An introduction to Zeta

Tobias Rossmann
National University of Ireland, Galway

Düsseldorf
September 2018

Zeta (2014–)

- ... is a package for Sage (http://www.sagemath.org/) (which also relies on Singular, Normaliz, LattE, ...)
- ... provides methods for computing various types (ask, subobject, ...) of zeta functions in "fortunate cases"
- ... is freely available (http://www.maths.nuigalway.ie/~rossmann/Zeta/)
- ... implements techniques outlined here (https://doi.org/10.1007/978-3-319-70566-8_25)

Sage (2005–)

is a free open-source mathematics software system licensed under the GPL. It builds on top of many existing open-source packages [...] Mission: Creating a viable free open source alternative to Magma, Maple, Mathematica and Matlab.

- Freely available (http://www.sagemath.org/)
- Based upon Python (https://www.python.org/)
- Includes GAP (https://www.gap-system.org/), Singular (http://www.singular.uni-kl.de/), ...
- Runs on Linux, Mac OS X, and Windows
- Command-line ("interactive shell") or browser-based ("notebook") interface

Sage as a symbolic calculator

Recall: the ask zeta function of $\mathrm{M}_1(\mathbf{Z}_p)$ is $\frac{1-p^{-1}T}{(1-T)^2}$.

In [1]:
```
var('p t') # create symbolic variables
Z = (1-p^(-1)*t)/(1-t)^2
```

In [2]:
```
Z.series(t,3) # power series expansion up to given order
```

Out[2]:
```
1 + (-1/p + 2)*t + (-2/p + 3)*t^2 + Order(t^3)
```

In [3]:
```
Z(p=1) # substitution
```

Out[3]:
```
-1/(t - 1)
```

In [4]:
```
Z+1 # arithmetic
```
Out[4]:
```
-(t/p - 1)/(t - 1)^2 + 1
```
In [5]:
```
_.simplify_full()
```
Out[5]:
```
(p*t^2 - (2*p + 1)*t + 2*p)/(p*t^2 - 2*p*t + p)
```
In [6]:
```
_.factor()
```
Out[6]:
```
(p*t^2 - 2*p*t + 2*p - t)/(p*(t - 1)^2)
```

Installing Zeta

- Linux-only at the moment.
- Steps explained in the manual (http://www.maths.nuigalway.ie/~rossmann/Zeta/#download). Quick & dirty:
  - download and extract a file (http://www.maths.nuigalway.ie/~rossmann/Zeta/Zeta-0.3.2-x86_64.tar.bz2)
  - start Sage from the same directory
- Cleaner option due to Tomer Bauer: https://pypi.org/project/zetalib/ (https://pypi.org/project/zetalib/)

Objectives of this tutorial:

- Illustrate what Zeta can (and cannot do).
- Describe valid input and the meaning of output.
- Give examples of theorems found (and sometimes proved) with the help of Zeta.

Getting started

In [7]:
```
import Zeta
```

```
Loading...

ZZZZZZZZZZZZZZZZZZZ                           tttt
Z:::::::::::::::::Z                        ttt:::t
Z:::::::::::::::::Z                        t:::::t
Z:::ZZZZZZZZ:::::Z                         t:::::t
ZZZZZ     Z:::::Z     eeeeeeeeeeee   ttttttt:::::ttttttt     aaaaaaaaaaaaa
        Z:::::Z     ee::::::::::::ee t:::::::::::::::::t     a::::::::::::a
       Z:::::Z     e::::::eeeee:::::et:::::::::::::::::t     aaaaaaaaa:::::a
      Z:::::Z     e::::::e     e:::::tttttt:::::::tttttt              a::::a
     Z:::::Z      e:::::::eeeee::::::e     t:::::t             aaaaaaa:::::a
    Z:::::Z       e:::::::::::::::::e      t:::::t           aa::::::::::::a
   Z:::::Z        e::::::eeeeeeeeeee       t:::::t          a::::aaaa::::::a
ZZZ:::::Z     ZZZZe:::::::e                t:::::t    ttttta::::a    a:::::a
Z::::::ZZZZZZZZ:::e::::::::e               t::::::tttt:::::a::::a    a:::::a
Z:::::::::::::::::Ze::::::::eeeeeeee       tt::::::::::::::a:::::aaaa::::::a
Z:::::::::::::::::Z ee:::::::::::::e         tt:::::::::::tta::::::::::aa:::a
ZZZZZZZZZZZZZZZZZZZ   eeeeeeeeeeeeee           ttttttttttt   aaaaaaaaaa  aaaa

                                                            VERSION 0.3.3-WIP
                                                                 Released: no
                             by
                                 Tobias Rossmann
```

Ask zeta functions in Zeta

Module representations = matrices of linear forms.

