MI-HET-???
May 26, 2025

Notes: Machine learning flat directions

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

...

Contents

1 Introduction

The AdS/CFT correspondence stands as one of the most profound dualities in theoretical physics, establishing a remarkable equivalence between gravitational theories in Anti-de Sitter (AdS) spacetimes and conformal field theories (CFTs) living on their boundaries. 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. On the CFT side, the dual conformal field theories often exhibit rich structures of exactly marginal operators whose coupling constants parametrize conformal manifolds—moduli spaces where the theory remains conformal but with varying coupling strengths. The AdS/CFT correspondence maps these conformal manifolds to flat directions in the supergravity scalar potential, where the cosmological constant remains fixed while scalar field configurations vary continuously.

Understanding these flat directions is therefore of great importance for several interconnected reasons. First, they encode the conformal manifolds of the dual CFTs, providing direct access to the space of exactly marginal deformations and their associated beta functions. Second, they reveal the vacuum structure of supergravity theories, determining which background configurations preserve supersymmetry and their associated stability properties.

However, the explicit characterization of these 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 derivatives vanish—quickly become too complex for traditional symbolic manipulation, rendering analytical approaches computationally intractable. This complexity paradox is particularly frustrating given that physically meaningful solutions often admit relatively simple parametric descriptions once discovered.

The emergence of machine learning techniques in theoretical physics offers new avenues for tackling such problems. Rather than attempting to solve the full symbolic system directly, we can leverage numerical methods to sample the solution space and then employ symbolic regression to extract analytical patterns from the resulting data. This hybrid approach potentially circumvents the computational bottlenecks that plague purely symbolic methods while maintaining the goal of obtaining exact analytical expressions.

In this work, we demonstrate the viability of this machine learning approach by applying it to a five-scalar subsector of a 13 scalar consistent truncation of three-dimensional chiral $\mathcal{N} = (2, 0)$ supergravity arising from $AdS_3 \times S^3$ compactification from 6d supergravity, or from the compactification of type IIB supergravity on $AdS_3 \times S^3 \times T^4$. Our methodology combines gradient descent sampling of the flat direction manifold with a novel annealed importance sampling algorithm for polynomial symbolic regression. We show that this approach successfully identifies the underlying three-dimensional conformal manifold and automatically discovers the polynomial constraints that characterize it analytically.

The paper is organized as follows. Section 2 establishes the supergravity setup, presenting the scalar potential in its full complexity and motivating the truncation to five fields. Section 3 details our numerical methodology, covering both the gradient descent sampling procedure and the annealed importance sampling approach to symbolic regression. Section 4 analyzes the discovered supergravity solutions, including their geometric interpretation and spectrum analysis. Finally, Section 5 concludes 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

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

which comprises the Einstein-Hilbert term R, the kinetic term for scalar fields parametrized by the matrix $M_{\bar{M}\bar{N}}$, a Chern-Simons contribution $\mathcal{L}_{\rm CS}$, and a scalar potential V.

The gauging structure is encoded in an embedding tensor that takes the general form

$$\Theta_{\bar{K}\bar{L}|\bar{M}\bar{N}} = \theta_{\bar{K}\bar{L}\bar{M}\bar{N}} + \frac{1}{2}\Big(\eta_{\bar{M}[\bar{K}}\theta_{\bar{L}]\bar{N}} - \eta_{\bar{N}[\bar{K}}\theta_{\bar{L}]\bar{M}}\Big) + \theta\, \eta_{\bar{M}[\bar{K}}\eta_{\bar{L}]\bar{N}} \,. \tag{2.2}$$

where we have a fully antisymmetric component $\theta_{\bar{K}\bar{L}\bar{M}\bar{N}} = \theta_{[\bar{K}\bar{L}\bar{M}\bar{N}]}$, a symmetric traceless part $\theta_{\bar{L}\bar{K}} = \theta_{(\bar{L}\bar{K})}$, and a constant parameter θ. The metric $\eta_{\bar{K}\bar{L}}$ is the SO(8,4)-invariant bilinear form used for index contractions.

The scalar degrees of freedom parametrize the coset space

$$\frac{\mathrm{SO}(8,4)}{\mathrm{SO}(8)\times\mathrm{SO}(4)} \,, \tag{2.3}$$

through the symmetric matrix $M_{\bar{K}\bar{L}} = \mathcal{V}_{\bar{K}}{}^{\bar{M}}\mathcal{V}_{\bar{L}}{}^{\bar{N}}\delta_{\bar{M}\bar{N}}$, where $\mathcal{V}_{\bar{K}}{}^{\bar{M}}$ represents the coset representative.

The gauge covariant derivatives are constructed using the embedding tensor according to

$$D_\mu = \partial_\mu + A_\mu{}^{\bar{M}\bar{N}}\, \Theta_{\bar{M}\bar{N}|\bar{P}\bar{Q}}\, T^{\bar{P}\bar{Q}} \,, \tag{2.4}$$

where $A_\mu{}^{\bar{M}\bar{N}}$ are the gauge fields and $T^{\bar{P}\bar{Q}}$ are the generators of the SO(8,4) algebra satisfying

$$\big(T^{\bar{M}\bar{N}}\big)_{\bar{P}}{}^{\bar{Q}} = 2\,\delta_{\bar{P}}{}^{[\bar{M}}\,\eta^{\bar{N}]\bar{Q}} \,. \tag{2.5}$$

The covariant derivative acting on the scalar matrix becomes

$$D_\mu M_{\bar{M}\bar{N}} = \partial_\mu M_{\bar{M}\bar{N}} + 4\, A_\mu{}^{\bar{P}\bar{Q}}\, \Theta_{\bar{P}\bar{Q}|(\bar{M}}{}^{\bar{K}}\, M_{\bar{N})\bar{K}}\,, \tag{2.6}$$

ensuring gauge invariance of the scalar kinetic terms.

The Chern-Simons contribution is given by

$$\mathcal{L}_{\rm CS} = -\varepsilon^{\,\mu\nu\rho}\, \Theta_{\bar{M}\bar{N}|\bar{P}\bar{Q}}\, A_\mu{}^{\bar{M}\bar{N}} \left(\partial_\nu\, A_\rho{}^{\bar{P}\bar{Q}} + \frac{1}{3}\, \Theta_{\bar{R}\bar{S}|\bar{U}\bar{V}}\, f^{\bar{P}\bar{Q},\bar{R}\bar{S}}{}_{\bar{X}\bar{Y}}\, A_\nu{}^{\bar{U}\bar{V}}\, A_\rho{}^{\bar{X}\bar{Y}} \right)\,, \tag{2.7}$$

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 the SO(8,4) Lie algebra, and $\varepsilon^{\mu\nu\rho}$ is the three-dimensional Levi-Civita tensor.

The scalar potential can be expressed in terms of the embedding tensor components as

$$\begin{aligned} V = \frac{1}{12}\,\theta_{\bar{K}\bar{L}\bar{M}\bar{N}}\theta_{\bar{P}\bar{Q}\bar{R}\bar{S}} \Big(& M^{\bar{K}\bar{P}} M^{\bar{L}\bar{Q}} M^{\bar{M}\bar{R}} M^{\bar{N}\bar{S}} - 6\, M^{\bar{K}\bar{P}} M^{\bar{L}\bar{Q}} \eta^{\bar{M}\bar{R}} \eta^{\bar{N}\bar{S}} \\ & + 8\, M^{\bar{K}\bar{P}} \eta^{\bar{L}\bar{Q}} \eta^{\bar{M}\bar{R}} \eta^{\bar{N}\bar{S}} - 3\, \eta^{\bar{K}\bar{P}} \eta^{\bar{L}\bar{Q}} \eta^{\bar{M}\bar{R}} \eta^{\bar{N}\bar{S}} \Big) \\ + \frac{1}{8}\,\theta_{\bar{K}\bar{L}}\theta_{\bar{P}\bar{Q}} \Big(& 2\, M^{\bar{K}\bar{P}} M^{\bar{L}\bar{Q}} - 2\, \eta^{\bar{K}\bar{P}} \eta^{\bar{L}\bar{Q}} - M^{\bar{K}\bar{L}} M^{\bar{P}\bar{Q}} \Big) + 4\,\theta\theta_{\bar{K}\bar{L}} M^{\bar{K}\bar{L}} - 32\,\theta^2\,. \end{aligned} \tag{2.8}$$

In order to describe the $\mathcal{N} = (2,0)$ theory, it is useful to break the SO(8,4) group as

$$\begin{aligned} \mathrm{SO}(8,4) &\longrightarrow \ \mathrm{GL}(3,\mathbb{R}) \times \mathrm{SO}(1,1) \times \mathrm{SO}(4)_{\rm global}\,, \\ X^{\bar{M}} &\longrightarrow \ \ \{X^{\bar{m}},\ X_{\bar{m}},\ X^{\bar{0}},\ X_{\bar{0}},\ X^{\bar{\alpha}}\}\,, \end{aligned} \tag{2.9}$$

where $\bar{m} \in [\![1,3]\!]$ and $\bar{\alpha} \in [\![9,12]\!]$ label the SL(3, $\mathbb{R}$) and $\mathrm{SO}(4)_{\rm global}$ vector representations, respectively. In this basis, the SO(8,4)-invariant tensor reads

$$\eta_{\bar{M}\bar{N}} = \begin{pmatrix} 0 & \delta_{\bar{m}}{}^{\bar{n}} & 0 & 0 & 0 \\ \delta^{\bar{m}}{}_{\bar{n}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\delta_{\bar{\alpha}\bar{\beta}} \end{pmatrix}. \tag{2.10}$$

The embedding tensor have the the following expression

$$\theta_{\bar{m}\bar{n}\bar{p}\,\bar{0}} = -2\,\lambda\,\varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \qquad \theta_{\bar{m}\bar{n}\bar{p}}{}^{\bar{0}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \qquad \theta_{\bar{m}\bar{n}}{}^{\bar{p}}{}_{\bar{0}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \tag{2.11}$$

or equivalently

$$X_{\bar{m}\bar{n}\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \qquad X_{\bar{m}}{}^{\bar{n}\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \qquad X^{\bar{m}}{}_{\bar{n}}{}^{\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \qquad X^{\bar{m}\bar{n}}{}_{\bar{p}} = \varepsilon_{\bar{m}\bar{n}\bar{p}}\,, \tag{2.12}$$

