FindBounce: Package for multi-field bounce actions☆,☆☆

Victor Guada [a,*], Miha Nemevšek [a], Matevž Pintar [b]

[a] *Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia*
[b] *C3M d.o.o., Tehnološki park 21, 1000 Ljubljana, Slovenia*

A B S T R A C T

We are launching FindBounce, a Mathematica package for the evaluation of the Euclidean bounce action that enters the decay rate of metastable states in quantum and thermal field theories. It is based on the idea of polygonal bounces, which is a semi-analytical approach to solving the bounce equation by discretizing the potential into piecewise linear segments. This allows for a fast and robust evaluation of arbitrary potentials with specified precision and any number of scalar fields. Time cost grows linearly with the number of fields and/or the number of segments. Computation with 20 fields takes $\sim$ 2 s with 0.5% accuracy of the action. The FindBounce function is simple to use with the native Mathematica look and feel, it is easy to install, and comes with detailed documentation and physical examples, such as the calculation of the nucleation temperature. We also provide timing benchmarks with comparisons to existing tools, where applicable.

Program summary
Program title: FindBounce
CPC Library link to program files: http://dx.doi.org/10.17632/tysw84skx3.1
Developer's repository link: https://github.com/vguada/FindBounce
Licensing provisions: GNU General Public License 3
Programming language: Mathematica
Nature of problem: Evaluation of the Euclidean bounce action that controls the decay rate of metastable local minima in thermal and quantum field theories.
Solution method: Semi-analytical solution of a system of coupled differential equations, based on the polygonal bounce idea (Guada et al., 2019).
Restrictions: Mathematica version 10 or above, works in $D = 3, 4$.

1. Introduction

False vacuum decay and phase transitions are pervasive topics in particle physics and cosmology. Most physical models of nature contain a portion of parameter space with a local ground state that can transform into a different one with a lower energy, either by quantum or thermal fluctuations. The theory of computing the lifetime of such metastable states began developing some 30 years ago [1–5] and still poses computational and conceptual challenges.

The task at hand is to find the so-called bounce solution, which is the dominant semi-classical contribution, that estimates the exponential factor B in the false vacuum decay rate

$$\Gamma \simeq A\, e^{-B} \left(1 + \mathcal{O}(\hbar)\right). \tag{1}$$

The pioneering work of Coleman [2] proposed a general method that applies to any potential. It requires solving a partial differential equation (PDE) in Euclidean space with fixed boundary conditions. An analytical approximation was given in the thin wall regime and an under(over)-shooting argument was made that is typically applied in numerical studies.

☆ The review of this paper was arranged by Prof. David W. Walker.
☆☆ This paper and its associated computer program are available via the Computer Physics Communication homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).
* Corresponding author.
E-mail addresses: victor.guada@ijs.si (V. Guada), miha.nemevsek@ijs.si (M. Nemevšek), matevz.pintar@c3m.si (M. Pintar).

One of the difficulties in automatic computation for arbitrary potentials resides in the numerical nature of the problem, particularly in the thin wall regime, where the two vacua are nearly degenerate. There, the initial field value has to be chosen with sufficient precision and needs to be sustained throughout the calculation in order to track the stiff PDEs with any chosen integrator. Finally, the field evolution has to stop at the appropriate moment before it slips back into the true vacuum or diverges. Controlling this behavior requires special care and may lead to numerical inaccuracies.

Closed form analytical solutions are clearly not available for arbitrary potentials. The most common approach is the thin-wall approximation [2] that is applicable to a general class of potentials with nearly degenerate minima. Other cases of soluble problems include the Fubini–Lipatov instanton [6,7] and its generalization [8], pure quartic [9] and logarithmic [10,11] potentials. For most other cases, numerical approaches are used. For example, single field renormalizable potentials can be reduced to a single parameter problem by rescaling and then solved numerically using the shooting procedure [12,13]. In many cases, having bounds on the action may be useful for restricting some particular solutions [14–17]. Finally, new approaches were proposed recently, based on the tunneling potential [18], machine learning techniques [19] and real time formalism [20–23].

The issues related to finding the bounce become more challenging when theories with multiple scalar fields are considered. Here, the path in field space is not fixed a priori and in general traverses a non-trivial potential landscape. In such case, finding the initial field value via the shooting procedure becomes a highly non-linear problem in the multi-dimensional field space.

There are a number of approaches that tackle the problem of multi-field bounce calculation. These include an improved action method that converts the saddle point into a minimum [24–26], numerical functional minimization [27], path deformation and shooting [28,29], frictionless dimensional continuation [30,31], semi-analytical techniques [32], multiple shooting [33], tunneling potential [34] and numerically solving coupled PDEs with variable coefficients [35], as well as machine learning techniques [36]. Existing publicly available tools are mostly based on numerical methods and rely on variations of path deformation [29] or multiple shooting [33,35].

Apart from the approaches described above, perhaps the simplest class of potentials with constant and linear field dependence was studied in [37]. The box-like and triangular shaped potentials with two segments were solved in closed form for $D = 4$. It turns out this approximation coincides with the thin wall and is therefore quite accurate for nearly degenerate potentials. Its validity was assessed in [38] for a single field and in [39] for multiple fields. Moreover, the combination of linear and quadratic was solved in [40], while the analytical continuation to Minkowski space was considered in [41].

Recently, the approach of [37] was extended [42] from the triangular bi-linear setting to an arbitrary polygonal shape, which allows for an arbitrarily precise evaluation. Moreover, both $D = 3, 4$ space–time dimensions were worked out and, most non-trivially, an arbitrary number of fields can be considered. We refer to this approach as the polygonal bounce method and is the methodological fulcrum of the FindBounce package. It is a semi-analytical procedure with a simple analytical solution on each segment of the piece-wise linear potential. Any number of segments can be glued together into a single smooth field value function from which an arbitrarily precise bounce action follows.

The purpose of this work is to implement the polygonal bounce method of [42] in Mathematica and release a publicly available documented software package called FindBounce. It serves to compute the bounce action B for theories with arbitrary number of fields in flat space–time dimensions $D = 3, 4$. This is done by the main function FindBounce, which computes the bounce action and returns the associated field solution. The output of the FindBounce is given by a dedicated data structure called BounceFunction. It stores the parameters of the field solution that can be used for further manipulation.

There are a number of phenomenological processes that require knowledge of the bounce action and the associated field configuration. Apart from the requirement that a given beyond the Standard Model (BSM) model should be long-lived enough, the bounce action at finite temperatures is important as well. It is needed to extract the parameters related to the production of gravitational waves from a first-order phase transition, see e.g. [43–48] and a recent review on LISA capabilities [49], EW baryogenesis [50–52] with the review in [53], as well as the production of primordial inter-galactic magnetic fields [54–58].

In Section 2, we provide a telegraphic overview of the basics regarding false vacuum decay, together with a short review of polygonal bounce features, needed for understanding the FindBounce implementation. The reader who is interested in a quick tryout of the FindBounce package can skip to Section 3, where the minimal short guide to installation is presented with the most basic examples. More detailed inner workings of the package, with a description of the FindBounce function and available options are available in Section 4. Examples with benchmarks, performance, timing and comparisons can be found in Section 5. We leave the concluding remarks and an outlook for future developments to Section 6.

2. Basics of false vacuum decay

The object of interest is the Euclidean action S_D of a real scalar field φ, defined in D dimensions as

$$S_D = \frac{2\pi^{D/2}}{\Gamma(D/2)} \int_0^\infty \rho^{D-1} \, d\rho \left(\frac{1}{2}\dot{\varphi}^2 + V(\varphi) \right), \tag{2}$$

and $V(\varphi)$ is the potential. We already assumed that the solution is $O(D)$ symmetric, as proven for single [59] and multi-field theories [60]. Therefore, S_D and the field solution depend only on the Euclidean radius $\rho^2 = t^2 + \sum x_i^2$, which determines the shape of the solution and the size of the appearing bubble. The bounce solution interpolates between the two (or more) minima of the potential $V(\varphi)$, as in Fig. 1, and obeys the following Euler–Lagrange equation with boundary conditions

$$\ddot{\varphi} + \frac{D-1}{\rho}\dot{\varphi} = \frac{dV}{d\varphi}, \qquad \varphi(0) = \varphi_0, \qquad \varphi(\infty) = \tilde{\varphi}_N,$$
$$\dot{\varphi}(0, \infty) = 0. \tag{3}$$

In the particle analogy [2], the field rolls down the inverted potential from φ_0 with zero velocity and comes to a stop in the false vacuum $\tilde{\varphi}_N$ at $\rho = \infty$. The usual shooting procedure involves numerically integrating the bounce equation (3) and varying φ_0 until the boundary conditions are met.
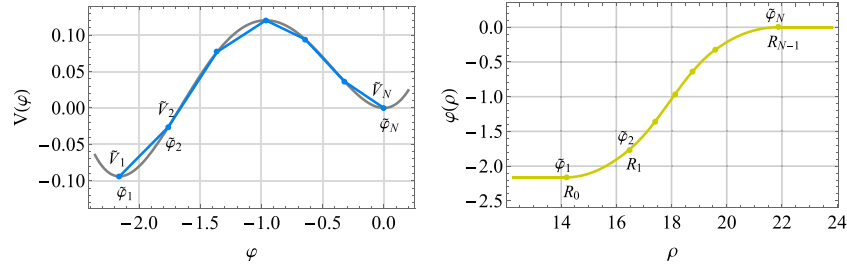
Fig. 1. Left: Linearly off-set quartic potential in gray and the polygonal approximation with $N = 7$ in blue. Right: The bounce field configuration corresponding to the potential on the left, computed with the polygonal bounce approximation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Numerical approaches based on shooting require a very precise initial value at $\rho = 0$, which has to be set carefully due to the $1/\rho$ friction term. Thereafter, high precision has to be kept throughout the integration, especially in the vicinity of the thin wall regime, where the bubble radius becomes large. In addition, the integration has to stop at a certain finite $\rho < \infty$, before it diverges or oscillates back to the true vacuum.

The polygonal bounce approach, described below, sidesteps these issues. Because the potential is composed of linear segments, the Euclidean time support region of the field is automatically compactified from $[0, \infty]$ to a finite region $[R_0, R_{N-1}]$, as seen on the right panel of Fig. 1. Outside of this range, the field value is constant. This compactification regularizes the solution and makes it numerically stable, especially in thin wall cases.

2.1. Polygonal bounce method

The polygonal bounce idea was developed in [42], where further details regarding the construction and its properties can be found. Here we review the salient features, necessary for understanding the `FindBounce` implementation. The basic idea is to extend the triangular case in [37] to an arbitrary number of segments and space–time dimensions. Moreover, the work of [42] shows how to go beyond the linear approximation and compute the bounce with any number of fields.

Polygonal construction. To establish the notation, let us start with the single field case and define the potential V_s on a linear segment s, with the corresponding field solution $\varphi_s(\rho)$

$$V_s(\varphi) \simeq \underbrace{\left(\frac{\tilde{V}_{s+1} - \tilde{V}_s}{\tilde{\varphi}_{s+1} - \tilde{\varphi}_s} \right)}_{8a_s} (\varphi_s - \tilde{\varphi}_s) + \tilde{V}_s - \tilde{V}_N, \tag{4}$$

$$\varphi_s(\rho) = v_s + \frac{4}{D} a_s \rho^2 + \frac{2}{D-2} \frac{b_s}{\rho^{D-2}},$$

with $s = 1, \ldots, N$, and $D > 2$. The list of field values $\{\tilde{\varphi}\}$ is called the field segmentation, see the left panel of Fig. 1. By default, the segmentation in `FindBounce` is taken to be homogeneous and consisting of $N = 31$ points, or equivalently $N - 1$ segments. The number of field points can be controlled by the option ``FieldPoints``, as explained in Section 4.1. Once the segmentation is fixed, the potential is evaluated on these points. This determines $\tilde{V}_s = V(\tilde{\varphi}_s)$ and thus the a_s coefficients in (4) are known.

