RESEARCH ARTICLE

Fortschritte der Physik
Progress of Physics

www.fp-journal.org

Evolutionary Algorithms for Multi-Center Solutions

*Sami Rawash and David Turton**

Large classes of multi-center supergravity solutions have been constructed in the study of supersymmetric black holes and their microstates. Many smooth multi-center solutions have the same charges as supersymmetric black holes, with all centers deep inside a long black-hole-like throat. These configurations are constrained by regularity, absence of closed timelike curves, and charge quantization. Due to these constraints, constructing explicit solutions with several centers in generic arrangements, and with all parameters in physically relevant ranges, is a hard task. In this work, an optimization algorithm, based on evolutionary algorithms and Bayesian optimization is presented, that systematically constructs numerical solutions satisfying all constraints. Explicit examples of novel five-center and seven-center machine-precision solutions are exhibited.

1. Introduction

Black hole solutions in classical gravitational theories typically involve a small handful of parameters, such as mass, angular momentum, and charge. However black holes have an entropy proportional to their horizon area, which suggests that they have a vast number of internal degrees of freedom. Black holes also contain curvature singularities, and the semiclassical description of black hole evaporation leads to the information paradox.[1,2] These facts present three corresponding major challenges for a fundamental theory of quantum gravity: to identify the black hole internal degrees of freedom, to resolve the singularities inside black holes, and to provide a consistent description of black hole evaporation.

In String Theory, black hole entropy arises from an exponential number of internal quantum microstates.[3] It is therefore of significant interest to study the gravitational description of heavy pure states, in order to investigate string-theoretic singularity resolution and black hole evaporation. Large families of such pure states are well-described by smooth, horizonless supergravity solutions which, in the best-understood examples, provide a valuable description of black hole microstates.[4–7]

S. Rawash, D. Turton
Mathematical Sciences and STAG Research Centre
University of Southampton
Highfield, Southampton SO17 1BJ, UK
E-mail: d.j.turton@soton.ac.uk

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/prop.202300255

DOI: 10.1002/prop.202300255

In recent years, an increasing number of String Theory, and Particle Physics, problems have been addressed with optimization algorithms and machine learning, see e.g., refs. [8–15]. In this work we present an algorithm, based on evolutionary algorithms (EAs) and Bayesian optimization, to construct smooth horizonless supergravity solutions. The category of microstate solutions that we study are supersymmetric multi-center solutions, also known as bubbling solutions.[16–25]

Supersymmetric multi-center solutions involve non-trivial topology supported by flux. These solutions are specified by a set of harmonic functions on a three-dimensional Euclidean space. This formalism is typically used to construct supergravity fields in four, five or six macroscopic dimensions. Depending on the details, large families of multi-center solutions in five and/or six macroscopic dimensions can have the features of being horizonless and smooth, up to possible orbifold singularities, see e.g., ref. [19]. We shall moreover focus on the "scaling" regime in which the centers lie deep inside a long black-hole-like throat.[18–20]

Multi-center solutions are one of two main classes of smooth horizonless supersymmetric solutions. The other class is known as *superstrata*.[26–36] Of the two classes, superstrata have a proposed holographic description that has passed precision holographic tests, involving protected correlators that can be reliably compared across moduli space between supergravity and the dual symmetric product orbifold CFT.[37–42] Two-center bubbling solutions have a similarly well-established holographic description[43–48] and also a string worldsheet description.[49–56] By contrast, multi-center solutions with three or more centers do not have a proposed holographic description, and it has been argued that they do not describe microstates of a single supersymmetric black hole, though they could describe microstates of other black objects.[57] Nevertheless, multi-center solutions provide interesting examples of gravitational solutions that closely resemble black holes, especially in the scaling regime. Indeed, multi-center solutions have been used to investigate potentially observable signatures of string theoretic black hole microstructure in gravitational wave observations.[58–60]

Constructing multi-center solutions with several centers is a hard problem. This is because asymptotic flatness, charge quantization, smoothness and absence of closed timelike curves (CTCs) comprise a set of non-trivial algebraic constraints. These constraints make the positions of the centers and the coefficients of the poles of the harmonic functions highly interdependent. The most important set of these constraints is known as the bubble equations. The distances between the centers are generically irrational real quantities. The bubble equations constrain these

distances in terms of quantized parameters; this is a strong set of constraints. For further discussion, see e.g., ref. [21].

Despite this difficulty, several solutions with three or four centers have been constructed analytically, see e.g., refs. [18, 20, 22–25]. However, fewer solutions with five or more centers have been constructed, and until relatively recently these typically involved taking all centers to lie on a line, so that the configuration is axisymmetric, see e.g., ref. [18]. An important step forward was recently made, by considering the dependent variables in the bubble equations to be a subset of the coefficients of the poles of the harmonic functions (which we call "flux parameters"), rather than the distances between the centers.[21] In this form, the bubble equations are a linear system, involving a symmetric matrix $\mathcal{M}$. It was moreover conjectured that a configuration does not contain CTCs if and only if $\mathcal{M}$ is positive-definite. While this perspective simplifies the task of finding physically relevant solutions, the strong nature of the constraints between generically irrational distances and quantized parameters remains.

One can construct exact solutions with this method by arranging non-generic locations of centers, however this is not easy to implement in practice, and is currently limited to quite special arrangements of centers.[21] An alternative approach is to construct approximate solutions to the bubble equations, as discussed in ref. [21] and done in refs. [22, 23]. One can do so in an iterative approach by first choosing a set of locations of the centers and solving for the flux parameters, obtaining generically irrational values. One then rounds any irrational flux parameters to nearby rational values that enable all quantization constraints to be satisfied. One then takes the rounded fluxes and attempts to re-solve the bubble equations in the traditional approach, to find the distances between the centers. This method has been used to numerically construct four-center solutions in axisymmetric or near-axisymmetric configurations.[23] However, it is unclear whether this method is generically tractable for more than four centers.[21]

In this paper we present a novel algorithmic method to construct numerical solutions with any number of centers, and with no symmetry imposed on the locations of the centers. The basic idea is to proceed in two steps. First, we generate a suitably good starting configuration which has certain desired physical properties, but is not yet a solution to the bubble equations. Second, we systematically vary the positions of the centers to construct a sequence of approximate solutions with increasing precision.

These two steps require two separate algorithms, since they optimize over different variables. In the first step, we generate a good starting configuration by optimizing over flux parameters. In the second step, we optimize over the locations of the centers.

The starting configuration is required to have two key physical properties. The first requirement is that the configuration be in the scaling regime mentioned above, in which the centers lie deep inside a long black-hole-like throat. We implement this by first finding an exact solution to the *homogeneous* form of the bubble equations, which will be reviewed in Section 2. We then round the flux parameters as described above.

The second requirement is that the supergravity charge radii are large, such that the solutions are weakly curved. We shall primarily have in mind solutions corresponding to bound states of D1 branes, D5 branes, and momentum P in a compact direction, in five or six macroscopic dimensions. In six dimensions, the (dimensionful) D1 and D5 charges, Q_1, Q_5 control the main curvature scale of the solution (and contribute to the ADM mass), so we require them to be appropriately large.

The charges Q_1, Q_5, considered as functions of the flux parameters, are computationally expensive to evaluate. Bayesian optimization is well-suited to the task of optimizing computationally expensive functions (see e.g., ref. [61]); we therefore employ it for the first step. The reason that the charges are computationally expensive to evaluate is that, given some flux parameters, one must first solve the bubble equations for the remaining flux parameters, and then evaluate the expressions for the charges Q_1, Q_5, which will be given in Section 2.

In a nutshell, Bayesian optimization is a strategy to choose points (in our case, values of flux parameters) on which to evaluate, or sample, the function to be optimized, known as the "objective" function (in our case, $\min(Q_1, Q_5)$). After a point is sampled, the accrued knowledge of the objective function is updated, and then used to decide the next point to sample. The next point is selected according to a specified strategy that balances exploitation of more favourable regions (where Q_1, Q_5 are known to be larger) versus exploration of unknown regions (where Q_1, Q_5 have not yet been computed). We will describe this in detail in Section 3.2.

After a successful run of the Bayesian optimization algorithm, we have a configuration in the scaling regime, with appropriately large charges Q_1, Q_5, which is not yet a solution. It can be regarded as an approximate solution, but with low precision. In the second step, we construct numerical solutions by varying the positions of the centers.

Our two-step approach means that after a successful first step, it is reasonable to expect that if there is a genuine solution nearby, it is likely to require incremental adjustments to the positions of the centers, rather than a wide search. EAs are well-suited to problems where incremental changes result in incremental improvements (see e.g., ref. [12]). We therefore employ an EA in the second step.

EAs work by generating a population of *individuals*, comprising certain data known as *genes*, and quantifying their *fitness* via a function known as the fitness function. The algorithm then generates subsequent generations of individuals following the principles of the Darwinian theory: selection, reproduction and mutation. By iterating this process over several generations, the algorithm aims to construct new individuals with higher fitness.

In our algorithm, an individual is a multi-center supergravity configuration that approximately solves the bubble equations (in their full, *inhomogeneous* form). Its genes are (an appropriate subset of) the positions of the centers. The coefficients of the poles of the harmonic functions are determined by the previous step. The fitness function quantifies the precision to which the bubble equations are approximately satisfied, with higher fitness being an approximate solution with a lower error. Once a multi-center configuration with the desired fitness is generated, we investigate the absence of CTCs by computing the eigenvalues of the matrix $\mathcal{M}$.

Our algorithm is designed to generate solutions with any number of centers, in a generic configuration. We have run the algorithm on several configurations of three, five and seven centers, and we shall exhibit explicit examples of novel five- and seven-center configurations. Generating solutions with a higher number of centers is feasible, although naturally this is computationally more expensive. The algorithm is implemented in Python, and the code is publicly available.[1]

This paper is structured as follows. In Section 2, we review multi-center scaling solutions, and their construction in the formalism of ref. [21]. In Section 3, we first describe our overall method, and then describe the Bayesian optimization algorithm and the EA we have developed. In Section 4, we describe explicit examples of five-center and seven-center scaling configurations obtained with our method, and comment on the performance of the algorithm. We discuss our results in Section 5.

