Numerical RG-time integration of the effective potential: Analysis and benchmark

Friederike Ihssen,[1] Franz R. Sattler,[1] and Nicolas Wink[2,*]

[1]*Institut für Theoretische Physik, Universität Heidelberg, Philosophenweg 16, D-69120 Heidelberg, Germany*

[2]*Institut für Kernphysik, Technische Universität Darmstadt, D-64289 Darmstadt, Germany*

(Received 13 March 2023; accepted 5 April 2023; published 7 June 2023)

We investigate the renormalization group (RG)-time integration of the effective potential in the functional renormalization group in the presence of spontaneous symmetry breaking and its subsequent convexity restoration on the example of a scalar theory in $d = 3$. The features of this setup are common to many physical models and our results are, therefore, directly applicable to a variety of situations. We provide exhaustive work-precision benchmarks and numerical stability analyses by considering the combination of different discrete formulations of the flow equation and a large collection of different algorithms. The results are explained by using the different components entering the RG-time integration process and the eigenvalue structure of the discrete system. Particularly, the combination of Rosenbrock methods, implicit multistep methods, or certain (diagonally) implicit Runge-Kutta methods with exact or automatic differentiation Jacobians proves to be very potent. Furthermore, a reformulation in a logarithmic variable circumvents issues related to the singularity bound in the flat regime of the potential.

DOI: 10.1103/PhysRevD.107.114009

I. INTRODUCTION

Phase transitions and critical phenomena are an integral part of describing physics in relativistic quantum theories, such as quantum chromodynamics, electroweak theory or condensed matter systems. The emergence of these phenomena can be best understood by following the scale dependence of the physics, starting from microscopic descriptions and coarse graining by integrating out fluctuations to understand and quantify collective phenomena at macroscopic scales. This process is encapsulated in renormalization group (RG) methods, which are thus a suitable choice for the study of criticality and scaling since they are adapted to the physics of scales.

One realization of such methods is the functional renormalization group (FRG), which constitutes a powerful method to deal in a nonperturbative manner with the scale integration of fluctuations. For a recent review see [1], wherein a detailed account of current applications is given. Its basic premise is to introduce a regularization scale k below which fluctuations are suppressed due to a momentum-dependent mass insertion, the regulator. This scale is then lowered, including successive momentum scales in a Wilsonian manner until one reaches $k \to 0$ in order to obtain the full theory with all fluctuations integrated out. Although the FRG is an exact method, it is necessary to use a truncation of the flow to turn the FRG flow equation into a numerically accessible tool.

Dealing with collective phenomena and phase transitions, one may employ a derivative expansion, i.e., a truncation up to some power of momenta in the one-particle irreducible (1PI) effective action $\Gamma_k[\phi]$. In spite of neglecting higher momentum dependencies, derivative expansions are usually a good choice to explore these systems in the FRG, as they allow one to focus on taking into account higher interaction processes, i.e., model the effective action for high orders in the fields, see, e.g., [2–5].

The resulting FRG equations are often remarkably similar for many different physical systems and feature common numerical difficulties, which enables one to explore often-appearing issues related to the RG-time integration within a minimal setup, whose basic features generalize to most more elaborate theories. In this work, we choose such a setup and discuss the process of spontaneous symmetry breaking in a simple scalar theory with an interaction potential that has a full field dependence without truncation.

To assess the nature of common numerical difficulties, we analyze the FRG flow of this model up to leading order in the derivative expansion, the local potential approximation. Furthermore, in such a setup, the flow equation

[*]nicolas.wink@tu-darmstadt.de

reduces to a single partial differential equation (PDE), though more elaborate approximations may feature a larger system of equations.

A majority of the obstacles in the numerical solution of flow equations can be explained by the presence of convexity restoration of the 1PI effective action in the regime of broken symmetry. Suppose one has an effective potential $V(\phi)$, where ϕ is the expectation value of a scalar field. The equation of motion of the 1PI effective action singles out the minimum ϕ_0 of $V(\phi)$ as the solution to the equation of motion and thus the physical point. Therefore, ϕ also serves as an order parameter and $\phi_0 = 0$ signifies the symmetric phase and $\phi_0 > 0$ the symmetry broken phase. In the symmetry broken phase in the vicinity of a second-order phase transition, the potential fulfills $V(|\phi| \leq \phi_0) =$ const due to the convexity of $\Gamma_{k\to 0}[\phi]$, which has to be restored during the FRG flow of the model. This region of the potential is called the flat part. At any finite RG-scale/regularization, however, the potential in the flat region has a finite slope that approaches zero only asymptotically, leading to technical difficulties in actual implementations.

This paper mostly deals with the numerical issues due to convexity restoration during the FRG flow of the effective potential. To this end, we apply a heuristically driven approach to investigate the issue of choosing an appropriate RG-time integration routine.

Starting from the method of lines, it has recently been worked out on a conceptual level how the effective potential in field direction should be discretized to achieve stability, see [6–15] for applications thereof. In essentially all previous works, the choice of algorithm to evolve the system in RG-time was rather arbitrary. Notable exceptions are [16], where the RG-time evolution was considered in a semianalytical approach, and [17], where a fully implicit spacetime evolution based on Chebyshev polynomials was used. The latter approach, however, is plagued by stability issues, which are especially pronounced for the case considered in this work.

Crucially, it should also be noted, that the use of explicit time stepper methods is impractical, since the wave speed in the flat region of the potential diverges exponentially fast for large RG-times [6]. Following the Courant-Friedrichs-Lewy (CFL) condition, which is necessary for both stability and validity of the results, leads to exponentially small step sizes. As any explicit method will suffer from such a bound, this issue can only be solved by using implicit methods. The aim of this paper is to start tackling the path to an informed choice and understanding of the pertinence of numerical time integrators by surveying and analyzing the performance of commonly used algorithms.

II. THE EFFECTIVE POTENTIAL AND ITS DISCRETIZATION

To keep matters simple, we look at the φ_3^4 theory, i.e., a single scalar field with $\mathbb{Z}_2$ symmetry, in the broken phase in three Euclidean spacetime dimensions. Working with the FRG, we are primarily aiming at the calculation of the scale-dependent effective potential; for details and references, see Appendix A. To leading order in the derivative expansion, the quantum effective action is given by

$$\Gamma_k[\phi] = \int_x \left\{ \frac{1}{2} (\partial_\mu \phi)^2 + V(t, \rho) \right\}, \tag{1}$$

where $\phi = \langle \varphi \rangle$ denotes the expectation value of the field, k is the RG-scale, and $t = -\log(k/\Lambda)$ is the RG-time relative to the initial UV scale Λ. Concerning the field dependence, the effective potential $V(t, \rho)$ must be a function of the $\mathbb{Z}_2$ invariant $\rho = \phi^2/2$ only. Finally, the integral subscript x denotes integration over the entire spacetime in the usual fashion.

Using the Litim regulator $R_k(p) = (k^2 - p^2)\Theta(k^2 - p^2)$, we obtain the flow of the effective potential by inserting the ansatz of the effective action (1) in Appendix A into the Wetterich equation (A1) and projecting on the effective potential

$$\partial_t V(t, \rho) = -A_d \frac{k^{d+2}}{k^2 + \partial_\rho V(t, \rho) + 2\rho \partial_\rho^2 V(t, \rho)}. \tag{2}$$

The prefactor $A_d = 2/d(2\pi)^{-d} \pi^{d/2} \Gamma(d/2)^{-1}$ in (2) collects prefactors from the Wetterich equation and the momentum integration. Its sign is positive and only depends on the spacetime dimension d. Throughout this work, we use $d = 3$, but note that the qualitative statements in the broken phase are independent of this choice for any $d > 2$.

Despite this simple truncation, numerical problems related to the RG-time integration are already present, and results generalize directly to other theories and more complicated settings.

Following the ideas of [6] and subsequent works, the fundamental variable is given by the derivative of the effective potential

$$u(t, \rho) = \partial_\rho V(t, \rho). \tag{3}$$

For simplicity, we will omit the arguments of functions if they are obvious in the current context. The equation for $u(t, \rho)$ is now conveniently formulated using the flux

$$f(t, m^2) = -A_d \frac{k^{d+2}}{k^2 + m^2}, \tag{4}$$

where the square of the (curvature) mass m^2 is given by

$$m^2(t, \rho) = u(t, \rho) + 2\rho \partial_\rho u(t, \rho). \tag{5}$$

The flow of the derivative of the effective potential $u(t, \rho)$ is then given by

$$\partial_t u = \partial_\rho f(t, m^2). \tag{6}$$

For future reference, we also introduce the flow equation of the squared curvature mass,

$$\partial_t m^2 = (1 + 2\rho\partial_\rho)\partial_\rho f(t, m^2). \tag{7}$$

Finally, we consider the logarithm of the regularized two-point function

$$\varpi(t, \rho) = \log(k^2 + m^2), \tag{8}$$

with the corresponding flow equation given by

$$\partial_t \varpi = -e^{-\varpi}[(1 + 2\rho\partial_\rho)\partial_\rho(A_d k^{d+2} e^{-\varpi}) - 2k^2]. \tag{9}$$

These flow equations are potentially interesting as they might be beneficial for numerical applications. The flow of the squared mass m^2 is closer in its form to typical advection-diffusion equations, while the log of the regularized two-point functions makes the constraint $k^2 + m^2 > 0$ for all finite $k > 0$ ($t < \infty$) manifest. Such logarithmic transformations are commonly employed in chemical and biological systems for precisely the same reason, as, e.g., chemical concentrations can become arbitrarily small, but must be positive. See, e.g., [18,19] and references therein for explicit examples where this is transformation is implemented. For the initial conditions considered in this work (quartic potentials), this constraint can be shown to hold for all RG-scales k, see [20] for a detailed discussion.

A. Discretization

To keep subsequent discussions simple, we employ a straightforward grid discretization of the field space and resort to a first-order finite difference scheme, see also [9].

We discretize the (finite) domain on a one-dimensional ordered grid,

$$\{\rho_i\} = \{\rho_1 = 0, \rho_2, \ldots, \rho_{N_\rho} = \rho_{\max}\}. \tag{10}$$

To shorten equations in the following, we introduce the shorthand notation $u_i = u(t, \rho_i)$, $m_i^2 = m^2(t, \rho_i)$, and $\varpi_i = \varpi(t, \rho_i)$.

The flux (4) is strictly negative. Hence, we can define the first-order up- and downwind derivative operators,

$$\mathcal{D}_{u(d)} a_i = \frac{a_{i\pm1} - a_i}{\rho_{i\pm1} - \rho_i}, \tag{11}$$

where a serves as placeholder for any variable of interest, the $+$ sign has to be taken for the upwind operator $\mathcal{D}_u$, and the $-$ sign is taken for the downwind operator $\mathcal{D}_d$.

We start by discretizing the standard formulation (6) and subsequently derive the discretization of the mass (7) and log (9) version. Following the ideas of [6], the derivative applied to the flux (4) in (6) has to be an upwind derivative, while the derivative in the term $(1 + 2\rho\partial_\rho)$ of (5) has to be a downwind derivative.

To summarize, the advection part of the equation is discretized with an upwind derivative, while the diffusive part is, taking the asymmetry of 2ρ into account, discretized with a central derivative.

With this at hand, we can easily write down stable discretizations of three different versions (6), (7), and (9). To that end, it is beneficial to introduce the combined derivative operator,

$$\tilde{\mathcal{D}} = (1 + 2\rho\mathcal{D}_d)\mathcal{D}_u, \tag{12}$$

which appears in two of the three equations.

In total, the discretized flow for the derivative of the effective potential (6) is given by

$$\partial_t u_i = \mathcal{D}_u f(t, (1 + 2\rho\mathcal{D}_d)u_i). \tag{13}$$

Utilizing the definition of the squared mass (5), we obtain the discretized version of (7),

$$\partial_t m_i^2 = \tilde{\mathcal{D}} f(t, m_i^2). \tag{14}$$

Finally, using (8), we obtain the discretized version of (9), the logarithm of the regularized two-point function,

$$\partial_t \varpi_i = -e^{-\varpi_i}[\tilde{\mathcal{D}}(A_d k^{d+2} e^{-\varpi_i}) + 2k^2]. \tag{15}$$

At this point, we would like to emphasize that the discretization of these formulations is equivalent. Therefore, the notion of stability underlying (13) implies the same stability for (14) and (15). This, however, only applies to the integration in RG-time if a strong stability preserving algorithm in combination with a CFL-type step size condition is used. Consequently, it does not tell us anything about the potential performance or stability of implicit algorithms with adaptive time stepping.

