

# GRID-FREE ADAPTIVE SEMI-LAGRANGIAN ADVECTION USING RADIAL BASIS FUNCTIONS

JÖRN BEHRENS\* AND ARMIN ISKE†

**Abstract.** This paper proposes a new grid-free adaptive advection scheme. The resulting algorithm is a combination of the semi-Lagrangian method (SLM) and the grid-free radial basis function interpolation (RBF). The set of scattered interpolation nodes is subject to dynamic changes at run time: Based on a posteriori local error estimates, a self-adaptive local refinement and coarsening of the nodes serves to obtain enhanced accuracy at reasonable computational costs. Due to well-known features of SLM and RBF, the method is guaranteed to be stable, it has good approximation behaviour, and it works for arbitrary space dimension. Numerical examples in two dimensions illustrate the performance of the method in comparison with existing grid-based advection schemes.

**Key words.** Adaptive advection schemes, semi-Lagrangian method, radial basis functions.

**1. Introduction.** Let  $I = [0, T] \subset \mathbb{R}$ ,  $T > 0$ , be a compact time interval, and let the computational domain be  $\Omega = \mathbb{R}^d$ ,  $d \geq 1$ . We consider the linear *advection equation*

$$(1.1) \quad \frac{d}{dt}c(t, x) = 0$$

in  $\bar{\Omega} = I \times \Omega$ , where  $c : \bar{\Omega} \rightarrow \mathbb{R}$  models the time-dependent distribution of the *concentration* in  $\Omega$ , i.e. for each pair  $(t, x) \in \bar{\Omega}$ ,  $c(t, x)$  returns the concentration at  $x \equiv x(t) \in \Omega$  at time  $t \in I$ . We rewrite equation (1.1) in the Eulerian form

$$(1.2) \quad \frac{\partial c}{\partial t} + \frac{dx}{dt} \cdot \nabla c = 0,$$

and we shall throughout this paper assume that the *velocity field*

$$\frac{dx}{dt} = a(t, x)$$

is known. The interpretation of (1.2) is that the scalar function  $c$  is constant along *trajectories*. Each of these trajectories is entirely determined by  $a : \bar{\Omega} \rightarrow \mathbb{R}^d$ . Given the initial distribution of the concentration

$$(1.3) \quad c(t, x)|_{t=0} = c^0(x),$$

$c^0$  with a compact support in  $\mathbb{R}^d$ , our aim is to numerically solve equation (1.1). Note that the formulation of equation (1.1) in the whole  $\mathbb{R}^d$  together with the initial condition (1.3) helps to avoid considering explicit boundary conditions.

One well-established method class for solving (1.1) are the *semi-Lagrangian methods* (SLM). We combine one *adaptive* such method, to be explained in the following Section 2, with the *grid-free* radial basis function interpolation, which is subject of the discussion in Section 3. This results in a new grid-free adaptive advection scheme, whose performance is illustrated by numerical examples in Section 4, including a comparison with existing grid-based adaptive advection schemes.

---

\*Zentrum Mathematik, Technische Universität München, Arcisstraße 21, D-80290 München, Germany (behrens@mathematik.tu-muenchen.de).

†Zentrum Mathematik, Technische Universität München, Arcisstraße 21, D-80290 München, Germany (iske@mathematik.tu-muenchen.de).

**2. Adaptive Semi-Lagrangian Advection.** In this section, we briefly review the (backward) SLM, where special attention is paid to its adaptive version [1]. For a specific discussion on the SLM and its applications in meteorology and environmental modelling we refer to the survey [14], whereas for a more general overview we recommend the textbook [11], Section 7.

The SLM integrates the Lagrangian form of the advection equation along trajectories. Therefore, the starting point of this particular advection scheme is the discretization

$$\frac{c(t + \tau, x) - c(t, x^-)}{\tau} = 0, \quad x \in \Omega,$$

of (1.1), where  $t \in I$  is the current time,  $\tau > 0$  the time step size, and  $x^-$  the *upstream point* corresponding to  $x$ . The point  $x^-$  gives us the unique location of that particle at time  $t$ , which by traversing along its trajectory arrives at  $x$  at time  $t + \tau$ . In particular,  $c(t + \tau, x) = c(t, x^-)$ .