The segmented solutions $\varphi_s(\rho)$ then need to be glued together into a continuously differentiable function. Thereby, the free parameters v_s, b_s, R_s of each segment are fixed by the following matching conditions: $\varphi_s(R_s) = \tilde{\varphi}_{s+1} = \varphi_{s+1}(R_s)$ and $\dot{\varphi}_s(R_s) = \dot{\varphi}_{s+1}(R_s)$. This results in the following recursive relations

$$v_s = v_1 - \frac{4}{D-2} \sum_{\sigma=1}^{s-1} (a_{\sigma+1} - a_\sigma) R_\sigma^2,$$

$$b_s = b_1 + \frac{4}{D} \sum_{\sigma=1}^{s-1} (a_{\sigma+1} - a_\sigma) R_\sigma^D. \tag{5}$$

Fixing the solution onto the pre-determined segmentation at R_s, i.e. $\varphi_s(R_s) = \tilde{\varphi}_{s+1}$, leads to

$$v_s + \frac{4}{D} a_s R_s^2 + \frac{2}{D-2} \frac{b_s}{R_s^{D-2}} = \tilde{\varphi}_{s+1}, \tag{6}$$

which is a fewnomial equation for R_s that admits the following simple solution

$$D = 3 : R_s = \frac{1}{2\sqrt{a_s}} \left(\frac{\delta_s}{\xi} + \xi \right),$$

$$D = 4 : R_s^2 = \frac{1}{2a_s} \left(\delta_s + \sqrt{\delta_s^2 - 4a_s b_s} \right), \tag{7}$$

with $\delta_s = \tilde{\varphi}_{s+1} - v_s$ and $\xi^3 = \sqrt{36 a_s b_s^2 - \delta_s^3} - 6\sqrt{a_s} b_s$. Therefore, the entire solution is constructed recursively from the starting values for v_s and b_s, given by the initial Euclidean radius R_{in}.
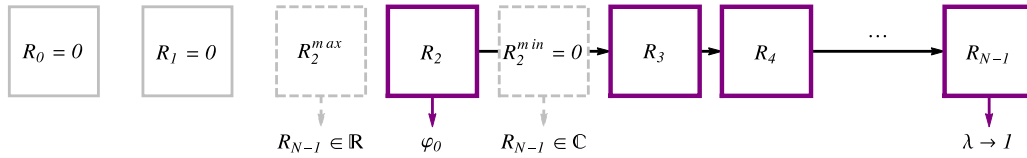
Fig. 2. A schematic overview of finding the initial radius R_{in}. In this particular example, the solution starts from segment $s = 2$, and is bounded by $R_2^{min} = 0$ and R_2^{max}, which can be computed from the segmentation, see [42] for details. Note that starting from these two boundary radii, the final radius R_{N-1} changes from real to complex, which happens only when starting from a segment, where a solution exists, in this example it is $s = 2$.

The polygonal approach exhibits features that simplify the search for the bounce solution. First, one replaces the field value φ_0, which is used in the numerical shooting algorithms, by the initial radius R_{in}. Thus the solution is constructed in terms of the Euclidean space–time variable and can be rescaled, e.g. $\varphi_s(\rho; R_{in}) \to \varphi_s(\rho; \lambda R_{in})$. This can be seen by considering the initial condition in (3), which gives either

$$\text{case A}: v_1 = \varphi_0 = \tilde{\varphi}_{in+1} - \frac{4}{D}a_{in}R_{in}^2, \quad b_1 = 0;$$
$$\text{or case B}: v_1 = \tilde{\varphi}_1 - \frac{4}{D-2}a_1R_0^2, \quad b_1 = \frac{4}{D}a_1R_0^D. \tag{8}$$

The entire solution is determined by the initial radius, either R_{in} in case A or R_0 in case B.

These two different cases are useful and come as a consequence of segmentation and the resulting lack of precision to perform the usual shooting. If the segmentation is sufficiently detailed, case A prevails and the field starts rolling from φ_0, otherwise it waits (as in the thin wall case) at the true minimum $\tilde{\varphi}_1$ until R_0 and then begins the evolution in ρ. The latter case is particularly suited for thin wall potentials.

Another useful feature of the method is that one can find the segment in field space, where the starting field value φ_0 has to be, prior to finding the exact solution. This can be done by scrolling through all the segments, starting from $\tilde{\varphi}_1$, bracketing the maximal and minimal values of the initial radius R_{in} (R_0 on the minimum, R_1 on the first segment and so on), running the polygonal setup and checking the complexity of the final radius. This procedure is sketched in Fig. 2, see also [42] for more details.

It turns out [42] that the argument of the square root in (7) equals zero when the final boundary conditions are met. Finding this transition point amounts to solving

$$\sum_{\sigma=in}^{N-1} (a_{\sigma+1} - a_\sigma) R_\sigma^D = 0 \tag{9}$$

in terms of R_{in}. This can done by rescaling R_{in} with Derrick's theorem or with a favorite root finding algorithm, e.g. by using the native FindRoot method in Mathematica. Notice that once the initial radius is found, the initial field value can be expressed directly in terms of it

$$\varphi_0 = \tilde{\varphi}_{in+1} - \frac{4}{D}a_{in}R_{in}^2. \tag{10}$$

A convenient measure for checking the accuracy of the solution comes from Derrick's theorem [61] that relates the integrated kinetic and potential pieces of the Euclidean action

$$\mathcal{T} = \frac{2\pi^{D/2}}{\Gamma(D/2)} \int_0^\infty \rho^{D-1} \, d\rho \frac{1}{2}\dot{\varphi}^2,$$
$$\mathcal{V} = \frac{2\pi^{D/2}}{\Gamma(D/2)} \int_0^\infty \rho^{D-1} \, d\rho V(\varphi). \tag{11}$$

Upon rescaling the solution to $\varphi(\rho/\lambda)$ the action should remain at the minimum, if the correct bounce solution was found, thereby relating $\mathcal{T}$ and $\mathcal{V}$

$$S_D^{(\lambda)} = \lambda^{D-2}\mathcal{T} + \lambda^D\mathcal{V}, \qquad \frac{dS_D^{(\lambda)}}{d\lambda}\bigg|_{\lambda=1} = 0,$$
$$\lambda = \sqrt{\frac{(2-D)\mathcal{T}}{D\mathcal{V}}} \to 1. \tag{12}$$

The second derivative $d^2S_D/d\lambda$ is always negative, hence the action is unstable with respect to rescaling, which may cause numerical issues. One way to solve the polygonal bounce is to simply rescale R_{in} by λ and iterate until $\lambda = 1$ is reached. In practice, finding the root of $\lambda - 1$ by varying R_{in} also turns out to be convenient and is implemented in the FindBounce package.

Going beyond the linear approximation. Once the basic polygonal bounce is obtained, it can be used for various purposes. For example, the $V(\varphi)$ can be expanded to higher orders and the bounce solution can be perturbatively improved. Specifically, expanding to second order gives

$$V_s(\varphi) \simeq \tilde{V}_s - \tilde{V}_N + \partial\tilde{V}_s (\varphi_s - \tilde{\varphi}_s) + \frac{\partial^2\tilde{V}_s}{2} (\varphi_s - \tilde{\varphi}_s)^2. \tag{13}$$

The $\partial \tilde{V}_s$ and $\partial^2 \tilde{V}_s$ parameters are determined by requiring the potential to be continuous and differentiable

$$\partial \tilde{V}_s = 8\left(a_s + \alpha_s\right), \qquad 8\alpha_s = 8a_s - d\tilde{V}_{s+1},$$

$$\partial^2 \tilde{V}_s \equiv \frac{d\tilde{V}_{s+1} - \partial \tilde{V}_s}{\bar{\varphi}_{s+1} - \bar{\varphi}_s} = \frac{d\tilde{V}_{s+1} - 8\left(a_s + \alpha_s\right)}{\bar{\varphi}_{s+1} - \bar{\varphi}_s}. \tag{14}$$

Here, $d\tilde{V}_s \equiv dV/d\varphi(\bar{\varphi}_s)$ are the first derivatives evaluated on the segmentation points. The functional form of the derivative of the potential $dV/d\varphi$ can be controlled by the ``Gradient'' option of the FindBounce function, as explained in Section 4.1.

The complete bounce solution is then constructed on top of the polygonal one $\varphi = \varphi_{PB} + \xi$ with the addition of the non-homogeneous integral pieces

$$\xi = \nu + \frac{4}{D}\alpha\rho^2 + \frac{2}{D-2}\frac{\beta}{\rho^{D-2}} + \mathcal{I}(\rho),$$

$$\mathcal{I} = \int_{\rho_0}^{\rho} dy\, y^{1-D} \int_{\rho_1}^{y} dx\, x^{D-1} \delta dV(x), \tag{15}$$

$$\mathcal{I}_s^{D=3} = \partial^2 \tilde{V}_s \left(\frac{\nu_s - \bar{\varphi}_s}{6}\rho^2 + b_s \rho + \frac{a_s}{15}\rho^4\right),$$

$$\mathcal{I}_s^{D=4} = \partial^2 \tilde{V}_s \left(\frac{\nu_s - \bar{\varphi}_s}{8}\rho^2 + \frac{b_s}{2}\ln\rho + \frac{a_s}{24}\rho^4\right). \tag{16}$$

Similarly to the purely poly-linear case, the α_s parameters are already known and so are the constants that enter in $\mathcal{I}_s$. What remains to be solved are the ν_s, β_s and the new matching radii R'_s.

At first sight this construction might look somewhat cumbersome. In particular, it may seem that the matching is not easy to perform due to the non-linear terms in (16). Keeping in mind though that this is a perturbative expansion on top of the polygonal one, we expect the new matching radii to be close to the polygonal solution $R'_s = R_s(1 + r_s)$ with $r_s < 1$. Thus the matching can be performed to the linear order in r_s and one ends up with a single linear equation [42], which is easy to solve and evaluates quickly.

Such 2nd order extension increases the precision of the polygonal approximation, as demonstrated in [42]. The FindBounce function performs it by default and evaluates the derivatives of $V(\varphi)$ automatically. As discussed in Section 4.1, this behavior can be controlled by the ``Gradient''->Automatic option (default value) and can be turned off with ``Gradient''->None, if the simplest polygonal output is required by the user.

One may also consider including higher orders in the expansion of the potential. However, the matching conditions in this case cannot be solved in closed form and would require computationally expensive numerical root finding at each segment when constructing the bounce solution. It thus turns out that it is practically more convenient to obtain precise bounce configurations by increasing the number field points using the ``FieldPoints'' option.

Multi-field solution. The calculation of the bounce solution in the presence of multiple scalar fields is a non-trivial computational issue. It is technically involved because one needs to find the initial field value in a higher dimensional field space and then integrate the coupled system of potentially stiff PDEs, which is usually done numerically.

There are a number of works [24,27–35] that tackle this issue in different ways. In general, these numerical approaches work either by field path deformation, combined with the single field numerical integration or with multiple shooting. The path deformation approach decouples the single field shooting, freezes the solution and minimizes the potential on this background. This procedure is then iterated. The drawback of this approach is that at each step either kinetic or potential piece of the Euclidean action is not completely extremized, which leads to oscillatory iterations and slows down the convergence. Conversely, direct multi-field shooting faces a highly non-linear increase with the number of fields. Both typically suffer from difficulties in the thin wall regime and provide a purely numerical output.

The polygonal approach to multi-fields addresses these common shortcomings by constructing a semi-analytical solution with the following features.

1. The functional ρ dependence of the multifield field solution remains as it was in the single field case in Eq. (4). This enables a fast numerical evaluation as well as the possibility to iterate. The final result has a closed analytical form, allowing for further manipulation and parameter extraction.
2. Although the solution is built iteratively, a single iteration takes into account both the kinetic and the potential minimization simultaneously when deforming the path in field space via the explicit ρ dependence. This remedies the oscillatory behavior and the solution converges quickly, with the number of iterations reported in the ``PathIterations'' output of the FindBounce method.
3. As in the single field case, the solution is found in terms of the Euclidean radii variables and exists on a compactified region. Therefore it enjoys all the regularities of the single field case and works well within the thin wall regime, as well as outside.
4. Similarly to the extended case above, the multi-field approach is based on a semi-analytical perturbative expansion. By expanding the solution around a particular path, a linear system for path deformation is obtained. Such systems are easily solvable with fast numerical methods and scale linearly with the number of segments and most importantly with the number of fields.

The starting point for the multifield bounce construction is an initial path in field space, denoted by $\bar{\varphi}_{is}$ where i is the field index $i = 1, \ldots, n_f$ and $s = 1, \ldots, N$. This ansatz is fairly arbitrary and can be controlled by the ``FieldPoints'' option to be ether a straight line connecting the two minima, a triangle through an intermediate (e.g. saddle) point, or any path given by the user, see Section 4.1.

Once the initial path is given, the polygonal $\bar{\varphi}_{is}(\rho)$ is computed along this longitudinal direction and provides the basis for perturbation $\varphi_{is}(\rho) = \bar{\varphi}_{is} + \zeta_{is}$. The bounce equation in Eq. (3) now becomes

$$\underbrace{\ddot{\bar{\varphi}}_{is} + \frac{D-1}{\rho}\dot{\bar{\varphi}}_{is}}_{8\bar{a}_{is}} + \underbrace{\ddot{\zeta}_{is} + \frac{D-1}{\rho}\dot{\zeta}_{is}}_{8a_{is}} = \frac{dV}{d\varphi_i}\left(\bar{\varphi} + \zeta\right). \tag{17}$$

In order to stick to the polygonal form, we have to expand the right-hand size around a deformed path $\tilde{\bar{\varphi}}_{is} + \tilde{\zeta}_{is}$ up to the leading constant order in ζ_{is}. Such an expansion keeps the ρ dependence of the field perturbation ζ in the polygonal form

$$\zeta_{is} = v_{is} + \frac{4}{D}a_{is}\rho^2 + \frac{2}{D-2}\frac{b_{is}}{\rho^{D-2}}, \tag{18}$$

which evaluates quickly and, most importantly, can be iterated.