B. Initial and boundary conditions

Investigating the convexity restoration of the potential is easiest done with a quartic UV potential. Consequently, we fix the initial potential to be

$$V(t = 0, \rho) = m_\Lambda^2 \rho + \frac{\lambda_\Lambda}{2}\rho^2. \tag{16}$$

The coefficient λ_Λ has to be positive for the potential to be bounded from below. On the other hand, the coefficient m_Λ^2 is chosen to be negative, such that the flow starts in the symmetry broken phase. Only then can the final potential have a flat region, because the flow is symmetry restoring, see, e.g., [20]. We are only interested in this regime of the theory, since it is responsible for the numerical difficulties that we are investigating. The other branch of solutions is, from this perspective, trivial.

Throughout this paper, we omit units, since we do not identify the calculations presented here with a specific physical system. Furthermore, we employ the freedom to rescale the field to fix $\lambda_\Lambda = 1$.

We fix the initial UV scale to $\Lambda = 7.5$, which is sufficiently large to incorporate the full dynamics related to convexity restoration. Finally, to stay deeply within the symmetry broken regime in the limit of vanishing regularization ($t \to \infty$), we set $m_\Lambda^2 = -2.5$.

Since we are interested in the flattening of the potential, which is entirely dominated by the infrared part of the flow, we will not pay any attention to any potential UV issues, such as RG consistency, see [21] for more details thereon. This procedure is fully compatible with usual calculations in effective low-energy effective theories.

With these parameters, the minimum of the effective potential is initially at $\rho_0(t=0) = 2.5$ and freezes in at $\rho_0(t \to \infty) = 2.296$. To fully incorporate the associated dynamics, we chose $\rho_{\max} = 7.5$ and $N_\rho = 256$. These parameters correspond to realistic choices that we would use in associated studies of the phase structure of the theory, i.e., these are parameters of typical calculations.

Finally, we have to specify the boundary conditions to complete the description of the discretization. The left boundary of the computational domain is always located at $\rho = 0$, hence we need to specify the downwind derivative at this boundary. In the current setup, we do not expect any (nor can they appear) irregularities at $\rho = 0$, and the combination $2\rho\mathcal{D}_d$ can safely be set to zero. In certain cases, the situation might be more complicated, see [7] for a detailed discussion. At the right boundary $\rho_{\max}$, the flow is sufficiently suppressed and upwinding does not matter [6]. Hence, we can replace the upwind derivative at the boundary with a downwind derivative. In summary,

$$\begin{aligned} 2\rho\mathcal{D}_d &\to 0 \quad \text{at } \rho = 0, \\ \mathcal{D}_u &\to \mathcal{D}_d \quad \text{at } \rho = \rho_{\max}. \end{aligned} \tag{17}$$

The discussion above fully specifies the discretization of the field domain and results in a system of ordinary differential equations (ODEs).

C. Fermionic extension

In principle, the general statements of this work should hold, in general, since the mechanisms of convection dominated evolution are independent of the theory. In order to test if our analysis is indeed robust against variations in the flow equation, we extend the system for this subsection with an additional quarklike term and increase the number of scalar fields from $N = 1$ to $N = 4$. The ansatz for the effective action (1) changes to

$$\begin{aligned} \Gamma_k[\phi] = \int_x \bigg\{ &\bar\psi_{a,f} i\gamma_\mu\partial_\mu\psi_{a,f} + \frac{1}{2}(\partial_\mu\phi)^2 \\ &+ \frac{h}{\sqrt{2}}\bar\psi_{a,f}(i\sigma + \gamma_5\vec\tau\cdot\vec\pi)\psi_{a,f} + V(t,\rho)\bigg\}, \end{aligned} \tag{18}$$

where the quark fields $\psi_{a,f}$ carry color indices $a = 1, 2, 3$ and flavor indices $f = 1, 2$, while the Dirac indices are suppressed for simplicity. In the current context, the scalar field $\phi = (\sigma, \vec\pi)$ has four components, in contrast to the $\mathbb{Z}_2$ theory of the previous section with one component. The scalars and fermions are coupled through a Yukawa interaction h, where we used the Pauli matrices τ_i. For a detailed discussion of this model, called the quark-meson model, see, e.g., [22–24]. This extension is an oft-used low-energy effective theory of QCD, which allows one to make predictions about the phase structure of QCD in low-energy settings and thus constitutes another physically relevant model.

From the effective action (18), we get, with the Litim regulator, a similar flow equation as for the $\mathbb{Z}_2$ case,

$$\begin{aligned} \partial_t V(t,\rho) = -A_d k^{d+2}\bigg(&\frac{3}{k^2 + \partial_\rho V(t,\rho)} \\ &+ \frac{1}{k^2 + \partial_\rho V(t,\rho) + 2\rho\partial_\rho^2 V(t,\rho)} - \frac{24}{k^2 + h^2\rho}\bigg). \end{aligned} \tag{19}$$

Compared to (2), we get an additional term for the Goldstone degrees of freedom, similar to the one of the radial mode but purely advective, i.e., it contains no diffusion. Furthermore, there is a fermionic contribution, which acts effectively as a source term in the equation, see [9,10,13,15] for previous studies with fermions.

In the current quark-meson (QM) case, we follow the same logic as for the $\mathbb{Z}_2$ theory and work with the derivative of the effective potential (3). The discretization of the equation then proceeds in the exact same fashion as outlined in Sec. II A and leads to an equation almost identical to (13),

$$\partial_t u_i = \mathcal{D}_u f_{\mathrm{QM}}(t, u_i, (1 + 2\rho\mathcal{D}_d)u_i), \tag{20}$$

where the QM flux reads

$$\begin{aligned} f(t, m_\pi^2, m_\sigma^2) = -A_d k^{d+2}\bigg(&\frac{3}{k^2 + m_\pi^2} + \frac{1}{k^2 + m_\sigma^2} \\ &- \frac{24}{k^2 + h^2\rho}\bigg), \end{aligned} \tag{21}$$

where we defined the pion and sigma masses as

$$\begin{aligned} m_\pi^2(t,\rho) &= u(t,\rho), \\ m_\pi^2(t,\rho) &= u(t,\rho) + 2\rho\partial_\rho u(t,\rho). \end{aligned} \tag{22}$$

To keep matters simple, we will only consider the standard formulation (20), i.e., directly discretizing the derivative of the effective potential.

Similarly, as in the $\mathbb{Z}_2$ theory, we aim for a realistic setting, which also means choosing physically sensible initial conditions. In the current quark-meson model, the fermionic degrees of freedom, i.e., quarks, drive the symmetry breaking and the initial conditions are usually chosen to be in the symmetric phase. This can be achieved by setting $m_\Lambda = 0$ and leaving the remaining parameters untouched.

Furthermore, the Yukawa coupling is an additional parameter not present in the $\mathbb{Z}_2$ theory, which we fix to $h = 1.5$ in order to make the symmetry breaking scale comparable to the one in the $\mathbb{Z}_2$-theory case.

D. Typical solutions

Before diving into the analysis of different numerical schemes for the RG-time integration, it is helpful to discuss the qualitative features of the solution. We discuss the solution in the range $t \in [0, 6]$. At the final RG-time $t = 6$, the flow is effectively frozen and is almost indistinguishable from the $t \to \infty$ limit. This will be discussed further in Sec. IV B.

In the case of the standard formulation (13), the qualitative behavior is straightforward and is shown in Fig. 1. For small RG-times t, the linear shape of $u(t,\rho)$ stays roughly intact and gets shifted only slightly. At $t \approx 1.5$ the smallest values, located around $\rho = 0$, get comparable to the singularity bound $-k^2$, cf. (6), and the convexity restoration dynamics set in. This is particularly well visible if the RG-time evolution of the individual grid points is visualized, shown in the right panel of Fig. 1. At this point, the zero crossing, which determines the solution of the equation of motion, starts to freeze in exponentially fast. In the remaining RG-time evolution, the region at field values smaller than the zero crossing $\rho \le \rho_0(t)$ gets pushed by the singularity bound toward zero.

The challenging aspect in these calculations is usually that the solution of the equation of motion is located at the boundary of the flat region. This makes a precise resolution of the region around $\rho_{0,\mathrm{fin}} = \rho_0(t \to \infty)$ difficult, as the point of interest itself becomes nonanalytic and quantities, such as the mass, cf. (5), contain one-sided derivatives. The nonanalyticity, in the presence of the singularity bound in the equation, necessitates the use of a diffusive scheme, regularizing the nonanalyticity.

This problem is particularly well visible when considering the discretization in terms of the mass (14). The solution thereof is shown in Fig. 2. In this case, the nonanalyticity at $\rho_{0,\mathrm{fin}}$ turns into a jump, which becomes sharper and sharper as the RG-time becomes larger. From the right panel in Fig. 2, it is evident that there is one grid point inside the jump, reflecting its diffusive nature. Please note that this grid point survives the $t \to \infty$ limit and that the details of the diffusive smearing of the shock strongly depend on the chosen spatial discretization. Furthermore, this point is also contained in the solution of the standard formulation (13), being equivalent on the discrete level. In the mass formulation (14), it is also quite well visible that each grid point in the flat regime goes through a rapid change, underlining the difficult RG-time evolution.

Finally, we consider the log formulation (15) of the problem. In such a formulation, the singularity bound is not manifest in the equations, as $\varpi(t,\rho) = \log(k^2 + m^2) \in \mathbb{R}$, in contrast to $u(t,\rho) > -k^2$, solving the issue of the singularity bound. The price to pay is that the points in the flat region tend toward negative infinity, as seen in Fig. 3.

This lets the height of the jump of m^2 diverge linearly with the RG-time, which may lead to additional complications, particularly, concerning the stability of the spatial discretization in the vicinity of the jump. Similar to the case of the mass formulation above, there is only one point inside the jump discontinuity that is interpreted in the same manner.

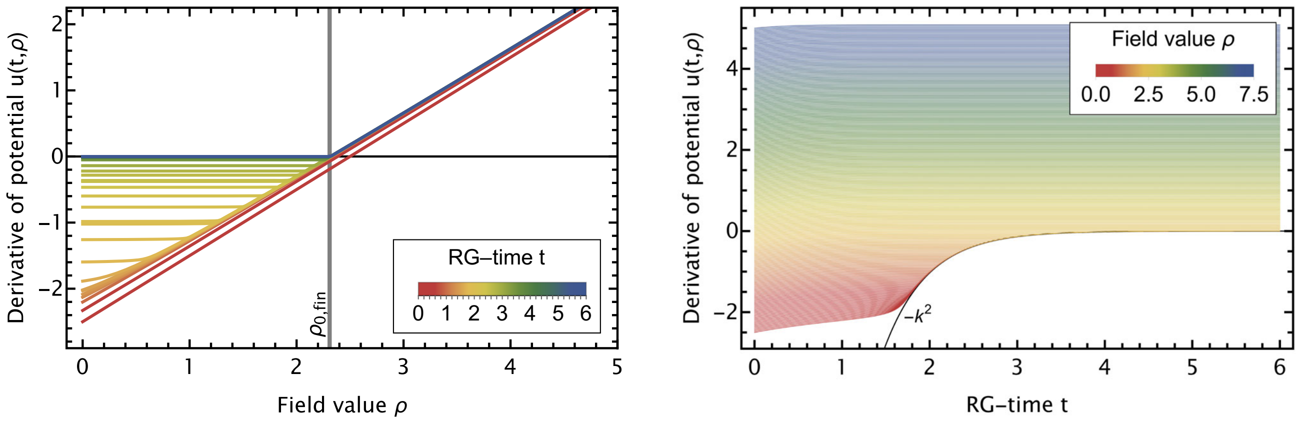


FIG. 1. Solution of the standard formulation (13) in the (ρ, t) plane. Left: the linearly interpolated ρ-dependent solution is shown for different RG-times t. Right: the RG-time evolution of individual grid points is shown.

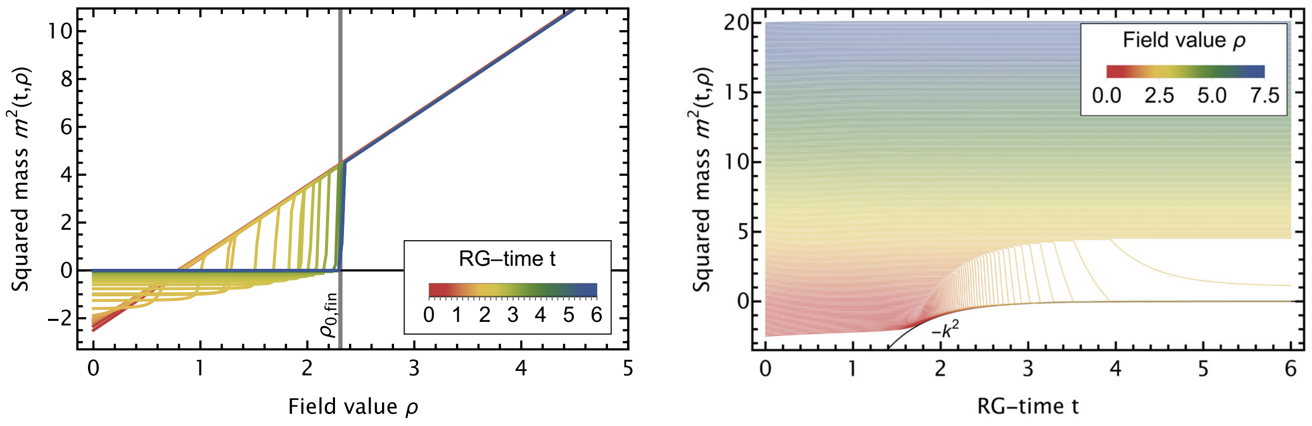


FIG. 2. Solution of the mass formulation (14) in the (ρ, t) plane. Left: the linearly interpolated ρ-dependent solution is shown for different RG-times t. Right: the RG-time evolution of individual grid points is shown. The shock is smeared due to the diffusive character of the spatial discretization. The effect thereof, a single grid point gets frozen inside the jump, can be clearly seen in the right panel.

