Journal of Computational Physics 501 (2024) 112787

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Numerical evaluation of oscillatory integrals via automated steepest descent contour deformation

A. Gibbs [a,*], D.P. Hewett [a], D. Huybrechs [b]

[a] Department of Mathematics, University College London, London, UK
[b] Department of Computer Science, KU Leuven, Leuven, Belgium

ARTICLE INFO

Keywords:
Oscillatory quadrature
Numerical steepest descent method
Saddle point method
Integrals with coalescing saddles

ABSTRACT

Steepest descent methods combining complex contour deformation with numerical quadrature provide an efficient and accurate approach for the evaluation of highly oscillatory integrals. However, unless the phase function governing the oscillation is particularly simple, their application requires a significant amount of a priori analysis and expert user input, to determine the appropriate contour deformation, and to deal with the non-uniformity in the accuracy of standard quadrature techniques associated with the coalescence of stationary points (saddle points) with each other, or with the endpoints of the original integration contour. In this paper we present a novel algorithm for the numerical evaluation of oscillatory integrals with general polynomial phase functions, which automates the contour deformation process and avoids the difficulties typically encountered with coalescing stationary points and endpoints. The inputs to the algorithm are simply the phase and amplitude functions, the endpoints and orientation of the original integration contour, and a small number of numerical parameters. By a series of numerical experiments we demonstrate that the algorithm is accurate and efficient over a large range of frequencies, even for examples with a large number of coalescing stationary points and with endpoints at infinity. As a particular application, we use our algorithm to evaluate cuspoid canonical integrals from scattering theory. A Matlab implementation of the algorithm is made available and is called PathFinder.

1. Introduction

In this paper we consider numerical evaluation of the integral

$$I = \int_{\Gamma} f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z, \tag{1}$$

where Γ is a contour in $\mathbb{C}$, possibly starting and/or ending at infinity, f and g are functions of a complex variable, and $\omega > 0$ is a frequency parameter. Such integrals arise in numerous application areas, particularly in wave phenomena and quantum mechanics, and are generally challenging to evaluate numerically, especially when ω is large, because the presence of the exponential factor $\mathrm{e}^{\mathrm{i}\omega g(z)}$ means that the integrand may undergo rapid oscillations and/or significant variations in amplitude along the integration contour.

* Corresponding author.
E-mail address: andrew.gibbs@ucl.ac.uk (A. Gibbs).

https://doi.org/10.1016/j.jcp.2024.112787
Received 14 July 2023; Received in revised form 6 December 2023; Accepted 18 January 2024

Available online 24 January 2024

The numerical evaluation of oscillatory integrals is a well-studied topic, and a number of different approaches have been developed for integrals of the form (1), applicable under certain assumptions on f, g and Γ. For an overview of the field and links to relevant literature we refer the reader to [7] and [1, §36.15].

When f and g are analytic, Cauchy's theorem provides the possibility of deforming the integration contour so as to make numerical evaluation easier. This is the basis of *steepest descent (SD) methods*, in which one aims to deform Γ onto a contour, or, more typically, a union of contours, which we term the *steepest descent (SD) deformation*, on which $\Re[g(z)]$ is constant, so that the exponential factor $\mathrm{e}^{\mathrm{i}\omega g(z)}$ is no longer oscillatory. By the Cauchy-Riemann equations, these contours coincide with the steepest descent curves of $-\Im[g(z)]$, and they connect endpoints of the original integration contour, valleys at infinity (sectors in which the integrand decays rapidly as $|z| \to \infty$), and *stationary points* of g, which are points $\xi \in \mathbb{C}$ at which $g'(\xi) = 0$.[1] Along each SD contour, away from stationary points the integrand typically decays exponentially, with the rate of decay increasing with increasing ω, and as $\omega \to \infty$ the value of the integral is dominated by local contributions close to the endpoints of Γ and any stationary points traversed by the SD deformation. In the *asymptotic* steepest descent method (described e.g. in [3,21]), one exploits this to obtain an asymptotic expansion for the integral, valid as $\omega \to \infty$, by performing a local Taylor expansion of the integrand around the endpoints and relevant stationary points, and reducing the local integrals along the SD contours to simpler integrals that can be expressed in terms of known special functions.

In the *numerical* steepest descent (NSD) method (described e.g. in [7, §5]) one evaluates the integrals along the SD contours numerically. This involves numerically tracing an appropriate segment of each SD contour in the SD deformation and applying suitable numerical quadrature rules to evaluate the associated contributions to the integral.

In principle, NSD is a highly accurate and efficient method for evaluating integrals of the form (1) for moderate or large ω. Indeed, under appropriate assumptions, the NSD method outputs approximations which, for a fixed number of quadrature points N, are asymptotic to (1) as $\omega \to \infty$, with the asymptotic accuracy improving with increasing N (see, e.g., [7, Thm 5.7]). Furthermore, if f and g are sufficiently well behaved it can also be the case that the NSD approximation converges to (1) as $N \to \infty$, for fixed $\omega > 0$, with a cost that remains bounded as $\omega \to \infty$.

In practice, however, applying the NSD method to an integral of the form (1) often requires significant expert user input. This is because:

(P1) Determining the SD contour deformation corresponding to a given g and Γ requires careful a priori analysis.
(P2) Parametrizing SD contours from or near stationary points, and evaluating integrals along them, is fraught with numerical difficulties, especially when stationary points are close to other stationary points or endpoints of Γ.

The issues described in (P1) and (P2) are particularly troublesome when one wishes to evaluate (1) for multiple instances of a phase function $g(z) = g(z, \mathbf{c})$ depending on a set of parameters $\mathbf{c} \in \mathbb{C}^q$. This is because, firstly, the number and location of the stationary points, and the nature of the SD deformation, have to be determined for each different value of $\mathbf{c}$, and, secondly, stationary points may coalesce as $\mathbf{c}$ approaches certain regions in parameter space, leading to a non-uniformity in the accuracy of the resulting NSD approximations.

The problem of stationary point coalescence in the context of NSD was studied in detail in [11] in the special case of the cubic phase function $g(z, c) = \frac{z^3}{3} - cz$, for $c \in \mathbb{C}$, which for $c \neq 0$ has a pair of order one stationary points which coalesce as $c \to 0$ (at $z = 0$) into a single stationary point of order two for $c = 0$.[2] In this case, the SD deformation and contour parametrization were carried out manually by analytically inverting the phase (illustrating (P1)), but the resulting integrals were found to be nearly singular for small c, leading to poor accuracy of standard NSD approximations (illustrating (P2)). It was shown in [11] how to construct a family of non-standard quadrature rules for this integral which perform uniformly well for $c \approx 0$ using complex-valued Gaussian quadrature, producing quadrature nodes that in general lie off the SD deformation. In principle, similar rules could be developed for more complicated coalescences involving higher order stationary points and/or endpoints of Γ. However, for each type of coalescence a bespoke quadrature rule would have to be developed, and a general catalogue of such rules is not yet available in the literature.

In contrast to [11], our aim is not to develop an optimized method for a specific instance of (1), but rather to present a relatively simple algorithm that can evaluate (1) accurately, for a general class of f and g, without the need for expert user input or a priori analysis, even in the case of coalescing stationary points, thus addressing problems (P1) and (P2). Our specific focus in this paper is on the case where f is entire and g is a polynomial. The extension of our approach to more general cases where f and/or g have pole or branch point singularities is the subject of ongoing research.[3] Necessarily, in aiming for generality and robustness we will sacrifice some efficiency. In particular, in contrast to standard NSD, the error in the approximations produced by our algorithm does not in general decay as $\omega \to \infty$. Nonetheless, our algorithm is designed to be rapidly convergent as $N \to \infty$ with approximately ω-independent error and ω-independent computational cost, and the fact that this is realised in practice is demonstrated by extensive numerical experiments in §5.

[1] Stationary points are often referred to as "saddle points" because they are saddle points of the functions $\Im[g(z)]$ and $\Re[g(z)]$, which cannot possess local maxima or minima by the maximum modulus principle.

[2] The *order* of a stationary point ξ is the multiplicity of ξ as a root of g'.

[3] We note that NSD-based methods for cases where f has a singularity at or close to the endpoint of the integration contour have been presented previously - see e.g. [12,9], where such methods are applied in the context of integral equation methods for high frequency scattering.

Our algorithm follows the basic principles of NSD, combining complex contour deformation with numerical quadrature. However, in contrast to standard NSD our algorithm does not trace SD contours directly from stationary points. Instead, stationary points are enclosed in a bounded "non-oscillatory region" within which the integrand is guaranteed to undergo at most a fixed number of oscillations. The original contour Γ is replaced by a "quasi-SD deformation" comprising a union of straight-line contours in the non-oscillatory region, for which numerical quadrature is straightforward, and SD contours outside the non-oscillatory region, on which standard NSD quadrature techniques can be applied. By excluding a neighbourhood of the stationary points from the region in which SD contours are traced, our algorithm avoids the problems mentioned in (P2) associated with stationary-point/stationary-point and/or stationary-point/endpoint coalescence. This not only "uniformizes" the accuracy of our algorithm compared to standard NSD, but it also enables us to tackle the problem (P1) by automating the contour deformation step. For the latter, we first perform low accuracy SD contour tracing outside the non-oscillatory region to build a graph describing the global connections (via SD contours) between the endpoints of Γ, the different components of the non-oscillatory region, and the valleys at infinity, and then determine the quasi-SD deformation using a shortest path algorithm, before refining the accuracy of the SD contour tracing at the quadrature stage.

One other problem with standard NSD is that it typically degenerates as $\omega \to 0$, because the quadrature points diverge to infinity [7, §5.2.4]. This issue has been addressed in the special case $g(z) = z$ for bounded Γ in [2,6]; however, it remains an open problem for general $g(z)$. Our algorithm is well-behaved in the limit as $\omega \to 0$ for general polynomial $g(z)$, since it reduces to standard non-oscillatory quadrature for sufficiently small ω for any bounded Γ.

Our algorithm is implemented in the open-source Matlab code "PathFinder", available at github.com/AndrewGibbs/PathFinder [8]. The basic user input to the code is a function handle for the amplitude `f`, the coefficients of the polynomial phase `g`, endpoints `a` and `b` (complex numbers, or angles in the case of infinite endpoints), the frequency parameter `omega`, and a parameter `N` controlling the number of quadrature points to be used. Approximating the integral (1) using PathFinder can be done with the following Matlab command:

```
PathFinder(a,b,f,g,omega,N,'infcontour',[A B])
```
(2)

Here `'infcontour'` is an optional input for which the user should supply a Boolean array `[A B]` (whose default value is `[false false]`) such that `A` (respectively `B`) is `true` if the endpoint `a` (resp. `b`) is infinite and `false` if it is finite. Examples of PathFinder code will be given in §5. Advanced users can also adjust a small number of other tuning parameters, whose role will be discussed during the presentation of our algorithm.

An outline of the paper is as follows. In §2 we provide a detailed description of our algorithm, first presenting an overview of the main steps, and then providing details of how each step is realised in PathFinder. In §3 we present some theoretical results underpinning our approach. In §4 we discuss some further implementation details, and in §5 we exhibit numerical results demonstrating the performance of our algorithm on a range of challenging integrals with large ω and complicated stationary point configurations.

We end this introduction by remarking that integrals with coalescing stationary points are of fundamental importance in numerous applications, including the study of optics and high frequency (short wavelength) acoustics, where they describe the wave field in the vicinity of geometrical singularities (or "catastrophes") in the classical ray-theoretic framework, Kelvin's celebrated ship-wave problem, and the theory of molecular collisions in quantum mechanics and theoretical chemistry. A catalogue of such integrals, along with links to relevant literature, can be found in [1, §36]. In §5.5 we show how PathFinder can be applied to accurately calculate these types of integrals.

2. Algorithm description

In this section we present our algorithm for the numerical approximation of (1) when f is entire[4] and g is a polynomial.

We start with some definitions and basic facts. Let

$$g(z) = \sum_{j=0}^{J} \alpha_j z^j, \tag{3}$$

for some $J \in \mathbb{N}$, $J \geq 1$, and $\alpha_j \in \mathbb{C}$, $j = 0, \dots, J$, with $\alpha_J \neq 0$. Then g has at most $J-1$ *stationary points*, which are the solutions of

$$g'(z) = \sum_{j=1}^{J} j\alpha_j z^{j-1} = 0. \tag{4}$$

We denote the set of all stationary points by $\mathcal{P}_{\text{stat}}$. We define the *valleys* at infinity to be the sectors of angular width π/J centred on the angles

$$\mathcal{V} := \left\{ \frac{(2(m-1)+1/2)\pi - \arg(\alpha_J)}{J} : \quad m = 1, \dots, J \right\}. \tag{5}$$

[4] When Γ is infinite we additionally implicitly assume that f is sufficiently well-behaved at infinity (i.e., does not grow too fast at infinity in the relevant directions) for the integral (1) to converge. Note that in many cases of interest, numerical evaluation of the integral to high accuracy after path deformation only requires f to be analytic in a small (and shrinking with increasing ω) neighbourhood of the stationary points.

These have the property that if $z = re^{i\theta}$ with $\theta \in (v - \pi/(2J), v + \pi/(2J))$ for some $v \in \mathcal{V}$ then $e^{i\omega g(z)} \to 0$ as $r \to \infty$. For each $\eta \in \mathbb{C} \setminus \mathcal{P}_{\text{stat}}$ there exists a unique SD contour γ_η beginning at η and ending either at a stationary point $\xi \in \mathcal{P}_{\text{stat}}$ or at a valley $v \in \mathcal{V}$, on which $\Re g(z) = \Re g(\eta)$ for $z \in \gamma_\eta$ (see, e.g., [4]).