In [8]:
```
R.<a,b,c,d> = QQ[] # create polynomial ring
A = matrix([[a,b],[c,d]])
A
```

Out[8]:
```
[a b]
[c d]
```

Zeta can attempt to compute "generic local zeta functions".

In [9]:
```
Z = Zeta.local_zeta_function(A, 'ask')
Z
```

Out[9]:
```
(q^2 - t)/(q^2*(t - 1)^2)
```

Hence, for all but finitely many primes q, the ask zeta function of $\mathrm{M}_2(\mathbf{Z}_q)$ is $\frac{1-q^{-2}T}{(1-T)^2}$. We knew that already.

Note. By default, Zeta attempts to compute ask zeta functions via ∘-duals; this can be overwritten.

Experimental mathematics... using Zeta

My typical applications of Zeta:

- Combine Zeta and databases/classification results to look for patterns. Examples:
    - Loop over all nilpotent Lie algebras of small dimension.
    - Loop over all matrices of a given dimension and given shape.
- "Guess & verify" formulae.
- Search for counterexamples.

Guess & verify: triangular matrices

What is the ask zeta function of

$$\mathfrak{tr}_d(\mathbf{Z}_p) = \begin{bmatrix} * & \cdots & * \\ & \ddots & \vdots \\ & & * \end{bmatrix}?$$

Dimension 2

In [10]:

```
R.<a,b,c> = QQ[]
A = matrix([[a,b],[0,c]])
A
```

Out[10]:

```
[a b]
[0 c]
```

In [11]:

```
Zeta.local_zeta_function(A, 'ask')
```

Out[11]:

```
-(q - t)^2/(q^2*(t - 1)^3)
```

Dimension 3

In [12]:
```
R.<a,b,c,d,e,f> = QQ[]
A = matrix([[a,b,c],[0,d,e],[0,0,f]])
Zeta.local_zeta_function(A, 'ask')
```

Out[12]:
```
(q - t)^3/(q^3*(t - 1)^4)
```

Dimension 4

In [13]:
```
R = PolynomialRing(QQ, 'x', 10)
x = R.gens()
A = matrix([[x[0],x[1],x[2],x[3]], [0,x[4],x[5],x[6]], [0,0,x[7],x[8]], [0,0,0,
x[9]]])
print A
%time Zeta.local_zeta_function(A, 'ask')
```

```
[x0 x1 x2 x3]
[ 0 x4 x5 x6]
[ 0  0 x7 x8]
[ 0  0  0 x9]
CPU times: user 1.46 s, sys: 98 ms, total: 1.56 s
Wall time: 7.18 s
```

Out[13]:
```
-(q - t)^4/(q^4*(t - 1)^5)
```

Higher dimensions

In [14]:
```
def generic_upper_triangular_matrix(d):
    R = PolynomialRing(QQ, 'x', binomial(d+1, 2))
    A = matrix(R, d, d)
    it = iter(R.gens())
    for i in range(d):
        for j in range(i, d):
            A[i,j] = next(it)
    return A
```

In [15]:
```
A = generic_upper_triangular_matrix(5)
print A
```

```
[ x0  x1  x2  x3  x4]
[  0  x5  x6  x7  x8]
[  0   0  x9 x10 x11]
[  0   0   0 x12 x13]
[  0   0   0   0 x14]
```

In [16]:
```
%time Zeta.local_zeta_function(A, 'ask')
```

```
CPU times: user 11.8 s, sys: 200 ms, total: 12 s
Wall time: 34.9 s
```

Out[16]:
```
(q - t)^5/(q^5*(t - 1)^6)
```

These computations inspired the following.

Theorem (R. '18). $Z^{\mathrm{ask}}_{\mathfrak{tr}_d(\mathbf{Z}_p)}(T) = \frac{(1-p^{-1}T)^d}{(1-T)^{d+1}}$.

