BCCS
TECHNICAL REPORT SERIES

Tracking of passive, neutrally buoyant particles using the Bergen Ocean Model

Tomas Torsvik, Helge Avlesen, Øyvind Thiem

REPORT No. 27 **January 18, 2011**

Supported by the Research Council of Norway through the project "Understanding coral distribution and conditions for growth in Norwegian Waters" (Cordino).
Contract number 146526/420

UNI Research
the University of Bergen research company

BERGEN, NORWAY

BCCS Technical Report Series is available at http://www.bccs.uni.no/publications/

Requests for paper copies of this report can be sent to:

Bergen Center for Computational Science, Høyteknologisenteret,
Thormøhlensgate 55, N-5008 Bergen, Norway

Abstract

The report presents an online particle tracking model that has been developed as an optional module in the Bergen Ocean Model. The method is based on equations describing the advection-diffusion process for Lagrangian particles, which has been formulated in a manner that is consistent with the Eulerian equations governing the ocean model. The theoretical basis for the model is presented in section 2, with more background material presented in appendices A and B. Section 3 describes the numerical implementation of the particle tracking model, and a number of benchmark test cases are presented in section 4.

Contents

1 Introduction

Ocean Circulation Models are often used for tracking of water masses and modeling transport of various substances such as pollution, nutrition, passive floaters, and organic matter. Transport is usually modeled either as a tracer concentration subject to advection and diffusion, or by Lagrangian transport of individual particles. A Lagrangian Stochastic Model (LSM) can be used to describe the transport (advection and dispersion) of clusters of particles, especially when high spatial and temporal resolution is required. If the stochastic component can be considered completely random, the LSM can be simplified to a random displacement model (RDM), with similar properties as the classical advection-diffusion problem (see Brickman and Smith [2002] for a further discussion on LSM and RDM).

There are a number of reasons why using a particle tracking model may be more attractive than modeling a tracer concentration (see e.g. Dimou and Adams [1993]). Sources are more naturally represented in a particle tracking model, where new particles can easily be introduced at different times, whereas it can be difficult to resolve a point source with a concentration model. A particle tracking model can provide information about the behavior or fate of individual agents, such as behavior of fish larvae under changing conditions, or settling of different size of particles. Particle tracking models are also a more natural choice when we are interested in integrated properties, such as residence time, rather than the concentration distribution itself.

This report describes the background theory and numerical implementation of a RDM which has been designed to run on-line with input from the Bergen Ocean Model (BOM). A number of benchmark test cases have been utilized to test the performance of the model.

2 Particle tracking in a C-grid, σ-coordinate ocean model

A particle tracking model for a σ-coordinate model with orthogonal curvilinear horizontal coordinates was presented by Blumberg et al. [2004]. Similar models have also been presented by Tompson and Gelhar [1990] and Dimou and Adams [1993]. The following is a short description of the main equations in a σ-coordinate ocean model, a description of a particle tracking model, and some elements that are important for the discretization of the equations.

2.1 Ocean model

Many ocean models apply the σ-coordinate system, where the vertical layers expand and contract depending on the bottom profile. The transformation between Cartesian coordinates (x_c, y_c, z_c, t_c) and σ-coordinates (x, y, σ, t) is defined by

$$x = x_c\,, \qquad y = y_c\,, \qquad \sigma = \frac{z_c - \eta}{H + \eta}\,, \qquad t = t_c\,, \tag{2.1}$$

where $\eta(x_c, y_c, t_c)$ is the free surface elevation and $H(x_c, y_c)$ is the water depth. To simplify notation, we introduce the total depth $D \equiv H + \eta$.

The governing equations for a σ-coordinate ocean model consist of the continuity equation

$$\frac{\partial UD}{\partial x} + \frac{\partial VD}{\partial y} + \frac{\partial \omega}{\partial \sigma} + \frac{\partial \eta}{\partial t} = 0\,, \tag{2.2}$$

where U and V are the horizontal velocity components in x and y directions, respectively, and ω is the vertical velocity relative to the σ-coordinate system. The relationship between the Cartesian vertical velocity component W and the σ-coordinate vertical velocity ω is given by

$$W = \omega + U\left(\sigma\frac{\partial D}{\partial x} + \frac{\partial \eta}{\partial x}\right) + V\left(\sigma\frac{\partial D}{\partial y} + \frac{\partial \eta}{\partial y}\right) + (1+\sigma)\frac{\partial \eta}{\partial t}\,.$$

The momentum equations on flux form are given by

$$\begin{aligned}\frac{\partial UD}{\partial t}+\frac{\partial U^2D}{\partial x}+\frac{\partial UVD}{\partial y}+\frac{\partial U\omega}{\partial \sigma}-fVD+\frac{D}{\rho_0}\frac{\partial P_{atm}}{\partial x}+gD\frac{\partial \eta}{\partial x}=\\ \frac{\partial}{\partial \sigma}\left(\frac{K_M}{D}\frac{\partial U}{\partial \sigma}\right)-g\frac{D^2}{\rho_0}\int_\sigma^0\left(\frac{\partial \rho}{\partial x}-\frac{\sigma}{D}\frac{\partial D}{\partial x}\frac{\partial \rho}{\partial \sigma}\right)\,d\sigma+DF_x\,,\end{aligned} \tag{2.3}$$

$$\begin{aligned}\frac{\partial VD}{\partial t}+\frac{\partial UVD}{\partial x}+\frac{\partial V^2D}{\partial y}+\frac{\partial V\omega}{\partial \sigma}+fUD+\frac{D}{\rho_0}\frac{\partial P_{atm}}{\partial y}+gD\frac{\partial \eta}{\partial y}=\\ \frac{\partial}{\partial \sigma}\left(\frac{K_M}{D}\frac{\partial V}{\partial \sigma}\right)-g\frac{D^2}{\rho_0}\int_\sigma^0\left(\frac{\partial \rho}{\partial y}-\frac{\sigma}{D}\frac{\partial D}{\partial y}\frac{\partial \rho}{\partial \sigma}\right)\,d\sigma+DF_y\,,\end{aligned} \tag{2.4}$$

where g is the acceleration of gravity, ρ is the density and ρ_0 is the density reference level, P_{atm} is the atmospheric pressure, f is the Coriolis parameter, and K_M is the vertical eddy viscosity. Motion induced by small scale processes (sub-grid scale) are parametrized by horizontal and vertical eddy viscosity terms (A_M, K_M) and horizontal and vertical eddy diffusivity terms (A_H, K_H). The horizontal viscosity terms in eqs. (2.3) and (2.4) are given by

$$DF_{x,y}=\frac{\partial}{\partial x}\left(A_M\frac{\partial(UD,VD)}{\partial x}\right)+\frac{\partial}{\partial y}\left(A_M\frac{\partial(UD,VD)}{\partial y}\right)\,. \tag{2.5}$$

The transport equation for a conservative tracer C is given by

$$\begin{aligned}\frac{\partial CD}{\partial t}+\frac{\partial CUD}{\partial x}+\frac{\partial CVD}{\partial y}+\frac{\partial C\omega}{\partial \sigma}=\\ \frac{\partial}{\partial x}\left(A_HD\frac{\partial C}{\partial x}\right)+\frac{\partial}{\partial y}\left(A_HD\frac{\partial C}{\partial y}\right)+\frac{\partial}{\partial \sigma}\left(\frac{K_H}{D}\frac{\partial C}{\partial \sigma}\right)\,.\end{aligned} \tag{2.6}$$

Transport equations for salinity S and temperature T are on the same form as the transport equation (2.6), and can be written by replacing C with S and T, respectively.

