Machine Learning L-functions

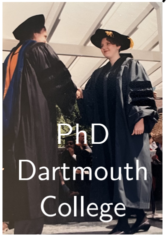
Tamara Veenstra

IDA-CCR La Jolla & University of Redlands (Emerita)

February 21, 2026

Southern California Number Theory Day

My career journey to and around Southern California



1997

2001

SCAMPs+
2007-2022

2023

[1] OK, really this is a photo of Torrey Pines State Beach and not CCRL.

What is IDA-CCR La Jolla and SCAMP?

Descriptions from IDA website

- “The Center for Communications and Computing (CCC) is a federally funded research and development center operated by the Institute for Defense Analyses (IDA). CCC consists of three distinct research centers: CCR La Jolla, CCR Princeton, CCS Bowie.”

What is IDA-CCR La Jolla and SCAMP?

Descriptions from IDA website

- "The Center for Communications and Computing (CCC) is a federally funded research and development center operated by the Institute for Defense Analyses (IDA). CCC consists of three distinct research centers: CCR La Jolla, CCR Princeton, CCS Bowie."
- "CCRL was founded in 1989 in La Jolla, California" (San Diego) and "now employs many PhD mathematicians, statisticians, and computer scientists working on problems in cryptography, cryptanalysis, machine learning, high-performance computing, and network security, as well as related areas of pure and applied mathematics."

What is IDA-CCR La Jolla and SCAMP?

Descriptions from IDA website

- "The Center for Communications and Computing (CCC) is a federally funded research and development center operated by the Institute for Defense Analyses (IDA). CCC consists of three distinct research centers: CCR La Jolla, CCR Princeton, CCS Bowie."
- "CCRL was founded in 1989 in La Jolla, California" (San Diego) and "now employs many PhD mathematicians, statisticians, and computer scientists working on problems in cryptography, cryptanalysis, machine learning, high-performance computing, and network security, as well as related areas of pure and applied mathematics."
- "SCAMP is an annual 10-week summer program (at each of the 3 centers) that brings together a diverse group of creative mathematicians and other scientists to provide practical solutions to critical real-world problems."

Machine Learning and L-functions

- Big question: There is much excitement in the world of machine learning and AI! Can we use these tools to discover new mathematics?

Machine Learning and L-functions

- Big question: There is much excitement in the world of machine learning and AI! Can we use these tools to discover new mathematics?
- Some things are hard to compute directly. Can get information about them from other features? Can we find new patterns and relationships from analysing large sets of data? Do these patterns reveal interesting structure?

Machine Learning and L-functions

- Big question: There is much excitement in the world of machine learning and AI! Can we use these tools to discover new mathematics?
- Some things are hard to compute directly. Can get information about them from other features? Can we find new patterns and relationships from analysing large sets of data? Do these patterns reveal interesting structure?
- The research in this talk originated at the Mathematics and Machine Learning Program at Harvard's Center for Mathematical Sciences and Applications in Fall of 2024. (Number theory week.)

Machine Learning and L-functions

- Big question: There is much excitement in the world of machine learning and AI! Can we use these tools to discover new mathematics?
- Some things are hard to compute directly. Can get information about them from other features? Can we find new patterns and relationships from analysing large sets of data? Do these patterns reveal interesting structure?
- The research in this talk originated at the Mathematics and Machine Learning Program at Harvard's Center for Mathematical Sciences and Applications in Fall of 2024. (Number theory week.)
- Joint work with Joanna Bieri[2], Giorgi Butbaia, Edgar Costa, Alyson Deines[3], Kyu-Hwan Lee, David Lowry-Duda, Tom Oliver, and Yidi Qi.

[2]Also SoCal: University of Redlands
[3]Also SoCal: IDA-CCR La Jolla

L-functions

L-functions are fundamental objects in number theory

- Generalize the Riemann zeta function.

L-functions

L-functions are fundamental objects in number theory

- Generalize the Riemann zeta function.
- Associated with various objects in number theory.

L-functions

L-functions are fundamental objects in number theory

- Generalize the Riemann zeta function.
- Associated with various objects in number theory.
- Can study families of L-functions.

L-functions

L-functions have certain properties

- Dirichlet series

$$L(s) = \sum_{n \geq 1} a_n n^{-s}$$

Enough to know a_{p^n} for prime p since $a_{nm} = a_n a_m$ if $gcd(n, m) = 1$.

L-functions

L-functions have certain properties

- Dirichlet series

$$L(s) = \sum_{n \geq 1} a_n n^{-s}$$

Enough to know a_{p^n} for prime p since $a_{nm} = a_n a_m$ if $gcd(n, m) = 1$.

- Functional equation

$$\Lambda(s) := N^{s/2} \Gamma_L(s) \cdot L(s) = \varepsilon \overline{\Lambda}((1 + w) - s),$$

L-functions

L-functions have certain properties

- Dirichlet series

$$L(s) = \sum_{n\geq 1} a_n n^{-s}$$

  Enough to know a_{p^n} for prime p since $a_{nm} = a_n a_m$ if $gcd(n, m) = 1$.

