Published for SISSA by Springer

Received: *February 8, 2021*
Revised: *May 20, 2022*
Accepted: *June 27, 2022*
Published: *August 8, 2022*

Neural network approximations for Calabi-Yau metrics

Vishnu Jejjala,[a] **Damián Kaloni Mayorga Peña**[a,b] **and Challenger Mishra**[c]

[a] *Mandelstam Institute for Theoretical Physics, School of Physics, NITheP, and CoE-MaSS, University of the Witwatersrand, P.O. Box WITS, Johannesburg 2050, South Africa*

[b] *Data Laboratory, Universidad de Guanajuato, Loma del Bosque No. 103, Col. Lomas del Campestre, Leon C.P. 37150, Guanajuato, Mexico*

[c] *Department of Computer Science & Technology, University of Cambridge, 15 J.J. Thomson Ave., Cambridge CB3 0FD, U.K.*

E-mail: vishnu@neo.phys.wits.ac.za, damian.mayorgapena@wits.ac.za, cm2099@cam.ac.uk

Abstract: Ricci flat metrics for Calabi-Yau threefolds are not known analytically. In this work, we employ techniques from machine learning to deduce numerical flat metrics for K3, the Fermat quintic, and the Dwork quintic. This investigation employs a simple, modular neural network architecture that is capable of approximating Ricci flat Kähler metrics for Calabi-Yau manifolds of dimensions two and three. We show that measures that assess the Ricci flatness and consistency of the metric decrease after training. This improvement is corroborated by the performance of the trained network on an independent validation set. Finally, we demonstrate the consistency of the learnt metric by showing that it is invariant under the discrete symmetries it is expected to possess.

Keywords: Differential and Algebraic Geometry, Discrete Symmetries, Superstring Vacua

ArXiv ePrint: 2012.15821

https://doi.org/10.1007/JHEP08(2022)105

Contents

JHEP08(2022)105

1 Introduction

Superstring theory supplies an architectural framework for obtaining the real world from a consistent theory of quantum gravity. In the critical setting, string theory makes the remarkable prediction that spacetime is ten dimensional. Commensurate with $\mathcal{N} = 1$ supersymmetry in four dimensions, we may use a compact Calabi-Yau threefold to reduce the theory down from ten dimensions [1, 2].

A Calabi-Yau space is an n-dimensional complex manifold $\mathcal{M}$ with a Kähler metric with local holonomy in $\mathrm{SU}(n)$. Mathematicians were already interested in these manifolds in the 1950s. Calabi conjectured [3] that given a closed $(1,1)$-form $\frac{1}{2\pi}C_1(\mathcal{M})$ representing the first Chern class of a Kähler manifold $\mathcal{M}$, there is a unique Kähler metric in the same Kähler class whose Ricci tensor is the closed $(1,1)$-form $C_1(\mathcal{M})$. This result implies the existence of a unique Ricci flat Kähler metric within each Kähler class. Unfortunately, the proof of the conjecture, viz., Yau's theorem [4, 5], does not explicitly construct the flat metric. Except for the trivial Calabi-Yau manifold, which is the even dimensional torus, we did not until recently have analytic closed form expressions for flat metrics on Calabi-Yau spaces. This situation is changing as [6–8] have tackled the problem for K3.

Luckily, topology on its own is a sufficiently powerful tool to enable string model building. Starting from the work of [9] and [10], numerous semi-realistic models of particle physics have been constructed on appropriately chosen Calabi-Yau spaces. These models are semi-realistic because, while the visible sector recapitulates the matter content and the interactions of the minimal supersymmetric Standard Model (MSSM), we do not have a detailed understanding of the Yukawa couplings or the mass hierarchies among the generations. To go beyond an analysis of the spectrum and break supersymmetry in a controlled manner, we must fix the $\mathcal{N} = 1$ Kähler potential. Doing this demands knowledge of the Ricci flat metric on the base manifold. Determining the mass of the electron from first principles therefore requires an understanding of the geometry as well as the topology of Calabi-Yau spaces.

Significant progress has been made in obtaining numerical metrics on Calabi-Yau geometries. Using the Gauss-Seidel method, Headrick and Wiseman [11] constructed numerical Ricci flat metrics on a one parameter family of K3 surfaces obtained as blowups of $T^4/\mathbb{Z}_2$. Donaldson [12, 13] subsequently developed an algorithm to solve for the metrics numerically. The essence of this approach is to consider generalizations of the Fubini-Study metric induced from the embedding of a hypersurface (or a complete intersection) in an ambient space. In particular, weighted projective spaces are endowed with a simple choice for a Kähler metric: the Fubini-Study (FS) metric. For Calabi-Yau manifolds constructed as hypersurfaces or complete intersections in (products of) weighted projective spaces, a Kähler metric can be obtained from the pullback of the ambient Fubini-Study metric onto the hypersurface or complete intersection defining the embedding. Generalizing the expression for the Kähler potential of this metric, Donaldson provides a family of so called "balanced" metrics that in a particular $k \to \infty$ limit, recovers the Ricci flat Calabi-Yau metric. This idea was then applied to finding numerical metrics on the Fermat quintic [14]. Since then, there have been a number of related numerical approaches, notably involving energy functionals [15], the Hermitian Yang-Mills equations [16–21], and general scaling properties [22].

Beginning with [23–26], modern methods in machine learning have successfully been applied to various problems in high energy theoretical physics and mathematics; see [27] for a review. For example, topological invariants in knot theory [28–30] are machine learnable. Topological properties of complete intersection Calabi-Yau threefolds have also been successfully reproduced [31–34]. Less work has been done in purely geometric directions. Ashmore, He, and Ovrut [35] used machine learning to improve the performance of Donaldson's algorithm to obtain approximate Ricci flat metrics on the Fermat quintic. In

JHEP08(2022)105

this work, we apply neural networks to obtain numerical metrics on the quintic Calabi-Yau threefold. We as well perform experiments on the Tian-Yau Calabi-Yau. Though preliminary, these experiments point the way toward a flat metric.

The organization of this paper is as follows. In section 2, we make some general remarks on Calabi-Yau spaces. In section 3, we briefly review Donaldson's algorithm and discuss measures of flatness. In section 4, we survey Ricci flow. In section 5, we introduce neural networks. In section 6, we describe our methodology. In section 7, we present our results for the torus, the K3 surface, the Fermat quintic, the Dwork family, and the Tian-Yau manifold. In section 8, we assess how well the neural network that yields the Ricci flat metric learns discrete symmetries of the Calabi-Yau. Finally, in section 9, we discuss the results and provide a prospectus for our future work.

Note added: as this work was nearing completion, two other papers by Anderson, Gerdes, Gray, Krippendorf, Raghuram, and Ruehle [36] and Douglas, Lakshminarasimhan, and Qi [37] appeared that also found numerical metrics on Calabi-Yau threefolds using machine learning techniques. Many of our methods and conclusions overlap with theirs. See also [38–41].

2 General remarks

Recall that a complex n-dimensional compact *Calabi-Yau manifold* $\mathcal{M}$ is determined by any of the following equivalent statements:

- The first real Chern class of $\mathcal{M}$ is zero.
- A positive power of the canonical bundle of $\mathcal{M}$ is trivial.
- $\mathcal{M}$ has a Kähler metric with local holonomy in $\mathrm{SU}(n)$.
- $\mathcal{M}$ admits a metric with vanishing Ricci curvature. This metric is unique in each Kähler class.

In cases where the manifold is simply connected, this is equivalent to saying that $\mathcal{M}$ possesses a unique, nowhere vanishing $(n, 0)$-form, that the positive power of the canonical bundle is the first power, and the local holonomy is global. As we are interested in finding the Ricci flat metric, we adopt the weaker definition.

As a complex Kähler manifold, the metric of $\mathcal{M}$ is a Hermitian matrix that can be derived from a Kähler potential $K(z, \bar{z})$:

$$g_{a\bar{b}} = \partial_a \partial_{\bar{b}} K(z, \bar{z})\,. \tag{2.1}$$

The metric can be used to construct the Kähler form as

$$J = \frac{\mathrm{i}}{2} g_{a\bar{b}}\, dz^a \wedge d\bar{z}^{\bar{b}}\,. \tag{2.2}$$

This is a closed $(1, 1)$-form. The corresponding Ricci tensor is given by:

$$R_{a\bar{b}} = \partial_a \partial_{\bar{b}} \log \det g\,. \tag{2.3}$$

For further details, we direct the reader to the classic references [42, 43].

JHEP08(2022)105

The Calabi-Yau spaces we will study are realized as hypersurfaces in products of projective space. The trivial example of a Calabi-Yau manifold is the torus T^{2n}. The torus T^2 can be embedded in projective space $\mathbb{P}^2$ as a cubic equation

$$T^2: \quad z_1^3 + z_2^3 + z_3^3 = 0 \subset \mathbb{P}^2\,, \tag{2.4}$$

where $[z_1 : z_2 : z_3]$ identifies a point in $\mathbb{P}^2$. Similarly, we can generalize this type of embedding to construct K3 as a quartic hypersurface in $\mathbb{P}^3$ and as well write quintic hypersurfaces in $\mathbb{P}^4$:

$$\text{K3}: \qquad z_1^4 + z_2^4 + z_3^4 + z_4^4 = 0 \subset \mathbb{P}^3\,, \tag{2.5}$$

$$\text{Fermat quintic}: \qquad z_1^5 + z_2^5 + z_3^5 + z_4^5 + z_5^5 = 0 \subset \mathbb{P}^4\,, \tag{2.6}$$

$$\text{Dwork family}: \quad z_1^5 + z_2^5 + z_3^5 + z_4^5 + z_5^5 - 5\psi z_1 z_2 z_3 z_4 z_5 = 0 \subset \mathbb{P}^4\,, \quad \psi^5 \neq 1\,. \tag{2.7}$$

Any pair of complex analytic K3 surfaces are diffeomorphic as smooth four real dimensional manifolds. The Hodge number $h^{1,1} = 20$ for K3. The quintic hypersurface has $h^{1,1} = 1$ and $h^{1,2} = 101$. We will also study the complete intersection Calabi-Yau manifold given by the configuration matrix

$$\text{Tian-Yau}: \quad \left[\begin{array}{c\|ccc} \mathbb{P}^3 & 3 & 0 & 1 \\ \mathbb{P}^3 & 0 & 3 & 1 \end{array}\right]_{\chi=-18} \qquad \Longleftrightarrow \qquad \begin{cases} \alpha^{ijk} z_i z_j z_k &= 0\,, \\ \beta^{ijk} w_i w_j w_k &= 0\,, \\ \gamma^{ij} z_i w_j &= 0\,, \end{cases} \tag{2.8}$$

where w_i are coordinates on the first $\mathbb{P}^3$ and z_i are coordinates on the second $\mathbb{P}^3$. The Tian-Yau manifold has $h^{1,1} = 14$ and $h^{1,2} = 23$. A freely acting $\mathbb{Z}_3$ quotient yields a Calabi-Yau manifold with $\chi = -6$ [44], corresponding to three generations of elementary particles; this was one of the initial testbeds for string phenomenology [45]. We will study numerical metrics on these geometries.

On the manifold $\mathbb{P}^n$, an atlas is given by $n+1$ coordinate charts on which each of the $z_i \neq 0$. In homogeneous coordinates, a possible Kähler potential for the projective space $\mathbb{P}^n$ has the form

$$K(z, \bar{z}) = \frac{1}{\pi} \log(|z|^2)\,, \qquad |z|^2 = \sum_{a=1}^{n+1} z^a \bar{z}^{\bar{a}}\,. \tag{2.9}$$

This potential leads to the following metric:

$$g_{a\bar{b}}(z, \bar{z}) = \frac{|z|^2 \delta^{a\bar{b}} - \bar{z}^a \bar{z}^{\bar{b}}}{\pi |z|^4}\,. \tag{2.10}$$

This is the *Fubini-Study* (FS) metric on $\mathbb{P}^n$. It is induced from the definition of $\mathbb{P}^n$ as a quotient space S^{2n+1}/S^1, where S^{2n+1} carries the round metric. The corresponding metric on each of the $n+1$ charts can be obtained via matrix multiplication with the Jacobian of the transformation from homogeneous to affine coordinates. In a similar fashion, the pullback of the Fubini-Study metric on $\mathbb{P}^n$ to the hypersurface induced by the embedding yields the Fubini-Study metric on the Calabi-Yau. This Fubini-Study metric is not the Ricci flat Kähler metric. In figure 1, we have plotted the Ricci scalar for the Fubini-Study metric on a section of the Fermat quintic.

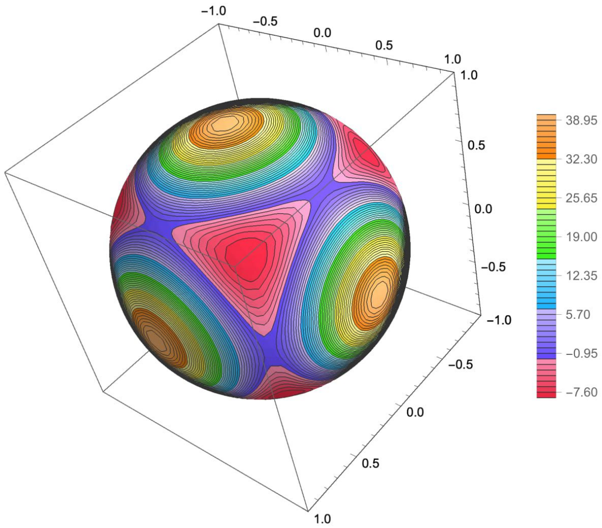

Figure 1. Visualizing the scalar curvature of the metric obtained by restricting the Fubini-Study metric of $\mathbb{P}^4$ to the quintic hypersurface. The three axes represent real parts of the three independent coordinates of the Fermat quintic (2.6) in a patch $z_i = 0$ for some i. The contour of scalar curvature is then presented atop a three dimensional sphere, and is evidently not uniform.

For $\mathbb{P}^n$, one naturally has the $n+1$ affine charts, each labeled by an index i corresponding to the coordinate that is set to one (i.e., $z^i = 1$). Additionally, given the structure of the embedding equations for T^2, K3, and the quintic, we have a natural collection of charts labeled by $z_i \neq 0$ and $z_{j \neq i}$, the latter of which is obtained from solving the equation defining the hypersurface. Therefore, each chart for the hypersurface will be labelled by two indices (i, j).

The Tian-Yau manifold is given as the intersection of three transversely intersecting loci on $\mathbb{P}^3 \times \mathbb{P}^3$. One has then two $\mathbb{C}^*$ actions and therefore two coordinates that can be set to one. For simplicity we define $z^{k+4} = w^k$, such that the index in the coordinate z runs from $1, \ldots, 8$. Taking the affine patch resulting from setting the coordinates $z^l = 1$ and $z^m = w^{m-4} = 1$, with $l \leq 4$ and $m > 4$, we can use the two cubic equations to solve for coordinates z^s and z^t with $s \leq 4$, $s \neq l$ and $t > 4$ $t \neq m$. Finally we use the bilinear equation to obtain a third dependent coordinate which can belong to either of the $\mathbb{P}^3$'s, say z^r, $r \neq l, m, s, t$. We thus have a chart in the Tian-Yau manifold defined by five indices $(l, m; s, t, r)$.

