ReLU Can Compute Numerical Integration
Mechanistic Interpretation of a Non-linear Activation

Chun Hei Yip [1] **Rajashree Agrawal** [2] **Jason Gross** [2]

Abstract

We hypothesize that the core challenge to mechanistically interpreting MLP layers is the difficulty of compressing non-linear behavior. In this work, we extend the analysis from Nanda et al. (2023) and Zhong et al. (2023), to develop an end-to-end interpretation of the 1 layer MLP-only modular addition transformer model with symmetric embeds. We present a new interpretation of the ReLU: that it implement quadrature schemes to carry out numerical integration, and we provide anecdotal and mathematical evidence in support.

1. Introduction

An open problem in mechanistic interpretation of toy models is understanding dense non-linear activations (Elhage et al., 2021). Current approaches largely either treat non-linear activations as detecting sparse binary ("on-off") features with scaling (Bricken et al., 2023; Olah et al., 2020; Marks et al., 2024; Elhage et al., 2022; Cunningham et al., 2023), or treat them as black-boxes, only describing input-output behavior (Nanda et al., 2023; Zhong et al., 2023).

The goal of mechanistic interpretability is to find a *compressed* explanation of the model's behavior on some dataset without loosing too much *correspondence* (Gross et al., 2024). The "brute force" explanation of an algorithm—"run the computation on all points of interest"—requires time roughly $\mathcal{O}(n \cdot d)$ to validate for a model with n parameters running on a dataset of size d. For a dataset whose description can be compressed into size d', a "complete" interpretation using an algorithm of size $a \ll n, d$ would compress this explanation to $\mathcal{O}(a(n + d'))$: we would pay $\mathcal{O}(a \cdot n)$ to check that the proposed algorithm corresponds to the model, and $\mathcal{O}(a \cdot d')$ to check that the algorithm explains the the data.

The key challenge in interpreting non-linearities is their expressiveness. While k-layer linear networks are not any more expressive than 1-layer linear networks of the same width, deep non-linear networks are not easily compressed. Concretely, if we have k matrices A_i of shape $m \times m$, a complete description of input-output behavior of $\prod_i A_i$ or $\sum_i A_i$ requires only m^2 parameters, independent of k. Hence we can systematically compress the brute force explanation of size $\mathcal{O}(km^2d)$ down to $\mathcal{O}(m^2d + km^{2.4})$ via matrix multiplication (Coppersmith & Winograd, 1990). By contrast, no such compression of $x \mapsto \mathrm{ReLU}(A\,\mathrm{ReLU}(Bx))$ nor $x \mapsto \mathrm{ReLU}(Ax) + \mathrm{ReLU}(Bx)$ is possible in general unless we assume that activations are sparse.

In some of the most detailed mechanistic interpretations, compression is bottlenecked on interpreting densely connected non-linearities. Nanda et al. (2023) and Zhong et al. (2023) provide exciting mechanistic interpretations of transformers with non-linear ReLU activations trained to perform modular addition. They infer an algorithm where the binary features are frequencies, by comprehensively analyzing the linear components of the models and black-boxing the ReLUs, describing only the input-output behavior. While the existing interpretation of Zhong et al. (2023) permits compressing the brute-force explanation of most of the model so much that it can be checked in time linear in the parameter count, checking that the ReLU in the MLP has the specified input-output behavior requires $\mathcal{O}(p \cdot d_{\mathrm{mlp}})$ time, where p is the prime modulus and d_{mlp} is the number of neurons. For $p \gg d_{\mathrm{model}}$ the hidden dimension, this asymptotically exceeds the parameter count of the model.

In this work, we successfully interpret a ReLU followed by a densely connected linear layer. Our interpretation allows a compression of the complete model behavior which can be validated on the full dataset in time linear in the parameter count. We find that the ReLU implements a quadrature scheme to carry out numerical integration, and provide empirical and mathematical evidence in support.

We describe the model architecture in section 2, following Nanda et al. (2023) and Zhong et al. (2023) in analysing the model by way of Fourier transform; here we present the input-output behavior specification of the MLP following Zhong et al. (2023). In section 3, we describe how using the amplitude-phase form of the Fourier series rather than the

[1]University of Cambridge [2]Independent. Correspondence to: Jason Gross <jgross@mit.edu>.