2.2 The particle tracking model

A Lagrangian description of particles with positions $\vec{X}(t)=(X_1(t),X_2(t),X_3(t))$ subject to advection and particle diffusion is given by the *Langevin equation*

$$\frac{d\vec{X}}{dt}=A(\vec{X},t)+B(\vec{X},t)\xi(t)\,, \tag{2.7}$$

where $A(\vec{X},t)$ represent deterministic forces acting on the particles, $B(\vec{X},t)$ represents random forces, and $\xi(t)$ is a vector composed of random numbers. The most fundamental diffusion process is the Wiener process. If we define

$$W(t)=\int_0^t \xi(s)\,ds\,,$$

the Langevin equation (2.7) becomes equivalent to the Ito stochastic differential equation

$$d\vec{X}=\vec{X}(t+dt)-\vec{X}=A(\vec{X},t)dt+B(\vec{X},t)dW(t)\,, \tag{2.8}$$

where the random Wiener process has the properties

$$\begin{aligned}\mu=E[dW(t)]=0 \qquad &\text{(zero mean)}\,,\\ \text{Var}\,[dW(t)]=E[dW(t)^2]=dt \qquad &\text{(variance proportional to } dt\text{)}\,.\end{aligned}$$

The discrete form of eq. (2.8) is

$$\Delta\vec{X}_{n+1} = \vec{X}_{n+1} - \vec{X}_n = A(\vec{X}_n, t_n)\Delta t + B(\vec{X}_n, t_n)\sqrt{\Delta t}Z_n \tag{2.9}$$

where Z_n is a vector of independent random numbers with zero mean and unit variance. If we define

$$f = f(\vec{X}, t|\vec{X}_0, t_0)$$

as the conditional probability density function for the positions $\vec{X}(t)$ for particles with initial positions $\vec{X}_0$ at time t_0, the *Fokker-Planck equation*

$$\frac{\partial f}{\partial t} + \frac{\partial}{\partial \vec{X}}(Af) = \nabla^2\left(\frac{1}{2}BB^T f\right), \tag{2.10}$$

determines the evolution of f in the limit as the number of particles becomes very large and the time step used to solve the transport equation becomes very small.

The Fokker-Plank equation (2.10) is the key link between transport of a conserved tracer in the ocean model and the transport of a cloud of particles. In order to construct a particle tracking model for neutrally buoyant particles that is consistent with the transport of a concentration in the ocean model, the Fokker-Planck equation (2.10) must be consistent with the the transport equation (2.6). By adding

$$\frac{\partial}{\partial x}\left(C\frac{\partial A_H D}{\partial x}\right) + \frac{\partial}{\partial y}\left(C\frac{\partial A_H D}{\partial y}\right) + \frac{\partial}{\partial \sigma}\left(C\frac{\partial}{\partial \sigma}\left(\frac{K_H D}{D^2}\right)\right)$$

to both sides of eq. (2.6), we can rearrange the terms in the transport equation

$$\begin{aligned}
&\frac{\partial CD}{\partial t} + \frac{\partial}{\partial x}\left\{\left[U + \frac{1}{D}\frac{\partial}{\partial x}(A_H D)\right]CD\right\} \\
&+ \frac{\partial}{\partial y}\left\{\left[V + \frac{1}{D}\frac{\partial}{\partial y}(A_H D)\right]CD\right\} + \frac{\partial}{\partial \sigma}\left\{\left[\frac{\omega}{D} + \frac{1}{D}\frac{\partial}{\partial \sigma}\left(\frac{K_H}{D^2}D\right)\right]CD\right\} \\
&= \frac{\partial^2}{\partial x^2}(A_H CD) + \frac{\partial^2}{\partial y^2}(A_H CD) + \frac{\partial^2}{\partial \sigma^2}\left(\frac{K_H}{D^2}CD\right).
\end{aligned} \tag{2.11}$$

Comparing the modified transport equation (2.11) with the Fokker-Planck equation (2.10), we find that we can construct a particle tracking model that is consistent with tracer transport if we choose

$$A \equiv \begin{bmatrix} U + \frac{1}{D}\frac{\partial}{\partial x}(A_H D) \\ V + \frac{1}{D}\frac{\partial}{\partial y}(A_H D) \\ \frac{\omega}{D} + \frac{1}{D}\frac{\partial}{\partial \sigma}\left(\frac{K_H}{D^2}D\right) \end{bmatrix} \tag{2.12}$$

and

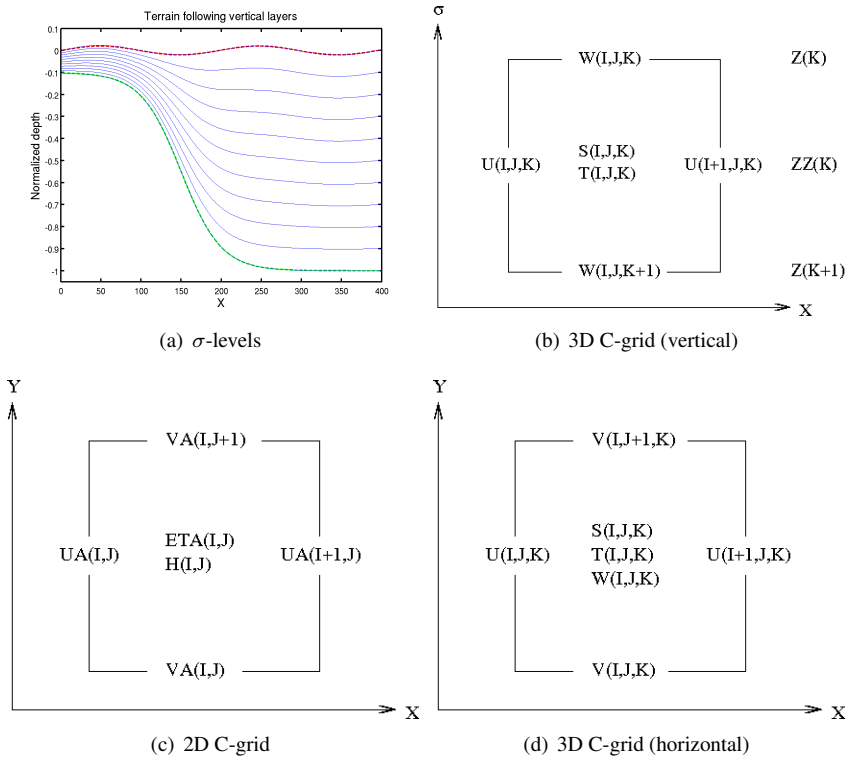
$$\frac{1}{2}BB^T \equiv \begin{pmatrix} A_H & 0 & 0 \\ 0 & A_H & 0 \\ 0 & 0 & \frac{K_H}{D^2} \end{pmatrix} \tag{2.13}$$