For a finite set  $X = \{x_1, \dots, x_n\} \subset \Omega$  of (current) *nodes*, the SLM requires at each time step  $t \rightarrow t + \tau$  computing a vector

$$c_X^{t+\tau} = (c(t + \tau, x_1), \dots, c(t + \tau, x_n))^T \in \mathbb{R}^n$$

of new concentration values at time  $t + \tau$  from the corresponding vector  $c_X^t$  of the previous time step. Recall that at time  $t = 0$  the values in  $c_X^0$  are given by the initial condition (1.3). The computation of each new concentration value  $c(t + \tau, x)$ ,  $x \in X$ , is accomplished as follows.

- (a) Compute  $\tilde{x}$ , an approximation of the upstream point  $x^-$ ;
- (b) Interpolate  $c(t, \tilde{x})$  using the values  $c_X^t$ ;
- (c) Advect by letting  $c(t + \tau, x) = c(t, \tilde{x})$ .

In [14] (see also [11], equation (7.66a)) it was recommended to use a recursion of the form

$$\alpha_{k+1} = \tau \cdot a(t + \tau/2, x - \alpha_k/2)$$

in order to compute after merely a few iterations a sufficiently accurate linear approximation  $\alpha \in \mathbb{R}^d$  of the trajectory arriving at  $x$ . The point  $\tilde{x}$  in step (a) is then obtained by  $\tilde{x} = x - \alpha$ . However, as pointed out in [14] and confirmed by the numerical examples in [1], the interpolation in step (b) is clearly the most critical part for the success of the above integration scheme. Usually, the computation of  $c(t, \tilde{x})$  in step (b) is done by finite element methods (cf. [11, 14]). Due to the arising computational costs required for the maintenance of a *finite element mesh*, such methods are, however, prohibitively expensive in higher dimensions. In consequence, their range of applications is usually limited to lower dimensional problems.

This severe restriction motivates *grid-free* methods, often also referred to as *mesh-less* methods. Generally speaking, a grid-free SLM is essentially based upon a grid-free interpolation scheme. To this end, we prefer to work with radial basis functions, which are modern and powerful tools for multivariate scattered data interpolation. Be it sufficient for the moment to say that this particular interpolation scheme works in arbitrary space dimension  $d \geq 1$  and for arbitrary spatial distribution of the interpolation nodes. More details on radial basis functions are discussed in the following Section 3.

In addition, we also pick up the idea of *adaptation*, as introduced in [1] for a (grid-based) SLM; for the purpose of achieving a good compromise between the required complexity and the method's accuracy, the node set  $X$  is subject to adaptive changes during the SLM (see [1] for further details). In combination, this leads us to a *generic* formulation of a grid-free adaptive SLM, each of whose time steps  $t \rightarrow t + \tau$  involves the following computations.

ALGORITHM 1. (*Adaptive SLM using RBF.*)

**INPUT:**  $X \equiv X(t)$ , and  $c_X^t$ .

**REPEAT** {

(1) **FOR** each  $x \in X$  **DO** {

(a) Compute  $\tilde{x}$ , and determine a set  $\mathcal{N} \equiv \mathcal{N}(\tilde{x}) \subset X$  of its neighbours;

(b) Compute  $s$ , the radial basis function interpolant satisfying  $s|_{\mathcal{N}} = c(t, \cdot)|_{\mathcal{N}}$ ;

(c) Advect by letting  $c(t + \tau, x) = s(\tilde{x})$ .

