Numerical Calabi-Yau Metrics from Holomorphic Networks

Yidi Qi

Stony Brook University

Seminar Seires on String Phenomenology

February 16, 2021

Based on arxiv:2012.04797 with Michael R. Douglas and Subramanian Lakshminarasimhan

Introduction

Calabi-Yau manifolds were proven to admit Ricci flat metrics and they play an important role in string compactification.

However, it is generally believed that no closed form expression exists for compact CY manifolds.

Introduction

Calabi-Yau manifolds were proven to admit Ricci flat metrics and they play an important role in string compactification.

However, it is generally believed that no closed form expression exists for compact CY manifolds.

Works on numerical approximations:

Headrick and Wiseman [hep-th/0506129]
Donaldson [math.DG/0512625]
Douglas, Karp, Lukic, Reinbacher (DKLR) [hep-th/0606261, hep-th/0612075]
Braun, Ovrut *et al* [0712.3563, 0805.3689]
Anderson *et al* [0904.2186, 1004.4399, 1103.3041]
Headrick and Nassar [0908.2635]
Cui and Gray [1912.11068]

The Embedding Method

A simple example of Calabi-Yau manifolds is the quintics hypersurface in $\mathbb{CP}^4$:

$$0 = f(Z^1, Z^2, Z^3, Z^4, Z^5) = \sum_{i=1}^{5} (Z^i)^5 + \psi Z^1 Z^2 Z^3 Z^4 Z^5$$

One can get a parameterized family of metrics by pulling back the Fubini-Study metrics, which is defined by the Kähler potential:

$$K = \log \sum_{i,\bar{j}=1}^{d+1} h_{i\bar{j}} Z^i \bar{Z}^{\bar{j}},$$

where h is a positive definite hermitian matrix.

The Embedding Method

A simple example of Calabi-Yau manifolds is the quintics hypersurface in $\mathbb{CP}^4$:

$$0 = f(Z^1, Z^2, Z^3, Z^4, Z^5) = \sum_{i=1}^{5} (Z^i)^5 + \psi Z^1 Z^2 Z^3 Z^4 Z^5$$

One can get a parameterized family of metrics by pulling back the Fubini-Study metrics, which is defined by the Kähler potential:

$$K = \log \sum_{i,\bar{j}=1}^{d+1} h_{i\bar{j}} Z^i \bar{Z}^{\bar{j}},$$

where h is a positive definite hermitian matrix.

This can be generalized by replacing the Zs with a basis of sections s^I of a line bundle $\mathcal{L}^k$, which are the homogeneous polynomials of degree k in Z^i:

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

Geometric quantities

A CY manifold admits two fundamental differential forms:

- The Kähler form:

$$\omega = \partial_i \partial_{\bar{j}} K \, dZ^i \wedge d\bar{Z}^{\bar{j}},$$

  which can be used to write the volume form:

$$d\mu_g \equiv \det \omega_g = \det_{i,\bar{j}} \partial_i \partial_{\bar{j}} K$$

Geometric quantities

A CY manifold admits two fundamental differential forms:

- The Kähler form:

$$\omega = \partial_i \partial_{\bar{j}} K \, dZ^i \wedge d\bar{Z}^{\bar{j}},$$

  which can be used to write the volume form:

$$d\mu_g \equiv \det \omega_g = \det_{i,\bar{j}} \partial_i \partial_{\bar{j}} K$$

- The non-vanishing holomorphic n-form:

$$\Omega = \Omega_{i_1 \ldots i_n} dZ^1 \wedge \ldots \wedge dZ^n$$

  And the associated volume form:

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

Geometric quantities

A CY manifold admits two fundamental differential forms:

- The Kähler form:

$$\omega = \partial_i \partial_{\bar{j}} K \, dZ^i \wedge d\bar{Z}^{\bar{j}},$$

  which can be used to write the volume form:

$$d\mu_g \equiv \det \omega_g = \det_{i,\bar{j}} \partial_i \partial_{\bar{j}} K$$

- The non-vanishing holomorphic n-form:

$$\Omega = \Omega_{i_1 \ldots i_n} dZ^1 \wedge \ldots \wedge dZ^n$$

  And the associated volume form:

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

For a Ricci flat metric, the ratio $\eta = \frac{d\mu_g}{d\mu_\Omega}$ can be set to 1 by choice of normalization.

Geometric quantities

With the embedding method

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

one can minimize an energy function: [Headrick, Nassar, arxiv:0908.2635]

$$E = \int_M d\mu_{ref} \left(\eta - 1\right)^2$$

It can be shown that the approximation error decreases exponentially in the order k of the polynomials.

