COMPUTATION OF 2D STOKES FLOWS VIA LIGHTNING AND AAA RATIONAL APPROXIMATION*

YIDAN XUE†, SARAH L. WATERS†, AND LLOYD N. TREFETHEN†

Abstract. Low Reynolds number fluid flows are governed by the Stokes equations. In two dimensions, Stokes flows can be described by two analytic functions, known as Goursat functions. Brubeck and Trefethen [9] recently introduced a lightning Stokes solver that uses rational functions to approximate the Goursat functions in polygonal domains. In this paper, we present the "LARS" algorithm (Lightning AAA Rational Stokes) for computing 2D Stokes flows in domains with smooth boundaries and multiply-connected domains using lightning and AAA rational approximation [36]. After validating our solver against known analytical solutions, we solve a variety of 2D Stokes flow problems with physical and engineering applications. Using these examples, we show rational approximation can now be used to compute 2D Stokes flows in general domains. The computations take less than a second and give solutions with at least 6-digit accuracy.

Key words. Stokes flow, biharmonic equation, lightning solver, AAA algorithm, rational approximation

MSC codes. 41A20, 65N35, 76D07

1. Introduction.

At small Reynolds numbers, where viscous forces dominate inertial forces, fluid flows are governed by the Stokes equations. This is typically true when the fluid is highly viscous or the length scale is small. Stokes flows have numerous physical and biological applications, including microcirculation [42], microfluidic devices [50], swimming microorganisms [28], fluid mixing [17] and lubrication [37]. Many of the flow characteristics in these problems can be analysed in two dimensions.

For two-dimensional Stokes flows, analytical solutions have been derived for certain problems using the Wiener-Hopf method [25,47], the method of images [17,20,24], and expansion of the stream function [35]. Analytical solutions usually only exist for problems with very simple geometries and boundary conditions. For more complex cases, one can approximate the solutions semi-analytically using the unified transform method [19,31,32] and extended lubrication theory [43], or solve the Stokes equations using numerical methods including finite element methods [29], boundary integral methods [38], and the lattice Boltzmann method [27]. It should be noted that there is no clear boundary between semi-analytical and numerical methods. For example, it is sometimes necessary to approximate the boundary conditions using orthogonal polynomials when using the unified transform method [32].

Brubeck and Trefethen [9] recently introduced a "lightning" Stokes solver, which uses rational functions to approximate the Goursat functions [23], which are two analytic functions that represent Stokes flow in 2D (see section 2 for details). The lightning solver differs from other numerical methods, since it treats corner singularities of the Stokes equations by clustering the poles of rational functions exponentially nearby. This enables its root-exponential convergence and thus "lightning" computation [22]. In [9], most Stokes flows were computed to at least 8-digit accuracy in less than a second. For the classic lid-driven cavity problem, the lightning solver captured several self-similar Moffatt eddies [35] near the bottom corners, which can be

*Submitted to the editors DATE.

Funding: YX would like to thank financial support from the UK EPSRC (EP/W522582/1). SLW and YX are grateful to funding from the UK MRC (MR/T015489/1).

†Mathematical Institute, University of Oxford, Oxford OX2 6GG, UK (xue@maths.ox.ac.uk, waters@maths.ox.ac.uk, trefethen@maths.ox.ac.uk).

challenging to resolve using more standard discretisation methods. The Goursat representation for 2D Stokes flows and the lightning solver will be reviewed in sections 2 and 3 of this paper respectively.

However, Brubeck and Trefethen [9] primarily focus on 2D Stokes flow problems in polygons. In this paper, we introduce LARS (Lightning-AAA Rational Stokes), a solver that can be implemented easily to compute 2D Stokes flows in general domains with custom boundary conditions in less than a second. LARS uses several rational approximation algorithms: the lightning solver for sharp corners [9, 22], the AAA rational approximation for smooth boundaries [10,36] and the series method for multiply connected domains [3, 18, 44].

It is well known that even for regions with analytic boundaries, the Goursat functions may only be analytically continuable a very short distance across the boundary, an effect known as the "crowding phenomenon" [11, 21]. This means that poles may need to be placed close to curved boundaries to achieve good rational approximations. Recently Costa and Trefethen [10] showed that using AAA rational approximation [36] to place poles outside the curved boundary enables fast and effective solution of Laplace problems. The AAA algorithm, derived from "Adaptive Antoulas-Anderson", automatically searches for a rational approximation in barycentric form for a vector of boundary values on a given boundary [36], and has been implemented in the Chebfun toolbox in MATLAB [14]. In this work, we apply AAA rational approximation to compute Stokes flows in domains with curved boundaries. Using an example case of Stokes flows in a channel with a smooth constriction, we compare our solution against a solution approximated using extended lubrication theory [43], and present these results in section 4.

We then introduce an algorithm for computing Stokes flows in multiply connected domains. For Laplace problems in multiply connected domains, the solution can be approximated using a Laurent series with a logarithmic term [3]. The series method has been applied to compute numerical solution to Laplace problems [44]. It has also been applied to 2D Stokes flows in domains bounded by cylinders [18, 39]. In this paper, we present an algorithm using the series method to compute Stokes flows in general multiply connected domains. We validate the computed stream function for Stokes flows between two cylinders with different boundary conditions against an analytical solution [17] in section 5.

We emphasise that the main contribution of this paper is the development of a new algorithm for solving Stokes flow problems, and a summary of our numerical method is given in subsection 6.1. In subsection 6.2, we apply LARS to compute various 2D Stokes flow problems to demonstrate its broad application. Using these examples, we show that rational approximation can now be used to compute 2D Stokes flows in general domains. The computation usually takes a fraction of a second for a solution to at least 6-digit accuracy.

2. 2D Stokes flow and biharmonic equations. Define (x, y) as the usual Cartesian coordinate system with associated velocity components (u, v). The steady-state Stokes equations in two dimensions are

$$\mu\nabla^2\mathbf{u} = \nabla p, \tag{2.1}$$

$$\nabla \cdot \mathbf{u} = 0, \tag{2.2}$$

where $\mathbf{u} = (u, v)^T$ is the 2D velocity field, p is the pressure and μ is the viscosity. We consider 2D Stokes flow problems (2.1) and (2.2) in a bounded domain Ω. Two boundary conditions are imposed on the domain boundary $\partial\Omega$.

Since the flow is 2D and incompressible, a stream function ψ can be defined by

$$u = \frac{\partial \psi}{\partial y}, \; v = -\frac{\partial \psi}{\partial x}. \tag{2.3}$$

Next we define the vorticity magnitude ω as

$$\omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} = -\nabla^2 \psi. \tag{2.4}$$

Taking the curl of (2.1) gives

$$\nabla^2 \omega = 0. \tag{2.5}$$

The stream function thus satisfies the biharmonic equation

$$\nabla^4 \psi = 0. \tag{2.6}$$

The Stokes problem now becomes that of finding a solution for the biharmonic equation (2.6) in the domain of interest, subject to given boundary conditions. In the complex plane $z = x + iy$, where $x, y \in \mathbb{R}$ and $i = \sqrt{-1}$, we have

$$\frac{\partial}{\partial z} = \frac{1}{2}\left(\frac{\partial}{\partial x} - i\frac{\partial}{\partial y}\right), \; \frac{\partial}{\partial \bar{z}} = \frac{1}{2}\left(\frac{\partial}{\partial x} + i\frac{\partial}{\partial y}\right), \tag{2.7}$$

where $\bar{z} = x - iy$ is the complex conjugate of z. Equation (2.6) can then be written in complex form as

$$\frac{\partial^4 \psi}{\partial^2 z \partial^2 \bar{z}} = 0, \tag{2.8}$$

which has a solution

$$\psi(z, \bar{z}) = \mathrm{Im}[\bar{z} f(z) + g(z)], \tag{2.9}$$

where $f(z)$ and $g(z)$ are two analytic functions, known as Goursat functions [23].

The flow velocity, pressure and vorticity can be expressed in terms of Goursat functions as

$$u - iv = -\overline{f(z)} + \bar{z} f'(z) + g'(z), \tag{2.10}$$

$$\frac{p}{\mu} - i\omega = 4f'(z), \tag{2.11}$$

where $\overline{f(z)}$ is the complex conjugate of $f(z)$. In the rest of this paper, (2.10) and (2.11) will be used to determine the Goursat functions by imposing boundary conditions on $\partial\Omega$ (e.g., $-\overline{f(z_0)} + \bar{z}_0 f'(z_0) + g'(z_0) = 0$ for a zero velocity boundary condition at z_0), from which we can then calculate quantities of interest in the simulation domain Ω.

