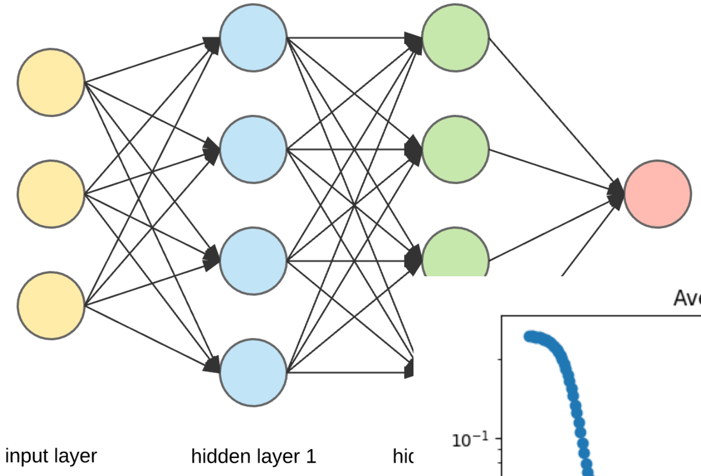
Lecture 2: ML CY metrics

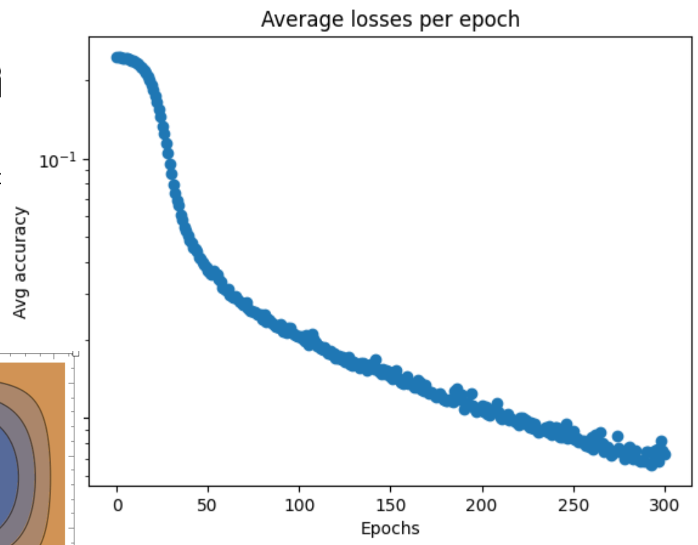
Simple NNs for tricky geometry

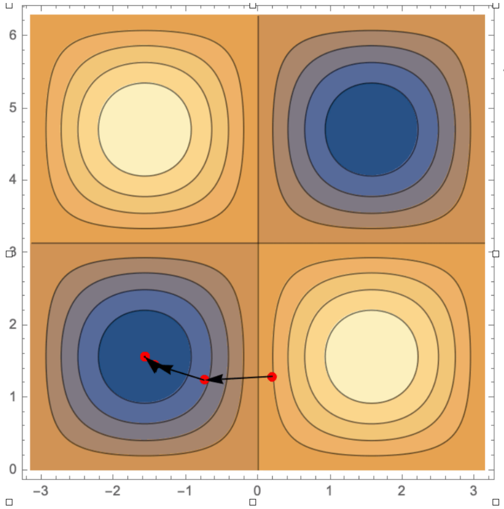
Magdalena Larfors, Uppsala University
Nordita Winter School 2024

Summary lecture 1

- Neural Networks:
  universal function approximators
- NNs: parametrized maps
  $f_\theta\colon\ \mathbb{R}^n \to \mathbb{R}^m$
- Train NN = change θ to reduce loss
- Stochastic Gradient Descent with Backpropagation (or some refinement)
- Use ML libraries:
  PyTorch, JAX, TensorFlow/Keras







Unsupervised and semi-supervised learning

- Lecture 1: supervised learning
  had labelled data (x, y)
  trained network using e.g. $L_{MSE} = \frac{1}{N}\sum\left(y(x_i) - f_\theta(x_i)\right)^2$
- Universal function approximators
  → NN can also predict unknown functions
- Unlabelled data → unsupervised learning: clustering techniques
  e.g. heterotic orbifold models Mutter et al:18, heterotic line bundle models Otsuka-Takemoto:20, type IIB flux vacua Cole-Shiu:17,18 ...

Unsupervised and semi-supervised learning

- Sometimes have unlabelled data with known constraints
  e.g. the function we want solves known constraint/equation

- In this case can use semi-supervised learning
- Encode constraints as custom (addition to) loss function
- In ML literature called PINN (Physics Informed Neural Networks)

- Examples:
  - Numerically solve Navier-Stokes equation
  - Compute the Ricci-flat metric on a CY manifold

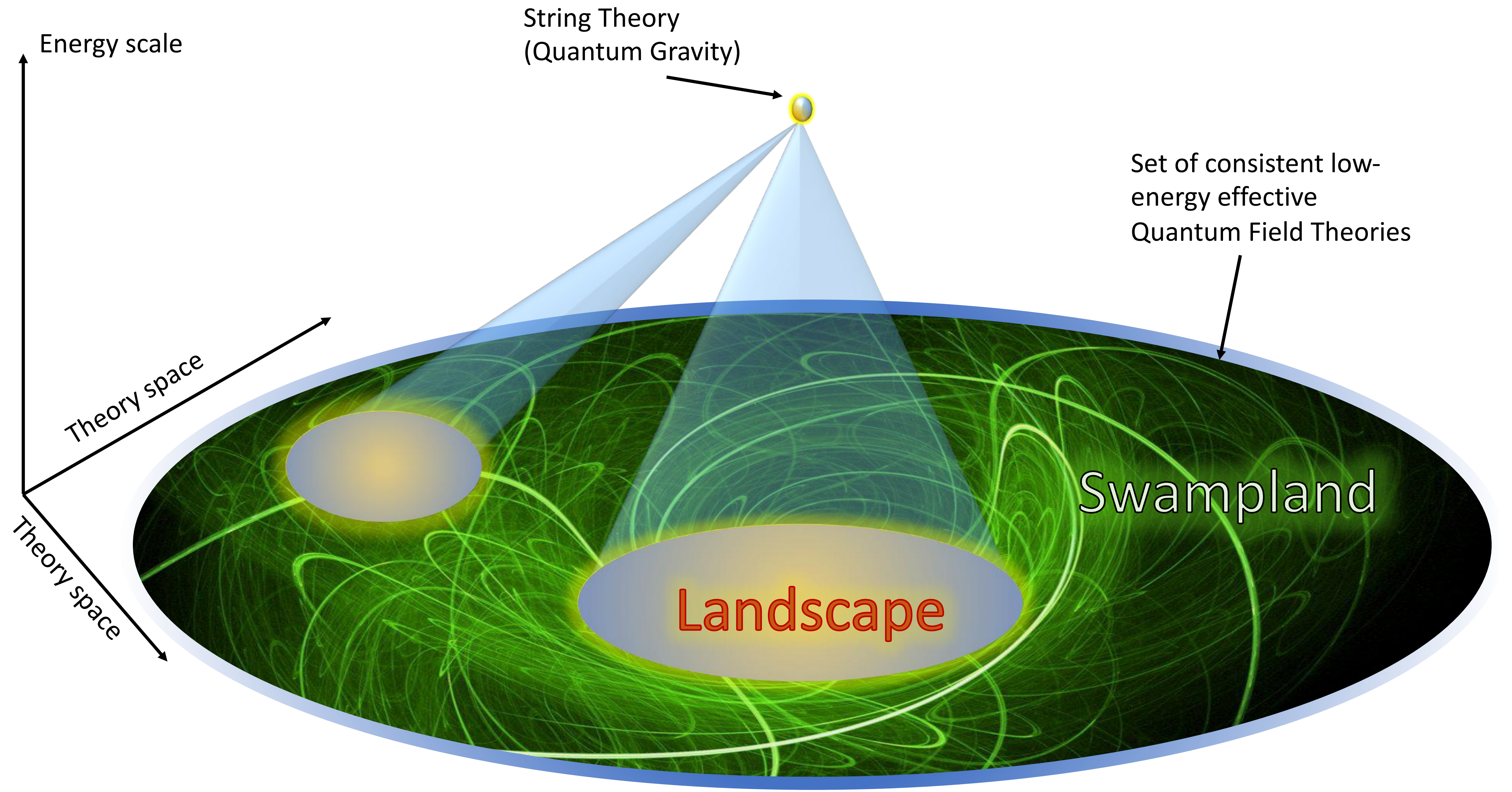
Outline

- String theory and Calabi-Yau (CY) geometry
- ML of Ricci flat CY metrics
- Data generation
- Semi-supervised learning and custom loss functions
- Comparing architectures

Motivation & problem set-up

String theory and Calabi-Yaus

- String theory: theory of quantum gravity
- **String compactifications**: connect with 4d particle physics, cosmology
- Topology and geometry of compact dimensions are key
- **Calabi-Yau manifolds** are popular (compact) example spaces:
  - Give SUSY Minkowski vacua (with moduli), used in flux comp's, …
  - Admit Ricci-flat metric
  - Many example manifolds constructed
  - Topology well understood (computed in examples)
- The *Ricci-flat CY metric* gives info on curvature, massive KK modes, …
  Can we compute it in examples?



