Pergamon

Marine Models 2 (2000) 35–56

MARINE MODELS

ONLINE

A Matlab environment for analysis of fluid flow and transport around a translating sphere☆

Uffe Høgsbro Thygesen *, Thomas Kiørboe

Danish Institute for Fisheries Research, Charlottenlund Slot, DK-2920 Charlottenlund, Denmark

Received 9 October 2000; received in revised form 1 September 2001; accepted 21 September 2001

Abstract

We present a software environment, implemented in `Matlab`, which addresses a sphere moving steadily in a fluid. The sphere leaks solute which is transported through the fluid. The environment allows the fluid flow to be approximated with Stokes' flow, or the Navier–Stokes equations can be solved numerically. Subsequently, the advection–diffusion equation for the concentration of the solute is solved numerically. Our purpose for developing the environment was to investigate solute concentrations around sinking marine snow, but the environment has more general applicability. The allowable parameter range depends on computational ressources; on our PC we investigated Reynolds numbers up to 20 and Peclet numbers up to 20,000. The environment features a graphical user interface which makes it useful to people who have never used Matlab, but the experienced Matlab user can also operate from the command prompt.

Keywords: Steady flow around a sphere; Navier–Stokes equations; Stokes flow; Advection–diffusion equation; Heat equation; Marine snow

1. Introduction

We have developed a Matlab[1] environment, named `Snow`, for computation and analysis of fluid flow and concentration fields around a sphere. The sphere moves

* Corresponding author.
E-mail addresses: uht@dfu.min.dk (U.H. Thygesen); tk@dfu.min.dk (T. Kiørboe).

☆ Revised version of a manuscript submitted to Marine Models Online.

[1] Matlab is a registered trademark, see The Mathworks, Inc. (1999).

1369-9350/00/$ - see front matter
PII: S1369-9350(01)00003-7

with constant velocity through a fluid and is a sink or a source of some substance, which is transported through the surrounding fluid by advection and diffusion.

The fluid flow is governed by the Navier–Stokes equations. The steady-state concentration field is governed by a linear advection–diffusion equation. See e.g. Acheson (1990) for the underlying physical theory.

Our original interest in this problem was to study the situation of marine snow in the ocean. Sinking particles leak organic solutes which form a chemical trail in the water. This trail may be sensed by zooplankton, which attempt to follow the trail and colonise the particles. See Kiørboe and Thygesen (2001) for a description of this situation.

However, the model has wider applicability. Firstly, negative production and transport corresponds to particles which consume some substance (e.g. oxygen) in the surrounding fluid. Secondly, the same advection–diffusion equation models both molecular diffusion and heat transfer, so the model also describes temperature fields in, for example, a cold fluid flowing past a hot sphere. Finally, if we add a uniform flow field to the Stokes flow we obtain the flow around a *spherical pump*; the model can then describe the catch rate of predators which generate a feeding current as in Kiørboe and Visser (1999).

Regarding mathematical analysis of the model, there exist approximate analytical solutions to the problem in terms of asymptotic expansions. See for instance Acrivos and Taylor (1962) and Acrivos and Goddard (1965); these expansions are used in Jackson (1989) in a context similar to ours. These expansions, however, assume Stokes flow and either very low or very high Peclet numbers. It is an open question as to how sensitive the conclusions are to these assumptions. In addition to the expansion techniques, the engineering literature is abundant with empirical relationships which, however, typically concern overall scalar descriptors such as the Sherwood or the Nusselt number.

This motivated us to implement the following functionality in the environment: to analyse the fluid flow through Stokes' approximation, or by direct numerical solution, and, subsequently, to analyse the transport through direct numerical solution of the advection–diffusion equation. This functionality is available from `Matlab` through a graphical user interface, which does not require familiarity with Matlab or the underlying mathematics and numerics. For the experienced Matlab user the functionality is also available as a set of classes which can be activated from the Matlab command line, or from another Matlab application.

From a numerical point of view, the advection–diffusion equation is linear and, in principle, straightforward to solve. For typical parameters, however, the transport is dominated by advection which leads to long trails, sharp gradients, and numerically sensitive calculations. Concerning the fluid flow, it is in general a notoriously difficult task to solve the Navier–Stokes equations. In our particular situation, where the geometry is simple and the flow is slow with Reynolds numbers below 20, the process is feasible but still time consuming and prone to convergence problems.

The paper is organised as follows. In Section 2 we give an overview of the graphical user interface and show example screen captures. Section 3 describes the physics of fluid flow and solute transport and discusses appropriate mathematical models.

Section 4 describes the numerical analysis of the model, including discretisation and iteration schemes. Section 5 gives some details on the software architecture, for the benefit of the advanced user who wants to by-pass the graphical user interface, or make changes to the environment. Section 6 is a brief summary of the application to marine snow which motivated the development of the software. Information about installation, system requirements, and parameter recommendations are found in Appendix A.

The numerical analysis performed by the environment has earlier been described in condensed form in an appendix in Kiørboe, Ploug, and Thygesen (2001).

2. An overview of the environment

This section assumes that the environment has been installed and started as described in Appendix A.

The graphical environment contains four windows: one for specification of the grid to be used for the computations, Fig. 1. Another for specification, computation, and visualisation of the fluid flow, Fig. 2. A third, Fig. 3, for specification of the transport, and for computation and visualisation of the concentration field. And finally a fourth, Fig. 4, for post-processing the solution; e.g. computing the width of the plume, where the concentration exceeds some threshold level.

In a typical session, the user first specifies the computational grid in the window in Fig. 1. How far away from the sphere should the fields be computed, and how fine a mesh is required? In addition, it is possible to make the grid finer downstream than upstream, by varying a parameter γ between 0 and 1; see Section 4 below for details.

Next, the fluid flow field is specified in the window in Fig. 2. Is the problem of

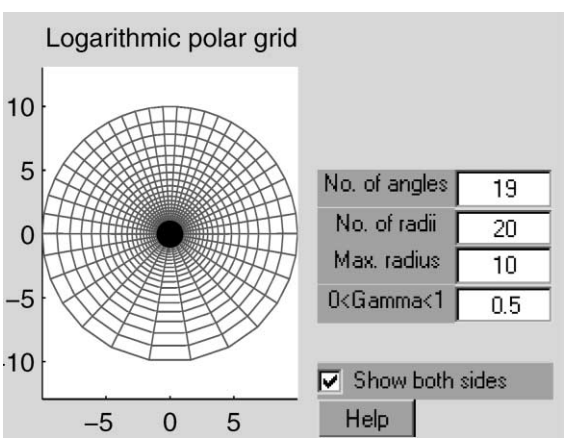

Fig. 1. The window for specification of the computational grid. This particular grid is too small and coarse for most applications, but shows the structure. Notice that the grid is finer downstream (up) than upstream since $\gamma > 0$.