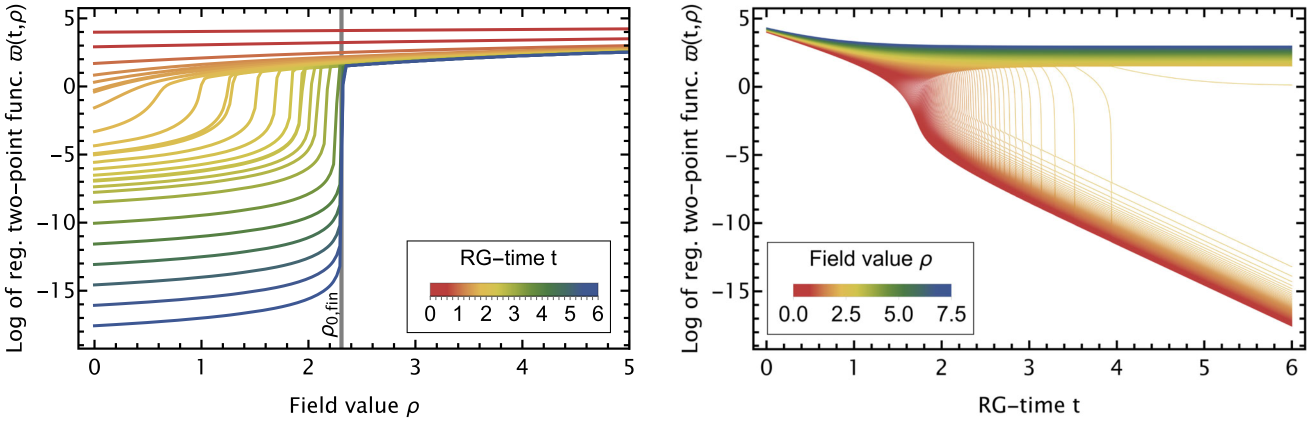


FIG. 3. Solution of the log formulation (15) in the (ρ, t) plane. Left: the linearly interpolated ρ-dependent solution is shown for different RG-times t. Right: the RG-time evolution of individual grid points is shown. The shock is smeared due to the diffusive character of the spatial discretization. The effect thereof, a single grid point gets frozen inside the jump, can be clearly seen in the right panel.

A further peculiarity can be seen when considering the RG-time evolution of the individual grid points in the right panel of Fig. 3. Every time a grid point enters the flat regime, every other point in the flat region is shifted to its predecessor, leading to a stairlike behavior. In the other formulations, this is not visible, as it happens entirely in the flat region. Nevertheless, such a behavior is certainly relevant in order to properly resolve the flat region and to avoid the singularity bound.

Finally, we would like to note that there is no known, stable discretization scheme that avoids the issues or qualitative features discussed above.

For completeness, we have also included the visualization of the solution in the case of added fermions at this point in Fig. 4. A discussion thereof can be found in Sec. IV D.

All solutions discussed in this section correspond to our reference solutions. They are calculated as close to machine precision as possible, and details thereon are given in Sec. IV A.

III. FRAMEWORK FOR NUMERICAL EXPERIMENTS

In this section, we describe the implementation details and general setup we have used to analyze and benchmark the RG-time integration of the sets of ODEs resulting from the discussion in Sec. II A.

The solution of ODEs with an implicit algorithm requires explicit choices for several different components:

(i) Algorithm.
(ii) Nonlinear solver.
(iii) Linear solver.
(iv) Time step controller.
(v) Error norms.
(vi) Determination of the Jacobian.

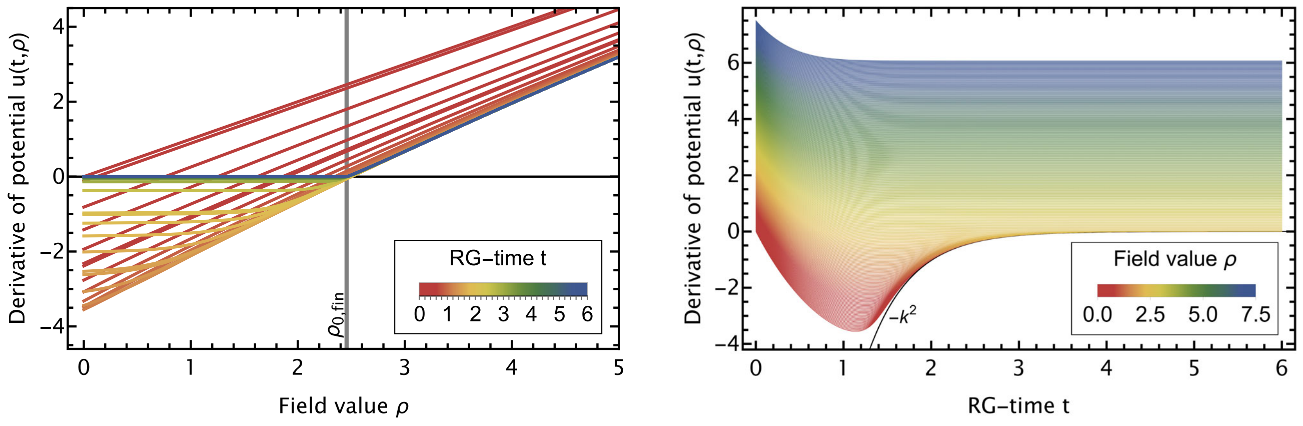


FIG. 4. Solution of the standard formulation in the case with fermionic extension (19) in the (ρ, t) plane. Left: the linearly interpolated ρ-dependent solution is shown for different RG-times t. Right: the RG-time evolution of individual grid points is shown.

Note that not all algorithms require all the other components, e.g., Rosenbrock methods have a Newton iteration step built in and, consequently, do not require a nonlinear solver. Nevertheless, all components can be discussed on a relatively general footing, since we found many patterns to be independent of the concrete algorithm in use.

To discuss all these different aspects, we found the DifferentialEquations.jl [25] package most suitable. All our tests are written in the JULIA language [26] and available on GitHub [27].

A. Components required for numerical solution

In the following, we discuss the different components and how they affect our benchmark process or the possibility to solve the system in general.

1. Algorithm

The first and most prominent component in the numerical solution is the choice of a time stepping algorithm itself. Broadly, time stepping algorithms fall into two categories, implicit and explicit schemes, with numerous candidates in both categories, as well as the possibility of hybrid schemes from which to choose. However, in the present context, it is a simple choice.

To see this, consider that, in the flat regime of the potential, the local wave speed grows exponentially fast (see [6] and Sec. IV B; note that the wave speed is defined as the largest eigenvalue of the local Jacobian). As a consequence, the maximum step size in RG-time becomes exponentially small by virtue of the CFL condition, i.e.,

$$\Delta t < \frac{C\Delta x}{\alpha} \sim e^{-bt}, \tag{23}$$

with C being the Courant number of the problem and b the scale exponent of the wave speed α.

While a straight forward CFL condition as in (23) only holds exactly in the infinite N limit of the O(N) theory, cf. [6], the flat part of the potential is convection dominated, and thus (23) provides a good approximation also in the present case. Furthermore, the presence of diffusive terms usually decreases the allowed step size, worsening the situation. In summary, the required exponentially small time step size renders explicit time stepper methods infeasible, and we will focus on implicit algorithms only.

This still leaves us with a vast landscape of implicit algorithms. For a detailed introduction (see, e.g., [28,29]), here we only state their rough features. The popular family of Runge-Kutta schemes splits into two different classes.

The first and most commonly used class of algorithms are the diagonally implicit Runge-Kutta (DIRK) methods, which includes well-known algorithms such as `ImplicitEuler`or the trapezoid/implicit midpoint algorithm used in the Crank-Nicolson scheme. The second class contains the fully implicit Runge-Kutta (FIRK) schemes. While in DIRK schemes for each internal stage the system of equations can be solved subsequently, in FIRK schemes a coupled system of equations has to be solved. To be more precise, in explicit RK schemes, each stage is obtained from previous stages only, while in DIRK methods, the right-hand side of each stage can contain itself. In FIRK methods, the right-hand side may contain all stages. As a direct consequence, DIRK schemes are a lot easier and cheaper to implement and are typically also faster, if they capture the stiffness of the system. Consequently, DIRK methods are the most commonly implemented implicit methods in toolboxes for PDEs.

The next big class of algorithms under investigation in this work includes the Rosenbrock methods, including their derived Rosenbrock-W extensions. They incorporate the Jacobian directly in the time step update, essentially performing the step of a Newton iteration, and they can be seen as an extension of the DIRK methods. As a consequence, for nonautonomous equations also the

RG-time gradient of the flux appears in the formula. They show impressive stability and accuracy properties for a wide range of problems, but their implementation is comparatively tedious. In schemes collected as Rosenbrock-W, the Jacobian and time gradient are not updated at every step, but in a lazy manner, depending on the algorithm.

The last big class of implicit algorithms includes the implicit multistep schemes. Compared to the RK schemes, no internal stages are evaluated, but the information from previous steps is used in the update step. Some of the most used libraries to solve stiff systems of ODEs, including the SUNDIALS library [30,31] or the ODEPACK package [32], focus on implicit multistep methods. Both of the aforementioned libraries are also included in our study via their JULIA interface [25]. A big advantage of implicit multistep methods is that they easily allow for adaptive choice of order, greatly improving performance.

To conclude this subsection on algorithms, we would like to mention and comment on some other classes of algorithms.

The first of these are implicit extrapolation methods, which are included in the survey plots of Figs. 8–13. We found them, in general, to be considerably underperforming compared to other algorithms. However, one appealing feature is their prospects regarding parallelizability, see, e.g., [33].

Another class of algorithms that we deemed promising when starting this project were exponential integrators, see, e.g., [34]. These algorithms are built around the idea to split the equation in a linear part, which is integrated exactly, and a nonlinear part, which can be handled explicitly. We deemed this promising, since at asymptotically large RG-times all values are simply exponentially decaying in the standard formulation, cf. Fig. 3. However, we could not find a suitable splitting of the equation, nor did automatic splitting based on a linearization at each time step, as implemented in [25], work, i.e., reach a final RG-time of at least $t = 6$ in any formulation. These shortcomings might be overcome in the future with implicit exponential integrators. However, this would include an implicit algorithm again, making the extra effort questionable.

For completeness, we also mention that we tried the algorithms offered by the IRKGaussLegendre.jl package [35], but found them unsuitable due to the general failure of functional iteration, commented on below.

2. Nonlinear solver

Most implicit algorithms require the solution of a nonlinear system of equations, with the notable exception of Rosenbrock methods, which incorporate a Newton solver iteration step already. In the context of solving systems of ODEs numerically, there are essentially two commonly employed methods:

Newton iteration.—This includes the classical (quasi)Newton methods for solving systems of equations. We found this class of methods to be the only one that works in practice. Particularly, `NLNewton` in the ODE JULIA framework [25] worked well and was insensitive to changes of numerical tuning parameters.

Functional iteration.—Unfortunately, we were unable to get neither direct iteration nor Anderson accelerated variants thereof to work properly, i.e., to converge to sufficient accuracy within a reasonable number of iterations. This can, at least partially, be explained by the peculiar eigenvalue spectrum of the Jacobian in the flat region, which will be discussed further in Sec. IV B.

3. Linear solver

All RG-time evolution algorithms, which can seriously be considered in the present context, require the solution of systems of linear equations at each RG-time step. In our setup, a plethora of different methods is available via the interface to the JULIA package LinearSolve.jl. We found that all commonly used algorithms converge, but that there are significant differences concerning their performance. Lower–upper (LU) decomposition-based algorithms and, in particular, their sparse variants performed extremely well. Krylov subspace and QR decomposition-based algorithms, on the other hand, performed significantly worse, increasing the run-time by several orders of magnitude. This can, again, be explained by the peculiar eigenvalue spectrum of the Jacobian, discussed in Sec. IV B. In the current context, this connection is particularly obvious in the QR algorithm, where the rate of convergence is proportional to the ratio of eigenvalues.

