preprint number
June 2019

Numerical CY metrics from holomorphic networks

Michael R. Douglas, Subramanian Lakshminarasimhan and Yidi Qi

Department of Physics and Simons Center for Geometry and Physics, Stony Brook University

Abstract

We try ML inspired ansatzes for computing Ricci-flat Kähler metrics.
Dedicated to the memory of Tudor Ciobanu.

1 Introduction

Kähler manifolds with Ricci flat metrics, known as Calabi-Yau manifolds, are the first and still the most important starting point for compactification of string theory to produce realistic physical models. Their study led to the discovery of mirror symmetry and a great deal of interesting mathematics. A brief survey of these topics appears in [**?**], and textbooks on mirror symmetry include [**?**, **?**].

No closed form expressions are known for these Ricci flat metrics, and it is generally believed that none exist.[1] One can get an approximate description using methods of numerical general relativity, as pioneered by Headrick and Wiseman [**?**]. The special properties of Kähler geometry lead to many simplifications, starting with the representation of the metric by a single function. Donaldson introduced many ideas such as a representation by projective embeddings and approximation by balanced metrics [**?**]. Subsequent works simplified and improved the numerical methods [**?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**, **?**] so that a fairly simple program can get accurate results.

To simplify a bit, the approach used in these works is to expand the Kähler potential in a polynomial basis. As is the case for spectral methods, this is very good for approximating smooth functions with variations on length scales $1/k$, where k is the order of the polynomials used. This is sometimes good enough, but by varying moduli one can easily produce metrics with structure on arbitrarily short scales, say by approaching a singular limit, as was studied in [**?**]. And since on a D-dimensional manifold, the number of basis functions grows as $\mathcal{O}(k^D)$, one cannot take k very large (the "curse of dimensionality"). Indeed, almost all

[1] But see [**?**, **?**] for an analytic approach to the K3 metric.

of these works restrict attention to manifolds with large discrete symmetry groups to get a manageable number of coefficients.

In recent years, there has been much success with machine learning (ML) inspired approaches to scientific computation, and especially the solution of PDE's such as Navier-Stokes and the Schrödinger equation. This is a rapidly developing area and we are not aware of a review covering the material we will use, but a simple illustration of the approach we follow is [**?**]. This work solves problems in structural optimization – mathematically, the unknown is a single function in a bounded domain of $\mathbb{R}^2$ and the solution is the minimum of a nonlocal energy functional (the integrated stress). What is novel is to represent the unknown function, not in terms of finite elements or a Fourier basis, but as the output of a neural network. This provides a space of approximating functions parameterized by the network weights, which can be efficiently computed using technologies developed for machine learning. Standard numerical optimization applied to the energy considered as a function of the weights leads to high quality solutions, which are not limited by the constraints of previous methods.

These methods have only begun to be explored for higher dimensional geometry. Works which study Calabi-Yau metrics include [**?**, **?**]. In [**?**], the Kähler potential is represented by a gradient boosted decision tree. The problem is solved at various lower degrees k, and the ML model does extrapolation in the degree (Richardson extrapolation). The work [**?**] develops a method more similar to what we will describe, with separate functions for the metric components and explicitly imposing the Kähler condition.

The idea we propose and study here is to represent the Kähler potential as the output of a **holomorphic network**, a feedforward neural network with complex weights, whose inputs are the complex coordinates or low degree sections, and which uses the activation function $z \to z^2$, a holomorphic function. Its outputs are a subspace of the space of holomorphic sections which is parameterized by weight matrices in a simple and efficiently computable way. Thus, we can find the metric in this class which is closest to Ricci flat by numerical optimization of an error function with respect to the weights, much as in ref. [**?**].

Since each layer doubles the degree of the section, one can in principle work with large values of k. One faces the usual problems of deep networks in ML, such as exploding gradients, but one can use ML techniques to deal with them, such as a variation on batch normalization.

This formulation of the numerical Calabi-Yau metric problem is formally very similar to supervised learning – both problems amount to fitting a function given a set of input-output values (x_i, y_i), where x_i is sampled from an input distribution and y_i is known. Thus we can use a standard ML package (TensorFlow) to construct our networks and do the optimization. Our code is described in §**??** and is available at `https://github.com/yidiq7/Manifold` .

We obtain results for a variety of quintic hypersurfaces, including the familiar one-parameter family of Fermat quintics, and also hypersurfaces with much less symmetry. Our main focus is to understand the accuracy of these metrics and its dependence on parameters, both degree k and the moduli and amount of symmetry of the manifold.

A brief summary of the approach and the mathematical questions it raises can be found

in [?].

Polynomial activation functions have appeared in the ML literature [?, ?], but our motivations (holomorphy and contact with higher dimensional geometry) are new.

2 Numerical methods inspired by machine learning

In this section we alternate between the points specific to our problem and review of the concepts we use from machine learning. Some useful background material includes [?, ?] on complex geometry and [?] on machine learning. We also review the basic definitions we use from complex geometry in Appendix A.

2.1 Review of the embedding method

Following Donaldson [?], most work on numerical Calabi-Yau metrics represents the metric using an embedding by holomorphic sections of a very ample line bundle $\mathcal{L}$. Let us briefly review this.

We recall that a line bundle is defined by choosing patches U_α on the manifold M and holomorphic transition functions $f_{\alpha\beta}$ on the overlaps of patches $U_\alpha \cup U_\beta$ satisfying the consistency conditions $f_{\alpha\beta}f_{\beta\gamma}f_{\gamma\alpha} = 1$. A section s is then a holomorphic function s_α on each patch satisfying $s_\alpha = f_{\alpha\beta}s_\beta$. Now this is ambiguous considered as a function, because we can always multiply by a set of holomorphic functions λ_α defined on each patch, taking $s_\alpha \to \lambda_\alpha s_\alpha$. But the ratio of a pair of sections is unambiguous. This can be rephrased as the statement that a vector of N sections is an unambiguous map ι from M to $\mathbb{CP}^{N-1}$, since λ_α acts to rescale the entire vector. The very ample condition then implies that the map ι is an embedding.

Pulling back the Fubini-Study metric on $\mathbb{CP}^{N-1}$, the embedding then leads to the Kähler potential

$$K = \log \sum_{I,\bar{J}} h_{I,\bar{J}} s^I \bar{s}^{\bar{j}} \tag{1}$$

where s^I is a basis of $N = h_0(\mathcal{L})$ holomorphic sections. This gives us an N^2 real dimensional family of metrics parameterized by the hermitian matrix $h_{I,\bar{J}}$. One then does integrals over M by Monte Carlo, as in [?]. This setup can be used in various ways, but conceptually the simplest is to choose a measure of how close the metric is to Ricci flatness and numerically optimize it, as done in [?]. This is easy to program (as we will review below), especially compared with methods that use explicit coordinate patches or a grid.

