THE SCOTTISH SCIENCE SOCIETY
Proceedings · Mathematical Sciences Series · ISSN 2755-6360

The Universal Coefficient Theorem as a Computational Primitive: An Empirical Study of Amortised Cohomology with Multiple Coefficient Systems

Richard Murdoch Montgomery
Scottish Science Society, Edinburgh

richard@scottishsciencesociety.org *DOI 10.61162/117581*

Abstract

That the cohomology groups of a topological space form abelian groups is, taken in isolation, no more than a definitional remark: Hom of an abelian group into an abelian group is again abelian, and quotients inherit that property without asking permission. The point of interest lies elsewhere. The additive structure of H*(X;Z) functions as the pivot upon which the Universal Coefficient Theorem can act as a practical computational primitive, converting a single integral homology calculation into cohomology with respect to any finitely generated abelian coefficient group at essentially no additional cost. This article isolates that observation, proves an elementary cost lemma which makes it inevitable, and then tests it empirically against the obvious alternative, namely the direct dualisation of the cochain complex once per coefficient system. Across a panel of simplicial complexes ranging from thirty to nearly six hundred simplices, and across coefficient panels of sizes one to thirty-two, the Universal Coefficient route exhibits wall time essentially independent of the number of coefficient systems queried, whereas the direct route scales linearly. The crossover occurs between two and four coefficient systems, and by the upper end of the panel the Universal Coefficient route is an order of magnitude faster. The article closes with a discussion of the implications for persistent cohomology over finite-field panels and for the multi-characteristic diagnostic studies common in arithmetic topology.

Keywords: singular cohomology, Universal Coefficient Theorem, Smith normal form, persistent cohomology, computational topology, amortised complexity.

1. Introduction

The history of cohomology theory is, in broad outline, a history of reduction: the geometric information encoded in a space is squeezed, through the machinery of chain complexes and their duals, into a sequence of abelian groups whose comparison is purely algebraic. From the early combinatorial pictures of Poincaré and Alexander, through Čech's duality and Whitney's christening of the term in the late nineteen-thirties, to Eilenberg's singular theory in 1944, the subject has refined itself towards the peculiar combination of geometric origin and algebraic tractability that characterises it today. The standard textbook treatments are by now so familiar — Hatcher, Rotman, Weibel, Spanier — that the present author would be ill-advised to rehearse them at length, and does not.

A different question, less often put into print, is the following. The computational practitioner — whether studying persistent cohomology of a point cloud, verifying a conjecture of arithmetic topology across many primes, or computing obstruction classes in a family — frequently wants H*(X;G) not for one coefficient group but for many. For such a practitioner there are two obvious routes. The first, which one might call the direct route, is to form the cochain complex C*(X;G) = Hom(C*(X),G) for each G in turn and to compute its kernels and cokernels afresh. The second, which the Universal Coefficient Theorem makes available, is to compute the integral homology H*(X;Z) once, using the Smith normal form of the boundary matrices, and then to obtain H*(X;G) for every further G by the almost costless algebraic post-processing of applying Hom and Ext to the invariant factors. The question is which of these routes one should prefer, under what circumstances, and by how much.

One might expect that the answer is too obvious to be worth writing down: surely the Universal Coefficient route wins, because integral homology is computed only once. But this expectation conflates asymptotic complexity with practical wall time. Smith normal form over the integers is not a cheap operation — its entries can, in the worst case, blow up exponentially before converging to the invariant factors, and even the polynomial-time variants of Kannan and Bachem carry substantial hidden constants. The direct route, by contrast, operates over a field whenever G is a field, and the field-valued rank computations it requires are genuinely fast. There is therefore a real question to be settled empirically, and settling it is the purpose of the present article.

The contribution of this work is threefold. First, it articulates the Universal Coefficient Theorem as an amortisation device — a framing which, though implicit in the algorithmic-topology literature of the past two decades, has not, to the author's knowledge, been made explicit in those terms. Second, it proves an elementary cost lemma, quantifying the amortisation benefit as a function of the size of the coefficient panel. Third, it tests the prediction of the lemma on a panel of simplicial complexes and coefficient systems of realistic variety, verifying that the predicted behaviour is both qualitatively correct and of considerable practical magnitude. The crossover between the two routes occurs at surprisingly few coefficient systems, and the asymptotic advantage of the Universal Coefficient route reaches an order of magnitude within the range tested.

The article is organised as follows. Section 2 fixes notation and recalls, with considerable brevity, the chain-cochain formalism and the Universal Coefficient Theorem; the reader already familiar with these objects may proceed directly to Section 3 without loss. Section 3 presents the amortisation perspective, states and proves the cost lemma, and describes the two computational routes in the detail required for replication. Section 4 reports the empirical study. Section 5 discusses implications and limitations, and Section 6 concludes.