- Functional equation

$$\Lambda(s) := N^{s/2}\Gamma_L(s) \cdot L(s) = \varepsilon\overline{\Lambda}((1 + w) - s),$$

where:

- $\Gamma_L(s)$ are defined in terms of Γ-functions.
- The degree d is roughly the number of these Γ-factors
- $\varepsilon \in \{z \in \mathbb{C} : |z| = 1\}$ is the root number (for our examples today $\varepsilon = \pm 1$)
- N is the conductor of $L(s)$,
- $w \in \mathbb{N}$ is the (motivic) weight of $L(s)$.

L-functions: What do they know? What can we learn from them?

L-functions can arise from many sources, and we have a database of them!

www.lmfdb.org: The L-functions and Modular Forms Database

L-functions: What do they know? What can we learn from them?

L-functions can arise from many sources, and we have a database of them!

www.lmfdb.org: The L-functions and Modular Forms Database

They contain arithmetic information about their number theoretic sources:

- Class number formula for a number field K:

$$\lim_{s \to 1} (s-1) L(K,s) = \frac{2^{r_1} \cdot (2\pi)^{r_2} \cdot \mathrm{Reg}_K \cdot h_K}{w_K \cdot \sqrt{|D_K|}}$$

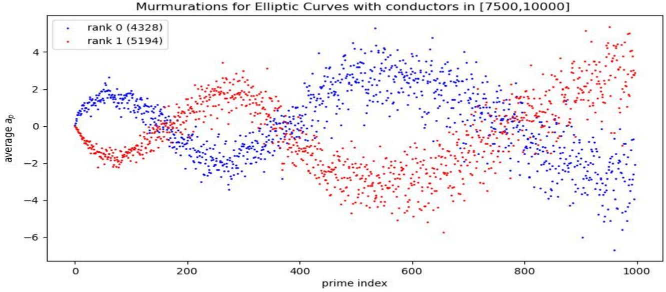
- Birch and Swinnerton-Dyer conjecture for an elliptic curve E:

$$L(E,s) \text{ vanishes to order } r := \operatorname{rank} E$$

Question: can we learn anything directly from the coefficients a_n? (ML)

An unexpected result!

In 2022, "Murmurations of Elliptic Curves" paper, He, Lee, Oliver, and Pozdnyakov observed fluctuations in average a_p that give information about the rank/root number.[4]



These statistical correlations between the root numbers of L-functions and Dirichlet coefficients seem to lead to ML's ability to predict accurately.

[4] The root number $\epsilon = (-1)^r$ where r is the rank.

Old School Cool: Mestre-Nagao Sums

Motivating heuristic: Mestre-Nagao sums

$$S(B) = \frac{1}{\log B} \sum_{p<B} \frac{a_p(E) \log p}{p}$$

- Used to predict the rank of elliptic curves
- Linear relationship $\Rightarrow$ maybe simple (linear) machine learning techniques will work

Agnostic Murmurations

Our first question involves studying how strong the murmuration pattern is across different types of datasets. That is, what if we just take a large set of L-functions in the LMFDB?

The dataset from LMFDB contains:

- 248,359 rational L-functions with root analytic conductor at most 4.
- 186,114 primitive L-functions.
- for each L-function all a_p for primes $p \leq 1000$
- https://zenodo.org/records/14774042
- Types of L-functions: Dirichlet, Artin, ECQ, CMF, G2Q, ECNF, HMF, BMF. (with a lot of overlap)

Data Normalizations

For each rational L-function, $L(s) = \sum_{n \geq 1} a_n n^{-s}$, dataset includes 168 a_p for $p \leq 1000$.

We use two normalizations:

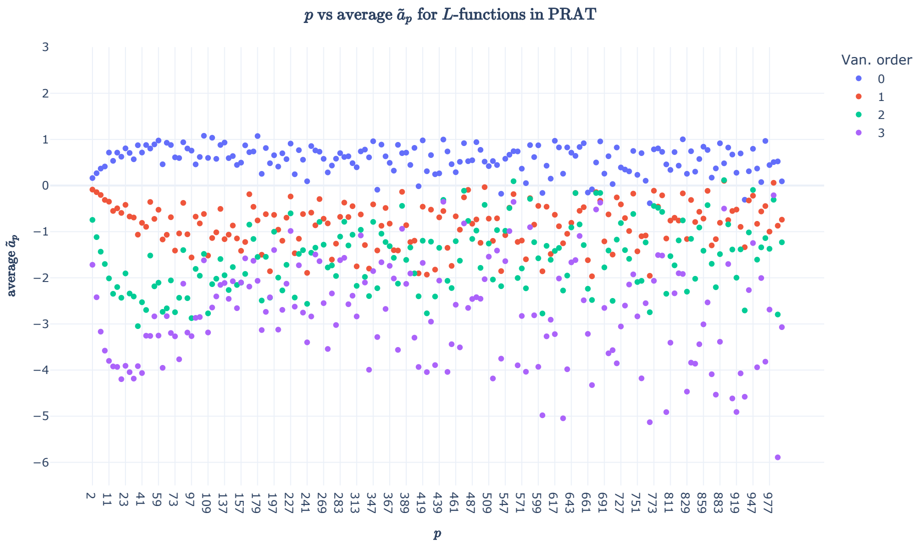
Murmurations:

$$\widetilde{a_p} = \frac{a_p}{p^{(w-1)/2}}$$

Hasse Bound/Machine Learning:

$$\overline{a_p} = \frac{a_p}{dp^{w/2}}$$

Agnostic Murmurations



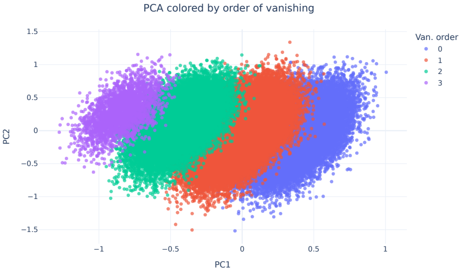
Starting with some simple ML techniques

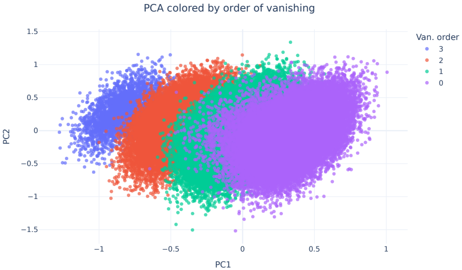
- Principal Component Analysis
  - Unsupervised dimensionality reduction technique
  - Assumes directions with the highest variance contain the most important information
  - Creates new, orthogonal (perpendicular) axes created by linear combinations of the original variables
  - Unlabeled data, applied to entire dataset

Starting with some simple ML techniques

- Principal Component Analysis
  - Unsupervised dimensionality reduction technique
  - Assumes directions with the highest variance contain the most important information
  - Creates new, orthogonal (perpendicular) axes created by linear combinations of the original variables
  - Unlabeled data, applied to entire dataset
- Linear Discriminant Analysis
  - Supervised dimensionality reduction technique
  - Goal is to maximize separability between classes
  - Learns linear decision boundary based on maximizing distance between the means of the two classes and minimizing variance within class
  - Labeled data, separate data into training and testing sets (usual 80/20 split)

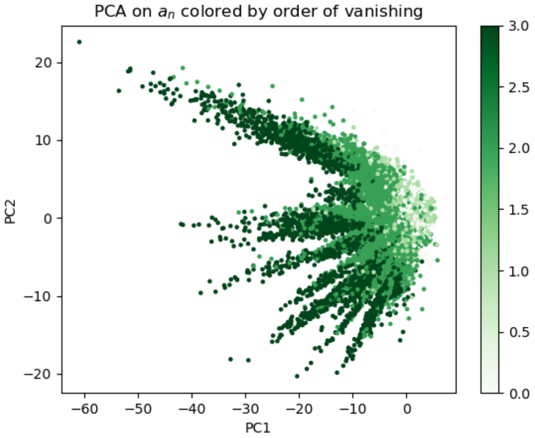
Principal component analysis





What's happening here?

Something very different happens for composite n. (PCA on a_n for $n \leq 100$.)



Linear Discriminant Analysis

Training and testing sets for a_ps for primitive, rational L-functions with degree 4, motivic weight 1 and order of vanishing ≤ 3.

Dataset	Training obs.	Validation obs.	Accuracy	Explained Variance	Counts	
PRAT*	140 924	35 232	0.959	0.982	0	53 344
					1	90 327
					2	29 648
					3	2 837
BMF	65 442	16 361	0.958	0.979	0	28 280
					1	44 773
					2	8 724
					3	26
ECNF	90 791	22 698	0.956	0.983	0	42 558
					1	61 243
					2	9 661
					3	27
G2Q	50 224	12 556	0.971	0.997	0	10 827
					1	29 155
					2	19 988
					3	2 810
HMF	25 571	6 393	0.963	0.988	0	14 443
					1	16 582
					2	938
					3	1

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.
- Simple techniques worked surprisingly well. Techniques employed for specific classes of L-function should generalize.

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.
- Simple techniques worked surprisingly well. Techniques employed for specific classes of L-function should generalize.
- More complicated techniques also worked well (e.g. Neural Nets).

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.
- Simple techniques worked surprisingly well. Techniques employed for specific classes of L-function should generalize.
- More complicated techniques also worked well (e.g. Neural Nets).
- The data set was somewhat unbalanced between types of L-functions, so that may be impacting the results.

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.
- Simple techniques worked surprisingly well. Techniques employed for specific classes of L-function should generalize.
- More complicated techniques also worked well (e.g. Neural Nets).
- The data set was somewhat unbalanced between types of L-functions, so that may be impacting the results.
- How do these tools perform for non-rational L-functions?

Upshot

- Rational L-functions as a dataset seem to be agnostic to their source, when normalized accordingly.
- Simple techniques worked surprisingly well. Techniques employed for specific classes of L-function should generalize.
- More complicated techniques also worked well (e.g. Neural Nets).
- The data set was somewhat unbalanced between types of L-functions, so that may be impacting the results.
- How do these tools perform for non-rational L-functions?
- How well does one type of L-function learn on a different type?

Non-rational L-functions (Maass forms)

Maass forms are similar to modular forms.

	Classical modular form	Maass form
Domain	$\mathcal{H}$	$\mathcal{H}$
Symmetry group contained in	$\mathrm{GL}_2(\mathbb{Z})$	$\mathrm{GL}_2(\mathbb{R})$
Fourier expansion	$\sum_{n\geq 1} a_n e^{2\pi inz}$	$\sum_{n\geq 1} a_n \sqrt{y} K_{iR}(2\pi\|ny\|) e^{2\pi inx}$ $K_{iR}(u)$: Bessel function
a_n	algebraic	transcendental in general
$L(f,s)$	$\sum_{n\geq 1} a_n n^{-s}$	$\sum_{n\geq 1} a_n n^{-s}$
Difficulty to compute L	Straightforward	Much harder!!!