The particle tracking model in discrete form for particle p with drift coefficient A defined by eq. (2.12) and diffusion coefficient B defined by eq. (2.13), is given by

$$\begin{aligned}
X^{(p)}_{1,(n-1)} - X^{(p)}_{1,(n)} &= \left[U^{(p)}_{(n)} + \frac{1}{D}\frac{\partial}{\partial x}(A_H D)\right]\Delta t + \sqrt{2A_H\Delta t}\, Z_n \\
X^{(p)}_{2,(n-1)} - X^{(p)}_{2,(n)} &= \left[V^{(p)}_{(n)} + \frac{1}{D}\frac{\partial}{\partial y}(A_H D)\right]\Delta t + \sqrt{2A_H\Delta t}\, Z_n \\
X^{(p)}_{3,(n-1)} - X^{(p)}_{3,(n)} &= \left[\frac{\omega^{(p)}_{(n)}}{D} + \frac{1}{D}\frac{\partial}{\partial \sigma}\left(\frac{K_H}{D^2}D\right)\right]\Delta t + \sqrt{\frac{2K_H}{D^2}\Delta t}\, Z_n
\end{aligned} \tag{2.14}$$

2.3 Discretization of equations

The governing equations for BOM (eqs. (2.2)-(2.6)) and for the particle tracking model (PTM) (eq. (2.14)) are discretized on grid with an equidistant Arakawa C-grid configuration for the horizontal dimensions (X, Y), and terrain-following σ-levels for the vertical dimension. The discretization of BOM is described in the user guide Berntsen [2000], and a sketch of the configuration, including the placement of the main variables, is shown in Fig. 1.



(a) σ-levels (b) 3D C-grid (vertical)

(c) 2D C-grid (d) 3D C-grid (horizontal)

Figure 1: σ-level C-grid coordinate system. Figure 1(a) show a vertical profile with σ-level discretization. Figures 1(b) and 1(d) show the grid discretization for a single 3D cell in vertical and horizontal, respectively. Figure 1(c) show the grid coordinates for a 2D horizontal grid cell.

Fig. 1(a) illustrates that the σ-layers follow the dynamic surface elevation, i.e. the σ levels are time dependent in physical space. Note however that the surface wave motion in Fig. 1(a) is large relative to the water depth, which violates a key assumption in the ocean model, and therefore this particular large amplitude surface wave motion can not be simulated by BOM.

The discretization is in principle independent the physical size of the region to be simulated, in the sense that the discretization does not contain any information about the size of the physical domain, or of the spatial discretization length scales Δx and Δy. A convention made explicit in BOM 5.0 is that the origin in physical space corresponds to the corner between cells (1,1), (1,2), (2,1), and (2,2) in the grid space. The relation between the physical and grid coordinates is shown in Fig. 2. Since the depth matrix is specified at cell centers, the input data for the bathymetry should correspond to the points $(-0.5\Delta x, -0.5\Delta y)$, $(0.5\Delta x, -0.5\Delta y)$, $(-0.5\Delta x, 0.5\Delta y)$, etc., in physical space.

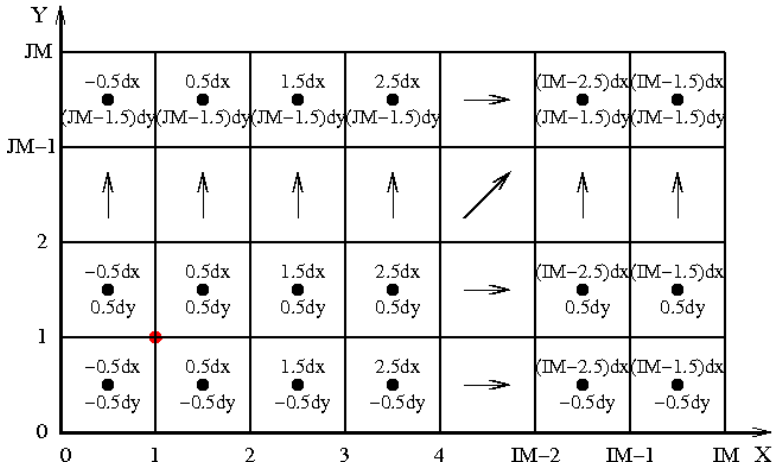


Figure 2: Conversion between physical (Cartesian) coordinates and grid coordinates in BOM. The location of the origin in physical space is marked by a red dot.

2.3.1 Discretization for the particle tracking model

The PTM use physical Cartesian coordinates (x_c, y_c, z_c) for input and output, but use a non-dimensional σ-coordinate system (x_p, y_p, σ_p) to keep track of the particles within the model. The mapping between physical and internal coordinates is defined by

$$x_p = \frac{x_c}{dx} + 1.0\,, \qquad y_p = \frac{y_c}{dy} + 1.0\,, \qquad \sigma_p = \frac{z_c - \eta}{H + \eta}\,.$$

With this convention, the *physical horizontal coordinate* $(x_c, y_c) = (0.0, 0.0)$ is mapped to the *grid horizontal coordinate* $(x_p, y_p) = (1.0, 1.0)$, i.e. the grid coordinates correspond to distances in $(\Delta x, \Delta y)$ from the west and south grid boundaries, respectively. Note that the grid coordinates specified in this way does not correspond directly to the index values (i, j) needed for the calculation of the model variables $(U, V, W, \text{etc.})$.

By default, the particles are restricted to live within the domain

$$0 < x_c < (IM - 2)\Delta x\,, \quad 0 < y_c < (JM - 2)\Delta y\,,$$

i.e. the particles are not tracked if they enter last cell towards any of the boundaries (this convention can be over-ruled when specifying boundary conditions for the particles). The reason for this convention is that the flow field can be highly artificial in the last cell towards the boundary, where boundary conditions for BOM are implemented. As a result, the effective boundary condition for particle tracking should be at the boundary between the last cell and the cell next-to-last near the boundary of the computational domain.

3 Implementation of the numerical particle tracking model

The PTM given by eq. (2.14) has been implemented and incorporated into BOM as an independent module. Both BOM and PTM are written in FORTRAN 95/2003. BOM is parallelized using a domain decomposition method, and a similar structure is also used for PTM. This enables PTM to take advantage of the framework provided by BOM for setup and message passing in the model. A brief description on how to use PTM is given in Appendix C.