Picture source: Palti (19)

Calabi-Yau manifolds: details

- Complex: local coordinates $z_i, \bar{z}_{\bar{j}}$
  holomorphic top form $\Omega = dz_1 \wedge dz_2 \wedge \cdots \wedge dz_n$
- Kähler: metric determined by **Kähler potential** $K(z, \bar{z})$

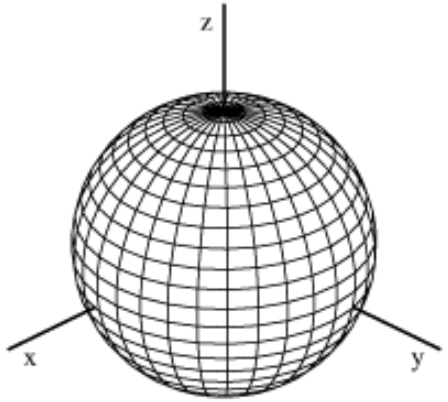
$$g_{i\bar{j}} = \partial_i \partial_{\bar{j}} K, \quad g_{ij} = g_{\bar{i}\bar{j}} = 0$$

Kähler form $J = \frac{i}{2} \sum g_{i\bar{k}} dz^j \wedge d\bar{z}^{\bar{k}}$

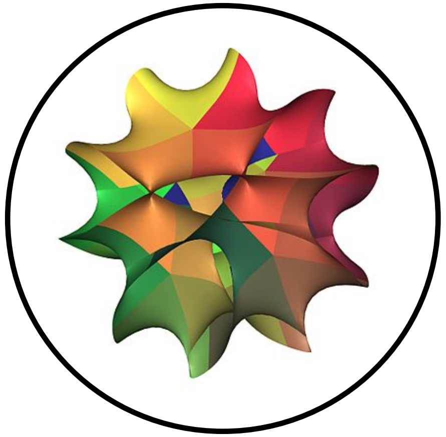
- Come in families parametrized by *complex structure/Kähler moduli*
- Satisfy topological restriction ($c_1 = 0$); unique Ricci-flat CY metric

Calabi-Yau manifolds: algebraic construction

- Non-compact CYs are not hard
- Build *compact* CYs from simpler ambient spaces (compact, complex, Kähler)



$x^2 + y^2 + z^2 = 1$ in $\mathbb{R}^3$

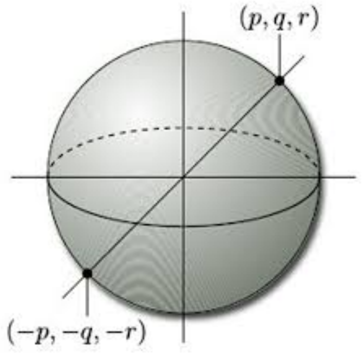


$Z_0^5 + Z_1^5 + Z_2^5 + Z_3^5 + Z_4^5 = 0$ in $\mathbb{P}^4$

- Many examples collected in databases:
  CICY 3-folds Candelas et al:88, CY hypersurfaces in toric spaces Kreuzer-Skarke:00, …

Addition: why is $\mathbb{P}^4$ a simpe ambient choice?

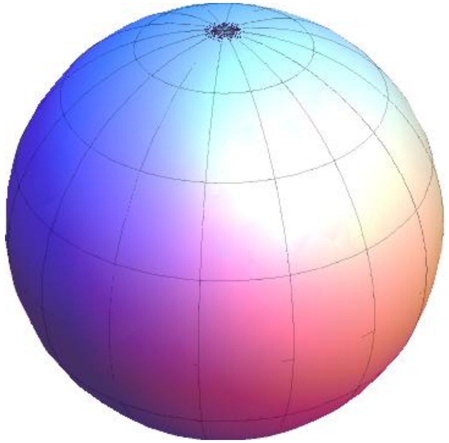
- Want compact, complex space: can't use $\mathbb{C}^d$ as it is non-compact
- $\mathbb{P}^d$ space of complex line through origin of $\mathbb{C}^{d+1}$
  $\mathbb{P}^d: \{(z_1, z_2, \dots z_{d+1}) \in \mathbb{C}^{d+1}: (z_1, z_2, \dots z_{d+1}) \sim \lambda(z_1, z_2, \dots z_{d+1})\}$
- Pictorially, easier to visualize real projective space, e.g. $\mathbb{RP}^2$ is hemisphere of 2-sphere in $\mathbb{R}^3$ with antipodal identification on equator



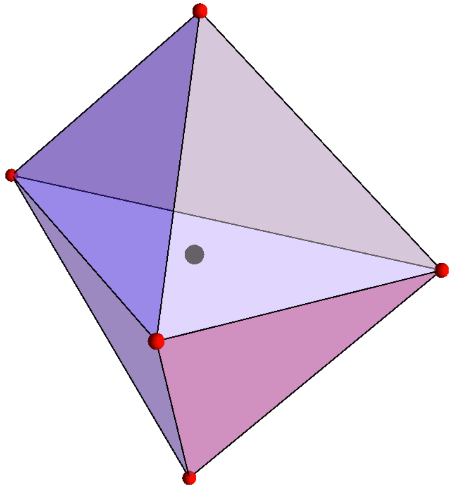
- For complex projective space, exists map to sphere
  $\mathbb{P}^1 \sim S^2/\mathrm{U}(1)$; $\mathbb{P}^4 \sim S^9/\mathrm{U}(1)$
  $\mathbb{P}^d$ have "FS metric" which is basically the round metric of sphere
  → given this, $\mathbb{P}^4$ is rather simple

CY manifolds from simpler ambient spaces

- CICY 3-folds Candelas et al:88
- Ambient: cpl projective spaces
  $$\mathbb{P}^{n_1} \times \mathbb{P}^{n_2} \times .. \mathbb{P}^{n_m}$$



- CY from KS list Kreuzer-Skarke:00, …
- Ambient: toric variety given by lattice polytope



We use the (relative) simplicity of the ambient space to compute things on CY

CY manifolds and Ricci flat metrics

Calabi:54, Yau:78

- Let X be an n-dimensional compact, complex, Kähler manifold with vanishing first Chern class.
  Then in any Kähler class $[J]$, X admits a unique Ricci flat metric g_{CY}.

- Problem: there is *no analytical expression* for g_{CY}.
- Impose Ricci-flatness: solve *4th order PDE* for Kähler pot. This is hard.

Ricci-flat CY metrics

Calabi:54, Yau:78

- Let X be an n-dimensional compact, complex, Kähler manifold with vanishing first Chern class.
- Then in any Kähler class $[J]$, X admits a unique Ricci flat metric g_{CY}.
- There is *no analytical expression* for g_{CY}.

But on CY spaces, we know more! Kähler form $J_{CY} \rightsquigarrow g_{CY}$ satisfies

- $J_{CY} = J + \partial\bar{\partial}\phi$ — same Kähler class; ϕ is a function
- $J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega}$ — *complex Monge-Ampere equation*

κ constant on X: *2nd order PDE for* ϕ

Ricci-flat CY metrics

Calabi:54, Yau:78

- Let X be an n-dimensional compact, complex, Kähler manifold with vanishing first Chern class. Then in any given Kähler class $[J]$, X admits a unique Ricci flat metric g_{CY}.
- There is *no analytical expression* for g_{CY}.

We can compute these in examples!

Kähler form $J_{CY} \leadsto g_{CY}$ satisfies

- $J_{CY} = J + \partial\bar{\partial}\phi$ same Kähler class; ϕ is a function
- $J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega}$ complex Monge-Ampere equation
  κ constant on X: 2nd order PDE for ϕ

Setting up the problem:

Find Ricci flat CY metric $g_{CY} \iff$ find J_{CY} that solves MA equation

$$J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega}$$

where κ is some complex constant.

Numerical method: Sample large set of random points on CY.

- Compute Ω and a reference J at all points
- Solve MA eq. numerically for $J_{CY} = J + \partial\bar{\partial}\phi$
- Check approximation: does MA eq hold and is Ricci tensor 0?

Numerical CY metrics – a longstanding quest

- Donaldson algorithm

  Donaldson:05, Douglas-et.al:06, Douglas-et.al:08, Braun-et.al:08, Anderson-et.al:10, ...,

- Energy functionals

  Headrick–Nassar:13, Cui–Gray:20, Ashmore–Calmon–He–Ovrut:21, ...

- Machine learning

  Ashmore–He–Ovrut:19, Douglas–Lakshminarasimhan–Qi:20,
  Anderson–Gerdes–Gray–Krippendorf–Raghuram–Ruehle:20, Jejjala–Mayorga–Peña:20 ,
  Larfors-Lukas-Ruehle-Schneider:21, 22, Ashmore–Calmon–He–Ovrut:21,
  Berglund–Butbaia–Hübsch–Jejjala–Mayorga Peña–Mishra–Tan:22,
  Gerdes–Krippendorf:22...