We let $\mathcal{P}_{\text{endp}}$ denote the set of finite endpoints of the integration contour Γ, which could have zero, one or two elements. We assume for now that any infinite endpoint of Γ is at one of the valleys $v \in \mathcal{V}$; see §4.4 for extensions.

We now provide a high-level overview of our algorithm. The following steps will be explained in more detail in sections 2.1-2.6.

1. Compute the set of stationary points $\mathcal{P}_{\text{stat}}$ (the solutions of (4)).
2. For each $\xi \in \mathcal{P}_{\text{stat}}$, select a radius $r_\xi > 0$ for which the function $e^{i\omega g(z)}$ is considered "non-oscillatory" on the closed ball Ω_ξ of radius r_ξ centred at ξ. These balls may overlap. However, if two balls overlap significantly, indicating near coalescence, one of the stationary points (along with its associated ball) is removed from the set $\mathcal{P}_{\text{stat}}$. This removal process continues recursively until no pair of balls is judged to overlap too much.
   We call $\{\Omega_\xi\}_{\xi \in \mathcal{P}_{\text{stat}}}$ the *non-oscillatory balls*, and their union
   $$\Omega := \bigcup_{\xi \in \mathcal{P}_{\text{stat}}} \Omega_\xi \tag{6}$$
   the *non-oscillatory region*.
3. Find the local minima of $|e^{i\omega g(z)}|$ on the boundary of the non-oscillatory region Ω. We call these points *exits*, and denote by $\mathcal{P}_{\text{exit}}$ the set of all exits.
4. For each $\eta \in \mathcal{P}_{\text{exit}} \cup (\mathcal{P}_{\text{endp}} \setminus \Omega)$, trace the SD contour γ_η from η, and determine whether
   (i) γ_η enters Ω at some point $z \in \partial\Omega \setminus \{\eta\}$, or
   (ii) γ_η converges towards a valley $v \in \mathcal{V}$ without entering Ω.
   We call points $z \in \partial\Omega$ determined in case (i) *entrances*, and denote by $\mathcal{P}_{\text{entr}}$ the set of all entrances.
5. Construct a graph G with a vertex for each of the elements of $\mathcal{P}_{\text{stat}}$, $\mathcal{P}_{\text{endp}}$, $\mathcal{P}_{\text{exit}}$, $\mathcal{P}_{\text{entr}}$ and $\mathcal{V}$. Add edges between the vertices of G as follows:
   - For each $\xi \in \mathcal{P}_{\text{stat}}$, add an edge between each pair of elements of $(\mathcal{P}_{\text{stat}} \cup \mathcal{P}_{\text{endp}} \cup \mathcal{P}_{\text{exit}} \cup \mathcal{P}_{\text{entr}}) \cap \Omega_\xi$.
   - For each pair $\xi, \xi' \in \mathcal{P}_{\text{stat}}$, $\xi \neq \xi'$, for which $\Omega_\xi \cap \Omega_{\xi'} \neq \emptyset$, add an edge between ξ and ξ', if not already added in the previous step.
   - For each $\eta \in \mathcal{P}_{\text{exit}} \cup (\mathcal{P}_{\text{endp}} \setminus \Omega)$, add an edge between η and the entrance $z \in \mathcal{P}_{\text{entr}}$ or the valley $v \in \mathcal{V}$ to which the SD contour γ_η leads.

   Find the shortest path (in the graph-theoretic sense) between the vertices corresponding to the endpoints of Γ.
6. Generate quadrature nodes and weights for the evaluation of each of the contour integrals corresponding to the edges in the shortest path. For an edge between two points in the non-oscillatory region, use a straight-line contour. For an edge between an exit or an endpoint of Γ to an entrance or a valley, use a refined version of the SD contour traced in step 4. The union of all the contours corresponding to the edges of the shortest path defines the "quasi-SD deformation" of the original integration contour. Finally, use the quadrature nodes and weights to approximate the integrals over the contours in the quasi-SD deformation and sum them to obtain an approximation of the original integral (1).

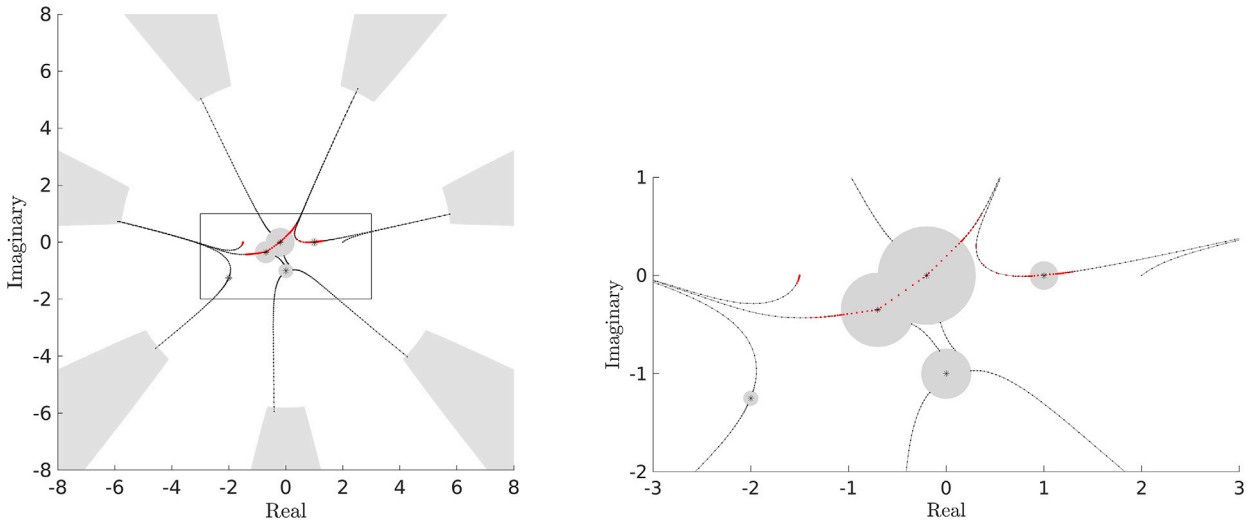
In Figs. 2.1 and 2.2 we illustrate the outcome of the above steps for the particular choice of phase function

$$\begin{aligned} g(z) = &\frac{z^7}{7} + z^6\left(\frac{7}{20} + \frac{13}{30}\mathrm{i}\right) + z^5\left(-\frac{1047}{2000} + \frac{543}{1000}\mathrm{i}\right) + z^4\left(-\frac{4409}{8000} - \frac{5077}{8000}\mathrm{i}\right) \\ &+ z^3\left(\frac{711}{2000} - \frac{4441}{6000}\mathrm{i}\right) + z^2\left(\frac{237}{800} - \frac{207}{800}\mathrm{i}\right) + z\left(\frac{63}{1000} - \frac{77}{2000}\mathrm{i}\right) \end{aligned} \tag{7}$$

and the parameters $\omega = 40$, $a = -1.5$, $b = 2$, $N = 10$, using the default parameter set for PathFinder (see Table 4.1). For this choice of g there is one order 2 stationary point and 4 order one stationary points. In Fig. 2.1 we plot these stationary points, along with their non-oscillatory balls, and the SD contours traced from the exits. Such plots can be generated in PathFinder by adding the optional `'plot'` flag. The ball centred at the stationary point $\xi = -\mathrm{i}$ contains two entrances, reached by SD contours from the balls above. In Fig. 2.2 we plot the graph G, using the optional PathFinder input flag `'plot graph'`. This graph, in addition to edges corresponding to the SD contours shown in Fig. 2.1, contains edges corresponding to contours between points in the non-oscillatory region, including connections within the two overlapping balls. The shortest path between a and b, which is highlighted with thick lines in Fig. 2.2, corresponds to the quasi-SD deformation, the integral over which is equal to (1) by Cauchy's Theorem. The integral is discretised using N quadrature points on each contour in the quasi-SD deformation that makes a non-negligible contribution to the integral (see §2.6) - these points are plotted in Fig. 2.1 in red.

The process of computing all the SD contours and the selection of a subset thereof via the shortest path algorithm addresses problem (P1). Surrounding stationary points by balls, and only tracing SD contours outside the balls, means that we avoid having to determine the local structure of the SD contours and compute integrals along them near stationary points, addressing problem (P2).

In the following subsections we provide further details of how we carry out the steps outlined above in PathFinder.



(a) SD contours and quasi-SD deformation (b) Zoom of boxed region near the origin

Fig. 2.1. Output of algorithm when applied with phase (7), $\omega = 40$, $a = -1.5$, $b = 2$, $N = 10$, and the default parameter set (see Table 4.1). Here we observe stationary points $\xi \in \mathcal{P}_{\text{stat}}$ (black stars) surrounded by balls Ω_ξ (grey) whose union is the non-oscillatory region Ω, SD contours (black lines) traced from exits $\eta \in \mathcal{P}_{\text{exit}}$ and finite endpoints $\eta \in \mathcal{P}_{\text{endp}}$ to either valleys $v \in \mathcal{V}$ at infinity or to entrances $z \in \mathcal{P}_{\text{entr}}$, and quadrature points (red points) allocated along the appropriate contours in the quasi-SD deformation. The "region of no return" R_v (see §3.2) around each of the valleys $v \in \mathcal{V}$ at infinity is also shaded grey. (For interpretation of the colours in the figure(s), the reader is referred to the web version of this article.)

2.1. Step 1 - Computing stationary points

Computing the stationary points of g (the roots of $g'(z)$) requires us to find the complex roots of the polynomial (4). In our implementation we compute stationary points using the Matlab `roots` command, which applies a companion matrix approach. We note that obtaining highly accurate values for the positions of stationary points is not critical to our algorithm, since the stationary points are enclosed in the non-oscillatory region and we never trace SD contours from them. Indeed, the difficulty in distinguishing numerically between multiple roots and roots of higher order contributes to the motivation for considering such non-oscillatory regions.

2.2. Step 2 - Defining the non-oscillatory region

The non-oscillatory region Ω was defined in (6) to be a union of balls centred at the elements of $\mathcal{P}_{\text{stat}}$. We choose the radii of the balls as follows. First fix some user-defined constant $C_{\text{ball}} > 0$. Then, given $\xi \in \mathcal{P}_{\text{stat}}$, define

$$r_\xi := \max\{r > 0 : |z - \xi| \le r \Rightarrow \omega|g(z) - g(\xi)| \le C_{\text{ball}}\}. \tag{8}$$

This definition enforces an upper bound on the number of oscillations within each ball. Accordingly, the region Ω shrinks to the stationary points as $\omega \to \infty$ and expands to fill the whole complex plane as $\omega \to 0$.

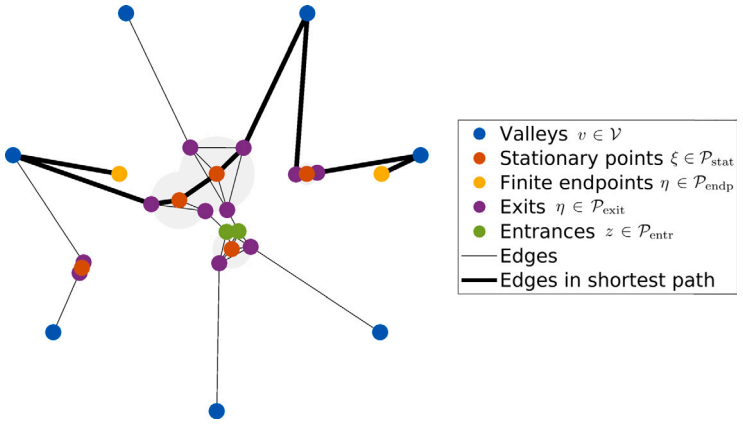
In our implementation we approximate r_ξ numerically as follows. Let $N_{\text{ball}} \in \mathbb{N}$ be a user-defined parameter. For each $n \in \{1, \dots, N_{\text{ball}}\}$ we consider the ray $\{z = \xi + re^{i2\pi n/N_{\text{ball}}}, r > 0\}$, and compute the smallest positive root $r_n > 0$ of the function $u_n(r) := \omega^2|g(\xi + re^{i2\pi n/N_{\text{ball}}}) - g(\xi)|^2 - C_{\text{ball}}^2$, which is a polynomial in r of degree $2J$. For this root-finding problem we use the Matlab `roots` command; in case this command produces no positive real roots (because of stability issues) we resort to a bisection approach instead. We then take as our approximation to r_ξ the positive number $\min_{n\in\{1,\dots,N_{\text{ball}}\}} r_n$.

When elements of $\mathcal{P}_{\text{stat}}$ are close it is natural to amalgamate their respective non-oscillatory balls. To do this systematically we adopt an iterative approach. Let $\delta_{\text{ball}} > 0$ be a user-defined parameter.

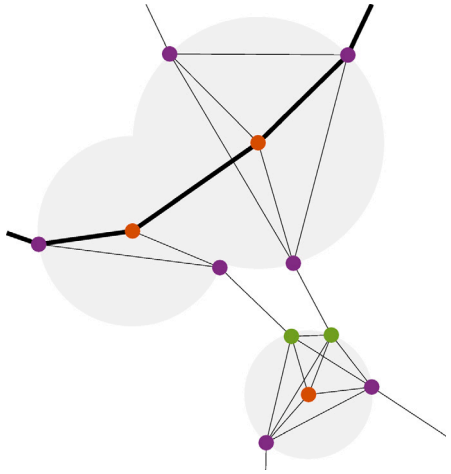
- For each pair $\xi_1, \xi_2 \in \mathcal{P}_{\text{stat}}$ compute

$$d_{\xi_1,\xi_2} := |\xi_1 - \xi_2| / \max(r_{\xi_1}, r_{\xi_2}).$$

- If $\min_{\xi_1,\xi_2} d_{\xi_1,\xi_2} < \delta_{\text{ball}}$ let ξ_1, ξ_2 be a pair realising the minimum. Remove from $\mathcal{P}_{\text{stat}}$ whichever of ξ_1, ξ_2 has the smaller associated ball radius (or choose arbitrarily between them if $r_{\xi_1} = r_{\xi_2}$).
- Repeat the previous step until either $\min_{\xi_1,\xi_2} d_{\xi_1,\xi_2} \ge \delta_{\text{ball}}$, or there is only one element of $\mathcal{P}_{\text{stat}}$ remaining.