In contrast to the single field case, the determination of the a_{is} requires some care, because the segmentation changes upon iteration. Subtracting the initial longitudinal path $\bar{a}_{is}$, we get

$$8a_{is} \simeq \frac{dV}{d\varphi_i}\left(\tilde{\bar{\varphi}}_{is} + \tilde{\zeta}_{is}\right) - 8\bar{a}_{is},$$
$$\frac{dV}{d\varphi_i} \simeq \frac{1}{2}\left(d_i\tilde{V}_s + d_i\tilde{V}_{s+1} + d_{ij}^2\tilde{V}_s\tilde{\zeta}_{js} + d_{ij}^2\tilde{V}_{s+1}\tilde{\zeta}_{js+1}\right). \tag{19}$$

The `FindBounce` package evaluates the gradient on segmentation points $d_i\tilde{V}_s \equiv dV/d\varphi_i(\tilde{\bar{\varphi}}_s)$ using the vector of functions given by the ''Gradient'' option. By default, the gradient is extracted analytically from $V(\varphi)$, but it can also be pre-computed and given by the ''Gradient'' option.

Similarly, the mixed second derivatives are given by the ''Hessian'' matrix $d_{ij}^2\tilde{V}_s \equiv d^2V/d\varphi_i d\varphi_j(\tilde{\bar{\varphi}}_s)$, see Section 4.1. The essential reason why the Hessian is needed is because the path in field space is typically curved, which requires the inclusion of second order derivatives that couple the path deformation displacements $\tilde{\zeta}_i$ in different field directions, which produces a coupled linear system for $\tilde{\zeta}_i$.

The rest of the construction proceeds as in the extended polygonal case by matching the field values and derivatives

$$b_{is} = b_{i1} + \sum_{\sigma=1}^{s-1}\frac{4}{D}\left(a_{i\sigma+1} - a_{i\sigma}\right)R_\sigma^D + \frac{1}{2}\left(\dot{\tilde{\varphi}}_{i\sigma+1} - \dot{\tilde{\varphi}}_{i\sigma}\right)R_\sigma^{D-1}, \tag{20}$$

$$v_{is} = v_{i1} - \sum_{\sigma=1}^{s-1}\frac{4}{D-2}\left(a_{i\sigma+1} - a_{i\sigma}\right)R_\sigma^2 - \frac{1}{D-2}\left(\dot{\tilde{\varphi}}_{i\sigma+1} - \dot{\tilde{\varphi}}_{i\sigma}\right)R_\sigma. \tag{21}$$

These recursion relations closely parallel the ones in (5), apart from the addition of field indices and the subtraction of the longitudinal ansatz $\bar{\varphi}$ that has discrete discontinuities in derivatives between segments.

The rest of the setup proceeds similarly to the single field case; we refer the reader to [42] for a detailed discussion. Again, we distinguish between the two cases. Case A has a moving initial point φ_{i0}, while in case B it is fixed in the true vacuum. The matching conditions are derived by requiring the fields to be continuously differentiable along the deformed path, i.e. $\zeta_{is}(R_s) = \tilde{\zeta}_{is+1} = \zeta_{is+1}(R_s)$ and $\dot{\zeta}_{is}(R_s) = \dot{\zeta}_{is+1}(R_s)$.

The linear system for $\tilde{\zeta}_{is}$ is then solved with the `Mathematica`'s native `LinearSolve` function. This procedure is iterated until any of the following requirement is achieved: maximum number of iterations controlled by ''MaxPathIterations'', the amount of path deformation set by ''PathTolerance'' or the precision of the action set by ''ActionTolerance'', as explained in Section 4.1.

3. Quick installation and running guide

3.1. Download and installation

The `FindBounce` package is released in the `.paclet` file format, which contains all the code, documentation and other necessary resources. The latest version of `.paclet` file can be downloaded from the repository "releases" page (https://github.com/vguada/FindBounce/releases) and can be installed by evaluating the following code in `Mathematica`.

```
In[1]:= PacletInstall["full/path/to/FindBounce-X.Y.Z.paclet"]
```

This will permanently install the `FindBounce` package to the `$UserBasePacletsDirectory`. To update the documentation, it may be necessary to restart `Mathematica`. `Mathematica` will always use the latest installed version. All the previously installed versions of FindBounce can be enumerated by evaluating `PacletFind[''FindBounce'']`. More detailed information about the FindBounce package can be found with `PacletInformation[''FindBounce'']`. All the versions can be uninstalled with `PacletUninstall[''FindBounce'']`.

3.2. Running

Once the package is installed, load it with `Needs`.

```
In[1]:= Needs["FindBounce`"]
```

To access the documentation, open the notebook interface help viewer and search for `FindBounce`. Let us show how `FindBounce` can be used on a simple example

In[2]:= `V[x_]:= 0.5 x^2 + 0.5 x^3 + 0.12 x^4;`

In[3]:= `extrema = x/.Sort@Solve[D[V[x],x]==0];`

The bounce is obtained with the `FindBounce` function

In[4]:= `bf = FindBounce[V[x],x,{extrema[[1]],extrema[[3]]}]`

where the order of the minima is arbitrary.

Out[4]= `BounceFunction[` `]`

Here, all the options have their default values and the results can be extracted as follows:

In[5]:= `bf["Action"]`

Out[5]= `73496.`

In[6]:= `bf["Dimension"]`

Out[6]= `4`

Notice that the summary box outputs rounded values, e.g. for the action, while directly accessing values from the `BounceFunction` object is done with default precision.

This concludes the simplest demonstration for single field use. More details regarding the other available options and their use are available in Section 4.1. Before moving on, let us briefly comment on `FindBounce` argument overloading. It is possible to study arbitrary purely polylinear potentials by providing a set of points and their potential values $\{\tilde{\varphi}, V(\tilde{\varphi})\}$:

`FindBounce[{{x1,V1},{x2,V2}, . . . }]`

An example of such use is given in Section 5.3. Finally, the `FindBounce` function is overloaded for use with multi-field potentials, in which case the evaluation is done by the following syntax:

`FindBounce[V[x,y, . . . ],{x,y, . . . },{m1,m2}]`

where $m_{1,2}$ are the two multi-field minima; see Section 5.5 for definite examples.

3.3. Contributing

Please use the `issues` (https://github.com/vguada/FindBounce/issues) page on the `GitHub` repository to submit bugs or feature ideas. Pull requests are welcome, however in case of major changes, please open an issue first to discuss what you would like to change. For developers, the instructions on how to run the tests, build the package `.paclet` file from the source code, and create the documentation, can be found in the `CONTRIBUTING.md` file.

4. Using the `FindBounce` and its output

4.1. FindBounce options

In this section we list and describe the available options of the `FindBounce` function. They are directly accessible within `Mathematica` using the `FindBounce` function documentation, which includes detailed descriptions and examples. The options and default values can be listed with the syntax

`Options[FindBounce]`

returning the options described below.

- `"ActionTolerance"` controls the relative variation of the bounce action $|\Delta S|/S$ between iterations of the initial radius R_{in} while solving the boundary conditions in (5). `FindBounce` also supervises the change of the action after each deformation of the path. The default tolerance value is 10^{-6}. See for example the right panel of Fig. 4.
- `‘‘BottomlessPotential’’` is a Boolean variable used to specify whether the combination of polygonal and quartic should be used. This option can be used to deal with single field potentials unbounded from below, as discussed in Section 5.4. The default value is `False`.

- ''Dimension'' defines the number of space–time dimensions, given by the integer D. The default value is $D = 4$ for zero temperature tunneling via quantum fluctuations. In thermal field theory, the $D = 3$ action is needed, which is obtained by the following evaluation.

  ```
  In[7]:= bf3 = FindBounce[V[x],x,{extrema[[1]],extrema[[3]]},"Dimension"-> 3];
  ```

- ''FieldPoints'' controls the segmentation in field space. It is specified either by the integer number of field values or by an explicit list of field values $\{\tilde{\varphi}_1, \ldots, \tilde{\varphi}_N\}$ given by the user.

  - ''FieldPoints''-> 31 by default. When specified by a single integer N, the segmentation is homogeneously split into N equidistant field points. For multiple scalar fields, the initial trajectory is taken to be a straight line connecting the two minima, e.g. the black dot-dashed line in Fig. 8.
  - ''FieldPoints''-> $\{\tilde{\varphi}\}$. Arbitrary segmentations can be given with an explicit set of field coordinates as an input. The order of the minima is not important, FindBounce recognizes the higher one as the false vacuum state. For single field potentials field points remain fixed, while for multi-fields they move when the path is being updated.

- ''Gradient'' controls the evaluation of the bounce beyond the poly-linear approximation of $V(\varphi)$, as in (13). There are three available option values.

  - ''Gradient''-> Automatic by default. FindBounce computes the bounce by taking into account the 2nd order approximation to $V(\varphi)$ in (13). The gradient function(s) of the potential needed in (19) are obtained analytically by running Grad[V[x,y, . . .],{x,y, . . . }].
  - ''Gradient''-> ''FiniteDifference''. The set of gradient functions required in (19) are approximated by finite differences, where the small variations of the fields $\Delta\varphi$, are proportional to the total length of the path as $\Delta\varphi \equiv 10^{-4}L_{path}$. This option is suitable for non-analytic potentials, or when the default derivative, given by Mathematica, may be complicated and leads to delays in evaluation.
  - ''Gradient''-> $\{dV/d\varphi_i\}$. The set of gradient functions can also be pre-computed, stored and given as an input with this option. This can be used in multiple evaluations and scanning to save the computation time. See for example Section 5.5.
  - ''Gradient''-> None. With this setting, the 2nd order extension is turned off and the polygonal method is implemented in the poly-linear approximation. This may be necessary when the derivative of the potential is discontinuous, as in Section 5.2.

- ''Hessian'' option for multi-field bounce calculations regulates the evaluation of the second derivatives $d^2V/d\varphi_i d\varphi_j$ in (19).

  - ''Hessian''-> Automatic is the default behavior, where Mathematica computes the Hessian matrix analytically by running Grad[Grad[V[x,y,. . .],{x,y,. . . }],{x,y,. . . }].
  - ''Hessian''-> ''FiniteDifference'' approximates the Hessian matrix required in (19) with numerical finite differences. Similarly to the ''Gradient'' option, the variations of the fields are computed from the path length and given by $\Delta\varphi \equiv 10^{-4}L_{path}$.
  - ''Hessian''-> $\{d^2V/d\varphi_i d\varphi_j\}$. Similarly to the ''Gradient'' option above, the Hessian matrix of functions can be provided externally by the user to speed up the calculation.

- ''MaxPathIterations'' can be used to control the maximum number of times the path can be iterated after starting from the initial ansatz; the default value is 3. See for example Section 5.5.
- ''MaxRadiusIterations'' sets the maximum number of iterations to compute the initial radius R_{in} that satisfies Eq. (12); the default value is 100. Generically, FindBounce takes about $O(1)$ iterations to compute the action up to the default tolerance value. However, this option may be overridden by the ActionTolerance requirement, which prioritizes the precision of the action and thus allows for a larger number of iterations.
- ''MidFieldPoint'' allows the user to control the segmentation by setting a single arbitrary intermediate field point between the two minima, such as saddle points.

  - ''MidFieldPoint''-> None by default: the segmentation is a homogeneously discretized straight line in field space connecting the two minima.
  - ''MidFieldPoint''-> Automatic. The straight line connecting the two minima is divided by $\tilde{\varphi}_{max}$ into two homogeneous segmentations. The intermediate field point $\tilde{\varphi}_{max}$ is the local maximum of the V on the straight line, and is computed with FindMaximum. This option is suitable for nearly flat or very asymmetric potentials, where automatic segmentation may not detect the maximum unless a large value for ''FieldPoints'' is used.
  - ''MidFieldPoint''-> $\tilde{\varphi}_{int}$ The segmentation is divided by $\tilde{\varphi}_{int}$ into two homogeneous segmentations. It consists of two straight lines that connect the two minima with the intermediate field point $\tilde{\varphi}_{int}$.

  See Section 5.2 for single field and section Section 5.5 for a multi-field example.

- ''PathTolerance'', controls the search with multiple scalar fields, where the path in field space changes with each iteration. Its value specifies the maximal allowed deviation of the path from one iteration to another. It is defined as the maximum length of deformation of any field point after each iteration, normalized to the total length of the path: $\max_s |\tilde{\zeta}_s| /L_{path}$ with default value 10^{-2}.

The FindBounce stops if either ''MaxPathIterations'', ''PathTolerance'' or ''ActionTolerance'' is satisfied.

4.2. Bounce function output and manipulation

The results of the calculation are bundled in the `BounceFunction` container, which stores various parameters and other properties of the solution. The `BounceFunction` is rendered in `Mathematica` notebooks with a summary box that contains some minimal amount of information: the shape of the field solution, Euclidean action and the number of space–time dimensions. The list of all the available properties can be accessed with the following syntax:

```
In[8]:= bf["Properties"]
```

```
Out[8]= {Action, BottomlessPotential, Bounce, Coefficients, CoefficientsExtension,
        Dimension, Path, PathIterations, Radii}
```

- ''`Action`'' gives the value of the Euclidean bounce action.
- ''`BottomlessPotential`'' returns the constant factor of the quartic potential V_0 of Eq. (24). See Section 5.4 for an example.
- ''`Bounce`'' returns the piece-wise smooth function that characterizes the bounce solution $\varphi(\rho)$. It can be evaluated as a continuous function, see section Section 5.5 for an example.
- ''`Coefficients`'' provides the constant factors $\{v_{is}, a_{is}, b_{is}\}$ in Eqs. (4) and (18) that define the multi-field (polygonal) bounce solution in each segment. See Section 5.4 for an example.
- ''`CoefficientsExtension`'' provides the constant factors $\{v_{is}, \alpha_{is}, \beta_{is}, \partial^2 \tilde{V}_s\}$ in Eqs. (14) and (15) that define the extension of the multi-field (polygonal) bounce solution ξ and $\mathcal{I}$ in each segment.
- ''`Dimension`'' returns the number of space–time dimensions in which the bounce was computed, where finite (zero) temperature corresponds to $D = 3$ ($D = 4$).
- ''`PathIterations`'' reports the number of times the path in field space was deformed from the initial ansatz and an upper limit set by the ''`MaxPathIterations`'' option.
- ''`Path`'' gives a list of points $\tilde{\varphi}_s = \varphi(R_s)$ that defines the trajectory of the bounce in field space. This output can be used as an initial path ansatz to save time when finding the bounce solution for similar potentials, see section Section 5.5 for an example.
- ''`Radii`'' returns the list of radii R_s where the segments are joined from R_{in} to R_{N-1}.