Parametrizing the scalar matrix following [1], we have for the scalar matrix

$$M_{\bar{M}\bar{N}} = \begin{pmatrix} m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\xi^2 & (\xi^2+\phi)m^{-1} & 0 & 0 & -\sqrt{2}\,[1+(\xi^2+\phi)m^{-1}]\xi \\ m^{-1}(\xi^2-\phi) & m^{-1} & 0 & 0 & -\sqrt{2}\,m^{-1}\xi \\ 0 & 0 & e^{2\tilde{\varphi}} & 0 & 0 \\ 0 & 0 & 0 & e^{-2\tilde{\varphi}} & 0 \\ -\sqrt{2}\,\xi^T[1+m^{-1}(\xi^2-\phi)] & -\sqrt{2}\,\xi^T m^{-1} & 0 & 0 & 1+2\,\xi^T m^{-1}\xi \end{pmatrix}, \tag{2.13}$$

which encodes 22 scalars of the theory ($22 = 32 - 10$, with 10 scalars gauge fixed using translations in the gauge group), which have been parametrized by $m = \nu\nu^T \in \mathrm{GL}(3,\mathbb{R})$ parametrizing the coset $\mathrm{GL}(3,\mathbb{R})/\mathrm{SO}(3)$, ϕ a 3×3 antisymmetric matrix,ξ a 3×4 matrix, and $\xi^2 = \xi\xi^T$, a dilaton $\tilde{\varphi}$. With this parametrization, the potential takes the from ref. [1]:

$$\begin{aligned} V = 4\,e^{-4\tilde{\varphi}} + 2\,e^{-2\tilde{\varphi}}\Big[& -\mathrm{tr}\left(m+m^{-1}\right) + \mathrm{tr}\left(\phi m^{-1}\phi\right) - 2\ \mathrm{tr}\left(\phi m^{-1}\xi^2\right) - 2\ \mathrm{tr}\left(\xi^2\right) \\ & - \mathrm{tr}\left(\xi^2 m^{-1}\xi^2\right) + \frac{1}{2}\ \det\left(m^{-1}\right)\left(1 - \mathrm{tr}\left(\phi^2\right) - \mathrm{tr}\left(\xi^4\right) + \mathrm{tr}\left(\xi^2\right)^2\right) \\ & + \frac{1}{2}\,\mathrm{T}\left(m^{-1}(\xi^2-\phi), (\xi^2+\phi)m^{-1}, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2\right) \\ & + \frac{1}{4}\,\mathrm{T}\left(m^{-1}, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2, m + (\xi^2+\phi)m^{-1}(\xi^2-\phi) + 2\,\xi^2\right)\Big], \end{aligned} \tag{2.14}$$

where $\mathrm{T}\,(A,B,C) = \varepsilon_{mnp}\,\varepsilon_{qrs}\,A^{mq}B^{nr}C^{ps}$. We can further restrict ourselves to a set of 13 scalars which is still a consistent truncation, with

$$\begin{aligned} \xi &= \begin{pmatrix} 0 & 0 & 0 & x_1 \\ 0 & 0 & 0 & x_2 \\ 0 & 0 & 0 & x_3 \end{pmatrix}, \\ \phi &= \begin{pmatrix} 0 & x_4 & x_5 \\ -x_4 & 0 & x_6 \\ -x_5 & -x_6 & 0 \end{pmatrix}, \\ \tilde{\varphi} &= \tilde{x}_{13}, \\ \nu &= e^{(6\,\tilde{x}_7 + 3\,\tilde{x}_8 + \sqrt{3}\,\tilde{x}_9)/6} \begin{pmatrix} 1 & \frac{x_{10}}{\sqrt{2}} & \frac{x_{11}}{\sqrt{2}} + \frac{x_{10}x_{12}}{4} \\ 0 & e^{-\tilde{x}_8} & \frac{e^{-\tilde{x}_8}\,x_{12}}{\sqrt{2}} \\ 0 & 0 & e^{-(\tilde{x}_8 + \sqrt{3}\,\tilde{x}_9)/2} \end{pmatrix}. \end{aligned} \tag{2.15}$$

As an initial simplification, we restrict our analysis to the scalar fields $x_1, x_2, x_4, \tilde{x}_8$, and x_{10}. This choice is made to facilitate a more tractable starting point for the investigation, and corresponds to a randomly selected subset of the original set of thirteen scalar fields. It is important to note that this subset no longer defines a consistent truncation of the full theory. Nevertheless, despite the fact that the effective theory defined by these five scalars may be formally ill-defined—due to potential higher-order interactions that could source the remaining scalar fields—the solutions we obtain remain valid solutions of the complete theory. This is because we compute the critical points of the scalar potential by first evaluating $\nabla V = 0$, and only then setting the remaining scalar fields to zero. By adopting this procedure—taking the gradient of the potential prior to truncation—we ensure that the resulting configurations satisfy the equations of

motion and are thus genuine solutions of the full theory. Under this truncation, the scalar potential (2.14) reduces to:

$$
\begin{aligned}
V = \frac{1}{8}\, e^{-2\,\tilde{x}_8} \Big[& 4 + 4\,x_1^4 + e^{4\,\tilde{x}_8}\big(2 + x_{10}^2\big)^2\big(1 + x_2^4\big) \\
& - 4\, e^{3\,\tilde{x}_8}\,\big(2 + x_{10}^2\big)\Big(2 - x_1^2 + \sqrt{2}\,x_1 x_{10} x_2^3 + x_2^4 + 2\,x_1 x_2 x_4 + x_4^2 - x_2^2\big(1 - x_1^2 + x_4^2\big)\Big) \\
& - 8\, e^{\tilde{x}_8}\,\Big(2 + x_1^4 + \sqrt{2}\,x_1^3 x_{10} x_2 - x_2^2 - 2\,x_1 x_2 x_4 + x_4^2 - x_1^2\big(1 - x_2^2 + x_4^2\big)\Big) \\
& + 4\, e^{2\,\tilde{x}_8}\,\Big(2 - 4\,x_1^2 + x_1^4 - 4\,x_2^2 + 4\,x_1^2 x_2^2 + x_2^4 + x_{10}^2\big(1 + 3\,x_1^2 x_2^2\big) + 4\,x_4^2 + x_4^4\Big) \\
& + 8\sqrt{2}\,x_{10}\, e^{2\,\tilde{x}_8}\,\Big(x_1^3 x_2 + x_1^2 x_4 - x_2^2 x_4 + x_1\big(x_2^3 - x_2 x_4^2\big)\Big)\Big].
\end{aligned}
\tag{2.16}
$$

Let us make a comment at this point. Looking at the potential given in Eq. 2.16 and since we have an explicit formula, one might argue that the problem of identifying flat directions could be addressed by directly computing the gradient of the expression and subsequently simplifying the resulting equations. However, attempting to carry out this procedure using a symbolic solver such as Mathematica quickly reveals its limitations: the resulting expressions are too complex 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. In this work, we aim to identify such solutions, which we seek using numerical methods.

An additional advantage of this approach is its broader applicability. In scenarios where an explicit analytic expression for the potential is inaccessible—yet numerical evaluation remains feasible—our method retains its validity. Hence, the numerical strategy adopted here may prove effective even in cases where an exact analytic expression for the potential is out of reach.

3 Numerical analysis

3.1 Gradient descent and local analysis

To identify the flat directions in the potential, we will use numerical tools. The procedure is as follows: first, we sample the underlying manifold, and then we use numerical techniques to extract analytical information from the resulting cloud of points. The first step is thus to sample the manifold.

To do so, we perform a basic gradient descent. We begin by randomly and uniformly initializing points within a hypercube of range $[-2, 2]$. This choice is important to ensure that points are not restricted to the inner range $[-1, 1]$. For this example, we choose to generate 10^5 points. This number is motivated by the dimensionality of our problem. If we aim to populate all possible directions and want approximately $\mathcal{O}(10)$ points per direction, then we require $\mathcal{O}(10^5)$ points. Given that the true intrinsic dimensionality of the manifold is less than or equal to 5, this value serves as a conservative upper bound for the number of points needed to adequately sample the manifold.

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

$$
\mathcal{L} = \sum_{i=1}^{n_{points}} ||\nabla V(\vec{x}^{,i})||^2 \tag{3.1}
$$

where $\vec{x}^{\,i} = (x_1^i, x_2^i, x_4^i, \tilde{x}_8^i, x_{10}^i)$ denotes the data points.

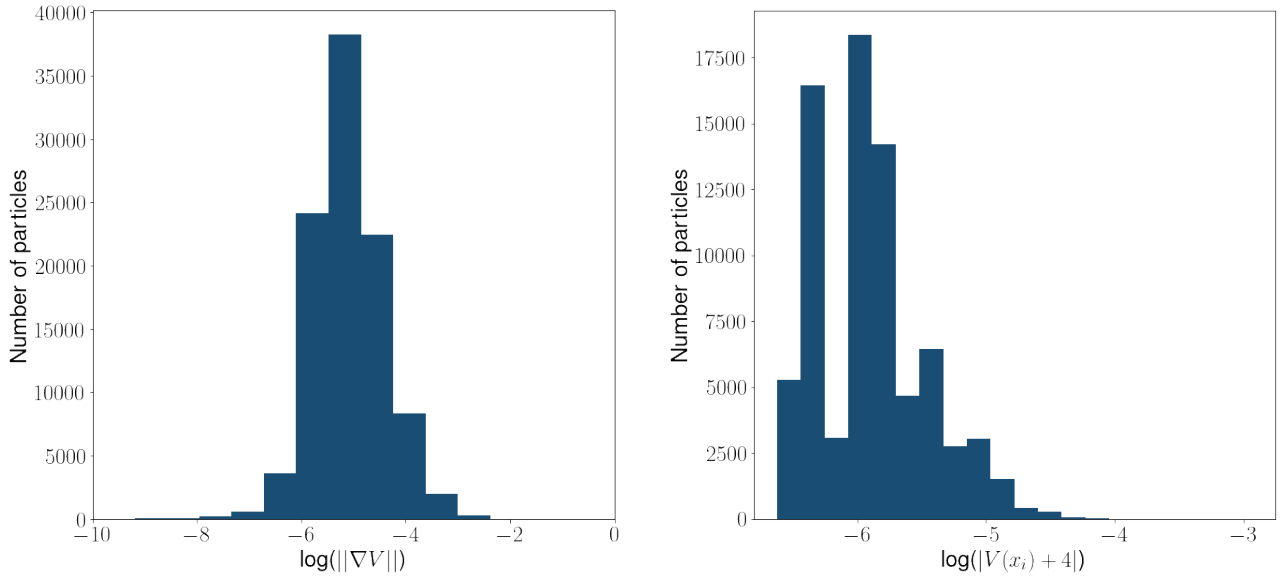

Fig. 1 Left panel: Histogram of the logarithm of the norm of the gradient for each point. Right panel: Histogram of the distance of the value of the potential for each point.

Through trial and error, we observed that periodically reinitializing the optimizer significantly improved the convergence rate. This technique bears similarity to the concept of warm restarts introduced in [2]. 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 reinitialized, 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.

