# Hypergeometric $L$-functions in average polynomial time

Edgar Costa, Kiran S. Kedlaya, and David Roe

Costa, Roe: Department of Mathematics, Massachusetts Institute of Technology
Kedlaya: Department of Mathematics, University of California, San Diego
edgarc@mit.edu, kedlaya@ucsd.edu, roed@mit.edu
slides at https://kskedlaya.org/slides/; see also arXiv:2005.13640, prerecorded talk

(virtual) Algorithmic Number Theory Symposium (ANTS-XIV)
University of Auckland (Te Whare Wānanga o Tāmaki Makaurau)
July 2, 2020

Kedlaya was supported by NSF (grant DMS-1802161) and UC San Diego (Warschawski Professorship). Costa and Roe were supported by the Simons Collaboration on Arithmetic Geometry, Number Theory, and Computation.

The MIT campus sits on the traditional unceded territory of the Wampanoag Nation; we acknowledge the painful history of genocide and forced removal from this territory. The UCSD campus sits on the ancestral homelands of the Kumeyaay Nation; the Kumeyaay people continue to have an important and thriving presence in the region.

# Contents

# Computing an arithmetic $L$-function

An arithmetic $L$-function over $\mathbb{Q}$ of some degree $r$ generally has the form

$$\prod_p \det(1 - p^{-s} F_p)^{-1}$$

where for all but finitely many $p$, $F_p$ is some $r \times r$ matrix. Rewrite

$$\det(1 - p^{-s} F_p)^{-1} = \exp\left(\sum_{f=1}^{\infty} \frac{1}{f} p^{-fs} \operatorname{Trace}(F_p^f)\right);$$

to compute the Dirichlet series up to $X$, we need $\operatorname{Trace}(F_p^f)$ for all prime powers $p^f \leq X$.

We are interesting in computing the hypergeometric $L$-function associated to a hypergeometric datum $(\alpha, \beta) \in (\mathbb{Q} \cap [0,1))^{r \times 2}$, for which $\operatorname{Trace}(F_p^f)$ is computed by a finite hypergeometric sum. In this paper, we focus on $f = 1$ and compute this trace modulo $p$.

# Computing an arithmetic $L$-function

An arithmetic $L$-function over $\mathbb{Q}$ of some degree $r$ generally has the form

$$\prod_p \det(1 - p^{-s} F_p)^{-1}$$

where for all but finitely many $p$, $F_p$ is some $r \times r$ matrix. Rewrite

$$\det(1 - p^{-s} F_p)^{-1} = \exp\left(\sum_{f=1}^{\infty} \frac{1}{f} p^{-fs} \operatorname{Trace}(F_p^f)\right);$$

to compute the Dirichlet series up to $X$, we need $\operatorname{Trace}(F_p^f)$ for all prime powers $p^f \leq X$.

We are interesting in computing the hypergeometric $L$-function associated to a hypergeometric datum $(\alpha, \beta) \in (\mathbb{Q} \cap [0,1))^{r \times 2}$, for which $\operatorname{Trace}(F_p^f)$ is computed by a finite hypergeometric sum. In this paper, we focus on $f = 1$ and compute this trace modulo $p$.

# Computing an arithmetic $L$-function

An arithmetic $L$-function over $\mathbb{Q}$ of some degree $r$ generally has the form

$$\prod_p \det(1 - p^{-s} F_p)^{-1}$$

where for all but finitely many $p$, $F_p$ is some $r \times r$ matrix. Rewrite

$$\det(1 - p^{-s} F_p)^{-1} = \exp\left(\sum_{f=1}^{\infty} \frac{1}{f} p^{-fs} \operatorname{Trace}(F_p^f)\right);$$

to compute the Dirichlet series up to $X$, we need $\operatorname{Trace}(F_p^f)$ for all prime powers $p^f \leq X$.

We are interesting in computing the hypergeometric $L$-function associated to a hypergeometric datum $(\alpha, \beta) \in (\mathbb{Q} \cap [0,1))^{r \times 2}$, for which $\operatorname{Trace}(F_p^f)$ is computed by a finite hypergeometric sum. In this paper, we focus on $f = 1$ and compute this trace modulo $p$.

