CONSTRUCTING A V_2 SELF MAP AT $P = 3$

by

BENJAMIN WILLIAM REID

A DISSERTATION

Presented to the Department of Mathematics
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

June 2017

DISSERTATION APPROVAL PAGE

Student: Benjamin William Reid

Title: Constructing a v_2 Self Map at $p = 3$

This dissertation has been accepted and approved in partial fulfillment of the requirements for the Doctor of Philosophy degree in the Department of Mathematics by:

Hal Sadofsky	Chair
Boris Botvinnik	Core Member
Daniel Dugger	Core Member
Victor Ostrik	Core Member
Santiago Jaramillo	Institutional Representative

and

Scott L. Pratt	Dean of the Graduate School

Original approval signatures are on file with the University of Oregon Graduate School.

Degree awarded June 2017

DISSERTATION ABSTRACT

Benjamin William Reid

Doctor of Philosophy

Department of Mathematics

June 2017

Title: Constructing a v_2 Self Map at $p = 3$

Working at the prime $p = 3$, we construct a stably finite spectrum, Z, with a v_2^1 self map f. Further, both $\mathrm{Ext}_A(H^*(Z), \mathbb{Z}_3)$ and $\mathrm{Ext}_A(H^*(Z), H^*(Z))$ have a vanishing line of slope $1/16$ in $(t-s, s)$ coordinates, and the map f is represented by an element α of Ext where multiplication by α is parallel to the vanishing line. To accomplish this construction, we prove a result about the connection between particular self maps of spectra and their effect on the Margolis homology of related modules over the Steenrod Algebra.

CURRICULUM VITAE

NAME OF AUTHOR: Benjamin William Reid

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, OR
Virginia Polytechnic Institute and State University, Blacksburg, VA

DEGREES AWARDED:

Doctor of Philosophy, Mathematics, 2017, University of Oregon
Bachelor of Science, Mathematics, 2011, Virginia Polytechnic Institute and State University
Bachelor of Science, Computer Science, 2010, Virginia Polytechnic Institute and State University

AREAS OF SPECIAL INTEREST:

Algebraic Topology
Stable Homotopy Theory

PROFESSIONAL EXPERIENCE:

Graduate Teaching Fellow, University of Oregon, 2011-2017

PUBLICATIONS:

Oleg Lazarev, Matt Mizuhara, Holly Swisher, Benjamin Reid, *Extension of a proof of the Ramanujan congruences for multipartitions*, Ramanujan Journal, 2016.

To Taylor, for her love and constant encouragement.

To my parents, for always believing in me.

To Hal, for his patience and advice.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Motivation and the Nilpotence Theorem

The Nilpotence Theorem [DHS88] identifies certain properties that a self map on a spectrum X, $f : \Sigma^d X \to X$, must have in order to be non-nilpotent. In particular, these maps are detected by the Morava K theories [Rav16], a set of related cohomology theories for each prime p. For a fixed prime p, the theories are indexed by the integers $n \geq 0$ and denoted $K(n)$ for each n.

Definition 1.1. A p-local, finite spectrum X is called *type* n if n is the smallest integer such that $K(n)_*(X)$ is nontrivial.

Furthermore, every p-local, finite spectrum is type n for some n. On these spectra, we define a particular type of self map based on the Morava K theories. The following definition comes from [Rav16].

Definition 1.2. For a type n spectrum, X, we define a v_n *self map* to be a map $f : \Sigma^d X \to X$ such that $K(n)_*(f)$ is an isomorphism, and $K(m)_*(f)$ is trivial for $m \neq n$.

Some power of any such map induces multiplication by some power of v_n in $K(n)$ homology. We say that a v_n^i map on X is a v_n self map that induces multiplication by v_n^i in $K(n)$ homology.

The Periodicity Theorem [HS98] tells us that such type n spectra and v_n maps exist for each n. In addition, every type n spectrum has a v_n^i self map for some power $i \geq 0$. This theorem does not give a construction for the map or the power i.

Consider the mod p Moore spectrum, $M(p)$, which is a type 1 spectrum. In [Ada66], Adams showed that $M(2)$ admits a v_1^4 map, but no smaller power, and that for $p \geq 3$, $M(p)$ has a v_1^1 map. Taking the cofiber of a v_n map gives a type $n+1$ spectrum, which has a v_{n+1} map. For $p \geq 5$, the cofiber of $v_1 : \Sigma^{2(p-1)}M(p) \to M(p)$ has a v_2^1 map [Smi70]. At $p = 3$, the cofiber of our v_1 map has a v_2^9 map instead [BP04]. We construct a related finite spectrum that has a v_2^1 map instead.

Theorem 1.3. *There exists a p-local, finite spectrum, Z_f, with a map $v_2^1 : \Sigma^{16}Z_f \to Z_f$.*

One application of the Nilpotence theorem to these v_n maps is outlined in Theorem 9 of [Hop87]. For $n > 0$, and X a type n spectrum, there is a map

$$f : \text{Center } [X, X]_* \to \mathbb{Z}_p[v_n]$$

such that $\ker(f)$ consists of nilpotent elements, and given any element $b \in \mathbb{Z}_p[v_n]$, then $b^j \in \operatorname{im}(f)$ for some j. This map f is induced by the $K(n)$ Hurewicz homomorphism. This tells us that the v_n self maps on X generate the center of $[X, X]$ modulo nilpotents.

Another application of these v_n self maps on finite spectra can is that they can give us examples of periodic families of stable homotopy elements. Given a finite spectrum X with a v_n self map $f : \Sigma^d X \to X$, and two maps $i : S^\ell \to X$ and $j : X \to S^t$, we can construct a stable homotopy element from the composition

$$S^{d+\ell} \xrightarrow{\Sigma^d i} \Sigma^d X \xrightarrow{f} X \xrightarrow{j} S^i$$

In fact, we can create a family of related, possibly nonzero stable homotopy elements by iterating f. As an example, iterating f twice would give us the element:

$$S^{2d+\ell} \xrightarrow{\Sigma^{2d} i} \Sigma^{2d} X \xrightarrow{\Sigma^d f} \Sigma^d X \xrightarrow{f} X \xrightarrow{j} S^i$$

We use a technique from [PS94] to construct our example of a finite spectrum and a corresponding v_2 map. An advantage of using this technique is that along the way, we construct a spectrum which has a vanishing line in the E_2 page of the Adams spectral sequence that is, in some sense, parallel to v_2. As in [Mil81] for a v_1 map, if we are able to compute this E_2 page in a band around the vanishing line, that may lead to information about the homotopy of the mapping telescope $v_2^{-1} Z_f$

For this construction, and to ensure the vanishing line in Ext behaves in the expected manner, we will be working with modules over the Steenrod Algebra, and computing Margolis homology groups for these modules. We give a little background about these topics below.

The Steenrod Algebra and its Dual

Recall from [Ste62] that the mod 3 Steenrod Algebra, which we will denote A, is the Hopf algebra of stable mod 3 cohomology operations. It is generated, as an algebra over $\mathbb{Z}_3$, by the Bockstein operation (β, in degree 1), and the reduced p-th power operations (P^i, in degree $4i$), modulo $P^0 = 1, \beta^2 = 0$ and the Adem relations given below:

$$P^a P^b = \sum_i (-1)^{a+1} \binom{2(b-i)-1}{a-3i} P^{a+b-i} P^i$$

for $a < 3b$, and

$$P^a\beta P^b = \sum_i (-1)^{a+i}\binom{2(b-i)}{a-3i}\beta P^{a+b-i}P^i + \sum_i (-1)^{a+i+1}\binom{2(b-i)-1}{a-3i-1}P^{a+b-i}\beta P^i$$

for $a \leq 3b$.

Given these relations, we can write any monomial in A in the form

$$\beta^{e_0}P^{s_1}\beta^{e_1}\cdots P^{s_k}\beta^{e_k}$$

where $e_i \in \{0,1\}$, and $s_i \in \{0,1,2,\ldots\}$. We call the monomial admissible if $s_i \geq 3s_{i+1} + e_i$. These admissible monomials form a basis for A as a $\mathbb{Z}_3$ vector space, that is, every element of A can be written as a $\mathbb{Z}_3$ linear combination of admissible monomials.

The dual of the Steenrod algebra, A_*, is also a Hopf algebra, with a simpler multiplicative structure. Specifically, A_* is a tensor product of a polynomial algebra on generators ξ_k of degree $2 \cdot 3^k - 2$, and an exterior algebra on generators τ_k of degree $2 \cdot 3^k - 1$.

$$A_* = \mathbb{Z}_3[\xi_1, \xi_2, \ldots] \otimes E(\tau_0, \tau_1, \ldots)$$

Dualizing the basis for A_* given by monomials in the $\{\xi_i\}$ and $\{\tau_j\}$ gives us another basis for A, called the Milnor basis. It contains elements Q_i dual to τ_i, and P_t^s dual to $\xi_t^{3^s}$.

Margolis Homology

Given a nilpotent element x of A, we can view x as a differential on any A module. If $x^n = 0$ and M is an A module, then we can define the homology of M with respect to x by:

$$H_*(M; x) = \frac{\ker(x^{n-1})}{\operatorname{im}(x)}$$

Remark 1.4. We note that there are other ways we could define the x homology of a module M, depending on the value of n. For example, we could flip the powers and take

$$H'_*(M; x) = \frac{\ker(x)}{\operatorname{im}(x^{n-1})}$$

Given a short exact sequence of A modules, we get a long exact sequence in homology with respect to x. If $n = 2$, then this long exact sequence behaves as we expect. However, if $n > 2$, then our long exact sequence will alternate three terms of H_* with three terms of H'_*.

As in [Mar83], we focus on particular nilpotent elements of A when defining this type of homology. Specifically, we consider the elements Q_n and P^s_t with $s < t$ from the Milnor basis of A. We know from [Mar83] that for these elements, we have $(Q_n)^2 = 0$ and $(P^s_t)^3 = 0$. We use the following notation in this paper:

Definition 1.5. Let M be an A module. For elements $Q_n \in A$ and $P^s_t \in A$ with $s < t$ define:

$$H_*(M; Q_n) = \frac{\ker(Q_n)}{\operatorname{im}(Q_n)}, \qquad H_*(M; P^s_t) = \frac{\ker((P^s_t)^2)}{\operatorname{im}(P^s_t)}$$

In some sense, the Margolis homology of a bounded below A module M with respect to these particular differentials gives us a way to measure how close M is to being free over A. From [MW81], we have the following results:

- If $H_*(M; P_t^s) = 0$ for all $P_t^s \in A$ with $s < t$, and $H_*(M; Q_t) = 0$ for all $Q_t \in A$, then M is free over A.

- If M is of finite type, $H_*(M; P_t^s) = 0$ for all $P_t^s \in A$ with $s < t$ and $p|P_t^s| < 2d$, and $H_*(M; Q_t) = 0$ for all $Q_t \in A$ with $|Q_t| < d$, then $\mathrm{Ext}_A^{s,t}(M, M)$ has a vanishing line of slope d.

u_k Maps for A Modules

If x is either a Q_n or P_t^s differential, then we define $C(x) \subseteq A$ to be the sub Hopf algebra generated by x. If $x = Q_n$, then $C(x) = E(x)$, an exterior algebra, and we can compute

$$\mathrm{Ext}_{C(x)}^{*,*}(\mathbb{Z}_3, \mathbb{Z}_3) = \mathbb{Z}_3[u]$$

The polynomial generator, u, is in bidegree $(1, |x|)$

Similarly, if $x = P_t^s$ with $s < t$, then $C(x) = \mathbb{Z}_3[x]/x^3$, a truncated polynomial algebra, and we can compute

$$\mathrm{Ext}_{C(x)}^{*,*}(\mathbb{Z}_3, \mathbb{Z}_3) = E[y] \otimes \mathbb{Z}_3[u]$$

Here, the exterior generator, y, is in bidegree $(1, |x|)$, and the polynomial generator, u, is in bidegree $(2, 3|x|)$.

These computations give rise to the notion of “slope” for these differentials, corresponding to multiplication in (s,t) coordinates by the polynomial generators u in both cases. More precisely, we have

Definition 1.6. For each of these differentials Q_n and P^s_t with $s<t$, we define the *slope* via the following formulas:

$$s(Q_n) = |Q_n|, \qquad \text{and} \qquad s(P^s_t) = \frac{3|P^s_t|}{2}$$

The Q_n and P^s_t differentials can be linearly ordered by slope, and we let x_k denote the kth differential in this ordering. The first few of these differentials are given in Table 1. below.

TABLE 1. The first four differentials, ordered by slope

k	x_k	element	$s(x_k)$
0	x_0	Q_0	1
1	x_1	Q_1	5
2	x_2	P^0_1	6
3	x_3	Q_2	17

In computing Ext groups of A modules, we make use of the following lemma from [Mar83, Theorem 19.7] about the x_k differentials:

Lemma 1.7. *Suppose M is a bounded below A module, with bottom degree m, such that $H_*(M;x_i) = 0$ for $i \leq k$. Let B be the sub Hopf algebra of A containing $\{x_0, x_1, \ldots, x_k\}$, and let d be the bottom nonzero degree of A/B. Then M is free through degree $d+m-1$. That is, there is an surjection from a free A module to M such that the bottom degree of the kernel is in degree at least $d+m$.*

Specifically, if we are constructing a minimal resolution for a bounded below A module, M, then the bottom degree of the first stage of the resolution will be at least $d + m$. Since free A modules have no x_i homology for any i, there is an isomorphism between the x_i homology of M and of the kernel of the surjection. In particular, the x_i homology of the kernel vanishes for $i \leq k$, so the kernel is also free through degree $2d+m-1$, and therefore the bottom degree of the second stage of the resolution must be at least $2d+m$. We can repeat this process inductively to see that the bottom degree of each stage of the resolution must increase by at least d.

We use this fact several times. Given a module with vanishing x_0 (and x_1) homology, the bottom nonzero degree of A/B will be 4, represented by P^1. Then the bottom degree in our minimal resolution for M will increase by at least 4 at each stage, giving us a vanishing line in $\mathrm{Ext}_A^{s,t}(M, \mathbb{Z}_3)$. For a module with vanishing x_0, x_1, and x_2 homology, the bottom degree of A/B will be 12, represented by P^3. Then the bottom degree in our minimal resolution for M will increase by at least 12 at each stage.

To each x_k, we let u_k be the polynomial generator of $\mathrm{Ext}^{*,*}_{C(x_k)}(\mathbb{Z}_3, \mathbb{Z}_3)$. Recall that if $x_k = Q_n$, then the bidegree of u_k is $(1, 2\cdot 3^n - 1)$, and if $x_k = P_t^s$, then the bidegree of u_k is $(2, 3[2\cdot 3^t - 2]^{3^s})$. The following definition gives us a connection between these elements of Ext and self maps of spectra. It utilizes the fact that the inclusion $C(x_k) \hookrightarrow A$ gives a map of algebras $\mathrm{Ext}_A^{*,*}(M, M) \to \mathrm{Ext}^{*,*}_{C(x_k)}(M, M)$.

Definition 1.8 (Definition 2.4 from [PS94])**.** Given an A module M, we say that $f \in \mathrm{Ext}_A^{*,*}(M, M)$ is a u_k^i-map of M if f restricts to $u_k^i \otimes 1_M \in \mathrm{Ext}^{*,*}_{C(x_k)}(M, M)$. Similarly, a spectrum X has a u_k^i-map if there is an element $g \in [X, X]$ which is represented at the E_2 term of the Adams spectral sequence by a u_k^i-map of $H^*(X)$.

Remark 1.9. The restriction map mentioned in this definition comes from the fact that A is a free $C(x_k)$ module [MM65], so a free A resolution of M is automatically a free $C(x_k)$ resolution of M as well. Thus, our inclusion $C(x_k) \hookrightarrow A$ gives us the restriction map $\mathrm{Ext}_A^{s,t}(M, M) \to \mathrm{Ext}_{C(x_k)}^{s,t}(M, M)$.

To build the element $u_k^i \otimes 1_M$ that we will need for comparisons, we begin with the minimal $C(x_k)$ resolution of $\mathbb{Z}_3$:

$$\mathbb{Z}_3 \leftarrow C(x_k) \leftarrow \Sigma^d C(x_k) \leftarrow \dots$$

where the suspension d depends on the type of differential. Then the element u_k^i is represented by a map from an appropriate (either i or $2i$) stage of the resolution to $\Sigma^{i \cdot s(x_k)} \mathbb{Z}_3$.

If we then tensor this resolution on the right by M, we get a $C(x_k)$ resolution of M:

$$\mathbb{Z}_3 \otimes M \leftarrow C(x_k) \otimes M \leftarrow \Sigma^d C(x_k) \otimes M \leftarrow \dots$$

Then the element $u_k^i \otimes 1_M$ is respresented by the element mapping from the same stage of the resolution to $\Sigma^{i \cdot s(x_k)} \mathbb{Z}_3 \otimes M$, which is u_k^i on the first factor, and the identity on the second factor.

We will see in the next section the importance of these u_k maps. In particular, Lemma 1.12 illustrates the relationship between u_k maps on certain types of spectra and v_n maps on related finite spectra.

Stably Finite Spectra and v_n Maps

In constructing our u_k^i maps, we will be working with a special category of spectra as defined in [PS94].

Definition 1.10. The category of *stably finite* p-local spectra is the thick subcategory of p-local spectra generated by finite spectra and locally finite generalized mod p Eilenberg-MacLane spectra (i.e. bounded below generalized $\mathbb{Z}_p$ Eilenberg-MacLane spectra with finitely many homotopy groups in each dimension).

In particular, we know that the cohomology of such a stably finite p-local spectrum will be an A module of finite type.

Instead of directly constructing a v_2 self map on a finite spectrum, we instead create a stably finite spectrum with an appropriate u_k^i map, and take advantage of a nice relationship between u_k maps and v_n maps whenever $x_k = Q_n$. In particular, we use the following results to see that such a constructed u_k^i map induces a v_n map on a related finite spectrum.

Lemma 1.11 (Lemma 3.1 from [PS94]). *Suppose that W is a spectrum, M is a finite spectrum, and there is a map $g : W \to M$ such that the fiber of g has a finite Adams resolution. If W has a self map f, then M has an associated self map $\bar{f}$ such that $gf = \bar{f}g$.*

The proof of this Lemma essentially boils down to the fact that $[K, M] = 0$ if K is an Eilenberg-MacLane spectrum, a result from [Mar74].

Thus, once we have found our stably finite spectrum with a u_k^i map, we simply need to find a map from it to an appropriate finite spectrum. We then use the following result to show that the induced map on the finite spectrum is actually a v_n map. This result comes from the proof of Corollary 3.5 in [PS94]

Lemma 1.12. *Suppose that $x_k = Q_n$, and that W is a stably finite spectrum with a u_k^i map, f. Then $K(n)_*(f)$ is an isomorphism, and if M is a finite spectrum with a map $g : W \to M$ whose fiber has a finite Adams resolution, then the self map $\bar{f}$ on M associated to f is a v_n map.*

At $p = 3$, since $x_3 = Q_2$, we will need to find a stably finite spectrum with a u_3^i map for some i, along with a suitable map to some finite spectrum. Then by appealing to this Lemma, we have the desired v_2 map on a finite spectrum.

To find such a u_3 map, we use one further result about stably finite spectra:

Proposition 1.13 (Proposition 2.8 from [PS94])**.** *If X is a stably finite p-local spectrum such that $H_*(H^*(X); x_j) = 0$ for $j < k$ and $H_*(H^*(X); x_k) \neq 0$, then X has a u_k^i-map for some $i \geq 1$.*

This means that we will need to construct a spectrum whose cohomology has no homology with respect to the differentials x_0, x_1, and x_2. We construct such a spectrum by starting with one that has no x_0 homology, and iteratively killing off the x_1 and x_2 homology by taking the cofibers of maps that are isomorphisms with respect to x_1 and x_2 homology. The resulting spectrum, which we denote Z below, will then be guaranteed to have a u_3^i map for some i.

CHAPTER II

A RESULT ABOUT U_K SELF MAPS AND X_K HOMOLOGY

Before proceeding to the construction of our self maps, we prove a result about x_k homology and its connection with u_k maps.

General Results

The following two lemmas help us to identify u_k self maps. We do not need to restrict to the $p = 3$ case for these two results. Given a stably finite spectrum, W, we use the proposition below in order to be able to work with the x_k homology of $H^*(W)$.

Proposition 2.1. *If W is a stably finite spectrum, then $H_*(H^*(W); x_k)$ will be a finite dimensional vector space over $\mathbb{Z}_p$.*

Proof. Suppose that W is a stably finite spectrum. If W is finite, then $H^*(W)$ is finite, and thus $H_*(H^*(W); x_k)$ is finite as well. If W is a generalized Eilenberg-MacLane spectrum, then its cohomology is a sum of copies of A, which has no x_k homology. Otherwise, W is obtained by a finite number of cofibrations involving finite spectra and generalized Eilenberg-MacLane spectra. By examining the long exact sequences in x_k homology, we can see that $H_*(H^*(W); x_k)$ will be finite dimensional as a vector space over $\mathbb{Z}_p$ $\square$

We make use of the first lemma below to take advantage of this fact. We will use this later to determine that certain maps we create are u_k maps.