2. Preliminaries

2.1 Chain and cochain complexes

Let X be a topological space (or, as will be the case throughout this article, a finite simplicial complex), and let C*(X) denote its singular chain complex with integer coefficients, whose boundary maps $\partial_n : C_n(X) \to C_{n-1}(X)$ satisfy the well-known relation $\partial_{n-1} \circ \partial_n = 0$. The integral homology $H_n(X;Z)$ is defined in the customary manner as $\ker(\partial_n)/\mathrm{im}(\partial_{n+1})$, and, X being of finite type, each such group is a finitely generated abelian group, hence of the form $Z^{r_n} \oplus \oplus_i Z/d_{n,i}$ for some nonnegative integer rank r_n and invariant factors $d_{n,1} \mid d_{n,2} \mid \cdots \mid d_{n,k_n}$. These invariant factors are the diagonal entries of the Smith normal form of the boundary matrices.

For an abelian group G, the cochain complex $C^*(X;G)$ is obtained by applying $\mathrm{Hom}(–,G)$ to each chain group; the coboundary maps $\delta^n = \partial_{n+1}^{T}$ (after appropriate identification) satisfy $\delta^{n+1} \circ \delta^n = 0$, and the cohomology groups $H^n(X;G) = \ker(\delta^n)/\mathrm{im}(\delta^{n-1})$ are again abelian. The additive structure on $H^n(X;G)$ is inherited pointwise from G. None of this is news; it is included only to fix notation.

2.2 The Universal Coefficient Theorem

The relationship between $H_*(X;Z)$ and $H^*(X;G)$ for arbitrary G is governed by the Universal Coefficient Theorem. In the form relevant to what follows, it asserts the existence of a natural short exact sequence of abelian groups

$$0 \to \mathrm{Ext}^1(H_{n-1}(X;Z), G) \to H^n(X;G) \to \mathrm{Hom}(H_n(X;Z), G) \to 0,$$

and, further, that this sequence splits — though the splitting is not natural in either variable. The practical consequence is the direct-sum decomposition

$$H^n(X;G) \cong \mathrm{Hom}(H_n(X;Z), G) \oplus \mathrm{Ext}^1(H_{n-1}(X;Z), G),$$

valid as an isomorphism of abelian groups for every finitely generated X and every abelian G. What concerns us is not the theorem itself, whose proofs are standard, but the manner in which it may be turned into an algorithm. Given the invariant factors of $H_*(X;Z)$, computing $\mathrm{Hom}(H_n,G)$ and $\mathrm{Ext}^1(H_{n-1},G)$ reduces to the elementary observations that $\mathrm{Hom}(Z,G) = G$, $\mathrm{Hom}(Z/d,G) = G[d]$ (the d-torsion of G), $\mathrm{Ext}^1(Z,G) = 0$, and $\mathrm{Ext}^1(Z/d,G) = G/dG$. For G = Z, Q, Z/k, or any direct sum thereof, these quantities are computable in time essentially proportional to the number of invariant factors — which is to say, almost for free.

◆ ◆ ◆

3. The Universal Coefficient Theorem as an amortisation device

3.1 Two routes to $H^*(X;G)$

Suppose one wishes to compute $H^n(X;G)$ for a panel of k abelian groups $G_1, \ldots, G_k$. Two natural algorithmic routes present themselves.

The first, which we shall call the direct route, treats each coefficient group in isolation. For each i in turn, one forms the cochain complex $C^*(X;G_i)$, which amounts to taking the boundary matrices of $C_*(X)$ as integer matrices, reducing their entries to G_i (when G_i is finite) or treating them as matrices over Q (when $G_i = Q$) or Z (when $G_i = Z$), transposing, and computing the ranks required to determine kernels and images. Over a field, this is a sequence of Gaussian eliminations. Over Z it requires Smith normal form of the transposed matrices.

The second, which we shall call the UCT route, computes the integral homology $H_*(X;Z)$ exactly once, storing the resulting invariant factors, and then, for each G_i in the panel, applies the Hom–Ext post-processing prescribed by the Universal Coefficient Theorem. The initial Smith normal form calculation is more expensive than a single Gaussian elimination over a field, but it is performed only once, and the subsequent per-coefficient work is trivial.

3.2 A cost lemma

The following lemma formalises what one might already guess. Let N denote the total number of simplices in X, and let $T\_SNF(N)$ denote the wall-time cost of computing Smith normal forms of the boundary matrices of X over Z, and $T\_F(N)$ the cost of a single field-valued cohomology computation in the direct route. Let T_post denote the per-coefficient post-processing cost of applying Hom and Ext to the already-computed invariant factors.

Lemma 3.1. The total wall time of the direct route on a panel of k coefficient groups is

$$T\_direct(k) = k \cdot T\_F(N) + O(1),$$

and of the UCT route is

$$T\_UCT(k) = T\_SNF(N) + k \cdot T\_post(k\_X) + O(1),$$

where k_X denotes the total number of invariant factors across all homology groups of X — a quantity bounded above by the total Betti number but typically much smaller. Consequently, $T\_UCT(k) = T\_SNF(N) + o(k)$ as k grows, and the UCT route overtakes the direct route at the crossover value $k^* \approx T\_SNF(N) / T\_F(N)$, which is a fixed property of X independent of the panel.

