Characterizing 4-string contact interaction using machine learning

Atakan Hilmi Fırat

Massachusetts Institute of Technology

SFT 2022, Prague CZ
September 13, 2022

Based on work in progress with Harold Erbin

Outline

1. Motivation(s)
2. Strebel quadratic differential
   - Review and general remarks
   - 4-punctured sphere case study
3. Machine learning
   - Neural networks 101
   - Neural network for accessory parameter when $n = 4$
4. Geometry of 4-string contact interaction
   - Local coordinates and mapping radii
   - Critical graph and the lengths of non-contractible curves
   - Indicator function
5. Discussion
   - Advantages
   - Future directions

Sociological motivation

- Even though closed SFT has been constructed, it is notoriously hard to perform any type of calculation. This is primarily due to
  1. Its non-polynomial structure

$$S[\Psi] = \frac{1}{2}\langle \Psi, Q_B \Psi \rangle + \sum_{2g-2+2n>0} \frac{\kappa^{2g-2+n}}{n!}\{\Psi^n\}_{g,n} .$$

  2. The lack of *user-friendly* description for the local coordinates and the region of integration $\mathcal{V}_{g,n}$ for higher string interactions

$$\{\Psi^n\}_{g,n} = \int_{\mathcal{V}_{g,n}} \langle \Sigma_{g,n} | \,\text{ghost insertions}\, | \Psi^n \rangle \, .$$

- Even though (1) is always with us to stay, it is imperative to get a better handle on (2). Here we will propose a *machine learning (ML)* approach to (2) in the case of classical ($g = 0, n \geqslant 3$) vertices.

Mathematics motivation

- The *canonical* approach for defining classical string vertices over the years has been through Strebel quadratic differential. That is, placing punctures at $z = \xi_1, \cdots, \xi_n$, the quadratic differential

$$\varphi = \phi(z)dz^2 = \sum_{i=1}^{n} \left[\frac{-1}{(z-\xi_i)^2} + \frac{c_i}{z-\xi_i} \right] dz^2 \,,$$

  with appropriately chosen *accessory parameters* $c_1, \cdots, c_n$ are all one needs to characterize the *n*-string contact interaction.

- Unfortunately determining accessory parameters for arbitrary n as a function of moduli is *extremely* difficult problem! However, we claim it is possible to train a *neural network (NN)* that *learns* the behavior of these functions — we test the algorithm in the case of $n = 4$.

Physics motivation

- Once a description of contact interactions are obtained to some order, we can truncate the theory and begin performing calculations.

- Primary target is establishing the tachyon vacuum for closed bosonic SFT, or lack thereof, as previous works (by Belopolsky, Moeller, Yang, Zwiebach) were somewhat inconclusive. **Maybe having more orders of interaction would help?**

- For now, we benchmark our pipeline by calculating 4-tachyon contact term. With our methods, we find the coefficient of this term to be

$$v_4 \approx 72.4 \quad \text{where} \quad V(t, \cdots) = -t^2 - \sum_{n=3}^{\infty} \frac{v_n}{n!} t^n + \cdots .$$

This result is consistent with the literature.

Table of Contents

Some review and nomenclature I

- We are interested in quadratic differentials with double poles at the punctures $z = \xi_1, \cdots, \xi_n$ with residue equal to -1 and regular at $z = \infty$. The most general such differential is

$$\varphi = \phi(z) dz^2 = \sum_{i=1}^{n} \left[\frac{-1}{(z - \xi_i)^2} + \frac{c_i}{z - \xi_i} \right] dz^2 \,,$$

with

$$\sum_{i=1}^{n} c_i = 0, \quad \sum_{i=1}^{n} (-1 + c_i z_i) = 0, \quad \sum_{i=1}^{n} (-2z_i + c_i z_i^2) = 0.$$

- Solving these conditions for $n = 3$ eventually results Witten's vertex.

Some review and nomenclature II

- We define *horizontal trajectory* as a trajectory for which $\varphi > 0$. A *critical trajectory* is a horizontal trajectory that begins and ends at $\varphi = 0$. The union of critical trajectories is called *critical graph*.