# Finite hypergeometric sums

Using Gross–Koblitz to compute Gauss sums in the Beukers–Cohen–Mellit formula using the Morita $p$-adic Gamma function $\Gamma_p$, we get for $q = p$

$$\mathrm{Trace}(F_p) = H_p\left(\begin{smallmatrix}\alpha \\ \beta\end{smallmatrix}\middle| z\right) := \frac{1}{1-p}\sum_{m=0}^{p-2}(-p)^{\eta_m(\alpha)-\eta_m(\beta)}p^{D+\xi_m(\beta)} \left(\prod_{j=1}^{r}\frac{\Gamma_p(\alpha_j+\frac{m}{1-p})/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j+\frac{m}{1-p})/\Gamma_p(\beta_j)}\right)[z]^m$$

where $\eta_m, \xi_m, D$ are some combinatorial invariants of $\alpha, \beta$ and $[z] \in \mathbb{Z}_p^\times$ is the unique $(p-1)$-st root of unity congruent to $z$ modulo $p$. (We rig up $D$ to ensure $\eta_m(\alpha) - \eta_m(\beta) + D + \xi_m(\beta) \geq 0$; since $\Gamma_p$ takes values in $\mathbb{Z}_p^\times$, everything in sight is in $\mathbb{Z}_p$ rather than $\mathbb{Q}_p$.)

# Finite hypergeometric sums

Using Gross–Koblitz to compute Gauss sums in the Beukers–Cohen–Mellit formula using the Morita $p$-adic Gamma function $\Gamma_p$, we get for $q = p$

$$\mathrm{Trace}(F_p) = H_p\left(\begin{matrix}\alpha\\ \beta\end{matrix}\middle| z\right) := \frac{1}{1-p}\sum_{m=0}^{p-2}(-p)^{\eta_m(\alpha)-\eta_m(\beta)}p^{D+\xi_m(\beta)} \left(\prod_{j=1}^{r}\frac{\Gamma_p(\alpha_j+\frac{m}{1-p})/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j+\frac{m}{1-p})/\Gamma_p(\beta_j)}\right)[z]^m$$

where $\eta_m, \xi_m, D$ are some combinatorial invariants of $\alpha, \beta$ and $[z] \in \mathbb{Z}_p^\times$ is the unique $(p-1)$-st root of unity congruent to $z$ modulo $p$. (We rig up $D$ to ensure $\eta_m(\alpha) - \eta_m(\beta) + D + \xi_m(\beta) \geq 0$; since $\Gamma_p$ takes values in $\mathbb{Z}_p^\times$, everything in sight is in $\mathbb{Z}_p$ rather than $\mathbb{Q}_p$.)

# Quadratic versus linear complexity

The implementations in `Magma` and `Sage` compute $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ one $p$ at a time. Since the sum is over $O(p)$ terms, computing all prime Dirichlet coefficients up to $X$ requires $O(\frac{X^2}{\log X})$ arithmetic operations.

In our paper, we use the method of remainder forests (cf. Sutherland's paper) to amortize the computation over all $p \leq X$. This reduces the complexity to $O(X \log^3 X)$ (for fixed $\alpha, \beta$).

Reminder: we are only computing $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right) \pmod{p}$. However, we expect that one can work modulo $p^e$ with similar complexity (times some power of $e$). It would still remain to compute $H_{p^f}\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ for all $p^f \leq X$ with $f \geq 2$; this requires $O(\frac{X^{3/2}}{\log X})$ as written, but other techniques can reduce this to $O(X \log^? X)$ even without amortization.

# Quadratic versus linear complexity

The implementations in `Magma` and `Sage` compute $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ one $p$ at a time. Since the sum is over $O(p)$ terms, computing all prime Dirichlet coefficients up to $X$ requires $O(\frac{X^2}{\log X})$ arithmetic operations.

In our paper, we use the method of remainder forests (cf. Sutherland's paper) to amortize the computation over all $p \leq X$. This reduces the complexity to $O(X \log^3 X)$ (for fixed $\alpha, \beta$).

Reminder: we are only computing $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right) \pmod{p}$. However, we expect that one can work modulo $p^e$ with similar complexity (times some power of $e$). It would still remain to compute $H_{p^f}\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ for all $p^f \leq X$ with $f \geq 2$; this requires $O(\frac{X^{3/2}}{\log X})$ as written, but other techniques can reduce this to $O(X \log^? X)$ even without amortization.

