User’s Guide to:

A *Mathematica* Package for Computing McClure’s Obstructions to H∞ Structure on BP

by Niles Johnson and Justin Noel

This document is a companion to the paper
"H_{\infty} Orientations on BP"
by Niles Johnson and Justin Noel
http://math.uchicago.edu/~justin/H−infty−BP.pdf
and to the *Mathematica* package
http://math.uchicago.edu/~justin/McClureDefs.m

Computer calculation guide

The variable ’pVal’ is the prime at which we work.
The variable ’order’ controls the order of our power series calculations.
If order = k, then we work modulo x^{k+1}.

The file ’McClureDefs.m’ holds the other definitions for these calculations.
We give examples of most of these functions below.

Parallel Processing Note: We make use of the multicore support in *Mathematica* 7.
If you use a single−core machine, you may see warnings about parallel processing not working;
this should not affect the calculations.

Warning: *your choice for the value of ’order’ will drastically affect how long you have to wait for these calculations.*
We recommend starting with pVal = 3 and order = 16, but
depending on your machine, you may want to choose order = 5 or 10.
At the prime 3, order = 26 is a challenging computation for modern desktop computers.

In[1]:=
```
pVal = 3;
order = 16;
Get["McClureDefs.m"];
```

Series are represented as lists of coefficients, and below we give example calculations.

- #### log, exp, and formal sum

The function ’arraylogBP’ holds the coefficients of the log series for BP. We store this list (and others) as a ’SparseArray’, and the function ’Normal’ displays it’s values.

```
arraylogBP
```

```
SparseArray[<3>, {17}]
```

```
Normal[arraylogBP]
```

```
{0, 1, 0, L[1], 0, 0, 0, 0, 0, L[2], 0, 0, 0, 0, 0, 0, 0}
```

We calculate the exponential series, ’arrayexpBP’, as the compositional inverse to ’arraylogBP’.

```
Normal[arrayexpBP]
```

```
{0, 1, 0, -L[1], 0, 3 L[1]^2, 0, -12 L[1]^3, 0,
 55 L[1]^4 - L[2], 0, 222 L[1]^5 - 9 L[1] (55 L[1]^4 - L[2]) + 3 L[1] L[2],
 0, 405 L[1]^6 - 36 L[1]^2 (55 L[1]^4 - L[2]) - 9 L[1]^2 L[2] -
  11 L[1] (222 L[1]^5 - 9 L[1] (55 L[1]^4 - L[2]) + 3 L[1] L[2]), 0, 417 L[1]^7 - 84 L[1]^3 (55 L[1]^4 - L[2]) +
  27 L[1]^3 L[2] - 55 L[1]^2 (222 L[1]^5 - 9 L[1] (55 L[1]^4 - L[2]) + 3 L[1] L[2]) -
  13 L[1] (405 L[1]^6 - 36 L[1]^2 (55 L[1]^4 - L[2]) - 9 L[1]^2 L[2] -
     11 L[1] (222 L[1]^5 - 9 L[1] (55 L[1]^4 - L[2]) + 3 L[1] L[2])), 0}
```

The function ’compser’ calculates the composite of two series.
compser[f(x),g(x)] = f(g(x)).

```
ExpandAll[compser[arrayexpBP, arraylogBP]]
```

```
{0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

The function ’arrayiseriesBP’ computes the i−series for BP as exp(i * log(x)).
Here we compute the p−series:

```
ExpandAll[arrayiseriesBP[pVal]]
```

```
{0, 3, 0, -24 L[1], 0, 648 L[1]^2, 0, -22 680 L[1]^3, 0,
 906 120 L[1]^4 - 19 680 L[2], 0, -39 161 880 L[1]^5 + 1 948 536 L[1] L[2], 0,
 1 782 778 248 L[1]^6 - 144 725 616 L[1]^2 L[2], 0, -84 205 559 448 L[1]^7 + 9 647 551 656 L[1]^3 L[2], 0}
```

We have implemented the Hazewinkel generators and the Araki generators
as a list of replacement rules, demonstrated below.
The variable ’nVal’ is set to ’order+1’, a useful shorthand since all of our coefficient lists are this long.