3 Donaldson's algorithm

Donaldson [12] supplied a constructive algorithm for obtaining the Ricci flat metric starting from the Fubini-Study metric. Weighted projective spaces are endowed with a simple choice of Kähler metric derived from the following Kähler potential given in equation (2.9). Another valid Kähler potential is

$$K^{(k)}(z, \bar{z}) = \frac{1}{k\pi} \log \left(h^{\alpha\bar{\beta}} s_\alpha \bar{s}_{\bar{\beta}} \right) , \tag{3.1}$$

where s_α is an element of a basis for degree k holomorphic polynomials over $\mathcal{M}$. Take N_k to be the dimension of such a basis and define

$$H_{\alpha\bar{\beta}} = \frac{N_k}{\mathrm{Vol}_\Omega} \int_{\mathcal{M}} d\mathrm{Vol}_\Omega \left(\frac{s_\alpha \bar{s}_{\bar{\beta}}}{h^{\sigma\bar{\rho}} s_\sigma \bar{s}_{\bar{\rho}}} \right) . \tag{3.2}$$

At level k, one can use $h^{\alpha\bar{\beta}} = (H_{\alpha\bar{\beta}})^{-1}$ and proceed iteratively until reaching a stable "balanced" metric $H_{\alpha\bar{\beta}}$. This is guaranteed to exist by virtue of Donaldson's theorem. Furthermore, as k increases, the metric $g_{a\bar{b}}$ obtained from the balanced metric approaches the desired Ricci flat Calabi-Yau metric. Donaldson's algorithm will provide a reference point to which we compare our results.

3.1 Flatness measures

As we are computing metrics numerically, we need some diagnostic that tells us how close we are to the flat metric. A number of such diagnostics have appeared in the literature [14, 16, 17, 19, 35]. We examine the σ-measure:

$$\sigma = \frac{1}{\mathrm{Vol}_\Omega} \int_{\mathcal{M}} d\mathrm{Vol}_\Omega \left| 1 - \frac{\mathrm{Vol}_\Omega}{\mathrm{Vol}_J} \cdot \frac{J^n}{\Omega \wedge \bar{\Omega}} \right| . \tag{3.3}$$

This makes use of the fact the volume can be computed using either the holomorphic and anti-holomorphic top forms or the closed $(1,1)$ Kähler form J:

$$\mathrm{Vol}_\Omega = \int_{\mathcal{M}} \Omega \wedge \bar{\Omega} , \qquad \mathrm{Vol}_J = \int_{\mathcal{M}} J^n , \tag{3.4}$$

with J^n denoting the n-fold wedge product of J. Note that this measure becomes zero whenever $J^n = \alpha \Omega \wedge \bar{\Omega}$, with $\alpha = \mathrm{Vol}_J \,/\, \mathrm{Vol}_{\mathrm{CY}}$.

Another measure comes from integrating the Ricci curvature scalar directly, i.e.,

$$||R|| = \frac{\mathrm{Vol}_J^{1/n}}{\mathrm{Vol}_\Omega} \int_{\mathcal{M}} d\mathrm{Vol}_J \, |R| . \tag{3.5}$$

This is known as the $||R||$-measure and in some sense it is equivalent to the sigma measure. Namely, it can be shown that as the σ-measure approaches zero, the $||R||$-measure goes to zero as well.

In order to have a Ricci flat Calabi-Yau metric, we must in addition check that it is Kähler and verify that the expressions agree on different coordinate charts in the atlas. We will refer to these conditions as the κ- and μ-measures. These are defined in (6.16) and (6.19), respectively.

4 Ricci flow

Ricci flow gives a partial differential equation for a Riemannian metric g. It was introduced by Hamilton [46] and famously employed by Perelman to prove the Poincaré conjecture in three dimensions [47–49]. See [50, 51] for surveys of the method. In high energy theory, Ricci flow was used to find numerical black hole solutions [52] and a numerical Kähler-Einstein metric on dP_3 [53], while [54, 55] discuss Ricci flow in a holographic context. Kähler-Ricci flow applies the method to a Kähler manifold. We specialize our notation to this case. The differential equation tells us that

$$\frac{\partial}{\partial\lambda} g_{a\bar{b}}(\lambda) = -\mathrm{Ric}_{a\bar{b}}(\lambda) = \frac{\partial^2}{\partial z^a \partial \bar{z}^{\bar{b}}} \log\det g\,, \tag{4.1}$$

where λ is a parameter defining a family of metrics. This differs by a factor of two from the Riemannian case [56]. We may take $g(0)$ to be the Fubini-Study metric and evolve this according to the differential equation. A fixed point of this flow is the flat metric. It turns out the Ricci flow preserves the Kähler class [57, 58]. The right hand side of (4.1) therefore provides another measure of how close we are to being Ricci flat, and the Ricci flow evolves the Fubini-Study metric to the flat metric on a Calabi-Yau.

For many purposes, it is convenient to augment the metric with a scalar function f. When we do this, the Ricci flow equation (4.1) arises from a variational principle. Ricci flow is then a gradient flow. This is suggestive because neural networks often employ gradient flow in order to accomplish deep learning. Define the Perelman functional

$$\begin{aligned}\mathcal{F}(g,f) &= \int_{\mathcal{M}} d\mu\; e^{-f}(R + |\nabla f|^2) \\ &= \int_{\mathcal{M}} dm\; (R + |\nabla f|^2)\,,\end{aligned} \tag{4.2}$$

where R is the scalar curvature. In physics language, we have introduced a dilaton f. The second equality in (4.2) recasts f in terms of the measure

$$f := \log\frac{d\mu}{dm}\,. \tag{4.3}$$

Using this definition, variation of $\mathcal{F}$ gives an equation of motion for the modified Ricci flow:

$$\frac{\partial}{\partial\lambda} g_{a\bar{b}} = -(\mathrm{Ric}_{a\bar{b}} + \nabla_a\nabla_{\bar{b}} f)\,. \tag{4.4}$$

Mapping (4.2) to a string action, the Ricci flow can be thought of as a beta function equation for the spacetime metric, which we treat as a coupling for the non-linear sigma model on the worldsheet. In this case, the fixed points are conformal and correspond to Ricci flat target spaces that satisfy the vacuum Einstein equation. Solutions to (4.4) realize Perelman's energy monotonicity condition:

$$\frac{d}{d\lambda}\mathcal{F} = 2\int_{\mathcal{M}} d\mu\; e^{-f}|\mathrm{Ric}_{a\bar{b}} + \nabla_a\nabla_{\bar{b}} f|^2 = 2\int_{\mathcal{M}} dm\; |\mathrm{Ric}_{a\bar{b}}|^2\,. \tag{4.5}$$

This implies the existence of a coupled set of partial differential equations for the metric and dilaton:

$$\frac{\partial}{\partial\lambda} g_{a\bar{b}} = -(\mathrm{Ric}_{a\bar{b}} + \nabla_a\nabla_{\bar{b}} f)\,, \tag{4.6}$$

$$\frac{\partial}{\partial\lambda} f = -\Delta f - R\,. \tag{4.7}$$

JHEP08(2022)105

The previous two equations are the same as (4.4) and (4.3). In particular, the second equation (4.7) is a backward heat equation. To efficiently address this coupled system, we must consistently solve the modified Ricci flow forward in time and the heat equation backward in time. The backward heat equation is not parabolic and therefore we have no guarantee of a solution to exist for a given initial condition $f(\lambda = 0)$. In order to solve this system, one has to bring (4.6) and (4.7) into the following form:

$$\frac{\partial}{\partial\lambda} g_{a\bar{b}} = -\mathrm{Ric}_{a\bar{b}}\,, \tag{4.8}$$

$$\frac{\partial}{\partial\lambda} f = -\Delta f + |\nabla f|^2 - R\,. \tag{4.9}$$

In this fashion, we can solve for g for $\lambda \in [0, T]$, with T such that $g(\lambda)$ is smooth in $[0, T]$. One can then use the solution for g and solve backwards in λ starting from a boundary condition $f(\lambda = T)$.

5 Neural networks

A fully connected neural network correlates an input vector to an output vector in order to approximate a true result. This correlation is highly non-linear. We can write the k-layer neural network as a function

$$\mathbf{v}_{\mathrm{out}} = f_\theta(\mathbf{v}_{\mathrm{in}}) = L_\theta^{(k)}(\sigma^{(k-1)}(\cdots L_\theta^{(2)}(\sigma^{(1)}(L_\theta^{(1)}(\mathbf{v}_{\mathrm{in}}))))) \approx \mathbf{v}_{\mathrm{true}}\,, \tag{5.1}$$

where

$$L_\theta^{(m)}(\mathbf{v}) = W_\theta^{(m)} \cdot \mathbf{v} + \mathbf{b}_\theta^{(m)}\,. \tag{5.2}$$

The $\mathbf{b}_\theta^{(m)}$ is called a bias vector, and $W_\theta^{(m)}$ is called a weight matrix. The superscript denotes that we are in the m-th hidden layer of the neural network. If $W_\theta^{(m)}$ is an $n_m \times n_{m-1}$ matrix, there are n_m neurons in this layer. In choosing the architecture, we specify k and n_m for $m = 1, \ldots, k$ as well as the functions $\sigma^{(m)}$. The elements of the bias vectors and the weight matrices are collectively termed hyperparameters, which we label by the subscript θ. These are fixed in training. The non-linearity acts elementwise. Three standard choices are

$$\begin{aligned} \text{logistic sigmoid}: &\qquad \sigma(x) = \frac{1}{1+e^{-x}}\,, \\ \text{ReLU}: &\qquad \sigma(x) = x\,\Theta(x)\,, \\ \tanh: &\qquad \sigma(x) = \tanh x\,. \end{aligned} \tag{5.3}$$

The training is executed by minimizing a specified loss function on the training set. Optimization of the hyperparameters is accomplished, for example, by stochastic gradient descent (SGD) or adaptive moment estimation (Adam). In fixing the hyperparameters, we pass the entire training set through the neural network multiple times; each time we do this is called an epoch. Because the datasets are large, each epoch is reached by splitting the training set into several batches. The number of epochs and the size of each batch are therefore parameters that enter the training. Validation is performed by testing the

JHEP08(2022)105

trained neural network on inputs unseen during training. The universal approximation theorem [59, 60] states that, with mild assumptions, a feedforward neural network with a single hidden layer and a finite number of neurons can approximate continuous functions on compact subsets of $\mathbb{R}^n$. The performance of the neural network is gauged by whether a distance function $d(\mathbf{v}_{\text{out}}, \mathbf{v}_{\text{true}})$ is sufficiently small when this is suitably averaged over a dataset.

6 Methodology for numerical Calabi-Yau metrics

6.1 Generation of points

As inputs we take the affine coordinates describing various points in the manifold of interest. For the computation of the numerical volumes and integrals in general, it is necessary to work with a uniform distribution of points. For this purpose, the following method has been employed. The same method was used in [17, 35]. The general philosophy for Calabi-Yaus defined by degree $n+1$ hypersurfaces in $\mathbb{P}^n$ is based on the fact that lines in $\mathbb{P}^n$ are uniformly distributed with respect to the $\mathrm{SU}(n+1)$ symmetry of the Fubini-Study metric. Therefore, sampling the manifold with points at the intersection of each line with the hypersurface permits us to evaluate numerical integrations in a straightforward manner, taking the Fubini-Study metric as a measure of point distribution. The point selection process proceeds as follows.

- First we generate a real vector $\mathbf{v}$ of random entries $-1 \leq v_i \leq 1$, $i = 1 \ldots 2n$. We only keep those vectors satisfying $|v| \leq 1$.
- Project the points to the hypersphere S^{2n-1} by setting $\hat{v} = v/|v|$. Furthermore, use $\hat{v}$ to build a point in $P \in \mathbb{P}^n$:

$$P = [\hat{v}_1 + \mathrm{i}\hat{v}_2 : \ldots : \hat{v}_{2n-1} + \mathrm{i}\hat{v}_{2n}] \,. \tag{6.1}$$

- Build a line $L_{kj} = \{P_k + \lambda P_j \,|\, \lambda \in \mathbb{C}\}$ using two points p_k and p_j constructed in the manner highlighted above.
- For each line, one takes the points $\{p_l\}$ resulting from the intersection of the line with the hypersurface. As they arise from random lines uniformly distributed with respect to the $\mathrm{SU}(n+1)$ symmetry of the hypersphere, they are uniformly distributed with respect to the Fubini-Study metric on the hypersurface.

A point $p = [z_1 : \ldots : z_{n+1}] \in \mathbb{P}^n$ can be described in affine patches. If the coordinate $z_m \neq 0$, then we can define the affine coordinates in the m-th patch as

$$x_r^m = \begin{cases} z_r/z_m \,, & r < m \,, \\ z_{r+1}/z_m \,, & r > m \,. \end{cases} \tag{6.2}$$

There is a preferred presentation for any given point. Assume that for the point p we have $\max_i(|z_i|^2) = |z_m|^2$. We therefore know that in the m-th patch, the point p is then described by the coordinates,

$$x^m = (x_1^m, x_2^m \dots, x_n^m) \in D^n \,, \quad |x_n^m| \leq 1 \tag{6.3}$$

with D a unitary disk in $\mathbb{C}$. Note that in all other patches, the presentation of the same point will lie outside of the corresponding polydisk D^n [22]. Furthermore, the hypersurface equation permits us to get rid of one coordinate that we denote the dependent coordinate. In principle, one is allowed to choose which coordinate to take as the independent one. For matters of numerical stability [35], it is recommended that we take as the dependent coordinate the one for which $|\partial Q/\partial x_r^m|$ is the maximum, with Q being the corresponding hypersurface equation. Assume that for the point p, this happens for the affine coordinate x_l^m. For the entire manifold, we then split its points into different patches $S^{(m,k)}$, with $k = l$ if $l < m$ or $k = l + 1$ if $l > m$. Note that even though, there is a preferred patch for each point, in principle it has a presentation in all of the other patches, provided none of its homogeneous coordinates is zero. Recall that all the hypersurface equations considered in this work remain invariant under permutation of the coordinates. For this reason we expect the local metric expressions over the different patches to be identical.

A slight modification of the point selection procedure has to be implemented for the complete intersection Tian-Yau manifold. For simplicity we take (2.8) to have the following form

$$Q_1 = w_1^3 + w_2^3 + w_3^3 + w_4^3 = 0 \,, \tag{6.4}$$

$$Q_2 = z_1^3 + z_2^3 + z_3^3 + z_4^3 = 0 \,, \tag{6.5}$$

$$Q_3 = w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 = 0 \,. \tag{6.6}$$

For the Tian-Yau the selection of points start with a random set of points in $\mathbb{P}^3$ exactly as done in (6.1).