The evolution of the loss function, along with the learning rate schedule, is shown in Fig. 2. As can be seen in this figure, the convergence rate improves significantly each time the optimizer is reinitialized. We also observe that, following the last few reinitializations, the loss exhibits a small bump immediately after the restart.

We interpret this behavior as a consequence of the increased learning rate: some points that previously had low loss values may momentarily worsen before benefiting from faster convergence. This effect 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.

Although this strategy introduces the risk of desampling certain regions in favor of others, we did not observe any such issue in the remainder of our analysis.

The Adam optimizer was employed throughout the gradient descent procedure. It was reinitialized at iterations 250, 500, 750, with the learning rate fixed at 10^{-2}. At iteration 1000, the optimizer was reinitialized once more, this time with a reduced learning rate of 10^{-3}. A final reinitialization was performed at iteration 1500, setting the learning rate to 10^{-4}, and the optimization was continued for an additional 1000 epochs.

Upon completion, it was observed that all data points converged to values close to -4. Specifically, nearly all data points attained values within an absolute error of 10^{-4}, with the exception of four points whose deviations were on the order of 10^{-3}. Similarly, 99,739 out of 100,000 data points exhibited gradient

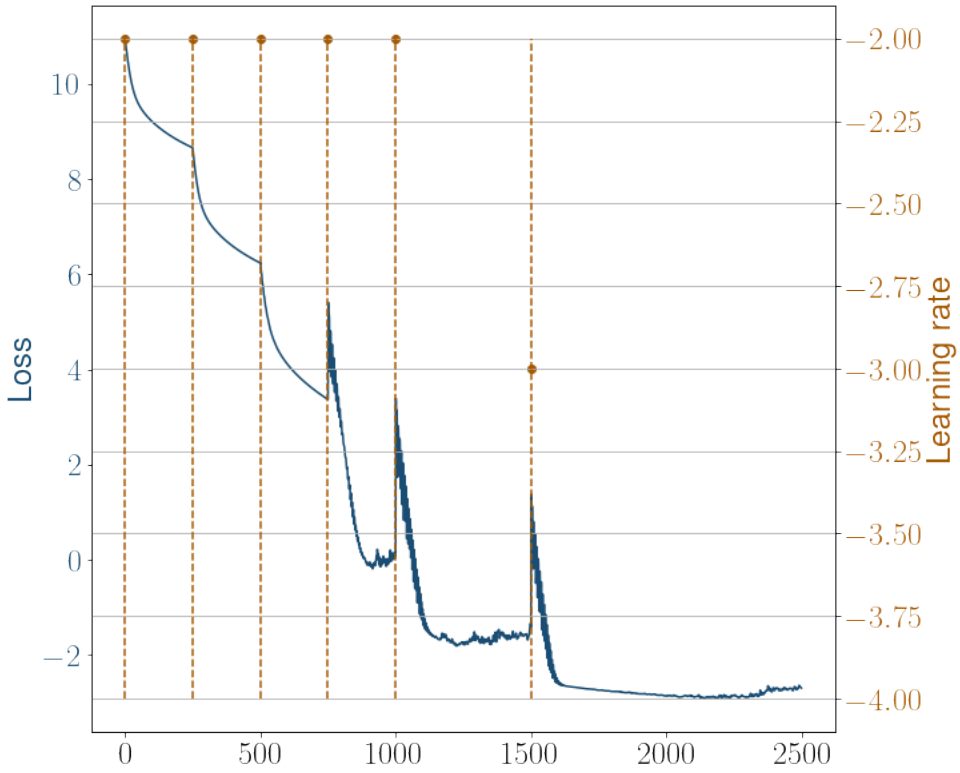

Fig. 2 Evolution of the loss function during gradient descent and corresponding learning rate schedule, both plotted on a logarithmic scale. Dashed lines indicate epochs at which the optimizer was reinitialized.

norms smaller than 10^{-3}. These results are summarized in Figure 1.

As a first visualization, we present a triangular plot of the data in Fig. 3. This figure shows all possible 2D projections of the data, along with the 1D histograms of each coordinate after gradient descent.

We can make a few comments on those plots. First, we observe non-trivial correlations in the data. Some of these may result from larger basins of attraction, such as in the x_1/x_2 plot. However, the structures seen in the $x_4/\tilde{x}_8$ or x_4/x_{10} plots likely reflect genuine features of the manifold, and will show latter that this is indeed the case. We also note that all directions appear to be well populated.

Once the gradient descent has been completed and the flat direction 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 can first perform some exploratory analyses to better understand the data. Specifically, we aim to determine the dimensionality of the manifold and wether 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). 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.

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. 4.

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 a dimensionality of 2.

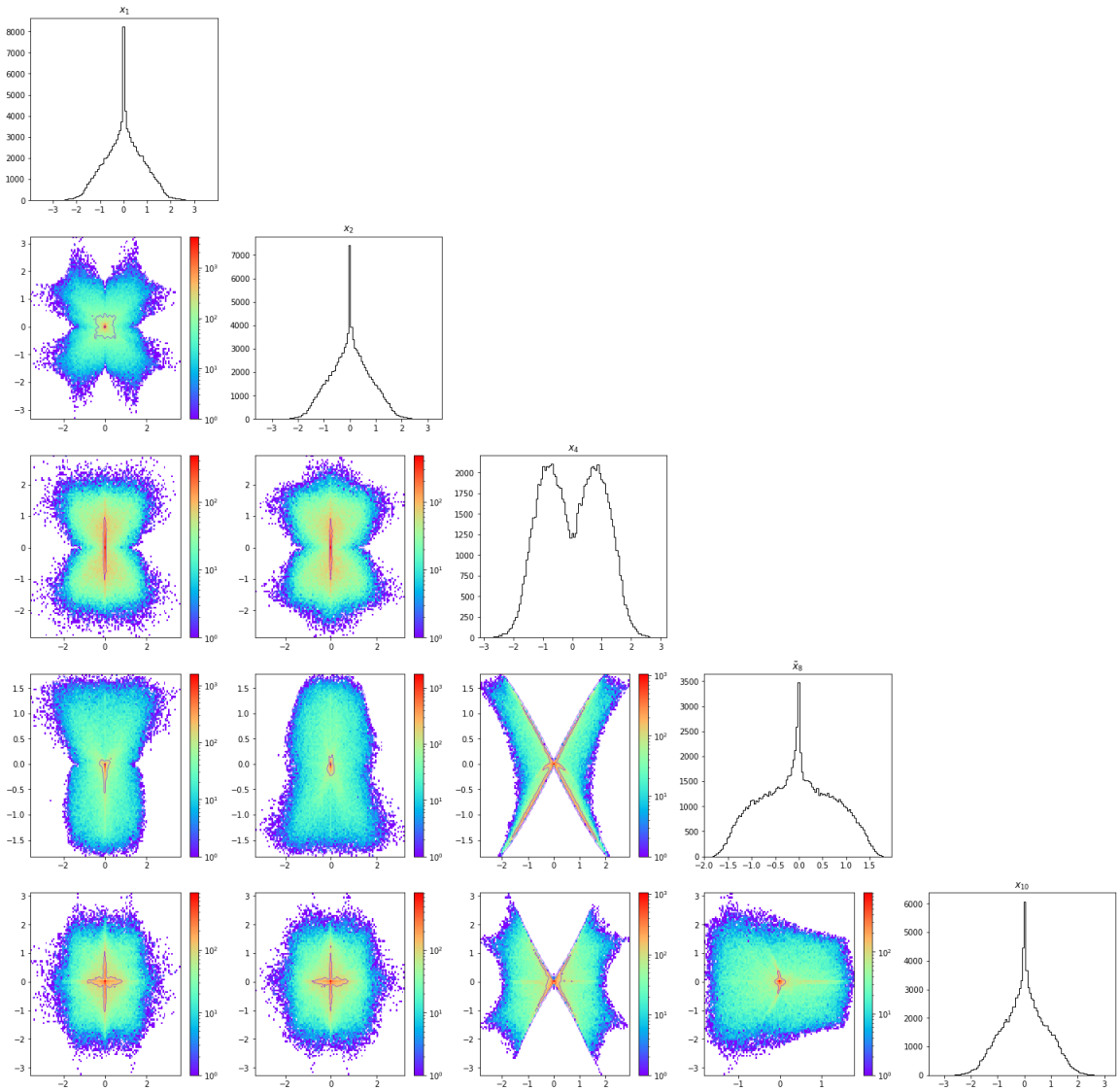

Fig. 3 Triangular plot showing $2d$ projections and $1d$ histograms of the data after the gradient descent

This can be attributed to the fact that if the true 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 dimensionalities of 4 or 5. This behavior 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 conclude that the manifold under investigation has an intrinsic dimension of 3.

One possible scenario is that our data actually consists of several distinct three-dimensional manifolds, and the points previously identified with dimension 4 may lie at the intersections of these manifolds. To illustrate, consider the intersection of two lines: at the intersection point, the local dimensionality estimated by the previous PCA algorithm would be 2. To rule out this possibility, we apply a clustering algorithm

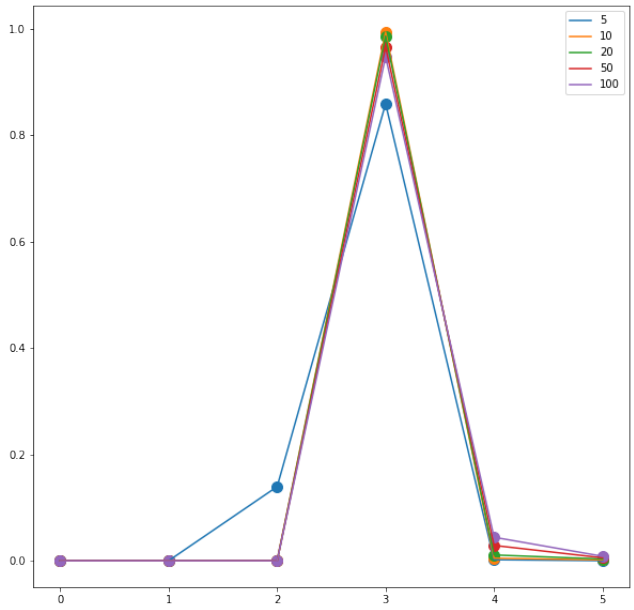

Fig. 4 Results of the local PCA analysis. The x-axis shows the dimension inferred by the algorithm, and the y-axis indicates the proportion of points for which that dimension was found. Each curve corresponds to a different value of k in the k-nearest neighbours.

based on local density, where we only select the points of dimension 3. This, way, we remove the possible intersection points with dimension 4. For the purpose of the clustering, we use the HDBSCAN algorithm[1].

We set the minimum cluster size to 10 points. Out of the 98,606 3-dimensional points, the algorithm identifies three clusters: one of size 10, one of size 91,747, and one of size 6,849.