Let us focus on the case of M a hypersurface $f = 0$ in a weighted projective space W. The homogeneous coordinates will be Z^i with degrees n_i. Now a holomorphic section is simply a weighted homogeneous polynomial, so this ansatz is easy to work with mathematically. Physically one can define a holomorphic section as a ground state of a magnetic Schrödinger operator, *i.e.* a state in the lowest Landau level (see for example [?]). Thus, in the terminology of numerical methods, this is a spectral method. A spectral method expands continuous

functions in terms of the low lying eigenfunctions of some operator, such as the translation operator (leading to a Fourier basis) or a Laplacian. The upper cutoff on eigenvalue corresponds to a fixed lower cutoff on the length scale, and thus spectral methods are usually not well adapted to problems with multiple scales.

In our problem, let us denote the degree of the polynomial (equivalently of the line bundle) by k. An expansion in polynomials of degree k can represent functions with up to k nodes, so with variations on length scales roughly down to $1/k$. However, by adjusting moduli it is easy to produce CY metrics with multiple scales – for example, we might have a small resolved cycle in a large bulk manifold. Thus we would like a more flexible ansatz than Eq. (1) but one which is still easy to program.

One option would be to use a higher degree line bundle and a subset of its sections, keeping more sections in regions with more variation. Mathematically, this amounts to restricting the rank of the matrix $h_{I\bar{J}}$. This could be done by imposing the form $h = U^\dagger \lambda U$ with λ a diagonal matrix of restricted rank. Let us denote the space of metrics of the form Eq. (1) with a degree k embedding and in which $\operatorname{rank} h \leq D$ as $\mathcal{G}_{k,D}$. It has real dimension $\mathcal{O}(ND)$ so this mitigates the curse of dimensionality, however to use this truncation we have to search for an appropriate subspace of sections, which is not straightforward. Thus we leave this interesting idea for future investigation.

2.2 Feed-forward networks

Let us briefly review the definition of a feed-forward network (FFN, also called MLP for multilayer perceptron). It is a parameterized space of functions

$$F_w : \mathcal{X} \to \mathcal{Y}, \tag{2}$$

with an input $x \in \mathcal{X} \cong \mathbb{R}^D$ and an output $y \in \mathcal{Y} \cong \mathbb{R}^{D'}$ (we will generally take $D' = 1$). It is the composition of a series of functions or layers indexed $0, 1, \ldots, d$. The number of layers $d + 1$ is the depth.

The parameters or weights w are a series of matrices $\{w_{(0)}, w_{(1)}, \ldots, w_{(d)}\}$ which enter the following expression:

$$\begin{aligned} y_j(w;x) &= \sum_{i_d=1}^{D_d} w^{(d)}_{j,i_d} \theta(z^{i_d}_{(d-1)}) && (3) \\ &\vdots \\ z^{i_2}_{(1)} &= \sum_{i_1=1}^{D_1} (w_{(1)})^{i_2}_{i_1} \theta(z^{i_1}_{(0)}) && (4) \\ z^{i_1}_{(0)} &= \sum_{i=1}^{D} (w_{(0)})^{i_1}_{i} x^i && (5) \\ & && (6) \end{aligned}$$

The "activation function" $\theta(x)$ is a nonlinear function, for which a popular choice is the "ReLU" function

$$\theta_{ReLU}(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases} . \tag{7}$$

The term "unit" is sometimes used to denote a sum followed by a single application of θ, so this network will have $D_1 + \ldots + D_d$ units.

It has been shown that feed-forward networks can approximate arbitrary functions. This is the case even for depth two ($d = 1$) [**?**], but in this case one can need an exponentially large number of units, as would be the case for simpler methods of interpolation (the "curse of dimension"). By using more layers, one can gain many advantages – complicated functions can be represented with many fewer units, and local optimization techniques are much more effective. How exactly this works is not well understood theoretically and there are many interesting observations and hypotheses as to how these advantages arise.

As an aside, one basic application of this in physics and statistics is to use a feed-forward network with a single output to represent a family P_w of probability distributions on $\mathcal{X}$ parameterized by the weights w, by taking the output y to be the probability density function.[2] The expectation value of a function $\mathcal{O}(x)$ in the distribution P_w is then

$$\mathbb{E}_w[\mathcal{O}] \equiv \frac{\int d\mu(x)\ y(w;x)\,\mathcal{O}(x)}{\int d\mu(x)\ y(w;x)}. \tag{8}$$

Here $d\mu(x)$ is a measure on $\mathcal{X}$, which mathematically is required to turn the function $y(w;x)$ on $\mathcal{X}$ into a density on $\mathcal{X}$. Given coordinates x^i on $\mathcal{X}$, this could be Lebesgue measure, but we are free to make other choices.

In a high dimensional configuration space $\mathcal{X}$, one often does these integrals by sampling, choosing a set of N_p points $x_{(i)} \in \mathcal{X}$ and taking

$$d\mu(x) = \frac{1}{N_p} \sum_{i=1}^{N_p} \delta(x - x_{(i)}). \tag{9}$$

The sample points $x_{(i)}$ could be distributed according to an *a priori* probability measure Dx on $\mathcal{X}$, or we could adapt the sampling to prefer regions with large measure (importance-based sampling). Many options have been developed in statistics.

An analogous idea could be used to get a parameterized family of quantum wave functions, in this case taking all of (x, w, y) to be complex, and taking

$$\mathbb{E}_w[\mathcal{O}] \equiv \frac{\int d\mu(x)\ |y(w;x)|^2 \mathcal{O}(x)}{\int d\mu(x)\ |y(w;x)|^2}. \tag{10}$$

One would also have to choose an appropriate $\theta(x)$.

[2] Another very common choice for representing a probability distribution over a finite set S is to use a network with a separate output y_a for each option $a \in S$, and take the final layer to be the "softmax" $P(a) = \exp y_a / \sum_{b \in S} \exp y_b$ to get a normalized probability distribution $P(a)$.

2.3 Multilayer holomorphic embeddings

The idea we will pursue in this work is to use Eq. (3) to define a subspace of sections. Thus we take the input space $\mathcal{X}$ to be the ambient weighted projective space, and we choose $\theta(x)$ to be a nonlinear homogeneous holomorphic function. The simplest choice and the one we will use is to take

$$\theta(x) = x^2. \tag{11}$$

We then take the successive layers Eq. (5), Eq. (15), *etc.* to define subspaces of a space of sections. To get a real valued Kähler potential, we replace Eq. (3) with

$$K(w;Z) = \log \sum_{i_d=1,\bar{j}_d=1}^{D_d} h^{(d)}_{i_d,\bar{j}_d} z_d^{i_d} \bar{z}_d^{\bar{j}_d} \tag{12}$$

$$z_d^{i_d} \equiv \theta(z^{i_d}_{(d-1)}) \tag{13}$$

where $z_{(d-1)}$ and the previous layers are as above.