(a) Graph G



(b) Zoom of the graph G

Fig. 2.2. The graph G corresponding to the problem considered in Fig. 2.1. The thick line represents the shortest path between the endpoints of Γ, which in this case are both finite. The balls (shaded grey) are included for ease of comparison with Fig. 2.1. The lower figure zooms in on the centre of the upper figure, showing the multiple edges that are constructed inside the balls.

2.3. Step 3 - Determining the exits

The exits associated with each $\xi \in \mathcal{P}_{\text{stat}}$ are defined to be the local minima on $\partial\Omega_\xi \setminus \bigcup_{\xi' \in \mathcal{P}_{\text{stat}}, \xi' \neq \xi} \Omega^\circ_{\xi'}$ of the function $|e^{i\omega g(z)}|$, equivalently of the function $-\Im g(z)$.

For each $\xi \in \mathcal{P}_{\text{stat}}$ the function $-\Im g(z)$ restricted to the $\partial\Omega_\xi$ is a trigonometric polynomial. Using this fact, in our implementation we determine the local minima of $-\Im g(z)$ on $\partial\Omega_\xi$ by finding the roots of the derivative of $-\Im g(z)$ in the angular direction (which is also a trigonometric polynomial) by the companion matrix approach of [5, §2.2], and keep only the real roots corresponding to local minima. We discard all those minima corresponding to points inside $\bigcup_{\xi' \in \mathcal{P}_{\text{stat}}, \xi' \neq \xi} \Omega^\circ_{\xi'}$, and add the remaining minima to the set $\mathcal{P}_{\text{exit}}$.

2.4. Step 4 - Tracing the SD contours

Given $\eta \in \mathcal{P}_{\text{exit}} \cup (\mathcal{P}_{\text{endp}} \setminus \Omega)$, the SD contour γ_η beginning at η is the unique curve on which $\Re g(z)$ is constant, with $-\Im g(z)$ decreasing along γ_η. It can be parametrized in terms of a parameter $p \geq 0$ as $z = h_\eta(p)$, where $h_\eta(p)$ is defined implicitly by

$$g(h_\eta(p)) = g(\eta) + ip, \qquad h_\eta(0) = \eta. \tag{9}$$

Differentiating (9) with respect to p gives

$$h'_\eta(p) = \frac{i}{g'(h_\eta(p))} =: F(h_\eta(p)), \qquad h_\eta(0) = \eta, \tag{10}$$

which is a first order ODE initial value problem for $h_\eta(p)$. By solving (10) numerically one can trace the contour γ_η until it either (i) enters the non-oscillatory region Ω, or (ii) one can be sure that it will tend to a valley $v \in \mathcal{V}$, without entering Ω. For (ii) we appeal

to the theoretical result in Theorem 3.3, which provides a "region of no return" R_υ associated with each valley $\upsilon \in \mathcal{V}$ for which it is guaranteed that if an SD contour enters R_υ it will never leave R_υ, and will converge to υ.

Staying away from stationary points ensures that the factor $1/g'$ in the right-hand side of (10) does not get too large.

In our implementation we trace the SD contour using a predictor-corrector approach, combining a forward Euler step for (10) and a Newton iteration for (9), to generate approximations $h_\eta^{(n)} \approx h_\eta(p_n)$ on a mesh $0 = p_0 < p_1 < p_2 < \ldots < p_{n_{\max}}$, where the total number of steps $n_{\max}$ is determined as part of the algorithm, as discussed below.

As the initial value we take $h_\eta^{(0)} = \eta$. Then, given $h_\eta^{(n)}$, to compute $h_\eta^{(n+1)}$ we first apply a forward Euler step for the ODE (10), with adaptive step length

$$p_{n+1} - p_n = \delta_{\mathrm{ODE}} \min\left(2\frac{|g'(h_\eta^{(n)})|^2}{|g''(h_\eta^{(n)})|}, |g'(h_\eta^{(n)})| \operatorname{dist}(h_\eta^{(n)}, \mathcal{P}_{\mathrm{stat}}) \right),$$

where $\delta_{\mathrm{ODE}} \in (0,1)$ is a user-specified parameter. The first argument of the minimum is included to ensure stability of the solver - note that $F'(h) = -\frac{\mathrm{i}g''(h)}{(g'(h))^2}$ and we might expect instability if the local step length were as large as $2/|F'(h)| = 2\frac{|g'(h)|^2}{|g''(h)|}$. The second argument is included to ensure that the solver "slows down" as it approaches the non-oscillatory region Ω, so that we can detect whether γ_η enters Ω or not. To ensure that $|h_\eta^{(n+1)} - h_\eta^{(n)}| \le \delta_{\mathrm{ODE}} d$, where $d := \operatorname{dist}(h_\eta^{(n)}, \mathcal{P}_{\mathrm{stat}}) = \min_{\xi \in \mathcal{P}_{\mathrm{stat}}} |h_\eta^{(n)} - \xi|$, we require that $p_{n+1} - p_n \le \frac{\delta_{\mathrm{ODE}} d}{|F(h_\eta^{(n)})|} = \delta_{\mathrm{ODE}} d |g'(h_\eta^{(n)})|$. This also ensures that $h_\eta^{(n+1)}$ remains far enough from $\mathcal{P}_{\mathrm{stat}}$, so that (10) doesn't get too large.

After each forward Euler step, we correct $h_\eta^{(n+1)}$ by running a Newton iteration to enforce (9) (with $p = p_{n+1}$ fixed), until the Newton step size $|\frac{g(h_\eta^{(n+1)}) - g(\eta) - \mathrm{i}p_{n+1}}{g'(h_\eta^{(n+1)})}|$ is smaller than $\delta_{\mathrm{coarse}} \operatorname{dist}(h_\eta^{(n+1)}, \mathcal{P}_{\mathrm{stat}})$, for some user-specified tolerance $\delta_{\mathrm{coarse}} > 0$.

We repeat this process for $n = 0, 1, 2, \ldots$ until either

(i) $h_\eta^{(n)} \in \Omega_\xi$ for some $\xi \in \mathcal{P}_{\mathrm{stat}}$, in which case we add $z = h_\eta^{(n)}$ to the set $\mathcal{P}_{\mathrm{entr}}$ of entrances. Note that in general the point $z = h_\eta^{(n)}$ will lie inside Ω_ξ° rather than on $\partial\Omega_\xi$, but will be closer to $\partial\Omega_\xi$ the smaller δ_{ODE} is;
or

(ii) $h_\eta^{(n)} \in R_\upsilon$ for some $\upsilon \in \mathcal{V}$, in which case, by Theorem 3.3, γ_η converges to the valley υ. Here the "region of no return" R_υ is defined by

$$R_\upsilon := \{ z \in \mathbb{C} : |\arg z - \upsilon|_{2\pi} < \pi/(2J) \text{ and } G(|z|, |\arg z - \upsilon|_{2\pi}) > 0 \}, \tag{11}$$

where

$$|\theta|_{2\pi} := \min_{m \in \mathbb{Z}} |\theta - 2\pi m|, \tag{12}$$

and, for $r > 0$ and $\theta \in (0, \pi/(2J))$,

$$G(r,\theta) := J|\alpha_J| r^{J-1} \min\left(1/\sqrt{2}, \cos J\theta\right) - \sum_{j=1}^{J-1} j|\alpha_j| r^{j-1}. \tag{13}$$

For further explanation of the meaning of R_υ see §3.2 below.

A necessary condition for a point z to lie in R_υ is that $|z| \ge r_*$, where $r_* > 0$ is the unique positive solution of the polynomial equation $G(r_*, \pi/(4J)) = 0$, i.e. the solution of

$$\frac{J|\alpha_J| r_*^{J-1}}{\sqrt{2}} = \sum_{j=1}^{J-1} j|\alpha_j| r_*^{j-1}.$$

Having found r_* once and for all (using the Matlab `roots` command), to check that a point z lies in R_υ we first check that $|z| \ge r_*$. If so, we then check that $|\arg z - \upsilon|_{2\pi} < \pi/(2J)$. If so, we then check that $G(|z|, |\arg z - \upsilon|_{2\pi}) > 0$. The point of introducing r_* is so that we don't compute $G(|z|, |\arg z - \upsilon|_{2\pi})$ unless absolutely necessary.

In either case, tracing of the SD contour stops and we set $n_{\max} = n$ for this contour.

2.5. Step 5 - Finding the shortest path

The construction of the graph G requires no further explanation. To find the shortest path in G between the endpoints of the original contour Γ we apply the standard Dijkstra shortest path algorithm [15, §10.6.2].

2.6. Step 6 - Evaluating the contour integrals

The quasi-SD contour deformation corresponding to the graph-theoretic shortest path between the endpoints of G calculated during step 5 involves integrals over three types of contour:

Type 1: Straight line contours between points in the non-oscillatory region;
Type 2: Infinite SD contours from exits/endpoints to valleys;
Type 3: Finite SD contours from exits/endpoints to entrances.

Some of these contours will make a larger contribution to the value of the original integral (1) than others. It is natural to neglect contours that make a negligibly small contribution. In our implementation, we only compute the contribution from a contour γ in the quasi-SD deformation if at least one of the finite endpoints η of γ satisfies $|\mathrm{e}^{\mathrm{i}\omega g(\eta)}|/M > \delta_{\text{quad}}$, where $\delta_{\text{quad}} \geq 0$ is a small, user-specified parameter and

$$M := \max |\mathrm{e}^{\mathrm{i}\omega g(\xi)}|,$$

where the maximum is taken over all $\xi \in \mathcal{P}_{\text{stat}} \cup \mathcal{P}_{\text{endp}} \cup \mathcal{P}_{\text{exit}}$ appearing in the shortest path corresponding to the quasi-SD deformation.

In our implementation, for Type 1 contours we use Gauss-Legendre quadrature, for Type 2 contours we use either Gauss-Laguerre quadrature (which is the default choice in PathFinder) or truncated Gauss-Legendre quadrature, and for Type 3 contours we use (possibly truncated) Gauss-Legendre quadrature, as detailed below. By default our implementation uses the same number N of quadrature points on each contour in the quasi-SD deformation whose contribution we compute, regardless of the type of integral (we comment on this in §3.3.4). Accordingly, if N_{cont} is the number of these contours then the total number of quadrature points used in the algorithm, N_{tot}, is given by

$$N_{\text{tot}} = N N_{\text{cont}}. \tag{14}$$

2.6.1. Evaluation of integrals over Type 1 contours

Let $z_0, z_1 \in \Omega$, and let γ be the straight-line contour in $\mathbb{C}$ starting at z_0 and ending at z_1, parametrized by

$$z_{[z_0,z_1]}(t) = \frac{1}{2}\Big((z_1 - z_0)t + (z_0 + z_1)\Big), \qquad t \in [-1, 1]. \tag{15}$$

Given $N \in \mathbb{N}$, let t_m^{Leg} and w_m^{Leg}, for $m = 1, \ldots, N$, denote the nodes and weights for standard N-point Gauss-Legendre quadrature on $[-1, 1]$. Our quadrature approximation to the integral over γ is then:

$$\int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z \approx \frac{z_1 - z_0}{2}\sum_{m=1}^{N} w_m^{\text{Leg}} f(z_{[z_0,z_1]}(t_m^{\text{Leg}}))\mathrm{e}^{\mathrm{i}\omega g(z_{[z_0,z_1]}(t_m^{\text{Leg}}))}. \tag{16}$$

2.6.2. Evaluation of integrals over Type 2 contours

Let $\eta \in \mathcal{P}_{\text{exit}} \cup (\mathcal{P}_{\text{endp}} \setminus \Omega)$ be such that the SD contour γ from η leads to a valley. Parametrizing γ by (9), for $p \in [0, \infty)$, noting (10), and rescaling $p = \tilde{p}/\omega$, we have

$$\int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z = \frac{\mathrm{e}^{\mathrm{i}\omega g(\eta)}}{\omega}\int_0^\infty \tilde{f}(\tilde{p})\mathrm{e}^{-\tilde{p}}\,\mathrm{d}\tilde{p}, \tag{17}$$

where

$$\tilde{f}(\tilde{p}) := f(h_\eta(\tilde{p}/\omega))h_\eta'(\tilde{p}/\omega) = \mathrm{i}\frac{f(h_\eta(\tilde{p}/\omega))}{g'(h_\eta(\tilde{p}/\omega))}.$$

By tracing contours outside of Ω, the contours remain a positive distance from $\mathcal{P}_{\text{stat}}$. This ensures that $\tilde{f}$ is analytic in a complex neighbourhood of $[0, \infty)$.