2. Multi-Center Scaling Supergravity Solutions

2.1. Multi-Center Solutions

For concreteness, we primarily consider 5D $\mathcal{N}=1$ Super-Einstein-Maxwell-Yang-Mills supergravity, whose bosonic field content is the metric, three Abelian vector multiplets, and an SU(2) triplet of non-Abelian vector multiplets. If one turns off the non-Abelian multiplets, one recovers the STU model.

Multi-center solutions are specified by a set of harmonic functions on a three-dimensional Euclidean "base" space, which have poles at the location of the centers. The index $a=0,1,\ldots,n-1$ labels the centers, and $r_a=|\vec{r}-\vec{r}_a|$ is the distance from the a-th center in the three-dimensional base. In the Abelian sector the harmonic functions are ($i=0,1,2$):

$$H=\sum_{a=0}^{n-1}\frac{q_a}{r_a}\,,\qquad K^i=\sum_{a=0}^{n-1}\frac{k_a^i}{r_a},\qquad L^i=l_0^i+\sum_{a=0}^{n-1}\frac{l_a^i}{r_a},$$

$$M=m_0+\sum_{a=0}^{n-1}\frac{m_a}{r_a}, \tag{2.1}$$

where $q_a\in\mathbb{Z}$. In the non-Abelian sector,[62] denoting the gauge coupling by g, we have

$$P=1+\sum_{a=0}^{n-1}\frac{\lambda_a}{r_a}\,,\qquad Q=\sum_{a=0}^{n-1}\frac{\sigma_a\lambda_a}{r_a}\,. \tag{2.2}$$

The harmonic function H defines a four-dimensional Gibbons-Hawking metric via

$$ds_4^2=H^{-1}(d\psi+A)^2+Hds_3^2\,, \tag{2.3}$$

where ds_3^2 is the flat metric on $\mathbb{R}^3$, and A is a one-form related to H via $\star_3\,dA=dH$. For the full five-dimensional fields, we refer the reader to ref. [21].

[1] GitHub URL: https://github.com/SamiRawash/Multicenter-Scaling-Solutions.

Only certain subsets of possible coefficients of the poles in Equations (2.1) and (2.2) lead to physically sensible solutions: one needs to impose further constraints. First, asymptotic flatness requires $\sum_a q_a=1$. Second, upon uplifting to Type IIB supergravity compactified on $S^1\times T^4$, the coefficients k_a^i are quantized in terms of integer flux parameters n_a^i as follows,[47]

$$k_a^0=\frac{g_s\alpha'}{2R_y}n_a^0\,,\qquad k_a^1=\frac{g_s\alpha'^3}{2V_4R_y}n_a^1\,,\qquad k_a^2=\frac{R_y}{2}n_a^2\,, \tag{2.4}$$

where the coordinate volume of T^4 is $(2\pi)^4V_4$ and that of the S^1 is $2\pi R_y$.

In this paper we focus on smooth horizonless supersymmetric solutions.[2] The following relations are imposed by absence of event horizons and singularities (the first three relations) and asymptotic flatness (the last two relations), see e.g., ref. [21, App. A.3],

$$\begin{aligned}
&l_a^i=-\frac{|\epsilon^{ijk}|}{2}\frac{k_a^jk_a^k}{q_a}+\frac{\delta^{0i}}{2g^2}\,,\qquad \sigma_a=\frac{k_a^0}{q_a}\,,\\
&m_a=\frac{k_a^0}{2q_a^2}\left(k_a^1k_a^2-\frac{1}{2g^2}\right),\\
&l_0^0l_0^1l_0^2=1\,,\qquad m_0=-\frac{1}{2}\sum_{a,i}l_0^ik_a^i.
\end{aligned} \tag{2.5}$$

The absence of Dirac-Misner singularities imposes the so-called "bubble equations",[19,62,68] which constrain the relation between the positions of the centers and the local charges:

$$\sum_{b\neq a}\frac{q_aq_b}{r_{ab}}\Pi_{ab}^0\left(\Pi_{ab}^1\Pi_{ab}^2-\frac{1}{2g^2}\mathbb{T}_{ab}\right)=\sum_{b,i}q_aq_bl_0^i\Pi_{ab}^i\,, \tag{2.6}$$

where

$$\Pi_{ab}^i=\frac{k_b^i}{q_b}-\frac{k_a^i}{q_a},\qquad \mathbb{T}_{ab}=\frac{1}{q_a^2}+\frac{1}{q_b^2}\,. \tag{2.7}$$

Here r_{ab} is the $\mathbb{R}^3$ Euclidean distance between centers a and b, Π_{ab}^i are the magnetic fluxes, and we will refer to the coefficients k_a^i as flux parameters. The bubble equations are a set of n equations among which only $(n-1)$ are independent: summation over a leads to a trivial identity, due to the antisymmetry of the Π_{ab}^i.

[2] The supersymmetric multi-center formalism can also be used to construct solutions with physical singularities such as shockwaves,[63] which give collective descriptions of families of pure states. Similar but different multi-center formalisms exist for non-supersymmetric solutions.[64–67]

The asymptotic charges of the multi-center solutions are:[19][3]

$$\begin{aligned} Q_1 &= -\sum_{a,b,c} q_a q_b q_c \Pi^1_{ab}\Pi^2_{ac} + \frac{1}{2g^2}\sum_a \frac{1}{q_a}\,, \\ Q_5 &= -\sum_{a,b,c} q_a q_b q_c \Pi^0_{ab}\Pi^2_{ac}\,, \\ Q_p &= -\sum_{a,b,c} q_a q_b q_c \Pi^0_{ab}\Pi^1_{ac}\,, \\ J_L &= -\frac{1}{2}\sum_{a,b,c,d} q_a q_b q_c q_d \Pi^0_{ab}\Pi^1_{ac}\Pi^2_{ad} + \frac{1}{4g^2}\sum_{a,b}\frac{q_b \Pi^0_{ab}}{q_a}\,, \\ \vec{J}_R &= \frac{1}{4}\sum_{a,b,a\neq b} q_a q_b \Pi^0_{ab}\left(\Pi^1_{ab}\Pi^2_{ab} - \frac{1}{2g^2}\mathbb{T}_{ab}\right)\frac{\vec{r}_a - \vec{r}_b}{|\vec{r}_a - \vec{r}_b|}\,. \end{aligned} \tag{2.8}$$

2.2. Scaling Solutions and Their Construction

Of particular interest are solutions to the bubble equations (2.6) in which the distances between the centers can be made uniformly parametrically small by scaling $r_{ab} \to \lambda r_{ab}$ with $\lambda \ll 1$, while keeping the asymptotic charges approximately constant. These solutions are known as "scaling" solutions.[18–20] Note that the rescaling $r_{ab} \to \lambda r_{ab}$ is equivalent to multiplying the RHS of (2.6) by λ, with $\lambda \ll 1$. It will be useful for us to note that in the limit $\lambda \to 0$, one obtains the homogeneous bubble equations[20] (see also for instance[23]),

$$\sum_{b\neq a}\frac{q_a q_b}{r_{ab}}\Pi^0_{ab}\left(\Pi^1_{ab}\Pi^2_{ab} - \frac{1}{2g^2}\mathbb{T}_{ab}\right) = 0\,. \tag{2.9}$$

Therefore, in the scaling regime of small λ, solutions to the full inhomogeneous bubble equations (2.6) are also approximate solutions to the homogeneous bubble equations (2.9), up to terms of order λ. We will exploit this to construct new scaling solutions.

The full inhomogeneous bubble equations (2.6) have typically been considered as equations in which the variables to be solved for are the distances r_{ab}, see e.g., ref. [19]. This perspective has two disadvantages.[21] First, it is generically difficult to find solutions for r_{ab}. Second, after solving the equations, one often finds that the resulting r_{ab} do not represent possible distances between points in 3D Euclidean space; for instance, the triangle inequality might not be respected.

A recently developed alternative approach is to exploit the feature that the bubble equations (2.6) are linear in the flux parameters k^2_a. Thus, instead of solving for the distances, one can first specify the positions of the centers, and then solve for the flux parameters k^2_a with $a = 2, 3, \ldots n$.[21][4] While this procedure is general and not restricted to scaling solutions, let us now review it in the context of scaling solutions. We introduce a scaling parameter λ that rescales the positions of the centers while keeping the shape of the distribution fixed: i.e., we write the distance between the centers as $r_{ab} = \lambda d_{ab}$, where d_{ab} remain constant in the scaling process. We define[5]

$$\begin{aligned} \bar{A}^2_{ab} &= \frac{q_a q_b}{d_{ab}}\Pi^0_{ab}\Pi^1_{ab}, \quad \dot{A}^2_{ab} = -s q_a q_b l^2_0, \\ \bar{B}^2_{ab} &= \sum_{b=0}^{n-1}\frac{q_a q_b}{d_{ab}}\frac{1}{2g^2}\mathbb{T}_{ab}\Pi^0_{ab}, \quad \dot{B}^2_a = s\sum_{b=0}^{n-1} q_a q_b (l^0_0\Pi^0_{ab} + l^1_0\Pi^1_{ab}), \end{aligned} \tag{2.10}$$

where we have introduced the constant s which takes values 0 or 1. These values correspond respectively to the homogeneous and inhomogeneous bubble equations, as we shall see momentarily. We then introduce $(\alpha\,, \beta = 1, \ldots, n-1)$[6]

$$\begin{aligned} \bar{\mathcal{M}}^2_{\alpha\beta} &= \bar{A}^2_{(\alpha+1)(\beta+1)} - \delta^\beta_\alpha \sum_{c=0}^{n-1}\bar{A}^2_{(\alpha+1)c}\,, \\ \dot{\mathcal{M}}^2_{\alpha\beta} &= \dot{A}^2_{(\alpha+1)(\beta+1)} - \delta^\beta_\alpha \sum_{c=0}^{n-1}\dot{A}^2_{(\alpha+1)c}\,, \end{aligned} \tag{2.11}$$

in terms of which, we write the following linear system of equations in the fluxes Π^2_{ab}:

$$\mathcal{M}^2_{\alpha\beta}\Pi^2_{1(\alpha+1)} \equiv \left(\bar{\mathcal{M}}^2_{\alpha\beta} + \lambda\dot{\mathcal{M}}^2_{\alpha\beta}\right)\Pi^2_{1(\alpha+1)} = \bar{B}^2_\beta + \lambda\dot{B}^2_\beta\,. \tag{2.12}$$

For $s = 1$ this linear system is equivalent to the inhomogeneous bubble equations (2.6), while for $s = 0$ the system is equivalent to the homogeneous bubble equations (2.9).