BOM is parallelized using a domain decomposition procedure, where each processor is assigned a subregion, called a block, within the computational domain (see Fig. 3). Communication between neighboring blocks is achieved through the use of ghost zones,

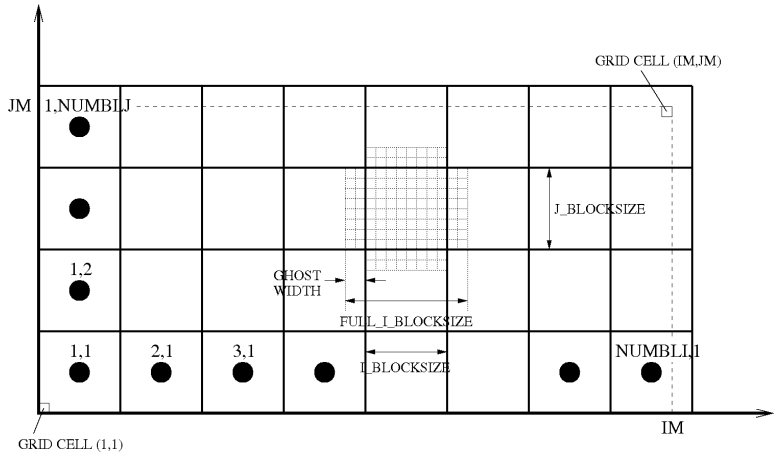


Figure 3: Domain decomposition in BOM

i.e. the subdomain for each processor overlap (by default the overlap is 4 cells, 2 on each side of the boundary). Most processes in the model only rely on local data around each cell, in which case the different processors only need to exchange data along the block boundaries.

The same domain decomposition structure is used also for PTM. All particles are initiated on all processors, but only made active within the processor where the particle is located in physical space. When a particle moves from block A to block B, relevant particle data must first be sent from processor A to processor B, then the particle is activated on B and de-activated on A. This procedure is fairly simple, especially since we can reuse existing parallelization structures in BOM, and is also efficient if particles are evenly distributed throughout the computational domain. However, it is not efficient for highly localized concentrations of particles, in which case a few processors do almost all the particle tracking work, while processors with few particles are mostly idle.

3.1 Structure of the particle tracking model

PTM for BOM is contained within a single module file, called *mod_particle.f90*, and all BOM subroutines dealing with particles should contain a "`USE MOD_PARTICLE`" statement in the preamble. PTM consist of three major parts: (1) setup of PTM, and initialization of the particle field, (2) Advection and random dispersion of particles, and (3) Analysis and writing data to files.

3.1.1 Setup of the particle tracking model

Particle tracking is initiated by calling the subroutine `setup_particle`. A flow chart for the routine is shown in Fig. 4. Several of the setup procedures are only done on a single processor, and the results are subsequently broadcasted to all other processors. Input parameters and initial particle positions are read from separate files (see Appendix C.1,C.2 for examples). Boundary conditions for the global domain are specified as (1) absorbing, (2) reflecting, or (3) periodic boundary conditions, and an offset distance (in index space) between the global BOM boundary location and the boundary location for PTM can be specified for each boundary. The subroutine also sets up arrays for exchanging particles between blocks (see also Section 3.1.2). At the end of the setup procedure, particles with initiation time $t = 0.0$ s are activated.

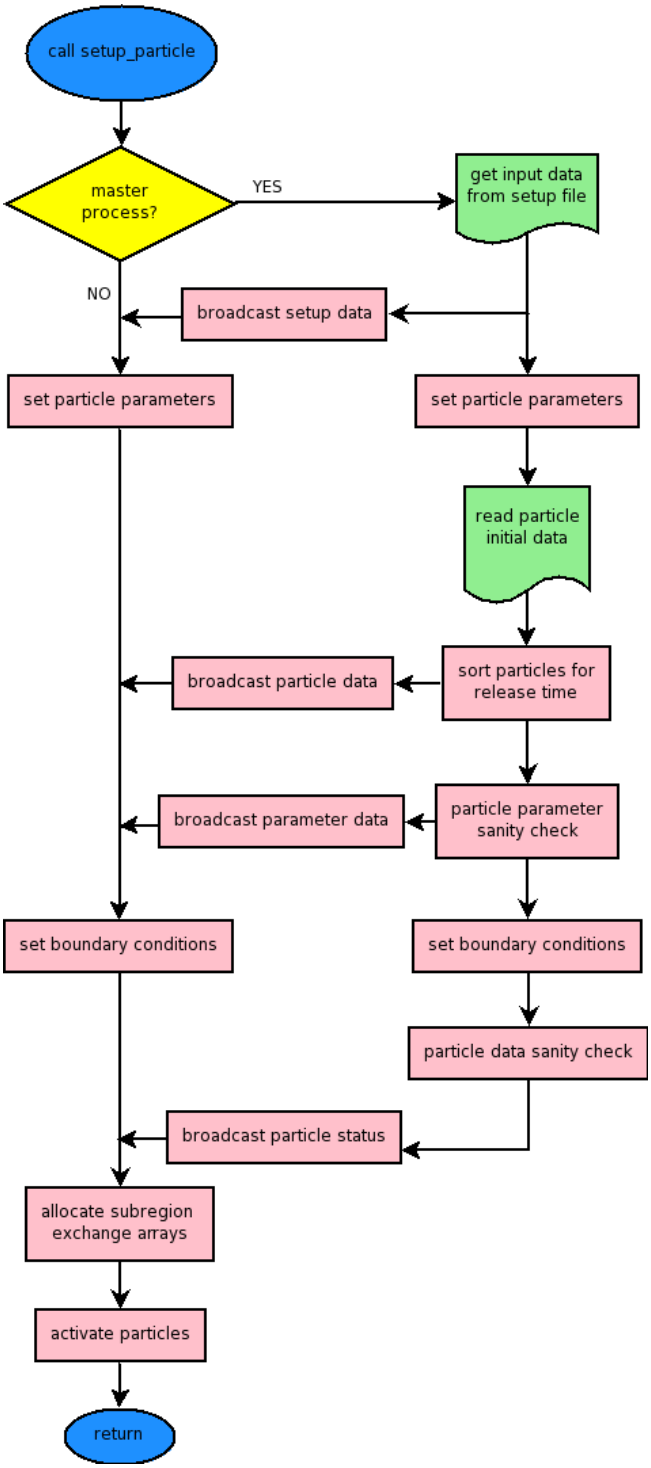


Figure 4: Flow chart for the `particle_setup` subroutine.