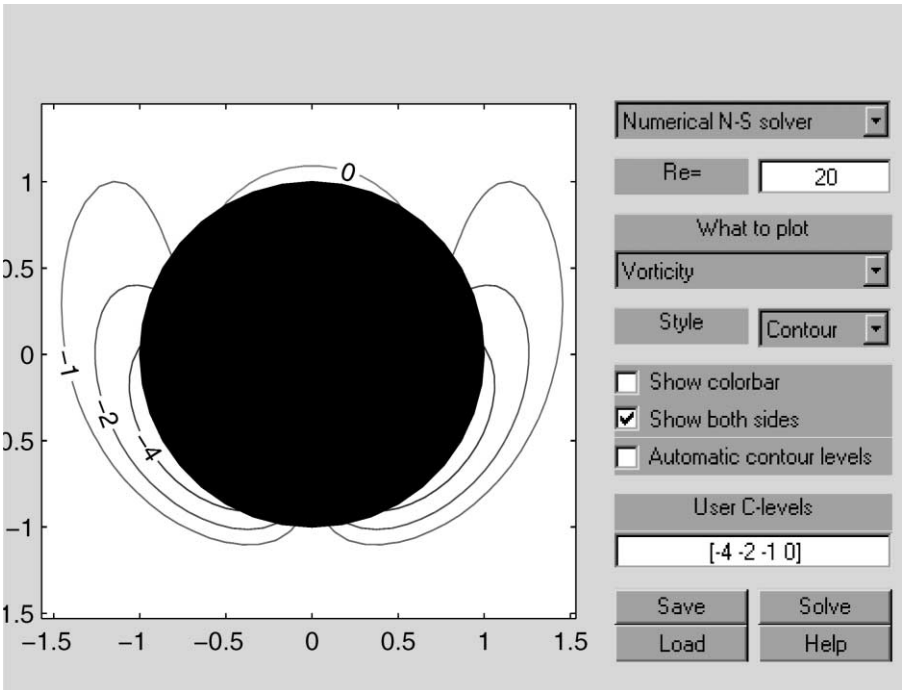

Fig. 2. The window for specification and analysis of the fluid flow field. This plot shows lines of constant vorticity for a Reynolds number of 20. The cap at the top of the sphere indicates a change in sign of the vorticity; hence separation is present. The plot supports the notion that vorticity is generated at the surface and transported through the fluid.

interest that of a translating sphere, or a spherical pump? In the former case, is the Stokes approximation adequate, or are full numerical solutions needed, in which case, for what Reynolds number? The resulting fluid flow can be visualised in a number of ways. For instance, one may plot the stream function, the individual flow components, deformation, or vorticity. These can be visualised as contour plots, colour-coded "surf" plots, or the fields can be plotted along the polar axis or along a line in the equatorial plane. A zoom capability is also available, using the mouse. Numerically computed flow fields can then be saved to disk for later retrieval.

With the fluid flow in place, we turn attention to the transport problem. First, if the Peclet number is high, a different computational grid may be required, i.e. the concentration field is solved on a finer mesh than was used for computing fluid flow fields. The transport problem is now specifed in the window in Fig. 3. In addition to the Peclet number, we must also specify the parameter λ, which determines the ratio of molecular diffusivity to total diffusivity (see Section 3.4 below). λ lies between 0 and 1. The boundary conditions at the sphere are chosen as either Dirichlet or Neumann conditions (see Section 3.3 below for a discussion about these boundary conditions and their biological significance). As a numerical method, the third/fourth order upwind scheme is a reasonable default, but those with an interest in the

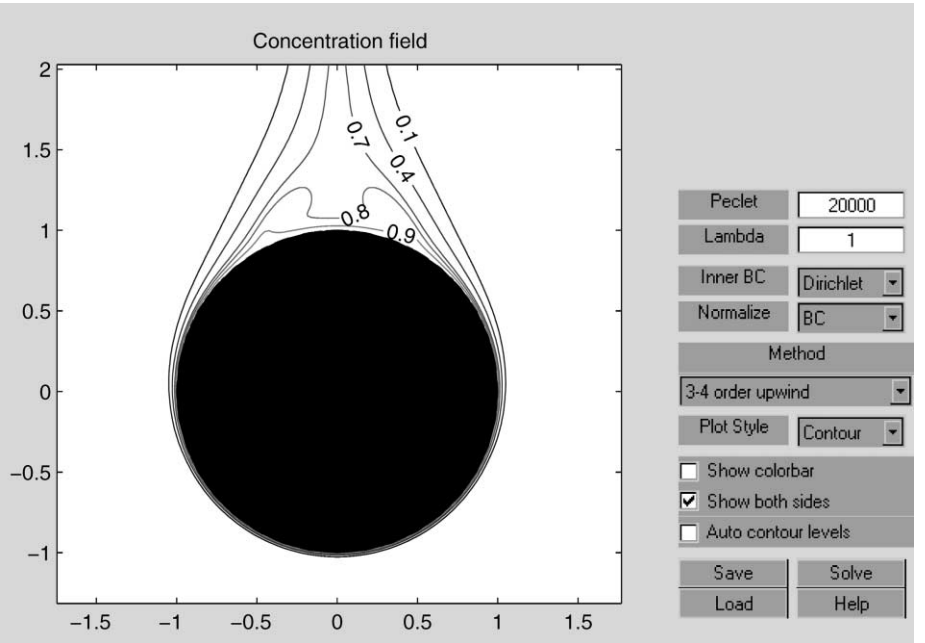

Fig. 3. The window for computation and analysis of the concentration field. The graph shows lines of constant concentration in a concentration field with Peclet number 20,000; the underlying flow field is the one from Fig. 2 with Reynolds number 20. Notice how the recirculation behind the sphere in this case leads to an area with high and fairly constant concentration.