Note that this is a "nice" formula which is not of constant rank type.

Uniformity: $\sqrt{-1}$

In [17]:
```
R.<a,b> = QQ[]
A = matrix([[a,b],[-b,a]])
A
```

Out[17]:
```
[ a  b]
[-b  a]
```

In [ ]:
```
Zeta.local_zeta_function(A,'ask')
```

In [19]:
```
Z = Zeta.local_zeta_function(A,'ask', symbolic=True)
print Z
```

```
-(q^2*sc_0*t - q^2*t - 2*q*sc_0*t + q^2 + sc_0*t + t^2 - t)/(q^2*(t - 1)^3)
```

In [20]:
```
Zeta.common.symbolic_count_varieties[0]
```

Out[20]:
```
Subvariety of 1-dimensional torus defined by [x^2 + 1]
```

In [21]:
```
Z(sc_0=2).factor() # p == 1 mod 4
```

Out[21]:
```
-(q^2*t + q^2 - 4*q*t + t^2 + t)/(q^2*(t - 1)^3)
```

In [22]:
```
Z(sc_0=0).factor() # p == 3 mod 4
```

Out[22]:
```
(q^2 - t)/(q^2*(t - 1)^2)
```

Circulant matrices

In [23]:
```
R.<a,b> = QQ[]
A = matrix([[a,b],[b,a]])
A
```

Out[23]:
```
[a b]
[b a]
```

In [24]:
```
Zeta.local_zeta_function(A,'ask')
```

Out[24]:
```
-(q^2*t + q^2 - 4*q*t + t^2 + t)/(q^2*(t - 1)^3)
```

In [25]:
```
R.<a,b,c> = QQ[]
A = matrix([[a,b,c],[b,c,a],[c,a,b]])
A
```

Out[25]:
```
[a b c]
[b c a]
[c a b]
```

In [ ]:
```
Zeta.local_zeta_function(A, 'ask')
```

Guess & verify? Diagonal matrices

What is the ask zeta function of $\mathfrak{d}_d(\mathbf{Z}_p) \subset \mathrm{M}_d(\mathbf{Z}_p)$?

In [27]:
```
def generic_diagonal_matrix(d):
    R = PolynomialRing(QQ, 'x', d)
    A = matrix(R,d)
    for i in range(d):
        A[i,i] = R.gen(i)
    return A
```

In [28]:
```
Zeta.local_zeta_function(generic_diagonal_matrix(1), 'ask')
```

Out[28]:
```
(q - t)/(q*(t - 1)^2)
```

In [29]:
```
Zeta.local_zeta_function(generic_diagonal_matrix(2), 'ask')
```

Out[29]:
```
-(q^2*t + q^2 - 4*q*t + t^2 + t)/(q^2*(t - 1)^3)
```

In [30]:
```
Zeta.local_zeta_function(generic_diagonal_matrix(3), 'ask')
```

Out[30]:
```
(q^3*t^2 + 4*q^3*t - 6*q^2*t^2 + q^3 - 12*q^2*t + 12*q*t^2 - t^3 + 6*q*t - 4*t
^2 - t)/(q^3*(t - 1)^4)
```

In [31]:
```
%time Zeta.local_zeta_function(generic_diagonal_matrix(4), 'ask')
```

```
CPU times: user 916 ms, sys: 94.2 ms, total: 1.01 s
Wall time: 28.7 s
```

Out[31]:
```
-(q^4*t^3 + 11*q^4*t^2 - 8*q^3*t^3 + 11*q^4*t - 56*q^3*t^2 + 24*q^2*t^3 + q^4
- 32*q^3*t + 96*q^2*t^2 - 32*q*t^3 + t^4 + 24*q^2*t - 56*q*t^2 + 11*t^3 - 8*q*
t + 11*t^2 + t)/(q^4*(t - 1)^5)
```

Hadamard products

- Recall: $\mathrm{ask}(\theta \oplus \tilde{\theta}) = \mathrm{ask}(\theta) \cdot \mathrm{ask}(\tilde{\theta})$.
- Hadamard product of power series: $(\sum a_n T^n) \star (\sum b_n T^n) = \sum a_n b_n T^n$.
- Hence: $\mathsf{Z}_{\theta \oplus \tilde{\theta}}(T) = \mathsf{Z}_{\theta}(T) \star \mathsf{Z}_{\tilde{\theta}}(T)$

Diagonal matrices (again)

- The ask zeta function of $\mathfrak{d}_d(\mathbf{Z}_p)$ is the dth Hadamard power of $\frac{1-p^{-1}T}{(1-T)^2}$.
- Surprise! This was computed by Brenti (1994).