# Quadratic versus linear complexity

The implementations in `Magma` and `Sage` compute $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ one $p$ at a time. Since the sum is over $O(p)$ terms, computing all prime Dirichlet coefficients up to $X$ requires $O(\frac{X^2}{\log X})$ arithmetic operations.

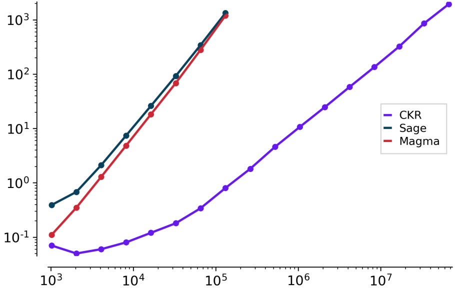
In our paper, we use the method of remainder forests (cf. Sutherland's paper) to amortize the computation over all $p \leq X$. This reduces the complexity to $O(X \log^3 X)$ (for fixed $\alpha, \beta$).

Reminder: we are only computing $H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right) \pmod p$. However, we expect that one can work modulo $p^e$ with similar complexity (times some power of $e$). It would still remain to compute $H_{p^f}\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right)$ for all $p^f \leq X$ with $f \geq 2$; this requires $O(\frac{X^{3/2}}{\log X})$ as written, but other techniques can reduce this to $O(X \log^? X)$ even without amortization.

# Timings

In this example $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}), z = \frac{1}{5}$. This $L$-function has weight 1, so $H_p\left(\begin{matrix}\alpha \\ \beta\end{matrix}\middle| z\right)$ is uniquely determined by its reduction mod $p$. (See §5.4 of the paper for more implementation details, and §5.5 for a worked example.)

| $X$ | Amortized | Sage | Magma |
|---|---|---|---|
| $2^{10}$ | 0.07s | 0.39s | 0.11s |
| $2^{11}$ | 0.05s | 0.68s | 0.35s |
| $2^{12}$ | 0.06s | 2.12s | 1.29s |
| $2^{13}$ | 0.08s | 7.39s | 4.83s |
| $2^{14}$ | 0.12s | 26.0s | 18.2s |
| $2^{15}$ | 0.18s | 92.3s | 68.4s |
| $2^{16}$ | 0.34s | 343s | 280s |
| $2^{17}$ | 0.80s | 1328s | 1190s |




| $X$ | $2^{18}$ | $2^{19}$ | $2^{20}$ | $2^{21}$ | $2^{22}$ | $2^{23}$ | $2^{24}$ | $2^{25}$ | $2^{26}$ |
|---|---|---|---|---|---|---|---|---|---|
| Amortized | 1.81s | 4.59s | 10.7s | 24.6s | 58.0s | 135s | 322s | 857s | 1948s |

# Contents

# Setup

Modulo $p$, the trace formula becomes

$$H_p\left(\begin{matrix}\alpha\\ \beta\end{matrix}\middle| z\right) \equiv \sum_{m=0}^{p-2} \pm p^* \left(\prod_{j=1}^{r} \frac{\Gamma_p(\alpha_j+m)/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j+m)/\Gamma_p(\beta_j)}\right) z^m \pmod p.$$

Call the $m$-th summand $P_m$. Suppose we had $f(m), g(m) \in \mathbb{Z}[m]$ so that

$$P_{m+1} \equiv \frac{f(m)}{g(m)} P_m \pmod p.$$

We could then set

$$B(m) := \begin{pmatrix} g(m) & 0 \\ g(m) & f(m) \end{pmatrix} = g(m) \begin{pmatrix} 1 & 0 \\ 1 & f(m)/g(m) \end{pmatrix}$$

and then use remainder products to compute

$$B(0)\dots B(p-2) \equiv g(0)\cdots g(p-2) \begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod p.$$

# Setup

Modulo $p$, the trace formula becomes

$$H_p\left(\begin{smallmatrix}\alpha\\ \beta\end{smallmatrix}\middle| z\right) \equiv \sum_{m=0}^{p-2} \pm p^* \left(\prod_{j=1}^{r} \frac{\Gamma_p(\alpha_j+m)/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j+m)/\Gamma_p(\beta_j)}\right) z^m \pmod{p}.$$

