COMPUTATION OF WEIGHT 1 CUSP FORMS OVER $\mathbb{F}_2$

LENNART MEIER

Throughout this document a modular forms of level n over a field K will always mean a Katz modular form for $\Gamma_1(n)$ over K (we will always assume that the characteristic of K does not divide the level and $n \geq 5$).[1] We denote the space of such modular forms of weight k by $M_k(\Gamma_1(n); K)$ and the corresponding space of cusp forms by $S_k(\Gamma_1(n); K)$. The dimension of the spaces only depends on the characteristic of K and

$$\dim_{\mathbb{Q}} M_k(\Gamma_1(n); \mathbb{Q}) \leq \dim_{\mathbb{F}_p} M_k(\Gamma_1(n); \mathbb{F}_p)$$

with equality for $k \geq 2$ and similarly for cusp forms (see e.g. [Mei17, Lemma 4.1]).

For $k \geq 2$, the dimensions of these spaces are easily computable using Riemann–Roch (see e.g. p. 108 of [DS05]). It is much more tricky to compute the dimensions of $S_1(\Gamma_1(n); K)$. For K of characteristic zero, a fast algorithm was found and implemented by Buzzard and Lauder [BL17]. The associated webpage `http://people.maths.ox.ac.uk/lauder/weight1/` contains tables up to level 1500 containing not only the dimensions, but bases, associated Galois extensions etc.

In [Buz14] Buzzard also considers the case of K of finite *odd* characteristic. The smallest level, where he shows that the dimension of $S_1(\Gamma_1(n); \mathbb{F}_p)$ is bigger than that of $S_1(\Gamma_1(n); \mathbb{Q})$ is level 74 (with $p = 3$). Buzzard restricts to modular forms where the $(\mathbb{Z}/n)^{\times}$ action is via a fixed Dirichlet character. As over $\mathbb{F}_p$ not every representation of $(\mathbb{Z}/n)^{\times}$ is 1-dimensional, this is potentially a non-trivial restriction. Apart from this restriction, his search appears to be exhaustive, i.e. there is no smaller level with non-liftable forms in odd characteristic.

Gabor Wiese also wrote a `MAGMA` package computing weight 1 cusp forms in characteristic 2, but only for $\Gamma_0(n)$. Schaeffer [Sch14] has a fast algorithm as well, but requires again a Dirichlet character.

Our modest aim was to complement these results for $K = \mathbb{F}_2$ without imposing any Dirichlet character. We do not propose a new algorithm, but rather give implementations of two variants of an algorithm proposed by Edixhoven in [Edi06] (which was already the basis of Wiese's work).

Proposition 1 ([Edi06], Prop 4.2)**.** *Let $g = \sum_{i=1}^{\infty} a_i q^i$ be a weight 2 cusp form of level n over $\mathbb{F}_2$. Let $B = \frac{n^2}{6} \prod_{l|n}(1 - \frac{1}{l^2})$, where the product runs over all primes dividing n (the so-called* Sturm bound*). Assume that $a_i = 0$ for all odd $i \leq B$. Then $g = f^2$ for a weight-1 cusp form f of level n over $\mathbb{F}_2$.*

As the space of weight 2 cusp forms is computable in `SAGE`, this leads easily to an algorithm that gives us even the q-expansions of a basis of the space of weight 1 cusp forms over $\mathbb{F}_2$. A sample implementation is the following:[2]

Date: April 12, 2018.

[1] If K does not contain an n-th root of unity, $\Gamma_1(n)$-level structures should be understood to provide an embedding of μ_n instead of $\mathbb{Z}/n$, or else one has to use a non-standard notion of q-expansions.

[2] I am not very experienced in `SAGE`, so the implementations below are probably far from optimal, but sufficient for the small levels we are considering.

```
def deg(n): #index of Gamma_1(n) in SL_2(Z)
    d = n^2
    for p in list(factor(n)):
        d = d*(1-1/(p[0]^2))
    return d

def fill(L, length): #fill list with zeros
    n = length - len(L)
    return L} + ([0]*n)

def CuspF2q(n, speed=12):
    #gives a basis of the space weight 1 cusp forms for Gamma1(n) over F_2
    # in vector notation
    M = CuspForms(Gamma1(n),2, GF(2))
    Prec =2*(floor(deg(n)/speed))+2 #for speed =12 this is the Sturm bound

    V = VectorSpace(GF(2), Prec)
    L = [fill(list(f.qexp(Prec)),Prec) for f in M.basis()]
    LV = [V(l) for l in L]
    W = V.subspace(LV)

    def square(Listje): return list(reduce(lambda s, t: s} + t,
                                           zip(Listje,[0]*Prec), ()))[:Prec]
    Lsquare = map(square, L)
    LVsquare = [V(l) for l in Lsquare]
    Wsquare = V.subspace(LVsquare)

    Meet = W.intersection(Wsquare)
    Base = [B[0::2] for B in Meet.basis()]
    #Odd entries deleted = taking preimage of Frobenius
    # to obtain weight 1 cusp form
    return Base

def CuspF2qexp(n, speed=12): #transforming vector notation into power series
    C = CuspF2q(n, speed)
    L = [[b[i] for i in range(b.length())] for b in C]
    R.<q> = PowerSeriesRing(GF(2))
    qexps = [R(l) for l in L]
    return qexps
```

This can be sped up by ignoring q-expansions and computing directly with Hecke algebras (as this can be done via modular symbols). The proposition above becomes in this language:

Proposition 2 ([Edi06], Prop 4.10)**.** *Let again* $B = \frac{n^2}{6}\prod_{l|n}(1-\frac{1}{l^2})$*. Set* $V = S_2(\Gamma_1(n);\mathbb{F}_2)$*. Let* $\mathbb{T}_{odd}$ *be the sub vector space of* $\operatorname{End}_{\mathbb{F}_2}(V)$ *generated by the Hecke operators* T_i *with* $i \leq B$ *odd and* $\mathbb{T}$ *be the full Hecke algebra, i.e. the subspace of* $\operatorname{End}_{\mathbb{F}_2}(V)$ *generated by* all *Hecke operators. Then*

$$\dim_{\mathbb{F}_2} S_1(\Gamma_1(n);\mathbb{F}_2) = \dim_{\mathbb{F}_2}\mathbb{T} - \dim_{\mathbb{F}_2}\mathbb{T}_{odd}.$$

Note that $\dim_{\mathbb{F}_2} \mathbb{T}$ agrees with $\dim_{\mathbb{F}_2} S_2(\Gamma_1(n); \mathbb{F}_2)$, which is half the dimension of that of cuspdial symbols.

A sample implementation of the resulting algorithm is the following (using the same function `deg(n)` as above):

```
def vect(A, dim): #transforms a matrix into a list of n^2 elements
    v = [ ]
    for i in range(dim):
        v.extend(list(A.row(i)))
    return v

def CuspF2(n):
    M = ModularSymbols(Gamma1(n),2,base_ring=GF(2)).cuspidal_subspace()
    Prec =floor(deg(n)/12)+1 #Half the Sturm bound

    T = M.hecke_algebra()
    dim = T.module().dimension()

    Lodd = [T.hecke_matrix(2*n+1) for n in range(Prec)]
    V = VectorSpace(GF(2), dim^2)
    LoddV = [V(vect(l, dim)) for l in Lodd]
    Vodd = V.subspace(LoddV)
    dimodd = Vodd.dimension()
    return dim/2- dimodd
```

Remark 3. There is a number of other approaches possible to compute weight 1 cusp forms, but some of these involve weight k cusp forms for $k > 2$ and these spaces of cusp forms grow very fast in dimension and are thus expensive to compute.

Running the second algorithm shows that

$$\dim_{\mathbb{F}_2} S_1(\Gamma_1(n); \mathbb{F}_2) = \dim_{\mathbb{Q}} S_1(\Gamma_1(n); \mathbb{Q})$$

for all odd $n < 70$ except for $n = 65$, where we have $\dim_{\mathbb{F}_2} S_1(\Gamma_1(65); \mathbb{F}_2) = 2$ while $S_1(\Gamma_1(65); \mathbb{Q}) = 0$ (which be obtain by `MAGMA` or the tables by Buzzard and Lauder).

Running the first algorithm for $n = 65$, gives us the q-expansions of a basis of $S_1(\Gamma_1(n); \mathbb{F}_2)$.

$$\begin{aligned} f_1 = q^2 &+ q^{10} + q^{12} + q^{14} + q^{16} + q^{26} + q^{28} + q^{34} + q^{38} + q^{42} + q^{44} + q^{50} + q^{54} + q^{60} + q^{66} \\ &+ q^{68} + q^{70} + q^{76} + q^{80} + q^{86} + q^{92} + q^{96} + q^{102} + q^{112} + q^{114} + q^{116} + q^{118} + q^{122} \\ &+ q^{128} + q^{130} + q^{132} + q^{138} + q^{140} + q^{142} + q^{148} + q^{154} + q^{156} + q^{164} + q^{170} + q^{172} + \cdots \\ f_2 = q^4 &+ q^6 + q^{12} + q^{14} + q^{20} + q^{22} + q^{30} + q^{32} + q^{34} + q^{38} + q^{44} + q^{46} + q^{48} + q^{52} + q^{58} \\ &+ q^{60} + q^{66} + q^{70} + q^{74} + q^{78} + q^{82} + q^{84} + q^{86} + q^{92} + q^{96} + q^{100} + q^{108} + q^{110} + q^{112} \\ &+ q^{116} + q^{118} + q^{122} + q^{134} + q^{138} + q^{142} + q^{148} + q^{150} + q^{156} + q^{160} + q^{162} + q^{164} + q^{170} + \cdots \end{aligned}$$

Acknowledgments. I want to thank Gabor Wiese for a helpful email exchange.

References

[BL17] Kevin Buzzard and Alan Lauder. A computation of modular forms of weight one and small level. *Ann. Math. Qué.*, 41(2):213–219, 2017. 1

[Buz14] Kevin Buzzard. Computing weight one modular forms over $\mathbb{C}$ with $\overline{\mathbb{F}}_p$. In *Computations with modular forms*, volume 6 of *Contrib. Math. Comput. Sci.*, pages 129–146. Springer, Cham, 2014. 1

[DS05] Fred Diamond and Jerry Shurman. *A first course in modular forms*, volume 228 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 2005. 1

[Edi06] Bas Edixhoven. Comparison of integral structures on spaces of modular forms of weight two, and computation of spaces of forms mod 2 of weight one. *J. Inst. Math. Jussieu*, 5(1):1–34, 2006. With appendix A (in French) by Jean-François Mestre and appendix B by Gabor Wiese. 1, 2

[Mei17] Lennart Meier. Decomposition results for rings of modular forms. *arXiv preprint arXiv:1710.03461*, 2017. 1

[Sch14] George J Schaeffer. Hecke stability and weight 1 modular forms. *Mathematische Zeitschrift*, pages 1–33, 2014. 1

Mathematical Institute, Budapestlaan 6, 3584 CD Utrecht, Nederland
E-mail address: f.l.m.meier@uu.nl