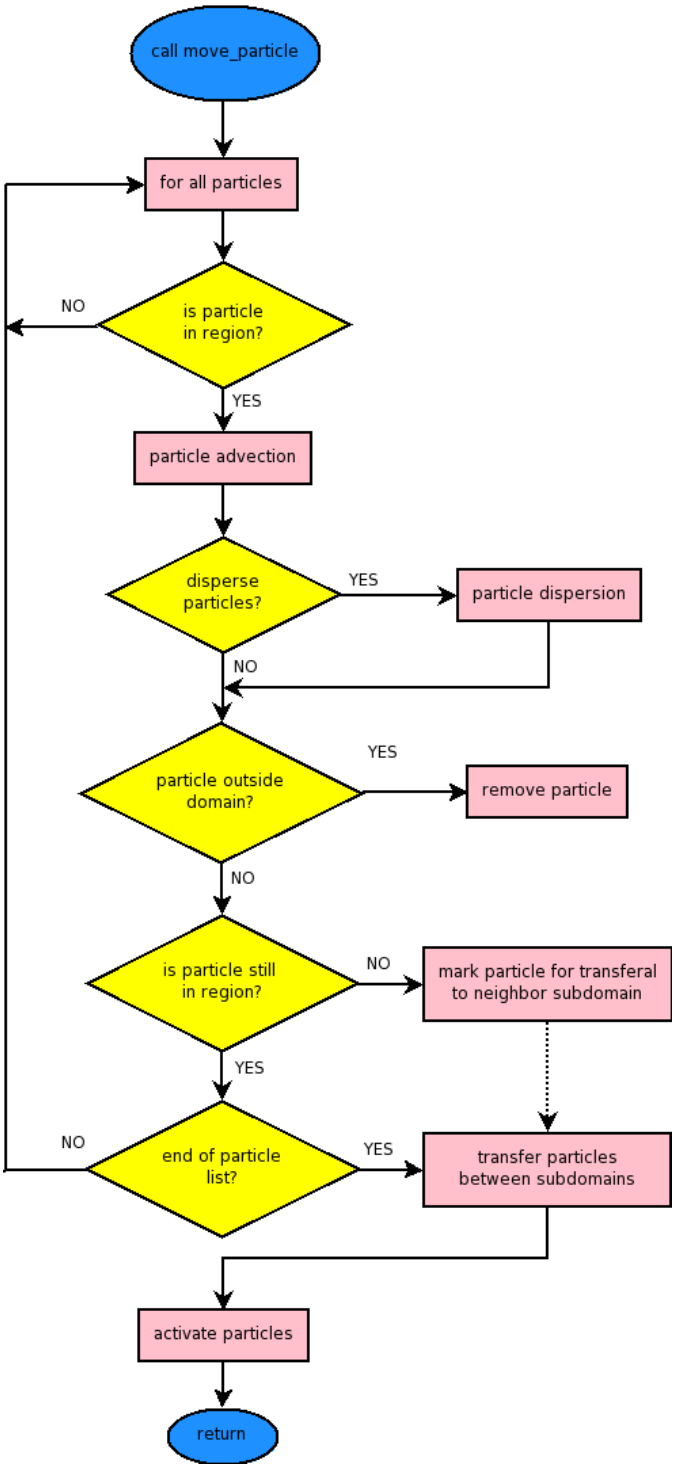


Figure 5: Flow chart for the `move_particle` subroutine.

3.1.2 Particle advection and dispersion

As mentioned earlier, each processor is responsible for advection and dispersion of particles within its own subregion of the domain. A flow chart for the procedure is shown in Fig. 5. Particle advection is performed through either (1) a single Euler forward time step, or (2) a two step Heun (predictor-corrector) method, with velocities estimated from either (i) linear or (ii) bi-linear interpolation. The random dispersion step is calculated using either (A) diffusion coefficients from BOM or (B) constant diffusion coefficients. Once a new particle position has been found, the model checks if the new position is within the valid vertical and horizontal range specified by the computational domain, and take appropriate action if this is not the case (e.g. bump particle back into the valid range, de-activate particle, etc.).

If the particle is still active, but has been moved within the region of a neighboring block, the particle data is copied to the appropriate exchange array (there are 4 arrays corresponding to transport east, west, north or south). Once new positions have been found for all particles, particle data is transferred between processors using one-sided communication. The sending processor is given access to write directly to memory allocated on the receiving processor. In the current implementation, the sending process first send a single integer specifying the number of particles being transferred (the default value of this integer is 0 in the receiver memory space). Thereafter the sender copies the appropriate exchange array of particle data to the accessible memory space provided by the receiver. This method reduces communication compared to standard two-sided send/receive. Data is only transferred between processors when needed, and the receiving process need only to check a single integer value to determine if any data has been received.

The last action of the subroutine is to activate particles with initiation time corresponding to the current time step.

3.1.3 Analysis and output

In the current implementation, the analysis option is limited to keeping track of which particles are activated on any given processor, and which particles have been de-activated for various reasons. Users can request writing (1) tracks of individual particles to separate files (up to 999 particle tracks), (2) the location of all active particles at a specific time to a file, and (3) the number of particles that have been de-activated within each horizontal grid cell to file at the end of the simulation.

4 Benchmark test cases for particle tracking

Benchmark testing has been performed on a Dell Optiplex 760 computer with an Intel Core Quad (2.5 GHz) processors, using the Intel ifort compiler (version 11.1), and a Cray XT4 (hexagon) system with a total of 5552 AMD Opteron quad-core (2,3 GHz) processors, using the Portland pgf90 compiler (version 10.6). A set of different compiler options were used for the test cases to test sensitivity to optimization and computational accuracy (see Table 1).

compiler option	ifort compiler	pgf90 compiler
A	-O3 / -O1$^{(*)}$	-O3
B	-r8 -O3 / -r8 -O1$^{(*)}$	-r8 -O3
C	-g	-g
D	-r8 -g	-r8 -g

$^{(*)}$ Optimization with -O3 lead to errors in some subroutines when using the ifort compiler.

Table 1: Compiler flags. Sensitivity to optimization and computational accuracy was tested using a set of different compiler options.

In addition to the compiler options from Table 1, the influence of time stepping methods and interpolation methods mentioned in Section 3.1.2 will also be examined in with the benchmark test cases. In the interest of brevity, the notation in Table 2 is introduced to describe the test case parameters; EF/PC represent time stepping with Euler Forward and Predictor-Corrector methods; lin/bil represent linear and bi-linear interpolation; S/D represent compilation with single and double floating point precision.

Label	time stepping	interpolation	compiler option
EF-lin-S	Euler Forward	linear	-O3 / -O1
EF-lin-D	Euler Forward	linear	-r8 -O3 / -r8 -O1
EF-bil-S	Euler Forward	bi-linear	-O3 / -O1
EF-bil-D	Euler Forward	bi-linear	-r8 -O3 / -r8 -O1
PC-lin-S	Predictor-Corrector	linear	-O3 / -O1
PC-lin-D	Predictor-Corrector	linear	-r8 -O3 / -r8 -O1
PC-bil-S	Predictor-Corrector	bi-linear	-O3 / -O1
PC-bil-D	Predictor-Corrector	bi-linear	-r8 -O3 / -r8 -O1

Table 2: Labels for test case parameters

4.1 Test case 1: 3D box

The test case is a 3D box with closed boundaries in all directions, and flat bottom (see parameters in Table 3). In the absence of any external forcing, the fluid should remain stationary throughout the simulation.

IM	JM	KB	dx	dy	$H_{\max}$
52	52	51	2.0 m	2.0 m	50.0 m

Table 3: 3D box case parameters

4.1.1 Test case 1A: Particles completely at rest

This test was performed using 7 particles without any random dispersion, over 1 hour with 1 s time step. Tests were made using compiler options A-D in Table 1. The test results confirm that all particles remain stationary throughout the simulation.

