Classical Modular Forms in Pari/GP

Karim Belabas and Henri Cohen,
IMB, Université de Bordeaux

DESY Zeuthen, October 24, 2017

Introduction

The aim of this talk is to announce and describe a new package for working with classical modular forms in Pari/GP, which has a number of features which do not exist in other software packages such as Sage or Magma.

It is a distant relative to programs that we wrote with N. Skoruppa and D. Zagier more than 2 decades ago.

Four parts:

1. Theory and Algorithms.
2. Implementation details.
3. Examples.
4. Advanced Examples.

Introduction

The aim of this talk is to announce and describe a new package for working with classical modular forms in `Pari/GP`, which has a number of features which do not exist in other software packages such as `Sage` or `Magma`.

It is a distant relative to programs that we wrote with N. Skoruppa and D. Zagier more than 2 decades ago.

Four parts:

1. Theory and Algorithms.
2. Implementation details.
3. Examples.
4. Advanced Examples.

Introduction

The aim of this talk is to announce and describe a new package for working with classical modular forms in Pari/GP, which has a number of features which do not exist in other software packages such as Sage or Magma.

It is a distant relative to programs that we wrote with N. Skoruppa and D. Zagier more than 2 decades ago.

Four parts:

1. Theory and Algorithms.
2. Implementation details.
3. Examples.
4. Advanced Examples.

Existing Methods

There exist many methods for computing spaces of classical modular forms. We mention four:

- The use of Modular symbols, and variants due to Yu. Manin, L. Merel, J. Cremona, etc... This is the method used in `Sage` and `magma`.
- The use of theta functions associated to quaternion algebras, together with Brandt matrices. This is the method explained in a book of H. Hijikata, A. Pizer, and T. Schemanske.
- The direct use of the Eichler–Selberg trace formula, generalized by Hijikata and the author.
- The use of products of two Eisenstein Series, based on a theorem of L. Borisov and P. Gunnells.

Existing Methods

There exist many methods for computing spaces of classical modular forms. We mention four:

- The use of Modular symbols, and variants due to Yu. Manin, L. Merel, J. Cremona, etc... This is the method used in `Sage` and `magma`.
- The use of theta functions associated to quaternion algebras, together with Brandt matrices. This is the method explained in a book of H. Hijikata, A. Pizer, and T. Schemanske.
- The direct use of the Eichler–Selberg trace formula, generalized by Hijikata and the author.
- The use of products of two Eisenstein Series, based on a theorem of L. Borisov and P. Gunnells.

Existing Methods

There exist many methods for computing spaces of classical modular forms. We mention four:

- The use of Modular symbols, and variants due to Yu. Manin, L. Merel, J. Cremona, etc... This is the method used in `Sage` and `magma`.
- The use of theta functions associated to quaternion algebras, together with Brandt matrices. This is the method explained in a book of H. Hijikata, A. Pizer, and T. Schemanske.
- The direct use of the Eichler–Selberg trace formula, generalized by Hijikata and the author.
- The use of products of two Eisenstein Series, based on a theorem of L. Borisov and P. Gunnells.

Existing Methods

There exist many methods for computing spaces of classical modular forms. We mention four:

- The use of Modular symbols, and variants due to Yu. Manin, L. Merel, J. Cremona, etc... This is the method used in `Sage` and `magma`.
- The use of theta functions associated to quaternion algebras, together with Brandt matrices. This is the method explained in a book of H. Hijikata, A. Pizer, and T. Schemanske.
- The direct use of the Eichler–Selberg trace formula, generalized by Hijikata and the author.
- The use of products of two Eisenstein Series, based on a theorem of L. Borisov and P. Gunnells.

Additional Weights

Our package implements the last two methods, which are complementary, and which allow the construction of the spaces $M_k(\Gamma_0(N), \chi)$ for $k \geq 2$ integral. We can also construct the following:

- Modular forms of weight $k = 1$ are constructed using two complementary methods: first forms associated to Hecke characters of finite order on imaginary and real quadratic fields; and second, when this is not sufficient, as quotients of higher weight forms using G. Schaeffer's Hecke stability method.
- Modular forms of weight $k = 1/2$ are constructed using their explicit description by J.-P. Serre and H. Stark as unary theta series.
- Modular forms of half-integral weight $k \geq 3/2$ are constructed as quotients of forms of integral weight by the two "coprime" forms $\theta(\tau)$ and $\theta(2\tau)$.

Additional Weights

Our package implements the last two methods, which are complementary, and which allow the construction of the spaces $M_k(\Gamma_0(N), \chi)$ for $k \geq 2$ integral. We can also construct the following:

- Modular forms of weight $k = 1$ are constructed using two complementary methods: first forms associated to Hecke characters of finite order on imaginary and real quadratic fields; and second, when this is not sufficient, as quotients of higher weight forms using G. Schaeffer's Hecke stability method.
- Modular forms of weight $k = 1/2$ are constructed using their explicit description by J.-P. Serre and H. Stark as unary theta series.
- Modular forms of half-integral weight $k \geq 3/2$ are constructed as quotients of forms of integral weight by the two "coprime" forms $\theta(\tau)$ and $\theta(2\tau)$.

Additional Weights

Our package implements the last two methods, which are complementary, and which allow the construction of the spaces $M_k(\Gamma_0(N), \chi)$ for $k \geq 2$ integral. We can also construct the following:

- Modular forms of weight $k = 1$ are constructed using two complementary methods: first forms associated to Hecke characters of finite order on imaginary and real quadratic fields; and second, when this is not sufficient, as quotients of higher weight forms using G. Schaeffer's Hecke stability method.
- Modular forms of weight $k = 1/2$ are constructed using their explicit description by J.-P. Serre and H. Stark as unary theta series.
- Modular forms of half-integral weight $k \geq 3/2$ are constructed as quotients of forms of integral weight by the two "coprime" forms $\theta(\tau)$ and $\theta(2\tau)$.

Additional Weights

Our package implements the last two methods, which are complementary, and which allow the construction of the spaces $M_k(\Gamma_0(N), \chi)$ for $k \geq 2$ integral. We can also construct the following:

- Modular forms of weight $k = 1$ are constructed using two complementary methods: first forms associated to Hecke characters of finite order on imaginary and real quadratic fields; and second, when this is not sufficient, as quotients of higher weight forms using G. Schaeffer's Hecke stability method.
- Modular forms of weight $k = 1/2$ are constructed using their explicit description by J.-P. Serre and H. Stark as unary theta series.
- Modular forms of half-integral weight $k \geq 3/2$ are constructed as quotients of forms of integral weight by the two "coprime" forms $\theta(\tau)$ and $\theta(2\tau)$.

Additional Features I

The constructions using trace formulas (as well as those using modular symbols) together with the additional methods above and the explicit construction of Eisenstein series allow us to construct spaces of classical modular forms and to perform most operations that can be required: construction of the new space, action of Hecke operators, calculation of Hecke eigenforms, etc...

But already some limitations are apparent: computing the action of the Atkin–Lehner operators is limited to certain cases (except if the level is squarefree), and other computations are nearly impossible.

Additional Features I

The constructions using trace formulas (as well as those using modular symbols) together with the additional methods above and the explicit construction of Eisenstein series allow us to construct spaces of classical modular forms and to perform most operations that can be required: construction of the new space, action of Hecke operators, calculation of Hecke eigenforms, etc...

But already some limitations are apparent: computing the action of the Atkin–Lehner operators is limited to certain cases (except if the level is squarefree), and other computations are nearly impossible.

Products of two Eisenstein Series

This is where the last method mentioned above enters: using a theorem essentially due to Borisov–Gunnells one can show that $M_k(\Gamma_0(N), \chi)$ is generated by products of at most two Eisenstein series, possibly divided by another.

Also, we can by very tedious computations obtain the Fourier expansion of $E|_k\gamma$ for any Eisenstein series and any $\gamma \in M_2^+(\mathbb{Q})$. This allows us to do many more operations on modular forms than before, and this is where we believe our package is more complete:

Products of two Eisenstein Series

This is where the last method mentioned above enters: using a theorem essentially due to Borisov–Gunnells one can show that $M_k(\Gamma_0(N), \chi)$ is generated by products of at most two Eisenstein series, possibly divided by another.

Also, we can by very tedious computations obtain the Fourier expansion of $E|_k\gamma$ for any Eisenstein series and any $\gamma \in M_2^+(\mathbb{Q})$. This allows us to do many more operations on modular forms than before, and this is where we believe our package is more complete:

Additional Features II

- Computation of the Fourier expansion of $F|_k\gamma$ for any modular form F (including half-integral weight) and any $\gamma \in M_2^+(\mathbb{Q})$, so in particular expansions and valuations at any cusp, regular or not.
- Action of all the Atkin–Lehner operators and of Atkin–Lehner eigenvalues and pseudo-eigenvalues.
- Computation of all modular symbols and all period polynomials associated to a modular form.
- Computation of Petersson products of two arbitrary modular forms, not necessarily eigenforms, and in particular decomposition of any form in the new space on normalized eigenforms.
- Fast numerical evaluation of an arbitrary modular form, even at a point very near the real axis.

Additional Features II

- Computation of the Fourier expansion of $F|_k\gamma$ for any modular form F (including half-integral weight) and any $\gamma \in M_2^+(\mathbb{Q})$, so in particular expansions and valuations at any cusp, regular or not.
- Action of all the Atkin–Lehner operators and of Atkin–Lehner eigenvalues and pseudo-eigenvalues.
- Computation of all modular symbols and all period polynomials associated to a modular form.
- Computation of Petersson products of two arbitrary modular forms, not necessarily eigenforms, and in particular decomposition of any form in the new space on normalized eigenforms.
- Fast numerical evaluation of an arbitrary modular form, even at a point very near the real axis.

Additional Features II

- Computation of the Fourier expansion of $F|_k\gamma$ for any modular form F (including half-integral weight) and any $\gamma \in M_2^+(\mathbb{Q})$, so in particular expansions and valuations at any cusp, regular or not.
- Action of all the Atkin–Lehner operators and of Atkin–Lehner eigenvalues and pseudo-eigenvalues.
- Computation of all modular symbols and all period polynomials associated to a modular form.
- Computation of Petersson products of two arbitrary modular forms, not necessarily eigenforms, and in particular decomposition of any form in the new space on normalized eigenforms.
- Fast numerical evaluation of an arbitrary modular form, even at a point very near the real axis.

Additional Features II

- Computation of the Fourier expansion of $F|_k\gamma$ for any modular form F (including half-integral weight) and any $\gamma \in M_2^+(\mathbb{Q})$, so in particular expansions and valuations at any cusp, regular or not.
- Action of all the Atkin–Lehner operators and of Atkin–Lehner eigenvalues and pseudo-eigenvalues.
- Computation of all modular symbols and all period polynomials associated to a modular form.
- Computation of Petersson products of two arbitrary modular forms, not necessarily eigenforms, and in particular decomposition of any form in the new space on normalized eigenforms.
- Fast numerical evaluation of an arbitrary modular form, even at a point very near the real axis.

