September 23, 2025

Machine Learning the Conformal Manifold of Holographic CFT2's

Bastien Duboeuf[1], Camille Eloy[1] and Gabriel Larios[2]

[1] *ENS de Lyon, CNRS, LPENSL, UMR5672, 69342, Lyon cedex 07, France*

[2] *Mitchell Institute for Fundamental Physics and Astronomy, Texas A&M University, College Station, TX, 77843, USA*

Abstract

We investigate the structure of conformal manifolds around $\mathrm{AdS}_3 \times S^3$ which lift from continuous flat directions in the scalar potential of gauged supergravity resulting from six-dimensional $\mathcal{N} = (1, 1)$. Our approach combines numerical exploration and symbolic inference. For the latter, we develop a symbolic regression algorithm based on Annealed Importance Sampling (AIS) combined with Sequential Monte Carlo (SMC), well-suited to uncovering polynomial constraints in high-dimensional parameter spaces. The algorithm reconstructs a set of polynomial relations that provides an explicit analytic parametrization of a new family of solutions.

Contents

1 Introduction **1**

2 Supergravity Setup **3**

3 Annealed Importance Sampling for Polynomial Symbolic Regression **5**

4 Annealed Importance Sampling for Polynomial Symbolic Regression **8**
4.1 General idea 8
4.2 Detailed procedure 9

5 Numerical Analysis and Results **12**
5.1 Sampling the manifold: gradient descent 12
5.2 Local analysis: extracting the dimension 15
5.3 AIS-SMC algorithm and results 16
5.3.1 Initialization of the algorithm 17
5.3.2 Analysis of a typical run 17
5.3.3 Statistics 19

6 Supergravity Solutions **21**

7 Conclusion **23**

A Some Details on the Supergravity Setup **24**

B Comparison with AIFeynman **25**

1 Introduction

The AdS/CFT correspondence stands as one of the most profound dualities in theoretical physics, establishing a remarkable equivalence between string theory solutions on anti-de Sitter (AdS) spacetimes and conformal field theories (CFTs) living on their boundaries [1]. This correspondence has revolutionized our understanding of both quantum gravity and strongly coupled field theories, providing unprecedented insights into the holographic nature of gravity. Within this holographic framework, supergravity theories in AdS backgrounds serve as the low-energy effective descriptions of string theory compactifications, making them natural laboratories for exploring the gravitational side of the duality. The CFT side plays a crucial role in statistical physics, as CFT are known to describe phase transitions ; they are also fixed point of Renormalisation Group flow and this is this latter aspect that is our interest in this paper. An interesting question is indeed to know whether those CFT are isolated fixed points of RG flows, or belong to a continuous family of CFT. If this is the case, the space of continuous deformations that takes from one CFT to another is called a conformal manifold. These deformations are parametrised by exactly marginal operators, or in other words operators that preserves the conformal symmetry, *i.e.* whose β functions exactly vanish.

From the AdS/CFT correspondence perspective, conformal manifolds on the boundary theory are dual to continuous families of AdS solutions where the (possibly warped) AdS factor stays undeformed. If a consistent truncation on these solutions exists, conformal manifolds can also be identified as flat directions in the scalar potential of the truncation. Along these directions, the scalar field configurations vary continuously while the cosmological constant remains unchanged. Supersymmetry is believed to be

necessary for the existence of holographic conformal manifolds, as non-supersymmetric AdS solutions are expected to be unstable [2, 3]. However, recent investigations have identified AdS_4 configurations that appear to evade this requirement, with no evidence of standard decay channels – neither perturbative nor non-perturbative – being present [4]. The situation is even richer in the context of AdS_3/CFT_2, as current-current deformations, given by products $J\bar{J}$ of (anti-)holomorphic conserved currents, are exactly marginal in two dimensions [5], although they may break supersymmetry. The question of the exact form of the gravity dual to $J\bar{J}$ deformations is however open. A well-known example is given in ref. [6, 7], and recent works have demonstrated the existence of vast families of classically marginal deformations of the $AdS_3 \times S^3 \times T^4$ and $AdS_3 \times S^3 \times S^3 \times S^1$ solutions of type IIB supergravity that are perturbatively stable despite supersymmetry breaking [8–10]. These deformations were shown in ref. [10] to be equivalent to current-current deformations of the worldsheet Wess-Zumino-Witten models [11] describing the undeformed solutions [12, 13], giving strong indications that their CFT duals arise from $J\bar{J}$ deformations as well.

Although extensive, the families of deformations found in the aforementioned papers have not exhausted all possibilities for marginal deformations of three-dimensional supergravity solutions. A complete classification would require a systematic study of the flat directions in the supergravity scalar potential V that defines the AdS configurations, as those directions correspond to classical marginal deformations of the holographic CFT in the large N limit. These flat directions are continuous sets in the space of scalar fields along which the value of the potential stay fixed, *i.e.* along which $\nabla V = 0$. However, the explicit characterisation of flat directions presents formidable technical challenges. Supergravity scalar potentials, even in truncated models, typically involve dozens of scalar fields with intricate non-linear interactions. The resulting expressions for critical points – where all first-order derivatives vanish – quickly become too complex for traditional symbolic manipulation, rendering analytical approaches computationally intractable.

The emergence of machine learning techniques opens new avenues for addressing such complex problems in theoretical physics. Instead of solving the full symbolic system analytically from the outset, one can employ numerical methods to sample the solution space and subsequently apply symbolic regression techniques to extract analytical patterns from the data. This hybrid approach has the potential to bypass the computational bottlenecks inherent to purely symbolic methods, while still having the potential to uncover exact analytical expressions.

Machine learning strategies have previously been applied to identify new isolated vacua in SO(8) supergravity [14, 15]. More broadly, there has been increasing interest in applying machine learning and numerical techniques across various domains of high-energy physics. This includes, for instance, the characterisation of Calabi–Yau metrics and hypersurfaces [16–23], as well as broader efforts to explore the string theory landscape [24–26]. Additional applications include studies in CFT [27], investigations of the supergravity landscape [28, 29], and explorations of the AdS/CFT correspondence [30]. More generally, machine learning has found utility in the study of string theory, geometry, and fundamental physics [31–33].

In this work, we demonstrate the viability of the aforementioned machine learning approach by applying it to a five-scalar subsector of a 13 scalar consistent truncation of six-dimensional non-chiral $\mathcal{N} = (1, 1)$ supergravity on $AdS_3 \times S^3$, or to type IIB supergravity on $AdS_3 \times S^3 \times T^4$. Our methodology combines gradient descent sampling of the conformal manifold with a symbolic regression technique. There exists a large body of literature on symbolic regression, using methods from genetic programming [34–37], to deep learning [38, 39], generative models [40], diffusion models [41], and equation learning with the nodes being symbolic operations [42]. A state-of-the-art algorithm is AIFeynman [43]. It uses neural networks to identify structures in the dataset (such as translational symmetries, multiplicative separability, compositionality...) to recursively define simpler problems on which they can fit the solutions with polynomials and basic functions. However, this method is

slow, usually limited to low-dimensional spaces, and, by construction, can only fit one expression at a time. Alternatively, we develop here a symbolic regression algorithm based on Annealing Importance Sampling (AIS) [44], since this approach is much better suited to the study of geometric loci defined by intersecting polynomials in high-dimensional spaces

The study of the five-parameter potential is separated into several parts. We first use a gradient descent to efficiently sample the underlying conformal manifold. Combined with numerical analyses, including a local principal component analysis, we can identify the existence of a three-dimensional continuous family. We then use an AIS technique combined with a sequential Monte Carlo approach (SMC) [45] in order to do symbolic regression. As we will indeed demonstrate in the paper, there exist polynomial constraints on the 5 parameters, viewed as embedding coordinates, that project them onto the 3 dimensional manifolds. We manage to identify 8 of those constraints on the data, not all independent, which once solved provide an explicit three-dimensional family of solutions.

The paper is organized as follows. In sec. 2, we establish the supergravity setup, presenting the scalar potential in its full complexity and motivating the restriction to five fields. Sec. 4 details our annealed importance sampling approach to symbolic regression. Our main results, both numerical an analytical, are presented in sec. 5, and sec. 6 gives some details on the new supergravity solutions discovered by the numerical analysis. We conclude with prospects for extending this approach to higher-dimensional cases and its broader implications for systematic studies of conformal manifolds in holographic theories.

2 Supergravity Setup

Three-dimensional $\mathcal{N} = 8$ (half-maximal) gauged supergravity is governed by the Lagrangian [46, 47]

$$e^{-1}\mathcal{L} = R + \frac{1}{8} g^{\mu\nu} D_\mu M^{\bar{M}\bar{N}} D_\nu M_{\bar{M}\bar{N}} + e^{-1}\mathcal{L}_{\text{CS}} - V, \tag{2.1}$$

which comprises an Einstein-Hilbert term R, a kinetic term for scalar fields parametrised by the matrix $M_{\bar{M}\bar{N}}$, a Chern-Simons contribution $\mathcal{L}_{\text{CS}}$, and a scalar potential V, see app. A for details on the definition of the terms. The scalar degrees of freedom parametrise the coset space

$$\frac{\text{SO}(8,4)}{\text{SO}(8) \times \text{SO}(4)}, \tag{2.2}$$

through the symmetric matrix $M_{\bar{K}\bar{L}}$.[1] With appropriate gauging (see eq. (A.9)), this SO(8, 4) theory is a consistent truncation of six-dimensional $\mathcal{N} = (1, 1)$ supergravity on S^3 [48–50]. It features an AdS_3 stationary point at the scalar origin ($M_{\bar{K}\bar{L}} = \delta_{\bar{K}\bar{L}}$, the SO(8, 4) identity matrix), corresponding to an $\text{AdS}_3 \times S^3$ solution in 6d. In the following, we will be interested in exploring flat directions of the potential V around this point, *i.e.* stationary points that are continuously connected to the origin. In three dimensions, the flat directions constitute a family of AdS_3 vacua sharing the same cosmological constant. The corresponding solutions in six dimensions are of the form $\text{AdS}_3 \times M^3$, with M^3 some deformation of the round sphere S^3, parametrised by the constant scalar vevs defining the 3d vacua.

We parametrise the scalars of the theory following ref. [51] (see also app. A for some details): 13

[1] In full generality, the coset space of half-maximal supergravity in three dimensions is $\text{SO}(4+p, 4+q)/\big(\text{SO}(4+p) \times \text{SO}(4+q)\big)$ for p vectors and q tensors in 6d. We consider here the case $p = 4$ only.

scalars parametrised by a symmetric $\mathrm{GL}(3,\mathbb{R})$ matrix $m = \nu\nu^T$ where

$$\nu = e^{(6\,\tilde{x}_7+3\,\tilde{x}_8+\sqrt{3}\,\tilde{x}_9)/6} \begin{pmatrix} 1 & \frac{x_{10}}{\sqrt{2}} & \frac{x_{11}}{\sqrt{2}} + \frac{x_{10}x_{12}}{4} \\ 0 & e^{-\tilde{x}_8} & \frac{e^{-\tilde{x}_8}\,x_{12}}{\sqrt{2}} \\ 0 & 0 & e^{-(\tilde{x}_8+\sqrt{3}\,\tilde{x}_9)/2} \end{pmatrix}, \tag{2.3}$$

the matrices

$$\phi = \begin{pmatrix} 0 & x_4 & x_5 \\ -x_4 & 0 & x_6 \\ -x_5 & -x_6 & 0 \end{pmatrix}, \quad \xi = \begin{pmatrix} 0 & 0 & 0 & x_1 \\ 0 & 0 & 0 & x_2 \\ 0 & 0 & 0 & x_3 \end{pmatrix}, \quad \xi^2 = \xi\xi^T, \tag{2.4}$$

and a dilaton $\tilde{\varphi} = \tilde{x}_{13}$. With this parametrization, the potential takes the form:

$$\begin{aligned} V = 4\,e^{-4\tilde{\varphi}} + 2\,e^{-2\tilde{\varphi}}\Big[& -\mathrm{tr}\left(m+m^{-1}\right) + \mathrm{tr}\left(\phi m^{-1}\phi\right) - 2\ \mathrm{tr}\left(\phi m^{-1}\xi^2\right) - 2\ \mathrm{tr}\left(\xi^2\right) \\ & - \mathrm{tr}\left(\xi^2 m^{-1}\xi^2\right) + \frac{1}{2}\ \det\left(m^{-1}\right)\left(1 - \mathrm{tr}\left(\phi^2\right) - \mathrm{tr}\left(\xi^4\right) + \mathrm{tr}\left(\xi^2\right)^2\right) \\ & + \frac{1}{2}\,\mathrm{T}\left(m^{-1}(\xi^2-\phi), (\xi^2+\phi)m^{-1}, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2\right) \\ & + \frac{1}{4}\,\mathrm{T}\left(m^{-1}, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2\right)\Big], \end{aligned} \tag{2.5}$$

where $\mathrm{T}\,(A,B,C) = \varepsilon_{mnp}\,\varepsilon_{qrs}\,A^{mq}B^{nr}C^{ps}$. For later convenience, we define a thirteen-dimensional vector

$$\vec{X} = (x_1, x_2, x_3, x_4, x_5, x_6, \tilde{x}_7, \tilde{x}_8, \tilde{x}_9, x_{10}, x_{11}, x_{12}, \tilde{x}_{13}), \tag{2.6}$$

encompassing all parameters. Note here that all dilaton fields are denoted with a tilde. This notation is adopted in anticipation of a later redefinition of the form $x_i = e^{\tilde{x}_i}$ for these fields.

The search for flat directions of the potential (2.5) can then start form a study of its gradient ∇V. However, carrying out the search for stationary points analytically is by far too complex, even reducing the number of variables or using a symbolic solver such as Mathematica [52]: the resulting expressions are too convoluted to be simplified into a manageable form, and do not yield usable relationships that express some variables in terms of others. This complexity does not, however, rule out the possibility that simpler solutions satisfying the condition of vanishing gradient may exist although the solver does not find them.

In this work, we aim to identify such solutions. A preliminary numerical exploration suggests that attention can be focused on the five scalars $x_1, x_2, x_4, \tilde{x}_8$, and x_{10}. To ensure that the solutions we obtain in this truncation remain valid solutions of the complete theory, we first compute ∇V with all scalar fields included, *i.e.* using eq. (2.5), and only then setting the remaining fields,

$$\vec{y} = (x_3, x_5, x_6, \tilde{x}_7, \tilde{x}_9, x_{11}, x_{12}, \tilde{x}_{13}), \tag{2.7}$$

to zero, as detailed in sec. 5.1. By performing the differentiation before the truncation, we ensure that the resulting configurations satisfy the full equations of motion and are therefore legitimate solutions of the complete theory.

