Learning the Size and Shape of Calabi-Yau Manifolds

Magdalena Larfors

Durham University & Uppsala University

Eurostrings 2022, Lyon

Based on work with Andre Lukas, Fabian Ruehle, Robin Schneider
(2111.01436 and in progress)

and `cymetric` package by Ruehle and Schneider:
`https://github.com/pythoncymetric/cymetric`

Does machine learning help string theory?

Yes. ML allows to

- Explore and analyse complex settings, such as the string theory landscape
- Explore, classify and approximate complex features
  e.g. for (Calabi-Yau) geometries relevant for string compactifications

Previous work

- Predicting topological properties of Calabi-Yau manifolds
  (supervised/unsupervised learning; faster than algebraic geometry methods)
- Approximate the Calabi-Yau metric
- Construct and study string derived semi-realistic standard models
  (e.g. reinforcement learning, genetic algorithms)
- Understanding string landscape: e.g. swampland conjectures.

Motivation: why compute the CY metric?

- Calabi-Yau manifolds are most common compactification spaces.
  Many examples:
  CICY 3-folds and 4-folds *Candelas-et.al:88, Greene-et.al:89, Haupt-et.al:14*
  CY hypersurfaces in 4D toric ambient spaces *Kreuzer–Skarke:00*

  Algebraic geometry $\rightsquigarrow$ topology known (in principle)
  But no closed form expression of CY metric (in dim ≥ 3).
- Couplings and masses in low-energy EFT depend on metric:
  e.g. set normalisation of Yukawa couplings so determine mass hierarchies...
- Can be used to test swampland conjectures, e.g. *Ashmore–Ruehle:21*.

This talk in summary:

Present development of package that can

- machine learn metrics on CY manifolds (focus on 3D)...
- with multiple complex and Kähler moduli ...
- from both CICY and Kreuzer-Skarke CY lists

Novel features and challenges (w.r.t previous work)

- multiple Kähler moduli $\rightsquigarrow$ new procedures to control Kähler class
- toric ambient space $\rightsquigarrow$ new procedures for point sampling

The cymetric package

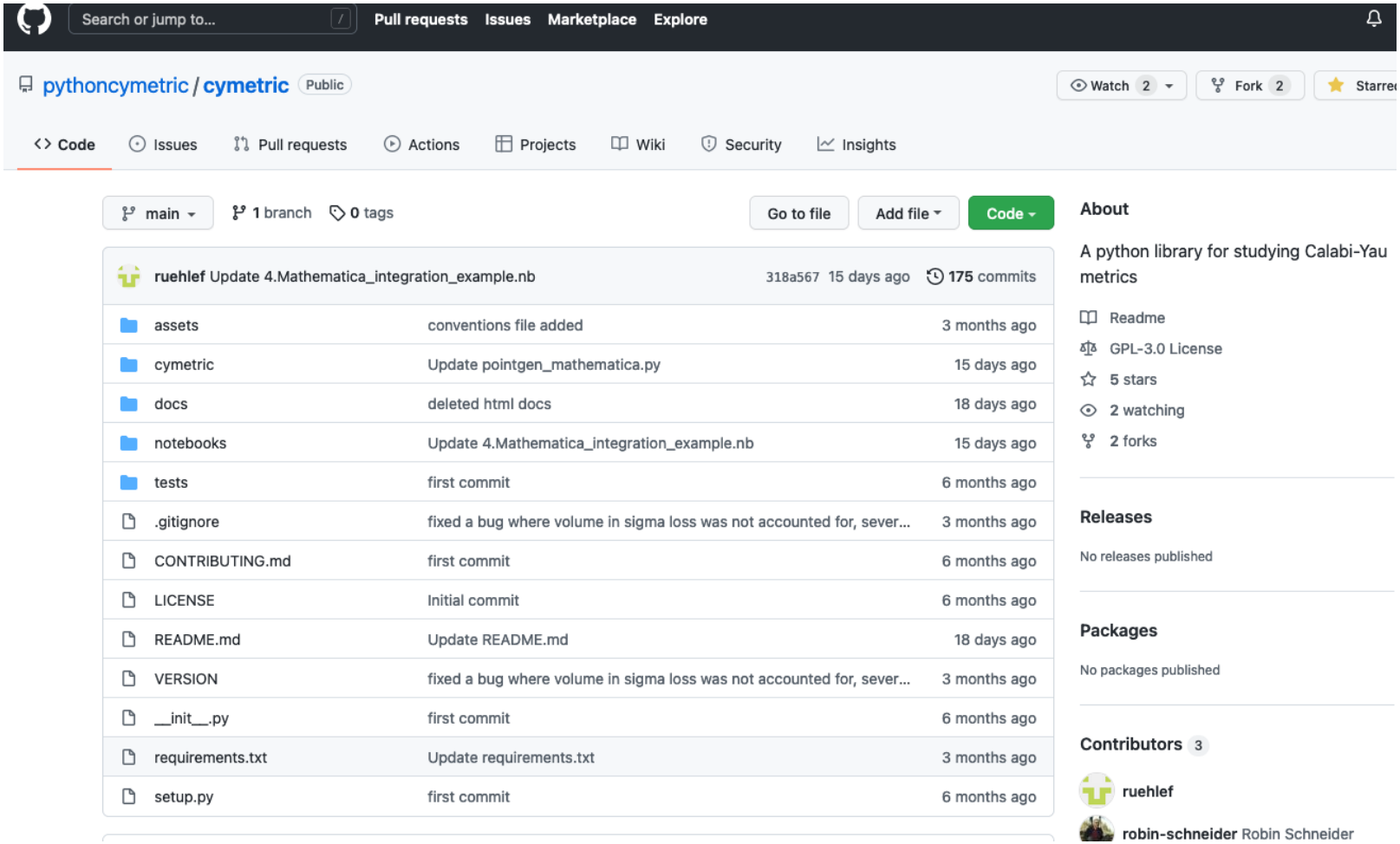
The package cymetric: low-threshold, open source