3. The lightning solver. Our numerical method is developed from the recently introduced "lightning" solver for 2D Stokes flow [9], which uses rational functions with clustered poles near sharp corners to approximate the Goursat functions, and from the related AAA-least squares method for problems with curved boundaries [10] (see the next section). These methods are based on the general idea of solving the Laplace equation in polygons or curved domains using rational functions [22]. A typical rational function consisting of m poles $\beta_1, ..., \beta_m$ and a polynomial of degree n has the form

$$r(z) = \sum_{j=1}^{m} \frac{a_j}{z - \beta_j} + \sum_{j=0}^{n} b_j z^j, \tag{3.1}$$

where a_j and b_j are complex coefficients to be determined from the boundary conditions. In [22] the first and second parts of (3.1) are called the "Newman" and "Runge" terms, respectively.

For Stokes flow problems, two rational functions, $\hat{f}(z)$ and $\hat{g}(z)$, are defined for the two Goursat functions, $f(z)$ and $g(z)$:

$$\hat{f}(z) = \sum_{j=1}^{m} \frac{a_j^f}{z - \beta_j} + \sum_{j=0}^{n} b_j^f z^j, \tag{3.2}$$

$$\hat{g}(z) = \sum_{j=1}^{m} \frac{a_j^g}{z - \beta_j} + \sum_{j=0}^{n} b_j^g z^j. \tag{3.3}$$

Determining these unknown coefficients is a non-linear problem because of the Newman terms of (3.1). However, it becomes a standard linear least-squares problem if we fix the location of poles beforehand. It has been shown in previous work [9, 22] that root-exponential convergence can be achieved if the poles are exponentially clustered near each sharp corner of the domain. For a polygonal domain Ω with K corners $w_1, ..., w_K$, we place N poles near each corner using

$$\beta_{kn} = w_k + Le^{i\theta_k} e^{-\sigma(\sqrt{N} - \sqrt{n})}, \; k = 1, ..., K, \; n = 1, ..., N, \tag{3.4}$$

where L is the characteristic length scale, θ_k is the exterior bisector of corner w_k and σ is a constant (normally set as 4), as [9, 22]. Note that these lightning poles are only used when the domain boundary has sharp corners and they do not appear in smooth boundary problems. As we will show in the next section, the pole vector β for smooth boundaries can be obtained easily using the AAA algorithm [36].

The representation (3.1) of a rational function can be ill-conditioned. Here we carry out a Vandermonde with Arnoldi (VA) orthogonalization [8] for the Runge terms and the group of poles near each corner to construct a well-conditioned basis for the linear system. There are two issues to note here. Firstly, unlike Laplace problems, Stokes flow problems involve the derivatives of Goursat functions. These need to be calculated based on the new basis from the VA orthogonalization (see Equations (4.4)–(4.6) in [9]). Secondly, the Laurent series used in multiply connected problems will also need to be orthogonalized, which will be further discussed in Section 5.

The sample points are selected along the boundary $\partial\Omega$, and are also clustered near the sharp corners [9, 22]. Along smooth boundary components, the sample points are evenly distributed, although improvements would certainly be possible here for cases of strong curvature. By applying two boundary conditions at each sample point using (2.10) or (2.11), we obtain a well-conditioned least-squares problem $Ax \approx b$. The real matrix A has size $2M \times 4(m + n + 1)$ and the real vector b has size $2M$, where M is the number of sample points. The columns of the matrix A correspond to the real and imaginary parts of the complex coefficients a_j^f, b_j^f, a_j^g and b_j^g, while its rows correspond to the two boundary conditions applied at M sample points, the values of which are stored in the vector b. The solution x gives the optimal coefficients for the two rational approximations $\hat{f}(z)$ and $\hat{g}(z)$ (for the two Goursat functions $f(z)$ and $g(z)$), which satisfy the boundary conditions on $\partial\Omega$ in a least-squares sense. The least-squares problem can be solved easily using the backslash command in MATLAB.

4. AAA rational approximation for a curved boundary. In biological and engineering applications, many boundary components are curved [26, 28, 42, 43]. The solutions associated with such problems are analytic. However, they may be analytically continuable only a very short distance across the boundary, an effect known as the "crowding phenomenon" [7, 11, 21, 34]. In such cases, accurate rational approximations will need to have poles very close to the boundary.

One way to tackle this phenomenon in rational approximation is to use the AAA algorithm [36]. In brief, the AAA algorithm searches for poles for a rational approximation speedily, reliably and automatically. This algorithm has been found to be very fast and effective in rational approximation of conformal maps near singularities [21]. In a recent work, Costa and Trefethen [10] applied the AAA algorithm to solve Laplace problems. The algorithm was shown to be able to place poles near the boundary of an arbitrary domain in a configuration effective for rational approximation. Using these poles, 8-digit accuracy was easily achieved.

Here we further apply AAA rational approximation to Stokes flow. We demonstrate the outstanding ability of this algorithm to place poles effectively near curved boundaries using an example case of Stokes flow in a channel with a smooth constriction. We choose this case because the pressure drop across the constriction for a given inlet flux has been determined semi-analytically using extended lubrication theory [43]. We first present the problem in subsection 4.1, before computing the problem using polynomials in subsection 4.2 and rational functions in subsection 4.3.

4.1. Stokes flow in a smoothly constricted channel. We consider Stokes flow through a channel with characteristic length L_0 and height h_0 with $\delta = h_0/L_0$, and inlet flux q_0. We introduce dimensionless variables after [43]:

$$X = \frac{x}{L_0},\ Y = \frac{y}{h_0},\ U = \frac{u}{q_0/h_0},\ V = \frac{v}{q_0/L_0},\ P = \frac{p}{\mu q_0 L_0/h_0^3}, \tag{4.1}$$

where we denote dimensionless variables with capitals and the dimensionless velocity field is $\mathbf{U} = (U, V)^T$. Equations (2.1) and (2.2) can then be written in the dimensionless form as

$$\delta^2 \frac{\partial^2 U}{\partial X^2} + \frac{\partial^2 U}{\partial Y^2} = \frac{\partial P}{\partial X}, \tag{4.2}$$

$$\delta^4 \frac{\partial^2 V}{\partial X^2} + \delta^2 \frac{\partial^2 V}{\partial Y^2} = \frac{\partial P}{\partial Y}, \tag{4.3}$$

$$\frac{\partial U}{\partial X} + \frac{\partial V}{\partial Y} = 0. \tag{4.4}$$

Figure 1a presents the Stokes flow problem in a channel with a smooth constriction from $X = -1$ to $X = 1$ and $\delta = 1$. The shape function for constriction is defined as

$$H(X) = 1 - \frac{\lambda}{2}(1 + \cos(\pi X)),\ 0 \le \lambda < 1, \tag{4.5}$$

where λ is the maximum dimensionless amplitude of constriction. The boundary curves corresponding to different values of λ from 0 to 0.8 are shown in Figure 1b.

A Poiseuille inlet velocity profile $\mathbf{U}(Y) = (6(Y - Y^2), 0)^T$, zero velocity on the walls ($\mathbf{U} = \mathbf{0}$), and a parallel outflow profile with zero pressure ($V = 0$, $P = 0$) are imposed on the domain boundary.

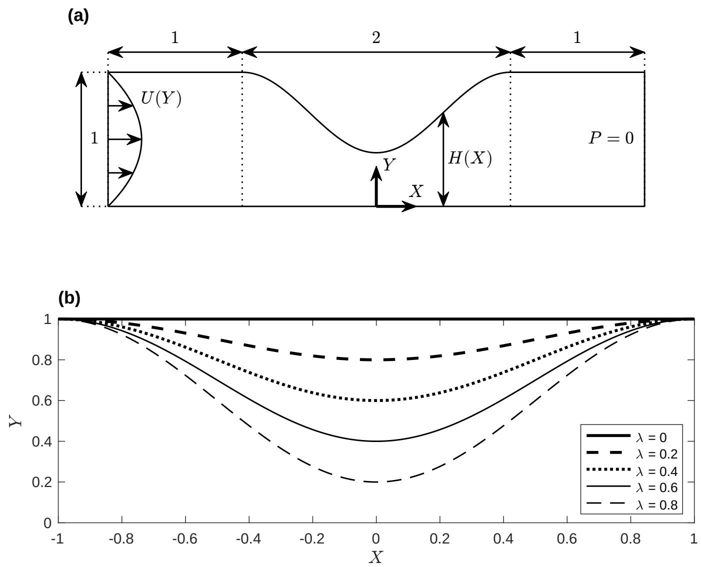

FIG. 1. *Schematic of Stokes flow through a smoothly constricted channel, after* [43]. *(a) Geometry and boundary conditions. (b) Shape function $H(X)$ of the upper boundary for different λ.*