- We construct a line $L_{kj} = \{P_k + \lambda P_j \,|\, \lambda \in \mathbb{C}\}$ and a plane $P_{lmn} = \{P_l + \alpha P_m + \beta P_n \,|\, \alpha, \beta \in \mathbb{C}\}$. Take the line to belong to the first $\mathbb{P}^3$ and the plane to belong to the second $\mathbb{P}^3$.
- We then find all points in $L_{kj} \times P_{lmn}$ that satisfy the defining equations of the Tian-Yau manifold.[1]

[1]This sampling procedure closely follows [17] invoking the distribution of zeroes of random sections of a given line bundle on the manifold $\mathcal{M}$ [61]. For the Tian-Yau manifold, we start sampling points from unitary spheres in each of the $\mathbb{P}^3$ ambient spaces. This is consistent with the projections:

$$\Phi_{\mathcal{O}_{\mathcal{M}}(1,0)} = \pi_1 |_{\mathcal{M}} \,, \quad \Phi_{\mathcal{O}_{\mathcal{M}}(0,1)} = \pi_2 |_{\mathcal{M}} \,.$$

where $O_{\mathcal{M}}(1,0)$ and $\mathcal{O}_{\mathcal{M}}(0,1)$ are the line bundles dual to the pullback of the hyperplane class in the first and second $\mathbb{P}^3$s, respectively. Points sampled in this manner will be uniformly distributed with respect to the measure

$$dA \sim \pi_1^*(\omega_{\mathbb{P}_1^3}) \wedge \pi_2^*(\omega_{\mathbb{P}_2^3}) \wedge \pi_2^*(\omega_{\mathbb{P}_2^3}) \,|_{\mathcal{M}} \,,$$

as the sampling takes lines from the first $\mathbb{P}^3$ and points from the second one. In an earlier version, we considered increasing the number of points by permuting points from the first and second $\mathbb{P}^3$ factors. As emphasized by the referee, this is not correct as the density dA is not invariant under this procedure.

Since lines and planes are both uniformly distributed over the hypersphere S^7 it is safe to assume that the points generated in this manner are uniformly distributed with respect to the symmetry of the Fubini-Study metric of the ambient space. Note that the set of transversely intersecting equations provides three dependent coordinates. Therefore the patches in this case will be described by two coordinates set to one using the $\mathbb{C}^*$ actions in the ambient space plus three dependent coordinates obtained from solving the Tian-Yau system of equations.

In contrast to the hypersurfaces describing the Torus, the K3 surface and the Dwork family, we can notice that arbitrary coordinate permutations do not necessarily leave (6.4)–(6.6) invariant. Therefore, not all local metrics are equivalent up to permutations. In order to work out the matching of patches, we need to follow a different approach. For the Tian-Yau manifold, we consider only training with the σ-measure as a loss, as it can be considered locally on a single patch. In a forthcoming work we expect to consider a more global training process for this manifold in which we account for the matching of metrics on patch overlaps as well as deviations of the metric from being Kähler.

Now, let us briefly sketch how numerical integration proceeds. For simplicity, we focus on the hypersurfaces, but the same results can be straightforwardly extrapolated to the complete intersection case. Taking points in their corresponding preferred patch $S^{(m,l)}$ makes the numerical integration over $\mathcal{M}$ relatively feasible. Picking the preferred presentation for each of the points makes the different patches to be almost disjoint. (The choice is ambiguous for points at the boundary regions. However, this does not affect the general integration process as the intersections constitute a set of measure zero.) Assume that we want to obtain numerical estimates for the volume of $\mathcal{M}$, using the sample points generated with the method previously illustrated. This volume is given by the expression

$$\mathrm{Vol}_\Omega = \int_{\mathcal{M}} d\,\mathrm{Vol}_\Omega = \int_{\mathcal{M}} d\,\mathrm{Vol}_{\mathrm{FS}} \left(\frac{d\,\mathrm{Vol}_\Omega}{d\,\mathrm{Vol}_{\mathrm{FS}}} \right) . \tag{6.7}$$

As the set of points of interest are uniformly distributed with respect to the Fubini-Study metric and $d\,\mathrm{Vol}_{\mathrm{FS}}$ is the Fubini-Study differential volume, we can numerically approximate the volume as generated with the method previously illustrated. This volume is given by the expression

$$\mathrm{Vol}_\Omega = \frac{1}{N} \sum_{l=1}^{N} w_M(p_l) \,, \quad w_M(p_l) = \frac{d\,\mathrm{Vol}_\Omega}{d\,\mathrm{Vol}_{\mathrm{FS}}} \,, \tag{6.8}$$

where N is the number of points under consideration. Similarly, any integral over $\mathcal{M}$ can be evaluated numerically in the following manner:

$$\int_{\mathcal{M}} d\,\mathrm{Vol}_\Omega \;\; f(z) = \frac{1}{N} \sum_{l=1}^{N} f(p_l) w_M(p_l) \,. \tag{6.9}$$

6.2 Neural network architecture

Our goal is to approximate the Hermitian Calabi-Yau metric from the output of a neural network. As an $n \times n$ Hermitian matrix, the metric g can be parameterized in terms of the following product

$$g = L \cdot D \cdot L^\dagger \,, \tag{6.10}$$

JHEP08(2022)105

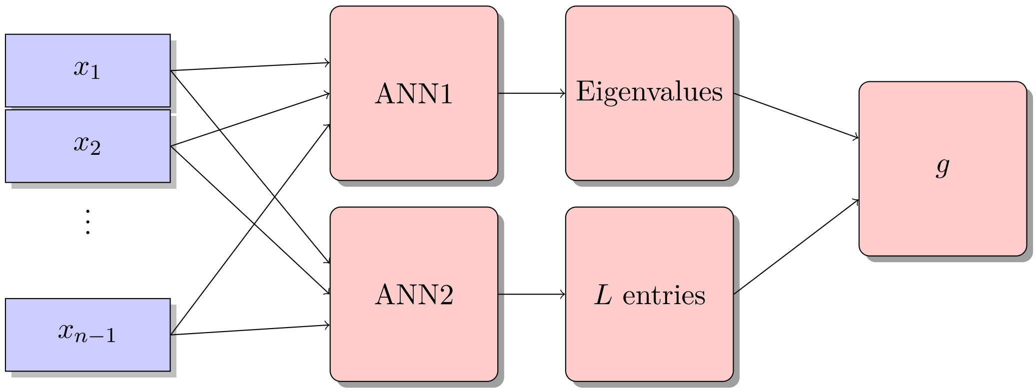

Figure 2. Flow chart of the algorithm. Two separate neural networks provide the eigenvalues as well as the entries of the diagonal matrix L. They get combined into the metric g that is used to minimize the combined loss function for both networks simultaneously.

where L is a lower triangular matrix with ones along the diagonal and D is a diagonal matrix with real entries. L contains $n(n-1)$ real parameters, and D must consist of n real and positive numbers. This is a variant of the classical Cholesky decomposition. Since we want the metric to be generated from neural network outputs, we employ two neural networks for this process. The first artificial neural network (ANN1) produces n outputs $o_1^{(1)}, \ldots, o_n^{(1)}$ that will serve to construct the matrix D. *A priori* these outputs need not to be positive, and for this reason we construct D with the output exponentials or squares

$$D = \mathrm{diag}(e^{o_1^{(1)}}, \ldots, e^{o_n^{(1)}}), \quad \text{or,} \;\; D = \mathrm{diag}((o_1^{(1)})^2, \ldots, (o_n^{(1)})^2)\,. \tag{6.11}$$

The second artificial neural network (ANN2) outputs $o_i^{(2)}$ are combined into the entries of the matrix L. Figure 2 displays the architecture of the neural network. For example, in the K3 case the metric L takes the form

$$L = \begin{pmatrix} 1 & 0 \\ o_1^{(2)} + \mathrm{i} o_2^{(2)} & 1 \end{pmatrix}\,. \tag{6.12}$$

The best architecture found for our purposes is to take ANN1 and ANN2 to be multi-layer perceptron (MLP) neural networks with three hidden layers. In most experiments, each hidden layer has 500 neurons. No specific initialization was used. We use ADAM optimization. All experiments were implemented in `PyTorch`. Because different hyperparameters are more successful in different experiments, we will quote these when we present our results.

To justify this choice of architecture, let us make some observations. The idea behind this separation has to do with the different structure of the data and the factorization itself. Since we want the diagonal entries to be positive, we pass them though an exponential. For the other neural network we have to pair the outputs such that we form the off diagonal entries of the matrix L. The σ-measure depends only upon the eigenvalues. As such, the modularity of the neural network enables us to highlight this fact in training and, taking advantage of the LDL decomposition, nudge the performance appropriately. Moreover, because we can switch off the neural network corresponding to the matrix L, we can independently extract the determinant.

6.3 Loss functions

As we have already noted in section 3.1, the σ- and $||R||$-measures are positive and bounded quantities that measure how close the metric approximation is to the Ricci flat metric. The σ-measure has the advantage that we do not need to take derivatives of the neural network outputs.

Additionally, we must take into account two other properties, in the case of the neural network the metric is not directly obtained from a Kähler potential. The Kähler property of the metric has to be checked, and this is guaranteed when the Kähler form is closed:

$$dJ = 0 \qquad \Longrightarrow \qquad \partial_a g_{b\bar{c}} - \partial_b g_{a\bar{c}} = 0\,. \tag{6.13}$$

Let us then define the quantity

$$k_{ab\bar{c}} = \partial_a g_{b\bar{c}} - \partial_b g_{a\bar{c}}\,. \tag{6.14}$$

We can define the Frobenius norm of k as follows

$$|k|^2 = \sum_{a,b,\bar{c}} |k_{ab\bar{c}}|^2\,. \tag{6.15}$$

From this, we define the κ-measure:

$$\kappa = \frac{\mathrm{Vol}_J^{1/n}}{\mathrm{Vol}_\Omega} \int_{\mathcal{M}} d\,\mathrm{Vol}_J \;\; |k|^2\,. \tag{6.16}$$

An additional property to check is that the boundary conditions for the metric are satisfied. Take a point p that has a presentation in the patch $S^{(m,l)}$ as well as in the patch $S^{(m',l')}$. Assume that the metric in the first patch is described by $g^{(m,l)}$ and by $g^{(m',l')}$ on the second patch. Then at point p,

$$g^{(m,l)} = J^{(m,l)}_{(m',l')} g^{(m',l')} \bar{J}^{(m,l)}_{(m',l')}\,. \tag{6.17}$$

where $J^{(m,l)}_{(m',l')}$ are the Jacobians of transformation between coordinate patches. Now define the matrix

$$M(m',l';m,l) = g^{(m,l)} - J^{(m,l)}_{(m',l')} g^{(m',l')} \bar{J}^{(m,l)}_{(m',l')}\,, \tag{6.18}$$

and define the μ-measure

$$\mu = \frac{1}{N_p!} \sum_{m',l'} \sum_{m.l \neq m',l'} \frac{1}{\mathrm{Vol}_\Omega} \int_{\mathcal{M}} d\,\mathrm{Vol}_J \;\; |M(m',l';m,l)|^2\,, \tag{6.19}$$

where $|M(m',l';m,l)|$ denotes the Frobenius norm and N_p is the number of patches.

A good measure of how close the metric approximation is to the actual flat metric can be constructed in the following manner:

$$\mathrm{Loss} = \alpha_\sigma \sigma + \alpha_\kappa \kappa + \alpha_\mu \mu\,, \tag{6.20}$$

JHEP08(2022)105

with α_σ, α_κ and α_μ are real positive coefficients that ensure the three quantities of interest make commensurate contributions to the loss function.

In this paper, we primarily focus on the torus, K3, and the quintic. These manifolds have a fixed Kähler class. If $h^{1,1} > 1$ for a Calabi-Yau threefold, we must ensure that the numerical metric we obtain lies in the same Kähler class as the seed, which may be the Fubini-Study metric. We can impose this by adding an extra term in the loss function that checks that we remain in the same Kähler class by integrating the Kähler form. Alternatively, we can employ methodology that preserves the Kähler class by working directly with the Kähler potential at the expense of calculating derivatives to obtain the metric, which admittedly is computationally expensive, or using Kähler-Ricci flow. While our investigation of the Tian-Yau manifold comprises the first experiments on a Calabi-Yau geometry with $h^{1,1} > 1$, we do not obtain a flat metric because the machine learning does not apply the full loss function.

6.4 Ricci flow with a neural network

One alternative approach to obtaining the Ricci flat metric of a Calabi-Yau manifold would be to follow the Ricci flow starting with the Fubini-Study metric (in case such manifold is embedded as a hypersurface, or complete intersection in a weighted projective space). This contrasts with the general approach of constructing a functional for which the flat metric can be obtained as a minimum for such a functional. This more direct implementation poses interesting challenges. In the first place one has to ensure that numerical errors do not propagate as one evolves the flow in the parameter λ, potentially driving us away from the desired flat metric.

As a proof of principle, we have applied this method to learn the flat metric on the square torus treated as a real manifold. We take the domain $x \in [0, 2\pi)$ and $y \in [0, 2\pi)$. We take the metric coming from the embedding into $\mathbb{R}^3$ given by

$$g(\lambda = 0) = \begin{pmatrix} (c + a\,\cos y)^2 & 0 \\ 0 & a^2 \end{pmatrix} \tag{6.21}$$

with a and c being the corresponding torus radii. We present this in figure 3. The approach here is slightly different as we employ Hamilton's Ricci flow formalism to flow the metric (6.21) to the Ricci flat metric. The evolution of the Ricci scalar as function of the y parameter is obtained using a numerical partial differential equation solver in `Mathematica`.

There are several challenges to overcome in the implementation of real Ricci flow on Riemannian manifolds. One such challenge is the likely appearance of curvature singularities at finite time as one evolves the flow. This is a typical feature due to the non linear nature of the Ricci flow equation. Additionally, due to gauge fixing, coordinate singularities might appear as well. Complex and especially Kähler manifolds permit us to deal with this problem in a simpler form, as the metric can be described in terms of a single function. In itself, Kähler-Ricci flow is more robust as convergence is proven [62]. Also, for Kähler-Ricci flow on Calabi-Yau manifolds one generically does not expect the appearance of coordinate singularities.

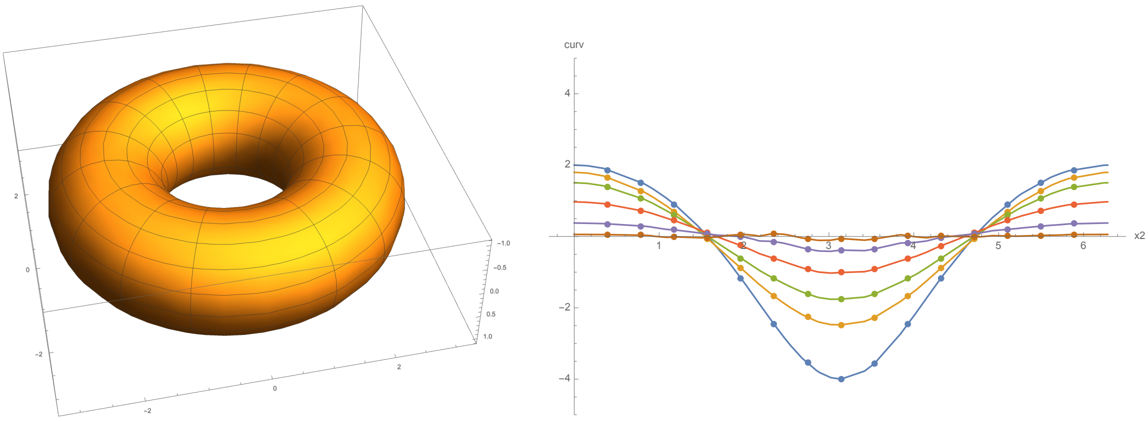

Figure 3. The scalar curvature of the real torus on the left is plotted as a function of the angular coordinate of the torus. By solving the Ricci flow equations we see a gradual shift to a metric of approximately zero scalar curvature.