Copyright 2024 by the author(s).

sine-cosine allows us to describe the MLP as implementing a quadrature scheme to carry out numerical integration to double the frequency of trigonometric functions, and present suggestive evidence in support. In section 4, we demonstrate how understanding of numerical integration allows us to compute and verify a non-vacuous worst-case bound on the error of the model in time linear in the parameter count.

2. Setup

We work with the 'pizza' model from Zhong et al. (2023): a one-layer ReLU transformer with constant attention $= \frac{1}{2}$ for all tokens, trained to compute $M : (x, y) \mapsto (x+y) \bmod p$. The model takes input (x, y) encoded as one-hot vectors, and outputs $\mathrm{logit}(x, y, z)$ for all possible values $(\bmod p)$, with the largest logit representing its predicted answer. We represent the model computation of logits as

$$\mathrm{logit}(x, y, z) = \sum_i \mathrm{ReLU}(\mathrm{OV}(x)_i + \mathrm{OV}(y)_i + \mathrm{embed}(y)_i) \cdot (W_{\mathrm{out}})_{zi} + \text{residual terms}$$

Here, the W_{out} matrix represents the unembed layer and final linear layer combined (we ignore the bias term because it is very small), and the notation $\mathrm{OV}(x)$ represents the OVE circuit applied to the one-hot vector for x.

Zhong et al. (2023) state that the model computation of logits can be approximated as

$$\mathrm{logit}(x, y, z) \propto |\cos(k(x-y)/2)| \cos(k(x+y-z))$$

and offer an argument of how this expression arises. As is standard in mechanistic interpretability, part of their argument is justified visually, by noting in Appendix A via plots that $|\cos(t)| - |\sin(t)| \approx \cos(2t)$, and a more complicated function in Appendix L. In order to check that a parameterisation using their stated algorithm is valid, we'd have to sum d_{mlp} terms for all p points separately.

3. Finding numerical integration

Let us first set up some notation for the model architecture. Since attention is uniform, the model can be expressed as follows:

$$\begin{aligned} F_1(x, y) &= \mathrm{OV}(x) + \mathrm{OV}(y) + \mathrm{Embed}(y) \\ F_2(x, y) &= W_{\mathrm{in}} F_1(x, y) + b_{\mathrm{in}} \\ \mathrm{logits}(x, y) &= W_{out}\, \mathrm{ReLU}(F_2(x, y)) + b_{out} + W_U F_1(x, y) \end{aligned}$$

where W_{in}, W_{out}, W_U, b_{in}, b_{out}, are parameters of the model.

As explained in (Nanda et al., 2023), we carry out Fourier transform on the OV vectors in the token dimension, and

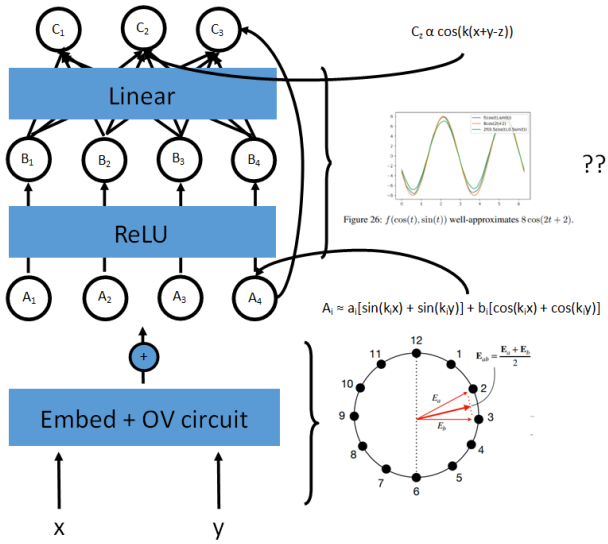

Figure 1. A diagram showing our model architecture and existing interpretations. The question marks indicate that the explanation there is incomplete: a brute force algorithm was used to plot the graph to ensure that the result was valid over $[-\pi, \pi]$.

get that each neuron, pre-ReLU, is "single-frequency":

$$\mathrm{OV}(x)_i \approx a_i \cos(k_i x) + b_i \sin(k_i x) = c_i \cos(k_i x + \phi_i)$$

We shall present the main results of the derivation, for details see Appendix A.