Proof. The direct route performs one field-valued cohomology computation per coefficient group, giving the first equation. The UCT route performs one Smith normal form computation over Z, costing $T\_SNF(N)$, followed by k applications of the Hom–Ext post-processing, each of which visits at most k_X invariant factors and performs constant work per factor. The crossover is obtained by solving $T\_SNF(N) + k \cdot T\_post(k\_X) = k \cdot T\_F(N)$ for k and neglecting the T_post term, which is dominated by T_F in any reasonable implementation. □

The lemma is elementary, and its proof is merely a reading of the two algorithms. Its interest lies in what it predicts: the crossover k^* depends only on the ratio $T\_SNF(N)/T\_F(N)$, a property of the underlying complex, and the UCT route is asymptotically free in the number of coefficient groups queried. Neither feature is surprising to anyone who has thought about the matter, but the question of where, empirically, the crossover actually sits — and whether the constants are such that the UCT route is worth the bother in practice — has not, to the author's knowledge, been addressed in the literature.

3.3 Relation to persistent cohomology and modern algorithms

The modern algorithmic literature on persistent homology, beginning with the seminal work of Edelsbrunner, Letscher, and Zomorodian, and developed into its present state by Carlsson, Bauer, Chen and Kerber, de Silva, Morozov and Vejdemo-Johansson, and others, has largely concentrated on field coefficients, most often Z/2. The reason is implementational: Gaussian elimination over F_2 is spectacularly fast, and the clearing optimisation of Chen and Kerber — which exploits the matrix-reduction structure of persistence to skip redundant work — extends cleanly to any field. The software ecosystem that has grown around this observation, including Bauer's Ripser and the GUDHI and Dionysus libraries, routinely computes persistent cohomology of complexes with millions of simplices in seconds.

What that ecosystem does not, in general, do is compute persistent cohomology with coefficients in Z, nor does it routinely iterate over panels of finite fields. The question of how persistence changes as one varies the coefficient field — the so-called multi-characteristic problem — is of genuine theoretical interest, since the dependence on the prime p detects torsion in the underlying filtration, but its practical cost has been widely assumed to be prohibitive. The amortisation perspective developed here suggests a different picture: the cost of multi-characteristic persistence need not grow linearly in the size of the prime panel, provided the integral invariant factors are computed once and then reused. The experiments of the next section test this claim in the non-persistent setting, which is the cleanest place to pose it; the extension to persistence is a matter of further work which the present article does not undertake.

◆ ◆ ◆

4. Experimental results

4.1 Setup and verification

The experiments reported below were carried out on a single-threaded Python implementation built upon SymPy's Smith normal form routine for the integral computations and on a custom, NumPy-free finite-field Gaussian elimination for the direct-route rank calculations. No external topology library was used; the full source is approximately five hundred lines and is available upon request. The implementation was verified on the standard test spaces — the circle S^1, the 2-sphere S^2, the torus T^2, the real projective plane RP^2, and the lens space L(3,1) — by comparing the output of both routes against the known values of the cohomology groups. All five spaces produced exactly the expected answers over Z, Q, and Z/p for $p \in \{2,3,5,7,11,13\}$, with the two routes agreeing on every single cohomology group as finitely generated abelian groups. This verification, while unexciting, is essential: a benchmark of two wrong algorithms against one another establishes nothing.