To identify the flat directions in the potential, we will combine numerical and symbolic tools. The procedure is as follows: first, we sample the underlying manifold by performing a gradient descent on a 5-d hypercube. The resulting cloud of points is subsequently analysed using local principal component analysis (PCA) and clustering algorithms, which allow us to infer the dimension and topological structure of the manifold. After these properties are ascertained, we finally extract analytical constraints defining the manifold thanks to symbolic regression methods. In the next section, we introduce the symbolic

regression algorithm, and defer the details of the numerical methods and results to the following one.

3 Annealed Importance Sampling for Polynomial Symbolic Regression

The numerical sampling of the space of vacua described above results in clouds of points embedded in a higher-dimensional space. In the realm of symbolic regression, the aim is to uncover interpretable mathematical expressions that best describe the embedding of these loci of solutions. In the present context, the form of the supergravity potential (2.5) determining the solutions implies that the loci can be defined through polynomial expressions such that $p(\vec{x}^{\,i}) = 0$, with $i \in \{1, \ldots, \texttt{n\_sample}\}$. Finding such polynomials involves navigating a vast, discrete, and often rugged search space of possible symbolic models, which poses significant challenges for traditional sampling methods. Markov Chain Monte Carlo (MCMC) techniques such as Metropolis-Hastings or Gibbs sampling, while widely used, can struggle with poor mixing and often get trapped in local optima, especially in high-dimensional or multimodal spaces [**?**].

To address these challenges, we employ an Annealed Importance Sampling (AIS) [44] combined with Sequential Monte Carlo (SMC) methods [53]. AIS aims to estimate ratios of partition functions by constructing a sequence of intermediate distributions that smoothly transition from an initial, tractable distribution (e.g. a prior over symbolic expressions) to the complex target posterior distribution. This annealing process is guided by a temperature-like parameter that gradually emphasises the data likelihood, allowing for more efficient exploration of the probability landscape.

SMC realises this procedure by encoding the probability distributions into a set of particles—each representing a candidate symbolic expression— and weights, and making them evolve so as to focus computational effort on high-probability regions. This combination of importance sampling, resampling, and mutation maintains diversity among the particles and prevents premature convergence to suboptimal models. These features make AIS-SMC particularly well-suited for symbolic regression tasks, where the search space is not only high-dimensional but also structured and discontinuous.

Let us now explain in more detail the procedure. The goal is to obtain a probability distribution $d(z)$, where z runs over a space of polynomials. This function assigns high probabilities to those polynomials that annihilate the data, and therefore will be used to identify such symbolic expressions. We reconstruct this density function by series of density functions $\pi_n(z)$ such that $\pi_n \to d(z)$ as $n \to \infty$. On every annealing step, π_n is defined in terms of an unnormalised density γ_n and a normalisation constant as $\pi_n(z) = \gamma_n(z)/Z_n$. The unnormalised density at level n is given in terms of a prior distribution $d_0(z)$ over the hypothesis space and a loss function $L(z)$:

$$\gamma_n(z) = d_0(z)\, \exp\left(-\beta_n L(z)\right), \tag{3.1}$$

where the inverse temperature constants β_n are taken to evolve monotonically as $\beta_1 < \beta_2 < \cdots < \beta_N$.

Sequential importance sampling [**?**] further assumes that the density functions $\pi_n(z)$ can be given in terms of importance densities $q_n^k(z)$ and weights w_n^k as $\pi_n(z) = \sum w_n^k q_n^k(z)/Z_n$, and these importance densities can be sequentially given as

$$q_n(z_n) = q_{n-1}(z_{n-1}) q_{n|n-1}(z_n|z_{n-1})\,. \tag{3.2}$$

At each epoch, we π_n can be approximated via a set of particles and weights $\{z_n^k,\ w_n^k\}$ so that[2]

$$\pi_n(z) \approx \sum_k w_n^k\ \delta(z - z_n^k)\,, \tag{3.3}$$

[2]In the following, we always normalise the weights so that $\sum_k w_n^k = 1$.

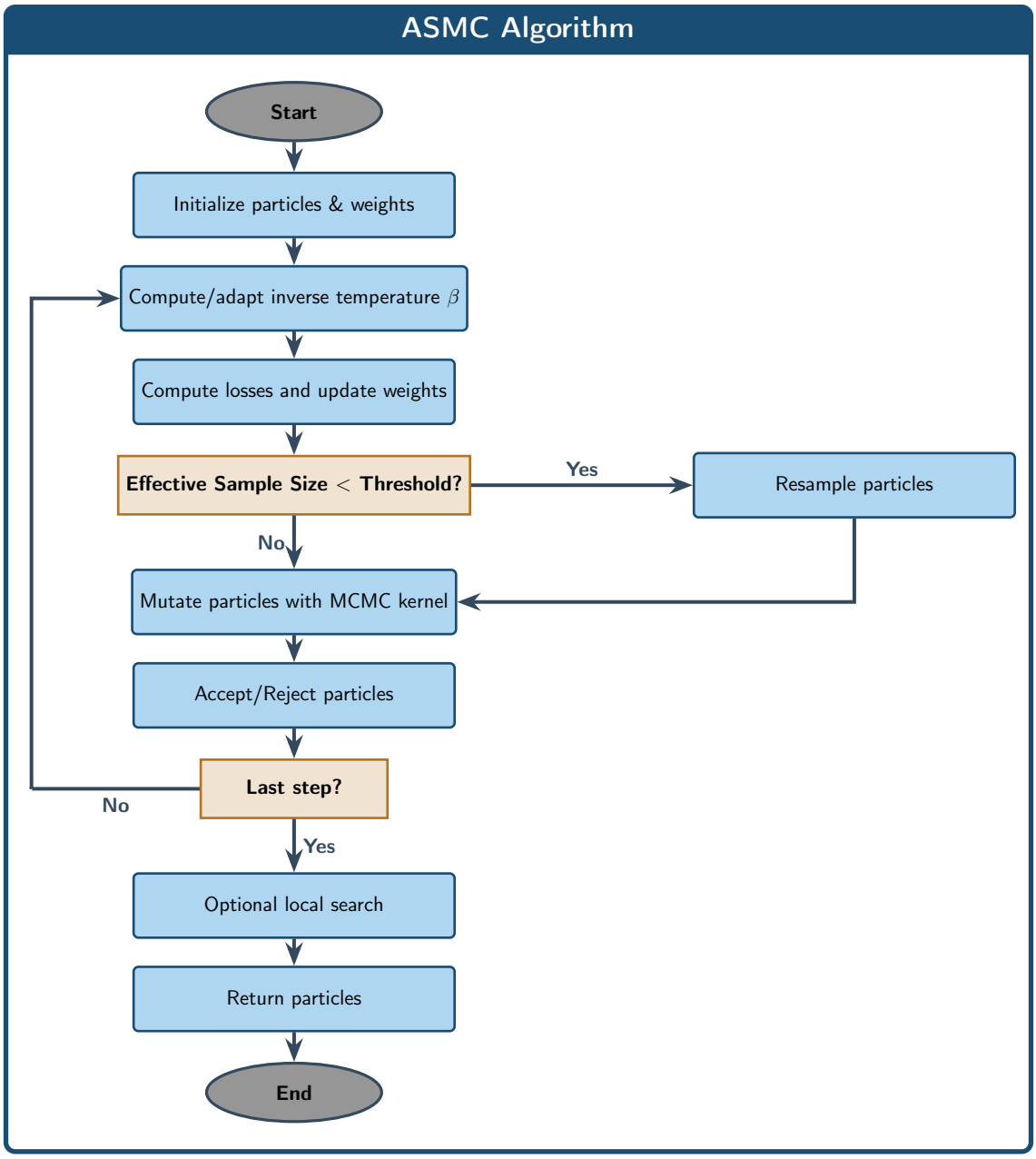

Fig. 1 Flow chart of the Annealed Importance Sampling - Sequential Monte Carlo algorithm

To obtain a new set $\{z_{n+1}^k,\ w_{n+1}^k\}$ approximating π_{n+1}, we sample from the forward propagation kernel $q_{n+1}(z_{n+1}|z_n)$ that defines how we generate the state at time $n+1$ given the state at time n. Similarly, the evolution of the weights is given by **(gl: for $k > 1$, this is a bit schematic...)** ⟸

$$w_{n+1} = \frac{\gamma_{n+1}}{q_{n+1}} = \frac{\gamma_{n+1}}{q_n\ q_{n+1|n}} = w_n\ \alpha_{n+1}\,,$$

where the incremental importance weights α_n^k are given by

$$\begin{aligned} \alpha_{n+1}^k &= \frac{\gamma_{n+1}(z_{n+1})}{\gamma_n(z_n)} \frac{1}{q_{n+1|n}^k(z_{n+1}|z_n)} \\ &= e^{-\beta_n L(z_n)+\beta_{n+1}L(z_{n+1})} \frac{1}{q_{n+1|n}^k(z_{n+1}|z_n)}\,. \end{aligned} \tag{3.4}$$

For the implementation of the forward propagation kernel, we perform an AIS-style MCMC algorithm. On every step, we make some move in the space of polynomials, and then accept or reject those new polynomials based on a fixed rate. The moves that have been allowed in our implementation are:

- **Coefficient pertubation**. Given a polynomial z_n, we choose one of its coefficients at random and modify it by a Gaussian noise distributed as $\mathcal{N}(0, \sigma^2)$. E.g.

$$2x_1 + 3x_2^2 \mapsto 2.1x_1 + 3x_2^2 \,.$$

- **Variable multiplication**. Given a polynomial z_n, we pick randomly one of its monomials, and multiply it by one of the available variables. E.g.

$$2x_1 + 3x_2^2 \mapsto 2x_1x_2 + 3x_2^2 \,.$$

- **Variable division**. Given a polynomial z_n, we choose randomly one of its monomials, and divide it by one of its variables. E.g.

$$2x_1x_2 + 3x_2^2 \mapsto 2x_1 + 3x_2^2 \,.$$

Each of these operations is chosen randomly, with probabilities p_{shift}, p_{multiply}, and p_{divide}. After performing these updates, the change is accepted based on the temperature-dependent rate **(gl: need to double check this)** ⟸

$$A(z_n, z_{n-1}) = \min\left(1, \frac{\gamma_n(z_n^k) q(z_{n-1}^k | z_n^k)}{\gamma_{n-1}(z_{n-1}^k) q(z_n^k | z_{n-1}^k)}\right) . \tag{3.5}$$

We then draw $u \sim \text{Uniform}(0, 1)$, accept the new particle if $u < A(z_n, z_{n-1})$ and reject it otherwise. The kernel $q(z_{n-1}^k | z_n^k)$ thus encapsulates the effects of performing these operations on the polynomials at step n.

To compute the weights and kernels via (4.11) and (4.9) we need to consider the loss function in (4.5). In the present context, we use the following loss, where the polynomials have been parameterised as $z = \sum_k c_k X_k$ with X_k denoting the possible monomials up to a given degree,[3]

$$L(z) = \sum_i z(x_i)^2 + \frac{\lambda}{\sum_k |c_k|} \,. \tag{3.6}$$

Here, the first term is just the sum of the square of the polynomial evaluated on the data, and is therefore minimal when the polynomial annihilates the data. The second term is a regularisation factor that prevents the algorithm to send all the coefficient to 0, which would give a trivial solution to the problem. We typically take $\lambda \sim \mathcal{O}(10^3)$. Together with the loss function, we also need to specify a cooling schedule for β. We will discuss different approaches in the next section.

Once the Annealing loop is over, we end up with a total of n_{sample} particles, which in principle should be close to annihilate our data, but whose coefficients may need some refinement. To deal with it, for each polynomial we now only modify its coefficients with a perturbation $\epsilon \sim \mathcal{N}(0, \sigma')$ and keep the particles when the loss function decreases.

[3] So, if we have x and y as variables, and the maximum degree is 2, then X_k are $1, x, y, x^2, y^2, xy$.

4 Annealed Importance Sampling for Polynomial Symbolic Regression

4.1 General idea

The numerical sampling of the space of vacua described above results in clouds of points embedded in a higher-dimensional space. The aim of symbolic regression is to uncover interpretable mathematical expressions that best describe the embedding of these loci of solutions. In the present context, the form of the supergravity potential (2.5) determining the solutions implies that the loci can be defined through polynomials p_m such that $p_m(\vec{x}^{(i)}) = 0$, with $i \in \{1, \dots, n_{\text{points}}\}$. Such polynomials will be called "annihilating polynomials" in the following. Finding annihilating polynomials involves navigating a vast, discrete, and often rugged search space of possible symbolic models, which poses significant challenges for traditional sampling methods. Markov Chain Monte Carlo (MCMC) techniques such as Metropolis-Hastings or Gibbs sampling, while widely used, can struggle with poor mixing and often get trapped in local optima, especially in high-dimensional or multimodal spaces [**?**]. **(ce: Citation.)** ⟸

To address these challenges, we formulate the search for annihilating polynomials as a determination of a density probability on the space of polynomials that stresses higher probabilities on polynomials p that minimises $\sum_{i=1}^{n_{\text{points}}} p(\vec{x}^{(i)})^2$. Such a density probability is *a priori* hard to find, but can be approximated using variations of Important Sampling methods [54]. Importance sampling is a method to approximate expectation values with respect to a target distribution d from a weighted average from another distribution π. It is based on the observation that the expectation

$$\mathbb{E}_d[f] = \int f(z) d(z)\, \mathrm{d}z \tag{4.1}$$

of a function f with respect to d can be computed from the expectation $\mathbb{E}_\pi[wf]$, now computed with respect to the distribution π, with unnormalised $w(z) = d(z)/\pi(z)$. Therefore, if it is easier to draw a sample $\{z^{(k)}\}_{k \in [\![1, n_{\text{particles}}]\!]}$ from π than from d, one can estimate $\mathbb{E}_d[f]$ from the weighted average

$$\mathbb{E}_d[f] \simeq \frac{\sum_{k=1}^{n_{\text{particles}}} w(z^{(k)})\, f(z^{(k)})}{\sum_{i=1}^{n_{\text{particles}}} w(z^{(k)})}. \tag{4.2}$$

The accuracy of this estimation depends on the size of the sample and, crucially, on the choice of the distribution π.

To have an accurate estimation of d, we employ an Annealed Importance Sampling (AIS) [44] combined with Sequential Monte Carlo (SMC) methods [45, 53]. AIS aims at estimating d sequentially from multiple intermediate distributions π_n that smoothly transition from an initial, tractable distribution d_0 (chosen to be easy to sample from) to the complex target distribution d. These distributions are approximated by weighted samples $\{z_n^{(k)}, w_n^{(k)}\}$ of particles, each representing a candidate symbolic expression, such that the resulting weighted average estimates π_n: **(ce: Normalise the weights?)** ⟸

$$\frac{\sum_{k=1}^{n_{\text{particles}}} w_n^{(k)}\, f(z_n^{(k)})}{\sum_{k=1}^{n_{\text{particles}}} w_n^{(k)}} \underset{n_{\text{particles}} \to +\infty}{\longrightarrow} \mathbb{E}_{\pi_n}[f], \tag{4.3}$$

or equivalently

$$\pi_n(z) \simeq \frac{\sum_{k=1}^{N} w_n^{(k)}\, \delta(z - z_n^{(k)})}{\sum_{k=1}^{N} w_n^{(k)}}. \tag{4.4}$$

The π_n are incrementally deformed using a Markov kernel to gradually reproduce the target d and the particles get reweighed. This annealing process is guided by a temperature-like parameter that gradually emphasises the data likelihood, allowing for more efficient and tunable exploration of the polynomials landscape.