This construction gives us a class of metrics for each choice of depth d and layer widths $D_1, \ldots, D_d$, obtained from embeddings with degree $k = 2^d$. The total number of real weights is

$$N_w = 2\left(DD_1 + D_1D_2 + \ldots + D_{d-1}D_d\right) + D_d^2. \tag{14}$$

Generally $D_i < h_0(2^i)$ so this will not span the complete basis of sections, in other words we have restricted the embedding and are only using a subset of metrics. While the final layer z_d is a linear subspace of $H^0(\mathcal{L}^{2^d})$, this subspace is nonlinearly parameterized by the weights. The hope is that it is flexible enough to describe the metrics of interest.

A variation on this is to take the inputs to be a complete basis of sections s^I of degree k_0, so

$$z^{i_1}_{(0)} = \sum_{I=1}^{h_0(k_0)} (w_{(0)})^{i_1}_i s^I \tag{15}$$

where I is an index for the basis. For the hypersurface $f = 0$ in $\mathbb{CP}^{D-1}$, the basis will be the symmetrized degree k_0 monomials, quotiented by the ideal generated by f (if $k_0 \geq \deg f$).

Other combinations of layers and activation functions are of course possible, subject to the constraint that every activation (intermediate value) is homogeneous (a section of a definite line bundle). Thus one cannot have skip connections, but one could take other products of outputs from previous layers.

2.4 Biholomorphic embeddings

Another variation is to use biholomorphic sections, meaning products of holomorphic and antiholomorphic sections, as the activations (intermediate values) of the network. Thus, we

would take

$$x^{i_1}_{(0)} = \sum_{i,j=1}^{D} (w_{(0)})^{i_1}_{i,\bar{j}} z^i \bar{z}^{\bar{j}} \tag{16}$$

$$x^{i_2}_{(1)} = \sum_{i=1}^{D_1} (w_{(1)})^{i_2}_{i_1} \theta(x^{i_1}_{(0)}) \tag{17}$$

$$\vdots \tag{18}$$

$$K(w;Z) = \log \sum_{i=1}^{D_d} (w_{(d)})_i \theta(x^i_{(d-1)}). \tag{19}$$

The first layer weights $w_{(0)}$ should be a hermitian matrix, so that the activations and subsequent weight matrices will all be real. One still needs the activation function θ to be homogeneous, and $z \to z^2$ is still the natural choice. The total number of real weights is now

$$N_w = D^2 + DD_1 + D_1 D_2 + \ldots + D_{d-2} D_{d-1} + D_d. \tag{20}$$

Again, there are many variations on this.

A potential advantage of this representation over the holomorphic network is that the higher degree space of Kähler potentials contains the lower degree spaces in a simpler way. For example, suppose we wanted to represent

$$K = \left(\sum_{i=1}^{D} |z_i|^2 \right)^n \tag{21}$$

with $n = 2^d$. This is a degree n Kähler potential which leads to exactly the same metric as a degree 1 potential (up to an overall rescaling). Representing it with the holomorphic network requires exponentially wide intermediate layers, but the biholomorphic network can do it with width 1 intermediate layers.

2.5 Generalizations to other ambient spaces; discrete symmetries

A sensible network must respect the gradings etc.

For hypersurfaces in weighted projective space, we need to modify the construction. The simplest option is to take the inputs to be a basis of all sections of the same degree as the defining function f. Another option is not to have a single feed-forward structure but rather a DAG of layers connected in all ways consistent with homogeneity. One might also generalize the activation function, allowing two layers with outputs z and z' to send the product zz' to the next layer.

Next let us consider toric varieties. In physics terms, a toric variety is the moduli space of a $U(1)^r$ gauge theory with four supercharges (say, a $(2,2)$ sigma model). There are many equivalent mathematical definitions; the closest to the physics is that it is the symplectic

(or GIT) quotient of $\mathbb{C}^N$ by a $U(1)^r$ action. Thus a toric manifold M is specified by an $N \times r$ integral matrix Σ of $U(1)^r$ charges and a moment map, a real-valued function on the Lie algebra of $U(1)^r$. If it is a smooth manifold, it will have $b^2 = r$, and the fields (the coordinates on $\mathbb{C}^N$) are sections of line bundles with $c_1(\mathcal{L})$ given by the corresponding row of Σ (say this in a more mathematical way). We can then multiply coordinates and take weighted sums in any way that preserves the grading, to get a nonlinearly defined subspace of the space of sections of some $\mathcal{L}$. All of these constructions should be unified in some representation theoretic way.

A basic example is to take all of the monomials in the defining polynomial, *i.e.* a basis of sections of $\mathcal{N}$, project to a subspace, and take the sum of the squares of the resulting functions. Thus we are comparing the representing power of $\mathrm{Sym}^2 H^0(\mathcal{L})$ with $\sigma(P \cdot H^0(\mathcal{L})$ for some projector P.

How can we compare the general ability to represent functions of a basis and a subset? We can define the associated kernels, which here are the Bergman kernels. Given a metric on $\mathcal{L}$ and a volume form, we can define the inner product on sections and the corresponding Bergman kernel

$$\rho(z, \bar{z}') = \sum_i s(z)\bar{s}(\bar{z}'). \tag{22}$$

We can then look at $\rho_1 - \rho_2$ and show that it has small norm in some sense. This will probably not be the case, but ρ_2 is parameterized and what we really care about is how well the best ρ_2 can approximate ρ_1.

2.6 Supervised learning, sampling and data

So far our discussion has not had much to do with machine learning. Let us briefly review a simple ML problem, supervised learning, to see the similarities and differences. It turns out that our problem is similar enough to this that we could try to adapt standard ML software to do our computations, however as we will explain there are some technical problems in doing this.

In supervised learning, we have a data set of N_{data} items, each of which is an input-output pair (x_n, y_n). These are supposed to be drawn from a probability distribution $\mathcal{P}$ on $\mathcal{X} \times \mathcal{Y}$. The goal is to choose the function Eq. (2) from $\mathcal{X}$ to $\mathcal{Y}$ which best describes the general relation $\mathcal{P}$ between input and output, in the sense that it minimizes some definition of the expected error (an objective or "loss" function). The procedure of making this choice given the data set (and perhaps limited "prior" information about $\mathcal{P}$) is called training the network.

A simple choice of objective function is the mean squared error,

$$\mathcal{E} = \mathbb{E}_{\mathcal{P}}\left[(f_w(x) - y)^2\right]. \tag{23}$$

If we estimate this by evaluating it on our data set, we get the training error

$$\mathcal{E}_{train} = \frac{1}{N_{data}} \sum_{n=1}^{N_{data}} (f_w(x_n) - y_n)^2. \tag{24}$$

A standard ML training procedure is the following. We start with an FFN as in Eq. (3), with the weights initialized to random values – in other words, we draw the w from some distribution independent of the data. A common choice is for each matrix element $w_{i_m}^{(m),i_{m+1}}$ to be an independent Gaussian random variable with mean zero and variance $1/\sqrt{D_m}$. This choice is made so that the expected eigenvalues of the weight matrix are order D_m^0.