}

(2) Adaptively modify  $X$  by refinement/coarsening of nodes, using a posteriori error estimations at each  $x \in X$ ;

**UNTIL** ( $X$  has not been changed in step (2))

**OUTPUT:**  $X \equiv X(t + \tau)$ , and  $c_X^{t+\tau}$ .

Note that the above Algorithm 1 is entirely grid-free, if the computations in all of its steps (1a), (1b), (1c), and (2) do not make use of any grid. While the steps (1b),(1c) are grid-free by the nature of radial basis functions, in step (1a) various different (grid-free) options are possible for the selection of  $\mathcal{N}$ . In order to make one concrete example, for some specific number  $k$ , the set  $\mathcal{N}$  may for instance be given by the  $k$  nearest neighbours of  $\tilde{x}$  in  $X$ .

The grid-free implementation of step (2) is, however, rather challenging. Indeed, the practicability of step (2) requires sophisticated rules for the adaptive refinement and coarsening of the nodes. But this delicate point is beyond the scope of this paper. We are currently investigating and evaluating various different grid-free strategies for step (2) to be presented in the future.

The purpose of this paper is more specific. Note that grid-free (meshless) methods are novel techniques for numerically solving partial differential equations. In order to mention two which are related to radial basis functions, recent theoretical developments include meshless Galerkin-type methods [15] and collocation methods [7, 8]. Their competitiveness in practical applications, however, remains to be shown by numerical comparison with classical and well-established methods, such as finite element methods. This contribution makes one step into this direction. We intend to draw the attention to the availability of radial basis functions for numerically solving advection equations. To this end, we show their utility by a numerical comparison against a finite element method of recommendation from [1]. The numerical tests are presented in Section 4.

**3. Radial Basis Function Interpolation.** In this section we explain relevant features of the grid-free radial basis function interpolation which enters step (1b) of the Algorithm 1. For a comprehensive review on radial basis functions we recommend the survey papers [4, 6, 12, 13]. Let us refer to the notations in Algorithm 1. Suppose that  $\tilde{x}$  is an approximation to the upstream point belonging to the node  $x \in X$ , and

let  $\mathcal{N} \equiv \mathcal{N}(\tilde{x}) = \{x_1, \dots, x_k\} \subset X$  denote a fixed set of points in the neighbourhood of  $\tilde{x}$ . Recall that the values of  $c(t, \cdot)$  at  $\mathcal{N}$  are known from the previous time step. According to the radial basis function interpolation scheme, for a fixed *radial* function  $\phi : [0, \infty) \rightarrow \mathbb{R}$ , the interpolant  $s : \mathbb{R}^d \rightarrow \mathbb{R}$  satisfying

$$s|_{\mathcal{N}} \equiv c(t, \cdot)|_{\mathcal{N}},$$

as required in step (1b) of Algorithm 1, is of the form

$$(3.1) \quad s = \sum_{j=1}^k \lambda_j \phi(\|\cdot - x_j\|) + p.$$

Here,  $\|\cdot\|$  denotes the Euclidean norm on  $\mathbb{R}^d$ , and  $p$  is an element of  $\Pi_m^d$ , the linear space of all  $d$ -variate polynomials of degree less than  $m$ . The following table provides a list of commonly used radial basis functions along with the corresponding order  $m$  of the polynomial part  $p$  in (3.1).

Thin Plate Splines	$\phi = r^2 \log(r)$	$m = 2$
Multiquadrics	$\phi = (r^2 + 1)^{1/2}$	$m = 1$
Gaussians	$\phi = e^{-r^2}$	$m = 0$
Inverse Multiquadrics	$\phi = (r^2 + 1)^{-1/2}$	$m = 0$