The smallest cluster (10 points) is likely an artefact of local fluctuations in the data density and is not interpreted as physically meaningful. The last group, comprising 6,849 points, consists of points that the algorithm could not assign to any cluster. We interpret these as outliers or edge points rather than representatives of a separate manifold.

What the algorithm does indicate, however, is that over 90% of the data belongs to a single dominant cluster. In Fig. 5, we show 3D projections of the data to visualize the clustering.

In these scatter plots, turquoise points belong to the main cluster, while purple points are those that the algorithm failed to assign. In one of the plots, a few yellow points can be seen; these correspond to the smallest cluster of 10 points. From visual inspection, it appears that the unassigned (purple) 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.

From this analysis, we conclude that the gradient descent procedure has produced a sampling of a single, connected, three-dimensional manifold.

After performing both PCA and clustering, we thus confirm that the data obtained after gradient descent samples a three-dimensional manifold.

[1]Details of the algorithm can be found in the original HDBSCAN paper or documentation. Briefly, it is a density-based clustering method that extends DBSCAN by converting it into a hierarchical clustering algorithm and then extracting a flat clustering based on the stability of clusters.

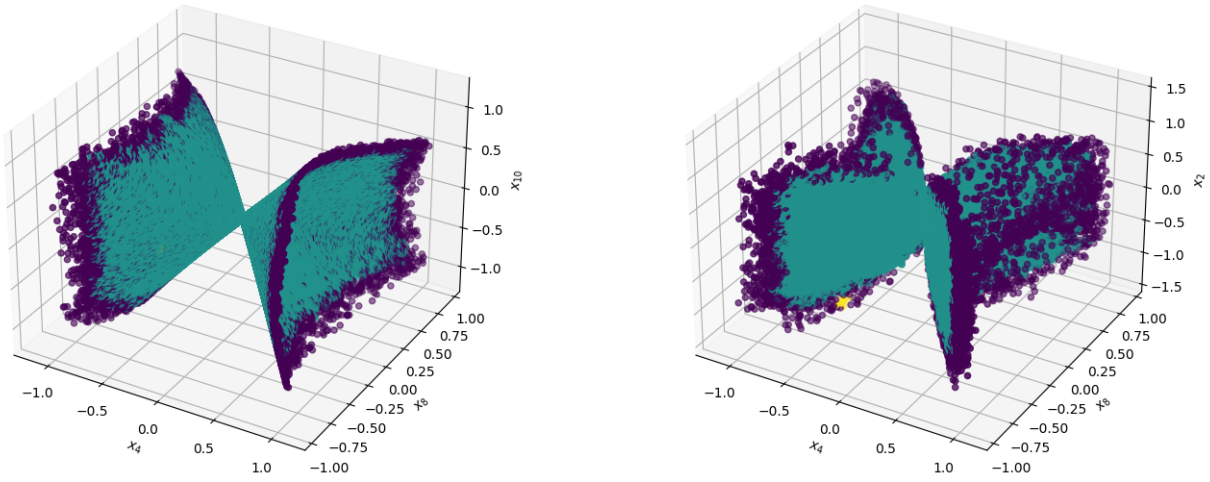

Fig. 5 3D plots of the data in selected coordinates. Left: (x_4, x_8, x_{10}). Right: (x_2, x_4, x_8).

3.2 Annealed Importance Sampling for polynomial symbolic regression

Now we have sampled the manifold and extracted some basic information about it (namely its dimension and the fact that it is made of one block), we would like to see if we can extract some analytic formula to characterise it. What we have at the moment, are points on a 3-dimensional manifold, which are embedded in a 5-dimensional space. Therefore we can conclude that, in order to characterise the manifold, we need two independent constraints on the embedding 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 (2.16), we see that if we use $x_8 = e^{\tilde{x}_8}$, that, up to a global $e^{-2\tilde{x}_8}$ factor, this potential is actually a polynomial of the embedding coordinates. Therefore, the components of $|\nabla V|$ are also polynomials in those variables. We conclude that the constraints on the embedding coordinates we are looking for are polynomial constraints of the form $p(\vec{x}) = 0$, and that there should be at least two of those. Note that we are now taking $\vec{x} = (x_1, x_2, x_4, \tilde{x}_8, x_{10})$. Of course if one takes directly the gradient of (2.16), 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 (polynomial) that vanish once evaluated on our data points. There are a number of way one can deal with it, using already existing methods such as AI Feynman methods [3], or ... **(bd: litterature review)**. In the spirit of trying to build a generalisable method, we develop here our own method, which is based on symbolic regression using an Annealing Importance Sampling method. ⟸

In the realm of symbolic regression, the aim is to uncover interpretable mathematical expressions that best describe a given dataset. In the problem discussed in this paper, we are looking for polynomial expressions such that $p(\vec{x}^i) = 0$, with $i \in \{1, \ldots, \texttt{n\_sample}\}$. This task 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, while widely used, can struggle with poor mixing and getting trapped in local optima, especially in high-dimensional or multimodal spaces.

To address these challenges, we employ an Annealed Importance Sampling (AIS) combined with

Sequential Monte Carlo (SMC) methods. AIS constructs 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 emphasizes the data likelihood, allowing for more efficient exploration of the probability landscape.

SMC enhances this procedure by propagating a population of particles—each representing a candidate symbolic expression—through the sequence of distributions. At each step, particles are reweighted based on the incremental change in the distribution, resampled to focus computational effort on high-probability regions, and mutated via operations. 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 details how does this procedure goes. The goal is to reconstruct some distribution function $d(z)$, where here z in going to be some polynomial. We will try to reconstruct this density function by series of density function $\pi_n(z_n) = \gamma_n(z_n)/Z_n$ with $n = 1, \dots, N$ is going to be the number of annealing steps, π_n is defined in terms of an unnormalized density γ_n and we have the normalizing constant $Z_n = \int \gamma_n(z)\mathrm{d}z$. We also assume we have a sequence of inverse temperature constants β_n, where $0 = \beta_1 < \beta_2 < \cdots < \beta_N = 1$. We then define the unnormalized density at level n in terms of a prior distribution $p_0(z)$ over the hypothesis space and a loss function $L(z)$:

$$\gamma_n(z) := d_0(z)\ \exp\left(-\beta_n L(z)\right). \tag{3.2}$$

At each step, we have a set of particles and weights $\{z_{n-1}^k, w_{n-1}^k\}$ approximating π_{n-1} (meaning $\mathbb{E}_{\pi_{n-1}}[f] \approx \frac{\sum_k w_{n-1}^k f(z_{n-1}^k)}{\sum_k w_{n-1}^k}$), and we want to obtain a new set $\{z_n^k, w_n^k\}$ approximating π_n. To do so, for each particle z_{n-1}^k, we propose a new particle $z_n^k \sim q(z_n|z_{n-1}^k)$ and calculate the new unnormalized importance weight w_n^k. The latter are updated using the formula

$$w_n^k = w_{n-1}^k \times \alpha_n^k \tag{3.3}$$

where α_n^k is the incremental importance weight. The standard form for α_n^k, which requires introducing an auxiliary backward transition kernel $q(z_{n-1}|z_n)$, is:

$$\alpha_n^k = \frac{\gamma_n(z_n^k) q(z_{n-1}^k|z_n^k)}{\gamma_{n-1}(z_{n-1}^k) q(z_n^k|z_{n-1}^k)} \tag{3.4}$$

The weights are finally normalized and we get $w_n^k \to \frac{w_n^k}{\sum_j w_n^j}$.

Let's us now focus on the forward $q(z_n^k|z_{n-1}^k)$ and backward propagation kernel $q(z_{n-1}^k|z_n^k)$ for now. The forward propagation kernel defines how we generate the state at time n given the state at time $n-1$. It gives the probability to transition from z_{n-1} to z_n. The backward kerne represents a hypothetical probability of transitioning back from state z_n to state z_{n-1}. The purpose of q is to give a proposition for z_n given z_{n-1}. For the implementation of the two, we decide to implement an AIS-style MCMC code : we make some move in the space of polynomials, and then accept or reject those new polynomials based on an acceptance rate. So we first need to choose what are the available moves in the space of polynomials. Here are the choices we have made for our symbolic regression task :

- Coefficient pertubation. Given a polynomial z_{n-1}, we choose randomly one of his coefficient and modify it by a random noise from the guassian distribution $\mathcal{N}(0, \sigma^2)$. So for eg : $2x_1 + 3x_2^2 \to 2.1x_1 + 3x_2^2$.
- Variable multiplication. Given a polynomial z_{n-1}, we choose randomly one of his monomial, and multiply it by one of the available variable. So for eg : $2_1^x + 3x_2^2 \to 2x_1x_2 + 3x_2^2$.
- Variable division. Given a polynomial z_{n-1}, we choose randomly one of his monomial, and divide it by one of its variable. So for eg : $2x_1 + 3x_2^2 \to 2x_1x_2 + 3x_2^2$.

This implementation enables a direct computation of both the forward and backward propagation kernels. In particular, for the case of coefficient perturbations, the transition is symmetric. That is,

$$q(z_n^k \mid z_{n-1}^k) = q(z_{n-1}^k \mid z_n^k), \tag{3.5}$$

which significantly simplifies the computation of the incremental importance weight.

However, this symmetry does not hold in general. For other types of operations—such as multiplication or division by a variable—the transition from z_{n-1}^k to z_n^k is generally not equivalent to the reverse transition from z_n^k to z_{n-1}^k. In such cases $q(z_n^k \mid z_{n-1}^k) \neq q(z_{n-1}^k \mid z_n^k)$ and particular care must be taken in the computation of the backward propagation kernel.

The acceptance ratio for the MCMC algorithm is given by :

$$A(z_n, z_{n-1}) = \min\left(1, \frac{\gamma_n(z_n^k) q(z_{n-1}^k | z_n^k)}{\gamma_{n-1}(z_{n-1}^k) q(z_n^k | z_{n-1}^k)}\right) \tag{3.6}$$

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.

Before closing the section, we need to discuss what is the Loss function we choose. It is required to compute the unormalized distribution γ_n. Assume that we have $z = \sum_k c_k X_k$, where c_k are the coefficients of the polynomials, and X_k is a short notations for all the possible monomials up to a given degree (so if we have x and y as variables, and the maximum degree is 2, then X_k are $1, x, y, x^2, y^2, xy$). For the Loss function, we decided to take :

$$L(z) = \sum_i z(x_i)^2 + \frac{\lambda}{\sum_k c_k} \tag{3.7}$$

So the first term is just the sum of the squared of the polynomial evaluated on the data. We want to make this 0 so we find polynomial that annihilates our data. The second part is a regularisation factor : it prevents the algorithm to send all the coefficient to 0, which would give a trivial solution to the problem. We typically take $\lambda \sim \mathcal{O}(1000)$.