This procedure has the disadvantage that the weights variance tends to increase, leading to weight degeneracy. We use SMC methods to incorporate a resampling procedure to AIS, thus focussing computational effort on high-probability regions. This combination of importance sampling, mutation and resampling maintains diversity among the particles and prevents premature convergence to suboptimal models. These features make AIS-SMC particularly well-suited for symbolic regression tasks, where the search space is not only high-dimensional but also structured and discontinuous.

4.2 Detailed procedure

Let us now explain in more detail the procedure. The goal is to obtain a probability distribution $d(z)$, where z runs over a space of polynomials, that assigns high probabilities to those polynomials that annihilate the data, and therefore will be used to identify their symbolic expressions. We reconstruct this density function d by series of density functions $\pi_n(z)$, $n \in [\![1, n_{\text{epochs}}]\!]$, such that $\pi_n \to d$ as $n \to n_{\text{epochs}}$. On every annealing step, π_n is defined in terms of an unnormalised density γ_n and a normalisation constant as $\pi_n(z) = \gamma_n(z)/Z_n$. The unnormalised density at level n is given in terms of a prior distribution $d_0(z)$ over the space of polynomials and a loss function $L(z)$, as **(ce: Citation.)** ⟸

$$\gamma_n(z) = d_0(z)\, \exp\Big(- \beta_n L(z)\Big), \tag{4.5}$$

where the inverse temperature constants β_n are taken to evolve monotonically as $\beta_1 < \beta_2 < \cdots < \beta_{n_{\text{epochs}}}$. The importance given to each polynomial z is thus updated at each steps, with polynomials z with high loss being disfavoured. This update is controlled by the inverse temperature constants, enabling deeper exploration during the early stages and protecting the procedure from being trapped in local minima. The loss function is chosen to be minimal on annihilating polynomials, its design will be discussed at the end of the section. We chose the prior distribution d_0 to be flat on the space of polynomials, giving equal importance to all of them, which makes easy the initial sampling of the particles $\{z_0^{(k)}\}$.

At each epoch n, π_n is approximated *via* a set of particles and weights $\{z_n^{(k)},\ w_n^{(k)}\}$ in the sense of eq. (4.4). The sample of particles $z_n^{(k)}$ get mutated using a forward propagation Markov kernel $q(z_{n+1}^{(k)}|z_n^{(k)})$, defining the probability to get $z_{n+1}^{(k)}$ when the current state is $z_n^{(k)}$. For the implementation of the forward propagation kernel, we perform an AIS-style MCMC algorithm. On every epoch, we make some move in the space of polynomials, and then accept or reject those new polynomials based on a fixed rate. The moves that have been allowed in our implementation are the following.

- **Coefficient pertubation** Given a polynomial, we choose one of its coefficients at random and modify it by a Gaussian noise distributed as $\mathcal{N}(0, \sigma^2)$, *e.g.*

$$2x_1x_2 + 3x_2^2 \longmapsto 2.1x_1x_2 + 3x_2^2\,. \tag{4.6}$$

- **Variable multiplication** Given a polynomial, we pick randomly one of its monomials, and

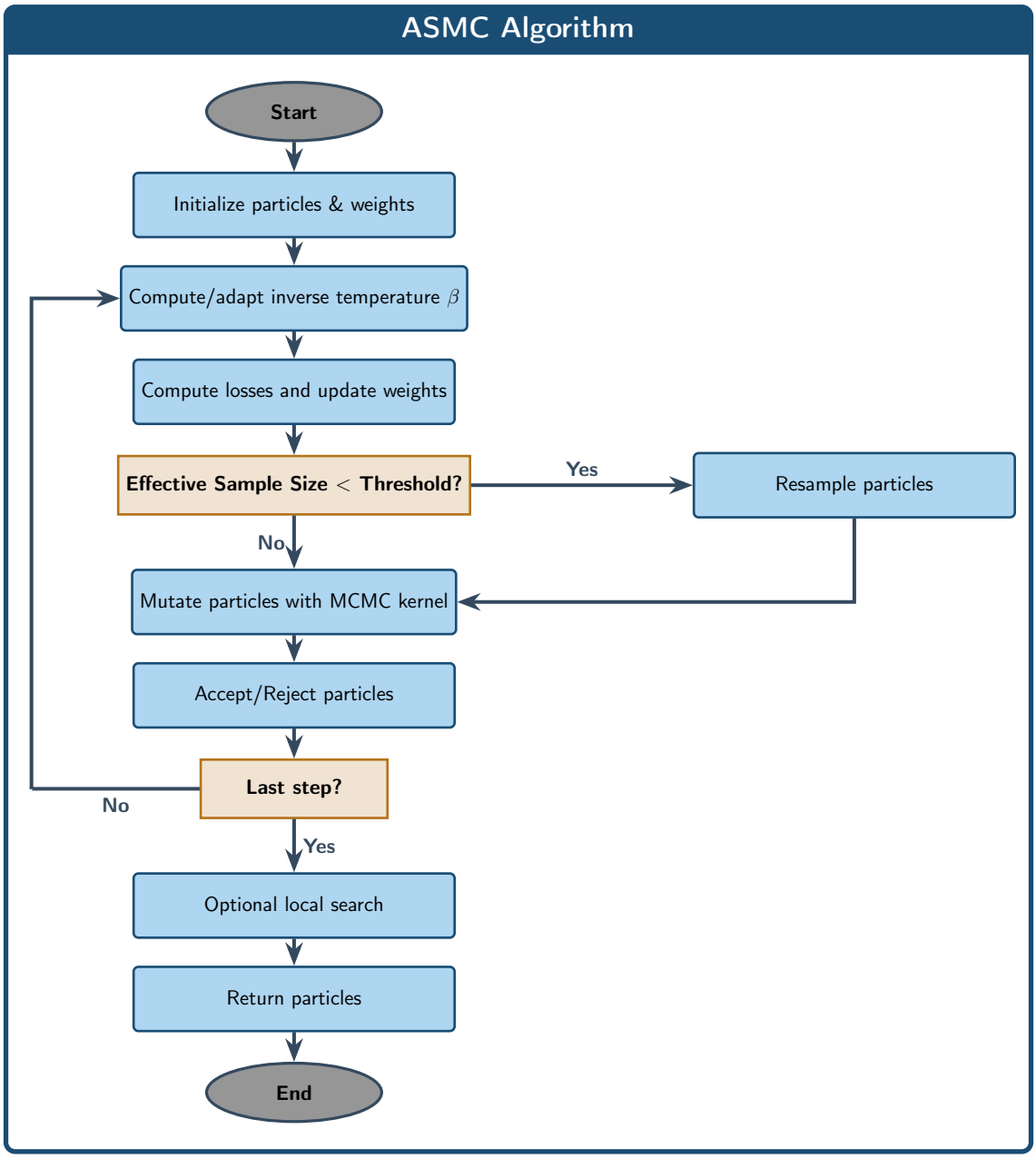

Fig. 2 Flow chart of the Annealed Importance Sampling - Sequential Monte Carlo algorithm

multiply it by one of the available variables, *e.g.*

$$2x_1x_2 + 3x_2^2 \longmapsto 2x_1x_2^2 + 3x_2^2\,. \tag{4.7}$$

- **Variable division** Given a polynomial, we choose randomly one of its monomials, and divide it by one of its variables, *e.g.*

$$2x_1x_2 + 3x_2^2 \longmapsto 2x_1 + 3x_2^2\,. \tag{4.8}$$

Each of these operations is chosen randomly, with probabilities p_{shift}, p_{multiply}, and p_{divide}. After performing these updates, the change is accepted based on the temperature-dependent rate

$$A(z_{n+1}^{(k)}, z_n^{(k)}) = \min\left(1, \frac{\gamma_{n+1}(z_{n+1}^{(k)})q(z_n^{(k)}|z_{n+1}^{(k)})}{\gamma_n(z_n^{(k)})q(z_{n+1}^{(k)}|z_n^{(k)})}\right)\,. \tag{4.9}$$

We then draw $u \sim \text{Uniform}(0,1)$, accept the new particle if $u < A(z_{n+1}^{(k)}, z_n^{(k)})$ and reject it otherwise. The kernel $q(z_{n-1}^k | z_n^k)$ thus encapsulates the effects of performing these operations on the polynomials at step n. **(ce: Why this comment?)** ⟸

Once the particles have been mutated, the evolution of the weights is given by

$$w_{n+1}^{(k)} = w_n^{(k)} \; \alpha_{n+1}^{(k)}, \tag{4.10}$$

where the incremental importance weights $\alpha_{n+1}^{(k)}$ are given by[4] [45]

$$\alpha_{n+1}^{(k)} = \frac{\gamma_{n+1}(z_{n+1}^{(k)})}{\gamma_n(z_n^{(k)})} \frac{q(z_n^{(k)}|z_{n+1}^{(k)})}{q(z_{n+1}^{(k)}|z_n^{(k)})} = \frac{\exp\big(-\beta_{n+1} L(z_{n+1}^{(k)})\big)}{\exp\big(-\beta_n L(z_n^{(k)})\big)} \frac{q(z_n^{(k)}|z_{n+1}^{(k)})}{q(z_{n+1}^{(k)}|z_n^{(k)})}. \tag{4.11}$$

To ensure an efficient exploration of the landscape, we would like to avoid having to many particles with low weights: as the algorithm progresses, particles with negligible weights contribute minimally to the approximation of the target distribution (see eq. (4.4)), leading to a concentration of the effective sample on a subset of particles with substantial weights. This can be estimated using the effective sample size [55]

$$\text{ESS}_{n+1} = \left(\sum_{k=1}^{n_{\text{particles}}} {\tilde{w}_{n+1}^{(k)}}^2 \right)^{-1}, \tag{4.12}$$

where the $\tilde{w}_{n+1}^{(k)}$ are normalised weights deduced from the $w_{n+1}^{(k)}$. ESS_{n+1} takes values between 1 (complete degeneracy, there is only one meaningful particle) and $n_{\text{particles}}$ (no degeneracy, all particles contribute equally). We resample the particles when the ESS falls below a predetermined threshold, typically $n_{\text{particles}}/2$. During resampling, new particles are drawn from the empirical distribution defined by the current particle weights $\tilde{w}_{n+1}^{(k)}$. This procedure results in a new particle set where high-weight particles may appear multiple times, while low-weight particles may be eliminated entirely. Following resampling, all particle weights are reset to uniform values $w_{n+1}^{(k)} = 1/n_{\text{particles}}$, as the information previously encoded in the weight distribution has been incorporated into the spatial distribution of the resampled particles. This resampling mechanism ensures that computational resources are concentrated on exploring the most promising regions of the state space.

A flow chart of the full AIS-SMC procedure is given in fig. 2. Note that in practice the reweighing and resampling of particles $z_{n-1}^{(k)}$ are done at the beginning of the epoch n.

To compute the acceptance ratio and weights *via* eq. (4.9) and (4.11) and we need to consider the loss function in (4.5). In the present context, we use the following loss, where the polynomials have been parameterised as $z = \sum_m c_m X_m$ with X_m denoting the possible monomials up to a given degree,[5]

$$L(z) = \sum_{i=1}^{n_{\text{points}}} z(x^{(i)})^2 + \frac{\lambda}{\sum_m |c_m|}. \tag{4.13}$$

Here, the first term is just the sum of the square of the polynomial evaluated on the data, and is therefore minimal when the polynomial annihilates the data. The second term is a regularisation factor that prevents the algorithm to send all the coefficient to 0, which would give a trivial solution to the problem. We typically take $\lambda \sim \mathcal{O}(10^3)$. Together with the loss function, we also need to specify a

[4] The incremental weight is in general defined as $\alpha_{n+1}^{(k)} = \frac{\gamma_{n+1}(z_{n+1}^{(k)})}{\gamma_n(z_n^{(k)})} \frac{K(z_n^{(k)}|z_{n+1}^{(k)})}{q(z_{n+1}^{(k)}|z_n^{(k)})}$, where $K(z_n^{(k)}|z_{n+1}^{(k)})$ is the backward propagation kernel [45]. For simplicity, we chose to identify K to the forward propagation kernel q.

[5] So, if we have x and y as variables, and the maximum degree is 2, then X_m are $1, x, y, x^2, y^2, xy$.

cooling schedule for β. We will discuss different approaches in the next section.

Once the Annealing loop is over, we end up with a total of $n_{\text{particles}}$ particles, which in principle should be close to annihilate our data, but whose coefficients may need some refinement. To deal with it, for each polynomial we run a fine tuning loop as described in sec. 5.3.2.

5 Numerical Analysis and Results

After describing the theory we want to address in sec. 2, and the symbolic regression method to be employed in sec. 4, we now proceed to detail the numerical analysis and present our main results. We start by describing how we sampled the space of solutions using gradient descent. Once the manifold correctly sampled, we determine its dimension using a local principal component analysis and a clustering procedure. We finally exploit the ASMC algorithm described in sec. 4 to find an analytical characterisation of the manifold.

5.1 Sampling the manifold: gradient descent

To perform the gradient descent, we initialise randomly and uniformly points within a hypercube of range $[-2, 2]$. This choice is important to ensure that points are not restricted to the inner range $[-1, 1]$, which will be crucial for the symbolic regression.[6] As the example we are focussing on has a five-dimensional parameter space, we choose to generate $n_{\text{points}} = 10^5$ points, since we aim at populating all five directions and want approximately $\mathcal{O}(10)$ points per direction. As the solutions lie within the five dimensional space, its intrinsic dimension is less than or equal to 5. The value 10^5 then serves as a conservative upper bound for the number of points needed to adequately sample the manifold.

We then perform a gradient descent on the points using TensorFlow's automatic differentiation [56]. The loss function is defined as

$$\mathcal{L} = \sum_{i=1}^{n_{\text{points}}} \left|\left| \nabla V(\vec{X}^{(i)}) \Big|_{\vec{y}^{(i)}=0} \right|\right|^2 , \tag{5.1}$$

with the vectors $\vec{X}$ and $\vec{y}$ given in (2.6) and (2.7). The exponents (i) denote the data points, with $i \in [\![1, n_{\text{points}}]\!]$. This way we only keep $(x_1^{(i)}, x_2^{(i)}, x_4^{(i)}, \tilde{x}_8^{(i)}, x_{10}^{(i)})$ alive after we have taken the derivative. As already mentioned above, we use the analytic formula for ∇V, but we could have performed the gradient descent by fully using automatic differentiation.