Numerical CY metrics – a longstanding quest

- Donaldson algorithm

  Donaldson:05, Douglas-et.al:06, Douglas-et.al:08, Braun-et.al:08, Anderson-et.al:10, …,

- Energy functionals

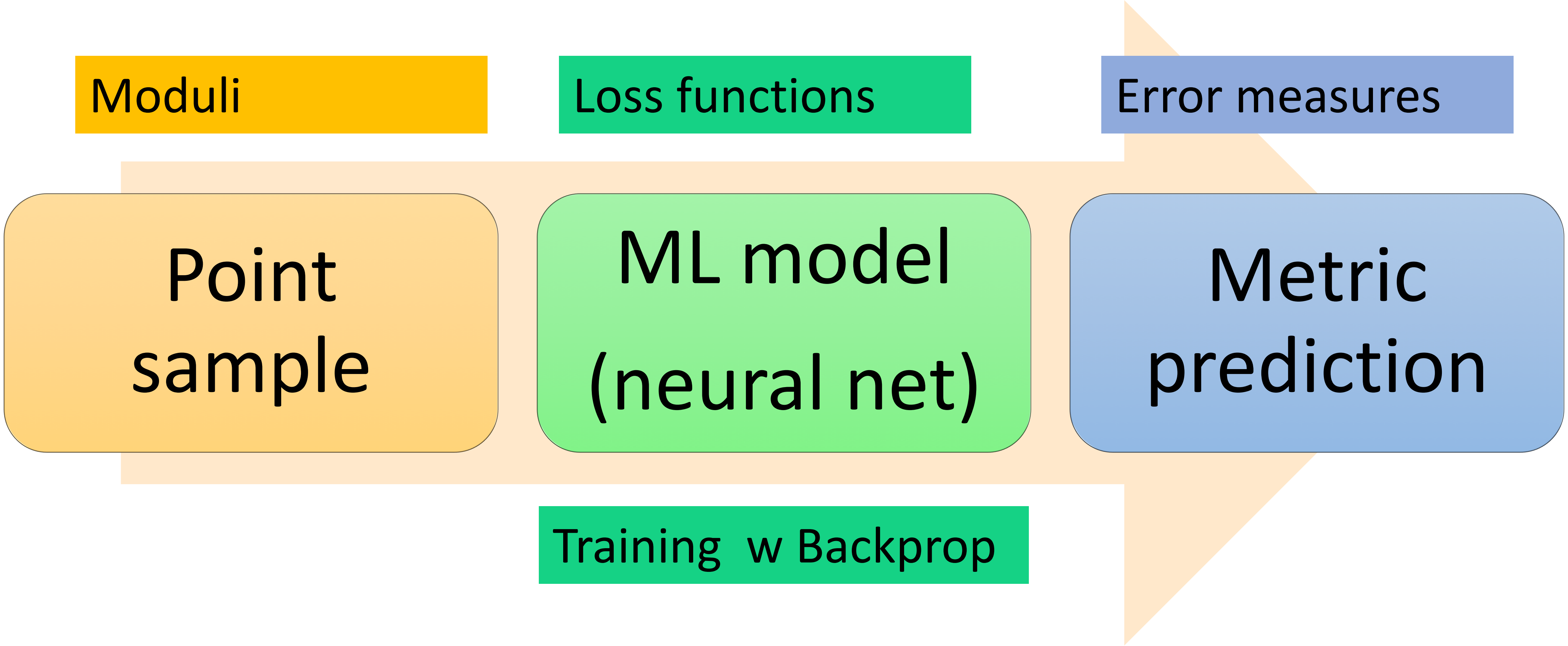
  Headrick–Nassar:13, Cui–Gray:20, Ashmore–Calmon–He–Ovrut:21, …

- Machine learning

  Ashm
  Ande *ña:20 ,*
  Larfo
  Bergl
  Gerde

https://github.com/yidiq7/MLGeometry	TensorFlow/Keras
https://github.com/pythoncymetric/cymetric	TensorFlow/Keras Mathematica & SAGE
https://github.com/ml4physics/cyjax	JAX

Machine Learning implementation



ML implementation

- Create data sample

- Train ML model with points sampled from CY
  (at given point in moduli space)

- The trained NN is the (approximation of) the Ricci flat metric

- Test accuracy of prediction

Error measures used to measure accuracy

After training, evaluate performance (on separate test set):

$\boxed{\text{does the MA equation hold? is the metric Ricci flat?}}$

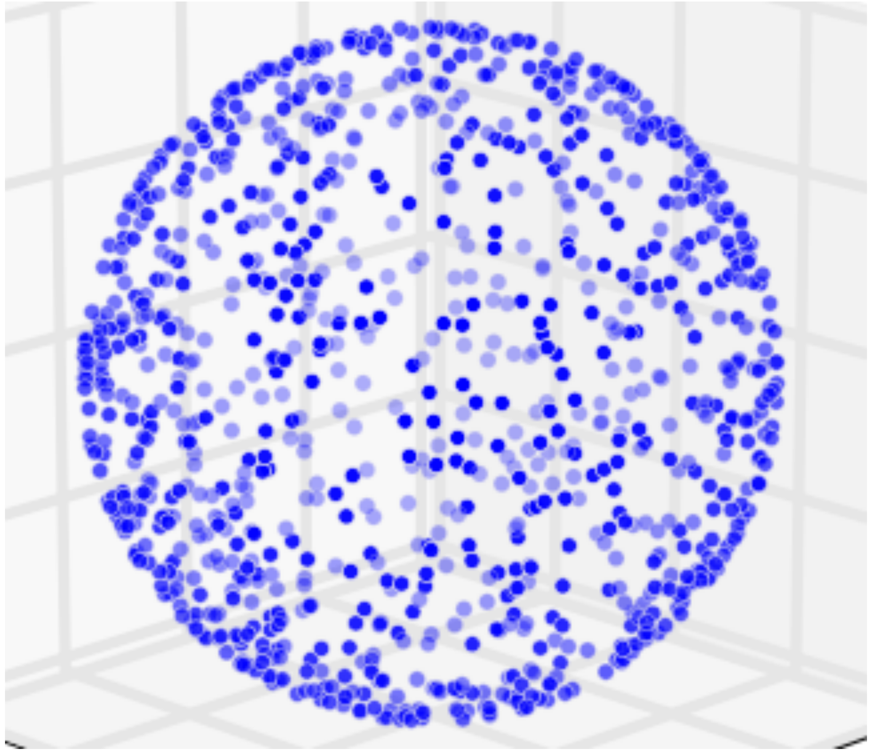
Check via established benchmarks:

$$\sigma = \frac{1}{\mathrm{Vol}_{\mathsf{CY}}} \int_X \left| 1 - \kappa \, \frac{\Omega \wedge \overline{\Omega}}{(J_{\mathrm{pr}})^3} \right| \, , \; \mathcal{R} = \frac{1}{\mathrm{Vol}_{\mathsf{CY}}} \int_X |R_{\mathrm{pr}}| \, .$$

using Monte Carlo integration for any function f

$$\int_X \mathrm{dVol}_{\mathsf{CY}} f = \int_X \frac{\mathrm{dVol}_{\mathsf{CY}}}{\mathrm{d}A} \mathrm{d}A \, f = \frac{1}{N} \sum_i w_i f|_{p_i} \quad \text{with} \quad w_i = \frac{\mathrm{dVol}_{\mathsf{CY}}}{\mathrm{d}A}|_{p_i}$$

Creating data -- Point generators



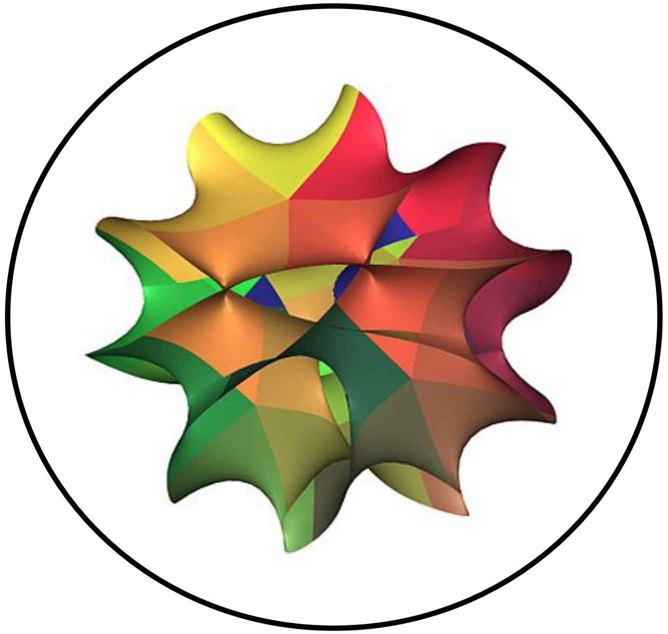
Point generators

We need

- random set of points on CY
- sampled w.r.t. known measure dA

..so we can

- determine global metric of CY
  (*not* enough to work locally, in a patch)
- compute integrals
  (e.g to check accuracy)



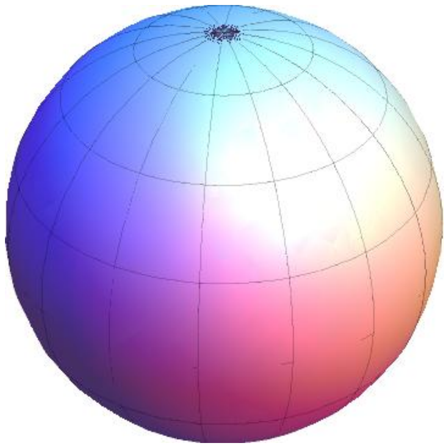
Point generators

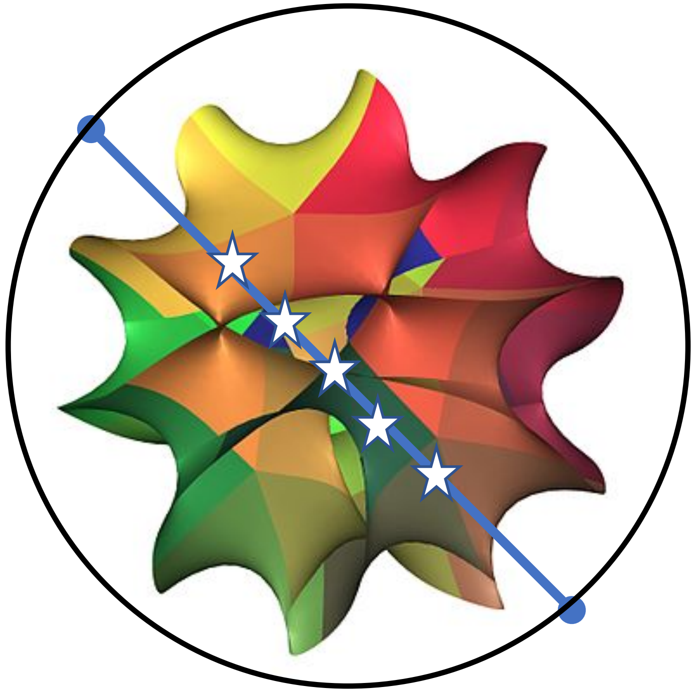
Start simple: Quintic $X: p = 0 \subset \mathbb{P}^4$