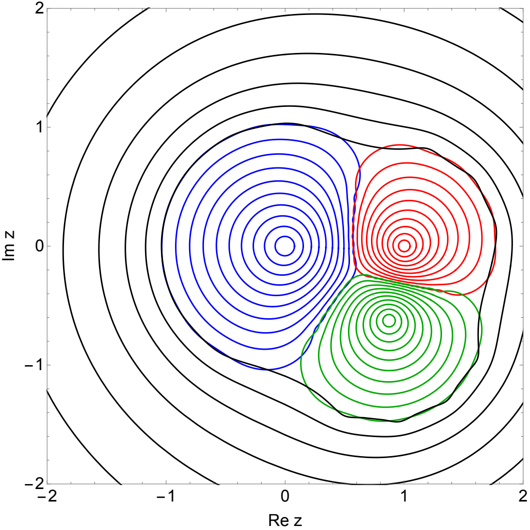
Strebel Differential

Strebel differential is the quadratic differential whose critical graph is

1. **Measure zero –** that is, it defines a polyhedron on Riemann sphere whose vertices are at $\varphi = 0$, edges are critical trajectories, and faces contains punctures,
2. **Connected –** that is, there is no string propagator in the geometry.

- It can be shown such differential is unique on a given surface and the metric $ds = \sqrt{|\phi(z)|}|dz|$ defines the metric of minimal (reduced) area – hence consistent SFT. The lengths of non-contractible curves can be smaller than 2π though.

Strebel differential when $\xi = 0.8734356 - 0.624202i$



Analytical condition for Strebel differential

- Define the *complex length* between zeros z_i, z_j as

$$\ell(z_i, z_j) \equiv \int_{z_i}^{z_j} \sqrt{\phi(z)}dz\,.$$

Path of integration here is chosen such that branch cuts are avoided.

- Notice in the case of Strebel differential $\ell(z_i, z_j)$ is real, as there exists a path for which integrand is real (critical trajectory). So Strebel differential satisfies

$$0 = \mathrm{Im}\,\ell(z_i, z_j) = \mathrm{Im}\int_{z_i}^{z_j} \sqrt{\phi(z)}dz\,.$$

between all pairs of zeros of φ. This condition is also sufficient.

Pre-loss function

- Now define the *pre-loss function* $\mathcal{L}$ as follows

$$\boxed{\mathcal{L}(c_1, \cdots, c_{n-3}; \xi_1, \cdots, \xi_n) \equiv \binom{2n-4}{2}^{-1} \sum_{i \geqslant j} (\mathrm{Im}\, \ell(z_i, z_j))^2 \,.}$$

- Such function is non-negative, $\mathcal{L} \geqslant 0$, and it attains its unique global minimum as a function of c_i's at fixed ξ_i when differential is Strebel.
- Then Strebel differential can be found by optimizing pre-loss function to this global minimum. One great advantage is whatever the optimization algorithm is, it is going to be independent of number of punctures and topology of the critical graph!

Strebel differential on 4-punctured sphere

- Specialize to 4-punctured sphere and place punctures at $0, 1, \xi, \infty$. It can be shown that the quadratic differential takes the form

$$\phi(z)dz^2 = \frac{-z^4 + az^3 + (2\xi - (1+\xi)a)z^2 + a\xi z - \xi^2}{z^2(z-1)^2(z-\xi)^2} dz^2 \,,$$

  with $a = a(\xi, \xi^*)$ being the accessory parameter.

- There are certain transformation properties accessory parameter have:

$$\begin{aligned} \xi \to \xi^* &\implies a \to a^* \,, && \text{(Involution)} \\ \xi \to 1 - \xi &\implies a \to -a + 4 \,, && \text{(Shift)} \\ \xi \to \frac{1}{\xi} &\implies a \to \frac{a}{\xi} \,. && \text{(Inversion)} \end{aligned}$$

Special points

- For certain symmetric points, we can fix the accessory parameter:

$$\left(\xi = \frac{1}{2},\, a = 2\right), \quad \left(\xi = \frac{1}{2} \pm i\frac{\sqrt{3}}{2},\, a = 2 \pm \frac{2}{\sqrt{3}}\right), \cdots$$

- In fact, when $\xi \in \mathbb{R}$ one can show

$$a(\xi = \xi^*) = \begin{cases} 0 & \xi \leqslant 0 \\ 4\xi & 0 \leqslant \xi \leqslant 1 \\ 4 & 1 \leqslant \xi \end{cases},$$

by noticing critical graph degenerates there.