Although this perspective has simplified the task of solving the bubble equations, it remains a fact that generic solutions obtained in this way will not respect the quantization conditions in Equation (2.4). This can be seen as follows. If we choose generic locations of the centers, generic relative distances will be irrational numbers. Then generic solutions will give irrational values of the flux parameters k^2_α, which is in conflict with the quantization conditions in Equation (2.4).

As described in the Introduction, using this method one can construct exact solutions with quantized fluxes by arranging a set of non-generic locations of centers, such that all relative distances are rational. For instance one can take all centers to lie on a line, or on a circle, as discussed in ref. [21].[7] While these constructions provide interesting and valuable exact solutions, the requirement to work with non-generic locations of centers is a significant limitation.

To proceed further, an alternative approach is to construct approximate solutions to the bubble equations. One can do so with an iterative approach, as follows. One first chooses a set of center locations, then solves for k^2_α, generically obtaining irrational values. One then rounds the k^2_α to nearby rational numbers to a desired precision, obtaining an approximate solution, as discussed in ref. [21] and done in refs. [22, 23].

[3] We use conventions in which J_L and J_R are interchanged with respect to ref. [19].

[4] The bubble equations are also linear in $k^{0,1}_a$, so a similar analysis can be carried out for them.

[5] A numerical typo in ref. [21, Eq. (3.22)] has been corrected.

[6] To be clear, the '2' are superscript labels for the value of the index i, not exponents. To avoid potential confusion on this point, we have suppressed the superscript '2' on the matrix $\mathcal{M}^2$ in the Introduction and Discussion sections.

[7] For earlier examples of solutions with all centers on a line, see e.g., ref. [18].

This can be further improved by taking the rounded flux parameters k_α^2, re-solving the bubble equations (in the traditional way) to obtain a new set of distances r_{ab}, and then arranging center positions to have the resulting relative distances. If this could be done analytically, one can obtain exact solutions, however typically this is hard, for the reasons discussed below Equation (2.9). More realistically, one can employ this method to improve the precision of the approximate solution, as done in ref. [23]. However, for more than four centers, doubt has been expressed as to the feasibility of this method.[21]

3. Constructing Numerical Scaling Solutions

3.1. Overview of the Method

In this section we describe our method to construct numerical solutions. The method involves two main steps. In the first step, we generate an approximate scaling solution with rounded flux parameters k_α^2 and large charges Q_1, Q_5. The condition of large charges is imposed using a Bayesian optimization algorithm, optimizing over flux parameters. In the second step, we fix the flux parameters and optimize instead over the locations of the centers, using an EA.

In this subsection we first describe the overall method, focusing primarily on the first step of generating an appropriately good starting configuration. The following subsections will describe the respective algorithms in detail.

A key ingredient in the first step is the construction of configurations in the scaling regime. To do this, we follow the discussion below (2.9). We solve the homogeneous bubble equations (2.9) in the form (2.12) by taking the position of the centers $\vec{r}_a$ and the coefficients q_a, l_0^i, $k_a^{0,1}$ and k_0^2 to be independent variables, and solving for k_α^2 (recall $\alpha = 1, \ldots, n-1$). The initialization/optimization of the independent variables will be described in the next subsection.

This will enable us, at a later step, to rescale $r_{ab} \to \lambda r_{ab}$ with $\lambda \ll 1$ to obtain a configuration in the scaling regime, as done in ref. [23].[8] However, before doing this rescaling we round the flux parameters, and impose $Q_1, Q_5 > \tilde{Q}$, as follows.

We round the flux parameters, $k_\alpha^2 \to \tilde{k}_\alpha^2$, to a certain precision, which will be a hyperparameter of the algorithm, and which we call `k_rounding`. From here onwards, tildes denote rounded quantities. After rounding, Equation (2.9) will no longer be exactly satisfied for the same configuration of centers.

Having constructed an approximate solution to the homogeneous bubble equations, we then impose $Q_1, Q_5 > \tilde{Q}$ using a Bayesian optimization algorithm, optimizing over a subset of the independent flux parameters, and iterating over the steps so far, as described in the next subsection.

After a successful run of the Bayesian optimization algorithm, we have an approximate solution to the homogeneous bubble equations, with charges Q_1, Q_5 in the desired range. We next generate another approximate solution in the scaling regime by rescaling the positions of the centers obtained in Equation (3.5):

$$r_a^i \to \bar{r}_a^i \equiv \lambda\, r_a^i\,, \qquad \lambda \ll 1\,, \tag{3.1}$$

where for concreteness we take $\lambda = 10^{-5}$.

At this stage, we have a starting configuration with the desired physical properties. Before using it as an input to the EA, we next impose two conditions that further indicate whether the configurations can be considered sufficiently good starting configurations.

To describe the first condition, let us consider the homogeneous bubble equations (2.9) for $a = n-1$:

$$\sum_{b\neq n-1} \frac{q_{n-1}q_b}{r_{(n-1)b}} \Pi^0_{(n-1)b}\left(\Pi^1_{(n-1)b}\Pi^2_{(n-1)b} - \frac{1}{2g^2}\mathbb{T}_{(n-1)b}\right) = 0\,. \tag{3.2}$$

Let us further examine the generic case in which all the terms in this sum are non-zero. Then a necessary condition to have a solution is that not all of the terms in the sum have the same sign. Since the distances are positive, the expressions $q_{n-1}q_b\Pi^0_{(n-1)b}(\Pi^1_{(n-1)b}\Pi^2_{(n-1)b} - \frac{1}{2g^2}\mathbb{T}_{(n-1)b})$ should not all have the same sign.

After rounding the flux parameters, $k_\alpha^2 \to \tilde{k}_\alpha^2$, we have rounded fluxes $\tilde{\Pi}^2$. In the EA, we will keep these fixed and change the location of the centers. So, before running the EA, we examine whether or not all of the following expressions have the same sign (note the presence of the rounded fluxes $\tilde{\Pi}^2$):

$$q_{n-1}q_b\Pi^0_{(n-1)b}\left(\Pi^1_{(n-1)b}\tilde{\Pi}^2_{(n-1)b} - \frac{1}{2g^2}\mathbb{T}_{(n-1)b}\right). \tag{3.3}$$

We have found that if these quantities have the same sign, it is a reliable indicator that there is unlikely to be a nearby scaling solution. Therefore, only if these quantities do not all have the same sign, we proceed.

Next, we perform a preliminary investigation of the absence of CTCs. Let us first review the case in which only Abelian fields are turned on. To rule out CTCs, two algebraic combinations of the harmonic functions (2.1) must be globally positive. Generically, the stronger of these conditions is that the quartic $E_{7(7)}$ invariant, as a function of the harmonic functions (2.1), is globally positive.[69] Investigating this condition is non-trivial, and typically done numerically. The generalization to configurations with both Abelian and non-Abelian fields was discussed in refs. [21, 62]. The authors of ref. [21] conjectured that the condition for absence of CTCs is equivalent to requiring that the matrix $\mathcal{M}^2$ defined in Equation (2.12) is positive-definite. We thus investigate absence of CTCs in the solutions found by the algorithm by examining this condition on $\mathcal{M}^2$.

Although $\mathcal{M}^2$ depends on the positions of the centers, and thus will be modified by the EA, we have observed that small modifications of the distances do not tend to change the eigenvalues much. Of course, after the EA, one must recheck the condition on $\mathcal{M}^2$. However, checking the condition at this stage provides a good indication of whether the condition will be respected in the final solution. If $\mathcal{M}^2$ is positive definite, we proceed to use this configuration as a seed for the EA.

Note that the condition (2.9) for scaling solutions, and the scaling limit $r_{ab} \to \lambda r_{ab}$, correspond to "zooming in" to the core of the solutions, such that the asymptotics become AdS_2 fibered over S^3; for a general discussion, see ref. [70]. We wish to "undo" this limit and construct solutions with $\mathbb{R}^{4,1}$ asymptotics. This re-

[8] We thank Pierre Heidmann for a discussion on this point.

quires restoring the inhomogeneous terms in the bubble equations (2.6).

In the second step of the method, the EA modifies the positions of the centers to construct numerical solutions to the full inhomogeneous bubble equations (2.6), as we will describe in Section 3.3.

3.2. The Bayesian Optimization Algorithm

We now describe the Bayesian optimization algorithm that we use to implement the first part of our method. As described in the previous section, we begin by specifying the positions of the centers $\vec{r}_a$ and the coefficients q_a, l_0^i, $k_{\dot{a}}^{0,1}$ and k_0^2, considering them to be independent variables, and solving the homogeneous bubble equations (2.9) for k_α^2 (recall $\alpha = 1, \dots, n-1$).

We wish to impose large charges, $Q_1, Q_5 > \bar{Q}$, for some appropriate $\bar{Q}$. To do so, in principle one could attempt to maximise the first two expressions in Equation (2.8) as functions of both k_0^2 and the full set of $k_{\dot{a}}^{0,1}$. However in practice, it is neither computationally efficient, nor necessary, to maximize Q_1, Q_5 with respect to a substantial fraction of the flux parameters $k_{\dot{a}}^{0,1}$ — it suffices to focus on the flux parameters of a small number of the centers. We thus introduce a hyperparameter n_{B} and maximize Q_1, Q_5 only with respect to the flux parameters of n_{B} of the centers. In our examples, it will suffice to take n_{B} to be equal to 1 or 2.

The centers whose flux parameters are dependent variables of the optimization process will be labeled with the index $\bar{a} = 0, \dots, n_{\text{B}} - 1$, and the remainder will be labeled with the index $\dot{a} = n_{\text{B}}, \dots, n-1$. In other words, we will fix the value of the flux parameters $k_{\dot{a}}^{0,1}$ when initializing the algorithm, and we then maximize the value of the global charges with respect to k_0^2 and $k_{\bar{a}}^{0,1}$.