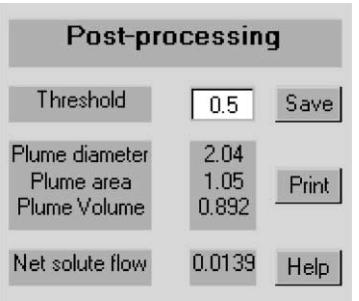

Fig. 4. The window for post-processing the solutions.

numerics can compare with two alternatives. By default the solution satisfies a normalised boundary condition (i.e. the dimensionless concentration or its gradient has magnitude 1 at the surface of the sphere), but the solution may be multiplied by a constant so that the dimensionless flow is normalised to 1. Once the concentration field has been solved, it can be visualised with options similar to those of the fluid flow field, or saved to disk for later retrieval.

The last window, in Fig. 4, contains various information about the solution. Our

original interest was concerned with the extent of the plume, which is defined as the region in which the concentration exceeds a given threshold. This window shows the width, cross-sectional area, and volume of the plume. In addition, the total solute flow away from the sphere is given.

3. The mathematical model

The mathematical model consists of Navier–Stokes equations for the flow around the sphere, and an advection–diffusion equation for the transport.

3.1. Co-ordinate system and dimensionless variables

The model is formulated in dimensionless variables in a spherical co-ordinate system following the sphere.

r denotes the distance from the centre of the sphere, ϕ denotes longitude, and θ denotes latitude with $\theta = 0$ pointing downstream (i.e., up for the case of a sinking particle) and $\theta = \pi$ pointing upstream.

The situation is rotationally symmetric. This means that the longitude ϕ does not appear in the equations.

All computations are done in dimensionless variables. To use the environment in an application, one must determine the dimensionless variables which describe the application, and then use these as inputs to the program. After the computation, one must convert the dimensionless results back to the dimensional setting in the application.

The characteristic length is taken as the radius a of the sphere, and the free fluid flow speed U is used as the characteristic speed. The fluid flow and the concentration fields are then described by the Reynolds number

$$Re = \frac{aU}{\nu}$$

and the Peclet number

$$Pe = \frac{Ua}{\mathrm{D}}$$

where ν is the kinematic viscosity (for seawater, approximately 10^{-6} m^2 s^{-1}) and D is the diffusivity (approximately 10^{-9} m^2 s^{-1} for small biologically relevant molecules). These two parameters, the Reynolds and the Peclet number, completely specify the physics up to scale, and are the only physical input needed to the program. Their ratio is termed the Schmidt number (in the context of heat transfer, the Prandtl number Pr) and is a material constant:

$$Sc = \frac{Pe}{Re} = \frac{\nu}{\mathrm{D}}$$

with an approximate value of 1000 for small molecules in seawater.

Interpretation of the results from the program may involve conversion back to dimensional quantities, using Table 1.

While most of these conversions are straightforward, those involving solute flows and concentrations require some explanation. The concentration of the solute in the fluid is made dimensionless with an affine transformation

$$C = cC' + C_\infty \tag{1}$$

where C is dimensional concentration, C' is dimensionless concentration, C_∞ is the ambient concentration found very far from the sphere, and c is a scaling factor. This equation implies a relationship between dimensional and dimensionless solute flows as follows:

$$F = cUa^2F'. \tag{2}$$

The scaling factor c can be fixed in a number of ways, depending on circumstances, as discussed below.

3.1.1. Normalised solute flow from the sphere

If one knows the actual solute flow F from the sphere (e.g. because one knows the internal production), then one may choose to scale the dimensionless concentration so that the dimensionless solute flow F' away from the sphere is 1. In that case the scaling factor c is found from Eq. (2):

$$c = \frac{F}{a^2U}$$

Table 1
Conversion between dimensional and dimensionless quantities

Quantity	With dim.	Dim.-less	Relation
Radial co-ordinate	r	r'	$r = r'a$
Dist. from symmetry axis	x	x'	$x = x'a$
Dist. from equatorial plane	z	z'	$z = z'a$
Latitude	θ	θ'	$\theta = \theta'$
Stream function	ψ	ψ'	$\psi(r,\theta) = a^2U\psi'(r',\theta')$
Fluid flow velocity	u	u'	$u(r,\theta) = u'(r',\theta')U$
Fluid vorticity	Ω	Ω'	$\Omega(r,\theta) = \Omega'(r',\theta')\frac{U}{a}$
Max. fluid deformation	Δ	Δ'	$\Delta(r,\theta) = \Delta'(r',\theta')\frac{U}{a}$
Concentration	C	C'	$C = c{\cdot}C' + C_\infty$
Conc. derivative	$\frac{\partial C}{\partial r}$	$\frac{\partial C'}{\partial r'}$	$\frac{\partial C}{\partial r} = \frac{\partial C'}{\partial r'}\frac{c}{a}$
Solute flow	F	F'	$F = ca^2UF'$

after which dimensional concentrations can be found from Eq. (1). To use this normalisation, choose "Flow" in the "Normalise" menu in window 3.

3.1.2. Normalised concentration

If one knows the dimensional concentration at the surface of the sphere, and this concentration is constant $= C_a$ over the surface, then it is natural to use a Dirichlet boundary condition (see Section 3.3 below) and normalise the concentration so that dimensionless concentration at the surface is 1. In this case the scaling factor c becomes

$$c = C_a - C_\infty$$

and dimensional solute flows and concentrations can now be found from Eqs. (1) and (2). To use this normalisation, choose "BC" in the "Normalise" menu, and "Dirichlet" in the "Inner BC" menu in window 3.

3.1.3. Normalised radial derivative of concentration

If the radial derivative of the concentration is constant over the surface of the sphere and known (which then also implies that the solute flow is known), then it is natural to use a Neumann boundary condition (see Section 3.3 below) to normalise the concentration so that the dimensionless radial derivative of concentration at the surface is -1. In this case the scaling factor c becomes

$$c = -a\left.\frac{\partial C}{\partial r}\right|_{r=a}$$

and dimensional solute flows and concentrations can now be found from Eqs. (1) and (2). To use this normalisation, choose "BC" in the "Normalise" menu, and "Neumann" in the "Inner BC" menu in window 3.

In the remainder of the paper, all variables are dimensionless, except when explicitly stated otherwise. For ease of notation we drop the prime, as is customary in the field.

3.2. Model of the fluid flow

The fluid flow around the sphere is assumed to be incompressible, axially symmetric and in steady state, and can therefore be described by the time-independent stream function, denoted $\psi(r,\theta)$, see e.g. Acheson (1990, p. 173). The physical significance of the stream function ψ is that $2\pi\psi(r,\theta)$ is the flow through the circle in space given by r and θ.