A preliminary avenue to explore the flow would be to consider a neural network approximation of the metric. For simplicity, instead of considering gradient flow for the Perelman loss, we can consider solving the differential equation taking the neural network to approximate the solution for the metric g. In that case we do not need to worry about the dilaton and take the flow equation to be

$$\partial_\lambda g_{a\bar{b}} = -R_{a\bar{b}} \,. \tag{6.22}$$

In order to set the boundary conditions, we start with a given known metric (e.g., the Fubini-Study metric) at $\lambda = 0$:

$$g(\lambda = 0) = g_{\text{FS}} \,. \tag{6.23}$$

On the neural network side, this means that we start by training the network to reproduce the Fubini-Study metric. We can then update the data at $\lambda = \Delta\lambda$ by computing the Ricci tensor $R_{a\bar{b}}$ on the neural network approximation and then train the network at $\lambda = \Delta\lambda$ with the following data:

$$g_{a\bar{b}}(\lambda = \Delta\lambda) = g_{a\bar{b}}(0) - R_{a\bar{b}}(0)\Delta\lambda \,, \tag{6.24}$$

with $R_{a\bar{b}}$ being the neural network approximation for the Ricci tensor. As such we are required to approximate not simply the metric, but also the corresponding Ricci tensor as accurately as possible. This requires a learning paradigm where derivative observations are available, synthetically generated or otherwise. Therefore, we would need to compute second derivatives of the network approximating the flat metric. Standard autograd tools in widely used machine learning packages (e.g., `PyTorch`, `TensorFlow`) allow fast gradient computations of neural networks. It has been noted in the machine learning literature that derivative observations can improve predictors as well as generalization. This has been demonstrated by the use of both Bayesian and non-Bayesian tools. In [63], for example, it was shown that derivative observations can improve the predictive power of Gaussian processes. It was recently shown in [64] that the same holds for neural networks. The

JHEP08(2022)105

authors proposed a new training paradigm: *Sobolev training*, wherein the loss function is constructed out of observations of the function values as well as derivatives, up to some order. Additionally, there are theoretical guarantees for existence of networks (with ReLU or leaky ReLU activations) that can approximate a function, with the network's derivatives approximating the function's derivatives. A consequence of Sobolev training is that it has lower sample complexity than regular training.

This learning paradigm is directly applicable to our situation, where both the metric and its derivatives (connections, Ricci tensor) have geometric meanings. The distinction from the method proposed in [64] is that the values of the metric or its derivatives are not readily available, except at the beginning of the flow. As such, we propose to generate this data at any step of the flow, from the network approximating the metric at the previous step, à la (6.24). We use the following dynamic loss function:

$$\mathrm{Loss}(\lambda) = \alpha_0 \mathrm{MSE}(g_{NN}(\lambda), g(\lambda)) + \alpha_1 \mathrm{MSE}(\nabla g_{NN}(\lambda), \nabla g(\lambda)) + \alpha_2 \mathrm{MSE}(\nabla^2 g_{NN}(\lambda), \nabla^2 g(\lambda))\,, \tag{6.25}$$

where λ is the flow parameter, MSE denotes the mean square error, $g_{NN}(\lambda)$ is the approximation metric and $g(\lambda)$ is the target metric, when the flow parameter is λ. The parameters α_0, α_1, and α_2 are weights set to ensure that the different mean squared error losses are of the same order.

While we have preliminary results for the complex torus and K3, the conclusions we may draw from the implementation are so far only tentative. Part of the issue here is in designing a loss function that efficiently localizes to the solution of a coupled set of partial differential equations, (4.6) and (4.7). The results for T^2, K3, and Calabi-Yau threefolds will be reported in forthcoming work.

7 Results

Overview. In this section, we document the results of our machine learning experiments in modelling Ricci flat metrics on Calabi-Yau manifolds. We consider the complex torus in one dimension, the quartic K3 surface, the Fermat quintic, a second member of the Dwork family of quintics, and the Tian-Yau manifold. For these geometries, we model the flat metric by a neural network whose architecture was chosen keeping simplicity in mind. Since neural network are essentially black boxes, the simplicity of the network is beneficial for characterizing what is happening. Crucially, we do not have a sufficiently comprehensive understanding of how the machine learns in order to go beyond numerics. Indeed, this may not be possible to achieve.

In order to optimize performance of the neural network, we have scanned over hyperparameters. In most experiments, we use a three layer neural network with 500 neurons in each layer. In selecting this architecture, we have studied neural networks with 100, 300, 500, and 700 neurons per layer. We have also used different learning rates (10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}), settling primarily on 10^{-2} and 10^{-3}. We have employed batch sizes of 500, 1000, and 5000 in training. Finally, we find that among tanh, ReLU, and logistic sigmoid, the tanh activation function works best for our problem. This is of course not a complete scan over the space of hyperparameters. It is, however, sufficient for us to identify combinations of the parameters that lead to good numerical approximations for the Ricci flat Calabi-Yau metric.[2]

7.1 The torus

Consider the torus $T^2 = \mathbb{C}/\Lambda$ with $\Lambda = \langle 1, \tau \rangle_{\mathbb{Z}}$. A general point on the torus can be described by the complex coordinate $z \in \mathbb{C}$, with the identification $z \sim z + \lambda$ for any $\lambda \in \Lambda$. In this way the metric on the torus inherited from $\mathbb{C}$ is

$$ds^2 = dz\, d\bar{z}\,, \tag{7.1}$$

which is obviously Ricci flat. By contrast, treating $T^2 \simeq S^1 \times S^1 \subset \mathbb{R}^3$, the induced metric from the embedding is not Ricci flat.

As in (2.4), consider the torus T^2 defined by a ternary cubic equation in $\mathbb{P}^2$:

$$z_1^3 + z_2^3 + z_3^3 = 0\,, \tag{7.2}$$

with homogeneous coordinates $[z_1 : z_2 : z_3]$.

We can consider the Fubini-Study metric in $\mathbb{P}^2$ restricted to the torus. In particular, let us consider the patch where $z_3 = 1$ and take z_2 to be a function of the coordinate z_1. In terms of the patching scheme discussed in section 6.1, this is the $(3, 2)$ patch of the torus. There are six patches in total. In figure 4, we have depicted the $(3, 2)$ patch corresponding to the complex plane spanned by the complex coordinate z_1. We have included the overlapping regions with the other five patches as well as the distribution of points generated using the hypersphere method. As expected from the symmetry, one gets roughly the same number of points in every preferred patch. Note that for each value of z_1, one generically obtains three different values for z_2, which lie in the torus defined by (7.2). However, one can concentrate on one of the Riemann sheets, particularly since the values of the metric at each of the three roots coincide.

The Fubini-Study metric in the patch under consideration reads

$$ds^2 = \left(\frac{(1 + |z_1|^4 + |z_2|^4)}{\pi |z_2|^4 (1 + |z_1|^2 + |z_2|^2)^2} \right) dz_1 d\bar{z}_1\,, \tag{7.3}$$

with $z_2 = (-1 - z_1^3)^{1/3}$. The Ricci flat metric can be obtained from the following relation

$$J \sim \Omega \wedge \bar{\Omega} = \frac{dz_1 \wedge d\bar{z}_1}{9|z_2|^4}\,. \tag{7.4}$$

The top holomorphic form Ω on a torus is a $(1, 0)$ form; J is the Kähler form. For the metric, we consider a Kähler ansatz of the form

$$J' = \frac{(\alpha_1 + \alpha_2 |z_1|^4 + \alpha_3 |z_2|^4) dz_1 \wedge d\bar{z}_1}{\pi |z_2|^4 (\beta_1 + \beta_2 |z_1|^2 + \beta_3 |z_2|^2)^2}\,. \tag{7.5}$$

[2]In addition to a 2.6 GHz 6 core Intel i7 processor, we used a Helvetios cluster, each node of which has two Skylake processors running at 2.3 GHz with 18 cores each. We thank Rahul Sharma for access to the latter resource.

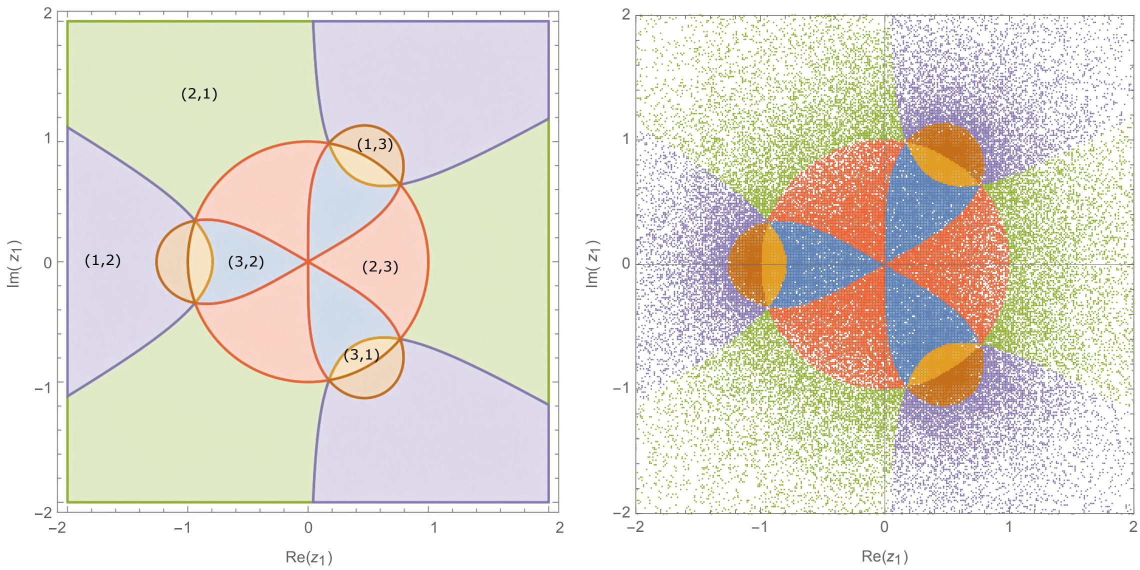

Figure 4. The torus can be viewed as three Riemann sheets glued together through branch cuts stretching between the cubic roots of -1 and the point at infinity. Three copies of the complex plane span the chart $(3, 2)$. The different regions in color correspond to the intersection with other patches. In the plot on the right, we present the distribution of sample points following the method indicated in section 6.1.

In order to approach the flat metric starting from this ansatz, we devise an algorithm that runs over the space of real parameters $\{\alpha_i, \beta_i\}$ in order to minimize the norm squared of the Ricci tensor. The ansatz (7.5) is overparameterized; depending on the initialization conditions for the algorithm we would obtain any of the possible solutions leading to the flat metric. Indeed, we find that the optimal parameters are of the order of one part in 10^7 with the exception of $\alpha_2 = 654.92$ and $\beta_2 = 378.08$. Therefore the metric obtained is given by (7.4) up to an overall coefficient. This is illustrated in figure 5. Notice that α_1, β_1 constant with the other coefficients zero leads to the same metric, so the solution is by no means unique.

7.2 The K3 surface

For the K3 surface, we consider the hypersurface equation (2.5) in $\mathbb{P}^3$. We have in total 12 (m, l) patches for the K3 hypersurface. To evaluate the ability of the network to generalize to unseen data, we have divided the set of points into a training and a test set, which is half the size of the training set. The results are summarized in figure 6. A weighted combination of the σ, μ, and κ-measures specifies the loss function. After considering various activation functions we observed that the best results are obtained for a tanh activation function. Epoch by epoch, we see that the losses all decrease. As well, we see good agreement between the training and test datasets with as few as 1000 points for training. In the case of the K3, the best σ values are obtained for the tanh activation function, for which we find a final σ of 0.060 for training and 0.068 for the test set. This is to be contrasted with a value of $\sigma = 1.37$ obtained for the Fubini-Study metric.

JHEP08(2022)105

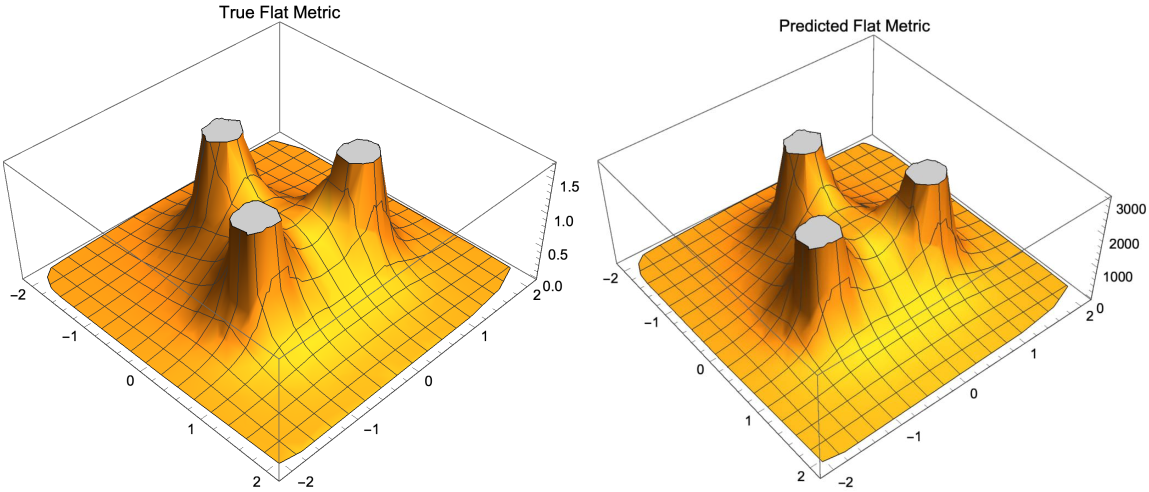

Figure 5. True and predicted Ricci flat metrics on the complex torus (7.2). The axes are given by the real and imaginary parts of the independent coordinate z_1. The three peaks appear at points $z_1^3 = -1$ for which the coordinate $z_2 = 0$. Note that these apparent singularities for the metric are not there if one takes z_2 as the dependent coordinate for those points. The predicted metric differs from the true metric by an overall scaling thus yielding a vanishing Ricci curvature.