BouncePlot. In addition to the `BounceFunction` described above, a plotting function wrapper `BouncePlot` is available, such that the field configuration(s) can be plotted with ease. The `BouncePlot` behaves similarly to the native `Plot`, where the default options can be changed as shown in the example below. Multi-field bounce solutions, given as a list of functions, can also be plotted simultaneously.

```
In[9]:= BouncePlot[{bf3,bf}, PlotLegends-> Placed[{"D=3","D=4"}, {Right,Center}]]
```
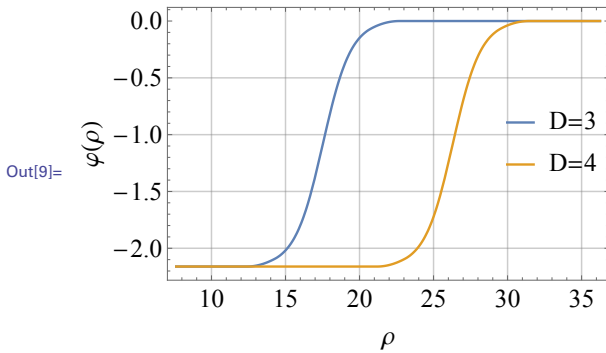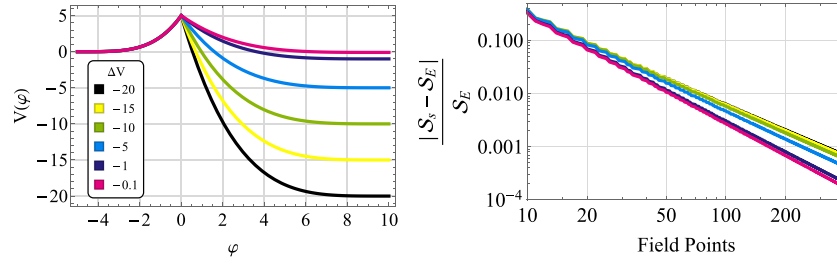

5. Examples

This section contains a number of examples, test cases and demonstrations of the `FindBounce` method. Each subsection contains a simple self contained code that can be easily reproduced in `Mathematica`. All of the examples given here (and more) are available also in the `Mathematica` documentation of `FindBounce` after installation.

We start with the single field benchmark in Section 5.1 that showcases the use of some available options and return values of `FindBounce` listed above in Section 4. One of the main aspects is the performance in terms of the precision of the action and timing in thick and thin wall regimes. In particular, timing scales linearly with the number of field points.

We consider the exactly solvable quartic-quartic case in Section 5.2 that shows how `FindBounce` deals with such non-trivial cuspy potentials. In Section 5.3 we use the purely polygonal example with $N = 5$ points to reproduce the curious case of the disappearing instanton in the presence of additional local minima. We show how `FindBounce` can estimate the escape point φ_0 of unstable potentials in Section 5.4 by combining the exact linear solution to an exact (unstable) quartic one.

For multi-field applications, we first perform the simplest study with two fields in Section 5.5, where we demonstrate how to control the precision and speed of the evaluation as well as the use of an arbitrary path ansatz. We also demonstrate the use of `FindBounce` on effective and thermal potentials and give a demonstration on computing the nucleation temperature. In the final sub-section, Section 5.7, we address the bounce calculation with an arbitrary number of fields and test the `FindBounce` method with up to twenty fields. We compare the results with other existing methods and demonstrate that the time demand of the `FindBounce` function scales *linearly* with the number of fields.

Fig. 3. Left: The benchmark potential from Eq. (22) for different values of α going from thick $\alpha = 0.6$ to thin wall $\alpha = 0.99$. Right: The bounce action $\mathcal{S}_s$ for each potential configuration and a given number of field points s, normalized to $s = 400$ and computed in $D = 4$.

Fig. 4. Left: Evaluation time with respect to the number of field points, averaged over two intervals of α corresponding to thin and thick wall regimes. Right: The bounce field configuration and action with $N = 31$ (default) field points for different tolerance value of the action controlled by ``ActionTolerance''. Reference values of the action for ``FieldPoints''-> 10 and 100 field points with ``ActionTolerance''-> 10^{-6} (default) are shown on the green background. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.1. Single field benchmark

Let us consider a generic renormalizable scalar potential with terms up to the quartic power. Such potentials in general feature an unstable ground state, as seen on the left panel of Fig. 3. Using the re-scaling properties of both the field φ and the ρ from (3), one can rewrite V as a function of a single parameter α, as [62]

$$V(\varphi) \equiv \frac{1}{2}\varphi^2 + \frac{1}{2}\varphi^3 + \frac{\alpha}{8}\varphi^4. \tag{22}$$

Here $0 \leq \alpha \leq 1$ covers all the possible scenarios going from thick to thin wall respectively. The bounce configuration for a particular α and the number of field points N in 3 space–time dimension can be obtained by FindBounce with:

```
In[1]:= V[x_,a_]:= 0.5 x^2 + 0.5 x^3 + 0.125 a x^4;
```

```
In[2]:= extrema[a_]:= x /. Sort@NSolve[(D[V[x,a], x]) == 0, x];
```

```
In[3]:= bf[a_,n_Integer]:= FindBounce[V[x,a], x, extrema[a][[{1,3}]],
        "FieldPoints"-> n, "Dimension"-> 3];
```

```
In[4]:= bf[0.6,100]
```

Out[4]= BounceFunction[▦ ⟋   Action: 44.5
                                 Dimension: 3]

```
In[5]:= bf[0.6,100][{"Action","Dimension"}]
```

Out[5]= {44.5098, 3}

The resulting action for different values of α is plotted on the right panel of Fig. 3 and is normalized to $N = 400$ field points, which is already very precise. The accuracy of the action improves with the number of field points and goes below the percent level with $N = 30$ segmentation points in both, thin and thick wall. As expected, the convergence is faster for thin walls since the minimal $N = 3$ case already coincides with the thin wall solution. Note also that in this case, the 2nd order correction of the potential is taken into account by default and the convergence is faster than the pure polygonal calculation.

The construction of the bounce solution, in particular the v_s, b_s parameters, already indicates that adding more segments does not require significant additional computing cost: evaluation time of the sums in (5) grows linearly. Thus it is to be expected that the total time should grow linearly with the number of field points, which is indeed the case, as displayed in Fig. 4. Note that the FindBounce performance is similar for both thick and thin walls, with a $\sim 10\%$ faster evaluation in the thin wall regime.

Fig. 5. Left: The piecewise quartic potential for different values of the potential difference between the vacua, going from the thin wall $\Delta V = -0.1$ to thick wall $\Delta V = -20$ regime. Right: The bounce action S_s for different number of field points, normalized to the exact result S_E of the quartic-quartic potential.

This behavior should be contrasted with numerical approaches based on under–over-shooting. There, the very thin wall limit requires an exponential amount of precision in finding φ_0.[1] This may cause numerical approaches to fail or significantly reduce the speed of computation. `FindBounce` instead relies on the R_{in}, which is found by extremizing the action, similar to the original thin wall approach [2]. Such change of variables thus provides a more stable universal behavior near the thin wall limit.

Speed and accuracy can be controlled as explained in Section 4.1. The bounce field configuration and the action can be computed with different requirements for ``ActionTolerance``, as shown on the right panel of Fig. 4. Even though the boundary conditions are not exactly satisfied at the first segment around $\rho \simeq 3$, the rest are joined analytically and the solution is smooth. Despite the discontinuity, the bounce action is fairly precise and within the limits of required action tolerance. For comparison, we also show the Euclidean action with different number of ``FieldPoints``-> 10, 100, computed with the default ``ActionTolerance``-> 10^{-6}, which shows how the action converges with the number of field points.

As a final comment, one can use the ``FieldPoints`` option to specify a fixed custom segmentation from which the bounce is obtained. This feature may be useful when dealing with non-homogeneous potentials that contain flat pieces, followed by local features. In such cases, rather than increasing the number of field points, constructing a custom segmentation may be more beneficial. In the following subsection we give one such example, where a bi-homogeneous segmentation, set by the ``MidFieldPoint`` option gives a more stable output.

5.2. Quartic-quartic potential

Perhaps the cleanest way of testing the `FindBounce` package is to compare it to one of the few non-trivial analytical closed-form solutions that are available [9,37,40,63]. Here, we consider the two joined quartic potentials studied in [40].

For the sake of illustration we choose a potential with fixed local extrema and leave a single parameter ΔV, which controls the potential difference between the false and true vacua. As shown on the left of Fig. 5, we vary ΔV from -20 (thick wall) to $\Delta V = -0.1$ (thin wall). The corresponding action values are computed with different number of field points and are shown on the right panel. They are evaluated at $D = 4$ and normalized to the known exact value.

The `FindBounce` syntax used for this calculation is again simple.

```
In[1]:= DV = -20;
        V[x_]:= Piecewise[ {{0.008 (x + 5)^4, x < 0},
          {DV + (5 - DV)(x/10 - 1)^4, x >= 0}} ];

In[2]:= FindBounce[V[x], x, {-5, 10}, "Gradient"-> None, "MidFieldPoint"-> 0]
```

Notice that the first derivative of the potential, required in Eq. (14), is not well defined at the origin $\varphi = 0$, therefore the default evaluation of the gradient, as well as the automatic extension of the polygonal approach, was turned off with ``Gradient``-> None. If this were not the case, `FindBounce` would issue a warning message and automatically return the solution computed without the gradient extension. Due to the absence of the second order correction, the convergence is a bit slower compared to the previous example in Section 5.1. Nevertheless, `FindBounce` finds the solution within 1% accuracy for 50 (100) field points in thin (thick) wall regime.

The other option used above is the ``MidFieldPoint``, which was used to set the intermediate field point of the segmentation to $\varphi = 0$, see the left panel of Fig. 5. This feature is optional but stabilizes and improves the accuracy of the bounce action, especially with a small number of field points.

5.3. Intermediate minima and disappearing instantons

A number of physically motivated models may feature a non-trivial potential with many local minima. Such situations appear in multiple axion and relaxion-type potentials [64–72]. Here we demonstrate the use of the polygonal approach to analyze the minimal polygonal potential with an intermediate minimum, i.e. the two-triangle construction with $N = 5$ field points, shown on the left of Fig. 6.

The value of the potential at the mid-point $\tilde{V}_3$ controls the depth of the intermediate minima and the resulting bounce solution. As this is lowered below the highest extremum $\tilde{V}_3 < \tilde{V}_5$, two types of transitions are possible. The direct tunneling from $\tilde{V}_5$ to $\tilde{V}_1$ (the ABC)

---

[1] As in [2], when φ_0 is very close to the true vacuum $\tilde{\varphi}_1$, the solution is given by $\varphi(\rho) - \tilde{\varphi}_1 = 2(\varphi_0 - \tilde{\varphi}_1) I_{D/2-2}(m\rho)/(m\rho)$, where $m^2 \equiv V''(\tilde{\varphi}_1)$. Thus in the thin wall limit $R_{in} \gg m^{-1}$ the initial condition $\varphi(R_{in}) \equiv \varphi_0$ is exponentially tuned.
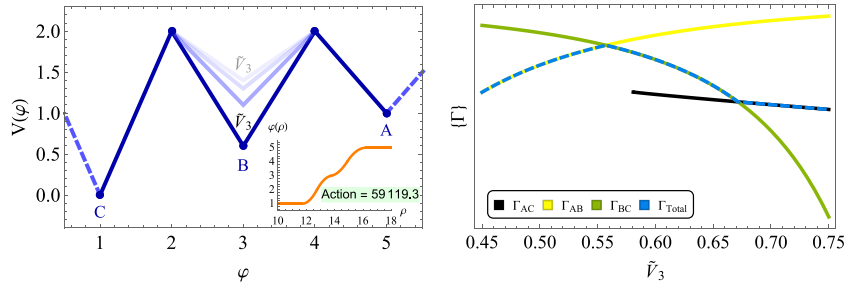
Fig. 6. Left: The minimal example of a potential with an intermediate minimum. The inset shows the bounce solution with the typical double bubble wall shape. Right: Decay rates for the direct (ABC) and the two subsequent (AB, BC) transitions, together with the total decay rate (23).

instanton, or a two-step transition first from $\tilde{V}_5$ to $\tilde{V}_3$ (AB) and then from $\tilde{V}_3$ to $\tilde{V}_1$, the BC instanton. The right panel of Fig. 6 shows the associated rates and the total decay rate, defined by

$$\Gamma_{total} \approx \frac{\Gamma_{AB}\Gamma_{BC}}{\Gamma_{AB} + \Gamma_{BC}} + \Gamma_{AC}. \tag{23}$$

The point of emphasis is that the direct ABC transition exists only up to a certain value of $\tilde{V}_3$ and then suddenly disappears. This behavior of a disappearing instanton was pointed out in [73] and explained in the thin wall approximation. In order to construct the bounce of the direct solution, the field should traverse from $\tilde{\varphi}_1$ to $\tilde{\varphi}_3$ with a radius associated to the BC transition and finally from $\tilde{\varphi}_3$ to $\tilde{\varphi}_5$ at the AB radius. Clearly, to have a meaningful solution $R_{BC} < R_{AB}$. However, when we lower the intermediate minimum, $\tilde{V}_3$ comes increasingly close to $\tilde{V}_1$, thus R_{AB} grows larger, and thereby Γ_{AB} decreases until direct tunneling via the ABC instanton becomes impossible. This is seen on the right panel of Fig. 6, where the Γ_{ABC} in black suddenly disappears. Of course, the two-step decay still exists, i.e. Γ_{AB} and Γ_{BC} are in fact non-zero.[2]

To study such particularly simple settings, FindBounce allows the user to manually set individual values of the potential as a list of points in field space $\left\{ \left\{ \tilde{\varphi}_1, \tilde{V}_1 \right\}, \ldots, \left\{ \varphi_N, \tilde{V}_N \right\} \right\}$. For example, the direct ABC instanton of Fig. 6 is obtained with the following syntax.

```
In[1]:= FindBounce[{{1,0},{2,2},{3,0.6},{4,2},{5,1}}]
```

```
Out[1]= BounceFunction[ ⊞  Action: 5.91 × 10⁴
                             Dimension: 4 ]
```

Note also that the order of points in the list is arbitrary and can be given from left to right or vice versa.