- python, Mathematica, Sage interface
- main functionality demonstrated in notebooks
- small experiments can be run on laptop (large on cluster)

  We hope this **adopts and adapts** to your favourite string theory problem – and contributions to the package are welcome

https://github.com/pythoncymetric/cymetric

The cymetric package



Outline

Calabi-Yau Manifolds

Calabi-Yau Theorem

- Let X be a (compact), complex, Kähler manifold
  - holomorphic top-form Ω
  - real, closed (1,1)-form $J \rightsquigarrow$ Kähler metric $g = \partial\bar{\partial}K$
- Exists unique $J_{CY} = J + \partial\bar{\partial}\phi$ such that corresponding metric g_{CY} is Ricci flat.
- This unique J_{CY} solves the Monge-Ampère equation

$$J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega} = \kappa\, \mathrm{d\ Vol}_{CY}$$

  where κ is some complex constant.
- CICY and KS CY:
  J and Ω can be computed analytically from ambient space data
- In general, no analytic expression is known for g_{CY} on compact CYs.

Calabi-Yau Manifolds

CY constructions

- CICY: complete intersection of hypersurfaces in $\mathcal{A} = \mathbb{P}^{n_1} \times \cdots \times \mathbb{P}^{n_h}$
  Specified by homogeneous polynomials p_r
- KS CY: hypersurface in 4D toric ambient space $\mathcal{A} \sim$ reflexive polytope.
  Specified by polynomial p.

CY data: Ω and J

- Ω is computed as a residue in ambient space coordinates

$$\text{CICY: } \Omega = \frac{dz_1 \wedge dz_2 \wedge dz_3}{\det(\partial p_r / \partial z^q)} \qquad \text{KS CY: } \Omega = \frac{dz_1 \wedge dz_2 \wedge dz_3}{\partial p / \partial z_4}$$

  This also gives the CY volume form d $\text{Vol}_{CY} = \Omega \wedge \bar{\Omega}$
- Constructing Kähler form J
  - Ambient spaces have Fubini–Study Kähler forms, e.g. for $\mathbb{P}^n$ $J_n = \frac{1}{2\pi} \sum |z_\mu^n|^2$
  - Restrict to CY using defining polynomials: basis J_α of $(1,1)$ forms
  - Fubini-Study Kähler form on CY given by $J_{\text{FS}} = t^\alpha J_\alpha$

  The corresponding g_{FS} is non-Ricci-flat, and J_{FS} does not satisfy the MA equation.

Calabi-Yau Manifolds

Unique Ricci flat metric g_{CY} given by J_{CY} that solves the MA equation

$$J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega} = \kappa\, \mathrm{d\ Vol}_{CY}$$

where κ is some complex constant.

Lacking analytic expression for g_{CY} (or J_{CY}), develop numerical approximations:

- 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
- Machine learning
  Ashmore–He–Ovrut:19, Douglas–Lakshminarasimhan–Qi:20,
  Anderson–Gerdes–Gray–Krippendorf–Raghuram–Ruehle:20, Jejjala–Mayorga–Peña:20 ,
  ML-Lukas-Ruehle-Schneider:21,..

Learning CY metrics with `cymetric`

`cymetric`

ML package `cymetric` decomposes into

1. point generators using Schiffman–Zelditch theorem
   CICYs: Apply algorithm by *Douglas et. al: 06, Braun et.al:08*.
   KS CYs: We generalise algorithm to toric ambient spaces.
2. custom TensorFlow neural networks that predict the CY metric
   (at given point in moduli space)

No ML in point generation; methods used also for other numerical methods.

`cymetric`: Point generators

Natural attempts (problematic)

- Give random values to all ambient coordinates, reject all points off the CY.
- Give random values to some ambient coordinates, use defining polynomials to solve for the rest.

Luckily, this problem has (almost completely) been solved before.