The Adam optimizer was employed throughout the gradient descent procedure [57]. As an accelerator of convergence, we observed that periodically reinitialising the optimizer significantly improved the convergence rate. It was reinitialised at iterations 250, 500, 750, with the learning rate α fixed at 10^{-2}. At iteration 1000, the optimizer was reinitialised once more, this time with a reduced learning rate of 10^{-3}. A final reinitialisation was performed at iteration 1500, setting the learning rate to 10^{-4}, and the optimization was continued for an additional 500 epochs. The evolution of the loss function (5.1), along with the learning rate schedule, is shown in fig. 3. As can be seen in this figure, the convergence rate improves significantly each time the optimizer is rebooted. We also observe that, following the last few reinitialisations, the loss exhibits a small bump immediately after the restart. We interpret this behaviour as follows: the learning rate gets internally adjusted by the Adam optimizer during the descent, and may well be smaller than the instructions when the reset occurs. The learning rate then gets suddenly increased, and some points that previously had low loss values may momentarily worsen

[6] The rationale is that our symbolic regression evaluates the data points $\vec{x}^{(i)}$ using a candidate polynomial function z. When all data points lie within the interval $[-1, 1]$, the algorithm tends to favor high-degree polynomials artificially lowering the loss. Introducing data points with absolute values greater than 1 mitigates this bias.

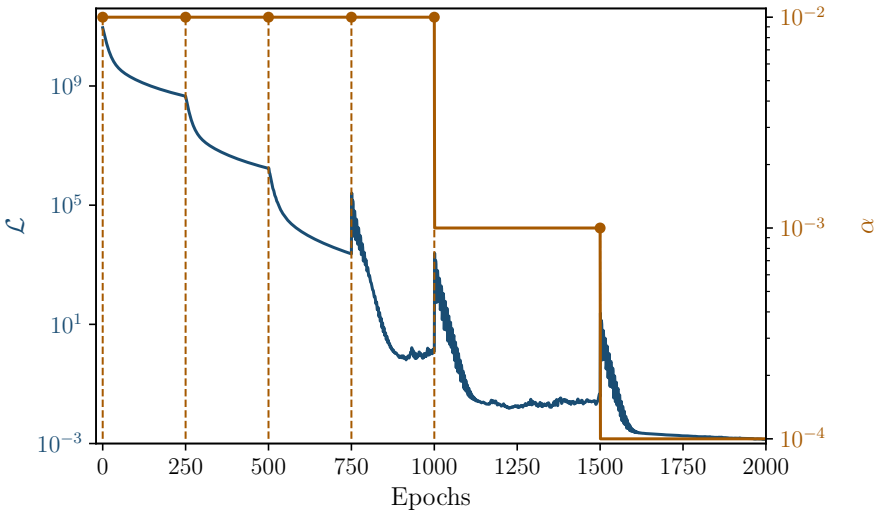

Fig. 3 Evolution of the loss function $\mathcal{L}$ in (5.1) during gradient descent (right axis, in blue) and corresponding learning rate schedule α (left axis in orange), both plotted in logarithmic scale. Dashed lines indicate epochs at which the optimizer was reinitialized.

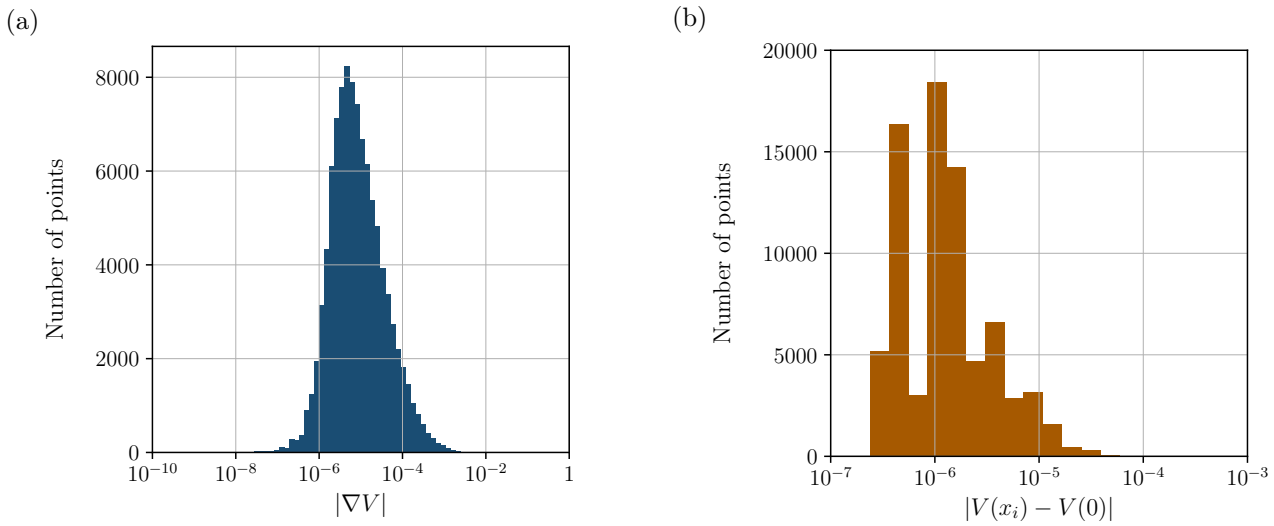

Fig. 4 (a) Histogram of the the norm of the gradient for each point (x-axis is in log scale). (b) Histogram of the distance of the value of the potential to $V(0)$ for each point (x-axis is in log scale).

before benefiting from faster convergence. This effect of faster convergence after the reinitialization is likely due to points initially located in regions with weak attractive basins being pushed toward areas where the potential gradient is steeper, thus accelerating their convergence. This strategy introduces the risk of desampling certain regions in favour of others, introducing the risk to hide some flat directions in the potential. However, we believe that with enough points, it is very unlikely for a flat direction to be completely desampled. This method bears similarity to the concept of warm restarts introduced in ref. [58], where the authors reset the learning rate to some value at each reset, without reinitialising the whole optimizer. However, we found that simply scheduling the learning rate without resetting the optimizer yielded slower convergence. We interpret this as follows: when the optimizer is reinitialised, it effectively "forgets" its past gradient history. As a result, the actual learning rate used corresponds more closely to the specified value, rather than being internally adjusted based on accumulated past gradients. This effect seems to contribute to faster convergence in our case.

Upon completion, the loss function converges around 0.002 **(gl: 10^{-3}?)**. The distribution of the ⟸

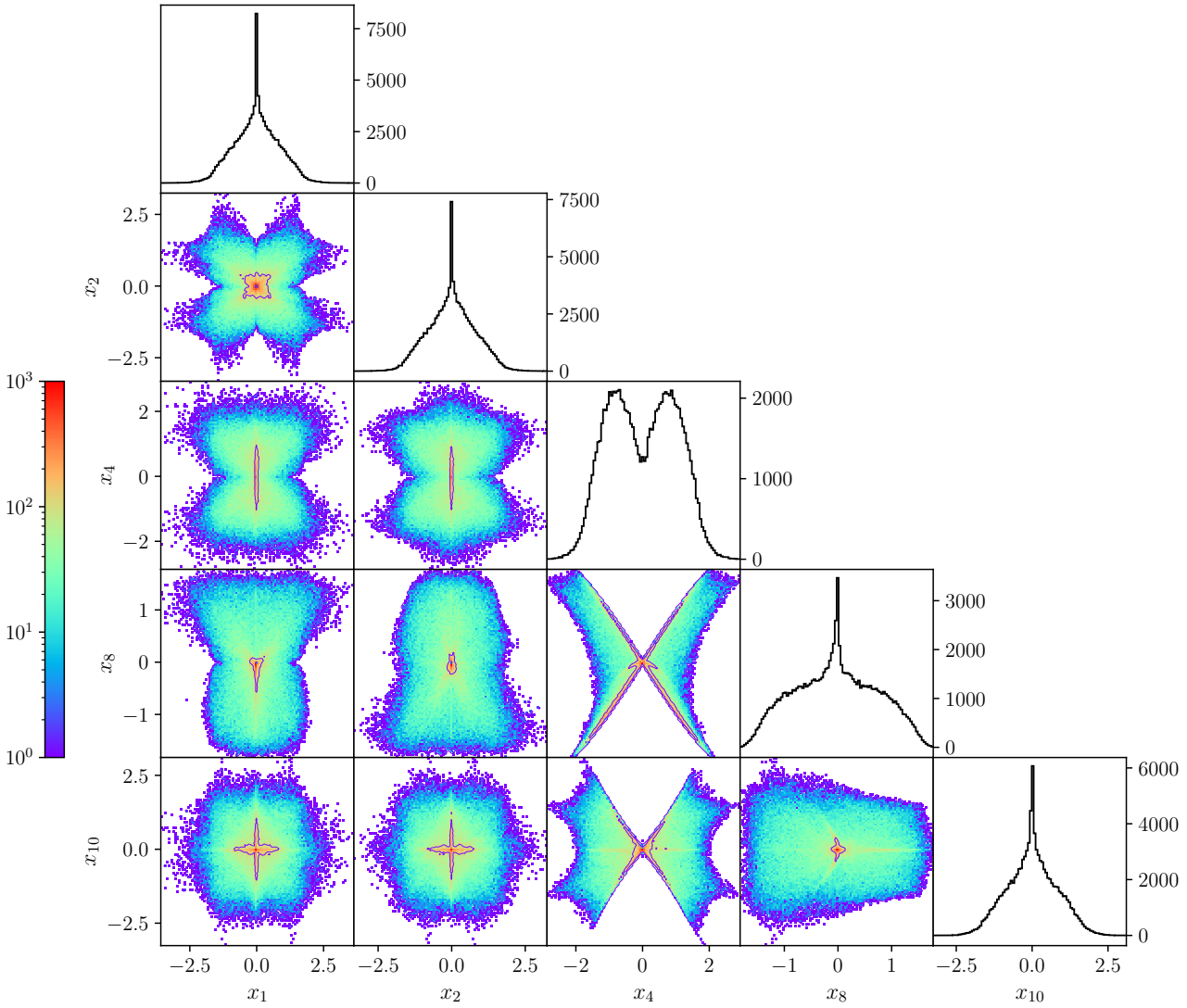

Fig. 5 Triangular plot showing all $2d$ projections and $1d$ histograms of the data after the gradient descent.

values of the gradient at each of the data points is plotted in fig. 4(a). More than 93% of the points converged to values of $|\nabla V|$ lower than 10^{-4}, and more than 99% of them exhibited gradient norms smaller than 10^{-3}, yielding a satisfying sampling of the flat directions. As can be observed in fig. 4(b), all data points converged to values close to $V(0) = -4$. Specifically, nearly all points attained $V(0)$ within an absolute error of at least 10^{-4}, with the exception of four points whose deviations were of the order of 10^{-3} (not visible on the graph). As the origin $\vec{X} = 0$ of the parameter space corresponds to the three-dimensional truncation of the round $\mathrm{AdS}_3 \times S^3$ solution of half-maximal supergravity in six dimensions with $V(0) = -4$, this confirms that the data points lie within flat directions of the potential.

As a first visualisation, we present a tomography of the data in fig. 5. This figure shows all possible 2d projections of the data, along with the 1d histograms of each coordinate after gradient descent. Note that all directions appear to be well populated. There also seem to be non-trivial correlations in the data, see for example the x_1/x_2, $x_4/\tilde{x}_8$ or x_4/x_{10} graphs. They could be genuine correlations, or may result from larger basins of attraction. Thanks to the subsequent analysis, we find that the structures in the x_1/x_2 plot are artefacts of the gradient descent, whereas the ones seen in the $x_4/\tilde{x}_8$ or x_4/x_{10} plots reflect genuine features of the manifold.

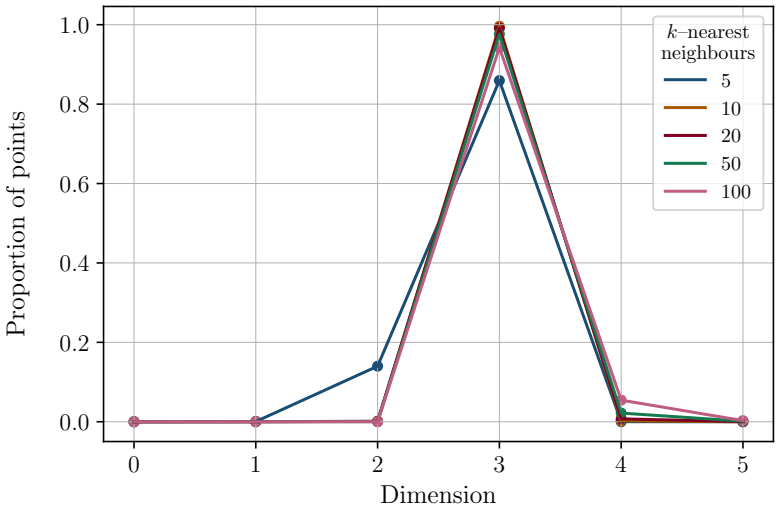

Fig. 6 Results of the local PCA analysis. The x-axis shows the dimension inferred by the algorithm, and the y-axis indicates the proportion of points for which that dimension was found. Each curve corresponds to a different value of k in the k-nearest neighbours.

5.2 Local analysis: extracting the dimension

Once the gradient descent has been completed and the flat directions sampled, the next step is to identify the structure of the underlying manifold. Our goal is to eventually obtain an analytical expression, not just a numerical description. Before applying symbolic regression to search for such an expression, we first perform some exploratory analyses to better understand the data. Specifically, we aim at determining the dimension of the manifold and whether it consists of a single connected component or multiple disjoint components (*e.g.* two intersecting hyperplanes). To this end, we apply a local Principal Component Analysis (PCA,see for example ref. [59]). For each point, we identify its k nearest neighbours and perform a PCA on that local neighbourhood. This procedure allows us to determine how many principal directions are needed to explain a given proportion ϵ of the data variance. In other words, it provides an estimate of the local dimensionality around each point, *i.e.* the dimension of its local tangent space. We perform this analysis for several values of k, namely $k \in \{5, 10, 20, 50, 100\}$, and we fix $\epsilon = 0.99$. The results are presented in fig. 6. We observe that for every choice of k, there is a prominent peak at $d = 3$, suggesting that the underlying manifold is three-dimensional. For $k = 5$, a noticeable fraction of points are assigned dimension 2. This can be attributed to the fact that if the genuine dimension is 3, then selecting only 5 neighbours may not sufficiently populate all three directions, leading the algorithm to underestimate the dimensionality for a fraction of the points. Additionally, for $k \geq 20$, we observe an increasing number of points being assigned dimensions 4 or 5. This behaviour can be explained by the loss of locality when the number of neighbours becomes too large: increasing k results in a coarser approximation, and the algorithm may then incorporate points that are no longer truly local. This artificial enlargement of the neighbourhood can cause the estimated local dimensionality to rise. We thus infer that the manifold under investigation has an intrinsic dimension of 3.

