MI-HET-???
May 16, 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

2 Supergravity setup

- Context: 6d $\mathrm{AdS}_3 \times S^3$, truncation to 3d, potential, conformal manifold.
- Choice of truncation from 32 to 13 to 5 variables.

Potential The 22 scalars of the theory can be parametrised by ($22 = 32 - 10$, with 10 scalars gauge fixed using translations in the gauge group)

- $m = \nu\nu^T \in \mathrm{GL}(3,\mathbb{R})$ 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 $\tilde{\varphi}$.

We can further restrict ourselves to a set of 13 scalars (see ref. [**?**]), with

$$\begin{gathered}
\xi = \begin{pmatrix} 0 & 0 & 0 & x_1 \\ 0 & 0 & 0 & x_2 \\ 0 & 0 & 0 & x_3 \end{pmatrix}, \quad \phi = \begin{pmatrix} 0 & x_4 & x_5 \\ -x_4 & 0 & x_6 \\ -x_5 & -x_6 & 0 \end{pmatrix}, \quad \tilde{\varphi} = x_{13} \\
\nu = e^{(6\,x_7 + 3\,x_8 + \sqrt{3}\,x_9)/6} \begin{pmatrix} 1 & \frac{x_{10}}{\sqrt{2}} & \frac{x_{11}}{\sqrt{2}} + \frac{x_{10}x_{12}}{4} \\ 0 & e^{-x_8} & \frac{e^{-x_8}\,x_{12}}{\sqrt{2}} \\ 0 & 0 & e^{-(x_8 + \sqrt{3}\,x_9)/2} \end{pmatrix}.
\end{gathered} \tag{2.1}$$

Scalar potential from ref. [**?**]:

$$\begin{aligned} V = 4\,e^{-4\tilde{\varphi}} + 2\,e^{-2\tilde{\varphi}}\Big[&- \operatorname{tr}\left(m + m^{-1}\right) + \operatorname{tr}\left(\phi m^{-1}\phi\right) - 2\operatorname{tr}\left(\phi m^{-1}\xi^2\right) - 2\operatorname{tr}\left(\xi^2\right) \\ &- \operatorname{tr}\left(\xi^2 m^{-1}\xi^2\right) + \frac{1}{2}\det\left(m^{-1}\right)\Big(1 - \operatorname{tr}\left(\phi^2\right) - \operatorname{tr}\left(\xi^4\right) + \operatorname{tr}\left(\xi^2\right)^2\Big) \\ &+ \frac{1}{2}\,\mathrm{T}\left(m^{-1}(\xi^2 - \phi), (\xi^2 + \phi)m^{-1}, m + (\xi^2 + \phi)m^{-1}(\xi^2 - \phi) + 2\,\xi^2\right) \\ &+ \frac{1}{4}\,\mathrm{T}\left(m^{-1}, m + (\xi^2 + \phi)m^{-1}(\xi^2 - \phi) + 2\,\xi^2, m + (\xi^2 + \phi)m^{-1}(\xi^2 - \phi) + 2\,\xi^2\right)\Big], \end{aligned} \tag{2.2}$$

where $\mathrm{T}\,(A, B, C) = \varepsilon_{mnp}\,\varepsilon_{qrs}\,A^{mq}B^{nr}C^{ps}$.

As a first simplified example we have considered only the parameters x_1, x_2, x_4, x_8 and x_{10}. The potential (2.2) becomes