For $0 < \delta \ll 1$, corresponding to a channel with small aspect ratio, we can use lubrication theory [37]. We expand U, V and P in terms of δ:

$$U(X,Y) = U_0(X,Y) + \delta^2 U_2(X,Y) + \delta^4 U_4(X,Y) + \cdots , \tag{4.6}$$

$$V(X,Y) = V_0(X,Y) + \delta^2 V_2(X,Y) + \delta^4 V_4(X,Y) + \cdots , \tag{4.7}$$

$$P(X,Y) = P_0(X,Y) + \delta^2 P_2(X,Y) + \delta^4 P_4(X,Y) + \cdots , \tag{4.8}$$

and solve (4.2)–(4.4) at different orders of δ. The solutions for the pressure drop across the smoothly constricted channel at different orders of δ have been given in [43] :

$$\Delta P_0(\lambda) = \frac{3(3\lambda^2 - 8\lambda + 8)}{(1-\lambda)^{5/2}}, \tag{4.9}$$

$$\Delta P_2(\lambda) = \frac{12\pi^2\lambda^2}{5(1-\lambda)^{3/2}}, \tag{4.10}$$

$$\Delta P_4(\lambda) = \frac{8\pi^4(428(-1+\sqrt{1-\lambda}) - 214(-2+\sqrt{1-\lambda})\lambda - 53\lambda^2)}{175\sqrt{1-\lambda}}. \tag{4.11}$$

In classical lubrication theory (CLT), only the leading order solution ΔP_0 is used for approximating the pressure drop. Tavakol et al. [43] approximate the solution using the leading order term with higher order correction terms, leading to 2nd-order extended lubrication theory (ELT):

$$\Delta P(\lambda) = \Delta P_0(\lambda) + \delta^2 \Delta P_2(\lambda) + \mathcal{O}(\delta^4), \tag{4.12}$$

and 4th-order ELT:

$$\Delta P(\lambda) = \Delta P_0(\lambda) + \delta^2 \Delta P_2(\lambda) + \delta^4 \Delta P_4(\lambda) + \mathcal{O}(\delta^6). \tag{4.13}$$

Tavakol et al. [43] show that including higher order terms significantly improves the approximation accuracy of the pressure drop across the constriction for channels with high aspect ratios, e.g. when δ approaches 1. In the following sections, we set $\delta = 1$ for all our computations, and we compare our results with the 4th-order ELT for the same δ.

4.2. Polynomial approximation for smooth Stokes flow problems. We first approximate the Stokes flow problem using the lightning Stokes solver [9]. When used without AAA, the lightning Stokes solver tackles smooth boundary problems by means of the polynomial or "Runge" term of (3.1). A recent example is provided as Figure 7.3 of [9], where 10-digit accuracy is achieved in a smooth bent channel using a polynomial of degree 300.

However, the polynomial approximation behaves poorly in this constricted channel problem, especially when the amplitude λ is close to 1, due to the crowding phenomenon. Figure 2 presents the pressure drop across the constriction for different λ using polynomial approximations of degrees 200, 300 and 400. The numbers of sample points for the approximations are 4200, 6300 and 8400, scaled with the polynomial degree. For example, when the polynomial degree is 200, there are 600 points evenly distributed on each segment of the domain boundary. We treat the upper boundary as 4 boundary segments $X \in (-2, -1)$, $X \in (-1, 0)$, $X \in (0, 1)$ and $X \in (1, 2)$, where 600 points are sampled on each segment. In the complex plane $Z = X + iY$, the pressure drop is calculated between $Z = -1 + 0.5i$ and $Z = 1 + 0.5i$ using (2.11).

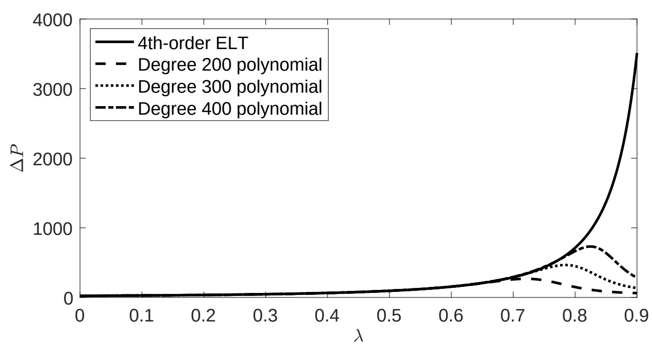

FIG. 2. *Pressure drop as a function of constriction parameter λ when $\delta = 1$ computed using polynomials with degrees* 200, 300 *and* 400*. The simulation results are compared with the solutions derived using a* 4*th-order extended lubrication theory* [43]*. The numbers of sample points for the polynomial approximations are* 4200*,* 6300 *and* 8400*.*

In Figure 2, the polynomial approximation is compared with the semi-analytical solution approximated by 4th-order ELT [43]. For $\lambda \leq 0.7$, the polynomial approximation provides a reasonable estimate of the pressure drop across the constriction (compared with the 4th-order analytical approximation). However, the degree 200 polynomial fails to approximate the pressure drop for $\lambda > 0.7$, followed by the degree 300 polynomial for $\lambda > 0.75$ and the degree 400 one for $\lambda > 0.8$. Note that the computational cost rises sharply as we increase the polynomial degree from 200 to 400. Further increasing the degree of the polynomials may approximate our problem at a larger λ, but this is certainly not a practical method for all $0 \leq \lambda < 1$. This is where the AAA algorithm comes into play.

4.3. AAA rational approximation for the upper boundary. The key idea now is to use the AAA algorithm to place poles outside the curved boundary (which is the upper boundary of the channel in this case) [36] to help the lightning solver compute the Stokes flow [9]. This method has proven to be very effective for solving

Laplace problems [10]. The AAA rational approximation can be computed using the MATLAB code `aaa.m` in Chebfun [14], and the workflow is simple:

1. Create a vector Z_b of sample points along the curved boundary.
2. Apply a boundary function F to Z_b, e.g., the Schwarz function [11]: $F = \overline{Z_b}$.
3. Run the AAA algorithm for these sample points and boundary values.
4. Remove the poles inside the domain Ω.

In MATLAB, the poles can be obtained easily by executing

```
F = conj(Zb);
[r,pol] = aaa(F,Zb,'tol',1e-8);
inpoly = @(z,w) inpolygon(real(z),imag(z),real(w),imag(w));
jj = inpoly(pol,Z);
beta = pol(~jj);
```

where Z_b is the vector of sample points along the curved boundary we aim to place poles near, and Z is the vector of sample points along all of $\partial\Omega$. For AAA-lightning computations of the constricted channel problem, we sample 600 points clustered towards singularities at two ends using `tanh(linspace(-14,14,600))` on each boundary segment.

Here we use the Schwarz function [11] as the boundary function and set the tolerance of the AAA algorithm as 10^{-8} for fast computation. Our numerical experiments show that the use of different boundary functions has negligible effect on the location of poles for this geometry. Here we use the Schwarz function, because it is purely based on the boundary geometry instead of the boundary data, so the poles of the rational approximation of the Schwarz function capture the singularities of the boundary geometry. The pole vector `beta` will be used to construct the Newman part of our rational function (3.1). For better numerical stability, VA orthogonalization [8] was performed on these poles and the polynomial using `VAorthog(Z,n,beta)`. This is the same as in the original lightning Stokes solver [9] except that the vector of poles is obtained using AAA. We will call these poles placed by the AAA algorithm "AAA poles" in the rest of this paper.

Figure 3 presents the Stokes flow in the constricted channel for different λ computed using the algorithm described above. The streamlines in each case are represented by light grey lines, while a colour scale shows the velocity magnitude. The AAA poles are marked as red dots. Using these poles and a polynomial of degree 100, a 5 to 6-digit accurate solution can be computed in 1-2 seconds on a standard laptop. For each computation, running the AAA algorithm, constructing the rational function basis, and solving the least-squares problem takes about 0.8, 0.4 and 0.2 second, respectively. Note that only 3-digit accuracy is achieved in the last case, with $\lambda = 0.9$, although the streamlines are qualitatively promising. For higher accuracy, higher degrees of polynomials and more sample points are required. In this paper, "α-digit accuracy" means that the maximum error of the approximation on the domain boundary is below $10^{-\alpha}$.

In Figure 3, the AAA algorithm places poles vertically along the centreline of the constriction, with clustering near the bottom. This phenomenon is very similar to that of the poles placed near a sharp corner using (3.4) in the original lightning Stokes solver [9]. However, for this specific problem, it is interesting to note that the AAA poles are clustered towards a point slightly above the boundary instead of on the boundary. This presumably corresponds to a branch point of the analytic continuation across the boundary.

