy Theory of* $(\infty, 1)$*-Categories*. Cambridge University Press, 2018. Cited on page 55.

[BK] C. Barwick and D. M. Kan. *Relative categories: Another model for the homotopy theory of homotopy theories*. In: Indag. Math. (N.S.) 23.1–2 (2012), pp. 42–68. arXiv: 1011.1691 [math.AT]. Cited on page 54.

[BP] R. Benedetti and C. Petronio. *Lectures on hyperbolic geometry*. Springer-Verlag, 1992. Cited on page 2.

[BV] J. M. Boardman and R. M. Vogt. *Homotopy invariant algebraic structures on topological spaces*. Springer-Verlag, 1973. Cited on page 55.

[Ci] D.-C. Cisinski. *Higher Categories and Homotopical Algebra*. Cambridge University Press, 2019. Last version and errata available at http://www.mathematik.uni-regensburg.de/cisinski/publikationen.html. Cited on pages 45, 47, 48, 50, 52, 53, 54, 59, 60, 67, 70, 71, 75, 77.

[CL] D. Crowley and C. Löh. *Functorial seminorms on singular homology and (in)flexible manifolds*. In: Algebr. Geom. Topol. 15.3 (2015), pp. 1453–1499. arXiv: 1103.4139 [math.GT]. Cited on pages iii, iv, vi, 3, 5, 8, 17, 18, 19, 20, 26, 31, 34, 38.

[CLO] D. A. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, and Algorithms. An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Fourth Edition. Springer, 2015. Cited on page 43.

[CMuV] C. Costoya, V. Muñoz, and A. Viruel. *On Strongly Inflexible Manifolds*. In: Int. Math. Res. Not. (Mar. 2022). arXiv: 2101.01961 [math.GT]. Cited on page 20.

[CMV1] C. Costoya, D. Méndez, and A. Viruel. *Homotopically rigid Sullivan algebras and their applications*. In: *An Alpine Bouquet of Algebraic Topology*. Amer. Math. Soc., 2018, pp. 103–121. arXiv: 1701.03705 [math.AT]. Cited on page 21.

[CMV2] C. Costoya, D. Méndez, and A. Viruel. *Realisability problem in arrow categories*. In: Collect. Math. 71.3 (2020), pp. 383–405. arXiv: 1901.03152 [math.AT]. Cited on page 21.

[CV] C. Costoya and A. Viruel. *Every finite group is the group of self-homotopy equivalences of an elliptic space*. In: Acta Math. 213.1 (2014), pp. 49–62. arXiv: 1106.1087 [math.AT]. Cited on page 20.

[DGPS] W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. *SINGULAR — A computer algebra system for polynomial computations*. http://www.singular.uni-kl.de. Cited on page 31.

[DK1] W. G. Dwyer and D. M. Kan. *Simplicial localizations of categories*. In: J. Pure Appl. Algebra 17.3 (1980), pp. 267–284. Cited on page 54.

[DK2] W. G. Dwyer and D. M. Kan. *Calculating simplicial localizations*. In: J. Pure Appl. Algebra 18.1 (1980), pp. 17–35. Cited on page 54.

[DK3] W. G. Dwyer and D. M. Kan. *Function complexes in homotopical algebra*. In: Topology 19.4 (1980), pp. 427–440. Cited on page 54.

[DS1] D. Dugger and D. I. Spivak. *Mapping spaces in quasi-categories*. In: Algebr. Geom. Topol. 11.1 (2011), pp. 263–325. arXiv: 0911.0469 [math.AT]. Cited on page 75.

[DS2] D. Dugger and D. I. Spivak. *Rigidification of quasi-categories*. In: Algebr. Geom. Topol. 11.1 (2011), pp. 225–261. arXiv: 0910.0814 [math.CT]. Cited on page 53.

[Ei] D. Eisenbud. *Commutative Algebra with a View Toward Algebraic Geometry*. Springer-Verlag, 1995. Cited on pages 21, 26.

[FHT] Y. Félix, S. Halperin, and J.-C. Thomas. *Rational Homotopy Theory*. Springer-Verlag, 2001. Cited on pages vi, 17, 22, 26, 31, 41.

[FL] D. Fauser and C. Löh. *Exotic finite functorial semi-norms on singular homology*. In: Glasg. Math. J. 61.2 (2019), pp. 287–295. arXiv: 1605.04093 [math.GT]. Cited on pages v, 8, 18, 19.

[Fr] G. Friedman. *Survey article: An elementary illustrated introduction to simplicial sets*. In: Rocky Mountain J. Math. 42.2 (2012), pp. 353–423. arXiv: 0809.4221 [math.AT]. Cited on page 45.