The fluid flow field is found by differentiating the stream function:

$$u_\phi = 0,\ u_r = \frac{1}{r^2\sin\theta}\frac{\partial\psi}{\partial\theta},\ u_\theta = -\frac{1}{r\sin\theta}\frac{\partial\psi}{\partial r}$$

Here u_ϕ, u_θ and u_r are the components of the fluid flow in longitudinal, latitudinal, and radial directions, respectively.

Navier–Stokes equations for water can be formulated as a fourth order nonlinear partial differential equation in the stream function, see van Dyke (1964, p. 124):

$$\left(L^2 - \frac{Re}{r^2 \sin\theta}\left[\frac{\partial\psi}{\partial\theta}\frac{\partial}{\partial r} - \frac{\partial\psi}{\partial r}\frac{\partial}{\partial\theta} + \left(2\cot\theta\frac{\partial\psi}{\partial r} - \frac{2}{r}\frac{\partial\psi}{\partial\theta}\right)\right]\right)\zeta = 0 \tag{3}$$

where

$$\zeta = L^2\psi \tag{4}$$

and the operator[2] L^2 is shorthand for

$$L^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r^2}\frac{\partial^2}{\partial\theta^2} - \frac{\cot\theta}{r^2}\frac{\partial}{\partial\theta}.$$

The boundary conditions of this equation state that the velocity is zero at the surface of the sphere, and approaches free flow far from the sphere:

$$\psi = 0 \quad \text{at the surface of the sphere,} \tag{5}$$

$$\frac{\partial\psi}{\partial r} = 0 \quad \text{at the surface of the sphere,} \tag{6}$$

$$L^2\psi \to 0 \quad \text{as } r \to \infty, \text{ and} \tag{7}$$

$$\psi \to \frac{1}{2}r^2\sin^2\theta \quad \text{as } r \to \infty. \tag{8}$$

The vorticity $\nabla \times u$ is another useful descriptor for the fluid flow field, which is equal to two times the angular velocity (rotation) of a fluid element. The environment also allows plotting of the maximum deformation rate, which is defined as the spectral radius (largest eigenvalue) of the rate of strain matrix, cf. Kiørboe and Visser (1999, Eq. (A8)). Our interest in the maximum deformation rate is that some zooplankters appear to react to this particular fluid flow component.

3.2.1. Analytical approximate solutions

The best known analytical approximation is due to Stokes himself, and omits the nonlinear terms in the Navier–Stokes equation, which corresponds to setting the Reynolds number $Re = 0$. This leads to the following solution:

$$\psi_{\text{Stokes}}(r,\theta) = \frac{1}{4}\left(2r^2 + \frac{1}{r} - 3r\right)\sin^2\theta.$$

In addition to the Stokes stream function, the environment also implements the free flow stream function

[2] This operator is typically given the symbol D^2 in the literature, e.g. van Dyke (1964). We use L^2 to avoid confusion with the diffusivity D.

$$\psi_{\text{Free}}(r,\theta) = \frac{1}{2}r^2\sin^2\theta.$$

One should of course be aware that in a free fluid flow, the water penetrates the sphere, which make the subsequent analysis of transport dubious. The free flow is mainly implemented to enable comparison.

Finally, the environment implements the *spherical pump* stream function of

$$\psi_{\text{Pump}}(r,\theta) = \psi_{\text{Stokes}}(r,\theta) - \psi_{\text{Free}}(r,\theta) = \frac{1}{4}\left(\frac{1}{r} - 3r\right)\sin^2\theta,$$

which is a Stokes-type approximation (i.e., the nonlinear inertial terms are neglected) but for the fluid flow problem where the sphere acts as a pump rather than translates through the fluid. To be precise, the spherical pump stream function solves the Navier–Stokes equations with $Re = 0$ and with the boundary conditions (5), (8) replaced by

$$\psi_{\text{Pump}}(r,\theta) = \psi_{\text{Free}}(r,\theta) \text{ at the surface of the sphere,}$$

$$\psi_{\text{Pump}}(r,\theta) = 0 \text{ as } r \longrightarrow \infty.$$

Those who prefer to work from the command line, the `streamfct` class makes it possible to define other stream functions which are given by closed-form expressions. Analytical approximations such as Oseen's and Proudman–Pearson's approximations are predefined in the environment. These approximations are discussed in van Dyke (1964). It was our experience, however, that they were less than useful for the subsequent analysis of transport, which is why they are not implemented in the graphical user interface.

3.3. The transport

At a point in the fluid, the flux of the solute is (in original, dimensional variables)

$$u \cdot C - \mathrm{D}\nabla C. \tag{9}$$

Here D is diffusivity and C is the concentration of the solute while u is the fluid flow vector. ∇ is the gradient operator. The first term in this equation is advection; the solute flux is in the direction to the fluid flow field. The second term is diffusion; the solute flux is in the opposite direction of the concentration gradient.

Conservation of the solute and steady state implies that this flux is divergence free, i.e. the concentration satisfies the *advection–diffusion equation* (in dimensionless variables)

$$Pe\ u \cdot \nabla C - \nabla^2 C = 0. \tag{10}$$

Here we have used constant diffusivity D, i.e. $\nabla \mathrm{D} = 0$, and the fluid flow is divergence free, i.e. $\nabla \cdot u = 0$. In addition we have used Table 1 to convert to dimensionless variables.

Far from the sphere, the dimensionless concentration approaches 0, i.e.

$$C|_{r=\infty} = 0. \tag{11}$$

It is less obvious from physics which boundary condition to use at the surface of the sphere. We investigate two choices of boundary conditions. First, the Dirichlet condition, where the concentration is fixed and constant over the surface of the sphere:

$$C|_{r=1} = c_0. \tag{12}$$

Secondly, the Neumann condition, where the radial derivative (hence the flux) is fixed and constant over the surface of the sphere.

$$\left.\frac{\partial C}{\partial r}\right|_{r=1} = c_0'. \tag{13}$$

The constants c_0 and c_0' are either 1 or fixed such that the resulting flow is one, cf. Section 3.1.