`cymetric`: Point generators

Creating a point sample on CICY 3-fold *Douglas-et.al:06, Braun-et.al:08*

Simplest case: Quintic hypersurfact in $\mathcal{A} = \mathbb{P}^4$

- Generate random points distributed w.r.t FS metric on $\mathbb{P}^4$
  use standard tools to pick uniformly distributed points on S^9, then rescale.
- Connect 2 such points by a 1-parameter line and intersect with CY hypersurface: 5 points on CY
- Theorem[Shiffman and Zelditch]: These points are distributed according to the FS measure:

$$dA = J_{\text{FS}} \wedge J_{\text{FS}} \wedge J_{\text{FS}} \, .$$

- This generalizes to other CICYs: Generate random points on $\mathbb{P}^r$ from S^{2r+1}, distribute free parameters equal to number of hypersurfaces and construct lines, surfaces, ... , then intersect with CY hypersurfaces.
- Implemented in `cymetric` as `CICYPointGenerator`

cymetric: Point generators

Creating a point sample on KS CY 3-fold, part 1

Can relate ambient toric variety $\mathcal{A}$ to projective spaces

$\Longrightarrow$ can apply Shiffman–Zelditch theorem, and generalise the CICY algorithm.

- The sections of the toric Kähler cone (generated by J_α) $\sim$ coordinates of $\mathbb{P}^r$

$$\Phi_\alpha : \quad [x_0 : x_1 : \ldots] \quad \rightarrow \quad [s_0^{(\alpha)} : s_1^{(\alpha)} : \ldots : s_{r_\alpha}^{(\alpha)}] \,,$$

- The FS metrics on $\mathbb{P}^r$ give a (non-FS) Kähler metric on $\mathcal{A}$.
- Can build random sections

$$S = \sum_{j=0}^{r_\alpha} a_j^{(\alpha)} s_j^\alpha$$

  using i.i.d. Gaussian coefficients $a_j^{(\alpha)} \sim \mathcal{N}(0,1)$
- Shiffman–Zelditch $\Longrightarrow$ Such S are distributed according to the FS measure.

cymetric: Point generators

Creating a point sample on KS CY 3-fold, part 2

Got map $\Phi_\alpha : \quad [x_0 : x_1 : \ldots] \quad \to \quad [s_0^{(\alpha)} : s_1^{(\alpha)} : \ldots : s_{r_\alpha}^{(\alpha)}]$ and can generate random point sample using sections $S = \sum_{j=0}^{r_\alpha} a_j^{(\alpha)} s_j^\alpha$.

- Now express the CY 3-fold in terms of Kähler cone sections $s_j^{(\alpha)}$
  - Problem 1: too many sections!
  - Problem 2: relations among sections
- Resolve problems: first find relations among sections ...
  - Groebner basis analysis using Singular (access via Sage)
  - Linear algebra routine works generically
- ... then combine relations + hypersurface eq:
  CY 3-fold as non-complete intersection in $\hat{\mathcal{A}} \cong \bigotimes_{\alpha=1}^{h^{1,1}} \mathbb{P}^{r_\alpha}$.
- End result: sample of random points on CY distributed wrt FS measure.

$$dA = \Phi_\alpha^*(J_{\mathrm{FS},\alpha}) \wedge \Phi_\beta^*(J_{\mathrm{FS},\beta}) \wedge \Phi_\gamma^*(J_{\mathrm{FS},\gamma}) \,.$$

- Implemented in `cymetric` as `ToricPointGeneratorMathematica`

`cymetric`: Point generators

Summary

Both for CICY and KS CY manifolds, `cymetric` point generators provide a sample of random points on CY distributed wrt known (FS) measure.

github notebooks illustrate how the point generators work in practice.

cymetric: Neural networks

Problem: find Ricci flat CY metric $g_{CY} \iff$ find J_{CY} that solves the MA eq.

$$J_{CY} \wedge J_{CY} \wedge J_{CY} = \kappa\, \Omega \wedge \bar{\Omega} = \kappa\, \mathrm{d}\ \mathrm{Vol}_{CY}$$

where κ is some complex constant.

Recall:

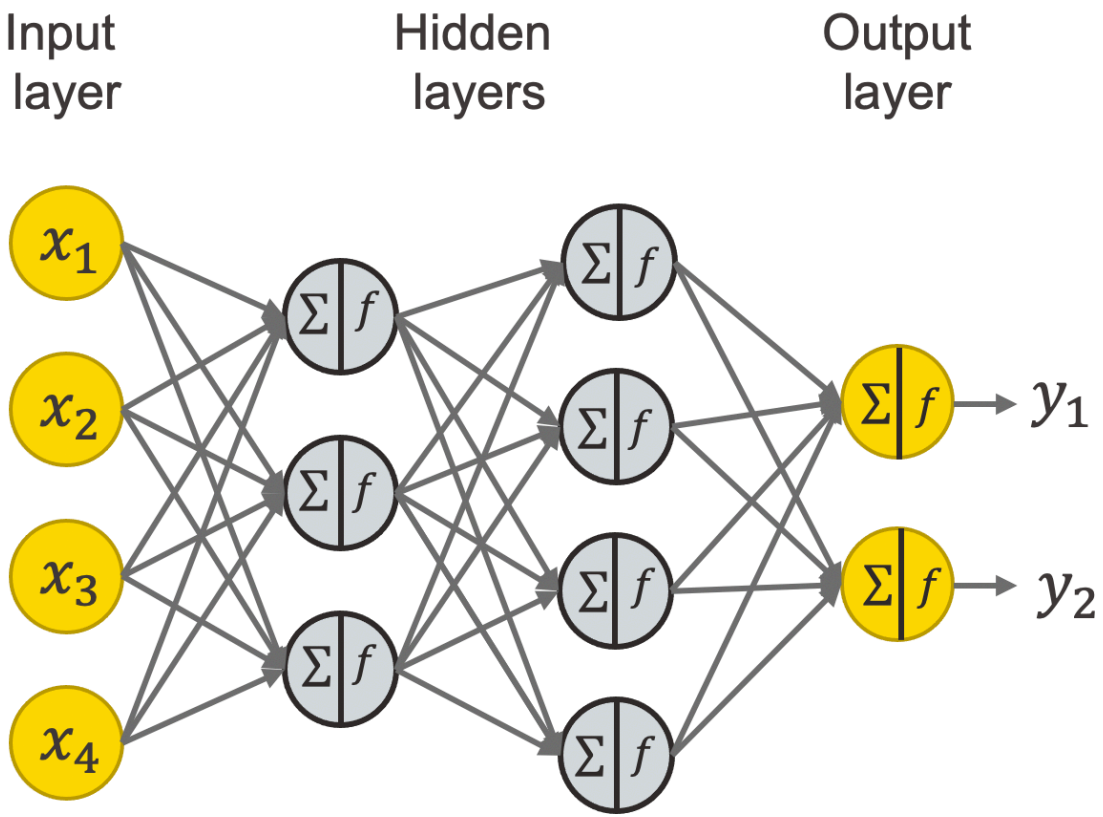
Know Ω and J_{FS} s.t. $[J_{CY}] \sim [J_{\mathrm{FS}}]$.

Have sample of points on CY randomly distributed w.r.t to known measure $\mathrm{d}A$.

Plan:

Input points into NN that we train and test using our physics knowledge.

cymetric: Neural networks



NN training:
parameters updated via stochastic gradient descent to minimise loss functions.
User controls network hyperparameters
(width, depth, batch size, activation function, learning rate, etc)

cymetric: Neural networks

Input and performance

Input: N points p_i, randomly distributed w.r.t to known measure $\mathrm{d}A$ on CY.

After training, measure performance: does the MA equation hold? is the metric Ricci flat?

Check via established benchmarks:

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

where we use random point sample to Monte Carlo integrate any function f

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

`cymetric`: Neural networks

`cymetric` models: Different Ansätze for the NN prediction

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

Learning CY metrics with `cymetric`

Custom loss terms controls learning - user chooses α_i

$$\mathcal{L} = \alpha_1 \mathcal{L}_{\text{MA}} + \alpha_2 \mathcal{L}_{\text{dJ}} + \alpha_3 \mathcal{L}_{\text{transition}} + \alpha_4 \mathcal{L}_{\text{Ricci}} + \alpha_5 \mathcal{L}_{\text{K-class}}.$$

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

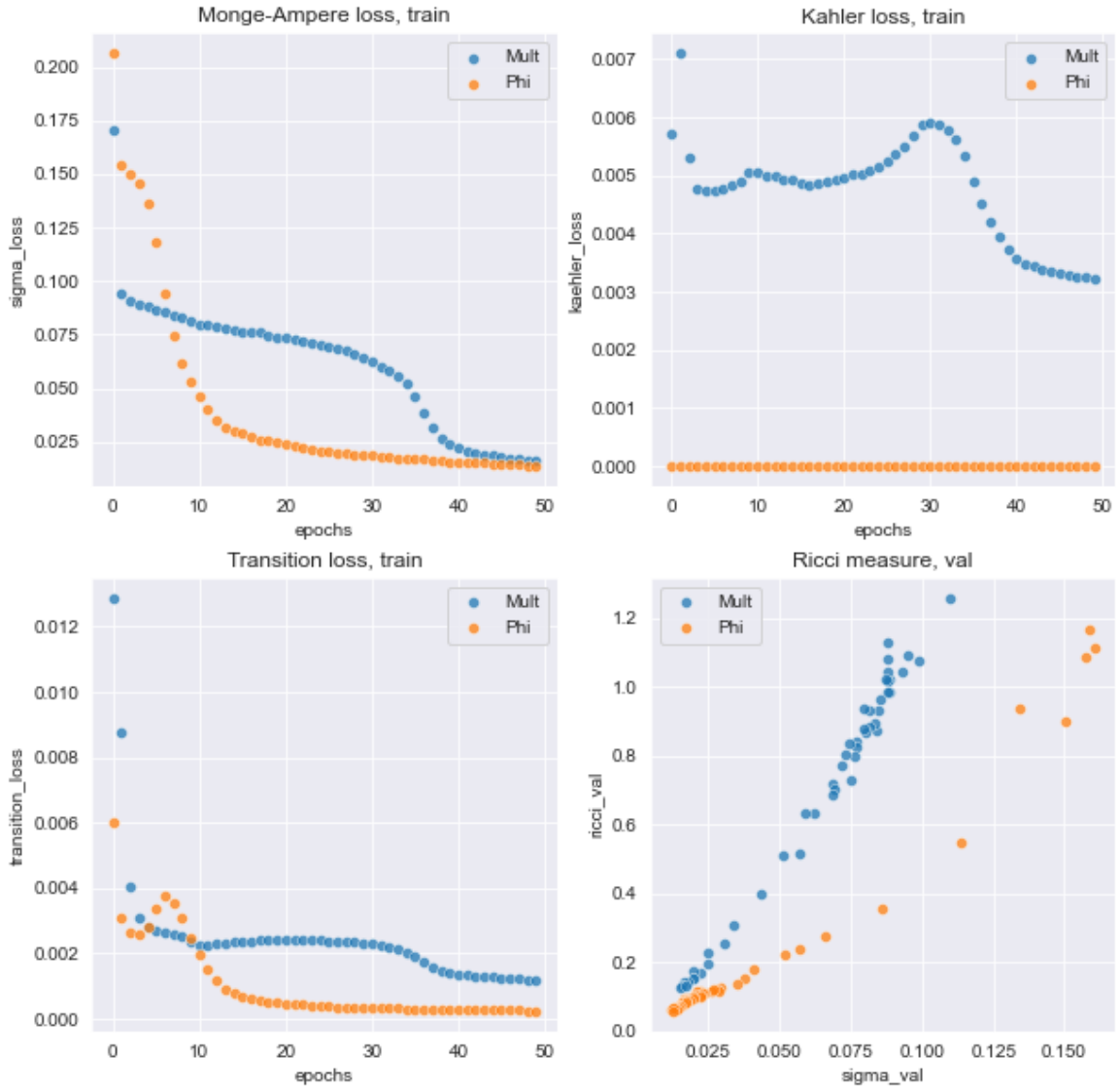
$$\mathcal{L}_{\text{dJ}} = \sum_{ijk} ||\Re c_{ijk}||_n + ||\Im c_{ijk}||_n , \quad \text{with } c_{ijk} = g_{i\bar{j},k} - g_{k\bar{j},i} \quad \text{and } g_{i\bar{j},k} = \partial_k g_{i\bar{j}}$$