Call the $m$-th summand $P_m$. Suppose we had $f(m), g(m) \in \mathbb{Z}[m]$ so that

$$P_{m+1} \equiv \frac{f(m)}{g(m)} P_m \pmod{p}.$$

We could then set

$$B(m) := \begin{pmatrix} g(m) & 0 \\ g(m) & f(m) \end{pmatrix} = g(m) \begin{pmatrix} 1 & 0 \\ 1 & f(m)/g(m) \end{pmatrix}$$

and then use remainder products to compute

$$B(0)\dots B(p-2) \equiv g(0)\cdots g(p-2) \begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}.$$

# Setup

Modulo $p$, the trace formula becomes

$$H_p\left(\begin{smallmatrix}\alpha \\ \beta\end{smallmatrix}\middle| z\right) \equiv \sum_{m=0}^{p-2} \pm p^* \left(\prod_{j=1}^{r} \frac{\Gamma_p(\alpha_j + m)/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j + m)/\Gamma_p(\beta_j)}\right) z^m \pmod{p}.$$

Call the $m$-th summand $P_m$. Suppose we had $f(m), g(m) \in \mathbb{Z}[m]$ so that

$$P_{m+1} \equiv \frac{f(m)}{g(m)} P_m \pmod{p}.$$

We could then set

$$B(m) := \begin{pmatrix} g(m) & 0 \\ g(m) & f(m) \end{pmatrix} = g(m) \begin{pmatrix} 1 & 0 \\ 1 & f(m)/g(m) \end{pmatrix}$$

and then use remainder products to compute

$$B(0)\dots B(p-2) \equiv g(0)\cdots g(p-2) \begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}.$$

# Setup

Modulo $p$, the trace formula becomes

$$H_p \left( {\alpha \atop \beta} \middle| z \right) \equiv \sum_{m=0}^{p-2} \pm p^* \left( \prod_{j=1}^{r} \frac{\Gamma_p(\alpha_j + m)/\Gamma_p(\alpha_j)}{\Gamma_p(\beta_j + m)/\Gamma_p(\beta_j)} \right) z^m \pmod{p}.$$

Call the $m$-th summand $P_m$. Suppose we had $f(m), g(m) \in \mathbb{Z}[m]$ so that

$$P_{m+1} \equiv \frac{f(m)}{g(m)} P_m \pmod{p}.$$

We could then set

$$B(m) := \begin{pmatrix} g(m) & 0 \\ g(m) & f(m) \end{pmatrix} = g(m) \begin{pmatrix} 1 & 0 \\ 1 & f(m)/g(m) \end{pmatrix}$$

and then use remainder products to compute

$$B(0) \dots B(p-2) \equiv g(0) \cdots g(p-2) \begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}.$$

# Two related issues

- The factor $\pm p^*$ is determined by the **zigzag function*** at $\frac{m}{p-1}$:

$$Z_{\alpha,\beta} : [0,1] \to \mathbb{Z}, \quad Z_{\alpha,\beta}(x) := \#\{j : \alpha_j \le x\} - \#\{j : \beta_j \le x\}.$$

This creates a "discontinuity" when $\frac{m}{p-1}$ passes through $\alpha_j$ or $\beta_j$.

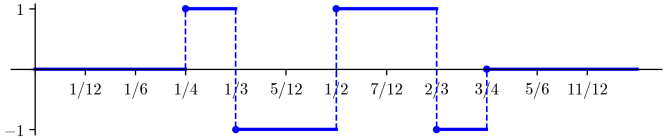



Figure: $Z_{\alpha,\beta}(x)$ for $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$

- Similar "discontinuities" arise from the functional equation for $\Gamma_p$:

$$\Gamma_p(x+1) = \begin{cases} -x\Gamma_p(x) & x \notin p\mathbb{Z}_p \\ -\Gamma_p(x) & x \in p\mathbb{Z}_p \end{cases}$$

* $Z_{\alpha,\beta}$ also determines the weight and Hodge numbers of the $L$-function.

# Two related issues

- The factor $\pm p^*$ is determined by the **zigzag function*** at $\frac{m}{p-1}$:

$$Z_{\alpha,\beta} : [0,1] \to \mathbb{Z}, \quad Z_{\alpha,\beta}(x) := \#\{j : \alpha_j \leq x\} - \#\{j : \beta_j \leq x\}.$$

This creates a “discontinuity” when $\frac{m}{p-1}$ passes through $\alpha_j$ or $\beta_j$.