4. RG-time step controller

Adaptive control over the RG-time step size is, in general, a very attractive feature, for the obvious reasons. Additionally, in FRG applications, it is common to redo calculations for changing initial conditions (either to tune initial conditions or to scan parameter spaces for phase structures) and the characteristics of the flow can change substantially. Manual control over the RG-time step size is rather inefficient in these scenarios, making it particularly relevant in FRG flows.

The most common options for controller of the adaptive step size are of control loop type, i.e., integral, proportional-integral, or proportional-integral-derivative controller. We have tested all three options, and additionally Gustafsson acceleration, which are the four variants implemented in [25]. We found no noteworthy difference and simply chose the default option of the individual RG-time step algorithms.

5. Error norms

Both for the purpose of checking if a Newton solver has converged, as well for choosing the size of an adaptive time step, an error norm of the residual is used. For large problems, where a LU decomposition is not feasible, such a norm is also needed to check if an iterative algorithm has converged.

Also in view of Sec. IV B, with regard to stability, it is important to ensure that the error norm takes into account spatially local accuracy. To be more explicit, consider absolute and relative precision limits s_{abs} and s_{rel}. For a residual vector r and a solution vector before the iteration (u) and after ($\tilde{u}$) on a grid of size N, we could define the normalized error using an l_2 norm as

$$e_{l_2} = n_{l_2} \frac{\|r\|_2}{s_{\mathrm{abs}} + s_{\mathrm{rel}} \max(\|u\|_2, \|\tilde{u}\|_2)}, \tag{24}$$

with $n_{l_p} = N^{\frac{1}{p}}$. Generically, this is the standard kind of error norm implemented in many discretization frameworks for finite difference and finite elements. However, such a definition can allow locally large errors in cases where locally strong dynamics occur, which quickly destabilize the system.

Therefore, an error norm that evaluates the relative precision locally is more pertinent,

$$e_{\mathrm{loc}} = n_{l_p} \left\| \frac{r_i}{s_{\mathrm{abs}} + s_{\mathrm{rel}} \max(|u_i|, |\tilde{u}_i|)} \right\|_p. \tag{25}$$

In DifferentialEquations.jl and in the SUNDIALS suite, the error norm (25) is implemented with $p = 2$, which is sufficient for the problems considered in this work. However, we would like to comment that, in the presence of spatially localized phenomena with fast dynamics, we found it necessary to choose $p = \infty$ in order to ensure stability of the evolution.

6. *(Automatic) Jacobian construction*

Finally, we have to consider the construction and approximation of the Jacobian, required for most time step algorithms. First, it should be noted that spatial discretizations dealing with convection dominated PDEs usually have sparse Jacobians. Utilizing this structure, i.e., working with sparse matrices, is of course important and closely related to the discussion about linear solvers, cf. Sec. III A 3. In practice, there are three different options to calculate the Jacobian:

(i) Automatic differentiation.
(ii) Exact/symbolic calculation.
(iii) Finite differences.

We found that the use of Jacobians that are accurate on the level of the working precision greatly benefit the numerical RG-time integration. This is achieved by the first two options.

The first option, i.e., automatic differentiation, is an option to obtain the Jacobian at working precision, by repeated application of the chain rule at the level of the compiler. In the JULIA language, this technique is readily available via ForwardDiff.jl [36] and the calculation of Jacobians is automatically supported in DifferentialEquations.jl. We found this to be the most convenient setup and used it as default throughout this work for all JULIA-based solvers, i.e., all except the SUNDIALS (`CVODE_BDF`), and ODEPACK (`lsoda`) routines. In these cases, we resorted to their internal default.

The second option, implementing the explicit Jacobian, performed very similar to the case of automatic differentiation, but comes with the additional overhead of deriving it. Sometimes, this task can be efficiently automated, making it also a very viable option, particularly if automatic differentiation is not an option.

Finally, we found the practical usability of finite difference to be strongly dependent on the heuristics employed by different libraries. The external libraries, mentioned above, performed very well, utilizing their own approximation heuristics. On the other hand, we did not manage to get the JULIA-based solver combined with finite differences, i.e., using FiniteDiff.jl, to work at all. We would like to add that we have previously made similar experiences in *Mathematica*, leading us to the stated conclusion.

B. Setup of the numerical experiments

All benchmark runs measuring execution times were run 5 times to avoid outliers; the reported value corresponds to the median. In all cases, we found the median to be very close to the minimum, confirming the absence of outliers. The implementation is available at GitHub [27]. The study was performed on a laptop with Windows 11 operating system, an Intel Core i7-11800H CPU with a maximal capacity of 4.60 GHz, and 32 GB RAM. To facilitate comparison, no forms of parallelization were used; all calculations were restricted to a single core and thread. All floating-point operations were performed in the double-precision floating-point format. The final RG-time is chosen as $t = 6$, unless specified otherwise. At $t = 6$ the flow is safely in the asymptotic regime, cf. Sec. IV B.

IV. RESULTS

This section presents our analysis and results, focusing on the RG-time evolution solver algorithm. The other components were found to be generic regarding their effect on the numerical RG-time integration and are discussed in the previous section, i.e., Sec. III.

A. Work precision

For solving flow equations in everyday applications, the most important quantity measuring the performance is the work-precision relation, i.e., how the accuracy behaves with computing time. Therefore, we surveyed 49 different algorithms, collected in Appendix B. Beforehand, a reference solution was generated using the `KenCarp58` algorithm.

For this reference solution, the absolute and relative accuracy goals were chosen as 10^{-15}, the highest accuracy feasible with double-precision floating-point numbers. By comparing different algorithms, i.e., `RadauIIA5`, `QNDF`, and `Rodas4`, at this target accuracy, we found the maximally actual achievable accuracy to be roughly 10^{-12}, measured in the ℓ_2 norm. Therefore, we excluded all results in the survey with $\|a_i - a_i^{\mathrm{ref}}\|_{\ell_2} < 1.25 \times 10^{-12}$. Slight remnants of the resulting saturation effect when investigating the work-precision relation are still visible, particularly in the results for the mass formulation (14).

In order to investigate the work-precision relation, we scanned all possible combinations of absolute and relative target accuracies in the range $10^{-8} - 10^{-12}$ in steps of one ($\log_{10}$ scale). We excluded all runs that did not reach the final RG-time $t = 6$, which was mostly the case at lower accuracy goals and explains the lower scan range of 10^{-8}. The full survey for work-precision relations are collected in Fig. 8 for the standard formulation (13), in Fig. 9 for the mass formulation (14), and in Fig. 10 for the log formulation (15). The general performance is similar for all three formulations, hence we refrain from separate discussions for each formulation.

The DIRK algorithms, the `KenCarp` family, except `KenCarp5`, and `TRBDF2` are looking very promising. Particularly, `KenCarp3` and `TRBDF2` are showing a stellar performance when taking into account that they are low-order methods. It is also noteworthy that the popular choices of `ImplicitEuler` and Trapezoid performed extremely poor.

The Rosenbrock methods are relatively independent of the specific choice of algorithm. While they do require significantly more time to push the high precision boundary, they shine through their remarkable stability and overperformance in actually achieved accuracy.

Rosenbrock-W methods are running on average slightly faster, but their decreased stability dims their attractiveness.

We turn to implicit multistep methods, for the purpose of this discussion, including SUNDIALS's `CVODE_BDF` and ODEPACK's lsoda. While being unable to push toward really small accuracies, the variable-order BDF implementations, i.e., `QBDF`, `QNDF`, and `CVODE_BDF`, are attractive due to their very short run-times. `FBDF` performed similarly, but performed worse than `QBDF` and `QNDF` due to reduced stability, i.e., requiring a significantly smaller minimal allowed step size.

Turning toward the FIRK methods, we found `RadauIIA5` to be very stable and well performing, while `RadauIIA3` was one of the worst performing algorithms that still managed to solve the system. We could not find an obvious explanation for the drastic difference between the two very similar algorithms. We suspect it to be related to the lack of a pseudostability region for `RadauIIA3`, which will be discussed in Sec. IV C.

Finally, the family of implicit extrapolation algorithms performed significantly worse than the other families of algorithms, rendering them irrelevant for normal applications to flow equations.

For the reader's convenience, a collection of well-performing algorithms, and `ImplicitEuler` for reference, is shown in Fig. 5. The depicted algorithms, i.e., `QNDF`, `TRBDF2`, `Rodas4`, `RadauIIA5`, and `CVODE_BDF`, can be considered excellent choices when being confronted with a novel problem/equation in the context of derivative expansions in flow equations.

B. Jacobian eigenvalues

Further insight into the difficulties encountered inside the flat region can be gained by investigating the Jacobian of the system. For simplicity, we restrict ourselves to the standard formulation (13). The results are, however, directly applicable to all three formulations. The (negative) eigenvalues of the (discrete) Jacobian are shown in Fig. 6 as a function of RG-time. Some peculiarities of the system become immediately obvious. First, each grid point in the flat region comes with its own exponentially growing eigenvalue. Please note that this statement is entirely dependent on the spatial discretization, but the situation is similar in all known, stable discretizations. Second, their exponential growth sets in subsequently. The eigenvalue that can be associated to the grid point inside the kink/jump is the last one to show dynamic behavior. It shows similar growth to the other points in the flat region, but freezes in the nonflat region. This marks a unique RG-time where extrapolation, similar to the one discussed in [6], can be performed safely. Finally, the nonflat region of the potential starts freezing in exponentially fast, once the flow becomes smaller than the local mass scale. This is the expected behavior of RG flows. Please note that the wiggly eigenvalue that is significantly more suppressed than all other eigenvalues is related to the boundary condition (17) at ρ_{max}.

To summarize, the eigenvalues split in two groups, an exponentially growing one in the flat region and an exponentially suppressed one in the nonflat region. Consequently, there is an exponentially growing gap between these two groups. This significantly reduces the performance of all algorithms whose rate of convergence is related to the ratio of eigenvalues, cf. Sec. III. Additionally, it hints toward potential design principles for new spatial discretizations, i.e., reducing the number of exponentially growing eigenvalues per evaluation node in the flat region. Hereby, the insights from [16] might be particularly useful, providing linearized equations in the flat region.

The exponential increase of the Jacobian eigenvalues also shows that the system becomes increasingly susceptible to large errors and instabilities due to perturbations. Such perturbations can quickly arise due to numerical errors done in previous time steps, making

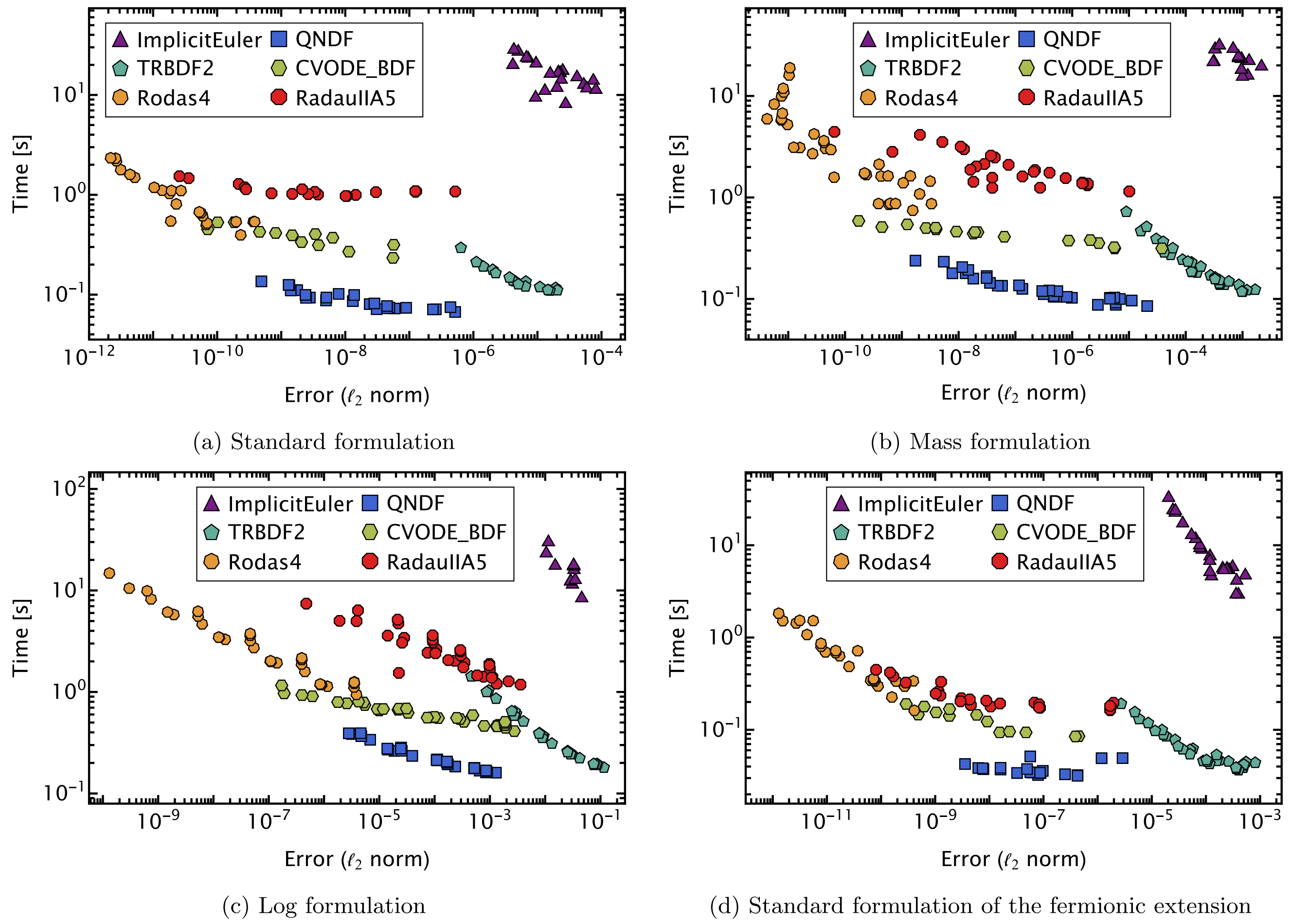


FIG. 5. Work-precision diagrams for all three formulations, the fermionic case, and selected algorithms. Noteworthy is the very efficient nature of the implicit multistep algorithm `QNDF`, the high accuracy of the Rosenbrock algorithm `Rodas4`, and the poor performance of `ImplicitEuler`. The complete survey is depicted in Figs. 8–10 and a detailed discussion is given in Sec. IV A.