Maass Forms and Murmurations?

LMFDB recently added about 35K Maass forms.

- Murmurations $\Rightarrow$ predicting root number from Fourier coefficients??
- For Maass forms, the root number

$$\epsilon = (-1)^{\sigma(f)} w_N$$

  where w_N is the **Fricke sign** and $\sigma(f)$ is the parity of the **symmetry** (f even/odd.)
- Also, $f(z) = w_N f\left(\frac{-1}{Nz}\right)$ for Maass forms of level N (Fricke involution).

Incomplete data in LMFDB

Unfortunately, for about half of the Maass forms, the Fricke sign (and hence root number) is missing. The Fricke sign is complicated to compute. In general, it requires:

- trace formula
- Hejhal's algorithm
- modularity properties

Best Case Scenario: Only need the trace formula and Hejhal's algorithm. These are the known Fricke signs in the LMFDB. (19,993 out of 35,416).

Worst Case Scenario: Zeros of the Bessel functions make the previous numerical calculations unstable, so need modularity conditions to produce an overdetermined system from which we can eventually deduce the Fricke sign. These are the unknown Fricke signs in the LMFDB. (15,423 out of 35,416)

We would like to predict w_N in these cases when it is hard to compute!

Our Dataset

- 35,416 rigorously computed Maass forms from the LMFDB.
- All with weight 0, trivial character, and level N in the range from 1 to 105.
- Dataset contains the first 1000 Fourier coefficients a_n for each Maass form.

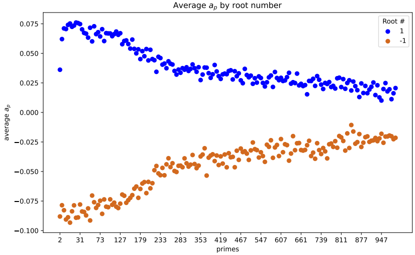
Separated by root number

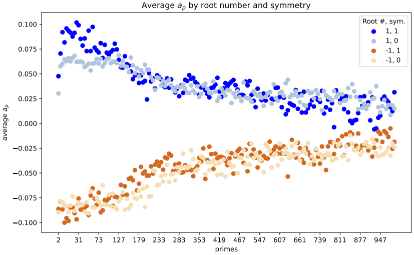
root number	total
1	10895
-1	9098
0 (unknown)	15423

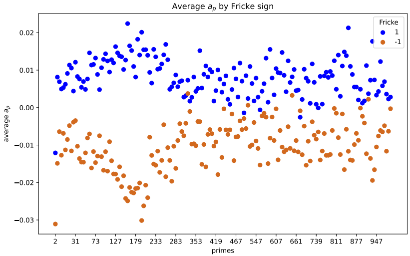
Separated by symmetry (σ) and Fricke sign (w):

	$w = -1$	$w = 1$	$w = 0$ (unknown)	Total
$\sigma = 0$ (even)	5,009	7,171	6,173	18,353
$\sigma = 1$ (odd)	3,724	4,089	9,250	17,063
Total	8,733	11,260	15,423	35,461

Murmuration type patterns? root number? symmetry? Fricke sign?







Experiments on training and validation data

Performed LDA on all the training data (12,795 observations). Tested accuracy on validation data (3199 observations). [80/20 split, stratified on root number.]

LDA features	accuracy on testing set
using all a_n	0.9677
using only a_p	0.8647

Experiments on training and validation data

Performed LDA on all the training data (12,795 observations). Tested accuracy on validation data (3199 observations). [80/20 split, stratified on root number.]

LDA features	accuracy on testing set
using all a_n	0.9677
using only a_p	0.8647

Surprising result:

- a_n is better than a_p!
- Why??

Why are a_n so much better than a_p?

Feature indices[5]	LDA accuracy	number of a_p
all n	96.8%	1000
prime n	86.5%	168
n is a prime power	87.1%	193
n is divisible by $2, 3$ or 5	79.2%	734
n is divisible by $2, 3$ or 5 or n is prime	90.0%	899
n is divisible by 1 or 2 prime factors	95.5%	702
n even	70.7%	500
n odd	94.7%	500
n 45-smooth	75.7%	507
$\{n \in \mathbb{Z} : 1 \leq n \leq 500\}$	93.9%	500

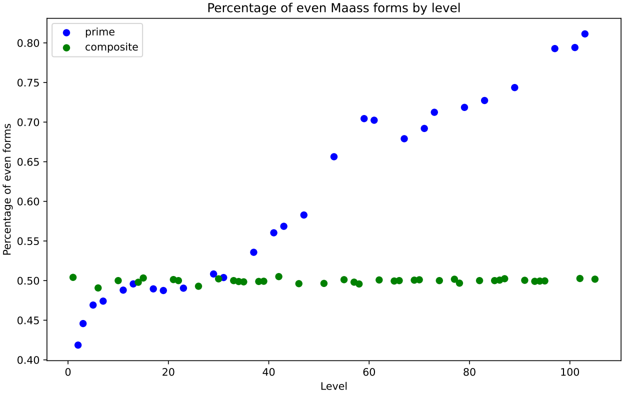
[5] Exploring subsets: $1 \leq n \leq 1000$ unless otherwise specified.