```
vnSubstitutionsHazewinkel[pVal, nVal]
```

$$\left\{L[2] \to \frac{1}{9}\left(v[1]^4 + 3\,v[2]\right),\ L[1] \to \frac{v[1]}{3}\right\}$$

```
vnSubstitutionsAraki[pVal, nVal]
```

$$\left\{L[1] \to -\frac{w[1]}{24},\ L[2] \to \frac{w[1]^4 - 24\,w[2]}{472\,320}\right\}$$

To allow changing generators, all the rest of our formulas reference the variable ’vnSubstitutionsJN’.
By default, this is set to the Hazewinkel generators; to change, edit the file ’McClureDefs.m’

vnSubstitutionsJNName

Hazewinkel

vnSubstitutionsJN

$$\left\{L[2] \to \frac{1}{9}\left(v[1]^4 + 3\,v[2]\right),\ L[1] \to \frac{v[1]}{3}\right\}$$

Here is the p−series, expressed with the generators v[i] specified above instead of the rational generators, L[i].

arraypseriesBPvnJN

$$\{0, 3, 0, -8\,v[1], 0, 72\,v[1]^2, 0, -840\,v[1]^3, 0, 9000\,v[1]^4 - 6560\,v[2], 0, -88\,992\,v[1]^5 + 216\,504\,v[1]\,v[2], 0, 658\,776\,v[1]^6 - 5\,360\,208\,v[1]^2\,v[2], 0, 1\,199\,088\,v[1]^7 + 119\,105\,576\,v[1]^3\,v[2], 0\}$$

For the reduced p−series, the last term is unknown

arraypseriesRedBPvnJN

$$\{3, 0, -8\,v[1], 0, 72\,v[1]^2, 0, -840\,v[1]^3, 0, 9000\,v[1]^4 - 6560\,v[2], 0, -88\,992\,v[1]^5 + 216\,504\,v[1]\,v[2], 0, 658\,776\,v[1]^6 - 5\,360\,208\,v[1]^2\,v[2], 0, 1\,199\,088\,v[1]^7 + 119\,105\,576\,v[1]^3\,v[2], 0, \text{Unk}\}$$

The formal sum over BP, exp(log(x) + log(y)), is given as a 2−dimensional array.
The function ’arrayPlusPolyBPi’ computes exp(i * log(x) + log(y));
here we display the formal sum by using arrayPlusPolyBPi[1], making the Hazewinkel substitutions, and printing this array as a matrix.
Note that, as expected, this is a symmetric matrix, with the coefficient of x and y each being 1.
This matrix is upper−(anti)triangular because we are working modulo (x,y)^{order+1}.
Note: this output is cutoff in the pdf version of this file

MatrixForm[ExpandAll[Normal[arrayPlusPolyBPi[1]] /. vnSubstitutionsJN]]

0	1	0	0	0
1	0	$-v[1]$	0	$v[1]^2$
0	$-v[1]$	0	$3\,v[1]^2$	0
0	0	$3\,v[1]^2$	0	$-13\,v[1]^3$
0	$v[1]^2$	0	$-13\,v[1]^3$	0
0	0	$-6\,v[1]^3$	0	$52\,v[1]^4 - 42\,v[2$
0	$-v[1]^3$	0	$27\,v[1]^4 - 28\,v[2]$	0
0	0	$6\,v[1]^4 - 12\,v[2]$	0	$-106\,v[1]^5 + 362\,v[1]$
0	$-3\,v[2]$	0	$-27\,v[1]^5 + 163\,v[1]\,v[2]$	0
0	0	$45\,v[1]\,v[2]$	0	$30\,v[1]^6 - 1770\,v[1]^2$
0	$v[1]^5 + 6\,v[1]\,v[2]$	0	$-31\,v[1]^6 - 568\,v[1]^2\,v[2]$	0
0	0	$-15\,v[1]^6 - 108\,v[1]^2\,v[2]$	0	$655\,v[1]^7 + 6333\,v[1]^3$
0	$-2\,v[1]^6 - 9\,v[1]^2\,v[2]$	0	$226\,v[1]^7 + 1517\,v[1]^3\,v[2]$	0
0	0	$42\,v[1]^7 + 210\,v[1]^3\,v[2]$	0	0
0	$3\,v[1]^7 + 12\,v[1]^3\,v[2]$	0	0	0
0	0	0	0	0
0	0	0	0	0

power operation, Euler class, a_i

Here we display the potential power operation for BP, using the Hazewinkel generators.
In the notation of our companion paper, the entry in row i and column j is the coefficient of x^i ξ ^j.
Note: this output is cutoff in the pdf version of this file