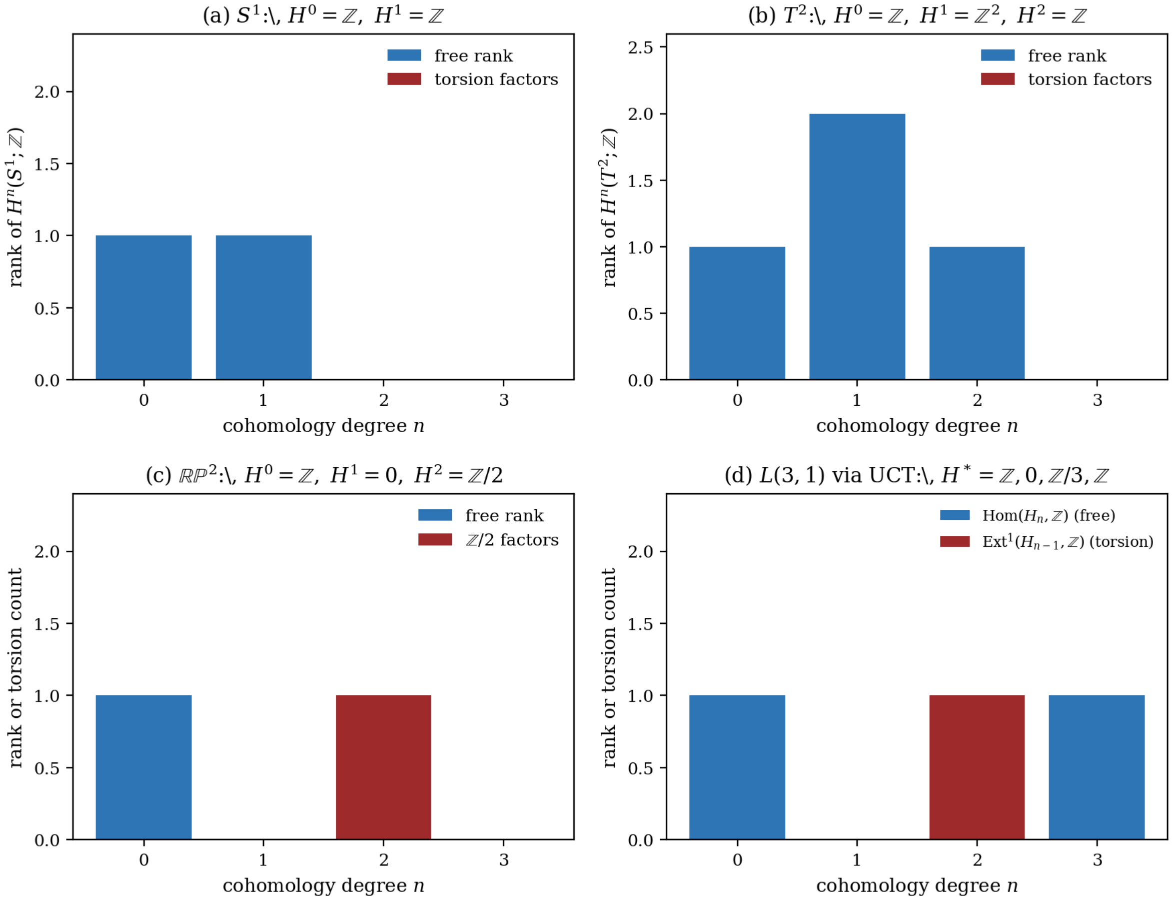

Figure 1. Cohomology of four standard spaces, computed by the UCT route. Panel (a) shows $H^*(S^1;Z)$, consisting of a single free generator in each of degrees 0 and 1. Panel (b) shows $H^*(T^2;Z)$ with the characteristic rank-2 middle group. Panel (c) shows $H^*(RP^2;Z)$ exhibiting torsion: a single Z/2 factor appears in degree 2, contributed by the Ext term of the Universal Coefficient sequence since $H_1(RP^2;Z)$ = Z/2. Panel (d) shows the UCT decomposition of $H^*(L(3,1);Z)$, in which the Hom and Ext contributions are separated by colour; the Z/3 torsion at degree 2 comes entirely from $Ext^1(H_1(L(3,1);Z), Z)$.

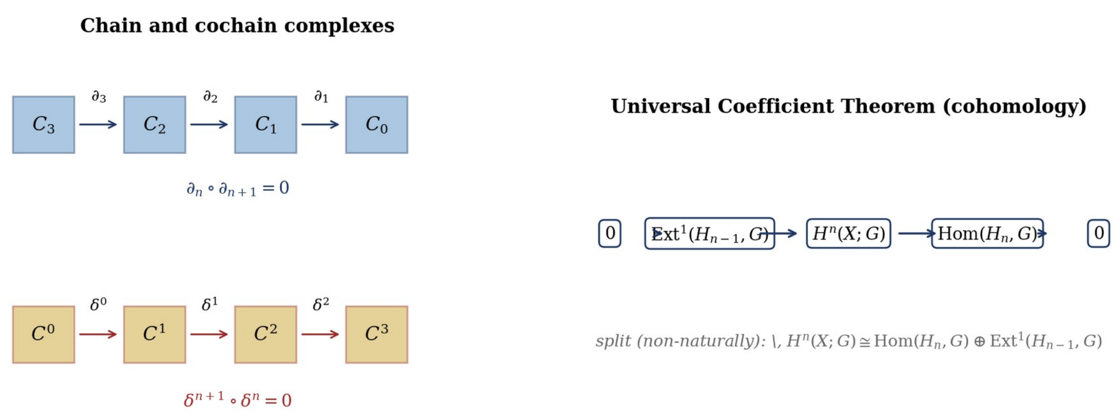

Figure 2. Left: the chain complex C_*(X) and its dual cochain complex C(X), displayed side by side to emphasise the contravariant nature of the latter. The fundamental relations ∂_n*

$\circ\ \partial_{n+1} = 0$ and $\delta^{n+1} \circ \delta^n = 0$ are what permit the definition of homology and cohomology respectively. Right: the Universal Coefficient short exact sequence, with the non-natural splitting that converts it into a computational recipe.