After determining the local dimension of the space of solutions, we need to ascertain its topology. One possible scenario is that our data consists of several three-dimensional manifolds, intersecting at least at the origin $\vec{X} = 0$, and the points previously identified with dimension 4 may lie at the intersections of these manifolds. Consider for example the intersection of two lines: at the intersection point, the local dimension estimated by the previous PCA algorithm would be 2. To rule out this possibility, we apply a clustering algorithm on the points with tangent spaces of dimension 3 only, as

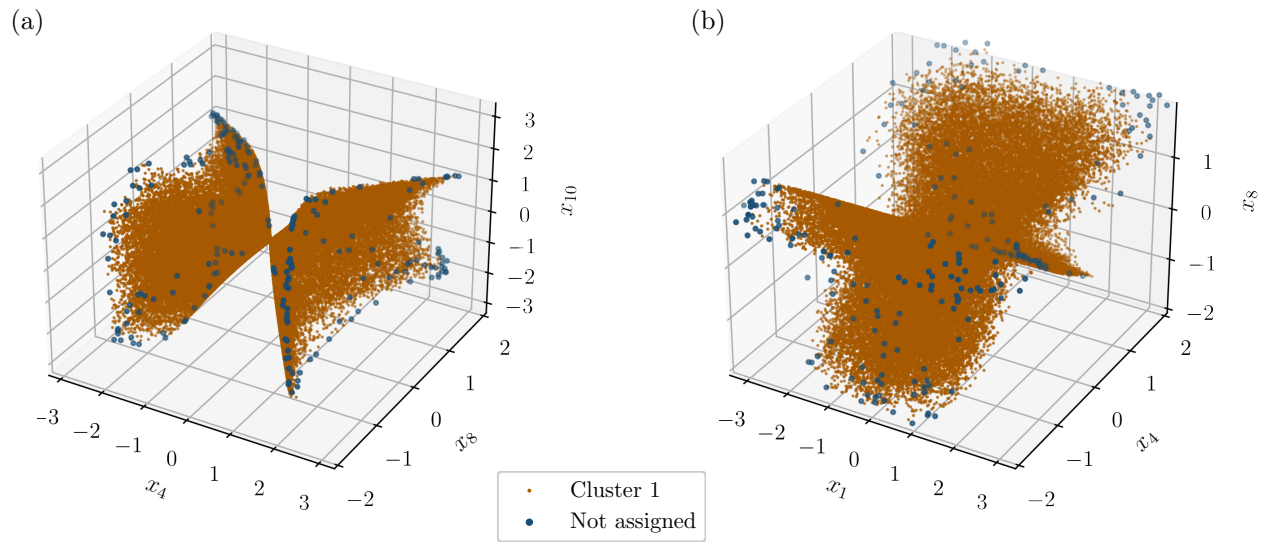

Fig. 7 3d projections of the data set in selected coordinates: (a) (x_4, x_8, x_{10}) and (b) (x_1, x_4, x_8). The colors are assigned by the clustering algorithm: the orange points correspond to the cluster, while the blue points were not assigned to any cluster. The size of each class of points has been tuned to favour the visualisation.

identified by the PCA with $k = 20$. This, way, we remove the possible intersection points with local dimension 4 or 5. For the purpose of the clustering, we use the density-based algorithm HDBSCAN [60]. As it can be observed in fig. 5, the points obtained after the gradient descent are denser around the origin, leading the algorithm to disregard points with a norm greater than 1. To avoid this effect, we de-sample the densest areas by randomly selecting no more than 10 points per 0.25-sided hyper-cube, and apply the HDBSCAN algorithm on the remaining points, with minimum cluster size set to 5 points. For our initial 10^5 points, the algorithm identifies one cluster made of 5 points, one other with 71,993 points, and fails to assign any cluster to 218 points. We show in fig. 7 some 3d projections of the data to visualize the clustering. In these scatter plots, orange points belong to the main cluster, while blue points are those that the algorithm failed to assign to any cluster. The smallest cluster, made of 5 points, is likely an artefact of local fluctuations in the data density and is not interpreted as physically meaningful. The size of the different points has been adjusted to facilitate the visualisation. From visual inspection, it appears that the unassigned (blue) points lie mostly on the boundary of the sampled region. We therefore interpret their unassigned status not as evidence of belonging to another manifold, but rather as a result of insufficient local density near the edges of the dataset. Thus, the algorithm does indicate that over 99% of the data belongs to a single dominant cluster.

5.3 AIS-SMC algorithm and results

From the preceding analysis, we conclude that the gradient descent procedure has produced a sampling of a single, connected, three-dimensional manifold. Therefore to characterise the manifold, we need two independent constraints on the coordinates $\vec{x} = (x_1, x_2, x_4, \tilde{x}_8, x_{10})$. If we have a look at the form of the potential, we can observe that if we use $x_i = e^{\tilde{x}_i}$, then, up to some potential global factor and field redefinitions, this potential is actually a polynomial on the $\vec{X}$ coordinates. Therefore, the components of $\nabla V|_{\vec{y}=0}$ are also polynomials on the $\vec{x} = (x_1, x_2, x_4, x_8, x_{10})$ coordinates. We conclude that the constraints on $\vec{x}$ we are looking for are polynomial constraints of the form $p(\vec{x}) = 0$, and that there should be at least two of those. Of course if one takes directly the gradient of (2.5), one ends up with such conditions, but none are usable directly to solve for two of the variables in terms of the others. The

problem we are facing here is therefore a problem of symbolic regression: we are looking for analytic expressions that vanish once evaluated on our data points. We now proceed to present the symbolic regression problem, utilizing the points obtained through gradient descent as our training dataset. In the following section, we detail the implementation of the AIS-SMC framework previously outlined in sec. 4.

5.3.1 Initialization of the algorithm

We initialized the algorithm with

$$\begin{aligned}
&\texttt{n_points} = 10{,}000,\\
&\texttt{n_epochs} = 1000,\\
&\texttt{beta} = 1e\text{-}6,\\
&\texttt{p_modify} = 0.5,\\
&\texttt{p_multiply} = 0.25,\\
&\texttt{p_divide} = 0.25,\\
&\texttt{lambda} = 1000,\\
&\texttt{sigma} = 0.1,\\
&\texttt{n_particles} = 1000.
\end{aligned} \tag{5.2}$$

The convenience of these parameters was determined empirically –a systematic analysis of its optimization is left for future work. For the representation of polynomials, we adopt a vectorial approach: each polynomial is represented as an array of its coefficients $\{c_m\}$, where $m \in \{1, \dots, n_{\text{mon}}\}$ and n_{mon} denotes the number of available monomials, determined by the chosen maximum degree `max_degree` and the number of variables. We also impose a constraint on the maximum number of terms within a single polynomial `max_num_monomials` to promote sparsity. [7] For our simulations, we chose:

$$\begin{aligned}
&\texttt{max_degree} = 4,\\
&\texttt{max_num_monomials} = 6.
\end{aligned} \tag{5.3}$$

With these parameters and given that we are dealing with 5 variables, $n_{\text{mon}} = 126$.

Regarding the temperature schedule, as can be seen in (4.5), what matters to determine γ_n is not the inverse temperature, but rather the product of the inverse temperature and the loss function. Therefore, instead of implementing an arbitrary schedule for β, we employed an adaptive temperature approach, whose aim is to ensure that the number of particles to be updated at each run is close to

$$\texttt{adaptive\_temperature\_ratio} = 0.8 - 0.5 \times \left(\frac{\texttt{epoch}}{\texttt{n\_epochs}} \right). \tag{5.4}$$

This encourages exploration during the early stages of training and gradually transitions to a more selective regime.

5.3.2 Analysis of a typical run

After a run, the 1000 particles are typically distributed into less than 50 different types of polynomials. Each particles of a given type share the same monomials, but their coefficients fluctuate a bit. We analyse them using the following procedure. We select the particle that features the minimal loss (without the regularisation, $\lambda = 0$, see eq. (4.13)), we fine-tune the coefficients by running a quick

[7] An alternative approach would be to encourage sparsity through the prior distribution or within the loss function formulation.

exploitation phase of 10,000 steps during which we randomly select a coefficient and modify it with a perturbation $\epsilon \sim \mathcal{N}(0, \sigma')$, with $\sigma' = 0.01$, and keep the new particle only if the loss function (without the regularisation) is getting smaller. There are two possible outputs after this phase. (i) The coefficients stay finite, and the particle is a candidate annihilator polynomial. We then filter out all particles of the same type as this candidate, as they will ultimately give the same polynomial. Or (ii), the coefficients are just getting smaller and smaller during the exploitation phase, indicating that the only way to minimize the loss with the given monomials is to have very small coefficients. The particle is then disregarded. We repeat the same steps on the best particles of the remaining set until we have explored all particles.

Here is an example for a given run. The best particle after the Annealing loop is

$$-0.035\,x_1 + 0.673\,x_2 + 0.305\,x_1x_4 + 0.396\,x_1x_{10} - 0.630\,x_2x_8 - 0.284\,x_2x_8x_{10}^2, \tag{5.5}$$

with loss $L^{(\lambda=0)} \simeq 2.1 \times 10^3$. The coefficients have been rounded to the nearest thousandth. After 10,000 steps of exploitation, this particle becomes

$$0.396\,x_2 + 0.396\,x_1x_4 + 0.280\,x_1x_{10} - 0.396\,x_2x_8 - 0.198\,x_2x_8x_{10}^2, \tag{5.6}$$

with loss $L^{(\lambda=0)} \simeq 8 \times 10^{-5}$, or equivalently

$$2\,x_2 + 2.000\,x_1x_4 + 1.415\,x_1x_{10} - 2.000\,x_2x_8 - 1.000\,x_2x_8x_{10}^2, \tag{5.7}$$

where we normalised the coefficients by setting the one of x_2 to 2. We removed the x_1 monomial, because its coefficient was smaller that 10^{-3}. We are in case (i), the polynomial is a candidate annihilator polynomial. After filtering out the 971 particles that are of the same type, the next best particle is

$$1.039\,x_2 + 0.383\,x_1x_4 + 0.466\,x_1x_{10} - 0.955\,x_2x_8 - 0.233\,x_2x_{10}^2 - 0.006\,x_1x_2x_8x_{10}, \tag{5.8}$$

with a loss $L^{(\lambda=0)} \simeq 2.4 \times 10^3$. After the exploitation phase, it becomes

$$0.006\,x_2 + 0.004\,x_1x_4 + 0.002\,x_1x_{10} - 0.006\,x_2x_8 - 0.001\,x_2x_{10}^2, \tag{5.9}$$

with a loss $L^{(\lambda=0)} \simeq 4 \times 10^{-2}$. The loss is low only because the coefficients are themselves very low. This is an example of case (ii), we disregard this particle. Reproducing similar steps for the remaining particles, we get some uninteresting particles and two new candidate polynomials (with normalised coefficients):

$$1.416\,x_2 + 1.416\,x_1x_4 - 1.417\,x_2x_8 + x_1x_8x_{10} - 1.001\,x_2x_4x_8x_{10}, \quad L^{(\lambda=0)} \simeq 1 \times 10^{-3}, \tag{5.10a}$$

$$-1.414\,x_1 + 1.414\,x_1x_8 - 1.414\,x_2x_4x_8 + x_2x_8x_{10}, \quad L^{(\lambda=0)} \simeq 2 \times 10^{-4}. \tag{5.10b}$$

Given that the coefficients of the gradient ∇V are only integers and square roots of integers, the coefficients of the polynomials annihilating ∇V must be combinations of rational numbers or square roots thereof. We then deduce from eq. (5.7), (5.10a) and (5.10b) the candidate annihilator polynomials for the above example:

$$\begin{gathered} 2\,x_2 + 2\,x_1x_4 + \sqrt{2}\,x_1x_{10} - 2\,x_2x_8 - x_2x_8x_{10}^2, \\ \sqrt{2}\,x_2 + \sqrt{2}\,x_1x_4 - \sqrt{2}\,x_2x_8 + x_1x_8x_{10} - x_2x_4x_8x_{10}, \\ -\sqrt{2}\,x_1 + \sqrt{2}\,x_1x_8 - \sqrt{2}\,x_2x_4x_8 + x_2x_8x_{10}. \end{gathered} \tag{5.11}$$

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	$\varnothing$
Frequency	92.6%	75.0%	51.9%	1.7%	0.2%	0.7%	0.7%	0.3%	0.1%
Maximum # of representatives	1000	1000	991	869	879	710	207	18	–

Tab. 1 Statistics of the ability of the ASMC algorithm to produce the polynomials (5.12) on 1000 runs with parameters (5.2) and (5.3). The frequencies represent the percentage of runs featuring a given polynomial in its outputs, and the maximum number of representatives gives the maximum number of particles representing the polynomial in a given run. The column $\varnothing$ counts the runs that failed to produce any annihilator polynomial.

5.3.3 Statistics

We performed 1000 independent runs with the same parameters as above, each involving 1000 particles. It takes approximately 10 min to do a single run on a regular computer, including annealing loop and local search described in the previous section. The code finds the following 8 different polynomials:

$$p_1 = -\sqrt{2}\,x_1 + \sqrt{2}\,x_1x_8 + x_2x_8x_{10} - \sqrt{2}\,x_2x_4x_8, \tag{5.12a}$$

$$p_2 = 2\,x_2 - 2\,x_2x_8 + \sqrt{2}\,x_1x_{10} + 2\,x_1x_4 - x_2x_8x_{10}^2, \tag{5.12b}$$

$$p_3 = \sqrt{2}\,x_2 - \sqrt{2}\,x_2x_8 + \sqrt{2}\,x_1\,x_4 + x_1x_8x_{10} - x_2x_4x_8x_{10}, \tag{5.12c}$$

$$p_4 = 2\,x_2 - 2\,x_2x_8 + \sqrt{2}\,x_1x_8x_{10} + 2\,x_1x_4x_8 - 2\,x_2x_4^2x_8, \tag{5.12d}$$

$$p_5 = -\sqrt{2}\,x_1^2x_4 + \sqrt{2}\,x_2^2x_4 + \sqrt{2}\,x_1x_2x_4^2 - x_1^2x_{10} - x_2^2x_{10}, \tag{5.12e}$$

$$p_6 = -2\,x_1 - 2\,x_2x_4 - 2\,x_1x_4^2 + 2\,x_1x_8 + \sqrt{2}\,x_2x_{10} + x_1x_8x_{10}^2, \tag{5.12f}$$

$$p_7 = -2 + 4\,x_8 - 2\,x_8^2 + 2\,x_4^2x_8 - x_8^2x_{10}^2, \tag{5.12g}$$

$$p_8 = -2\,x_2x_4 - 2\,x_1x_4^2 + 2\,x_2x_4x_8 + \sqrt{2}\,x_2x_{10} - \sqrt{2}\,x_2x_8x_{10} + x_1x_8x_{10}^2, \tag{5.12h}$$

where $x_8 = e^{\tilde{x}_8}$. They are found with different frequencies, with some polynomials occurring more often than others, as reported in tab. 1. The algorithm typically identifies an average of 2.2 distinct polynomials per run, with a maximum number of 4, demonstrating its capacity to uncover multiple solutions simultaneously. The algorithm failed at finding a solution in only 3 runs. This is summed up in fig. 8. Note that in 77% of the time the algorithm finds more than a single solution, demonstrating the robustness of the method.