By default, PathFinder evaluates the integral on the right-hand side of (17) by Gauss-Laguerre quadrature. Let t_m^{Lag} and w_m^{Lag}, for $n = 1, \ldots, N$, denote the standard Gauss-Laguerre nodes and weights on $[0, \infty)$. Our quadrature approximation to the integral over γ is then:

$$\int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z \approx \frac{\mathrm{e}^{\mathrm{i}\omega g(\eta)}}{\omega}\sum_{m=1}^{N} w_m^{\text{Lag}}\tilde{f}(t_m^{\text{Lag}}). \tag{18}$$

To evaluate $\tilde{f}(t_m^{\text{Lag}})$ we need accurate computations of $h_\eta(t_m^{\text{Lag}}/\omega)$ for $m = 1, \ldots, N$. For this, for each m we run a Newton iteration on (9) with $p = t_m^{\text{Lag}}/\omega$ fixed, until the magnitude of the increment is smaller than a user-specified tolerance $\delta_{\text{fine}} > 0$. Typically we take δ_{fine} to be considerably smaller than the tolerance δ_{coarse} used in the Newton iteration in step 4, since when carrying out quadrature we require higher accuracy in our approximation of the SD contour than is required for determining the global structure of the quasi-SD deformation in step 4. As the initial guess for the Newton method we use a piecewise linear interpolant of the points $\{(p_0, h_\eta^{(0)}), (p_1, h_\eta^{(1)}), \ldots, (p_{n_{\max}}, h_\eta^{(n_{\max})})\}$ computed in step 4, where $n_{\max}$ denotes the total number of steps taken in the ODE solve in step 4 before the contour tracing algorithm terminated. If $p_{n_{\max}} < t_N^{\text{Lag}}/\omega$ then before running the Newton iteration we first need to

restart the contour tracing algorithm of step 4 to extend the SD contour until $p_{n_{\max}} \geq t_N^{\mathrm{Lag}}/\omega$, so that there are points to interpolate between.

As an alternative, one can evaluate the integral over a Type 2 contour using truncated Gauss-Legendre quadrature, as suggested in [18]. To activate this alternative in PathFinder one should add the optional input `'inf quad rule', 'legendre'`. In this case we truncate the integral to

$$\int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z \approx \frac{\mathrm{e}^{\mathrm{i}\omega g(\eta)}}{\omega}\int_0^P \tilde{f}(\tilde{p})\mathrm{e}^{-\tilde{p}}\,\mathrm{d}\tilde{p}, \tag{19}$$

for some $P > 0$, then apply Gauss-Legendre quadrature on $[0, P]$, to obtain the approximation

$$\int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z \approx \frac{P\mathrm{e}^{\mathrm{i}\omega g(\eta)}}{2\omega}\sum_{m=1}^{N} w_m^{\mathrm{Leg}}\tilde{f}(z_{[0,P]}(t_m^{\mathrm{Leg}}))\mathrm{e}^{-z_{[0,P]}(t_m^{\mathrm{Leg}})}, \tag{20}$$

where we compute $h_\eta(z_{[0,P]}(t_m^{\mathrm{Leg}}/\omega))$ (which is required for the evaluation of $\tilde{f}(z_{[0,P]}(t_m^{\mathrm{Leg}}))$) by the same Newton iteration discussed above for $h_\eta(t_m^{\mathrm{Lag}}/\omega)$. For the truncation point P we take

$$P = L, \tag{21}$$

where

$$L := -\log\left(\delta_{\mathrm{quad}} M/|\mathrm{e}^{\mathrm{i}\omega g(\eta)}|\right), \tag{22}$$

which describes the point at which the magnitude of the exponential part of the integrand drops below δ_{quad} times its maximum value M on the quasi-SD deformation.

2.6.3. Evaluation of integrals over Type 3 contours

Let $\eta \in \mathcal{P}_{\mathrm{exit}} \cup (\mathcal{P}_{\mathrm{endp}} \setminus \Omega)$ be such that the SD contour γ from η leads to an entrance $z \in \mathcal{P}_{\mathrm{entr}}$. In this case we apply (possibly truncated) Gauss-Legendre quadrature as in formulas (19) and (20), but now with

$$P = \min(p_{n_{\max}}/\omega, L), \tag{23}$$

where $p_{n_{\max}}$ is defined as in §2.4 and L is defined as in §2.6.2.

In the case where the minimum is attained by $p_{n_{\max}}/\omega$, so that the whole contour is considered, a potential inconsistency arises, because the application of the higher accuracy Newton iteration described in §2.6.2 for the calculation of $h_\eta(z_{[0,P]}(t_m^{\mathrm{Leg}}/\omega))$ corresponds implicitly to a slight shifting of the endpoint of the contour γ away from the entrance $z = h_\eta^{(n_{\max})}$ added to the graph G in step 5. To avoid this inconsistency, in our implementation, in step 4, whenever the contour tracing terminates in case (i), we run a Newton iteration on the final point $h_\eta^{(n_{\max})}$ with the high accuracy tolerance δ_{fine}, before adding it to the list of entrances $\mathcal{P}_{\mathrm{entr}}$. Note that this may mean that $h_\eta^{(n_{\max})}$ lies very slightly outside Ω.

3. Theoretical results

In this section we collect some theoretical results that motivate the design of our algorithm.

3.1. Removal of stationary points

In §2.2 we described our algorithm for removing stationary points from the set $\mathcal{P}_{\mathrm{stat}}$ when they are close. When removing stationary points and their associated non-oscillatory balls, we need to ensure that the removed stationary points still lie inside one of the remaining non-oscillatory balls, so that we don't encounter any stationary points along the trajectory in our ODE solve for the SD contour tracing (see the discussion in §3.3.2 below). In this section we provide a sufficient condition on the parameter δ_{ball} for this to be guaranteed.

Proposition 3.1. *Suppose that in the removal algorithm of §2.2, n stationary points have been removed from $\mathcal{P}_{\mathrm{stat}}$. Then for any stationary point ξ that was removed, there exists $\xi' \in \mathcal{P}_{\mathrm{stat}}$ such that $|\xi - \xi'| \leq n\delta_{\mathrm{ball}} r_{\xi'}$.*

Proof. We proceed by induction on n. The result is trivially true for $n = 0$. Assume that it is true after the removal of n points, and suppose that the $(n+1)$st point is now to be removed. Let ξ_1, ξ_2 denote the pair of points selected as realising $\min_{\xi_1,\xi_2} d_{\xi_1,\xi_2}$, and without loss of generality suppose that ξ_2 is the point to be removed (so that $r_{\xi_1} \geq r_{\xi_2}$). Then $|\xi_2 - \xi_1| \leq \delta_{\mathrm{ball}} r_{\xi_1} \leq (n+1)\delta_{\mathrm{ball}} r_{\xi_1}$, so the claimed property holds for ξ_2. Furthermore, by the inductive hypothesis, for each point ξ previously removed, there exists $\xi' \in \mathcal{P}_{\mathrm{stat}}$ such that $|\xi - \xi'| \leq n\delta_{\mathrm{ball}} r_{\xi'}$. If $\xi' \neq \xi_2$ then ξ' will still be present in $\mathcal{P}_{\mathrm{stat}}$ after the removal of ξ_2, and $|\xi - \xi'| \leq n\delta_{\mathrm{ball}} r_{\xi'} \leq (n+1)\delta_{\mathrm{ball}} r_{\xi'}$. On the other hand, if $\xi' = \xi_2$ then ξ' will not be present in $\mathcal{P}_{\mathrm{stat}}$ after the removal of ξ_2, but ξ_1 will be, and by the triangle inequality

$$|\xi - \xi_1| \le |\xi - \xi_2| + |\xi_2 - \xi_1| \le n\delta_{\text{ball}} r_{\xi_2} + \delta_{\text{ball}} r_{\xi_1} \le (n+1)\delta_{\text{ball}} r_{\xi_1},$$

completing the inductive step. □

As a consequence, we obtain the following.

Corollary 3.2. *If $J > 2$ and $0 < \delta_{\text{ball}} \le 1/(2(J-2))$ then, after the removal algorithm has run, for every stationary point ξ there exists $\xi' \in \mathcal{P}_{\text{stat}}$ such that $\xi \in \Omega_{\xi'}$ and $\operatorname{dist}(\xi, \partial\Omega_{\xi'}) \ge r_{\xi'}/2$.*

3.2. Region of no return for SD contours

The following result establishes a *region of no return*: once an SD contour enters this region, we can say with certainty which valley it will converge to. The idea behind this result is that in the region of no return the highest degree term $\alpha_J z^J$ of the polynomial g is sufficiently dominant over the lower degree terms that the SD contours inside the region converge to the same valley as those corresponding to the monomial phase $\alpha_J z^J$.

Theorem 3.3 *(Region of no return). Let g, $\mathcal{V}$ and R_v, for $v \in \mathcal{V}$, be as in (3), (5) and (11). The regions R_v, $v \in \mathcal{V}$, contain no stationary points of g. Furthermore, if an SD contour enters R_v for some $v \in \mathcal{V}$, it never leaves R_v.*

Proof. That R_v contains no stationary points follows because if $G(r,\theta) > 0$ then

$$J|\alpha_j||z|^{J-1} > \sum_{j=1}^{J-1} j|\alpha_j||z|^{j-1} \ge \left| \sum_{j=1}^{J-1} j\alpha_j z^{j-1} \right|,$$

so that $g'(z) \ne 0$.

Now fix $v \in \mathcal{V}$. Given $\theta' \in (0, \pi/(2J))$ and $R > 0$ we define the sector

$$S_v(R,\theta') := \{z \in \mathbb{C} : |\arg z - v|_{2\pi} < \theta' \text{ and } |z| > R\},$$

with $|\cdot|_{2\pi}$ defined as in (12). We also define the function

$$\tilde{G}(R,\theta') := |J||\alpha_J| R^{J-1} \min\left(\sin J\theta', \cos J\theta'\right) - \sum_{j=1}^{J-1} j|\alpha_j| R^{j-1}, \tag{24}$$

which for each fixed θ' is a polynomial in R of degree $J-1$.

We claim that if $\theta' \in (0, \pi/(2J))$ and $\tilde{G}(R,\theta') > 0$, then if an SD contour enters $S_v(R,\theta')$ it never leaves $S_v(R,\theta')$. To prove this, we show that if an SD contour intersects $\partial S_v(R,\theta')$ then the direction of descent always points into $S_v(R,\theta')$. Since $\partial S_v(R,\theta')$ is the union of the sets

$$\{z \in \mathbb{C} : |\arg z - v|_{2\pi} \le \theta' \text{ and } |z| = R\}$$

and

$$\{z \in \mathbb{C} : |\arg z - v|_{2\pi} = \theta' \text{ and } |z| \in [R, \infty)\},$$

it suffices to show that, in polar coordinates (r,θ),

$$\mathfrak{I}\frac{\partial g}{\partial r} > 0, \quad \text{for } |\theta - v|_{2\pi} \le \theta' \text{ and } r = R, \tag{25}$$

$$\mp\,\mathfrak{I}\frac{1}{r}\frac{\partial g}{\partial \theta} > 0, \quad \text{for } \theta = v \pm \theta' \pmod{2\pi} \text{ and } r \ge R. \tag{26}$$

For (25), let $|\theta - v|_{2\pi} \le \theta'$. Since

$$\frac{\partial g(re^{i\theta})}{\partial r} = \sum_{j=1}^{J} j\alpha_j e^{ij\theta} r^{j-1}$$

and $\mathfrak{I}[\alpha_J e^{iJ\theta}] = |\alpha_J| \cos(J|\theta - v|_{2\pi})$ (using the definition of v) we have that

$$\mathfrak{I}\frac{\partial g(re^{i\theta})}{\partial r} \ge J|\alpha_J| r^{J-1} \cos(J|\theta - v|_{2\pi}) - \sum_{j=1}^{J-1} j|\alpha_j| r^{j-1},$$

so a sufficient condition for (25) to hold is that

$$J|\alpha_J|R^{J-1}\cos(J\theta') - \sum_{j=1}^{J-1} j|\alpha_j|R^{j-1} > 0. \tag{27}$$

For (26), let $\theta = v \pm \theta' \pmod{2\pi}$. Since

$$\frac{1}{r}\frac{\partial g(re^{i\theta})}{\partial\theta} = \sum_{j=1}^{J} ij\alpha_j e^{ij\theta} r^{j-1},$$

and $\Im[i\alpha_J e^{iJ\theta}] = \Im[i\alpha_J e^{iJ(v\pm\theta')}] = \mp|\alpha_J|\sin(J\theta')$ we have that

$$\mp\Im\frac{1}{r}\frac{\partial g(re^{i\theta})}{\partial\theta}\bigg|_{\theta=v\pm\theta'} \geq J|\alpha_J|r^{J-1}\sin(J\theta') - \sum_{j=1}^{J-1} j|\alpha_j|r^{j-1} =: \phi(r).$$

The function $\phi(r)$ has the property that if $R > 0$ and $\phi(R) > 0$ then $\phi(r) > 0$ for all $r \geq R$. To see this, note that

$$\phi(r) = r^{J-1}\left(J|\alpha_J|\sin(J\theta') - \sum_{j=1}^{J-1} j|\alpha_j|r^{j-J}\right),$$

and that the term in brackets is a strictly decreasing function of r, which tends to $-\infty$ as $r \to 0$ and to $J|\alpha_J|\sin(J\theta') > 0$ as $r \to \infty$. Hence a sufficient condition for (26) is that $\phi(R) > 0$, i.e.

$$J|\alpha_J|R^{J-1}\cos(J\theta') - \sum_{j=1}^{J-1} j|\alpha_j|R^{j-1} > 0. \tag{28}$$

Since the assumption $\tilde{G}(R,\theta') > 0$ implies both (27) and (28), our claim is proved.

The statement of the theorem then follows by noting that the region R_v is the union of all the sectors $S_v(R,\theta')$ such that $0 < \theta' \leq \pi/(2J)$ and $\tilde{G}(R,\theta') > 0$. We note that if $0 < \theta' < \pi/(4J)$ then $\sin J\theta' < \sin\pi/4$, so that if $\tilde{G}(R,\theta') > 0$ then $\tilde{G}(R,\pi/(4J)) > 0$. This implies that the union can actually be taken over $\pi/(4J) \leq \theta' < \pi/(2J)$ only, justifying the definition of the function G in (13). □

3.3. Quadrature error

In §2.2 we defined the non-oscillatory region as a union of balls on which the exponential $e^{i\omega g(z)}$ undergoes a bounded number of oscillations. In this section we show that the definition (8) strikes a balance between the accuracy of our quadrature approximations to the integrals outside and inside this region.