In order to maximize the value of the global charges we use a Bayesian optimization algorithm. The reason for this choice is that the global charges Q_1, Q_5 in (2.8) are computationally expensive to evaluate: for any given trial configuration, one must first solve the homogeneous bubble equations for k_α^2, and then use the result to compute Q_1, Q_5.

3.2.1. Bayesian Optimization

Let us now provide an intuitive description of how the Bayesian optimization algorithm works. Bayesian optimization (BO) is an approach to find the global maximum (or minimum) of a "black-box" function, called the objective function. By black-box function, we mean either a function over which we have no analytic control (for example a stochastic function), or a function that is computationally expensive to evaluate, as in the case at hand. This means that we do not have a global knowledge of the function, i.e., we do not know its value on every point of the domain, however we have the freedom to evaluate it on a finite set points.

In this context, Bayesian optimization algorithm is a strategy to obtain the maximum of such functions that works better than a random search. It works as follows (for a review, see e.g., ref. [61]). First, a Gaussian process prior is placed on the objective function. Then, the objective function is evaluated on a set of points $[x_1, \dots, x_{n_0}]$ of the domain. At this stage, the data $\{[x_1, \dots, x_{n_0}], [f(x_1), \dots, f(x_{n_0})]\}$ represent all our knowledge on the objective function. Of course, there are infinitely many functions whose value is $[f(x_1), \dots, f(x_{n_0})]$ when evaluated on $[x_1, \dots, x_{n_0}]$, but, by assuming that the objective function follows a Gaussian process model, we estimate that not all such functions are equally probable.

In the next step, we construct a "surrogate" function, which, among the infinite functions that have the same value of the objective on the points $[x_1, \dots, x_{n_0}]$, has the highest probability of representing the objective[9]: as such, it is our best estimate of the objective based on the knowledge we have so far, and has the advantage of being much quicker to evaluate.

The last ingredient of the algorithm is the acquisition function, also known as acquisition strategy. By evaluating the *surrogate* function on a finite set of points of the domain, it chooses the next sampling point of the objective (i.e., the point of the domain that is more likely to pay off when the objective is evaluated on it), according to some specified strategy. There are many different possible acquisition strategies; see e.g., ref. [61].

By evaluating the objective function on this new point, we increase the knowledge we have on the objective. Thus, after each additional sampling point the surrogate function is updated, and the acquisition function is again used to choose the next sampling point, iterating until an acceptably good approximate maximum (in our case, $Q_1, Q_5 > \bar{Q}$) of the objective is found, or a previously set computational limit is reached (in our case, a maximum number of iterations N).

3.2.2. Implementation

We now explain this first part of our algorithm in more detail. We initialize the independent variables as follows. The q_a must all sum to 1. If n is odd, we use alternating values ± 1, starting and ending with 1. If n is even, the first $n-1$ use alternating values ± 1, starting and ending with -1, while $q_n = 2$:

$$q_a = \begin{cases} (1, -1, 1, \dots, -1, 1) & n \text{ odd} \\ (-1, 1, -1, \dots, -1, 2) & n \text{ even} \end{cases} \quad ; \quad l_0^i = 1 \ \ \forall\, i. \tag{3.4}$$

Furthermore, we choose the position of the n centers so that they lie inside a cube with edge length equal to two. We set up the coordinate system in such a way that the first center is at the origin, the second is at $y = 0, z = 0$ and the third center is at $z = 0$, so that we have a total of $3n - 6$ free coordinates. We sample the remaining non-zero r_a^i from the following uniform distribution:

$$r_a^i \ \sim \ U(-1, 1)\,. \tag{3.5}$$

In practice, this sampling is obtained from a discrete distribution whose step-size is controlled by the hyperparameter `prec_pos`. Similarly, the coefficients $k_{\dot{a}}^{0,1}$ are sampled with the discrete uniform distribution $U(-10^{\texttt{prec\_k}}, 10^{\texttt{prec\_k}})$, with step-size equal to 1. In the following we will set the parameter `prec_k` $= 2$.

[9] This is why this optimization algorithm is called Bayesian: given the knowledge on the objective, the prior is used to obtain a posterior, which in this case is the surrogate function.

Algorithm 1 BO algorithm

function $f_{\text{OBJ}}\{k_{\vec{a}}^{0,1}, k_0^2\}$
  $k_\alpha^2 \leftarrow$ Solve the homogeneous bubble equations (2.9)
  $\tilde{k}_\alpha^2 \leftarrow$ Round k_α^2
  Compute Q_1, Q_5 via (2.8)
  return min(Q_1, Q_5)
Assume Gaussian process prior
Evaluate f_{OBJ} on $\mathbf{n_0}$ points $\{k_{\vec{a}}^{0,1}, k_0^2\}$ sampled with $U(-10^2, 10^2)^{\otimes(2n_B+1)}$
$Q_{\max} \leftarrow$ The maximum output of f_{OBJ} found so far
Generate the surrogate function using the available data
while $n < N$ or $Q_{\max} < \bar{Q}$ **do**
  Let $\{k_{\vec{a}}^{0,1}, k_0^2\}_n$ be the point returned by the acquisition function
  Evaluate $f_{\text{OBJ}}(\{k_{\vec{a}}^{0,1}, k_0^2\}_n)$
  $Q_{\max} \leftarrow$ if $f_{\text{OBJ}}(\{k_{\vec{a}}^{0,1}, k_0^2\}_n) > Q_{\max}$, update $Q_{\max}$
  Update the surrogate function with the new data
  n++

Having initialized all the parameters we are not optimizing over, we now maximize the objective function

$$f_{\text{OBJ}} = \min\left(Q_1, Q_5\right), \tag{3.6}$$

as a function of the variables $\{k_{\vec{a}}^{0,1}, k_0^2\}$, which (having set `prec_k = 2`) we also allow to take integer values in $[-100, 100]^{\otimes(2n_B+1)}$. The evaluation of f_{OBJ} works as follows. We first solve the homogeneous bubble equations (2.9). Next, we round the flux parameters $k_\alpha^2 \to \tilde{k}_\alpha^2$. Last, we use (2.8) to compute Q_1, Q_5 and select the minimum of the two.

We evaluate the objective function on $\mathbf{n_0}$ points in the $(2n_B + 1)$-dimensional space of $\{k_{\vec{a}}^{0,1}, k_0^2\}$ sampled from the discrete uniform distribution $U(-10^2, 10^2)^{\otimes(2n_B+1)}$. Assuming a Gaussian process prior as described above, we use knowledge of f_{OBJ} evaluated at this set of points to generate the surrogate function. We use $\mathbf{n_0} = 200$.

We then evaluate the surrogate function on a much higher number (of order 10^4) of randomly sampled points, with discrete uniform distribution $U(-10^2, 10^2)^{\otimes(2n_B+1)}$ and step-size 1. We use an acquisition function based on the Probability of Improvement method (see e.g., ref. [71]) to choose the next point of the domain that is most worth evaluating with the objective function. The knowledge of f_{OBJ} at this new point is then used to update the surrogate function. This process is iterated until a point $\{k_{\vec{a}}^{0,1}, k_0^2\}$ such that $f_{\text{OBJ}}(\{k_{\vec{a}}^{0,1}, k_0^2\}) > \bar{Q}$ is found, or a previously set computational limit (the maximum number of iterations N) is reached. This procedure is summarized in Algorithm 1.

While this algorithm is not guaranteed to find a solution, we found that in practice this approach is much more successful than a random search.

After a successful run of the Bayesian optimization algorithm, we impose the two conditions described at the end of Section 3.1, to be confident that a nearby scaling solution exists and that there are no CTCs in the configuration at this point. If and only if these two tests are passed, we proceed to the EA.

3.3. The Evolutionary Algorithm

A successful run of the Bayesian optimization algorithm outputs an approximate solution to the homogeneous bubble equations (2.9) with the desired characteristics. The approximate nature of this solution is due to the rounding of the flux parameters k_α^2. Our task is now to obtain a numerical solution of the inhomogeneous bubble equations (2.6) by moving the positions of the centers in the $\mathbb{R}^3$ base space.

We shall do so by using an EA, which is an optimization algorithm inspired by Darwin's theory of evolution. The starting point is a *population*, i.e., a set of *individuals* that are approximate solutions to the problem we wish to solve. The properties of each individual are called *genes*. We measure how good an approximate solution is via the *fitness* function, which is the function we want to maximize. The fittest individuals are selected to reproduce, by passing some of their genes to their offspring and in the reproduction process some mutations are implemented, i.e., random modifications of the genes of the offspring. Then the offspring take the place of the less fit individuals, which die, such that the population size is constant. This process is iterated until a sufficiently good solution is found, or a previously set computational limit is reached.

For the case at hand, each individual is an approximate solution to the bubble equations (2.6). An individual's genes are (a subset of) the positions of the centers together with a set of *strategy parameters*, to be described momentarily.

After implementing the translational and rotational symmetry, the number of free coordinates is $3n - 6$, while the number of independent bubble equations is $n - 1$. Implementing a genetic algorithm on all the $3n - 6$ coordinates would be computationally expensive. Thus, we select a subset of the coordinates to be fixed to the values of the BO algorithm output, $\bar{r}_a^i$ of Equation (3.1). We observed that fixing approximately one coordinate on each of the last $n - 3$ centers provides a good balance between effectiveness and computational cost, and we shall give explicit examples of this in Section 4. We shall denote by d the number of degrees of freedom, i.e., the number of unfixed coordinates over which we run the EA.

To describe the algorithm further, we introduce the multi-index $A = (a, i)$ combining the center label a and the three Euclidean coordinates i. For the p^{th} individual in the population, we denote the set of coordinates to be varied by $r_A^{(p)}$. As noted above, we work directly with the positions of the centers as genes, so the distances between the centers are always well-defined.