We have then

$$F_2(x, y)_i \approx d_i \cos(k_i(x-y)/2) \cos(k_i(x+y)/2 + \phi_i)$$

Furthermore, the columns of W_{out} are also 'single-frequency', with frequencies for each column matching the corresponding OV frequency, that is:

$$(W_{\mathrm{out}})_{zi} \approx e_i \cos(k_i z + \psi_i)$$

Some manipulation of the logit expression gives

$$\begin{aligned} &\mathrm{logit}(x, y, z) \\ &= \sum_i \mathrm{ReLU}((F_2(x, y))_i)(W_{out})_{zi} \\ &\approx \sum_k |\cos(k(x-y)/2)| \\ &\{\sum_i r_i\, \mathrm{ReLU}[\cos(k(x+y)/2 + \phi_i)] \cos(\psi_i)\} \cos(kz) \\ &+ \{\sum_i r_i\, \mathrm{ReLU}[\cos(k(x+y)/2 + \phi_i)] \sin(\psi_i)\} \sin(kz) \end{aligned}$$

We note from Figure 2 that $\psi_i \approx -2\phi_i$.

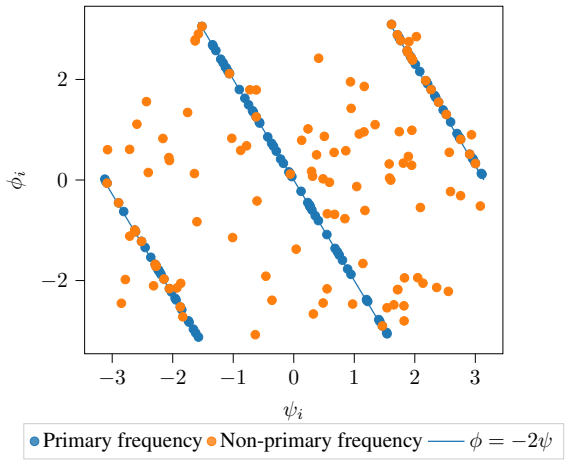

Figure 2. Angles for frequency $k = 12$. $\psi_i \approx -2\phi_i$ for the primary frequency of each neuron but not in general.

Furthermore, noting that the weighted sum by r_i resembles a quadrature scheme, suppose we replaced

$$\sum_i r_i \,\mathrm{ReLU}[\cos(k(x+y)/2+\phi_i)]\cos(-2\phi_i)$$

with

$$R_k \int_{-\pi}^{\pi} \mathrm{ReLU}[\cos(k(x+y)/2+\phi)]\cos(-2\phi)\,\mathrm{d}\phi$$

and similarly for the second term. Computing the integrals:

$$\int_{-\pi}^{\pi} \cos(-2\phi) ReLU \cos(\tfrac{k}{2}+\phi)\,\mathrm{d}\phi = \tfrac{2}{3}\cos(k)$$
$$\int_{-\pi}^{\pi} \sin(-2\phi) ReLU \cos(\tfrac{k}{2}+\phi)\,\mathrm{d}\phi = \tfrac{2}{3}\sin(k)$$

We would then recover

$$\begin{aligned}\mathrm{logit}(x,y,z) &\approx \sum_k \frac{2}{3} R_k(\cos(k(x+y))\cos(kz)+\sin(k(x+y))\sin(kz)) \\ &= \sum_k \frac{2}{3} R_k \cos(k(x+y-z))\end{aligned}$$

which is the derivation of the final unembed layer shown in (Nanda et al., 2023).