$$\mathcal{L}_{\text{transition}} = \frac{1}{d} \sum_{(s,t)} \left|\left| g_{\text{pr}}^{(t)} - T_{(s,t)} \cdot g_{\text{pr}}^{(s)} \cdot T_{(s,t)}^{\dagger} \right|\right|_n$$

$$\mathcal{L}_{\text{Ricci}} = ||R||_n = \left|\left| \partial\bar{\partial} \ln \det g_{\text{pr}} \right|\right|_n ,$$

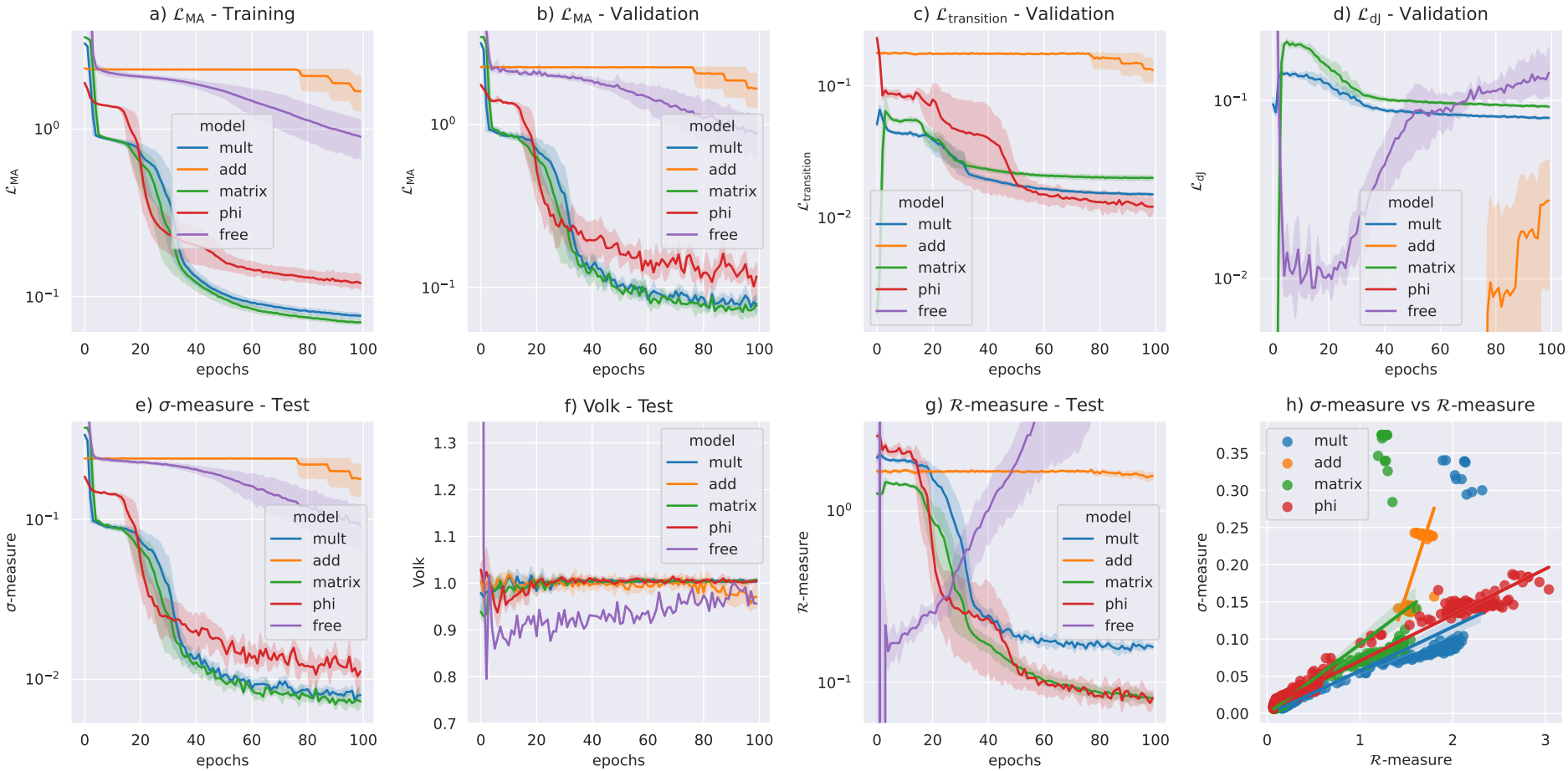
$$\mathcal{L}_{\text{K-class}} = \frac{1}{h^{11}} \sum_{i=1}^{h^{11}} \left|\left| \mu_{J_{\text{FS}}}(\mathcal{L}_i) - \int_X (J_{\text{pr}})^{n-1} \mathcal{F}_i \right|\right|_n .$$