In the reproduction process, random mutations occur. The magnitude of the mutations is controlled by the set of strategy parameters. Each direction $r_A^{(p)}$ has an independent strategy parameter $\sigma_A^{(p)}$, which is a gene of the individual, and which also undergoes variation and selection itself. As we will describe in the following, we will initialize the positions of the individuals in the population by sampling from a Gaussian distribution with fixed standard deviation, which will be taken as the initial value of the strategy parameter for every individual and every direction. The genes of the p-th individual in the population are then written as

$$\{r_A^{(p)}, \sigma_A^{(p)}\}, \qquad A = (a, i). \tag{3.7}$$

All genes $\{r_A^{(p)}, \sigma_A^{(p)}\}$ undergo variation and selection. An individual whose genes are likely to survive in the evolutionary process will have a good set of positions $r_A^{(p)}$, that will be quantified by having high fitness, and a set of strategy parameters $\sigma_A^{(p)}$ that are likely to give rise to fit offspring, as will become clearer when we discuss reproduction and mutation.

The evolutionary mechanism is iterated over a certain number of generations controlled by the hyperparameter `generations`; in each new generation a number of offspring is generated, specified by the hyperparameter `offspring_per_generation`. The optimal values of these hyperparameters depend on the number of degrees of freedom of the problem, i.e., the number of centers of the configuration, as we shall see in examples.

3.3.1. Fitness Function

We now define the fitness function, i.e., the function the EA algorithm seeks to maximize. We seek solutions to the inhomogeneous bubble equations (2.6), so we use these to construct the fitness function. By rearranging the bubble equations, we write:

$$\sum_{a\neq b} \frac{q_a q_b}{r_{ab}} \Pi^0_{ab} \left(\Pi^1_{ab}\Pi^2_{ab} - \frac{1}{2g^2}\mathbb{T}_{ab} \right) - \sum_{b,i} q_a q_b l^i_0 \Pi^i_{ab} = \epsilon_a \,, \tag{3.8}$$

where if $\epsilon_a = 0 \ \forall a$, then the solution is exact. We seek configurations $\{r_A\}$ that minimize the errors associated to all the bubble equations. We do so by instructing the algorithm to minimize $\sum_a |\epsilon_a|$. That is, we define the algorithm's fitness function to be

$$f(r_A) = \frac{1}{\sum_a |\epsilon_a|} \,. \tag{3.9}$$

It will also be useful to define the following inverse fitness function:

$$f_{\text{inv}}(r_A) = \sum_a |\epsilon_a| \,. \tag{3.10}$$

Note that since the sum of the bubble equations is zero by construction, the inverse fitness function f_{inv} typically over-states the error of a configuration, particularly as the number of centers grows larger. A more accurate measure is the largest absolute value of the errors of the individual equations. So, for later use when discussing our results, we also define the following maximum-error fitness and inverse fitness functions,

$$\tilde{f}(r_A) = \frac{1}{\max_a |\epsilon_a|} \,, \tag{3.11}$$

and

$$\tilde{f}_{\text{inv}}(r_A) = \max_a |\epsilon_a| \,. \tag{3.12}$$

3.3.2. Population Initialization

We initialize a population of `pop_size` individuals by starting with the configuration of centers obtained in (3.1), and adding to it a random variable $\delta r_A^{(p)}$ sampled from the Gaussian distribution $\mathcal{N}(0, \texttt{var\_pos})$ with mean 0 and standard deviation `var_pos`, where `var_pos` is a hyperparameter of the algorithm:

$$r_A^{(p)} = \bar{r}_A + \delta r_A^{(p)} \qquad \text{for } p = 1, \dots, \texttt{pop\_size} \,. \tag{3.13}$$

The optimal value of `var_pos` depends on the number of centers. If `var_pos` is too small, we generate a population that is too similar to the seed solution; if `var_pos` is too large, we obtain a large number of unfit individuals in the initial population. It is not practical to compute optimal value of `var_pos` directly from the bubble equations Equation (2.6), so we optimize it via a simple grid search. Concretely, we examine the populations generated by a set of candidate values of `var_pos`, and use the population fitness to select the optimal `var_pos`.

Once an individual is generated, we implement the opposition-based technique,[72,73] i.e., we compare this individual with the one obtained through reflection symmetry with respect to the initial configuration (3.1), and keep the one which has the highest fitness. In other words, given the individual with position $r_A^{(p)}$, we consider the individual with position $r'^{(p)}_A$ given by:

$$r'^{(p)}_A = 2\bar{r}_A - r_A^{(p)} \,, \tag{3.14}$$

and add to the population (only) the fitter of the two individuals. We initialize the strategy parameter as $\sigma_A^{(p)} = \texttt{var\_pos}$ for all p, A.

3.3.3. Selection

The selection mechanism of the EA dictates which individuals pass their genes to the offspring, and which individuals are replaced in the new generation. Recall that the number of new offspring per generation is a hyperparameter of the algorithm. To produce an offspring, the algorithm selects two parents that reproduce and one individual that will be replaced. This selection process occurs probabilistically, favouring potential parents with highest fitness, and where those with highest inverse fitness are most likely to die off.

Our algorithm implements two different selection mechanisms, amongst which the user can choose. The two methods are (see e.g., ref. [74] for more details):

- Fitness proportional selection. The probability of an individual to be chosen as a parent is:

$$P^{(p)} = \frac{f(r_A^{(p)})}{\sum_p f(r_A^{(p)})} \,, \tag{3.15}$$

  where f is the fitness function (3.9). The probability of an individual to die off is governed by the same equation, with the fitness function replaced by the inverse fitness function, defined in Equation (3.10).
- Sigma scaling. The probability of an individual to be chosen as parent is given by a modified version of Equation (3.15), with the fitness function being replaced by the following auxiliary fitness function f', which involves a shift controlled by

the mean $\bar{f}$ and standard deviation σ_f of the fitnesses of the population:

$$f'(r_A^{(p)}) = \max\left(f\left(r_A^{(p)}\right) - (\bar{f} - c\sigma_f), 0\right), \tag{3.16}$$

where c is a constant which is usually set to 2. Similarly, we also apply this method to the selection of the individual that dies by replacing the fitness function with the inverse fitness function.

The first method is less computationally expensive, however it can be less effective due to the following disadvantage. It is sensitive to adding a constant shift to all f, and depending on this shift there can be too much or too little selection pressure. For instance, if there is too much selection pressure, the best individuals tend to dominate the population very quickly, which can lead to premature convergence.

By contrast, the second method is more computationally expensive, but tends to produce an appropriate amount of selection pressure.

3.3.4. Reproduction and Mutation

Once two parents $\{r_A^{(1)}, \sigma_A^{(1)}\}$ and $\{r_A^{(2)}, \sigma_A^{(2)}\}$ are selected, their offspring $\{r_A, \sigma_A\}$ is generated as follows. First, separately for each gene, i.e., for each element of the multi-index A, we assign equal probability to one of the following three reproduction methods to occur. Before possible mutation, this gene will be set equal to the gene of a parent, or the average of the two parental genes:

$$\begin{aligned} &1) \quad \{r_A, \sigma_A\} = \{r_A^{(1)}, \sigma_A^{(1)}\}; \\ &2) \quad \{r_A, \sigma_A\} = \{r_A^{(2)}, \sigma_A^{(2)}\}; \\ &3) \quad \{r_A, \sigma_A\} = \left\{\frac{r_A^{(1)} + r_A^{(2)}}{2}, \frac{\sigma_A^{(1)} + \sigma_A^{(2)}}{2}\right\}. \end{aligned} \tag{3.17}$$

Next, we implement a mutation mechanism, which can enable the population to escape from local minima of the fitness function. Recall that each offspring has a total of d genes ($A = 1, \ldots, d$). For each gene of an offspring, we assign a probability of order $2/d$ for the gene to mutate away from the value assigned in (3.17). So on average, approximately two genes of each offspring will mutate.

It is desirable to build in the flexibility for mutations in different genes to have different strengths. We therefore implement uncorrelated mutation, with different step sizes for different genes (see e.g., ref. [74]). If a gene is selected for mutation, first σ_A mutates, then the mutated σ'_A sets the scale for the mutation of the position r_A. The mutation of σ_A is controlled by two Gaussian random variables: $\delta\sigma$ is sampled only once for each offspring, while $\delta\sigma_A$ is sampled separately for each gene. Both are sampled from the Gaussian distribution $\mathcal{N}(0,1)$. The mutation strength of σ_A is controlled by the parameters $\tau' = 1/\sqrt{d}$ and $\tau = 1/\sqrt{2\sqrt{d}}$ via:

$$\sigma_A \to \sigma'_A = \sigma_A \exp\left(\tau'\,\delta\sigma + \tau\,\delta\sigma_A\right), \tag{3.18}$$

where following,[74] we set $\tau' = 1/\sqrt{d}$ and $\tau = 1/\sqrt{2\sqrt{d}}$. The way this treats different directions differently is as follows. The mutation $\exp(\tau'\,\delta\sigma)$ is common to all directions and allows for an overall change of the mutation step-size. By contrast, the term $\exp(\tau\,\delta\sigma_A)$ introduces different mutation strengths in different directions.

Once σ_A has mutated to σ'_A, the position r_A mutates with a Gaussian random variable δr_A drawn from $\mathcal{N}(0,1)$, with mutation strength set by the new σ'_A:

$$r_A \to r'_A = r_A + \sigma'_A\,\delta r_A\,. \tag{3.19}$$

As the fitness of the population improves, and the algorithm explores narrower regions of the phase space, we improve upon the mutation mechanism (3.18) to achieve the following set of goals. The mutation should enable an appropriately fine exploration of the narrower region, while not becoming too small in magnitude. Separately, the mutation should be able to jump outside local minima. To enable this, we introduce a hyperparameter `generation_update`. After `generation_update` generations, we update the strategy parameter of the individuals according to the following prescription. After this update, the algorithm returns to the mutation (3.18) for the next `generation_update` generations. The algorithm contains two different methods to update the strategy parameter, among which the user can choose. These are:

- Random update. We introduce the hyperparameters `percentage_random_update` and `factor_random_update`, randomly select (`percentage_random_update`)% of the individuals and rescale $\sigma_A \to \sigma_A/$`factor_random_update`, while rescaling the strategy parameter of the remaining individuals as $\sigma_A \to$ (`factor_random_update`) σ_A.
- Variance update. We update the strategy parameter according to the variance of the positions, as follows. For each A, we define the average position

$$\hat{r}_A = \frac{1}{\texttt{pop\_size}} \sum_q r_A^{(q)}, \tag{3.20}$$

and reset the strategy parameters in direction A of all individuals to have the same value,

$$\sigma_A^{(p)} \to \sigma_A'^{(p)} = \sqrt{\frac{1}{\texttt{pop\_size}} \sum_q \left(r_A^{(q)} - \hat{r}_A\right)^2} \qquad \forall p\,. \tag{3.21}$$

The variance update method works as follows. If, for instance, the whole population is concentrated in a particular region of parameter space, it tends to enable finer exploration of the local region. By contrast, if for instance the population is concentrated in two or more separate regions, the variance update tends to enable the population to explore a larger region of the parameter space.