4. How numerical integration works

The question then becomes a mathematical one of evaluating the efficiency of the quadrature scheme

$$\int_{-\pi}^{\pi} f(\phi)\,\mathrm{d}\phi \approx \sum_i \frac{r_i}{R_k} f(\phi_i)$$

We first provide empirical evidence. For ease of further investigation, we split the ReLU function as

$$\mathrm{ReLU}(x) = \frac{x}{2} + \frac{|x|}{2}$$

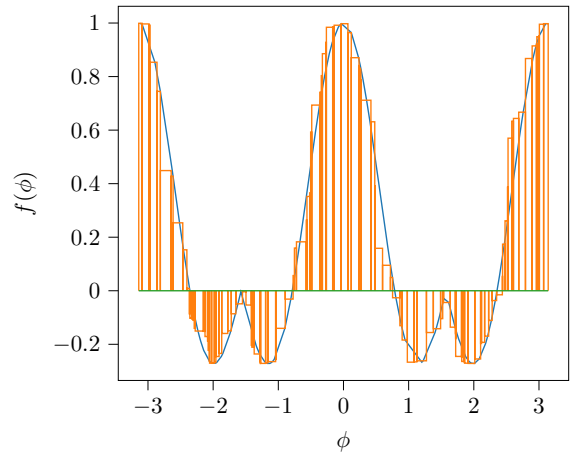

Figure 3. Converting the weighted sum into rectangles to estimate an integral (for frequency $k = 12$).

(i.e. respectively the 'id' component and the 'abs' component)

The maximum empirical error (obtained by evaluating the expression for each value of $x+y \pmod p$) is shown below:

Error Bound Type \ Freq.	12	18	21	22
Normalised abs error	0.04	0.03	0.04	0.03
Normalised id error	0.06	0.05	0.04	0.04
Numerical abs $\int$ bound	0.60	0.41	0.46	0.41
Naive abs bound	0.74	0.74	0.74	0.74

The naive bound is given by the average value of

$$\left|\int_{-\pi}^{\pi} f(\phi)\,\mathrm{d}\phi\right|$$

and the error is given by

$$\left|\int_{-\pi}^{\pi} f(\phi)\,\mathrm{d}\phi - \sum_i \frac{r_i}{R_k} f(\phi_i)\right|$$

We compute the error both by brute force exactly (the first two rows) and by mathematical analysis (the third row). The exact error is much smaller than the size of the integral, and the mathematical error bound is also smaller than the size of the integral. This gives convincing evidence that the model is indeed performing numerical integration.

Below we show how the mathematical error bound is produced.

One way to make sense of the quadrature scheme is to interpret the RHS as rectangles with width r_i sampled at the point ϕ_i gives the visualization of Figure 2, as follows:

First normalise the widths r_i such that $\sum_i r_i = 2\pi$, the range of integration. Then, sort the angles and line up the

rectangles with width r_i correspondingly to form end-points $a_0 = -\pi, a_1 = -\pi + r_1, a_2 = -\pi + r_1 + r_2, \ldots, a_n = -\pi + r_1 + ... + r_n = \pi$. Note that since the function is 2π-periodic, we can shift the intervals formed above by any constant. In what follows, when bounding approximation error, we use the shift that gives the lowest bound.

Additionally, by utilising the fact that the function $f(x)$ is actually π-periodic rather than 2π-periodic, we can overlap the two halves of sampled points to try and reduce the error of integration. (In this way, the rectangles in the approximation are narrower and so the error would be smaller.)

5. Error bound calculation

The discrepancy is: $\left|\int_{-\pi}^{\pi} f(x) - f(\phi_i)\,\mathrm{d}x\right|$ where ϕ_i is the angle s.t. $x \in [a_{i-1}, a_i]$. A crude bound is:

$$\begin{aligned}
&\leq \textstyle\int_{-\pi}^{\pi} |f(x) - f(\phi_i)|\,\mathrm{d}x \\
&\leq \textstyle\int_{-\pi}^{\pi} |x - \phi_i| \cdot \sup_x |f'(x)|\,\mathrm{d}x \\
&\leq \sup_x |f'(x)| \textstyle\sum_i \left(\int_{a_{i-1}-\phi_i}^{a_i-\phi_i} |x|\,\mathrm{d}x\right)
\end{aligned}$$

We must also include approximation error from $\psi_i \approx 2\phi_i$: we have $\sum_i r_i\,|\cos(k_i(x+y)/2 - \phi_i)| \cdot (\cos\psi_i - \cos 2\phi_i) \leq \sum_i r_i\,|\cos(k_i(x+y)/2 - \phi_i)| \cdot |\psi_i - 2\phi_i| \leq \sum_i r_i |\psi_i - 2\phi_i|$.