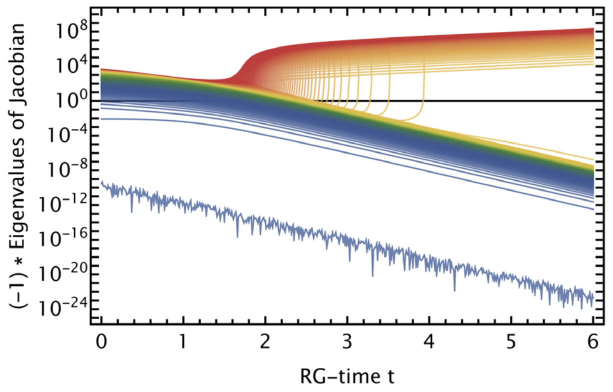


FIG. 6. RG-time evolution of the negative eigenvalues of the Jacobian. The significantly suppressed, wiggly line can be attributed to the boundary condition at large field values, i.e., the $\rho_{\max}$ boundary. The remaining eigenvalues split in two categories, the ones associated with the flat regime, which shown exponential growth, and the remaining ones, which show exponential suppression. A detailed discussion of the implications thereof is given in Sec. IV B.

high precision during the time stepping an important prerequisite for stability.

C. Stability in the infinite RG-time limit

There is another aspect when solving FRG equations in everyday applications that is highly relevant: the long RG-time stability of the RG-time integration. In order to test this faithfully, we switch to fixed RG-time step size evolution, where the step size is denoted by Δt. The flow is integrated until we detect an inconsistency/instability in the flow, i.e., either the solution becomes nonmonotonic or the positivity bound is violated. The last successful RG-time step then determines $t_{\max}$. Ideally, there would be some maximal step size Δt_{cr}, for which all smaller RG-time steps $\Delta t < \Delta t_{\mathrm{cr}}$ provide stable evolution, i.e., $t_{\max} \to \infty$. In practice, we cannot integrate until $t_{\max} \to \infty$. Therefore, we chose the maximal cutoff at $t = 50$, which is the same for all practical purposes. The external libraries are excluded for technical reasons in this survey; we do not expect them to perform as well as the pure JULIA

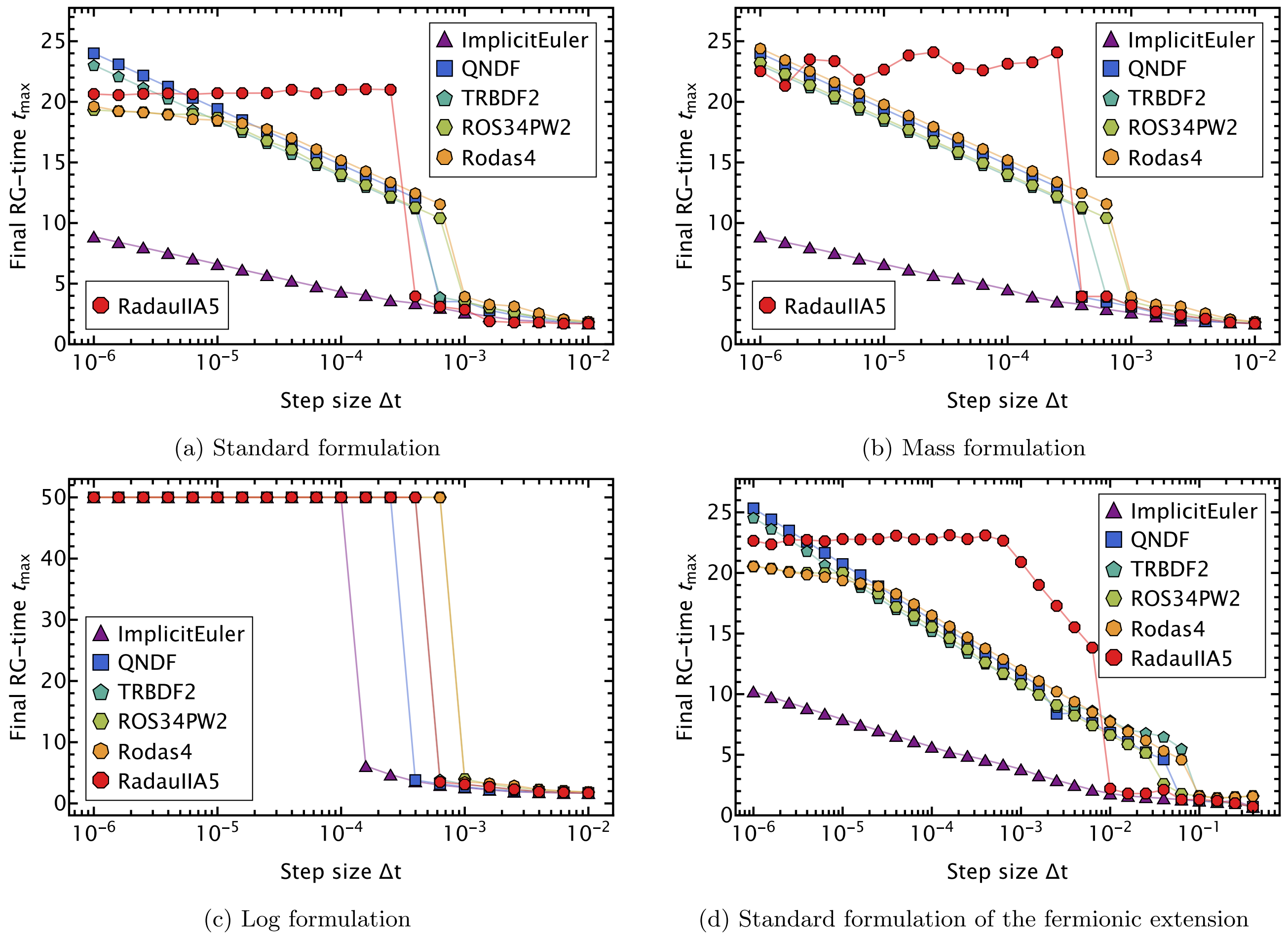


FIG. 7. RG-time stability for all three formulations, the fermionic case, and selected algorithms. Noteworthy is the notion of pseudostability in the standard (13) [including the fermionic case (20)] and mass formulations (14) and the proper stability in the logarithmic formulation (15). The complete survey is depicted in Figs. 11–13 and a detailed discussion is given in Sec. IV C.

implementations, due to the lack of autodifferentiation in the calculation of the Jacobian.

The full survey of results can be found in Fig. 11 for the standard formulation (13), in Fig. 12 for the mass formulation (14), and in Fig. 13 for the log formulation (15). The selection of algorithms is depicted in Fig. 7. There are some immediate observations: First, the only formulation that shows proper stability is the log formulation (15). At $\Delta t \approx 10^{-3} - 10^{-3}$ the final RG-time becomes independent of the step size. Second, the other two formulations do not exceed $t_{\mathrm{max}} \approx 20$–25, i.e., they do not feature stability. However, at roughly the same Δt, where the log formulation shows stability, they feature a sharp increase in t_{max}. This sharp increase precisely makes the difference between breaking down during the flattening of the potential and being able to properly resolve this essential part of the evolution. We call this notion "pseudostability." Luckily, a large part of the investigated algorithms, particularly the Rosenbrock and implicit multistep methods, posses this property. The final RG-time reached by most algorithms in the standard and mass formulation is limited by double precision in the (non)linear solver part. This can be easily seen by estimating the relevant orders of magnitude in their discrete equations.

Furthermore, for the method `RadauIIA5`, the jump in the final RG-time is so large that we cannot differentiate between pseudostability and proper stability of the algorithm. Because of the notion of the severe size of the jump in the mass formulation, cf. Fig. 7(b), one might suspect that the method is stable, but limited by finite numerical precision. However, this statement has to be contrasted with the observations of the fermionic case, discussed in Sec. IV D.

Particularly, the log formulation, despite its infinitely growing gap in the solution, gets quite appealing due to the RG-time stability. We would like to note that the generalization thereof to more complicated theories, e.g., a $O(N)$ theory with $N > 1$, is ambiguous and has to be worked out.

In general, the notion of pseudostability hints toward the existence of two different stability criteria: One being related to stability with respect to nonanalyticities, i.e., the kink in the current example, and the other one being related

TABLE I. Overview of all methods explored in this work. Their names are given as used in the DifferentialEquations.jl package [25]. If the order of the algorithms is variable and may be changed during run-time, it is denoted as "v." Additionally, we give a subjective rating of each algorithm for the application at hand, which should serve as guidance only. The ratings are explained in (B1), with S being the best and D the worst.

Name	Reference	Order	Rating
	DIRK		
`ImplicitEuler`		1	C
`Trapezoid`		2	C
`TRBDF2`	[37]	2	S
`SDIRK2`	[30]	2	C
`Kvaerno3`	[38]	3	C
`KenCarp3`	[39]	3	S
`Cash4`	[30]	4	D
`Hairer4`	[40]	4	A
`Hairer42`	[40]	4	A
`Kvaerno4`	[38]	4	B
`KenCarp4`	[39]	4	S
`KenCarp47`	[41]	4	S
`Kvaerno5`	[38]	5	C
`KenCarp5`	[39]	5	B
`KenCarp58`	[41]	5	S
	FIRK		
`RadauIIA3`	[42]	3	C
`RadauIIA5`	[43]	5	S
	Implicit multistep		
`QNDF1`	[43]	1	B
`QBDF1`	[43]	1	B
`ABDF2`	[44]	2	B
`QNDF2`	[43]	2	A
`QBDF2`	[43]	2	A
`QNDF`	[43]	v	S
`QBDF`	[43]	v	S
`FBDF`	[45]	v	A

Name	Reference	Order	Rating
	Rosenbrock		
`ROS3P`	[46]	3	B
`Rodas3`	[40]	3	B
`RosShamp4`	[47]	4	B
`Veldd4`	[48]	4	A
`Velds4`	[48]	4	A
`GRK4T`	[49]	4	A
`GRK4A`	[49]	4	A
`Ros4LStab`	[40]	4	A
`Rodas4`	[40]	4	A
`Rodas42`	[40]	4	A
`Rodas4P`	[50]	4	A
`Rodas4P2`	[51]	4	A
`Rodas5`	[52]	5	A
	Rosenbrock-W		
`Rosenbrock23`	[43]	2	C
`Rosenbrock32`	[43]	3	D
`ROS34PW1a`	[53]	3	B
`ROS34PW1b`	[53]	3	B
`ROS34PW2`	[53]	3	A
`ROS34PW3`	[53]	3	A

(Table continued)

TABLE I. *(Continued)*

Name	Reference	Order	Rating
	External libraries		
`CVODE_BDF`	[25,30]	v	S
`lsoda`	[25,32,54,55]	v	A
	Implicit extrapolation		
`ImplicitEulerExtrapolation`	[25]	v	C
`ImplicitDeuflhardExtrapolation`	[25]	v	D
`ImplicitHairerWannerExtrapolation`	[25]	v	C

to the existence of a singularity bound and can be circumvented by the log formulation.