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 coefficient may need some more fine tuning. To deal with it, we ran a quick exploitation phase, where, for each polynomial, we now only modify its coefficients, with a pertubation $\epsilon \sim \mathcal{N}(0, \sigma_1)$, and keep the new particle, only if the Loss function is now getting smaller. This allow to fine-tune the coefficients. ⟸

(bd: Results for naive choices of parameters (numbers of iteration and particles, probabilities, β) and motivate them (we want some exploration and then exploitation, not too long computations): for multiple runs we find multiple polynomials (good polynomial: after exploitation we convert the coefficients to integers and square roots, and recompute loss without regularisation, select with threshold). Quantify it nicely: success rates for each

polynomials, and absolute number of success, failing rate. Total time needed, without cluster or fancy computers.)

Results We have used the numerical analysis outlined in the previous sections to the search for extrema of the scalar potential (2.16). The code finds the 7 different following polynomials:

$$p_1 = -\sqrt{2}\,x_1 + \sqrt{2}\,x_1x_8 + x_2x_8x_{10} - \sqrt{2}\,x_2x_4x_8, \tag{3.8a}$$
$$p_2 = 2\,x_2 - 2\,x_2x_8 + \sqrt{2}\,x_1x_{10} + 2\,x_1x_4 - x_2x_8x_{10}^2, \tag{3.8b}$$
$$p_3 = 2\,x_2 - 2\,x_2x_8 + \sqrt{2}\,x_1x_8x_{10} + 2\,x_1x_4x_8 - 2\,x_2x_4^2x_8, \tag{3.8c}$$
$$p_4 = \sqrt{2}\,x_2 - \sqrt{2}\,x_2x_8 + \sqrt{2}\,x_1\,x_4 + x_1x_8x_{10} - x_2x_4x_8x_{10}, \tag{3.8d}$$
$$p_5 = -2\,x_1 - 2\,x_2x_4 - 2\,x_1x_4^2 + 2\,x_1x_8 + \sqrt{2}\,x_2x_{10} + x_1x_8x_{10}^2, \tag{3.8e}$$
$$p_6 = -2\,x_2x_4 - 2\,x_1x_4^2 + 2\,x_2x_4x_8 + \sqrt{2}\,x_2x_{10} - \sqrt{2}\,x_2x_8x_{10} + x_1x_8x_{10}^2, \tag{3.8f}$$
$$p_7 = -\sqrt{2}\,x_1^2x_4 + \sqrt{2}\,x_2^2x_4 + \sqrt{2}\,x_1x_2x_4^2 - x_1^2x_{10} - x_2^2x_{10}, \tag{3.8g}$$

where $x_8 = e^{\tilde{x}_8}$. They are found with different frequencies, depending on the chosen parameters.

Let us clarify a point regarding the expected structure of the solutions to the system defined by the critical points of the scalar potential, i.e., the variety defined by the vanishing of the gradient ∇V.

In algebraic geometry, the set of solutions to the equations $\nabla V = 0$ defines an algebraic variety over a certain base field. In our case, the potential V and its gradient ∇V are polynomial functions with coefficients lying in a specific field extension of $\mathbb{Q}$—typically of the form $\mathbb{Q}(\sqrt{d_1}, \sqrt{d_2}, \dots)$, where $d_i \in \mathbb{Q}$. That is, the coefficients of the polynomials are combinations of rational numbers and square roots of rational numbers.

The conformal manifold (or the moduli space of solutions) is then a subvariety of this ambient space, defined as the common zero locus of the polynomials in ∇V. From the perspective of algebraic geometry, this variety is cut out by an ideal in the polynomial ring with coefficients in the aforementioned field extension of $\mathbb{Q}$. Since the defining equations have coefficients in this field, the variety itself is defined over that same field.

Now, consider the numerical solutions obtained via methods such as annealed importance sampling. If these numerical procedures converge to points on the same variety, and assuming that the variety is irreducible over the given field, then any point (i.e., any solution) on the variety must lie in the same field extension over which the variety is defined. This follows from basic principles in algebraic geometry: the set of algebraic points on a variety defined over a field K are themselves algebraic over K, and the field of definition of these points must be a (possibly finite) extension of K.

Therefore, even if the numerical solver does not explicitly reveal this, one should expect that the coordinates of the solutions (i.e., the values of the scalar fields at critical points) lie in the same field extension of $\mathbb{Q}$ as the coefficients of the original system of polynomials ∇V. That is, they should belong to $\mathbb{Q}(\sqrt{d_1}, \sqrt{d_2}, \dots)$ or a finite extension thereof.

This argument justifies the expectation that if the scalar potential V and its gradient ∇V involve only rational numbers and square roots, then the solutions to $\nabla V = 0$—regardless of how they are found—should also lie within the same field extension.

This is this argument that allowed us to round the coefficients of the polynomials given by the annealed importance sampling algorithm, and found the polynomials in 3.8.

1. **β in power 3, $\sigma = 0.1$, `sample_size` $= 10000$**

For 100 runs with parameters

$$\begin{gathered}\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000, \\ \texttt{sample size} = 10000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^3, \quad \sigma = 0.1, \\ p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25,\end{gathered} \tag{3.9}$$

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	76%	18%	10%	6%	4%	1%	2%	6%

(3.10)

The dynamics of the appearance of the target polynomials is plotted in fig. 6.

Averaged number of particles reproducing a target in a successful run: 973 ± 103

Mean loss with regularisation for particles reproducing a target: 854 ± 427

Mean loss without regularisation for particles reproducing a target: 602 ± 417

After the AIS, we performed an exploitation phase to fine tune the coefficients. The distributions of the particles with respect to their loss (without regularisation), is plotted in fig. 7. The fine tuning (with `local_search`) is made using the loss with regularisation.

We list below, for each polynomials in eq. (3.8), one example of representing particles directly after AIS and after the fine tuning, and the associated losses (without regularisation):

$$p_1: \begin{aligned} &-1.939\,x_1 + 2.120\,x_1\tilde{x}_8 + x_{10}x_2\tilde{x}_8 - 2.266\,x_2x_4\tilde{x}_8 + 0.029\,x_2^2x_4, \quad L^{(\lambda=0)} \simeq 754, \\ &-1.416\,x_1 + 1.414\,x_1\tilde{x}_8 + x_{10}x_2\tilde{x}_8 - 1.414\,x_2x_4\tilde{x}_8 + 0.00001\,x_2^2x_4, \quad L^{(\lambda=0)} \simeq 0.0040, \end{aligned} \tag{3.11}$$

$$p_2: \begin{aligned} &2.213\,x_2 - 2.075\,x_2\tilde{x}_8 + 1.674\,x_1x_{10} + 2.015\,x_1x_4 - x_{10}^2x_2\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 244, \\ &2.002\,x_2 - 2.002\,x_2\tilde{x}_8 + 1.417\,x_1x_{10} + 2.001\,x_1x_4 - x_{10}^2x_2\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 0.015, \end{aligned} \tag{3.12}$$

$$p_3: \begin{aligned} &2\,x_2 - 1.808\,x_2\tilde{x}_8 + 1.168\,x_1\tilde{x}_8x_{10} + 1.828\,x_1x_4\tilde{x}_8 - 1.864\,x_2x_4^2\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 180, \\ &2\,x_2 - 1.998\,x_2\tilde{x}_8 + 1.412\,x_1\tilde{x}_8x_{10} + 1.998\,x_1x_4\tilde{x}_8 - 1.998\,x_2x_4^2\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 0.016, \end{aligned} \tag{3.13}$$

$$p_4: \begin{aligned} &1.458\,x_2 - 1.341\,x_2\tilde{x}_8 + 1.497\,x_1x_4 + x_1\tilde{x}_8x_{10} - 1.008\,x_2x_4\tilde{x}_8x_{10} + 0.020x_2^2x_4\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 218, \\ &1.419\,x_2 - 1.418\,x_2\tilde{x}_8 + 1.416\,x_1x_4 + x_1\tilde{x}_8x_{10} - 1.001\,x_2x_4\tilde{x}_8x_{10} - 0.0003\,x_2^2x_4\tilde{x}_8, \quad L^{(\lambda=0)} \simeq 0.074, \end{aligned} \tag{3.14}$$

$$p_5: \begin{aligned} &-2.179\,x_1 - 1.571\,x_2x_4 - 1.953\,x_1x_4^2 + 1.983\,x_1\tilde{x}_8 + 1.530\,x_2x_{10} + x_1\tilde{x}_8x_{10}^2, \quad L^{(\lambda=0)} \simeq 210, \\ &-2.008\,x_1 - 2.001\,x_2x_4 - 1.999\,x_1x_4^2 + 1.995\,x_1\tilde{x}_8 + 1.412\,x_2x_{10} + x_1\tilde{x}_8x_{10}^2, \quad L^{(\lambda=0)} \simeq 0.228, \end{aligned} \tag{3.15}$$

$$p_6: \begin{aligned} &-2.856\,x_2x_4 - 2.499\,x_1x_4^2 + 2.721\,x_2x_4\tilde{x}_8 + 1.677\,x_2x_{10} - 1.187\,x_2\tilde{x}_8x_{10} + x_1\tilde{x}_8x_{10}^2, \quad L^{(\lambda=0)} \simeq 202, \\ &-2.043\,x_2x_4 - 2.005\,x_1x_4^2 + 2.030\,x_2x_4\tilde{x}_8 + 1.393\,x_2x_{10} - 1.401\,x_2\tilde{x}_8x_{10} + x_1\tilde{x}_8x_{10}^2, \quad L^{(\lambda=0)} \simeq 1.28, \end{aligned} \tag{3.16}$$