5.4. Potentials unbounded from below

In this section, we demonstrate how FindBounce can be used to deal with unstable potentials. Many BSM theories contain portions of parameter space with unstable field directions, such that local ground states may tunnel into the unstable region. This seems to be the case for the Standard Model as well, with the estimated lifetime significantly longer than the age of the universe, see e.g. [80–85].

Note that the quartic-quadratic potentials do not admit the bounce solution at tree level due to scale invariance [86], however the bounce may exists after the inclusion of quantum corrections. In any case, one might expect the unstable direction to be dominated by the negative quartic term at large field values.

The escape point can be found either by the usual numerical shooting method or with the polygonal approach. However, due to the steepness of the unstable direction, polygonal segments might need to be extend to large values, which may require a large number of segments. In order to provide a good estimate for large φ_0, we demonstrate how the poly-linear potential with many segments can be joined with an exact unstable quartic solution. This functionality is implemented in FindBounce single field potentials in $D = 4$. It can be turned on with the ``BottomlessPotential''-> True, option, together with the field values $\tilde{\varphi}_N$ and $\tilde{\varphi}_2$. The latter is the point where the PB is connected to the unstable quartic.

To understand how the estimate works, consider the quartic potential V_q and the associated solution

$$V_q(\varphi) = V_0 - a_q \left(\varphi - v_q \right)^4, \qquad \varphi_q(\rho) = v_q + \frac{b_q}{1 + \frac{1}{2}a_q b_q^2 \rho^2}, \tag{24}$$

where v_q and b_q are constants of integration and $a_q > 0$ is a dimensionless parameter of the potential. Assuming the unstable φ^4 term dominates for large field values, a_q is fixed by equating it to the φ^4 coefficient of the input potential $V(\varphi)$.

Similar to the pure polygonal construction explained in section Section 2, the quartic piece of the potential can be matched to the polygonal ones. In particular, the coefficients b_q, v_q are then determined by the boundary conditions $\varphi_q(R_2) = \varphi_{PB}(R_2) = \tilde{\varphi}_2$ and $\dot{\varphi}_q(R_2) = \dot{\varphi}_{PB}(R_2)$. What remains to be determined is the matching radius R_2, which can be found with FindRoot, similar to the polygonal case above. Finally, we fix V_0 by requiring the potential to be continuous.

---

[2] Strictly speaking, one should perform the two-step analysis more carefully, and allow the field to develop in real time after tunneling from $\tilde{V}_5$ in the vicinity of $\tilde{V}_3$. This would lead to oscillations with subsequent decay, see e.g. [74–78] and potential enhancement of the rate. We leave this interesting question for future studies [79].
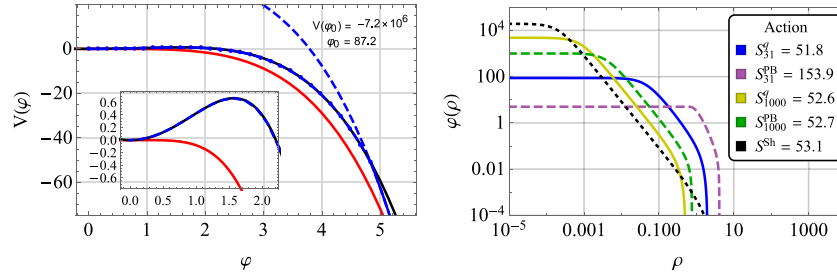
Fig. 7. Left: Potential unbounded from below. The pure quartic, polygonal-quartic and input potentials are shown in red, solid blue and black, respectively. The dashed blue line is the pure quartic potential that is joined to the piece-wise polygonal potential. Right: The bounce field configuration with its bounce action. The solid lines show the field configuration with PB+quartic estimate, the dashed are the pure polygonal solutions and the dotted line is the numerical result obtained with over–under-shooting. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

We demonstrate the use of `FindBounce` in such unbounded transitions with the following example, where we specify the minimum at $\varphi = 0$ and the connecting point to the quartic at $\varphi = 5$.

```
In[1]:= V[x_] := 0.5 x^2 + 0.05 x^3 − 0.125 x^4;
```

```
In[2]:= bf = FindBounce[V[x], x, {0, 5}, "BottomlessPotential"-> True];
```

```
In[3]:= {vq, aq, bq} = bf["Coefficients"];
        V0 = bf["BottomlessPotential"];
```

```
In[4]:= Plot[V0 − aq[[1]](x − vq[[1]])^4, {x, bf["Path"][[1,1]],bf["Path"][[2,1]]}]
```

The potentials of interest are shown on the right side of Fig. 7 where the red, solid blue and black lines represent the pure quartic V_q, polygonal-quartic and the input potential, respectively. The dashed blue line is the pure quartic potential (24) that was joined to the polygonal potential. Its parameters are given by ''Coefficients'' and `BottomlessPotential` as shown in the syntax below. The bounce solution given by `FindBounce` is then shown on the right of Fig. 7. Notice that the materialization of the bounce happens at $\varphi_0 \sim 90$, much above the connecting point at $\varphi = 5$.

5.5. Two field benchmark

Many extensions of the SM feature additional scalar fields, see [87] for a review. Extra scalars can couple to the SM Higgs and may alter the vacuum structure, potentially triggering a first order phase transition. The number of additional fields in generic SM extensions may be large. However, in many cases it may be sufficient to consider the dynamics of two fields only.

As a simple multi-field example in `FindBounce`, let us consider such two field potential with parameters chosen such that the field dynamics produces a large curvature in field space. We start with the simplest tree-level potential at zero temperature, compute the bounce action and show some additional features of `FindBounce`. In Section 5.6 we show how to deal with finite temperature and thermal corrections.

Let us then consider the following example, where we call `FindBounce` on the two minima and ask it to connect them with an intermediate point, set by ''MidFieldPoint''. We also show how the pre-computed gradient function can be specified.

```
In[1]:= V[h_,s_] := -100 h^2 + 0.1 h^4 - 60 s^2 + 0.3 s^4 + 3 h^2 s^2;
        minima = {{0.,10.},{22.4,0.}};
```

```
In[2]:= bf = FindBounce[V[h,s], {h,s}, minima, "MidFieldPoint"-> {6,6},
         "Gradient"-> {-200 h + 0.4 h^3 + 6 h s^2, -120 s + 1.2 s^3 + 6 s h^2}];
```

```
In[3]:= {Ri,Rf} = bf["Radii"][[{1,-1}]];
        Show[
         ContourPlot[V[h,s], {h,-1,25}, {s,-1,11},Contours->50],
         ParametricPlot[ Through@bf["Bounce"][r], {r, Ri, Rf}]]
```

```
In[4]:= BouncePlot[bf,
         PlotLabel->Row[{"Action = ",Round@bf["Action"]}],
         PlotStyle->{Purple, Orange}]
```

The code above returns the bounce field configuration corresponding to the solid blue line in field space, shown on the left of Fig. 8. The Euclidean time profiles $(h(\rho), s(\rho))$ can also be plotted easily and are shown on the right of Fig. 8.

The dashed lines in Fig. 8 represent the different choices of the initial path. By default, `FindBounce` chooses a straight line from one minimum to the other, seen by the black dashed line. In case there is a specific point that the segmentation should follow, such as the saddle point, or an arbitrary point in the above example, it is specified with the ''MidFieldPoint'' option. Finally, one can start with a completely arbitrary initial path, set by the ''FieldPoints'' option. In the example above, the dashed red line on the
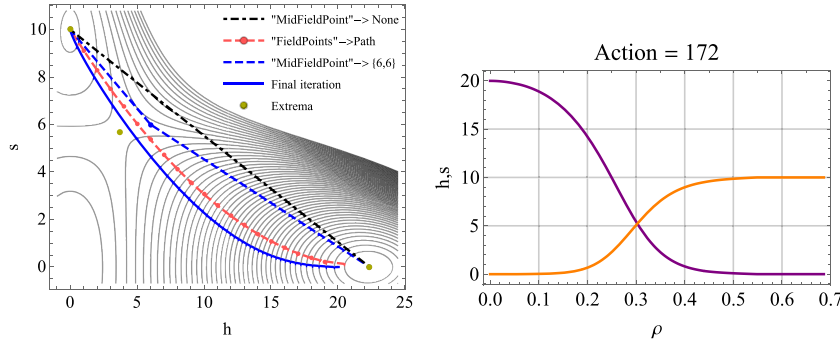
Fig. 8. Left: Potential contours in solid black lines, the final trajectory of the bounce field in solid blue line and three different initial paths in dashed. Black dot-dashed line is the default straight line ansatz, the straight dashed blue line includes the intermediate point at {6,6}, while the red dashed line is a parabola, set by hand. Right: The field configuration $(h(\rho), s(\rho))$ of the final iteration and the associated bounce action in the caption. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

left of Fig. 8 was obtained with a parabola connecting the two minima. The latter option is particularly useful when we already have some idea about the path in field space, e.g. when performing potential parameter scans.

Whatever the choice of the initial path is, FindBounce iterates the path deformation procedure until it reaches at least one of the following three conditions.

1. The maximum number of iterations, controlled by ''MaxPathIterations''. Here, zero means no perturbation of the initial path; i.e. all the dashed lines in Fig. 8 were obtained by setting ''MaxPathIterations''-> 0.
2. Path deformation measure, controlled set up by ''PathTolerance'', as explained in Section 4.1.
3. ''ActionTolerance'' that directly measures the change of the Euclidean action between iterations.

The resulting field configurations are shown in solid blue line on the left of Fig. 8.

5.6. Thermal corrections and nucleation temperature

As a phenomenological application of FindBounce, let us compute the nucleation temperature of a SM extension with a real scalar singlet. This model may support a first order phase transition and might successfully explain phenomena such as baryogenesis [88–92], dark matter [93–96] and act as a possible source of gravitational waves [97–101].

We consider the SM Higgs h together with the singlet scalar singlet field s. For simplicity, we assume an additional $\mathbb{Z}_2$ symmetry and define the tree-level renormalizable potential

$$V_{\text{tree}}(h, s) = -\frac{1}{2}\mu_h^2 h^2 + \frac{1}{2}\mu_s^2 s^2 + \frac{1}{4}\lambda_h h^4 + \frac{1}{4}\lambda_s s^4 + \frac{1}{4}\lambda\, s^2 h^2. \tag{25}$$

One-loop thermal corrections to the potential above can be computed using the equations presented in Appendix A. For excellent references, see [102,103], a review [104] or textbook [105]. The exact thermal one-loop functions $J_{B/F}$ in (A.3) were implemented efficiently in C++ [106] and can be employed in Mathematica with the provided interface. However, we remain in the high-T limit and use the closed form given in Eqs. (A.4) and (A.5). These are valid up to $\mathcal{O}(T^4)$, while neglecting the contributions of the quartic coupling of the potential for simplicity.

As shown in [107], the thermal one-loop corrections to the potential in (25) at high-T are given by

$$V(h, s, T) = \frac{1}{2}\left(c_h h^2 + c_s s^2\right) T^2, \tag{26}$$

$$c_h = \frac{2M_W^2 + M_Z^2 + m_h^2 + 2m_t^2}{4v^2} + \frac{\lambda}{4!}, \qquad c_s = \frac{2\lambda + 3\lambda_s}{12}. \tag{27}$$

Here we considered the W, Z gauge bosons and the top quark contributions to the effective potential and neglected the other light fermions.