Table of Contents

Machine learning 101

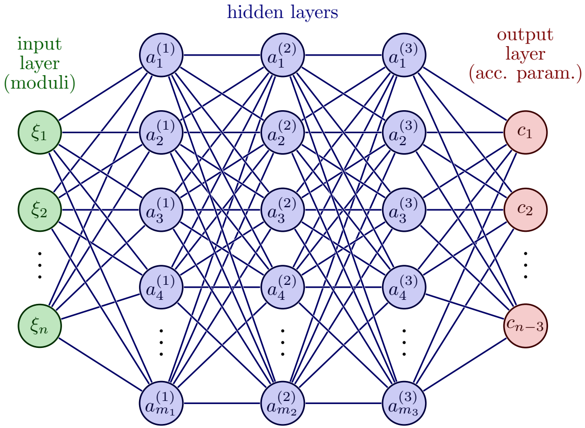
Machine learning (Mitchell)

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E.

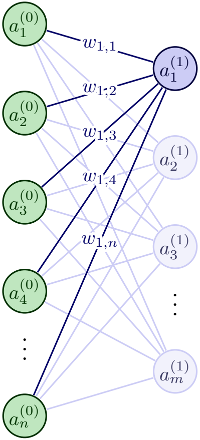
- T: Finding $c_1, \cdots, c_{n-3}$ as a function of moduli $\xi_1, \cdots \xi_n$,
- P: How close one gets $\mathcal{L} \approx 0$ everywhere on the moduli space,
- E: Finite number of points on the moduli space (i.e. *training set*) and the average of their pre-loss function $\mathcal{L}$.

Neural networks I

- Neural networks (NN) consist of many connected *layers* and each layers contain number of *nodes*.



- At each node following operation is performed:



$$= \sigma\left(w_{1,0}a_0^{(0)} + w_{1,1}a_1^{(0)} + \ldots + w_{1,n}a_n^{(0)} + b_1^{(0)}\right)$$

$$= \sigma\left(\sum_{i=1}^{n} w_{1,i}a_i^{(0)} + b_1^{(0)}\right)$$

$$\begin{pmatrix} a_1^{(1)} \\ a_2^{(1)} \\ \vdots \\ a_m^{(1)} \end{pmatrix} = \sigma\left[\begin{pmatrix} w_{1,0} & w_{1,1} & \ldots & w_{1,n} \\ w_{2,0} & w_{2,1} & \ldots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,0} & w_{m,1} & \ldots & w_{m,n} \end{pmatrix}\begin{pmatrix} a_1^{(0)} \\ a_2^{(0)} \\ \vdots \\ a_n^{(0)} \end{pmatrix} + \begin{pmatrix} b_1^{(0)} \\ b_2^{(0)} \\ \vdots \\ b_m^{(0)} \end{pmatrix}\right]$$

$$a^{(1)} = \sigma\left(\mathbf{W}^{(0)}a^{(0)} + \mathbf{b}^{(0)}\right)$$

- Here $\mathbf{W}^{(0)}$ are weights, $\mathbf{b}^{(0)}$ are bias, and σ is some *non-linear* activation function.

Learning process

- Parameters $c_1, \cdots, c_{n-3}$ are determined by weights and bias.

- We can define the *loss function* as

$$L(\mathbf{W}, \mathbf{b}; \text{training set}) \equiv \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(c_1^{(i)}, \cdots, c_{n-3}^{(i)};\ \xi_1^{(i)}, \cdots, \xi_n^{(i)}),$$

  where N is the number of training points. "Learning" here means performing iterative gradient descent in the space of weights $\mathbf{W}$ and bias $\mathbf{b}$ such that the loss function gets minimized.