Roughly, the Dirichlet condition corresponds to a situation where the solute flows freely inside the sphere, hence providing a uniform distribution of the solute. At the other extreme, the Neumann condition corresponds to the situation where the solute flow inside the sphere is very restricted, and the solute therefore travels to the closest point on the surface and escapes from there. In this case, we also assume that the internal production is uniform and fixed. It is beyond our scope to model in detail the production and flow of the solute in the interior of the sphere, which is why we investigate these two boundary conditions only.

3.4. Including turbulent diffusion

The diffusion in the previous subsection is pure molecular diffusion, i.e. the macroscopic description of the random motion of molecules. Diffusion, however, can also be a macroscopic (time-averaging) description of the apparently random velocity fluctuations in turbulent fluid flow.

In this case, the diffusivity D in Eq. (9) is a *turbulent diffusivity* D^* which varies in space:

$$\mathrm{D}^*(r,\theta) = \alpha\epsilon^{1/3}r^{4/3}.$$

Here, α is a universal constant sometimes referred to as Richardson's constant, and although it is not very precisely known, it has an approximate value of 1.37. ϵ is the dissipation rate, which in oceanographic applications lies between 10^{-8} $\mathrm{m^2s^{-1}}$ and 10^{-4} $\mathrm{m^2s^{-1}}$.

This statistical approximation of turbulent dispersion of nearby particles goes back to Richardson (1926) and can qualitatively be explained as follows. The turbulence adds a random time-varying velocity field $\tilde{u}(r,\theta,\phi,t)$ to the mean velocity field. Two passive tracers at seperate positions in the fluid will therefore experience a random velocity difference because this random component $\tilde{u}$ varies in space. The closer the two tracers are, the smaller this velocity difference will be, because of the spatial

autocorrelation of the random velocity $\tilde{u}$. Therefore the equivalent diffusivity D^* grows with r. See Shraiman and Siggia (2000) for a recent discussion and further references.

Our software environment allows molecular and turbulent diffusion to be simultaneously present in the model. In dimensionless variables this means that the flux in Eq. (9) is replaced by

$$Pe\, u \cdot C - (\lambda + (1-\lambda) r^{4/3}) \nabla C.$$

The parameter λ is specified by the user in the interval between 0 and 1 and describes the ratio of molecular to total (molecular plus turbulent) diffusivity at the surface of the sphere:

$$\lambda = \frac{D}{D + D^*(a,\theta)} = \frac{D}{D + \alpha \epsilon^{1/3} a^{4/3}}.$$

Thus $\lambda = 1$ corresponds to pure molecular diffusion, while $\lambda = 0$ corresponds to pure turbulent diffusion.

The approximation of turbulence with spatially varying diffusivity is only valid for distances r which are smaller than the integral scale of the embedding turbulent flow. For the applications in marine biology — which was our original interest — this does not cause concern, but this should be kept in mind if applying the environment to other fields.

Note that concentration fields computed in this way with $\lambda < 1$ are *time-averaged* fields. At any point in time the actual concentration field may be very much different, and a fixed point in space may experience large concentration fluctuations due to the turbulence.

4. Numerical analysis

The advection–diffusion Eq. (10) and, unless one reconciles with an analytical approximation, cf. Section 3.2.1, the Navier–Stokes Eq. (3) are discretised using a finite difference scheme on a rectangular grid in polar coordinates (r,θ).

Using circular symmetry, we are interested in the solution in the region

$$\theta \in [0,\pi],\ r \in [1,R_{\max}],\ \phi = 0.$$

Here we should in principle take $R_{\max} = \infty$ but we will reconcile with a large $R_{\max}$, specified by the user. See the comment in Appendix C regarding this approximation.

The computational grid is rectangular in the spherical co-ordinates r, θ, and contains M radii and N angles. M and N are specified by the user.

The grids are not uniform. In the radial direction, the spacing is logarithmic, i.e.

$$r_i = h^{i-1}, \quad i = 1,\ldots,M$$

which enhances resolution near the surface of the sphere, where gradients are steep. In the latitudinal direction, the spacing is given by

$$\theta_j + \gamma\sin(\theta_j) = kj - \frac{k}{2}, \quad j = 1,\ldots,N.$$

Here the step sizes h and k are determined by $h^{M-1} = R_{\max}$ and $kN = \pi$. The parameter $\gamma \in [0,1)$ is defined by the user and makes the grid denser downstream than upstream, which is useful for solving the transport Eq. (10) at high Peclet numbers. We have found $\gamma = 0.7$ to be an appropriate value for Peclet numbers up to 20,000; this grid has a density downstream which is $(1 + 0.7)/(1-0.7) \approx 5.7$ higher than upstream. When solving the Navier–Stokes equations, we prefer using $\gamma = 0$, i.e. the angle θ is equidistantly spaced.

The partial differential equations are discretised on these computational grids using finite difference schemes. The Navier–Stokes equations for the fluid flow use a fourth-order central difference scheme. For the advection–diffusion equation for the concentration the user may choose between second-order and fourth-order central schemes, and a third/fourth order upwind scheme. The second-order scheme is the fastest, but has higher numerical diffusion than the fourth order scheme. For high Peclet numbers, the use of upwind schemes is necessary to avoid unphysical unboundedness (ripples) in the solution. In summary, we recommend the third/fourth order upwind scheme for the advection–diffusion equation.

In the event that different grids for the fluid flow and the concentration are used, the flow field is interpolated onto the diffusion grid before solving the advection–diffusion equation. This is done using cubic splines.

4.1. Solution of the algebraic equation systems

The discretisation transforms the partial differential equations into algebraic equations.

The advection–diffusion equation is linear and is solved in one step using a solver for sparse matrix systems which is built into Matlab.

The Navier–Stokes equations are nonlinear and are solved iteratively by constructing a sequence ψ_i and ζ_i of approximative solutions to Eqs. (3) and (4). The iteration law is that ψ_i and ζ_i satisfy the PDE (4) with the boundary conditions (5), (8), whereas ζ_i and ψ_{i+1} satisfy (3) with boundary condition (7) at the exterior boundary and the condition $\zeta_i(1,\theta) = \xi_i(\theta)$ at the inner boundary $r = 1$. Thus, the Navier–Stokes equation is solved by iteratively solving Dirichlet problems. The inner boundary condition $\xi_i(\theta)$ is not specified by the physics but must correspond to (6), and a technique originating from stochastic adaptive control is employed to guarantee this by refining guesses on $\xi_i(\theta)$. See the documentation in the source code for details.

5. Program details

This section is included for those who want to modify the environment, or access some of its components without using the graphical user interface. It can be skipped by the user who only wants to use the graphical user interface.