Does symmetry matter?

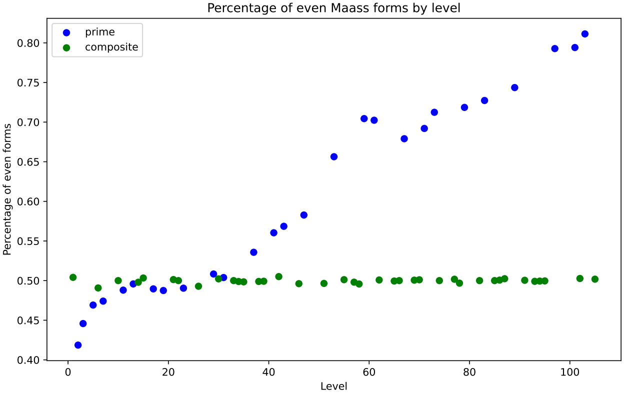
- Unbalanced data set: for known root numbers, 7813 odd and 12180 even
- For unknown root numbers: 9250 odd and 6173 even.
- There is some computational bias happening in the symmetry. See next slide.
- Training/testing sets as before.

LDA features	all symmetry (normalized)	even symmetry	odd symmetry
using all a_n	0.968	0.960	0.966
using only a_p	0.865	0.834	0.898

How unbalanced is the dataset?



How unbalanced is the dataset?



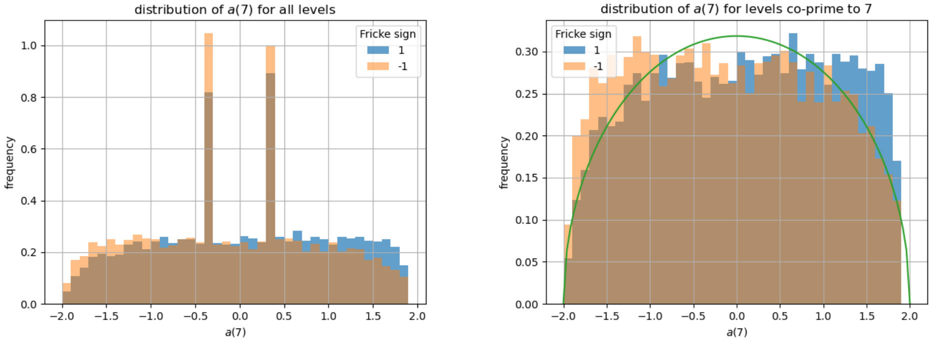
Should Maass forms be so unbalanced with symmetry by prime level?

- Asymptotically, no. They should be evenly split.
- Computationally, maybe? At larger prime levels, there are reasons why the even

Is LDA cheating?

Fourier coefficients contain information about Fricke sign/root number when $p|N$.

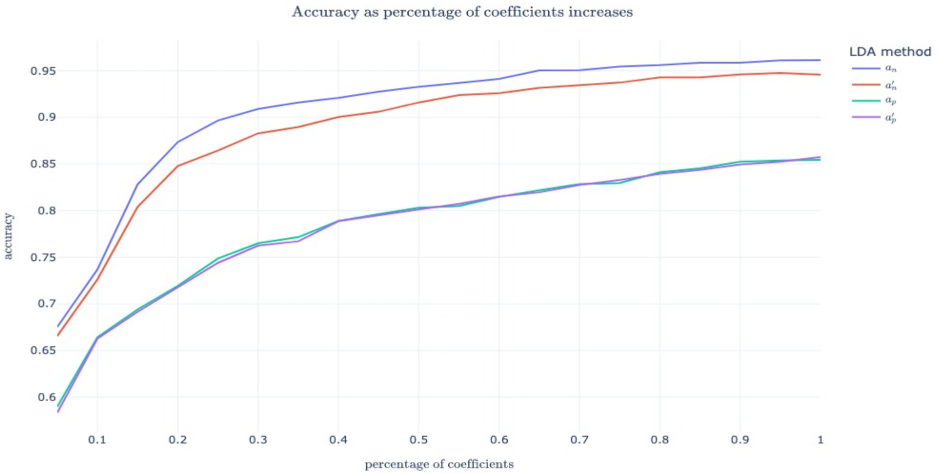
- If Fricke sign is known, $a_p = -w_p/\sqrt{p}$ and $w_N = \pm 1 = \prod_{p|N} w_p$.
- If Fricke sign is unknown, $a_n := 0$ for $\gcd(n, N) > 1$.



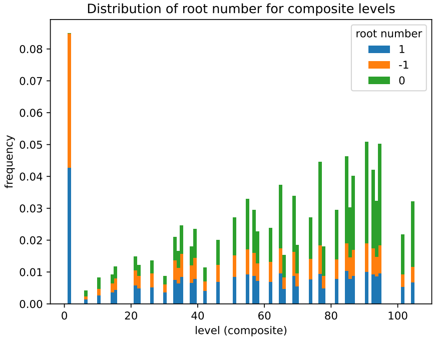
Spikes at $\pm 1/\sqrt{p}$ versus Sato-Tate circular distribution.