Lemma 2.2. *Let V and V' be finite dimensional vector spaces over a field F. If R is an augmented F algebra, then we have an isomorphism:*

$$Ext_R^s(V, V') \cong Ext_R^s(F, F) \otimes Hom(V, V') \tag{2.1}$$

Further, this isomorphism in natural with respect to Yoneda multiplication.

In the context of this Lemma, we view the vector spaces V and V' as trivial R modules, that is R acts through its augmentation $\varepsilon : R \to F$.

Proof. To first prove isomorphism (2.1), we pick bases I and J for V and V', respectively. By assumption, I and J are finite. By writing V and V' as direct sums of copies of F, we can use the properties of Ext to then write an isomorphism:

$$\mathrm{Ext}_R^s(V, V') \cong \bigoplus_{I,J} \mathrm{Ext}_R^s(F, F)$$

This, in turn, gives us the isomorphism

$$\bigoplus_{I,J} \mathrm{Ext}_R^s(F, F) \cong \mathrm{Ext}_R^s(F, F) \otimes \mathrm{Hom}(V, V')$$

This establishes (2.1); it remains to show that it is natural with respect to Yoneda multiplication in Ext. To do this, we need to describe the isomorphism of (2.1). Using our chosen bases, an element of $\mathrm{Hom}(V, V')$ is a matrix with entries in F. Let $E_{i,j}$ be the matrix that is 0 in all entries except the (i, j) entry, which is 1. So $E_{i,j}$ takes the ith basis element of V to the jth basis element of V' and is zero on the other basis elements of V. Consider the ith projection map $\pi_i : V \to F$, and the jth inclusion map $f_j : F \to V'$. Combining these maps with the functoriality of Ext (contravariant in the first position, and covariant in the second), we have a map $E_{i,j} : \mathrm{Ext}_R(F,F) \to \mathrm{Ext}_R(V,V')$. Then the image of $x \otimes E_{i,j}$ under the map $\mathrm{Ext}^s_R(F,F) \otimes \mathrm{Hom}(V,V') \to \mathrm{Ext}^s_R(V,V')$ from (2.1) is $E_{i,j}(x)$.

We then look at the following diagram, for finite dimensional F-vector spaces V, V', V'', and show that it commutes:

$$\begin{tikzcd}
\mathrm{Ext}^s_R(V,V') \otimes \mathrm{Ext}^{s'}_R(V',V'') \arrow[r] & \mathrm{Ext}^{s+s'}_R(V,V'') \\
 & \mathrm{Ext}^{s+s'}_R(F,F) \otimes \mathrm{Hom}(V,V'') \arrow[u] \\
\mathrm{Ext}^s_R(F,F) \otimes \mathrm{Hom}(V,V') \otimes \mathrm{Ext}^{s'}_R(F,F) \otimes \mathrm{Hom}(V',V'') \arrow[uu] \arrow[ur] &
\end{tikzcd} \tag{2.2}$$

We start with an element $x \otimes E_{i,j} \otimes y \otimes E_{j',k}$ in the bottom left corner. Going up, we have $E_{i,j}(x) \otimes E_{j',k}(y)$. We then need to describe the product of these two elements of Ext. To do this, we take some R resolution of the field F

$$F \leftarrow B_0 \leftarrow B_1 \leftarrow \dots \tag{2.3}$$

Then the element x is represented by a map $\tilde{x} : B_s \to F$, and y by a map $\tilde{y} : B_{s'} \to F$. In particular, since B_s is projective, and the map $B_0 \to F$ is surjective, we can lift $\tilde{x}$ to $\tilde{x}_0$, a map $B_s \to B_0$

$$\begin{tikzcd}
B_s \arrow[r, dashed, "\tilde{x}_0"] \arrow[dr, "\tilde{x}"'] & B_0 \arrow[d] \\
 & F
\end{tikzcd}$$

We can then lift this map s' additional times to $\tilde{x}_{s'} : B_{s+s'} \to B_{s'}$. The product xy in Ext is then represented by the composition $\tilde{y} \circ \tilde{x}_{s'}$ as shown along the top of the diagram below.

$$\begin{tikzcd}
B_{s+s'} \arrow[r, dashed, "\tilde{x}_{s'}"] \arrow[d] & B_{s'} \arrow[r, "\tilde{y}"] \arrow[d] & F \\
\vdots \arrow[d] & \vdots \arrow[d] & \\
B_s \arrow[r, "\tilde{x}_0"] \arrow[dr, "\tilde{x}"] & B_0 \arrow[d] & \\
& F &
\end{tikzcd}$$

We can use (2.3) to create particular resolutions of V, V', V'' by tensoring on the right by those vector spaces. We can then represent $E_{i,j}(x)$ and $E_{j',k}(y)$ by maps

$$\tilde{x} \otimes E_{i,j} : B_s \otimes V \to F \otimes V' \qquad \tilde{y} \otimes E_{j',k} : B_{s'} \otimes V' \to F \otimes V''$$

Then, the product of $E_{i,j}(x) \otimes E_{j',k}(y)$ is given by the composition along the top of the following diagram:

$$\begin{tikzcd}
B_{s+s'} \otimes V \arrow[r, dashed, "\tilde{x}_{s'} \otimes E_{i,j}"] \arrow[d] & B_{s'} \otimes V' \arrow[r, "\tilde{y} \otimes E_{j',k}"] \arrow[d] & F \otimes V'' \\
\vdots \arrow[d] & \vdots \arrow[d] & \\
B_s \otimes V \arrow[r, "\tilde{x}_0 \otimes E_{i,j}"] \arrow[dr, "\tilde{x} \otimes E_{i,j}"'] & B_0 \otimes V' \arrow[d] & \\
& F \otimes V' &
\end{tikzcd} \tag{2.4}$$

The composition of these maps in (2.4) gives $E_{i,k}$ on the right factor exactly when $j = j'$, otherwise the composition of the vector space maps is 0. Thus, the product of our two ext elements is just $E_{i,k}(xy)$ when $j = j'$ and 0 otherwise.

Going to the right in diagram (2.2), we have $xy \otimes E_{i,j}E_{j',k}$. The product of matrices on the right is $E_{i,k}$ when $j = j'$, and 0 otherwise. As in our description of the isomorphism (2.1), we know the image of this element will be $E_{i,k}(xy)$ when $j = j'$ and 0 otherwise. This coincides with the image from the other path around the square, so we can conclude that it commutes.

From this, we can conclude that our isomorphism (2.1) is natural with respect to the Yoneda multiplication in Ext. $\square$

As we saw in Definition 1.8, u_k maps on M are defined by their restriction to $\mathrm{Ext}_{C(x_k)}(M, M)$. Our goal in this chapter is to establish a connection between u_k maps on M and the x_k homology of M. The following lemma assists us in computing these Ext groups in the case that $x_k = Q_n$ for some n. This result holds for any prime, not just $p = 3$. Recall that in this situation the sub Hopf algebra $C(x_k)$ is exterior.

Lemma 2.3. *Let p be a prime, and A the mod p Steenrod algebra. Let $x_k = Q_n \in A$ for some n, and let M and M' be bounded below A modules with finitely generated x_k homology. Then, for $s \geq 1$, we have an isomorphism*

$$Ext^{s,t}_{C(x_k)}(M, M') \cong \mathbb{Z}_p[u_k] \otimes Hom(V, V'), \tag{2.5}$$

which is natural with respect to the two modules, and where $V = H_*(M; x_k)$, and $V' = H_*(M'; x_k)$.

Proof. We first decompose $M = W \oplus V$, and $M' = W' \oplus V'$, where W and W' are free $C(x_k)$ module [Mar83, Theorem 15.20 a]. We can then use this to decompose $\mathrm{Ext}(M, M')$ in the following way:

$$\begin{aligned} \mathrm{Ext}^{s,t}_{C(x_k)}(M, M') \cong \mathrm{Ext}^{s,t}_{C(x_k)}(W, W') \oplus \mathrm{Ext}^{s,t}_{C(x_k)}(W, V') \\ \oplus \mathrm{Ext}^{s,t}_{C(x_k)}(V, W') \oplus \mathrm{Ext}^{s,t}_{C(x_k)}(V, V') \end{aligned} \tag{2.6}$$

We know that $C(x_k)$ is injective as a $C(x_k)$ module. Then a direct sum of copies of $C(x_k)$ is also injective [Mar83, Theorem 15.27 c], so W' is injective. Also, W is clearly projective, so for $s \geq 1$, the first three terms on the right side of (2.6) are 0, giving us

$$\mathrm{Ext}^{s,t}_{C(x_k)}(M, M') \cong \mathrm{Ext}^{s,t}_{C(x_k)}(V, V') \tag{2.7}$$

Then, we know (2.7) is a natural isomorphism, coming from the naturality of the Yoneda product and the maps $M \twoheadrightarrow V$ and $V' \hookrightarrow M'$. We can now appeal to Lemma 2.2 to establish the natural isomorphism:

$$\mathrm{Ext}_{C(x_k)}(V, V') \cong \mathrm{Ext}_{C(x_k)}(\mathbb{Z}_p, \mathbb{Z}_p) \otimes \mathrm{Hom}(V, V') \cong \mathbb{Z}_p[u_k] \otimes \mathrm{Hom}(V, V')$$

□

A similar result about Ext over $C(x_k)$ when $x_k = P^s_t$ would be helpful, but we do not have such a result.

Main Theorem

We return here to the specific case that $p = 3$, and that A denotes the mod 3 Steenrod algebra.

Before stating the next theorem, we introduce a bit of notation. Let M be an A module, and let $P_\bullet$ be a projective A resolution of M. Then, given an element $\alpha \in \operatorname{Ext}_A^{s,t}(M, M)$, α is represented by one or more maps $f : P_s \to \Sigma^t M$. This map f induces a map $\bar{f}$ defined to make the lower right triangle in the diagram below commute:

$$\begin{tikzcd}
M & \cdots \arrow[l, "\partial_0"'] & P_{s-1} \arrow[l, "\partial_{s-1}"'] & P_s \arrow[l, "\partial_s"'] \arrow[dl] \arrow[dd, "f"] & \cdots \arrow[l, "\partial_{s+1}"'] \\
 & & \ker(\partial_{s-1}) \arrow[u, hook] \arrow[dr, dashed, "\bar{f}"'] & & \\
 & & & \Sigma^t M &
\end{tikzcd}$$

Theorem 2.4. *Let M be a stably finite A module such that $H_*(M; x_m) = 0$ for $m < k$, and $H_*(M; x_k) \neq 0$. Using the same notation as above, let $\alpha \in Ext_A^{s,t}(M, M)$. Then some power of α is a u_k^i map if, and only if, the induced map $\bar{f} : \ker(\partial_{s-1}) \to \Sigma^t M$ is an isomorphism on x_k homology.*

Proof. The "only if" direction of this proof comes from [PS94, Proposition 2.7]. We must still prove that if the induced map $\bar{f} : \ker(\partial_{s-1}) \to \Sigma^t M$ is an isomorphism on x_k homology, then some power of α is a u_k^i map for some i. We proceed by considering two cases, $x_k = Q_n$, and $x_k \neq Q_n$.

Case 1. $x_k = Q_n$ for some n

As in Remark 1.9, our free A resolution is also a free $C(x_k)$ resolution, so we can consider the image of α in $\operatorname{Ext}_{C(x_k)}^{s,t}(M, M)$.

Let $V = H_*(M; x_k)$ be the x_k homology of M. Consider the short exact sequence

$$0 \to \ker(\partial_0) \to P_0 \xrightarrow{\partial_0} M \to 0$$

Since P_0 is free over A, it must have zero x_k homology. Thus, by examining the long exact sequence in x_k homology, we see that

$$H_*(\ker(\partial_0); x_k) = \Sigma^{|x_k|} V$$

Repeating this argument $s-1$ more times, we have that

$$H_*(\ker(\partial_{s-1}); x_k) = \Sigma^{s|x_k|} V$$

Then, by assumption $\bar{f}$ gives us an isomorphism $\hat{f} : \Sigma^{s|x_k|} V \to \Sigma^t V$.

By Lemma 2.3, we know that

$$\mathrm{Ext}_{C(x_k)}(M, M) \cong \mathbb{Z}_3[u_k] \otimes \mathrm{Hom}_{\mathbb{Z}_3}(V, V).$$

Under the isomorphism outlined in this lemma, we can see that α maps to $u_k^s \otimes \hat{f}$.

Since M is assumed to be stably finite, V is finitely generated as a vector space. Thus, some power of $\hat{f}$ is the identity on V, and we can conclude that some power of α restricts to $u_k^i \otimes 1$ for some i, that is, some power of α is a u_k^i map.

Case 2. $x_k \neq Q_n$

For this case, we note that we must have s even, this is the only opportunity for us to have an isomorphism on x_k homology.

Our subalgebra $C(x_k)$ is equal to $\mathbb{Z}_3[x_k]/x_k^3$. Since $C(x_k)$ is a graded PID, we can describe graded, finitely generated $C(x_k)$ modules using the graded, cyclic module $\mathbb{Z}_3$, $B = C(x_k)/(x_k^2)$, and $C(x_k)$. We then decompose M, as a $C(x_k)$

module, that is, for some index sets I, J, L, we can write:

$$M = \bigoplus_{i \in I} \Sigma^{a_i} C(x_k) \oplus \bigoplus_{j \in J} \Sigma^{b_j} B \oplus \bigoplus_{\ell \in L} \Sigma^{c_\ell} \mathbb{Z}_3$$

The x_k homology of M comes from the copies of B and $\mathbb{Z}_3$ in this decomposition, and since M is stably finite, there are finitely many copies of B and $\mathbb{Z}_3$. We assume, without loss of generality, that these factors are ordered by suspension, that is $b_j \leq b_{j+1}$ and $c_\ell \leq c_{\ell+1}$. Let $W = H_*(M; x_k)$.

Similarly to the previous case, we compute the x_k homology of $\ker(\partial_{s-1})$ in our resolution of M. Consider the A resolution where

$$P_0 = \bigoplus_{i \in I} \Sigma^{a_i} A \oplus \bigoplus_{j \in J} \Sigma^{b_j} A \oplus \bigoplus_{\ell \in L} \Sigma^{c_\ell} A$$

Then, as a $C(x_k)$ module, we have

$$\ker(\partial_0) = \bigoplus_{j \in J} \Sigma^{b_j + 2|x_k|} \mathbb{Z}_3 \oplus \bigoplus_{\ell \in L} \Sigma^{c_\ell + |x_k|} B \oplus \bigoplus_{I'} C(x_k)$$

The copies of $C(x_k)$ do not contribute to the x_k homology, so we do not keep track of their suspensions.

At the next stage of the resolution, $\ker(\partial_1)$ can be computed similarly, with

$$\ker(\partial_1) = \Sigma^{3|x_k|} \left(\bigoplus_{j \in J} \Sigma^{b_j} B \oplus \bigoplus_{\ell \in L} \Sigma^{c_\ell} \mathbb{Z}_3 \right) \oplus \bigoplus_{I''} C(x_k)$$

as a $C(x_k)$ module. Specifically, this tells us that

$$\Sigma^{3|x_k|} H_*(M; x_k) \cong H_*(\ker(\partial_1); x_k)$$

This process generalizes, and we can describe $\ker(\partial_{s-1})$. Since $s-1$ is odd, we know that $\ker(\partial_{s-1})$ will have a copy of B for each copy of B in M, and a copy of $\mathbb{Z}_3$ for each copy of $\mathbb{Z}_3$ in M. Each of these copies is shifted by up by $s \cdot 3|x_k|/2$. Since only the B's and $\mathbb{Z}_3$'s contribute to the x_k homology, and the relative distance between the suspensions are not changed, we have that

$$H_*(\ker(\partial_{s-1}); x_k) = \Sigma^{s\cdot 3|x_k|/2} W = \Sigma^{s\cdot 3|x_k|/2} H_*(M; x_k)$$

We are assuming $\bar{f}$ gives us an isomorphism $\hat{f} : \Sigma^{s\cdot 3|x_k|/2} W \to \Sigma^t W$.

Since $\bar{f}$ is an A module map, it is also a $C(x_k)$ module map. Recall that the x_k homology of our modules comes from the generator in each copy of B, and from the generator in each copy of $\mathbb{Z}_3$. To help us describe the effect of the map $\bar{f}$, we rewrite our decomposition of the module M as $M = C \oplus D \oplus E$, where C is a sum of copies of $C(x_k)$, D is a sum of copies of B, and E is a sum of copies of $\mathbb{Z}_3$.

If $g : \mathbb{Z}_3 \to B$ is a map of $C(x_k)$ modules, then we must have $g(1) = \lambda x$ for some $\lambda \in \mathbb{Z}_3$. Therefore, since $\bar{f}$ is a map of $C(x_k)$ modules, then the composition

$$\Sigma^{s\cdot 3|x_k|/2} E \hookrightarrow C' \oplus \Sigma^{s\cdot 3|x_k|/2}(D \oplus E) \xrightarrow{\bar{f}} \Sigma^t(C \oplus D \oplus E) \twoheadrightarrow \Sigma^t D$$

must land in $x_k \cdot \Sigma^t D$. Here, C' denotes a sum of copies of $C(x_k)$. This composition is then 0 in x_k homology. Thus, since $\bar{f}$ induces an isomorphism in x_k homology, the composition

$$\Sigma^{s\cdot 3|x_k|/2} E \hookrightarrow C' \oplus \Sigma^{s\cdot 3|x_k|/2}(D \oplus E) \xrightarrow{\bar{f}} \Sigma^t(C \oplus D \oplus E) \twoheadrightarrow \Sigma^t E$$

must be an isomorphism. Similarly, the composition

$$\Sigma^{s\cdot 3|x_k|/2} D \hookrightarrow C' \oplus \Sigma^{s\cdot 3|x_k|/2}(C \oplus D \oplus E) \xrightarrow{\bar{f}} \Sigma^t(C \oplus D \oplus E) \twoheadrightarrow \Sigma^t D$$

can be written as a matrix of the form $P + x_k \cdot Q$, where P is invertible.

We must show that some power of $\hat{f}$ is the identity on x_k homology. To do this, we represent $\bar{f}$ as a block matrix:

$$F = \begin{bmatrix} P & Q \\ R & S \end{bmatrix}$$

where P represents the part of the map sending copies of B to copies of B, R represents the part sending copies of B to copies of $\mathbb{Z}_3$, Q represents the part sending copies of $\mathbb{Z}_3$ to copies of B, and S represents the part sending copies of $\mathbb{Z}_3$ to copies of $\mathbb{Z}_3$. In particular, we know that the blocks on the diagonal (P and S) are isomorphisms, and thus invertible over $C(x_k)$.

Further, based on our ordering of the suspensions in the decomposition, we can say a little more about the structure of the blocks. P is an upper block triangular matrix, where the blocks along the diagonal are invertible over $\mathbb{Z}_3$, with the size of these blocks determined by the number of copies of B with the same suspension. Blocks above the diagonal may contain multiples of x_k when the suspensions of copies of B differ by $|x_k|$. Otherwise, they are all zero. Similarly, S is a block diagonal matrix, with the nonzero blocks along the diagonal being invertible over $\mathbb{Z}_3$. We can also describe R as a block matrix containing only elements of $\mathbb{Z}_3$, and Q is a block matrix containing multiples of x_k.

Then, we can write $F = N + Tx_k$, where N and T are matrices over $\mathbb{Z}_3$. In addition, N is a lower triangular block matrix over $\mathbb{Z}_3$, with invertible blocks along the diagonal. Thus, N is invertible over $\mathbb{Z}_3$. Tx_k is a matrix over the augmentation ideal of $C(x_k)$. Since this augmentation ideal is nilpotent, this is enough to guarantee that F is invertible over $C(x_i)$. Then, because the group of invertible matrices over $\mathbb{Z}_3$ is finite, there is some power m such that $F^m = I$. This means that $(\hat{f})^m$ is the identity on x_k homology.

We must now show that the restriction of α^m to $\mathrm{Ext}_{C(x_k)}^{ms,mt}(M, M)$ is equivalent to $u_k^{ms/2} \otimes 1_M$. From above, we know that

$$\ker(\partial_{ms-1}) = C' \oplus \Sigma^{ms \cdot 3|x_k|/2} (D \oplus E)$$

where C' is a large sum of copies of $C(x_k)$. Further, P_{ms} can be interpreted as a free $C(x_k)$ module with a copy of $C(x_k)$ mapping surjectively onto each summand of D and E.