4.1.2 Test case 1B: Particle dispersion in non-stratified fluid

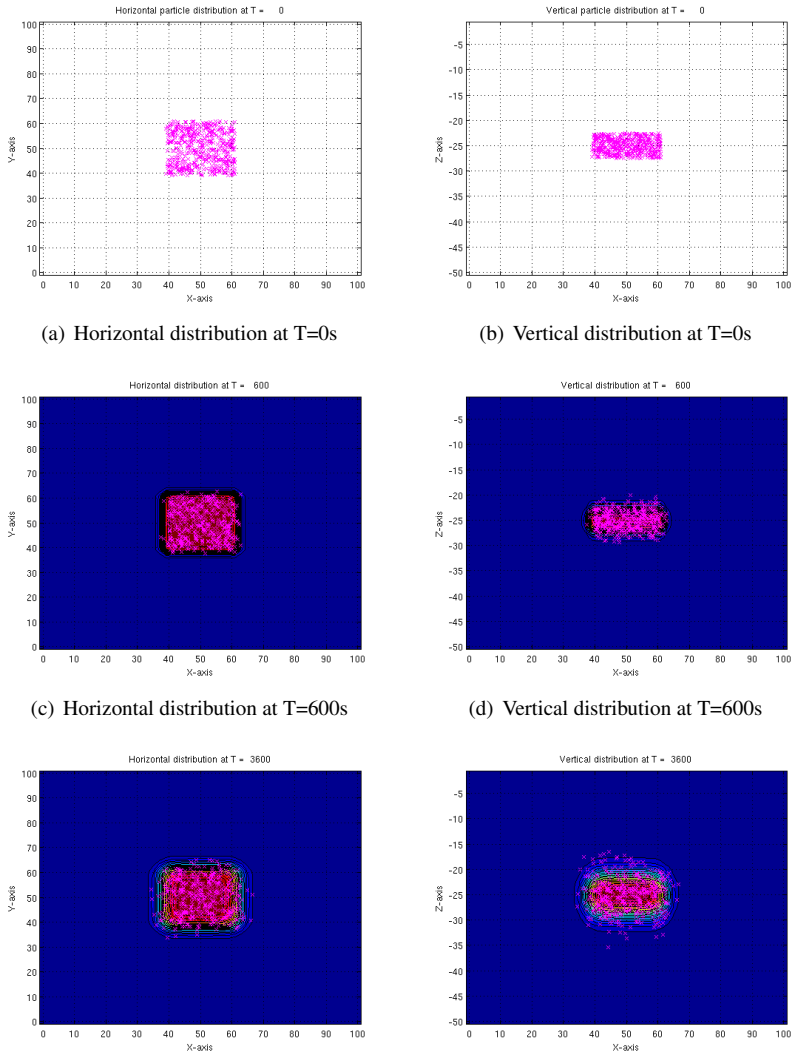
This test was performed using 500 particles with random dispersion, over 1 hour with 1 s time step. The particles and a passive tracer patch are initially located within a box confined by the corner coordinates (39 m, 39 m, -22.5 m) and (61 m, 61 m, -27.5 m). The particles are seeded randomly within this region, while the tracer concentration is uniform within the same region. The horizontal and vertical viscosity and diffusivity coefficients are constant for this test case.

A_M	A_H	$A_M 2D$	K_M	K_H
1.0×10^{-3}	1.0×10^{-3}	1.0×10^{-1}	1.0×10^{-3}	1.0×10^{-3}

Table 4: Constant viscosity and diffusivity coefficients.

Results for the simulation, showing horizontal and vertical distribution of the particles and the tracer at different times, are shown in Fig. 6. The tracer concentration is integrated over all vertical layers for the horizontal plots, and over all layers along the y-axis for the

vertical plots. There are 20 contour levels in the tracer contour plots, meaning that the dark blue area roughly represent a tracer concentration of less than 5 % of the maximum level.



(a) Horizontal distribution at T=0s

(b) Vertical distribution at T=0s

(c) Horizontal distribution at T=600s

(d) Vertical distribution at T=600s

(e) Horizontal distribution at T=3600s

(f) Vertical distribution at T=3600s

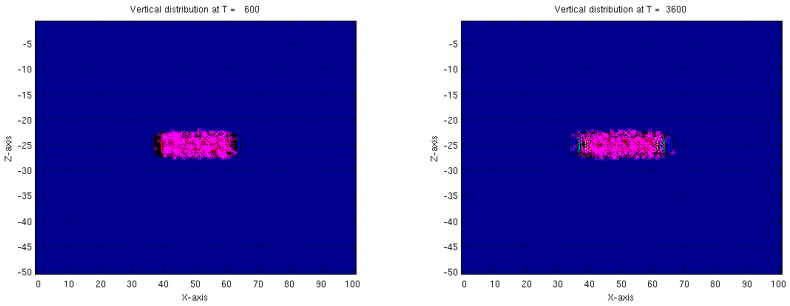
Figure 6: 3D box: Particle dispersion in non-stratified fluid

4.1.3 Test case 1C: Particle dispersion in stratified fluid

This test case is similar to test case 1B, but the fluid is now stratified with a specific temperature and salinity profile, and the vertical viscosity and diffusivity coefficients are calculated using the Mellor-Yamada 2.5 scheme. Figure 7 show the vertical distribution of particles at two different times. The horizontal diffusion process is similar to the results presented in Fig. 6.

4.2 Test case 2: 2D uniform flow over a ridge

The test case is a 2D slice model of homogeneous fluid with two open boundaries, a constant, uniform in-flow through the western boundary, and an out-flow through the eastern



(a) Vertical distribution at T=600s (b) Vertical distribution at T=3600s

Figure 7: 3D box: Particle dispersion in stratified fluid

boundary. A single ridge at the seabed, defined by

$$h(x) = \begin{cases} -h_0 + \dfrac{h_r}{2}\left[1 + \cos\left(\dfrac{(x - x_c)\pi}{r}\right)\right], & -r < x - x_c < r \\ -h_0\,, & \text{elsewhere} \end{cases}$$

is located midway between the in-flow and out-flow boundaries. The main geometrical parameters for the test case are given in Table 5.

IM	JM	KB	dx	h_0	x_c	h_r	r
248	1	101	1.0 m	100.0 m	123.0 m	10.0 m	50.0 m

Table 5: 2D flow-over-bump parameters

The velocity at the in-flow boundary was $U = 0.25\,\mathrm{ms}^{-1}$, and a total of 7 particles were used in the particle tracking model. The particles were seeded initially at $x = 10$ m, and at the vertical locations as listed in Table 6.

Coord.	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
z	-99.5	-97.5	-95.0	-92.5	-90.0	-85.0	-80.0

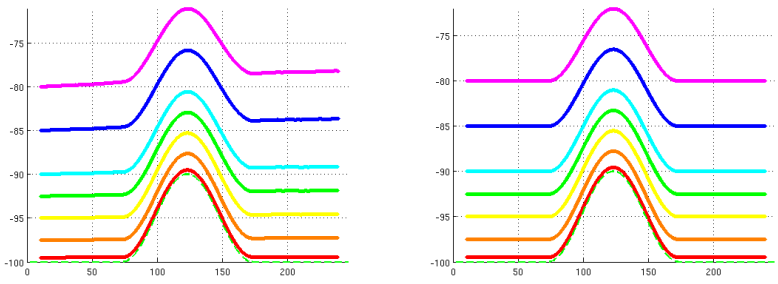
Table 6: Test case 2: particle seeding locations (in meters)

4.2.1 Test case 2A: non-dispersive particle tracking without bottom friction in hydrostatic flow

The first test case was a hydrostatic simulation performed without any bottom friction, in which case we expect the vertical location of the particles to be equal at the in-flow and out-flow boundaries. Test cases were performed with all the parameter options listed in Table 2.