We have tracked the appearance of the polynomials (5.12) during each of the 1000 runs. To do so,

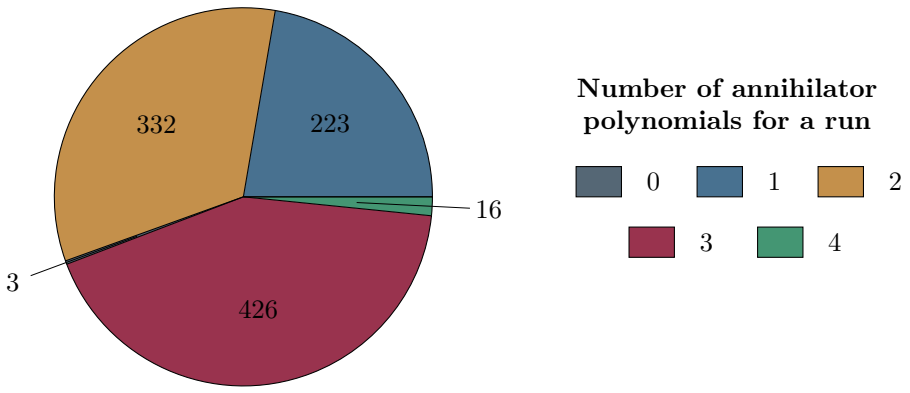

Fig. 8 Pie chart of the repartition of the number of distinct polynomials found in a single run.

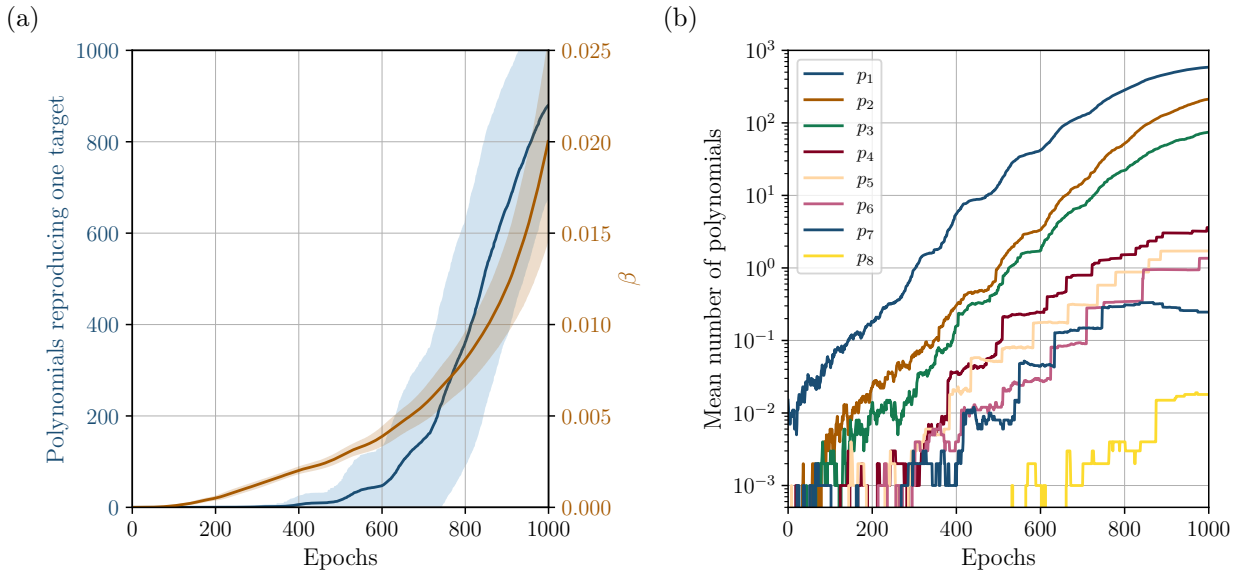

Fig. 9 (a) Evolution of the mean number of particles reproducing a target polynomial, along with its 1σ deviation, for the parameters described in (5.2) (left scale) and evolution of the inverse temperature β and its 1σ deviation (right scale). (b) Dynamics of the mean number of particles reproducing each polynomial in eq. (5.12) (the scale of the y-axis is logarithmic).

we have counted at each epoch the number of particles including the same monomials as the annihilator polynomials.[8] The total number of particles reproducing at least one of the annihilator polynomials listed in eq. (5.12) is plotted in fig. 9(a), together with the evolution of the inverse temperature β, both averaged on the 1000 runs. The proportion of annihilator polynomials among the particles start to be significant after approximately 700 epochs. It then increases quickly and, towards the end of each run, an average of 88% of the particles do reproduce one of the annihilator polynomials, with a standard deviation of 20%. The evolution of this proportion is explained by the structure of the method: the code favours the particles with highest weights, and thus those that cause a significant improvement to the loss function, at each resampling. Once an annihilator polynomials is reached, it colonises larger and larger proportions of the particles at each resampling.

The evolution of the inverse temperature is dictated by eq. (5.4). Note that β is equal to 0.02 on average at the end of the runs. The temperature is thus still quite high, which favours diversity. This is a key ingredient to get more than one candidate polynomials per run. The end value of β is intimately linked to the choice of target adaptative temperature ratio in eq. (5.4): the lower this acceptance ratio, the higher the inverse temperature β. On the one hand, if the ratio is too low, very few particles get mutated and it is difficult to explore the space of polynomials and to have diverse outputs. On the other hand, if the ratio is too high, there are too many mutations: the output features a large number of polynomials, but lots of them do not converge to an annihilator polynomials because the algorithm is not selective enough. It is a matter of balance between exploration and exploitation.

The dynamics of appearance of each annihilator polynomial is plotted in fig. 9(b), averaged on the 1000 runs. The dynamics depend strongly on the polynomial and we observe three different classes. In the first case, for p_1, p_2 and p_4, the first occurrences appear typically after only few dozen of epochs; and in the the second one, constituted of p_3, p_5, p_7 and p_8, after few hundreds of epochs. The last class has p_6 as its only representative. This polynomial is very difficult to produce, and when it appears it is only towards the end of the runs. The stair-step patterns, with sudden jumps in the population of the

[8] Thus, a particle featuring the same monomials as one of the polynomials (5.12) will be counted, even if the coefficients do not match, and if there are additional monomials.

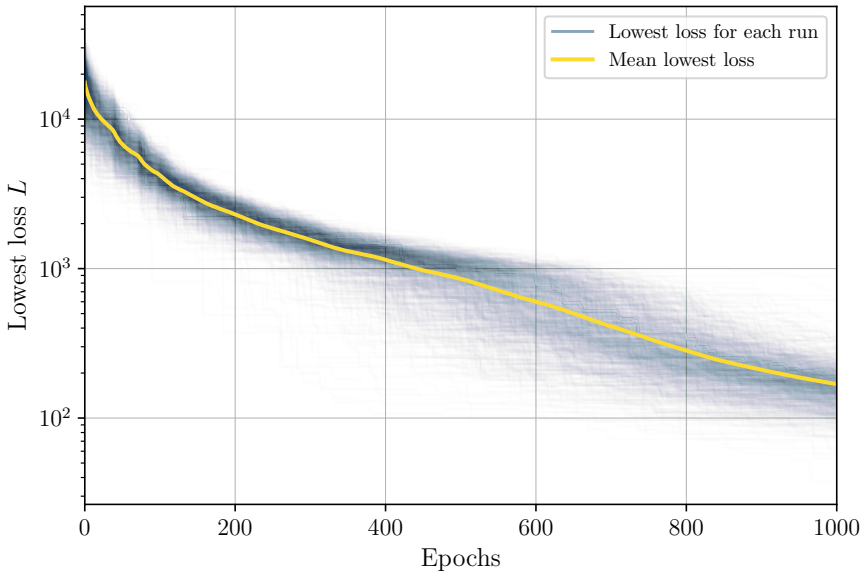

Fig. 10 Evolution of the lowest loss during each of the 1000 runs (in blue) and their average (in yellow). The losses are computed with the regularisation factor λ, see eq. (4.13).

polynomials alternating with phases of stagnation, are due to the use of resampling in the algorithm. When a particle get close enough to an annihilating polynomials, its loss gets lowered significantly faster than the one of the other particles. This induces an increase of its weight and it will populate a large part of the sample at the next resampling. The coefficients get improved during the stagnation phases, inducing an increase of the polynomials weights, and thus even larger colonizations during the following resamplings. This also induces a competition between the annihilator polynomials, the ones that get bettered the more easily (typically those with fewest terms) are favoured. This mechanism explains the decrease in the averaged population of the polynomial p_8 observed after 800 epochs in fig. 9(b).

The best loss for each of the 1000 runs is shown in fig. 10 in blue, together with their mean value in yellow, with a logarithmic y-axis. The losses here do include the regularisation factor λ, see eq. (4.13), and are given without the exploitation phase discussed in sec. 5.3. For a given run, we are plotting the best loss at each epoch, the curves thus do not follow single particles. On average the loss get bettered by two orders of magnitude during a run, illustrating the convergence of the algorithm. The best loss is however quite high at the last epoch. This is linked to the very low value of β: the exploration is favoured with respect to exploitation during all the process. Although the loss is high, 1000 epochs are sufficient to select candidate annihilator polynomials thanks to the exploitation phase performed on the best particles, as illustrated in sec. 5.3.

6 Supergravity Solutions

In the previous section, we introduced a numerical method that enabled symbolic regression, yielding a set of polynomials that vanish on our dataset, as presented in eq. (5.12). We know from sec. 5 that the manifold we are aiming at parametrising is three-dimensional. As the parameter space is of dimension 5, we only need two constraints to define the solutions, and the eight polynomials of eq. (5.12) are not independent. We are however only interested in finding an analytic parametrisation of the solutions manifold, and the easiest to find it is to solve the system

$$p_i = 0, \quad \forall i \in 1, \dots, 8. \tag{6.1}$$

This leads to the following rules between the parameters:

$$\begin{cases} e^{\tilde{x}_8} = \dfrac{x_1^2 + x_2^2}{x_2^2 + \big(x_1 - x_2 x_4\big)^2}, \\ x_{10} = \sqrt{2}\, x_4\, \dfrac{x_2^2 - x_1^2 + x_1 x_2 x_4}{x_1^2 + x_2^2}. \end{cases} \tag{6.2}$$

Alternatively, the system can be recast as:

$$\begin{cases} x_1 = \dfrac{x_2}{e^{\tilde{x}_8} - 1}\Big(x_4\, e^{\tilde{x}_8} \pm \sqrt{-1 + \big(2 + x_4^2\big)\, e^{x_8} - e^{2\tilde{x}_8}}\Big), \\ x_{10} = \mp\, e^{-\tilde{x}_8}\, \sqrt{-2 + 2\,\big(2 + x_4^2\big)\, e^{\tilde{x}_8} - 2\, e^{2\tilde{x}_8}}, \end{cases} \tag{6.3}$$

or

$$\begin{cases} x_1 = \dfrac{x_2}{\sqrt{2}}\, \dfrac{e^{\tilde{x}_8/2}}{e^{\tilde{x}_8} - 1}\Big(- x_{10}\, e^{\tilde{x}_8/2} \pm \sqrt{2 - 4\, e^{\tilde{x}_8} + e^{2\tilde{x}_8}\big(2 + x_{10}^2\big)}\Big), \\ x_4 = \pm \dfrac{e^{-x_8/2}}{\sqrt{2}}\, \sqrt{2 - 4\, e^{\tilde{x}_8} + e^{2\tilde{x}_8}\big(2 + x_{10}^2\big)}. \end{cases} \tag{6.4}$$

As anticipated, this defines a three-parameter manifold. We tested analytically that $\nabla V = 0$ for any of these rules. The three-parameter manifold then defines a three-dimensional space of flat directions of the half-maximal supergravity scalar potential.[9]

The supergravity solution can be shown to preserve a $\mathrm{U}(1) \times \mathrm{U}(1)$ gauged symmetry and breaks all supersymmetry, except at the origin where we recover the $\mathrm{SO}(4)$ isometries and the $\mathcal{N} = (0,4)$ supercharges. Using the parametrisation (6.2), the (x_1, x_2, x_4) moduli space is most nicely parametrised using the change of coordinates

$$x_1 = r\cos(\theta)\cos(\Phi), \quad x_2 = r\cos(\theta)\sin(\Phi) \quad \text{and} \quad x_4 = r\sin(\theta), \tag{6.5}$$

for which the Zamolodchikov metric reads

$$\mathrm{d}^2 s_{\text{Zam.}} = -\,\mathrm{d}r^2 - r^2\left(\mathrm{d}\theta^2 - r\cos(\theta)\,\mathrm{d}\theta\mathrm{d}\Phi + \sin(\theta)\,\mathrm{d}r\mathrm{d}\Phi + \frac{1}{2}\big(3 + r^2 - \cos(2\theta)\big)\,\mathrm{d}\Phi^2\right). \tag{6.6}$$

The spectrum of scalar fields around the flat directions is the following:

$$\begin{aligned} \big(m_{(0)}\ell_{\text{AdS}}\big)^2 : \quad & 0\ [5], \quad 8\ [1], \quad r^2\big(4 + r^2\big)\ [8], \\ & 2\,r\left(3\,r + r^3 - r\cos(2\theta) \pm \big(2 + r^2\big)\sqrt{2 + r^2 - 2\cos(2\theta)}\right)\ [2+2]. \end{aligned} \tag{6.7}$$

The masses are normalised with respect to the AdS length $\ell_{\text{AdS}}^2 = -2/V(0) = 1/2$. The numbers between brackets indicate the multiplicity of each mode. The three-dimensional spectrum is thus stable (*i.e.* satisfies the Breitenlohner-Freedman bound $\big(m_{(0)}\ell_{\text{AdS}}\big)^2 \geq -1$ [61]) if

$$\theta = \pm\frac{1}{2}\,\arccos\left(-1 + \sqrt{3} + \frac{-7 + 4\sqrt{3})}{2\,r^2}\right) \quad \text{and} \quad r \geq \sqrt{-2 + \frac{7}{2\sqrt{3}}}. \tag{6.8}$$

[9] One might argue that only two of the eight polynomials are sufficient to fully characterise the solution. That is, choosing any pair $(i,j) \in 1,\ldots,8$ may suffice to extract a complete description. In practice, this is not entirely accurate. While such a pair can yield partial constraints – for example, recovering eq. (6.2) – it may also produce alternative (and potentially less general) parameterisations. Upon inspection, all such partial rules are found to be consistent with, and included in, the most general expressions given in eq. (6.2).