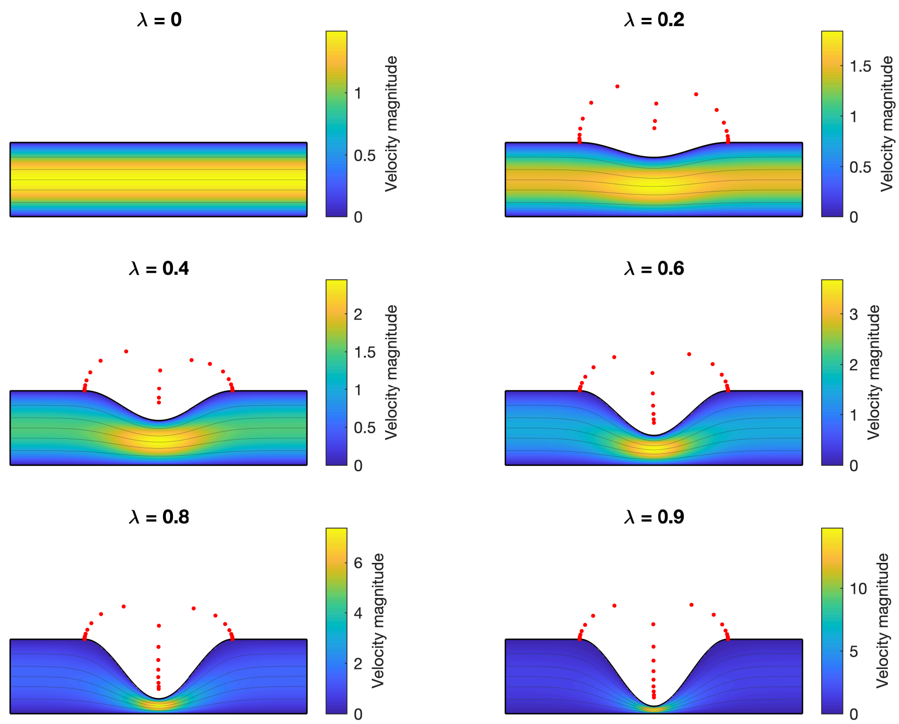

FIG. 3. *Stokes flow in a smoothly constricted channel for different* λ *from* 0 *to* 0.9 *and* $\delta = 1$. *The solution is computed using the lighting solver with a polynomial of degree* 100 *with poles placed by the AAA algorithm. The locations of poles are marked by red dots. The streamlines and velocity magnitude in each case are represented by light grey lines and a colour scale, respectively. Note that the colour scale has a different range for each case, scaled by the maximum velocity magnitude. The computation for each case takes* 1–2 *seconds on a standard laptop.*

In addition, AAA places poles near the juncture points where the horizontal part transitions into the curved part. The AAA algorithm detects these points as singularities, because the transitions are not twice continuously differentiable (although the first derivative is continuous, see (4.5)).

Figure 4 shows a comparison of pressure drop across the constriction approximated by lubrication theory at different orders of δ and the AAA-lightning computation. The gap between lubrication theory and lightning simulation becomes smaller when higher order terms are included in the asymptotic analysis. Compared with our computation, the maximum differences in ΔP for all λ are approximately 20% and 4%, respectively, when using CLT and 2nd-order ELT. These figures agree with the gaps between the approximations and computations using a finite element method reported in [43]. For 4th-order ELT, the maximum difference in ΔP reduces to 2.2%.

5. Series methods for multiply connected domains. In the previous paper [9], application of the lightning Stokes solver was focused on simply connected domains. However, there has been increasing interest in 2D Stokes flows in multiply connected domains [15, 16, 17, 18, 31, 33, 39, 40]. In this section, we describe a method to extend the lightning Stokes solver to multiply connected problems.

When solving a Laplace problem in a multiply connected domain, it is known that

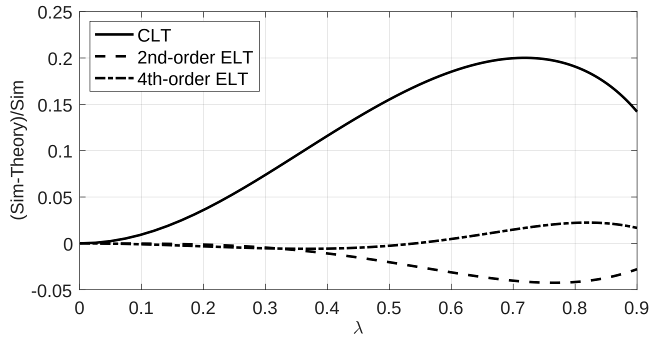

Fig. 4. *Relative differences between the pressure drop across the constriction when $\delta = 1$ approximated by lubrication theory at different orders [43] and the AAA-lightning computation, which we presume is effectively exact for the purposes of this comparison. The computation uses a degree 100 polynomial with poles placed by the AAA algorithm.*

only a logarithmic term and a Laurent series are needed for each smooth hole [44]. The logarithmic term prevents the solution from being multi-valued, thanks to the "logarithmic conjugation theorem" presented by Axler [3], with its proof dating to Walsh in 1929 [46]. The series method has been shown to be effective for computing conformal maps in multiply connected domains [45].

Similar series methods have also been applied to 2D Stokes flow problems in multiply connected domains by Price et al. [39] and Finn et al. [18]. However, these applications are limited to 2D Stokes flows in domains bounded by cylinders. With the AAA algorithm [36], VA orthogonalization [8] and the lightning solver [9, 22], we are now able to compute 2D Stokes flows in more general multiply connected domains and thus address much broader applications.

5.1. Algorithm. For multiply connected domain problems with p smooth holes, we define a rational function (before adding the logarithmic terms) in the form

$$r(z) = \sum_{j=1}^{m} \frac{a_j}{z-\beta_j} + \sum_{j=0}^{n} b_j z^j + \sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}(z-z_i)^{-j}, \tag{5.1}$$

where the first and second parts are the Newman and Runge terms from (3.1). The third part represents a Laurent series expansion to degree q at point z_i in the ith hole.

Based on the logarithmic conjugation theorem [3], a logarithmic term is needed for each Goursat function. Moreover, unlike Laplace problems, a derivative term $g'(z)$ also appears in (2.10), which describes the velocity field in Stokes flow problems. The velocity components u and v are each expressed as

$$u(z) = \mathrm{Re}[-f(z) + \bar{z}f'(z) + g'(z)], \tag{5.2}$$

$$v(z) = \mathrm{Im}[-f(z) - \bar{z}f'(z) - g'(z)]. \tag{5.3}$$

To ensure the velocity field is not multi-valued, an extra term $(z-z_i)\log(z-z_i) - z$ is required in $g(z)$ for the ith hole. So the imaginary part of $\log(z-z_i)$ term in $g'(z)$ and $f(z)$ can cancel out in (5.2) and (5.3) [39]. The rational representation of Goursat functions can thus be written as

$$\hat{f}(z) = \sum_{j=1}^{m} \frac{a_j^f}{z-\beta_j} + \sum_{j=0}^{n} b_j^f z^j + \sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}^f (z-z_i)^{-j} + \sum_{i=1}^{p} d_i^f \log(z-z_i), \tag{5.4}$$

$$\begin{aligned}\hat{g}(z) = \sum_{j=1}^{m} \frac{a_j^g}{z-\beta_j} + \sum_{j=0}^{n} b_j^g z^j + \sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}^g (z-z_i)^{-j} + \sum_{i=1}^{p} d_i^g \log(z-z_i)\\ - \sum_{i=1}^{p} \overline{d_i^f}[(z-z_i)\log(z-z_i) - z],\end{aligned} \tag{5.5}$$

where a_j^f, b_j^f, c_{ij}^f, d_i^f, a_j^g, b_j^g, c_{ij}^g and d_i^g are complex coefficients to be determined by solving a least-squares problem.

In the lightning Stokes solver, VA orthogonalization has been carried out for both the Newman and the Runge parts. The VA orthogonalization for the polynomial part can be found in the Section 4.1 of [9]. The VA orthogonalization for the Laurent series is very similar to that for polynomials, and can be realised by appending a new module at the end of the existing MATLAB code `VAorthog` (see Appendix A in [9]):

```
Q = ones(M,1); H = zeros(nl+1,nl);
for k = 1:nl
   q = 1./(Z-Z_i).*Q(:,k);
   for j = 1:k, H(j,k) = Q(:,j)'*q/M; q = q - H(j,k)*Q(:,j); end
   H(k+1,k) = norm(q)/sqrt(M); Q(:,k+1) = q/H(k+1,k);
end
Hes{length(Hes)+1} = H; R = [R Q(:,2:end)];
```

where `nl` is the degree of Laurent series (i.e. q in (5.4) and (5.5)), `Hes` is the upper-Hessenberg matrix from the Arnoldi process. Similarly, the new VA basis can be constructed using `VAeval` with an additional module appended in the end:

```
H = Hes{1}; Hes(1) = [];
Q = ones(M,1); D = zeros(M,1);
Zpki = 1./(Z-Z_i); Zpkid = -1./(Z-Z_i).^2;
for k = 1:nl
    hkk = H(k+1,k);
    Q(:,k+1) = (Q(:,k).*Zpki - Q(:,1:k)*H(1:k,k))/hkk;
    D(:,k+1) = (D(:,k).*Zpki - D(:,1:k)*H(1:k,k) + Q(:,k).*Zpkid)/hkk;
end
R0 = [R0 Q(:,2:end)]; R1 = [R1 D(:,2:end)];
```

where `R0` is the function basis for the Goursat functions $f(z)$ and $g(z)$ and `R1` is the function basis for their derivatives $f'(z)$ and $g'(z)$. The first column representing the constant term is always omitted at the storing step, since the constant term has been included in the polynomial part. The logarithmic terms are added in the MATLAB code `makerows`, when evaluating the function values on the domain boundary $\partial\Omega$. Example codes for computing Stokes flow in general multiply connected domains can be found in Appendix A.