The environment was developed using Matlab's features for object oriented programming. It is our experience that this development style makes it relatively easy to make changes to the model, which is important in research where it is most often not clear what the program should do until after one has seen preliminary results, so that development is done iteratively. Other important benefits of object orientation in Matlab are that it makes it easy to test and validate the program, debugging each class interactively from the command line, which makes it relatively easy for other developers to use, expand, or modify the components in the software. A disadvantage of object orientation in Matlab is that it may involve some overhead of memory and CPU-time, because of the functional structure of Matlab, and because of the absence of pointers and references.

5.1. Architecture

The environment consists of a set of classes which represents fluid flow fields, computational grids, etc. Table 2 lists these classes. It is beyond the scope of this paper to go through in detail all the classes available from the command line; more information about a class and its available methods can be obtained from the command line by e.g.

Table 2
The classes in the application

Class name	Brief description
Visual classes	
`GUI`	The over-all graphical user interface (GUI)
`GUIgrid`	The GUI for the computational grid
`GUIflow`	The GUI for the stream function and fluid flow field
`GUIcon`	The GUI for the transport and concentration field
`GUIpost`	The GUI for post-processing of the solution
Mathematical classes	
`compgrid`	Computational grid
`flowfield`	Numerical values of fluid velocities
`streamfct`	A stream function given by an analytical, closed-form expression
`NumStreamFct`	Numerically solved stream function
`NumField`	Numerical values of a scalar field
`SparseTensor`	A sparse four-dimensional tensor, typically representing a discretised partial differential operator
`TensorMatrix`	A tensor-matrix equation; a discretised version of a partial differential equation
Auxiliary classes	
`rectgrid`	Rectangular, uniform grid — auxiliary class for compgrid, rarely interfaced directly
`scalarfield`	"Abstract" class representing a scalar field, not used directly

```
>> help compgrid
```

or

```
>> methods compgrid
```

The classes can be divided into three groups: those that involve the graphical user interface; those that represent mathematical objects; and finally, auxillary classes, which the user does not interface. The classes in the first group all begin with `GUI`, e.g. `GUIgrid`, which is the graphical user interface for specification of the computational grid.

5.2. Working from the command line

A user who is experienced with Matlab may in some situations prefer to work from the command line rather than using the graphical user interface. Reasons to prefer the command line are

- The ability to make macros, or scripts, which perform computations on a list of scenarios.
- The ability to combine the computations made within the environment with other operations, defined by the user.

When working from the command line, the Matlab search path must include the `Snow/bin` directory.

For instance, the following code extract assigns the default computational grid to variable `cg`, the analytical Stokes solution to `psi`, lets `ff` be the fluid flow field corresponding to this stream function evaluated on the computational grid `cg`, and finally plots the flow field in the default style.

```
>> cg = compgrid
>> psi = streamfct("stokes")
>> ff = flowfield(psi,cg)
>> plot(ff)
```

It is straightforward to combine the graphical user interface with work from the command line. For instance, if one has specified a fluid flow field in the graphical user interface and wishes to investigate this field from the command line, or use it in some other computation, then it may be retrieved with the command

```
>> ff = flowfield(GUIflow)
```

For more information about how to extract information from GUI componenents, look at the methods for those components (e.g. `>> methods("GUIcon")`).

5.3. Program validation

The numerical Navier–Stokes solver was validated by comparing the results to those published in Dennis and Walker (1971) and Chang and Maxey (1994) and Chang & Maxey (1994). We have found that contour plots of the vorticity are the most useful for comparison.

The solver for the concentration fields was first validated by solving for a Peclet number of zero (pure diffusion), where an analytical solution is available for comparison. Subsequently, we validated the concentration fields obtained from Stokes flow and small or large Peclet numbers against the approximate analytical solutions in Acrivos and Taylor (1962) and Acrivos and Goddard (1965).

At higher Reynolds and Peclet numbers the solution was evaluated by comparing both concentration fields and Sherwood numbers with those obtained from previous numerical solutions, summarised in Clift, Grace, and Weber (1978).

Based on this validation we believe the results to be accurate, but we cannot guarantee its suitability for use outside the tested parameter ranges.

6. An application

The model was developed with marine snow in mind. Marine snow is milimetre to centimetre sized aggregates formed from smaller primary particles by coagulation and other processes. Marine snow aggregates sink at velocities of up to 100 m or more per day, and are thus characterised by Reynolds numbers up to about 20. Marine snow is believed to be the main vehicle for vertical material transport in the ocean, and is a site of elevated biological activity (“hot spots”). Microorganisms residing on or comprising the aggregate exchange solutes with the ambient water and make particulate material soluble. As a result, various solutes (oxygen, dissolved organics, inorganic nutrients) are released and/or taken up by the sinking aggregate. The hydrodynamic disturbance generated by the sinking aggregate, and the chemical trail painted by leaking solutes in its wake, may provide cues to small zooplankters that colonise and feed on aggregates. We have used the model to explore these various processes in Kiørboe, Ploug, and Thygesen (2001) and Kiørboe and Thygesen (2001). We provide a few examples below and refer to these papers for further details and references to the literature.

6.1. Solute exchange

The exchange of solutes between a marine snow aggregate and the ambient water may be limited by the rate at which the solute is consumed or produced by the aggregate, or by the rate at which it is being transported by diffusion and advection towards the aggregate. These two situations correspond to a Neumann and a Dirichlet boundary condition, respectively. In the latter situation, solute flow may be enhanced by advection. The Sherwood number quantifies this enhancement, and is the ratio of solute flow in the presence and absence of advection. The Sherwood number has

been used in the past to evaluate the effect of advection (swimming, sinking) on nutrient and oxygen uptake in marine organisms and is, hence, a useful property in biological oceanography. Semianalytical estimates of Sherwood numbers are available in the literature for small ($\approx$0) Reynolds numbers and numerical results are available for large ($>$ 100) Reynolds numbers, see Clift et al. (1978). However, to our knowledge there are no data available for intermediate Reynolds numbers (0–20) and diffusion coefficients characteristic of small biological molecules in water (10^{-9} m^2s^{-1}). Our software environment allows estimation of Sherwood numbers in this range, using the following procedure.