The next step is to minimize Eq. (24) as a function of the weights. A simple algorithm for this is gradient descent, a stepwise process in which the weights at time $t+1$ are derived from those at t as

$$w(t+1) = w(t) - \epsilon(t) \frac{\partial \mathcal{E}_{train}}{\partial w}\bigg|_{w=w(t)} . \tag{25}$$

While this will only find a local minimum, it works better for these problems than one might have thought. One trick for improving the quality of the result is to make the step size $\epsilon(t)$ decrease with time, according to a "learning schedule" chosen empirically to get good results for the task at hand.

An improvement on this procedure is "stochastic gradient descent" or SGD. This is much like Eq. (25) except that instead of evaluating the training error $\mathcal{E}_{train}$ on the full data set, one evaluates it on a subset or "batch," with the batch varied from one step to the next so that their union covers the full data set. This was originally done for computational reasons but it also turns out to produce a noise term with beneficial properties, for example in helping to escape local minima.

Finally, once the optimization is deemed to have converged, one judges the results by estimating Eq. (23). This estimate must be made by using an independent data set from that used in training as otherwise we are rewarding our model for matching both signal and noise.[3] However in most applications we do not have any direct access to $\mathcal{P}$, rather we only have an empirical data set. Thus one starts by dividing the full data set into disjoint "training" and "testing" subsets, evaluates Eq. (24) on the training set for training, and then evaluates the sum of errors over the testing set to estimate $\mathcal{E}$. The final model can be very accurate, surprisingly so when compared to expectations from standard statistical theory. Let us cite [] as a few papers which study these theoretical questions.

While our problem is not one of supervised learning, it will be useful to phrase it in terms as similar as possible, so that we can most easily use ML software. The workflow of the supervised learning task involves defining a set of data points (x_n, y_n) which are independent of the weights, repeated evaluation of the network at each x_n to get a prediction $f(x_n)$ for the corresponding y_n, and optimization of an objective function which is a sum of terms which each depend on a single data point. The network is normally defined by concatenating layers, such as multiplication by a weight matrix (a fully connected layer), application of an activation function, and so on. These layers are implemented in associated software libraries, such as Keras for Tensorflow. As we explain next, while we will have to implement some new layers for our problem, otherwise our workflow is the same.

[3] In classification problems, one often uses networks with many more parameters than data points and which can completely fit the dataset, so that the minimum of $\mathcal{E}_{train}$ is zero! In this case $\mathcal{E}_{train}$ is clearly a poor estimate for $\mathcal{E}$.

2.7 Batch normalization

A very common problem with deep networks is the exploding/vanishing gradient problem. To see this, consider the gradient of the objective function Eq. (23),

$$\frac{\partial}{\partial w}\mathcal{E} = 2\mathbb{E}_{\mathcal{P}}\left[(f_w(x) - y)\cdot\frac{\partial f_w(x)}{\partial w}\right]. \tag{26}$$

Using the chain rule, the gradients of f for a given layer can be written as a product of the gradient of the subsequent layers with respect to their inputs, times the output of the preceding layers. These products include a weight at each layer, so one has the general scaling $\nabla\mathcal{E} \sim w^{d-1}$. Using this in gradient descent, one has $\dot{w} \sim w^{d-1}$, which tends to drive the weights to infinity or zero.

One of the most successful solutions to this problem is batch normalization [**?**]. Rather than operate directly on the weights, the basic idea is to "whiten" the inputs to each layer. For each component of the input, one subtracts its mean value mean and rescales by the variance to standardize its distribution. This removes the dependence on the scales of weights from other layers. Furthermore, it is a differentiable transformation and thus one can optimize a network with such a layer.

Batch normalization has two further ingredients. First, additional learned weights (β, γ) are provided to compensate for the standardization. Explicitly, if the standardized data points are $\hat{x}_i$, one uses instead $\gamma\hat{x}_i + \beta$, with the new weights optimized along with the original ones. Second, the standardization is done on a per-batch basis, again to reduce the computational requirements.

Does batch normalization, either in its standard definition or some variant, make sense in our setup? For us, an overall rescaling of the vector of sections by a local holomorphic function λ is allowed, but not individual rescalings of the components. Furthermore, shifts are not allowed. Thus standard batch normalization is not sensible, but we can take the same rescaling for every component,

$$z^i_{(n)} \to \lambda(z_{(n)})z^i_{(n)}. \tag{27}$$

Here λ could be a general function, but the natural choice in order to normalize the vector is

$$z^i_{(n)} \to \hat{z}^i_{(n)} = \frac{\gamma}{\sqrt{\sigma^2+\epsilon}}\cdot z^i_{(n)} \tag{28}$$

$$\sigma^2 \equiv \mathbb{E}_{\text{batch}}\left[\sum_i |z^i_{(n)}|^2\right]. \tag{29}$$

This could be done per-batch, or even on a per point basis. The weight γ is redundant with the next layer weights, but using it changes the optimization problem, so we might include it. The parameter ϵ is included for numerical stability and should be small.

Finally, the backpropagation through this layer uses

$$\frac{\partial \hat{z}^i_{(n)}}{\partial z^j_{(n)}} = \frac{\gamma}{\sqrt{\sigma^2+\epsilon}}\delta^i_j - \frac{\gamma}{2(\sigma^2+\epsilon)^{3/2}} z^i_{(n)} \frac{\partial \sigma^2}{\partial z^j_{(n)}} \tag{30}$$

$$= \frac{\gamma}{\sqrt{\sigma^2+\epsilon}} - \frac{\gamma}{2(\sigma^2+\epsilon)^{3/2}} z^i_{(n)} \bar{z}^j_{(n)}. \tag{31}$$

There is a similar expression for the biholomorphic network in terms of the real quantities $x^i_{(n)}$,

$$\frac{\partial \hat{x}^i_{(n)}}{\partial x^j_{(n)}} = \frac{\gamma}{\sqrt{\sigma^2+\epsilon}} - \frac{\gamma}{(\sigma^2+\epsilon)^{3/2}} x^i_{(n)} x^j_{(n)}. \tag{32}$$

Presumably, we can modify the Tensorflow batch normalization layer to do this.

3 Implementation

3.1 Geometric quantities

There is the holomorphic three-form

$$\Omega = \dots \tag{33}$$

and the associated volume form

$$d\mu_\Omega \equiv \mathcal{N}_\Omega \Omega \wedge \bar{\Omega}. \tag{34}$$

The normalization $\mathcal{N}_\Omega$ will be described shortly. This volume form depends on the complex structure but is independent of the Kähler form and thus the embedding we use to represent M.

The Kahler form and associated volume element are

$$\omega_g = \partial_i \partial_{\bar{j}} K \tag{35}$$

$$d\mu_g \equiv \omega_g^3 = \det \omega_g = \det_{i,\bar{j}} \partial_i \partial_{\bar{j}} K. \tag{36}$$

Now the integral $\int_M \omega_g^3$ is a topological invariant, so one can choose the normalization of $d\mu_\Omega$ to make $\eta = 1$ for the Ricci flat metric.