Sample 2 points on $\mathbb{P}^4$; connect & intersect

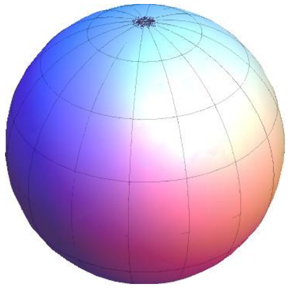
- Repeat M times
  $\rightsquigarrow 5M$ random points on X
- Shiffman-Zelditch theorem:
  points distributed w.r.t. FS measure on X

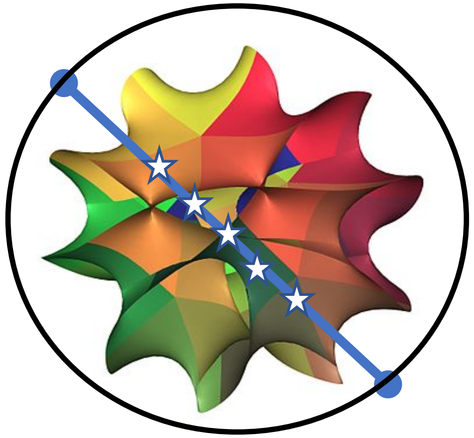
Douglas et. al: hep-th/0612075





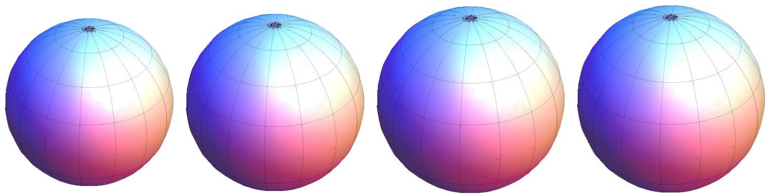
Point generators



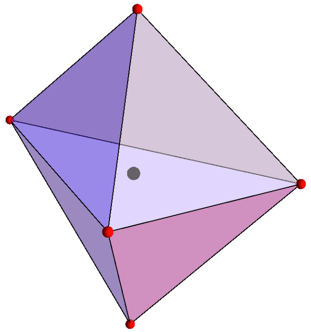


Quintic $X: p = 0 \subset \mathbb{P}^4$ Douglas et. al: hep-th/0612075

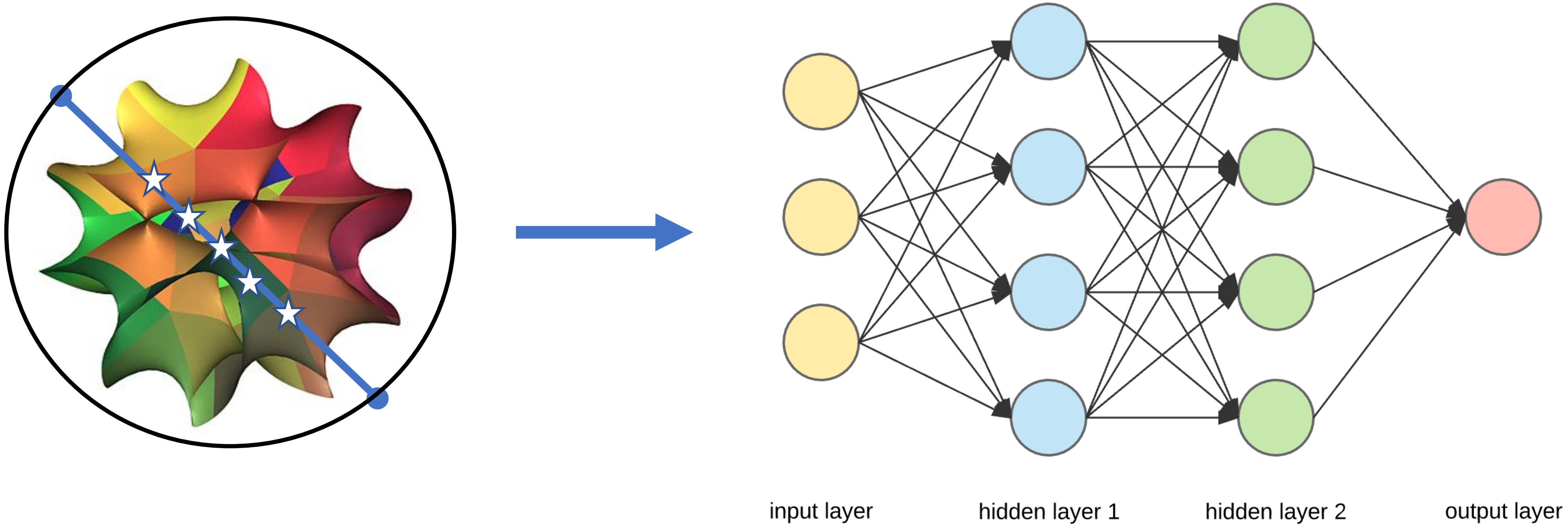
- Sample 2 points on $\mathbb{P}^4$; connect & intersect
- Shiffman-Zelditch theorem:
  points distributed w.r.t. FS measure on X



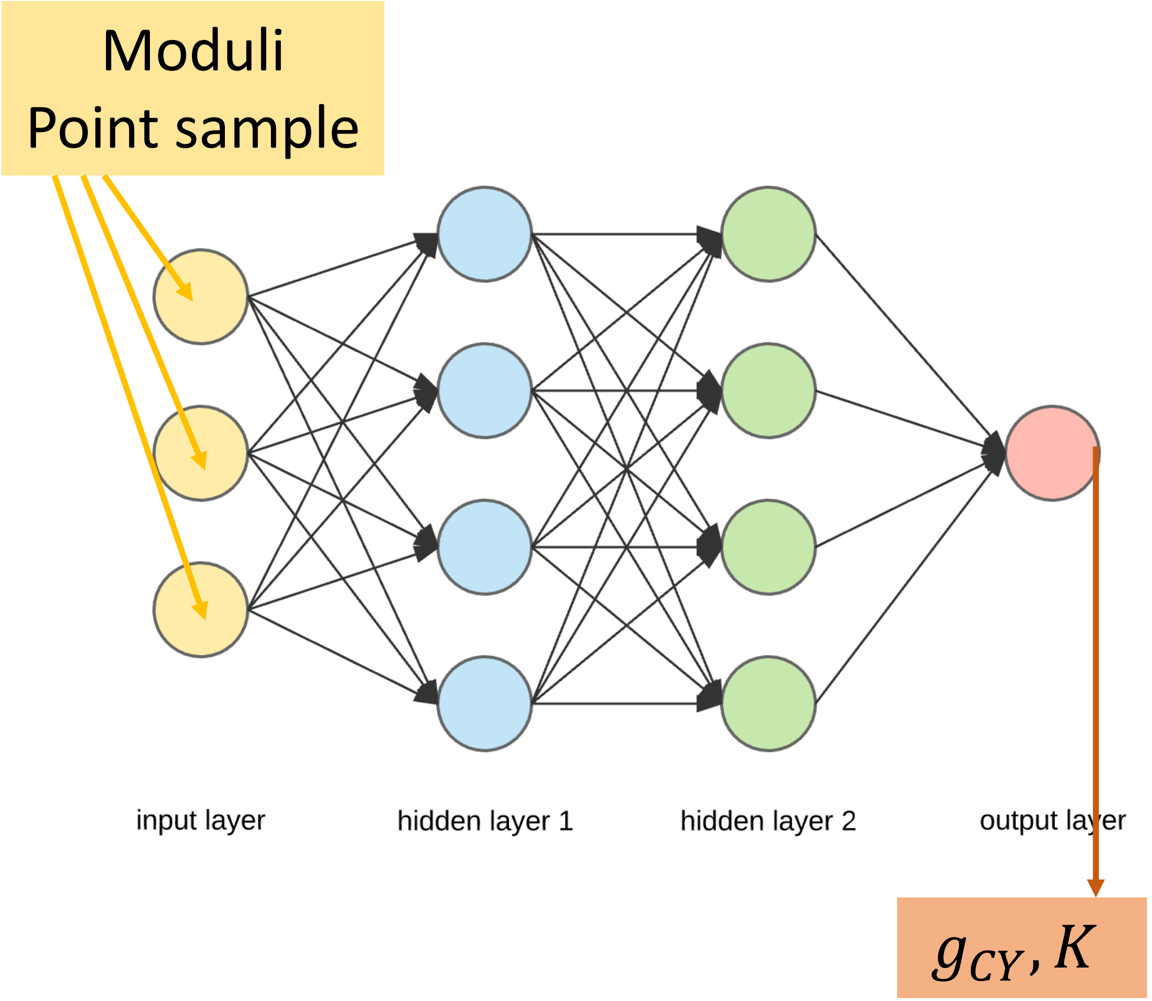
- Generalizations:
  CICY Douglas et.al 0712.3563, ...,
  Kreuzer-Skarke
  ML, Lukas, Ruehle, Schneider 2111.01436 & 2205.13408



ML for CY metrics



ML models: Set-up and training



Architectural choices

- What to predict
  CY metric or Kähler pot?
- Encode constraints in NN or loss?
  (global, complex, Kähler…)

Then train

- Minimize loss functions

And check performance

- Error measures

So what loss functions should we use?