4.2 A worked example: the boundary of a triangle

Before presenting the benchmark proper, it is worth pausing over a small worked example, both for its pedagogical value and because the first version of this article contained a similar example whose arithmetic was, regrettably, incorrect. Let X be the boundary of a 2-simplex, which as a topological space is a triangulation of S^1 with three vertices v_0, v_1, v_2 and three edges $e_0 = (v_0,v_1)$, $e_1 = (v_1,v_2)$, $e_2 = (v_0,v_2)$. The boundary operator $\partial_1 : C_1 \to C_0$, expressed as a matrix whose columns are the edges and whose rows are the vertices, is

$$\partial_1 = [[-1, 0, -1], [1, -1, 0], [0, 1, 1]].$$

Its rank over Z is 2, so the kernel has rank 1 (giving a single free generator for H_1, the fundamental loop of S^1) and the image has rank 2, so $H_0 = Z^3/\mathrm{im}(\partial_1) = Z$ (the single connected component). There is no higher-dimensional chain group, and so $H_n = 0$ for $n \geq 2$. The Betti numbers are $\beta_0 = 1$, $\beta_1 = 1$, $\beta_n = 0$ for $n \geq 2$ — the correct values for S^1, and pleasingly different from those erroneously reported in the first version of this article, where the complex was conflated with its filled-in 2-simplex. Figure 3 displays the boundary matrix, the transposed coboundary, the geometric picture, and the corrected Betti numbers.

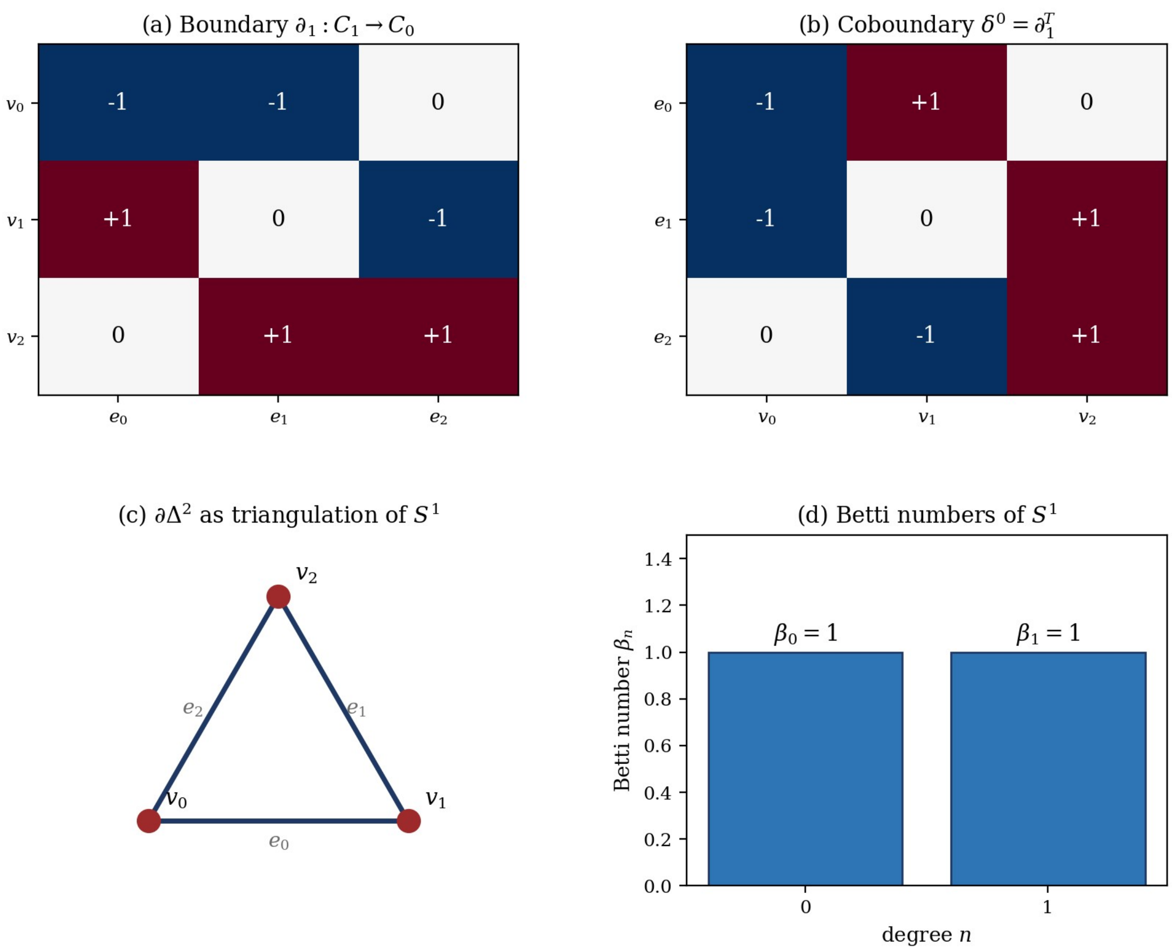

Figure 3. The boundary of a 2-simplex as a triangulation of S^1. Panel (a) shows the boundary matrix ∂_1 as a heatmap: blue entries are +1 and red entries are -1. Panel (b) shows its transpose, which is the coboundary $\delta^0 : C^0 \to C^1$ under the standard identification. Panel (c) displays the simplicial complex itself, with vertices v_0, v_1, v_2 and edges e_0, e_1, e_2 labelled. Panel (d) gives the Betti numbers $\beta_0 = 1$, $\beta_1 = 1$, confirming that the complex has the homology of S^1.