- Then one can check how NN performs by looking whether $\mathcal{L} \approx 0$ everywhere on the moduli space $\mathcal{M}_{0,n}$. What we do here is essentially *unsupervised learning*.

Why does it work? – A simple argument

- At the end of gradient descent, the loss function satisfies

$$\forall W : \quad 0 = \frac{dL}{dW} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{n-3}\frac{d\mathcal{L}^{(i)}}{dc_j}\frac{dc_j}{dW}\,,$$

and same for b. This is *almost always* satisfied when

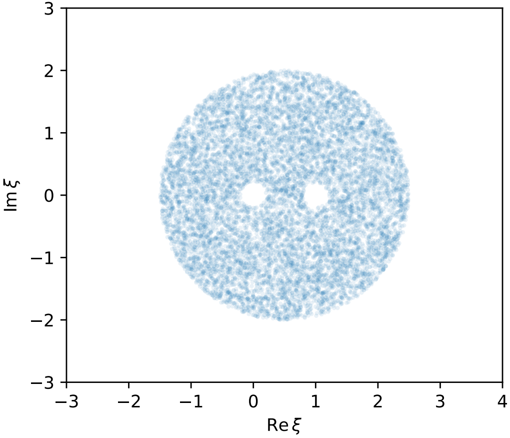
$$\forall i \in \{\text{training set}\} : \quad \frac{d\mathcal{L}^{(i)}}{dc_j} = 0, \quad j = 1, 2, \cdots n-3\,.$$

We say NN learns the function if we can replace $\forall i$ with nearly $\forall \mathcal{M}_{0,n}$.

- There may have been multiple critical points of the function $\mathcal{L}$, but by our experiments it appears we always extremize to the unique global minimum ($\mathcal{L} = 0$) describing Strebel differential.

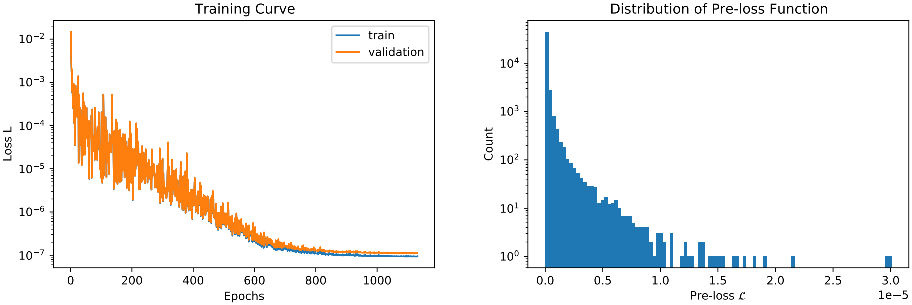
Training set

- We construct training set by uniformly sampling $N = 10^6$ points in the moduli space $\mathcal{M}_{0,4}$ by excluding small regions near $\{0, 1, \infty\}$.
- It is crucial to training region to roughly cover the expected vertex region $\mathcal{V}_{0,4}$.



The training statistics I

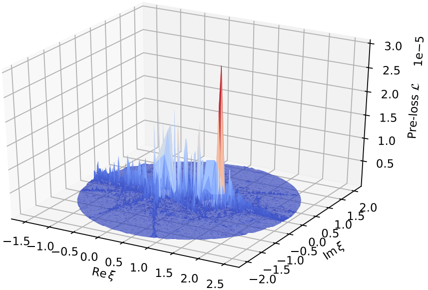
- We use *Google Jax* to set up and train our NN. We setup complex NN with $[1, 512, 128, 1028, 1]$ nodes. Here are some training statistics:



- No overfitting and it takes ≈ 2 hours to train on a laptop's GPU!

The training statistics II

- NN under performs when it is too close to punctures/real line.