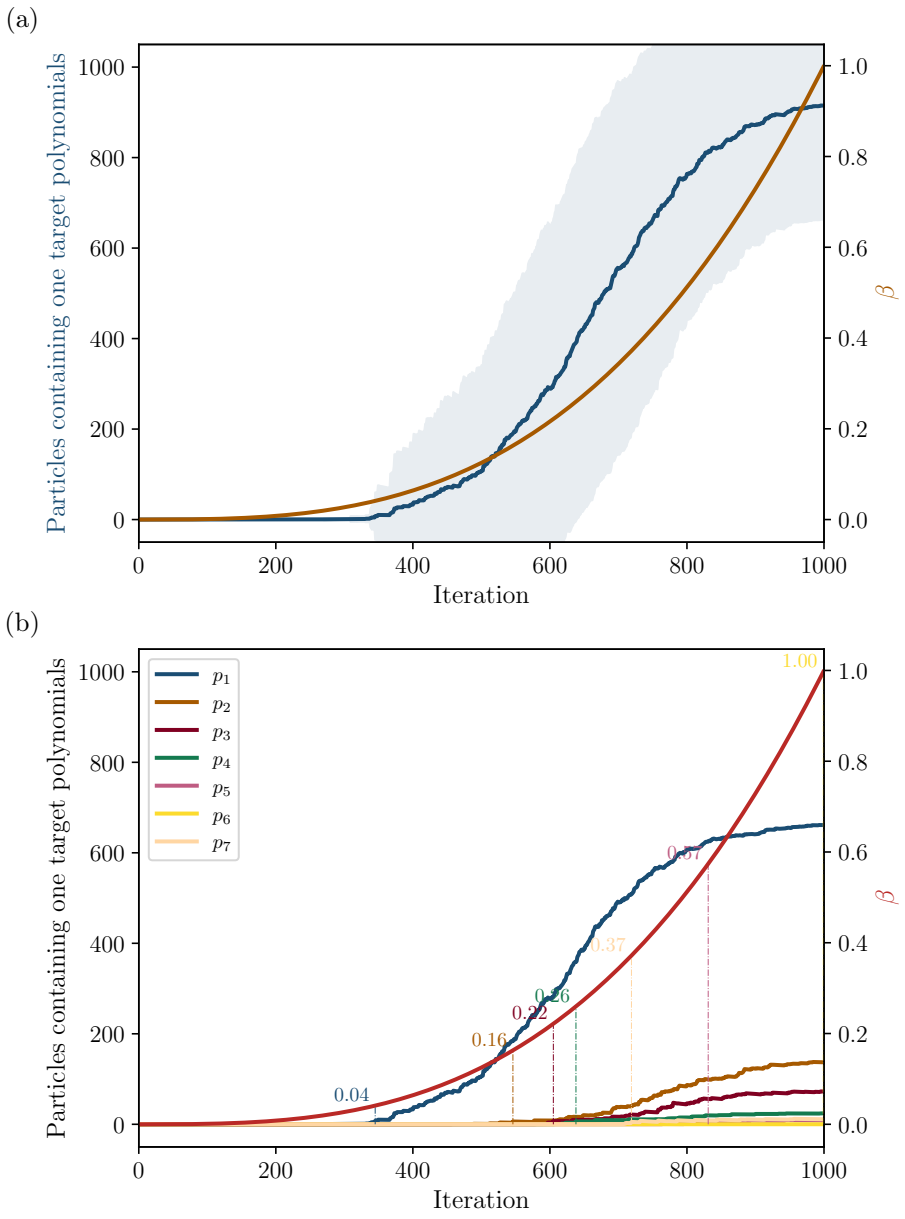

Fig. 6 β in power 3, $\sigma = 0.1$, `sample size` $= 10000$ (a) Evolution of the mean numbers of particles reproducing a target polynomials and its 1σ deviation for the parameters (3.9). (b) Evolution of the mean numbers of particles reproducing each targets polynomial. The vertical lines indicate when there are an average 5 particles reproducing a given polynomial, and the value of β at that moment.

$$p_7 : \begin{array}{ll} -1.354\,x_1^2x_4 + 1.243\,x_2^2x_4 + 1.304\,x_1x_2x_4^2 - 1.053\,x_1^2x_{10} - x_2^2x_{10} + 0.052\,x_2x_4^2x_{10}, & L^{(\lambda=0)} \simeq 373, \\ -1.413\,x_1^2x_4 + 1.415\,x_2^2x_4 + 1.413\,x_1x_2x_4^2 - 0.997\,x_1^2x_{10} - x_2^2x_{10} + 0.0003\,x_2x_4^2x_{10}, & L^{(\lambda=0)} \simeq 0.038. \end{array} \tag{3.17}$$

2. **β in power 4, $\sigma = 0.1$, `sample size` $= 10000$**

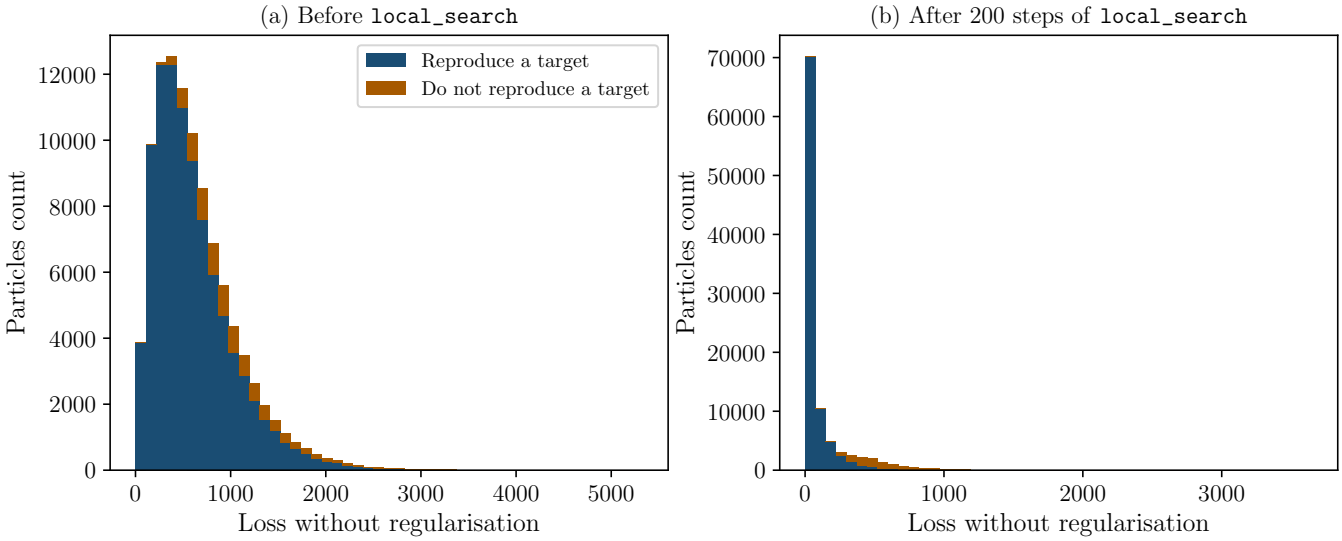

Fig. 7 β in power 3, $\sigma = 0.1$, `sample size` $= 10000$ Distributions of all the 100,000 particles with respect to their loss (without regularisation) (a) directly after AIS with parameters (3.9) and (b) after 200 steps of `local_search`. The fine tuning (with `local_search`) is made using the loss with regularisation.

For 100 runs with parameters

$$\begin{gathered}\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000, \\ \texttt{sample size} = 10000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^4, \quad \sigma = 0.1, \\ p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25,\end{gathered} \tag{3.18}$$

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	74%	17%	7%	8%	2%	0%	2%	6%

(3.19)

The dynamics of the appearance of the target polynomials is plotted in fig. 8.

Averaged number of particles reproducing a target in a successful run: 951 ± 200

Mean loss with regularisation for particles reproducing a target: 709 ± 320

Mean loss without regularisation for particles reproducing a target: 452 ± 309

3. **β in power 5, $\sigma = 0.1$, `sample size` $= 10000$**

For 100 runs with parameters

$$\begin{gathered}\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000, \\ \texttt{sample size} = 10000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^5, \quad \sigma = 0.1, \\ p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25,\end{gathered} \tag{3.20}$$

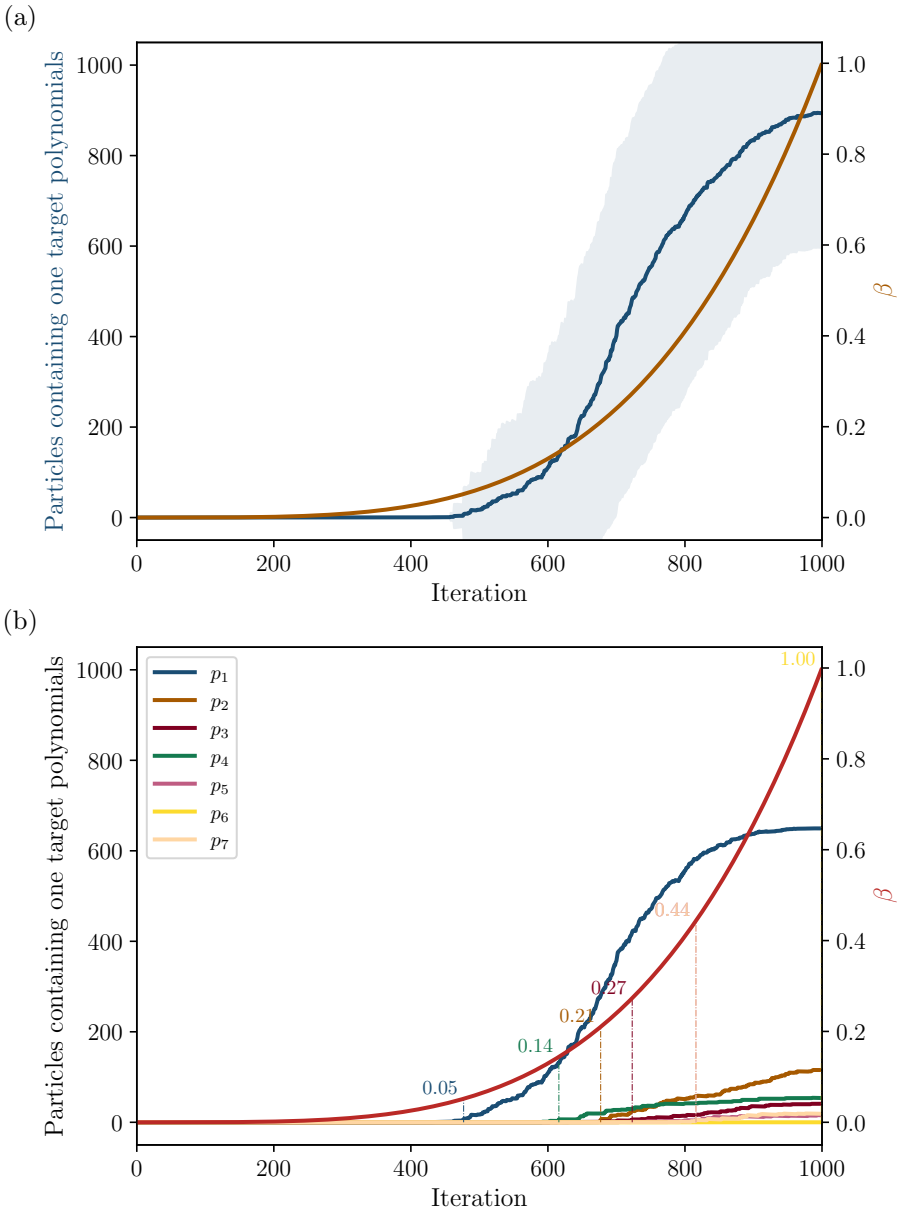

Fig. 8 β in power 4, $\sigma = 0.1$, `sample size` $= 10000$ (a) Evolution of the mean numbers of particles reproducing a target polynomials and its 1σ deviation for the parameters (3.18). (b) Evolution of the mean numbers of particles reproducing each targets polynomial. The vertical lines indicate when there are an average 5 particles reproducing a given polynomial, and the value of β at that moment.