Additional Features II

- Computation of the Fourier expansion of $F|_k\gamma$ for any modular form F (including half-integral weight) and any $\gamma \in M_2^+(\mathbb{Q})$, so in particular expansions and valuations at any cusp, regular or not.
- Action of all the Atkin–Lehner operators and of Atkin–Lehner eigenvalues and pseudo-eigenvalues.
- Computation of all modular symbols and all period polynomials associated to a modular form.
- Computation of Petersson products of two arbitrary modular forms, not necessarily eigenforms, and in particular decomposition of any form in the new space on normalized eigenforms.
- Fast numerical evaluation of an arbitrary modular form, even at a point very near the real axis.

We now begin by explaining in more detail the results and algorithms that we used. The first main tool is the trace formula, initially due to M. Eichler and A. Selberg, and generalized by D. Zagier, H. Hijikata, and the author. It is of the following form, assuming $k \geq 2$ and $\chi(-1) = (-1)^k$:

$$\mathrm{Tr}_{S_k(\Gamma_0(N),\chi)}\, T(n) = A_1 + A_2 + A_3 + A_4 \ ,$$

where the A_i will be mentioned below. Note two important and related facts:

- This formula is valid for any n, coprime to the level N or not.
- Even though closely related to each other, the action of a Hecke operator $T(n)$ is not necessaily the same on $\Gamma_0(N)$ and on $\Gamma_0(M)$ for $N \mid M$, so more properly one should write $T_N(n)$ instead of $T(n)$.

Part I: Algorithms; Trace Formulas I

We now begin by explaining in more detail the results and algorithms that we used. The first main tool is the trace formula, initially due to M. Eichler and A. Selberg, and generalized by D. Zagier, H. Hijikata, and the author. It is of the following form, assuming $k \geq 2$ and $\chi(-1) = (-1)^k$:

$$\mathrm{Tr}_{S_k(\Gamma_0(N),\chi)}\, T(n) = A_1 + A_2 + A_3 + A_4 \ ,$$

where the A_i will be mentioned below. Note two important and related facts:

- This formula is valid for any n, coprime to the level N or not.
- Even though closely related to each other, the action of a Hecke operator $T(n)$ is not necessaily the same on $\Gamma_0(N)$ and on $\Gamma_0(M)$ for $N \mid M$, so more properly one should write $T_N(n)$ instead of $T(n)$.

Trace Formulas II

No use in a talk to give formulas explicitly. Note simply that:

- It involves class numbers of imaginary quadratic orders. Thus, essential to precompute a sufficiently large table beforehand. Done using classical recursions on Hurwitz class numbers.
- It involves the sum of four multiplicative elementary arithmetic functions. Can either use multiplicativity, or precompute their values. For small level (up to 1000, say), precomputation is faster, larger levels multiplicativity.
- We use a cache method both to store the Hurwitz class number and to avoid recomputing already computed traces (e.g., computing class numbers up to 2.10^7 only requires 15s).

Trace Formulas II

No use in a talk to give formulas explicitly. Note simply that:

- It involves class numbers of imaginary quadratic orders. Thus, essential to precompute a sufficiently large table beforehand. Done using classical recursions on Hurwitz class numbers.
- It involves the sum of four multiplicative elementary arithmetic functions. Can either use multiplicativity, or precompute their values. For small level (up to 1000, say), precomputation is faster, larger levels multiplicativity.
- We use a cache method both to store the Hurwitz class number and to avoid recomputing already computed traces (e.g., computing class numbers up to 2.10^7 only requires 15s).

Trace Formulas II

No use in a talk to give formulas explicitly. Note simply that:

- It involves class numbers of imaginary quadratic orders. Thus, essential to precompute a sufficiently large table beforehand. Done using classical recursions on Hurwitz class numbers.
- It involves the sum of four multiplicative elementary arithmetic functions. Can either use multiplicativity, or precompute their values. For small level (up to 1000, say), precomputation is faster, larger levels multiplicativity.
- We use a cache method both to store the Hurwitz class number and to avoid recomputing already computed traces (e.g., computing class numbers up to 2.10^7 only requires 15s).

Trace Formulas III

Simpler to work directly with the new space: we use a trace formula for the new space obtained by a nontrivial Möbius inversion from the full cuspidal space (J. Bober, A. Booker, M. Lee, and the author).

Notation: $\mathrm{Tr}(N,n)$ ($\mathrm{Tr}^{\mathrm{new}}(N,n)$) for trace of $T(n)$ on $S_k(\Gamma_0(N),\chi)$ (resp., $S_k^{\mathrm{new}}(\Gamma_0(N),n)$) ($k$ and χ are fixed).

$$\mathrm{Tr}^{\mathrm{new}}(N,n)=\sum_{\mathfrak{f}|M|N}\sum_{\substack{d|\gcd(M/\mathfrak{f},N_1)\\ d^2|n}}\chi_{\mathfrak{f}}(d)d^{k-1}\beta_{n/d^2}(N/M)\,\mathrm{Tr}(M/d,n/d^2).$$

$N=N_1N_2$ with $\gcd(N_1,N_2)=1$, N_1 squarefree, N_2 squarefull, $\mathfrak{f}$ conductor, $\chi_{\mathfrak{f}}$ primitive character equivalent to χ, $\beta_m(N)$ elementary arithmetic function defined by

$$\zeta^{-2}(s)\prod_{p|m}(1-1/p^s)=\sum_{N\geq 1}\beta_m(N)/N^s\ .$$

Trace Formulas III

Simpler to work directly with the new space: we use a trace formula for the new space obtained by a nontrivial Möbius inversion from the full cuspidal space (J. Bober, A. Booker, M. Lee, and the author).

Notation: $\mathrm{Tr}(N,n)$ ($\mathrm{Tr}^{\mathrm{new}}(N,n)$) for trace of $T(n)$ on $S_k(\Gamma_0(N),\chi)$ (resp., $S_k^{\mathrm{new}}(\Gamma_0(N),n)$) ($k$ and χ are fixed).

$$\mathrm{Tr}^{\mathrm{new}}(N,n)=\sum_{\mathfrak{f}|M|N}\sum_{\substack{d|\gcd(M/\mathfrak{f},N_1)\\ d^2|n}}\chi_{\mathfrak{f}}(d)d^{k-1}\beta_{n/d^2}(N/M)\,\mathrm{Tr}(M/d,n/d^2).$$

$N=N_1N_2$ with $\gcd(N_1,N_2)=1$, N_1 squarefree, N_2 squarefull, $\mathfrak{f}$ conductor, $\chi_{\mathfrak{f}}$ primitive character equivalent to χ, $\beta_m(N)$ elementary arithmetic function defined by

$$\zeta^{-2}(s)\prod_{p|m}(1-1/p^s)=\sum_{N\geq 1}\beta_m(N)/N^s\ .$$

Trace Formulas III

Simpler to work directly with the new space: we use a trace formula for the new space obtained by a nontrivial Möbius inversion from the full cuspidal space (J. Bober, A. Booker, M. Lee, and the author).

Notation: $\mathrm{Tr}(N,n)$ ($\mathrm{Tr}^{\mathrm{new}}(N,n)$) for trace of $T(n)$ on $S_k(\Gamma_0(N),\chi)$ (resp., $S_k^{\mathrm{new}}(\Gamma_0(N),n)$) ($k$ and χ are fixed).

$$\mathrm{Tr}^{\mathrm{new}}(N,n)=\sum_{\mathfrak{f}|M|N}\sum_{\substack{d|\gcd(M/\mathfrak{f},N_1)\\ d^2|n}}\chi_{\mathfrak{f}}(d)d^{k-1}\beta_{n/d^2}(N/M)\,\mathrm{Tr}(M/d,n/d^2).$$

$N=N_1N_2$ with $\gcd(N_1,N_2)=1$, N_1 squarefree, N_2 squarefull, $\mathfrak{f}$ conductor, $\chi_{\mathfrak{f}}$ primitive character equivalent to χ, $\beta_m(N)$ elementary arithmetic function defined by

$$\zeta^{-2}(s)\prod_{p|m}(1-1/p^s)=\sum_{N\geq 1}\beta_m(N)/N^s\ .$$

Construction of $S_k(\Gamma_0(N),\chi)$

Three steps:

1. First note that $\mathcal{T}^{\mathrm{new}}(N)=\sum_{n\geq 1}\mathrm{Tr}^{\mathrm{new}}(N,n)q^n$ is simply equal to the sum of the normalized eigenforms, hence $\mathcal{T}^{\mathrm{new}}(N)\in S_k^{\mathrm{new}}(\Gamma_0(N),\chi)$.
2. Easy to show the $T(m)\mathcal{T}^{\mathrm{new}}(N)$ for $m\leq B$ generate S_k^{new} for B sufficiently large. Since dimension known ($\dim=\mathrm{Tr}^{\mathrm{new}}(N,1)$) we know when to stop.
3. Finally

$$S_k(\Gamma_0(N),\chi)=\bigoplus_{\mathrm{f}(\chi)|M|N}\ \bigoplus_{d|N/M} B(d)S_k^{\mathrm{new}}(\Gamma_0(M),\chi)\,.$$

Note: difficult but possible to express explicitly $\mathcal{T}(N)$ in terms of $\mathcal{T}^{\mathrm{new}}(M)$ for $M\mid N$ and conversely.

Construction of $S_k(\Gamma_0(N),\chi)$

Three steps:

1. First note that $\mathcal{T}^{\rm new}(N)=\sum_{n\geq 1}\mathrm{Tr}^{\rm new}(N,n)q^n$ is simply equal to the sum of the normalized eigenforms, hence $\mathcal{T}^{\rm new}(N)\in S_k^{\rm new}(\Gamma_0(N),\chi)$.
2. Easy to show the $T(m)\mathcal{T}^{\rm new}(N)$ for $m\leq B$ generate $S_k^{\rm new}$ for B sufficiently large. Since dimension known ($\dim=\mathrm{Tr}^{\rm new}(N,1)$) we know when to stop.
3. Finally

$$S_k(\Gamma_0(N),\chi)=\bigoplus_{\mathrm{f}(\chi)|M|N}\ \bigoplus_{d|N/M}B(d)S_k^{\rm new}(\Gamma_0(M),\chi)\,.$$

Note: difficult but possible to express explicitly $\mathcal{T}(N)$ in terms of $\mathcal{T}^{\rm new}(M)$ for $M\mid N$ and conversely.