$$\begin{aligned} V = \frac{1}{8}\,e^{-2\,x_8}\Big[&4 + 4\,x_1^4 + e^{4\,x_8}\left(2 + x_{10}^2\right)^2\left(1 + x_2^4\right) \\ &- 4\,e^{3\,x_8}\left(2 + x_{10}^2\right)\Big(2 - x_1^2 + \sqrt{2}\,x_1 x_{10} x_2^3 + x_2^4 + 2\,x_1 x_2 x_4 + x_4^2 - x_2^2\left(1 - x_1^2 + x_4^2\right)\Big) \\ &- 8\,e^{x_8}\Big(2 + x_1^4 + \sqrt{2}\,x_1^3 x_{10} x_2 - x_2^2 - 2\,x_1 x_2 x_4 + x_4^2 - x_1^2\left(1 - x_2^2 + x_4^2\right)\Big) \\ &+ 4\,e^{2\,x_8}\Big(2 - 4\,x_1^2 + x_1^4 - 4\,x_2^2 + 4\,x_1^2 x_2^2 + x_2^4 + x_{10}^2\left(1 + 3\,x_1^2 x_2^2\right) + 4\,x_4^2 + x_4^4\Big) \\ &+ 8\sqrt{2}\,x_{10}\,e^{2\,x_8}\Big(x_1^3 x_2 + x_1^2 x_4 - x_2^2 x_4 + x_1\left(x_2^3 - x_2 x_4^2\right)\Big)\Big]. \end{aligned} \tag{2.3}$$

3 Numerical analysis

3.1 Gradient descent and local analysis

- Gradient descent: implementation, sampling, time needed, loss.

  In order to look for flat direction in this potential, we will use numerical tools. The path we will follow is the following : first we will sample the underlying manifold, then we will use numerical tools to exctract analytics information on this cloud of points. Therefore the first step is to sample the manifold. To do so, we choose to do a basic gradient descent. We start by randomly uniformy initializing points in an hypercube of length $[-2, 2]$. This will be important latter to have points which are not only in $[-1, 1]$. We choose to have 10^5 points. For this example. This number is motivated by the dimensionality of our problem. If all possible directions are going to be populated, and if wen want $\mathcal{O}(10)$ points in each direction, then we need $\mathcal{O}(10^5)$ points. As the true internal dimension of the manifold is lesser or equal than 5, this number provides an upper bound for the number of points needed to sufficiently sample the manifold. We then perfom a gradient descent on the point using the automatic differentiation method of tensorflow. For the loss function, we use

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

  with $\vec{x} = (x_1, x_2, x_4, x_8, x_{10})$. We choose to use the Adam optimizer and use a learning rate of 10^{-2}. We run the gradient descent for 10000 epochs to make sure of convergence. As a first visualisation,

