1 Basic setup

We wish to compute in the ring $\mathcal{O}_L = \mathbf{F}_q[[t^{1/q^\infty}]]$, the t-adic completion of $\mathbf{F}_q[t^{1/q^\infty}]$. For concreteness we set $q = 3$, and begin by constructing $\mathbf{F}_q$ and $\mathbf{F_q}^\times$ in Sage:

```
q = 2
Fq.<a> = GF(q)
Fqx = Fq.list()[:]; Fqx.remove(0)
```

For computational purposes, we bound the power of q that can occur in the denominators of the exponents of powers of t:

```
bound = 10
```

Now we construct the ring $R = \mathbf{F}_q[t^{1/q^{bound}}]$:

```
vars = ["t"] + ["t_"+str(i) for i in range(1,bound+1)]
R0 = PolynomialRing(Fq,vars)
I = R0.ideal([R0(vars[i])^q-R0(vars[i-1]) for i in range(1,bound+1)])
R = R0.quo(I)
```

We will abbreviate $t^{q^{1/n}}$ by t_n:

```
t = R.gens()[0]
t_1,t_2,t_3,t_4 = var("t_1,t_2,t_3,t_4")
[t_1,t_2,t_3,t_4] = R.gens()[1:5]
```

For example, we can compute in Sage that, in R, we have $t_4^{q^4} - t = 0$, and likewise test that $tt_1 = t_1^{q+1}$: True.

Now we can't actually work in the completion $\mathcal{O}_L$ of R in Sage, but we can at least construct the normalized t-adic valuation. For example, we want $v(t_2^5 + t_4^{q^{10}}) = 5/q^2 = \frac{5}{4}$. Here's a slick way to code this:

```
def v(r):
    S.<x> = PolynomialRing(Fq)
    phi = R0.hom([x]+[0 for i in range(bound)],S)
    s0 = (r^(q^bound)).lift() # lift r to a polynomial
    s = phi(s0)
    return s.valuation() / (q^bound)
```

Unfortunately, this won't work when bound is reasonable large, because taking large powers is a no-no in Sage.

So here is a hacky solution:

```
print "Defining v(r)"
def v(r):
    if r == 0:
        return Infinity
```

```
        r1 = r/r.lc()
        mvs = []
        d = zip(R.variable_names(),
                      ["("+str(q^(-i))+")" for i in range(len(R.variable_names()))])
        #       probably should build this dict once and store it,
        #       rather than creating it every time v(r) is called
        d.append(("*","+"))
        d.append(("^","*"))
        while r1 != 0:
            r1 = r1/r1.lc()
            #print r1.lift()
            m = r1.lm() # get the normalized leading monomial
            ms = str(m) # turn this into a string
            for dd in d:
                ms = ms.replace(dd[0],dd[1])
            mv = QQ(sage.calculus.calculus.symbolic_expression_from_string(ms))
           # mv is the valuation of the leading monomial of r1
            mvs.append(mv)
            r1 = r1 - m
        return min(mvs)
```

Let

```
    r = var("r")
    r = t_2^5 + t_4^(q^10)
```

Then $v(r) = \frac{5}{4}$.

2 A hopefully convergent sequence

Let $K = \mathbf{F}_q((\pi))$ and let $\mathcal{F}$ be a Lubin–Tate formal $\mathcal{O}_K$-module. Setting $K_n = K(\mathcal{F}[\pi^n]), K_\infty = \bigcup K_n$, we have $L = \mathrm{Frac}(\mathcal{O}_L) = \hat{K}_\infty$, and the galois group $H = \mathrm{Gal}(K_\infty/K)$ acts on L.

We define $A_n = \mathbf{F}_q^\times \times \mathbf{F_q}^{n-1}$:

```
    def A(n):
        return map(flatten, CartesianProduct(Fqx,map(list,VectorSpace(Fq,n-1).list())))
```

Jared conjectures that

$$\pi = \lim_{n\to\infty} \prod_{(a_0,\ldots,a_{n-1})\in A} (\sum a_i t^{q^i})^{q^{-n}}$$

should converge to an element $\pi \in \mathcal{O}_L$ which is fixed by the action of H. Write π_n for the expression in the limit.

```
    def pi(n):
        t_n = R.gens()[n]
        return prod([sum([a[i]*t_n^(q^i) for i in range(n)]) for a in A(n)])
```

Examples:

```
pi_1 = pi(1)
print 1
pi_2 = pi(2)
print 2
pi_3 = pi(3)
print 3
pi_4 = pi(4)
print 4
pi_5 = pi(5)
#print 5
pi_6 = pi(6)
#print 6
#pi_7 = pi(7)
#print 7
#pi_8 = pi(8)
#print 8
#pi_9 = pi(9)
#print 9
```

$$\pi_1 = t_1$$
$$\pi_2 = t_1 t_2 + t_1$$
$$\pi_3 = t t_2 t_3 + t t_2 + t_1 t_2 + t t_3 + t + t_1$$

Hmm, it is difficult to tell whether these are converging!

$$v(\pi_2 - \pi_1) = \frac{3}{4}$$
$$v(\pi_3 - \pi_2) = 1$$
$$v(\pi_4 - \pi_3) = \frac{5}{4}$$
$$v(\pi_5 - \pi_4) = \frac{3}{2}$$
$$v(\pi_6 - \pi_5) = \frac{7}{4}$$

The data for a few small choices of q:

$$q = 2 : v(\pi_{n+1} - \pi_n) = \frac{2}{4} + \frac{n}{4}, n = 1, \ldots, 8$$
$$q = 3 : v(\pi_{n+1} - \pi_n) = \frac{5}{9} + \frac{4n}{9}, n = 1, \ldots, 5$$
$$q = 4 : v(\pi_{n+1} - \pi_n) = \frac{12}{16} + \frac{9n}{16}, n = 1, \ldots, 3$$
$$q = 5 : v(\pi_{n+1} - \pi_n) = \frac{20}{25} + \frac{16n}{25}, n = 1, \ldots, 3$$

OK, this does look like a Cauchy sequence! Of course, that could be an artifact of working with just this small example. Can we prove it?