Following [35,107], the tree level potential (25) can be redefined as a function of the critical temperature and couplings of the singlet fields $\{T_C, \lambda_s, \lambda\}$ respectively. Taking into account the constraints that lead to a first order phase transition and requiring that the deeper minimum becomes our Higgs vacuum as T is lowered, the effective quadratic couplings turn into:

$$\mu_h^2(T) = \lambda_h v_h^2(T), \qquad \mu_s^2(T) = -\lambda_s v_s^2(T), \qquad \lambda_h = \frac{m_h^2}{2v^2}. \tag{28}$$

The corresponding vacuum expectation values (vevs) are then

$$v_h^2(T) = v^2\left(1 - \frac{2c_h}{m_h^2}T^2\right),$$

$$v_s^2(T) = \frac{1}{\lambda_s}\left(\left(\frac{m_h}{2v}v_h^2(T)\sqrt{2\lambda_s} + c_s T_C^2\right) - c_s T^2\right), \tag{29}$$

where v is the SM vev at zero temperature and the minima of the potential are $\{v_h(T), 0\}$ and $\{0, v_s(T)\}$.

With the thermal and quantum corrections in place, we can show how the nucleation temperature T_N can be computed with FindBounce. The T_N is defined as the temperature when the probability for a single bubble to be nucleated within one horizon volume is ~ 1 [108]. Assuming radiation domination and the SM degrees of freedom in the thermal plasma, the above requirement roughly translates to $B = S_3/T_N \approx 140$.

Let us consider the benchmark in [35,36] and use FindBounce to determine the variation of B with temperature.

```
In[1]:=  V[h_, s_, T_] := -uh2[T]/2*h^2+us2[T]/2*s^2+lh/4 h^4+ls/4*s^4+l/4*s^2 h^2;
         ch = (2 mW^2 + mZ^2 + mh^2 + 2 mt^2)/(4v^2) + l/24;
         cs = (2 l + 3 ls)/12;
         uh2[T_] := mh^2/(2v^2) vh2[T];
         us2[T_] := -ls*vs2[T];
         vh2[T_] := v^2(1 - 2 ch/mh^2 T^2);
         vs2[T_] := ((mh/(2 v) vh2[TC]*Sqrt[2 ls] + cs*TC^2) - cs*T^2)/ls;
```

```
In[2]:=  {mW, mZ, mh, v, mt} = {80.4, 91.2, 125.1, 246.2, 173.2}; (*GeV*)
         {TC, l, ls, lh} = {110(*GeV*), 1.5, 0.65, mh^2/(2 v^2)};
```

```
In[3]:=  (*S3/T at finite temperature*)
         BT[T_?NumericQ,fp_] := 1/T*FindBounce[
          V[h, s, T], {h, s},{{Sqrt@vh2[T], 0},{0,Sqrt@vs2[T]}},
          "Dimension" -> 3, "FieldPoints"-> fp ]["Action"];
```

```
In[4]:=  LogPlot[{BT[T,10],BT[T,31], 140}, {T, 70, 100},
            AxesLabel -> {"T(GeV)", "S3/T"}, PerformanceGoal -> "Speed"]
```

Out[4]=
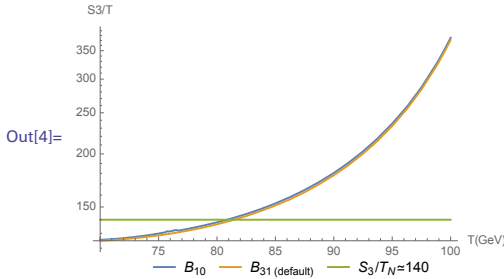

```
In[5]:=  Round[T /.FindRoot[BT[T,10] == 140,{T,100}]]
```

Out[5]= 81

The nucleation temperature we get is approximately 81 GeV, where $B_{10}(B_{31})$ is the normalized action, obtained with 10(31) field points.

Clearly, one can improve the precision by increasing the number of field points ``FieldPoints'' and action tolerance with ``ActionTolerance''. However, it should be kept in mind that one should also consistently consider quantum corrections, daisy resummation terms, the pre-factor A in (1) and contributions from lighter fermions, among others. Taking the above leading contributions only, FindBounce with only a few field points (10) already gives a reasonably accurate estimate, while being computationally inexpensive.

5.7. Beyond two fields

We devote this section to the estimates of the computational time of FindBounce with an arbitrary number of fields. We start with a simple example and show the Mathematica code to obtain the minima and compute the bounce configurations. Our main result is that the time requirement increases linearly with respect to the number of fields and that the calculation with 10 fields takes ~ 1 second. We compare the results to other available packages in the literature and test FindBounce with up to 20 fields.

From [35], we consider the multi-field potential as a function of the number of fields,

$$V(\varphi) = \left(\sum_{i=1}^{n_\varphi} c_i \, (\varphi_i - 1)^2 - c_{n_\varphi + 1} \right) \sum_{i=1}^{n_\varphi} \varphi_i^2, \tag{30}$$

where c_i take values between 0 to 1. The position of the minima is a point in field space with components close to 0 and 1. Given the number of fields $n_\varphi = 6$ and a constant set of random parameters c_i, the code to compute the bounce is:

```
In[1]:=  nf = 6;
         SeedRandom[1];
         c = RandomReal[1, nf + 1];
         phi = Table[Symbol["phi" <> ToString[i]], {i, nf}];
```

```
In[2]:=  V[phi_] := (Sum[c[[i]](phi[[i]]-1)^2,{i,nf}]-c[[-1]])Sum[phi[[i]]^2,{i,nf}];
```

```
In[3]:= extrema = Table[FindRoot[D[V[phi] == 0, {phi}],
          Transpose@{phi, ConstantArray[phi0, nf]}], {phi0,0,1}];
```

```
In[4]:= {minima, d2V} = {phi /. extrema, D[V[phi],{phi},{phi}] /. extrema};
        ei = Table[DeleteDuplicates@Sign[Eigenvalues@d2V[[i]]], {i,2}];
        typeV = Table[If[Length@ei[[i]] > 1,
          "Saddle", If[ei[[i, 1]] > 0, "Minimum", "Maximum"]], {i,2}]
```
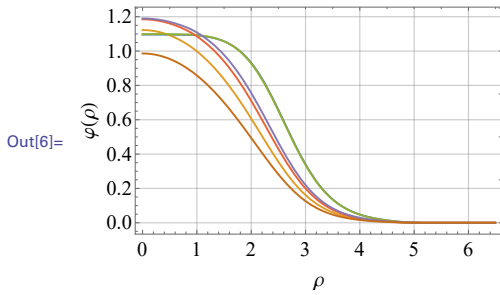
```
Out[4]= {Minimum,Minimum}
```

```
In[5]:= bf = FindBounce[V[phi], phi, minima]
```

Out[5]= BounceFunction[[+] Action: 202.
                             Dimension: 4]

```
In[6]:= BouncePlot[ bf, PlotLegends -> LineLegend[Automatic, LegendLabel-> "Fields"]]
```

Out[6]=

Since c_i were chosen at random, FindRoot might not find the minimum of the potential but a saddle point instead. In such case FindBounce returns $Failed. For more general potentials, the extrema can be computed with Vevacious [109], which is a tool that finds all the tree-level extrema of a generic one-loop effective potential.

With the multi-field potential defined in (30), we are ready to compute the action, estimate the computation time and compare to the existing tools. We vary the number of fields from the single field case going up to twenty fields. The results are collected in Table 1 and on the left of Fig. 9, together with results obtained by other methods. The parameters for $n_\varphi \leq 8$ were taken from [35], while for $n_\varphi > 8$ they were chosen randomly, as listed in Appendix B. The comparison includes the FindBounce with 10, 30 and 100 field points, CosmoTransition(CT) [29], AnyBubble(AB) [33], BubbleProfile(BP) [35] and SimpleBounce(SB) [110].

We find that the action computed with $N \lesssim 20$ field points is accurate up to roughly 1%, as shown on the left panel of Fig. 9. Similarly, bounces with ``FieldPoints''-> 10,30 in Table 1 are accurate to 1% level or below. Table 1 also shows how the accuracy of the action improves with the number of field points. In particular, the $N = 100$ case reproduces the values of the action obtained by other methods. Therefore, it is clear that one can use FindBounce with any type of potential to get the arbitrarily precise bounce action by adjusting the ``FieldPoints'' and ``ActionTolerance'' settings.

Note that the parameters c_i that regulate the multi-field potential for $n_\varphi \leq 8$ were chosen in a way that the bounce solution belongs to the thick wall regime. This is well suited for numerical shooting methods and typically gives the action of $\mathcal{O}(10)$, as seen in Table 1. For larger number of fields $n_\varphi > 8$, the parameters were chosen at random, see Table B.2 and the resulting action in Table B.3 contains both small and large values, in particular $n_\varphi = 15, 19$ that belong to the thin wall regime. Clearly, FindBounce can deal with both cases quite efficiently.

Let us turn to the timing performance of FindBounce. As explained in the PB method section Section 2.1, we expect this increase to be linear. Indeed, as shown on the right panel of Fig. 9 in solid lines, the time consumption of FindBounce scales linearly with the number of fields n_φ. This behavior is independent of the number of field points, i.e. for different colors of the solid lines in Fig. 9 and covers both thin and thick wall cases. Moreover, the time demand of FindBounce with respect to the number of field points also scales linearly in the multi-field scenarios.

A reliable timing comparison to other tools could be somewhat involved. The running time clearly depends on the CPU capabilities and the optimization efficiency of the implementation in a given computer language. In our setup, the computation time of FindBounce for 10(20) fields turns out to be less than 1(2) seconds for the default value with 31 field points. This was computed on a desktop system using the Mathematica native functions AboluteTiming and RepeatedTiming running on a desktop iMac 10.12.6, equipped with an Intel Core i7, processor speed 3.4 GHz and 16 GB of DDR3 RAM clocked at 1.6 GHz.

For the other implementations, the timing reports were adopted from [35] and [110]. We find FindBounce to be comparable to these tools in terms of speed performance, as shown on the right side of Fig. 9. In particular, the time demand with $N = 10$ field points is comparable to the values quoted by SimpleBounce and CosmoTransition. It should be noted though that the above tools are implemented in C++ and Python, while FindBounce was coded straightforwardly in Mathematica, with no significant numerical optimization.
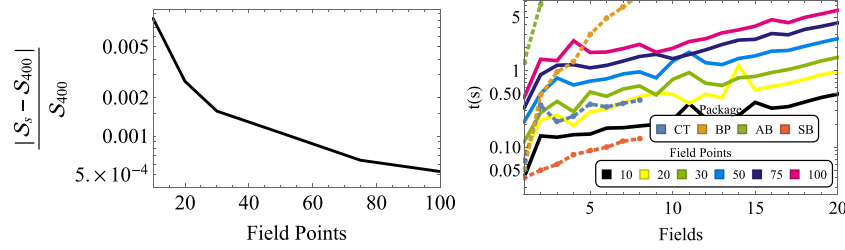
Fig. 9. Left: The multi-field bounce action S_s average, normalized to the one with $N = 400$ field points. Right: Multi-field time measure with respect to number of fields for several field points. The solid lines were obtained by `FindBounce` with ``ActionTolerance'' $\rightarrow 10^{-4}$. The dashed lines with dots are the time measurements from other existing tools, see text for details.

Table 1
Comparison of the bounce action obtained by different methods in the literature for various number of fields n_φ. It includes `FindBounce` with 10, 30 and 100 field points and `CosmoTransition`, `AnyBubble`, `BubbleProfile` and `SimpleBounce`. The action values for the other methods were adopted from [35,110].