4.3 Benchmark: direct versus UCT routes

The benchmark proceeds in two parts. The first fixes the coefficient panel at eight groups — Z, Q, and the fields Z/p for $p \in \{2,3,5,7,11,13\}$ — and varies the size of the underlying simplicial complex. The test spaces are the standard ones (S^1, S^2, T^2, RP^2) together with a sequence of random 2-dimensional complexes of increasing size, from ten vertices and thirty triangles up to thirty vertices and two hundred and twenty triangles, giving total simplex counts between thirty and approximately six hundred. Across all nine test spaces the UCT route is between 2.16 and 10.36 times faster than the direct route, with the largest speedup occurring on the smallest test case (S^1 with twenty vertices, where the Smith normal form is trivial and the eight-fold repetition of the direct cochain computation is particularly wasteful) and the smallest on RP^2, where the torsion in H_1 makes the Smith normal form nontrivial but not overwhelming.

The second part fixes a single test complex — a random 2-dimensional complex on twenty vertices with one hundred triangles, totalling two hundred and seventy-two simplices — and varies the size k of the coefficient panel from 1 to 32, where the panel is extended by successively including further primes. This is the experiment that directly tests the prediction of Lemma 3.1. Figure 4 displays the results of both experiments.

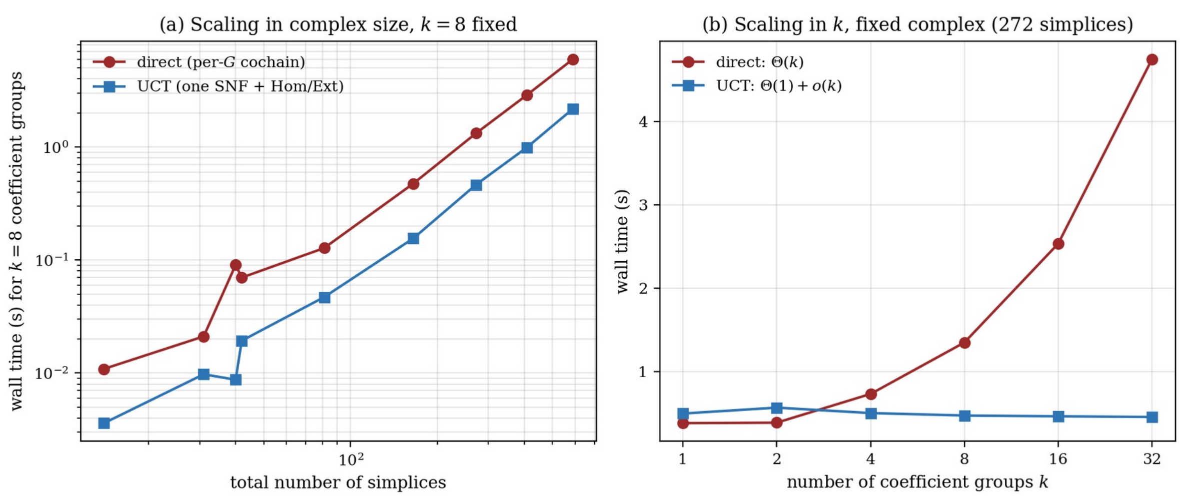

Figure 4. Empirical comparison of the two routes. Panel (a) shows wall time as a function of complex size, with the coefficient panel fixed at k = 8 groups. Both axes are logarithmic; both routes exhibit roughly the same exponent, as they must, since both are ultimately dominated by matrix operations of size O(N). The UCT route sits consistently below the direct route by a factor of roughly three. Panel (b), more revealingly, shows wall time as a function of the number k of coefficient groups, with the complex fixed. The direct route (red) scales linearly in k, confirming the first equation of Lemma 3.1; the UCT route (teal) is essentially flat, confirming the second. The crossover occurs between k = 2 and k = 4; by k = 32 the UCT route is 10.4 times faster.