Experiments: Fermat Quintic



Train NN for 50 epochs (width 64, depth 3, GELU activation functions, batch size of 64, learning rate of 1/1000, all $\alpha = 1$); on laptop about 1.5 hours/model.

Experiments: Fermat Quintic



Multiple Kähler moduli: preserving the Kähler class

The Kähler class $[J]$ is fixed by

$$\int_C J = \mathrm{vol}(C)\ , \quad \int_D J \wedge J = \mathrm{vol}(D)\ , \quad \int_X J \wedge J \wedge J = \mathrm{vol}(X)\ ,$$

Are we guaranteed this is fixed during training?

- Yes, if there is only one Kähler class (e.g. Quintic)
- Yes, if metric Ansatz ensures correction to J is exact
- Yes, if NN trained with Kähler class loss function $\mathcal{L}_{\text{K-class}}$ (works for all metric Ansätze).

Multiple Kähler moduli: preserving the Kähler class

Loss function preserving the Kähler class

- Could define a loss function fixing curve, divisor and CY volumes (but have not; this requires sampling points on curves and divisors).
- Instead use that $\mathcal{O}_X(k)$ (line bundle over X with $c_1 = [k^\alpha J_\alpha]$) has slope

$$\mu_J := \int_X J \wedge J \wedge c_1(\mathcal{O}_X(k)) = -\frac{i}{2\pi} \int_X J \wedge J \wedge F = d_{\alpha\beta\gamma} t^\alpha t^\beta k^\gamma \ ,$$

The slope is topological, so agrees for metrics in the same Kähler class!

- Loss function: for h^{11}-dim basis of line bundles with $k^1 = (1, 0, 0, ...)$ etc. compute

$$\mathcal{L}_{\text{K-class}} = \frac{1}{h^{11}} \sum_{i=1}^{h^{11}} \left|\left| \mu_{J_{\text{FS}}}(L_i) - \int_X J_{\text{pr}} \wedge J_{\text{pr}} \wedge F_i \right|\right|_n$$

- Requires more points than contained in mini-batch; NN code more involved.
- Cross-check after training: compute volume and line bundle slopes from intersection numbers, from FS metric and from CY metric.

Example: Bicubic

The bicubic is given by a homogeneous degree (3,3) polynomial in $\mathcal{A} = \mathbb{P}^2 \times \mathbb{P}^2$.
It has 2 Kähler moduli and 83 complex structure moduli.
We choose the complex structure moduli, i.e. specify the (3,3) polynomial.

Choose several Kähler moduli paired with line bundles of vanishing slope:

case	1	2	3	4	5
t	$\begin{pmatrix} 1.414 \\ 1.414 \end{pmatrix}$	$\begin{pmatrix} 0.687 \\ 1.878 \end{pmatrix}$	$\begin{pmatrix} 0.421 \\ 1.955 \end{pmatrix}$	$\begin{pmatrix} 0.299 \\ 1.977 \end{pmatrix}$	$\begin{pmatrix} 0.962 \\ 1.753 \end{pmatrix}$
$\mathcal{O}_X(k)$	$\mathcal{O}_X(1,-1)$	$\mathcal{O}_X(1,-2)$	$\mathcal{O}_X(1,-3)$	$\mathcal{O}_X(1,-4)$	$\mathcal{O}_X(2,-3)$

`cymetric` point generation and training

- Generate 100 000 points for each choice of Kähler parameters
- Train ϕ-model for 100 epochs (width 64, depth 3, GELU activation functions, batch size of 64, learning rate of 1/1000).
  Training has been carried out on a single CPU in about two hours.