We note that, as already mentioned in §1, in contrast to standard NSD approximations the error in our method does not in general decay as $\omega \to \infty$. In the special case where we are tracing a single infinite SD contour from a fixed (ω-independent) endpoint to a valley at infinity, without stationary points nearby, the error in our method would indeed decay as $\omega \to \infty$ in the same way as for standard NSD (e.g., [7, Thm. 5.5]), up to errors introduced by our ODE for SD contour tracing. However, in the general case our algorithm traces steepest descent contours from points on the edges of the non-oscillatory balls, whose radii depend in a non-trivial way on ω and the distribution of stationary points, so the standard NSD theory does not apply. Also, the NSD-style Gauss-Laguerre quadrature along SD contours forms just one step in our algorithm, and other steps (such as the quadrature inside the non-oscillatory region) have their own non-trivial ω-dependence. As a result, the ω-dependence of the overall error is much harder to predict for our algorithm than for standard NSD in the simplest setting. Nonetheless, in practice we observe errors that remain bounded as $\omega \to \infty$ for fixed N. We shall provide some theoretical justification for this below, and back this up with extensive numerical evidence in §5.

3.3.1. Quadrature in the non-oscillatory region

The Type 1 straight line contour integrals between points in the non-oscillatory region are evaluated using Gauss-Legendre quadrature, as detailed in §2.6.1. To assess the accuracy of this we note the following theorem, which is a simple consequence of the standard error analysis presented in [17, Chap. 19].

Theorem 3.4. *Let $z_0, z_1 \in \mathbb{C}$. Suppose that γ is a straight-line contour in $\mathbb{C}$ starting at z_0 and ending at z_1 and that there exists $\rho > 0$, $C > 0$ and $\xi_* \in \mathbb{C}$ such that f is analytic and bounded in $z_{[z_0,z_1]}(B_\rho)$, where B_ρ is a standard Bernstein ellipse (relative to $[-1,1]$) and $z_{[z_0,z_1]}$ is defined as in (15), and*

$$\omega|g(\xi_*) - g(z)| \leq C, \qquad z \in z_{[z_0,z_1]}(B_\rho). \tag{29}$$

Let I and Q denote the left- and right-hand sides of (16), respectively. Then, for some $\tilde{C} > 0$, depending only on ρ,

$$|I - Q| \leq \tilde{C}|z_1 - z_0|\|f\|_{L^\infty(z_{[z_0,z_1]}(B_\rho))} e^{-\omega\Im[g(\xi_*)]} e^C \rho^{-2N}. \tag{30}$$

Proof. Noting that

$$I = \mathrm{e}^{\mathrm{i}\omega g(\xi_*)} \int_\gamma f(z)\mathrm{e}^{\mathrm{i}\omega(g(z)-g(\xi_*))}\,\mathrm{d}z$$

and that

$$|f(z)\mathrm{e}^{\mathrm{i}\omega(g(z)-g(\xi_*))}| \le \|f\|_{L^\infty(z_{[z_0,z_1]}(B_\rho))}\mathrm{e}^C, \qquad z \in z_{[z_0,z_1]}(B_\rho),$$

the result follows from [17, Thm 19.3]. □

Theorem 3.4 motivates the definition of the non-oscillatory region in (8). Indeed, if the assumptions of Theorem 3.4 hold with ρ and C independent of ω then the bound (30) guarantees ω-independent exponential convergence for ω bounded away from zero. However, even when (8) is satisfied, the relationship between ξ_*, ρ, C and ω is beyond our control in general because the ellipse may extend beyond the non-oscillatory region, so that $C > C_{\text{ball}}$. Thus we cannot control the factor e^C entirely based on condition (8).

Still, the bound (30) shows that the quadrature error decreases with increasing N. The precise rate of decrease depends on a balance between the decay of ρ^{-2N} and the growth of e^C and $\|f\|_{L^\infty(z_{[z_0,z_1]}(B_\rho))}$ for increasing ρ. We quantify this in the special case of monomial phase in §3.3.3.

3.3.2. Quadrature for the SD contours

For Type 2 or Type 3 integrals along SD contours we use either Gauss-Laguerre or (possibly truncated) Gauss-Legendre quadrature, as detailed in §2.6.2 and §2.6.3. We expect these rules to converge rapidly to the true value of the integral as the number of quadrature points N tends to infinity, provided that the integrand is analytic and bounded in a suitable region of the complex $\tilde{p}$ plane.

For Gauss-Laguerre the following result appeared recently in [19, Thm 6.3].

Theorem 3.5. *Suppose that $\tilde{f}$ is analytic inside and on the parabola $P_\rho := \{z \in \mathbb{C} : \sqrt{-z} = \rho\}$ for some $\rho > 0$, where the branch cut is along the positive real axis and $\sqrt{-z}$ is real and positive on the negative real axis, that $\tilde{f}$ grows at most algebraically as $z \to \infty$ inside the parabola, and that the integral*

$$\mathcal{K}_\rho := \int_{P_\rho} |e^{-z}\sqrt{-z}\tilde{f}(z)|\,\mathrm{d}z$$

is finite. Let I and Q denote the left- and right-hand sides of (18), respectively. Then

$$|I - Q| \le \mathcal{K}_\rho \frac{e^{-\omega\Im[g(\eta)]}}{\omega} e^{-4\rho\sqrt{N}}. \tag{31}$$

This result implies that our Gauss-Laguerre quadrature approximation should converge root-exponentially as $N \to \infty$, provided that f is sufficiently well-behaved at infinity. The presence of singularities in the complex $\tilde{p}$-plane limits the size of ρ, and hence the convergence rate. We know from (17) that our integrand is singular at points $\tilde{p} \in \mathbb{C}$ where $g'(h_\eta(\tilde{p}/\omega)) = 0$, i.e. where $h_\eta(\tilde{p}/\omega) = \xi$ for some stationary point ξ. Since we only trace SD contours outside the non-oscillatory region (which contains the stationary points), we know that there cannot be singularities on the SD contour itself. If the start point η lies on an SD contour emanating from a stationary point ξ then we expect there to be a singularity in the $\tilde{p}$-plane at $\tilde{p} = \omega\Im[g(\xi) - g(\eta)] < 0$. We show in §3.3.3 that in the special case of monomial phase this singularity lies at $\tilde{p} = -C_{\text{ball}}$, which implies root-exponential convergence independent of ω for ω bounded away from zero. Determining the locations of the other possible singularities in the complex $\tilde{p}$-plane is more challenging, since it involves study of the (multivalued) inverse of g. We leave further theoretical investigation of this to future work.

3.3.3. Results for monomial phase

It is instructive to consider the special case of a monomial phase $g(z) = z^J$ for some $J \in \mathbb{N}$. In this case there is a single stationary point of order $J - 1$ at $\xi = 0$, and $g(0) = 0$. Following the prescription (8), we obtain a ball radius

$$r_0 = (C_{\text{ball}}/\omega)^{1/J}.$$

We first consider a Type 1 integral in the non-oscillatory region. For simplicity we choose $f(z) \equiv 1$. Specifically, we consider the evaluation of the integral

$$\int_0^{r_0\mathrm{e}^{\mathrm{i}\theta}} \mathrm{e}^{\mathrm{i}\omega g(z)}\,\mathrm{d}z,$$

for some $\theta \in [0, 2\pi]$. Taking $\xi_* = 0$, we can apply Theorem 3.4 with any $\rho > 1$, and the resulting scaled and translated Bernstein ellipse surrounding $[0, r_0\mathrm{e}^{\mathrm{i}\theta}]$ is contained in the disc $|z| \le sr_0$, where ρ and s are related by

$$\rho = 2s - 1 + \sqrt{(2s-1)^2 - 1} = 2s - 1 + 2\sqrt{s^2 - s}.$$

Hence condition (29) is satisfied, independently of θ, with

$$C = C_{\text{ball}} s^J,$$

which is independent of ω but dependent on J. When s is large, we have $\rho \approx 4s$, and in this regime the error bound provided by (30) for Gauss-Legendre quadrature is approximately proportional to

$$(C_{\text{ball}}/\omega)^{1/J} \mathrm{e}^{C_{\text{ball}} s^J} (4s)^{-2N}.$$

As a function of s, with J and N fixed, this quantity is minimised where its s-derivative vanishes, which occurs where

$$C_{\text{ball}} J s^J - 2N = 0,$$

i.e. where

$$s = \left(\frac{2N}{C_{\text{ball}} J} \right)^{1/J}.$$

Accordingly, the error bound is approximately proportional to

$$(C_{\text{ball}}/\omega)^{1/J} 16^J \left(\frac{8eN}{C_{\text{ball}} J} \right)^{-2N/J}.$$

Thus we expect super-exponential convergence as $N \to \infty$ for fixed J. However, we expect the convergence to be slower the larger J is.

Next we consider a Type 2 integral over an SD contour, again with $f(z) \equiv 1$. Specifically, we consider the evaluation of the integral

$$\int_{r_0 \mathrm{e}^{\mathrm{i}\upsilon}}^{\infty \mathrm{e}^{\mathrm{i}\upsilon}} \mathrm{e}^{\mathrm{i}\omega g(z)} \, \mathrm{d}z,$$

where $\upsilon = ((2j + 1/2)\pi)/J$ for some $j \in \{1, \dots, J\}$. Following our method, the contour is parametrized by

$$h_\eta(p) = (r_0^J + p)^{1/J} \mathrm{e}^{\mathrm{i}\upsilon}, \qquad p \in [0, \infty),$$

and, recalling (10) and (17), after rescaling $p = \tilde{p}/\omega$ the integral becomes

$$\frac{\mathrm{e}^{-C_{\text{ball}}} \mathrm{e}^{\mathrm{i}\upsilon}}{\omega^{1/J} J} \int_0^\infty (C_{\text{ball}} + \tilde{p})^{1/J - 1} \mathrm{e}^{-\tilde{p}} \, \mathrm{d}\tilde{p}.$$

The integrand has a branch point at

$$\tilde{p} = -C_{\text{ball}},$$

but we note that the distance between the branch point and the positive real $\tilde{p}$-axis equals C_{ball}, which is independent of both ω and J.

For truncated Gauss-Legendre the relevant theory can be found in [17, Chap. 19] (and see also [18]). Due to the branch point at $\tilde{p} = -C_{\text{ball}}$, as $N \to \infty$ we obtain exponential convergence to the integral over the interval $[0, P]$, where P is given by either (21) or (23). In the case where $P = L$, by the definition of L in (22), we expect the truncation error to have relative order δ_{quad}.

3.3.4. Number and distribution of quadrature points

PathFinder uses a fixed number N of quadrature points on each contributing contour, and that number is the same both for integrals within and outside the non-oscillatory region, i.e., for Gauss–Legendre and Gauss–Laguerre quadrature. Thus, increasing the single parameter N provides a way of uniformly improving accuracy.

The theoretical results in this section (specifically, Theorems 3.4 and 3.5) imply that the precise rate of improvement with respect to N depends on the type of integral being approximated. They suggest even that a different strategy for the distribution of quadrature points may be superior. Indeed, exponential convergence of Gauss–Legendre for Type 1 integrals in the non-oscillatory region is not balanced with root-exponential convergence of Gauss–Laguerre for Type 2 integrals outside. Similarly, convergence rates of Gauss-Laguerre and truncated Gauss-Legendre outside the non-oscillatory region are different. Our choice of a fixed parameter N is inspired on the one hand by simplicity, and on the other hand by the lack of robust methods to optimize parameters in alternative schemes. For example, we have shown in §3.3.3 that the convergence rate of Gauss-Legendre for Type 1 integrals may depend on the order of nearby stationary points. While this can be quantified precisely for the case of monomial phase, it is not at all clear how to generalise this analysis when a cluster of multiple stationary points is present. Hence, stationary point order is a quantity that we

deliberately do not explicitly compute, estimate or rely on in any way. Implicitly, of course, it plays a big role, and it does so mainly via the definition of the ball of the radius in (8).

The main practical benefit of the theoretical analysis of quadrature error in this section is the guarantee that N is a robust parameter for improving accuracy. Concerning possible future improvements, rather than attempting to optimize the quadrature point distribution a priori, we believe a more promising development would be the ability to invoke standard adaptive quadrature schemes along the contours for a given function f. However, it should be borne in mind that quadrature forms just one step in our algorithm, and that the other steps (particularly the SD path tracing) incur a non-negligible cost overhead, that should also be considered when trying to further optimize performance.

4. Further implementation aspects

In this section we discuss some additional aspects of the implementation of our algorithm in PathFinder.

4.1. Default parameter values

In Table 4.1 we list the user-specified parameters in our algorithm, along with the default values used in all our numerical results in §5. These were determined as the result of extensive numerical experiments on a range of examples, not detailed here. Instructions on how to adjust these parameters away from their default values can be found at github.com/AndrewGibbs/PathFinder.

Table 4.1
User-specified parameters and their default values in PathFinder.

Parameter	Domain	Meaning	Default
C_{ball}	$(0,\infty)$	Governs maximum number of oscillations across each non-oscillatory ball (and hence the ball radius)	2π
N_{ball}	$\mathbb{N}$	Number of rays used when determining the ball radius	16
δ_{ball}	$(0,1)$	Governs when overlapping balls should be amalgamated	$10^{-3}/(2\max(J-2,1))$
δ_{ODE}	$(0,1)$	Governs the local step size in the ODE solver for SD path tracing	0.1
δ_{coarse}	$(0,1)$	Tolerance for the increment in the Newton iteration in the SD path tracing	10^{-2}
δ_{fine} $(<\delta_{\text{coarse}})$	$(0,1)$	Tolerance for the increment in the Newton iteration in the quadrature	10^{-13}
δ_{quad}	$(0,1)$	Governs when the contribution from an integral on the quasi-SD deformation is computed	10^{-16}
N	$\mathbb{N}$	Number of quadrature points to use in each integral evaluated in step 6	no default

4.2. Small ω

While our algorithm is geared towards the case where ω is moderate or large, we make a brief comment on the case where ω is small. If Γ is infinite then the integral (1) typically diverges for $\omega = 0$. However, if Γ is finite then the integral converges for $\omega = 0$ and for small enough ω it is non-oscillatory. In PathFinder we detect and deal with this case in the following way. If both endpoints are finite, then before starting step 1 of the algorithm we construct non-oscillatory balls around the endpoints (using the process in §2.2) and check whether the balls intersect non-trivially. If so, we apply standard Gauss-Legendre quadrature to evaluate (1); if not, the balls are discarded and we proceed with the rest of the algorithm.