5.2. Stokes flow between two translating and rotating cylinders. Here we consider Stokes flow between two cylinders to illustrate the speed and accuracy of our new Stokes flow solver via comparison of numerical results with Finn and Cox [17], based on previous work on cylinders with simpler motions [6, 20, 41]. Figure 5 shows the problem setting, where the outer cylinder has a radius of a_{out} centred at the origin and the inner cylinder has a radius of a_{in} centred at $(\epsilon, 0)$. The outer and inner cylinders rotate with angular velocities ω_{out} and ω_{in}, respectively. The inner cylinder can also translate with velocity $(u^*, v^*)^T$. We describe the problem using dimensionless variables:

$$(u, v) = u^*(U, V), \quad p = \frac{u^* a_{out}}{\mu} P, \tag{5.6}$$

where capitals denote dimensionless variables. The system is then characterised by the following dimensionless parameters

$$V^* = \frac{v^*}{u^*}, \; A_{in} = \frac{a_{in}}{a_{out}}, \; E = \frac{\epsilon}{a_{out}}, \; (\Omega_{in}, \Omega_{out}) = \frac{a_{out}}{u^*}(\omega_{in}, \omega_{out}). \tag{5.7}$$

The analytical solution to this problem can be derived by superposing solutions corresponding to each type of motion (rotational and translational), see Equation (68) in [17] (the details are omitted here).

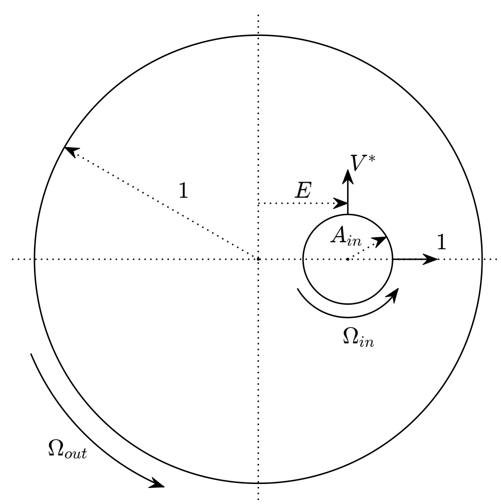

FIG. 5. *Schematic of a translating and rotating cylinder in a rotating cylinder, after* [17] *(dimensionless quantities).*

We used our algorithm to compute the Stokes flow in all nine example cases presented in Figure 9 of [17]. Our results are shown in Figure 6. The parameter values for each case are listed in Table 1. The same parameter values were used in [17], except that we change $E = 0.7$ to $E = 0.65$ in case 'g' to prevent the boundaries of the two cylinders touching each other.

For each case, 100 sample points are evenly distributed along the inner cylinder boundary, with another 500 points along the outer cylinder boundary slightly clustered towards the narrower gap between two cylinders:

```
pw = ceil(1/(1-epsilon))+1;
sp = tanh(linspace(-pw,pw,500));
Z_b = a_out*exp(1i*pi*(sp-1)');
```

The solution is computed using a rational approximation (5.1) consisting of a degree 20 polynomial and two degree 50 Laurent series about $(E, 0)$ and $(1/E, 0)$. The computation for each case takes tens of milliseconds on a standard laptop.

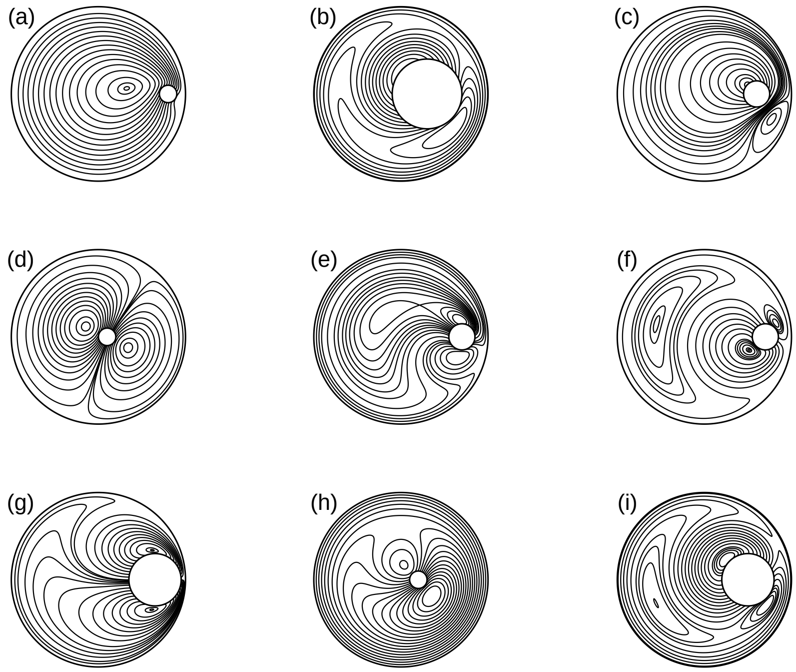

FIG. 6. *Streamlines for Stokes flow between two cylinders for nine different boundary conditions, following* [17]*. The parameter values are listed in Table* 1*. The stream function is* 0 *on the outer cylinder.*

TABLE 1
Parameter values for nine example cases as presented in Figure 6*.*

Case	A_{in}	E	V^*	Ω_{in}	Ω_{out}
a	0.1	0.8	2	-3	1
b	0.4	0.3	1	5	-3
c	0.15	0.6	1	10	0
d	0.1	0.1	2	0	0
e	0.15	0.7	0	3.33	0.66
f	0.15	0.7	-1	0	0.66
g	0.3	0.65	0	0	-0.2
h	0.1	0.2	1	5	-1
i	0.3	0.5	1	2	-2

Here we use an additional Laurent series about the reflection of the centre of inner cylinder to better compute the flow field in the narrow gap between two cylinders in cases ‘a’ and ‘g’. The algorithm using two Laurent series is found to be much more efficient than using a Laurent series about the inner cylinder with a much higher degree polynomial. When the inner cylinder is very close to the outer cylinder, the analytic continuation of Goursat functions may have branch points outside the outer cylinder, leading to the crowding phenomenon that we have tackled using the AAA algorithm in the last section. Hence one may develop another algorithm that places AAA poles near the gap outside the domain. However, controlling the region of AAA poles and dealing with the logarithmic terms can be challenging and requires future work.

Next, we validate the values of the stream function ψ computed using our Stokes flow solver against the analytical solution [17]. Figure 7 shows the pointwise error (Error = Sim−Theory) of our computation of the stream function in all nine cases. All solutions are obtained to 12 to 14-digit accuracy, which is close to machine precision.

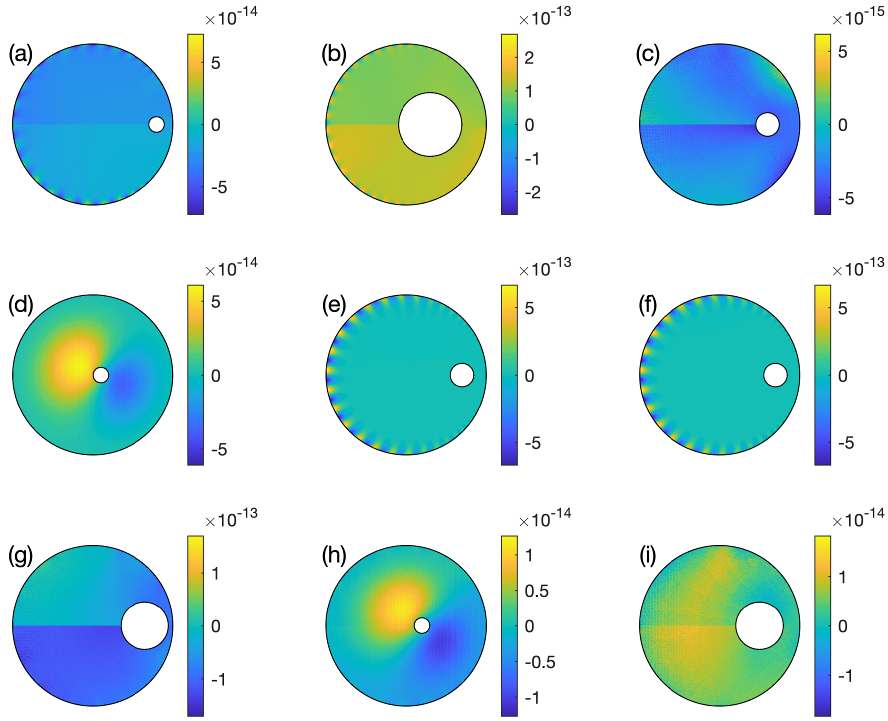

FIG. 7. *Pointwise error of the stream function computed using our Stokes flow solver in all nine cases.*

6. Algorithm summary and applications. In this section, we first summarise the LARS algorithm, and then apply it to a variety of 2D Stokes flow problems to demonstrate the utility of the algorithm.