Geometric quantities

With the embedding method

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

one can minimize an energy function: [Headrick, Nassar, arxiv:0908.2635]

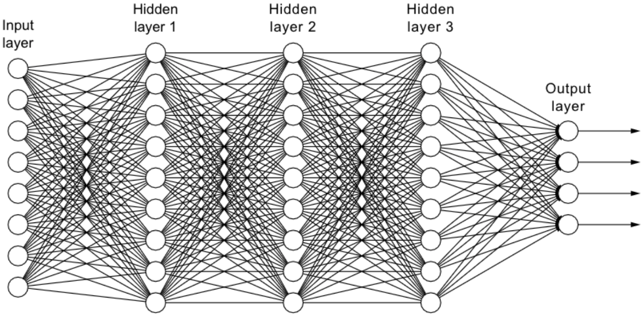
$$E = \int_M d\mu_{ref} \, (\eta - 1)^2$$

It can be shown that the approximation error decreases exponentially in the order k of the polynomials.

However, the number of parameters in h also increases in $\mathcal{O}(k^6)$, which demands a lot of computational resource when k goes higher.

Most of the previous work only consider the symmetric hypersurfaces to reduce the number of parameters.

Feed-forward Networks



$$F_w = W^{(d)} \circ \theta \circ W^{(d-1)} \circ \ldots \circ \theta \circ W^{(1)} \circ \theta \circ W^{(0)}$$

- $W^{(i)}$: Weight matrix
- θ: Non-linear activation function

Supervised Learning

Feed-forward network:

- A parameterized function: $F_w : \mathcal{X} \to \mathcal{Y}$
- The universal approximation theorem shows that feed-forward networks can approximate arbitrary functions [Cybenko, 1989; Hornik, 1991]

Supervised Learning

Feed-forward network:

- A parameterized function: $F_w : \mathcal{X} \to \mathcal{Y}$
- The universal approximation theorem shows that feed-forward networks can approximate arbitrary functions [Cybenko, 1989; Hornik, 1991]

Loss function:

- Mean squared error (MSE):

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

- Mean absolute percentage error (MAPE):

$$\mathcal{E} = \mathbb{E}_{\mathcal{P}} \left[\frac{|f_w(x) - y|}{y} \right]$$

Supervised Learning

Feed-forward network:

- A parameterized function: $F_w : \mathcal{X} \to \mathcal{Y}$
- The universal approximation theorem shows that feed-forward networks can approximate arbitrary functions [Cybenko, 1989; Hornik, 1991]

Loss function:

- Mean squared error (MSE):

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

- Mean absolute percentage error (MAPE):

$$\mathcal{E} = \mathbb{E}_{\mathcal{P}} \left[\frac{|f_w(x) - y|}{y} \right]$$

Training:

- One simple algorithm to minimize the loss is the gradient descent:

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

Supervised Learning

Compare the supervised learning with the CY metrics problem:

Dataset:

- Input x: the points on the hypersurface
- Output y: the holomorphic volume form $d\mu_\Omega \equiv \mathcal{N}_\Omega \Omega \wedge \bar{\Omega}$

Parameterized function F_w: mapping the points to the metric volume form $d\mu_g$

Loss function: The Monte Carlo integration of

$$\mathcal{E} = \int_M d\mu_{ref} \, (\eta - 1)^2$$

Training algorithm

Holomorphic embeddings

The first way to construct F_w is by what we called **holomorphic networks**:

We replace the homogeneous polynomials in the Kähler potential

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

by a complex feed-forward network which takes the coordinates as the input:

$$F_w = W^{(d)} \circ \theta \circ W^{(d-1)} \circ \ldots \circ \theta \circ W^{(1)} \circ \theta \circ W^{(0)}$$

We choose the activation functions to be:

$$\theta(z) = z^2$$

so the output of F_w is a vector of sections of $\mathcal{L}^k$ with $k = 2^d$, and the Kähler potential can be represented by

$$K_w = \log |F_w(Z)|^2$$

Bihomogeneous embeddings

The performance for the holomorphic networks is not so good in our experiments.

Alternatively, we can also construct F_w by what we called **bihomogeneous networks**, which takes the real and imaginary parts of the $Z^i \bar{Z}^{\bar{j}}$ as inputs.