Assume that  $p_1, \dots, p_q$  is a basis of  $\Pi_m^d$ , i.e.  $q = \binom{m-1+d}{d}$ , where  $q \leq k$ . Then, the coefficient vector  $(\lambda^T, \mu^T)^T \in \mathbb{R}^{k+q}$  of the interpolant  $s$ ,  $\lambda = (\lambda_1, \dots, \lambda_k)^T \in \mathbb{R}^k$  for the major part and  $\mu = (\mu_1, \dots, \mu_q)^T \in \mathbb{R}^q$  for the polynomial part, is to be computed subject to constraints

$$\sum_{j=1}^k \lambda_j p(x_k) = 0 \quad \text{for all } p \in \Pi_m^d.$$

Altogether, this amounts to solving the linear system

$$\begin{bmatrix} A_{\phi, \mathcal{N}} & P \\ P^T & 0 \end{bmatrix} \cdot \begin{bmatrix} \lambda \\ \mu \end{bmatrix} = \begin{bmatrix} (c(t, x_j))_{1 \leq j \leq k} \\ 0 \end{bmatrix},$$

where

$$A_{\phi, \mathcal{N}} = (\phi(\|x_j - x_\ell\|))_{1 \leq j, \ell \leq k} \in \mathbb{R}^{k \times k}, \quad P = (p_\ell(x_j))_{1 \leq j \leq k; 1 \leq \ell \leq q} \in \mathbb{R}^{k \times q}.$$

For each of the radial basis functions mentioned above the linear system has a solution which is unique provided that the point set  $\mathcal{N}$  is  $\Pi_m^d$ -*nondegenerate*, i.e.

$$(3.2) \quad p(x_j) = 0 \quad 1 \leq j \leq k \quad \implies \quad p \equiv 0 \quad \text{for } p \in \Pi_m^d$$

holds true. In this case, the interpolation scheme achieves to reproduce any polynomial  $p \in \Pi_m^d$  from its values in  $\mathcal{N}$ . Note that the above constraint (3.2) on the distribution of the points in  $\mathcal{N}$  is rather weak. Indeed, for  $m = 2$ , i.e.  $\Pi_m^d$  are the linear polynomials, condition (3.2) requires that the points in  $\mathcal{N}$  must not all lie on a straight line. Moreover, for  $m = 1$  the condition (3.2) is trivial, and for  $m = 0$  it is empty.

In our numerical examples, we considered working with thin plate splines for a test case in two dimensions, i.e.  $d = 2$ . This popular scattered data interpolation method,

dating back to Duchon [5], has been identified as the analogue of the classical natural cubic spline interpolation method for univariate functions (see [12], Section 3.2). According to [5] the thin plate spline interpolation scheme yields optimal interpolants in the Beppo-Levi space

$$\text{BL}^2 = \{f \in C(\mathbb{R}^2, \mathbb{R}) : D^\alpha f \in L_2(\mathbb{R}^2) \text{ for all } |\alpha| = 2\}$$

being equipped with the semi-norm

$$|f|_{\text{BL}^2}^2 = \int_{\mathbb{R}^2} (f_{x_1 x_1}^2 + 2f_{x_1 x_2}^2 + f_{x_2 x_2}^2) dx$$

which represents the bending energy of a thin plate of infinite extent. Due to the results in [16], we obtain global error estimates of the form

$$\|c(t, \cdot) - s\|_{L_\infty(\tilde{\Omega})} \leq C_f \cdot h$$

where

$$h \equiv h_{\mathcal{N}, \tilde{\Omega}} = \sup_{y \in \tilde{\Omega}} \min_{x \in \mathcal{N}} \|x - y\|$$

denotes the *fill distance* of  $\mathcal{N} \subset \tilde{\Omega}$  in a bounded and open domain  $\tilde{\Omega} \subset \Omega$  satisfying an interior cone condition.