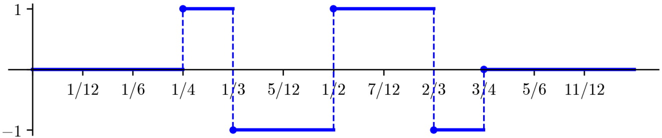



Figure: $Z_{\alpha,\beta}(x)$ for $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$

- Similar “discontinuities” arise from the functional equation for $\Gamma_p$:

$$\Gamma_p(x+1) = \begin{cases} -x\Gamma_p(x) & x \notin p\mathbb{Z}_p \\ -\Gamma_p(x) & x \in p\mathbb{Z}_p \end{cases}$$

* $Z_{\alpha,\beta}$ also determines the weight and Hodge numbers of the $L$-function.

# Resolution of the issues

We resolve both issues by "ferrying".†

- We break the summation at $\lfloor \alpha_j(p-1) \rfloor, \lfloor \beta_j(p-1) \rfloor$, and separate primes into classes modulo $\mathrm{lcd}(\alpha, \beta)$.
- Within each range and congruence class, we do a single amortized computation of matrix products.
- We then do non-amortized computations of transition matrices to "portage" or "ferry" across the breaks.

For each $p$, we put the ranges and transitions together to obtain a product computing a scalar multiple of $\begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}$.


†At ANTS-XIII in Madison, "portage" would have been a better metaphor.

# Resolution of the issues

We resolve both issues by “ferrying”.[†]

- We break the summation at $\lfloor \alpha_j(p-1) \rfloor, \lfloor \beta_j(p-1) \rfloor$, and separate primes into classes modulo $\mathrm{lcd}(\alpha, \beta)$.
- Within each range and congruence class, we do a single amortized computation of matrix products.
- We then do non-amortized computations of transition matrices to “portage” or “ferry” across the breaks.

For each $p$, we put the ranges and transitions together to obtain a product computing a scalar multiple of $\begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}$.


[†]At ANTS-XIII in Madison, “portage” would have been a better metaphor.

# Resolution of the issues

We resolve both issues by "ferrying".[†]

- We break the summation at $\lfloor \alpha_j(p-1) \rfloor, \lfloor \beta_j(p-1) \rfloor$, and separate primes into classes modulo $\mathrm{lcd}(\alpha, \beta)$.
- Within each range and congruence class, we do a single amortized computation of matrix products.
- We then do non-amortized computations of transition matrices to "portage" or "ferry" across the breaks.

For each $p$, we put the ranges and transitions together to obtain a product computing a scalar multiple of $\begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}$.


[†]At ANTS-XIII in Madison, "portage" would have been a better metaphor.

# Resolution of the issues

We resolve both issues by "ferrying".[†]

- We break the summation at $\lfloor \alpha_j(p-1) \rfloor, \lfloor \beta_j(p-1) \rfloor$, and separate primes into classes modulo $\mathrm{lcd}(\alpha, \beta)$.
- Within each range and congruence class, we do a single amortized computation of matrix products.
- We then do non-amortized computations of transition matrices to "portage" or "ferry" across the breaks.

For each $p$, we put the ranges and transitions together to obtain a product computing a scalar multiple of $\begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}$.


[†] At ANTS-XIII in Madison, "portage" would have been a better metaphor.

# Resolution of the issues

We resolve both issues by “ferrying”.[†]

- We break the summation at $\lfloor \alpha_j(p-1) \rfloor, \lfloor \beta_j(p-1) \rfloor$, and separate primes into classes modulo $\mathrm{lcd}(\alpha, \beta)$.
- Within each range and congruence class, we do a single amortized computation of matrix products.
- We then do non-amortized computations of transition matrices to “portage” or “ferry” across the breaks.

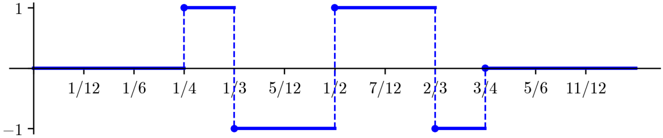
For each $p$, we put the ranges and transitions together to obtain a product computing a scalar multiple of $\begin{pmatrix} 1 & 0 \\ \sum_{m=0}^{p-2} P_m & P_{p-1} \end{pmatrix} \pmod{p}$.


[†]At ANTS-XIII in Madison, “portage” would have been a better metaphor.