First, choose the appropriate flow field in window 2 (Stokes flow or a numerical solution). Then, in window 3, choose the relevant Peclet number, consider only molecular diffusion by setting $\lambda = 1$, chose a Dirichlet boundary condition, and normalise the results by the boundary condition. Window 4 will provide the estimate of the dimensionless solute flow F'. The corresponding dimensional flow is, according to Table 1

$$F_{Pe} = ca^2UF' = F'\,Pe\,\mathrm{D}(C_a - C_\infty).$$

This flow should be compared with the flow in pure diffusion, which can be shown to be

$$F_0 = 4\pi\mathrm{D}(C_a - C_\infty).$$

The Sherwood number is then the solute flow at the specified *Re* and *Pe* divided by the solute flow for pure diffusion:

$$Sh = \frac{F_{Pe}}{F_0} = \frac{F'Pe}{4\pi}.$$

As a sidestep, this method can also be used to assess the accuracy of the numerical discretisation. To this end, notice that if the Peclet number is low, the procedure above should result in a Sherwood number of 1. However, the discretisation introduces some numerical diffusion, leading to higher observed Sherwood numbers. For instance, applying the procedure to a uniform 40-by-39 grid extending 20 radii (not a very large grid) leads to an estimated Sherwood number of 1.05, whenever the Peclet number is smaller than 10^{-6}.

6.2. The chemical trail

For Reynolds numbers typical of many marine snows, the plume of elevated or depleted solute concentration is long and slender, as in Fig. 5. Zooplankters may be able to follow such chemical trails, and bacteria may enjoy the elevated concentrations of organic solutes in the wake of a sinking marine snow aggregate. For realistic estimates of e.g. amino acid leakage rates from aggregates, the length of the plume with concentrations significantly exceeding ambient may be substantial, up to the order of 100 radii.

The model estimates the steady state distribution of solutes. One may ask how

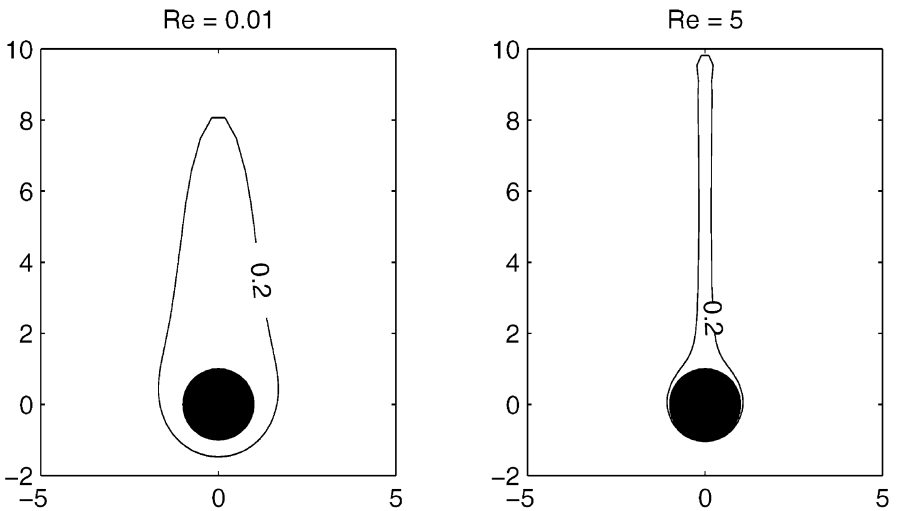

Fig. 5. The chemical trail after marine snow of different sizes. The plot shows lines of constant concentration for $Re = 0.01$ (left) and $Re = 5$ (right), corresponding to particle sizes of 35 μm and 5 mm, respectively.

representative steady state is, i.e., what is the time scale of solute transport. At Peclet numbers much less than 1, solute transport is governed by diffusion. The diffusion time scale is $L^2/6D$, where L is the length scale. Thus, the time to diffuse, e.g., 20 radii away from a 0.5 cm radius non-sinking aggregate is of the order $(5\ \text{cm})^2/(6{\cdot}10^{-5}\ \text{cm}^2\text{s}^{-1}) \approx 20$ days. This is much longer than the likely lifetime of an aggregate. At Peclet numbers much greater than 1, advection governs solute transport, and here the time scale is rather given by L/U, where U is the aggregate sinking velocity. Thus, the time required to establish the concentration field within 20 radii of a 0.5 cm aggregate sinking at 0.1 cm s^{-1} is only 100 s. Thus, steady state is not a bad assumption (in a non-turbulent environment).

6.3. Hydrodynamic disturbance

Many zooplankters may perceive and respond to fluid disturbances, specifically fluid deformation Δ. Thus, the fluid deformation generated by a sinking aggregate may potentially allow zooplankters to remotely detect and colonise aggregates. The deformation field generated by a typical marine snow aggregate of 0.5 cm radius sinking at 0.1 cm s^{-1} ($Re = 5$) is seen in Fig. 6. For a small (1 mm) copepod it takes a deformation rate of at least 0.1 s^{-1} to elicit a behavioural response. Using Table 1, this corresponds to a dimensionless deformation of 0.5. Thus, the 0.5 contour line in Fig. 6 indicates where a small copepod may detect a sinking aggregate. Evidently, in this case the sinking aggregate is detected less than a radius away.

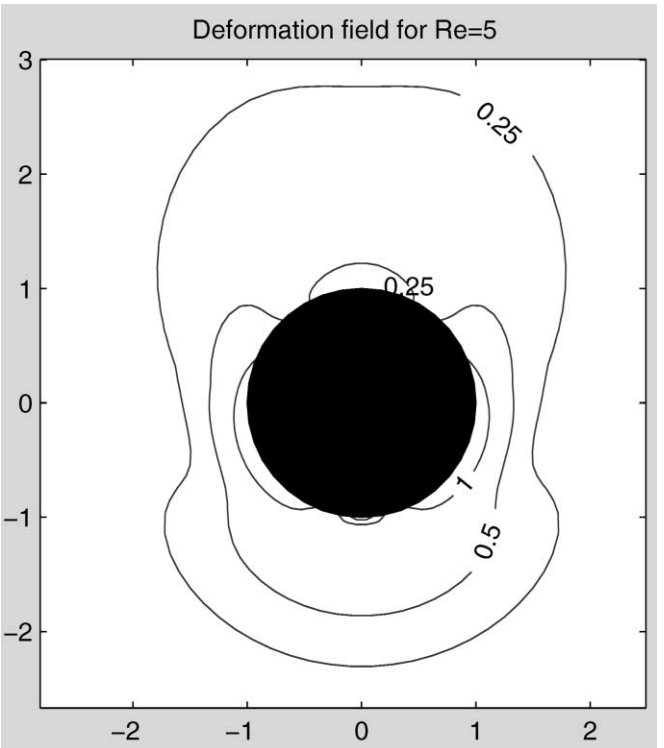

Fig. 6. Contour plot of the fluid deformation around marine snow of radius 0.5 cm, corresponding to a Reynolds number of 5.