Using the specific frequency $k = 12$, the angle discrepancy gives us a bound of 0.15. The code for the analysis is linked here.

We can bound (analytically)

$$\sup_x |f'(x)| \leq 2$$

and the integral within the sum can be evaluated easily in $O(d_{mlp})$ time. This gives us a way of bounding the error in integration in time linear in the number of sample points (p).

Using this method on our data gives a bound of 0.45, so combined with above, this gives a bound of 0.60. We can compare this bound with the brute force bound and the naive bound as shown in the table above. We also include results for the other frequencies.

Note that our bounding argument is $O(n_{\text{freq}} d_{mlp})$, while the procedure to check the maximum error is $O(p d_{mlp})$. Since there are $p d_{mlp}$ effective parameters in the MLP layer, our argument is of order square root in the number of parameters (assuming $p \approx d_{mlp}$ and $n_{\text{freq}} \ll p$). This showcases the tradeoff between computational complexity and bound accuracy as set out in Gross et al. (2024). The fact that we get a valid bound (better than the naive estimate) in sub-linear time shows that our guess is very likely on the right track. However, our large bound shows that there is more to understand about how the quadrature scheme works, such as how do we align the intervals, what do the values of the ϕ_i and r_i play here, etc.

6. Discussion, interpretation, and further work

First, note the correspondence between the complexity of the argument and mechanistic understanding of the model. In the first stage, restricting each node to a single frequency (using mechanistic insight) means we only need $\mathcal{O}(p \cdot d_{mlp})$ time complexity to compute the neuron activations (the $x-y$ term can be grouped together and considered later since it is the same factor for all neurons of the same frequency). In the second stage, there is a lack of explanation of how the ReLU layer works in Zhong et al. (2023, Appendix L), where they just evaluated all the combinations in the simplified expression, to check that we indeed get $\cos(k(x+y))$ terms. Our numerical integration argument can infer about the accuracy of the scheme just by looking at the angle samples ϕ_i and ψ_i, which is $\mathcal{O}(p + d_{mlp})$ complexity. This corresponds to a better explanation of how the ReLU layer works. In particular, this argument should allow us to produce a large family of hand-coded, parameterised models which we know will perform the task well, and this family of models approximates what happens when they are trained empirically. Producing this family of models is future work.

Second, the argument presented here is not fully rigorous. We have not obtained a proof that the network as a whole indeed well-approximates numerical integration. (In fact, it is difficult even to obtain good error bounds for the 'single frequency' approximation of the neurons.) This reflects a general problem of mechanistic interpretability – a lot of the evidence is merely demonstrative or anecdotal, and something which looks 'obvious' to some human eyes may actually be impossible to prove. Thus, to have truly convincing interpretability work, we should put more effort into rigorous proofs or guarantees about the correctness of our observations and approximations.

Acknowledgements

We would like to thank the Mentorship for Alignment Research Students (MARS) program of the Cambridge AI Safety Hub (CAISH) for setting up this collaboration, and providing funding for compute and in-person research sprints. We are grateful to Euan Ong, Alex Gibson, and Soufiane Noubir, and Lawrence Chan for helpful discussions and support during the development of this project.

Impact Statement

This paper presents work whose goal is to advance the field of Mechanistic Interpretability. There are many potential

societal consequences of our work, none which we feel must be specifically highlighted here.

References

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Coppersmith, D. and Winograd, S. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990. ISSN 0747-7171. doi: https://doi.org/10.1016/S0747-7171(08)80013-2. URL https://www.sciencedirect.com/science/article/pii/S0747717108800132. Computational algebraic complexity editorial.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., Grosse, R., McCandlish, S., Kaplan, J., Amodei, D., Wattenberg, M., and Olah, C. Toy models of superposition. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/toy_model/index.html.