Figure 8 shows a comparison between simulations using single and double precision real values. The particle tracks for the single precision case show a clear upward drift, which becomes stronger further up in the water column, while the double precision simulation show particles at approximately the same vertical level near the in-flow and out-flow boundaries. For this test case, the single precision simulation is clearly not adequate.

Table 7 shows the difference in vertical position between the start and end of the simulations for all particles and all test cases. The difference in vertical drift is clearly visible when comparing (S) single and (D) double precision cases. The use of (bil) bi-linear interpolation as opposed to (lin) linear interpolation slightly improves the results, as can be seen



(a) Single precision (b) Double precision

Figure 8: 2D flow-over-ridge: Particle tracks with single and double precision simulations

Case	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
EF-lin-S	7.88e-2	2.29e-1	4.39e-1	6.53e-1	8.71e-1	1.31	1.77
EF-lin-D	-9.94e-4	-2.39e-3	-3.36e-3	-3.55e-3	-3.22e-3	-3.84e-3	-3.24e-3
EF-bil-S	6.74e-2	2.02e-1	3.91e-1	5.85e-1	7.69e-1	1.16	1.60
EF-bil-D	-9.51e-4	-2.31e-3	-3.31e-3	-3.65e-3	-3.61e-3	-4.28e-3	-2.44e-3
PC-lin-S	7.88e-2	2.29e-1	4.39e-1	6.53e-1	8.70e-1	1.31	1.77
PC-lin-D	-9.94e-4	-2.39e-3	-3.36e-3	-3.55e-3	-3.22e-3	-3.84e-3	-3.23e-3
PC-bil-S	6.74e-2	2.02e-1	3.91e-1	5.85e-1	7.69e-1	1.16	1.60
PC-bil-D	-9.51e-4	-2.31e-3	-3.31e-3	-3.65e-3	-3.61e-3	-4.28e-3	-2.44e-3

Table 7: Test case 2A. Vertical displacement (in meters) of particles from start to end time.

when comparing e.g. results EF-lin-D and EF-bil-D, but the difference in precision is only a few per cent. The use of a higher order (PC) predictor-corrector time stepping scheme as opposed to the (EF) Euler forward method has almost no influence on the results, which in fact are numerically identical to the order of magnitude shown in the table (except for Part. 7 in cases EF-lin-D and PC-lin-D).

4.2.2 Test case 2B: non-dispersive particle tracking with bottom friction

Most practical applications include a bottom friction term which creates a boundary layer near the seabed, with associated enhanced mixing and reduced horizontal velocities. The bottom friction is modeled in BOM with the term

$$C_D = \max\left[0.0025, \frac{\kappa^2}{\left(\ln(z_b/z_0)\right)^2}\right] \tag{4.1}$$

where z_b is the distance to the nearest grid point to the bottom, and the von Karman constant κ is 0.4. In the following simulations we have used the parameter $z_0 = 0.0002$ m, and perform simulations with case parameters EF-lin-D and EF-bil-D.

Case	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
EF-lin-D	2.26e-1	1.28e-1	5.41e-2	6.59e-2	8.52e-2	9.57e-2	9.07e-2
EF-bil-D	2.13e-1	1.17e-1	-4.68e-3	2.99e-2	6.67e-2	9.07e-2	8.59e-2

Table 8: Test case 2B. Vertical displacement (in meters) of particles from start to end time.

The particle tracks shown in Fig. 9 clearly show the effect of reduced horizontal velocity near the seabed, reflected in the reduced distance traveled by the two particles closest to the bottom. What is less apparent from the plot, but evident when comparing start and

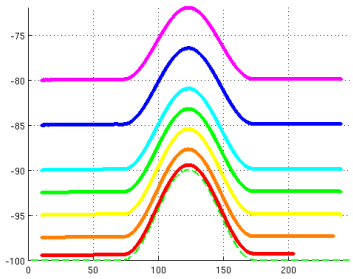


Figure 9: 2D flow-over-bump: Particle tracks with bottom friction.

end vertical position (Table 8), is that the development of a bottom boundary layer cause the two bottom particles to drift upwards by approximately 22 cm and 12 cm, respectively. This is different from the results presented in Table 7, where the particles higher in the water column showed the largest vertical drift. It is not clear from these results if the bi-linear interpolation improves the accuracy of the results, but the differences between linear and bi-linear results are again fairly small.

It is worth mentioning that the the turbulence in the bottom boundary layer is not resolved properly with the resolution used for this test case. The details of the flow close to the bottom are therefore not represented correctly in this test case.

4.2.3 Test case 2C: non-hydrostatic simulation results

The test case results presented in Sections 4.2.1 and 4.2.2 were based on hydrostatic flow simulations. For high resolution simulations, non-hydrostatic effects may be of importance. Simulations with case parameters EF-lin-D and EF-bil-D were performed with and without bottom friction, and the results are summarized in Table 9.

Case	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
without bottom friction							
EF-lin-D	-2.63e-5	9.10e-6	7.99e-5	1.73e-4	2.85e-4	4.97e-4	6.89e-4
EF-bil-D	-1.30e-5	2.98e-5	8.52e-5	1.86e-4	2.96e-4	4.97e-4	7.02e-4
with bottom friction							
EF-lin-D	2.43e-1	1.40e-1	4.50e-2	1.39e-1	2.09e-1	2.69e-1	2.77e-1
EF-bil-D	2.40e-1	1.28e-1	2.64e-2	1.32e-1	2.08e-1	2.71e-1	2.78e-1

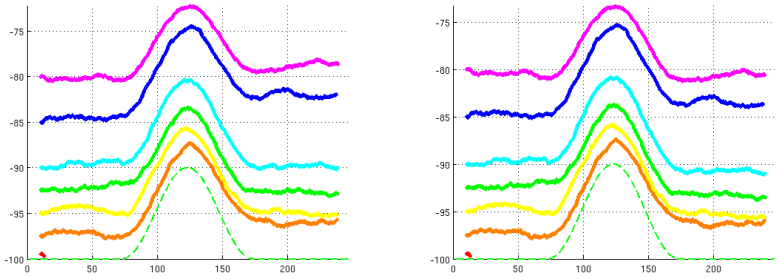
Table 9: Test case 2C. Vertical displacement (in meters) of particles from start to end time.

The results without bottom friction show less deviation between the start and end vertical position than the comparable results for test case 2A in Table 7. There is also greater consistency between the EF-lin-D and EF-bil-D results, except for Part. 2, for the non-hydrostatic simulations with bottom friction than the comparable hydrostatic results in Table 8. These results may indicate some improved performance when non-hydrostatic effects are taken into account, but the improvements are not very large for the case without bottom friction, and difficult to assess for the case with bottom friction.