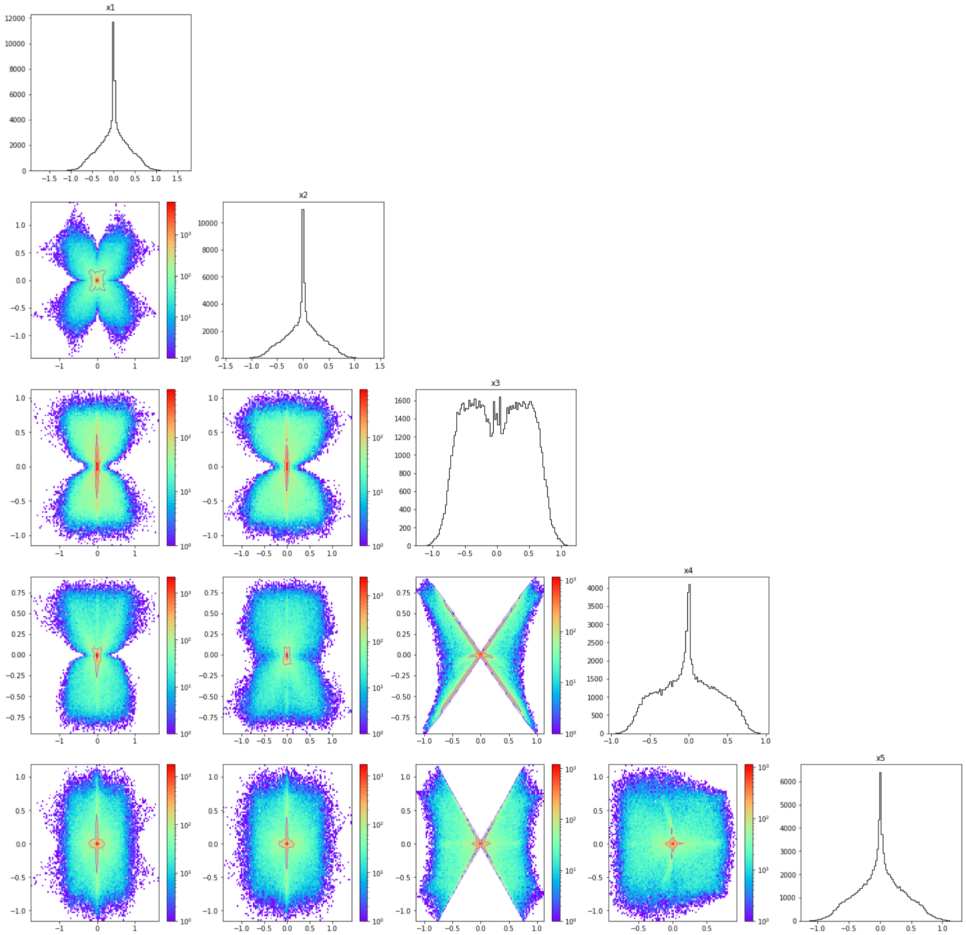

Fig. 1 Triangular plot showing $2d$ projections and $1d$ histograms of the data after the gradient descent

we show a triangular plot of the data in Fig. 1. This shows all possible $2d$ projections of the data as well as the $1d$ histograms of the data after the gradient descent.

Comments :

1. We see some non trivial correlation between the data. Note however that some features we see can be caused by the fact that some places may be greater basin of attractions (this is for example what happend in the x_1/x_2 plot). However, the structure we see in the x_4/x_8 or x_4/x_{10} is a remnant of true features of the manifold
2. All directions seems to be well populated
3. A great majority of the points fall in the $\vec{0}$ point. This is not too surprising. This just means that the round sphere solution has a greater basin of attraction. Even so, those points are going to be sufficiently good for the rest of the analysis, and we do not try to move the points along flat directions (to depopulate the origin et repopulate the rest).

- Local PCA: graphs of number of patches with given local dimension, and with varying size of local patch, discuss optimal patch size (we want at least fex points in each direction).

  Once the gradient descent is done and we sampled the flat direction, it remains to determine what is the underlying manifolds. We would like to obtain some analytics expression for this, not only