In our particular application, however, we are mainly interested in the *local* approximation order of thin plate spline interpolation. To this end, we recall from the previous paper [9], that for  $c \equiv c(t, \cdot) \in C^r(\mathbb{R}^2, \mathbb{R})$ ,  $1 \leq r \leq 2$ , the local approximation order of the thin plate spline interpolation method is  $r$ , i.e.

$$|c(\tilde{x} + h\Delta x) - s^h(\tilde{x} + h\Delta x)| = \mathcal{O}(h^r), \quad h \rightarrow 0,$$

holds for every  $\Delta x \in \mathbb{R}^2$ . Here, for  $h > 0$ ,  $s^h$  denotes the thin plate spline interpolant satisfying

$$c(\tilde{x} + h\Delta x_j) = s^h(\tilde{x} + h\Delta x_j), \quad 1 \leq j \leq k,$$

where  $\Delta x_j = x_j - \tilde{x}$ ,  $1 \leq j \leq k$ . If  $c$  is only continuous, then the local approximation by  $s^h$  is at least consistent, i.e.

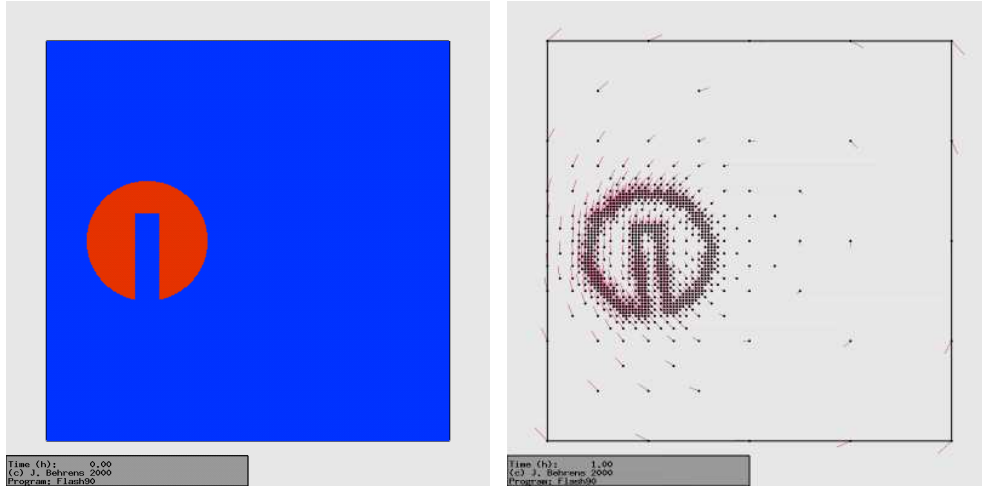
$$|c(\tilde{x} + h\Delta x) - s^h(\tilde{x} + h\Delta x)| \rightarrow 0, \quad h \rightarrow 0.$$

**4. Numerical Results.** We have implemented Algorithm 1, and we considered applying the method on the *slotted cylinder*, a common test case suggested by Zalesak [17]. In this test case, the domain  $\Omega = [-0.5, 0.5] \times [-0.5, 0.5] \subset \mathbb{R}^2$  is the shifted unit square, and the initial distribution of the concentration at time  $t_0 = 0$  is given by

$$c(0, x) = \begin{cases} 4 & \text{for } x \in D \\ 0 & \text{otherwise,} \end{cases}$$

where  $D \subset \Omega$  is the slotted disc, as displayed in the Figure 4.1 (left), of radius  $r = 0.15$ , centred at  $(-0.25, 0)$ , with slot width 0.06 and length 0.22. The slotted cylinder is rotated by the steady flow field  $a(x) = -\omega(-x_2, x_1)$ ,  $x = (x_1, x_2)$ , where  $\omega = 0.3636 \times 10^{-4}$ . We let  $\tau = 1800$  sec. for each time step size, so that one revolution of the slotted

FIG. 4.1. *Initial time step: The slotted cylinder (2D view, left); Distribution of the nodes and their trajectories (right).*



cylinder around the origin requires 96 time steps. Moreover, we let  $I = [0, 480\tau]$ . Note that, although the domain  $\Omega$  is not the whole  $\mathbb{R}^2$ , as required in the introduction, the test case complies with the original intention. The concentration never interacts with its boundary  $\partial\Omega$ , and therefore we do not have to consider boundary conditions.