# Contents

# Setup

Take $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$, $z = \frac{1}{5}$. We see that the $L$-function has weight 1 by plotting the zigzag function (again):

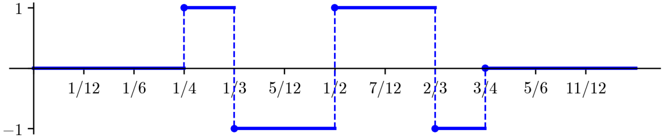



In particular, computing $H_p$ modulo $p$ is enough to determine it exactly. Denote the intervals we see by $I_0, \ldots, I_5$.

Since we are only working modulo $p$, the only intervals that contribute to the sum are $I_2 = (\frac{1}{3}, \frac{1}{2})$ and $I_4 = (\frac{2}{3}, \frac{3}{4})$. **However**, we do still have to compute over the other integrals in order to update the product!

# Setup

Take $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$, $z = \frac{1}{5}$. We see that the $L$-function has weight 1 by plotting the zigzag function (again):

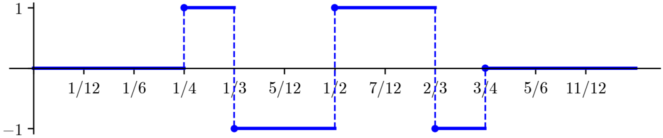



In particular, computing $H_p$ modulo $p$ is enough to determine it exactly. Denote the intervals we see by $I_0, \ldots, I_5$.

Since we are only working modulo $p$, the only intervals that contribute to the sum are $I_2 = (\frac{1}{3}, \frac{1}{2})$ and $I_4 = (\frac{2}{3}, \frac{3}{4})$. **However**, we do still have to compute over the other integrals in order to update the product!

# Setup

Take $\alpha = (\frac{1}{4}, \frac{1}{2}, \frac{1}{2}, \frac{3}{4}), \beta = (\frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3})$, $z = \frac{1}{5}$. We see that the $L$-function has weight 1 by plotting the zigzag function (again):




In particular, computing $H_p$ modulo $p$ is enough to determine it exactly. Denote the intervals we see by $I_0, \ldots, I_5$.

Since we are only working modulo $p$, the only intervals that contribute to the sum are $I_2 = (\frac{1}{3}, \frac{1}{2})$ and $I_4 = (\frac{2}{3}, \frac{3}{4})$. **However**, we do still have to compute over the other integrals in order to update the product!

# Amortized products

For simplicity, we focus on the case $p \equiv 7 \pmod{12}$. In the intervals that contribute to the sum, we take in the matrix product

$$\begin{aligned} f_{2,7}(k) &= 5184k^4 + 8640k^3 + 4428k^2 + 852k + 55, \\ g_{2,7}(k) &= 25920k^4 + 69120k^3 + 63360k^2 + 23040k + 2880, \\ f_{4,7}(k) &= 5184k^4 + 12096k^3 + 9612k^2 + 2820k + 175, \\ g_{4,7}(k) &= 25920k^4 + 86400k^3 + 106560k^2 + 57600k + 11520. \end{aligned}$$

Suppose we did the remainder forest and then took $p = 67$. We'd see

$$S_2(67) = \begin{pmatrix} 65 & 0 \\ 34 & 5 \end{pmatrix}, \qquad S_4(67) = \begin{pmatrix} 54 & 0 \\ 25 & 41 \end{pmatrix}.$$

# Amortized products

For simplicity, we focus on the case $p \equiv 7 \pmod{12}$. In the intervals that contribute to the sum, we take in the matrix product

$$\begin{aligned}
f_{2,7}(k) &= 5184k^4 + 8640k^3 + 4428k^2 + 852k + 55, \\
g_{2,7}(k) &= 25920k^4 + 69120k^3 + 63360k^2 + 23040k + 2880, \\
f_{4,7}(k) &= 5184k^4 + 12096k^3 + 9612k^2 + 2820k + 175, \\
g_{4,7}(k) &= 25920k^4 + 86400k^3 + 106560k^2 + 57600k + 11520.
\end{aligned}$$

Suppose we did the remainder forest and then took $p = 67$. We'd see

$$S_2(67) = \begin{pmatrix} 65 & 0 \\ 34 & 5 \end{pmatrix}, \qquad S_4(67) = \begin{pmatrix} 54 & 0 \\ 25 & 41 \end{pmatrix}.$$