So F_w will be a network with real weights and a single output

$$F_w = W^{(d)} \circ \theta \circ W^{(d-1)} \circ \ldots \circ \theta \circ W^{(1)} \circ \theta \circ W^{(0)}$$

and the Kähler potential is

$$K_w = \log F_w(\mathrm{Re} Z^i \bar{Z}^{\bar{j}}, \mathrm{Im} Z^i \bar{Z}^{\bar{j}})$$

Training Algorithm

The activation function $\theta(z) = z^2$ is actually not a usual choice in a typical machine learning problem:

- The weights in the front layers have higher degrees than the rest
- The gradient is easy to blow up or vanish during the optimization

We used a two-phase training method:

- Train with the Adam algorithm [Kingma, Ba, 2015] which computes an individual adaptive learning rate for each parameter
- Then train with L-BFGS near the minima

Supervised Learning

Dataset:

- Input x: the points on the hypersurface $\rightarrow Z^i \bar{Z}^{\bar{j}}$
- Output y: the holomorphic volume form $d\mu_\Omega \equiv \mathcal{N}_\Omega \Omega \wedge \bar{\Omega}$

Parameterized function: $\det \partial\bar{\partial} \log F_w[x_i]$

Loss function: The Monte Carlo integration of

$$\mathcal{E} = \int_M d\mu_{ref} \, (\eta - 1)^2$$

Training algorithm: Adam and L-BFGS

Why machine learning?

- Hardware (GPU) and software (Tensorflow/Keras) optimized for tensor operations
- Automatic differentiation techniques to speed up the training process

Implementation

Our code on Github: `http://github.com/yidiq7/MLGeometry`

```
class twolayers(tf.keras.Model):

    def __init__(self, n_units):
        super(twolayers, self).__init__()
        self.bihomogeneous = bnn.Bihomogeneous()
        self.layer1 = bnn.Dense(25, n_units[0], activation=tf.square)
        self.layer2 = bnn.Dense(n_units[0], n_units[1], activation=tf.square)
        self.layer3 = bnn.Dense(n_units[1], 1)

    def call(self, inputs):
        x = self.bihomogeneous(inputs)
        x = self.layer1(x)
        x = self.layer2(x)
        x = self.layer3(x)
        x = tf.math.log(x)
        return x
```

Experiments and Results

We scanned through CY manifolds with different symmetries:

- The Dwork quintics $f = f_1$ below with $\phi = 0$
- A two parameter family with less symmetry:

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

- Another two parameter family with no symmetry:

$$\begin{aligned} f_2 = f_1|_{\phi=0} + \alpha\Big(& z_2 z_0^4 + z_0 z_4 z_1^3 + z_0 z_2 z_3 z_4^2 + z_3^2 z_1^3 + z_4 z_1^2 z_2^2 + z_0 z_1 z_2 z_3^2 + \\ & z_2 z_4 z_3^3 + z_0 z_1^4 + z_0 z_4^2 z_2^2 + z_4^3 z_1^2 + z_0 z_2 z_3^3 + z_3 z_4 z_0^3 + z_1^3 z_4^2 + \\ & z_0 z_2 z_4 z_1^2 + z_1^2 z_3^3 + z_1 z_4^4 + z_1 z_2 z_0^3 + z_2^2 z_4^3 + z_4 z_2^4 + z_1 z_3^4 \Big) \end{aligned}$$

Experiments and Results

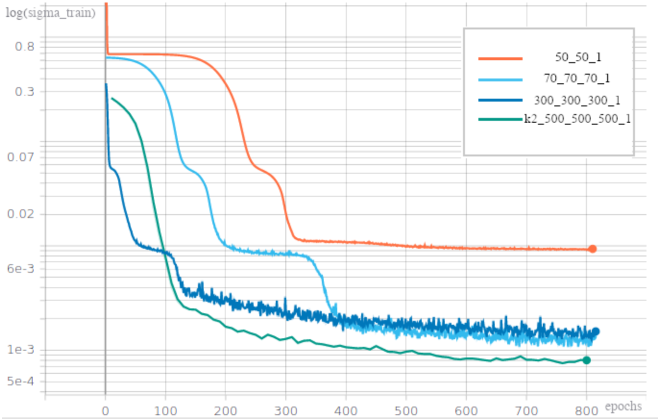


Figure: The training curves for the Dwork quintic with $\psi = 0.5$, trained with Adam optimizer and MAPE loss

Experiments and Results

We studied the dependence of the accuracy on different parameters:

For the geometry of CY:

- The shortest length scale $\sim$ distance in moduli space to a singualr CY $f_s = \nabla f_s = 0$:

$$\sin\theta \propto d \equiv \min_{Z_0 \in M} \frac{|\partial_i f(Z_0)|_H}{||f||_H |Z_0|^{n-1}}.$$

- The complexity of the CY

For the network:

- The depth d of the network, with $k = 2^d$
- The total number of parameters

Experiments and Results

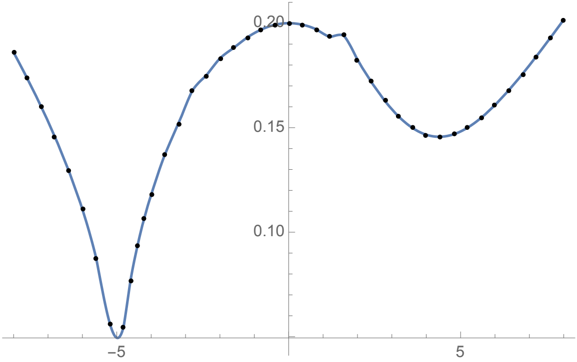


Figure: Distance to a singular CY for the Dwork quintics f_0, X axis is ψ, Y axis is the distance.