Loss functions encode math constraints

- Train the network to get *unknown* Ricci-flat metric (in given Kähler class)
- Use semi-supervised learning
  1. Encode mathematical constraints as custom loss functions
  2. Train network (adapt layer weights) to minimize loss functions
- E.g. satisfy Monge-Ampere eq $\rightsquigarrow$ minimize Monge-Ampere loss

$$\mathcal{L}_{\text{MA}} = \left|\left|1 - \frac{1}{\kappa}\frac{\det g_{\text{pr}}}{\Omega \wedge \bar{\Omega}}\right|\right|_n$$

- Depending on metric ansatz, need more or less loss functions.

More loss functions

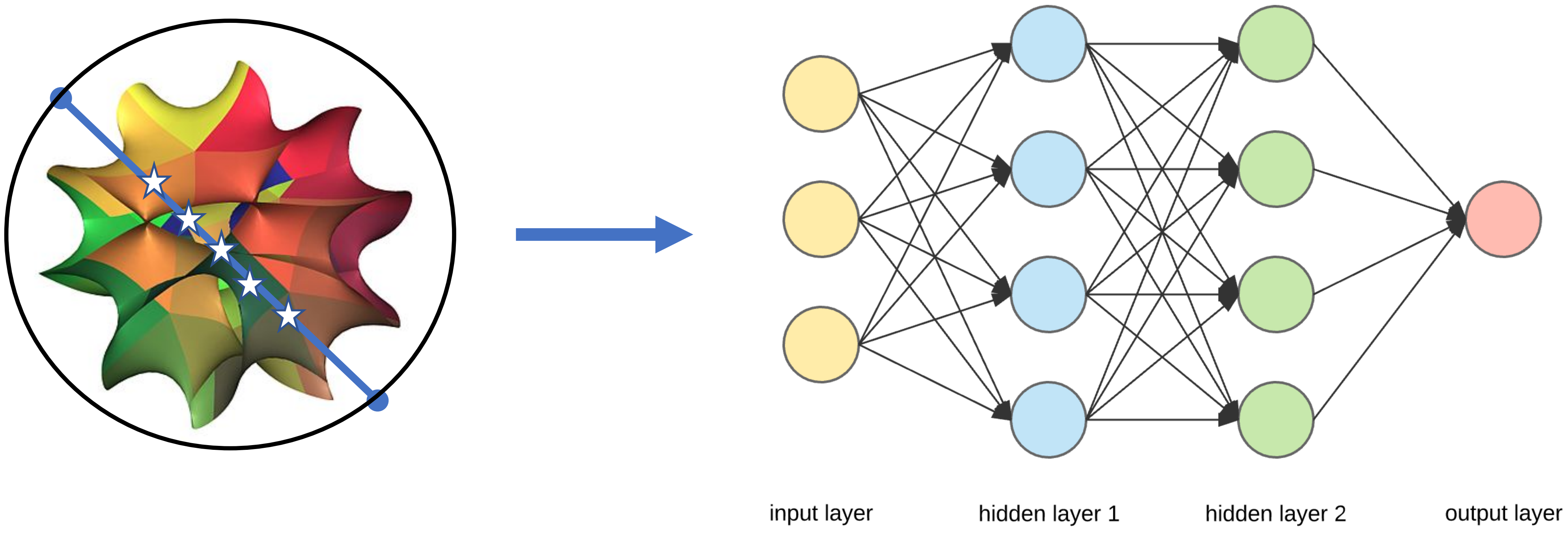
- Satisfy Monge-Ampere eq $\rightsquigarrow$ minimize MA loss
- *OR* Set Ricci tensor to zero $\rightsquigarrow$ minimize Ricci loss (requires derivatives)

Also might need to check

- manifold-ness: match metrics on patch overlaps (requires derivatives)
- Kähler-ity: $d\, J_{pr} = 0$ (requires derivatives)
- Preserve Kähler class $J_{pr} \sim J_{FS}$ (only needed when CY has several Kahler moduli)

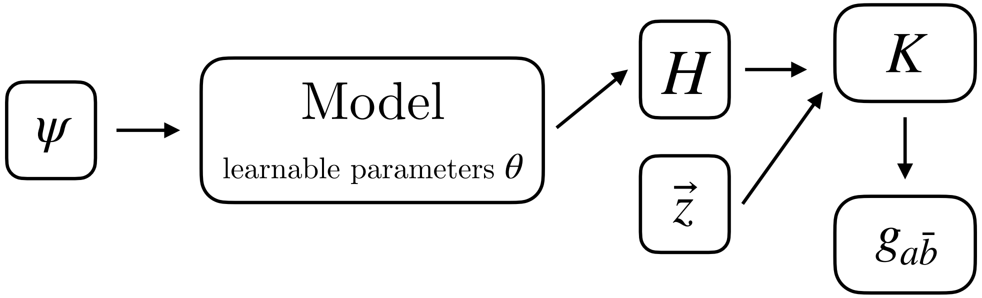
Architectural choices will determine which loss functions we need
Computing derivatives wrt input – (ab)use ML library’s autodiff implementation!

ML implementations

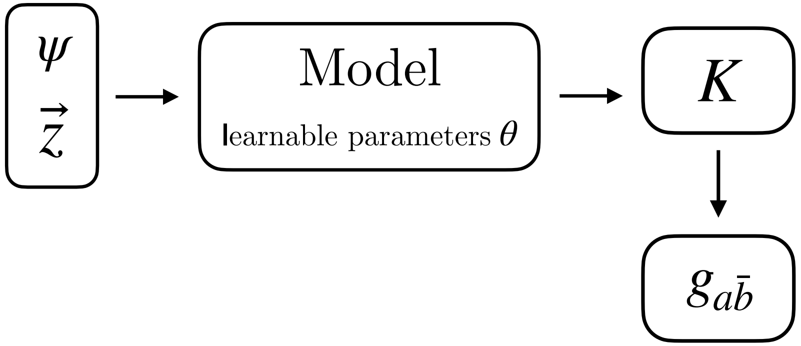


ML model architectures

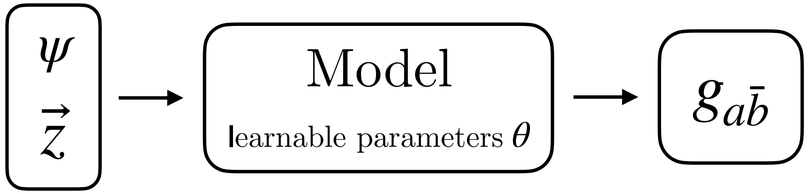
1. Learn Donaldson’s H matrix
   Anderson et al 2012.04656, Gerdes et al 2211.12520



2. Learn Kähler potential
   Anderson et al 2012.04656, Douglas et al 2012.04797, Larfors et al 2111.01436 & 2205.13408 , Berglund et al 2211.09801



3. Learn metric
   Anderson et al 2012.04656, Jejjala et al 2012.15821, Larfors et al 2111.01436 & 2205.13408



ML implementation– two paths

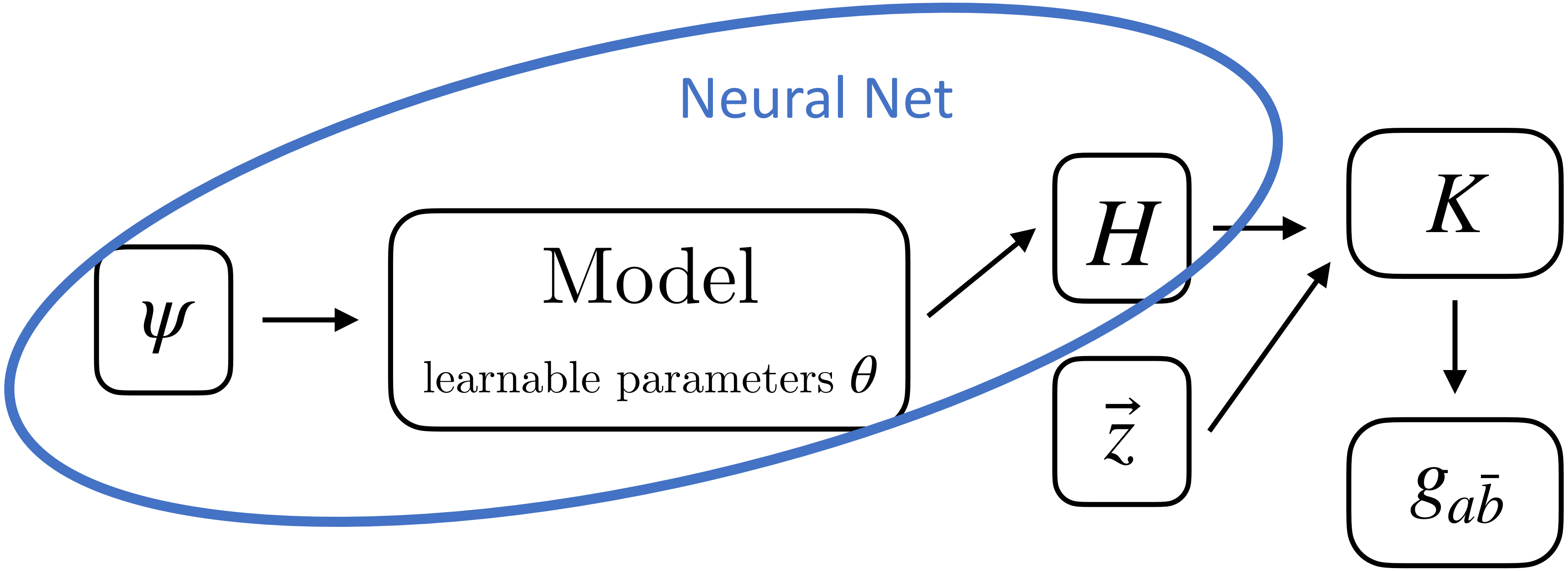
Algebraic CY metrics

- Expand K_{CY} in polynomial basis
- $K_k(z,\bar{z}) = \frac{1}{k}\sum \ln H_{a\bar{b}} s^a \bar{s}^{\bar{b}}$
- ML Hermitian matrix H for given moduli
- Compute Kähler pot from H