# More amortized products and the portage

In order to compute the correct sum, we also do similar computations over the other intervals. At $p = 67$, we get

$$S_0(67) = \begin{pmatrix} 38 & 0 \\ 0 & 62 \end{pmatrix}, \qquad S_1(67) = \begin{pmatrix} 50 & 0 \\ 0 & 47 \end{pmatrix},$$

$$S_3(67) = \begin{pmatrix} 1 & 0 \\ 0 & 16 \end{pmatrix}, \qquad S_5(67) = \begin{pmatrix} 1 & 0 \\ 0 & 38 \end{pmatrix}.$$

For the "ferries", we work directly with $p = 67$ to compute

$$T_0(67) = \begin{pmatrix} 1 & 0 \\ 0 & 6 \end{pmatrix}, \quad T_1(67) = \begin{pmatrix} 1 & 0 \\ 0 & 31 \end{pmatrix}, \quad T_2(67) = \begin{pmatrix} 1 & 0 \\ -1 & 12 \end{pmatrix},$$

$$T_3(67) = \begin{pmatrix} 1 & 0 \\ -1 & 40 \end{pmatrix}, \quad T_4(67) = \begin{pmatrix} 1 & 0 \\ -1 & 40 \end{pmatrix}, \quad T_5(67) = \begin{pmatrix} 1 & 0 \\ -1 & 31 \end{pmatrix}.$$

# More amortized products and the portage

In order to compute the correct sum, we also do similar computations over the other intervals. At $p = 67$, we get

$$S_0(67) = \begin{pmatrix} 38 & 0 \\ 0 & 62 \end{pmatrix}, \qquad S_1(67) = \begin{pmatrix} 50 & 0 \\ 0 & 47 \end{pmatrix},$$

$$S_3(67) = \begin{pmatrix} 1 & 0 \\ 0 & 16 \end{pmatrix}, \qquad S_5(67) = \begin{pmatrix} 1 & 0 \\ 0 & 38 \end{pmatrix}.$$

For the "ferries", we work directly with $p = 67$ to compute

$$T_0(67) = \begin{pmatrix} 1 & 0 \\ 0 & 6 \end{pmatrix}, \quad T_1(67) = \begin{pmatrix} 1 & 0 \\ 0 & 31 \end{pmatrix}, \quad T_2(67) = \begin{pmatrix} 1 & 0 \\ -1 & 12 \end{pmatrix},$$

$$T_3(67) = \begin{pmatrix} 1 & 0 \\ -1 & 40 \end{pmatrix}, \quad T_4(67) = \begin{pmatrix} 1 & 0 \\ -1 & 40 \end{pmatrix}, \quad T_5(67) = \begin{pmatrix} 1 & 0 \\ -1 & 31 \end{pmatrix}.$$

# A worked example (part 4)

Putting the product together, we get

$$S(67) = T_0(67)S_0(67)\cdots T_5(67)S_5(67) = \begin{pmatrix} 21 & 0 \\ 33 & 21 \end{pmatrix}$$

so $H_{67}\left(\begin{smallmatrix}\alpha \\ \beta\end{smallmatrix}\middle|\tfrac{1}{5}\right) \equiv \frac{33}{21} \equiv 59 \pmod{67}$. This checks with `Magma` and `Sage`:

```
H := HypergeometricData([[1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]]);
HypergeometricTrace(H, 5, 67);
-8

sage: from sage.modular.hypergeometric_motive \
....: import HypergeometricData as Hyp
sage: H = Hyp(alpha_beta=([1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]))
sage: H.trace(67, 1, 1/5)
-8
```

# A worked example (part 4)

Putting the product together, we get

$$S(67) = T_0(67)S_0(67)\cdots T_5(67)S_5(67) = \begin{pmatrix} 21 & 0 \\ 33 & 21 \end{pmatrix}$$

so $H_{67}\left(\begin{smallmatrix}\alpha \\ \beta\end{smallmatrix}\middle| \frac{1}{5}\right) \equiv \frac{33}{21} \equiv 59 \pmod{67}$. This checks with `Magma` and `Sage`:

```
H := HypergeometricData([[1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]]);
HypergeometricTrace(H, 5, 67);
-8
```

```
sage: from sage.modular.hypergeometric_motive \
....: import HypergeometricData as Hyp
sage: H = Hyp(alpha_beta=([1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]))
sage: H.trace(67, 1, 1/5)
-8
```