3.2 General implementation

This follows the approach taken in [] with the main difference being the use of the feed-forward network Eq. (12).

We start by sampling points on M for doing integrals, using the procedure of [**?**]. This sample will play the role of the input dataset in a supervised learning task. It will be naturally

distributed according to the Fubini-Study volume element and thus (using a theorem of Shiffman-Zelditch)

$$\lim_{N_{data}\to\infty} \frac{1}{N_{data}} \sum_i f(x_i) = \int_M \omega_{FS}^3 f(x), \tag{37}$$

where the Fubini-Study metric ω_{FS} and volume element is the special case of Eq. (35) with $k = 1$ and a fixed hermitian metric $h_{i\bar{j}}$. Given such a metric, to sample from M we first choose two points a and b on the ambient WP space[4] from a normal distribution with covariance $h_{i\bar{j}}$. We then find the points on the intersection of the line $\lambda a + \rho b$ with M, in other words the choices of λ/ρ for which $f(\lambda a + \rho b) = 0$. This equation will have $\deg f$ solutions (with multiplicity) and for our purposes (which do not look at correlations between points) we can simply add all of them to the data set as inputs x_n, evaluating the variable y_n for each.

By reweighting, we define an empirical approximation to $d\mu_\Omega$, which is

$$d\mu_{\Omega,\mathcal{D}} = \sum_{i\in\mathcal{D}} \frac{|\Omega(x_i)|^2}{\omega_{FS}(x_i)^3}. \tag{38}$$

The objective function measures the deviation of the metric from Ricci flatness. The simplest such measure is

$$\eta = \frac{\omega_g^3}{\Omega \wedge \bar{\Omega}} \tag{39}$$

The "least squares error" for Ricci flatness is then

$$\begin{aligned} \mathcal{E} &= \int_M \Omega \wedge \bar{\Omega} \, (\eta - 1)^2 && (40)\\ &= \int_M \omega_{FS}^3 \frac{\left(\omega_g^3 - \Omega \wedge \bar{\Omega}\right)^2}{\omega_{FS}^3 \Omega \wedge \bar{\Omega}} && (41)\\ &= \int_M \omega_{FS}^3 \frac{(f_w(x_i) - y_i)^2}{y_i} && (42) \end{aligned}$$

where

$$\begin{aligned} y_i &= \frac{\Omega \wedge \bar{\Omega}}{\omega_{FS}^3} && (43)\\ f_w(x_i) &= \frac{\omega_g^3}{\omega_{FS}^3}. && (44) \end{aligned}$$

The point of writing it this way is that y_i is independent of the weights, so we can also consider it as part of the "data set," the target value for f_w evaluated at the point x_i. An ML package will provide common objective functions such as least squared error and we will be able to use this one directly.

[4]More details on computations on WP can be found in the Penn group papers.

Note that we can choose a different function of η which has the same optimum yet has a statistics/information theory interpretation. For example, if we consider a volume form on the CY as a probability density, it is natural to look at the KL divergence:

$$\mathrm{KL}\left(d\mu_\Omega | d\mu_g\right) = -\int_M d\mu_\Omega \log \eta. \tag{45}$$

This will also have its minimum at the Ricci-flat metric. Still, it may be less convenient for optimization, so we compute with the least squares definition.

Thus we need to construct layers which implement Eq. (12), the two derivatives which produce $g_{i\bar{j}}$, and the nonlinear definition of f_w in terms of this. Now optimization requires evaluating the gradient of f_w and this is done by the backpropagation algorithm built into the ML package. Thus there is already a built-in ability to compute derivatives, the question is how to use it to compute the three derivatives implicit in the definition grad $\mathcal{E}$. This should be possible because we can chain gradient computations, for example see [**?**].

Now given that the inputs to the network are the coordinates Z^a of a point x_i, the derivatives with respect to Z can be obtained by interpolating an additive layer $Z \to Z + b$ and taking the b derivative. Presumably, we can combine these to get the required second and third derivatives.

3.3 Adaptive method

We could vary the dataset to lower the variance of the objective function. Interesting to work out but not clear it is worth implementing.

4 Results

4.1 Examples

1. The Fermat quintics Eq. (**??**).

2. A two parameter family with less symmetry,

$$f_1 = z_0^5 + z_1^5 + z_2^5 + z_3^5 + z_4^5 + \psi z_0 z_1 z_2 z_3 z_4 + \phi(z_3 z_4^4 + z_3^2 z_4^3 + z_3^3 z_4^2 + z_3^4 z_4) \tag{46}$$

3. Another two parameter family,

$$\begin{aligned} f_2 &= z_3 g(z) + z_4 h(z) && (47)\\ g &= z_0^4 + z_1^4 + z_2^4 + z_3^4 + a z_0 z_1 z_2 z_3 && (48)\\ h &= z_0^4 + z_1^4 + z_2^4 + z_4^4 + b z_0 z_1 z_2 z_4 && (49)\\ & && (50) \end{aligned}$$

4. Possibly, some randomly chosen f's.

Each should be worked out with

1. The standard space of metrics Eq. (1) for as many k's as we can reasonably do.
2. A two layer holomorphic network, where the inputs are the coordinates.
3. A two layer holomorphic network, where the inputs are quadratic ($k = 2$) and cubic ($k = 3$) in the coordinates.
4. Two layer biholomorphic networks with inputs $z\bar{z}$ and $z^2\bar{z}^2$.
5. Maybe, a few examples of three layer networks, perhaps with batch norm.

4.2 Observations

For each of these, we want the training and testing η and $\delta\eta$, as we are computing now. We probably also want the maximum $|\eta - 1|$.

Graph these as functions of the number of layers and number of parameters. One fair way to compare all of the different networks is to make the x-axis the total number of real parameters.

4.3 Interpretation

There are two geometric properties of the CY which we expect to control the accuracy. One is the smallest length scale on the CY. The other is the "complexity" of the manifold, a somewhat vague concept, but somehow inversely related to the amount of symmetry. Both of these properties are hard to measure and thus we look at measurable proxies.

A reasonable proxy for the smallest length scale is the distance in moduli space to the nearest singular Calabi-Yau manifold, for which $f = \nabla f = 0$ has a simultaneous solution in $\mathbb{CP}^4$. For example, for the Fermat family of quintics Eq. (**??**), these equations are Eq. (**??**) and $5z_i^5 = -\psi z_1 z_2 z_3 z_4 z_5 \, \forall 1 \leq i \leq 5$, which has a solution if $\psi = -5\omega$ for ω any fifth root of unity. The limit $\psi \to \infty$ is also singular.