Construction of $S_k(\Gamma_0(N),\chi)$

Three steps:

1. First note that $\mathcal{T}^{\mathrm{new}}(N)=\sum_{n\geq 1}\mathrm{Tr}^{\mathrm{new}}(N,n)q^n$ is simply equal to the sum of the normalized eigenforms, hence $\mathcal{T}^{\mathrm{new}}(N)\in S_k^{\mathrm{new}}(\Gamma_0(N),\chi)$.
2. Easy to show the $T(m)\mathcal{T}^{\mathrm{new}}(N)$ for $m\leq B$ generate S_k^{new} for B sufficiently large. Since dimension known ($\dim=\mathrm{Tr}^{\mathrm{new}}(N,1)$) we know when to stop.
3. Finally

$$S_k(\Gamma_0(N),\chi)=\bigoplus_{\mathfrak{f}(\chi)|M|N}\ \bigoplus_{d|N/M} B(d)S_k^{\mathrm{new}}(\Gamma_0(M),\chi)\ .$$

Note: difficult but possible to express explicitly $\mathcal{T}(N)$ in terms of $\mathcal{T}^{\mathrm{new}}(M)$ for $M\mid N$ and conversely.

Construction of $S_k(\Gamma_0(N),\chi)$

Three steps:

1. First note that $\mathcal{T}^{\rm new}(N)=\sum_{n\geq 1}\mathrm{Tr}^{\rm new}(N,n)q^n$ is simply equal to the sum of the normalized eigenforms, hence $\mathcal{T}^{\rm new}(N)\in S_k^{\rm new}(\Gamma_0(N),\chi)$.
2. Easy to show the $T(m)\mathcal{T}^{\rm new}(N)$ for $m\leq B$ generate $S_k^{\rm new}$ for B sufficiently large. Since dimension known ($\dim=\mathrm{Tr}^{\rm new}(N,1)$) we know when to stop.
3. Finally

$$S_k(\Gamma_0(N),\chi)=\bigoplus_{\mathfrak{f}(\chi)|M|N}\ \bigoplus_{d|N/M}B(d)S_k^{\rm new}(\Gamma_0(M),\chi)\ .$$

Note: difficult but possible to express explicitly $\mathcal{T}(N)$ in terms of $\mathcal{T}^{\rm new}(M)$ for $M\mid N$ and conversely.

Construction of $M_k(\Gamma_0(N),\chi)$

Of course: $M_k(\Gamma_0(N),\chi) = S_k(\Gamma_0(N),\chi) \oplus \mathcal{E}_k(\Gamma_0(N),\chi)$. For $k \geq 3$, basis of space of Eisenstein series $\mathcal{E}_k$: $G_k(\chi_1,\chi_2,d\tau)$, χ_1, χ_2 primitive, $d\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2) \mid N$, $\chi_1\chi_2 \sim \chi$, with

$$G_k(\chi_1,\chi_2,\tau) = \sum_{N_1|c,\ d}{}' \frac{\chi_1(d)\chi_2(c/N_1)}{(c\tau+d)^k} .$$

For $k = 1$, require χ_1 even; for $k = 2$ and χ trivial, exclude χ_1 trivial, but add $G_2(\tau) - dG_2(d\tau)$ for $d \mid N$.

Up to a multiplicative constant, Fourier expansions of $G_k(\chi_1,\chi_2)$ in large cyclotomic field $\mathbb{Q}(\chi_1,\chi_2)$. Would like in smaller fields, for instance in $\mathbb{Q}$ if χ is trivial or quadratic.

A generalization of Hilbert 90 (vanishing of an H^1) tells us this is possible, and in fact easy: take the trace of $\alpha^j G_k(\chi_1,\chi_2)$ from $\mathbb{Q}(\chi_1,\chi_2)$ to $\mathbb{Q}(\chi)$ for suitable j and restrict to suitable χ_1.

Construction of $M_k(\Gamma_0(N),\chi)$

Of course: $M_k(\Gamma_0(N),\chi) = S_k(\Gamma_0(N),\chi) \oplus \mathcal{E}_k(\Gamma_0(N),\chi)$. For $k \geq 3$, basis of space of Eisenstein series $\mathcal{E}_k$: $G_k(\chi_1,\chi_2,d\tau)$, χ_1, χ_2 primitive, $d\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2) \mid N$, $\chi_1\chi_2 \sim \chi$, with

$$G_k(\chi_1,\chi_2,\tau) = \sum_{N_1|c,\ d}{}' \frac{\chi_1(d)\chi_2(c/N_1)}{(c\tau+d)^k} .$$

For $k = 1$, require χ_1 even; for $k = 2$ and χ trivial, exclude χ_1 trivial, but add $G_2(\tau) - dG_2(d\tau)$ for $d \mid N$.

Up to a multiplicative constant, Fourier expansions of $G_k(\chi_1,\chi_2)$ in large cyclotomic field $\mathbb{Q}(\chi_1,\chi_2)$. Would like in smaller fields, for instance in $\mathbb{Q}$ if χ is trivial or quadratic.

A generalization of Hilbert 90 (vanishing of an H^1) tells us this is possible, and in fact easy: take the trace of $\alpha^j G_k(\chi_1,\chi_2)$ from $\mathbb{Q}(\chi_1,\chi_2)$ to $\mathbb{Q}(\chi)$ for suitable j and restrict to suitable χ_1.

Construction of $M_k(\Gamma_0(N),\chi)$

Of course: $M_k(\Gamma_0(N),\chi) = S_k(\Gamma_0(N),\chi) \oplus \mathcal{E}_k(\Gamma_0(N),\chi)$. For $k \geq 3$, basis of space of Eisenstein series $\mathcal{E}_k$: $G_k(\chi_1,\chi_2,d\tau)$, χ_1, χ_2 primitive, $d\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2) \mid N$, $\chi_1\chi_2 \sim \chi$, with

$$G_k(\chi_1,\chi_2,\tau) = \sum_{N_1|c,\ d}{}' \frac{\chi_1(d)\chi_2(c/N_1)}{(c\tau+d)^k} .$$

For $k = 1$, require χ_1 even; for $k = 2$ and χ trivial, exclude χ_1 trivial, but add $G_2(\tau) - dG_2(d\tau)$ for $d \mid N$.

Up to a multiplicative constant, Fourier expansions of $G_k(\chi_1,\chi_2)$ in large cyclotomic field $\mathbb{Q}(\chi_1,\chi_2)$. Would like in smaller fields, for instance in $\mathbb{Q}$ if χ is trivial or quadratic.

A generalization of Hilbert 90 (vanishing of an H^1) tells us this is possible, and in fact easy: take the trace of $\alpha^j G_k(\chi_1,\chi_2)$ from $\mathbb{Q}(\chi_1,\chi_2)$ to $\mathbb{Q}(\chi)$ for suitable j and restrict to suitable χ_1.

Construction of $M_k(\Gamma_0(N),\chi)$

Of course: $M_k(\Gamma_0(N),\chi) = S_k(\Gamma_0(N),\chi) \oplus \mathcal{E}_k(\Gamma_0(N),\chi)$. For $k \geq 3$, basis of space of Eisenstein series $\mathcal{E}_k$: $G_k(\chi_1,\chi_2,d\tau)$, χ_1, χ_2 primitive, $d\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2) \mid N$, $\chi_1\chi_2 \sim \chi$, with

$$G_k(\chi_1,\chi_2,\tau) = \sum_{N_1|c,\ d}{}' \frac{\chi_1(d)\chi_2(c/N_1)}{(c\tau+d)^k} .$$

For $k = 1$, require χ_1 even; for $k = 2$ and χ trivial, exclude χ_1 trivial, but add $G_2(\tau) - dG_2(d\tau)$ for $d \mid N$.

Up to a multiplicative constant, Fourier expansions of $G_k(\chi_1,\chi_2)$ in large cyclotomic field $\mathbb{Q}(\chi_1,\chi_2)$. Would like in smaller fields, for instance in $\mathbb{Q}$ if χ is trivial or quadratic.

A generalization of Hilbert 90 (vanishing of an H^1) tells us this is possible, and in fact easy: take the trace of $\alpha^j G_k(\chi_1,\chi_2)$ from $\mathbb{Q}(\chi_1,\chi_2)$ to $\mathbb{Q}(\chi)$ for suitable j and restrict to suitable χ_1.

Splitting

Next step: compute eigenforms. Use Hecke algebra to split into eigenspaces. Only linear algebra, but note:

- Hecke operators not sufficient. Example: $S_2^{\mathrm{new}}(\Gamma_0(512))$ cannot be split by $T(n)$ alone (use e.g., $T(3)+T(5)$ in addition).
- Splitting involves factoring polynomials over cyclotomic fields: can be expensive. Work over finite fields essential.

Splitting

Next step: compute eigenforms. Use Hecke algebra to split into eigenspaces. Only linear algebra, but note:

- Hecke operators not sufficient. Example: $S_2^{\mathrm{new}}(\Gamma_0(512))$ cannot be split by $T(n)$ alone (use e.g., $T(3)+T(5)$ in addition).
- Splitting involves factoring polynomials over cyclotomic fields: can be expensive. Work over finite fields essential.

Standard Tasks

Can easily perform:

- Fourier expansions at infinity.
- Action of Hecke operators $T(n)$ and expanding operators $B(d)$.
- Computation of the L-function and special values.
- Linear decomposition of a form on a basis.

But not sufficient, many other tasks need to be done. Before that, computation of more spaces.

Standard Tasks

Can easily perform:

- Fourier expansions at infinity.
- Action of Hecke operators $T(n)$ and expanding operators $B(d)$.
- Computation of the L-function and special values.
- Linear decomposition of a form on a basis.

But not sufficient, many other tasks need to be done. Before that, computation of more spaces.

Modular Forms of Weight 1

Well known much more difficult. At least three methods:

- Construction of explicit weight 1 forms using Hecke characters. Easy, but insufficient.
- Using two "coprime" forms, i.e., having no common zero. One can then divide a higher weight form by those two forms and look for intersection. Dimension must be small, so need weight 1 forms. two reasonable choices: (E_3, E_4), Eisenstein series of weight 1 and characters $(-3/n)$, $(-4/n)$: level 12 unfortunately. Or $(\theta^2, \theta^2(\tau))$ (θ standard theta function), level 8. If level coprime to 6, very expensive.
- G. Schaeffer's Hecke stability method: if finite dimensional space of meromorphic forms stable by any $T(n)$ with $(n, N) = 1$, all forms holomorphic. Best method, but linear algebra can be very expensive so again finite fields.

Modular Forms of Weight 1

Well known much more difficult. At least three methods:

- Construction of explicit weight 1 forms using Hecke characters. Easy, but insufficient.
- Using two "coprime" forms, i.e., having no common zero. One can then divide a higher weight form by those two forms and look for intersection. Dimension must be small, so need weight 1 forms. two reasonable choices: (E_3, E_4), Eisenstein series of weight 1 and characters $(-3/n)$, $(-4/n)$: level 12 unfortunately. Or $(\theta^2, \theta^2(\tau))$ (θ standard theta function), level 8. If level coprime to 6, very expensive.
- G. Schaeffer's Hecke stability method: if finite dimensional space of meromorphic forms stable by any $T(n)$ with $(n, N) = 1$, all forms holomorphic. Best method, but linear algebra can be very expensive so again finite fields.