We performed several runs with both Random and Variance updates. The Variance update has the advantage that it does not depend on the number of centers, while in the Random update we are introducing two new hyperparameters that could in principle

Table 1. Hyperparameters of the algorithm, optimized for five-center configurations.

pop_size	offspring_per_generation	generations	var_pos	generation_update
2000	50	16 500	10^{-4}	666

be optimized over, depending for instance on the number of centers. Of the two methods, typically the Variance update is more computationally expensive, and typically the performance of the algorithm is better than or comparable to the Random update, depending on the other parameters of the configuration.

For the sake of clarity, let us illustrate an example of how the variance update method works. Suppose we set the EA to run for 1000 generations, and that we specify `generation_update` = 100. The evolution of the strategy parameters σ_A will work as follows. Up to generation 99, the σ_A of each individual will evolve with random mutations via Equation (3.18). Then, at generation 100, the strategy parameters σ_A of all individuals in the population will be reset according to Equation (3.21). Next, for generations 101 to 199 we return to the random mutations described in Equation (3.18). Then, at generation 200 we again reset the σ_A according to Equation (3.21), and the pattern continues until the end.

4. Results

In this section we present two explicit examples of numerical scaling solutions obtained with the algorithm described above, which have five and seven centers respectively. In both of these examples we use the fitness proportional selection method and the variance update mechanism described in Section 3.3. In the examples that we present, the non-Abelian coupling constant g will be set equal to 1. We also comment on the performance of the algorithm.

4.1. A Five-Center Scaling Configuration

As a first application of our method, we provide an example of a five-center scaling configuration. We first optimize the hyperparameters of the EA for a five-center configuration. We do so with a grid search and obtain the values reported in **Table 1**.

We then follow the procedure described in Section 3.2. We use the Bayesian optimization algorithm to obtain initial positions and flux parameters, recorded in **Table 2**, that give a configuration with global charges $Q_1 \simeq 2938$ and $Q_5 \simeq 2016$.

By solving the homogeneous bubble equations (2.9), we obtain the $(n-1)$ remaining k_α^2 parameters. After rounding to a precision of `k_rounding` $= 10^{-4}$, we report their values in **Table 3**.

The parameters in Tables 2 and 3 define a numerical solution to the inhomogeneous bubble equations (2.6) with fitness $f(\bar{r}_a^i) \approx 4.26$. The configuration respects the condition regarding the possibility of a nearby scaling solution discussed around Equation (3.3). Moreover, the matrix $\mathcal{M}^2$ is positive-definite. We therefore proceed to use the configuration as a seed configuration in the EA. The genes of the EA are the subset of that are coordinates underlined in Table 2.

Table 2. Input parameters of the configuration. For ease of notation, the rescaling in Equation (3.1) with $\lambda = 10^{-5}$ is understood: in the first three rows of this table the coordinates of the centers are in units of 10^{-5}, i.e., $r_2^1 = 0.3314 \times 10^{-5}$. The underlined coordinates are those that are genes of the EA, i.e., the coordinates over which the evolution process occurs.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
x	0	0.3314	0.7491	−0.6923	0.4644
y	0	0	0.5648	−0.684	0.4799
z	0	0	0	−0.0792	0.549
q	1	1	−1	−1	1
k^0	17	−18	63	47	29
k^1	−61	66	25	72	80
k^2	60	–	–	–	–

Table 3. Solution of the homogeneous bubble equations, with the input parameters given in Table 2, after rounding.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
k^2	–	56.0815	−48.5265	−51.1402	47.9087

We plot in **Figure 1** how the fitness of the fittest individual and the average fitness of the population change over the generations in a representative run of the algorithm. By starting with a seed solution with fitness of order 1 we obtain, after around 14 000 generations, a numerical solution with fitness $\tilde{f} \simeq 2 \times 10^{10}$. This means that the EA generated a numerical solution that solves each bubble equation with a precision of at least 5×10^{-11}.

Note that these fitness/error figures are dimensionful, and depend on our choice of units. A useful dimensionless relative error can be defined by comparing the error in the bubble equations to

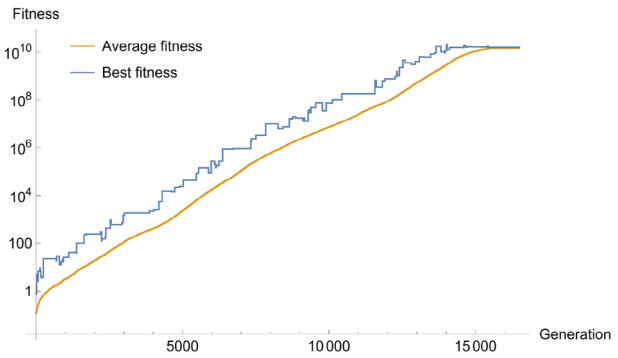

Figure 1. Fitness of the fittest five-center configuration, and average fitness of the population over the generations. The final plateau occurs at machine precision, as discussed in the text. The maximum fitness obtained is $\tilde{f} \simeq 2 \times 10^{10}$, corresponding to a dimensionless relative error of 1×10^{-15}.

Table 4. Output of the evolutionary algorithm with highest fitness, for the five-center configuration. Similarly to those in Table 2, these coordinates are in units of 10^{-5}.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}
x	0	0.3314045307889	0.7491026359717	−0.6923	0.4644170857933
y	0	0	0.5648088034751	−0.6840217263604	0.4799
z	0	0	0	−0.07910622500586	0.549

the largest summand on the left-hand side of the bubble equations. For this five-center configuration, the largest summand has absolute value approximately 5×10^4. The ratio of the largest error in the bubble equations, $\tilde{f}_{\text{inv}}(r_A) = \max_a |\epsilon_a|$, to this largest summand, is therefore approximately 1×10^{-15}. This is a typical best-case value at which the dimensionless relative error of the algorithm saturates, reflecting the fact that we work to machine precision.

As a side point, we observe that the fitness of the fittest individual in the population does not increase monotonically. The main reason is that occasionally the fittest individual is selected to die; this is part of the random nature of the algorithm, and is one way that the algorithm can escape local minima. A more minor reason is that the algorithm maximizes the sum-error fitness f defined in (3.9), while we have plotted the max-error fitness $\tilde{f}$ defined in (3.11), being a more accurate measure of the fitness.

The associated matrix $\mathcal{M}^2$ is positive-definite, thus we can have confidence that the geometry does not contain any CTCs. We report in **Table 4** the coordinates of the centers of this numerical solution.

We compute the global charges of this solution using Equation (2.8), obtaining:

$$Q_1 \simeq 2938\,, \qquad Q_S \simeq 2016\,, \qquad Q_P \simeq 2.998 \times 10^4\,, \\ J_L \simeq 4.090 \times 10^5\,, \qquad J_R \simeq 0.001545\,, \tag{4.1}$$

where $J_R \equiv |\vec{J}_R|$. We note that $J_R \ll 1$ as expected.

4.2. A Seven-Center Scaling Configuration

We now present an example of a seven-center scaling configuration. We report in **Table 5** the hyperparameters of the EA, optimized for seven-center configurations.

After running the Bayesian optimization algorithm, we obtain an initial configuration with global charges $Q_1 \simeq 736$ and $Q_S \simeq 410$, which is described in **Table 6**.

The homogeneous bubble equations (2.9) give the $n-1$ remaining flux parameters k_α^2; we round them to a precision of 10^{-5}, and report the result in **Table 7**.

The coefficients in Tables 6 and 7 provide an approximate solution to the inhomogeneous bubble equations (2.6) with fitness $f(\vec{r}_a^{\,i}) \approx 3.21$. As in the five-center example, only the underlined coordinates in Table 6 are taken to be genes of the EA.

Table 6. Input parameters of the solution. Similarly to those in Table 2, the coordinates in the first three rows are in units of 10^{-5}, and underlined coordinates are genes of the evolutionary algorithm.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}
x	0	0.8409	0.3858	−0.0195	0.2188	−0.6853	−0.6294
y	0	0	−0.4476	−0.8449	−0.2569	−0.82	0.8284
z	0	0	0	−0.2854	−0.9864	−0.3303	−0.9516
q	1	−1	1	−1	1	−1	1
k^0	6	−38	56	38	86	85	15
k^1	99	57	39	30	48	37	72
k^2	39	–	–	–	–	–	–

Table 7. Solution of the homogeneous bubble equations, with input parameters given in Table 6, after rounding.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}
k^2	–	−37.2187	38.5597	−38.568	38.4874	−38.6549	38.8499

As depicted in **Figure 2**, we obtain, after around 17 000 generations, a numerical solution with fitness $\tilde{f} \simeq 1.7 \times 10^{10}$. For this configuration, the absolute value of the largest term on the left-hand side of the bubble equations is 1.1×10^4. So the dimensionless relative error defined in the previous subsection is approximately 5.7×10^{-15}, again reflecting the fact that we work to machine precision.

The coordinates of the centers of the fittest configuration obtained by the EA are reported in **Table 8**.