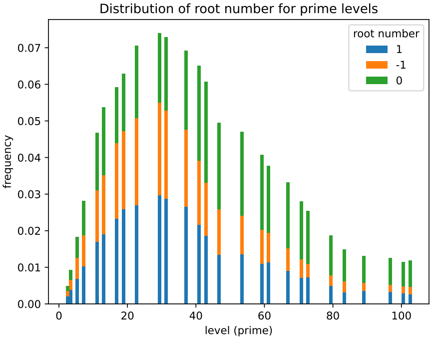
Eliminating Fricke sign info in a_p and how many a_n do we need?

For known Fricke signs we compare original a_n and a'_n where $a'_n := 0$ if $\gcd(n, N) > 1$

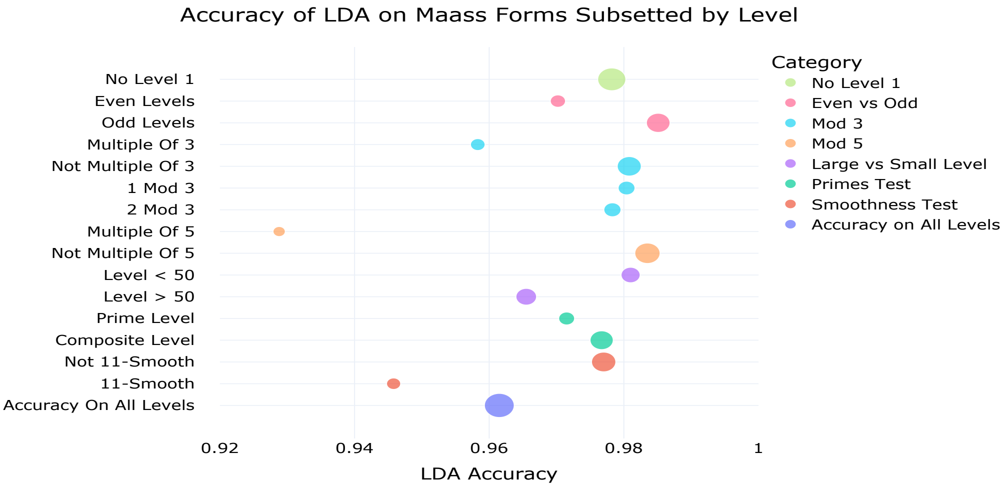


Level Analysis: Another way LDA might be cheating

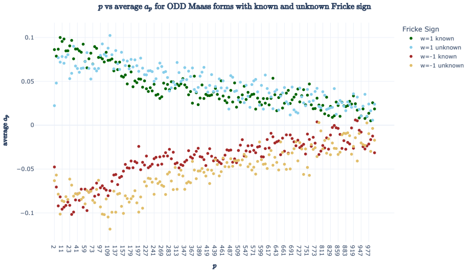


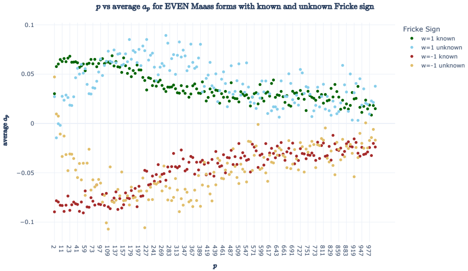


Are some levels easier to predict?



- We have unknown Fricke signs in the dataset.
- Using LDA to make predictions on these. How do we assess?
- Compare to murmurations.





- Also good accuracy with heuristic methods of computing Fricke sign.

Comparing to heuristic Hejhal's algorithm.

Instead of the computationally expensive rigorous computations of Fricke sign, can use *heuristic* guesses for Fricke signs.

- Given an eigenvalue and a set of Atkin-Lehner involution signs:
- evaluate Maass forms at several points in several different ways to produce candidate lists of Fourier coefficients
- If guessed well: get consistent system of equations and can solve for Fricke sign
- Try all possible values for eigenvalue and set of Atkin-Lehner involution signs and see if they match.

Precision in LMFDB was sufficient for this heuristic method to work on 4595 (out of 15423) forms with unknown Fricke signs.

Comparing predictions with Hejhal heuristic

Similar results to training and validation data!

LDA features	% agreement with Hejhal heuristic
a_n	95.45
a_{p_i}	82.93

Parity	% agreement with Hejhal heuristic
odd	96.09 (2509 out of 2611)
even	94.61 (1877 out of 1984)

Conclusion for Maass forms

- Really good accuracy!
- Predicting with LDA is very easy!
- Predictions with Neural Networks had similar accuracy.
- Even the heuristic calculations were surprisingly hard.
- Hope to use the predicted values to help determine the unknown signs precisely!

Agnostic Murmurations: Transfer Learning

- How well does one type of L-function generalize?
- Starting with rational L-functions.
- What happens when we train on one type, but test on another?