Machine Learning CY metrics

- ML model searches freely for CY metric
- Training objective: minimize loss
- Control evolution via NN architecture and loss functions [typically need all loss functions]

1. Learn Donaldson's H matrix



1. Learn Donaldson’s H matrix

Donaldson’s algorithm:

Iterative algorithm (no ML) that gives Kähler potential

$$K_k(z,\bar{z}) = \frac{1}{k}\sum \ln H_{a\bar{b}} s^a \bar{s}^{\bar{b}}$$

- s_a monomials of order k (sections of holomorphic line bundle)
- H: $N_k \times N_k$ Hermitian matrix, “balanced metric”
- Larger k gives larger set of s_a → more accurate K
- Problem: Curse of dimensionality, need to use discrete symmetries

1. Learn Donaldson's H matrix

Donaldson's algorithm: algebraic K from H

NN that predicts H

- Input layer: complex structure moduli
- Output layer: H matrix
- Predicted H + s_α at points → K in spectral basis → algebraic metric
- Either supervised learning
- or semi-supervised learning with MA/Ricci loss function

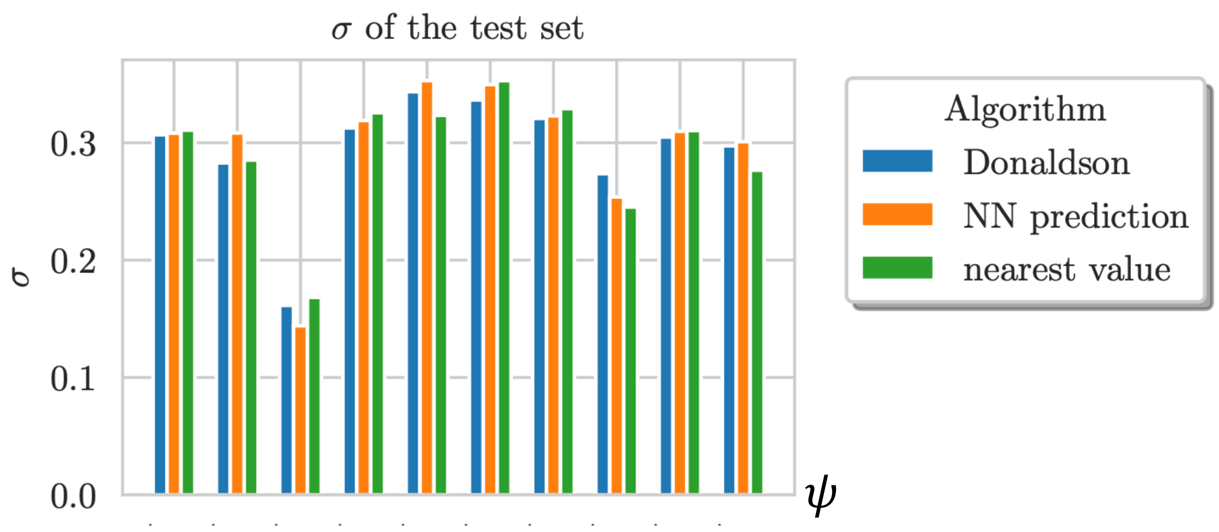
Anderson et al 2012.04656, Gerdes et al 2211.12520, cyjax

Example: supervised learning of H

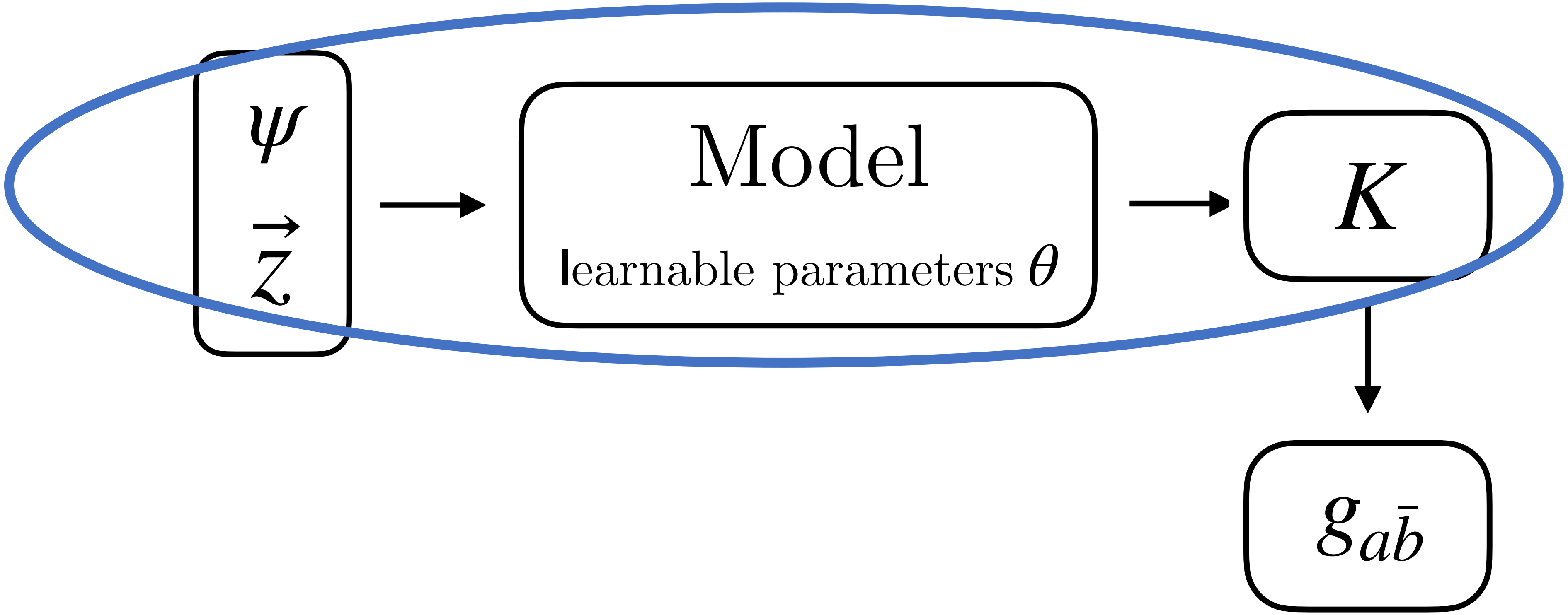
Anderson et al 2012.04656

- Quintic, 1 cpl modulus
- $k = 3 \rightarrow$ 35-dim basis of sections s_α
- Input Re ψ, Im ψ, Abs ψ
- Output Re, Im of H components; compare with Donaldson
- FF NN, 3 layers, ADAM opt.

Layer	Number of Nodes	Activation	Number of Parameters
input	3	–	–
hidden 1	100	leaky ReLU	400
hidden 2	1000	leaky ReLU	101 000
hidden 3	1000	leaky ReLU	1 001 000
output	N_k^2	identity	$1000\times N_k^2 + N_k^2$