D. Fermionic extension of model

The results presented so far have all been obtained in the $\mathbb{Z}_2$ theory. Here we briefly discuss results of the theory with fermions introduced in Sec. II C. For simplicity, we restrict ourselves to the standard formulations; the extensions of other formulations, particularly the log one, will be discussed elsewhere.

A typical RG-time evolution of the derivative of the effective potential is shown in Fig. 4. Most notably, during the initial phase of the flow, the fermionic contributions increase the expectation value of the field. However, as soon as the flattening of the potential sets in, the evolution becomes very similar to the $\mathbb{Z}_2$ case. Additionally, the nonanalyticity seems softer, i.e., it manifests itself in the discrete system over a larger range in RG-time.

The work-precision result, shown in Fig. 5(d) for selected algorithms, is very similar to the pure $\mathbb{Z}_2$ case. This was to be expected, due to the very similar final solution.

The infinite RG-time analysis, however, reveals that the requirements for pseudostability are reached at already much larger time steps ($\approx 10^{-2}$), which can easily be explained by the softer nonanalyticity. Furthermore, the `RadauIIA5` algorithm clearly shows only pseudostable behavior. This might indicate that the observed behavior in the $\mathbb{Z}_2$ case was also only pseudostability, limited by finite numerical precision, or special to this particular case.

V. CONCLUSION AND OUTLOOK

In this work, we investigated the numerical RG-time integration of FRG flow equations as encountered in a derivative expansion of a scalar theory in three spacetime dimensions with a $\mathbb{Z}_2$ symmetry. This example displays the prototypical behavior of a flat region of the effective potential with an accompanying nonanalyticity at its boundary, marking the expectation value of the theory.

Investigating three different formulations of the equation, we found that certain families of algorithms, combined with (almost) exact Jacobians, are suitable to perform the RG-time integration to desired accuracy. An overview of our heuristic opinion is collected in Table I.

Interestingly, a considerable number of algorithms feature the notion of pseudostability, i.e., there exists a maximum step size that allows for properly resolving the nonanalyticity at the minimum of the potential. For smaller step sizes, the only limiting factor is the accuracy to resolve the singularity bound.

However, the log formulation, introduced in (9), circumvents this issue. This allows for integration to arbitrarily large RG-times, i.e., small RG-scales.

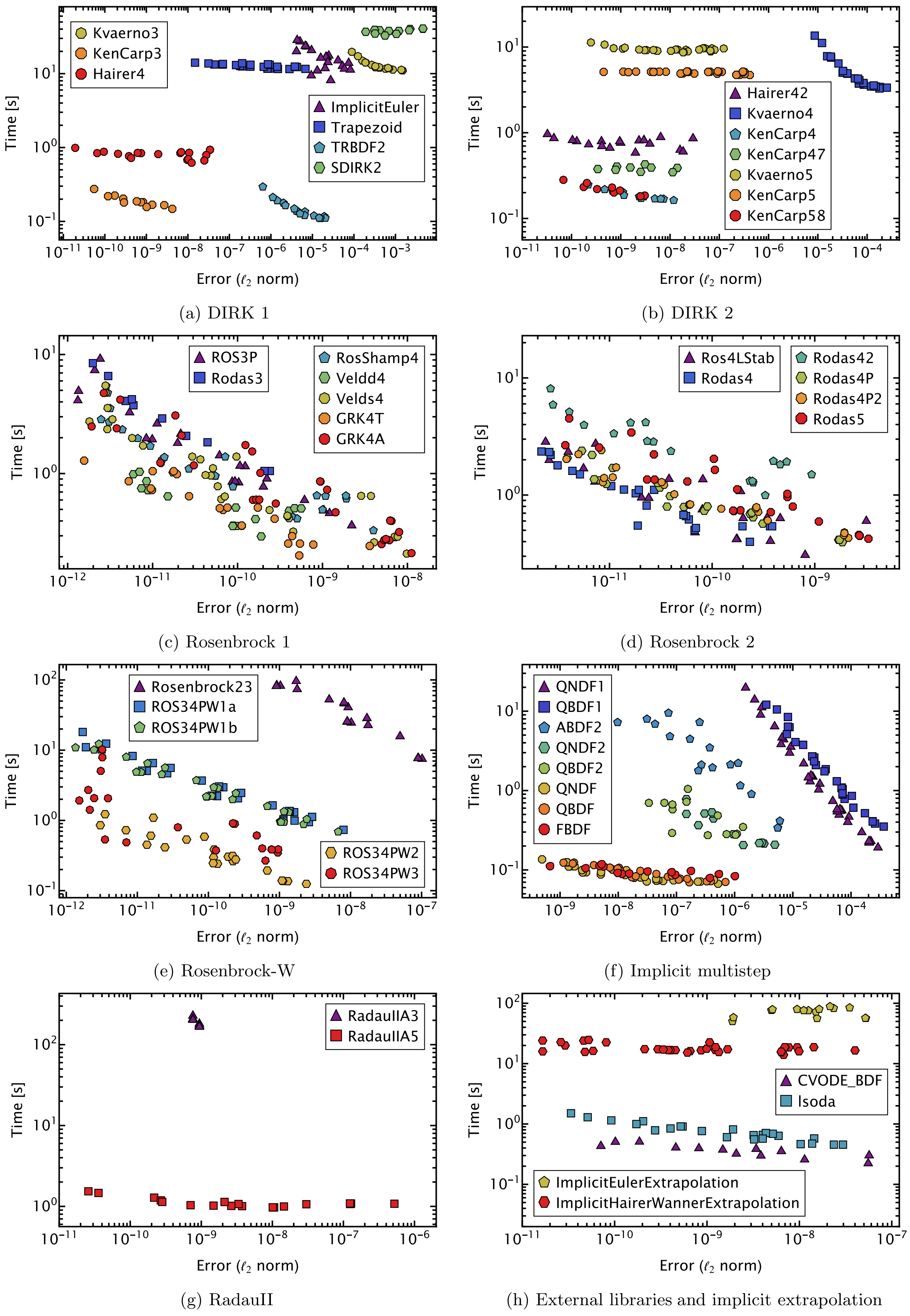
This immediately highlights future goals: First, to understand the different notions of stability arising from nonanalyticities and from the singularity bound. The discussion at the end of Sec. IV C could be a good starting point for a more formal investigation of the equation. The second goal is to understand the generalization of the log formulation and its feasibility in more general theories.

ACKNOWLEDGMENTS

We would like to thank Eduardo Grossi, Adrian Königstein, Jan M. Pawlowski, Fabian Rennecke, and Lorenz von Smekal for discussions. F. I. acknowledges funding by the Studienstiftung des deutschen Volkes. F. R. S. acknowledges funding by the GSI Helmholtzzentrum für Schwerionenforschung. N. W. acknowledges support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—Project No. 315477589—TRR 211 and by the State of Hesse within the Research Cluster ELEMENTS (Project ID 500/10.006).

APPENDIX A: FUNCTIONAL RENORMALIZATION GROUP

We briefly summarize the central ideas of the FRG and introduce our notation; for reviews see [1] and references therein. The RG describes the evolution of a quantum field theory along the (momentum) RG-scale k. It interpolates between the microscopic theory at small scales (large k) and the macroscopic theory at large scales (small k), by successively integrating out fluctuations. The FRG provides a modern functional implementation of this idea,



(a) DIRK 1

(b) DIRK 2

(c) Rosenbrock 1

(d) Rosenbrock 2

(e) Rosenbrock-W

(f) Implicit multistep

(g) RadauII

(h) External libraries and implicit extrapolation

FIG. 8. Work-precision survey for the standard formulation (13).

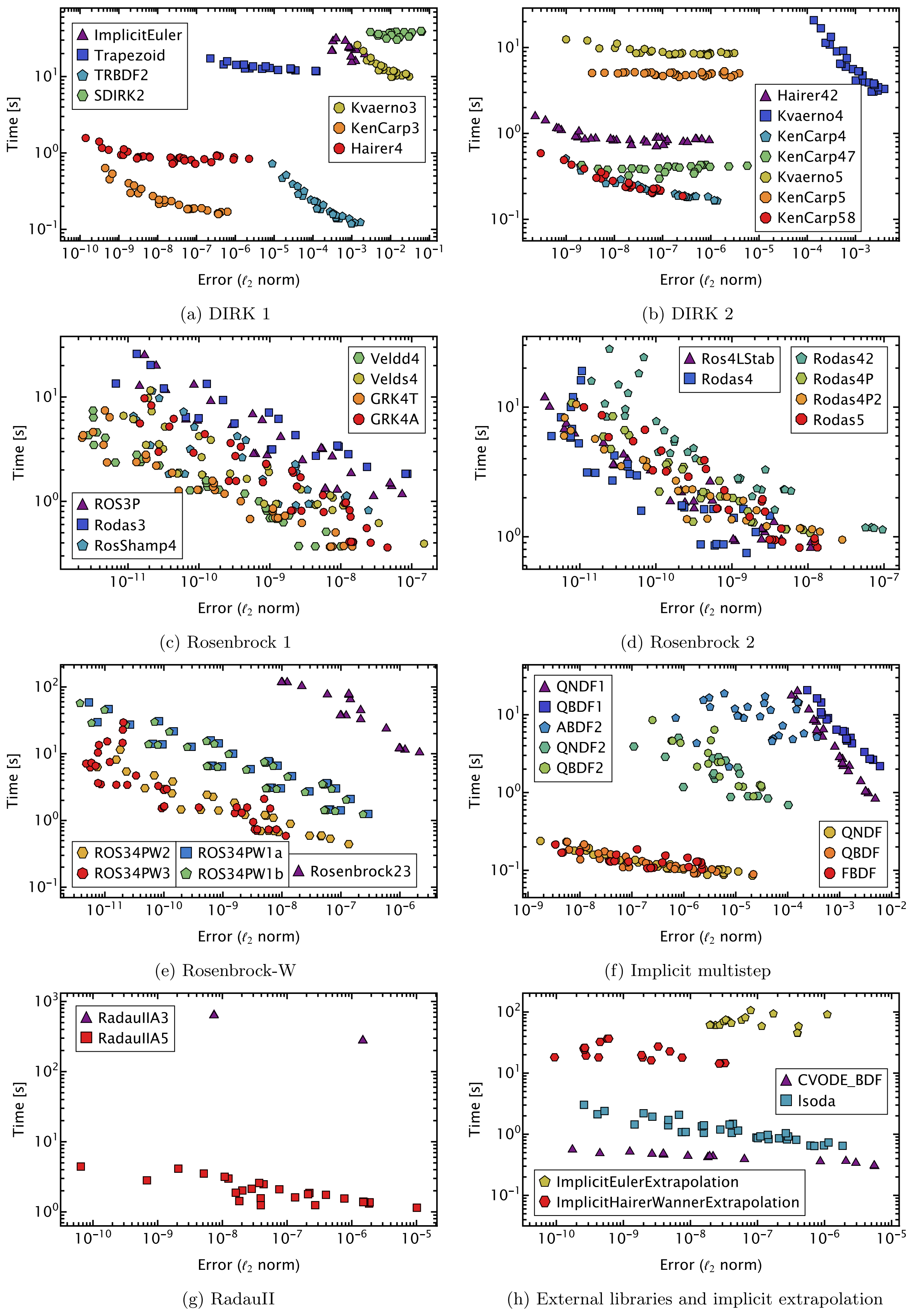


FIG. 9. Work-precision survey for the mass formulation (14).