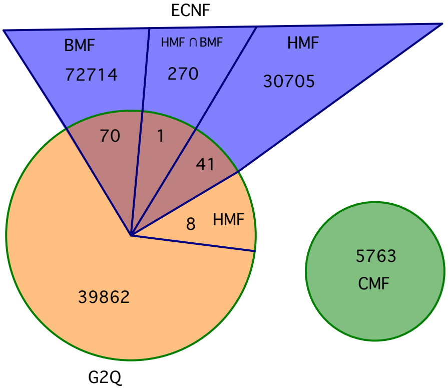
Now we restrict to:

- Primitive
- Order of vanishing 0 **and** 1
- Motivic weight 1

Which gives:

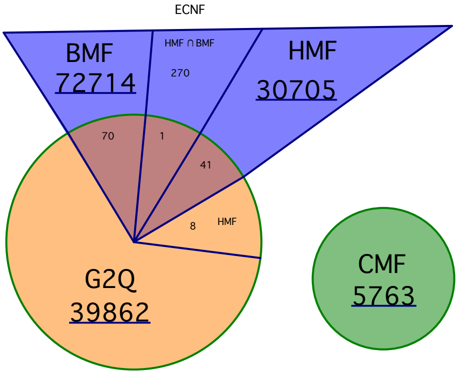
- All CMF's are ECQ's
- All CMF's have degree 2
- Everything else has degree 4

We looked at about 150k rational L-functions of small arithmetic complexity



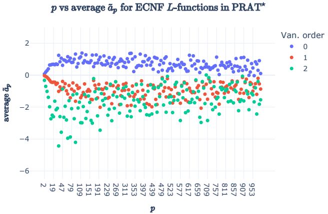
Here's the weird interesting L-function in the intersection ECNF, HMF, BMF, and G2Q: https://beta.lmfdb.org/L/4/2e13/1.1/c1e2/0/0

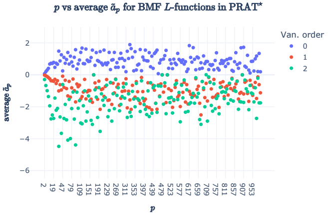
We looked at about 150k rational L-functions of small arithmetic complexity

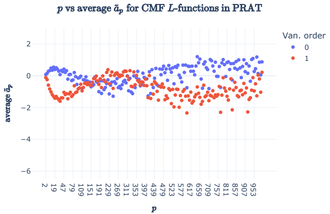


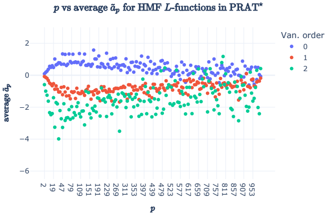
Our training and testing sets come from the four disjoint sets, that we'll refer to as BMF, HMF, G2Q, and CMF.

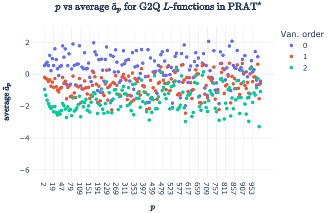
How strong are the murmuration type patterns?



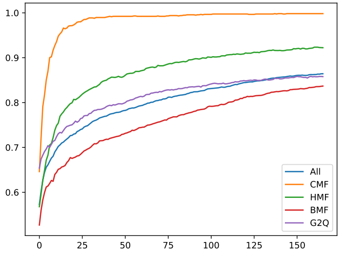




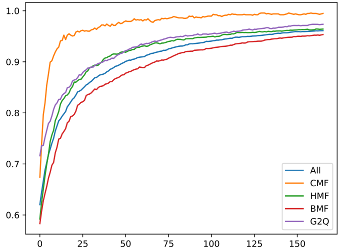




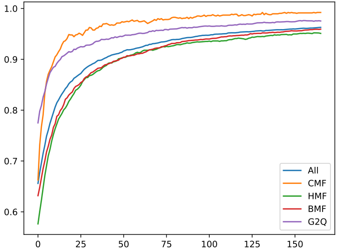
How does the transfer learning work (as we add more a_p)?



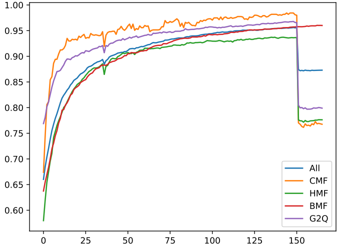
- CMF



- HMF



- G2Q



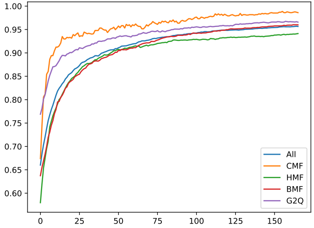
- BMF

What is happening with the BMFs??

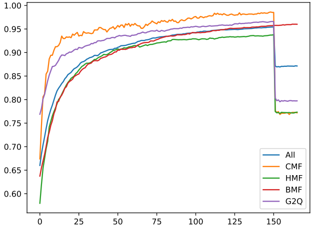
- In our original dataset, we discovered two labels were incorrect. (Just labels, not L-functions!) Two L-functions were labeled as ECNF when they should have been labeled as **both** ECNF and BMF.
- Transfer learning on this original dataset had no dips and BMF did much better!
- We double checked that the BMFs were correctly computed in the LMFDB!!
- So what's up with these two BMFs and these two primes???
  - L-function 4-643e2-1.1-c1e2-0-0
  - L-function 4-1879e2-1.1-c1e2-0-0
  - The primes are 167 and 887.