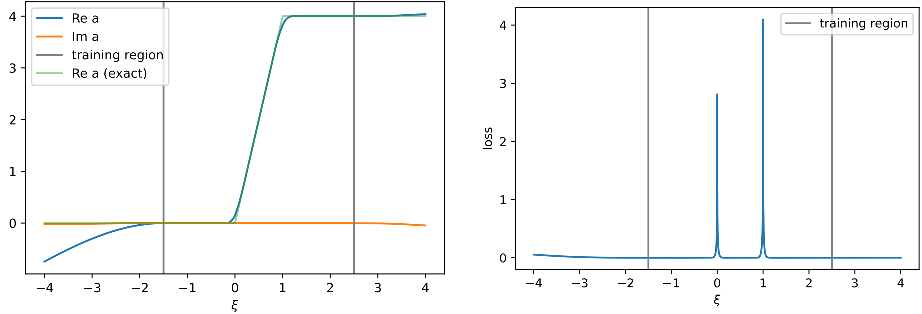


Loss statistics:

- mean $= 1.7 \times 10^{-7}$
- median $= 5.33 \times 10^{-8}$
- min $= 1.01 \times 10^{-11}$
- max $= 3.01 \times 10^{-5}$
- best point: $\xi = 0.53 + 1.34i$
- worst point: $\xi = 1.18 - 0.10i$.

The case when $\xi \in \mathbb{R}$ and $\xi = e^{i\pi/3}$

- Even so, the learned behavior matches quite well with the exact solution for $\xi \in \mathbb{R}$ in the training region.

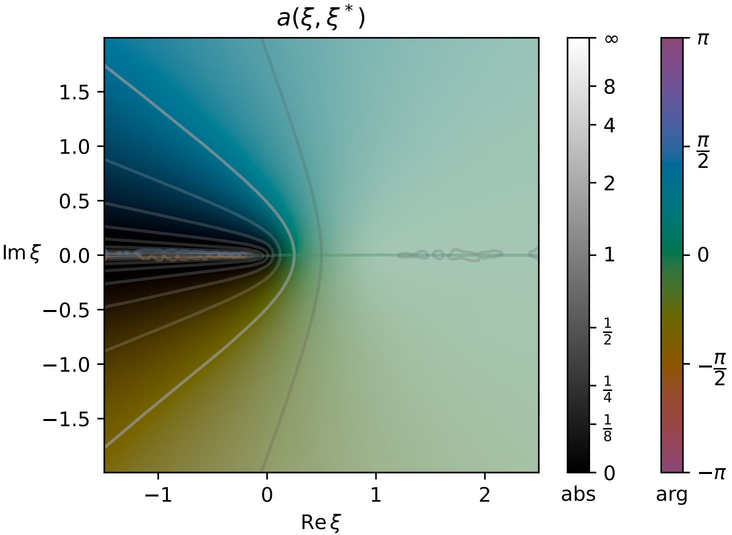


- Further, we find the value at the point $\xi = e^{i\pi/3}$ is relatively close to the exact value

$$a(\xi = e^{i\pi/3}) = 2.0007 + 1.1540j \approx 2.0000 + 1.1547\,.$$

The function $a = a(\xi, \xi^*)$, finally

- The overall behavior of the accessory parameter looks like:

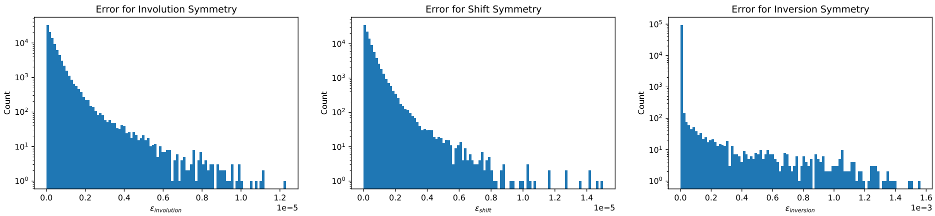


Checking symmetries

- NN learned the symmetries of accessory parameter a. We quantify this by computing the "error" ϵ_g given a point $\xi \in \mathcal{M}_{0,4}$

$$\epsilon_g(\xi, \xi^*) \equiv \left| a(g(\xi), g(\xi^*)) - g(a)(\xi, \xi^*) \right|^2 ,$$

where g represents one of transformation properties of a. Here are the statistics for the errors:

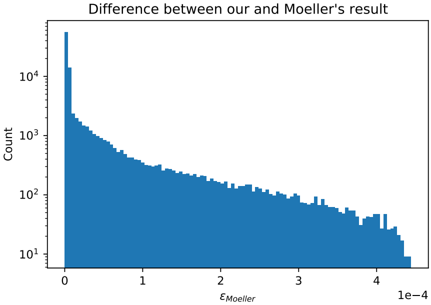


Comparing with Moeller's result

- We can compare our result with the fit provided by Moeller. [1] Again defining the error as

$$\epsilon_{Moeller}(\xi, \xi^*) \equiv |a(\xi, \xi^*) - a_{Moeller}(\xi, \xi^*)|^2 \,,$$

we get



The average error is

$$\langle \epsilon_{Moeller} \rangle \approx 0.005491 \,.$$

[1] hep-th/0408067

Table of Contents

Local coordinates I

- Strebel differential defines local coordinates by its critical graph. That is, there exists a conformal map

$$z = h_i(w_i) = \xi_i + (\rho_i w_i) + d_{i,1}(\rho_i w_i)^2 + d_{i,2}(\rho_i w_i)^3 + \cdots ,$$

from punctured disks $0 < |w_i| \leqslant 1$ to n-punctured sphere for which Strebel differential takes the form it takes for the flat cylinder

$$\varphi = -\frac{dw_i^2}{w_i^2} ,$$

and maps $|w_i| = 1$ to the critical trajectory surrounding the puncture. Here w_i are the local coordinates.

Local coordinates II

- Further, we can expand Strebel differential around the puncture ξ_i

$$\varphi = \left[-\frac{1}{(z-\xi_i)^2} + \frac{b_{i,-1}}{z-\xi_i} + b_{i,0} + b_{i,1}(z-\xi_i) + \cdots \right] dz^2 \, .$$

  Notice each coefficient b is determined by accessory parameters.

- Equating this with φ in local coordinates, one obtains d-coefficients in terms of b-coefficients, hence in terms of c_i. First few terms are

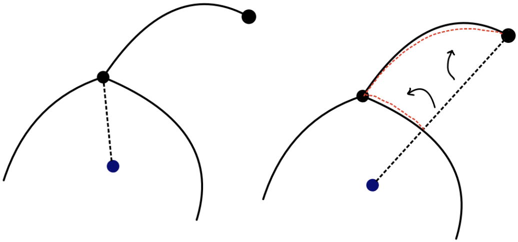
$$\begin{aligned} d_{i,1} &= \frac{1}{2} b_{i,-1}, \\ d_{i,2} &= \frac{1}{16}(7 b_{i,-1}^2 + 4 b_{i,0}), \\ d_{i,3} &= \frac{1}{48}(23 b_{i,-1}^3 + 28 b_{i,-1} b_{i,0} + 8 b_{i,1}) \, . \end{aligned}$$

Mapping radii

- This procedure determines local coordinates up to *mapping radii* ρ_i. But in fact, this can also be determined by evaluating the integral

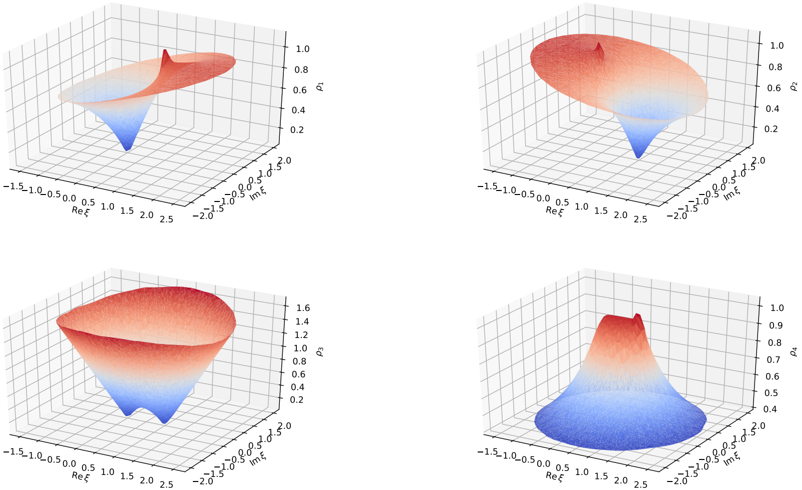
$$\log \rho_i = \lim_{\epsilon \to 0} \left(\mathrm{Im} \int_{\xi_i + \epsilon}^{z_c} \sqrt{\phi(z')} dz' + \log \epsilon \right) ,$$

where the sign of integral is chosen such that limit exists.
- Here z_c can be *any* point on the critical graph, not necessarily at the closest trajectories to the puncture. This is because path after it crosses the closest trajectory don't contribute to the imaginary part.