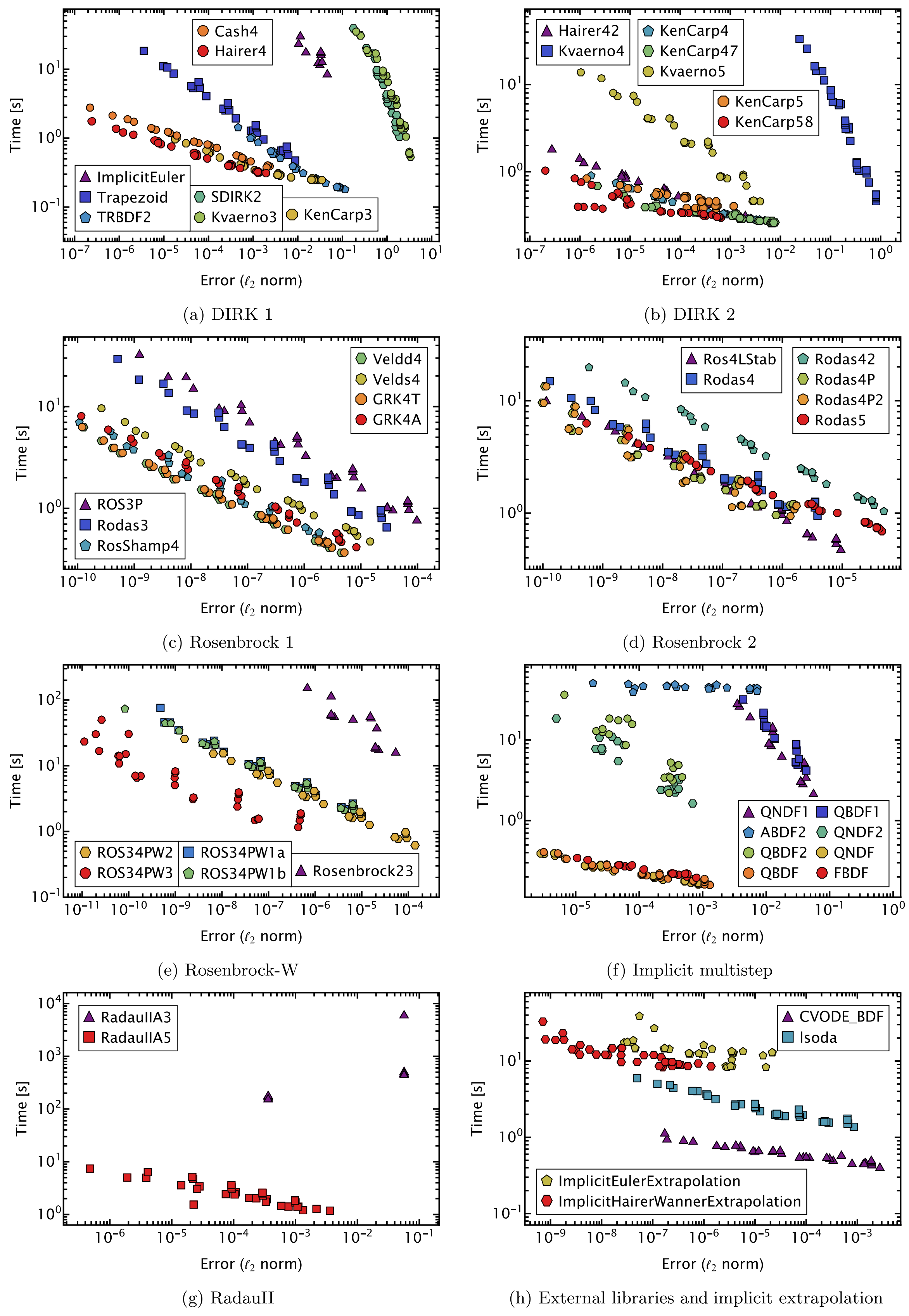


FIG. 10. Work-precision survey for the logarithmic formulation (15).

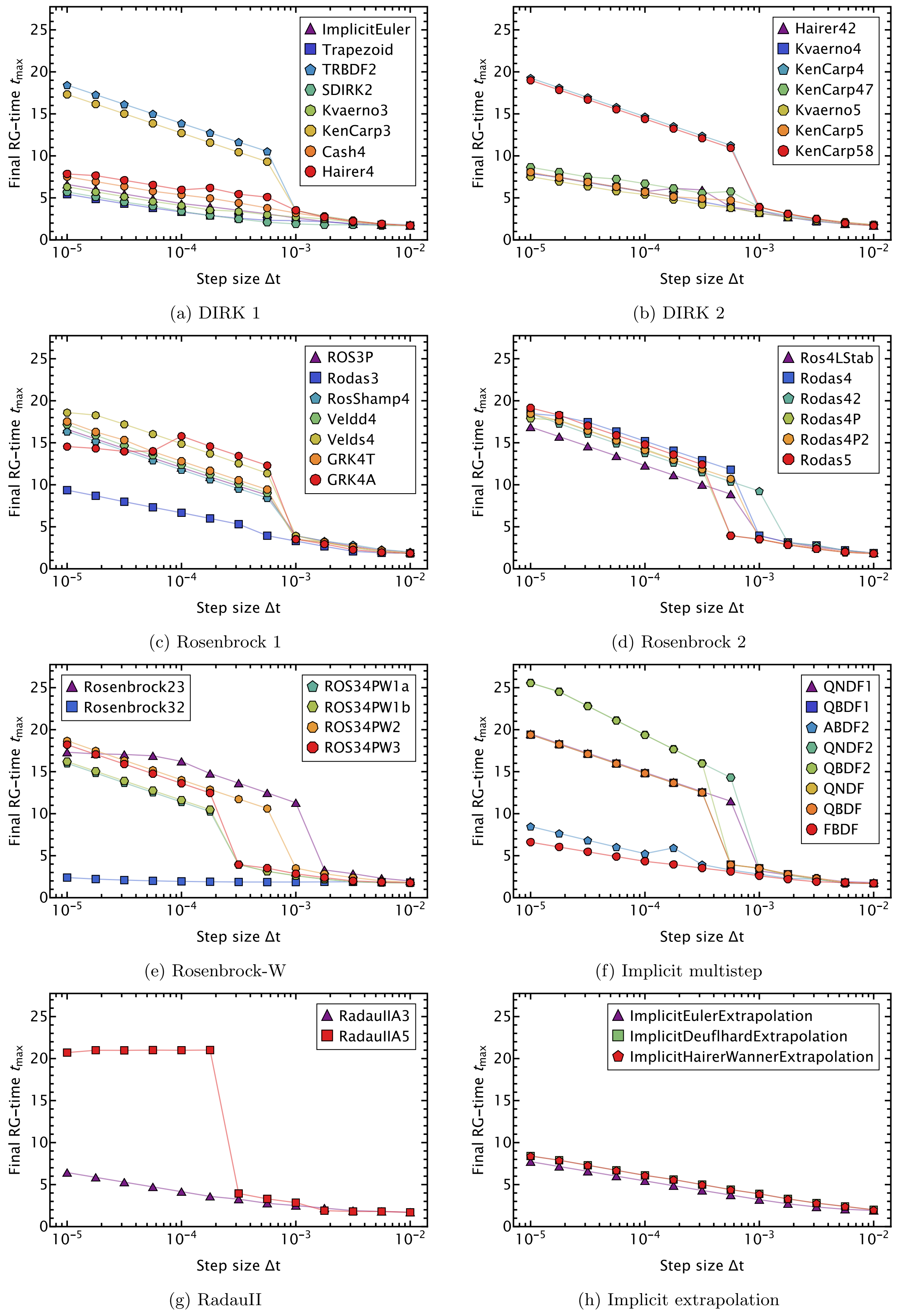


FIG. 11. Fixed step size RG-time stability survey for the standard formulation (13).

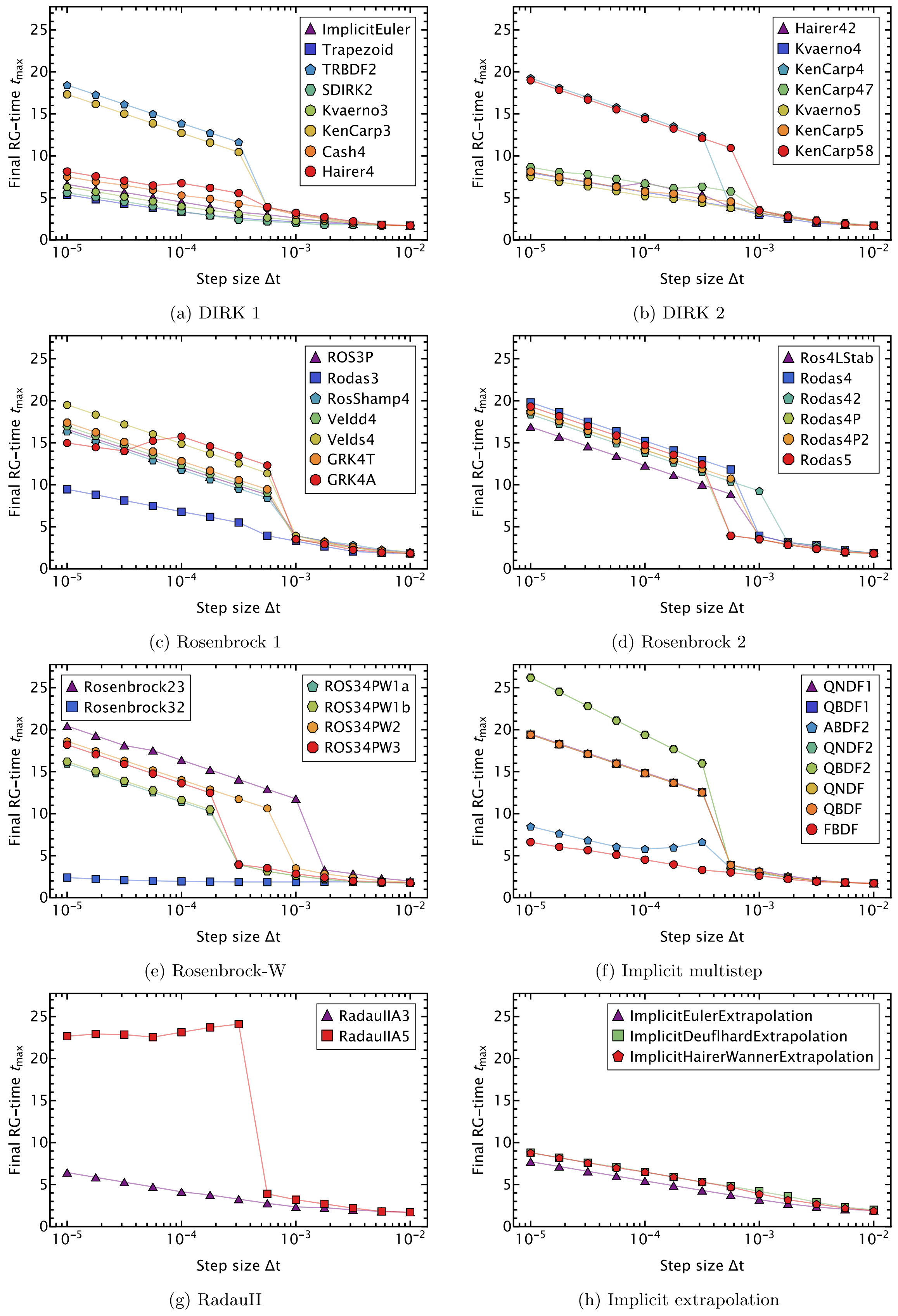
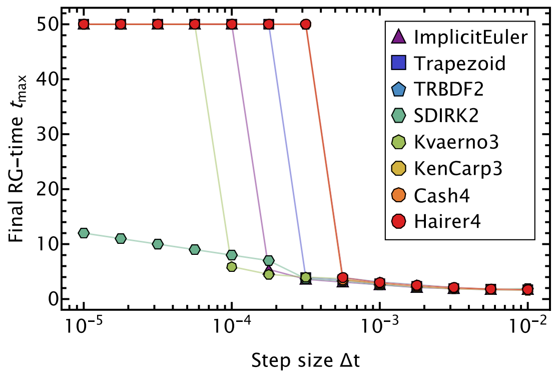
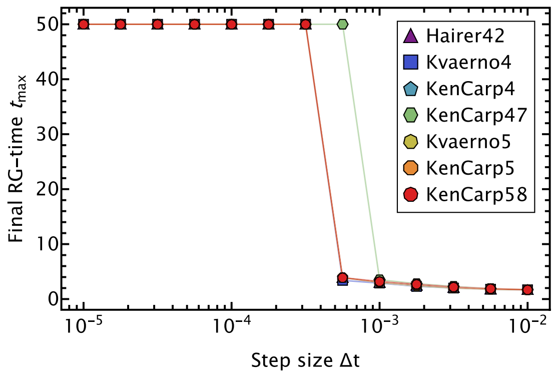


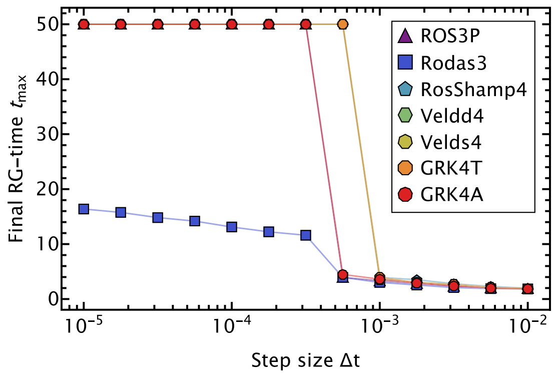
FIG. 12. Fixed step size RG-time stability survey for the mass formulation (14).