The generic singularity of a hypersurface is an ordinary double point (ODP). In a small neighborhood of a $D = 3$ ODP singularity it looks like

$$z_1^2 + z_2^2 + z_3^2 + z_4^2 = \epsilon \tag{51}$$

and the singular limit is $\epsilon \to 0$. This is usually called the conifold singularity in the string theory literature. A Ricci-flat metric on this noncompact manifold is known [**?**] and it looks like the total space of T^*S^3, with the volume of the S^3 shrinking to zero as $\epsilon \to 0$.[5]

[5]This can be seen by writing the real and imaginary parts of Eq. (**??**) separately. For $\epsilon > 0$ real, the S^3 is the submanifold $\mathrm{Im} z_i = 0$.

A Math review

A first reference is GSW volume II chapters 15 and 16, especially §15.3, 15.4 and 15.5 on the Hodge decomposition and Dolbeault cohomology. After that, a classic math reference is Griffiths and Harris, Principles of Algebraic Geometry. This needs serious study, but §0.2 on complex manifolds and Dolbeault cohomology and §0.7 on Kähler metrics are not hard.

To go deeper into the subject one should study toric geometry and toric hypersurfaces. The classic physics paper is Witten's hep-th/9301042, and a standard math introduction is [**?**]. There are a lot of string theory PhD theses which work this out, e.g. Bouchard hep-th/0609123.

Let us first give the computation of the volume of $\mathbb{CP}^n$ and a hypersurface M defined as $f = 0$ in this space. These volumes are topological quantities, meaning that they are invariant under continuous deformations of the metric. In fact if we take the Fubini-Study metric

$$K = \frac{i}{2\pi} \log \sum_i^{n+1} |Z^i|^2 \tag{52}$$

or its restriction to M, normalized as we just did, the volumes will be integers.

The formula Eq. (35) expresses the Kähler metric as a total derivative of the Kähler potential. On reviewing Dolbeault cohomology, one sees a closely related formula

$$\omega = \partial\bar{\partial} K \tag{53}$$

for the $(1,1)$-form ω, the Kähler form. This can be used to construct the volume form,

$$\sqrt{\det g} = \det \omega = \frac{1}{n!}\omega^n. \tag{54}$$

(see p.31 of GH; it would be nice to check the normalizations in a symbolic algebra system). Furthermore, since $\partial\omega = 0$, we have

$$\omega^n = \partial(\bar{\partial} K \wedge \omega^{n-1}). \tag{55}$$

Now if K were a single valued function, Eq. (**??**) would imply that the integral of ω over any two-cycle would vanish. It is not, but this argument tells us that varying K by a function, $K \to K + f$, does not change the integral of ω. The same holds for the volume form by Eq. (**??**).

Next, let us see that for K given by Eq. (**??**),

$$\int_\Sigma \omega \in \mathbb{Z}, \tag{56}$$

for any two-cycle Σ. Furthermore there is a two-cycle for which this integral is 1. The integral only depends on the homology class of the cycle. In fact $\mathbb{CP}^n$ has a unique generator of H^2, as can be seen by embedding $\mathbb{CP}^1$ into it as $(z^1, z^2) \to (z^1, z^2, 0, \ldots, 0)$, the fact that

$\mathbb{CP}^1$ is topologically a two-sphere, and the $SU(n+1)$ symmetry of $\mathbb{CP}^n$ which takes all such embeddings into each other. Thus we can just compute this integral on $\mathbb{CP}^1$. You should do this exercise.

A simpler way to do this integral and understand why it is quantized, is to break up $\mathbb{CP}^n$ into patches, and express the integral as a sum of terms involving the differences of K between different patches. In mathematics this comes from a relation between ω and an associated gauge field and line bundle $\mathcal{L}$. In other words, we can postulate a $U(1)$ gauge field A, with curvature $F = dA$, such that

$$\omega = F. \tag{57}$$

Then, the integral of F over a basis of homology two-cycles, is by definition the first Chern class of the line bundle. One writes $c_1(\mathcal{L}) = 1$, and in the math literature this $\mathcal{L}$ is called $\mathcal{O}(1)$. You can read about this in the references, but the simplest physical construction of F is the following: we consider a Dirac monopole at the origin in $\mathbb{R}^3$, and identify $\mathbb{CP}^1 \cong S^2$ with the sphere of unit radius centered at the origin. Then, A is the monopole field, and the fact that the integral Eq. (**??**) is quantized is just the Dirac quantization condition (where we set the charge of the electron to 1).

This is related to our problem as follows. We can relate the coordinates Z^i to solutions of the Schrödinger equation in the monopole magnetic field on $\mathbb{CP}^1$,

$$\psi_a^{(i)} = Z_a^i \exp -\frac{1}{2}|Z_a|^2. \tag{58}$$

Here $a = 1, 2$ labels the patches with $Z^a = 1$, and ψ_a is the wave function in patch a. Z_a^i is the value of the coordinate Z^i in that patch. In fact these solutions are ground states in the lowest Landau level.

Similarly, if one put k monopoles at the origin, one would get a solution with magnetic field k times as large. This defines the line bundle $\mathcal{O}(k)$, with $c_1 = k$.

Now, the math fact which I won't explain in detail, is that the numbers c_1 are also related to the number of zeroes of a section of the bundle. In the case at hand they are equal, for example Z^i has a single zero in the patch $a \neq i$ and no zero in the patch $a = i$. Similarly the section $(Z^i)^k$ of $\mathcal{O}(k)$ has k zeroes. In fact one can turn all the problems of integrating wedge products of these ω's, into counting zeroes of simultaneous equations. This is called intersection theory in the mathematics and in the simple case at hand works as follows: we can associate $\mathcal{O}(1)$ with a dual two-cycle Σ, the generator of H^2. Then the powers ω^k are associated with dual $2k$-cycles which each generate their own H^{2k}. This implies that

$$\int_{\mathbb{CP}^n} \omega^n = 1 \tag{59}$$

and thus the volume of $\mathbb{CP}^n$ in the Kähler metric derived from Eq. (**??**) is $1/n!$.

One can check this directly by writing the coordinates as

$$Z^i = r_i e^{i\theta_i} \tag{60}$$

upon which the condition $1 = \sum |Z^i|^2$ becomes $1 = \sum r_i^2$. Thus $\mathbb{CP}^n$ is closely related to the sphere S^{2n+1} – it is obtained by quotienting by the overall phase, $\theta_i \to \theta_i + \epsilon$. This is a circle with constant volume, so the two volumes are just related by an overall 2π (I think).

Next, let us consider M, the hypersurface $f = 0$ in $\mathbb{CP}^{n+1}$, where f is a degree k homogeneous polynomial. The volume form on M is $\omega^n/n!$ where ω is the restriction of ω as above to this surface. To do this restriction in practice, one must embed the n-dimensional cotangent space T^*M in the $n+1$-dimensional cotangent space $T^*\mathbb{CP}^{n+1}$. Now the homogeneous coordinates Z^i are also sections of a line bundle $\mathcal{L}$, defined by restricting $\mathcal{O}(1)$ to M. However we cannot use them all as coordinates in a patch as they are redundant. In terms of the cotangent bundle we can express this redundancy as

$$T^*\mathbb{CP}^n \cong T^*M \oplus \mathcal{N}^*_{\mathbb{CP}^{n+1}} M, \tag{61}$$

where $\mathcal{N}^*_{\mathbb{CP}^{n+1}} M$ is the conormal bundle, in other words the dual to the normal bundle in which vectors normal to M live. While the normal vector to M depends on the metric, one can choose a section of the conormal bundle which does not. It is a one-form which vanishes if we contract it with a tangent vector on M. Thus it is just the one-form df, since the condition that a vector v is tangent to M is $v^i \partial_i f = 0$.