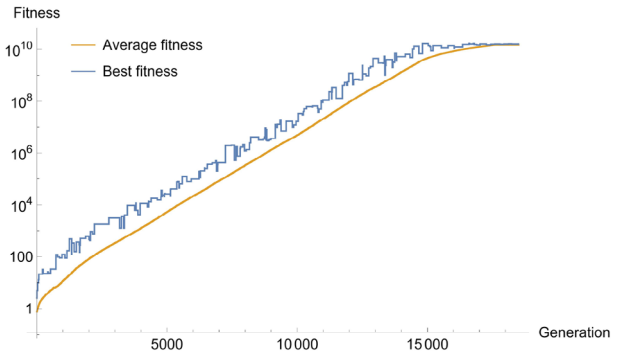

Figure 2. Fitness of the fittest seven-center configuration, and average fitness of the population over the generations. The final plateau again occurs at machine precision: the maximum fitness obtained is $\tilde{f} \simeq 1.7 \times 10^{10}$, corresponding to a dimensionless relative error of 5.7×10^{-15}.

Table 5. Hyperparameters of the algorithm, optimized for seven-center configurations.

pop_size	offspring_per_generation	generations	var_pos	generation_update
3000	100	18 500	10^{-4}	666

Table 8. Output of the evolutionary algorithm with highest fitness, for the seven-center configuration. Again, coordinates are in units of 10^{-5}.

	1^{st}	2^{nd}	3^{rd}	4^{th}	5^{th}	6^{th}	7^{th}
x	0	0.84088499	0.38578191	0.0195	0.21880766	−0.68529849	−0.62944182
y	0	0	−0.44759418	−0.84485200	−0.2569	−0.82004941	0.82839175
z	0	0	0	−0.28532553	−0.98638729	−0.33036911	−0.9516

The matrix $\mathcal{M}^2$ is positive-definite, and thus we can have confidence that the configuration is free of CTCs. Finally, we record the global charges of the solution:

$$Q_1 \approx 736.3\,, \qquad Q_5 \approx 410.0\,, \qquad Q_P \approx 8.887\times 10^4\,, \qquad J_L \approx 1.625\times 10^5\,, \qquad J_R \approx 0.007047\,. \tag{4.2}$$

We again observe that $J_R \ll 1$ as expected.

4.3. Performance of the Algorithm

We now make some general comments regarding the algorithm's performance. We have run the algorithm in detail for configurations of three, five, and seven centers. As the number of centers increases, naturally the runtime increases. This is primarily due to two factors. First, the number of bubble equations that need to be evaluated increases linearly with n. Second, a higher number of centers means a higher number of degrees of freedom of the EA, thus the population size and the number of offspring per generation should be increased accordingly, to optimize the algorithm's performance.

When run on a three-center configuration, the algorithm typically found of order 10 approximate solutions with fitness $\gtrsim 10^6$ in around 10 h (all run-times refer to a high-specification mainstream desktop machine). For a five-center configuration, typically around 13 runtime produced two solutions with fitness $\gtrsim 10^6$. For a seven-center configuration, in 40 runtime the algorithm produced two solutions with fitness $\gtrsim 10^5$, including the one reported above. Each of these runs was performed initially with a cutoff of 10 000 generations. Subsequently, longer runs were performed on the two examples presented earlier in this section to obtain the machine-precision solutions.

In light of the No Free Lunch Theorem, we have compared our algorithm with a random search on several examples, and observed it is always far superior, as follows. The random search is obtained by generating `offspring_per_generation` new individuals via Equation (3.13) for `generations` generations, and evaluating their fitness. We did so for different values of the standard deviation `var_pos`. For large values of `var_pos`, the relevant sampling space is too big, and the probability of finding good solutions is low. As we decrease `var_pos`, the performance of the random search increases until an optimized value. Decreasing `var_pos` further results in a loss of performance, as the individuals are too close to the seed solution $\bar{r}_A$, and thus their fitness is of the same order of $f(\bar{r}_A)$.

In all the examples we analysed, the random search provided an approximate solution with fitness no higher than around 10^{-2}. For completeness, in **Figure 3** we present an example of such a random search for the five-center configuration discussed in 4.1, with the values of `generations` and `offspring_per_generation` given in Table 1. This contrasts with the far superior performance of the EA.

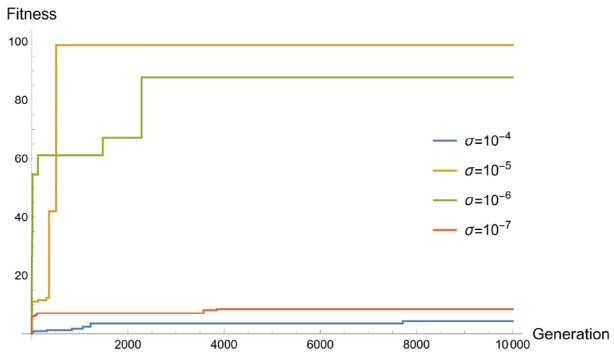

Figure 3. Random search over the initial configuration described in Tables 2 and 3. We repeat the analysis for four different values of `var_pos`, which are denoted with σ in the plot's legend.

5. Discussion

In this work we have presented an optimization algorithm, combining Bayesian optimization and an EA, using which we have constructed numerical multi-center solutions with several centers arranged in generic configurations, satisfying all flux quantization constraints.

The Bayesian optimization algorithm generates a configuration in the scaling regime with appropriately large D1 and D5 charges, by optimizing over a subset of the flux parameters. The output of this first step is a starting configuration that is an approximate solution to the bubble equations (2.6). The starting configuration contains two approximations: first, it was derived by solving the homogeneous form of the bubble equations; second, the flux parameters were rounded in order to respect the conditions of flux quantization.

The EA uses the configuration generated in the first step as a starting point to generate approximate solutions to the full inhomogeneous bubble equations, with increasing precision. It does so by optimizing over the positions of the centers. By working with the positions of the centers, the distances between centers are always well-defined. The fact that we work with quantized fluxes and well-defined distances is a distinguishing advantage of our method over previous approaches.

In Section 4 we exhibited two examples of novel machine-precision numerical solutions, one with five centers and one with seven centers. Although we have used the algorithm primarily to construct configurations with three, five, and seven centers, it can in principle be used to construct configurations with any number of centers in generic configurations, upon tuning the hyperparameters appropriately.

While the EA significantly improves upon previous methods, naturally it does not always find machine-precision solutions, depending on the starting configuration. This can be for one of two reasons. First, it is a general limitation of EAs that they do not always find optimal solutions, i.e., global minima of the fitness function: in the evolution process, the population might get stuck in a local minimum. It is well known that this problem is more

serious when the number of degrees of freedom of the problem is low, as the probability of having local minima decreases as the number of dimensions increase.

Second, given a set of poles in the harmonic functions describing the multi-center solution, it is not guaranteed that there exists a nearby exact solution to the bubble equations. For instance, by rounding the flux parameters, one could access a region of parameter space that does not admit solutions. It is thus possible that the EA fails to find a sufficiently good solution simply because such a solution does not exist. Indeed, we found three-center configurations in which the EA could not improve the fitness above a certain value, and a detailed analysis of one such example showed that a nearby exact solution did not exist. Despite these inherent limitations, most of the time the algorithm is successful at finding solutions.

As the number of centers grows, naturally more computational resources are required. It is worth noting that, in particular, the task of finding a good starting configuration requires significantly more computational resources. This is because it can take several iterations for the Bayesian optimization algorithm to find scaling configurations with appropriately large supergravity charges Q_1, Q_5, together with a positive-definite matrix $\mathcal{M}$ for absence of CTCs. So as the number of centers increases, a smaller fraction of potential starting configurations get passed on to the EA. In particular, apart from choosing the flux parameters that we initialize to have the same sign, as suggested in ref. [21, Footnote 17], finding a good $\mathcal{M}$ is otherwise done by a random search. The probability of randomly selecting initial parameters that lead to an $\mathcal{M}$ with all positive eigenvalues decreases as the number of centers (and thus the dimension of $\mathcal{M}$) increases. Therefore, it would be interesting to explore more efficient ways of selecting parameters that lead to a positive-definite $\mathcal{M}$; we leave this task to future work.

To conclude, we have implemented a novel application of EAs and Bayesian optimization to the study of multi-centered solutions to supergravity, and have presented two state-of-the-art machine-precision numerical solutions. Our algorithm could be used to create larger families of numerical multi-center solutions, which could in turn be used in phenomenological models.[58–60] The prospects for harnessing the power of computer science algorithms to solve physically interesting problems in String Theory and related fields appear bright, with an exciting future ahead.

Acknowledgements

We thank Iosif Bena, Johan Blåbäck, Óscar Dias, Bogdan Ganchev, Pierre Heidmann, Anthony Houppe, Daniel Mayerson, Rodolfo Russo, Kostas Skenderis, Marika Taylor, Nicholas Warner and Ben Withers for fruitful discussions. The work of S.R. was supported by a Royal Society URF Enhancement Award. The work of D.T. was supported by a Royal Society Tata University Research Fellowship. We thank the IPhT, CEA Saclay, and the University of Genoa for hospitality during the course of this work.