Gross, J., Agrawal, R., Kwa, T., Ong, E., Yip, C. H., Gibson, A., Noubir, S., and Chan, L. Compact proofs of model performance via mechanistic intepretability, June 2024. URL https://arxiv.org/abs/2406.11779.

Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., and Mueller, A. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models, 2024. URL https://arxiv.org/abs/2403.19647.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint*, 2023. doi: 10.48550/arXiv.2301.05217.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

Zhong, Z., Liu, Z., Tegmark, M., and Andreas, J. The clock and the pizza: Two stories in mechanistic explanation of neural networks, 2023.

A. Derivation of the logit expression

We start with

$$\mathrm{OV}(x)_i \approx a_i \cos(k_i x) + b_i \sin(k_i x) = a_i \cos(k_i x + \phi_i)$$

Then

$$\begin{aligned} F_1(x,y) &= \mathrm{OV}(x) + \mathrm{OV}(y) + \mathrm{Embed}(y) \\ &\approx a_i(\cos(k_i x) + \cos(k_i y)) + b_i(\sin(k_i x) + \sin(k_i y)) \\ &\quad + \mathrm{Embed}(y) \\ &= c_i(\cos(k_i x + \phi_i) + \cos(k_i y + \phi_i)) + \mathrm{Embed}(y) \end{aligned}$$

where we use the reverse direction of the cosine addition formula

$$\cos(k_i x + \phi_i) = \cos(k_i x)\cos(\phi_i) - \sin(k_i x)\sin(\phi_i)$$

We ignore the Embed term (which is the residual stream), and use the trigonometric identity

$$\cos(x) + \cos(y) = 2\cos((x+y)/2)\cos((x-y)/2)$$

to get

$$\begin{aligned} F_2(x,y) &= W_{\mathrm{in}} F_1(x,y) + b_{\mathrm{in}} \\ &\approx W_{\mathrm{in}} c_i(\cos(k_i x + \phi_i) + \cos(k_i y + \phi_i)) + b_{\mathrm{in}} \\ &\approx d_i \cos(k_i(x-y)/2)\cos(k_i(x+y)/2 + \phi_i) \end{aligned}$$

where $d_i = 2(W_{\mathrm{in}} c)_i$.

Finally, for the logit expression, using

$$(W_{\mathrm{out}})_{zi} \approx e_i \cos(k_i z + \psi_i)$$

we have

$$\begin{aligned} \mathrm{logits}(x,y) &= W_{out}\,\mathrm{ReLU}(F_2(x,y)) + b_{out} + W_U F_1(x,y) \\ &\approx W_{out}\,\mathrm{ReLU}(F_2(x,y)) \end{aligned}$$

(again ignoring the residual stream)

Then

$$\begin{aligned} &\mathrm{logits}(x,y,z) \\ &= \sum_i \mathrm{ReLU}((F_2(x,y))_i)(W_{out})_{zi} \\ &\approx \sum_i \mathrm{ReLU}(d_i \cos(k_i(x-y)/2) \\ &\cos(k_i(x+y)/2 + \phi_i))e_i \cos(k_i z + \psi_i)) \\ &= \sum_k |\cos(k(x-y)/2| \\ &\sum_i r_i\,\mathrm{ReLU}(\cos(k(x+y)/2 + \phi_i))\cos(kz + \psi_i) \\ &\approx \sum_k |\cos(k(x-y)/2)| \\ &\{\sum_i r_i\,\mathrm{ReLU}[\cos(k(x+y)/2 + \phi_i)]\cos(\psi_i)\}\cos(kz) \\ &+ \{\sum_i r_i\,\mathrm{ReLU}[\cos(k(x+y)/2 + \phi_i)]\sin(\psi_i)\}\sin(kz) \end{aligned}$$

where we sum over the 'key frequencies' k (in our example $k = 12, 18, 21, 22$), $r_i = d_i e_i$, and we use again the cosine addition formula.