Thus, to restrict ω and the metric to M, we want to choose a local set of coordinates in which one of the coordinates Z^i is replaced by f, and change basis from $(dZ^1, dZ^2, \ldots, dZ^{n+1})$ to $(dZ^1, dZ^2, \ldots, df, \ldots, dZ^{n+1})$, and omit df. To restrict the volume form we just need the Jacobian of the resulting matrix (with df omitted). Writing the new coordinates as $\hat{Z}^i$, one can see that this matrix is (for $n = 2$ and replacing $\hat{Z}^3$),

$$\begin{pmatrix} d\hat{Z}^1 & d\hat{Z}^2 & df \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \frac{\partial f}{\partial Z^1} & \frac{\partial f}{\partial Z^2} & \frac{\partial f}{\partial Z^3} \end{pmatrix} \begin{pmatrix} dZ^1 & dZ^2 & dZ^3 \end{pmatrix}. \tag{62}$$

To reexpress ω in the new frame, one must invert this matrix to get dZ^i in terms of the new frame. One can then omit the ∂f and $\bar{\partial}\bar{f}$ components.

As we discussed, we want to choose subpatches of each patch on $\mathbb{CP}^{n+1}$ on which the various components $|\partial f/\partial Z^i|$ are maximized, and then omit this $\hat{Z}^i$ in that patch. Thus this relation will take a slightly different form in each patch.

One can even carry out the argument in terms of intersection theory to get the volume of M in the restriction of the Fubini-Study metric. Now, the Kähler form ω from Eq. (**??**) is the curvature of $\mathcal{O}(1)$, which is just obtained by restriction. To do an integral over M in intersection theory, one can wedge the form with the curvature of the normal bundle. This is degree f as is explained in the "adjunction formulas" in §1.1 of GH, pages 145–147 in my copy. Thus we have the volume

$$\frac{1}{n!}\int_M \omega^n = \frac{\text{degree } f}{n!} \tag{63}$$

$$= \frac{n+2}{n!} \qquad \text{if degree } f = n+2 \tag{64}$$

as is the case for a Ricci flat manifold. So the volumes for T^2, K3 and the quintic should be 3, 2 and 5/6 respectively.

B Symmetries

While the general quintic hypersurface has no symmetries, we will restrict attention to the hypersurfaces

$$0 = f(z) = \sum_{i=1}^{5} z_i^5 + \psi z_1 z_2 z_3 z_4 z_5 \tag{65}$$

which have $S_5 \ltimes \mathbb{Z}_5^3$ symmetry (the special case $\psi = 0$ has an additional $\mathbb{Z}_5$). Let us call this symmetry group Γ and denote a general element as (α, σ) where

$$z_j \to \exp \frac{2\pi i \alpha_j}{5} \sigma_{j,k} z_k \tag{66}$$

where $\sigma_{j,k}$ is a permutation, then the parameters must satisfy $\sum \alpha_j = 0(5)$ to preserve f. In addition an overall rotation $z_j \to e^{2\pi i/5} z_j$ can be compensated by λ. These conditions are satisfied by the following "charge vectors"

$$\begin{aligned} \alpha^{(1)} &= (1, -1, 0, 0, 0) && (67)\\ \alpha^{(2)} &= (0, 1, -1, 0, 0) && (68)\\ \alpha^{(3)} &= (0, 0, 1, -1, 0) && (69)\\ & && (70) \end{aligned}$$

which can be grouped into a vector $\vec{\alpha} = (\alpha^{(1)}, \alpha^{(2)}, \alpha^{(3)})$.

These symmetries must preserve the Kähler potential up to an overall constant shift. This will force many of its parameters to be zero and relate other parameters.

For the general expression Eq. (1), this requires every nonzero matrix element $h_{I,\bar{J}}$ to transform by the same phase. Thus a nonzero element must satisfy

$$0 = \sum_{a=1}^{k} \vec{\alpha}(I_a) - \vec{\alpha}(J_a) \tag{71}$$

Furthermore the S_k action relates many of the matrix elements.

Let us consider some low order cases. For $k = 1$ the $\mathbb{Z}_5^3$ symmetry sets all the off-diagonal terms to zero, while the S_5 symmetry forces h to be proportional to the identity. This amounts to the statement that

$$X^{1,1} = \sum_{i=1}^{5} |z_i|^2 \tag{72}$$

is the only polynomial of degree $(1, 1)$ invariant under Γ.

For $k = 2$ the identity h is of course allowed, but there is an another allowed diagonal h. This is because there are two invariants of degree $(2, 2)$, $(X^{1,1})^2$ and

$$X^{2,2} = \sum_{i=1}^{5} |z_i|^4. \tag{73}$$

Clearly we can make a new invariant $X^{l,l}$ at each order l, and get invariants

$$\prod (X^{1,1})^{n_1} (X^{2,2})^{n_2} \dots \tag{74}$$

with $\sum_l l n_l = k$. This is however not a complete set. This is clear for $k \geq \deg f$ as f contains new invariants, for example $X^{5,0} \equiv z_1 z_2 z_3 z_4 z_5$ and its complex conjugate $X^{0,5}$.

In practice we can use an overcomplete set of parameters, so what we want is a general algorithm that produces a not too redundant set along with the linear transformation T mapping them into the matrix h. This could be done by first imposing the $\mathbb{Z}_5^3$ symmetry to set most of the matrix elements to zero, and then applying the following algorithm.

1. Create an empty set L which will contain an independent set of matrix elements $h_{I,\bar{J}}$.
2. Iterate through the matrix elements $h_{I,\bar{J}}$ which respect $\mathbb{Z}_5^3$ symmetry.
3. Considering all $\sigma \in S_5$, if no $(\sigma I, \sigma \bar{J})$ appears in the set L, add $(I, \bar{J})$ to L.
4. If we add $(I, \bar{J})$, the corresponding matrix elements of T are

$$T_{(\sigma I, \sigma \bar{J}),(I,\bar{J})} = 1. \tag{75}$$

In general this might not find all the relations, but it should be good enough.

For the neural networks, we presumably can only combine quantities with the same symmetry properties in each layer. This is quite a drastic restriction, for example in the first layer with inputs z_i, the weight matrix would have to be the identity matrix. As we go deeper there will be allowed combinations, but maybe not enough - we might want to generalize $z \to z^2$ to products of sections with different degrees.