```
MatrixForm[arrayPBPvnJN]
```

0	0	0	0	0
0	0	2	0	-2 v
0	3	0	$-8 v[1]$	0
1	0	$-9 v[1]$	0	63 v[
0	$-3 v[1]$	0	$53 v[1]^2$	0
0	0	$21 v[1]^2$	0	-435 v
0	$3 v[1]^2$	0	$-179 v[1]^3$	0
0	0	$-38 v[1]^3$	0	$1454 v[1]^4$
0	$-3 v[1]^3$	0	$341 v[1]^4 - 324 v[2]$	0
0	0	$36 v[1]^4 - 72 v[2]$	0	$-2748 v[1]^5 + 8$
0	$-9 v[2]$	0	$-333 v[1]^5 + 1833 v[1] v[2]$	0
0	0	$261 v[1] v[2]$	0	$1158 v[1]^6 - 38$
0	$3 v[1]^5 + 18 v[1] v[2]$	0	$-305 v[1]^6 - 6204 v[1]^2 v[2]$	0
0	0	$-85 v[1]^6 - 612 v[1]^2 v[2]$	0	$13157 v[1]^7 + 13$
0	$-6 v[1]^6 - 27 v[1]^2 v[2]$	0	$2374 v[1]^7 + 16191 v[1]^3 v[2]$	0
0	0	$234 v[1]^7 + 1170 v[1]^3 v[2]$	0	0
0	$9 v[1]^7 + 36 v[1]^3 v[2]$	0	0	0
0	0	0	0	0

This array has 'order'+2 rows, and 'order'+1 columns.
The first row is 0 because the power operation is divisible by x.
Row 's' is the coefficient of x^{s−1}, and is accurate modulo ξ ^{order− s + 3}
(we are making use of the fact that we know *all* of the coefficients in the power series 'x')

```
Dimensions[arrayPBPvnJN]
```

{18, 17}

The Euler class is the power series in ξ which appears in the second row of 'arrayPBPvnH' (The coefficient of x).

```
EulerClass
```

seriesList[0, 0, 2, 0, $-2 v[1]$, 0, $8 v[1]^2$, 0, $-40 v[1]^3$, 0, $170 v[1]^4 - 170 v[2]$, 0,
$-648 v[1]^5 + 2216 v[1] v[2]$, 0, $1424 v[1]^6 - 22512 v[1]^2 v[2]$, 0, $9752 v[1]^7 + 208104 v[1]^3 v[2]$]

The function 'aClass[r]' gives the series which is row r+2 above; the coefficient of x^{r+1}.
This series is accurate modulo ξ ^{order− r + 1}
Note that aClass[r] is undefined for r > 'order'.

```
aClass[2]
```

```
seriesList[1, 0, -9 v[1], 0, 63 v[1]^2, 0, -491 v[1]^3,
 0, 3336 v[1]^4 - 2331 v[2], 0, -19 299 v[1]^5 + 48 000 v[1] v[2], 0,
 75 190 v[1]^6 - 688 481 v[1]^2 v[2], 0, 190 305 v[1]^7 + 8 461 284 v[1]^3 v[2], 0, 0]
```

reduction modulo p−series

We use a simple division algorithm for reduction modulo the reduced p−series:
0. Call a term 'unreduced' if it is some constant multiplied by an integer outside the range [0,p−1]
1. Find the smallest 't' such that some summand of the t−th coefficient is 'unreduced'
2. Add the necessary multiple of the p−series to convert this to a reduced summand.
3. Repeat

Note that, for a single coefficient of ξ ^j different summands are multiplied by different powers of the v[i], and hence reducing one summand has no effect on the other summands of that coefficient

Here is the list of coefficients for the reduced Euler Class
(change Head of input to 'List' for this function)