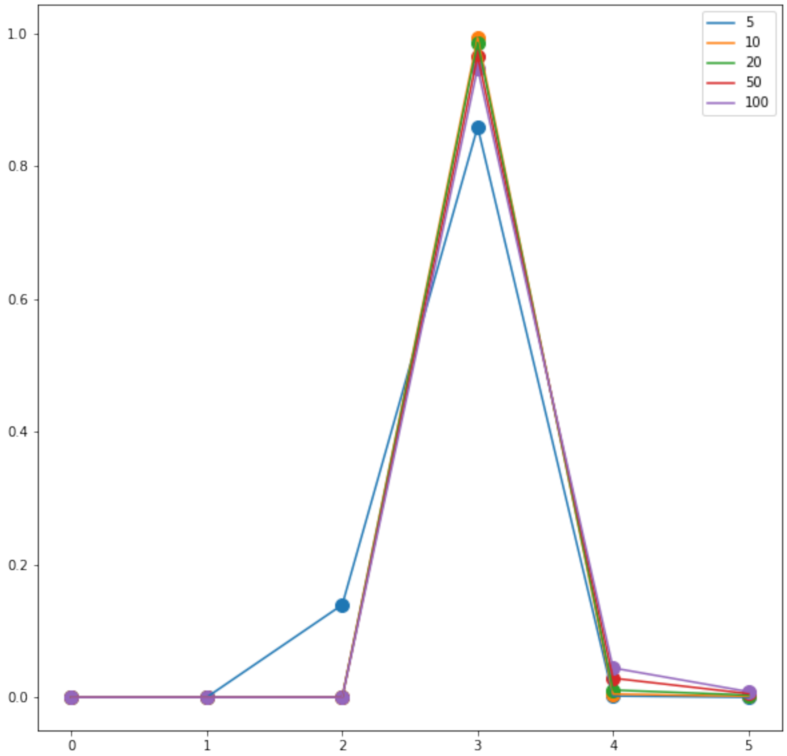

Fig. 2 Results of the local PCA analysis. On the x-axis is the dimension found by the algorithm, and on the y-axis the proportion of points. The different curves are for different k-nearest neighboors.

a numerical solution. Before proceeding to any sort of symbolic regression and look for such an analytic expression, there is a few things we can do to gain so understanding on the data, such as determining the dimensionality of the manifold, and wheter it a one piece or several pieces manifolds (like 2 intersecting hyperplane or so). To do so, we run a local Principal Component Analysis (PCA) algorithm. For each point, we select the k nearest neighboors, and we run a PCA analysis on thos points. This allows to determine how many directions are needed to explain a given percentage of the spreding of the data, call it ϵ. In other words, this allows to determine what is the local dimension around each point. We ran such an algorithm for a number of k nearest neighboors equal to $[5, 10, 20, 50, 100]$, and for $\epsilon = 0.99$. The results are shown in 2.

We see that, for every choice of nearest neighboor, a peak at $d = 3$, which indicates an underlying dimension of 3 for the manifold. We also see that for $k = 5$, that a number a point seems to indicate a dimensionality of 2. However, if the dimension is indeed 3, then 5 points may not be enough to sufficiently populate the three directions, explaining the failure of the algorithm to find 3. We also see that for $k \leq 20$, that there is more and more points indicating a dimension equals to 4 or 5. This can be explained by the fact that we increasing the number of neighboors, we are doing a rougher coarse graining, loosing information on the localilty. A too large number of neighboors implies including points which are not really local anymore, artificially increasing the local dimension. We conclude that the dimension of the manifold we are looking at is 3.

- Clusteting: HDBScan (only parameter: minimal number of points in a cluster), and graphs of 3d slices of the 5d space.

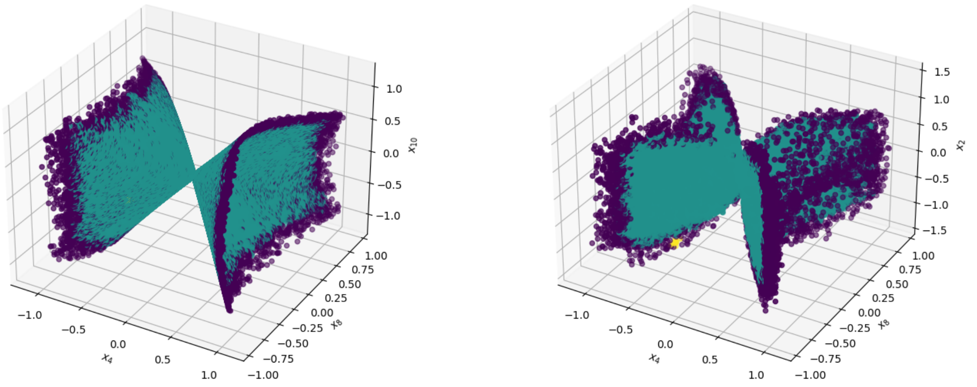

Fig. 3 3d plot of x_4, x_8 and x_{10}

Something that might happen, is that we have several manifolds of dimesion 3, and the dimension 4 points we saw above are actually the intersection of those manifolds. If we indeed picture the intersection of two lines, then at the intersection point, the local dimension found by the previous algorithm should be 2. To exclude such a scenario, we run a clustering algorithm, based on local density. For this, we use the HDBSAN algorithm. (details on how it works?). We choose a minimal cluster size of 10 points. Out of the 100000, the algorithm finds 3 cluster : one on dimension 10, one of dimension 91747 and a last one of dimension 6849 (not a total of 100000??). The last cluster are just the points that the algorithm did not managed to fit into any cluster. For the first one, we interpret it as some fluctuation in our data, making some region of different density as the other, but we do not believe this is some real result, more an artefact. Therefore, what the algorithm tells us, is that there seems to be on cluster with more than 90% of the data. We show on 3 three-dimensional projections of the data.