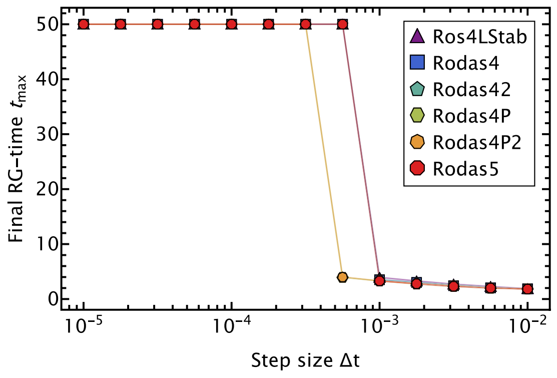
(a) DIRK 1



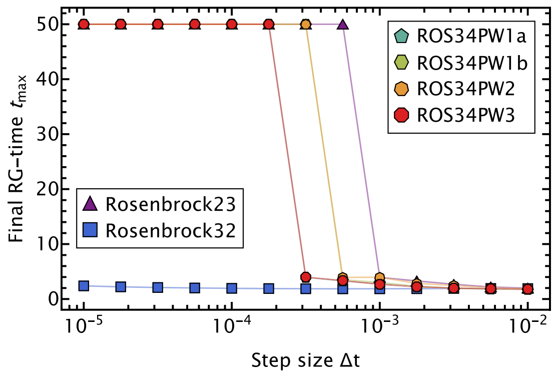
(b) DIRK 2



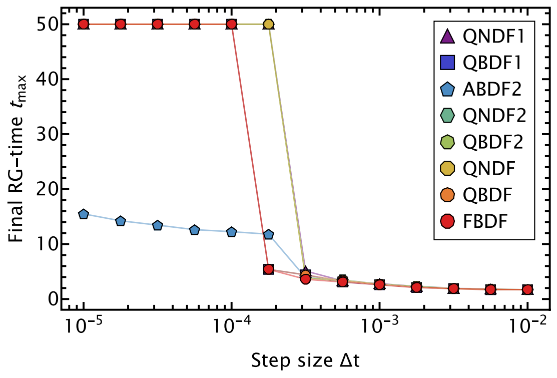
(c) Rosenbrock 1



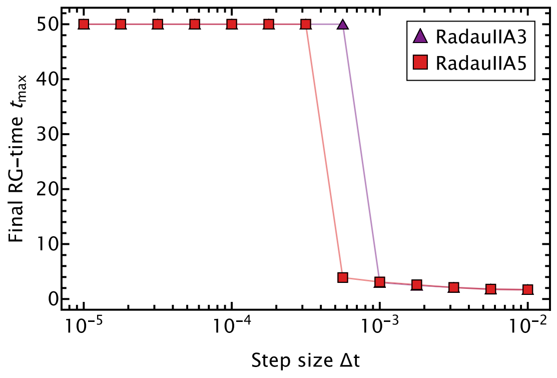
(d) Rosenbrock 2



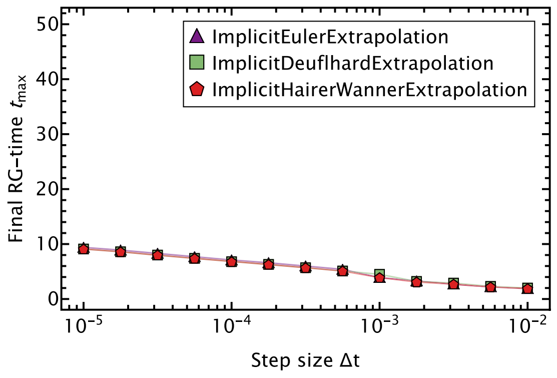
(e) Rosenbrock-W



(f) Implicit multistep



(g) RadauII



(h) Implicit extrapolation

FIG. 13. Fixed step size RG-time stability survey for the log formulation (15). The maximum RG-time was set to $t = 50$, which can, for all practical purposes, be identified with infinity.

$$\partial_t \Gamma_k[\phi] = \frac{1}{2}\mathrm{Tr}[(\Gamma^{(2)} + R_k)^{-1}\partial_t R_k], \tag{A1}$$

the Wetterich equation [56]. In (A1), the RG-time t is given by

$$t = -\ln\frac{k}{\Lambda}, \tag{A2}$$

with some (ultraviolet) reference scale Λ, which is usually used as initialization scale of the flow. In (A2) we would like to point out the extra minus sign compared to most FRG literature, which turns the RG-time evolution positive instead of negative. $\Gamma_k[\Phi]$ in (A1) is the quantum effective action (QEA) at a given RG-scale k, which has to be truncated to yield a finite set of equations, cf. Sec. II A. R_k denotes the regulator, which acts like a mass term $\sim k^2$ and hence renders loops infrared finite, while its decay in momentum space renders diagrams via its RG-time derivative in (A1) ultraviolet finite. Finally, the trace Tr collects summation/integration over indices/arguments, which includes an integration over spacetime and summation/integration over internal indices, which might be present, depending on the theory under investigation.

The QEA $\Gamma_k[\Phi]$ approaches the classical action $S[\Phi]$ of the theory for $t \to -\infty$ ($k \to \infty$). On the other hand, the regularization is removed in the limit $t \to \infty$ ($k \to 0$), where the physical theory, in a given truncation, is recovered. This RG evolution is provided by (A1), which typically results in a set of coupled PDEs. A common truncation scheme is the standard derivative expansion, whose leading order is called the local potential approximation. In this approximation, only the effective potential enters as a RG-scale-dependent quantity, and one is left with a single PDE.

APPENDIX B: OVERVIEW SOLVER

Table I collects all algorithms explored extensively in this work. The tables collect the name of the method, as implemented in [25], their reference(s), and their order (with v = variable). Additionally, we list a rating, which reflects our subjective view on the performance of the algorithm for the problem at hand. Hereby, we include the work-precision performance for all three formulations (13)–(15), as well as the fermionic extension of the problem (20). The RG-time stability at fixed step size, detailed in Sec. IV C, also played a significant role. Finally, when in doubt how to rank an algorithm, we have also considered the ease of implementation. For example, TRBDF2 is comparatively easy to implement, while Rosenbrock methods add another level of complexity. The rating scheme is as follows:

S Best performance, little to no drawbacks,

A Overall very good performance,

B Works well, but some drawbacks,

C Converges, but inefficient,

D Major convergence/stability issues. (B1)

We would like to stress again that this rating is subjective and based on our experience while completing this study. Nevertheless, we hope that it might be a helpful starting point for choosing an algorithm.

[1] N. Dupuis, L. Canet, A. Eichhorn, W. Metzner, J. M. Pawlowski, M. Tissier, and N. Wschebor, Phys. Rep. **910**, 1 (2021).

[2] I. Balog, H. Chaté, B. Delamotte, M. Marohnic, and N. Wschebor, Phys. Rev. Lett. **123**, 240604 (2019).

[3] W.-j. Fu, J. M. Pawlowski, and F. Rennecke, Phys. Rev. D **101**, 054032 (2020).

[4] G. De Polsi, I. Balog, M. Tissier, and N. Wschebor, Phys. Rev. E **101**, 042113 (2020).

[5] W.-j. Fu, Commun. Theor. Phys. **74**, 097304 (2022).

[6] E. Grossi and N. Wink, arXiv:1903.09503.

[7] A. Koenigstein, M. J. Steil, N. Wink, E. Grossi, J. Braun, M. Buballa, and D. H. Rischke, Phys. Rev. D **106**, 065012 (2022).

[8] F. Ihssen, J. M. Pawlowski, F. R. Sattler, and N. Wink, arXiv:2207.12266.

[9] N. Wink, Towards the spectral properties and phase structure of QCD, Ph.D. thesis, U. Heidelberg, ITP, 2020.

[10] E. Grossi, F. J. Ihssen, J. M. Pawlowski, and N. Wink, Phys. Rev. D **104**, 016028 (2021).

[11] A. Koenigstein, M. J. Steil, N. Wink, E. Grossi, and J. Braun, Phys. Rev. D **106**, 065013 (2022).

[12] M. J. Steil and A. Koenigstein, Phys. Rev. D **106**, 065014 (2022).

[13] J. Stoll, N. Zorbach, A. Koenigstein, M. J. Steil, and S. Rechenberger, arXiv:2108.10616.

[14] F. Ihssen and J. M. Pawlowski, arXiv:2207.10057.

[15] A. Koenigstein, Non-perturbative aspects of (low-dimensional) quantum field theories, Ph.D. thesis, Goethe University, Frankfurt (main), 2023.

[16] M. Peláez and N. Wschebor, Phys. Rev. E **94**, 042136 (2016).

[17] J. Borchardt and B. Knorr, Phys. Rev. D **94**, 025027 (2016).

[18] F. Ilinca and D. Pelletier, Int. J. Therm. Sci. **38**, 560 (1999).

[19] Y. Yi, Y. Hu, and J. Zhao, Commun. Nonlinear Sci. Numer. Simul. **101**, 105895 (2021).

[20] D. F. Litim, J. M. Pawlowski, and L. Vergara, arXiv:hep-th/0602140.

[21] J. Braun, M. Leonhardt, and J. M. Pawlowski, SciPost Phys. **6**, 056 (2019).

[22] D. U. Jungnickel and C. Wetterich, Phys. Rev. D **53**, 5142 (1996).

[23] B.-J. Schaefer and J. Wambach, Nucl. Phys. **A757**, 479 (2005).

[24] R. Alkofer, A. Maas, W. A. Mian, M. Mitter, J. París-López, J. M. Pawlowski, and N. Wink, Phys. Rev. D **99**, 054029 (2019).

[25] C. Rackauckas and Q. Nie, J. Open Res. Software **5**, 15 (2017).

[26] J. Bezanson, A. Edelman, S. Karpinski, and V. B. Shah, SIAM Rev. **59**, 65 (2017).

[27] Github link, https://github.com/NicolasW1/Numerical-RG-time-integration-of-the-effective-potential-Analysis-and-Benchmark.

[28] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations* (SIAM, 1995).

[29] K. Atkinson, W. Han, and D. E. Stewart, *Numerical Solution of Ordinary Differential Equations* (John Wiley & Sons, New York, 2011).

[30] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward, ACM Trans. Math. Software (TOMS) **31**, 363 (2005).

[31] D. J. Gardner, D. R. Reynolds, C. S. Woodward, and C. J. Balos, ACM Trans. Math. Software (TOMS) **48**, 31 (2022).

[32] A. C. Hindmarsh, Sci. Comput. **1**, 55 (1983).

[33] C. Elrod, Y. Ma, K. Althaus, C. Rackauckas *et al.*, in *2022 IEEE High Performance Extreme Computing Conference (HPEC)* (IEEE, 2022), pp. 1–9.

[34] M. Hochbruck and A. Ostermann, Acta Numer. **19**, 209 (2010).

[35] M. Antonana, J. Makazaga, and A. Murua, Numerical Algorithms **76**, 861 (2017).

[36] J. Revels, M. Lubin, and T. Papamarkou, arXiv:1607.07892.

[37] M. Hosea and L. Shampine, Applied Numerical Mathematics **20**, 21 (1996).

[38] A. Kværnø, BIT **44**, 489 (2004).

[39] C. A. Kennedy, *Additive Runge-Kutta Schemes for Convection-Diffusion-Reaction Equations* (NASA, Langley Research Center, 2001).

[40] G. Wanner and E. Hairer, *Solving Ordinary Differential Equations II* (Springer Berlin Heidelberg, New York, 1996), Vol. 375.

[41] C. A. Kennedy and M. H. Carpenter, Applied Numerical Mathematics **136**, 183 (2019).

[42] E. Hairer and G. Wanner, J. Comput. Appl. Math. **111**, 93 (1999).

[43] L. F. Shampine and M. W. Reichelt, SIAM J. Sci. Comput. **18**, 1 (1997).

[44] E. A. Celaya, J. A. Aguirrezabala, and P. Chatzipantelidis, Proc. Comput. Sci. **29**, 1014 (2014).

[45] L. F. Shampine, J. Numer. Math. **10**, 291 (2002).

[46] J. Lang and J. Verwer, BIT **41**, 731 (2001).

[47] L. F. Shampine, ACM Trans. Math. Software (TOMS) **8**, 93 (1982).

[48] M. van Veldhuizen, Computing **32**, 229 (1984).

[49] P. Kaps and P. Rentrop, Numer. Math. **33**, 55 (1979).

[50] G. Steinebach, Report, Technische Hochschule Darmstadt, Fachbereich Mathematik, 1995.

[51] G. Steinebach, in *Progress in Differential-Algebraic Equations II* (Springer, New York, 2020), pp. 165–184.

[52] G. Di Marzo, MSc Mathematics Thesis, 1993.

[53] J. Rang and L. Angermann, BIT **45**, 761 (2005).

[54] L. Petzold, SIAM J. Sci. Stat. Comput. **4**, 136 (1983).

[55] K. Radhakrishnan and A. C. Hindmarsh, Description and use of LSODE, the Livermore solver for ordinary differential equations, Technical Report, 1993.

[56] C. Wetterich, Phys. Lett. B **301**, 90 (1993).