7 Conclusion

We have presented a novel machine learning approach to systematically identify and characterise flat directions in supergravity scalar potentials. Our methodology combines gradient descent sampling with annealed importance sampling for symbolic regression on polynomials. We demonstrated its efficiency on a 5-scalar subsector of three-dimensional half-maximal supergravity, derived from the $\mathrm{AdS}_3 \times S^3$ compactification of six-dimensional $\mathcal{N} = (1,1)$ supergravity, or analogously of the $\mathrm{AdS}_3 \times S^3 \times T^4$ compactification of IIB supergravity.

We developed a robust pipeline that transitions from numerical exploration to analytical understanding. The gradient descent procedure successfully samples the flat directions manifold, while local PCA analysis reveals its intrinsic dimensionality. Most notably, our annealed importance sampling approach to symbolic regression automatically and quickly discovers polynomial constraints characterising the manifold, bypassing the computational complexity that renders direct symbolic manipulation intractable. The algorithm uncovered eight distinct polynomial relations with varying frequencies, suggesting a hierarchical structure in constraint discovery. Furthermore, upon 1000 runs the algorithm failed at finding any solution in only 3 cases, demonstrating the robustness of the method.

This approach opens several promising avenues for advancement. The scalability to higher-dimensional cases represents the most immediate challenge and opportunity. By increasing the number of particles, refining the annealing schedule, and optimizing polynomial search strategies, we anticipate extending this methodology to the full 13-scalar theory and potentially to other supergravity models. This will pave the way to an exhaustive characterisation of the flat directions of these models, with prominent applications to the AdS/CFT correspondence: flat directions of supergravity solutions having a CFT dual are in correspondence with the space of CFT deformations preserving the conformal symmetry (called the conformal manifold).

Our analysis reveals that the 5-dimensional scalar space contains a 3-dimensional conformal manifold. The discovered solutions preserve a $\mathrm{U}(1) \times \mathrm{U}(1)$ gauged symmetry and breaks all supersymmetries. The Zamolodchikov metric on the moduli space provides a concrete geometric description of the conformal manifold. While the specific solutions found may have limited direct physical applications, the demonstrated feasibility of our approach and its potential for systematic classification of flat directions across the supergravity landscape make it a valuable addition to the theoretical physicist's toolkit. As the solutions live in a 3d consistent truncation of both the $\mathrm{AdS}_3 \times S^3$ solution of $\mathcal{N} = (1,1)$ six-dimensional supergravity and of the $\mathrm{AdS}_3 \times S^3 \times T^4$ vacuum of IIB supergravity, we can uplift the solution to 6d and 10d using the tools of exceptional field theory (ExFT) [62, 63, 50]. The higher-dimensional solutions would be of the form $\mathrm{AdS}_3 \times M^3$ or $\mathrm{AdS}_3 \times M^7$, with M^3, and M^7 being deformations of the round S^3 and $S^3 \times \mathrm{T}^4$, respectively. These deformed spaces have as moduli the 3 parameters $\{x_1, x_2, x_4\}$ that define the solutions. ExFT gives further access to the full Kaluza-Klein spectrum for those compactifications [64–66], allowing a test of the perturbative stability of the solutions, and giving valuable information on the spectrum of states of the dual CFT. These applications relies entirely on the fact that we have an analytic parametrisation of the solutions thanks to the symbolic regression we performed. We live these for future works.

The marriage of machine learning techniques with traditional supergravity analysis represents a step toward more systematic approaches to understanding the rich geometric structures underlying these theories. As computational power increases and algorithms improve, we envision this methodology becoming a standard tool for exploring the intricate relationships between geometry, symmetry, and dynamics in supergravity theories.

A Some Details on the Supergravity Setup

We give here some details on the supergravity theory described schematically in sec 2, following ref. [46, 47].[10] The Lagrangian is given in eq. (2.1). The gauging structure is encoded in an embedding tensor [68, 69] that takes the general form

$$\Theta_{\bar{K}\bar{L}|\bar{M}\bar{N}} = \theta_{\bar{K}\bar{L}\bar{M}\bar{N}} + \frac{1}{2}\Big(\eta_{\bar{M}[\bar{K}}\theta_{\bar{L}]\bar{N}} - \eta_{\bar{N}[\bar{K}}\theta_{\bar{L}]\bar{M}}\Big) + \theta\,\eta_{\bar{M}[\bar{K}}\eta_{\bar{L}]\bar{N}}, \tag{A.1}$$

where $\theta_{\bar{K}\bar{L}\bar{M}\bar{N}} = \theta_{[\bar{K}\bar{L}\bar{M}\bar{N}]}$ is fully antisymmetric, $\theta_{\bar{L}\bar{K}} = \theta_{(\bar{L}\bar{K})}$ is symmetric and traceless, and θ is a scalar. The metric $\eta_{\bar{K}\bar{L}}$ is the SO(8,4)-invariant bilinear form used for index contractions. The gauge covariant derivatives are constructed using the embedding tensor according to

$$D_\mu = \partial_\mu + A_\mu{}^{\bar{M}\bar{N}}\,\Theta_{\bar{M}\bar{N}|\bar{P}\bar{Q}}\,T^{\bar{P}\bar{Q}}, \tag{A.2}$$

where $A_\mu{}^{\bar{M}\bar{N}}$ are the gauge fields and

$$\big(T^{\bar{M}\bar{N}}\big)_{\bar{P}}{}^{\bar{Q}} = 2\,\delta_{\bar{P}}{}^{[\bar{M}}\,\eta^{\bar{N}]\bar{Q}} \tag{A.3}$$

are the generators of the $\mathfrak{so}(8,4)$ algebra. The covariant derivative acting on the scalar matrix is thus

$$D_\mu M_{\bar{M}\bar{N}} = \partial_\mu M_{\bar{M}\bar{N}} + 4\,A_\mu{}^{\bar{P}\bar{Q}}\,\Theta_{\bar{P}\bar{Q}|(\bar{M}}{}^{\bar{K}}\,M_{\bar{N})\bar{K}}, \tag{A.4}$$

ensuring gauge invariance of the scalar kinetic terms. The embedding tensor also defines the scalar potential as following [70, 71]:

$$\begin{aligned} V = \frac{1}{12}\,\theta_{\bar{K}\bar{L}\bar{M}\bar{N}}\theta_{\bar{P}\bar{Q}\bar{R}\bar{S}}\Big(&M^{\bar{K}\bar{P}}M^{\bar{L}\bar{Q}}M^{\bar{M}\bar{R}}M^{\bar{N}\bar{S}} - 6\,M^{\bar{K}\bar{P}}M^{\bar{L}\bar{Q}}\eta^{\bar{M}\bar{R}}\eta^{\bar{N}\bar{S}} \\ &+ 8\,M^{\bar{K}\bar{P}}\eta^{\bar{L}\bar{Q}}\eta^{\bar{M}\bar{R}}\eta^{\bar{N}\bar{S}} - 3\,\eta^{\bar{K}\bar{P}}\eta^{\bar{L}\bar{Q}}\eta^{\bar{M}\bar{R}}\eta^{\bar{N}\bar{S}}\Big) \\ + \frac{1}{8}\,\theta_{\bar{K}\bar{L}}\theta_{\bar{P}\bar{Q}}\Big(&2\,M^{\bar{K}\bar{P}}M^{\bar{L}\bar{Q}} - 2\,\eta^{\bar{K}\bar{P}}\eta^{\bar{L}\bar{Q}} - M^{\bar{K}\bar{L}}M^{\bar{P}\bar{Q}}\Big) + 4\,\theta\theta_{\bar{K}\bar{L}}M^{\bar{K}\bar{L}} - 32\,\theta^2. \end{aligned} \tag{A.5}$$

Finally, the dynamics of the vector fields is governed by the Chern-Simons contribution

$$\mathcal{L}_{\rm CS} = -\varepsilon^{\mu\nu\rho}\,\Theta_{\bar{M}\bar{N}|\bar{P}\bar{Q}}\,A_\mu{}^{\bar{M}\bar{N}}\left(\partial_\nu\,A_\rho{}^{\bar{P}\bar{Q}} + \frac{1}{3}\,\Theta_{\bar{R}\bar{S}|\bar{U}\bar{V}}\,f^{\bar{P}\bar{Q},\bar{R}\bar{S}}{}_{\bar{X}\bar{Y}}\,A_\nu{}^{\bar{U}\bar{V}}\,A_\rho{}^{\bar{X}\bar{Y}}\right), \tag{A.6}$$

where $f^{\bar{M}\bar{N},\bar{P}\bar{Q}}{}_{\bar{K}\bar{L}} = 4\,\delta_{[\bar{K}}{}^{[\bar{M}}\eta^{\bar{N}][\bar{P}}\delta_{\bar{L}]}{}^{\bar{Q}]}$ are the structure constants of $\mathfrak{so}(8,4)$, and $\varepsilon^{\mu\nu\rho}$ is the three-dimensional Levi-Civita symbol.

The parametrisation of the gauging to get the truncation of six-dimensional half-maximal supergravity on S^3 is best described through the decomposition of SO(8, 4) as [51]

$$\begin{aligned} \mathrm{SO}(8,4) &\longrightarrow \ \mathrm{GL}(3,\mathbb{R}) \times \mathrm{SO}(1,1) \times \mathrm{SO}(4)_{\rm global}\,, \\ X^{\bar{M}} &\longrightarrow \ \{X^{\bar{m}},\ X_{\bar{m}},\ X^{\bar{0}},\ X_{\bar{0}},\ X^{\bar{\alpha}}\}, \end{aligned} \tag{A.7}$$

where $\bar{m} \in [\![1,3]\!]$ and $\bar{\alpha} \in [\![9,12]\!]$ label the $\mathrm{SL}(3,\mathbb{R})$ and $\mathrm{SO}(4)_{\rm global}$ vector representations. In this basis,

[10] See also ref. [67] for a review.

the SO(8, 4)-invariant tensor has the expression

$$\eta_{\bar{M}\bar{N}} = \begin{pmatrix} 0 & \delta_{\bar{m}}{}^{\bar{n}} & 0 & 0 & 0 \\ \delta^{\bar{m}}{}_{\bar{n}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\delta_{\bar{\alpha}\bar{\beta}} \end{pmatrix}, \tag{A.8}$$

and the embedding tensor has the following non-vanishing components:

$$\theta_{\bar{M}\bar{N}\bar{P}\,\bar{0}} = -\frac{1}{\sqrt{2}}\, X_{\bar{M}\bar{N}\bar{P}}\,, \qquad \theta_{\bar{0}\bar{0}} = -4\sqrt{2}, \tag{A.9}$$

with

$$X_{\bar{m}\bar{n}\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}, \qquad X_{\bar{m}}{}^{\bar{n}\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}, \qquad X^{\bar{m}}{}_{\bar{n}}{}^{\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}, \qquad X^{\bar{m}\bar{n}}{}_{\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}. \tag{A.10}$$

The gauge group is then

$$(T^1)^4 \times [\mathrm{SO}(4) \ltimes (T^3 \times T^3)], \tag{A.11}$$

where T^n is a translational group transforming in the representation of dimension n of SO(4). A possible parametrisation of the scalar matrix is then

$$M_{\bar{M}\bar{N}} = \begin{pmatrix} m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\xi^2 & (\xi^2+\phi)m^{-1} & 0 & 0 & -\sqrt{2}\,[1 + (\xi^2+\phi)m^{-1}]\xi \\ m^{-1}(\xi^2-\phi) & m^{-1} & 0 & 0 & -\sqrt{2}\,m^{-1}\xi \\ 0 & 0 & e^{2\tilde{\varphi}} & 0 & 0 \\ 0 & 0 & 0 & e^{-2\tilde{\varphi}} & 0 \\ -\sqrt{2}\,\xi^T[1 + m^{-1}(\xi^2-\phi)] & -\sqrt{2}\,\xi^T m^{-1} & 0 & 0 & 1 + 2\,\xi^T m^{-1}\xi \end{pmatrix}, \tag{A.12}$$

in terms of a symmetric $\mathrm{GL}(3,\mathbb{R})$ matrix m, a 3×3 antisymmetric matrix ϕ, a 3×4 matrix ξ, its 3×3 square $\xi^2 = \xi\xi^T$ and a dilaton $\tilde{\varphi}$. This encodes 22 out the 32 scalars of the theory, 10 of the scalars being gauge fixed using the translations in the gauge group. With this parametrisation, the potential takes the form (2.5). We can further consistently restrict ourselves to a set of 13 scalars by requiring invariance under the diagonal SO(3) subgroup of $\mathrm{SO}(4)_{\text{global}}$ [51], yielding to the parametrisation described in eq. (2.3) and (2.4).

B Comparison with AIFeynman

We have used the state-of-the-art technique `AIFeynman` [43] on our dataset, with again $x_8 = \exp(\tilde{x}_8)$, so as to identify polynomial expressions fitting the data. We have run it with the following configuration

$$\begin{aligned} &\texttt{BF\_try\_time} = 60, \\ &\texttt{polyfit\_deg} = 5, \\ &\texttt{NN\_epochs} = 1000. \end{aligned} \tag{B.1}$$

The first parameter fixes the time limit in seconds for each brute force call, `polyfit_deg` gives the maximum degree of the polynomial tried by the polynomial fit routine and `NN_epochs` is the number of training epoch for the internal neural network. The function used for the brute force tests are

$$\texttt{+*-/><} \sim \texttt{\R1} \tag{B.2}$$

The binary operations are addition, multiplication, substraction and division. The unary ones are inverse, increment, decrement, negation and square root. Finally there is a nonary one, the unity. For more details we refer to [43].