4.3. The case $J = 1$

In the case $J = 1$ (linear phase) there are no stationary points, and our algorithm simplifies dramatically. Furthermore, the SD contours are simply parallel straight lines in the direction of the single valley at angle $\pi/2 - \arg(\alpha_1)$, and there is no need to trace them numerically. Hence when $J = 1$ PathFinder skips the ODE contour tracing step and exploits the exact characterization of the SD contours mentioned above.

4.4. Specifying infinite endpoints

In the description of our algorithm in §2 we made the assumption that any infinite endpoint of the contour Γ should be at a valley $v \in \mathcal{V}$. PathFinder is actually more flexible than this. The user is permitted to specify an infinite endpoint at any $\theta \in [v - \pi/(2J), v + \pi/(2J)]$ and the code will automatically adjust this to equal v. The case $\theta = v \pm \pi/(2J)$ is delicate because the highest order term in the phase does not provide exponential decay along the contour. Nonetheless, we include it, because in applications one often encounters this case, with the integral converging conditionally (under appropriate assumptions on f) and the contour deformation to v being justified by Jordan's Lemma.

5. Numerical results

In this section we present numerical results illustrating the performance of our algorithm and its implementation in PathFinder. All results in this section were produced using PathFinder Version 1.0 [8].

5.1. A "generic" example

We begin by illustrating the performance of PathFinder on the integral

$$I = \int_{-1}^{1} (2z^4 + 7z^3 + z^2 + 8z + 2)\mathrm{e}^{\mathrm{i}\omega(3z^9+z^8+4z^7+z^6+5z^5+9z^4+2z^3+6z^2+5z+3)}\,\mathrm{d}z, \tag{32}$$

where, to convey the message that our approach is applicable to truly "generic" amplitudes and polynomial phase functions, the coefficients of f and g are chosen to be the first 5 digits of e and the first 10 digits of π, respectively. This can be approximated by PathFinder via the Matlab code (cf. (2))

```
PathFinder(-1,1,@(z) 2*z.^4+7*z.^3+z.^2+8*z+2,[3 1 4 1 5 9 2 6 5 3],omega,N)
```

In Fig. 5.1 we plot the quasi-SD deformations and quadrature point distributions (using the PathFinder `'plot'` option) for (32) for $\omega \in \{0.01, 1, 5, 50\}$ and $N = 10$. As explained in §2.2, for smaller ω the non-oscillatory balls are larger, and can overlap, while for larger ω they shrink around the stationary points. In more detail, in Fig. 5.1(a) ($\omega = 0.01$), ω is small enough that both endpoints are inside the same non-oscillatory ball. Hence the integral is treated as non-oscillatory and is approximated by Gauss-Legendre quadrature along a single straight-line contour. In Fig. 5.1(b) ($\omega = 1$), ω is still small enough that many of the balls overlap, and the quasi-SD deformation comprises two SD contours (one from an exit and one from an endpoint) plus four straight-line contours in the non-oscillatory region. In Fig. 5.1(c) ($\omega = 5$), ω is large enough that only two balls overlap, and the quasi-SD deformation comprises five SD contours (two from endpoints, two from exits to valleys, and one from an exit to an entrance), plus four straight-line contours in the non-oscillatory region. Finally, in Fig. 5.1(d) ($\omega = 50$), ω is so large that none of the balls overlap, and the quasi-SD deformation comprises eight contributing SD contours (two from endpoints and six from exits to valleys), plus three straight-line contours in the non-oscillatory region. However, in this case the two SD contours and one straight-line contour associated with the stationary point near $0.2 + 0.5\mathrm{i}$ are judged to make a negligible contribution to the integral, so are not assigned any quadrature points. We emphasize that this intricate behaviour is fully automated, with no expert input required from the user.

In Fig. 5.2(a) we plot the error in the PathFinder approximation of (32), compared to reference values computed using the Julia `QuadGK` package when $\omega < 500$, and using PathFinder with $N = 500$ when $\omega \geq 500$. For fixed ω we observe rapid convergence as $N \to \infty$, at a rate that appears independent of ω. In Fig. 5.2(b) we show the associated computation times, which remain bounded as ω increases.

5.2. Coalescence and the Airy function

The canonical example of an integral with two coalescing stationary points is provided by the integral representation for the Airy function, viz. (see [1, 9.5.4])

$$\mathrm{Ai}(x) = \frac{1}{2\pi\mathrm{i}} \int_{\infty\mathrm{e}^{-\mathrm{i}\pi/3}}^{\infty\mathrm{e}^{\mathrm{i}\pi/3}} \mathrm{e}^{z^3/3 - xz}\,\mathrm{d}z = \frac{1}{2\pi\mathrm{i}} \int_{\infty\mathrm{e}^{-\mathrm{i}\pi/3}}^{\infty\mathrm{e}^{\mathrm{i}\pi/3}} \mathrm{e}^{\mathrm{i}(-\mathrm{i}(z^3/3 - xz))}\,\mathrm{d}z, \qquad x \in \mathbb{C}, \tag{33}$$

which is of the form (1) with $f \equiv 1$, $\omega = 1$ and $g(z; x) = -\mathrm{i}(z^3/3 - xz)$. Up to a change of variable this is the same example for which, as mentioned in §1, a bespoke, complex Gaussian quadrature rule was developed in [11]. Ai can be approximated by PathFinder via the Matlab code

```
Ai = @(x) 1/(2i*pi) * PathFinder(-pi/3,pi/3,[],...
    -1i*[1/3 0 -x 0],1,N,'infcontour',[true true])
```

where the input `[]` for `f` indicates that $f \equiv 1$.

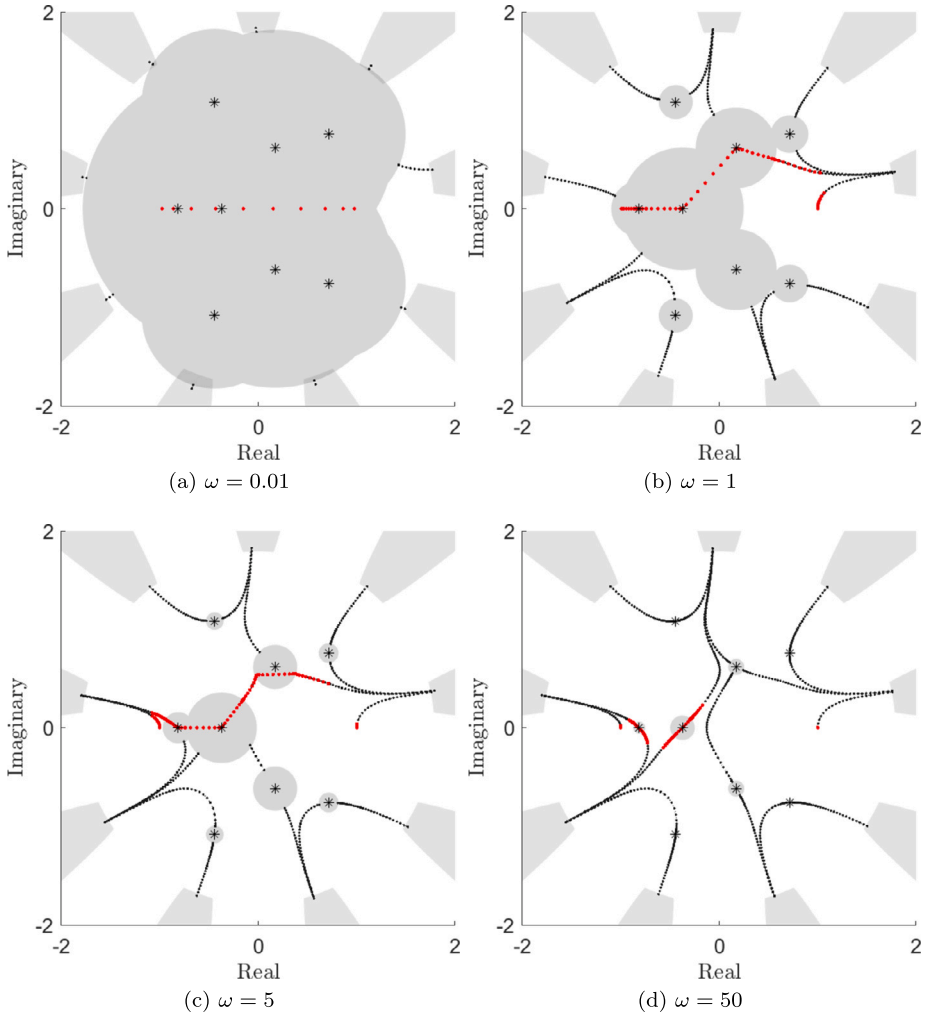


Fig. 5.1. PathFinder output (cf. Fig. 2.1) with $N = 10$ for the approximation of (32).

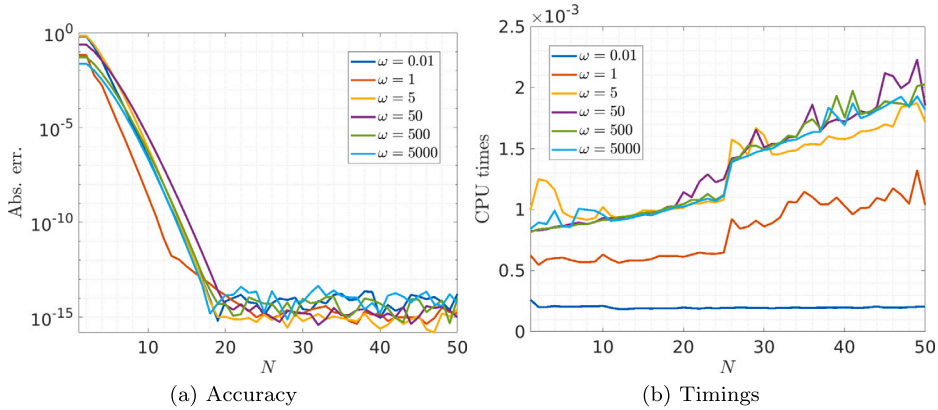
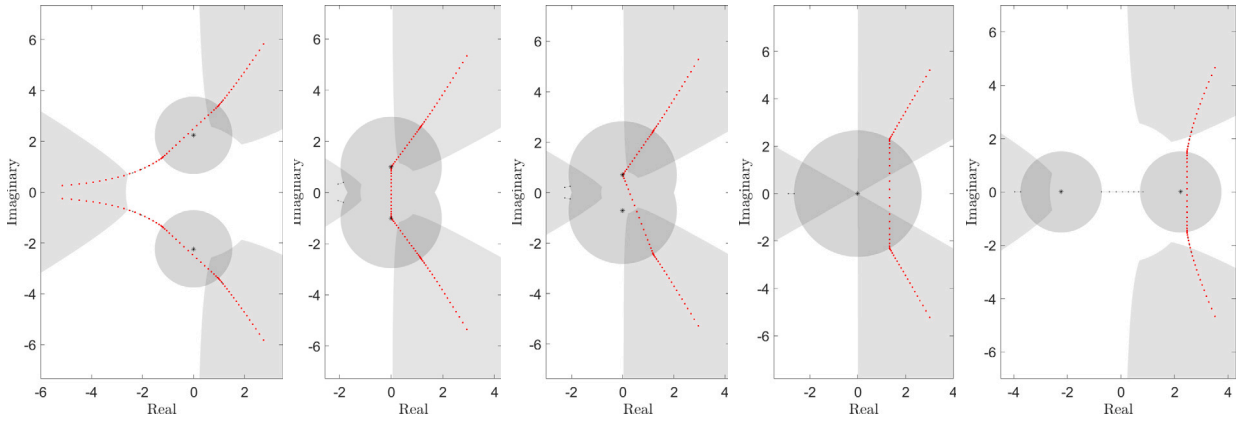


Fig. 5.2. Accuracy (a) and timings (b) of the PathFinder approximation of (32).



(a) $x = -5$ (b) $x = -1$ (c) $x = -0.5$ (d) $x = 0$ (e) $x = 5$

Fig. 5.3. PathFinder output with $N = 20$ for the approximation of $\mathrm{Ai}(x)$ via (33) at various x, showing the stationary point coalescence at $x = 0$.

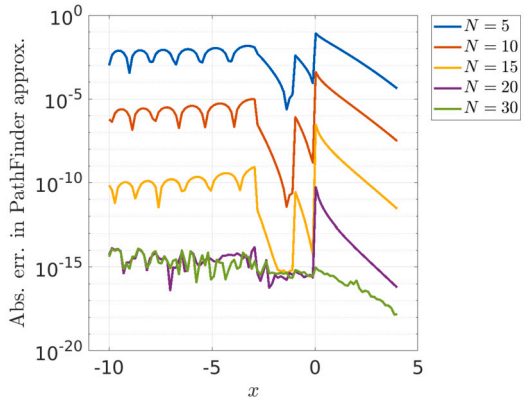


Fig. 5.4. Accuracy of PathFinder approximation of $\mathrm{Ai}(x)$ for different N.