Let $Q_\bullet(M)$ be the $C(x_k)$ resolution of M as described in Remark 1.9, and denote the differentials $\partial_i' : Q_i \to Q_{i-1}$. We claim that

$$\ker(\partial_{ms-1}') = \Sigma^{ms \cdot 3|x_k|/2} (D \oplus E) . \tag{2.8}$$

We can decompose each stage of $Q_\bullet$ as:

$$C(x_k) \otimes M \cong \bigoplus (C(x_k) \otimes C(x_k)) \bigoplus (C(x_k) \otimes B) \bigoplus (C(x_k) \otimes \mathbb{Z}_3)$$

As $C(x_k)$ modules, we have

$$C(x_k) \otimes B \cong C(x_k) \oplus \Sigma^{|x_k|} C(x_k).$$

We note that $Q_\bullet(M \oplus N) = Q_\bullet(M) \oplus Q_\bullet(N)$. In particular $Q_\bullet(B)$ is

$$\mathbb{Z}_3 \otimes B \xleftarrow{\partial_0'} C(x_k) \otimes B \xleftarrow{\partial_1'} \Sigma^{|x_k|} C(x_k) \otimes B \xleftarrow{\partial_2'} \Sigma^{3|x_k|} C(x_k) \otimes B \leftarrow \dots$$

Here, the kernel of ∂_0' is isomorphic, as a $C(x_k)$ module, to $\Sigma^{2|x_k|}\mathbb{Z}_3 \oplus \Sigma^{x_k} C(x_k)$. Similarly, the kernel of ∂_1' is isomorphic, as a $C(x_k)$ module, to $\Sigma^{3|x_k|}B$. Then $Q_2(B)$ is a copy of $C(x_k)$ mapping surjectively onto this copy of B. A similar argument holds for the $Q_\bullet(\mathbb{Z}_3)$.

This pattern repeats every two stages of the resolution, so that

$$\ker(\partial_{2n-1}') = \Sigma^{2n \cdot 3|x_k|/2}(D \oplus E)$$

and Q_{2n} has a copy of $C(x_k)$ mapping surjectively onto each summand of D and E. This shows our claim in (2.8) is true.

To finish the proof, we must construct a map between $Q_\bullet(M)$ and $P_\bullet$ to show that our two Ext elements are equivalent. We take the map that sends the copy of $C(x_k)$ in Q_{ms} mapping to a particular summand of $\ker(\partial_{ms-1}')$ isomorphically to the copy of $C(x_k)$ in P_{ms} mapping onto the same summand of $\ker(\partial_{ms-1})$.

Then the composition of this map and f is the same as $u_k^m \otimes 1$, so α^m is a $u_k^{ms/2}$ map. $\square$

Now, for a spectrum X, we can construct an Adams resolution for X. For a more thorough description, see [Rav03, Definition 2.1.3]. Let X_i be the ith stage of the Adams resolution, where $X_0 = X$. Given a self map $g : \Sigma^d X \to X$ coming from an element of $\mathrm{Ext}_A^{s,d+s}(H^*(X), H^*(X))$, we can lift g to a map $\bar{g} : \Sigma^d X \to X_s$. This also gives us a map in cohomology $\bar{g}^* H^*(X_s) \to H^*(\Sigma^d X)$.

Theorem 2.5. *Let X be a stably finite p-local spectrum, $H_*(H^*(X); x_m) = 0$ for $m < k$, and $H_*(H^*(X); x_k) \neq 0$, and let $f : \Sigma^d X \to X$ come from an element $\alpha \in Ext_A^{s,d+s}(H^*(X), H^*(X))$. Then some power of α is a u_k map if and only if $\bar{f}^*$ is an isomorphism on x_k homology.*

Proof. An Adams resolution of X gives a resolution, $P_\bullet$ of $H^*(X)$ by free A modules. In this correspondence, we have that $\ker(\partial_i) \cong \Sigma^i H^*(X_{i+1})$. Then, $\bar{f}^*$ corresponds to a map $\ker(\partial_{s-1}) \to \Sigma^d H^*(X)$. Now apply Theorem 2.4 with $M = H^*(X)$. □

CHAPTER III

CONSTRUCTING U_K MAPS

General Strategy for Constructing u_k Maps

In this chapter, we detail the construction of our u_1, u_2, and u_3 maps and the spectra on which they give us the desired self maps. Each of these three constructions follows the same general pattern, and we will refer back to the following numbered steps during our construction.

Step 0: Begin with a spectrum X such that $H_*(H^*(X); x_j) = 0$ for $j < k$. By proposition 1.13, X will have a u_k^i map for some i.

Step 1: Construct a minimal resolution for the cohomology of the spectrum as an A module.

Step 2: Construct an element of Ext_A in the appropriate bidegree to be a u_k map.

Step 3: Appeal to theorem 2.4 to show that some power of this map is a u_k^i map.

Step 4: Show that this element survives the Adams Spectral Sequence to give a self map on the corresponding spectrum.

Step 5: Lift the self map up the Adams tower for the spectrum.

Step 6: Take the cofiber of this lifted map to generate the spectrum for the next iteration. The cohomology of this new spectrum will have no x_j homology for $j < k+1$. To prepare for the next iteration of this process, we describe the A module structure of the cohomology of this newly created spectrum.

The last two steps are not required with our u_3 map, as we do not construct a u_4 map.

Constructing a u_1 Map

Step 0: We start with the mod 3 Moore spectrum $M(3)$. As a module over A, $H^*M(3)$ has no $x_0 = \beta$ homology. We hope to find a self map of $M(3)$ that is a power of u_1. It will turn out that this u_1 map is the v_1 map on $M(3)$ as seen in [Mil81].

Step 1: As outlined above, we start by computing $\mathrm{Ext}_A^{s,t}(H^*M(3), H^*M(3))$ using a minimal resolution of $H^*(M(3))$ in order to find a candidate for a u_1 map. A calculation by hand shows the beginnings of this minimal resolution as follows.

$$H^*M(3) \xleftarrow{\partial_0} P_0 = A \xleftarrow{\partial_1} P_1 = \Sigma^4 A \oplus \Sigma^5 A \oplus \Sigma^{12} A \oplus \cdots \xleftarrow{\partial_2} P_2 = \Sigma^9 A \oplus \cdots \xleftarrow{\partial_3} \cdots$$

Step 2: Now that we have the minimal resolution, we can construct our candidate u_1 map. Since the polynomial generator u_1 is in bidegree $(1, 5)$ as described in Section 1.5, we are looking for an A module map $P_1 \to \Sigma^5 H^*(M(3))$ to generate an element of $\mathrm{Ext}_A^{1,5}(H^*(M(3)), H^*(M(3)))$.

We have such a map $f \in \mathrm{Hom}^5(P_1, H^*(M(3)))$, which is unique up to scalar multiplication, sending $(0, 1, 0, \dots) \mapsto 1$ and sending all other generators of P_1 to 0. For dimensional reasons $f \circ \partial_2 = 0$, and similarly, since $\mathrm{Hom}^5(P_0, H^*(M(3))) = 0$, $f \notin \mathrm{im}(\partial_1^*)$. Thus, f gives us a nonzero element $\alpha \in \mathrm{Ext}_A^{1,5}(H^*(M(3)), H^*(M(3)))$.

Step 3: We claim that this Ext element is our u_1 map, and we'd like to be able to use Theorem 2.4 to prove this.

Proposition 3.1. *The map f described above induces a map $\bar{f} : \ker(\partial_0) \to \Sigma^5 H^*(M(3))$ which is an isomorphism on x_1 homology.*

$$
\begin{tikzcd}
H^*(M(3)) & A \arrow[l, "\partial_0"'] & P_1 \arrow[l, "\partial_1"'] \arrow[dl, two heads] \arrow[dd, "f"] & P_2 \arrow[l, "\partial_2"'] \\
& \ker(\partial_0) \arrow[u, hook] \arrow[dr, dashed, "\bar{f}"'] & & \\
& & \Sigma^5 H^*(M(3)) &
\end{tikzcd}
\tag{3.1}
$$

Proof. We start by noting that the map $\bar{f}$ is well defined, as the A module generators of $\ker(\partial_0)$ have unique lifts in P_1. Since ∂_0 maps 1 to 1 and β to β, then $\ker(\partial_0)$ is just a copy of A with these bottom two classes removed. As an A module, the generators in the lowest dimension of $\ker(\partial_0)$ are P^1 in dimension 4 and $P^1\beta$ in dimension 5. Then, the generator $(1, 0, 0, \dots) \in P_1$ is mapped via the surjection to the element P^1 in the kernel, and the generator $(0, 1, 0, \dots) \in P_1$ is mapped to $P^1\beta$ in the kernel. This tells us that

$$
\begin{aligned}
\bar{f}(P^1) &= f(1, 0, 0, \dots) = 0 \\
\bar{f}(P^1\beta) &= f(0, 1, 0, \dots) = 1
\end{aligned}
$$

The x_1 homology of $H^*(M(3))$ is generated by each of the nonzero cohomology classes, 1 and β. The x_1 homology of $\ker(\partial_0)$ is generated by the classes $P^1\beta - \beta P^1$ in dimension 5, and $\beta P^1\beta$ in dimension 6, that is:

$$
H_*(\ker(\partial_0); x_1) = \langle P^1\beta - \beta P^1, \beta P^1\beta \rangle
$$

We note that these classes are just x_1 applied to the two classes we removed from A to construct $\ker(\partial_0)$. These elements can be computed by hand using the long exact sequence in x_1 homology from the short exact sequence

$$
0 \to \ker(\partial_0) \to P_0 \to H^*(M(3)) \to 0
$$

We now need to find the image of these two classes under the map $\bar{f}$. Since $\partial_1(-\beta, 1, 0, \dots) = P^1\beta - \beta P^1$ and $\partial_1(0, \beta, 0, \dots) = \beta P^1 \beta$, we have the following for $\bar{f}$:

$$\begin{aligned} \bar{f}(P^1\beta - \beta P^1) &= f(-\beta, 1, 0, \dots) = 1 \\ \bar{f}(\beta P^1 \beta) &= f(0, \beta, 0, \dots) = \beta \end{aligned} \tag{3.2}$$

□

Corollary 3.2. *Some power of α is a u_1^i map for some i.*

Proof. $\bar{f}$ is an isomorphism on x_1 homology, so we can apply Theorem 2.4 □

Step 4: Next, we must show that α survives the Adams spectral sequence to give us a self map on $M(3)$.

We know that $H_*(H^*(M(3)); x_0) = 0$, so by Lemma 1.7 $H^*(M(3))$ is free through degree 3. This stems from the fact that P^1 in degree 4 is the element of lowest degree not in the sub Hopf algebra of A containing x_0 This indicates that at each step in our minimal resolution, the bottom suspension will increase by at least 4. This means that $\mathrm{Ext}_A^{s,t}(H^*(M(3)), H^*(M(3))) = 0$ for $(t - s) \leq 5$ as long as $s > 1$. This is illustrated in the figure below, where all Ext classes above and to the left of the line must be 0.

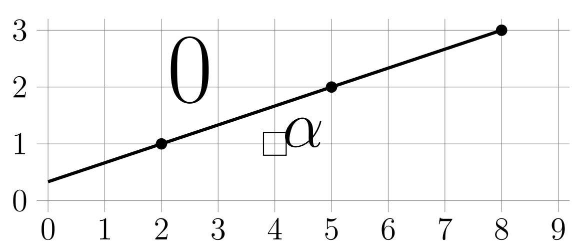

FIGURE 1. Vanishing edge for $\mathrm{Ext}^{s,t}(H^*(M(3)), H^*(M(3)))$ in $(t-s, s)$ coordinates

Our element $\alpha \in \mathrm{Ext}^{1,5}$ lies below this line at $(4,1)$ in $(t-s,s)$ coordinates. However, the differential d_2 on the E_2 page of the spectral sequence would send this to coordinates $(3,3)$ in the diagram, which must be 0. For dimensional reasons, we can also see that this class cannot be hit by the differential d_2. Similarly, all higher differentials would land even higher in the $t-s = 3$ column. Thus, all of the differentials in the Adams spectral sequence on α will be 0, and it is not in the image of any of these differentials, so α survives to the E_∞ page, giving us an essential self-map $f_{M(3)} : \Sigma^4 M(3) \to M(3)$

Step 5: We now lift our map $f_{M(3)}$ up the Adams resolution. Recall, we construct the first stage of the Adams resolution, denoted $M(3)_1$ by taking the fiber of the map $M(3) \to K(\mathbb{Z}_3)$. Since $f_{M(3)}$ comes from a map of Adams filtration 1, we can lift it to $\bar{f}_{M(3)}$ as in the diagram below.

$$\begin{tikzcd} & M(3)_1 \arrow[d] & \\ \Sigma^4 M(3) \arrow[ur, dashed, "\bar{f}_{M(3)}"] \arrow[r, "f_{M(3)}"] & M(3) \arrow[r] & K(\mathbb{Z}_3) \end{tikzcd} \tag{3.3}$$

By calculation, the cohomology of $M(3)_1$ corresponds to a shift down by one dimension of $\ker(\partial_0)$ from (3.1) above. This leaves the bottom few classes as P^1 in degree 3, $\beta P^1, P^1\beta$ in degree 4, and $\beta P^1 \beta$ in degree 5.

We note that the induced map in cohomology

$$\bar{f}^*_{M(3)} : H^*(M(3)_1) \to H^*(\Sigma^4 M(3))$$

is the desuspension of the map $\bar{f}$ from our Ext computations above. Namely, $\bar{f}^*_{M(3)}$ is an isomorphism in x_1 homology.

Step 6: Finally, we follow the method outlined in [PS94] to "kill off" the x_1 homology of $M(3)$ and produce a new spectrum with no x_1 homology.

By taking the cofiber of the map $\bar{f}_{M(3)} : \Sigma^4 M(3) \to M(3)_1$, we create a new spectrum, which we will call Y. Since $\bar{f}^*_{M(3)}$ is surjective, we have that $H^*(Y) = \ker \bar{f}^*_{M(3)}$.

Further, we have a short exact sequence in cohomology:

$$0 \to H^*(Y) \to H^*(M(3)_1) \xrightarrow{\bar{f}^*_{M(3)}} H^*(\Sigma^4 M(3)) \to 0.$$

When we examine the corresponding long exact sequence in x_1 homology, we see that since $\bar{f}^*_{M(3)}$ is an isomorphism on x_1 homology, $H^*(Y)$ will have no x_1 homology. As it also has no x_0 homology (since $H^*(M(3))$ and $H^*(M(3)_1)$ had no x_0 homology), we will be able to find a u_2^i map on Y for some i.

From our construction, we can describe $H^*(Y)$ as the submodule of $\Sigma^{-1}A$ containing all Serre-Cartan basis elements except $1, \beta, P^1\beta$, and $\beta P^1 \beta$.

We know we will need to construct an Adams resolution for Y, and an A-projective resolution of $H^*(Y)$ in our search for a u_2 map, so it will be helpful to describe the generators of $H^*(Y)$ as an A module. The following result summarizes the structure of the A module $H^*(Y)$.

Proposition 3.3. *$H^*(Y)$ is the submodule of A generated by the classes $P^3P^1\beta$ and those of the form P^{3^i} for $i \geq 0$.*

Proof. We begin with A and study the effects of removing the clases listed above. As an A module, A has a single generator, 1. When we remove this generator, the resulting module is generated by the indecomposable elements of A, namely, β, and elements of the form P^{3^i} for $i \geq 0$.

We next remove β to create a module isomorphic to $H^*(M(3)_1)$. This module still contains P^{3^i}, but may require other generators of the form $P^{3^i}\beta$. We claim that most of these elements are not, in fact, new generators. By the Adem relations, for $i \geq 1$, we have:

$$P^{3^i}\beta = P^1\beta P^{3^i-1} + \beta P^{3^i}$$

Thus, for $i \geq 1$, we can write the element $P^{3^i}\beta$ in terms of the generators of the form P^{3^i} that we already had. Then the only generator we gain at this step is $P^1\beta$.

The third step is to remove the class $P^1\beta$. The possibilities for new generators are $\beta P^1\beta$, and elements of the form $P^{3^i}P^1\beta$. Of these, $\beta P^1\beta$ is clearly a generator. By computation, we have

$$P^1P^1\beta = -P^2\beta = -(\beta P^1 + P^1\beta)P^1$$

so that $P^1P^1\beta$ is not a generator. Furthermore, by the Adem relations, for $i \geq 2$, we have

$$P^{3^i}P^1\beta = P^3P^{3^i-2}\beta + P^{3^i+1}\beta$$

Both terms on the right hand side can be written in terms of our already existing generators, so none of the elements of the form $P^{3^i}P^1\beta$ for $i \geq 2$ are generators of the module. After removing three of the four elements, we have an A module with generators: $\beta P^1\beta, P^3P^1\beta$, and the elements of the form P^{3^i}.

The final step is to remove $\beta P^1\beta$. This gives us the elements $P^{3^i}\beta P^1\beta$ as new possibilities for generators. For small i, we have:

$$P^1\beta P^1\beta = \beta P^2\beta = (\beta P^1\beta)P^1$$

and

$$P^3\beta P^1\beta = (\beta)P^3P^1\beta$$

so that neither of these classes are new generators. Further, for $i \geq 2$, we have the general formula:

$$P^{3^i}\beta P^1\beta = P^1\beta P^{3^i-1}P^1\beta + \beta P^{3^i}P^1\beta$$

Thus, none of these new candidates for generators are actually generators.

Putting all of this together, we have that our A module, $H^*(Y)$, is generated by the class $P^3P^1\beta$ in degree 16, and the classes of the form P^{3^i} in dimension $4(3^i) - 1$. $\square$

Note that unlike our original spectrum $M(3)$, the cohomology of Y is no longer finite, or even finitely generated. But we now have the information that we need in order to begin constructing our u_2 map.

Constructing a u_2 Map

Step 0: As we've shown in the preceeding section, we know that Y is a spectrum whose cohomology has no x_0 or x_1 homology, so we will be able to find a self map that is a power of u_2 by Proposition 1.13. We follow the same outline given at the beginning of the chapter to construct our u_2 map.

Step 1: To compute $\mathrm{Ext}_A^{s,t}(H^*(Y), H^*(Y))$, we'll need to construct a minimal projective resolution of H^*Y. Since u_2 is in homological degree 2, we'll need to construct at least two steps in this resolution. The work we did in the previous section helps us with the first step of this process. We take as our initial projective, P_0, a direct sum of copies of A, one for each generator of $H^*(Y)$, so that

$$P_0 = \Sigma^3 A \oplus \Sigma^{11} A \oplus \Sigma^{16} A \oplus \Sigma^{35} A \oplus_{i\geq 3} \Sigma^{4\cdot 3^i - 1} A \tag{3.4}$$

Then, the map $\partial_0 : P_0 \twoheadrightarrow H^*(Y)$ sends the generator of each copy of A to the corresponding generator of $H^*(Y)$. To compute the next projective, P_1, we need to find generators for $\ker(\partial_0)$. Since $H^*(Y)$ isn't finitely generated, we use Sage [S$^+$16] to compute the kernel through dimension 100, which will be enough for our purposes in this paper. The code can be found in Section 3.4, but we'll outline the general procedure here.

The process is carried out dimension by dimension, starting at the bottom. In each dimension n, we first compute a vector space basis for $H^n(Y)$ and a separate one for $(P_0)_n$. For each basis element of $(P_0)_n$, we write it as a list of basis elements of A, finitely many of which are nonzero. For example, when $n = 12$, we have the element $(\beta P^2, \beta, 0, \dots) \in (P_0)_{12}$. Since we know where ∂_0 sends the generators of P_0, we can determine where it sends each of our basis elements of $(P_0)_n$. As an example, we have:

$$
\begin{aligned}
\partial_0((\beta P^2, \beta, 0, \dots)) &= \partial_0(\beta P^2, 0, \dots) + \partial_0(0, \beta, 0, \dots) \\
&= (\beta P^2)P^1 + (\beta)P^3 \\
&= \beta P^3.
\end{aligned}
$$

We then write the image of each basis element of $(P_0)_n$ as an $\mathbb{Z}_3$ linear combination of the chosen basis elements for $H^n(Y)$. From there, we have all of the information we need to write down ∂_0 in this dimension as a linear map between our two vector spaces. We then have Sage compute the kernel of this map, and use this to determine which elements of $(P_0)_n$ are in the kernel of ∂_0.

To continue forming our minimal resolution, we must determine the generators of $\ker(\partial_0)$ as an A module. Clearly, the element in the bottom dimension $((P^2, 0, \dots) \in (P_0)_{11})$ must be a generator. For higher dimensions we use the following method to determine if an element in dimension n of the kernel is a module generator or not. First, we compute β on elements of the kernel in dimension $n-1$, then P^1 on elements of the kernel in dimension $n-4$, then P^3 and P^9 on elements in the appropriate dimensions. We don't need to check any other P^{3^i} in our range of dimensions. We store the images of these elements of lower dimension in a list for later comparison. Once we have computed all of these images, we test each element of dimension n in the kernel to see if it can be written as an A linear combination of elements in the list of images. If not, we add it to the list of generators in this dimension. We note that future checks in the same dimension require that we check to see if an element is an A linear combination of elements from the image list together with the generators we have already found.

For example, if dimension n of the kernel had elements a and b as a vector space basis, and P^1 on an element from dimension $n-4$ was $a+b$, we would only want to count one of the elements a or b as a module generator.

We give a list of the generators of the kernel in Table 2. below, which we will make use of again when we compute our u_2 map. As these are elements of P_0 of dimension less than 100, we write them as linear combinations of the generators of P_0 in dimensions $3, 11, 16, 35$.