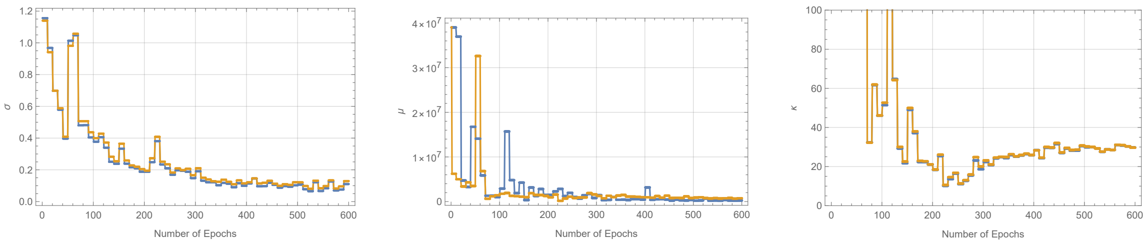

Figure 6. Evolution of the measures during the training process for the quartic K3 surface (2.5). In this case, we have used a tanh activation function. The measures on the training set are depicted blue and on the test set are depicted yellow. The training is performed with a loss function that includes all three measures.

7.3 The Fermat quintic

The Fermat quintic is described by (2.6). In this case we have 20 different (m, l) patches. For test and training sets of 10,000 points, we train employing a weighted combination of the three measures σ, μ and κ. In figure 7, we illustrate the sampling of the points used in the analysis. Upon training, we have once again observed that a tanh activation function gives the best outcomes. Employing this activation function we obtain a σ of 0.15 and 0.18 for training and test sets, respectively. These values can be compared to $\sigma = 0.49$ corresponding to the Fubini-Study metric.

As in the K3 case, we observe that all losses decrease during the training process. We also observe that the μ and the κ measures track each other. This turns out to be a ubiquitous empirical feature that we encounter consistently in our experiments. We have also trained on a particular loss and monitored the behavior of the other two. The results

JHEP08(2022)105

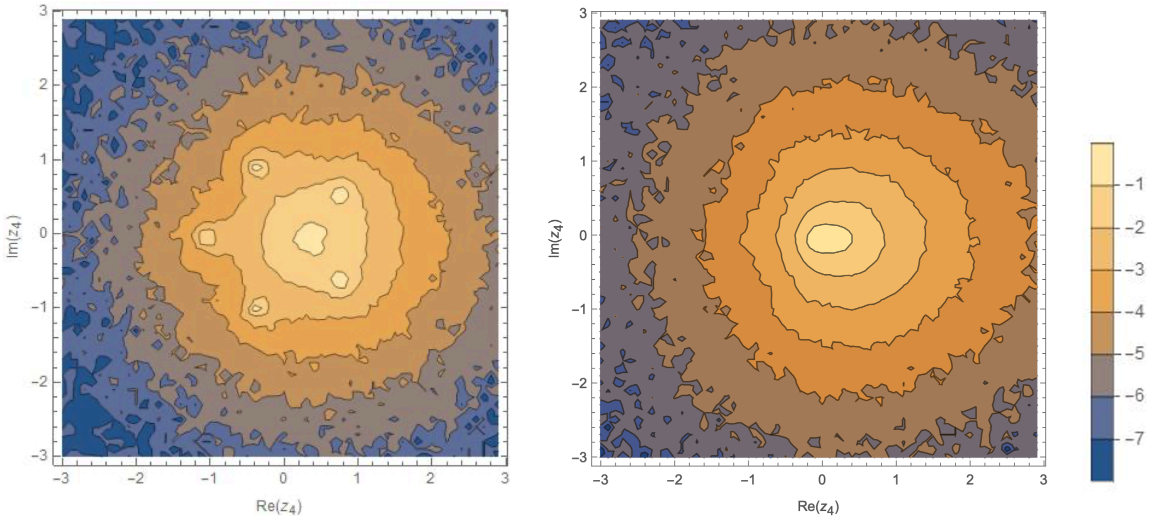

Figure 7. Distribution of log density points for the Fermat quintic (left) and Dwork quintic ($\psi = -\frac{1}{5}$, right), the points are projected to one complex plane along the coordinate z_4.

are shown in figure 8. There we observe that improvements in flatness (training with σ only) lead to an increase in the Kaehler and patch matching loss. Similarly, an improvent in either μ or κ does not automatically lead to an improved σ.

Additionally, we explored the behavior of the μ- and κ-measures for the quintic. The training process with non-zero weights α_σ, α_κ and α_μ in the loss function (6.20) is more involved as the matrix products in `PyTorch` have to be converted to real numbers. This reflects in longer computation times. For this reason, instead of training over 20,000 epochs as in the training with σ-measure only, we have chosen to train only during 400 epochs. The hyperparameters α_σ, α_κ, and α_μ are chosen from normalization of the loss values obtained in the first training epoch. For our case, we have chosen $\alpha_\sigma = 1$, $\alpha_\kappa = 1.25 \times 10^{-4}$, and $\alpha_\mu = 3.16 \times 10^{-7}$. In figure 9, we have included the results for the various loss functions using the same test and training points as in the previous experiments. We observe that as the network training evolves, we see a decrease of two orders of magnitude in the κ-measure and three orders of magnitude in the μ-measure.

7.4 The Dwork family

The neural network approximation can be extended to any member of the Dwork family given by (2.7). For this experiment, we take $\psi = -1/5$. Our results are summarized in figure 10. In this case we obtain the best σ for the tanh activation function. After training, the σ-measure drops to 0.0015 on the training set and to 0.28 on the test set. We also see that the ReLU activation function gives a σ of 0.30 on the training set and 0.31 on the test set. The value of σ for the Fubini-Study metric in this case is 0.50.

7.5 The Tian-Yau manifold

For the Tian-Yau manifold (2.8), we train and test with points in a single patch. We choose to work with defining equations (6.4)–(6.6). This is equivalent to making an explicit choice

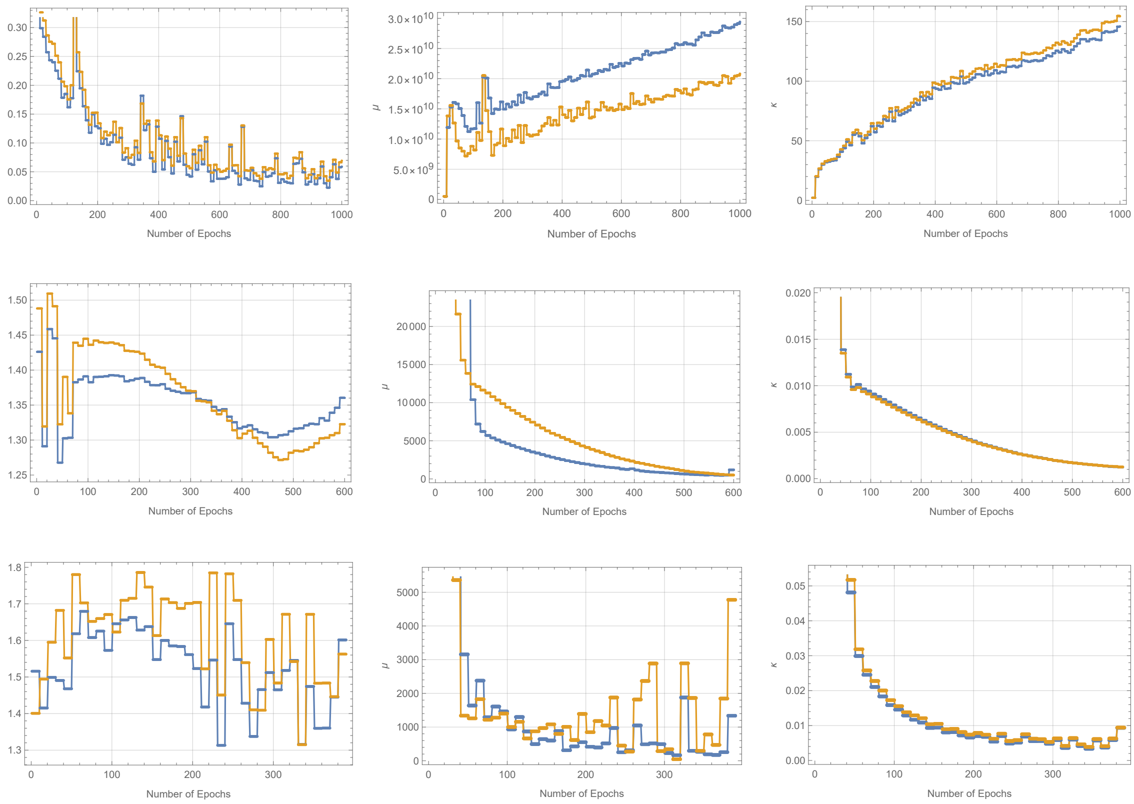

Figure 8. Evolution of the various measures after training separately with different weights on the Fermat Quintic. In the first row only σ is taken as the loss function for training. We observe that as σ decreases in the training process both κ and μ increase. In the second and third rows we train using μ only and κ only. We observe that in both these cases μ and κ decrease simultaneously. The σ measure instead remains stable. In all the figures the blue lines correspond to the training set and yellow to the test set.

of the complex structure moduli. There are in total 192 patches. Note that in contrast to the previous hypersurface examples, the patches are not all equivalent. This is due to the fact that all permutations among homogeneous coordinates have to act simultaneously on both $\mathbb{P}^3$ ambient spaces in order to leave (6.6) invariant. This leaves us with four inequivalent families of patches. The family we considered for our computations is based on the patch $(4, 6; 2, 7, 1)$. Here, we train using only the σ-measure as the loss function. Clearly, because we are not tracing the Kähler class or imposing the closedness of the Kähler form in the training, the endpoint of the procedure does not give us the Ricci flat metric. Nevertheless, the methodology shows that the σ-measure decreases. This is worthwhile as an intermediate step in determining metrics for more complicated geometries. The network approximation produces σ-measure values of 0.5 or better for the training and test sets. Similar values are obtained if one trains with a ReLU activation function instead of a hyperbolic tangent as shown in figure 11.

In contrast to the previous hypersurface cases where one has permutation symmetries that permit us to infer an identical structure of the metric in all of the patches, for the Tian-Yau manifold one has four different classes of patches. This makes it harder to evaluate the

JHEP08(2022)105

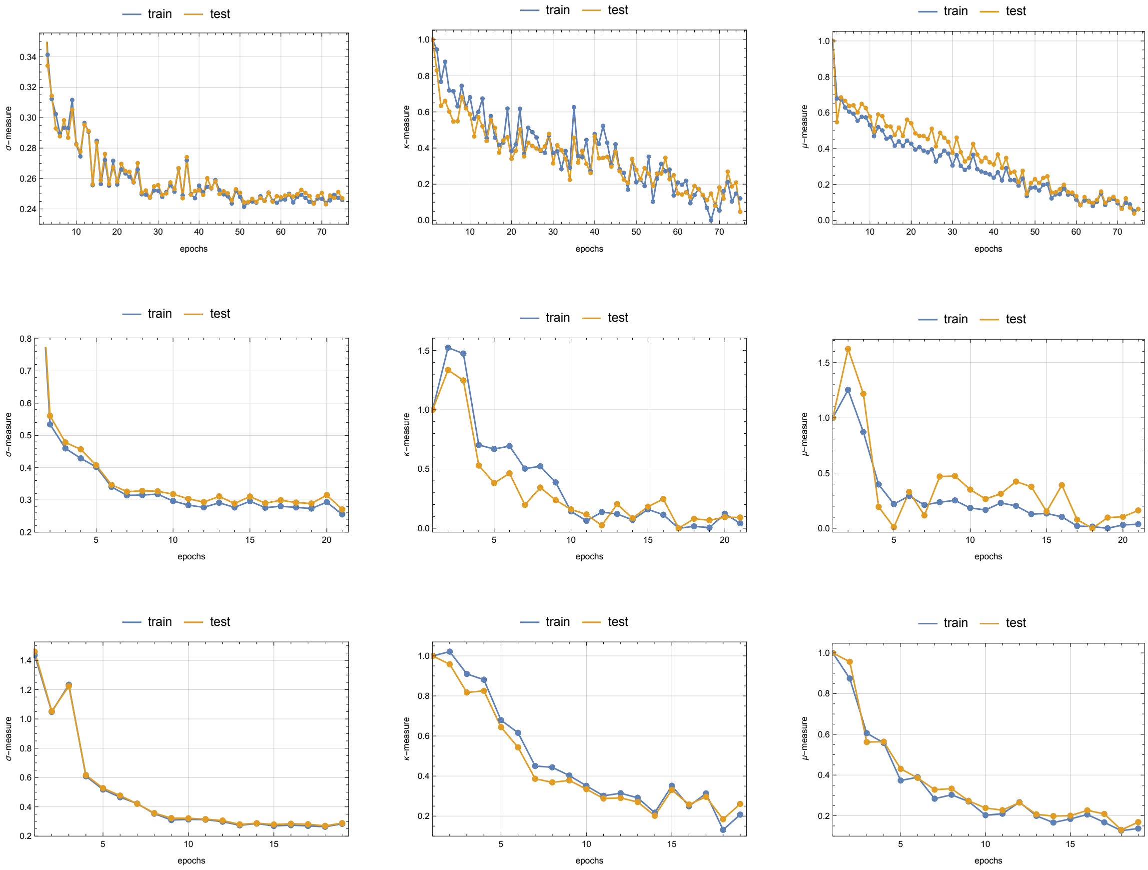

Figure 9. Evolution of the σ, κ, and μ-measures as defined in section 6.2, for the Fermat quintic (2.6). For ease of visualization, the σ measure is reported in the original scale, whereas the other measures are reported in the log scale after a normalisation. Our architecture is a set of two three layer networks with 500 nodes in each layer and tanh activation (see figure 2). We use a train-test split of $90:10$. We use variable training sizes: $28,000$ points in the case of the top row, and $10,000$ points for the bottom two rows. We use a batch size of 500, a learning rate of 10^{-4} (rows one and two) and 10^{-3} for the bottom row. The loss function is a weighted combination of all three measures. In all these cases, the σ measures fall below the corresponding σ measure for the pullback of the Fubini-Study metric. These graphs illustrate the robustness of our architecture and methodology in obtaining approximations to the Kähler, Ricci flat metric on the Fermat quintic. Remarkably, a small number of epochs are sufficient to achieve this learning.

μ-measure. One possibility to follow this approach is to employ a pair of neural networks per class.