In Fig. 5.3 we plot the quasi-SD deformations, along with the distribution of quadrature points for $N = 20$, for the evaluation of $\mathrm{Ai}(x)$ at $x \in \{-5, -1, -0.5, 0, 5\}$. Here one observes in detail how our algorithm deals with stationary point coalescence, as the non-oscillatory balls overlap, merge, then split. In Fig. 5.3(a) the quasi-SD deformation comprises four SD contours from exits, plus two straight-line contours inside balls (which do not go via stationary points). In Fig. 5.3(b) the balls overlap and this changes to two SD contours from exits plus three straight-line contours inside balls (which go via both stationary points). In Fig. 5.3(c) the balls overlap enough that both stationary points are contained in both balls, so we get two SD contours from exits plus just two straight-line contours inside balls (which go via only one of the stationary points). In Fig. 5.3(d) the balls have merged completely and in addition to the two SD contours from exits there is just one straight-line contour inside a ball (which does not go via the stationary point). In Fig. 5.3(e) the balls have split again, but we see the same deformation structure as in Fig. 5.3(d). Again, we emphasize that these calculations are fully automated.

In Fig. 5.4 we show the accuracy of the PathFinder approximation for this example as a function of $x \in [-10, 4]$, for different N. Our reference is the built-in Matlab command `airy`. We note that between $x = -3$ and $x = 0$ the error for the smaller values of N undergoes some jumps. These are due to the fact that near stationary point coalescence the topology of the quasi-SD deformation, the number of contours constituting it, and hence the total number of quadrature points along it (recall (14)), all change discontinuously as a function of x (as illustrated in Fig. 5.3). However, as N increases we see a clear, approximately exponential decrease in the error, and, although the rate of decrease depends slightly on x (because of the factors mentioned above), for $N = 30$ we achieve approximately 10^{-13} error uniformly across the interval.

5.3. A high order stationary point - comparison with Mathematica's implementation of Levin quadrature

We now consider the integral

$$I = \int_{-1}^{1} \sin(z) \mathrm{e}^{\mathrm{i}\omega z^9} \, \mathrm{d}z, \tag{34}$$

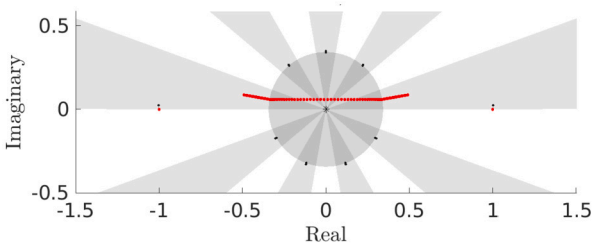
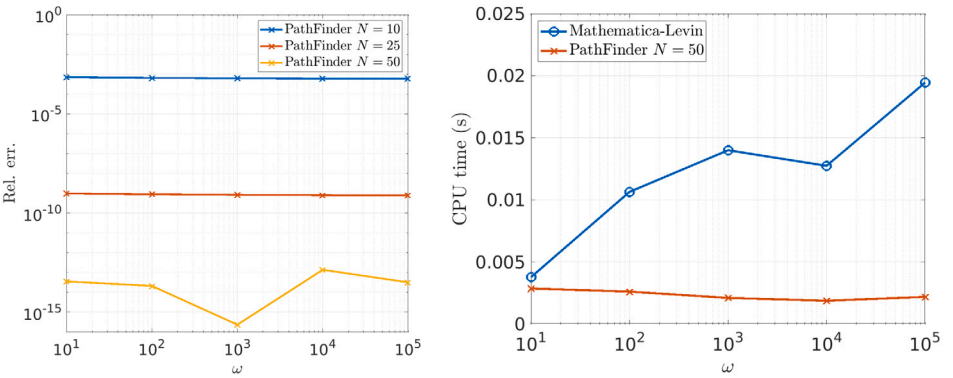


Fig. 5.5. PathFinder output for (34) with $\omega = 100,000$ and $N = 50$.



(a) Accuracy (compared to Mathematica) (b) Timings

Fig. 5.6. Accuracy (a) and timings (b) of the PathFinder approximation of (34), compared to Mathematica's `NIntegrate` command.

which has a stationary point of order 8 at the origin. The integral (34) can be approximated by PathFinder via the command

```
PathFinder(-1,1,@(z) sin(z),[1 0 0 0 0 0 0 0 0 0],omega,N)
```

Fig. 5.5 shows the quasi-SD deformation and quadrature point distribution obtained by PathFinder for $\omega = 100,000$ and $N = 50$. There are small contributions from the endpoints, but the main contribution comes from the ball containing the stationary point.

In the Mathematica documentation [20, pp. 75-86], it is stated that oscillatory integrals with monomial phase functions such as (34) can be evaluated efficiently using the built-in Mathematica function `NIntegrate`, via its implementation of Levin quadrature (which is described, e.g., in [7, §3.3]). To do this one can use the Mathematica command:

```
NIntegrate[Sin[x]Exp[omega*I*x^9],{x,-1,1},Method->{``LevinRule",”Kernel”->Exp[omega*I*x^9]}]
```

In Fig. 5.6(a) we show a plot of the relative accuracy of our PathFinder approximation, compared to the Mathematica approximation (using the default settings), as a function of ω, for different N values. For all three N values the accuracy of our approximation is approximately uniform in ω, and for $N = 50$ our approximation agrees with Mathematica's to approx 13 digits. In Fig. 5.6(b) we report the corresponding computation times (averaged over 100 identical runs) for the Mathematica routine and for the PathFinder approximation with $N = 50$. These results were obtained on a laptop (i7-1185G7, 32 GB RAM) running Mathematica v13.0 and Matlab v2021b. The results suggest that PathFinder is highly competitive with Mathematica for this problem, especially for large ω.

5.4. Coalescence to a high order stationary point

We now investigate the robustness of our algorithm in the presence of a large number of coalescing stationary points. Specifically, we consider the integral

$$\int_{-1}^{1} \mathrm{e}^{\mathrm{i}\omega(z^7/7 - r^6 z)}\,\mathrm{d}z, \tag{35}$$

where $r \geq 0$ is a parameter controlling the coalescence. For $r > 0$ there are 6 stationary points with $|\xi| = r$, namely the solutions of $\xi^6 = r^6$, and for $r = 0$ there is a single stationary point of order 6. To evaluate this integral in PathFinder for a given r, one can use the command

```
PathFinder(-1,1,[],[1/7 0 0 0 0 0 -r^6 0],omega,N)
```

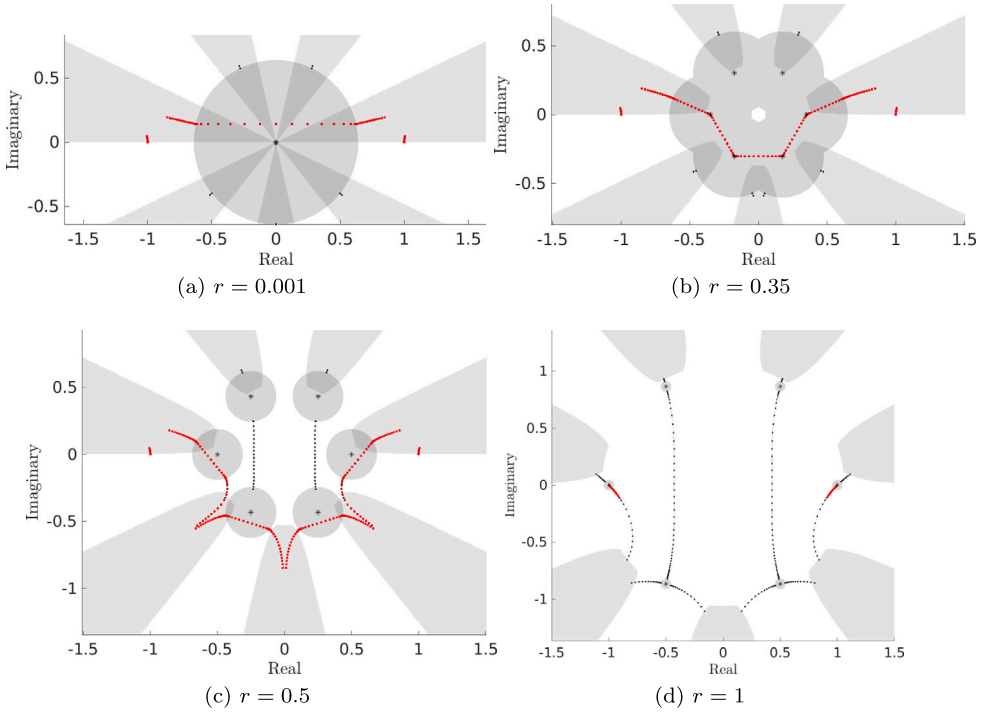


Fig. 5.7. PathFinder output for the approximation of (35) with $\omega = 1000$ and $N = 15$.

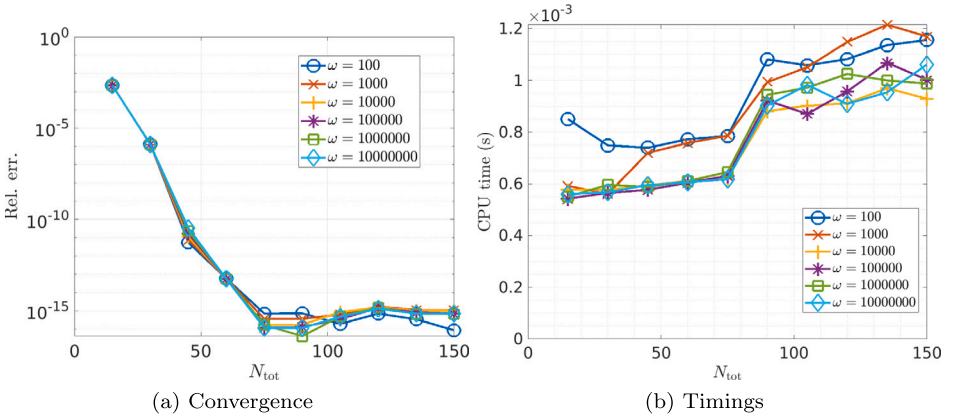
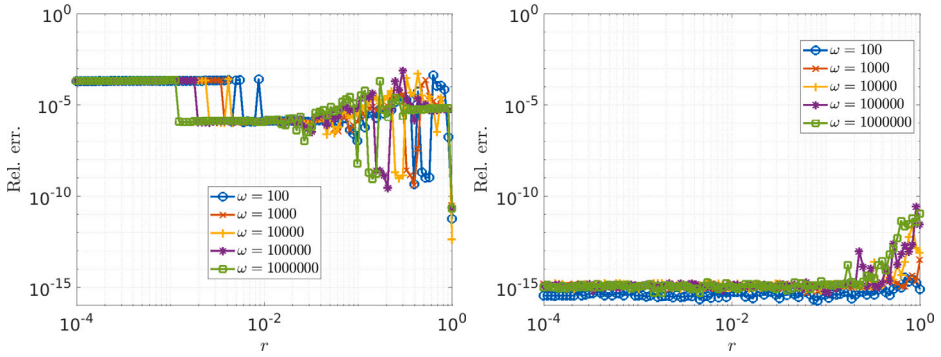


Fig. 5.8. Accuracy (a) and timings (b) for (35) for $p = 6$ and $r = 0.01$.

In Fig. 5.7 we plot the quasi-SD deformations and quadrature point distributions for a some different r values, showing how the balls first intersect and then merge as $r \to 0$.

In Fig. 5.8 we show convergence (with respect to a PathFinder reference with $N = 500$) and CPU times (averaged over 100 runs) for fixed $r = 0.01$. We see that both the error and the CPU time are essentially independent of ω in this case. In Fig. 5.9 we plot errors for two fixed N values $N = 10, 50$, as a function of r. We observe that as $r \to 0$, the error stays bounded. For $N = 10$ the error jumps up between $r = 10^{-3}$ and $r = 10^{-2}$, at a point depending on ω. This represents the point at which the balls around the stationary points merge, resulting in a reduction of N_{tot}, and hence a reduction in accuracy. But after this point we observe no further reduction in accuracy as $r \to 0$. We remark that for sufficiently small $r > 0$ the six stationary points are numerically indistinguishable, but this isn't a problem for our algorithm because in that case the problem will be treated identically to that of a monomial phase.



(a) $N = 10$ (b) $N = 50$

Fig. 5.9. Accuracy for (35) as $r \to 0$ for $p = 6$, for $N = 10$ (a) and $N = 50$ (b).

5.5. Canonical cuspoid integrals and their generalisations

In this section we show how our algorithm can be applied to the computation of some of the canonical integrals catalogued by Berry and Howls in [1, §36], which, as mentioned already in §1, are of fundamental importance in numerous application areas including optics, acoustics and quantum mechanics.

In this context, our algorithm is related to that of [13], where an adaptive contour deformation approach was applied to evaluate the cuspoid integrals considered in §5.5.1. The algorithm in [13] is similar in spirit to our approach, in that it deforms the integration contour so that it terminates in valleys at infinity, and splits the contour into portions close to stationary points plus portions away from stationary points. However, in contrast to our approach, the algorithm in [13] does not attempt to trace SD contours, and hence is susceptible to rounding errors associated with the "violent" behaviour of the exponential factor $\mathrm{e}^{\mathrm{i}\omega g(z)}$ when one is not on a true SD contour - see [13, §2]. Furthermore, while the algorithm in [13] was specialised to the case of integration over the real line, our algorithm can handle much more general contours, as we illustrate in §5.5.2.