TABLE 2. Generators of $\ker(\partial_0)$ and their dimensions

Dimension	Generator
11	$(P^2, 0, 0, 0)$
19	$(P^3P^1 + P^4, 2P^2, 0, 0)$
20	$(P^3P^1\beta + \beta P^3P^1, P^2\beta, 2P^1, 0)$
27	$(P^6, 2P^3P^1 + 2P^4, 0, 0)$
28	$(P^5\beta P^1, 2P^3P^1\beta + 2P^4\beta, 2P^3, 0)$
33	$(\beta P^6P^1\beta, 0, 2P^3P^1\beta + 2P^4\beta, 0)$
35	$(P^7P^1, P^5P^1 + P^6, 0, 0)$
39	$(P^9, P^6P^1 + P^7, 0, 2P^1)$
59	$(P^{10}P^3P^1, 2P^9P^3 + 2P^{10}P^2 + P^{12}, 0, P^5P^1 + P^6)$
64	$(0, 0, P^9P^3 + P^{10}P^2 + P^{12}, 2P^6P^1\beta)$
83	$(0, 2P^{14}P^3P^1 + P^{14}P^4 + P^{18}, 0, 2P^9P^3 + P^{11}P^1 + 2P^{12})$

Now that we have identified a set of module generators, we can assume P_1 has a copy of A, with an appropriate suspension, corresponding to each of them. Thus, we will have P_1 surjecting onto the kernel, which includes into P_0, and ∂_1 is the composition of these maps:

$$\begin{tikzcd} P_0 & & P_1 \arrow[ll, "\partial_1"'] \arrow[dl, two heads] \\ & \ker(\partial_0) \arrow[ul, hook] & \end{tikzcd}$$

Concretely, we've proven the following proposition:

Proposition 3.4.

$$P_1 = \Sigma^{11} A \oplus \Sigma^{19} A \oplus \Sigma^{20} A \oplus \dots \tag{3.5}$$

and the map ∂_1, at least in the range of dimensions we are concerned with.

$$\begin{aligned} \partial_1(1, 0, 0, \dots) &= (P^2, 0, \dots) \\ \partial_1(0, 1, 0, \dots) &= (P^3 P^1 + P^4, 2P^2, 0, \dots) \\ &\vdots \end{aligned} \tag{3.6}$$

We then use a similar process to compute the kernel of ∂_1 in order to get information about P_2. We first compute the map $P_1 \to \ker(\partial_0)$ as a map of vector spaces, and use Sage to find its kernel. Once we've done this, we then determine the generators of $\ker(\partial_1)$ as an A module as before. These generators are listed below in Table 3. through dimension 75. Again, we write them as a linear combination of the generators of P_1.

As in the the construction of P_1, this information tells us what the generators for the free module P_2 are. We can now build P_2 and describe the map ∂_2.

Proposition 3.5.

$$P_2 = \Sigma^{15} A \oplus \Sigma^{23} A \oplus \Sigma^{28} A \oplus \dots \tag{3.7}$$

and

$$\begin{aligned} \partial_2(1, 0, 0, \dots) &= (P^1, 0, 0, \dots) \\ \partial_2(0, 1, 0, \dots) &= (P^3, 2P^1, 0, \dots) \\ &\vdots \end{aligned}$$

TABLE 3. Generators of $\ker(\partial_1)$ and their dimensions

Dimension	Generator
15	(P^1)
23	$(P^3, 2P^1)$
28	$(P^3P^1\beta, P^2\beta + \beta P^2, 2P^2)$
31	$(0, P^3, 0, 2P^1)$
36	$(0, \beta P^3P^1, P^3P^1 + P^4, 0, 2P^2)$
37	$(0, \beta P^3P^1\beta, 2P^3P^1\beta + \beta P^4, 0, 2\beta P^2, P^1)$
39	$(0, 0, 0, P^3, 0, 0, P^1)$
44	$(0, P^5P^1\beta + P^6\beta, P^5P^1 + P^6, \beta P^3P^1, 2P^3P^1 + 2P^4, 0, P^2\beta)$
45	$(0, 0, P^5P^1\beta + 2P^6\beta + \beta P^5P^1 + \beta P^6, 0, 2P^4\beta, 2P^3, 2\beta P^2\beta)$
47	$(P^9, 0, 0P^4P^1, 0, 0, P^3, 2P^2)$
50	$(0, 0, 0, \beta P^4\beta P^1\beta, 2P^4\beta P^1\beta + 2\beta P^4P^1\beta,$ $P^3P^1\beta + 2P^4\beta + \beta P^3P^1 + 2\beta P^4)$
52	$(0, 0, 2P^6P^2 + P^7P^1, P^5P^1\beta + P^6\beta, P^5P^1 + P^6, 0,$ $2P^3P^1\beta + P^4\beta + 2\beta P^3P^1 + \beta P^4)$
59	$(0, 0, 0, 0, 0, 0, P^6)$
71	$(0, 2P^9P^3P^1 + P^{13}, 0, 2P^9P^2, 0, 0, 2P^9, P^6P^2 + 2P^7P^1 + 2P^8,$ $2P^3)$
72	$(0, P^9P^3P^1\beta + P^{12}P^1\beta, 2P^{12}P^1 + 2P^{13}, 0, P^9P^2 + 2P^{10}P^1, 0,$ $P^7\beta P^2 + 2P^9\beta + \beta P^7P^2, P^6P^2\beta + P^7P^1\beta + P^7\beta P^1$ $+\beta P^6P^2, 2P^3\beta, 2P^2)$

As our later computations will show, we do not need to construct any more of the minimal resolution for $H^*(Y)$ in order to find our u_2 map.

Step 2: Our candidate for a u_2^1 map will be in $\mathrm{Ext}^{2,12}(H^*(Y), H^*(Y))$, and we will have to construct a map $f : P_2 \to \Sigma^{12}H^*(Y)$. Because of the more complicated module structure of $H^*(Y)$, it is hard to immediately construct such an f. We start, however, with an element of $\mathrm{Ext}^{1,11}(H^*(Y), \mathbb{Z}_3)$ which is easier to describe based on our minimal resolution. From our description of the module P_1 in (3.5), we will use the class generated by the map

$$f_1 : P_1 \to \Sigma^{11}\mathbb{Z}_3$$

$$(1, 0, 0, \dots) \mapsto 1$$

sending all other generators of P_1 to 0. We then construct our desired map by working our way through several long exact sequences coming from the construction of Y. To make later suspensions work out nicely, we make use of the isomorphism

$$\mathrm{Hom}_A^{11}(P_1, k) \cong \mathrm{Hom}_A^{12}(P_1, \Sigma^{-1}k)$$

so that $f_1 : P_1 \to \Sigma^{12}(\Sigma^{-1}k)$.

The list below outlines the major steps we must make to construct our desired element, and is followed by the explicit computations. The first step is the map that we have defined above.

- $f_1 \in \mathrm{Hom}_A^{12}(P_1, \Sigma^{-1}\mathbb{Z}_3)$
- $f_2 \in \mathrm{Hom}_A^{12}(P_1, H^*(\Sigma^{-1}M(3)))$
- $f_5 \in \mathrm{Hom}_A^{12}(P_2, H^*(M(3)_1))$
- $f \in \mathrm{Hom}_A^{12}(P_2, H^*(Y))$

In order to complete the second step, we consider the short exact sequence in cohomology

$$0 \to k \hookrightarrow H^*\Sigma^{-1}M(3) \twoheadrightarrow \Sigma^{-1}k \to 0 \tag{3.8}$$

The surjective map in (3.8) gives us a map betweeen Hom groups:

$$\mathrm{Hom}_A^{12}(P_1, H^*(\Sigma^{-1}M(3))) \to \mathrm{Hom}_A^{12}(P_1, \Sigma^{-1}k). \tag{3.9}$$

Our map f_1 pulls back along (3.9) to a map

$$\begin{aligned} f_2 : P_1 &\to \Sigma^{12} H^*(\Sigma^{-1} M(3)) \\ (1, 0, 0, \dots) &\mapsto 1 \\ (\beta, 0, 0, \dots) &\mapsto \beta \end{aligned}$$

and sending the rest of P_1 to 0.

To accomplish the third step, we use our construction of an Adams resolution for $M(3)$ from (3.3) to get a map into $H^*(M(3)_1)$. We have a cofibration

$$\Sigma^{-1} M(3) \to \Sigma^{-1} K(\mathbb{Z}_3) \to M(3)_1.$$

Taking cohomology gives us a short exact sequence:

$$0 \to H^*(M(3)_1) \hookrightarrow \Sigma^{-1} A \twoheadrightarrow H^*(\Sigma^{-1} M(3)) \to 0 \tag{3.10}$$

Applying the functor $\mathrm{Ext}_A(H^*(Y), -)$ to (3.10) gives rise to a long exact sequence. The connecting homomorphism is

$$\mathrm{Ext}^{s,t}(H^*(Y), H^*(\Sigma^{-1} M(3))) \to \mathrm{Ext}^{s+1,t}(H^*(Y), H^*(M(3)_1)). \tag{3.11}$$

For $s \geq 1$, (3.11) is an isomorphism since $\mathrm{Ext}_A^{s,t}(H^*(Y), A) = 0$. To understand how this isomorphism works, we look at the diagram

$$\begin{tikzcd} 0 & \mathrm{Hom}^{12}(P_2, H^*(\Sigma^{-1} M(3))) \arrow[l] & \mathrm{Hom}^{12}(P_2, \Sigma^{-1} A) \arrow[l] & \mathrm{Hom}^{12}(P_2, H^* M(3)_1) \arrow[l] & 0 \arrow[l] \\ 0 & \mathrm{Hom}^{12}(P_1, H^*(\Sigma^{-1} M(3))) \arrow[l] \arrow[u, "\partial_2"] & \mathrm{Hom}^{12}(P_1, \Sigma^{-1} A) \arrow[l] \arrow[u, "\partial_2"] & \mathrm{Hom}^{12}(P_1, H^* M(3)_1) \arrow[l] \arrow[u, "\partial_2"] & 0 \arrow[l] \end{tikzcd}$$

The vertical maps here are induced by the map $\partial_2 : P_2 \to P_1$ in the minimal resolution of $H^*(Y)$. We know the rows in the diagram are exact since P_1 and P_2 are projective, so $\mathrm{Hom}(P_i, -)$ is exact. We then want to trace through the diagram to find the image of our map under the isomorphism in (3.11), moving from the bottom left to the top right. The first step is to pull back f_2 along the surjection $\Sigma^{-1}A \to H^*(\Sigma^{-1}M(3))$. The result is a map

$$\begin{aligned} f_3 : P_1 &\to \Sigma^{12}(\Sigma^{-1}A) \\ (1, 0, 0, \dots) &\mapsto 1 \\ (x, 0, 0, \dots) &\mapsto x \end{aligned}$$

for $x \in A$. This map sends all other generators of P_1 to 0. We note that this map is simply the projection from P_1 onto the first factor.

We next move up in the diagram to $\mathrm{Hom}^{12}(P_2, \Sigma^{-1}A)$. We define the map

$$f_4 : P_2 \to \Sigma^{12}[\Sigma^{-1}A]$$

as the composition $f_4 = f_3 \circ \partial_2$. This map on the first two generators gives:

$$\begin{aligned} f_4(1, 0, 0, \dots) &= f_3(P^1, 0, 0, \dots) = P^1 \\ f_4(0, 1, 0, \dots) &= f_3(P^3, 2P^1, 0, \dots) = P^3 \end{aligned}$$

We can use Table 3. to see the generators of the kernel of ∂_1 in order to describe f_4 in our range.

To finish moving through this diagram, the last step is to pull back along our inclusion $H^*M(3)_1 \hookrightarrow \Sigma^{-1}A$. Since f_4 maps nothing to 1 or β, we can successfully pull our map back, ending up with a map

$$f_5 : P_2 \to \Sigma^{12} H^*(M(3)_1)$$

sending $(1, 0, \dots) \mapsto P^1$ and so on as above with f_4. This map must generate a nonzero class in Ext as the class is the image of a nonzero Ext class under the connecting homomorphism (3.11) (which we know is an isomorphism).

Finally, we consider the cofibration

$$\Sigma^4 M(3) \xrightarrow{\bar{f}_{M(3)}} M(3)_1 \to Y$$

that we used to construct Y. In cohomology, we get the short exact sequence

$$0 \to H^*Y \hookrightarrow H^*M(3)_1 \twoheadrightarrow H^*(\Sigma^4 M(3)) \to 0$$

Above, in (3.2), we showed that the kernel of this surjective map is everything other than the classes $P^1\beta - \beta P^1$ and $\beta P^1 \beta$. Since the image of f_5 does not include these classes in $H^*(M(3)_1)$ we do not have any trouble pulling f_5 back into a map

$$f : P_2 \to \Sigma^{12} H^*(Y) \tag{3.12}$$

which is the correct suspension that we were looking for in order to generate our candidate for a u_2 map. The effect of this map is similar to that of f_4 and f_5. Given an element of P_2, we apply ∂_2, then project onto the first coordinate.

Proposition 3.6. *The map f we have constructed generates a nonzero class $\alpha \in Ext^{2,12}(H^*(Y), H^*(Y))$*

Proof. We first show that the composition $f \circ d_3 : P_3 \to H^*Y$ is 0. By exactness, the composition

$$P_3 \xrightarrow{\partial_3} P_2 \xrightarrow{\partial_2} P_1$$

is 0. In the process of creating f, we had a map $f_2 : P_1 \to H^*\Sigma^{-1}M(3)$. Since P_1 is projective, and $\Sigma^{-1}A \to \Sigma^{-1}M(3)$ is surjective, we lifted this map to a map $f_3 : P_1 \to \Sigma^{-1}A$. Then, the composition

$$P_3 \to P_2 \to P_1 \xrightarrow{f_3} \Sigma^{-1}A$$

must also be 0 by exactness. We then used the map $\partial_2 : P_2 \to P_1$ to construct a map $f_4 : P_2 \to \Sigma^{-1}A$. This process can be summarized by the diagram below, in which the triangles commute:

$$\begin{tikzcd}
P_3 \arrow[r, "\partial_3"] \arrow[dr, "0"] & P_2 \arrow[r, "\partial_2"] \arrow[d, dashed, "f_4"] & P_1 \arrow[dl, dashed, "f_3"] \arrow[d, "f_2"] \\
& \Sigma^{-1}A \arrow[r, two heads] & H^*(\Sigma^{-1}(M(3)))
\end{tikzcd} \tag{3.13}$$

Thus, the composition

$$P_3 \xrightarrow{\partial_3} P_2 \xrightarrow{f_4} \Sigma^{-1}A$$

must be 0 since the upper triangle commutes.

By injectivity of the maps $H^*M(3)_1 \to \Sigma^{-1}A$ and $H^*Y \to H^*M(3)_1$, we get that the compositions

$$P_3 \to P_2 \xrightarrow{f_5} H^*M(3)_1$$

and

$$P_3 \to P_2 \xrightarrow{f} H^*Y$$

must both be 0. Thus, our map f is in the kernel of ∂_3^*.

To show f gives us a nonzero Ext class, we also need to show it is not in the image of ∂_2, i.e. $f \neq h \circ \partial_2$ for some $h \in \mathrm{Hom}^{12}(P_1, H^*Y)$.

This is true for dimensional reasons. Suppose that such a nonzero h exists, and consider $(1, 0, \dots) \in P_2$. Applying ∂_2 gives us $(P^1, 0, \dots) \in P_1$ in dimension 15. The only nonzero thing this could map to under h in $\Sigma^{12}H^*Y$ is 1. However, h is an A-module map, and $(P^1, 0, \dots) = P^1(1, 0, \dots)$. But there is nothing for h to send $(1, 0, \dots)$ to in $\Sigma^{12}H^*Y$, so $h(P^1, 0, \dots) = 0$. But then we have $f(1, 0, \dots) = P^1$ and $(h \circ \partial_2)(1, 0, \dots) = 0$. Thus, f is not in the image of ∂_2, and we can conclude that f generates a nonzero class $\alpha \in \mathrm{Ext}^{2,12}(H^*Y, H^*Y)$. □

Step 3: We next want to use Theorem 2.4 to show that α and the map f give us a u_2 map. To apply the theorem, we need to examine the induced map

$$\bar{f} : \ker(\partial_1) \to H^*(Y). \tag{3.14}$$

From our construction of f above in (3.12), we can see that $\bar{f}$ takes an element of $\ker(\partial_1)$, lifts it to P_2, and applies f. However, the map f applies ∂_2, giving us the same element we started with, and then projects onto the first coordinate. The effect of this map on the first two generators of $\ker(\partial_1)$, taken from Table 3. is shown below.

$$\bar{f}(P^1, 0, \dots) = f(1, 0, \dots) = \pi_1(\partial_2(1, 0, \dots)) = \pi_1(P^1, 0, \dots) = P^1$$
$$\bar{f}(P^3, 2P^1, 0, \dots) = f(0, 1, 0, \dots) = \pi_1(\partial_2(0, 1, 0, \dots)) = \pi_1(P^3, 2P^1, 0, \dots) = P^3$$

We must show that this map is an isomorphism on x_2 homology. In order to accomplish this, we need to compute the x_2 homology of $H^*(Y)$, and subsequently $\ker \partial_1$.

From our description of the cohomology of Y, we can explicitly compute the classes that generate nonzero x_2 homology. Recall that $x_2 = P_1^0 = P^1$. The classes that are killed by x_2, but are not in the image of $(x_2)^2$ are P^1 in dimension 3, βP^2 and $2P^2\beta$ in dimension 8, and $\beta P^2 \beta$ in dimension 9. However,

$$\beta P^2 - 2P^2\beta = \beta P^2 + P^2\beta = P^1(\beta P^1),$$

so they generate the same x_2 homology class as their difference is in the image of x_2. Thus, our three nonzero classes are generated by $P^1, 2P^2\beta, \beta P^2 \beta$.

We now need to find the elements of $\ker \partial_1$ that generate its x_2 homology. We have a short exact sequence:

$$0 \to \ker \partial_0 \hookrightarrow P_0 \twoheadrightarrow H^*(Y) \to 0$$

We can take the long exact sequence in x_2 homology from this short exact sequence. Since P_0 is free over A, it has no x_2 homology, so the connecting homomorphism an isomorphism:

$$H_n(H^*(Y); x_2) \cong H'_{n+8}(H^*(\ker(\partial_0)); x_2)$$

where $H'_*(-; x_2)$ is the kernel of the action of x_2 modulo the image of $(x_2)^2$, as in Remark 1.4.

To find the image of our three classes under this isomorphism, we take the class a in $H^*(Y)$ and lift it to b in P_0. Then multiply b by $(x_2)^2$. This element must be in the kernel of ∂_0 as $(x_2)^2$ on a is 0. $(x_2)^2 b$ is automatically in the kernel of the action of x_2. And since $\partial_0(b) = a$, then $b \notin \ker \partial_0$, so $(x_2)^2 b$ is not in the image of $(x_2)^2$.

Using this process, we lift the classes $P^1, 2P^2\beta$, and $\beta P^2 \beta$. The lifts of these elements in $\ker(\partial_0)$ are

$$(2P^2, 0, 0, \dots), (2P^3\beta + \beta P^3, 0, 0, \dots), \text{ and } (2\beta P^3 \beta, 0, 0, \dots) \tag{3.15}$$

respectively. These classes generate the modified (H'_*) x_2 homology of $\ker(\partial_0)$.

We use a similar process with the short exact sequence:

$$0 \to \ker(\partial_1) \hookrightarrow P_1 \twoheadrightarrow \ker(\partial_0) \to 0$$

Taking the long exact sequence in x_2 homology again, and using the fact that P_1 is free over A, we get an isomorphism from the connecting homomorphism:

$$H'_{n+8}(\ker(\partial_0); x_2) \cong H_{n+12}(\ker(\partial_1); x_2)$$

This isomorphism works in a similar way to the one above. Take a class a in $\ker(\partial_0)$, lift it to P_1, and multiply by x_2. For similar reasons as above, this gives an element of $\ker(\partial_1)$ that generates a nonzero class in x_2 homology.

The classes that generate the x_2 homology of $\ker(\partial_0)$, given in (3.15), lift under this process, to

$$(2P^1, 0, 0, \dots), (2P^2\beta, 0, 0, \dots), \text{ and } (2\beta P^2\beta, 0, 0, \dots)$$

in $\ker(\partial_1)$, respectively. These classes generate $H_*(\ker(\partial_1); x_2)$.

Now, since our map $\bar{f} : \ker(\partial_1) \to \Sigma^{12} H^*(Y)$ is just a projection onto the first coordinate, we can see its effect on the x_2 homology generators:

$$\bar{f}(2P^1, 0, 0, \dots) = 2P^1$$
$$\bar{f}(2P^2\beta, 0, 0, \dots) = 2P^2\beta$$
$$\bar{f}(2\beta P^2\beta, 0, 0, \dots) = 2\beta P^2\beta$$

Then $\bar{f}$ gives us an isomorphism (though not the identity) on the x_2 homology. By Theorem 2.4, this tells us that some power of α is a power of a u_2 map.