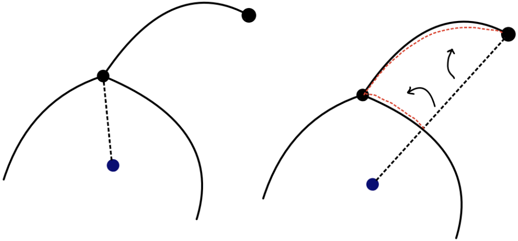
Mapping radii in the case of 4-punctured sphere

- The mapping radii for punctures $0, 1, \xi, \infty$ have the following behavior



Characterizing the critical graph

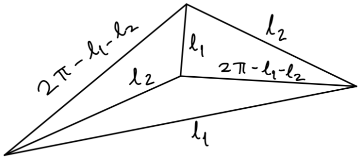
- Looking at $\mathrm{Im}\int_{\xi_i+\epsilon}^{z_c}\sqrt{\phi(z')}dz'$ allows us to detect which zeros are closest to given puncture. This is evident from the figure:

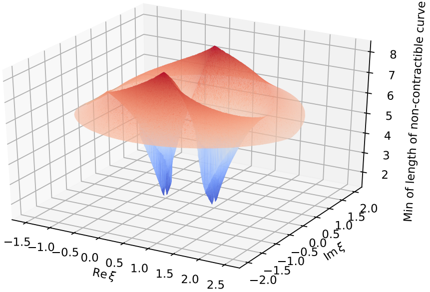


- In the figure on the left, $\mathrm{Im}\int_{\xi_i+\epsilon}^{z_c}\sqrt{\phi(z')}dz'$ would be monotonic as it moves away from the puncture, while for the figure on the right, the same quantity has to be non-monotonic as the contributions after it crosses the critical trajectory should add up to zero.

Finding the lengths of non-contractible curves

- It is possible to obtain the shape of the critical graph! This allow us to find the length of non-contractible curves as a function of moduli [2]





- Recall vertex region $\mathcal{V}_{0,n}$ is the region for which lengths of *all* non-contractible curves are larger than 2π.

[2] Here we cheated and used something specific for the construction of 4-punctured sphere in our code. But in principle this is just a technical issue and work in progress.

Indicator function

- Define *the indicator function* $\Theta : \mathcal{M}_{0,n} \to \{0,1\}$ as

$$\Theta(\xi) = \begin{cases} 1 & \text{if} \quad \xi \in \mathcal{V}_{0,n} \\ 0 & \text{if} \quad \xi \notin \mathcal{V}_{0,n} \end{cases} .$$

- An advantage of defining such function is that it would simplify the form of the moduli integration by replacement

$$\int_{\mathcal{V}_{0,n}} (\cdots) = \int_{\mathcal{M}_{0,n}} \Theta(\xi)(\cdots) .$$

  Any need for explicit description of the region $\mathcal{V}_{0,n}$ is absorbed to the indicator function.

- It is possible to train a neural network to learn this function as well!

Neural network for the indicator function for $n = 4$

- NN will input ξ and produce Θ_{pred}, now viewed as a probability distribution $\Theta_{pred} : \mathcal{M}_{0,n} \to [0, 1]$.

- Training set is constructed by assigning points in $\mathcal{M}_{0,n}$ 1 if the lengths of non-contractible curve is greater than 2π and 0 otherwise.