Experiments and Results

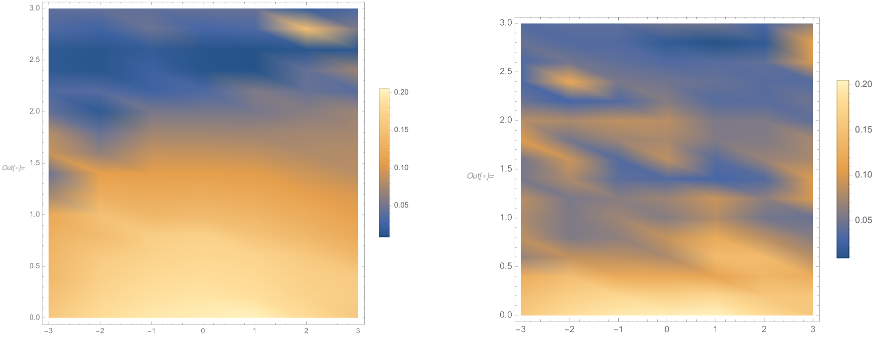
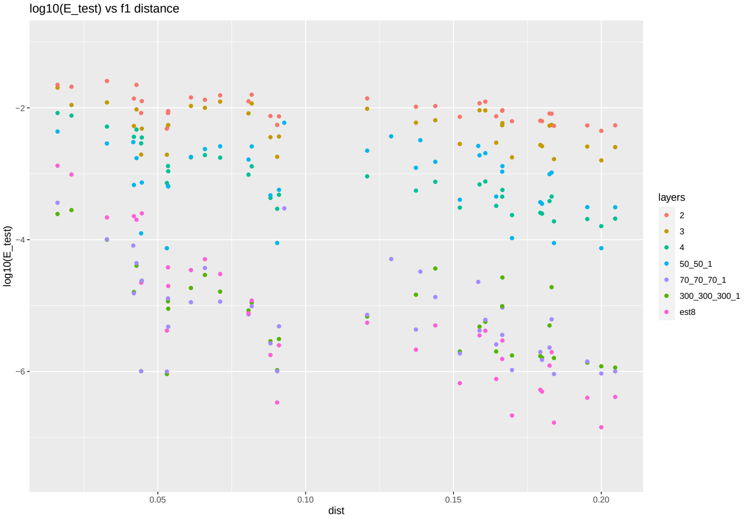
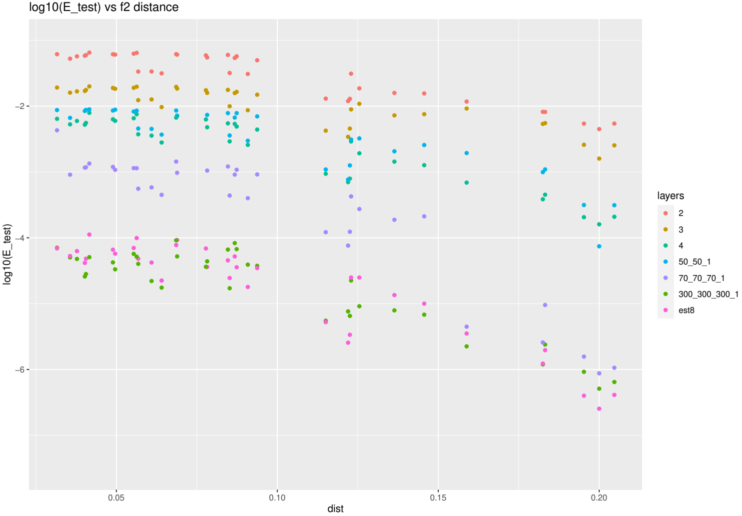


Figure: Distance to singular CY as function of ψ, ϕ in f_1 and ψ, α in f_2

Experiments and Results



Experiments and Results



Summary

- We developed a Tensorflow/Keras package to approximate numerical CY metrics with high accuracy
- Other interesting topics in geometry: SYZ special Lagrangian torus fibrations
- Questions in ML: hyperparameters tuning, over-parameterization, etc
- Other recent work using ML:

  Ashmore, He and Ovrut [1910.08605]
  Ashmore [2011.13929]
  Anderson, Gerdes, Gray, Krippendorf,Raghuram and Ruehle [2012.04656]
  Jejjala, Pena, Mishra [2012.15821]