6.1. The LARS algorithm. There are three steps to compute the solution to a 2D Stokes flow problem using the LARS algorithm:

1. Place the poles and sample points. For a given domain Ω, we first identify corner singularities and cluster lightning poles exponentially nearby following the lightning algorithm [9, 22]. We then identify smooth boundaries and approximate the Schwarz function on each section using the AAA algorithm [36]. After removing the poles inside Ω, we obtain the AAA poles for approximating these smooth boundaries [10]. The lightning poles and the AAA poles form the vector β_j to construct the Newman part in (5.4) and (5.5). The sample points are clustered towards corner singularities along the domain boundary $\partial\Omega$. For smooth domain boundaries and holes, we take equispaced sample points. All sample points along the entire $\partial\Omega$ are stored in a vector $Z = (Z_1, Z_2, ..., Z_M)^T$.

2. Construct the rational function bases. We perform VA orthogonalization [8] for the Newman terms, the Runge terms, and the Laurent series corresponding to each hole for all sample points Z. Using the VA orthogonalization we obtain a well-conditioned rational function basis R_0 (spanning the same spaces as the original basis) to evaluate every sample point in Z, and its derivative R_1. The matrices R_0 and R_1 have size $M \times (m + n + pq + 1)$, where each row corresponds to a sample point and each column corresponds to a coefficient a_j, b_j or c_{ij}.

3. Solve the least-squares problem and compute physical quantities. We now impose two boundary conditions on $\partial\Omega$ (with Z being the vector of sample points) to compute the coefficient values in two rational functions $\hat{f}(z)$ and $\hat{g}(z)$ that approximate the Goursat functions. For simplicity, it is assumed here that we have the boundary condition $(\tilde{u}(Z_i), \tilde{v}(Z_i))^T$ at each sample point Z_i on $\partial\Omega$. One can impose ψ, p, ω or a component in the 2D stress tensor easily after minor changes from the linear system presented below. Based on (5.2) and (5.3), we construct a linear system $Ax \approx b$ using R_0, R_1 and logarithmic terms, which will be solved using a least-squares approach to find the coefficient vector x that minimises $||Ax - b||_2$. The matrix A consists of 2×8 blocks:

$$
A = \begin{bmatrix} \mathrm{Re}\{cZ \times R_1 - R_0\} & \mathrm{Re}\{R_1\} & \mathrm{Re}\{cZ \times oZ - 2lZ\} & \mathrm{Re}\{oZ\} \\ \mathrm{Im}\{-cZ \times R_1 - R_0\} & \mathrm{Im}\{-R_1\} & \mathrm{Im}\{-cZ \times oZ\} & \mathrm{Im}\{-oZ\} \end{bmatrix.
$$

$$
\begin{matrix} -\mathrm{Im}\{cZ \times R_1 - R_0\} & -\mathrm{Im}\{R_1\} & -\mathrm{Im}\{cZ \times oZ\} & -\mathrm{Im}\{oZ\} \\ \mathrm{Re}\{-cZ \times R_1 - R_0\} & \mathrm{Re}\{-R_1\} & \mathrm{Re}\{-cZ \times oZ - 2lZ\} & \mathrm{Re}\{-oZ\} \end{bmatrix}, \tag{6.1}
$$

where we have

$$
cZ = \begin{bmatrix} \overline{Z_1} & & & \\ & \overline{Z_2} & & \\ & & \ddots & \\ & & & \overline{Z_M} \end{bmatrix}, \quad oZ = \begin{bmatrix} \frac{1}{Z_1 - z_1} & \frac{1}{Z_1 - z_2} & \cdots & \frac{1}{Z_1 - z_p} \\ \frac{1}{Z_2 - z_1} & \frac{1}{Z_2 - z_2} & \cdots & \frac{1}{Z_2 - z_p} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{Z_M - z_1} & \frac{1}{Z_M - z_2} & \cdots & \frac{1}{Z_M - z_p} \end{bmatrix},
$$

$$
lZ = \begin{bmatrix} \log(Z_1 - z_1) & \log(Z_1 - z_2) & \dots & \log(Z_1 - z_p) \\ \log(Z_2 - z_1) & \log(Z_2 - z_2) & \dots & \log(Z_2 - z_p) \\ \vdots & \vdots & \ddots & \vdots \\ \log(Z_M - z_1) & \log(Z_M - z_2) & \dots & \log(Z_M - z_p) \end{bmatrix}. \tag{6.2}
$$

The vector $b = (\tilde{u}(Z), \tilde{v}(Z))^T$ corresponds to the two boundary conditions on $\partial\Omega$. We compute the least-squares problem using the backslash command in MATLAB to obtain

$$
x = [\mathrm{Re}\{a_j^f, b_j^f, c_{ij}^f, a_j^g, b_j^g, c_{ij}^g, d_i^f, d_i^g\}, \mathrm{Im}\{a_j^f, b_j^f, c_{ij}^f, a_j^g, b_j^g, c_{ij}^g, d_i^f, d_i^g\}]^T, \tag{6.3}
$$

and thus all complex coefficients in $\hat{f}(z)$ and $\hat{g}(z)$, which satisfy the given boundary conditions in a least-squares manner.

After finding the Goursat functions, we construct function handles for physical quantities $u(z)$, $v(z)$, $p(z)$ and $\omega(z)$ using (2.10) and (2.11). The function handles enable the evaluation of any required physical quantity at any given point in Ω.

6.2. Application to other Stokes flow problems. Figure 8 shows 2D Stokes flow around a translating and rotating elliptical cylinder inside a fixed elliptical cylinder. This setting has potential biomedical applications to kidney stone removal problems [48]. The parameter values for Case 'c' in Table 1 are used here to set the boundary conditions, but with the outer cylinder replaced by an ellipse with eccentricity 0.6 and the inner cylinder by an ellipse with eccentricity 0.8, and A_{out} and A_{in} now representing the lengths of the semi-minor axes. The solution is computed to 7-digit accuracy with a degree 40 polynomial and a degree 120 Laurent series. Note that for non-circular holes, a higher degree Laurent series is usually required for good precision.

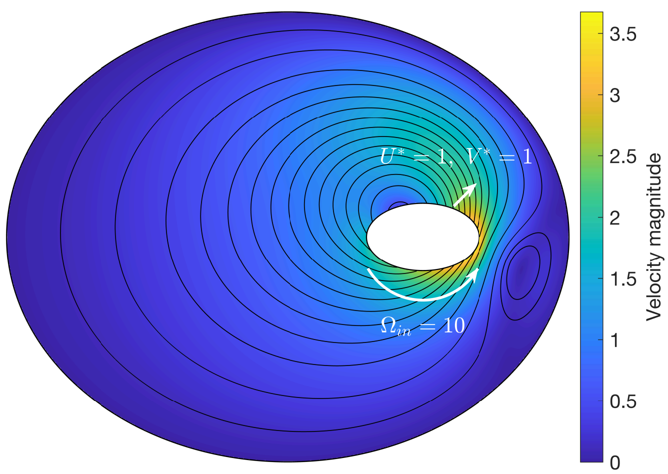

FIG. 8. *Stokes flow around a translating and rotating elliptical cylinder inside a fixed elliptical cylinder. The same parameter values as for Case 'c' in Table* 1 *are used here. The outer ellipse has eccentricity* 0.6 *and the inner ellipse has eccentricity* 0.8*. The translation and rotation of the inner ellipse are indicated by white arrows.*

Figure 9 shows the Stokes flow around a heart-shaped hole in a channel, illustrating how readily our solver can be extended to other shapes. Boundary conditions of constant pressure and parallel flow are imposed at the channel inlet and outlet, and a zero velocity condition is imposed on the walls. The solution is computed to 10-digit accuracy (12 to 13-digit accuracy along the hole boundary) with a degree 120 polynomial and a degree 80 Laurent series in 0.9 second. One can compute the solution to 6-digit accuracy with a degree 80 polynomial and a degree 40 Laurent series in 0.3 second.

Figure 10 shows two pairs of Moffatt eddies [35] near the cusp on the left side of the heart-shaped hole in Figure 9. If the stream function is set to 0 along the centreline, the first pair have stream functions on the order of 10^{-7} and the second on the order of 10^{-12}. The third pair in this theoretically infinite series of eddies would be at a level near machine precision. This example demonstrates the great accuracy of our methods. For this problem, it takes 45 microseconds per point to compute a stream function or velocity.