Example: Bicubic

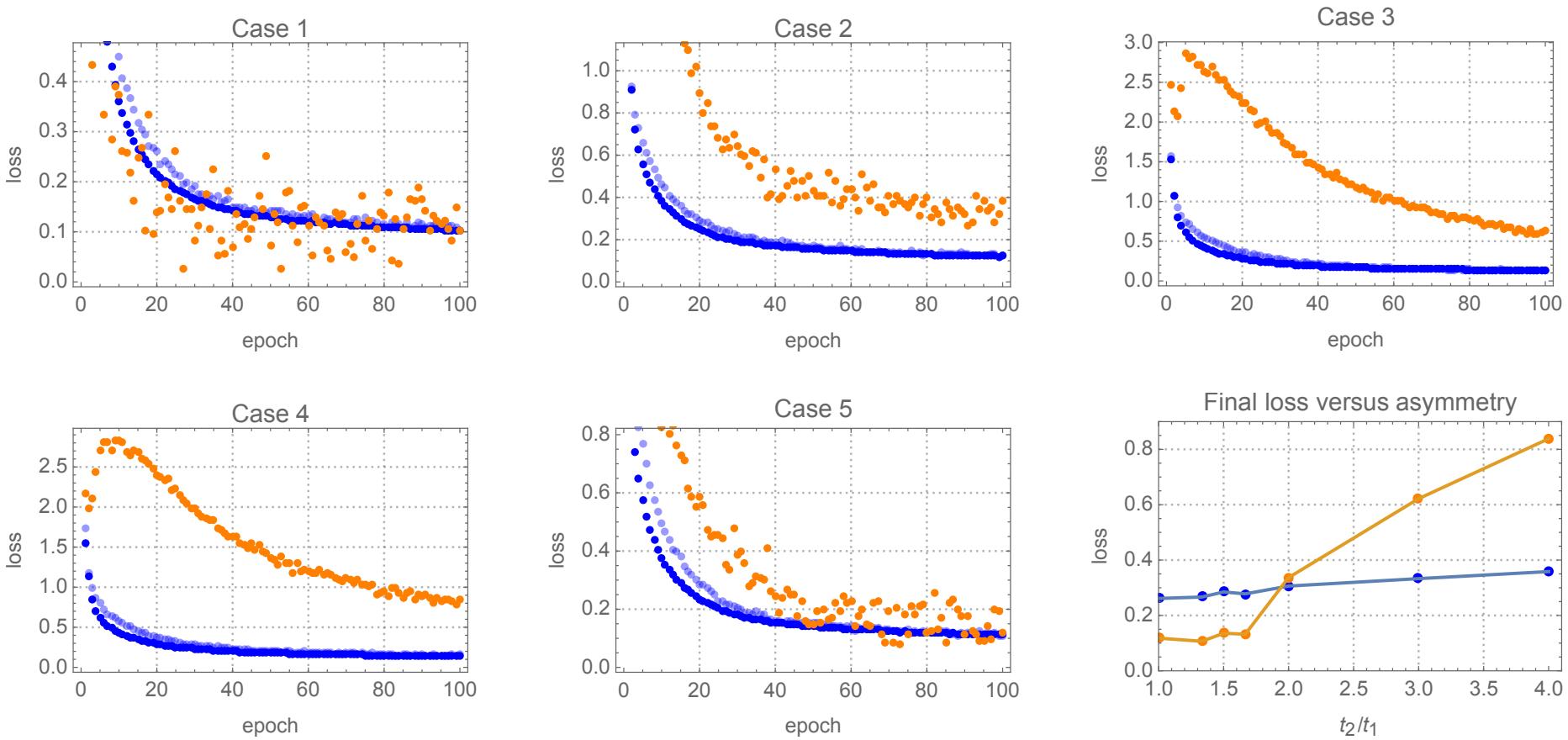


Figure: Bi-cubic training curves for several choices of Kähler parameters. The last plot represents the final loss, obtained by averaging over the last 10 epochs, as a function of t^2/t^1. (Training data, blue: $4\times$ sigma loss, orange: Kähler class loss).

Bicubic: checking the Kähler class

After training can compute the slope using the intersection numbers, FS and CY metric for all choices of Kähler moduli. They agree.