Step 4: We must now show that our element $\alpha \in \mathrm{Ext}^{2,12}(H^*(Y), H^*(Y))$ survives to the E_∞ page of the Adams spectral sequence to give us a self map on the spectrum Y. In terms of the $t-s, s$ coordinates on the E_2 page, we have a nonzero element at $s = 2, t - s = 10$. The differential d_r on this class is a map

$$d_r : \mathrm{Ext}^{2,12}(H^*(Y), H^*(Y)) \to \mathrm{Ext}^{2+r,12+r-1}(H^*(Y), H^*(Y))$$

That is, it takes our element in $s = 2, t - s = 10$ to $s = 2 + r, t - s = 9$. We'd like to show all of the Ext classes in the $t - s = 9$ column are zero for $r \geq 2$. We make a similar argument as for our u_1 map above.

We've constructed the first part of a minimal resolution for $H^*(Y)$, and we've shown in (3.7) that $\mathrm{Ext}^{2,15}(H^*(Y), \mathbb{Z}_3)$ is the first nonzero Ext class for $s = 2$. By Lemma 1.7, since $H^*(Y)$ has no x_0 or x_1 homology, it is free over A through degree 3, as P^1 in degree 4 is not in the sub Hopf algebra of A containing x_0 and x_1. Thus, every increase by 1 of s corresponds to an increase in the bottom degree of P_s by at least 4, giving us the vanishing line in Figure 2. below. Everything above and to the left of the solid line must be 0.

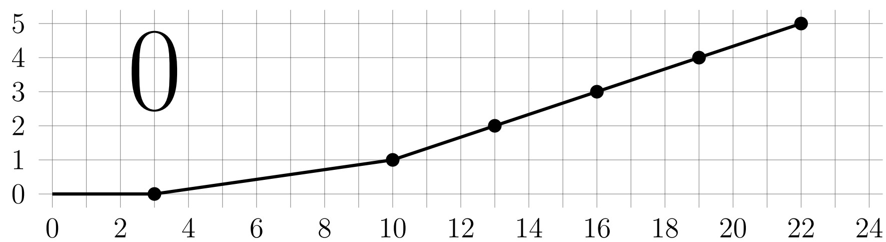

FIGURE 2. Vanishing edge for $\mathrm{Ext}^{s,t}(H^*(Y), \mathbb{Z}_3)$ in $(t - s, s)$ coordinates

These Ext classes come from maps sending some element of P_s to $1 \in \Sigma^t \mathbb{Z}_3$. If we want to compute $\mathrm{Ext}^{s,t}(H^*Y, H^*M(3))$, the "vanishing edge" that we are finding moves to the left by one. This is since we could send the same element in P_2 to the class $\beta \in \Sigma^{t-1} H^*M(3)$. This means for $s = 0$, our first nonzero class happens when $t = 2$. For $s = 1, 2, 3, 4$, the smallest nonzero classes could be in dimensions $10, 14, 18, 22$, respectively. This is summed up in Figure 3. below. As before, everything above and to the left of the line must be zero.

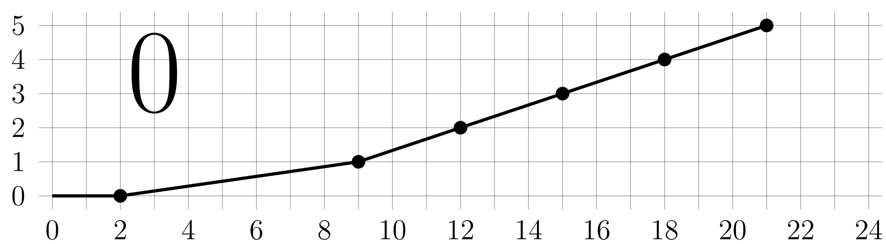

FIGURE 3. Vanishing edge for $\mathrm{Ext}^{s,t}(H^*(Y), H^*(M(3)))$ in $(t - s, s)$ coordinates

Like we did in computing our u_2 map, we can use the cofibrations from the construction of the spectrum Y to eventually get information about the vanishing edge for $\mathrm{Ext}^{s,t}(H^*(Y), H^*(Y))$. We have a cofibration

$$M(3) \to K(\mathbb{Z}_3) \to \Sigma M(3)_1,$$

which gives rise to a short exact sequence in cohomology:

$$0 \to H^*(\Sigma M(3)_1) \to A \to H^*(M(3)) \to 0.$$

This, in turn gives rise to a long exact sequence in Ext. The connecting homomorphism from this long exact sequence is given below:

$$\to \mathrm{Ext}_A^{s,t}(H^*(Y), H^*(M(3))) \xrightarrow{\delta} \mathrm{Ext}_A^{s+1,t}(H^*(Y), H^*(\Sigma M(3)_1)) \to$$

For $s \geq 1$, the terms on either side of these, $\mathrm{Ext}_A(H^*(Y), A)$, are 0, so the connecting homomorphism δ is an isomorphism. For $s = 0$, it is a surjection as the next term to the right is 0. We also note that

$$\mathrm{Ext}_A^{s+1,t}(H^*(Y), H^*(\Sigma M(3)_1)) \cong \mathrm{Ext}_A^{s+1,t+1}(H^*(Y), H^*(M(3)_1)). \tag{3.16}$$

This lets us find a vanishing edge for $\mathrm{Ext}_A^{s,t}(H^*(Y), H^*(M(3)_1))$. From our observations about the connecting homomorphism, the vanishing edge will be shifted up by one (in $(t-s, s)$ coordinates) from that of $\mathrm{Ext}_A^{s,t}(H^*(Y), H^*(M(3)))$. However, we can't say anything about the new information that "shows up" in the $s = 0$ row. The new vanishing edge is shown in 4. below, again everything above and to the left is zero:

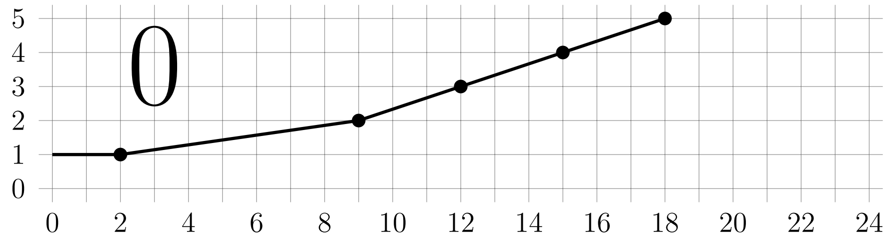

FIGURE 4. Vanishing edge for $\mathrm{Ext}^{s,t}(H^*(Y), H^*(M(3)_1))$ in $(t-s, s)$ coordinates

Finally, we use the cofibration

$$\Sigma^4 M(3) \to M(3)_1 \to Y,$$

and the corresponding short exact sequence in cohomology. Then in Ext, we have a long exact sequence, part of which is shown below:

$$\mathrm{Ext}_A^{s-1,t}(H^*Y, H^*\Sigma^4 M(3)) \to \mathrm{Ext}_A^{s,t}(H^*Y, H^*Y) \to \mathrm{Ext}_A^{s,t}(H^*Y, H^*M(3)_1) \quad (3.17)$$

From this, we know that if both of the end groups are zero for a particular pair of s, t, then the middle term must be zero as well. Adding in the Σ^4 in the first term shifts everything from Figure 3. to the left by 4, and the extra shift to account for the $s-1$ moves the diagram up and to the left by 1 more, giving us a vanishing edge shown in Figure 5. below:

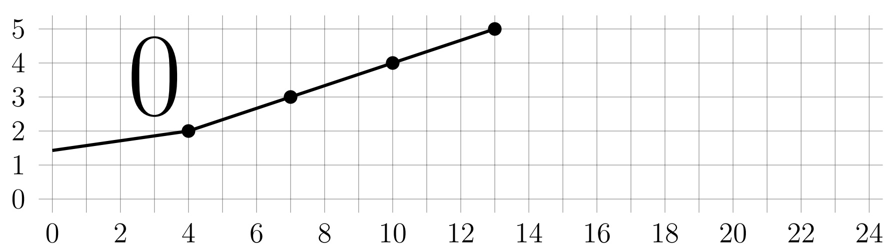

FIGURE 5. Vanishing edge for $\mathrm{Ext}^{s-1,t}(H^*(Y), H^*(\Sigma^4 M(3)))$ in $(t - s, s)$ coordinates

Finally, we note that the intersection of the "vanishing regions" of Figure 4. and Figure 5. gives us at least part of the vanishing region for $\mathrm{Ext}^{s,t}(H^*(Y), H^*(Y))$. In this instance, the vanishing region in Figure 5. lies entirely inside of the vanishing region in Figure 4., so this is our intersection. Figure 6. below provides the vanishing region again with added information about the class α, and the location of the first few differentials we are concerned with.

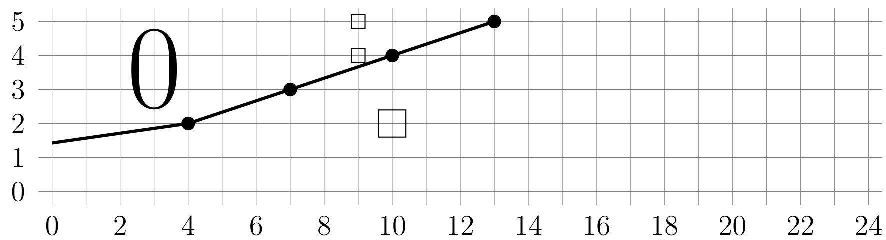

FIGURE 6. Vanishing edge for $\mathrm{Ext}^{s,t}(H^*(Y), H^*(Y))$ in $(t - s, s)$ coordinates. The large open square represents our class α, and the smaller open squares show the locations of the image of α under the differentials d_2 and d_3.

Since these differentials land in the vanishing region of our diagram, we can conclude that $d_r(\alpha) = 0$ for $r \geq 2$, and that our class survives the Adams spectral sequence, and gives us a map $f_Y : \Sigma^{10}Y \to Y$.

Step 5: Since our map f_Y comes from an element of Ext^2, we know that we will be able to lift it two stages up the Adams resolution. This is represented by the figure below:

$$\begin{tikzcd}
 & Y_2 \arrow[d] & \\
 & Y_1 \arrow[r] \arrow[d] & K_1 \\
\Sigma^{10}Y \arrow[r, "f_Y"] \arrow[uur, dashed, "\bar{f}_Y"] & Y \arrow[r] & K_0
\end{tikzcd} \tag{3.18}$$

Here, the spectra K_0 and K_1 are wedges of copies of $K(\mathbb{Z}_3)$, and the cohomology of K_0 and K_1 are isomorphic, as A modules, to P_0 and P_1 as described in (3.4) and (3.5) in the resolution of $H^*(Y)$.

Similarly to the u_1 case above, the cohomology of Y_2 corresponds to a shift down by two dimensions of $\ker \partial_1$. The induced map in cohomology, $\bar{f_Y}^*$ is the desuspension (by two dimensions) of $\bar{f} : \ker \partial_1 \to \Sigma^{12} H^*(Y)$ as defined in (3.14), and we have shown that it is an isomorphism in x_2 homology.

Step 6: We follow the same method as with our u_1 map to construct a spectrum, Z, with no x_2 homology by taking the cofiber of $\bar{f}_Y$. In the dimensions we are considering, $\bar{f}_Y^*$ is a surjection, so the cohomology of Z is just the kernel of $\bar{f_Y}^*$.

Unfortunately, unlike the previous case, we do not have a complete description of $H^*(Y_2)$ in order to compute $H^*(Z)$. However, we do know that $\bar{f_Y}^*$ is a map that projects to the first coordinate, so all of the generators of $H^*(Y_2)$ with first coordinate 0 will be in the kernel. Recall that these generators are the same as those in Table 3., but shifted down in dimension by 2. Those generators that are nonzero in the first coordinate will not be in the kernel of $\bar{f}_Y^*$. There is a possibility that we gain some extra generators that are A linear combinations of these two types of generators. These are detected using the same SAGE computation procedure as before. The module generators of $H^*(Z)$ up through dimension 70 are given in Table 4. below. Generators not coming directly from a generator of $H^*(Y_2)$ are denoted with a $*$. As before, the last entry in each tuple is the last nonzero entry.

TABLE 4. Generators of $H^*(Z)$ and their dimensions

Dimension	Generator
29	$(0, P^3, 0, 2P^1)$
34	$(0, \beta P^3 P^1, P^3 P^1 + P^4, 0, 2P^2)$
35	$(0, \beta P^3 P^1 \beta, 2P^3 P^1 \beta + \beta P^4, 0, 2\beta P^2, P^1)$
37	$(0, 0, 0, P^3, 0, 0, P^1)$
42	$(0, P^5 P^1 \beta + P^6 \beta, P^5 P^1 + P^6, \beta P^3 P^1, 2P^3 P^1 + 2P^4, 0, P^2 \beta)$
43	$(0, 0, P^5 P^1 \beta + 2P^6 \beta + \beta P^5 P^1 + \beta P^6, 0, 2P^4 \beta, 2P^3, 2\beta P^2 \beta)$
48	$(0, 0, 0, \beta P^4 \beta P^1 \beta, 2P^4 \beta P^1 \beta + 2\beta P^4 P^1 \beta, P^3 P^1 \beta + 2P^4 \beta$ $+\beta P^3 P^1 + 2\beta P^4)$
49*	$(0, P^6 P^2, 0, P^5, P^1, 0, 0, 2P^4)$
50	$(0, 0, 2P^6 P^2 + P^7 P^1, P^5 P^1 \beta + P^6 \beta, P^5 P^1 + P^6, 0, 2P^3 P^1 \beta$ $+P^4 \beta + 2\beta P^3 P^1 + \beta P^4)$
57	$(0, 0, 0, 0, 0, 0, P^6)$
69*	$(0, P^9 P^3 P^1 + P^{12} P^1, 0, 0, 0, 0, P^7 P^2, 2P^6 P^2)$
69	$(0, 2P^9 P^3 P^1 + P^{13}, 0, 2P^9 P^2, 0, 0, 2P^9, P^6 P^2 + 2P^7 P^1$ $+2P^8, 2P^3)$
70	$(0, P^9 P^3 P^1 \beta + P^{12} P^1 \beta, 2P^{12} P^1 + 2P^{13}, 0, P^9 P^2 + 2P^{10} P^1,$ $0, P^7 \beta P^2 + 2P^9 \beta + \beta P^7 P^2, P^6 P^2 \beta + P^7 P^1 \beta$ $+P^7 \beta P^1 + \beta P^6 P^2, 2P^3 \beta, 2P^2)$

Constructing a u_3 Map

Step 0: By construction, we know that $H^*(Z)$ has no x_0, x_1 or x_2 homology, so Proposition 1.13 tells us that we will be able to find some power of a u_3 map on this spectrum.

Step 1: We first need to compute $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Z))$ in order to find our candidate for a u_3 map. As before, we will accomplish this by constructing a projective resolution, $R_\bullet$, of $H^*(Z)$. Since $x_3 = Q_2$, we know that u_3 is in homological degree 1. This means we may be able to construct our u_3 map after only computing R_0 and R_1.

Based on Table 4., we know that the initial step in the resolution, R_0, will have one copy of A for each of the generators in the table, i.e.

$$R_0 = \Sigma^{29}A \oplus \Sigma^{34}A \oplus \Sigma^{35}A \oplus \Sigma^{37}A \oplus \dots \tag{3.19}$$

Let ∂_0' be the map $R_0 \to H^*(Z)$. As before, we use Sage to compute the kernel of ∂_0' and to find its A module generators. These generators and their dimensions are in Table 5.. As before, the last entry in each tuple is the last nonzero entry.

As before, we construct R_1 to be a free A module with a generator for each generator of $\ker(\partial_0')$, so

$$R_1 = \Sigma^{46}A \oplus \Sigma^{51}A \oplus \Sigma^{52}A \oplus \dots \tag{3.20}$$

We'll show that this is sufficient to construct a u_3 map on Z.

TABLE 5. Generators of $\ker(\partial_0')$ and their dimensions

Dimension	Generator
46	$(2P^4\beta + \beta P^3P^1 + \beta P^4, P^3, 0, \beta P^2, 2P^1)$
51	$(0, P^3P^1\beta + 2P^4\beta + \beta P^3P^1 + \beta P^4, 2P^3P^1 + 2P^4, 0, 2P^2\beta, 2P^2)$
52	$(\beta P^4\beta P^1\beta, 0, 2P^3P^1\beta + 2P^4\beta + 2\beta P^3P^1, 0, 0, P^2\beta + 2\beta P^2, P^1)$
53	$(P^5P^1 + P^6, 0, 0, P^3P^1 + P^4, 0, 0, 0, 2P^1)$
54	$(P^5P^1\beta, 2P^4P^1 + P^5, 0, 2P^4\beta + 2\beta P^3P^1 + 2\beta P^4, 2P^3, 0, 0, P^1\beta, 2P^1)$
59	$(0, P^5P^1\beta + 2P^6\beta, P^5P^1 + P^6, \beta P^5\beta, P^4\beta, P^3P^1 + P^4, 0, 0, \beta P^2)$
60	$(0, 0, P^5\beta P^1 + \beta P^6, 2\beta P^4\beta P^1\beta, 0, P^3P^1\beta + 2P^4\beta + 2\beta P^3P^1, P^3, 0, 2\beta P^2\beta)$
61	$(0, 0, 0, P^5P^1 + P^6, 0, 0, 0, P^3, 0, P^1)$
65	$(0, 0, P^6\beta P^1\beta, 0, \beta P^4\beta P^1\beta, P^4\beta P^1\beta + \beta P^4P^1\beta + 2\beta P^5\beta, 2P^3P^1\beta + 2P^4\beta)$
66	$(0, P^8, 0, P^6P^1\beta + P^7\beta, 2P^6, 0, 0, P^4\beta + \beta P^4, P^3P^1 + 2P^4, P^2\beta + \beta P^2)$
67	$(P^7\beta P^2\beta, 2P^6P^2\beta, P^7P^1, 0, 2\beta P^5P^1, 2P^5P^1 + 2P^6, 0, 2\beta P^4\beta, P^3P^1\beta + 2P^4\beta + \beta P^4)$
69	$(0, 0, 0, 0, 0, 0, 0, P^5, 0, P^3)$

Step 2: We'd like to construct an element in $\mathrm{Ext}_A^{1,17}(H^*(Z), H^*(Z))$ that is a u_3 map. To do this, we need a map from R_1 to $\Sigma^{17}H^*(Z)$.

Let Y_1, Y_2, K_0 and K_1 be the spectra in the construction from the Adams resolution for Y as defined in (3.18). We begin to construct our map by defining a map:

$$g_1 : H^*(K_1) \to \Sigma^{16}H^*(K_0)$$

We define this map on a small number of the generators of $H^*(K_1)$ as outlined in Table 6. below. In this, and the following tables, generator i refers to the tuple with 1 in the ith spot, and 0's in all other spots.

The map g_1 sends the rest of the generators of $H^*(K_1)$ to 0.

TABLE 6. Image of the map g_1 in $\Sigma^{16} H^*(K_0)$

Dimension	Generator #	Image under g_1
19	3	$(1, 0, 0, 0)$
27	5	$(0, 1, 0, 0)$
32	6	$(0, P^1\beta, 1, 0)$
63	10	$(0, 2P^9, 0, 2P^3)$

As before, we move through a few exact sequences to end up with the map that we originally wanted. The first step involves the inclusion

$$H^*(\Sigma Y_2) \hookrightarrow H^*(K_1)$$

from the cofibration in constructing the Adams resolution, which gives us a map

$$\mathrm{Hom}^{16}(H^*(K_1), H^*(K_0)) \to \mathrm{Hom}^{16}(H^*(\Sigma Y_2), H^*(K_0)). \tag{3.21}$$

Further, we have another inclusion

$$H^*(\Sigma Z) \hookrightarrow H^*(\Sigma Y_2)$$

from the construction of Z, which gives us a map

$$\mathrm{Hom}^{16}(H^*(\Sigma Y_2), H^*(K_0)) \to \mathrm{Hom}^{16}(H^*(\Sigma Z), H^*(K_0)). \tag{3.22}$$

Putting together (3.21) and (3.22), since $H^*(\Sigma Z)$ is contained in $H^*(K_1)$, g_1 is defined on $H^*(\Sigma Z)$ by restriction. Finally, a map from $H^*(Z)$ gives us a map from R_0 by composing with ∂_0'. Thus, given an element in R_0, we can apply the differential to obtain an element of $H^*(Z)$, which in turn gives us an element of

$H^*(Y_2)$ via inclusion, and then an element of $H^*(\Sigma^{-1}K_1)$. Let

$$g_2 : \Sigma R_0 \to \Sigma^{16} H^*(K_0)$$

be the composition of these maps and g_1 as defined above. This map is defined on the first few generators of ΣR_0 according to Table 7. below. The later generators

TABLE 7. Image of the map g_2

dimension	generator #	Image under g_2 in $\Sigma^{16}H^*(K_0)$
30	1	0
35	2	$(P^3P^1 + P^4, 2P^2, 0, 0)$
36	3	$(2P^3P^1\beta + \beta P^4, 2P^2\beta + 2\beta P^2, P^1, 0)$
38	4	0
43	5	$(P^5P^1 + P^6, 2P^3P^1 + 2P^4, 0, 0)$
44	6	$(P^5P^1\beta + 2P^6\beta + \beta P^5P^1 + \beta P^6, 2P^3P^1\beta$ $+2P^4\beta, 2P^3, 0)$
49	7	$(0, P^4\beta P^1\beta + \beta P^4P^1\beta, P^3P^1\beta + 2P^4\beta$ $+\beta P^3P^1 + 2\beta P^4, 0)$
50	8	0
51	9	$(2P^6P^2 + P^7P^1, P^5P^1 + P^6, 0, 0)$
58	10	0
70	11	0
70	12	0
71	13	$(2P^{12}P^1 + 2P^{13}, P^9P^2 + 2P^{10}P^1 + P^{11}, 0, P^5)$

do not necessarily go to zero, but they are not needed for the calculations that follow.