Another example of a complete intersection Calabi-Yau that could be addressed using similar modifications is the Schoen manifold described below:

$$\text{Schoen}: \quad \left[\begin{array}{c||cc} \mathbb{P}^2 & 3 & 0 \\ \mathbb{P}^2 & 0 & 3 \\ \mathbb{P}^1 & 1 & 1 \end{array}\right]_{\chi=0} . \tag{7.6}$$

The configuration matrix for this manifold is the transpose of the configuration matrix

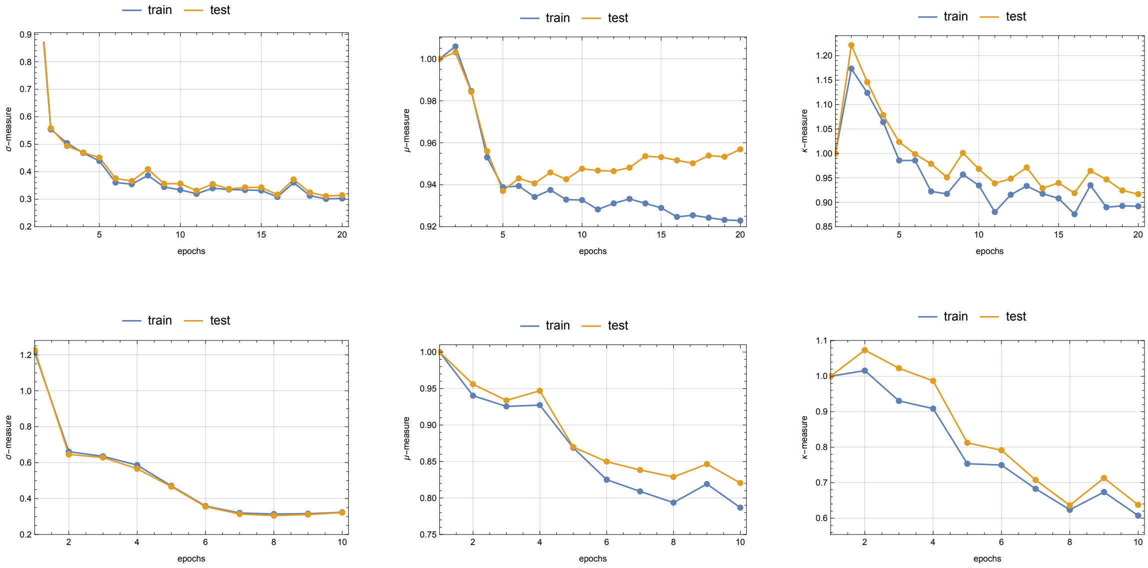

Figure 10. Evolution of σ, κ, and μ-measures for a member of the Dwork family of quintics (2.7) with $\psi = -\frac{1}{5}$. For ease of visualization, the σ measure is reported in the original scale, whereas the other measures are reported relative in the log scale following a normalisation. Following the case of the Fermat quintic, our architecture is a set of two three layer networks with 500 nodes in each layer and *tanh* activation (see figure 2). We use a train-test split of $90 : 10$. We use $10,000$ points for training, a batch size of 500, and a learning rate of 10^{-4} (top row) and 10^{-3} (bottom row). The loss function is a weighted combination of all three measures.

for the Tian-Yau manifold (2.8). The manifold is topologically distinct, however. The Hodge pair for this manifold is $(h^{1,1}, h^{1,2}) = (19, 19)$, and therefore the Euler characteristic is $\chi = 0$. This manifold is interesting in its own right for string phenomenology. It is a split of the bicubic complete intersection Calabi-Yau on which semi-realistic heterotic string models have been realized. We defer a more rigorous and thorough analysis of the Tian-Yau manifold and related geometries to future work.

8 Discrete symmetries

Discrete symmetries of Calabi-Yau manifolds play an important rôle in string model building. In the Standard Model, discrete symmetries are merely hypothesized to explain certain processes or their absence. A phenomenologically interesting example pertains to the absence of proton decay. In this case, a discrete R-symmetry in the minimal supersymmetric Standard Model protects the proton from decay. It also ensures a stable lightest supersymmetric partner, which is then a candidate for dark matter. A second example of the important rôle discrete symmetries play in particle phenomenology is given by the discrete structure of mixing matrices such as the CKM and the PMNS matrices in the quark and lepton sectors, respectively. A symmetry group invoked to explain certain such discrete structures is $\Delta(27)$, which is isomorphic to $(Z_3 \times Z_3) \rtimes Z_3$. (Curiously, $\Delta(27)$ as an orbifold group gives simple constructions of the Standard Model as a worldvolume theory on D3-

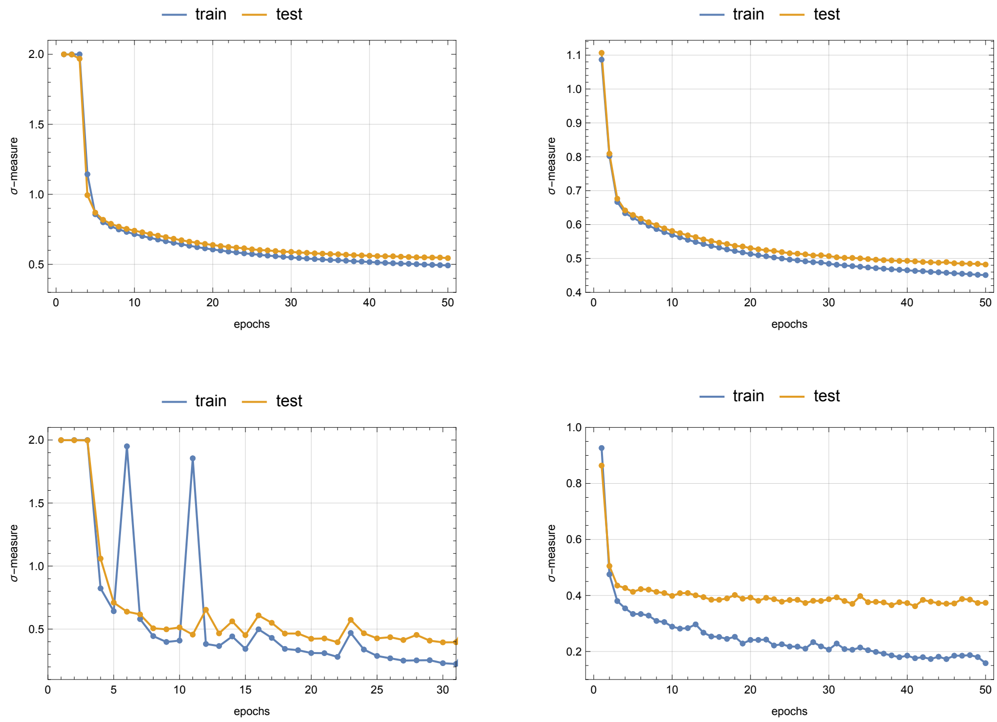

Figure 11. Evolution of the σ-measure during the course of training for the Tian-Yau manifold (2.8). In the experiments in the top row we use the activation functions ReLU (left) and *tanh* (right), a learning rate of 5×10^{-6}, a batch size of 200 and use 1000 nodes in each hidden layer of our neural network. We use $15,000$ points to train and use a train-test ratio of $90 : 10$. In the experiments in the bottom row, we use the same hyperparameters except the learning rates which are 5×10^{-4} and 5×10^{-5} respectively for the ReLU (left) and the tanh (right) architectures respectively. To mitigate the issue of over-fitting, we stop training when the training loss stabilises. Further, we use the sigma measure alone as the loss. Experiments using the logistic sigmoid activation did not yield good results.

branes [65, 66].) In the language of superstrings, these discrete symmetries cannot merely be hypothesized, but actually descend from isometries of the underlying compactification space. As such, constructing Calabi-Yau spaces with phenomenologically interesting discrete symmetries that remain unbroken after compactification is of paramount importance.

A number of theoretical investigations have led to the classification of discrete symmetries of Calabi-Yau threefolds. These have resulted in large datasets of discrete symmetries of complete intersection Calabi-Yau threefolds that are freely acting [67–70]. This allowed a classification of the linearly realised freely acting discrete symmetries of the resulting quotient Calabi-Yau threefolds, on which heterotic models are typically built [71, 72]. Further, it was shown that discrete symmetries can be enhanced at special loci in the complex structure moduli space [72, 73]. Similar attempts have been made on the Kreuzer-Skarke dataset [74], although due to the vastness of the dataset, a full classification has not been possible. These investigations point to the fact that discrete symmetries are rather rare in Calabi-Yau spaces, with phenomenologically interesting symmetries even more so. Many semi-realistic string derived standard models are built on Calabi-Yau quotients by freely

JHEP08(2022)105

acting symmetries. The quotient manifolds have topological properties distinct from the original manifold [68, 69, 75] and are often more suitable for model building.

With the advent of machine learning in studying the string landscape, one might perceive faster classification of such symmetries on the known Calabi-Yau databases, although there has not been much success in this direction so far [31, 76]. In this section, we demonstrate that the neural networks we have trained to approximate the Ricci flat metric is invariant under some of the discrete symmetries of the Calabi-Yau, without this being explicitly coded into the architecture. This acts as a sanity check in cases where we expect the metric to be invariant under a given discrete symmetry. Although this does not provide a method of actually detecting such symmetries (for which one would need to solve the corresponding inverse problem), we hope this stimulates further investigation into a machine motivated discrete symmetry learning of Calabi-Yau spaces.

We will discuss discrete symmetries for K3 and the quintic threefold. Although we do not inform our neural network architectures of any symmetry explicitly, we test whether such symmetries are nonetheless being learned (encoded in the architecture) as a consequence of learning the flat metric. As such, we quantify the extent of learned symmetry until epoch n, by δ_n, defined as,

$$\delta_n(\tau) := \frac{1}{N}\sum_z \text{abs}\left(\frac{g_{NN}(\theta_n; z) - g_{NN}(\theta_n; \tau.z)}{g_{NN}(\theta_n; z)}\right), \tag{8.1}$$

where $g_{NN}(\theta_n; z)$ denotes the metric constructed using the neural network(s) in figure 2, with θ_n denoting the network's parameters at epoch n. Here, τ denotes a specific instance of a discrete symmetry of the manifold in question, acting linearly on the combined homogeneous coordinates of the ambient space. The sum is over a random selection of N points used for training and testing. An overall decreasing behavior of $\delta_n(\tau)$ with increasing number of epochs, n, would be indicative of the symmetry τ being learned. Below, we outline how well certain symmetries were captured by our neural network architectures for the K3 surface and the Fermat quintic.

8.1 The quartic K3 surface

From the quartic K3 surface equation reproduced in (8.2), it is easy to note the symmetry permuting the homogeneous coordinates $[z_1 : z_2 : z_3 : z_4]$ of $\mathbb{P}^3$. The permutation symmetry is utilized during the training process to ensure that the metric agrees at the intersection of different patches. We set one of the coordinates to 1 defining a patch, in effect breaking the permutation symmetry S_4. There is a second set of symmetries that acts diagonally on the combined homogeneous coordinates. The action of this symmetry is given in (8.3) and an instance of it, τ, in (8.4).

$$\text{K3}: \quad z_1^4 + z_2^4 + z_3^4 + z_4^4 = 0 \subset \mathbb{P}^3 \tag{8.2}$$

$$z_p \to \omega_p\, z_p\,; \quad \text{with} \quad p \in \{1,2,3,4\} \quad \text{and} \quad \omega_p \in \mathbb{Z}_4\,. \tag{8.3}$$

$$\tau: \quad z_1 \mapsto i\, z_1\,,\; z_2 \mapsto -\, z_2\,,\; z_3 \mapsto -i\, z_3\,. \tag{8.4}$$

In figure 12, we report the extent of symmetry learned by monitoring δ_n over the epochs. We find that not all architectures capture this symmetry. In fact, the architecture

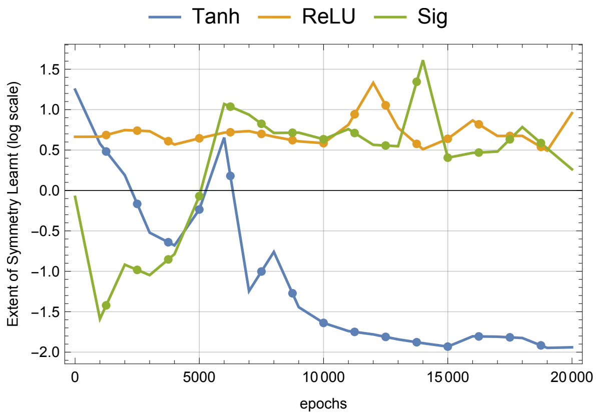

Figure 12. Discrete isometries of the quartic K3 surface defined in (8.2) being learnt. The plot shows the extent of learning given by (8.1) for the symmetry of the K3 surface described in (8.4). We observe similar behavior for other symmetries of the class (8.3).

with tanh activation function is the only one that shows a behavior indicative of the symmetry τ (8.4) being learned.

8.2 The Fermat quintic

For the Fermat quintic similar considerations apply, although the discrete symmetries themselves are different. In (8.6) we identify a class of symmetries that could have possibly been learned during the process of training. An example of this symmetry is given in (8.7) and it acts diagonally on the homogeneous coordinates $[z_1 : z_2 : z_3 : z_4 : z_5]$.

$$\text{Fermat quintic}: \quad z_1^5 + z_2^5 + z_3^5 + z_4^5 + z_5^5 = 0 \subset \mathbb{P}^4 \tag{8.5}$$

$$z_p \to \omega_p\, z_p\,; \quad \text{with} \quad p \in \{1,2,3,4,5\} \quad \text{and} \quad \omega_p \in \mathbb{Z}_5\,. \tag{8.6}$$

$$\tau: \quad z_1 \mapsto e^{2\pi \mathrm{i}/5}\, z_1\,, \quad z_2 \mapsto e^{4\pi \mathrm{i}/5}\, z_2\,, \quad z_3 \mapsto e^{8\pi \mathrm{i}/5}\, z_3, \quad z_4 \mapsto e^{6\pi \mathrm{i}/5}\, z_4\,. \tag{8.7}$$

In figure 13, we show the extent of learning for the symmetry described in (8.1). In this example, we note that the architecture with the logistic sigmoid activation function has captured the symmetry to the largest extent, while the other architectures show behavior consistent with this learning. The realization of the symmetries through machine learning indicates that the numerical Ricci flat metric is consistent. This is a non-trivial *a posteriori* confirmation of the methods employed.

9 Discussion and outlook

In this paper, we have reported preliminary investigations on machine learning numerical metrics on Calabi-Yau spaces. We created neural network models for metrics over Calabi-Yau spaces of complex dimensions up to three, choosing examples of the complex torus T^2 in dimension one and the quartic K3 surface in dimension two. In dimension three, we chose to work with the Fermat quintic, a second member of the Dwork family, as well

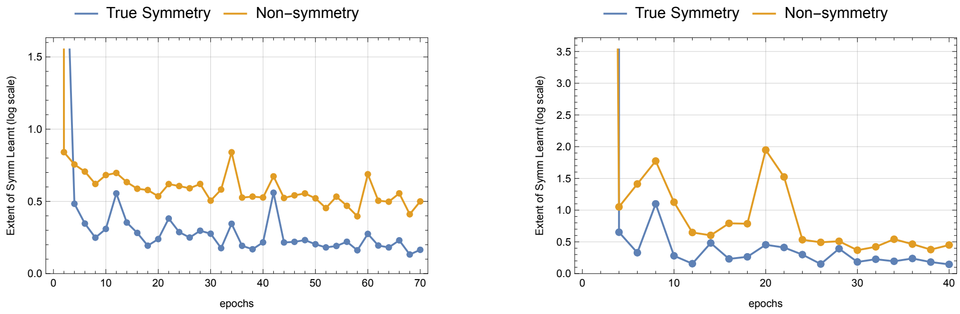

Figure 13. Discrete isometries of the Fermat quintic defined in (8.5) being learnt. The plot shows the extent of learning given by (8.1) for the symmetry of the quintic surface described in (8.7). The plots correspond to two different networks trained in figure 9. We observe similar behavior for other symmetries of the class (8.6).

as the phenomenologically interesting Tian-Yau manifold. Interestingly, the same neural network architecture described in figure 2 could approximate Ricci flat metrics on all these spaces. In these experiments, the tanh activation function yielded the best σ-measures on these spaces.