B. Analysis of the 'identity' component of ReLU

Recall that we broke down ReLU into two parts,

$$\mathrm{ReLU}(x) = \frac{x}{2} + \frac{|x|}{2}$$

The integrals then split into

$$\begin{aligned} \textstyle\int_{-\pi}^{\pi} \cos(-2\phi)\frac{1}{2}\cos(\frac{k}{2} + \phi)\,\mathrm{d}\phi &= \tfrac{2}{3}\cos(k) \\ \textstyle\int_{-\pi}^{\pi} \sin(-2\phi)\frac{1}{2}\cos(\frac{k}{2} + \phi)\,\mathrm{d}\phi &= \tfrac{2}{3}\sin(k) \\ \textstyle\int_{-\pi}^{\pi} \cos(-2\phi)\frac{1}{2}|\cos(\frac{k}{2} + \phi)|\,\mathrm{d}\phi &= 0 \\ \textstyle\int_{-\pi}^{\pi} \sin(-2\phi)\frac{1}{2}|\cos(\frac{k}{2} + \phi)|\,\mathrm{d}\phi &= 0 \end{aligned}$$

We see that the 'identity' part of the ReLU yields a zero integral. So does this part of the model contribute to the logits? It turns out that the answer is yes. To resolve this issue, we look at the discrepancy between the results suggested by previous work: Zhong et al. (2023) claim that logits are of the form

$$\mathrm{logit}(x,y,z) \propto |\cos(k(x-y)/2)|\cos(k(x+y-z))$$

while Nanda et al. (2023) claim that logits are of the form

$$\mathrm{logit}(x,y,z) \propto \cos(k(x+y-z))$$

To check which is correct, we regress the logits against the factors $|\cos(k(x-y)/2)|\cos(k(x+y-z))$, which gives an R^2 of 0.86, while if we regress them against just $\cos(k(x+y-z))$, we obtain an R^2 of 0.98. So overall, Nanda et al. (2023) give a more accurate expression, but this seems to go against the analysis we did above, which led to the expression in Zhong et al. (2023). (A similar value is obtained if we just use the MLP output and drop the residual streams.) However, if we only consider the contribution to the logits from the absolute value component of ReLU, the R^2 values become 0.99 and 0.85 respectively. Therefore, although the contribution from the identity component of ReLU is small, it does make a difference towards reducing the logit dependence on $x - y$, in

particular $|\cos(k(x-y)/2)|$. This is a good thing because when $\cos(k(x-y)/2)$ is small, the logit difference between the correct logit $(x+y)$ and other logits will also be small, which will lead to a higher loss. The identity component slightly counters this effect. Then

$$W_{\text{out}}\text{OV}(x)/2 + W_{\text{out}}\text{OV}(y)/2 + W_{\text{out}}\text{embed}(y)/2$$

Thus, we can store the matrices $\text{logit_id1}[:,x] = W_{\text{out}}\text{OV}(x)/2$ and $\text{logit_id2}[:,x] = W_{\text{out}}\text{embed}(x)/2$, then we have

$$\begin{aligned} F_2(x,y)_z &= \text{residual stream} + \text{absolute value terms} \\ &\quad + \text{logit_id1}[z,x] + \text{logit_id1}[z,y] + \text{logit_id2}[z,y] \end{aligned}$$

We carry out a 2D Fourier transform to find out the decomposition of the logit_id1 and logit_id2 matrices (because W_{out} and $\text{OV}(x)$ are sparse in the (1D) Fourier basis, so their product will naturally be sparse in the 2D Fourier basis). We get $\text{logit_id1}[z,x] \approx 2\Re(\sum_k a_k e^{i(kz-2kx)})$, where the frequencies k here are the same as section 3. Hence, the output from the identity component of ReLU is (ignoring logit_id2 for now, which comes from the residual stream and is smaller): $\sum_k D_k(\cos(kz-2kx)+\cos(kz-2ky)) + E_k(\sin(kz-2kx)+\sin(kz-2ky)) = \sum_k \cos(k(y-x))(D_k\cos(k(z-x-y)) + E_k\sin(k(z-x-y)))$.

The imaginary component of the FT is very small, $c_k \approx 0$; so the contribution is $\sum_k b_k \cos(k(y-x))\cos(k(x+y-z))$.