In this algorithm, one tries to fit one of the variables in term of the others. Here, we used it to fit x_{10} in terms of the others. The AIFeynman algorithm finds a solution:

```
x_10 = 1.414213551821*(x_4-((x_1/x_2)-((x_1/x_2)/x_8))
```
(B.3)

which after identifying the numerical factor with $\sqrt{2}$ and inverting the relation gives

$$-\sqrt{2}x_1 + \sqrt{2}x_1x_8 + x_2x_8x_{10} - \sqrt{2}x_2x_4x_8 = 0 \tag{B.4}$$

which corresponds to p_1 of Eq. (5.12). While the AIFeynman code is able to recover one of the constraints, it exhibits several drawbacks compared to our method. First, it is significantly slower: it required 5685 seconds (1 hour 34 minutes and 45 seconds) compared to approximately 600 seconds for a single run of our ASMC algorithm [11]. Moreover, the AIFeynman framework is based on expressing one variable as a function of the others, which assumes the invertibility of the underlying relation. In general, this is not guaranteed for the class of polynomials in Eq. (5.12), and obtaining a closed-form inverse can be nontrivial or even impossible for higher-degree expressions. In our case, each polynomial involves at most quadratic terms in any single variable, ensuring invertibility, but this property would not hold for more complex models. In addition, AIFeynman is limited to recovering one constraint per run, whereas our ASMC method can discover multiple constraints simultaneously. This is reflected in our results, where each run identified an average of 2.2 polynomials, with up to 4 found in the best case. Finally, the brute-force regression strategy used by AIFeynman makes it less effective for identifying higher-degree polynomials, whose complexity increases the search difficulty. In contrast, our method treats all polynomials as equally probable, regardless of their degree, enabling it to uncover more intricate structures more efficiently [12].

References

[1] J. M. Maldacena, *The Large N limit of superconformal field theories and supergravity*, *Adv. Theor. Math. Phys.* **2** (1998) 231–252, [hep-th/9711200].

[2] H. Ooguri and C. Vafa, *Non-supersymmetric AdS and the Swampland*, *Adv. Theor. Math. Phys.* **21** (2017) 1787–1801, [arXiv:1610.01533].

[3] E. Palti, *The Swampland: Introduction and Review*, *Fortsch. Phys.* **67** (2019), no. 6 1900037, [arXiv:1903.06239].

[4] A. Giambrone, A. Guarino, E. Malek, H. Samtleben, C. Sterckx, and M. Trigiante, *Holographic evidence for nonsupersymmetric conformal manifolds*, *Phys. Rev. D* **105** (2022), no. 6 066018, [arXiv:2112.11966].

[5] S. Chaudhuri and J. A. Schwartz, *A criterion for integrably marginal operators*, *Phys. Lett. B* **219** (1989) 291–296.

[11] This time only accounts for the annealing loop, without the local search algorithm described in section 5.3.2. Furthermore, this computational time reflects the overhead incurred by verifying at each iteration whether the particles reproduce one of the polynomials specified in Eq. (5.12), which substantially increases the overall execution time. In the absence of this verification step, the annealing loop would require approximately 350 seconds on standard computational hardware.

[12] Note that we do not claim that we have used the AIFeynman in the most optimal way.

[6] O. Aharony, M. Berkooz, and E. Silverstein, *Nonlocal string theories on AdS(3) x S**3 and stable nonsupersymmetric backgrounds*, *Phys. Rev. D* **65** (2002) 106007, [hep-th/0112178].

[7] X. Dong, D. Z. Freedman, and Y. Zhao, *Explicitly Broken Supersymmetry with Exactly Massless Moduli*, *JHEP* **06** (2016) 090, [arXiv:1410.2257].

[8] C. Eloy, M. Galli, and E. Malek, *Adding fluxes to consistent truncations: IIB supergravity on $AdS_3\times S^3\times S^3\times S^1$*, *JHEP* **11** (2023) 049, [arXiv:2306.12487].

[9] C. Eloy and G. Larios, *Nonsupersymmetric stable marginal deformations in AdS3/CFT2*, *Phys. Rev. D* **108** (2023), no. 12 L121901, [arXiv:2309.03261].

[10] C. Eloy and G. Larios, *Charting the Conformal Manifold of Holographic CFT_2's*, arXiv:2405.17542.

[11] D. Gepner and E. Witten, *String Theory on Group Manifolds*, *Nucl. Phys. B* **278** (1986) 493–549.

[12] L. Eberhardt, M. R. Gaberdiel, and W. Li, *A holographic dual for string theory on $AdS_3 \times S^3 \times S^3 \times S^1$*, *JHEP* **08** (2017) 111, [arXiv:1707.02705].

[13] L. Eberhardt, M. R. Gaberdiel, and R. Gopakumar, *The Worldsheet Dual of the Symmetric Product CFT*, *JHEP* **04** (2019) 103, [arXiv:1812.01007].

[14] I. M. Comsa, M. Firsching, and T. Fischbacher, *SO(8) Supergravity and the Magic of Machine Learning*, *JHEP* **08** (2019) 057, [arXiv:1906.00207].

[15] D. Berman, T. Fischbacher, G. Inverso, B. Scellier, and B. Scellier, *Vacua of ω-deformed SO(8) supergravity*, *JHEP* **06** (2022) 133, [arXiv:2201.04173].

[16] A. Ashmore, Y.-H. He, and B. A. Ovrut, *Machine Learning Calabi–Yau Metrics*, *Fortsch. Phys.* **68** (2020), no. 9 2000068, [arXiv:1910.08605].

[17] D. S. Berman, Y.-H. He, and E. Hirst, *Machine learning Calabi-Yau hypersurfaces*, *Phys. Rev. D* **105** (2022), no. 6 066002, [arXiv:2112.06350].

[18] M. Larfors, A. Lukas, F. Ruehle, and R. Schneider, *Numerical metrics for complete intersection and Kreuzer–Skarke Calabi–Yau manifolds*, *Mach. Learn. Sci. Tech.* **3** (2022), no. 3 035014, [arXiv:2205.13408].

[19] P. Berglund, G. Butbaia, T. Hüubsch, V. Jejjala, D. Mayorga Peña, C. Mishra, and J. Tan, *Machine-learned Calabi–Yau metrics and curvature*, *Adv. Theor. Math. Phys.* **27** (2023), no. 4 1107–1158, [arXiv:2211.09801].

[20] V. Jejjala, D. K. Mayorga Pena, and C. Mishra, *Neural network approximations for Calabi-Yau metrics*, *JHEP* **08** (2022) 105, [arXiv:2012.15821].

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

[22] M. Larfors, A. Lukas, F. Ruehle, and R. Schneider, *Learning Size and Shape of Calabi-Yau Spaces*, arXiv:2111.01436.

[23] Y.-H. He, *The Calabi–Yau Landscape: From Geometry, to Physics, to Machine Learning*. Lecture Notes in Mathematics. 5, 2021.

[24] Y.-H. He, *Deep-Learning the Landscape*, arXiv:1706.02714.

[25] J. Carifio, J. Halverson, D. Krioukov, and B. D. Nelson, *Machine Learning in the String Landscape*, *JHEP* **09** (2017) 157, [arXiv:1707.00655].

[26] F. Ruehle, *Evolving neural networks with genetic algorithms to study the String Landscape*, *JHEP* **08** (2017) 038, [arXiv:1706.07024].

[27] H.-Y. Chen, Y.-H. He, S. Lal, and M. Z. Zaz, *Machine Learning Etudes in Conformal Field Theories*, arXiv:2006.16114.

[28] N. Brady, D. Tennyson, and T. Vandermeulen, *Machine Learning the 6d Supergravity Landscape*, arXiv:2505.16131.

[29] C. Krishnan, V. Mohan, and S. Ray, *Machine Learning $\mathcal{N} = 8, D = 5$ Gauged Supergravity*, *Fortsch. Phys.* **68** (2020), no. 5 2000027, [arXiv:2002.12927].

[30] K. Hashimoto, S. Sugishita, A. Tanaka, and A. Tomiya, *Deep learning and the AdS/CFT correspondence*, *Phys. Rev. D* **98** (2018), no. 4 046019, [arXiv:1802.08313].

[31] F. Ruehle, *Data science applications to string theory*, *Phys. Rept.* **839** (2020) 1–117.

[32] Y.-H. He, E. Heyes, and E. Hirst, *Machine learning in physics and geometry*, *Handbook of Statistics* **49** (2023) 47–81, [arXiv:2303.12626].

[33] J. Bao, Y.-H. He, E. Hirst, J. Hofscheier, A. Kasprzyk, and S. Majumder, *Hilbert series, machine learning, and applications to physics*, *Phys. Lett. B* **827** (2022) 136966, [arXiv:2103.13436].

[34] J. R. Koza, *Genetic programming as a means for programming computers by natural selection*, *Statistics and computing* **4** (1994) 87–112.

[35] M. Virgolin, T. Alderliesten, C. Witteveen, and P. A. Bosman, *Improving model-based genetic programming for symbolic regression of small expressions*, *Evolutionary computation* **29** (2021), no. 2 211–237.

[36] D. L. Randall, T. S. Townsend, J. D. Hochhalter, and G. F. Bomarito, *Bingo: a customizable framework for symbolic regression with genetic programming*, in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 2282–2288, 2022.

[37] B. Burlacu, G. Kronberger, M. Kommenda, and M. Affenzeller, *Parsimony measures in multi-objective genetic programming for symbolic regression*, in *Proceedings of the genetic and evolutionary computation conference companion*, pp. 338–339, 2019.

[38] B. K. Petersen, M. Landajuela, T. N. Mundhenk, C. P. Santiago, S. K. Kim, and J. T. Kim, *Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients*, *arXiv preprint arXiv:1912.04871* (2019).

[39] P.-A. Kamienny, S. d'Ascoli, G. Lample, and F. Charton, *End-to-end symbolic regression with transformers*, *Advances in Neural Information Processing Systems* **35** (2022) 10269–10281.

[40] M. Valipour, B. You, M. Panju, and A. Ghodsi, *Symbolicgpt: A generative transformer model for symbolic regression*, *arXiv preprint arXiv:2106.14131* (2021).

[41] Z. Bastiani, R. M. Kirby, J. Hochhalter, and S. Zhe, *Diffusion-based symbolic regression*, *arXiv preprint arXiv:2505.24776* (2025).

[42] S. S. Sahoo, C. H. Lampert, and G. Martius, *Learning Equations for Extrapolation and Control*, *arXiv e-prints* (June, 2018) arXiv:1806.07259, [arXiv:1806.07259].

[43] S.-M. Udrescu and M. Tegmark, *AI Feynman: a Physics-Inspired Method for Symbolic Regression*, *Sci. Adv.* **6** (2020), no. 16 eaay2631, [arXiv:1905.11481].

[44] R. M. Neal, *Annealed importance sampling*, physics/9803008.

[45] P. Del Moral, A. Doucet, and A. Jasra, *Sequential monte carlo samplers*, *Journal of the Royal Statistical Society Series B: Statistical Methodology* **68** (2006), no. 3 411–436.

[46] H. Nicolai and H. Samtleben, *N=8 matter coupled AdS(3) supergravities*, *Phys. Lett. B* **514** (2001) 165–172, [hep-th/0106153].

[47] B. de Wit, I. Herger, and H. Samtleben, *Gauged locally supersymmetric D = 3 nonlinear sigma models*, *Nucl. Phys. B* **671** (2003) 175–216, [hep-th/0307006].

[48] M. Cvetic, H. Lu, and C. N. Pope, *Consistent Kaluza-Klein sphere reductions*, *Phys. Rev. D* **62** (2000) 064028, [hep-th/0003286].

[49] N. S. Deger, H. Samtleben, O. Sarioglu, and D. Van den Bleeken, *A supersymmetric reduction on the three-sphere*, *Nucl. Phys. B* **890** (2014) 350–362, [arXiv:1410.7168].

[50] O. Hohm, E. T. Musaev, and H. Samtleben, *O(d + 1, d + 1) enhanced double field theory*, *JHEP* **10** (2017) 086, [arXiv:1707.06693].

[51] C. Eloy, G. Larios, and H. Samtleben, *Triality and the consistent reductions on $AdS_3 \times S^3$*, *JHEP* **01** (2022) 055, [arXiv:2111.01167].

[52] Wolfram Research Inc., "Mathematica, Version 14.2." Champaign, IL, 2024.

[53] C. A. Naesseth, F. Lindsten, and T. B. Schön, *Elements of sequential monte carlo*, 2024.

[54] T. Kloek and H. K. van Dijk, *Bayesian estimates of equation system parameters: An application of integration by monte carlo*, *Econometrica* **46** (1978), no. 1 1–19.

[55] J. S. Liu and R. Chen, *Sequential monte carlo methods for dynamic systems*, *Journal of the American Statistical Association* **93** (1998), no. 443 1032–1044, [https://doi.org/10.1080/01621459.1998.10473765].

[56] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, *TensorFlow: Large-scale machine learning on heterogeneous systems*, 2015. Software available from tensorflow.org.

[57] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, arXiv:1412.6980.

[58] I. Loshchilov and F. Hutter, *Sgdr: Stochastic gradient descent with warm restarts*, arXiv:1608.03983.

[59] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine learning in Python*, *Journal of Machine Learning Research* **12** (2011) 2825–2830.

[60] R. J. G. B. Campello, D. Moulavi, and J. Sander, *Density-based clustering based on hierarchical density estimates*, in *Advances in Knowledge Discovery and Data Mining* (J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu, eds.), (Berlin, Heidelberg), pp. 160–172, Springer Berlin Heidelberg, 2013.

[61] P. Breitenlohner and D. Z. Freedman, *Stability in Gauged Extended Supergravity*, *Annals Phys.* **144** (1982) 249.

[62] O. Hohm and H. Samtleben, *Exceptional field theory. III. $E_{8(8)}$*, *Phys. Rev. D* **90** (2014) 066002, [arXiv:1406.3348].

[63] O. Hohm and H. Samtleben, *Consistent Kaluza-Klein Truncations via Exceptional Field Theory*, *JHEP* **01** (2015) 131, [arXiv:1410.8145].

[64] E. Malek and H. Samtleben, *Kaluza-Klein Spectrometry for Supergravity*, *Phys. Rev. Lett.* **124** (2020), no. 10 101601, [arXiv:1911.12640].

[65] E. Malek and H. Samtleben, *Kaluza-Klein Spectrometry from Exceptional Field Theory*, *Phys. Rev. D* **102** (2020), no. 10 106016, [arXiv:2009.03347].

[66] C. Eloy, *Kaluza-Klein spectrometry for* AdS_3 *vacua*, *SciPost Phys.* **10** (2021), no. 6 131, [arXiv:2011.11658].

[67] C. Eloy, *Supergravity : AdS_3 vacua and higher-derivative corrections.* PhD thesis, Lyon, Ecole Normale Superieure, Université de Lyon, 2021.

[68] H. Nicolai and H. Samtleben, *Maximal gauged supergravity in three-dimensions*, *Phys. Rev. Lett.* **86** (2001) 1686–1689, [hep-th/0010076].

[69] B. de Wit, H. Samtleben, and M. Trigiante, *On Lagrangians and gaugings of maximal supergravities*, *Nucl. Phys. B* **655** (2003) 93–126, [hep-th/0212239].

[70] H. Samtleben and Ö. Sarıoglu, *Consistent S^3 reductions of six-dimensional supergravity*, *Phys. Rev. D* **100** (2019), no. 8 086002, [arXiv:1907.08413].

[71] J. Schon and M. Weidner, *Gauged N=4 supergravities*, *JHEP* **05** (2006) 034, [hep-th/0602024].