In most of our experiments, we have taken a small number of points. This is in contrast with the relatively large number of hyperparameters in the neural networks employed. For the case of the Fermat quintic, where we have increased the number of points for training, we saw improving results on the test set, demonstrating that the neural networks are indeed capable of generalization. We expect this trend to apply in all other geometries considered. Additionally, we would like to employ the full power of the loss function to include the κ- and μ-measures in all cases, not just for the Fermat quintic. An important issue to address in this direction is to handle complex matrix products directly in order to compute Frobenius norms more efficiently. This was not feasible in the `PyTorch` implementation as it only allows us to manipulate real quantities. Once these pieces are in order, one would scan for an optimal network architecture that weighs in the capacity for generalization as well as loss minimization. Another avenue to explore is the possibility to constrain network hyperparameters in such a way that Kählericity is built in to the metric solution *ab initio*.

In section 8, we commented on discrete symmetries of Calabi-Yau manifolds. While discrete symmetries are very important from a string model building perspective, they are rare. As such, a full classification of them is computationally intensive. There are many ways of incorporating symmetries into a machine learning model involving neural networks. Some of the traditional methods include data augmentation [77, 78], feature averaging, DeepSets [79], and building more complex equivariant architectures [80]. The inverse problem of discovering symmetries of the dataset itself, or in our case, the manifolds, is not well-studied. An example of such a study is [81], wherein the marginal likelihood was employed to compute symmetries in the MNIST dataset [82]. In this paper, we lay the ground work for an alternate machine driven approach to discovering symmetries. Through examples of symmetries of the quartic K3 surface and the Fermat quintic, we demonstrate that discrete symmetries can be learned while a network is being trained on an auxiliary

JHEP08(2022)105

task. In this case, the task is to learn the flat metric. We found that our networks were able to learn discrete symmetries over the course of training without any effort on our part to inform the learning process or the network's architecture of these symmetries.

A discrete symmetry may be realized linearly in a given representation of a complete intersection Calabi-Yau, and simultaneously be realized non-linearly in an equivalent representation. An example of this phenomenon involves the Schoen manifold (reproduced below), a split of the bicubic. This manifold admits an equivalent representation which is also noted alongside:

$$\text{Schoen}: \quad \left[\begin{array}{c||cc} \mathbb{P}^1 & 1 & 1 \\ \mathbb{P}^2 & 3 & 0 \\ \mathbb{P}^2 & 0 & 3 \end{array}\right]_{\chi=0} \simeq \left[\begin{array}{c||cccccc} \mathbb{P}^1 & 0 & 0 & 0 & 0 & 1 & 1 \\ \mathbb{P}^2 & 1 & 1 & 0 & 0 & 1 & 0 \\ \mathbb{P}^2 & 1 & 1 & 0 & 0 & 1 & 0 \\ \mathbb{P}^2 & 0 & 0 & 1 & 1 & 0 & 1 \\ \mathbb{P}^2 & 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right]_{\chi=0} . \tag{9.1}$$

The manifolds in (9.1) are equivalent. There are symmetries of this manifold that are linearly realized in the extended representation but not in the split bicubic representation. Non-linearly realized symmetries are hard to classify using traditional methods. It would be interesting to understand if non-linear symmetries are also picked up by the neural networks during the training process. We hope to return to this analysis in the future.

The observations above warrant an analysis of the neural networks we have considered. The aim of such an analysis would be to understand how learning of the metric happens, as well as to derive hints for the underlying analytic function for the flat metric. Such an analysis is a necessity in mitigating the black box nature of the resulting neural network models. Although we wish to return to such analysis in a future work, a few comments are in order. A particular approach to analysis is using the recent Neural Tangent Kernel (NTK) formulation [83]. Under the NTK regime, we would have an analytic formulation of the function to which our networks would converge. This would allow us to make meaningful statements for the underlying analytic flat metric. Secondly, in order to understand the training dynamics, we would like to understand how the spectra of matrices of the network parameters evolve over the course of learning. In figure 14, we show this for the Fermat quintic. We note that during the course of training, the overall spectra lowers. We also note that condition number, which is the ratio of the largest and smallest eigenvalues, decreases over the course of the training. We observe similar behavior for other weight matrices in all the examples we have considered. We would like to understand if this provides hints for the underlying analytic function approximating the metric.

Finally, a Topological Data Analysis (TDA) approach, tracking persistent homology of the distribution of the network parameters might reveal further insights into the learning process. Persistence has been shown to be a useful tool for interpretability in deep learning [84, 85]. Recently, a new complexity measure of neural networks has been proposed with the aid of topological methods [86]. This has helped better understand certain machine learning best practices, like dropout and batch-normalization. We expect such topological methods to shed further light on creating more accurate and interpretable neural network models of Ricci flat metrics of Calabi-Yau manifolds.

JHEP08(2022)105

In importing machine learning methods to high energy theoretical physics, there has so far only been limited success in translating the learned associations into interpretable analytic formulae. (See [30, 87–89] for computations in line bundle cohomology on surfaces and in knot theory.) An open problem, of course, is to reverse engineer the performance of the neural network to obtain analytic expressions for flat metrics. So far, we have no results in this direction.

Work in progress details the Ricci flow approach more explicitly. There are a number of strategies we adopt. One of these is simply to use $\sum_{a,\bar{b}} |\mathrm{Ric}_{a\bar{b}}|$ as the loss function to update the metric according to the Ricci flow equation. When this loss is minimized to zero, we have arrived at the flat metric in the same Kähler class as the Fubini-Study metric, which is the starting point of the flow. Another strategy is to solve the partial differential equations (4.6) and (4.7) using machine learning techniques. Indeed, deep learning is particularly good at solving certain systems of equations [90, 91].

As approximations for the metric we have considered a system of two neural networks providing the metric entries in a natural matrix factorization. This method proves to work relatively well for all the geometries considered. Because of this we would like to highlight its modularity: it could serve not only to approximate the metric of Calabi-Yau manifolds, but more in general to complex and real manifolds as well.

In this work, we have initiated a neural network approach to finding numerical Calabi-Yau metric. As this program proceeds, it is worth considering how promising this approach would be. Donaldson's balanced metric at level k will have N_k^2 real parameters, with N_k the number of monomials up to order k:

$$N_k = \begin{cases} \begin{pmatrix} 5+k-1 \\ k \end{pmatrix} - \begin{pmatrix} k-1 \\ k-5 \end{pmatrix}\,, & k < 5 \\ \begin{pmatrix} 5+k-1 \\ k \end{pmatrix} - \begin{pmatrix} k-1 \\ k-5 \end{pmatrix}\,, & k \geq 5\,. \end{cases} \tag{9.2}$$

For $k \sim 12$, the balanced metric will require more parameters than the neural network. Using only the sigma measure, we find that the performance corresponds to $k \sim 16$ (see figure 15). This sets an upper bound on how well we can do. Assuming these bounds extrapolate to different Calabi-Yau spaces, in particular those used for string compactification to the Standard Model, we should ask whether a numerical metric near the upper bound would be good enough to compute Yukawa couplings to sufficient accuracy that we can compare model predictions to experimental measurements. At the moment, this discussion is perhaps premature because the errors in the Kähler measure are not yet small enough to apply the numerical metrics to model building.

These methods are broadly applicable. In particular, due to the modularity of the architecture, we can envision finding a suitable loss function that enables us to compute metrics on G_2 manifolds.[3] We can as well envision applying machine learning methods to finding new solutions to general relativity or supergravity. Numerical methods have been

[3] We thank Anthony Ashmore for a discussion on this point.

JHEP08(2022)105

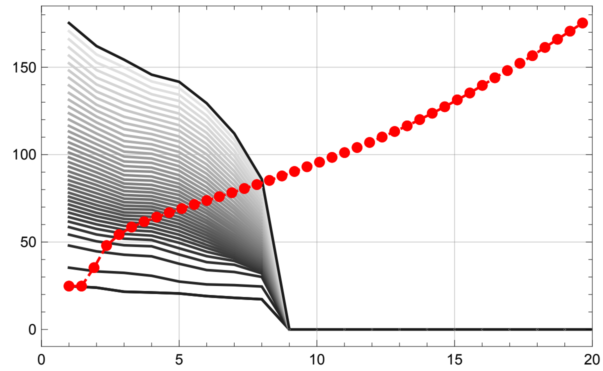

Figure 14. Evolution of the eigenspectra of the symmetrised weight matrix for the first layer of our neural network. This network corresponds to the top neural network in figure 2, and outputs the eigenvalues of the flat metric on the Fermat quintic. There is one curve for each epoch. The x-axis corresponds to the top twenty eigenvalues. The y-axis gives its numerical value. The topmost curve corresponds to the beginning of training, and the bottom, the end. The curve in red tracks the largest eigenvalue from the spectra at each epoch. We note a certain monotonicity in this figure, in that the largest eigenvalue as well as the overall spectra lowers as training progresses.

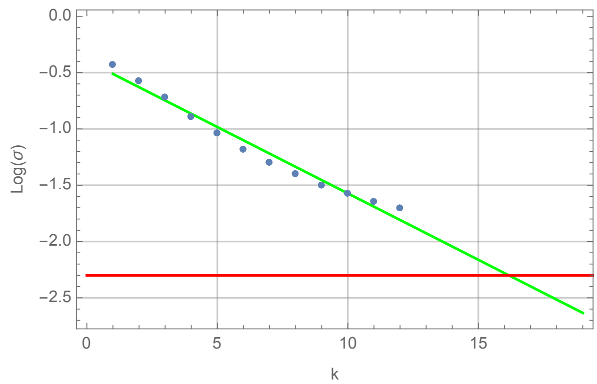

Figure 15. The value of σ reached by the neural network array on the Fermat quintic would be attained at level $k \sim 16$ using Donaldson's algorithm. The exponential extrapolation was constructed using the data of [35].

applied to construct stationary black hole solutions in various contexts, for example, by using shooting methods to solve ordinary differential equations or employing the Newton-Raphson procedure to solve partial differential equations [92–94]; neural networks may supply an additional tool as a supplement to these approaches.

We have so far only studied a handful of Calabi-Yau threefolds, including the Fermat quintic and Tian-Yau Calabi-Yau threefolds. We aim to find numerical Ricci flat metrics for the base manifolds implicated in top down constructions of the Standard Model. Interestingly, the configuration matrix in [9] corresponding to the split bicubic or Schoen manifold is the transpose of the Tian-Yau configuration matrix (2.8); its metric is amenable to attack using similar methods. We hope to create neural network models for Ricci flat metrics on this manifold as well as on the other complete intersection Calabi-Yaus as an ex-

JHEP08(2022)105

tension of this work. In future work, we also hope to report on a first principles theoretical computation of the mass of the electron based on these models.

Acknowledgments

We thank Anthony Ashmore, Per Berglund, Andrew Dancer, Yarin Gal, Mario Garcia-Fernandez, James Gray, Tristan Hübsch, Nikhil Raghuram, Lewis Smith, Mark van der Wilk, and Francisco Vargas for conversations and the Simons Center for Geometry and Physics for hospitality at the "Strings, Geometry, and Data Science" meeting in January 2020. We also thank Rahul Sharma and Siddharth Krishna for computational support which helped us address the anonymous referee's helpful comments. VJ is supported by the South African Research Chairs Initiative of the Department of Science and Technology and the National Research Foundation. VJ and DKMP are supported by a Simons Foundation Mathematics and Physical Sciences Targeted Grant, 509116. CM is supported by a Fellowship by the Accelerate Science program at the Computer Laboratory, University of Cambridge. CM acknowledges support for part of the work by the U.K. Government's Defence & Security Programme in support of the Alan Turing Institute.

References

[1] P. Candelas, G.T. Horowitz, A. Strominger and E. Witten, *Vacuum Configurations for Superstrings*, *Nucl. Phys. B* **258** (1985) 46 [INSPIRE].

[2] M.B. Green, J. Schwarz and E. Witten, *Superstring Theory. Volume 2. Loop Amplitudes, Anomalies and Phenomenology*, Cambridge University Press, Cambridge, U.K. (1988).

[3] E. Calabi, *The space of Kähler metrics*, in proceedings of the *International Congress of Mathematicians*, Amsterdam, The Netherlands, 2–9 September 1954, volume 2, North Holland, Amsterdam, The Netherlands (1956), pp. 206–207.

[4] S.-T. Yau, *Calabi's Conjecture and some new results in algebraic geometry*, *Proc. Nat. Acad. Sci.* **74** (1977) 1798 [INSPIRE].

[5] S.-T. Yau, *On the Ricci Curvature of a Compact Kähler Manifold and the Complex Monge-Ampère Equation. Part I*, *Comm. Pure. Appl. Math.* **31** (1978) 339.

[6] D. Gaiotto, G.W. Moore and A. Neitzke, *Wall-crossing, Hitchin Systems, and the WKB Approximation*, arXiv:0907.3987 [INSPIRE].

[7] S. Kachru, A. Tripathy and M. Zimet, *$K3$ metrics from little string theory*, arXiv:1810.10540 [INSPIRE].

[8] S. Kachru, A. Tripathy and M. Zimet, *$K3$ metrics*, arXiv:2006.02435 [INSPIRE].

[9] V. Braun, Y.-H. He, B.A. Ovrut and T. Pantev, *A Heterotic standard model*, *Phys. Lett. B* **618** (2005) 252 [hep-th/0501070] [INSPIRE].

[10] V. Bouchard and R. Donagi, *An* SU(5) *heterotic standard model*, *Phys. Lett. B* **633** (2006) 783 [hep-th/0512149] [INSPIRE].

[11] M. Headrick and T. Wiseman, *Numerical Ricci-flat metrics on* $K3$, *Class. Quant. Grav.* **22** (2005) 4931 [hep-th/0506129] [INSPIRE].

[12] S. Donaldson, *Scalar curvature and projective embeddings. Part I*, *J. Diff. Geom.* **59** (2001) 479.

[13] S.K. Donaldson, *Some numerical results in complex differential geometry*, math/0512625.

[14] M.R. Douglas, R.L. Karp, S. Lukic and R. Reinbacher, *Numerical Calabi-Yau metrics*, *J. Math. Phys.* **49** (2008) 032302 [hep-th/0612075] [INSPIRE].

[15] M. Headrick and A. Nassar, *Energy functionals for Calabi-Yau metrics*, *Adv. Theor. Math. Phys.* **17** (2013) 867 [arXiv:0908.2635] [INSPIRE].

[16] M.R. Douglas, R.L. Karp, S. Lukic and R. Reinbacher, *Numerical solution to the Hermitian Yang-Mills equation on the Fermat quintic*, *JHEP* **12** (2007) 083 [hep-th/0606261] [INSPIRE].

[17] V. Braun, T. Brelidze, M.R. Douglas and B.A. Ovrut, *Calabi-Yau Metrics for Quotients and Complete Intersections*, *JHEP* **05** (2008) 080 [arXiv:0712.3563] [INSPIRE].