Why does this happen, and why does it help explain the R^2 values we got above? We first list the approximate coefficients a_k:

Frequency	12	18	21	22
abs coefs (C_k)	13.9	15.1	12.1	11.2
id coefs (D_k)	-3.7	-3.9	-3.2	-3.3

Thus, the overall expression for the logits is

$$\begin{aligned} F_2(x,y)_z &\approx \textstyle\sum_k (C_k\,|\cos(k(y-x)/2)| \\ &\qquad + D_k\cos(k(y-x)))\cos(k(x+y-z)) \\ &= \textstyle\sum_k (2D_k^2\,|\cos(k(y-x)/2)|^2 \\ &\qquad + C_k\,|\cos(k(y-x)/2)|)\cos(k(x+y-z)) \\ &\qquad - D_k\cos(k(x+y-z)) \end{aligned}$$

using double angle formula. Since $D_k < 0$, the $\sum_k -D_k\cos(k(x+y-z))$ term gives some cushion for the base performance of the model (since as we discussed, the $\cos(k(x+y-z))$ term is why the model gives the highest logit when $z = x+y$). Moreover, the $2D_k^2|\cos(k(y-x)/2)|^2$ term also further improves the model since it is always non-negative. Hence, the contribution of the identity term evens out parts of the model and improves the logit difference when $|\cos(k(y-x)/2)|$ is small (where the absolute value part doesn't do well). Note that the model would work on its own if we only use the absolute value part, but since ReLU is composed of both the absolute value and identity part and the coefficients combine both parts in a way that improve model performance.

C. Further work

There are obvious problems with generalising this sort of approach to neural network interpretation. Not only does it depend on the specifics of this problem (how we can have a specific desired formula for the logits), but also it is highly labour intensive. Thus, in order for the approach to have any practical use, we need to develop automated tools to make these approximations and interpretations. For example, we may want to use the first few terms of the Fourier expansion (or other low-rank approximations) to approximate the action of various layers in a neural network, and then combine those to get algebraic expressions for certain neuron outputs of interest. Such algebraic expressions will natural admit phenomena like the numerical integration we described above. This sort of method may be particularly fruitful on problems which Fourier transforms play a large role, such as signal processing and solutions to partial differential equations.

D. Future technical work

To complete the technical work laid out in the introduction, we must accomplish two tasks which we discuss in this appendix section: constructing a parameterisation of the MLP which is checkable in less than $\mathcal{O}(p \cdot d_{mlp})$ time, and more generally constructing a parameterisation of the entire 'pizza' model that is checkable in time that is linear in the number of parameters; and establishing a bound on the error in the model's logits that does not neglect any terms.

D.1. Linear parameterisation

Constructing a parameterisation of the model which is checkable in less than $\mathcal{O}(p \cdot d_{mlp})$ time is a relatively straightforward task, given the interpretation in the body of the paper. We expect that the parameters are:

- A choice of n_{freq} frequencies k_i.
- A splitting of the neurons into groups by frequency, and an ordering of the neurons within each group.
- An assignment of widths r_i to each neuron, and an assignment of angles ϕ_i to each neuron.
- An assignment of orthogonal planes into which each frequency is embedded by the embedding matrix, and by the unembedding matrix.

- Rotations and scaling of the low-rank subset of the hidden model dimension for each of the O and V matrices.

D.2. Bounding the error of the MLP

To bound the error of our interpreation of the MLP precisely, we'd need to include a bound on the primary frequency contribution of the identity component (which integrates to 0 symbolically), and include bounds on the residual components – OVE on x and y, the MLP bias, and the embed of y, as inputs to ReLU; and UOVE on x and y and UE on y as output logits.

We could decompose every matrix in our model as a sum of the corresponding matrix from our parameterized model and a noise term. Expanding out the resulting expression for the logits (and expanding $|x + \varepsilon|$ as $|x| + (|x + \varepsilon| - |x|)$), we will have an expression which is at top-level a sum of our parameterized model result and a noise term which is expressed recursively as the difference between the actual model and the parameterized model. We can then ask two questions:

1. What worst-case bounds can we prove on the error terms at various complexities?
2. What are the empirical worst-case bounds on the relevant error terms?