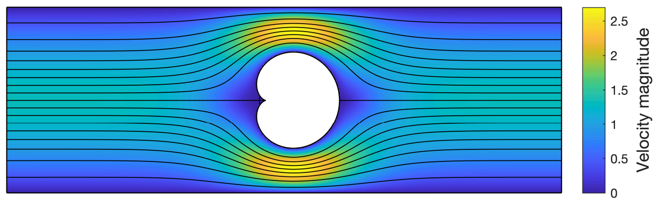

FIG. 9. *Stokes flow around a heart-shaped hole in a channel. The computation is carried out using a degree* 120 *polynomial and a degree* 80 *Laurent series.*

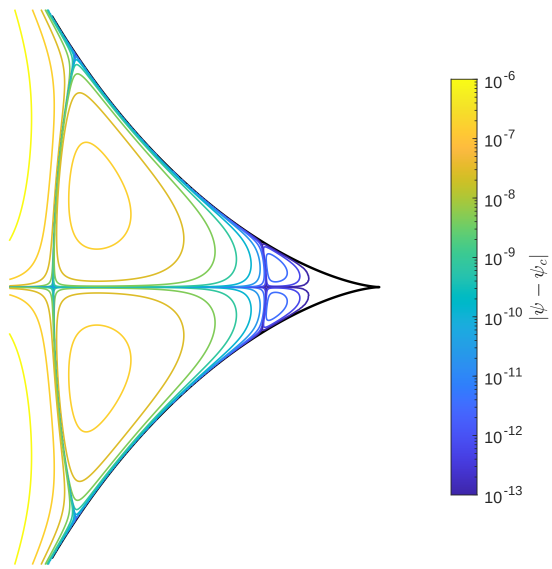

FIG. 10. *Moffatt eddies near the cusp of the heart-shaped hole in Figure* 9*. The colour scale represents the magnitude of the deviation of the stream function from* ψ_c*, the value along the centreline.*

We conclude this section with an example case that combines all the methods we have introduced: the lightning solver for sharp corners [9, 22], the AAA rational approximation for smooth boundaries [10, 36] and the series method for multiply connected domains [3,18,44]. This is an application of the complete LARS algorithm. Figure 11 shows the Stokes flow around a steady ellipse (or an elliptical hole) within a bifurcation. The bifurcation has two sharp corners, where poles are exponentially clustered, and a smooth corner, where poles are placed using the AAA algorithm. Boundary conditions of constant pressure and parallel flow are imposed on the channel inlet and outlets, but the upper branch has a higher outlet pressure than that of the lower branch. Zero velocity conditions are imposed on both the channel walls and the ellipse boundary.

For sharp corners, poles are exponentially clustered along the exterior bisectors, as shown in previous work [9]. For the smooth corner, AAA rational approximation places the poles outside the corner boundary using the boundary function $F = \bar{Z}$. It is interesting to note that the AAA algorithm clusters poles towards a branch point along

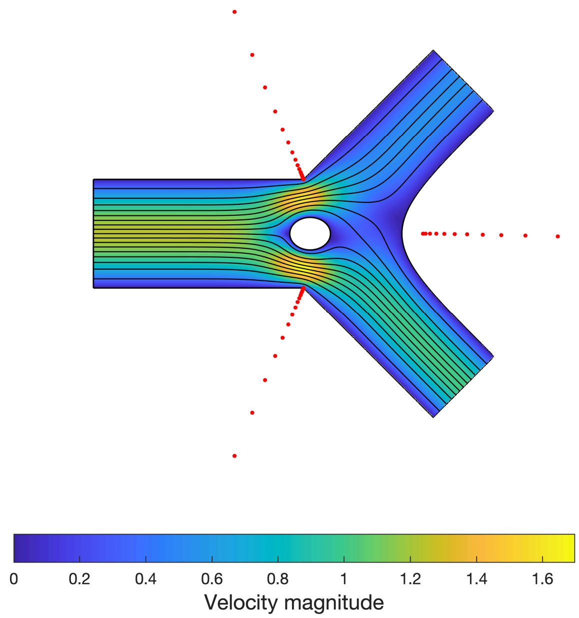

FIG. 11. *Stokes flow around a steady ellipse in a bifurcation with a smooth corner and two sharp corners. The computation is carried out to* 6*-digit accuracy using a degree* 96 *polynomial, a degree* 48 *Laurent series,* 48 *poles exponentially clustered near each sharp corner and AAA poles near the smooth corner.*

the exterior bisector of the smooth boundary automatically in 0.03 second, without any knowledge of the singularities near that geometry. A degree 48 Laurent series was added for the elliptical hole (or a 2D steady elliptical particle) with corresponding logarithmic terms. Next, we carried out VA orthogonalization [8] for a degree 96 polynomial, the Laurent series and three sets of poles near three corners. As the last step, we solved a linear least-squares problem using the backslash command in MATLAB to achieve a solution to 6-digit accuracy. The entire computation takes 1.5 seconds on a standard laptop. In engineering applications, e.g., for microparticle transport problems [2, 12] when Stokes flow needs to be simulated at multiple time steps, one can obtain a reasonably accurate solution (losing 1-digit accuracy) in a much shorter time by reducing the degrees of polynomial and series.

7. Discussion. In this paper, we have presented LARS, an algorithm that uses lightning and AAA rational approximation to compute 2D Stokes flows in bounded domains. The algorithm uses lightning approximation for sharp corners [9, 22], AAA rational approximation for curved boundaries [10, 36] and a series method for holes [3, 44]. Vandermonde with Arnoldi orthogonalization [8] is carried out for each part to construct a well-conditioned basis, and the Goursat functions for 2D Stokes flows [23] are approximated by solving a linear least-squares problem.

One advantage of LARS is its great speed and accuracy, thanks to the root-exponential convergence of the lightning algorithm [22]. We can compute a solution to a few digits of accuracy by just placing a few poles exponentially clustered near the singular corners or branch points near curved boundaries. The application of the lightning algorithm to Stokes flows in polygons has been presented in previous work [8]. Similar speed and accuracy have been shown in this work when computing Stokes flows in curved boundary domains using AAA poles, which are also clustered towards the branch point. The beauty of using lightning and AAA rational approximation to compute Stokes flows is that it allows a rational function to capture the singularities using poles located outside the domain, while preserving analyticity inside the domain. The advantage of rational approximation over polynomial approximation in computing Stokes flows was shown in the constricted channel example in section 4.

The LARS solver is also suitable for quasi-steady computations with moving boundaries, where the boundary position and velocity are updated at each time step based on the flow field. This is because it is fast enough to perform hundreds of simulations in a few minutes. Since this algorithm does not require the domain discretisation at each time step, it saves both computer memory and computation time. We believe the proposed solver has many more potential applications to time-dependent problems than the steady-state problems presented in section 6. For example, we are currently using it to investigate the transport of microparticles in channels with bifurcations [12] with applications to the transport of microthrombi in the human cerebral microvasculature [49].

It should be noted that the algorithm presented is not limited to computing Stokes flows, although this is the focus of our applications. It can be applied to other biharmonic problems, e.g. the vibration of plates in solid mechanics [30], with changes only in the boundary conditions.

One limitation of LARS is that it is only applicable to 2D geometries, because it is based on Goursat representations and rational functions. In fact, this is true for most applied complex variables techniques [1]. For 3D Stokes flow problems, one can use other numerical methods mentioned including finite element methods [29] or boundary integral methods [38].

In addition, LARS currently only works for flows in bounded domains. As discussed previously [9], rational approximations should be able to treat Stokes flow problems in unbounded domains. The computation of unbounded Stokes flows requires careful consideration of the boundary conditions at infinity. For periodic boundary problems, e.g. Stokes flow through a 2D channel with periodic boundary geometry, using trigonometric polynomials might lead to better results than using the Runge terms in (3.1). For example, see [4] for an extended AAA algorithm using a trigonometric barycentric formula to approximate periodic functions.

Lastly, there has been recent progress in lightning and AAA rational approximations that may lead to a better 2D Stokes flow solver. Baddoo and Trefethen [5] developed a log-lightning method that has been shown to have faster convergence than the original lightning method [22] when computing Green's function for a square. Following this, a log-lightning Stokes solver could be developed in future work. In addition, a new study considers AAA approximation on a continuum [13], rather than the vector of discrete points used in the original AAA approximation paper [36]. This can potentially lead to faster and more robust approximations.

To conclude, we have shown that it is now possible to compute 2D Stokes flows in very general domains using rational approximation. The "LARS" algorithm is simple and easy to implement for a variety of 2D Stokes flow problems. The computation usually takes less than a second to obtain a solution to at least 6-digit accuracy.