Action

n_φ	FB_{10}	FB_{30}	FB_{100}	CT	AB	BP	SB
1	52.1	52.6	52.4	52.6	52.4	54.1	52.4
2	20.8	20.8	20.8	21.1	20.8	20.8	20.8
3	20.8	20.7	20.7	22.0	22.0	22.0	22.0
4	57.9	56.2	55.8	55.9	56.4	55.9	55.8
5	16.3	16.3	16.3	16.3	16.3	16.3	16.4
6	24.6	24.5	24.5	24.5	24.5	24.4	24.5
7	36.9	36.7	36.7	36.7	36.6	36.7	36.7
8	46.4	46.1	46.0	46.1	46.0	46.0	46.0

Table B.2
Multi-field potential parameters c_i that define the potential in (30).

n_φ	c_i
3	0.68, 0.18, 0.30, 0.28
4	0.53, 0.77, 0.84, 0.01, 0.26
5	0.47, 0.23, 0.57, 0.14, 0.52, 0.66
6	0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
7	0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
8	0.24, 0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
9	0.21, 0.24, 0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
10	0.12, 0.21, 0.24, 0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
11	0.23, 0.21, 0.21, 0.24, 0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
12	0.12, 0.11, 0.12, 0.21, 0.24, 0.52, 0.34, 0.47, 0.23, 0.57, 0.14, 0.52, 0.66
13	0.54, 0.47, 0.53, 0.28, 0.35, 0.27, 0.42, 0.59, 0.33, 0.16, 0.38, 0.35, 0.17, 0.41
14	0.39, 0.23, 0.26, 0.40, 0.11, 0.42, 0.41, 0.27, 0.42, 0.54, 0.18, 0.59, 0.13, 0.29, 0.58
15	0.21, 0.22, 0.22, 0.23, 0.39, 0.55, 0.43, 0.12, 0.16, 0.58, 0.25, 0.50, 0.45, 0.35, 0.45, 0.12
16	0.42, 0.34, 0.43, 0.22, 0.59, 0.41, 0.58, 0.41, 0.26, 0.45, 0.16, 0.31, 0.39, 0.57, 0.43, 0.10, 0.46
17	0.24, 0.35, 0.39, 0.56, 0.37, 0.41, 0.52, 0.31, 0.52, 0.22, 0.58, 0.39, 0.39, 0.17, 0.46, 0.30, 0.37, 0.43
18	0.18, 0.17, 0.30, 0.22, 0.38, 0.48, 0.11, 0.49, 0.43, 0.47, 0.21, 0.29, 0.32, 0.36, 0.30, 0.56, 0.46, 0.42, 0.44
19	0.40, 0.14, 0.10, 0.43, 0.39, 0.27, 0.33, 0.59, 0.48, 0.36, 0.24, 0.28, 0.51, 0.59, 0.40, 0.39, 0.24, 0.35, 0.20, 0.14
20	0.42, 0.11, 0.47, 0.13, 0.16, 0.24, 0.58, 0.53, 0.38, 0.44, 0.18, 0.46, 0.47, 0.27, 0.53, 0.24, 0.33, 0.40, 0.32, 0.29, 0.44

6. Conclusions and outlook

The `FindBounce` package presented in this work performs the task of calculating the semi-classical contribution to the false vacuum decay rate, the so-called bounce field configuration and the associated Euclidean action. We demonstrated the basic use of the `FindBounce` function implemented in `Mathematica`, as well as the extended options and manipulation of the output. The current version of the package can deal efficiently with single and multi-field calculations, ranging up to 20 fields in a matter of seconds.

There are a number of physically relevant directions one can pursue, that may be implemented in the future versions of the `FindBounce` framework. An obvious question relates to the growth of the bubble after nucleation, which is governed by the bounce equations in Minkowski space–time [2]. An initial step in this direction was done in [41] and may be generalized straightforwardly to more segments and fields. Moreover, matching such solutions to subsequent tunneling may be needed when potentials with multiple minima are considered. We leave this fascinating subject for future studies [79].

Another ubiquitous phenomenon is the tunneling without barriers into an unstable region of the potential. In this version of the `FindBounce` package, we demonstrate how this is done for single field theories by matching the polygonal potential onto an unstable quartic potential. Extension to multi-field case requires additional care and we leave it for an updated version.

The total decay rate in (1) depends on both the bounce and the pre-factor A [3], which is in general not easily calculable. Numerical approaches were done in $D = 3$ [111,112] and for $D = 4$ [62,113], for recent developments in treating gauge fields, see also [84,85,114]. The use of the polygonal method in computing the pre-factor was demonstrated in [42], where the semi-analytical bounce background was used for numerical computation of the fluctuation operator eigenfunctions, while the WKB integrals could be performed analytically. It may be feasible to revisit the calculation of the pre-factor ab-initio by considering the polygonal (or its second

Table B.3

The bounce action of the single and multi-field potential in Eq. (B.1) and (30), respectively computed using `FindBounce` with 10, 30 and 100 number of field points.

Action

n_φ	PB_{10}	PB_{30}	PB_{100}	n_φ	PB_{10}	PB_{30}	PB_{100}
1	52.1	52.6	52.4	11	78.3	78.3	78.2
2	20.8	20.8	20.8	12	80.2	80.0	79.0
3	20.8	20.7	20.7	13	274	271	271
4	57.9	56.2	55.8	14	154	155	155
5	16.3	16.3	16.3	15	2.90×10^3	2.87×10^3	2.87×10^3
6	24.6	24.5	24.5	16	358	355	355
7	36.9	36.7	36.7	17	472	468	468
8	46.4	46.1	46.0	18	439	435	435
9	56.1	55.7	55.6	19	3.96×10^3	3.93×10^3	3.93×10^3
10	63.8	63.4	63.3	20	565	560	560

order extension) potential in a more direct way by calculating the eigenfunctions of the second derivative of $V(\varphi)$. This would offer an extension beyond the thin wall case [115], an arbitrary precision with more segments and, perhaps most interestingly, to multiple fields.

The semi-analytical approach to computing the bounce field configuration and the action at finite temperature is relevant for characterizing the strength of the potential phase transitions. In particular, we saw how one can determine the nucleation temperature T_N, that happens below the critical temperature, using the output from the `FindBounce` result. Similarly, one can get an analytical insight in the gravitational wave spectra by computing the α and β parameters [49], which are related to the strength and the position of the maximum in the frequency range. We intend to return to these issues in future installments of the `FindBounce` approach.

CRediT authorship contribution statement

Victor Guada: Conceptualization, Methodology, Software, Writing - original draft. **Miha Nemevšek:** Conceptualization, Writing - original draft, Writing - review & editing, Supervision. **Matevž Pintar:** Methodology, Software, Visualization, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Appendix A. One-loop effective potential

The calculation of the effective potential was first computed by Coleman and Weinberg [116] at one-loop and at higher loops by Jackiw and Iliopoulus, et al. [117,118] at zero temperature. For finite temperatures excellent references are [102,103], a review [104] and the textbook [105]. For recent works on the consistent use of potentials, see [119]. The quantum and thermal results at one-loop order in the $\overline{\text{MS}}$ scheme are:

$$\text{Quantum: } \Delta V_1(\varphi) = \sum_i \pm \frac{n_i m_i^4(\varphi)}{64\pi^2} \left(\log \frac{m_i^2(\varphi)}{\mu_R^2} - C_i \right), \tag{A.1}$$

$$\text{Thermal: } \Delta V_1(\varphi, T) = \sum_i \frac{n_i T^4}{2\pi^2} J_{B/F} \left(\frac{m_i^2(\varphi)}{T^2} \right), \tag{A.2}$$

where μ_R is the renormalization scale and $C_i = 3/2\,(5/6)$ for scalars and fermions (gauge bosons). The sum runs over all the species that couple to φ, where n_i and $m_i^2(\varphi)$ are the number of degrees of freedom and the field-dependent squared masses of the species i. The upper and lower signs correspond to bosons (B) and fermions (F), respectively. The relevant thermal functions are given by

$$J_{B/F}\left(y^2\right) = \mp \int_0^\infty dx\, x^2 \log\left(1 \pm e^{-\sqrt{x^2+y^2}}\right), \tag{A.3}$$

and can be expanded at high temperatures, $\left(y^2 \ll 1\right)$ such that

$$J_B(y^2) = -\frac{7\pi}{360} + \frac{\pi^2}{24}y^2 + \frac{y^4}{32}\log\frac{y^2}{a} + \mathcal{O}\left(y^6\right), \tag{A.4}$$

$$J_F(y^2) = -\frac{\pi^6}{45} + \frac{\pi^2}{12}y^2 - \frac{\pi}{6}\left(y^2\right)^{\frac{3}{2}} - \frac{y^4}{32}\log\frac{y^2}{16a} + \mathcal{O}\left(y^6\right),$$ (A.5)

with $a = \pi^2 e^{\frac{3}{2} - 2\gamma_E}$.

Appendix B. Multi field potential parameters

The multi-field potential for comparison with other packages, shown in Table 1, was taken from [27] and defined for each number of fields n_φ as

$$V(\varphi) = \frac{1}{10}\left(\varphi^4 - 8\varphi^3 + 10\varphi^2 + 8\right),$$ (B.1)

for single field potentials and by Eq. (30) for multifields. The relevant parameters c_i for higher number of fields are given by the list in Table B.2 with the corresponding actions listed in Table B.3. Note that the last component of the parameters $c_{n_\varphi+1}$ controls the degeneracy of the vacua. Tiny values of this parameter correspond to thin wall scenarios, which is the case for the potentials with 15 and 19 fields. There, the action is notably larger. Nevertheless, as shown in Fig. 9, the computational time of the bounce action is practically independent of whether the bounce is computed in thin or thick wall regime.

References

[1] I.Y. Kobzarev, L.B. Okun, M.B. Voloshin, Sov. J. Nucl. Phys. 20 (1975) 644; Yad. Fiz. 20 (1974) 1229.
[2] S.R. Coleman, Phys. Rev. D 15 (1977) 2929; Erratum, Phys. Rev. D 16 (1977) 1248.
[3] C.G. Callan Jr., S.R. Coleman, Phys. Rev. D 16 (1977) 1762.
[4] A.D. Linde, Phys. Lett. 100B (1981) 37; Nucl. Phys. B 216 (1983) 421; Erratum, Nucl. Phys. B 223 (1983) 544.
[5] S.R. Coleman, Cambridge University Press, 1985.
[6] S. Fubini, Nuovo Cimento A 34 (1976) 521.
[7] L. Lipatov, Sov. Phys.—JETP 45 (1977) 216–223, LENINGRAD-76-255.
[8] F. Loran, Modern Phys. Lett. A 22 (2007) 2217–2235, arXiv:hep-th/0612089 [hep-th].
[9] K.M. Lee, E.J. Weinberg, Nuclear Phys. B 267 (1986) 181.
[10] A. Ferraz de Camargo, R.C. Shellard, G.C. Marques, Phys. Rev. D 29 (1984) 1147.
[11] A. Aravind, B.S. DiNunno, D. Lorshbough, S. Paban, Phys. Rev. D 91 (2) (2015) 025026, arXiv:1412.3160 [hep-th].
[12] F.C. Adams, Phys. Rev. D 48 (1993) 2800, hep-ph/9302321.
[13] U. Sarid, Phys. Rev. D 58 (1998) 085017, hep-ph/9804308.
[14] I. Dasgupta, Phys. Lett. B 394 (1997) 116, hep-ph/9610403.
[15] A. Aravind, D. Lorshbough, S. Paban, Phys. Rev. D 89 (10) (2014) 103535, arXiv:1401.1230 [hep-th].
[16] R. Sato, M. Takimoto, Phys. Rev. Lett. 120 (9) (2018) 091802, arXiv:1707.01099 [hep-ph].
[17] A.R. Brown, Phys. Rev. D 97 (10) (2018) 105002, arXiv:1711.07712 [hep-th].
[18] J.R. Espinosa, J. Cosmol. Astropart. Phys. 1807 (2018) 036, arXiv:1805.03680 [hep-th].
[19] R. Jinno, arXiv:1805.12153 [hep-th].
[20] A. Andreassen, D. Farhi, W. Frost, M.D. Schwartz, Phys. Rev. Lett. 117 (23) (2016) 231601, arXiv:1602.01102 [hep-th].
[21] W.Y. Ai, B. Garbrecht, C. Tamarit, J. High Energy Phys. 12 (2019) 095, arXiv:1905.04236 [hep-th].
[22] M.P. Hertzberg, M. Yamada, Phys. Rev. D 100 (1) (2019) 016011, arXiv:1904.08565 [hep-th].
[23] J. Braden, M.C. Johnson, H.V. Peiris, A. Pontzen, S. Weinfurtner, Phys. Rev. Lett. 123 (3) (2019) 031601, arXiv:1806.06069 [hep-th].
[24] A. Kusenko, Phys. Lett. B 358 (1995) 51, hep-ph/9504418.
[25] S. Chigusa, T. Moroi, Y. Shoji, Phys. Lett. B 800 (2020) 135115, arXiv:1906.10829 [hep-ph].
[26] R. Sato, Phys. Rev. D 101 (1) (2020) 016012, arXiv:1907.02417 [hep-ph].
[27] P. John, Phys. Lett. B 452 (1999) 221, hep-ph/9810499.
[28] J.M. Cline, G.D. Moore, G. Servant, Phys. Rev. D 60 (1999) 105035, hep-ph/9902220.
[29] C.L. Wainwright, Comput. Phys. Comm. 183 (2012) 2006, arXiv:1109.4189 [hep-ph].
[30] T. Konstandin, S.J. Huber, J. Cosmol. Astropart. Phys. 0606 (2006) 021, hep-ph/0603081.
[31] J.h. Park, J. Cosmol. Astropart. Phys. 1102 (2011) 023, arXiv:1011.4936 [hep-th].
[32] S. Akula, C. Balázs, G.A. White, Eur. Phys. J. C 76 (12) (2016) 681, arXiv:1608.00008 [hep-ph].
[33] A. Masoumi, K.D. Olum, B. Shlaer, J. Cosmol. Astropart. Phys. 1701 (01) (2017) 051, arXiv:1610.06594 [gr-qc].
[34] J.R. Espinosa, T. Konstandin, J. Cosmol. Astropart. Phys. 1901 (2019) 051, arXiv:1811.09185 [hep-th].
[35] P. Athron, C. Balázs, M. Bardsley, A. Fowlie, D. Harries, G. White, Comput. Phys. Comm. 244 (2019) 448, arXiv:1901.03714 [hep-ph].
[36] M.L. Piscopo, M. Spannowsky, P. Waite, Phys. Rev. D 100 (1) (2019) 016002, arXiv:1902.05563 [hep-ph].
[37] M.J. Duncan, L.G. Jensen, Phys. Lett. B 291 (1992) 109.
[38] K. Dutta, C. Hector, T. Konstandin, P.M. Vaudrevange, A. Westphal, Phys. Rev. D 86 (2012) 123517, arXiv:1202.2721 [hep-th].
[39] A. Masoumi, K.D. Olum, J.M. Wachter, J. Cosmol. Astropart. Phys. 1710 (10) (2017) 022, arXiv:1702.00356 [gr-qc].
[40] K. Dutta, C. Hector, P.M. Vaudrevange, A. Westphal, Phys. Lett. B 708 (2012) 309, arXiv:1110.2380 [hep-th].
[41] G. Pastras, J. High Energy Phys. 1308 (2013) 075, arXiv:1102.4567 [hep-th].
[42] V. Guada, A. Maiezza, M. Nemevšek, Phys. Rev. D 99 (5) (2019) 056020, arXiv:1803.02227 [hep-th].
[43] E. Witten, Phys. Rev. D 30 (1984) 272.
[44] C.J. Hogan, Mon. Not. R. Astron. Soc. 218 (1986) 629.
[45] A. Kosowsky, M.S. Turner, R. Watkins, Phys. Rev. Lett. 69 (1992) 2026; Phys. Rev. D 45 (1992) 4514.
[46] C. Grojean, G. Servant, Phys. Rev. D 75 (2007) 043507, hep-ph/0607107.
[47] M. Hindmarsh, S.J. Huber, K. Rummukainen, D.J. Weir, Phys. Rev. Lett. 112 (2014) 041301, arXiv:1304.2433 [hep-ph]; Phys. Rev. D 92 (12) (2015) 123009, arXiv:1504.03291 [astro-ph.CO]; Phys. Rev. D 96 (10) (2017) 103520, arXiv:1704.05871 [astro-ph.CO].
[48] D. Cutting, M. Hindmarsh, D.J. Weir, Phys. Rev. D 97 (12) (2018) 123513, arXiv:1802.05712 [astro-ph.CO] and arXiv:1906.00480 [hep-ph].
[49] C. Caprini, et al., J. Cosmol. Astropart. Phys. 1604 (2016) 001, arXiv:1512.06239 [astro-ph.CO]; J. Cosmol. Astropart. Phys. 03 (2020) 024, arXiv:1910.13125 [astro-ph.CO].
[50] A.I. Bochkarev, S.V. Kuzmin, M.E. Shaposhnikov, Phys. Lett. B 244 (1990) 275.
[51] A.G. Cohen, D.B. Kaplan, A.E. Nelson, Phys. Lett. B 245 (1990) 561; Nucl. Phys. B 349 (1991) 727.
[52] N. Turok, J. Zadrozny, Nuclear Phys. B 358 (1991) 471; Phys. Rev. Lett. 65 (1990) 2331.
[53] D.E. Morrissey, M.J. Ramsey-Musolf, New J. Phys. 14 (2012) 125003, arXiv:1206.2942 [hep-ph].
[54] T. Vachaspati, Phys. Lett. B 265 (1991) 258.
[55] G. Sigl, A.V. Olinto, K. Jedamzik, Phys. Rev. D 55 (1997) 4582, astro-ph/9610201.