For the purpose of illustration, we use thin plate spline interpolation in step (1b) of Algorithm 1. In order to chop off overshoots, as they typically arise when applying thin plate spline interpolation on discontinuous data (cf. the details in [9]), we adopt the idea of *clipping*, as suggested in [10] for the interpolation by cubic elements.

To this end, we replace  $s$  in (1b) of Algorithm 1 by

$$\bar{s}(\tilde{x}) = \begin{cases} c_{\min}, & \text{if } s(\tilde{x}) < c_{\min} \\ c_{\max}, & \text{if } s(\tilde{x}) > c_{\max} \\ s(\tilde{x}) & \text{otherwise,} \end{cases}$$

where  $s$  is the thin plate spline interpolant to  $c(t, \cdot)$  on  $\mathcal{N}$  and

$$c_{\min} = \min_{x \in \mathcal{N}} s(x), \quad c_{\max} = \max_{x \in \mathcal{N}} s(x)$$

are its extremal values. This yields the semi-Lagrangian method **SLM-TPS**, which we wish to compare with the recommended method **SLM-CUB** in [1], using cubic finite elements with clipping. We use the self-adaptive grid  $\mathcal{T}$  from [1] for the maintenance of the node set  $X$  (being the vertex set of  $\mathcal{T}$ ). Moreover, in order to make a fair comparison between the two methods, exactly the same rules for the adaptive refinement and coarsening of the node set  $X$  (resp.  $\mathcal{T}$ ) were applied. The distribution of the nodes after the initial sweep in Algorithm 1 are displayed in Figure 4.1 (right) along with the linear approximations of their trajectories.

As to the selection of the set  $\mathcal{N}$  of neighbours around an upstream point  $\tilde{x}$  in step (1a) of Algorithm 1 for **SLM-TPS**, we first locate a triangle  $T \in \mathcal{T}$  containing  $\tilde{x}$ , before we let  $\mathcal{N} = \bigcup_{v \in \mathcal{V}_T} \mathcal{C}_v$ , where  $\mathcal{V}_T$  is the set of the three vertices of the triangle  $T$ , and for any vertex  $v$  in  $\mathcal{T}$ ,  $\mathcal{C}_v$  is the vertex set of its cell.

For diagnostic reasons, we record the ratios of first and second mass moments,

$$\text{RFM}(t) = \frac{\int_{\Omega} c_h(x, t) dx}{\int_{\Omega} c(x, 0) dx} \quad \text{and} \quad \text{RSM}(t) = \frac{\int_{\Omega} |c_h(x, t)|^2 dx}{\int_{\Omega} |c(x, 0)|^2 dx},$$

and the  $L_2$ -error

$$\eta_2(t)^2 = \|c_h(t, \cdot) - c(t, \cdot)\|_{L_2(\Omega)}^2 = \int_{\Omega} |c_h(t, x) - c(t, x)|^2 dx$$

between the numerical solution  $c_h(t, \cdot)$  and the analytic solution  $c(t, \cdot)$ . The Figure 4.2 reflects our results. As the graph of  $\text{RFM}(t)$  in Figure 4.2 shows, the method **SLM-TPS** is slightly inferior, when it comes to preserving the first mass moment: The method **SLM-CUB** differs by at most 0.18 % from the ideal value 1.0, whereas for **SLM-TPS** the maximal deviation is 0.35 %. However, in terms of the conservation of the second mass moment, method **SLM-TPS** is clearly superior over **SLM-CUB**. Indeed, the value  $\text{RSM}(t)$  achieved by **SLM-TPS** is, compared with **SLM-CUB**, for all  $t \in I$  closer to the ideal value 1.0 at a maximal deviation of 4.28 %, compared with 9.34 % obtained by using **SLM-CUB**. Finally, the  $L_2$ -error  $\eta_2(t)$  of **SLM-TPS** is for all  $t \in I$  smaller than the corresponding error of the method **SLM-CUB**.