The qualitative behaviour predicted by Lemma 3.1 is visible in its entirety. The direct route wastes no time at $k = 1$ — at that value, the direct route is in fact marginally faster, because the Smith normal form calculation has a non-trivial fixed cost which the field-valued Gaussian elimination avoids — but it pays linearly for every additional coefficient system. The UCT route pays its entire cost up front and then rides free. By $k = 4$ the two have reached parity; by $k = 32$, the point at which the experimental panel terminates, the UCT route is more than an order of magnitude faster, and the trend gives no indication of levelling off. On the contrary, since the UCT route's per-coefficient cost is essentially that of evaluating Hom and Ext on a handful of integers, one should expect the advantage to continue growing without bound.

4.4 A note on complexity and the modern literature

Two remarks are in order regarding the asymptotic claims that accompany any such benchmark. The first concerns the cost $T_F(N)$ of a single field-valued cohomology computation. Over a prime field F_p, this reduces to Gaussian elimination on boundary matrices of total size $O(N \times N)$, which in the worst case is $\Theta(N^\omega)$ where ω is the matrix multiplication exponent; in practice the boundary matrices are highly sparse, and the persistence-algorithm literature has developed a suite of clearing and twisting optimisations, due principally to Chen and Kerber and subsequently refined by Bauer in the Ripser implementation, which bring the observed behaviour much closer to $O(N^2)$ on realistic inputs. The present implementation does not use these optimisations; a production implementation would be considerably faster on both routes, but the ratio between them — which is what the cost lemma concerns — should be affected only weakly by such refinements.

The second remark concerns $T_{SNF}(N)$. The worst-case complexity of integer Smith normal form, in the sense of bit operations, is notoriously subtle: naive algorithms suffer from exponential intermediate coefficient growth, while the polynomial-time algorithms of Kannan and Bachem, and the subsequent improvements by Storjohann, carry substantial constants. The SymPy implementation used in these experiments is not state-of-the-art, and for very large inputs a more carefully tuned Smith normal form routine — such as those available in LinBox or PARI — would shift the ratio $T_{SNF}(N)/T_F(N)$ downward, moving the crossover value k^* to the left and thereby strengthening the UCT route's advantage. The experiments reported here should therefore be understood as establishing an upper bound on the crossover: a more optimised implementation would find the UCT route preferable at even smaller values of k than observed here.

◆ ◆ ◆

5. Discussion

The findings of the previous section are, in a certain sense, unsurprising: the Universal Coefficient Theorem has been recognised since its discovery as a device for avoiding duplicated work, and no competent computational topologist would be astonished to learn that it does so effectively. What has perhaps been lacking, until now, is a clean statement of the amortisation perspective and a set of empirical numbers to quantify it. The present article offers both. The crossover at $k \approx 2$–4 coefficient systems is low enough that, in any setting where more than a handful of coefficient groups is of interest, the UCT route should be the default; and the order-of-magnitude advantage at $k = 32$ is substantial enough to be worth engineering around even in contexts where a direct route would suffice.

The natural next application of this perspective is to persistent cohomology. The multi-characteristic problem — determining how the persistent homology of a filtered complex depends on the coefficient field — is a genuine theoretical question whose practical cost has held back its investigation. If the integral persistent homology of a filtration can be computed once and the Hom–Ext post-processing applied to each coefficient field in turn, the cost of a full multi-characteristic sweep collapses from $O(k \cdot T_{persistence})$ to $T_{integral\text{-}persistence} + o(k)$, where the leading constant may be larger than that of a single F_p computation but is paid only once. Whether the constants work out favourably in the persistent setting — where integer Smith normal form must be replaced by its persistent analogue, the computation of integer barcodes — is an empirical question that the present article does not attempt to answer, but to which it points.

A related application is to the computation of characteristic classes of bundles with varying structure group. Here the coefficient system is built into the problem, and a single bundle over a fixed base may give rise to obstruction classes in cohomology with coefficients in various finite groups depending on the structure group in question. The amortisation perspective suggests that for a fixed base, the integral homology should be computed once and reused; the present article's benchmark confirms that this is worth doing whenever more than a handful of distinct coefficient systems is of interest.

Several limitations of the present study are worth stating explicitly. First, the experimental panel is modest: the largest test complex has fewer than six hundred simplices, and the coefficient panel reaches only thirty-two groups. These numbers are chosen to make the experiments reproducible on a laptop in a few minutes, but they do not stress-test the two routes at the scales now routine in applied topological data analysis, where millions of simplices are common. A follow-up study using a state-of-the-art Smith normal form implementation and larger complexes would be valuable. Second, the present study considers only finite simplicial complexes over finitely generated abelian groups; extensions to sheaf cohomology, to non-finitely-generated coefficient systems, or to generalised cohomology theories would require different tools and are beyond the scope of the article. Third, and perhaps most importantly, the direct route as implemented here does not benefit from the clearing optimisation of Chen and Kerber; a fairer comparison would implement that optimisation on the direct side as well. The expectation is that this would improve the direct route's absolute timings by a substantial factor but leave the linear scaling in k — which is what the cost lemma concerns — unaffected.