In each dimension, we can computationally verify that the image of the generator in $H^*(K_0)$ is actually an element of $H^*(\Sigma Y_1)$. This gives us a map:

$$g_3 : \Sigma R_0 \to \Sigma^{17} H^*(Y_1)$$

which we can desuspend to

$$\Sigma^{-1}g_3 : R_0 \to \Sigma^{16}H^*(Y_1)$$

Now, from the cofibration

$$Y_1 \to K_1 \to \Sigma Y_2$$

we have the short exact sequence in cohomology

$$0 \to H^*(\Sigma Y_2) \hookrightarrow H^*(K_1) \twoheadrightarrow H^*(Y_1) \to 0.$$

Further, there is a corresponding long exact sequence from applying the functor $\mathrm{Ext}_A(H^*(Z), -)$. The connecting homomorphism in this long exact sequence is

$$\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_1)) \to \mathrm{Ext}_A^{s+1,t}(H^*(Z), H^*(\Sigma Y_2))$$

We can further apply an isomorphism similar to (3.16) to treat this connecting homomorphism as:

$$\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_1)) \to \mathrm{Ext}_A^{s+1,t+1}(H^*(Z), H^*(Y_2))$$

By working through this connecting homomorphism, illustrated in the diagram below, we take our element of $\mathrm{Ext}_A^{0,16}(H^*(Z), H^*(Y_1))$ and produce an element of $\mathrm{Ext}_A^{1,17}(H^*(Z), H^*(Y_2))$, which is very close to the desired Ext group.

$$
\begin{tikzcd}
0 & \mathrm{Hom}^{16}(R_1, H^*(Y_1)) \arrow[l] & \mathrm{Hom}^{16}(R_1, H^*(K_1)) \arrow[l] & \mathrm{Hom}^{16}(R_1, H^*(\Sigma Y_2) \arrow[l] & 0 \arrow[l] \\
0 & \mathrm{Hom}^{16}(R_0, H^*(Y_1)) \arrow[l] \arrow[u, "\partial_1'"] & \mathrm{Hom}^{16}(R_0, H^*(K_1)) \arrow[l] \arrow[u, "\partial_1'"] & \mathrm{Hom}^{16}(R_0, H^*(\Sigma Y_2) \arrow[l] \arrow[u, "\partial_1'"] & 0 \arrow[l]
\end{tikzcd}
$$

For each generator of R_0, we look at the image in $H^*(Y_1)$ and find a lift of that element in $H^*(K_1)$. Since $H^*(K_1) \to H^*(Y_1)$ is surjective, we can make this lift. It can be verified computationally that these lifts are also unique. This gives us a map

$$g_4 : R_0 \to \Sigma^{16} H^*(K_1)$$

As above in our u_2 map computations, we compose with the differential ∂_1' from the $R_\bullet$ resolution in order to form

$$g_5 : R_1 \to \Sigma^{16} H^*(K_1)$$

Again, we can computationally verify that the image of this map lies entirely in the kernel of $H^*(K_1) \to H^*(Y_1)$, so we can pull back to a map

$$g_6 : R_1 \to \Sigma^{16} H^*(\Sigma Y_2) = \Sigma^{17} H^*(Y_2)$$

This map, g_6, is summarized in Table 8. below, where we show the image of the first few generators of R_1. We only need the first 9 generators for later computations. As above, the remaining generators do not necessarily go to zero, but they are not needed for the computations in the next section.

TABLE 8. Image of the map g_6 in relevant dimensions

Dimension	Generator #	Image under g_6 in $\Sigma^{17}H^*(Y_2)$
46	1	$(0, P^3, 0, 2P^1)$
51	2	$(0, P^4\beta + \beta P^3P^1 + \beta P^4, P^3P^1 + P^4, 2P^2\beta, 2P^2)$
52	3	$(0, 2\beta P^3P^1\beta, P^3P^1\beta + \beta P^3P^1, 0, 0, 2P^1)$
53	4	0
54	5	$(0, 2P^5, 0, 2P^3, 0, 0, 2P^1)$
59	6	$(0, 2P^5P^1\beta + 2P^5\beta P^1 + 2P^6\beta + 2\beta P^5P^1 +\beta P^6, P^5P^1 + 2P^6, P^4\beta, P^3P^1 + P^4, 0, \beta P^2)$
60	7	$(0, 2\beta P^5\beta P^1 + \beta P^6\beta, P^5P^1\beta + P^5\beta P^1 + P^6\beta +2\beta P^6, 0, 2P^4\beta, 2P^3, 2\beta P^2\beta)$
61	8	0
65	9	$(0, \beta P^6\beta P^1\beta, 2\beta P^6P^1\beta, \beta P^4\beta P^1\beta, 2P^4\beta P^1\beta +2\beta P^4P^1\beta + \beta P^5\beta, P^3P^1\beta + P^4\beta)$

Now, recall the short exact sequence from the construction of Z

$$0 \to H^*(Z) \hookrightarrow H^*(Y_2) \xrightarrow{\bar{f_Y}^*} \Sigma^{10}H^*(Y) \to 0$$

the surjective map in the sequence above is a projection onto the first coordinate. All of the images in Table 8. have a zero first coordinate, so they are all in the kernel of the suspension of $\bar{f_Y}^*$. This means these elements are in $H^*(Z)$. Thus, we have defined the desired map $g : R_1 \to \Sigma^{17}H^*(Z)$.

Before checking to see if we have a u_3 map, we need to make sure that g generates a nonzero class in Ext. A similar diagram chase as in (3.13) from our u_2 construction tells us that the class g generates in Ext is in the kernel of ∂_2', so we must verify it is not the image of some map, h, in $\mathrm{Hom}^{17}(R_0, H^*(Z))$.

Consider the first generator in R_1, in dimension 46. It maps, via ∂_1' to the element $(2P^4\beta+\beta P^3P^1+\beta P^4, P^3, 0, \beta P^2, 2P^1)$ in R_0. In order for g to be the image of h, we would need for this element to also map to $(0, P^3, 0, 2P^1) \in \Sigma^{17}H^*(Z)$.

However, from (3.19), we know the first five generators of R_0 are in dimensions $29, 34, 36, 37$, and 42. These generators would need to map to elements in the same dimensions in $\Sigma^{17} H^*(Z)$, corresponding to the dimensions $12, 17, 19, 20$, and 25 in $H^*(Z)$. However, there is nothing in $H^*(Z)$ in these dimensions, so the bottom five generators would necessarily map to 0. This is a contradiction as an A linear combination of these generators must map to something nonzero.

Thus, g generates a nonzero class, $\alpha \in \mathrm{Ext}_A^{1,17}(H^*(Z), H^*(Z))$. It remains to show this element gives us a u_3 map.

Step 3: We must verify that g induces a map $\bar{g} : \ker \partial_0' \to \Sigma^{17} H^*(Z)$ that is an x_3 homology isomorphism. In order to check this, we must determine the x_3 homology of $H^*(Z)$, which we can then use to find the x_3 homology of $\ker \partial_0'$.

As before, we can start with the x_3 homology of $H^*(Y)$ since the cohomology is easy to describe. The nonzero x_3 homology of $H^*(Y)$ is generated by x_3 applied to the four "missing" cohomology classes: $1, \beta, P^1\beta, \beta P^1 \beta$. These classes are generated by the elements:

$$\begin{gathered} P^3P^1\beta + 2P^4\beta + 2\beta P^3P^1 + \beta P^4, \qquad \beta P^3 P^1 \beta + 2\beta P^4 \beta, \\ P^4\beta P^1 \beta + 2\beta P^5 \beta, \qquad \beta P^4 \beta P^1 \beta \end{gathered} \tag{3.23}$$

which are in dimensions $16, 17, 21$, and 22 respectively.

To find the x_3 homology of $H^*(Z)$ we will need to compute the x_3 homology of $H^*(Y_2)$, since Z is the cofiber of the map $\Sigma^{10} Y \to Y_2$. We have a short exact sequence in cohomology from the construction of the Adams resolution of Y:

$$0 \to H^*(\Sigma Y_1) \hookrightarrow H^*(K_0) \twoheadrightarrow H^*(Y) \to 0$$

We then look at the long exact sequence induced here in x_3 homology. Since $(x_3)^2 = 0$, we don't need to use $H'_*(-;x_3)$, our long exact sequence just uses the $H_*(-;x_3)$ form instead. Since $H^*(K_0)$ is a sum of copies of A, and A has no x_3 homology, then we get an isomorphism from the connecting homomorphism of the long exact sequence:

$$H_n(H^*(Y);x_3) \cong H_{n+17}(\Sigma H^*(Y_1);x_3) \cong H_{16}(H^*(Y_1);x_3)$$

The process for computing the isomorphism is as follows. Take $x \in H^*(Y)$ that generates nonzero x_3 homology, and lift to some $y \in H^*(K_0)$. Then by construction x_3y maps to 0 in $H^*(Y)$, so x_3y pulls back to an element of $H^*(\Sigma Y_1)$. Then, $x_3(x_3y) = 0$ in $H^*(\Sigma Y_1)$, and $y \notin H^*(\Sigma Y_1)$ as it is not in the kernel of the surjection. Then x_3y generates a nonzero element of x_3 homology in $H^*(\Sigma Y_1)$. Following this process with the four elements listed above, we get the following four generators of the x_3 homology of $H^*(Y_1)$, shown in Table 9..

TABLE 9. Generators of the x_3 homology of $H^*(Y_1)$

Dimension	Element
32	$(\beta P^6P^1\beta + 2\beta P^6\beta P^1 + 2\beta P^7\beta, P^4\beta P^1\beta + 2\beta P^4P^1\beta$ $+\beta P^4\beta P^1, P^3P^1\beta + 2P^4\beta + 2\beta P^3P^1 + \beta P^4, 0)$
33	$(0, 2\beta P^4\beta P^1\beta, 2\beta P^3P^1\beta + \beta P^4\beta)$
37	$(0, 2\beta P^5\beta P^1\beta, 2P^4\beta P^1\beta + \beta P^5\beta, 0)$
38	$(0, 0, \beta P^4\beta P^1\beta, 0)$

An identical procedure lets us lift these elements to the generators of the x_3 homology of $H^*(Y_2)$, which are summarized in Table 10. below.

TABLE 10. Generators of the x_3 homology of $H^*(Y_2)$

Dimension	Element
48	$(0,0,2\beta P^6P^1\beta+\beta P^6\beta P^1+\beta P^7\beta, 2\beta P^4\beta P^1\beta,$ $P^4\beta P^1\beta+\beta P^4P^1\beta+2\beta P^4\beta P^1+\beta P^5\beta,$ $2P^3P^1\beta+P^4\beta+\beta P^3P^1+2\beta P^4, 0, 0)$
49	$(0,0,0,0,\beta P^4\beta P^1\beta, 2\beta P^3P^1\beta+\beta P^4\beta, 0, 0)$
53	$(0,0,0,0,\beta P^5\beta P^1\beta, 2P^4\beta P^1\beta+\beta P^5\beta, 0, 0)$
54	$(0,0,0,0,0,2\beta P^4\beta P^1\beta, 0, 0)$

Now we have the x_3 homology generators for both $H^*(Y)$ and $H^*(Y_2)$. We have a short exact sequence from the construction of Z

$$0 \to H^*(Z) \hookrightarrow H^*(Y_2) \xrightarrow{\bar{f_Y}^*} H^*(\Sigma^{10}Y) \to 0 \tag{3.24}$$

The surjective map above is projection onto the first coordinate. Since all of our generators in the table above have a zero first coordinate, they are all in $H^*(Z)$. Further, they are in the kernel of x_3 but not the image, so they generate nonzero x_3 homology classes of $H^*(Z)$. However, there are other pieces to the x_3 homology of $H^*(Z)$. The short exact sequence in (3.24) gives us a long exact sequence in x_3 homology. Part of this long exact sequence is given below:

$$\begin{array}{ccccc} H_n(H^*(Y_2);x_3) & & & & \\ \downarrow & & & & \\ H_n(H^*(\Sigma^{10}Y);x_3) & \longrightarrow & H_{n+17}(H^*(Z);x_3) & \longrightarrow & H_{n+17}(H^*(Y_2);x_3) \\ & & & & \downarrow \\ & & & & H_{n+17}(H^*(\Sigma^{10}Y);x_3) \end{array}$$

Since we have computed the x_3 homology of $H^*(Y)$ and $H^*(Y_2)$, we can say that the two vertical maps in the diagram above must be zero for dimensional reasons. That is, the x_3 homology of $H^*(Y_2)$ is in dimensions $48, 49, 53$, and 54, while the x_3 homology of $H^*(\Sigma^{10}Y)$ is in dimensions $26, 27, 31$, and 32. Thus, we have a short exact sequence:

$$0 \to H_n(H^*(\Sigma^{10}Y); x_3) \hookrightarrow H_{n+17}(H^*(Z); x_3) \twoheadrightarrow H_{n+17}(H^*(Y_2); x_3) \to 0 \qquad (3.25)$$

Further, the inclusion here is really the connecting homomorphism in the long exact sequence, and we use the same process here that we did to compute the x_3 homology of $H^*(Y_1)$.

We take an element of $H^*(\Sigma^{10}Y)$ that generates a nonzero x_3 homology class, lift it to an element of $H^*(Y_2)$, then multiply by x_3 on the left. As in our explanation before, this will give a nonzero element of x_3 homology for $H^*(Z)$. These elements are given in Table 11. below.

TABLE 11. Additional generators of x_3 homology of $H^*(Z)$

Dimension	Element
43	$(0, P^5\beta P^1\beta + 2\beta P^6\beta, 2P^5P^1\beta + P^5\beta P^1 + P^6\beta + 2\beta P^6)$
44	$(0, 2\beta P^6\beta P^1\beta, \beta P^6 P^1\beta + 2\beta P^6\beta P^1 + 2\beta P^6\beta)$
48	$(0, \beta P^6\beta P^1\beta, P^6\beta P^1\beta + 2\beta P^6 P^1\beta + \beta P^6\beta P^1)$
49	$(0, 0, 2\beta P^6\beta P^1\beta)$

We now have generators of eight distinct nonzero x_3 homology classes for $H^*(Z)$. We note that for dimensional reasons, the pairs of classes in dimensions 48 and 49 cannot generate the same class in x_3 homology as $H^*(Z)$ contains nothing in dimensions 31 or 32, so their difference cannot be in the image of multiplication by x_3.

From our resolution of $H^*(Z)$, we have a short exact sequence

$$0 \to \ker \partial_0' \hookrightarrow R_0 \xrightarrow{\partial_0'} H^*(Z) \to 0$$

Then, since R_0 has no x_3 homology, in the long exact sequence in x_3 homology, we have an isomorphism

$$H_n(H^*(Z); x_3) \cong H_{n+17}(\ker \partial_0'; x_3)$$

We can then compute the x_3 homology of $\ker \partial_0'$ by lifting the elements of $H^*(Z)$ to R_0 and then multiplying by x_3. The results are summarized in Table 12. below.

TABLE 12. Generators of x_3 homology of $\ker(\partial_0')$. The first four generators are the ones lifted through the connecting homomorphism in (3.25). The last four are the ones lifted up the Adams resolution, listed in Table 10.

dimension	element
60	$(2\beta P^6\beta P^1\beta, 2P^5\beta P^1\beta + \beta P^6\beta, P^5P^1\beta + 2P^5\beta P^1 + 2P^6\beta$ $+\beta P^6, 0)$
61	$(0, 2\beta P^5\beta P^1\beta, \beta P^5P^1\beta + 2\beta P^5\beta P^1 + 2\beta P^6\beta, 0)$
65	$(0, \beta P^6\beta P^1\beta, P^6\beta P^1\beta + 2\beta P^6P^1\beta + \beta P^6\beta P^1, 0)$
66	$(0, 0, \beta P^6\beta P^1\beta, 0)$
65	$(0, \beta P^6\beta P^1\beta, \beta P^6P^1\beta + 2\beta P^6\beta P^1 + 2\beta P^7\beta, 0, \beta P^4\beta P^1\beta,$ $\beta P^4P^1\beta + 2\beta P^4\beta P^1 + 2\beta P^5\beta, 2P^3P^1\beta + P^4\beta + \beta P^3P^1$ $+2\beta P^4)$
66	$(0,0,0,0,0,0,\beta P^3P^1\beta + 2\beta P^4\beta)$
70	$(0,0,0,0,0,0,P^4\beta P^1\beta + 2\beta P^5\beta)$
71	$(0,0,0,0,0,0,2\beta P^4\beta P^1\beta)$

Finally, we can compute the images of each of these elements under the map $\bar{g}$. As we hoped, $\bar{g}$ maps each of these generators to either the corresponding element in $H^*(Z)$ or to twice that element, which is an isomorphism on x_3 homology. This tells us, by Theorem 2.4, that our element of $\mathrm{Ext}_A^{1,17}(H^*(Z), H^*(Z))$ is, in fact, a u_3 map.

Step 4: The last step is to show that this element of Ext survives the Adams Spectral Sequence in order to give us a self map $\Sigma^{16}Z \to Z$. We make a similar argument as the one for our u_2 map above using vanishing lines.

By our computations summarized in Table 4., we have shown that our groups $\mathrm{Ext}_A^{0,t}(H^*(Z), \mathbb{Z}_3)$ are zero for $t < 29$, and from (3.20), that $\mathrm{Ext}_A^{1,t}(H^*(Z), \mathbb{Z}_3)$ is zero for $t < 46$. For larger values of s, we use Lemma 1.7 again. $H^*(Z)$ has no x_0, x_1, or x_2 homology, so the Lemma tells us that it is free through degree 11, as P^3 is the element of smallest degree not in the sub Hopf algebra containing x_0, x_1 and x_2. Thus, each time s increases by 1, t must increase by at least 12, or $t-s$ must increase by at least 11. In Figure 7., everything above and to the left of the solid line must be 0, including the line $s = 0$ for $t - s < 29$.

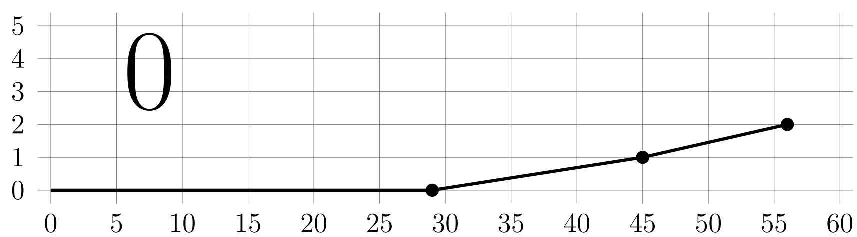

FIGURE 7. Vanishing edge for $\mathrm{Ext}_A^{s,t}(H^*(Z), \mathbb{Z}_3)$ in $(t-s, s)$ coordinates

We then proceed with our construction of vanishing regions as in the case for our u_2 map. Moving from $\mathrm{Ext}_A^{s,t}(H^*(Z), \mathbb{Z}_3)$ to $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(M(3)))$ has the effect of shifting our vanishing line to the left by one unit. Further moving to $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(M(3)_1))$ moves the vanishing line up by one unit. This Ext grid is given in Figure 8. below. As before everything above and to the left of the line must be zero, as well as everything along the $s = 1$ line for $t - s < 28$

We next construct the vanishing region for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y))$ by taking the intersection of the vanishing regions for $\mathrm{Ext}_A^{s-1,t}(H^*(Z), H^*(\Sigma^4 M(3)))$ and $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(M(3)_1))$ as in (3.17). The first of these pieces looks like Figure 9.

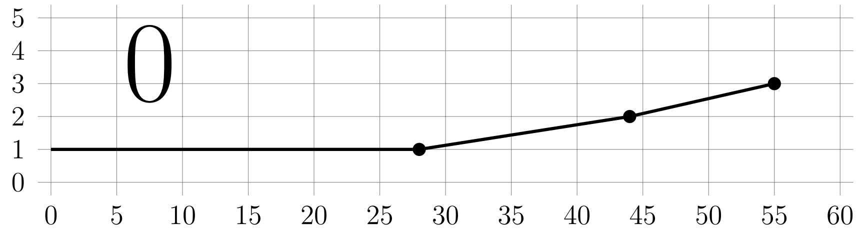

FIGURE 8. Vanishing edge for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(M(3)_1))$ in $(t-s, s)$ coordinates

shifted up by 1, and to the left by 6, and the second is depicted in Figure 8.. This intersection results in a vanishing region depicted by Figure 9. below, where everything above and to the left of the line, as well as the line $s = 1$ for $t - s < 23$ must be zero.