Modular Forms of Weight 1

Well known much more difficult. At least three methods:

- Construction of explicit weight 1 forms using Hecke characters. Easy, but insufficient.
- Using two "coprime" forms, i.e., having no common zero. One can then divide a higher weight form by those two forms and look for intersection. Dimension must be small, so need weight 1 forms. two reasonable choices: (E_3, E_4), Eisenstein series of weight 1 and characters $(-3/n)$, $(-4/n)$: level 12 unfortunately. Or $(\theta^2, \theta^2(\tau))$ (θ standard theta function), level 8. If level coprime to 6, very expensive.
- G. Schaeffer's Hecke stability method: if finite dimensional space of meromorphic forms stable by any $T(n)$ with $(n, N) = 1$, all forms holomorphic. Best method, but linear algebra can be very expensive so again finite fields.

Modular Forms of 1/2-Integral Weight

- For weight $1/2$: explicit description by the Serre–Stark theorem.
- For higher weight: the use of the coprime forms $\theta(\tau)$ and $\theta(2\tau)$ is here the best since level of a form of half integral weight always multiple of 4, so at worst double the level.
- Use of a theorem of J. Oesterlé and the author to control the dimension.

Eisenstein series: on $\Gamma_0(4N)$ known if N is squarefree. General case should be easy.

Modular Forms of 1/2-Integral Weight

- For weight $1/2$: explicit description by the Serre–Stark theorem.
- For higher weight: the use of the coprime forms $\theta(\tau)$ and $\theta(2\tau)$ is here the best since level of a form of half integral weight always multiple of 4, so at worst double the level.
- Use of a theorem of J. Oesterlé and the author to control the dimension.

Eisenstein series: on $\Gamma_0(4N)$ known if N is squarefree. General case should be easy.

Modular Forms of 1/2-Integral Weight

- For weight $1/2$: explicit description by the Serre–Stark theorem.
- For higher weight: the use of the coprime forms $\theta(\tau)$ and $\theta(2\tau)$ is here the best since level of a form of half integral weight always multiple of 4, so at worst double the level.
- Use of a theorem of J. Oesterlé and the author to control the dimension.

Eisenstein series: on $\Gamma_0(4N)$ known if N is squarefree. General case should be easy.

Modular Forms of 1/2-Integral Weight

- For weight $1/2$: explicit description by the Serre–Stark theorem.
- For higher weight: the use of the coprime forms $\theta(\tau)$ and $\theta(2\tau)$ is here the best since level of a form of half integral weight always multiple of 4, so at worst double the level.
- Use of a theorem of J. Oesterlé and the author to control the dimension.

Eisenstein series: on $\Gamma_0(4N)$ known if N is squarefree. General case should be easy.

Using Products of Eisenstein Series I

A number of desired operations on modular forms cannot be done using the above, except in special cases (e.g., squarefree level), most notably the Fourier expansions at all cusps. We now use an alternative method to generate modular form spaces, based on a generalization of a theorem of L. Borisov and P. Gunnells.

Recall the Eisenstein series $G_k(\chi_1, \chi_2)$ and their normalized counterparts $E_k(\chi_1, \chi_2)$. Clearly

$$E_\ell(\chi_1, \chi_2)(e\tau)E_{k-\ell}(\chi_3, \chi_4)(f\tau) \in M_k(\Gamma_0(N), \chi)$$

for $N = \mathrm{lcm}(\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2)e, \mathfrak{f}(\chi_3)\mathfrak{f}(\chi_4)f)$ and $\chi = \chi_1\chi_2\chi_3\chi_4$ (defining $E_0 = 1$ and usual modification in weight 2). Can hope they linearly generate $M_k(\Gamma_0(N), \chi)$.

Using Products of Eisenstein Series I

A number of desired operations on modular forms cannot be done using the above, except in special cases (e.g., squarefree level), most notably the Fourier expansions at all cusps. We now use an alternative method to generate modular form spaces, based on a generalization of a theorem of L. Borisov and P. Gunnells.

Recall the Eisenstein series $G_k(\chi_1, \chi_2)$ and their normalized counterparts $E_k(\chi_1, \chi_2)$. Clearly

$$E_\ell(\chi_1, \chi_2)(e\tau)E_{k-\ell}(\chi_3, \chi_4)(f\tau) \in M_k(\Gamma_0(N), \chi)$$

for $N = \mathrm{lcm}(\mathfrak{f}(\chi_1)\mathfrak{f}(\chi_2)e, \mathfrak{f}(\chi_3)\mathfrak{f}(\chi_4)f)$ and $\chi = \chi_1\chi_2\chi_3\chi_4$ (defining $E_0 = 1$ and usual modification in weight 2). Can hope they linearly generate $M_k(\Gamma_0(N), \chi)$.

Using Products of Eisenstein Series II

The theorem says that this is the case if the (integral) weight $k \geq 3$, and also in weight 2 under a suitable condition. Can avoid this case simply by multiplying.
Using products of Eisenstein series as a basis is heavy because of the characters χ_i (large cyclotomic fields), but has enormous advantages since Eisenstein series are completely explicit.

By a very tedious but straightforward computation we can compute $E|_k\gamma$ for any Eisenstein series E and any $\gamma \in M_2^+(\mathbb{Q})$.
This has numerous very useful consequences:

Using Products of Eisenstein Series II

The theorem says that this is the case if the (integral) weight $k \geq 3$, and also in weight 2 under a suitable condition. Can avoid this case simply by multiplying.
Using products of Eisenstein series as a basis is heavy because of the characters χ_i (large cyclotomic fields), but has enormous advantages since Eisenstein series are completely explicit.

By a very tedious but straightforward computation we can compute $E|_k\gamma$ for any Eisenstein series E and any $\gamma \in M_2^+(\mathbb{Q})$.
This has numerous very useful consequences:

Using Products of Eisenstein Series III

- Compute the Fourier expansion of $F|_k\gamma$ for any modular form F and any $\gamma \in M_2^+(\mathbb{Q})$, in particular expansions and valuations at all cusps, action of Atkin–Lehner operators W_Q.
- Computation of all period polynomials

$$\int_{s_1}^{s_2} F|_k(\gamma)(\tau)(X-\tau)^{k-2}\,d\tau$$

for any cusps s_1, s_2 by using expansions at all cusps and Yu. Manin's decomposition of modular symbols as a sum of Manin symbols.
- Computation of Petersson scalar products: can express them as finite bilinear combinations of period polynomials for suitable s_1, s_2, and γ.
- Numerical evaluation of a form: can show that to compute $F(\tau)$, using a suitable γ and the expansion of $F|_k\gamma$, can always reduce to the case $\Im(\tau) \geq 1/(2N)$, almost optimal.

Using Products of Eisenstein Series III

- Compute the Fourier expansion of $F|_k\gamma$ for any modular form F and any $\gamma \in M_2^+(\mathbb{Q})$, in particular expansions and valuations at all cusps, action of Atkin–Lehner operators W_Q.
- Computation of all period polynomials

$$\int_{s_1}^{s_2} F|_k(\gamma)(\tau)(X-\tau)^{k-2}\,d\tau$$

for any cusps s_1, s_2 by using expansions at all cusps and Yu. Manin's decomposition of modular symbols as a sum of Manin symbols.

- Computation of Petersson scalar products: can express them as finite bilinear combinations of period polynomials for suitable s_1, s_2, and γ.
- Numerical evaluation of a form: can show that to compute $F(\tau)$, using a suitable γ and the expansion of $F|_k\gamma$, can always reduce to the case $\Im(\tau) \geq 1/(2N)$, almost optimal.

Using Products of Eisenstein Series III

- Compute the Fourier expansion of $F|_k\gamma$ for any modular form F and any $\gamma \in M_2^+(\mathbb{Q})$, in particular expansions and valuations at all cusps, action of Atkin–Lehner operators W_Q.
- Computation of all period polynomials

$$\int_{s_1}^{s_2} F|_k(\gamma)(\tau)(X-\tau)^{k-2}\,d\tau$$

  for any cusps s_1, s_2 by using expansions at all cusps and Yu. Manin's decomposition of modular symbols as a sum of Manin symbols.
- Computation of Petersson scalar products: can express them as finite bilinear combinations of period polynomials for suitable s_1, s_2, and γ.
- Numerical evaluation of a form: can show that to compute $F(\tau)$, using a suitable γ and the expansion of $F|_k\gamma$, can always reduce to the case $\Im(\tau) \geq 1/(2N)$, almost optimal.

Using Products of Eisenstein Series III

- Compute the Fourier expansion of $F|_k\gamma$ for any modular form F and any $\gamma \in M_2^+(\mathbb{Q})$, in particular expansions and valuations at all cusps, action of Atkin–Lehner operators W_Q.
- Computation of all period polynomials

$$\int_{s_1}^{s_2} F|_k(\gamma)(\tau)(X-\tau)^{k-2}\,d\tau$$

  for any cusps s_1, s_2 by using expansions at all cusps and Yu. Manin's decomposition of modular symbols as a sum of Manin symbols.
- Computation of Petersson scalar products: can express them as finite bilinear combinations of period polynomials for suitable s_1, s_2, and γ.
- Numerical evaluation of a form: can show that to compute $F(\tau)$, using a suitable γ and the expansion of $F|_k\gamma$, can always reduce to the case $\Im(\tau) \geq 1/(2N)$, almost optimal.

Part II: Implementation Details

Must explain in particular how to represent modular forms and Dirichlet characters, functions acting on them, and functions on spaces of modular forms

In the modular symbol context, modular forms are represented by modular symbols, which are finite objects. In the trace formula context, the situation is different. We cannot represent them by their Fourier expansion at infinity since this is an infinite object. We could use Taylor expansions, which can be encoded with a finite amount of data, but this is difficult in practice (not in theory) for groups of large level. We have chosen to do as follows:

Part II: Implementation Details

Must explain in particular how to represent modular forms and Dirichlet characters, functions acting on them, and functions on spaces of modular forms

In the modular symbol context, modular forms are represented by modular symbols, which are finite objects. In the trace formula context, the situation is different. We cannot represent them by their Fourier expansion at infinity since this is an infinite object. We could use Taylor expansions, which can be encoded with a finite amount of data, but this is difficult in practice (not in theory) for groups of large level. We have chosen to do as follows:

Representation of Modular Forms I

A modular form will be represented as an mfclosure (our terminology), which is a `Pari/GP` object on which essentially a unique function mfcoefs can be applied, as well as functions derived from it: `mfcoefs(F,n)` gives the vector of Fourier coefficients $[a(0), a(1), \ldots, a(n)]$. To have a power series expansion, simply use `Ser(,q)`; for example `Ser(mfcoefs(mfdelta(),5),q)` returns

$$q - 24q^2 + 252q^3 - 1472q^4 + 4830q^5 + O(q^6)$$

To simplify notation we define the user function

```
mfser(F,n)=Ser(mfcoefs(F,n),q)
```

Representation of Modular Forms I

A modular form will be represented as an mfclosure (our terminology), which is a `Pari/GP` object on which essentially a unique function mfcoefs can be applied, as well as functions derived from it: `mfcoefs(F,n)` gives the vector of Fourier coefficients $[a(0), a(1), \ldots, a(n)]$. To have a power series expansion, simply use `Ser(,q)`; for example `Ser(mfcoefs(mfdelta(),5),q)` returns

$$q - 24q^2 + 252q^3 - 1472q^4 + 4830q^5 + O(q^6)$$

To simplify notation we define the user function `mfser(F,n)=Ser(mfcoefs(F,n),q)`

Representation of Modular Forms II

Advantage of this representation: no need to specify in advance the desired number of Fourier coefficients.
All the usual arithmetic and modular form operations on mfclosures are of course available.