The distribution of the concentration values after the final time step, i.e. after five revolutions of the slotted cylinder, are for both methods, **SLM-TPS** and **SLM-CUB**, displayed in Figure 4.3. In both figures, the colour code is such that the concentration values are linearly scaled between *blue*, corresponding to zero, and *red*, corresponding to  $c \equiv 4$ .

**5. Conclusion and Future Work.** A new grid-free adaptive semi-Lagrangian method for the numerical solution of the linear advection equation has been proposed. The method is based on the grid-free radial basis function interpolation, and therefore works in arbitrary space dimension. The resulting scheme performs very well on the test case of Zalesak's slotted cylinder. Its accuracy (in terms of the  $L_2$ -error) and the conservation of the second mass moment is even better than a comparable well-introduced grid-based method of recommendation in [1].

We plan to present further improvements of the method by involving advanced refinement and coarsening rules for the scattered (interpolation) nodes. These details have widely been omitted in this paper. However, recall that the formulation in Algorithm 1 does not depend on a particular adaption strategy.

Finally, as shown in [2], we remark that a successful intergration scheme for the linear advection equation is potentially useful for the purpose of solving nonlinear advection equations, such as the *shallow water equations*. Therefore, by following along the lines of [2], we plan to develop a corresponding grid-free advection scheme for this important class of applications.

**Acknowledgments.** The second author was partly supported by the European Union within the project MINGLE (Multiresolution in Geometric Modelling), contract no. HPRN-CT-1999-00117.

## REFERENCES

- [1] J. Behrens, An adaptive semi-Lagrangian advection scheme and its parallelization, *Mon. Wea. Rev.* **124**, 1996, 2386–2395.
- [2] J. Behrens, Atmospheric and ocean modeling with an adaptive finite element solver for the shallow-water equations, *Applied Numerical Mathematics* **26**, 1998, 217–226.
- [3] R. Bermejo, and A. Staniforth, The conversion of semi-Lagrangian advection schemes to quasi-monotone schemes, *Mon. Wea. Rev.* **120**, 1992, 2622–2632.
- [4] M. D. Buhmann, Radial basis functions, *Acta Numerica*, 2000, 1–38.
- [5] J. Duchon, Splines minimizing rotation-invariant semi-norms in Sobolev spaces, in *Constructive Theory of Functions of Several Variables*, W. Schempp and K. Zeller (eds.), Springer, Berlin, 1977, 85–100.
- [6] N. Dyn, Interpolation and Approximation by Radial and Related Functions, in *Approximation Theory VI, Vol. 1*, C. K. Chui, L. L. Schumaker, J. D. Ward (eds.), Academic Press, 1989, 211–234.
- [7] C. Franke and R. Schaback, Convergence Orders of Meshless Collocation Methods using Radial Basis Functions, *Advances in Computational Mathematics* **8** (1998), no. 4, 381–399.
- [8] C. Franke and R. Schaback, Solving Partial Differential Equations by Collocation using Radial Basis Functions, *Applied Mathematics and Computation* **93** (1998), no. 1, 73–82.
- [9] T. Gutzmer, and A. Iske, Detection of Discontinuities in Scattered Data Approximation, in *Numerical Algorithms* **16:2**, 1997, 155–170.
- [10] S. Gravel, and A. Staniforth, A mass-conserving semi-Lagrangian scheme for the shallow-water equations, *Mon. Wea. Rev.* **122**, 1994, 243–248.
- [11] K. W. Morton, *Numerical Solution of Convection-Diffusion Problems*, Chapman & Hall, London, 1996.
- [12] M. J. D. Powell, The theory of radial basis function approximation in 1990, in *Advances in Numerical Analysis II: Wavelets, Subdivision, and Radial Basis Functions*, W. A. Light (ed.), Clarendon Press, Oxford, 1992, 105–210.
- [13] R. Schaback, Multivariate interpolation and approximation by translates of a basis function, in *Approximation Theory VIII*, C. K. Chui and L. L. Schumaker (eds.), World Scientific, Singapore, 1995, 491–514.
- [14] A. Staniforth, and J. Côté, Semi-Lagrangian Integration Schemes for Atmospheric Models – A Review, in *Monthly Weather Review* **119**, 1991, 2206–2223.
- [15] H. Wendland, Meshless Galerkin methods using radial basis functions, *Math. Comp.* **68**, 1999, 1521–1531.
- [16] Z. Wu, and R. Schaback, Local error estimates for radial basis function interpolation of scattered data, *IMA J. Numer. Anal.* **13**, 1993, 13–27.
- [17] S. T. Zalesak, Fully multidimensional flux-corrected transport algorithms for fluids, *J. Comput. Phys.* **31**, 1979, 335–362.