Appendix A. MATLAB codes. The MATLAB codes for computing the example cases in sections 4 and 5 and Figures 10 and 11 have been posted in a

GitHub repository at https://github.com/YidanXue/LARS.

Acknowledgments. We are grateful for very helpful discussions with Stephen Payne and Howard Stone. We also thank two anonymous referees for their comments and suggestions.

REFERENCES

[1] M. J. Ablowitz and A. S. Fokas, *Complex variables: introduction and applications*, Cambridge University Press, 2003.

[2] D. Audet and W. Olbricht, *The motion of model cells at capillary bifurcations*, Microvasc. Res., 33 (1987), pp. 377–396.

[3] S. Axler, *Harmonic functions from a complex analysis viewpoint*, Amer. Math. Monthly, 93 (1986), pp. 246–258.

[4] P. J. Baddoo, *The AAAtrig algorithm for rational approximation of periodic functions*, SIAM J. Sci. Comput., 43 (2021), pp. A3372–A3392.

[5] P. J. Baddoo and L. N. Trefethen, *Log-lightning computation of capacity and Green's function*, Maple Transactions, 1 (2021).

[6] B. Ballal and R. Rivlin, *Flow of a Newtonian fluid between eccentric rotating cylinders: inertial effects*, Arch. Rational Mech. Anal., 62 (1976), pp. 237–294.

[7] L. Banjai, *Revisiting the crowding phenomenon in Schwarz–Christoffel mapping*, SIAM J. Sci. Comput., 30 (2008), pp. 618–636.

[8] P. D. Brubeck, Y. Nakatsukasa, and L. N. Trefethen, *Vandermonde with Arnoldi*, SIAM Rev., 63 (2021), pp. 405–415.

[9] P. D. Brubeck and L. N. Trefethen, *Lightning Stokes solver*, SIAM J. Sci. Comput., 44 (2022), pp. A1205–A1226.

[10] S. Costa and L. N. Trefethen, *AAA-least squares rational approximation and solution of Laplace problems*, in European Congr. Math., Hujdurović et al., ed., EMS Press, 2023.

[11] P. Davis and H. Pollak, *On the analytic continuation of mapping functions*, Trans. Amer. Math. Soc., 87 (1958), pp. 198–225.

[12] V. Doyeux, T. Podgorski, S. Peponas, M. Ismail, and G. Coupier, *Spheres in the vicinity of a bifurcation: elucidating the Zweifach–Fung effect*, J. Fluid Mech., 674 (2011), pp. 359–388.

[13] T. Driscoll, Y. Nakatsukasa, and L. N. Trefethen, *AAA rational approximation on a continuum*, SIAM J. Sci. Comput., to appear (2023).

[14] T. A. Driscoll, N. Hale, and L. N. Trefethen, *Chebfun Guide*, 2014, www.chebfun.org.

[15] A. Dvinsky and A. S. Popel, *Motion of a rigid cylinder between parallel plates in Stokes flow: Part 1: Motion in a quiescent fluid and sedimentation*, Comput. Fluids, 15 (1987), pp. 391–404.

[16] A. Dvinsky and A. S. Popel, *Motion of a rigid cylinder between parallel plates in Stokes flow: Part 2: Poiseuille and Couette flow*, Comput. Fluids, 15 (1987), pp. 405–419.

[17] M. D. Finn and S. M. Cox, *Stokes flow in a mixer with changing geometry*, J. Eng. Math., 41 (2001), pp. 75–99.

[18] M. D. Finn, S. M. Cox, and H. M. Byrne, *Chaotic advection in a braided pipe mixer*, Phys. Fluids, 15 (2003), pp. L77–L80.

[19] A. Fokas and A. Kapaev, *On a transform method for the Laplace equation in a polygon*, IMA J. Appl. Math., 68 (2003), pp. 355–408.

[20] R. A. Frazer, *III. On the motion of circular cylinders in a viscous fluid*, Proc. R. Soc. Lond. A, 225 (1926), pp. 93–130.

[21] A. Gopal and L. N. Trefethen, *Representation of conformal maps by rational functions*, Numer. Math., 142 (2019), pp. 359–382.

[22] A. Gopal and L. N. Trefethen, *Solving Laplace problems with corner singularities via rational functions*, SIAM J. Numer. Anal., 57 (2019), pp. 2074–2094.

[23] E. Goursat, *Sur l'équation* $\Delta\Delta u = 0$, Bull. Soc. Math. France, 26 (1898), pp. 236–237.

[24] G. B. Jeffery, *The rotation of two circular cylinders in a viscous fluid*, Proc. R. Soc. Lond. A, 101 (1922), pp. 169–174.

[25] J.-T. Jeong, *Slow viscous flow in a partitioned channel*, Phys. Fluids, 13 (2001), pp. 1577–1582.

[26] D. H. Kelley and J. H. Thomas, *Cerebrospinal fluid flow*, Annu. Rev. Fluid Mech., 55 (2023).

[27] T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, and E. M. Viggen, *The lattice Boltzmann method*, Springer, 2017.

[28] E. Lauga and T. R. Powers, *The hydrodynamics of swimming microorganisms*, Rep. Prog.

Phys., 72 (2009), p. 096601.
[29] A. Logg, K.-A. Mardal, and G. Wells, *Automated solution of differential equations by the finite element method: The FEniCS book*, vol. 84, Springer, 2012.
[30] A. E. H. Love, *A treatise on the mathematical theory of elasticity*, Cambridge University Press, Cambridge, 4th ed., 1927.
[31] E. Luca and D. G. Crowdy, *A transform method for the biharmonic equation in multiply connected circular domains*, IMA J. Appl. Math., 83 (2018), pp. 942–976.
[32] E. Luca and S. G. Llewellyn Smith, *Stokes flow through a two–dimensional channel with a linear expansion*, Q. J. Mech. Appl. Math., 71 (2018), pp. 441–462.
[33] M. Manga, *Dynamics of drops in branched tubes*, J. Fluid Mech., 315 (1996), pp. 105–117.
[34] R. Menikoff and C. Zemach, *Methods for numerical conformal mapping*, J. Comput. Phys., 36 (1980), pp. 366–410.
[35] H. K. Moffatt, *Viscous and resistive eddies near a sharp corner*, J. Fluid Mech., 18 (1964), pp. 1–18.
[36] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, *The AAA algorithm for rational approximation*, SIAM J. Sci. Comput., 40 (2018), pp. A1494–A1522.
[37] H. Ockendon and J. R. Ockendon, *Viscous flow*, Cambridge University Press, 1995.
[38] C. Pozrikidis, *Boundary integral and singularity methods for linearized viscous flow*, Cambridge University Press, 1992.
[39] T. Price, T. Mullin, and J. Kobine, *Numerical and experimental characterization of a family of two-roll-mill flows*, Proc. R. Soc. A, 459 (2003), pp. 117–135.
[40] K. Ranger, *A problem on the slow motion of a viscous fluid between two fixed cylinders*, Q. J. Mech. Appl. Math., 14 (1961), pp. 411–421.
[41] K. Ranger, *Eddies in two dimensional Stokes flow*, Int. J. Eng. Sci., 18 (1980), pp. 181–190.
[42] T. W. Secomb, *Blood flow in the microcirculation*, Annu. Rev. Fluid Mech., 49 (2017), pp. 443–461.
[43] B. Tavakol, G. Froehlicher, D. P. Holmes, and H. A. Stone, *Extended lubrication theory: improved estimates of flow in channels with variable geometry*, Proc. R. Soc. A, 473 (2017), p. 20170234.
[44] L. N. Trefethen, *Series solution of Laplace problems*, ANZIAM J., 60 (2018), pp. 1–26.
[45] L. N. Trefethen, *Numerical conformal mapping with rational functions*, Comput. Methods Funct. Theory, 20 (2020), pp. 369–387.
[46] J. Walsh, *The approximation of harmonic functions by harmonic polynomials and by harmonic rational functions*, Bull. Amer. Math. Soc., 35 (1929), pp. 499–544.
[47] N. Wiener and E. Hopf, *Über eine Klasse singulärer Integralgleichungen*, Sem.–Ber. Preuss. Akad. Wiss., 31 (1931), pp. 696–706.
[48] J. Williams, B. Turney, D. Moulton, and S. Waters, *Effects of geometry on resistance in elliptical pipe flows*, J. Fluid Mech., 891 (2020), p. A4.
[49] Y. Xue, T. Georgakopoulou, A.-E. van der Wijk, T. I. Józsa, E. van Bavel, and S. J. Payne, *Quantification of hypoxic regions distant from occlusions in cerebral penetrating arteriole trees*, PLoS Comput. Biol., 18 (2022), p. e1010166.
[50] L. Y. Yeo, H.-C. Chang, P. P. Chan, and J. R. Friend, *Microfluidic devices for bioapplications*, Small, 7 (2011), pp. 12–48.