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	69%	11%	7%	12%	0%	4%	0%	13%

(3.21)

The dynamics of the appearance of the target polynomials is plotted in fig. 9.

Averaged number of particles reproducing a target in a successful run: 960 ± 175

Mean loss with regularisation for particles reproducing a target: 606 ± 257

Mean loss without regularisation for particles reproducing a target: 359 ± 244

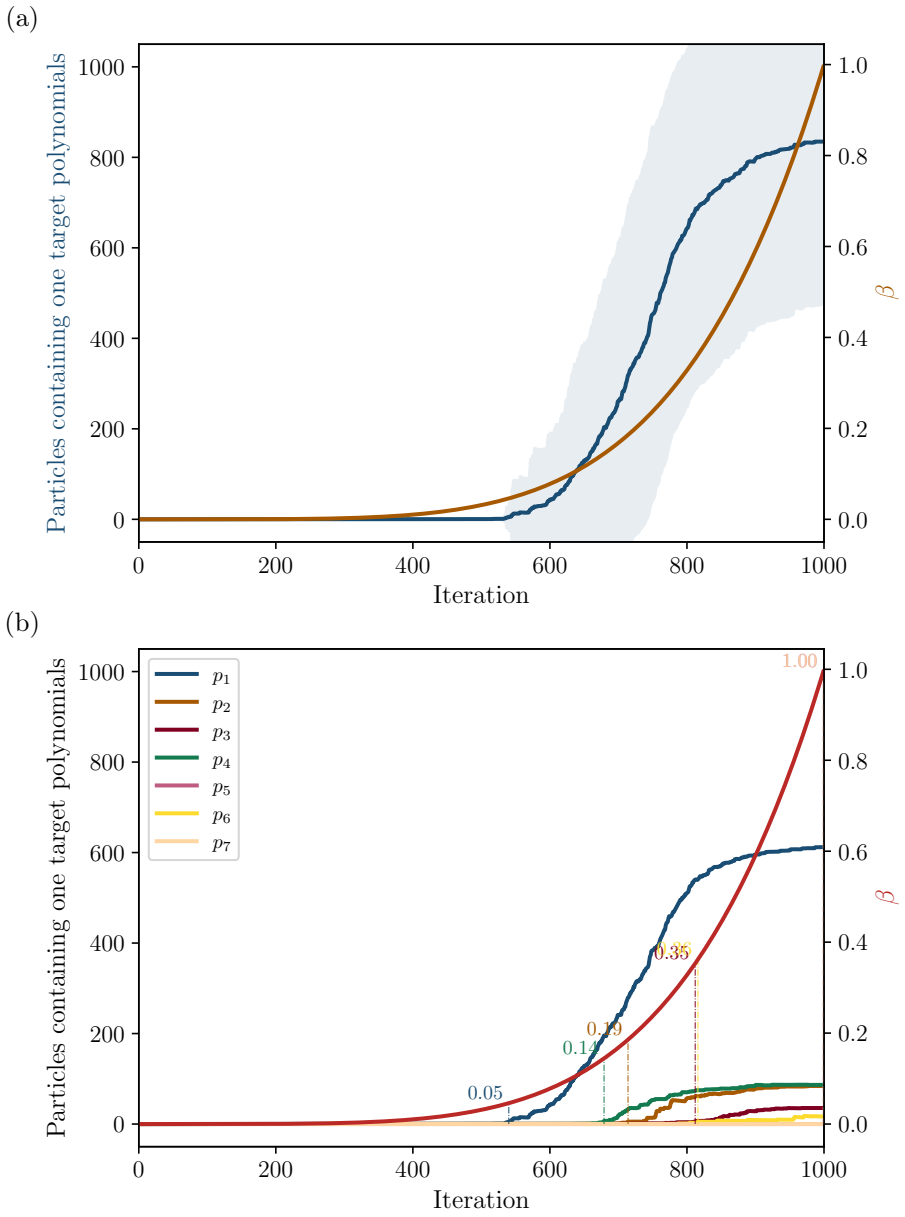

Fig. 9 β in power 5, $\sigma = 0.1$, `sample size` $= 10000$ (a) Evolution of the mean numbers of particles reproducing a target polynomials and its 1σ deviation for the parameters (3.22). (b) Evolution of the mean numbers of particles reproducing each targets polynomial. The vertical lines indicate when there are an average 5 particles reproducing a given polynomial, and the value of β at that moment.

4. **β in power 5, σ varying, `sample size` $= 10000$**

For 100 runs with parameters

$$\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000,$$
$$\texttt{sample size} = 10000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^5, \quad \sigma = 0.5 - \frac{0.45}{1 + \exp\left(10 - 20i/n_{\text{iter}}\right)},$$
$$p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25, \tag{3.22}$$

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	59%	25%	5%	9%	2%	4%	2%	13%

(3.23)

The sum of the percentages is higher than 100% because some runs find more than one polynomials. The code fails to find any polynomial in 13% of the time. The averaged time needed for a single run on a PC in $\sim 700\,\text{s}$. **(ce: This timing is with `analysis`, it may be less without.)** ⟸

Averaged number of particles reproducing a target in a successful run: 931 ± 230

Mean loss with regularisation for particles reproducing a target: 597 ± 264

Mean loss without regularisation for particles reproducing a target: 345 ± 246

5. **β in power 3, σ varying, `sample size` = 10000**

For 100 runs with parameters

$$\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000,$$
$$\texttt{sample size} = 10000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^3, \quad \sigma = 0.5 - \frac{0.45}{1 + \exp\left(10 - 20i/n_{\text{iter}}\right)},$$
$$p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25, \tag{3.24}$$

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	68%	17%	6%	9%	2%	1%	1%	11%

(3.25)

The dynamics of the appearance of the target polynomials is plotted in fig. 10.

Averaged number of particles reproducing a target in a successful run: 966 ± 115

Mean loss with regularisation for particles reproducing a target: 837 ± 438

Mean loss without regularisation for particles reproducing a target: 609 ± 430

6. **β in power 5, $\sigma = 0.1$, `sample size` = 1000**

For 100 runs with parameters

$$\texttt{max degree} = 4, \quad \texttt{max num monomials} = 6, \quad n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000,$$
$$\texttt{sample size} = 1000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^5, \quad \sigma = 0.1, \tag{3.26}$$
$$p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25,$$

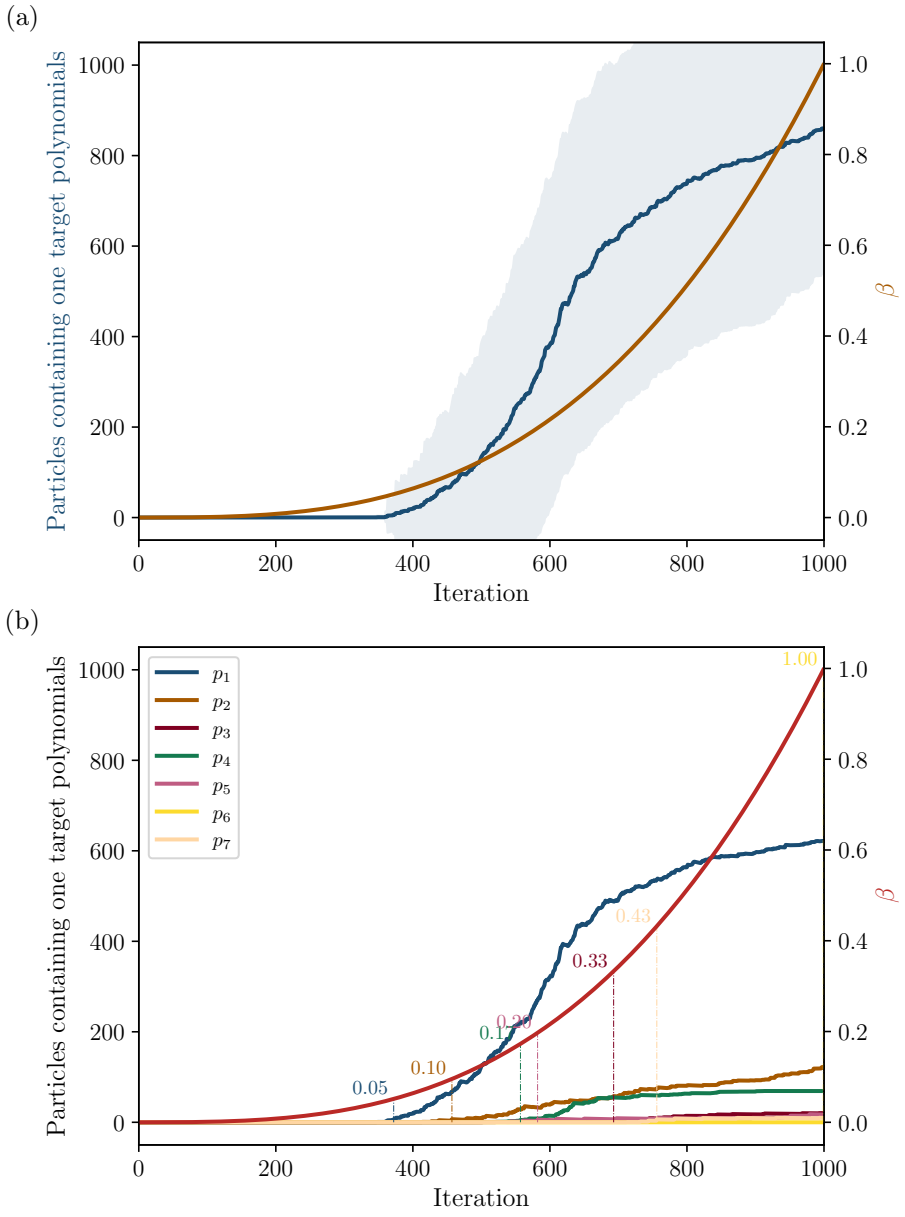

Fig. 10 β in power 3, σ varying, `sample size` = 10000 (a) Evolution of the mean numbers of particles reproducing a target polynomials and its 1σ deviation for the parameters (3.24). (b) Evolution of the mean numbers of particles reproducing each targets polynomial. The vertical lines indicate when there are an average 5 particles reproducing a given polynomial, and the value of β at that moment.

the polynomials are found at the following frequencies

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	$\varnothing$
Frequency	72%	27%	3%	19%	2%	0%	0%	12%

(3.27)

The dynamics of the appearance of the target polynomials is plotted in fig. 11.