[18] V. Braun, T. Brelidze, M.R. Douglas and B.A. Ovrut, *Eigenvalues and Eigenfunctions of the Scalar Laplace Operator on Calabi-Yau Manifolds*, *JHEP* **07** (2008) 120 [arXiv:0805.3689] [INSPIRE].

[19] L.B. Anderson, V. Braun, R.L. Karp and B.A. Ovrut, *Numerical Hermitian Yang-Mills Connections and Vector Bundle Stability in Heterotic Theories*, *JHEP* **06** (2010) 107 [arXiv:1004.4399] [INSPIRE].

[20] L.B. Anderson, V. Braun and B.A. Ovrut, *Numerical Hermitian Yang-Mills Connections and Kähler Cone Substructure*, *JHEP* **01** (2012) 014 [arXiv:1103.3041] [INSPIRE].

[21] A. Ashmore, *Eigenvalues and eigenforms on Calabi-Yau threefolds*, arXiv:2011.13929 [INSPIRE].

[22] W. Cui and J. Gray, *Numerical Metrics, Curvature Expansions and Calabi-Yau Manifolds*, *JHEP* **05** (2020) 044 [arXiv:1912.11068] [INSPIRE].

[23] Y.-H. He, *Machine-learning the string landscape*, *Phys. Lett. B* **774** (2017) 564 [INSPIRE].

[24] D. Krefl and R.-K. Seong, *Machine Learning of Calabi-Yau Volumes*, *Phys. Rev. D* **96** (2017) 066014 [arXiv:1706.03346] [INSPIRE].

[25] F. Ruehle, *Evolving neural networks with genetic algorithms to study the String Landscape*, *JHEP* **08** (2017) 038 [arXiv:1706.07024] [INSPIRE].

[26] J. Carifio, J. Halverson, D. Krioukov and B.D. Nelson, *Machine Learning in the String Landscape*, *JHEP* **09** (2017) 157 [arXiv:1707.00655] [INSPIRE].

[27] F. Ruehle, *Data science applications to string theory*, *Phys. Rept.* **839** (2020) 1 [INSPIRE].

[28] V. Jejjala, A. Kar and O. Parrikar, *Deep Learning the Hyperbolic Volume of a Knot*, *Phys. Lett. B* **799** (2019) 135033 [arXiv:1902.05547] [INSPIRE].

[29] S. Gukov, J. Halverson, F. Ruehle and P. Sułkowski, *Learning to Unknot*, *Mach. Learn. Sci. Tech.* **2** (2021) 025035 [arXiv:2010.16263] [INSPIRE].

[30] J. Craven, V. Jejjala and A. Kar, *Disentangling a deep learned volume formula*, *JHEP* **06** (2021) 040 [arXiv:2012.03955] [INSPIRE].

[31] K. Bull, Y.-H. He, V. Jejjala and C. Mishra, *Machine Learning CICY Threefolds*, *Phys. Lett. B* **785** (2018) 65 [arXiv:1806.03121] [INSPIRE].

[32] K. Bull, Y.-H. He, V. Jejjala and C. Mishra, *Getting CICY High*, *Phys. Lett. B* **795** (2019) 700 [arXiv:1903.03113] [INSPIRE].

[33] H. Erbin and R. Finotello, *Inception neural network for complete intersection Calabi-Yau 3-folds*, *Mach. Learn. Sci. Tech.* **2** (2021) 02LT03 [arXiv:2007.13379] [INSPIRE].

[34] H. Erbin and R. Finotello, *Machine learning for complete intersection Calabi-Yau manifolds: a methodological study*, *Phys. Rev. D* **103** (2021) 126014 [arXiv:2007.15706] [INSPIRE].

[35] A. Ashmore, Y.-H. He and B.A. Ovrut, *Machine Learning Calabi-Yau Metrics*, *Fortsch. Phys.* **68** (2020) 2000068 [arXiv:1910.08605] [INSPIRE].

[36] L.B. Anderson, M. Gerdes, J. Gray, S. Krippendorf, N. Raghuram and F. Ruehle, *Moduli-dependent Calabi-Yau and* SU(3)*-structure metrics from Machine Learning*, *JHEP* **05** (2021) 013 [arXiv:2012.04656] [INSPIRE].

[37] M.R. Douglas, S. Lakshminarasimhan and Y. Qi, *Numerical Calabi-Yau metrics from holomorphic networks*, arXiv:2012.04797 [INSPIRE].

[38] N. Raghuram, *On Calabi-Yau Metrics and Neural Networks*, talk given at *Strings, Geometry, and Data Science*, Simons Center for Geometry and Physics, Stony Brook, NY, U.S.A., 8 January 2020.

[39] F. Ruehle, *Machine Learning in Theoretical Physics*, talk given at *Bethe Colloquium*, University of Bonn, Bonn, Germany, 4 June 2020.

[40] M.R. Douglas, *Numerical Calabi-Yau Metrics from Holomorphic Networks*, in proceedings of *string_data 2020*, online, 14–16 December 2020.

[41] S. Krippendorf, *Calabi-Yau Metrics from Machine Learning*, in proceedings of *string_data 2020*, online, 14–16 December 2020.

[42] P. Candelas, *Lectures on complex manifolds*, in *Superstrings and Grand Unification*, proceedings of the *Winter School on High Energy Physics*, Puri, India, 3–17 January 1988, T. Pradhan ed., World Scientific, Singapore (1988).

[43] T. Hubsch, *Calabi-Yau manifolds: A Bestiary for physicists*, World Scientific, Singapore (1992).

[44] G. Tian and S. Yau, *Three dimensional algebraic manifolds with* $C_1 = 0$ *and* $\chi = -6$, in *Advanced Series in Mathematical Physics* **1**, World Scientific, Singapore (1987), pp. 543–559.

[45] B.R. Greene, K.H. Kirklin, P.J. Miron and G.G. Ross, *A Three Generation Superstring Model. Part 1. Compactification and Discrete Symmetries*, *Nucl. Phys. B* **278** (1986) 667 [INSPIRE].

[46] R.S. Hamilton, *Three-manifolds with positive Ricci curvature*, *J. Diff. Geom.* **17** (1982) 255.

[47] G. Perelman, *The Entropy formula for the Ricci flow and its geometric applications*, math/0211159 [INSPIRE].

[48] G. Perelman, *Ricci flow with surgery on three manifolds*, math.DG/0303109.

[49] G. Perelman, *Finite extinction time for the solutions to the Ricci flow on certain three-manifolds*, math/0307245 [INSPIRE].

[50] B. Chow et al., *The Ricci flow: techniques and applications*, American Mathematical Society, Providence, RI, U.S.A. (2007).

[51] B. Kleiner, J. Lott et al., *Notes on Perelman's papers*, *Geom. Topol.* **12** (2008) 2587.

[52] M. Headrick and T. Wiseman, *Ricci flow and black holes*, *Class. Quant. Grav.* **23** (2006) 6683 [hep-th/0606086] [INSPIRE].

[53] C. Doran, M. Headrick, C.P. Herzog, J. Kantor and T. Wiseman, *Numerical Kähler-Einstein metric on the third del Pezzo*, *Commun. Math. Phys.* **282** (2008) 357 [hep-th/0703057] [INSPIRE].

[54] S. Jackson, R. Pourhasan and H. Verlinde, *Geometric RG Flow*, arXiv:1312.6914 [INSPIRE].

[55] P. Fonda, V. Jejjala and A. Veliz-Osorio, *On the Shape of Things: From holography to elastica*, *Annals Phys.* **385** (2017) 358 [arXiv:1611.03462] [INSPIRE].

[56] J. Song and B. Weinkove, *Lecture notes on the Kähler-Ricci flow*, arXiv:1212.3653.

[57] H.-D. Cao, *Deformation of Kähler matrics to Kähler-Einstein metrics on compact Kähler manifolds*, *Invent. Math.* **81** (1985) 359.

[58] X. Chen and G. Tian, *Ricci flow on Kähler-Einstein manifolds*, *Duke Math. J.* **131** (2006) 17.

[59] G. Cybenko, *Approximation by superpositions of a sigmoidal function*, *Math. Control Signals Syst.* **2** (1989) 303.

[60] K. Hornik, *Approximation capabilities of multilayer feedforward networks*, *Neural Netw.* **4** (1991) 251.

[61] B. Shiffman and S. Zelditch, *Distribution of zeros of random and quantum chaotic sections of positive line bundles*, *Commun. Math. Phys.* **200** (1999) 661.

[62] H.D. Cao, *Deformation of Kähler matrics to Kähler-Einstein metrics on compact Kähler manifolds*, *Invent. Math.* **81** (1985) 359.

[63] A. Wu, M.C. Aoi and J.W. Pillow, *Exploiting gradients and Hessians in Bayesian optimization and Bayesian quadrature*, arXiv:1704.00060.

[64] W.M. Czarnecki, S. Osindero, M. Jaderberg, G. Świrszcz and R. Pascanu, *Sobolev training for neural networks*, arXiv:1706.04859.

[65] G. Aldazabal, L.E. Ibáñez, F. Quevedo and A.M. Uranga, *D-branes at singularities: A Bottom up approach to the string embedding of the standard model*, *JHEP* **08** (2000) 002 [hep-th/0005067] [INSPIRE].

[66] D. Berenstein, V. Jejjala and R.G. Leigh, *The Standard model on a D-brane*, *Phys. Rev. Lett.* **88** (2002) 071602 [hep-ph/0105042] [INSPIRE].

[67] P. Candelas, C.A. Lütken and R. Schimmrigk, *Complete intersection Calabi-Yau manifolds. Part 2. Three generation manifolds*, *Nucl. Phys. B* **306** (1988) 113 [INSPIRE].

[68] P. Candelas and R. Davies, *New Calabi-Yau Manifolds with Small Hodge Numbers*, *Fortsch. Phys.* **58** (2010) 383 [arXiv:0809.4681] [INSPIRE].

[69] P. Candelas and A. Constantin, *Completing the Web of $\mathbb{Z}_3$-Quotients of Complete Intersection Calabi-Yau Manifolds*, *Fortsch. Phys.* **60** (2012) 345 [arXiv:1010.1878] [INSPIRE].

[70] V. Braun, *On Free Quotients of Complete Intersection Calabi-Yau Manifolds*, *JHEP* **04** (2011) 005 [arXiv:1003.3235] [INSPIRE].

[71] A. Lukas and C. Mishra, *Discrete Symmetries of Complete Intersection Calabi-Yau Manifolds*, *Commun. Math. Phys.* **379** (2020) 847 [arXiv:1708.08943] [INSPIRE].

[72] C. Mishra, *Calabi-Yau manifolds, discrete symmetries and string theory*, Ph.D. Thesis, University of Oxford, Oxford, U.K. (2017).

[73] P. Candelas and C. Mishra, *Highly Symmetric Quintic Quotients*, *Fortsch. Phys.* **66** (2018) 1800017 [arXiv:1709.01081] [INSPIRE].

[74] A. Braun, A. Lukas and C. Sun, *Discrete Symmetries of Calabi-Yau Hypersurfaces in Toric Four-Folds*, *Commun. Math. Phys.* **360** (2018) 935 [arXiv:1704.07812] [INSPIRE].

[75] P. Candelas, A. Constantin and C. Mishra, *Hodge Numbers for CICYs with Symmetries of Order Divisible by* 4, *Fortsch. Phys.* **64** (2016) 463 [arXiv:1511.01103] [INSPIRE].

[76] S. Krippendorf and M. Syvaeri, *Detecting symmetries with neural networks*, *Mach. Learn. Sci. Technol.* **2** (2020) 015010.

[77] D. Beymer and T. Poggio, *Face recognition from one example view*, in proceedings of *IEEE International Conference on Computer Vision*, Cambridge, MA, U.S.A., 20–23 June 1995, IEEE, New York, NY, U.S.A. (1995), pp. 500–507.

[78] P. Niyogi, F. Girosi and T. Poggio, *Incorporating prior information in machine learning by creating virtual examples*, *Proc. IEEE* **86** (1998) 2196.

[79] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. Salakhutdinov and A. Smola, *Deep sets*, arXiv:1703.06114.

[80] T.S. Cohen and M. Welling, *Group equivariant convolutional networks*, in proceedings of the *33rd International Conference on International Conference on Machine Learning (ICML'16)*, New York, NY, U.S.A., 19–24 June 2016, pp. 2990–2999.

[81] M. van der Wilk, M. Bauer, S. John and J. Hensman, *Learning invariances using the marginal likelihood*, arXiv:1808.05563.

[82] L. Deng, *The mnist database of handwritten digit images for machine learning research [best of the web]*, *IEEE Signal Process. Mag.* **29** (2012) 141.

[83] A. Jacot, F. Gabriel and C. Hongler, *Neural tangent kernel: Convergence and generalization in neural networks*, arXiv:1806.07572.

[84] R. Brüel-Gabrielsson and G. Carlsson, *Exposition and interpretation of the topology of neural networks*, arXiv:1810.03234v2.

[85] M. Gabella, N. Afambo, S. Ebli and G. Spreemann, *Topology of learning in artificial neural networks*, arXiv:1902.08160.

[86] B. Rieck et al., *Neural persistence: A complexity measure for deep neural networks using algebraic topology*, arXiv:1812.09764.

[87] D. Klaewer and L. Schlechter, *Machine Learning Line Bundle Cohomologies of Hypersurfaces in Toric Varieties*, *Phys. Lett. B* **789** (2019) 438 [arXiv:1809.02547] [INSPIRE].

[88] C.R. Brodie, A. Constantin, R. Deen and A. Lukas, *Machine Learning Line Bundle Cohomology*, *Fortsch. Phys.* **68** (2020) 1900087 [arXiv:1906.08730] [INSPIRE].

JHEP08(2022)105

[89] C.R. Brodie, A. Constantin, R. Deen and A. Lukas, *Index Formulae for Line Bundle Cohomology on Complex Surfaces*, *Fortsch. Phys.* **68** (2020) 1900086 [arXiv:1906.08769] [INSPIRE].

[90] M. Raissi, P. Perdikaris and G.E. Karniadakis, *Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations*, *J. Comput. Phys.* **378** (2019) 686 [arXiv:1711.10561] [INSPIRE].

[91] Z. Li et al., *Fourier neural operator for parametric partial differential equations*, arXiv:2010.08895.

[92] M. Headrick, S. Kitchen and T. Wiseman, *A New approach to static numerical relativity, and its application to Kaluza-Klein black holes*, *Class. Quant. Grav.* **27** (2010) 035002 [arXiv:0905.1822] [INSPIRE].

[93] T. Wiseman, *Numerical construction of static and stationary black holes*, in *Black holes in higher dimensions*, G.T. Horowitz ed., Cambridge University Press, Cambridge, U.K. (2012), pp. 233–270 [arXiv:1107.5513] [INSPIRE].

[94] O.J.C. Dias, J.E. Santos and B. Way, *Numerical Methods for Finding Stationary Gravitational Solutions*, *Class. Quant. Grav.* **33** (2016) 133001 [arXiv:1510.02804] [INSPIRE].

JHEP08(2022)105