2. Learn Kähler potential directly



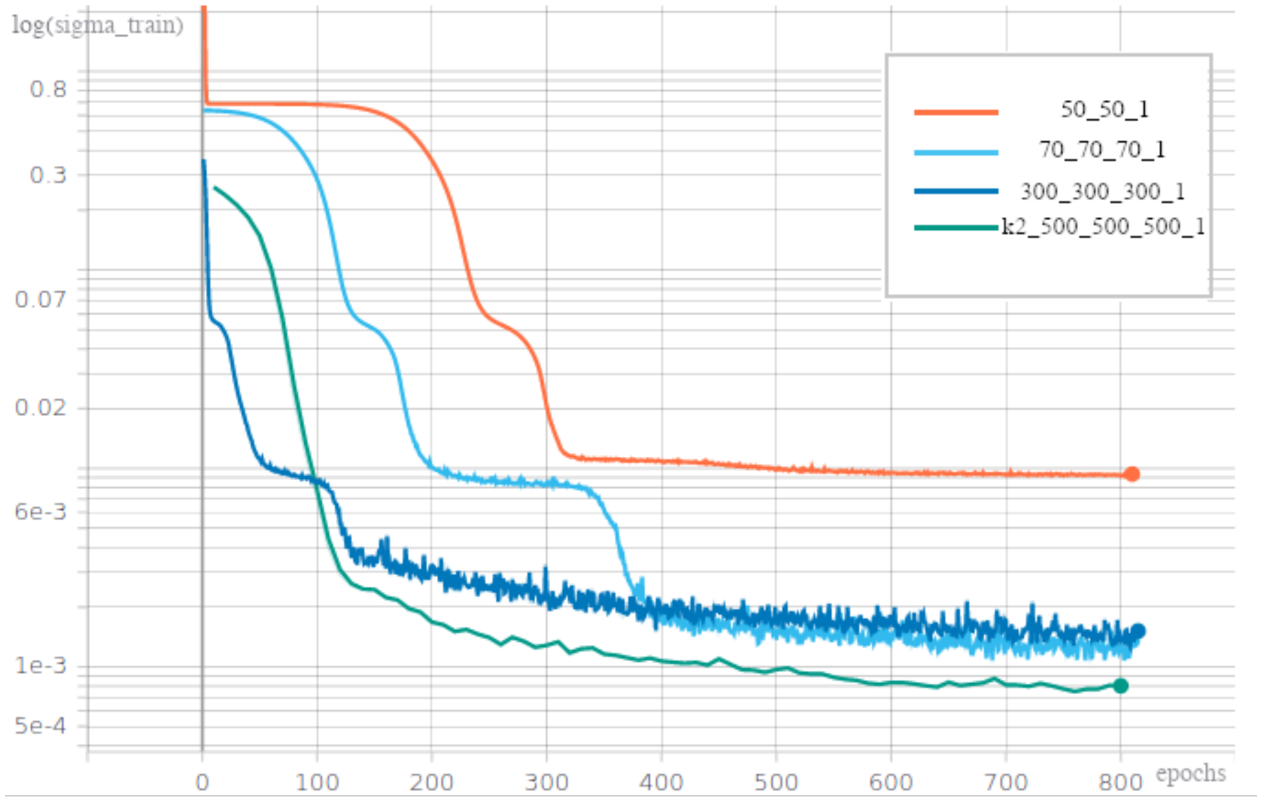
2. Learn Kähler potential directly

Douglas et al 2012.04797, holomorphic and bihomogeneous NN

- Input: points on CY
- Output: prediction for K
- Must ensure K is globally defined
  Guaranteed if expand in section basis (Donaldson, Headrick-Nassar)
  Or have embedding NN (holomorphic or bihomogeneous)
- Bihomogeneous NN:
  Input $x_a \to x_a\overline{x_b} \to Re, Im$; Act. fcn: $\sigma: x \to x^2$
- $K = \log W^d \circ \sigma \circ \cdots \circ \sigma \circ W^1(x_a\overline{x_b})$

Example: semisupervised learning of K

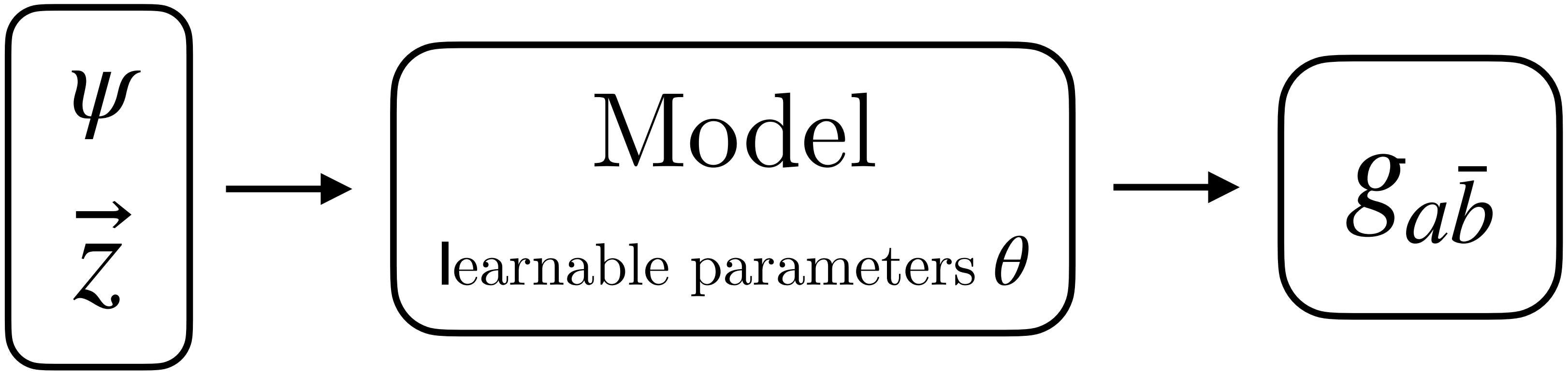
- Semi-supervised learning
- MAPE version of MA loss
- After training:
  NN → K → approximate CY metric
- Gradient blow-ups/deep NN



The training curves for Equation (3) with $\psi = 0.5$, trained with Adam optimizer and MAPE loss. The data for k2_500_500_500_1 was recorded every 10 epochs.

Douglas et al 2012.04797, mlgeometry

3. Direct ML of metric



3. Direct ML of metric: neural network

- Input: point on CY
  Quintic: input layer has 10 nodes = $Re(x_I\ ), Im(x_I)$
- Output: metric prediction - different Ansatze possible
  9 (or 1) node
- Semi-supervised learning using custom loss function
- After training:
  NN → approximate CY metric

Anderson et al 2012.04656, Larfors et al 2205.13408, cymetric

3. Direct ML of metric: neural network

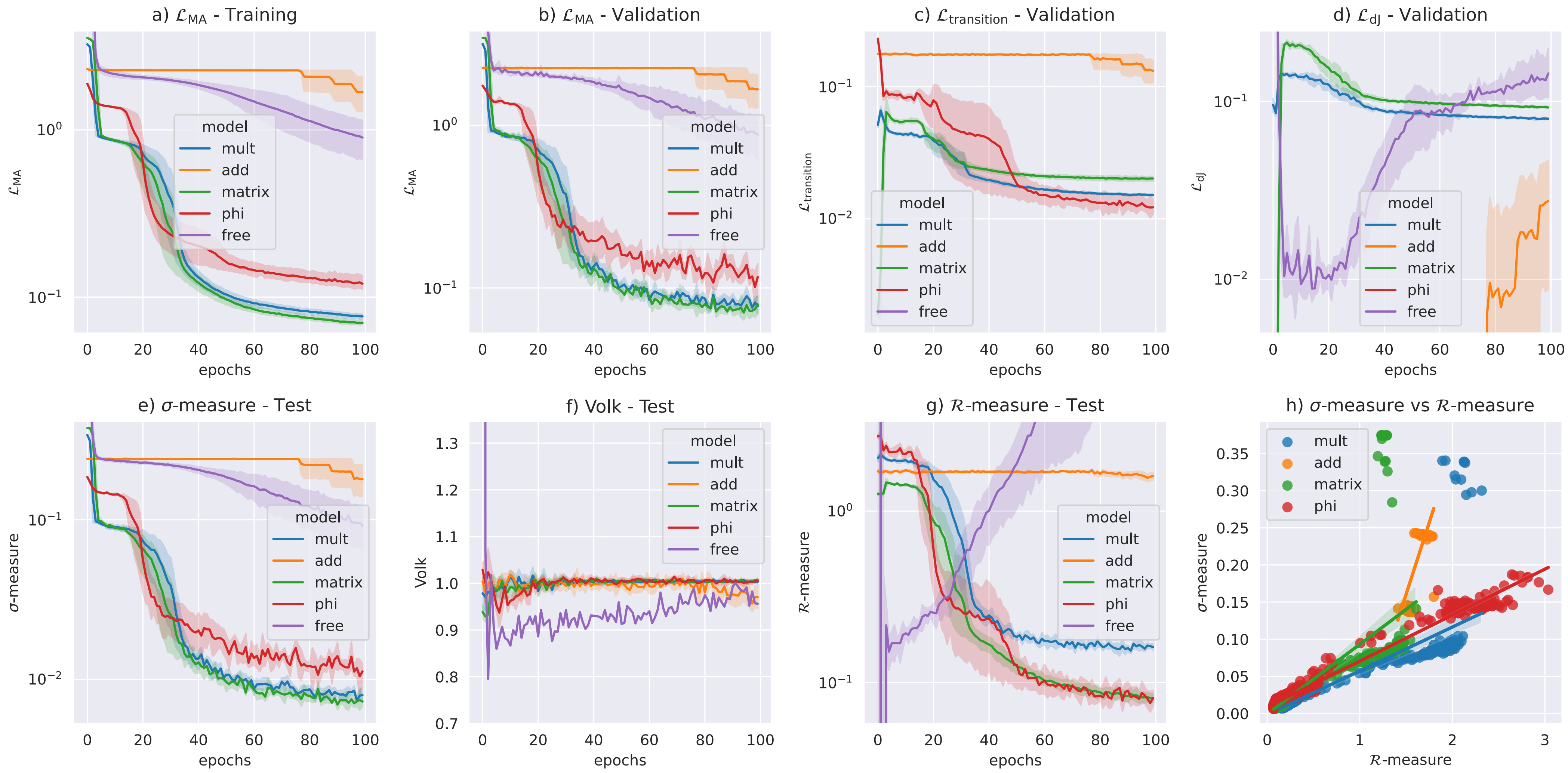
- Different Ansatze possible for metric prediction g_{pr}
  Encode more/less of math knowledge
- In the cymetric package, can choose between

Name	Ansatz
Free	$g_{\mathrm{pr}} = g_{\mathrm{NN}}$
Additive	$g_{\mathrm{pr}} = g_{\mathrm{FS}} + g_{\mathrm{NN}}$
Multiplicative, element-wise	$g_{\mathrm{pr}} = g_{\mathrm{FS}} + g_{\mathrm{FS}} \odot g_{\mathrm{NN}}$
Multiplicative, matrix	$g_{\mathrm{pr}} = g_{\mathrm{FS}} + g_{\mathrm{FS}} \cdot g_{\mathrm{NN}}$
ϕ-model	$g_{\mathrm{pr}} = g_{\mathrm{FS}} + \partial\bar{\partial}\phi$

Same as learning K

Example: direct learning of g

[Larfors et al 2205.13408]



Fermat quintic ,FF NN, fully connected, GELU, 64-64-64 network

More on cymetric package

the package I’ve worked most with ☺

- Most general:
  - point generators for CICYs and KS Cys
  - loss function for Kahler class preservation
- Makes most of ML (knows less math)
  - Agnostic about CY geometry apart from loss functions
  - PINN – physics constraints are enforced via custom loss functions

- Want other metric? Just replace some loss function!
- But be aware that all constraints are “soft”!