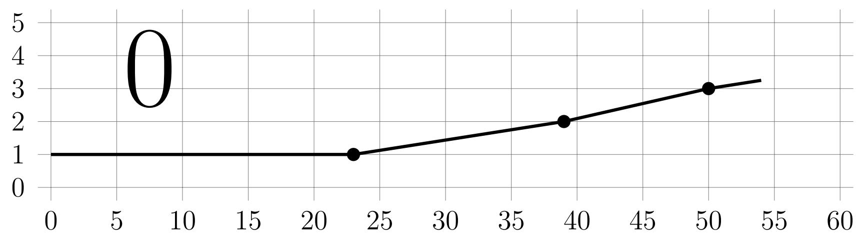

FIGURE 9. Vanishing edge for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y))$ in $(t-s, s)$ coordinates

We then move two stages up the Adams resolution for Y to first produce $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_1))$, and then $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_2))$. Each of these steps moves our vanishing line up by one, as seen below in Figure 10.. We still have everything above the line, and everything along $s = 3$ for $t - s < 23$ must be zero.

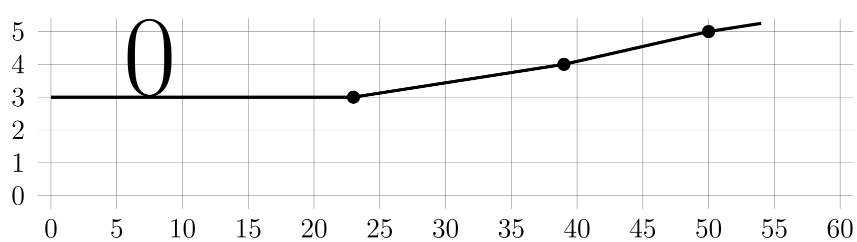

FIGURE 10. Vanishing edge for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_2))$ in $(t-s, s)$ coordinates

Finally, we form the vanishing region for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Z))$ by taking the intersection of the vanishing region for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(Y_2))$ (from Figure 10.) and the vanishing region for $\mathrm{Ext}_A^{s,t}(H^*(Z), H^*(\Sigma^{10}Y))$ (from shifting Figure 9. left by 10). The resulting diagram is identical to Figure 10. above.

Now, our u_3 map was an element in $\mathrm{Ext}_A^{1,17}(H^*(Z), H^*(Z))$, which would be in position $(16, 1)$ in the figure. The Adams Spectral Sequence differential d_r moves up r and left 1 spot on the grid. Since we only have to worry about d_r for $r \geq 2$, all of the differentials land in the $s - t = 15$ column, for $s \geq 3$. However, we have seen that we must have zeroes along the line $s = 3$ for $t - s < 23$, so none of these differentials can be nonzero. In addition, since we have an element of Ext^1, it cannot be in the image of any differentials. Thus, α survives the Adams Spectral Sequence to give us a map $f_Z : \Sigma^{16}Z \to Z$.

CHAPTER IV

CONSTRUCTION OF A FINITE SPECTRUM WITH A V_2 MAP

We have now constructed our u_3 map on the spectrum Z. We want to use this map, as well as the u_2 and u_1 maps we constructed in order to build a finite spectrum with a v_2 map.

First, consider our u_1 map $f_{M(3)} : \Sigma^4 M(3) \to M(3)$. To construct our second spectrum Y (and to kill the x_1 homology), we lifted this map to $\Sigma^4 M(3) \to M(3)_1$ and took the cofiber. This spectrum, as we have detailed in Section 3.2, is no longer finite.

However, we can construct a different spectrum, denoted $V(1)$ as in [BP04], which is finite, by taking the cofiber of $f_{M(3)}$. This is the cofiber sequence

$$\Sigma^4 M(3) \xrightarrow{f_{M(3)}} M(3) \to V(1)$$

We would like to use Lemma 1.11 to show that our self map on Y, f_Y, corresponds to a self map on the finite spectrum $V(1)$. To do so we need to show that the map $Y \to V(1)$ has a fiber with finite Adams resolution. In fact, we show that the fiber is just an Eilenberg-MacLane spectrum.

Consider the diagram below:

$$\begin{tikzcd}
* \arrow[r] \arrow[d] & \Sigma^{-1}K(\mathbb{Z}_3) \arrow[d] & \\
\Sigma^4 M(3) \arrow[r, "\bar{f}_{M(3)}"] \arrow[d, "="'] & M(3)_1 \arrow[r] \arrow[d] & Y \arrow[d] \\
\Sigma^4 M(3) \arrow[r, "f_{M(3)}"] & M(3) \arrow[r] & V(1)
\end{tikzcd}$$

The elements in the top row are the fibers of the vertical maps below them. By the 3×3 Lemma [Mac95][5.1], the top right corner of the map can be filled in with another copy of $\Sigma^{-1}K(\mathbb{Z}_3)$, and it is the fiber of the map $Y \to V(1)$.

Since the map $Y \to V(1)$ has a fiber with a finite Adams resolution, we know, by Lemma 1.11, that our self map $f_y : \Sigma^{10}Y \to Y$ induces a self map $h_Y : \Sigma^{10}V(1) \to V(1)$.

We use h_Y to create a finite spectrum associated to the locally finite spectrum Z. Let Z_f be the cofiber of h_Y, so Z_f has eight cells. We need to use Lemma 1.11 again on our u_3 self map on Z to create a self map on Z_f, and show that it is also a v_2 self map. In order to do this, we need to show that the fiber of the map $Z \to Z_f$ has a finite Adams resolution.

Let K_0 and K_1 be part of the Adams resolution for Y as defined in (3.18). Let G be the fiber of the composition $Y_2 \to Y_1 \to Y$. Consider the following diagram where all rows and columns are fiber sequences:

$$\begin{tikzcd}
\Sigma^{-1}K_1 \arrow[r] \arrow[d, "="'] & G \arrow[r] \arrow[d] & \Sigma^{-1}K_0 \arrow[d] \\
\Sigma^{-1}K_1 \arrow[r] \arrow[d] & Y_2 \arrow[r] \arrow[d] & Y_1 \arrow[d] \\
* \arrow[r] & Y \arrow[r, "="] & Y
\end{tikzcd}$$

Since G is in a fiber sequence with $\Sigma^{-1}K_1$ and $\Sigma^{-1}K_0$, and since K_1 and K_0 are spectra with finite Adams resolutions, then G also has a finite Adams resolution. Now, we let G' be the fiber of the composition $Y_2 \to Y \to V(1)$. Then we can construct the following diagram to show G' has a finite Adams resolution by the same argument we made for G.

$$\begin{tikzcd}
G \arrow[r] \arrow[d, "="'] & G' \arrow[r] \arrow[d] & \Sigma^{-1}K(\mathbb{Z}_3) \arrow[d] \\
G \arrow[r] \arrow[d] & Y_2 \arrow[r] \arrow[d] & Y \arrow[d] \\
* \arrow[r] & V(1) \arrow[r, "="] & V(1)
\end{tikzcd}$$

Let H be the fiber of $Z \to Z_f$. We construct another diagram below to show that H must also have a finite Adams resolution.

$$
\begin{tikzcd}
\Sigma^9 K(\mathbb{Z}_3) \arrow[r] \arrow[d] & G' \arrow[r] \arrow[d] & H \arrow[d] \\
\Sigma^{10} Y \arrow[r, "\bar{f}_Y"] \arrow[d] & Y_2 \arrow[r] \arrow[d] & Z \arrow[d] \\
\Sigma^{10} V(1) \arrow[r, "h_Y"] & V(1) \arrow[r] & Z_f
\end{tikzcd}
$$

We now claim, from Lemma 1.11 again, that our self map $f_Z : \Sigma^{16} Z \to Z$ induces a self map $h_Z : \Sigma^{16} Z_f \to Z_f$. It remains to conclude that h_Z is actually a v_2 map. But this follows from Lemma 1.12. Thus, we have constructed a v_2 self map on the eight-cell finite spectrum Z_f.

APPENDIX

SAGE CODE

Following is the code used to carry out the computations in Chapter 3. It is broken into five parts, which correspond to the construction of the cohomology of the spectra Y and Z, and the modules involved in the minimal resolutions of $H^*(Y)$ and $H^*(Z)$.

Below is sample output (lightly formatted for readability) from the first segment of code, through line 165. This output summarizes the generators of the kernel of the map $\partial_0 : P_0 \to H^*(Y)$ as given in Table 2. in the text.

```
[P^2, 0, 0, 0]  is a generator of the kernel in dimension  11

[P^3 P^1 + P^4, 2 P^2, 0, 0]  is a generator of the kernel in
dimension  19

[P^3 P^1 beta + beta P^3 P^1, P^2 beta, 2 P^1, 0]  is a generator of
the kernel in dimension  20

[P^6, 2 P^3 P^1 + 2 P^4, 0, 0]  is a generator of the kernel in
dimension  27

[P^5 beta P^1, 2 P^3 P^1 beta + 2 P^4 beta, 2 P^3, 0]  is a
generator of the kernel in dimension  28

[beta P^6 P^1 beta, 0, 2 P^3 P^1 beta + 2 P^4 beta, 0]  is a
generator of the kernel in dimension  33

[P^7 P^1, P^5 P^1 + P^6, 0, 0]  is a generator of the kernel in
dimension  35

[P^9, P^6 P^1 + P^7, 0, 2 P^1]  is a generator of the kernel in
dimension  39
```

```
[P^10 P^3 P^1, 2 P^9 P^3 + 2 P^10 P^2 + 2 P^12, 0, P^5 P^1 + P^6]
is a generator of the kernel in dimension  59
[0, 0, P^9 P^3 + P^10 P^2 + P^12, 2 P^6 P^1 beta]  is a generator of
the kernel in dimension  64
[0, 2 P^14 P^3 P^1 + P^14 P^4 + P^18, 0, 2 P^9 P^3 + P^11 P^1
+ 2 P^12] is a generator of the kernel in dimension  83
[P^16 P^4 beta P^1, P^14 P^4 beta P^1 + P^15 P^4 beta +
2 P^15 beta P^4 + 2 P^16 P^3 beta + 2 P^16 beta P^3 + P^17 beta P^2
+ P^19 beta + 2 beta P^14 P^4 P^1 + beta P^15 P^4 + beta P^16 P^3,
2 P^13 P^4 P^1 + 2 P^15 P^3 + P^18, 2 P^9 P^3 P^1 beta +
2 P^10 beta P^3 + P^11 beta P^2 + 2 P^12 P^1 beta + 2 P^13 beta +
beta P^11 P^2]  is a generator of the kernel in dimension  88
```

```
A3 = SteenrodAlgebra(p=3,basis='adem'); A3
beta = A3.Q(0); beta
P1 = A3.monomial((0,1,0)); P1
P3 = A3.monomial((0,3,0)); P3
P9 = A3.monomial((0,9,0)); P9

#Compute the kernel of the map $\oplus A_3 \to Y$
#Y has generators P1, P3, P3P1B, P9, P27, ... as an A_3 module

#how high to carry the computations
height = 100

#Keep track of the elements in the kernel of the first step
ker1list = [];

#Keep track of the generators of the kernel in the first step in terms of vector space
ker1gens = [];

#Keep track of the generators of Y1 in terms of steenrod elements, and their dimensions
Y1gens = [];
Y1dims = [];

#Elements in the sum of A_3's (This is really P0 in our projective resolution)
P0elements = [];

#Dictionary keeping track of the generators of Y and their dimension:
Ygens = [P1, P3, P3*P1*beta, P9]
Ygendims = [3,11,16,35]
numgens = 4;

#Keep track of Y
Ybasis = []

for i in xrange(height):
    ker1list.append([]);
    ker1gens.append([]);
    P0elements.append([]);

    #Build a basis for Y in this dimension so we can compute the kernel later
    Ybasis.append([]);

    Yi = A3.basis(i+1);
    for elt in Yi:
        #each element is in terms of the P^i's, this returns them in terms of admissible sequences
        #which we need later
        if (elt != beta) and (elt != P1*beta) and (elt != beta*P1*beta):
            Ybasis[i].append(elt.monomial_coefficients().keys()[0]);

    #A dictionary to keep track of the image of each of the v-space generators of P0
    maps = {};

    #Look at the copies of A3 in P0 corresponding to each of the generators
    for j in xrange(numgens):
        elementlist = A3.basis(i-Ygendims[j]);
        for elt in elementlist:
            target = elt*Ygens[j];
            vselt = [0]*numgens;
            vselt[j] = elt
            P0elements[i].append(tuple(vselt));
            maps[tuple(vselt)] = target

    #Dimension of Y as a vector space in this degree
```

```
    Ydimension = len(Ybasis[i]);

    #Dimension of P0 as a vector space in this degree
    P0dimension = len(P0elements[i]);

    m = matrix(GF(3),P0dimension,Ydimension);
    for j in xrange(P0dimension):
        #For each element in P0, decompose its image in terms of admissible sequences and coefficients
        image = maps[P0elements[i][j]].monomial_coefficients();
        for k in range(Ydimension):
            #Extract the coefficients to form our linear transformation matrix
            if image.has_key(Ybasis[i][k]):
                m[j,k] = image[Ybasis[i][k]];
    #print m

    #Compute the kernel of our linear transformation:
    ker = kernel(m);
    for elt in ker.basis():
        ker1list[i].append(list(elt));
        element = [0]*numgens;
        for j in xrange(P0dimension):
            for k in xrange(numgens):
                element[k] = element[k]+elt[j]*P0elements[i][j][k];

print '--------------------------------------'

#Now that we have the kernel, we want to rewrite it to emphasize the structure as an A3 module
#We look down from each dimension to see if we can replace an element with beta*something, P1*something, et

multipliers = [beta,P1,P3,P9];
multdims = [1,4,12,36];
nummults = 4;

for i in xrange(3,height):
    #if the kernel in this dimension is nonempty:
    if (len(ker1list[i])!=0):
        #Collect all elements that are images of things in lower dimensions:
        imagevectors = [];
        #Iterate through the multipliers
        for j in xrange(nummults):
            #Look the appropriate number of dimensions below to see if the kernel is nonempty:
            if (i-multdims[j] > 0) and (len(ker1list[i-multdims[j]])!=0):
                for elt in ker1list[i-multdims[j]]:
                    element = [0]*numgens;
                    for k in xrange(len(elt)):
                        for l in xrange(numgens):
                            element[l] = element[l]+elt[k]*P0elements[i-multdims[j]][k][l];
                    for k in xrange(numgens):
                        element[k] = multipliers[j]*element[k];

                    #We now have the image of an element from a lower dimension, in terms of elements of th
                    #sum of copies of A3, we need to rewrite it in terms of our Z/3 vector space so we can
                    #check spans below
                    imvect = [];
                    coefficients = [];
                    #Get the coefficients (in terms of admissible sequences) of each component of the image
                    #vectors
                    for k in xrange(numgens):
                        coefficients.append(element[k].monomial_coefficients());
                    #Use the basis elements in P0 to construct our vector, i.e. find the coefficient of eac
                    #element in P0 in our image vector
                    for basiselt in P0elements[i]:
                        #Each one should only have one nonzero entry, so adding them together is the same a
                        #picking the nonzero one...
                        basis = 0;
```

```
                        for k in xrange(numgens):
                            basis = basis+basiselt[k];
                        #convert this element to an admissible sequence:
                        sequence = basis.monomial_coefficients().keys()[0];
                        coeff = 0;
                        for k in xrange(numgens):
                            if coefficients[k].has_key(sequence):
                                coeff = coeff + coefficients[k][sequence];
                        imvect.append(coeff);
                    imagevectors.append(imvect)

        #Keep track of which elements we add in
        replacements = [];


        #Now, compute the generators of the kernel as an A_3 module:
        imagespan = (GF(3)^(len(P0elements[i]))).span(imagevectors)

        for j in xrange(len(ker1list[i])):
            if ker1list[i][j] not in replacements:
                if len((imagespan.intersection(span([ker1list[i][j]],GF(3)))).basis()) == 0:
                    ker1gens[i].append(ker1list[i][j]);
                    imagespan = (GF(3)^(len(P0elements[i]))).span(imagevectors+ker1gens[i]);

print '---------------------------'

#Print the generators:
for i in xrange(3,height):
    if len(ker1gens[i])!=0:
        for elt in ker1gens[i]:
            element = [0]*numgens;
            for j in xrange(len(elt)):
                for k in xrange(numgens):
                    element[k] = element[k]+elt[j]*P0elements[i][j][k]
            print element,' is a generator of the kernel in dimension ',i
            Y1gens.append(element);
            Y1dims.append(i-1);

#Part2
#Now that we've computed Y1 = ker:P0 -> Y, we form P1 and repreat the process to find Y2 = ker:P1 -> Y1

#Height for this part of the computation
height2 = 99;

#Basis for P1
P1elements = [];

#Basis of this kernel
ker2basis = [];

#Generators of this kernel
ker2gens = [];

#How many generators did we find in the previous step?
numgens2 = len(Y1gens);

#Start computing the maps based on the generators above:
for i in xrange(height2):
    P1elements.append([]);
    ker2basis.append([]);
    ker2gens.append([]);

    #Maps in terms of steenrod elements
    maps = {};

```

```
    #Maps in terms of basis elements in Y1
    kermaps = {};

    #Create elements in P1, and find where in Y1 they map (in terms of steenrod elements)
    for j in xrange(numgens2):
        elements = A3.basis(i-Y1dims[j]);
        generator = Y1gens[j];
        for elt in elements:
            source = [0]*numgens2;
            source[j] = elt;
            target = [0]*numgens;
            for k in xrange(numgens):
                target[k] = elt*generator[k];
            P1elements[i].append(tuple(source));
            maps[tuple(source)] = tuple(target);

    #Now rewrite each one in terms of Y1 elements (vector space form)
    Y1vecspace = (GF(3)^len(P0elements[i+1])).span_of_basis(ker1list[i+1]);
    for elt in P1elements[i]:
        image = maps[elt];
        coefficients = [];
        #Extract the coefficients in terms of admissible sequences
        for j in xrange(numgens):
            coefficients.append({});
            if image[j] != 0:
                coefficients[j] = image[j].monomial_coefficients();
        targetvector = [];
        #Write the target in terms of our vector space basis for P0
        for target in P0elements[i+1]:
            coefficient = 0;
            #There should be only one nonzero place
            targetmonomial = 0;
            #keep track of where it is
            spot = -1
            for j in xrange(numgens):
                if target[j] != 0:
                    targetmonomial = target[j].monomial_coefficients().keys()[0];
                    spot = j
            if coefficients[spot].has_key(targetmonomial):
                coefficient = coefficients[spot][targetmonomial];
            targetvector.append(coefficient)
        #Now, get this element in terms of the vector space basis:
        kervec = Y1vecspace.coordinates(targetvector);
        kermaps[elt] = kervec;

    #Create a matrix representing our linear transformation
    Mat = Matrix(GF(3),len(maps.keys()),len(ker1list[i+1]));
    for j in xrange(len(maps.keys())):
        Mat[j] = kermaps[P1elements[i][j]];

    Ker2 = kernel(Mat);
    for elt in Ker2.basis():
        element = [0]*numgens2;
        ker2basis[i].append(list(elt));
        for j in xrange(len(elt)):
            for k in xrange(numgens2):
                element[k] = element[k]+elt[j]*P1elements[i][j][k];

print '---------------------------'

#Now compute the images of things below
multipliers = [beta,P1,P3,P9];
multdims = [1,4,12,36];
nummults = 4;