Theorem. $\mathsf{Z}_{\mathfrak{d}_d(\mathbf{Z}_p)}(T) = \frac{h_d(-p^{-1},T)}{(1-T)^{d+1}}$, where $\mathrm{B}_d = \{\pm 1\} \wr \mathrm{S}_d$ and

$$h_d(X, Y) = \sum_{\sigma \in \mathrm{B}_d} X^{\mathrm{N}(\sigma)} Y^{\mathrm{d_B}(\sigma)}.$$

Theorem & counterexample: bounded denominators

Definition. $F(T) = \sum a_n T^n \in \mathbf{Q}[[T]]$ has **bounded denominators** if $Ba_n \in \mathbf{Z}$ for all n and some integer $B > 0$.

Obvious: if $F(T) \in \mathbf{Q}[[T]] \cap \mathbf{Q}(T)$ can be written over a denominator

$$C(1 - a_1 T^{e_1}) \cdots (1 - a_r T^{e_r})$$

for integers $C, a_i, e_i \geqslant 1$, then $F(T)$ has bounded denominators.

Question. Let $M \subset \mathrm{M}_d(\mathbf{Z}_p)$. Does $\mathsf{Z}_M(T)$ always have bounded denominators?

Supporting evidence:

$\mathrm{M}_d(\mathbf{Z}_p), \mathfrak{d}_d(\mathbf{Z}_p), \mathfrak{tr}_d(\mathbf{Z}_p), \mathfrak{so}_d(\mathbf{Z}_p), \ldots$

Not quite "typical" though

Theorem. Let $\mathfrak{g} \subset \mathfrak{gl}_d(\mathbf{Z}_p)$ be a Lie subalgebra. Then $\mathsf{Z}_{\mathfrak{g}}(T)$ has bounded denominators.

Sketch of proof. $p^{2d}\mathsf{Z}_{\mathfrak{g}}(T)$ enumerates orbits.

♦

In [32]:

```
R.<a,b,c,d> = QQ[]
A = matrix([[a,b,a],[b,c,d],[a,d,c]])
%time Z = Zeta.local_zeta_function(A,'ask')
Z
```

```
CPU times: user 1.6 s, sys: 127 ms, total: 1.72 s
Wall time: 5.39 s
```

Out[32]:

```
(q^4 + 5*q^3*t - 12*q^2*t + 5*q*t + t^2)/((q - t)*q^3*(t - 1)^2)
```

Conjugacy class zeta functions

- Zeta can compute local conjugacy class zeta functions attached to nilpotent Lie algebras.
- Formally: let $\mathfrak{g}$ be a nilpotent Lie $\mathbf{Z}$-algebra with $\mathfrak{g} \approx \mathbf{Z}^\ell$ additively.
- (We pretend to) choose an embedding $\mathfrak{g} \subset \mathfrak{n}_d(\mathbf{Z}[\frac{1}{N}])$.
- Zeta can attempt to compute the conjugacy class zeta functions of $\exp(\mathfrak{g} \otimes \mathbf{Z}_p)$ for $p \gg 0$.
- $\mathfrak{g}$ can be defined e.g. by structure constants.

Example: Heisenberg

In [33]:

```
# Basis (x,y,z) with [x,y]=z=(0,0,1), [y,x]=-z=(0,0,-1)
H = Zeta.Algebra([[(0, 0, 0), (0, 0, 1), (0, 0, 0)],
                  [(0, 0,-1), (0, 0, 0), (0, 0, 0)],
                  [(0, 0, 0), (0, 0, 0), (0, 0, 0)]])
```

In [34]:

```
Zeta.local_zeta_function(H, 'cc')
```

Out[34]:

```
-(t - 1)/((q^2*t - 1)*(q*t - 1))
```

Other types of zeta functions

Zeta can attempt to compute zeta functions enumerating

- ... subalgebras and ideals of additively finitely generated $\mathbf{Z}$-algebras,
- ... (twist isoclasses of) irreducible representations of f.g. nilpotent groups,
- ... submodules of $\mathbf{Z}^d$ of finite index which are invariant under a given set of matrices.