[Ga] A. Gaifullin. *Universal realisators for homology classes*. In: Geom. Topol. 17.3 (2013), pp. 1745–1772. Cited on page 5.

[GJ] P. G. Goerss and J. F. Jardine. *Simplicial Homotopy Theory*. Birkhäuser Basel, 2009. Cited on pages 45, 47, 77.

[GM] P. Griffiths and J. Morgan. *Rational Homotopy Theory and Differential Forms*. Second Edition. Springer, 2013. Cited on page 22.

[Go1] T. G. Goodwillie. *Calculus I: The First Derivative of Pseudoisotopy Theory*. In: K-Theory 4.1 (1990), pp. 1–27. Cited on page iv.

[Go2] T. G. Goodwillie. *Calculus II: Analytic Functors*. In: K-Theory 5.4 (1992), pp. 295–332. Cited on page iv.

[Go3] T. G. Goodwillie. *Calculus III: Taylor Series*. In: Geom. Topol. 7 (2003), pp. 645–711. Cited on pages iv, viii.

[Grth] M. Groth. *A short course on ∞-categories*. 2015. arXiv: 1007.2925 [math.AT]. Cited on pages 45, 52.

[Grv1] M. Gromov. *Volume and bounded cohomology*. In: Inst. Hautes Études Sci. Publ. Math. 56 (1982), pp. 5–99. URL: http://www.numdam.org/item?id=PMIHES_1982__56__5_0. Cited on pages iii, iv, vi, vii, 1, 2, 19.

[Grv2] M. Gromov. *Metric Structures for Riemannian and Non-Riemannian Spaces*. With Appendices by M. Katz, P. Pansu and S. Semmes. Birkhäuser Boston, 1999. Cited on pages iii, iv, vi, 2, 19.

[Ha] A. Hatcher. *Algebraic Topology*. Cambridge University Press, 2002. URL: https://www.math.cornell.edu/~hatcher/AT/ATpage.html. Cited on pages 57, 69.

[Hi] P. S. Hirschhorn. *Model Categories and Their Localizations*. American Mathematical Society, 2009. Cited on pages 54, 65, 68, 69.

[Ho] M. Hovey. *Model Categories*. American Mathematical Society, 2007. Cited on pages 65, 69, 70.

[Iv] N. V. Ivanov. *Notes on the bounded cohomology theory*. 2017. arXiv: 1708.05150 [math.AT]. Cited on pages 57, 61.

[IY] H. Inoue and K. Yano. *The Gromov invariant of negatively curved manifolds*. In: Topology 21.1 (1982), pp. 83–89. Cited on page 19.

[Jo1] A. Joyal. *Quasi-categories and Kan complexes*. In: J. Pure Appl. Algebra 175.1 (2002), pp. 207–222. Cited on page 55.

[Jo2] A. Joyal. *Notes on Quasi-Categories*. 2008. URL: http://www.math.uchicago.edu/~may/IMA/Joyal.pdf. Cited on page 55.

[K] J. Lurie. *Kerodon*. https://kerodon.net, "an online resource for homotopy-coherent mathematics". Cited on pages 45, 47, 48, 50, 52, 53, 54, 58, 63, 76.

[Lo] Z. L. Low. *Universes for category theory*. 2013. arXiv: 1304.5227 [math.CT]. Cited on page 48.

[Lö1] C. Löh. *ℓ^1-Homology and Simplicial Volume*. PhD thesis. WWU Münster, 2007. Available at http://nbn-resolving.de/urn:nbn:de:hbz:6-37549578216 or (with errata) at https://loeh.app.uni-regensburg.de/preprints/. Cited on pages iii, vii, 5.

[Lö2] C. Löh. *Isomorphisms in ℓ^1-homology*. In: Münster J. Math. 1 (2008), pp. 237–265. arXiv: math/0612589 [math.AT]. Cited on pages vii, 57, 61.

[Lö3] C. Löh. *Finite functorial semi-norms and representability*. In: Int. Math. Res. Not. IMRN 12 (2016), pp. 3616–3638. arXiv: 1404.6557 [math.AT]. Cited on pages iii, iv, 8.

[Lö4] C. Löh. *Exploring Formalisation. A Primer in Human-Readable Mathematics in Lean 3 with Examples from Simplicial Topology*. Springer-Verlag, 2022. Cited on page 9.

[Lö5] C. Löh. *Simplicial Volume*. Manifold Atlas Project, http://www.map.mpim-bonn.mpg.de/Simplicial_volume. Cited on page 2.

[Lu1] J. Lurie. *Higher Topos Theory*. Princeton University Press, 2009. Most recent version can be found here: https://www.math.ias.edu/~lurie/. Cited on pages 45, 47, 48, 50, 52, 53, 55, 57, 67, 74.