Keywords

black holes, string theory, supergravity solutions

Received: November 21, 2023

[1] S. W. Hawking, *Phys. Rev.* **1976**, *D14*, 2460.
[2] S. D. Mathur, *Class. Quant. Grav.* **2009**, *26*, 224001, arXiv:0909.1038 [hep-th].
[3] A. Strominger, C. Vafa, *Phys. Lett.* **1996**, *B379*, 99, arXiv:hep-th/9601029.
[4] O. Lunin, S. D. Mathur, *Nucl. Phys.* **2002**, *B623*, 342, arXiv:hep-th/0109154.
[5] S. D. Mathur, *Fortsch. Phys.* **2005**, *53*, 793, arXiv:hep-th/0502050.
[6] K. Skenderis, M. Taylor, *Phys. Rept.* **2008**, *467*, 117, arXiv:0804.0552 [hep-th].
[7] V. Balasubramanian, J. de Boer, S. El-Showk, I. Messamah, *Class. Quant. Grav.* **2008**, *25*, 214004, arXiv:0811.0263 [hep-th].
[8] S. A. Abel, A. Constantin, T. R. Harvey, A. Lukas, *Fortsch. Phys.* **2023**, *71*, 2200161, arXiv:2208.13804 [hep-th].
[9] P. Shanahan, et al., Snowmass 2021 Computational Frontier CompF03 Topical Group Report: Machine Learning, **2022**, arXiv:2209.07559 [physics.comp-ph].
[10] I. Bena, J. Blåbäck, M. Graña, S. Lüst, *Adv. Appl. Clifford Algebras* **2022**, *32*, 7, arXiv:2103.03250 [hep-th].
[11] F. Ruehle, *Phys. Rept.* **2020**, *839*, 1.
[12] S. Abel, J. Rizos, *JHEP* **2014**, *08*, 010, arXiv:1404.7359 [hep-th].
[13] D. Partipilo, *JHEP* **2022**, *09*, 096, arXiv:2205.06245 [hep-th].
[14] H. Erbin, A. H. Fırat, Characterizing 4-string contact interaction using machine learning, **2022**, arXiv:2211.09129 [hep-th].
[15] P. Berglund, G. Butbaia, T. Hübsch, V. Jejjala, D. Mayorga Peña, C. Mishra, J. Tan, Machine Learned Calabi–Yau Metrics and Curvature, **2023**, arXiv:2211.09801 [hep-th].
[16] I. Bena, N. P. Warner, *Adv. Theor. Math. Phys.* **2005**, *9*, 667, arXiv:hep-th/0408106.
[17] J. P. Gauntlett, J. B. Gutowski, *Phys.Rev.* **2005**, *D71*, 045002, arXiv:hep-th/0408122 [hep-th].
[18] I. Bena, C.-W. Wang, N. P. Warner, *JHEP* **2006**, *11*, 042, arXiv:hep-th/0608217.
[19] I. Bena, N. P. Warner, *Lect. Notes Phys.* **2008**, *755*, 1, arXiv:hep-th/0701216.
[20] I. Bena, C.-W. Wang, N. P. Warner, *JHEP* **2008**, *07*, 019, arXiv:0706.3786 [hep-th].
[21] J. Avila, P. F. Ramirez, A. Ruiperez, *JHEP* **2018**, *01*, 041, arXiv:1709.03985 [hep-th].
[22] P. Heidmann, *JHEP* **2017**, *10*, 009, arXiv:1703.10095 [hep-th].
[23] I. Bena, P. Heidmann, P. F. Ramirez, *JHEP* **2017**, *10*, 217, arXiv:1709.02812 [hep-th].
[24] M. Bianchi, J. F. Morales, L. Pieri, N. Zinnato, *JHEP* **2017**, *05*, 147, arXiv:1701.05520 [hep-th].
[25] D. R. Mayerson, Modave Lectures on Horizon-Size Microstructure, Fuzzballs and Observations, **2022**, arXiv:2202.11394 [hep-th].
[26] I. Bena, S. Giusto, R. Russo, M. Shigemori, N. P. Warner, *JHEP* **2015**, *05*, 110, arXiv:1503.01463 [hep-th].
[27] I. Bena, E. Martinec, D. Turton, N. P. Warner, *JHEP* **2016**, *05*, 064, arXiv:1601.05805 [hep-th].
[28] I. Bena, S. Giusto, E. J. Martinec, R. Russo, M. Shigemori, D. Turton, N. P. Warner, *Phys. Rev. Lett.* **2016**, *117*, 201601, arXiv:1607.03908 [hep-th].
[29] I. Bena, E. Martinec, D. Turton, N. P. Warner, *JHEP* **2017**, *06*, 137, arXiv:1703.10171 [hep-th].
[30] A. Tyukov, R. Walker, N. P. Warner, *JHEP* **2018**, *02*, 122, arXiv:1710.09006 [hep-th].
[31] I. Bena, S. Giusto, E. J. Martinec, R. Russo, M. Shigemori, D. Turton, N. P. Warner, *JHEP* **2018**, *02*, 014, arXiv:1711.10474 [hep-th].
[32] N. Ceplak, R. Russo, M. Shigemori, *JHEP* **2019**, *03*, 095, arXiv:1812.08761 [hep-th].
[33] P. Heidmann, N. P. Warner, *JHEP* **2019**, *09*, 059, arXiv:1903.07631 [hep-th].

[34] P. Heidmann, D. R. Mayerson, R. Walker, N. P. Warner, *JHEP* **2020**, *02*, 192, arXiv:1910.10714 [hep-th].
[35] B. Ganchev, A. Houppe, N. P. Warner, *JHEP* **2022**, *09*, 067, arXiv:2207.04060 [hep-th].
[36] N. Čeplak, *JHEP* **2023**, *08*, 047, arXiv:2212.06947 [hep-th].
[37] I. Kanitscheider, K. Skenderis, M. Taylor, *JHEP* **2007**, *04*, 023, arXiv:hep-th/0611171.
[38] I. Kanitscheider, K. Skenderis, M. Taylor, *JHEP* **2007**, *06*, 056, arXiv:0704.0690 [hep-th].
[39] S. Giusto, E. Moscato, R. Russo, *JHEP* **2015**, *11*, 004, arXiv:1507.00945 [hep-th].
[40] S. Giusto, S. Rawash, D. Turton, *JHEP* **2019**, *07*, 171, arXiv:1904.12880 [hep-th].
[41] S. Rawash, D. Turton, *JHEP* **2021**, *07*, 178, arXiv:2105.13046 [hep-th].
[42] B. Ganchev, S. Giusto, A. Houppe, R. Russo, *Eur. Phys. J. C* **2022**, *82*, 217, arXiv:2112.03287 [hep-th].
[43] S. Giusto, S. D. Mathur, A. Saxena, *Nucl. Phys.* **2004**, *B701*, 357, arXiv:hep-th/0405017.
[44] S. Giusto, S. D. Mathur, A. Saxena, *Nucl. Phys.* **2005**, *B710*, 425, arXiv:hep-th/0406103.
[45] O. Lunin, *JHEP* **2004**, *04*, 054, arXiv:hep-th/0404006.
[46] V. Jejjala, O. Madden, S. F. Ross, G. Titchener, *Phys. Rev.* **2005**, *D71*, 124030, arXiv:hep-th/0504181.
[47] S. Giusto, O. Lunin, S. D. Mathur, D. Turton, *JHEP* **2013**, *1302*, 050, arXiv:1211.0306 [hep-th].
[48] B. Chakrabarty, D. Turton, A. Virmani, *JHEP* **2015**, *11*, 063, arXiv:1508.01231 [hep-th].
[49] E. J. Martinec, S. Massai, *JHEP* **2018**, *07*, 163, arXiv:1705.10844 [hep-th].
[50] E. J. Martinec, S. Massai, D. Turton, *JHEP* **2018**, *09*, 031, arXiv:1803.08505 [hep-th].
[51] E. J. Martinec, S. Massai, D. Turton, *JHEP* **2019**, *11*, 019, arXiv:1906.11473 [hep-th].
[52] E. J. Martinec, S. Massai, D. Turton, *JHEP* **2020**, *12*, 135, arXiv:2005.12344 [hep-th].
[53] D. Bufalini, S. Iguri, N. Kovensky, D. Turton, *JHEP* **2021**, *08*, 011, arXiv:2105.02255 [hep-th].
[54] D. Bufalini, S. Iguri, N. Kovensky, D. Turton, *Phys. Rev. Lett.* **2022**, *129*, 121603, arXiv:2203.13828 [hep-th].
[55] D. Bufalini, S. Iguri, N. Kovensky, D. Turton, *JHEP* **2023**, *03*, 066, arXiv:2210.15313 [hep-th].
[56] E. J. Martinec, S. Massai, D. Turton, *Fortsch. Phys.* **2023**, *71*, 2300015, arXiv:2211.12476 [hep-th].
[57] G. Bossard, S. Lüst, *Gen. Rel. Grav.* **2019**, *51*, 112, arXiv:1905.12012 [hep-th].
[58] I. Bena, D. R. Mayerson, *Phys. Rev. Lett.* **2020**, *125*, 221602, arXiv:2006.10750 [hep-th].
[59] M. Bianchi, D. Consoli, A. Grillo, J. F. Morales, P. Pani, G. Raposo, *Phys. Rev. Lett.* **2020**, *125*, 221601, arXiv:2007.01743 [hep-th].
[60] I. Bah, I. Bena, P. Heidmann, Y. Li, D. R. Mayerson, *JHEP* **2021**, *10*, 138, arXiv:2104.10686 [hep-th].
[61] P. I. Frazier, A Tutorial on Bayesian Optimization, **2018**, arXiv:1807.02811 [stat.ML].
[62] P. F. Ramirez, *JHEP* **2016**, *11*, 152, arXiv:1608.01330 [hep-th].
[63] B. Chakrabarty, S. Rawash, D. Turton, *JHEP* **2022**, *02*, 202, arXiv:2112.08378 [hep-th].
[64] I. Bena, G. Bossard, S. Katmadas, D. Turton, *JHEP* **2017**, *01*, 127, arXiv:1611.03500 [hep-th].
[65] G. Bossard, S. Katmadas, D. Turton, *JHEP* **2018**, *02*, 008, arXiv:1711.04784 [hep-th].
[66] I. Bah, P. Heidmann, *JHEP* **2021**, *09*, 128, arXiv:2106.05118 [hep-th].
[67] I. Bah, P. Heidmann, P. Weck, *JHEP* **2022**, *08*, 269, arXiv:2203.12625 [hep-th].
[68] F. Denef, *JHEP* **2000**, *0008*, 050, arXiv:hep-th/0005049 [hep-th].
[69] I. Bena, N. P. Warner, *Phys. Rev.* **2006**, *D74*, 066001, arXiv:hep-th/0505166.
[70] I. Bena, P. Heidmann, D. Turton, *JHEP* **2018**, *12*, 028, arXiv:1806.02834 [hep-th].
[71] A. Agnihotri, N. Batra, *Distill* **2020**, *5*.
[72] H. R. Tizhoosh, in *International Conference on Computational Intelligence for Modelling, Control and Automation and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)*, Vienna, Austria, Vol. *1*, IEEE, **2005**, pp. 695–701.
[73] S. Rahnamayan, H. R. Tizhoosh, M. M. Salama, Opposition-based differential evolution algorithms, in *2006 IEEE International Conference on Evolutionary Computation*, IEEE **2006**, pp. 2010–2017.
[74] A. E. Eiben, J. E. Smith, *Introduction to Evolutionary Computing*, Vol. *53*, Springer, New York **2003**.