5.5.1. Cuspoid integrals

The so-called "cuspoid integrals" listed in [1, §36.2.4] are all of the form (1) with polynomial phase g and unit amplitude $f \equiv 1$, unit frequency $\omega = 1$, and integration along the real line. Our algorithm is ideally suited to the evaluation of these integrals, and to demonstrate this we compute two of them. In the notation of [1, §36], we consider the cusp catastrophe integral

$$\Psi_2(x,y) = P(y,x) = \int_{-\infty}^{\infty} \mathrm{e}^{\mathrm{i}(t^4+yt^2+xt)}\,\mathrm{d}t, \tag{36}$$

where P is the Pearcey function, and the swallowtail catastrophe integral

$$\Psi_3(x,y,z) = \int_{-\infty}^{\infty} \mathrm{e}^{\mathrm{i}(t^5+zt^3+yt^2+xt)}\,\mathrm{d}t. \tag{37}$$

Both exhibit coalescence of stationary points on certain algebraic varieties (see [1, §36.5(ii)]) on which both the first and second derivatives of the phase function vanish. In the case of (36) this occurs when

$$y = -\frac{3}{2}|x|^{2/3}, \tag{38}$$

and for (37) this occurs when

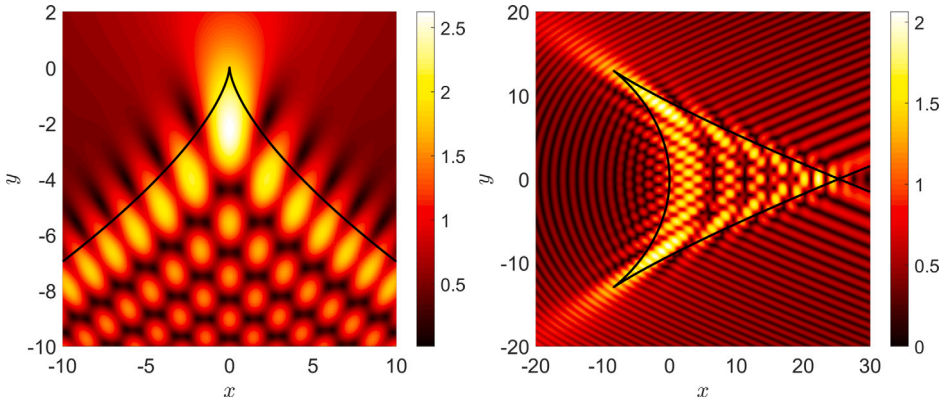
$$400x^3 - 360x^2z^2 - 135y^4 - 27y^2z^3 + 540xy^2z + 81xz^4 = 0. \tag{39}$$

The integrals (36) and (37) can be computed in Pathfinder via the commands

```
Psi2 = @(x,y) PathFinder(pi,0,[],[1 0 y x 0],1,N,'infcontour',[true true])

Psi3 = @(x,y,z) PathFinder(pi,0,[],[1 0 z y x 0],1,N,'infcontour',[true true])
```

Fig. 5.10 shows plots of the magnitude of (36) and (37) (the latter over the plane $z = -7.5$), computed using PathFinder with the default settings and $N = 50$. The plots agree qualitatively with those presented in [1, Figs 36.3.1 & 36.6.5], and, for (36), agree quantitatively (to all five decimal places presented) with the values presented in [13, Table 1]. Computation times on a small desktop computer (Intel i7-4790, 32 GB RAM) were less than a minute for the cusp (which required the computation of 10000 instances of



(a) $|\Psi_2(x,y)|$ (b) $|\Psi_3(x,y,-7.5)|$

Fig. 5.10. Magnitude plots of (36) and (37), with coalescence curves (38) and (39) (the latter with $z=-7.5$) superimposed in black. The computational grid was of size 100×100 for (a) and 500×500 for (b).

(36), averaging 0.005 s per instance) and less than an hour for the swallowtail (which required 250000 instances of (37), averaging 0.01 s per instance).

5.5.2. *Generalisations*

In [10] the authors considered a family of generalisations of certain canonical cuspoid integrals, with integration no longer over the real line, but rather over a complex contour starting and ending at valleys at infinity, and possibly with a non-unit amplitude function.

A specific aim of [10] was to investigate the relevance of such integrals to the study of the so-called "inflection point problem", a canonical problem in wave scattering originally introduced over 50 years ago by Popov in [14]. This problem, which remains unsolved in closed form, concerns two-dimensional time-harmonic wave propagation near a boundary with an inflection point, and seeks a solution for the wave field near the inflection point that describes the transition from an incoming "whispering gallery wave" supported on the concave portion of the boundary, to outgoing "creeping waves" along the convex portion of the boundary, along with a scattered "searchlight" beam (for details and further references see [16]).

In this context, in [10, §3.3] the authors studied the family of integrals

$$A_{ij}(x,y)=\int_{\Gamma_{ij}} f(t)\mathrm{e}^{\mathrm{i}(2t^5/5-xt^4/2-yt^2)}\,\mathrm{d}t, \tag{40}$$

where $f(t)$ is some amplitude to be specified, and Γ_{ij} denotes any contour from valley v_i to valley v_j, where $v_j:=(2(j-1)+1/2)\pi/5$, $j=1,\dots,5$. These integrals have stationary point coalescence on the cubic curve $y+4x^3/27=0$, which suggests that, by appropriately choosing f and Γ_{ij}, they might exhibit certain features of the solution of the inflection point problem. Indeed, in [10, §4] it was shown that as $x\to-\infty$ near the cubic curve, the integral A_{32} has the character of an incoming whispering gallery type wave, and that, as $x\to+\infty$ near the cubic curve, the integral A_{52} has the character of an outgoing creeping wave. However, plots of the resulting fields could not be presented in [10] due to the lack of a suitable numerical evaluation method and implementation.

Using PathFinder we are able to remedy this. In Figs. 5.11(a) and 5.11(b) we provide plots of the magnitude of A_{32} and A_{52} with $f\equiv1$. To evaluate the integrals we used the PathFinder code

```
A32 = @(x,y) PathFinder(9*pi/10,pi/2,[],[2/5 -x/2 0 -y 0 0],1,N,'infcontour',[true true])
```

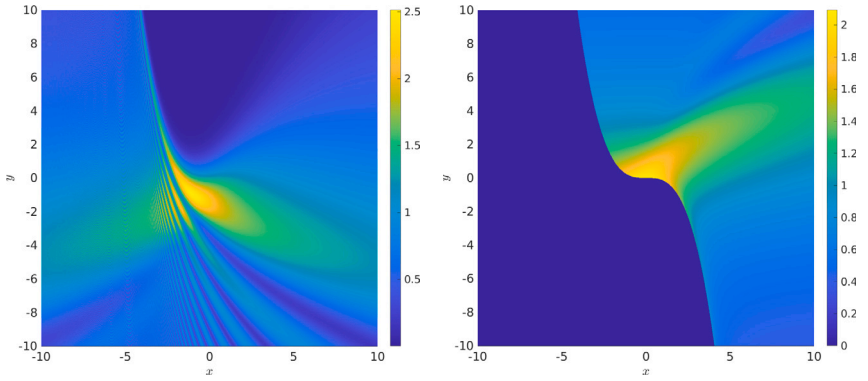
```
A52 = @(x,y) PathFinder(17*pi/10,pi/2,[],[2/5 -x/2 0 -y 0 0],1,N,'infcontour',[true true])
```

We only plot A_{52} above the cubic curve $y+4x^3/27=0$, because below this curve A_{52} becomes exponentially large (cf. [10, Fig. 12(i)]). In Figs. 5.11(c) and 5.11(d) we present corresponding plots of the modulated plane wave

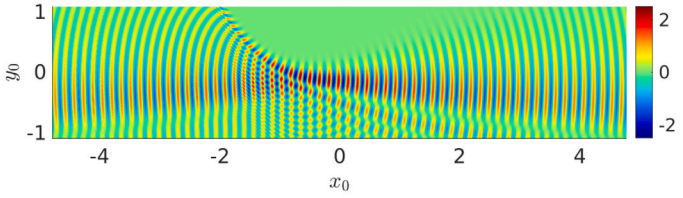
$$u(x_0,y_0)=A_{ij}(x,y)\mathrm{e}^{\mathrm{i}kx_0},$$

where (x_0,y_0) are outer variables, related to the inner variables (x,y) by $x=k^{1/5}x_0$, $y=k^{3/5}y_0$, which is an asymptotic solution of the Helmholtz equation $\Delta u+k^2u=0$ as $k\to\infty$ in the region $x_0=O(k^{-1/5})$, $y_0=O(k^{-3/5})$ [10, §1]. Here one observes the predicted incoming whispering gallery type behaviour of A_{32} near the top of Fig. 5.11(c) between $x_0=-2$ and $x_0=-1$, with oscillations giving way to an exponentially small field in the caustic shadow, and the predicted creeping wave type behaviour of A_{52} near the bottom of Fig. 5.11(d) between $x_0=1$ and $x_0=2$, with waves propagating along the cubic curve, shedding rays tangentially.

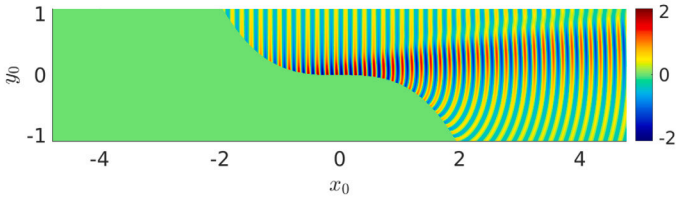
In ongoing and future studies we plan to use PathFinder to further investigate the properties of integrals of the form (40), and generalisations involving different choices of f and higher degree phase functions (see [10]), which we hope may shed new light on the inflection point problem and related problems in high frequency wave propagation.



(a) $|A_{32}(x,y)|$ (b) $|A_{52}(x,y)|$



(c) $\Re[A_{32}(k^{1/5}x_0, k^{3/5}y_0)\mathrm{e}^{\mathrm{i}kx_0}]$



(d) $\Re[A_{52}(k^{1/5}x_0, k^{3/5}y_0)\mathrm{e}^{\mathrm{i}kx_0}]$

Fig. 5.11. Plots of (40) with $f \equiv 1$, along with the associated approximate Helmholtz equation solutions for $k = 40$.

CRediT authorship contribution statement

A. Gibbs: Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **D.P. Hewett:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **D. Huybrechs:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Software, Supervision, Visualization, Writing – original draft, Writing – review & editing.

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.

Data availability

A link to the software is provided in the paper.

Acknowledgements

The authors acknowledge support from KU Leuven project C14/15/055 (DH and AG) and EPSRC projects EP/S01375X/1 and EP/V053868/1 (DPH and AG). We are grateful to Alex Townsend, Nick Trefethen, Marcus Webb, Sheehan Olver and Samuel Groth for helpful discussions in relation to this project.

References

[1] NIST Digital Library of Mathematical Functions, http://dlmf.nist.gov/, release 1.1.3 of 2021-09-15.
[2] A. Asheim, A. Deaño, D. Huybrechs, H. Wang, A Gaussian quadrature rule for oscillatory integrals on a bounded interval, Discrete Contin. Dyn. Syst. 34 (2013) 883–901.
[3] N. Bleistein, R.A. Handelsman, Asymptotic Expansions of Integrals, 2nd edition, Dover, 1986.
[4] W.M. Boothby, The topology of the level curves of harmonic functions with critical points, Am. J. Math. 73 (1951) 512–538.
[5] J.P. Boyd, Computing the zeros, maxima and inflection points of Chebyshev, Legendre and Fourier series: solving transcendental equations by spectral interpolation and polynomial rootfinding, J. Eng. Math. 56 (2006) 203–219.
[6] A.F. Celsus, A. Deaño, D. Huybrechs, A. Iserles, The kissing polynomials and their Hankel determinants, Trans. Math. Appl. 6 (2021) tnab005.
[7] A. Deaño, D. Huybrechs, A. Iserles, Computing Highly Oscillatory Integrals, SIAM, 2018.
[8] A. Gibbs, PathFinder v1.0, https://github.com/AndrewGibbs/PathFinder/releases/tag/v1.0, 2023.
[9] A. Gibbs, D.P. Hewett, D. Huybrechs, E. Parolin, Fast hybrid numerical-asymptotic boundary element methods for high frequency screen and aperture problems based on least-squares collocation, SN Partial Differ. Equ. Appl. 1 (2020) 1–26.
[10] D.P. Hewett, J.R. Ockendon, V.P. Smyshlyaev, Contour integral solutions of the parabolic wave equation, Wave Motion 84 (2019) 90–109.
[11] D. Huybrechs, A.B.J. Kuijlaars, N. Lejon, A numerical method for oscillatory integrals with coalescing saddle points, SIAM J. Numer. Anal. 57 (2019) 2707–2729.
[12] D. Huybrechs, S. Vandewalle, A sparse discretization for integral equation formulations of high frequency scattering problems, SIAM J. Sci. Comput. 29 (2007) 2305–2328.
[13] N. Kirk, J. Connor, C. Hobbs, An adaptive contour code for the numerical evaluation of the oscillatory cuspoid canonical integrals and their derivatives, Comput. Phys. Commun. 132 (2000) 142–165.
[14] M.M. Popov, The problem of whispering gallery waves in a neighbourhood of a simple zero of the effective curvature of the boundary, J. Sov. Math. (now J. Math. Sci.) 11 (1979) 791–797.
[15] K.H. Rosen, Discrete Mathematics and Its Applications, McGraw-Hill, 2019.
[16] V.P. Smyshlyaev, I.V. Kamotski, Searchlight asymptotics for high-frequency scattering by boundary inflection, St. Petersburg Math. J. 33 (2022) 387–403.
[17] L.N. Trefethen, Approximation Theory and Approximation Practice, SIAM, 2013.
[18] L.N. Trefethen, Exactness of quadrature formulas, SIAM Rev. 64 (2022) 132–150.
[19] H. Wang, Optimal convergence analysis of Laguerre spectral approximations for analytic functions, arXiv preprint arXiv:2304.05744, 2023.
[20] Wolfram Research, Inc., Advanced Numerical Integration in Mathematica, 2008, Downloaded 6 April 2023 from library.wolfram.com.
[21] R. Wong, Asymptotic Approximations of Integrals, SIAM, 2001.