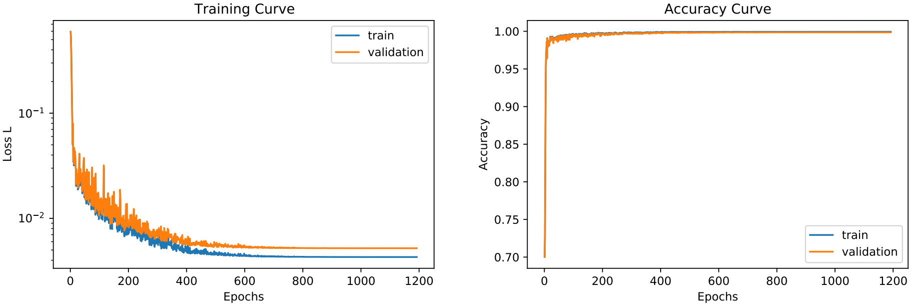
- Then we have a binary classification problem at our hand which we can perform a simple *supervised learning* with *cross-entropy loss*:

$$L = -\frac{1}{N} \sum_{i=1}^{N} \left[\Theta_{true,i} \log \Theta_{pred,i} + (1 - \Theta_{true,i}) \log(1 - \Theta_{pred,i}) \right] ,$$

where $\Theta_{true} \in \{0, 1\}$ are assigned truth values.

Learning statistics for indicator function NN

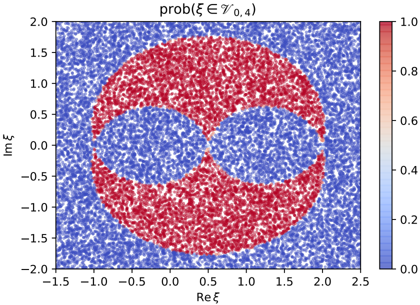
- We trained with $N = 10^6$ points on real NN with $[2, 256, 32, 128, 2]$ nodes. Here are the learning statistics for the training

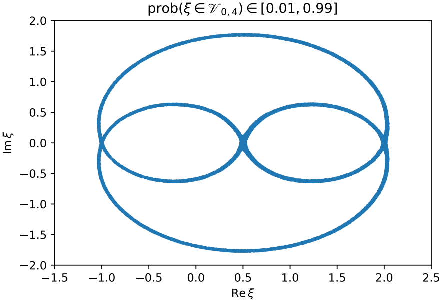


- We have achieved 99.94% training accuracy and 99.87% validation accuracy! No overfitting again.

The indicator function and the region $\mathcal{V}_{0,4}$

- Here are the probabilities indicator function Θ gives:





- In principle one can also train indicator function that would separate s, t, u regions, but we won't need them at this work.

Off-shell 4-tachyon contact term

- Finally, we can compute the off-shell 4-tachyon contact term by combining all ingredients. Its expression is given by

$$v_4 = \frac{2}{\pi} \int_{\mathcal{M}_{0,4}} d^2\xi \; \Theta(\xi) \prod_{i=1}^{4} \frac{1}{\rho_i^2} \, .$$

- We perform a numerical integration with trapezoid method. The numerical value we find is $\boxed{v_4 \approx 72.3868}$. We're still in the process of understanding its statistics. For reference, values in the literature are:

Belopolsky (1994)	72.39
Moeller (2004)	72.390 ± 0.003
Yang & Zwiebach (2005)	72.414

Table of Contents

Advantages

There are few advantages using ML over classical numerical methods:

1. **Scaleable** – algorithm presented here is manifestly independent of number of punctures. As long as it scales in favorable fashion, it gives a systematic numerical approach for obtaining local coordinates.
2. **Ease of use** – after network is trained one can initialize it and begin performing calculations immediately.
3. **Open to exploration** – neural networks are differentiable functions. This allows one to explore various properties of accessory parameters and obtain hidden patterns.

Future directions

- Obviously, we are going to try to obtain local coordinates for $n \geqslant 5$ and perform level truncation in CSFT. How do level and order truncations interact?
- It may be possible to train a NN for the differential that gives local coordinates in Feynman region by adjusting pre-loss function and training appropriately. This give an alternative way to sewing to obtain these local coordinates. It may be interesting to cross-check.
- It may be possible to perform *symbolic regression* now we have NNs.
- Higher genus? Hyperbolic vertices?