We also check that the volumes agree (*not* used in Kähler class loss function)

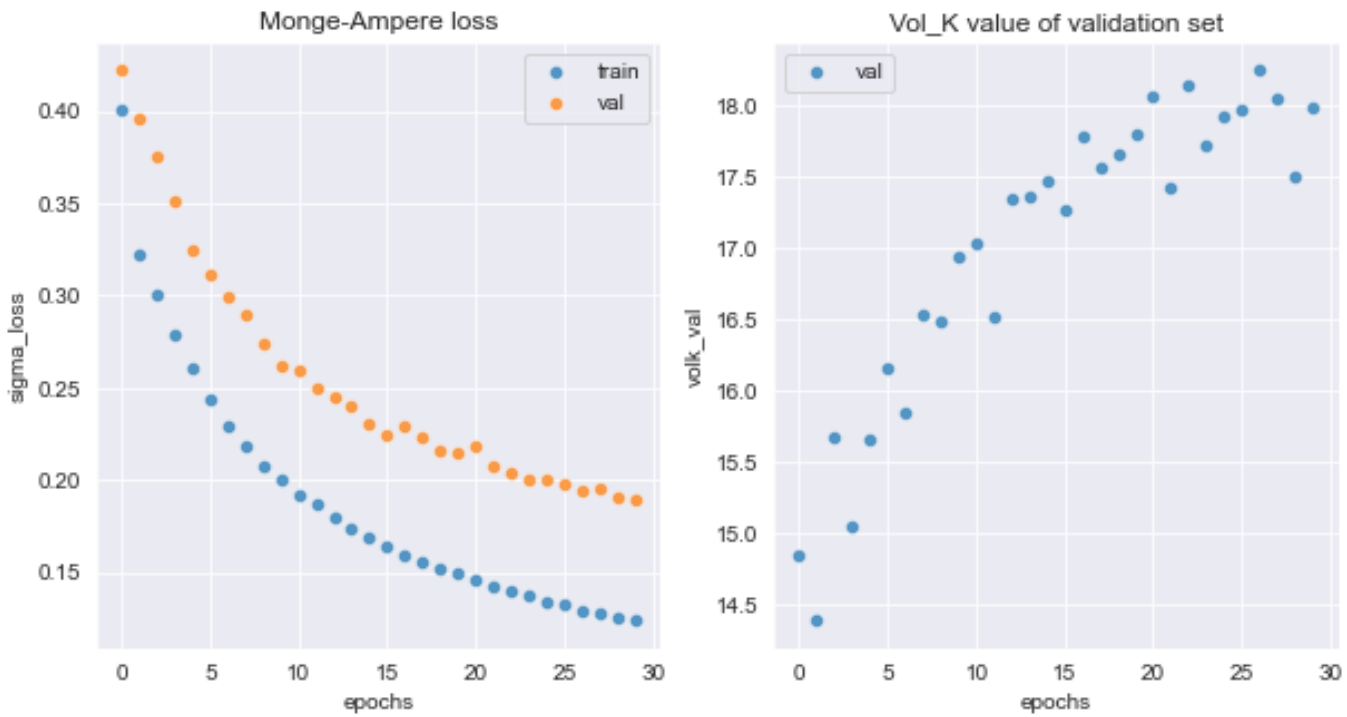
case	1	2	3	4	5	6	7
V_{exact}	8.49	4.97	2.93	2.02	6.87	7.59	6.16
V_{FS}	8.49	4.98	2.9	1.93	6.81	7.59	6.12
error	$< 1\%$	$< 1\%$	$\sim 1\%$	$\sim 4\%$	$\sim 1\%$	$< 1\%$	$< 1\%$
V_{CY}	8.57	5.03	2.91	1.93	6.88	7.63	6.08
error	$\sim 1\%$	$\sim 1\%$	$< 1\%$	$\sim 4\%$	$< 1\%$	$< 1\%$	$\sim 1\%$

Note V_{FS} is a check of the point sampling; better accuracy requires more sampling points.

Example: $h^{11} = 2$ Kreuzer–Skarke CY

Short KS run: train toric ϕ-model on 50 000 points for 30 epochs (width 64, depth 3, GELU activation functions, batch size of 64, learning rate of 1/1000).
Point generation: about 1 hour, training: about three hours (single CPU).

MA loss and volume (exact 20)



Conclusions and Outlook

This talk in summary:

Present development of package that can

- machine learn metrics on CY manifolds (focus on 3D)...
- with multiple complex and Kähler moduli ...
- from both CICY and Kreuzer-Skarke CY lists

Novel features and challenges (w.r.t previous work)

- multiple Kähler moduli $\rightsquigarrow$ new procedures to control Kähler class
- toric ambient space $\rightsquigarrow$ new procedures for point sampling

Conclusions and Outlook

Outlook

- Improve ML side: architecture and hyperparameter tuning for better performance.
- Test other point sampling routines (MCMC methods?).
- (Kähler) moduli dependence of metric?
  e.g. cpl structure moduli on Quintic *Anderson-et.al:20, Ashmore-Ruehle:21*.
- Solve Hermitian-Yang Mills equation *(in progress)*.
- Submanifolds and branes: calibrated/SUSY, non-calibrated/non-SUSY.
- Compactifications with flux e.g. Strominger-Hull system.
  just need other loss functions. *Larfors-Lukas-Ruehle:18, Anderson-et.al:20*.
- K3 metrics, compare with analytic metric.
  Kachru–Tripathy–Zimet:18,20; Jejjala-et.al:20
- What about other special holonomy manifolds?

Conclusions and Outlook

Results

- Machine learning provides computationally efficient approximations to CY metrics.
- The package `cymetric` provides the functionalities needed for any CY (in principle).
- It's time to use it for physics!

Thank you for listening!