```
arrayCoeffElimLast[List @@ EulerClass]
```

```
{0, 0, 2, 0, v[1], 0, 0, 0, 2 v[1]^3, 0, v[1]^4 + v[2], 0, 2 v[1]^5, 0, v[1]^2 v[2], 0, v[1]^7 + v[1]^3 v[2]}
```

To see the first 5 reductions, use the following:

```
arrayCoeffElimShowAll[List @@ EulerClass, 5]
```

```
Input

{0, 0, 2, 0, -2 * v[1], 0, 8 * v[1]^2, 0, -40 * v[1]^3,
 0, 170 * v[1]^4 - 170 * v[2], 0, -648 * v[1]^5 + 2216 * v[1] * v[2], 0,
 1424 * v[1]^6 - 22 512 * v[1]^2 * v[2], 0, 9752 * v[1]^7 + 208 104 * v[1]^3 * v[2]}

Reductions

{0, 0, 2, 0, v[1], 0, 0, 0, 32 * v[1]^3, 0,
 -670 * v[1]^4 - 170 * v[2], 0, 8352 * v[1]^5 - 4344 * v[1] * v[2], 0,
 -87 568 * v[1]^6 + 193 992 * v[1]^2 * v[2], 0, 668 528 * v[1]^7 - 5 152 104 * v[1]^3 * v[2]}

{0, 0, 2, 0, v[1], 0, 0, 0, 2 * v[1]^3, 0,
 -590 * v[1]^4 - 170 * v[2], 0, 7632 * v[1]^5 - 4344 * v[1] * v[2], 0,
 -79 168 * v[1]^6 + 193 992 * v[1]^2 * v[2], 0, 578 528 * v[1]^7 - 5 086 504 * v[1]^3 * v[2]}

{0, 0, 2, 0, v[1], 0, 0, 0, 2 * v[1]^3, 0,
 v[1]^4 - 170 * v[2], 0, 6056 * v[1]^5 - 4344 * v[1] * v[2], 0,
 -64 984 * v[1]^6 + 193 992 * v[1]^2 * v[2], 0, 413 048 * v[1]^7 - 5 086 504 * v[1]^3 * v[2]}

{0, 0, 2, 0, v[1], 0, 0, 0, 2 * v[1]^3, 0, v[1]^4 + v[2], 0, 6056 * v[1]^5 - 4800 * v[1] * v[2],
 0, -64 984 * v[1]^6 + 198 096 * v[1]^2 * v[2], 0, 413 048 * v[1]^7 - 5 134 384 * v[1]^3 * v[2]}

{0, 0, 2, 0, v[1], 0, 0, 0, 2 * v[1]^3, 0, v[1]^4 + v[2], 0, 2 * v[1]^5 - 4800 * v[1] * v[2],
 0, -48 840 * v[1]^6 + 198 096 * v[1]^2 * v[2], 0, 267 752 * v[1]^7 - 5 134 384 * v[1]^3 * v[2]}
```

Here is the power operation from above, but with each row reduced modulo the reduced p−series
Note: this output is cutoff in the pdf version of this file

MatrixForm[arrayPBPvnJNRed]

0	0	0	0	0	0	0	0	0	0	0
0	0	2	0	$v[1]$	0	0	0	$2\,v[1]^3$	0	$v[1]^4 + v[2]$
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	$2\,v[1]^3$	0	$v[1]^4 + v[2]$	0	$2\,v[1]^5$	0	$v[1]^2\,v[2]$
0	0	0	0	0	0	0	0	0	0	0
0	0	$v[1]^3$	0	$2\,v[2]$	0	$v[1]^5 + v[1]\,v[2]$	0	$v[1]^6 + 2\,v[1]^2\,v[2]$	0	$2\,v[1]^7 + v[1]^3\,v[2]$
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	$2\,v[1]^7$	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	$2\,v[1]^6$	0	$v[1]^7 + 2\,v[1]^3\,v[2]$	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

Here is the reduced Euler class

EulerClassRed

seriesList[0, 0, 2, 0, $v[1]$, 0, 0, 0, $2\,v[1]^3$, 0, $v[1]^4 + v[2]$, 0, $2\,v[1]^5$, 0, $v[1]^2\,v[2]$, 0, $v[1]^7 + v[1]^3\,v[2]$]

The function ’aClassRed’ is the reduced ’aClass’

aClassRed[2]

seriesList[1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, $v[1]^7$, 0, 0]

■ McClure[n]

Now we compute MC_n(ξ) using the expression given in our companion paper (Proposition 3.2, as of the last revision)

First, we compute the multiplicative inverse of the general series, ’seriesf’

seriesf

{f[0], f[1], f[2], f[3], f[4], f[5], f[6], f[7], f[8], f[9], f[10], f[11], f[12], f[13], f[14], f[15], f[16]}

To speed up the calculation, we substitute zero for those coefficients f[i] such that aClassRed[i] = 0;
this takes advantage of the ’sparseness’ in the a_i.
Also, we have Eu[n] = f[0]^n.