Acknowledgements

This work was supported financially by the Danish Natural Science Research Council under grant No. 9801391. We thank Andy Visser for fruitful discussions and helpful suggestions.

Appendix A. Installation and startup

The environment is distributed in one archive. To install it, simply extract this archive (using `unzip` on Unix, `WinZip` on Microsoft Windows, or your favourite archiving tool depending on your operating system) to a directory of your choice.

To start the environment, start Matlab and change directory to that directory, then issue the command `Snow`. This brings up a little startup window. Clicking OK then brings up the four windows in Figs. 1–4.

Appendix B. System requirements

The environment should run on any platform which has Matlab version 5.2, 5.3 or 6.0 installed. We have not tested the environment on earlier versions. We have tested the environment on PCs running Windows NT, Windows 98, and Linux, as well as on an HP running HP-UX. We will be pleased to hear of experiences with running the environment on other platforms.

There are no absolute minimum hardware requirements in the sense that it is possible to run the environment with moderate resources, if one is satisfied with small grids and low Reynolds numbers. As an example of the computational burden, the figures in this report were generated on a PC with a 266 MHz processor and 64 MB RAM running Windows NT. Computing the fluid flow displayed in Fig. 2 (for a Reynolds number of 20) took 6 minutes; the grid was 40-by-40. Computing the concentration field was done on a 70-by-70 grid and took 20 seconds.

Appendix C. Parameter recommendations

This section contains recommendations with regard to the valid input ranges of parameters used in the environment.

C.1. Physical parameters

Parameter	*Symbol*	*Range*
Reynolds number	Re	0–20
Peclet number	Pe	0–20,000 (often $\approx 1000 \cdot Re$)
Relative molecular diffusivity	λ	0–1

The environment should be applicable to higher Reynolds and Peclet numbers as well, as long as the flow is steady. When we list $Re = 20$ as an upper limit, it means that we have not validated solutions above this number.

C.2. Grid sizes

The most suitable grid depends on both the physical parameters and the purpose of the calculation. High Peclet numbers require fine resolution near the sphere. If one is interested in the extent of the tail, this requires a larger grid than if one only wants the variation near the sphere. Another consideration is that for high Peclet

numbers, the upstream influence is small, so the computational domain need not extend far beyond the region of interest.

In general, we recommend beginning with small grids, perhaps 30-by-30, extending 10 radii or even less, and then extending and refining gradually.

We recommend using uniform grids (i.e., $\gamma = 0$) for computation of flows, at least up to Reynolds numbers around 20. For computation of concentration fields, we recommend increasing the downstream resolution as the Peclet number increases, ranging from $\gamma = 0$ for a Peclet number of 0 up to $\gamma = 0.7$ for a Peclet number of 20,000.

The approximation of the outer boundary condition in Section 4 results in some underestimation of the concentration field near the outer boundary, in particular when the computational grid does not extend very far. As a result, the size of the plume is also underestimated. If, however, the transport is dominated by advection, the error does not propagate upstream and so does not affect fluxes and concentrations near the surface. It is easy to identify potential problems by inspecting a plot of the concentraton field along the axis of symmetry (a so-called polar plot). If this plot shows a steady decline followed by a sharp drop to 0 at the outer boundary, then the computational grid does not cover the entire plume.

C.3. Discretisation method

We recommend the third/fourth order upwind scheme for final computations. The central second order scheme is around twice as fast and exposes numerical problems more clearly, so it may be used in an early exploratory phase. The reasoning behind this statement is that the main numerical weakness of the second order central scheme is unboundedness, which leads to very visible ripples in the solution, whereas the main numerical problem with the third/fourth order upwind scheme is numerical diffusion, which can be harder to identify.

References

Acheson, D. J. (1990). *Elementary fluid dynamics*. Clarendon Press.

Acrivos, A., & Goddard, J. D. (1965). Asymptotic expansions for laminar forced-convection heat and mass transfer. *J. Fluid Mech., 23*, 273–291.

Acrivos, A., & Taylor, T. D. (1962). Heat and mass transfer from single spheres in Stokes flow. *Physics of Fluids, 5*(4), 387–394.

Chang, E. J., & Maxey, M. R. (1994). Unsteady flow about a sphere at low to moderate Reynolds number. Part 1. Oscillatory motion. *J. Fluid Mech., 277*, 347–379.

Clift, R., Grace, J. R., & Weber, M. E. (1978). *Bubbles, drops and particles*. New York: Academic Press, 380 pp.

Dennis, S. C. R., & Walker, J. D. A. (1971). Calculation of the steady flow past a sphere at low and moderate Reynolds number. *J. Fluid Mech., 48*(4), 771–789.

Jackson, G. A. (1989). Simulation of bacterial attraction and adhesion to falling particles in an aquatic environment. *Limnol. Oceanogr., 34*(3), 514–530.

Kiørboe, T., Ploug, H., & Thygesen, U. H. (2001). Fluid motion and solute distribution around sinking aggregates. i. Small scale fluxes and heterogeneity of nutrients in the pelagic environment. *Marine Ecology Progress Series, 211*, 1–13.

Kiørboe, T., & Thygesen, U. H. (2001). Fluid motion and solute distribution around sinking aggregates. ii. Implications for remote detection by colonizing zooplankters. *Marine Ecology Progress Series, 211*, 15–25.

Kiørboe, T., & Visser, A. W. (1999). Predator and prey perception in copepods due to hydromechanical signals. *Marine Ecology Progress Series, 179*, 81–95.

Richardson, L. F. (1926). Atmospheric diffusion shown on a distance neighbor graph. *Proc. Roy. Soc. London Ser. A, 110*, 709–737.

Shraiman, B. I., & Siggia, E. D. (2000). Scalar turbulence. *Nature, 405*, 639–646.

The Mathworks, Inc. (1999). *Using Matlab*. http://www.mathworks.com

van Dyke, M. (1964). *Perturbation methods in fluid mechanis*. Academic Press.