Averaged number of particles reproducing a target in a successful run: 401 ± 389

Mean loss with regularisation for particles reproducing a target: 406 ± 193 **(ce: Needs to be** ⟸

renormalized by the sample size.)

Mean loss without regularisation for particles reproducing a target: 272 ± 192 **(ce: Needs to be renormalized by the sample size.)** ⟸

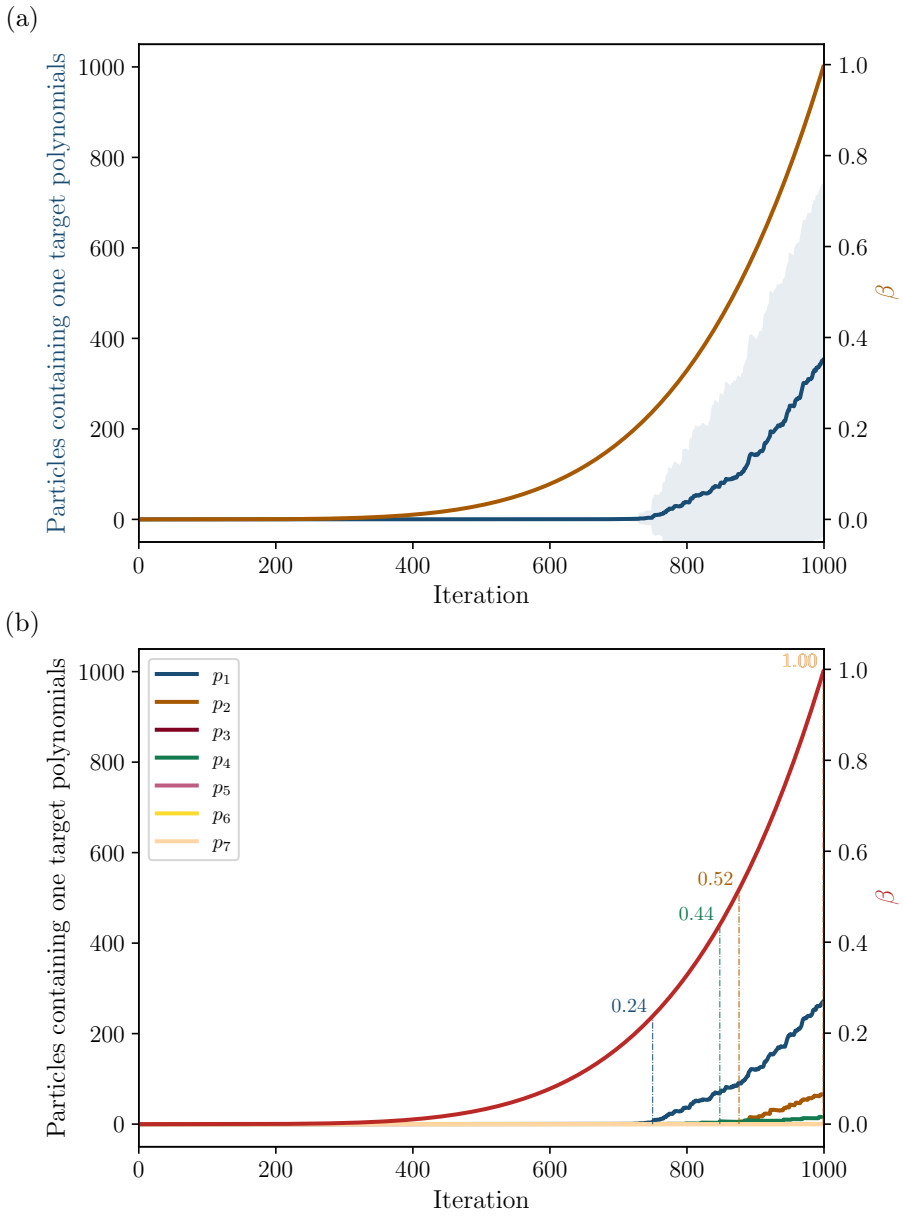

Fig. 11 β in power 5, $\sigma = 0.1$, `sample size` $= 1000$ (a) Evolution of the mean numbers of particles reproducing a target polynomials and its 1σ deviation for the parameters (3.26). (b) Evolution of the mean numbers of particles reproducing each targets polynomial. The vertical lines indicate when there are an average 5 particles reproducing a given polynomial, and the value of β at that moment.

⟸

(ce: Draw histograms of the polynomials distribution to discuss runs with multiple outputs?)

4 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. 3.8. We can now solve the system

$$p_i = 0, \quad \forall i \in 1, \dots, 7 \tag{4.1}$$

which leads to the following expressions:

$$\begin{cases} x_1 = \dfrac{x_2}{\sqrt{2}} \dfrac{e^{x_8/2}}{e^{x_8}-1} \Big(- x_5\, e^{x_8/2} + \sqrt{2 - 4\, e^{x_8} + e^{2x_8}\big(2 + x_{10}^2\big)} \Big), \\ x_4 = \dfrac{e^{-x_8/2}}{\sqrt{2}} \sqrt{2 - 4\, e^{x_8} + e^{2x_8}\big(2 + x_{10}^2\big)}. \end{cases} \tag{4.2}$$

Alternatively, the system can be recast as:

$$\begin{cases} \tilde{x}_8 = \dfrac{x1^2 + x_2^2}{x_2^2 + \big(x_1 - x_2 x_4\big)^2}, \\ x_{10} = \sqrt{2}, x_4, \dfrac{x_2^2 - x_1^2 + x_1 x_2 x_4}{x_1^2 + x_2^2}. \end{cases} \tag{4.3}$$

As anticipated, this defines a three-parameter solution corresponding to a three-dimensional slice of the half-maximal supergravity scalar potential.

Let us make a few remarks at this stage. First, it is possible to find solutions to Eq. 4.1 which, however, are not solutions of $\nabla V = 0$. For instance $x_1 = x_2 = 0$; such cases are excluded, as they do not satisfy the stationarity condition and therefore do not correspond to valid physical solutions.

Secondly, one might argue that only two of the seven polynomials are sufficient to fully characterize the solution. That is, choosing any pair $(i, j) \in 1, \dots, 7$ 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.4.3—it may also produce alternative (and potentially less general) parameterizations. Upon inspection, all such partial rules are found to be consistent with, and included in, the most general expressions given in Eq.4.3.

The solution preserves a $\mathrm{U}(1) \times \mathrm{U}(1)$ gauged symmetry.

The (x_1, x_2, x_4) moduli space is most nicely parametrised using the change of coordinates

$$x_1 = r\cos(\theta)\cos(\Phi), \quad x_2 = r\cos(\theta)\sin(\Phi) \quad \text{and} \quad x_4 = r\sin(\theta), \tag{4.4}$$

for which the Zamolodchikov metric reads

$$\mathrm{d}^2 s_{\text{Zam.}} = -\,\mathrm{d}r^2 - r^2 \left(\mathrm{d}\theta^2 - r\cos(\theta)\,\mathrm{d}\theta\mathrm{d}\Phi + \sin(\theta)\,\mathrm{d}r\mathrm{d}\Phi + \frac{1}{2}\big(3 + r^2 - \cos(2\theta)\big)\,\mathrm{d}\Phi^2 \right). \tag{4.5}$$

(ce: Tests other change of variables? Test choices of variables other than (x_1, x_2, x_4)?) ⟸

Bosonic spectrum Vector fields:

$$\begin{aligned} m_{(1)}\ell_{\rm AdS}: \quad & 0\ [2], \quad -2\ [5], \quad 2\ [1], \\ & -1 \pm \sqrt{(1+r^2)^2 - 2\,r^2\,\cos(2\theta)}\ [2+2], \\ & 1 \pm \sqrt{1+4\,r^2+r^4}\ [2+2]. \end{aligned} \tag{4.6}$$

The integers between square brackets indicate the multiplicity of each eigenvalue. The spectrum includes two massless vectors corresponding to the unbroken U(1) × U(1) gauge symmetry, although in three dimensions they are non-propagating.

Scalars:

$$\begin{aligned} \left(m_{(0)}\ell_{\rm AdS}\right)^2: \quad & 0\ [5], \quad 8\ [1], \quad r^2\ \left(4+r^2\right)\ [8], \\ & 2r\left(3\,r + r^3 \pm (2+r^2)\,\sqrt{2+r^2-2\,\cos(2\theta)} - r\cos(2\theta)\right)\ [2+2]. \end{aligned} \tag{4.7}$$

Gravitini:

$$m_{(3/2)}\ell_{\rm AdS}: \qquad \frac{1}{2}\left[1 \pm \sqrt{4+2\,r^2+r^4-2r^2\cos(2\theta)}\right]\ [4+4], \tag{4.8}$$

no SUSY enhancements other than $r = 0$.

No dependence on Φ.

5 Conclusion

- Conclusion: good prospects of improvement to be able to access higher dimensional cases. Classify flat directions.
- Appendix with some details on the code?

In this work, we have presented a novel machine learning approach to systematically identify and characterize flat directions in supergravity scalar potentials. Our methodology combines gradient descent sampling with annealed importance sampling for symbolic regression, demonstrating its effectiveness on a 5-scalar subsector of 3d (2,0) supergravity derived from $AdS_3 \times S^3 \times T^4$ compactifications of IIB supergravity.

We developed a robust pipeline that transitions from numerical exploration to analytical understanding. The gradient descent procedure successfully samples the flat direction manifold, while local PCA analysis reveals its intrinsic dimensionality. Most notably, our annealed importance sampling approach to symbolic regression automatically discovers polynomial constraints characterizing the manifold, bypassing the computational complexity that renders direct symbolic manipulation intractable.

Our analysis reveals that the 5-dimensional scalar space contains a 3-dimensional conformal manifold, with one direction potentially gauge-fixable. The discovered solution preserves $U(1)\times U(1)$ gauged symmetry and exhibits a rich spectrum structure. The Zamolodchikov metric on the moduli space, parametrized by spherical-like coordinates, provides a concrete geometric description of the conformal manifold.

The method demonstrates remarkable efficiency and reliability. Across different parameter choices, we achieve success rates of 69-76% for the most common polynomial constraint, with computation times

of approximately 700 seconds per run on standard hardware. The algorithm discovers seven distinct polynomial relations with varying frequencies, suggesting a hierarchical structure in constraint discovery.

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.

Beyond technical improvements, this work suggests a broader paradigm for studying conformal manifolds in string theory and supergravity. The ability to automatically extract analytical constraints from numerical data could prove invaluable for classifying flat directions across different theories, understanding moduli stabilization mechanisms, and exploring the landscape of consistent supergravity backgrounds.

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.

While the specific solution found in this 5-scalar model 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.

References

[1] 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].

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

[3] 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].