In [35]:
```
Zeta.local_zeta_function(H, 'subalgebras')
```
Out[35]:
```
-(q^2*t^2 + q*t + 1)/((q^3*t^2 - 1)*(q*t + 1)*(q*t - 1)*(t - 1))
```
In [36]:
```
Zeta.local_zeta_function(H, 'ideals')
```
Out[36]:
```
-1/((q^2*t^3 - 1)*(q*t - 1)*(t - 1))
```
In [37]:
```
Zeta.local_zeta_function(H, 'reps')
```
Out[37]:
```
(t - 1)/(q*t - 1)
```

Ideals of $\mathbf{Z}[\![X]\!]$

The **ideal zeta function** of a ring R is

$$\zeta_R(s) = \sum_{\mathfrak{a} \triangleleft R} |R/\mathfrak{a}|^{-s}.$$

Example. $\zeta_{\mathbf{Z}}(s) = \zeta(s)$ (= Riemann zeta function).

Theorem (Lustig 1955). $\zeta_{\mathbf{Z}[[X]]}(s) = \prod_{j=1}^{\infty} \zeta(js - j + 1)$.

Reproving Lustig's theorem

Proposition. $\zeta_{\mathbf{Z}[[X]]}(s) = \lim_{n\to\infty} \zeta_{\mathbf{Z}[X]/(X^n)}(s)$ (in any reasonable sense).

Sketch of proof. Maximal ideals of $\mathbf{Z}[[X]]$ are of the form (X, p) (p prime). It follows that every ideal of finite index contains X^n for some n.

♦

Note. Ideals of $\mathbf{Z}[X]/(X^n)$ = submodules of $\mathbf{Z}^n$ invariant under

$$\begin{bmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{bmatrix} = \text{ companion matrix of } X^n.$$

In [38]:
```
def fun(n):
    R.<x> = QQ[]
    A = companion_matrix(x^n).transpose()
    return Zeta.Algebra(rank=n, operators=[A])
```

In [39]:
```
Zeta.local_zeta_function(fun(2), 'subalgebras')
```

Out[39]:
```
1/((q*t^2 - 1)*(t - 1))
```

In [40]:
```
Zeta.local_zeta_function(fun(3), 'subalgebras')
```

Out[40]:
```
-1/((q^2*t^3 - 1)*(q*t^2 - 1)*(t - 1))
```

In [ ]:
```
Zeta.local_zeta_function(fun(4), 'subalgebras')
```

In [ ]:
```
Zeta.local_zeta_function(fun(4), 'subalgebras', verbose=True)
```

- More patience and computing power lead to the conjecture $\zeta_{\mathbf{Z}_p[X]/(X^n)}(s) = 1/\prod_{j=0}^{n-1}(1 - p^{j-1}t^j)$, where $t = p^{-s}$.
- This can be proved... Zeta's approach doesn't help though.
- Lustig's theorem follows by taking the product over all p.
- (There are some new theorems in it too...)

Higman's conjecture?

A strengthening of Higman's conjecture predicts that for each d, there exists $H_d(X, T) \in \mathbf{Q}(X, T)$ such that for each compact DVR $\mathfrak{O}$ with residue field of size q,

$$\mathsf{Z}^{\mathrm{cc}}_{\mathrm{U}_d(\mathfrak{O})}(T) = H_d(q, T).$$

- For $d = 2$, this is obvious.
- For $d = 3$, we proved this above... at least for almost all residue characteristics.
- For $d = 4$:

In [42]:
```
L = Zeta.lookup('n(4,ZZ)') # algebra already stored in database
```

In [43]:
```
%time Zeta.local_zeta_function(L, 'cc')
```

```
CPU times: user 4.98 s, sys: 263 ms, total: 5.25 s
Wall time: 1min 17s
```

Out[43]:
```
-(q*t - 1)^2/((q^3*t - 1)^2*(q^2*t - 1))
```

Thanks for your time!

Feature requests?

- The development of Zeta was primarily driven by my own research needs.
- What features or functionality would *you* find useful?