```
fZeroSubstitutions
```

```
{f[1] → 0, f[2] → f[2], f[3] → 0, f[4] → f[4], f[5] → 0, f[6] → f[6], f[7] → 0, f[8] → 0, f[9] → 0,
 f[10] → f[10], f[11] → 0, f[12] → f[12], f[13] → 0, f[14] → 0, f[15] → 0, f[16] → 0}
```

Now ’genmultinv’ is the multiplicative inverse of seriesf, using the zero substitutions and Eu[n]

```
ExpandAll[genmultinv]
```

```
{Eu[-1], 0, -Eu[-2] f[2], 0, Eu[-3] f[2]^2 - Eu[-2] f[4],
 0, -Eu[-4] f[2]^3 + 2 Eu[-3] f[2] f[4] - Eu[-2] f[6], 0,
 Eu[-5] f[2]^4 - 3 Eu[-4] f[2]^2 f[4] + Eu[-3] f[4]^2 + 2 Eu[-3] f[2] f[6], 0, -Eu[-6] f[2]^5 +
  4 Eu[-5] f[2]^3 f[4] - 3 Eu[-4] f[2] f[4]^2 - 3 Eu[-4] f[2]^2 f[6] + 2 Eu[-3] f[4] f[6] - Eu[-2] f[10],
 0, Eu[-7] f[2]^6 - 5 Eu[-6] f[2]^4 f[4] + 6 Eu[-5] f[2]^2 f[4]^2 - Eu[-4] f[4]^3 + 4 Eu[-5] f[2]^3 f[6] -
  6 Eu[-4] f[2] f[4] f[6] + Eu[-3] f[6]^2 + 2 Eu[-3] f[2] f[10] - Eu[-2] f[12], 0,
 -Eu[-8] f[2]^7 + 6 Eu[-7] f[2]^5 f[4] - 10 Eu[-6] f[2]^3 f[4]^2 + 4 Eu[-5] f[2] f[4]^3 -
  5 Eu[-6] f[2]^4 f[6] + 12 Eu[-5] f[2]^2 f[4] f[6] - 3 Eu[-4] f[4]^2 f[6] -
  3 Eu[-4] f[2] f[6]^2 - 3 Eu[-4] f[2]^2 f[10] + 2 Eu[-3] f[4] f[10] + 2 Eu[-3] f[2] f[12],
 0, Eu[-9] f[2]^8 - 7 Eu[-8] f[2]^6 f[4] + 15 Eu[-7] f[2]^4 f[4]^2 - 10 Eu[-6] f[2]^2 f[4]^3 +
  Eu[-5] f[4]^4 + 6 Eu[-7] f[2]^5 f[6] - 20 Eu[-6] f[2]^3 f[4] f[6] + 12 Eu[-5] f[2] f[4]^2 f[6] +
  6 Eu[-5] f[2]^2 f[6]^2 - 3 Eu[-4] f[4] f[6]^2 + 4 Eu[-5] f[2]^3 f[10] -
  6 Eu[-4] f[2] f[4] f[10] + 2 Eu[-3] f[6] f[10] - 3 Eu[-4] f[2]^2 f[12] + 2 Eu[-3] f[4] f[12]}
```

The function ’mult’ computes the product of two series. Here we check that genmultinv is the multiplicative inverse