[56] A. De Simone, G. Nardini, M. Quiros, A. Riotto, J. Cosmol. Astropart. Phys. 1110 (2011) 030, arXiv:1107.4317 [hep-ph].
[57] A.G. Tevzadze, L. Kisslinger, A. Brandenburg, T. Kahniashvili, Astrophys. J. 759 (2012) 54, arXiv:1207.0751 [astro-ph.CO].
[58] J. Ellis, M. Fairbairn, M. Lewicki, V. Vaskonen, A. Wickens, J. Cosmol. Astropart. Phys. 1909 (09) (2019) 019, arXiv:1907.04315 [astro-ph.CO].
[59] S.R. Coleman, V. Glaser, A. Martin, Comm. Math. Phys. 58 (1978) 211.
[60] K. Blum, M. Honda, R. Sato, M. Takimoto, K. Tobioka, J. High Energy Phys. 1705 (2017) 109;   Erratum, J. High Energy Phys. 1706 (2017) 060, arXiv:1611.04570 [hep-th].
[61] G.H. Derrick, J. Math. Phys. 5 (1964) 1252.
[62] G.V. Dunne, H. Min, Phys. Rev. D 72 (2005) 125004, hep-th/0511156.
[63] S.R. Coleman, F. De Luccia, Phys. Rev. D 21 (1980) 3305.
[64] P.W. Graham, D.E. Kaplan, S. Rajendran, Phys. Rev. Lett. 115 (22) (2015) 221801, arXiv:1504.07551 [hep-ph].
[65] J. Espinosa, C. Grojean, G. Panico, A. Pomarol, O. Pujolàs, G. Servant, Phys. Rev. Lett. 115 (25) (2015) 251803, arXiv:1506.09217 [hep-ph].
[66] E. Hardy, J. High Energy Phys. 11 (2015) 077, arXiv:1507.07525 [hep-ph].
[67] S.P. Patil, P. Schwaller, J. High Energy Phys. 02 (2016) 077, arXiv:1507.08649 [hep-ph].
[68] J. Jaeckel, V.M. Mehta, L.T. Witkowski, Phys. Rev. D 93 (6) (2016) 063522, arXiv:1508.03321 [hep-ph].
[69] R.S. Gupta, Z. Komargodski, G. Perez, L. Ubaldi, J. High Energy Phys. 02 (2016) 166, arXiv:1509.00047 [hep-ph].
[70] B. Batell, G.F. Giudice, M. McCullough, J. High Energy Phys. 12 (2015) 162, arXiv:1509.00834 [hep-ph].
[71] J. Jaeckel, V.M. Mehta, L.T. Witkowski, J. Cosmol. Astropart. Phys. 01 (2017) 036, arXiv:1605.01367 [hep-ph].
[72] N. Fonseca, E. Morgante, R. Sato, G. Servant, arXiv:1911.08473 [hep-ph].
[73] A.R. Brown, A. Dahlen, Phys. Rev. D 84 (2011) 105004, arXiv:1106.0527 [hep-th].
[74] L. Darmé, J. Jaeckel, M. Lewicki, Phys. Rev. D 96 (5) (2017) 056001, arXiv:1704.06445 [hep-ph].
[75] S. Sarangi, G. Shiu, B. Shlaer, Internat. J. Modern Phys. A 24 (2009) 741, arXiv:0708.4375 [hep-th].
[76] P.M. Saffin, A. Padilla, E.J. Copeland, J. High Energy Phys. 0809 (2008) 055, arXiv:0804.3801 [hep-th].
[77] S.-H.H. Tye, D. Wohns, arXiv:0910.1088 [hep-th].
[78] L. Darmé, J. Jaeckel, M. Lewicki, Phys. Rev. D 100 (9) (2019) 096012, arXiv:1907.04865 [hep-th].
[79] V. Guada, M. Nemevšek, L. Ubaldi, work in progress.
[80] G. Isidori, V.S. Rychkov, A. Strumia, N. Tetradis, Phys. Rev. D 77 (2008) 025034, arXiv:0712.0242 [hep-ph];
     A. Salvio, A. Strumia, N. Tetradis, A. Urbano, J. High Energy Phys. 1609 (2016) 054, arXiv:1608.02555 [hep-ph].
[81] G. Degrassi, S. Di Vita, J. Elias-Miro, J.R. Espinosa, G.F. Giudice, G. Isidori, A. Strumia, J. High Energy Phys. 1208 (2012) 098, arXiv:1205.6497 [hep-ph].
[82] D. Buttazzo, G. Degrassi, P.P. Giardino, G.F. Giudice, F. Sala, A. Salvio, A. Strumia, J. High Energy Phys. 1312 (2013) 089, arXiv:1307.3536 [hep-ph].
[83] S. Chigusa, T. Moroi, Y. Shoji, Phys. Rev. Lett. 119 (21) (2017) 211801, arXiv:1707.09301 [hep-ph].
[84] A. Andreassen, W. Frost, M.D. Schwartz, Phys. Rev. D 97 (5) (2018) 056006, arXiv:1707.08124 [hep-ph].
[85] S. Chigusa, T. Moroi, Y. Shoji, Phys. Rev. D 97 (11) (2018) 116012, arXiv:1803.03902 [hep-ph].
[86] J.R. Espinosa, Phys. Rev. D 100 (10) (2019) 105002, arXiv:1908.01730 [hep-th].
[87] G. Branco, P. Ferreira, L. Lavoura, M. Rebelo, M. Sher, J.P. Silva, Phys. Rep. 516 (2012) 1–102, arXiv:1106.0034 [hep-ph].
[88] S.J. Huber, M.G. Schmidt, Nuclear Phys. B 606 (2001) 183, hep-ph/0003122.
[89] S.J. Huber, T. Konstandin, T. Prokopec, M.G. Schmidt, Nuclear Phys. B 757 (2006) 172, hep-ph/0606298.
[90] H.H. Patel, M.J. Ramsey-Musolf, J. High Energy Phys. 1107 (2011) 029, arXiv:1101.4665 [hep-ph].
[91] J.M. Cline, K. Kainulainen, J. Cosmol. Astropart. Phys. 1301 (2013) 012, arXiv:1210.4196 [hep-ph].
[92] M. Chala, G. Nardini, I. Sobolev, Phys. Rev. D (5) 94 (2016) 055006, arXiv:1605.08663 [hep-ph].
[93] C.P. Burgess, M. Pospelov, T. ter Veldhuis, Nuclear Phys. B 619 (2001) 709, hep-ph/0011335.
[94] J. McDonald, Phys. Rev. Lett. 88 (2002) 091304, hep-ph/0106249.
[95] M. Gonderinger, Y. Li, H. Patel, M.J. Ramsey-Musolf, J. High Energy Phys. 1001 (2010) 053, arXiv:0910.3167 [hep-ph].
[96] J.M. Cline, K. Kainulainen, P. Scott, C. Weniger, Phys. Rev. D 88 (2013) 055025;   Erratum, Phys. Rev. D 92 (3) (2015) 039906, arXiv:1306.4710 [hep-ph].
[97] C. Caprini, R. Durrer, G. Servant, J. Cosmol. Astropart. Phys. 0912 (2009) 024, arXiv:0909.0622 [astro-ph.CO].
[98] J.R. Espinosa, T. Konstandin, J.M. No, G. Servant, J. Cosmol. Astropart. Phys. 1006 (2010) 028, arXiv:1004.4187 [hep-ph].
[99] P. Binetruy, A. Bohe, C. Caprini, J.F. Dufaux, J. Cosmol. Astropart. Phys. 1206 (2012) 027, arXiv:1201.0983 [gr-qc].
[100] E. Thrane, J.D. Romano, Phys. Rev. D 88 (12) (2013) 124032, arXiv:1310.5300 [astro-ph.IM].
[101] J. Ellis, M. Lewicki, J.M. No, V. Vaskonen, J. Cosmol. Astropart. Phys. 1906 (2019) 024, arXiv:1903.09642 [hep-ph].
[102] L. Dolan, R. Jackiw, Phys. Rev. D 9 (1974) 3320.
[103] S. Weinberg, Phys. Rev. D 9 (1974) 3357.
[104] M. Quiros, hep-ph/9901312.
[105] J.I. Kapusta, C. Gale, http://dx.doi.org/10.1017/CBO9780511535130.
[106] A. Fowlie, Comput. Phys. Comm. 228 (2018) 264, arXiv:1802.02720 [hep-ph].
[107] J.R. Espinosa, T. Konstandin, F. Riva, Nuclear Phys. B 854 (2012) 592, arXiv:1107.5441 [hep-ph].
[108] G.W. Anderson, L.J. Hall, Phys. Rev. D 45 (1992) 2685.
[109] J.E. Camargo-Molina, B. O'Leary, W. Porod, F. Staub, Eur. Phys. J. C 73 (10) (2013) 2588, arXiv:1307.1477 [hep-ph].
[110] R. Sato, arXiv:1908.10868 [hep-ph].
[111] A. Strumia, N. Tetradis, Nuclear Phys. B 542 (1999) 719, hep-ph/9806453.
[112] G. Munster, S. Rotsch, Eur. Phys. J. C 12 (2000) 161, cond-mat/9908246.
[113] J. Baacke, G. Lavrelashvili, Phys. Rev. D 69 (2004) 025009, hep-th/0307202.
[114] A. Andreassen, D. Farhi, W. Frost, M.D. Schwartz, Phys. Rev. D 95 (8) (2017) 085011, arXiv:1604.06090 [hep-th].
[115] R.V. Konoplich, S.G. Rubin, Yad. Fiz. 42 (1985) 1282;
      R.V. Konoplich, Theor. Math. Phys. 73 (1987) 1286;   Teor. Mat. Fiz. 73 (1987) 379.
[116] S.R. Coleman, E.J. Weinberg, Phys. Rev. D 7 (1973) 1888.
[117] R. Jackiw, Phys. Rev. D 9 (1974) 1686.
[118] J. Iliopoulos, C. Itzykson, A. Martin, Rev. Modern Phys. 47 (1975) 165.
[119] A. Andreassen, W. Frost, M.D. Schwartz, Phys. Rev. D 91 (1) (2015) 016009, arXiv:1408.0287 [hep-ph];   Phys. Rev. Lett. 113 (24) (2014) 241801, arXiv:1408.0292 [hep-ph].