[Lu2] J. Lurie. *Higher Algebra*. 2017. URL: https://www.math.ias.edu/~lurie/. Cited on pages iv, vii, viii, 59, 61, 62, 63, 64, 70, 73, 77.

[LW] C. Löh and J. Witzig. *Universal finite functorial semi-norms*. 2022. arXiv: 2209.12971 [math.CT]. Cited on pages v, 7, 14, 15.

[Ml1] S. Mac Lane. *Homology*. Reprint of the 1975 edition. Springer-Verlag, 1995. Cited on pages 22, 25, 33.

[Ml2] S. Mac Lane. *Categories for the Working Mathematician*. Second Edition. Springer-Verlag, 1998. Cited on pages 21, 45, 47, 48, 50, 59.

[Ma] S. Marlow, ed. *Haskell 2010 Language Report*. Available as https://haskell.org/onlinereport/haskell2010/ or https://haskell.org/definition/haskell2010.pdf. Cited on page 30.

[Mo] N. Monod. *An invitation to bounded cohomology*. In: *International Congress of Mathematicians. Vol. II*. EMS Publishing House, 2006, pp. 1183–1211. URL: https://egg.epfl.ch/~nmonod/articles/icm.html. Cited on page vii.

[MP] J. P. May and K. Ponto. *More Concise Algebraic Topology. Localization, Completion, and Model Categories*. University of Chicago Press, 2012. Cited on page 77.

[MV] B. A. Munson and I. Volić. *Cubical Homotopy Theory*. Cambridge University Press, 2015. Cited on pages 65, 68.

[nL] nLab authors. *nLab*. https://ncatlab.org/nlab/show/HomePage, "a wiki for collaborative work on Mathematics, Physics, and Philosophy". Cited on page 55.

[Qu] D. Quillen. *Rational homotopy theory*. In: Ann. of Math. 90 (1969), pp. 205–295. Cited on page 21.

[Rap] G. Raptis. *Bounded cohomology and homotopy colimits*. 2021. arXiv: 2103.15614 [math.AT]. Cited on page 60.

[Rat] J. G. Ratcliffe. *Foundations of Hyperbolic Manifolds*. Second Edition. Springer, 2006. Cited on page 2.

[Ri1] E. Riehl. *Homotopy coherent structures*. 2018. arXiv: 1801.07404 [math.CT]. Cited on page 51.

[Ri2] E. Riehl. *Homotopical categories: from model categories to* $(\infty, 1)$*-categories*. 2020. arXiv: 1904.00886 [math.AT]. Cited on page 45.

[RV] E. Riehl and D. Verity. *The theory and practice of Reedy categories*. In: Theory Appl. Categ. 29 (2014), pp. 256–301. arXiv: 1304.6871 [math.CT]. Cited on pages 70, 71.

[Sm] The Sage Developers. *SageMath, the Sage Mathematics Software System (Versions 9.5–9.7)*. https://www.sagemath.org. 2022. Cited on page 31.

[Sh] M. A. Shulman. *Set theory for category theory*. 2008. arXiv: 0810.1279 [math.CT]. Cited on page 48.

[Su] D. Sullivan. *Infinitesimal computations in topology*. In: Inst. Hautes Études Sci. Publ. Math. 47 (1977), pp. 269–331. URL: http://www.numdam.org/item?id=PMIHES_1977__47__269_0. Cited on pages 17, 21, 26.

[Sw] R. M. Switzer. *Algebraic Topology – Homotopy and Homology*. Reprint of the 1975 original. Springer-Verlag, 2002. Cited on page 60.

[tD] T. tom Dieck. *Algebraic topology*. EMS Publishing House, 2008. Errata at https://www.uni-math.gwdg.de/tammo/algtop.html. Cited on page 57.

[Tho] R. Thom. *Quelques propriétés globales des variétés différentiables*. In: Comment. Math. Helv. 28 (1954), pp. 17–86. Cited on page 5.

[Thu] W. P. Thurston. *Geometry and Topology of Three-Manifolds*. Lecture notes, Princeton. Available at http://library.msri.org/books/gt3m/. 1980. Cited on page 2.

[Vo] R. M. Vogt. *Homotopy limits and colimits*. In: Math. Z. 134 (1973), pp. 11–52. Cited on page 55.

[We] C. A. Weibel. *An Introduction to Homological Algebra*. Cambridge University Press, 1995. Errata at https://sites.math.rutgers.edu/~weibel/Hbook-corrections.html. Cited on page 77.

[Wi] J. Witzig. *Abstract Excision and* ℓ^1*-Homology*. 2022. arXiv: 2203.06120v1 [math.AT]. Cited on page viii.