FIG. 4.2. The ratio of the first and second mass moment,  $RFM$  (top) and  $RSM$  (middle), and the  $L_2$ -errors  $\eta_2(t)$ ,  $t \in I$  (bottom).

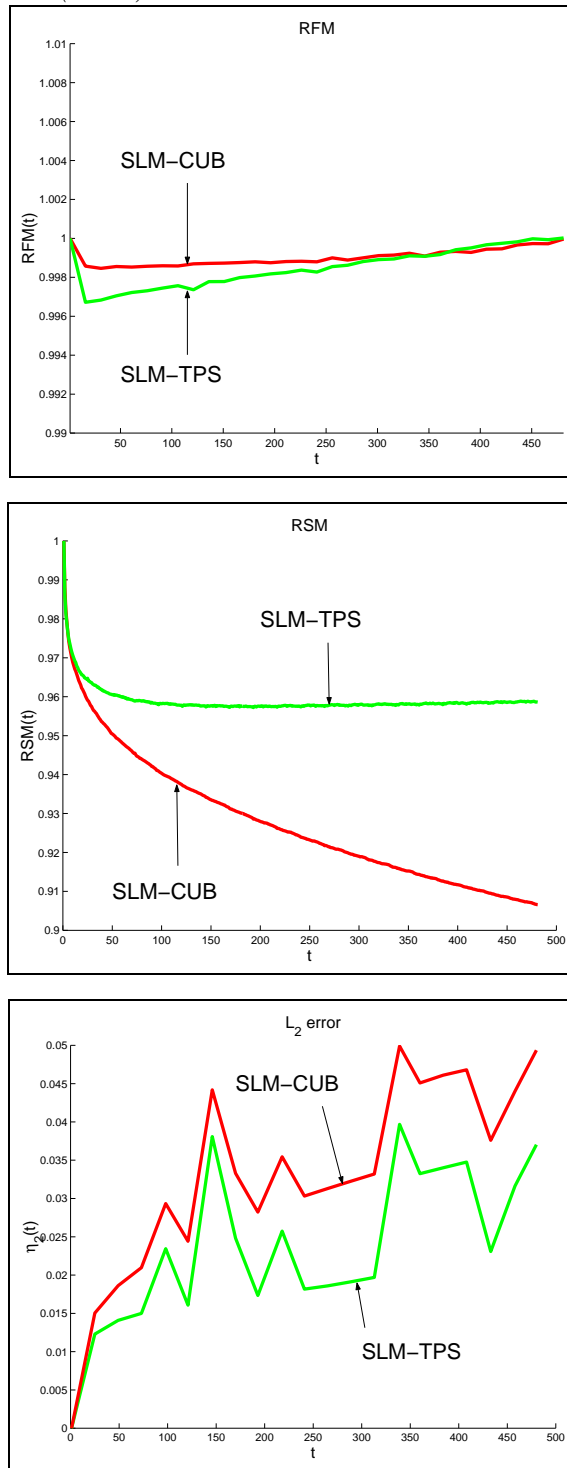


FIG. 4.3. Distribution of the concentration values after five revolutions of the slotted cylinder using the grid-free method **SLM-TPS** (top) and the grid-based **SLM-CUB** (bottom).