Another approach this suggests is to use the invariants $X^{l,l}$, $X^{5,0}$ and $X^{0,5}$ as our elementary variables. A straightforward way to use these is to decompose (k, k) in all possible ways, *e.g.*

$$Z^{2,2} = w_{1,1}(X^{1,1})^2 + w_2 X^{2,2} \tag{76}$$

$$Z^{3,3} = w_{1,1,1}(X^{1,1})^3 + w_{1,2} X^{1,1} X^{2,2} + w_3 X^{3,3} \tag{77}$$

$$Z^{4,4} = w_{1,1,1,1}(X^{1,1})^4 + w_{1,3} X^{1,1} X^{3,3} + w_{2,2}(X^{2,2})^2 + w_{1,1,2}(X^{1,1})^2 X^{2,2} + w_4 X^{4,4} \tag{78}$$

and so on. But instead of taking $K = Z^{k,k}$ defined this way, we could also use some of the Z's as intermediate layers. For example, we could exclude all the terms of more than

quadratic order in the X's, instead using the Z's:

$$Z^{3,3} = w_{1,2}X^{1,1}Z^{2,2} + w_3X^{3,3} \quad (79)$$

$$Z^{4,4} = w_{1,3}X^{1,1}Z^{3,3} + w_{2,2}(Z^{2,2})^2 + w_4X^{4,4} \quad (80)$$

$$Z^{5,5} = w_{1,4}X^{1,1}Z^{4,4} + w_{2,3}Z^{2,2}Z^{3,3} + w_5X^{5,5} + \tilde{w}X^{5,0}X^{0,5} \quad (81)$$

so that the result would depend on nonlinear combinations of the weights w. With this approach we could get a very high degree function with $O(k^2)$ weights.

References

[1] "Calabi-Yau metrics for string compactification," Nucl. Phys. **B898** (2015) 667-674, arXiv:1503.02899.

[2] Hori, K., Thomas, R., Katz, S., Vafa, C., Pandharipande, R., Klemm, A., Vakil, R. and Zaslow, E., 2003. *Mirror symmetry* (Vol. 1). AMS

[3] Aspinwall, P., Bridgeland, T., Craw, A., Douglas, M. R., Gross, M., Kapustin, A., Moore, G. W., Segal, G., Szendroi, B. and Wilson, P. M. H., 2009. *Dirichlet branes and mirror symmetry* (Vol. 4). AMS

[4] D. Gaiotto, G. W. Moore and A. Neitzke, "Four-dimensional wall-crossing via three-dimensional field theory," Commun. Math. Phys. **299**, 163 (2010) [arXiv:0807.4723 [hep-th]].

[5] S. Kachru, A. Tripathy and M. Zimet, "K3 metrics," [arXiv:2006.02435 [hep-th]].

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

[7] S. K. Donaldson, "Some numerical results in complex differential geometry," [math.DG/0512625].

[8] M. R. Douglas, R. L. Karp, S. Lukic and R. Reinbacher, "Numerical solution to the hermitian Yang-Mills equation on the Fermat quintic," JHEP **0712**, 083 (2007) [hep-th/0606261].

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

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

[11] V. Braun, T. Brelidze, M. R. Douglas and B. A. Ovrut, "Calabi-Yau Metrics for Quotients and Complete Intersections," JHEP **0805**, 080 (2008) [arXiv:0712.3563 [hep-th]].

[12] R. S. Bunch and S. K. Donaldson, "Numerical approximations to extremal metrics on toric surfaces," *Handbook of geometric analysis* No. 1, 1-28, Adv. Lect. Math. (ALM), 7, Int. Press, Somerville, MA, 2008, [arXiv:0803.0987 [math.DG]].

[13] R. Seyyedali, "Numerical Algorithms for Finding Balanced Metrics on Vector Bundles," Asian J. Math. 13 (2009), no. 3, 311-321, [arXiv:0804.4005 [math.DG]].

[14] V. Braun, T. Brelidze, M. R. Douglas and B. A. Ovrut, "Eigenvalues and Eigenfunctions of the Scalar Laplace Operator on Calabi-Yau Manifolds," JHEP **0807**, 120 (2008) [arXiv:0805.3689 [hep-th]].

[15] L. B. Anderson, J. Gray, D. Grayson, Y. H. He and A. Lukas, "Yukawa Couplings in Heterotic Compactification," Commun. Math. Phys. **297**, 95 (2010) [arXiv:0904.2186 [hep-th]].

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

[17] 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 **1006**, 107 (2010) [arXiv:1004.4399 [hep-th]].

[18] L. B. Anderson, V. Braun and B. A. Ovrut, "Numerical Hermitian Yang-Mills Connections and Kahler Cone Substructure," JHEP **1201**, 014 (2012) [arXiv:1103.3041 [hep-th]].

[19] W. Cui and J. Gray, JHEP **05**, 044 (2020) doi:10.1007/JHEP05(2020)044 [arXiv:1912.11068 [hep-th]].

[20] Hoyer, S., Sohl-Dickstein, J. and Greydanus, S., 2019. Neural reparameterization improves structural optimization. arXiv:1909.04240.

[21] A. Ashmore, Y. H. He and B. A. Ovrut, [arXiv:1910.08605 [hep-th]].

[22] L. B. Anderson, J. Gray, S. Krippendorf, N. Raghuram and F. Ruehle, "Machine learning SU(3) structure," To appear.

[23] Chrysos, G.G., Moschoglou, S., Bouritsas, G., Deng, J., Panagakis, Y. and Zafeiriou, S., 2020. Π− nets: Deep Polynomial Neural Networks. arXiv:2003.03828.

[24] Mannelli, S.S., Vanden-Eijnden, E. and Zdeborová, L., 2020. Optimization and Generalization of Shallow Neural Networks with Quadratic Activation Functions. arXiv:2006.15459.

[25] Douglas, M.R., Holomorphic feedforward networks, submitted to the Pure and Applied Mathematics Quarterly special issue in honor of Professor Bernie Shiffman.

[26] Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167.

[27] Candelas, P. and De la Ossa, X.C., Comments on conifolds 1990 Nucl. Phys. B, 342, p.246.

[28] Griffiths, P. and Harris, J., 2014. *Principles of algebraic geometry*. John Wiley & Sons.

[29] Huybrechts, D., 2006. *Complex geometry: an introduction*. Springer Science & Business Media.

[30] Bishop, C.M., 2006. *Pattern recognition and machine learning*. Springer.

[31] M. R. Douglas and S. Klevtsov, "Bergman Kernel from Path Integral," Commun. Math. Phys. **293**, 205 (2010) [arXiv:0808.2451 [hep-th]].

[32] Cybenko, G., 1989. Approximation by superpositions of a sigmoidal function. Mathematics of control, signals and systems, 2(4), pp.303-314.

[33] `https://github.com/tensorflow/tensorflow/blob/r2.0/tensorflow/python/ops/gradients_i`

[34] `http://home.ustc.edu.cn/ hanzr/pdf/Introduction%20to%20Toric%20Varieties-Fulton.pdf`