Note that the internal representation will be in direct Polish notation. For instance, the modular form $E_4(\Delta^2 + E_{24})$ is represented (with evident notation) by

```
Mul(E(4),Add(Mul(Delta(),Delta()),E(24)))
```

The function `mfdescribe` essentially outputs the above.

Representation of Modular Forms II

Advantage of this representation: no need to specify in advance the desired number of Fourier coefficients.
All the usual arithmetic and modular form operations on mfclosures are of course available.

Note that the internal representation will be in direct Polish notation. For instance, the modular form $E_4(\Delta^2 + E_{24})$ is represented (with evident notation) by

```
Mul(E(4),Add(Mul(Delta(),Delta()),E(24)))
```

The function `mfdescribe` essentially outputs the above.

Representation of Modular Forms II

Advantage of this representation: no need to specify in advance the desired number of Fourier coefficients.
All the usual arithmetic and modular form operations on mfclosures are of course available.

Note that the internal representation will be in direct Polish notation. For instance, the modular form $E_4(\Delta^2 + E_{24})$ is represented (with evident notation) by

```
Mul(E(4),Add(Mul(Delta(),Delta()),E(24)))
```

The function `mfdescribe` essentially outputs the above.

Representation of Modular Forms III

Thus, in the language of combinatorics, mfclosures can be leaves (modular forms constructed from scratch or from standard mathematical objects), such as Eisenstein series, the Δ and j function, modular forms attached to elliptic curves, quadratic forms (theta functions), eta quotients, etc... , or obtained by unary operations such as Hecke, Atkin–Lehner, diamond, and twisting operators, derivation/integration, and Serre derivation, or binary operations such as multiplication/division (more generally linear combinations), RC-brackets, etc...

In addition, a number of functions operate on mfclosures: we have seen `mfcoefs` (and `mfcoef` which gives a single coefficient), but also `mfeval` (evaluation at a point in $\mathbb{H}$), `mfslashexpansion` (expansion of $F|_k\gamma$), `lfunmf` (L-function attached to a modular form), `mftaylor` (Taylor expansion, for now only on the full modular group at the point $\tau_0 = i$), etc...

Representation of Modular Forms III

Thus, in the language of combinatorics, mfclosures can be leaves (modular forms constructed from scratch or from standard mathematical objects), such as Eisenstein series, the Δ and j function, modular forms attached to elliptic curves, quadratic forms (theta functions), eta quotients, etc... , or obtained by unary operations such as Hecke, Atkin–Lehner, diamond, and twisting operators, derivation/integration, and Serre derivation, or binary operations such as multiplication/division (more generally linear combinations), RC-brackets, etc...

In addition, a number of functions operate on mfclosures: we have seen `mfcoefs` (and `mfcoef` which gives a single coefficient), but also `mfeval` (evaluation at a point in $\mathbb{H}$), `mfslashexpansion` (expansion of $F|_k\gamma$), `lfunmf` (L-function attached to a modular form), `mftaylor` (Taylor expansion, for now only on the full modular group at the point $\tau_0 = i$), etc...

Representation of Modular Forms III

Thus, in the language of combinatorics, mfclosures can be leaves (modular forms constructed from scratch or from standard mathematical objects), such as Eisenstein series, the Δ and j function, modular forms attached to elliptic curves, quadratic forms (theta functions), eta quotients, etc... , or obtained by unary operations such as Hecke, Atkin–Lehner, diamond, and twisting operators, derivation/integration, and Serre derivation, or binary operations such as multiplication/division (more generally linear combinations), RC-brackets, etc...

In addition, a number of functions operate on mfclosures: we have seen `mfcoefs` (and `mfcoef` which gives a single coefficient), but also `mfeval` (evaluation at a point in $\mathbb{H}$), `mfslashexpansion` (expansion of $F|_k\gamma$), `lfunmf` (L-function attached to a modular form), `mftaylor` (Taylor expansion, for now only on the full modular group at the point $\tau_0 = i$), etc...

Dirichlet Characters

This is technical, will skip details: a Dirichlet character can be omitted (by default trivial character), an integer congruent to 0 or 1 mod 4 representing a discriminant (e.g., -23, 37), or an element of $(\mathbb{Z}/N\mathbb{Z})^x$ such as `Mod(61,633)`, corresponding to a semi-canonical isomorphism with its group of characters introduced by B. Conrey.

All the usual operations on Dirichlet characters are available (evaluation, induction, primitive character associated to it, multiplication, etc...)

Dirichlet Characters

This is technical, will skip details: a Dirichlet character can be omitted (by default trivial character), an integer congruent to 0 or 1 mod 4 representing a discriminant (e.g., -23, 37), or an element of $(\mathbb{Z}/N\mathbb{Z})^x$ such as `Mod(61,633)`, corresponding to a semi-canonical isomorphism with its group of characters introduced by B. Conrey.

All the usual operations on Dirichlet characters are available (evaluation, induction, primitive character associated to it, multiplication, etc...)

Functions on Spaces I

The basic initialization command for spaces of modular forms is `mfinit`: specify level, weight, character, and which space (new, cuspidal, old, or full). Dimension: `mfdim`. This gives a basis (including in weight 1) which is essentially random. To get the corresponding elements of the basis as mfclosures, use `mfbasis`. Matrix of Hecke or Atkin–Lehner operators `mfheckemat`, `mfatkininit` (which gives a little more than the matrix).

Second basic initialization command: to obtain the eigenforms use `mfeigenbasis`. Can also specify bound on the degree of the "field of definition" using e.g., `mfsplit(mf,1)` computes the rational eigenforms. To get the polynomials defining the fields, use `mffields`.

Functions on Spaces I

The basic initialization command for spaces of modular forms is `mfinit`: specify level, weight, character, and which space (new, cuspidal, old, or full). Dimension: `mfdim`. This gives a basis (including in weight 1) which is essentially random. To get the corresponding elements of the basis as mfclosures, use `mfbasis`. Matrix of Hecke or Atkin–Lehner operators `mfheckemat`, `mfatkininit` (which gives a little more than the matrix).

Second basic initialization command: to obtain the eigenforms use `mfeigenbasis`. Can also specify bound on the degree of the "field of definition" using e.g., `mfsplit(mf,1)` computes the rational eigenforms. To get the polynomials defining the fields, use `mffields`.

Functions on Spaces II

Examples:

? `mf=mfinit([26,2],0);`

Creates a basis of the new space of level 26 weight 2, no character.

? `L=mfbasis(mf);vector(#L,i,mfser(L[i],10))` get the corresponding mfclosures; there are two:

$[2q - 2q^3 + 2q^4 - 4q^5 - ..., -2q - 4q^2 + 10q^3 - 2q^4 + ...]$

These are of course not the eigenforms. To get them we do:

? `LE=mfeigenbasis(mf);vector(#LE,i,mfser(LE[i],10))`

$[q - q^2 + q^3 + q^4 - 3q^5 - ..., q + q^2 - 3q^3 + q^4 - q^5 - ...]$

Functions on Spaces II

Examples:

? `mf=mfinit([26,2],0);`

Creates a basis of the new space of level 26 weight 2, no character.

? `L=mfbasis(mf);vector(#L,i,mfser(L[i],10))` get the corresponding mfclosures; there are two:

$[2q - 2q^3 + 2q^4 - 4q^5 - ..., -2q - 4q^2 + 10q^3 - 2q^4 + ...]$

These are of course not the eigenforms. To get them we do:

? `LE=mfeigenbasis(mf);vector(#LE,i,mfser(LE[i],10))`

$[q - q^2 + q^3 + q^4 - 3q^5 - ..., q + q^2 - 3q^3 + q^4 - q^5 - ...]$

Functions on Spaces II

Examples:

? `mf=mfinit([26,2],0);`

Creates a basis of the new space of level 26 weight 2, no character.

? `L=mfbasis(mf);vector(#L,i,mfser(L[i],10))` get the corresponding mfclosures; there are two:

$[2q - 2q^3 + 2q^4 - 4q^5 - ..., -2q - 4q^2 + 10q^3 - 2q^4 + ...]$

These are of course not the eigenforms. To get them we do:

? `LE=mfeigenbasis(mf);vector(#LE,i,mfser(LE[i],10))`

$[q - q^2 + q^3 + q^4 - 3q^5 - ..., q + q^2 - 3q^3 + q^4 - q^5 - ...]$

Functions on Spaces III

Eigenforms can be over a number field:

```
? mf=mfinit([23,2]);LE=mfeigenbasis(mf);
? vector(#LE,i,mfser(LE[i],10))
```

$[Mod(1, y^2 + y - 1)q + Mod(y, y^2 + y - 1)q^2 + ...]$

To see it better:

```
[mffields(mf),vector(#LE,i,lift(mfser(LE[i],10)))]
```

$[[y^2 + y - 1], [q + yq^2 + (-2y - 1)q^3 + (-y - 1)q^4 + 2yq^5 ...]]$

Functions on Spaces III

Eigenforms can be over a number field:

? `mf=mfinit([23,2]);LE=mfeigenbasis(mf);`

? `vector(#LE,i,mfser(LE[i],10))`

$[Mod(1, y^2+y-1)q + Mod(y, y^2+y-1)q^2 + ...]$

To see it better:

`[mffields(mf),vector(#LE,i,lift(mfser(LE[i],10)))]`

$[[y^2+y-1], [q + yq^2 + (-2y-1)q^3 + (-y-1)q^4 + 2yq^5...]]$

Functions on Spaces IV

When nonquadratic characters occur, can get complicated output, unavoidable,

```
mf=mfinit([15,3,Mod(2,5)],0);mffields(mf)
```