On these scatter plots, the turquoise points are those belonging to the main cluster, and the purple are those for which the algorithm failed to give them a cluster. On of the two plot, we can see some few yellow points which correspond to the small cluster with 10 points. We can see, from what we can tell with those plots, that all points that could not be assigned to a cluster are on the boarder of the data. Therefore, we interpret the fact that those point could not be assigned, not because there are part of another cluster, and a manifold, but because of the lack of points around them due to the fact they are on the boundary of our data. We conclude from this analysis, that we possess here a sampling of a unique 3 dimensional manifold.

3.2 Annealed Importance Sampling for polynomial symbolic regression

- We can convert the potential to a polynomial by converting the variables that appear in exponentials to logs.
- As the potential is a polynomial, the solutions satisfy polynomials equations. Discuss that fact that we could get polynomials directly by taking the gradient of the potential, but those will be to

complicated to express the vacuum in a usable way. We search those using Annealed Importance Sampling.

Now we have sampled the manifold and exctrated 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 exctract some analytic formula to caracterise 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 caracterise the manifold, we need two constraints on the embedding coordinates $\vec{x} = (x_1, x_2, x_4, x_8, x_{10})$. If we have a look at the form of the potential (2.3), we see that if we use $\tilde{x_8} = e^{x_8}$, that 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 are two of those. Of course if one takes directly the gradient of (2.3), 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, or ..., or by doing rough gradient descent on the most general polynomial with arbitrary coefficients and trying to minimize its value once evaluated on the data points. 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.

- Annealed Importance Sampling: first explain general idea (construct density probability, role of temperature, links with Monte-Carlo), discuss β to control exploration and exploitation phases.

  In the realm of symbolic regression, the objective 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 $p(data) = 0$. 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.

- Then more details: discuss hypotheses to compute the weights, choice of transformations, choice of β, choice of loss, prior, initial sampling, choice of representation for the polynomials.

  Let us now explain in more details how does this procedure goes. The goal is to reconstruct some distribution function $p(z)$, where here z in going to be some polynomial. We will try to reconstruct this density function by a serie of density function $\pi_n(z_n) = \gamma_n(z_n)/Z_n$ with $n = 1, \ldots, N$ 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 an Loss function $L(z)$

  $$\gamma_n(z) := p_0(z)\, \exp\left(-\beta_n L(z)\right) \tag{3.2}$$

  At each step, we have a set of particles $\{z^k_{n-1}, w^k_{n-1}\}$ approximating π_{n-1} (meaning $\mathbb{E}_{\pi_{n-1}}[f] \approx \frac{\sum_k w^k_{n-1} f(z^k_{n-1})}{\sum_k w^k_{n-1}}$), and we want to obtain a new set $\{z^k_n, w^k_n\}$ approximating π_n. To do so, for each particle z^k_{n-1}, we propose a new particle $z^k_n \sim q(z_n|z^k_{n-1})$ and walculate the new unnormalized importance weight w^k_n.

  Let us first focus on how we propose a new particle $z^k_n \sim q(z_n|z^k_{n-1})$ given z^k_{n-1}. To do so, we choose $q(z_n|z_{n-1})$ to be a Markov kernel

- Discuss the fact that we allow float coefficients even though we know the coefficients will be only integers of square roots of integers?

- Analysis after AIS: select the best polynomials and do exploitation on the coefficients (without MC: we keep only the proposal of it betters the polynomial).