# A worked example (part 4)

Putting the product together, we get

$$S(67) = T_0(67)S_0(67)\cdots T_5(67)S_5(67) = \begin{pmatrix} 21 & 0 \\ 33 & 21 \end{pmatrix}$$

so $H_{67}\left(\begin{smallmatrix}\alpha \\ \beta\end{smallmatrix}\middle|\frac{1}{5}\right) \equiv \frac{33}{21} \equiv 59 \pmod{67}$. This checks with `Magma` and `Sage`:

```
H := HypergeometricData([[1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]]);
HypergeometricTrace(H, 5, 67);
-8
```

```
sage: from sage.modular.hypergeometric_motive \
....: import HypergeometricData as Hyp
sage: H = Hyp(alpha_beta=([1/4,1/2,1/2,3/4],[1/3,1/3,2/3,2/3]))
sage: H.trace(67, 1, 1/5)
-8
```

# Contents

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m,x]/(x^e)$ where $x$ is a proxy for $[z]-z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m,x]/(x^e)$ where $x$ is a proxy for $[z]-z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m,x]/(x^e)$ where $x$ is a proxy for $[z]-z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m,x]/(x^e)$ where $x$ is a proxy for $[z]-z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m,x]/(x^e)$ where $x$ is a proxy for $[z]-z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Raising the modulus

There are two main issues with working modulo a higher power of $p$.

- The general formula has $[z]$ (the $(p-1)$-st root of unity congruent to $z$ modulo $p$) instead of $z$. One can compute $[z]$ modulo $p^e$ (e.g., by a Newton-Raphson iteration) but this does not integrate well into the amortization.
- The general formula has $\Gamma_p(\alpha_j + \frac{m}{1-p})$ rather than $\Gamma_p(\alpha_j + m)$. One can compute $\Gamma_p$ using its Mahler expansion in a residue disc, but it takes $O(p)$ complexity to compute the coefficients (e.g., modulo $p^2$ one needs $(p-1)! \pmod{p^2}$ as in a search for Wilson primes).

To deal with the first issue, one can use Harvey's "generic prime" strategy: replace $\mathbb{Z}[m]$ with $\mathbb{Z}[m, x]/(x^e)$ where $x$ is a proxy for $[z] - z$.

To deal with the second issue, we replace $p$ by a second nilpotent variable $y$, and integrate Mahler coefficients into the amortized computation.

We have not tried this! But it should work well in practice for small $e$.

# Prime-power traces

We also need a plan for dealing with the $p^f$-Frobenius traces for $f > 1$.

For Dirichlet coefficients up to $X$, there are $O(\frac{X^{1/2}}{\log X})$ of these, and the primes involved are $O(X^{1/2})$. So we don't need to amortize if we can reduce the individual complexity from $O(p^f)$ to $O(p)$.

This is achieved by algorithms that compute a suitable matrix $F_p$. For example, one can compute the Frobenius structure on the hypergeometric differential equation and specialize it suitably (as in Lauder's **deformation method** for zeta functions).

# Prime-power traces

We also need a plan for dealing with the $p^f$-Frobenius traces for $f > 1$.

For Dirichlet coefficients up to $X$, there are $O(\frac{X^{1/2}}{\log X})$ of these, and the primes involved are $O(X^{1/2})$. So we don't need to amortize if we can reduce the individual complexity from $O(p^f)$ to $O(p)$.

This is achieved by algorithms that compute a suitable matrix $F_p$. For example, one can compute the Frobenius structure on the hypergeometric differential equation and specialize it suitably (as in Lauder's **deformation method** for zeta functions).

# Prime-power traces

We also need a plan for dealing with the $p^f$-Frobenius traces for $f > 1$.

For Dirichlet coefficients up to $X$, there are $O(\frac{X^{1/2}}{\log X})$ of these, and the primes involved are $O(X^{1/2})$. So we don't need to amortize if we can reduce the individual complexity from $O(p^f)$ to $O(p)$.

This is achieved by algorithms that compute a suitable matrix $F_p$. For example, one can compute the Frobenius structure on the hypergeometric differential equation and specialize it suitably (as in Lauder's **deformation method** for zeta functions).