```
ExpandAll[mult[genmultinv, seriesf /. f[0] → Eu[1] /. fZeroSubstitutions]]
```

```
{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

The formal summands of the MC_n are given by ’formalMcClureSummandsUNXP’;
Eu[i] will be replaced with EulerClassRed^i, and
f[i] will be replaced with aClassRed[i].

```
ExpandAll[formalMcClureSummandsUNXP[2]]
```

```
{0, -3 Eu[1] f[1] v[1], 6 f[1]^2 - 3 Eu[1] f[2], 0, 0, 0, 0, 0, 0}
```

This function is useful for checking the range of accuracy for MC_n, using the known ranges of accuracy for the a_i and the fact that EulerClassRed has no constant coefficient.

Finally, we have the function ’McClure[n]’, which computes our MC_n(\xi)

```
McClure[1]
```

```
{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}
```

```
McClure[4]
```

```
{0, 0, 0, 0, 60, 0, 20 v[1], 0, -45 v[1]^2, 0, 10 v[1]^3, 0,
 -165 v[1]^4 + 20 v[2], 0, -230 v[1]^5 - 90 v[1] v[2], 0, -490 v[1]^6 - 70 v[1]^2 v[2]}
```

```
McClure[pVal - 1]
```

{0, 0, −6, 0, v[1], 0, 4 v[1]^2, 0, −5 v[1]^3, 0, 5 v[1]^4 − 3 v[2],
0, 2 v[1]^5 + 4 v[1] v[2], 0, 10 v[1]^6 − v[1]^2 v[2], 0, −v[1]^7 + v[1]^3 v[2]}

Nicely formated output can be given by ’PrettyMcClure’

```
PrettyMcClure[pVal - 1]
```

$-6\,\xi^{2}+\xi^{4}\,v_{1}+4\,\xi^{6}\,v_{1}^{2}-5\,\xi^{8}\,v_{1}^{3}+\xi^{10}\left(5\,v_{1}^{4}-3\,v_{2}\right)+\xi^{12}\left(2\,v_{1}^{5}+4\,v_{1}\,v_{2}\right)+\xi^{14}\left(10\,v_{1}^{6}-v_{1}^{2}\,v_{2}\right)+\xi^{16}\left(-v_{1}^{7}+v_{1}^{3}\,v_{2}\right)$

We check whether the McClure series is 0 modulo the reduced p−series

```
PrettyRedMcClure[4]
```

0

```
PrettyRedMcClure[pVal - 1]
```

$\xi^{8}\,v_{1}^{3}+2\,\xi^{16}\,v_{1}^{7}+2\,\xi^{10}\,v_{2}+2\,\xi^{14}\,v_{1}^{2}\,v_{2}+\xi^{12}\left(v_{1}^{5}+v_{1}\,v_{2}\right)$

We also have a function explicitly for calculating the first obstruction, since this is the first term of interest.

MC_{2(p−1)} is always divisible by (2p−1)EulerClass^{2p−4}, so this function divides MC_{2(p−1)} by this factor before performing the rest of the calculations. This decreases the range of accuracy by (p−1), but shows non−zero values sooner (for p > 2).

```
FirstObstruction
```

{3, 0, −2 v[1], 0, −v[1]^2, 0, −4 v[1]^3, 0, −6 v[1]^4 − 2 v[2], 0, −6 v[1]^5 − 2 v[1] v[2],
0, −8 v[1]^6 − 2 v[1]^2 v[2], 0, −8 v[1]^3 v[2], 0, −13 v[1]^8 − 4 v[1]^4 v[2] − v[2]^2}

The ’Unk’ in the last term comes from the unknown last term of the reduced p−series.

```
FirstObstructionRed
```

{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2 Unk}

Exercises

1. At the prime 2, McClure[2] is non−zero modulo the reduced 2−series
2. At the prime 3, McClure[4] is non−zero modulo the reduced 3−series
3. At the prime 5, find n such that McClure[n] is non−zero modulo the reduced 5−series