```

```
for i in xrange(3,height2):
    print 'realigning dimension ',i
    #if the kernel in this dimension is nonempty:
    if (len(ker2basis[i])!=0):
        #Collect all elements that are images of things in lower dimensions:
        imagevectors = [];
        #Iterate through the multipliers
        for j in xrange(nummults):
            #Look the appropriate number of dimensions below to see if the kernel is nonempty:
            if (i-multdims[j] > 0) and (len(ker2basis[i-multdims[j]])!=0):
                for elt in ker2basis[i-multdims[j]]:
                    element = [0]*numgens2;
                    for k in xrange(len(elt)):
                        for l in xrange(numgens2):
                            element[l] = element[l]+elt[k]*P1elements[i-multdims[j]][k][l];
                    for k in xrange(numgens2):
                        element[k] = multipliers[j]*element[k];

                    #We now have the image of an element from a lower dimension, in terms of elements of th
                    #sum of copies of A3, we need to rewrite it in terms of our Z/3 vector space so we can
                    #check spans below
                    imvect = [];
                    coefficients = [];
                    #Get the coefficients (in terms of admissible sequences) of each component of the image
                    #vectors
                    for k in xrange(numgens2):
                        if element[k]!=0:
                            coefficients.append(element[k].monomial_coefficients());
                        else:
                            coefficients.append({});
                    #Use the basis elements in P0 to construct our vector, i.e. find the coefficient of eac
                    #element in P0 in our image vector
                    for basiselt in P1elements[i]:
                        #Each one should only have one nonzero entry, so adding them together is the same a
                        #picking the nonzero one...
                        basis = 0;
                        for k in xrange(numgens2):
                            basis = basis+basiselt[k];
                        #convert this element to an admissible sequence:
                        sequence = basis.monomial_coefficients().keys()[0];
                        coeff = 0;
                        for k in xrange(numgens2):
                            if coefficients[k].has_key(sequence):
                                coeff = coeff + coefficients[k][sequence];
                        imvect.append(coeff);
                    imagevectors.append(imvect)


        #Keep track of which elements we add in
        replacements = [];

        #Now, compute the generators of the kernel as an A_3 module:

        imagespan = (GF(3)^(len(P1elements[i]))).span(imagevectors)

        for j in reversed(range(len(ker2basis[i]))):
            if ker2basis[i][j] not in replacements:

                if len((imagespan.intersection(span([ker2basis[i][j]],GF(3)))).basis()) == 0:
                    ker2gens[i].append(ker2basis[i][j]);
                    imagespan = (GF(3)^(len(P1elements[i]))).span(imagevectors+ker2gens[i]);

print '---------------------------'

#Print the generators:
```

```
for i in xrange(3,height2):
    if len(ker2gens[i])!=0:
        for elt in ker2gens[i]:
            element = [0]*numgens2;
            for j in xrange(len(elt)):
                for k in xrange(numgens2):
                    element[k] = element[k]+elt[j]*P1elements[i][j][k]
            print element,' is a generator of the kernel in dimension ',i

#Part 3
#Try to compute the kernel of the map Y2 (= ker: P1 -> Y1) -> S10 (Y)
#Start by printing things in the appropriate dimensions:
height3 = 95;

#Keep track of the kernel of f_bar_star
kerfbasis = [];

#Keep track of the generators of the kernel (i.e. the generators of H*(Z))
kerfgens = [];

#Keep track of the generators of Z for the next step:
Zgens = [];
Zgendims = [];

for i in xrange(10):
    kerfbasis.append([]);
    kerfgens.append([]);

for i in xrange(10,height3):
    print '--------------'
    print 'dimension = ',i

    kerfbasis.append([]);
    kerfgens.append([]);

    #keep track of where basis elements in Y2 are mapped:
    fmaps = {};

    #print 'printing basis of Y2'
    for elt in ker2basis[i+1]:
        element = [0]*numgens2
        for j in xrange(len(elt)):
            for k in xrange(numgens2):
                element[k] = element[k]+elt[j]*P1elements[i+1][j][k]
        fmaps[tuple(elt)] = element[0];

    P2dimension = len(ker2basis[i+1]);
    Ydimension = len(Ybasis[i-10]);

    m = matrix(GF(3),P2dimension,Ydimension);
    for j in xrange(P2dimension):
        #For each element in P0, decompose its image in terms of admissible sequences and coefficients
        image = fmaps[tuple(ker2basis[i+1][j])].monomial_coefficients();
        for k in range(Ydimension):
            #Extract the coefficients to form our linear transformation matrix
            if image.has_key(Ybasis[i-10][k]):
                m[j,k] = image[Ybasis[i-10][k]];

    kerf = kernel(m);
    for elt in kerf.basis():
        lenkervect = len(ker2basis[i+1][0]);
        element = [0]*lenkervect
        #First write the element in terms of our vector space kernel above
        for j in xrange(P2dimension):
            for k in xrange(lenkervect):
```

```
                element[k] = element[k]+elt[j]*ker2basis[i+1][j][k];
        kerfbasis[i].append(element);
        #Then write it in terms of steenrod elements
        element2 = [0]*numgens2;
        for j in xrange(lenkervect):
            for k in xrange(numgens2):
                element2[k] = element2[k]+element[j]*P1elements[i+1][j][k]

print '-----------------------'
print 'done computing kernel'
print '-----------------------'

#Now determine the generators of the kernel (these are the generators of the cohomology of Z)
#Now compute the images of things below
multipliers = [beta,P1,P3,P9];
multdims = [1,4,12,36];
nummults = 4;

for i in xrange(3,height3):
    print 'realigning dimension ',i
    #if the kernel in this dimension is nonempty:
    if (len(kerfbasis[i])!=0):
        #Collect all elements that are images of things in lower dimensions:
        imagevectors = [];
        #Iterate through the multipliers
        for j in xrange(nummults):
            #Look the appropriate number of dimensions below to see if the kernel is nonempty:
            if (i-multdims[j] > 0) and (len(kerfbasis[i-multdims[j]])!=0):
                for elt in kerfbasis[i-multdims[j]]:
                    element = [0]*numgens2;
                    for k in xrange(len(elt)):
                        for l in xrange(numgens2):
                            element[l] = element[l]+elt[k]*P1elements[i-multdims[j]+1][k][l];
                    for k in xrange(numgens2):
                        element[k] = multipliers[j]*element[k];

                    #We now have the image of an element from a lower dimension, in terms of elements of th
                    #sum of copies of A3, we need to rewrite it in terms of our Z/3 vector space so we can
                    #check spans below
                    imvect = [];
                    coefficients = [];
                    #Get the coefficients (in terms of admissible sequences) of each component of the image
                    #vectors
                    for k in xrange(numgens2):
                        if element[k]!=0:
                            coefficients.append(element[k].monomial_coefficients());
                        else:
                            coefficients.append({});
                    #Use the basis elements in P0 to construct our vector, i.e. find the coefficient of eac
                    #element in P0 in our image vector
                    for basiselt in P1elements[i+1]:
                        #Each one should only have one nonzero entry, so adding them together is the same a
                        #picking the nonzero one...
                        basis = 0;
                        for k in xrange(numgens2):
                            basis = basis+basiselt[k];
                        #convert this element to an admissible sequence:
                        sequence = basis.monomial_coefficients().keys()[0];
                        coeff = 0;
                        for k in xrange(numgens2):
                            if coefficients[k].has_key(sequence):
                                coeff = coeff + coefficients[k][sequence];
                        imvect.append(coeff);
                    imagevectors.append(imvect)

```

```
        #Keep track of which elements we add in
        replacements = [];

        #Now, compute the generators of the kernel as an A_3 module:

        imagespan = (GF(3)^(len(P1elements[i+1]))).span(imagevectors)

        for j in reversed(range(len(kerfbasis[i]))):
            if kerfbasis[i][j] not in replacements:

                if len((imagespan.intersection(span([kerfbasis[i][j]],GF(3)))).basis()) == 0:
                    kerfgens[i].append(kerfbasis[i][j]);
                    imagespan = (GF(3)^(len(P1elements[i+1]))).span(imagevectors+kerfgens[i]);


print '----------------------'
print 'printing generators as A3 module:'
print '----------------------'

#Print the generators:
for i in xrange(3,height3):
    if len(kerfgens[i])!=0:
        for elt in kerfgens[i]:
            element = [0]*numgens2;
            for j in xrange(len(elt)):
                for k in xrange(numgens2):
                    element[k] = element[k]+elt[j]*P1elements[i+1][j][k]
            print element,' is a generator of the kernel in dimension ',i
            Zgens.append(element);
            Zgendims.append(i);


#Part 4
#Start computing a resolution of Z?
#H*Z <-- Q0 <-- Q1 <-- Q2 ....
#This step should build Q0, then compute the kernel of the map Q0 --> H*Z
#The generators of the kernel will then tell us what Q1 should be

#The height to carry out this computation
heightz1 = 95

#Number of generators of H*Z
numZgens = len(Zgendims);

#Vector space generators of Q0
Q0elements = [];

#Elements in the kernel of d0: Q0 -> H*Z
kerd0basis = [];

#Elements that generate the kernel of d0: Q0 -> H*Z
kerd0gens = [];

#Store the generators of the kernel for the next step so we can map Q1 -> ker d0
Z1gens = [];
Z1gendims = [];

for i in xrange(heightz1):
    print 'dimension = ',i
    #Holds the vs generators of Q0 in this dimension
    Q0elements.append([]);
    #Holds the vs generators of ker d0 in this dimension
    kerd0basis.append([]);
```

```
    kerd0gens.append([]);
    #Maps in terms of steenrod elements
    maps = {}
    #Maps in terms of vs elements in H*Z
    kermaps = {}
    for j in xrange(numZgens):
        if (i - Zgendims[j]) > -1:
            basis = A3.basis(i-Zgendims[j]);
            for elt in basis:
                Q0elt = [0]*numZgens
                Q0elt[j] = elt;
                targetelt = [0]*numgens2;
                for k in xrange(numgens2):
                    targetelt[k] = targetelt[k]+elt*Zgens[j][k]
                Q0elements[i].append(tuple(Q0elt));
                maps[tuple(Q0elt)] = tuple(targetelt);

    #Now, we rewrite each of these targets in terms of the vs basis for H*Z
    #recall that Z is a subset of Y2 is a subset of P1
    #i+1 from shift between P1 and Y2
    #The subspace representing Z is generated by the kernel from the previous
    #step of the computation (Y2 -> Y)
    Zvecspace = (GF(3)^len(P1elements[i+1])).span_of_basis(kerfbasis[i]);
    #Cycle through the vs generators of Q0 in this dimension
    for elt in Q0elements[i]:
        #The steenrod tuple this element maps to
        image = maps[elt];
        #Stores the monomial coefficients of each component
        coefficients = [];
        #Extract the coefficients of each admissible sequence in the image
        for j in xrange(numgens2):
            coefficients.append({});
            if image[j] !=0:
                #This stores the monomials that make up the jth component
                #of the image
                coefficients[j] = image[j].monomial_coefficients();
        #This will store info about the target in terms of the vs gens
        targetvector = [];
        #Cycle through the vs generators of P1 to match up the monomials
        for target in P1elements[i+1]:
            coefficient = 0;
            targetmonomial = 0;
            #spot where we find the nonzero piece
            spot = -1;
            #Each P1 generator should have one nonzero component, so find it
            for j in xrange(numgens2):
                if target[j] != 0:
                    #Extract that admissible sequence
                    targetmonomial = target[j].monomial_coefficients().keys()[0]
                    spot = j
            #Look at the jth component monomials to see if it has the one we
            #are looking for.
            #NOTE TO SELF: This only works as long as the target space has no
            #module generators that are in the same dimension
            #i.e. if d0(x) = (a, b, c, P1*B, P1*B, d, e,...) it'll be bad
            #FIXED (I THINK)
            if coefficients[spot].has_key(targetmonomial):
                coefficient = coefficients[spot][targetmonomial]
            targetvector.append(coefficient)

        #Write this targetvector as a linear combination of the basis of H*Z
        kervec = Zvecspace.coordinates(targetvector);
        kermaps[elt] = kervec;

    #Create a matrix representing our linear transformation Q0 -> H*Z
```

```
    Mat = Matrix(GF(3),len(maps.keys()),len(kerfbasis[i]));
    for j in xrange(len(maps.keys())):
        Mat[j] = kermaps[Q0elements[i][j]];

    Q0ker = kernel(Mat);
    for elt in Q0ker.basis():
        element = [0]*numZgens;
        kerd0basis[i].append(list(elt));
        for j in xrange(len(elt)):
            for k in xrange(numZgens):
                element[k] = element[k]+elt[j]*Q0elements[i][j][k];
    print '---------------'

#Now rewrite with A3 module structure:
print '-----------------------------------------------'
for i in xrange(heightz1):
    print 'realigning ',i
    if (len(kerd0basis[i])!=0):
        #keep track of the image of things in lower dimensions
        imagevectors = [];
        #iterate through the elements B, P1, P3, P9
        for j in xrange(nummults):
            #make sure that we're not looking in a negative dimension or
            #a dimension with nothing in it
            if (i - multdims[j] > 0) and (len(kerd0basis[i-multdims[j]])!=0):
                #iterate through the elements in the lower dimension
                for elt in kerd0basis[i-multdims[j]]:
                    element = [0]*numZgens
                    for k in xrange(len(elt)):
                        for l in xrange(numZgens):
                            element[l] = element[l]+elt[k]*Q0elements[i-multdims[j]][k][l]
                    #Now we have the element in a lower dimension, multiply
                    #each component by the current multiplier
                    for k in xrange(numZgens):
                        element[k] = multipliers[j]*element[k];

                    #Now rewrite our element in terms of our Z/3 vs basis so
                    #we can possibly replace elements below
                    imvect = [];
                    #holds a dictionary of coefficients for each component
                    coefficients = [];
                    #Get the coefficients (in terms of admissible sequences)
                    #of each component of the image vector
                    for k in xrange(numZgens):
                        if element[k]!=0:
                               coefficients.append(element[k].monomial_coefficients())
                        else:
                            coefficients.append({});
                    for basiselt in Q0elements[i]:
                        basis = 0;
                        for k in xrange(numZgens):
                            basis = basis+basiselt[k];
                        #This is the admissible sequence representing the
                        #basis element
                        sequence = basis.monomial_coefficients().keys()[0];
                        coeff = 0;
                        #Again, this only works since we don't have two gens
                        #in the same dimension for Z
                        for k in xrange(numZgens):
                            if coefficients[k].has_key(sequence):
                                coeff = coeff+coefficients[k][sequence];
                        imvect.append(coeff);
                    imagevectors.append(imvect);
        #Keep track of which elements we add in:
        replacements = [];
```

```
        imagespan = (GF(3)^(len(Q0elements[i]))).span(imagevectors);
        for j in reversed(range(len(kerd0basis[i]))):
            if len((imagespan.intersection(span([kerd0basis[i][j]],GF(3)))).basis()) == 0:
                kerd0gens[i].append(kerd0basis[i][j]);
                imagespan = (GF(3)^(len(Q0elements[i]))).span(imagevectors+kerd0gens[i]);


print '----------------------'
print 'printing generators as A3 module:'
print '----------------------'

#Print the generators:
numgens = 1;
for i in xrange(heightz1):
    if len(kerd0gens[i])!=0:
        for elt in kerd0gens[i]:
            element = [0]*numZgens;
            for j in xrange(len(elt)):
                for k in xrange(numZgens):
                    element[k] = element[k]+elt[j]*Q0elements[i][j][k]
            print element,' is a generator of the kernel in dimension ',i
            if element[6]!=0:
                print 'this is generator number ',numgens,' and has ',element[6],' in the seventh spot'
            Z1gens.append(element);
            Z1gendims.append(i);
            numgens = numgens + 1;

#Part 5:
#We've computed the kernel of d0: Q0 -> H*Z and found the generators
#We now get a new copy of A3 in Q1 for each of these generators and repeat the process, computing
#the kernel of d1: Q1 -> Q0 in the next step of our resolution

#The height to carry out this computation
heightz1 = 80

#Number of generators of H*Z
numZ1gens = len(Z1gendims);

#Vector space generators of Q0
Q1elements = [];

#Elements in the kernel of d0: Q0 -> H*Z
kerd1basis = [];

#Elements that generate the kernel of d0: Q0 -> H*Z
kerd1gens = [];

#Store the generators of the kernel for the next step so we can map Q1 -> ker d0
Z2gens = [];
Z2gendims = [];

for i in xrange(heightz1):
    print 'dimension = ',i
    #Holds the vs generators of Q0 in this dimension
    Q1elements.append([]);
    #Holds the vs generators of ker d0 in this dimension
    kerd1basis.append([]);

    kerd1gens.append([]);
    #Maps in terms of steenrod elements
    maps = {}
    #Maps in terms of vs elements in H*Z
    kermaps = {}
    for j in xrange(numZ1gens):
```

```
        if (i - Z1gendims[j]) > -1:
            basis = A3.basis(i-Z1gendims[j]);
            for elt in basis:
                Q1elt = [0]*numZ1gens
                Q1elt[j] = elt;
                targetelt = [0]*numZgens;
                for k in xrange(numZgens):
                    targetelt[k] = targetelt[k]+elt*Z1gens[j][k]
                Q1elements[i].append(tuple(Q1elt));
                maps[tuple(Q1elt)] = tuple(targetelt);

    #Now, we rewrite each of these targets in terms of the vs basis for Q0

    Q0vecspace = (GF(3)^len(Q0elements[i])).span_of_basis(kerd0basis[i]);
    #Cycle through the vs generators of Q0 in this dimension
    for elt in Q1elements[i]:
        #The steenrod tuple this element maps to
        image = maps[elt];
        #Stores the monomial coefficients of each component
        coefficients = [];
        #Extract the coefficients of each admissible sequence in the image
        for j in xrange(numZgens):
            coefficients.append({});
            if image[j] !=0:
                #This stores the monomials that make up the jth component
                #of the image
                coefficients[j] = image[j].monomial_coefficients();
        #This will store info about the target in terms of the vs gens
        targetvector = [];
        #Cycle through the vs generators of Q0 to match up the monomials
        for target in Q0elements[i]:
            coefficient = 0;
            targetmonomial = 0;
            #which spot did we find the nonzero element?
            spot = -1;
            #Each Q0 generator should have one nonzero component, so find it
            for j in xrange(numZgens):
                if target[j] != 0:
                    #Extract that admissible sequence
                    targetmonomial = target[j].monomial_coefficients().keys()[0]
                    spot = j;
            #Look at the jth component monomials to see if it has the one we
            #are looking for.
            if coefficients[spot].has_key(targetmonomial):
                coefficient = coefficients[spot][targetmonomial]
            targetvector.append(coefficient)

        #Write this targetvector as a linear combination of the basis of H*Z
        kervec = Q0vecspace.coordinates(targetvector);
        kermaps[elt] = kervec;

    #Create a matrix representing our linear transformation Q0 -> H*Z
    Mat = Matrix(GF(3),len(maps.keys()),len(kerd0basis[i]));
    for j in xrange(len(maps.keys())):
        Mat[j] = kermaps[Q1elements[i][j]];

    Q1ker = kernel(Mat);
    for elt in Q1ker.basis():
        element = [0]*numZ1gens;
        kerd1basis[i].append(list(elt));
        for j in xrange(len(elt)):
            for k in xrange(numZ1gens):
                element[k] = element[k]+elt[j]*Q1elements[i][j][k];
        print element,' is in the kernel in dimension ',i
    print '---------------'
```

REFERENCES CITED

[Ada66] J. F. Adams, *On the groups $J(X)$—IV*, Topology **5** (1966), 21–71. MR 0198470

[BP04] Mark Behrens and Satya Pemmaraju, *On the existence of the self map v_2^9 on the Smith-Toda complex $V(1)$ at the prime 3*, Homotopy theory: relations with algebraic geometry, group cohomology, and algebraic K-theory, Contemp. Math., vol. 346, Amer. Math. Soc., Providence, RI, 2004, pp. 9–49. MR 2066495

[DHS88] Ethan S. Devinatz, Michael J. Hopkins, and Jeffrey H. Smith, *Nilpotence and stable homotopy theory. I*, Ann. of Math. (2) **128** (1988), no. 2, 207–241. MR 960945

[Hop87] Michael J. Hopkins, *Global methods in homotopy theory*, Homotopy theory (Durham, 1985), London Math. Soc. Lecture Note Ser., vol. 117, Cambridge Univ. Press, Cambridge, 1987, pp. 73–96. MR 932260

[HS98] Michael J. Hopkins and Jeffrey H. Smith, *Nilpotence and stable homotopy theory. II*, Ann. of Math. (2) **148** (1998), no. 1, 1–49. MR 1652975

[Mac95] S. MacLane, *Homology*, Classics in Mathematics, Springer Berlin Heidelberg, 1995.

[Mar74] H. R. Margolis, *Eilenberg-Mac Lane spectra*, Proc. Amer. Math. Soc. **43** (1974), 409–415. MR 0341488

[Mar83] ______, *Spectra and the Steenrod algebra*, North-Holland Mathematical Library, vol. 29, North-Holland Publishing Co., Amsterdam, 1983, Modules over the Steenrod algebra and the stable homotopy category. MR 738973

[Mil81] Haynes R. Miller, *On relations between Adams spectral sequences, with an application to the stable homotopy of a Moore space*, J. Pure Appl. Algebra **20** (1981), no. 3, 287–312. MR 604321 (82f:55029)

[MM65] John W. Milnor and John C. Moore, *On the structure of Hopf algebras*, Ann. of Math. (2) **81** (1965), 211–264. MR 0174052

[MW81] Haynes Miller and Clarence Wilkerson, *Vanishing lines for modules over the Steenrod algebra*, J. Pure Appl. Algebra **22** (1981), no. 3, 293–307. MR 629336

[PS94] John H. Palmieri and Hal Sadofsky, *Self-maps of spectra, a theorem of J. Smith, and Margolis' killing construction*, Math. Z. **215** (1994), no. 3, 477–490. MR 1262528

[Rav03] D.C. Ravenel, *Complex cobordism and stable homotopy groups of spheres*, AMS Chelsea Publishing Series, AMS Chelsea Pub., 2003.

[Rav16] ———, *Nilpotence and periodicity in stable homotopy theory. (am-128)*, Annals of Mathematics Studies, Princeton University Press, 2016.

[S+16] W. A. Stein et al., *Sage Mathematics Software (Version 7.5)*, The Sage Development Team, 2016, `http://www.sagemath.org`.

[Smi70] Larry Smith, *On realizing complex bordism modules. Applications to the stable homotopy of spheres*, Amer. J. Math. **92** (1970), 793–856. MR 0275429

[Ste62] N. E. Steenrod, *Cohomology operations*, Lectures by N. E. Steenrod written and revised by D. B. A. Epstein. Annals of Mathematics Studies, No. 50, Princeton University Press, Princeton, N.J., 1962. MR 0145525