Take out the weird L-functions:

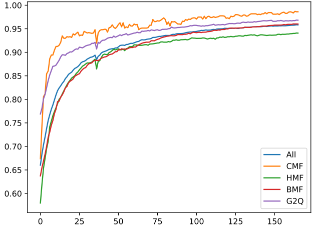
- Neither in training set



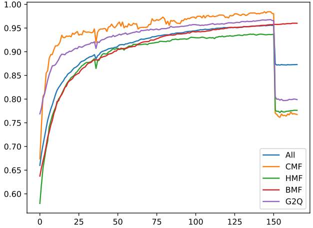
- Only the second in training set



- Only the first in training set



- Both used in training set



Outlier BMF L-functions

There are four sparse columns in the BMF dataset (and none in the others):

- 167 - one nonzero at 4-643e2-1.1-c1e2-0-0
- 479 - one nonzero at 4-643e2-1.1-c1e2-0-0
- 503 - all zeros
- 887 - one nonzero 4-1879e2-1.1-c1e2-0-0

Note: in the old BMF dataset, the columns for the primes 167, 479, 503, and 887 were all zeros. No other subset has any columns that are all zeros!

L-functions from elliptic curves over number fields

For rational L-functions, $L = \sum_{n\geq 1} a_n n^{-s}$, the values of a_p depend on how prime p splits in the number field K.

Let E/K be an elliptic curve defined over a number field K with ring of integers $\mathcal{O}_K$, then

$$L(E/K, s) = \sum_{\mathfrak{n} < \mathcal{O}_K} a_{\mathfrak{n}} N_{K/\mathbb{Q}}(\mathfrak{n})^{-s}$$

For quadratic fields, the possibilities are:

- $p\mathcal{O}_K = \mathfrak{p}_1\mathfrak{p}_2$, then $a_p = a_{\mathfrak{p}_1} + a_{\mathfrak{p}_2}$ [split]
- $p\mathcal{O}_K = \mathfrak{p}$, then $a_p = 0$ [inert]
- $p\mathcal{O}_K = \mathfrak{p}^2$, then $a_p = a_{\mathfrak{p}}$ [ramified]

BMF Number Fields

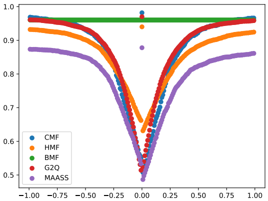
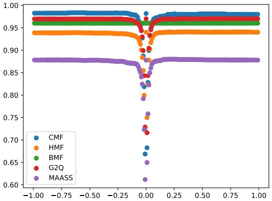
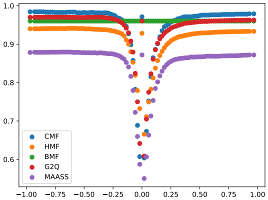
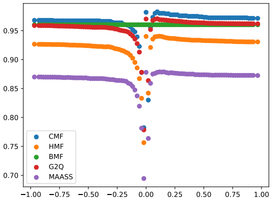
The smallest primes that are inert ($a_p = 0$ trivially) for most number fields in this set.

LMFDB Number Field	Count	167	479	503	887
2.0.4.1	40275	Inert	Inert	Inert	Inert
2.0.3.1	42808	Inert	Inert	Inert	Inert
2.0.8.1	36907	Inert	Inert	Inert	Inert
2.0.7.1	28322	Inert	Inert	Inert	Inert
2.0.11.1	30608	Inert	Inert	Inert	Inert
2.0.643.1	1	Split	Split	Split[6]	Inert
2.0.1879.1	1	Inert	Inert	Inert	Split

But why the differences between 167, 479, and 887??

[6] But $a_{503} = -24 + 24 = 0$ (nontrivially) in this case.

Changing values of L-function at given prime

prime	outlier BMF	random BMF	value
887			0.201
167			-.464

So many questions!

- Is this all an artifact of LDA? Didn't happen with linear support vector machines (good) or decision trees (bad). LDA fragile against outliers, but this is still striking!
- What would happen with a more complete/larger BMF dataset?
- Transfer learning does remarkably well on the different rational L-functions. What about Maass forms?

Training and testing on dataset with outlier BMF L-functions removed.

Testing Set

Training Set	CMF	HMF	BMF	G2Q	MAASS
CMF	0.998265	0.9280248	0.8331156	0.8656716	0.8524631
HMF	0.993929	0.9649894	0.9517981	0.9729086	0.8754689
BMF	0.982654	0.9377951	0.9602558	0.969773	0.8754689
G2Q	0.992194	0.9487054	0.9590181	0.977549	0.8732183
MAASS	0.9679098	0.8628888	0.8208073	0.7802584	0.8647162

Would it work better if we used all the a_n instead of a_p?

Future Directions

- Raise analytic rank/root analytic conductor?
- Could we just get a few more BMFs?
- Can we learn the origin of an L-function?
- Can we predict in advance what the outliers are?
- Is LDA fragile against incorrect values?
- Study in the context of Mestre-Nagao sums

Thank You!

- Maass Forms paper available: arxiv.org/abs/2501.02105.
- Code and data available on Zenodo: doi.org/10.5281/zenodo.15716014, doi.org/10.5281/zenodo.15490636
- Rational L-functions paper available: arxiv.org/abs/2502.10360
- Code and data available on Zenodo: doi.org/10.5281/zenodo.14774042
- Neural network analysis (ECQ) coming soon!
- Transfer learning results coming soon!
- Slides and references to papers available on my website, `tamarabveenstra.com`