Summary ML implementations of CY metrics

Can use NN to model H matrix, Kahler potential or CY metric

In all cases, NN is very simple:

- Feed-forward, fully connected NNs with 2-3 layers
- Custom loss functions and/or activation functions encode physics
- Trainable on laptops, in minutes, for simple CYs (quintic)

Pro/con for architecture choices

Learning H or K

Pro

- Kähler
- Globally defined
- Donaldson's alg:
  convergence as $k \to \infty$

Con

- Scaling (of spectral basis)
- No generalization beyond Kähler

Learning metric

Pro

- Always learn 9 comps of 3*3 Hermitian metric
- Generalizes
  (e.g. non-Kähler SH metric)

Con

- Not Kähler
- Not globally defined

Architecture: further developments

- Benchmark study on cubic CY in $\mathbb{P}^2$ (a.k.a. the torus)
  Ahmed & Ruehle 2304.00027
  Found accuracy improves w larger training set, more nodes, longer training

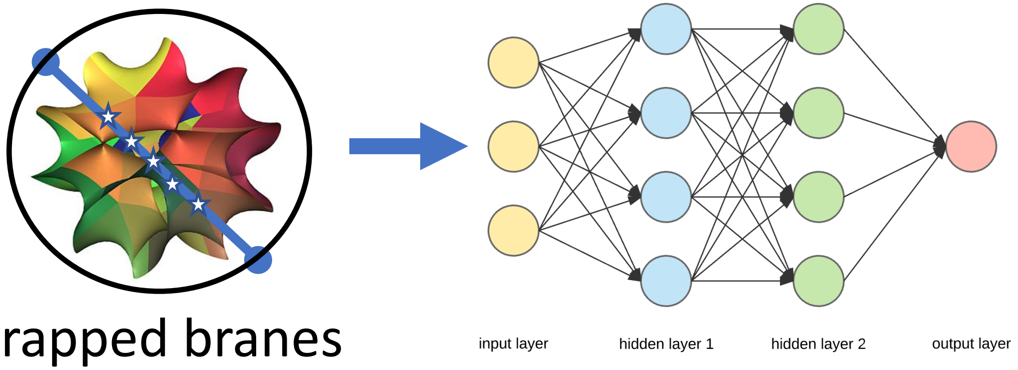
- Improve accuracy with architecture by going global
  - cymetric with spectral (=bihomogeneous) layer
    Berglund et al 2211.09801
  - Symmetries, equivariant NNs and Geometric Deep Learning
    in progress w Moritz Walden and Yacoub Hendi
- Improve accuracy (and training time) by going ultra-local
  Metric flows and Neural tangent kernel
  Halverson Ruehle 2310.19870

Summary of this lecture

- Simple NNs can learn Ricci flat CY metrics
- Mathematical constraints: encoded in NN or in loss functions
- ML packages for CY metrics

  applies to all CICY and Kreuzer-Skarke list
  at given point in moduli space
  architecture differs

- Lots of things (mostly) left to do:
  - Moduli-dependent CY metrics
  - Applications in physics:
    swampland conjectures, HYM equation, wrapped branes
  - Go beyond CY: G2 metrics, G-structure manifolds, ...



Plan for afternoon studies

- Reading:
    - R. Schneider "Heterotic Compactifications in the Era of Data Science", ch. 3, 5
      http://uu.diva-portal.org/smash/record.jsf?pid=diva2%3A1649343&dswid=-2157
    - L. Anderson, J. Gray and M. Larfors, arXiv:2312.17125
- Online tutorials:
    - Intro to CY metric optimization by Ed Hirst
      https://github.com/edhirst/OxfordCYTutorial/tree/main
    - ML of Ricciflat CY metrics by Yidi Qi
      https://colab.research.google.com/drive/1Z-jzToRkrTHayB83J0UKogbbBGO5Zdho?usp=sharing
    - Tutorials in the CYmetric, MLgeometry, cyJAX gitHub repo's

Additional slides

Point generators for KS CY manifolds - details

ML, Lukas, Ruehle, Schneider 2111.01436 & 2205.13408

- Can we relate ambient toric variety A to projective spaces?
  Yes!
  Use sections of line bundle dual to Kähler cone divisors;
  recall nef divisors are base-point free
- So Shiffman–Zelditch applies and quintic algorithm generalizes.

- Sections $s_j^{(\alpha)}$ of the toric Kähler cone generators $J_\alpha \sim$ coordinates of $\mathbb{P}^{r_\alpha}$
- Use Shiffman–Zelditch on $\mathbb{P}^{r_\alpha}$
- Express CY 3-fold as non-complete intersection in $\hat{A} \cong \bigotimes_{\alpha=1}^{h^{1,1}} \mathbb{P}^{r_\alpha}$
- Intersect $\rightsquigarrow$ sample of random points on CY distributed wrt FS measure.

cymetric: KS CY example

$h^{1,1} = 2, h^{2,1} = 80$ CY from the Kreuzer-Skarke list

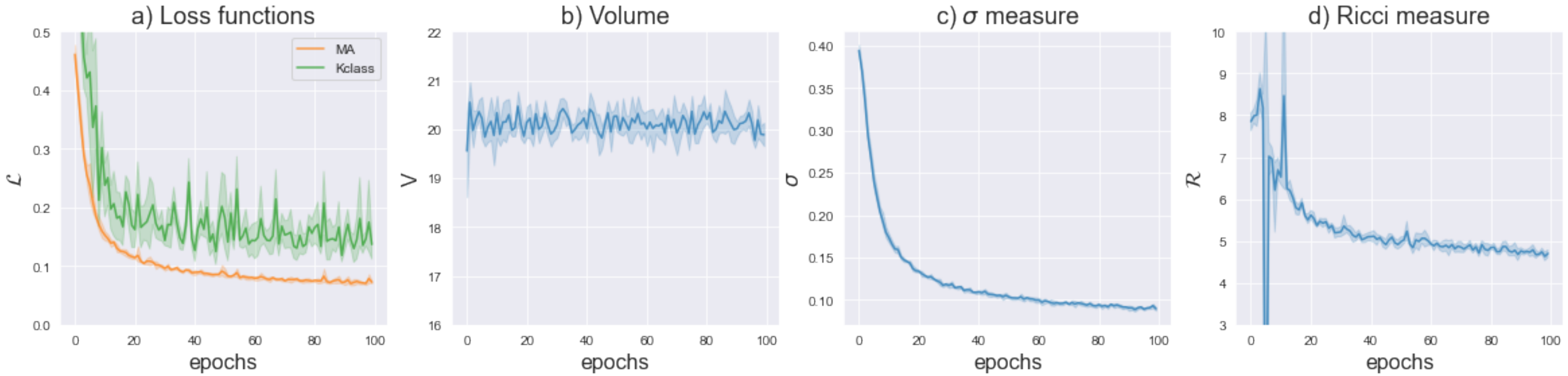
- Ambient space is $\mathbb{P}^1 \to \mathrm{A} \to \mathbb{P}^3$ w. toric coordinates $(x_0, \ldots, x_4)$

- CY hypersurface: $p(x_0, \ldots, x_4) = 0$ (80 terms; select randomly)
- 2 Kähler cone generators J_a ; $\quad J = t_1 J_1 + t_2 J_2$
- Morphisms to $\mathbb{P}^1$ and $\mathbb{P}^5$ using $H^0(J_a)$

- Point generation ~ 1 hour (generic cpl structure moduli, $t_a = 1$).

KS CY example

- $h^{1,1} = 2, h^{2,1} = 80$ Kreuzer-Skarke



- Toric ϕ-model, default loss, 200 000 points
- NN width 256, depth 3, GELU, batch (128, 10000), SGD w. momentum

Traditional methods

- Approximate K_{CY} via *algebraic expansion* in polynomial basis

$$K_k(z,\bar{z}) = \frac{1}{k}\sum \ln H_{a\bar{b}} p^a \bar{p}^{\bar{b}}$$

- Hermitian matrix H to be computed

<u>Donaldson algorithm</u>

- H_k : fixed point of iteration scheme
- Slow convergence at given k
- Proven $K \to K_{CY}$ as $k \to \infty$

<u>Energy functional</u>

- H_k : minimum of functional encoding MA equation
- Fast convergence at given k

Traditional methods

- Approximate K_{CY} via *algebraic expansion* in polynomial basis

$$K_k(z,\bar{z}) = \frac{1}{k}\sum \ln H_{a\bar{b}} p^a \bar{p}^{\bar{b}}$$

- Hermitian matrix H to be computed
- Problem: polynomial basis dim N_k grows with k, and $H \sim N_k^2$
  On quintic:

$$N_k = \begin{pmatrix} k+4 \\ k \end{pmatrix} - \begin{pmatrix} k-1 \\ k-5 \end{pmatrix} = 5, 15, 35, 70, 125, 205, 315, \ldots$$

- Use discrete symmetries to cut down N_k. Only works for some CYs