$[y^2 + Mod(-3t, t^2 + 1)]$

Quadratic extension of quadratic extension.

```
F=mfser(mfeigenbasis(mf)[1],10);f=liftall(F)
```

$q + (y - t - 1)q^2 + tyq^3 + ((-2t - 2)y + t)q^4 + ...$

Can get expansion over an absolute instead of relative number field if desired:

```
[T,a]=rnfequation(nfinit(t^2+1),y^2-3*t,1);subst(f,t,a)
```

$Mod(1, y^4 + 9)q + Mod(-1/3y^2 + y - 1, y^4 + 9)q^2 + ...$

Functions on Spaces IV

When nonquadratic characters occur, can get complicated output, unavoidable,

```
mf=mfinit([15,3,Mod(2,5)],0);mffields(mf)
```

$[y^2 + Mod(-3t, t^2 + 1)]$

Quadratic extension of quadratic extension.

```
F=mfser(mfeigenbasis(mf)[1],10);f=liftall(F)
```

$q + (y - t - 1)q^2 + tyq^3 + ((-2t - 2)y + t)q^4 + ...$

Can get expansion over an absolute instead of relative number field if desired:

```
[T,a]=rnfequation(nfinit(t^2+1),y^2-3*t,1);subst(f,t,a)
```

$Mod(1, y^4 + 9)q + Mod(-1/3y^2 + y - 1, y^4 + 9)q^2 + ...$

Functions on Spaces V

Can ask only rational spaces:

```
mf=mfinit([35,2],0);mffields(mf)
```

$[y, y^2 + y - 4]$

```
C=mfsplit(mf,1)[1][1];mfser(mflinear(mfbasis(mf),C),10)
```

$[q + q^3 - 2q^4 - q^5 + ...]$

This is in fact used by the function `mfsearch`:

```
S=mfsearch([40,2],[[2,1],[3,-2]])
```

[[34, [Vecsmall(...)]]]

This tells us that there is only one rational eigenform in weight 2 with $N \leq 40$, $a_2 = 1$, $a_3 = -2$, and it is in $S_2(\Gamma_0(34))$, and the beginning of its Fourier expansion is given by

```
mfser(S[1][2],10)
```

$q + q^2 - 2q^3 + q^4 - 2q^6 - 4q^7 + ...$

Note the advantage of using mfclosures: if we want the Fourier expansion to 100 terms, say, simply type `mfser(S[1][2],100)`.

Functions on Spaces V

Can ask only rational spaces:

```
mf=mfinit([35,2],0);mffields(mf)
```

$[y, y^2 + y - 4]$

```
C=mfsplit(mf,1)[1][1];mfser(mflinear(mfbasis(mf),C),10)
```

$[q + q^3 - 2q^4 - q^5 + ...]$

This is in fact used by the function `mfsearch`:

```
S=mfsearch([40,2],[[2,1],[3,-2]])
```

[[34, [Vecsmall(...)]]]

This tells us that there is only one rational eigenform in weight 2 with $N \leq 40$, $a_2 = 1$, $a_3 = -2$, and it is in $S_2(\Gamma_0(34))$, and the beginning of its Fourier expansion is given by

```
mfser(S[1][2],10)
```

$q + q^2 - 2q^3 + q^4 - 2q^6 - 4q^7 + ...$

Note the advantage of using mfclosures: if we want the Fourier expansion to 100 terms, say, simply type `mfser(S[1][2],100)`.

Functions on Spaces V

Can ask only rational spaces:

```
mf=mfinit([35,2],0);mffields(mf)
```

$[y, y^2 + y - 4]$

```
C=mfsplit(mf,1)[1][1];mfser(mflinear(mfbasis(mf),C),10)
```

$[q + q^3 - 2q^4 - q^5 + ...]$

This is in fact used by the function `mfsearch`:

```
S=mfsearch([40,2],[[2,1],[3,-2]])
```

[[34, [Vecsmall(...)]]]

This tells us that there is only one rational eigenform in weight 2 with $N \leq 40$, $a_2 = 1$, $a_3 = -2$, and it is in $S_2(\Gamma_0(34))$, and the beginning of its Fourier expansion is given by

```
mfser(S[1][2],10)
```

$q + q^2 - 2q^3 + q^4 - 2q^6 - 4q^7 + ...$

Note the advantage of using mfclosures: if we want the Fourier expansion to 100 terms, say, simply type `mfser(S[1][2],100)`.

Part III: Further examples I

```
mf=mfinit([96,4],0);M=mfheckemat(mf,5)
```

$[0, 64, 0, 0, -84, 0; 0, 4, 0, 36, 0, 0; 1, 0, -24/5, 0, 0, 294/5; 0, 2, 0, -12,$

```
factor(charpoly(M))
```

$[x - 10, 2; x - 2, 2; x + 14, 2]$

```
M=mfatkininit(mf,3)[2..3]
```

$[[0, -3, 0, 0, -24, 0; -1/3, 0, -4/3, 0, ...], 1]$

```
factor(charpoly(M[1]))
```

$[x - 1, 3; x + 1, 3]$

Part III: Further examples I

```
mf=mfinit([96,4],0);M=mfheckemat(mf,5)
```

$[0, 64, 0, 0, -84, 0; 0, 4, 0, 36, 0, 0; 1, 0, -24/5, 0, 0, 294/5; 0, 2, 0, -12,$

```
factor(charpoly(M))
```

$[x - 10, 2; x - 2, 2; x + 14, 2]$

```
M=mfatkininit(mf,3)[2..3]
```

$[[0, -3, 0, 0, -24, 0; -1/3, 0, -4/3, 0, ...], 1]$

```
factor(charpoly(M[1]))
```

$[x - 1, 3; x + 1, 3]$

Part III: Further examples I

```
mf=mfinit([96,4],0);M=mfheckemat(mf,5)
```

$[0, 64, 0, 0, -84, 0; 0, 4, 0, 36, 0, 0; 1, 0, -24/5, 0, 0, 294/5; 0, 2, 0, -12,$

```
factor(charpoly(M))
```

$[x - 10, 2; x - 2, 2; x + 14, 2]$

```
M=mfatkininit(mf,3)[2..3]
```

$[[0, -3, 0, 0, -24, 0; -1/3, 0, -4/3, 0, ...], 1]$

```
factor(charpoly(M[1]))
```

$[x - 1, 3; x + 1, 3]$

Part III: Further examples I

```
mf=mfinit([96,4],0);M=mfheckemat(mf,5)
```

$[0, 64, 0, 0, -84, 0; 0, 4, 0, 36, 0, 0; 1, 0, -24/5, 0, 0, 294/5; 0, 2, 0, -12,$

```
factor(charpoly(M))
```

$[x - 10, 2; x - 2, 2; x + 14, 2]$

```
M=mfatkininit(mf,3)[2..3]
```

$[[0, -3, 0, 0, -24, 0; -1/3, 0, -4/3, 0, ...], 1]$

```
factor(charpoly(M[1]))
```

$[x - 1, 3; x + 1, 3]$

Further examples II

Linear decompositions:

```
Th=mfTheta();
mf=mfinit([4,2]);mftobasis(mf,mfpow(Th,4))
```

$[0, 8]$,

```
mf=mfinit([4,5,-4]);mftobasis(mf,mfpow(Th,10))
```

$[64/5, 4/5, 32/5]$

Further examples II

Linear decompositions:

```
Th=mfTheta();
mf=mfinit([4,2]);mftobasis(mf,mfpow(Th,4))
```

$[0, 8]$,

```
mf=mfinit([4,5,-4]);mftobasis(mf,mfpow(Th,10))
```

$[64/5, 4/5, 32/5]$

Further examples III

Eisenstein series:

```
B=mfbasis([4,3,-4],3);
[mfser(E,10) | E <- B]
```

$[q + 4q^2 + 8q^3 + 16q^4 + 26q^5 + ..., -1/4 + q + q^2 - 8q^3 + q^4 + 26q^5 - ...]$

Numerical modular form functions such as evaluation at some point in the upper-half plane, computation of L-function values.

```
E4=mfEk(4);mf=mfinit(E4);mfeval(mf,E4,I)
1.4557628922687093224624220035988 69..
3*gamma(1/4)^8/(2*Pi)^6
1.4557628922687093224624220035988 69..
```

This is a consequence of complex multiplication.

Further examples III

Eisenstein series:

```
B=mfbasis([4,3,-4],3);
[mfser(E,10) | E <- B]
```

$[q + 4q^2 + 8q^3 + 16q^4 + 26q^5 + ..., -1/4 + q + q^2 - 8q^3 + q^4 + 26q^5 - ...]$

Numerical modular form functions such as evaluation at some point in the upper-half plane, computation of L-function values.

```
E4=mfEk(4);mf=mfinit(E4);mfeval(mf,E4,I)
```

1.455762892268709322462422003598869..

```
3*gamma(1/4)^8/(2*Pi)^6
```

1.455762892268709322462422003598869..

This is a consequence of complex multiplication.

Further examples IV

Special values:

```
D=mfDelta(); mf=mfinit(D); L = lfunmf(mf,D);
lfunmfspec(L);
```

outputs

$$[[1, 25/48, 5/12, 25/48, 1], [1620/691, 1, 9/14, 9/14, 1, 1620/691],$$
$$0.00741542092989613058900642774590022872 48,$$
$$0.00508351210839328686049429013743874732 26]$$

Plotting and finding zeros: (this is more properly part of the L-function package):

```
LF = lfuninit(L,[50]); ploth(t=0,50,lfunhardy(LF,t));
```

In 50 ms, this outputs:

Further examples IV

Special values:

```
D=mfDelta(); mf=mfinit(D); L = lfunmf(mf,D);
lfunmfspec(L);
```

outputs

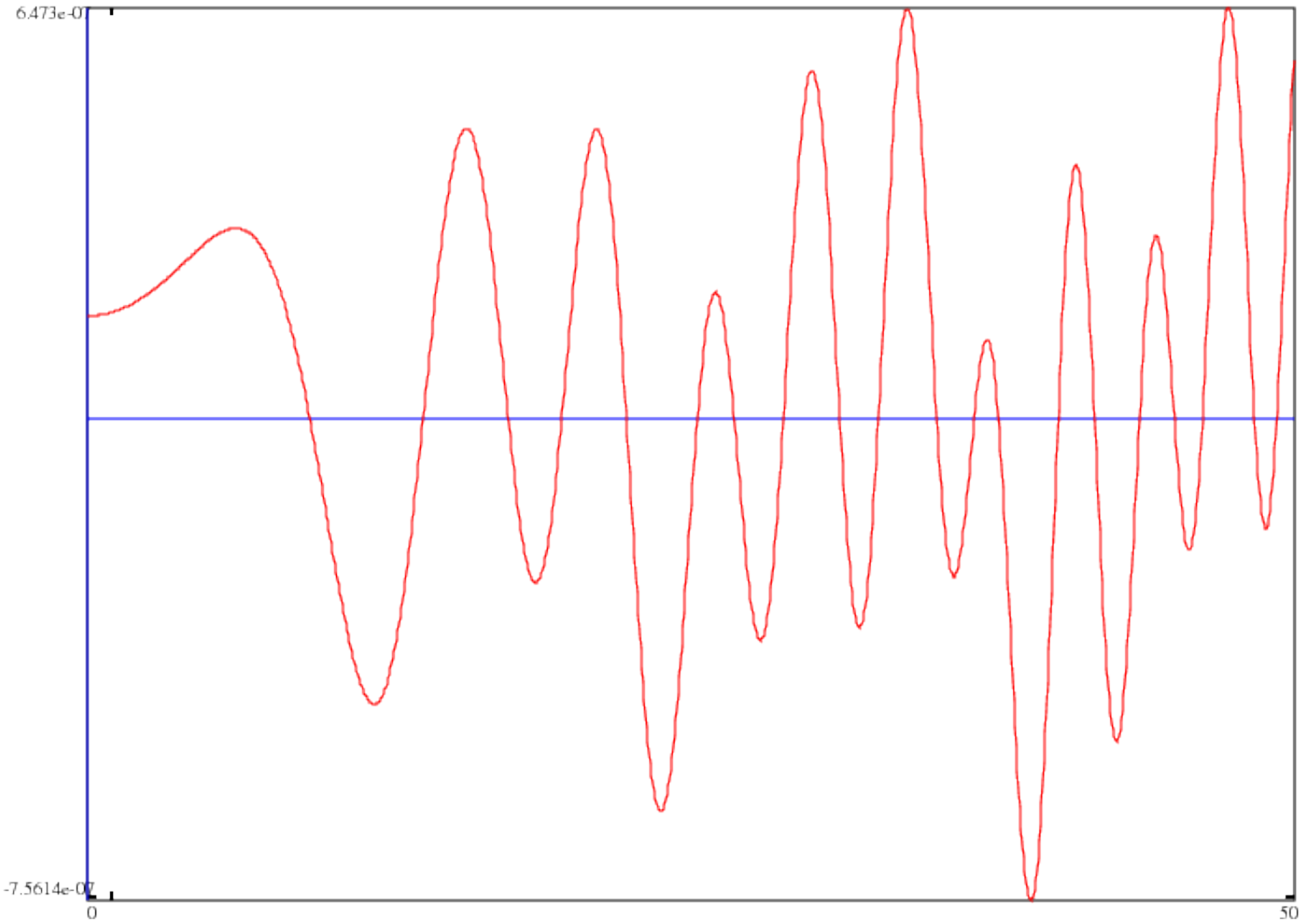
$$[[1, 25/48, 5/12, 25/48, 1], [1620/691, 1, 9/14, 9/14, 1, 1620/691],$$
$$0.00741542092989613058900642774590022872 48,$$
$$0.00508351210839328686049429013743874732 26]$$

Plotting and finding zeros: (this is more properly part of the L-function package):

```
LF = lfuninit(L,[50]); ploth(t=0,50,lfunhardy(LF,t));
```

In 50 ms, this outputs:

Plot



Further examples V

In 10 ms `lfunzeros(LF,20)` outputs

[9.22237939992110252224376719274347813 55,
13.907549861392134406446681328770219492,
17.442776978234473313551525137127262719,
19.656513141954961000127281756321302802]

Note that the `lfunmf` function is completely general and computes the L-function attached to any modular form, eigenform or not.

Further examples V

In 10 ms `lfunzeros(LF,20)` outputs

[9.22237939992110252224376719274347813 55,
13.907549861392134406446681328770219492,
17.442776978234473313551525137127262719,
19.656513141954961000127281756321302802]

Note that the `lfunmf` function is completely general and computes the L-function attached to any modular form, eigenform or not.

Further examples VI

Searching for modular eigenforms with given coefficients:

```
L=mfsearch([60,2],[[2,-1],[3,-3]])
```

[[53, [Vecsmall...]], [58, [Vecsmall...]]]

```
[mfser(L[1][2],10),mfser(L[2][2],10)]
```

$[q - q^2 - 3q^3 - q^4 + 3q^6 - 4q^7 + ..., q - q^2 - 3q^3 + q^4 - 3q^5 + 3q^6 - 2q^7 ...]$

Weight 1 Examples I

```
L=mfchargalois(148,,-1);
```

Finds all Galois equivalence classes of odd characters.

```
[mfdim([148,1,chi],1) | chi <- L]
```

$[0, 0, 0, 0, 1, 0, 1, 1, 0]$

Three characters for which there exists a form of weight 1.

```
mf=mfinit([148,1,L[7]],0);
```

```
mfser(mfeigenbasis(mf)[1],10)
```

$Mod(1, t^2+1)q + Mod(-t, t^2+1)q^3 + Mod(-1, t^2+1)q^7 + \ldots$

```
mfgaloistype(mf)
```

$[-24]$ Eigenform of type S_4 (the lowest possible level).

```
mfgaloistype([675,1,Mod(161,675)])
```

$[-60]$ Eigenform of type A_5 (the lowest possible level is 633).

Takes only 4 seconds to create the space and to find the form.

Weight 1 Examples I

```
L=mfchargalois(148,,-1);
```

Finds all Galois equivalence classes of odd characters.

```
[mfdim([148,1,chi],1) | chi <- L]
```

$[0, 0, 0, 0, 1, 0, 1, 1, 0]$

Three characters for which there exists a form of weight 1.

```
mf=mfinit([148,1,L[7]],0);

mfser(mfeigenbasis(mf)[1],10)
```

$Mod(1, t^2+1)q + Mod(-t, t^2+1)q^3 + Mod(-1, t^2+1)q^7 + \ldots$

```
mfgaloistype(mf)
```

$[-24]$ Eigenform of type S_4 (the lowest possible level).

```
mfgaloistype([675,1,Mod(161,675)])
```

$[-60]$ Eigenform of type A_5 (the lowest possible level is 633).

Takes only 4 seconds to create the space and to find the form.

Weight 1 Examples I

```
L=mfchargalois(148,,-1);
```

Finds all Galois equivalence classes of odd characters.

```
[mfdim([148,1,chi],1) | chi <- L]
```

$[0, 0, 0, 0, 1, 0, 1, 1, 0]$

Three characters for which there exists a form of weight 1.

```
mf=mfinit([148,1,L[7]],0);
```

```
mfser(mfeigenbasis(mf)[1],10)
```

$Mod(1, t^2+1)q + Mod(-t, t^2+1)q^3 + Mod(-1, t^2+1)q^7 + \ldots$

```
mfgaloistype(mf)
```

$[-24]$ Eigenform of type S_4 (the lowest possible level).

```
mfgaloistype([675,1,Mod(161,675)])
```

$[-60]$ Eigenform of type A_5 (the lowest possible level is 633).

Takes only 4 seconds to create the space and to find the form.

Weight 1 Examples II

More typical examples:

```
mf=mfinit([148,1,L[5]],0); mfgaloistype(mf)
```

Vecsmall([18]) This eigenform is of type D_{18}.

```
mfgaloistype([239,1,-239])
```

Vecsmall([6, 10, 30])

Here there are three eigenforms, of type D_n with $n = 6$, 10, and 30 respectively.

Weight 1 Examples II

More typical examples:

```
mf=mfinit([148,1,L[5]],0); mfgaloistype(mf)
```

Vecsmall([18]) This eigenform is of type D_{18}.

```
mfgaloistype([239,1,-239])
```

Vecsmall([6, 10, 30])

Here there are three eigenforms, of type D_n with $n = 6$, 10, and 30 respectively.

Half-Integral Weight Examples

```
mf=mfinit([12,5/2]);B=mfbasis(mf);
for(j=1,#B,print(mfser(B[j],8)))
```

$$1 + 12q^5 + 30q^8 + O(q^9)$$
$$q - 8q^5 + 14q^6 + 28q^7 - 20q^8 + O(q^9)$$
$$q^2 + 8q^5 - q^6 - 10q^7 + 18q^8 + O(q^9)$$
$$q^3 - 4q^5 + 4q^6 + 10q^7 - 10q^8 + O(q^9)$$
$$q^4 + 2q^5 - 2q^6 - 4q^7 + 5q^8 + O(q^9)$$

```
F=mfShimura(B[1],5)[2];mfser(F,6)
```
$$12/5 + 12q + 132q^2 + 132q^3 + 1092q^4 + 1512q^5 + O(q^6)$$

Half-Integral Weight Examples

```
mf=mfinit([12,5/2]);B=mfbasis(mf);
for(j=1,#B,print(mfser(B[j],8)))
```

$$1 + 12q^5 + 30q^8 + O(q^9)$$
$$q - 8q^5 + 14q^6 + 28q^7 - 20q^8 + O(q^9)$$
$$q^2 + 8q^5 - q^6 - 10q^7 + 18q^8 + O(q^9)$$
$$q^3 - 4q^5 + 4q^6 + 10q^7 - 10q^8 + O(q^9)$$
$$q^4 + 2q^5 - 2q^6 - 4q^7 + 5q^8 + O(q^9)$$

```
F=mfShimura(B[1],5)[2];mfser(F,6)
```

$$12/5 + 12q + 132q^2 + 132q^3 + 1092q^4 + 1512q^5 + O(q^6)$$

Advanced Examples I

As mentioned at the beginning, we now give a number of examples which we believe are not possible (at least in general) with other packages.

Fourier expansion of $F|_k\gamma$

```
mf=mfinit([32,4],0);F=mfbasis(mf)[1];
[Ser(mfslashexpansion(mf,F,[1,0;2,1],6,1,&A),q),A]
```

$[Mod(-(1/64)t, t^8+1)q + Mod(-(1/4)t^3, t^8+1)q^3 + Mod(-(11/32)t^5, t^8+1)q^5 + O(q^7), [0,8]]$

Note: t is canonically $e^{2\pi i/16}$, and 8 means that $q = e^{2\pi i\tau/8}$.

Advanced Examples I

As mentioned at the beginning, we now give a number of examples which we believe are not possible (at least in general) with other packages.

Fourier expansion of $F|_k\gamma$

```
mf=mfinit([32,4],0);F=mfbasis(mf)[1];
[Ser(mfslashexpansion(mf,F,[1,0;2,1],6,1,&A),q),A]
```

$[Mod(-(1/64)t, t^8+1)q + Mod(-(1/4)t^3, t^8+1)q^3 + Mod(-(11/32)t^5, t^8+1)q^5 + O(q^7), [0, 8]]$

Note: t is canonically $e^{2\pi i/16}$, and 8 means that $q = e^{2\pi i\tau/8}$.