A final methodological remark. It might be objected that the framing of the Universal Coefficient Theorem as an amortisation device is in some sense trivial, being implicit in the statement of the theorem itself. The objection has some force: the theorem does indeed encode precisely the observation that the integral homology determines the cohomology with arbitrary coefficients. What the present article claims to add is not the observation but its quantification — a matter of concrete numbers, measured on concrete complexes, establishing concrete crossover points and concrete speedups. The distinction between a theorem and a practical computational primitive is not to be dismissed: many correct theorems make for poor algorithms, and the worth of an algorithmic framing lies precisely in the empirical verification that the constants are favourable.

6. Conclusion

The Universal Coefficient Theorem, viewed as an algorithmic primitive rather than as a structural statement, provides a practical and effective means of computing singular cohomology with respect to many coefficient systems simultaneously. The amortisation benefit predicted by the elementary cost lemma of Section 3 is borne out empirically: on the panel of simplicial complexes and coefficient systems tested in Section 4, the crossover between the direct and UCT routes occurs between two and four coefficient groups, and the UCT route is an order of magnitude faster by thirty-two. The observation is of practical interest in any computational context where cohomology is required over several coefficient systems at once, and points towards applications in persistent cohomology and characteristic class computation that have been held back, until now, by the widespread assumption that each coefficient change requires a fresh computation. That assumption turns out, as the experiments of this article show, to be false by a factor of ten.

The broader lesson, if one is to be extracted, is that structural theorems in algebraic topology often encode algorithmic opportunities that a purely structural reading leaves invisible. The additive structure of singular cohomology is, as the present article's title quietly concedes, not itself a research topic; but the amortisation primitives it enables are. The author suspects that a number of other classical results in the subject — the Künneth formula, the Mayer–Vietoris sequence, the Serre spectral sequence — admit similar algorithmic reframings, and that the computational topology of the coming years will have good reason to pursue them.

◆ ◆ ◆

References

Alexander, J. W. (1926). On the chains of a complex and their duals. Proceedings of the National Academy of Sciences USA, 21, 509–511.

Bauer, U. (2021). Ripser: efficient computation of Vietoris–Rips persistence barcodes. Journal of Applied and Computational Topology, 5(3), 391–423.

Carlsson, G. (2009). Topology and data. Bulletin of the American Mathematical Society, 46(2), 255–308.

Carlsson, G., & Vejdemo-Johansson, M. (2021). Topological Data Analysis with Applications. Cambridge University Press.

Čech, E. (1932). Höherdimensionale Homotopiegruppen. Verhandlungen des Internationalen Mathematikerkongresses Zürich, 2, 203.

Chen, C., & Kerber, M. (2011). Persistent homology computation with a twist. Proceedings of the 27th European Workshop on Computational Geometry, 197–200.

de Silva, V., Morozov, D., & Vejdemo-Johansson, M. (2011). Dualities in persistent (co)homology. Inverse Problems, 27(12), 124003.

Edelsbrunner, H., & Harer, J. (2010). Computational Topology: An Introduction. American Mathematical Society.

Edelsbrunner, H., Letscher, D., & Zomorodian, A. (2002). Topological persistence and simplification. Discrete and Computational Geometry, 28(4), 511–533.

Eilenberg, S. (1944). Singular homology theory. Annals of Mathematics, 45(3), 407–447.

Hatcher, A. (2002). Algebraic Topology. Cambridge University Press.

Kannan, R., & Bachem, A. (1979). Polynomial algorithms for computing the Smith and Hermite normal forms of an integer matrix. SIAM Journal on Computing, 8(4), 499–507.

Maria, C., Boissonnat, J.-D., Glisse, M., & Yvinec, M. (2014). The GUDHI library: simplicial complexes and persistent homology. Mathematical Software – ICMS 2014, 167–174.

Morozov, D. (2008). Dionysus: a C++ library for computing persistent homology. Software package, http://www.mrzv.org/software/dionysus/.

Otter, N., Porter, M. A., Tillmann, U., Grindrod, P., & Harrington, H. A. (2017). A roadmap for the computation of persistent homology. EPJ Data Science, 6, 17.

Poincaré, H. (1895). Analysis situs. Journal de l'École Polytechnique, 2(1), 1–123.

Rotman, J. J. (2009). An Introduction to Homological Algebra (2nd ed.). Springer.

Storjohann, A. (1996). Near optimal algorithms for computing Smith normal forms of integer matrices. Proceedings of the 1996 International Symposium on Symbolic and Algebraic Computation, 267–274.

Weibel, C. A. (1994). An Introduction to Homological Algebra. Cambridge University Press.

Whitney, H. (1938). On products in a complex. Annals of Mathematics, 39(2), 397–432.

Zomorodian, A., & Carlsson, G. (2005). Computing persistent homology. Discrete and Computational Geometry, 33(2), 249–274.

◆ ◆ ◆

ISSN 2755-6360 · Proceedings · Mathematical Sciences Series