- 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 (2.3). For 100 runs with parameters

$$\begin{gathered} n_{\text{iter}} = 1000, \quad n_{\text{particles}} = 1000, \quad \text{reg} = 10^3, \quad \beta = 10^{-10} + \left(\frac{i}{n_{\text{iter}}}\right)^5, \quad \sigma = 0.5 - \frac{0.45}{1 + \exp\left(10 - 20i/n_{\text{iter}}\right)}, \\ \text{sample size} = 10000, \quad p_{\text{add}} = 0, \quad p_{\text{remove}} = 0, \quad p_{\text{modify}} = 0.5, \quad p_{\text{multiply}} = 0.25, \quad p_{\text{divide}} = 0.25, \end{gathered} \tag{3.3}$$

the code finds the 7 different following polynomials:

$$
\begin{align}
p_1 &= -\sqrt{2}\,x_1 + \sqrt{2}\,x_1\tilde{x}_8 + x_2\tilde{x}_8 x_{10} - \sqrt{2}\,x_2 x_4 \tilde{x}_8, \tag{3.4a}\\
p_2 &= 2\,x_2 + 2\,x_2\tilde{x}_8 + \sqrt{2}\,x_1 x_{10} + 2\,x_1 x_4 - x_2\tilde{x}_8 x_{10}^2, \tag{3.4b}\\
p_3 &= 2\,x_2 - 2\,x_2\tilde{x}_8 + \sqrt{2}\,x_1\tilde{x}_8 x_{10} + 2\,x_1 x_4 \tilde{x}_8 - 2\,x_2 x_4^2 \tilde{x}_8, \tag{3.4c}\\
p_4 &= \sqrt{2}\,x_2 - \sqrt{2}\,x_2\tilde{x}_8 + \sqrt{2}\,x_1\,x_4 + x_1\tilde{x}_8 x_{10} - x_2 x_4 \tilde{x}_8 x_{10}, \tag{3.4d}\\
p_5 &= -2\,x_1 - 2\,x_2 x_4 - 2\,x_1 x_4^2 + 2\,x_1\tilde{x}_8 + \sqrt{2}\,x_2 x_{10} + x_1\tilde{x}_8 x_{10}^2, \tag{3.4e}\\
p_6 &= -2\,x_2 x_4 - 2\,x_1 x_4^2 + 2\,x_2 x_4 \tilde{x}_8 + \sqrt{2}\,x_2 x_{10} - \sqrt{2}\,x_2\tilde{x}_8 x_{10} + x_1\tilde{x}_8 x_{10}^2, \tag{3.4f}\\
p_7 &= -\sqrt{2}\,x_1^2 x_4 + \sqrt{2}\,x_2^2 x_4 + \sqrt{2}\,x_1 x_2 x_4^2 - x_1^2 x_{10} - x_2^2 x_{10}, \tag{3.4g}
\end{align}
$$

where $\tilde{x}_8 = e^{x_8}$, with the distribution:

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

(3.5)

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 $\sim 1400\,\mathrm{s}$.

4 Supergravity solutions

- For each couple of candidate polynomials, we get the same expression for the solution, and it indeed defines a unique vacuum of the 3d SUGRA.
- Discussion of the vacuum: gauge group, Zham. metric, change of variables, 3d spectrum (and stability), spin 2 spectrum on S^3, one parameter seems to be a gauge parameter, uplift?

For every couples (p_i, p_j) of polynomials in eq. (3.4), the system

$$
\begin{cases}
p_i = 0,\\
p_j = 0,
\end{cases} \tag{4.1}
$$

gives the same solution

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

We checked that this defines a 3-parameter solution of three-dimensional half-maximal supergravity. Equivalently:

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

The solution preserves a U(1) × U(1) gauged symmetry.

Moduli space 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

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

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

Bosonic spectrum Vector fields:

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

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:

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

Gravitini:

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

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?