Advanced Examples II

More complicated:

```
mf=mfinit([36,3,-4],0);F=mfbasis(mf)[1];
[Ser(mfslashexpansion(mf,F,[1,0;2,1],6,1,&A),q),A]
```

$[Mod(-(1/54)t^4 - (1/54)t, t^6 + t^3 + 1) + Mod(-(1/3)t^5 - (1/3)t^2, t^6 + t^3 + 1)q^2 + Mod(-(2/9)t^4, t^6 + t^3 + 1)q^3 + O(q^4), [1/18, 9]]$

Here $t^6 + t^3 + 1$ is the 9th cyclotomic polynomial so t is canonically $e^{2\pi i/9}$, 9 means that $q = e^{2\pi i\tau/9}$, and $1/18$ means that the expansion must be multiplied by $e^{2\pi i\tau/18}$.

Recall: possible thanks to products of two Eisenstein series.

Can also have "raw" expansion:

```
Ser(mfslashexpansion(mf,F,[1,0;2,1],6,0,&A),q)
(0.00321570699383204349725401160683916288888 -
0.018237180611337186284569315270176352104 I) +
(0.25534814770632601173413088351847222465 -
0.21426253656217977544088113663575447764 I)q^2 + ...
```

Advanced Examples II

More complicated:

```
mf=mfinit([36,3,-4],0);F=mfbasis(mf)[1];
[Ser(mfslashexpansion(mf,F,[1,0;2,1],6,1,&A),q),A]
```

$[Mod(-(1/54)t^4 - (1/54)t, t^6 + t^3 + 1) + Mod(-(1/3)t^5 - (1/3)t^2, t^6 + t^3 + 1)q^2 + Mod(-(2/9)t^4, t^6 + t^3 + 1)q^3 + O(q^4), [1/18, 9]]$

Here $t^6 + t^3 + 1$ is the 9th cyclotomic polynomial so t is canonically $e^{2\pi i/9}$, 9 means that $q = e^{2\pi i\tau/9}$, and $1/18$ means that the expansion must be multiplied by $e^{2\pi i\tau/18}$.

Recall: possible thanks to products of two Eisenstein series.

Can also have "raw" expansion:

```
Ser(mfslashexpansion(mf,F,[1,0;2,1],6,0,&A),q)
```

$(0.00321570699383204349725401160683916288888 - 0.018237180611337186284569315270176352104\ I) + (0.255348147706326011734130883518472224 65 - 0.214262536562179775440881136635754477 64\ I)q^2 + \ldots$

Advanced Examples III

Atkin–Lehner operators in difficult cases:

```
mf=mfinit([32,4,8],0); [mfB,M,C]=mfatkininit(mf,32);
[M, C]
```

$[[1/8, -7/4; -1/16, -1/8], 0.35355339059327376220042218105242$

($C = 2^{-1/2}$).

```
F=mfbasis(mf)[1]; G=mfatkin(mf,F); mfser(G,10)
```

$(1/4)q + (7/2)q^3 + 7q^5 + 2q^7 - (1/4)q^9 + O(q^{11})$

Note: explicit constant C such that expansion of $C \cdot F|_k W_Q$ has same field of coefficients of F, as above. Similar question for general $F|_k\gamma$ under investigation.

Advanced Examples III

Atkin–Lehner operators in difficult cases:

```
mf=mfinit([32,4,8],0); [mfB,M,C]=mfatkininit(mf,32);
[M, C]
```

$[[1/8, -7/4; -1/16, -1/8], 0.353553390593273762200422181052424$
$(C = 2^{-1/2})$.

```
F=mfbasis(mf)[1]; G=mfatkin(mf,F); mfser(G,10)
```

$(1/4)q + (7/2)q^3 + 7q^5 + 2q^7 - (1/4)q^9 + O(q^{11})$

Note: explicit constant C such that expansion of $C \cdot F|_k W_Q$ has same field of coefficients of F, as above. Similar question for general $F|_k\gamma$ under investigation.

Advanced Examples III

Atkin–Lehner operators in difficult cases:

```
mf=mfinit([32,4,8],0); [mfB,M,C]=mfatkininit(mf,32);
[M, C]
```

$[[1/8, -7/4; -1/16, -1/8], 0.353553390593273762200422181052424$
$(C = 2^{-1/2})$.

```
F=mfbasis(mf)[1]; G=mfatkin(mf,F); mfser(G,10)
```

$(1/4)q + (7/2)q^3 + 7q^5 + 2q^7 - (1/4)q^9 + O(q^{11})$

Note: explicit constant C such that expansion of $C \cdot F|_k W_Q$ has same field of coefficients of F, as above. Similar question for general $F|_k\gamma$ under investigation.

Advanced Examples IV

Numerical computation of period polynomials

$$P(F,X) = \int_0^{i\infty} (X-\tau)^{k-2} F(\tau)\, d\tau$$

Needs the expansion of $F|_k W_N$, now easy. In particular, special values and periods.

More generally, numerical computation of general period polynomials and modular symbols

$$\int_{s_1}^{s_2} (X-\tau)^{k-2} F|_k\gamma(\tau)\, d\tau$$

for any cusps s_1 and s_2 and any $\gamma \in M_2^+(\mathbb{Q})$.

Advanced Examples IV

Numerical computation of period polynomials

$$P(F,X) = \int_0^{i\infty} (X-\tau)^{k-2} F(\tau)\, d\tau$$

Needs the expansion of $F|_k W_N$, now easy. In particular, special values and periods.

More generally, numerical computation of general period polynomials and modular symbols

$$\int_{s_1}^{s_2} (X-\tau)^{k-2} F|_k\gamma(\tau)\, d\tau$$

for any cusps s_1 and s_2 and any $\gamma \in M_2^+(\mathbb{Q})$.

Advanced Examples V

Computation of Petersson products of modular forms: By using Haberland-type formulas, can express Petersson products as bilinear expressions in general modular symbols, so now easy.

```
mf=mfinit([11,2],0);F=mfbasis(mf)[1];mfpetersson(mf,F,F)
0.00390834565612459898524738548138211386 18
```

```
mf=mfinit([23,2],1);B=mfbasis(mf);
for(i=1,#B,for(j=i,#B,print(mfpetersson(mf,B[i],B[j]))))
```

```
0.009593150872767286679013189786724534554 0
 − 0.006692042995757562005131315318410623119 2
0.016285193868524848684144505105135157673
```

Advanced Examples V

Computation of Petersson products of modular forms: By using Haberland-type formulas, can express Petersson products as bilinear expressions in general modular symbols, so now easy.

```
mf=mfinit([11,2],0);F=mfbasis(mf)[1];mfpetersson(mf,F,F)
```

0.00390834565612459898524738548138211386 18

```
mf=mfinit([23,2],1);B=mfbasis(mf);
for(i=1,#B,for(j=i,#B,print(mfpetersson(mf,B[i],B[j]))))
```

0.009593150872767286679013189786724534554 0
− 0.006692042995757562005131315318410623119 2
0.016285193868524848684144505105135157673

Advanced Examples V

Computation of Petersson products of modular forms: By using Haberland-type formulas, can express Petersson products as bilinear expressions in general modular symbols, so now easy.

```
mf=mfinit([11,2],0);F=mfbasis(mf)[1];mfpetersson(mf,F,F)
```

0.00390834565612459898524738548138211386 18

```
mf=mfinit([23,2],1);B=mfbasis(mf);
for(i=1,#B,for(j=i,#B,print(mfpetersson(mf,B[i],B[j]))))
```

0.009593150872767286679013189786724534554 0

− 0.006692042995757562005131315318410623119 2

0.016285193868524848684144505105135157673

Advanced Examples VI

Evaluation of a form near the real axis: For forms over the full modular group Γ or small index, can use modular transformations to significantly increase imaginary part. In general, not possible. Now easy: can always reduce to $\Im(\tau) \geq 1/(2N)$, almost optimal:

```
for(i=1,#B,print(mfeval(mf,B[i],1/Pi+I/10^7)))
```

```
0.090783713804535492499017418221928331504
  - 0.059239894966349901602857024156091190672 I
 - 0.045391856928135245866207977352609859692
  + 0.029619947256208807202238276196812855946 I
```

Advanced Examples VI

Evaluation of a form near the real axis: For forms over the full modular group Γ or small index, can use modular transformations to significantly increase imaginary part. In general, not possible. Now easy: can always reduce to $\Im(\tau) \geq 1/(2N)$, almost optimal:

```
for(i=1,#B,print(mfeval(mf,B[i],1/Pi+I/10^7)))
```

$$0.090783713804535492499017418221928331504$$
$$- 0.059239894966349901602857024156091190672\ I$$
$$- 0.045391856928135245866207977352609859692$$
$$+ 0.029619947256208807202238276196812855946\ I$$

Thank you for your attention !

The theory of modular forms is a fundamental tool used in many areas of mathematics and physics. It is also a very concrete and "fun" subject in itself and abounds with an amazing number of surprising identities.

This comprehensive textbook which includes numerous exercises aims to give a complete picture of the classical aspects of the subject, with an emphasis on explicit formulas. After a number of motivating examples such as elliptic functions and theta functions, the modular group, its subgroups, and general aspects of holomorphic and nonholomorphic modular forms are explained, with an emphasis on explicit examples. The heart of the book is the classical theory developed by Hecke and continued up to the Atkin–Lehner–Li theory of newforms and including the theory of Eisenstein series, Rankin–Selberg theory, and a more general theory of theta series including the Weil representation. The final chapter explores in some detail more general types of modular forms such as half-integral weight, Hilbert, Jacobi, Maass, and Siegel modular forms.

This book is essentially self-contained, the necessary tools being given in a separate chapter.

This book gives a beautiful introduction to the theory of modular forms, with a delicate balance of analytic and arithmetic perspectives. The target readership is graduate students in number theory, though it will also be accessible to advanced undergraduates and will, no doubt, serve as a valuable reference for researchers for years to come.

—Jennifer Balakrishnan, Boston University

This marvelous book is a gift to the mathematical community, and more specifically to anyone wanting to learn modular forms. The authors take a classical view of the material offering extremely helpful explanations in a generous conversational manner and covering such an impressive range of this beautiful, deep, and important subject.

—Barry Mazur, Harvard University

Modular forms are central to many different fields of mathematics and mathematical physics. Having a detailed and complete treatment of all aspects of the theory by two world experts is a very welcome addition to the literature.

—Peter Sarnak, Princeton University

ISBN 978-0-8218-4947-7
9 780821 849477
GSM/179

For additional information and updates on this book, visit
www.ams.org/bookpages/gsm-179

www.ams.org

GSM 179 **Modular Forms** **Cohen and Strömberg** AMS

GRADUATE STUDIES IN MATHEMATICS 179

Modular Forms:
A Classical Approach

Henri Cohen
Fredrik Strömberg

American Mathematical Society

712 pages • backspace 2 1/8" • 3-color cover PMS 632 (light blue), PMS 302(blue), 123 (yellow)