4.2.4 Test case 2D: Particle tracks with random dispersion

As a last case the particle tracking without bottom friction was simulated with random walk dispersion, corresponding to constant horizontal and vertical diffusivity coefficients of $A_H = K_H = 10^{-3}\mathrm{m}^2\mathrm{s}^{-1}$. Results for single and double precision real numbers are shown in Fig. 10. Note that the sequence of random numbers used in the random dispersion

simulation is identical for the two cases. This case illustrates that the random dispersion will in many cases overshadow a systematic background drift. A spurious drift is likely to be evident as a systematic error when analyzing tracks for a large number of particles, although the source of the error may not be apparent.



(a) Single precision (b) Double precision

Figure 10: Particle tracks with random dispersion

4.3 Test case 3: 3D uniform flow over a mound

This test case is quite similar to test case 2 (Section 4.2), with flow of homogeneous fluid in a channel with a constant, uniform in-flow through the western boundary, out-flow through the eastern boundary, and closed boundaries north and south. A single mound at the seabed, defined by

$$h(x) = -h_0 + \frac{h_r}{4}\left[1 + \cos\left(\frac{(x - x_c)\pi}{r}\right)\right]\left[1 + \cos\left(\frac{(y - y_c)\pi}{r}\right)\right]$$

within the region $(-r < x - x_c < r, -r < y - y_c < r)$, and a constant depth at $-h_0$ elsewhere. The main geometrical parameters for the test case are given in Table 10.

IM	JM	KB	dx	dy	h_0	x_c	x_c	h_r	r
45	30	51	4.0 m	4.0 m	100.0 m	86.0 m	56.0 m	2.5 m	40.0 m

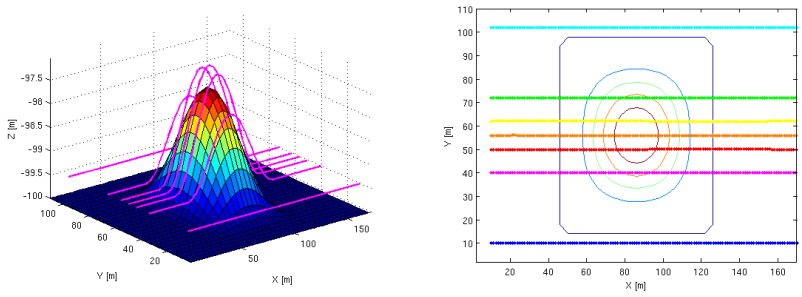
Table 10: 3D flow-over-bump parameters

The velocity at the in-flow boundary was $U = 0.25\,\mathrm{ms}^{-1}$, and a total of 7 particles were used in the particle tracking model. The particles were seeded at $x = 10$ m, $z = -99.5$ m, and in the y direction at the locations as listed in Table 11.

Coord.	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
y	10.0	40.0	50.0	56.0	62.0	72.0	102.0

Table 11: Test case 3: particle seeding locations (in meters)

The tracks of particle pairs (1,7), (2,6), and (3,5) should be symmetrical relative to the center line of the channel at $y = 56.0$ m. Figure 11 shows typical tracks for the 7 particles used in the analysis. Note that boundary cells are not shown for the plots in Fig. 11, which explains the absence of channel side walls in the topography plots. The simulations are run long enough for the particles to transverse the mound in the channel, but terminated before the particles reach the out-flow boundary.



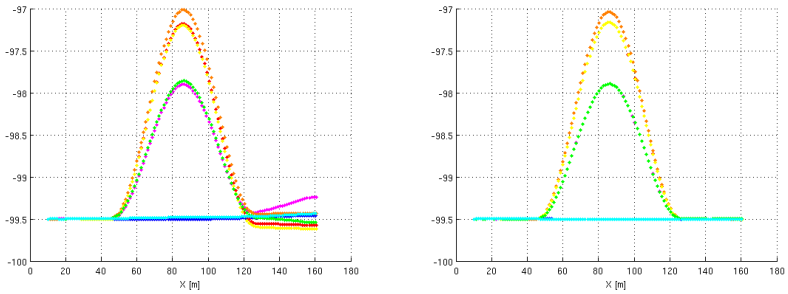
(a) Channel topography and particle tracks (b) Top view

Figure 11: Typical particle tracks for the 7 particles analyzed in test case 3

4.3.1 Test case 3A: non-dispersive particle tracking without bottom friction in hydrostatic flow

In a similar fashion as for test case 2, we repeat the exercise from Section 4.2.1 and compare the influence of real number precision, time stepping and interpolation methods on the vertical displacement of the particles.

A comparison between (S) single and (D) double precision simulations, Fig. 12, demonstrate that the single real precision simulation does not perform adequately with pure drift. All subsequent test cases in this section are therefore performed with (D) double real precision.



(a) Single precision (b) Double precision

Figure 12: 3D flow-over-bump: Particle tracks with single and double precision simulations

Table 12 shows the difference in vertical position between the start and end of the simulations for all particles and all test cases. The single precision simulation shown in Fig. 12(a) is also included. Based on the analysis of the vertical displacement, there does not appear to be any advantage of using (bil) bi-linear as opposed to (lin) linear interpolation. In this case we also see a small difference between the (EF) Euler forward and (PC) predictor-corrector time stepping methods, but this effect is smaller than the effect of interpolation.

Table 13 show relative horizontal displacement between particles that are initially located symmetrically across the channel center line $y = 56.0$ m. The displacement is the difference in the distance between the particles at the starting and ending time. A negative value in the displacement indicate that the particles are closer to each other at the end of the simulation than at the beginning, which may be the case for Δy, but not for Δx since the initial x-coordinates for all particles are identical (see Table 11).

The results in Table 13 show that there is a tendency of negative displacement in Δy (ignoring case EF-lin-S), indicating that particles tend to come closer together in the y-

Case	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
EF-lin-S	4.00e-2	2.59e-1	-7.16e-2	6.67e-2	-1.16e-1	-4.03e-2	6.17e-2
EF-lin-D	-3.39e-4	-1.32e-4	5.71e-6	-1.04e-4	-2.22e-5	3.36e-5	-9.34e-4
EF-bil-D	-8.96e-4	-6.21e-5	-4.54e-5	-1.08e-4	-3.19e-5	-6.01e-5	-8.97e-4
PC-lin-D	-3.40e-4	-1.32e-4	6.46e-6	-1.05e-4	-2.31e-5	3.48e-5	-9.40e-4
PC-bil-D	-8.93e-4	-6.32e-5	-4.62e-5	-1.08e-4	-3.30e-5	-6.13e-5	-8.94e-4

Table 12: Test case 3A. Vertical displacement (in meters) of particles from start to end time.

	Parts. (1,7)		Parts. (2,6)		Parts. (3,5)	
Case	Δx	Δy	Δx	Δy	Δx	Δy
EF-lin-S	1.33e-2	3.50e-2	5.98e-2	4.75e-1	4.03e-2	-5.20e-2
EF-lin-D	5.88e-3	-1.65e-3	4.03e-2	-4.61e-3	2.39e-2	-1.45e-3
EF-bil-D	3.70e-5	-8.58e-4	1.04e-4	-3.12e-4	3.08e-4	-1.34e-3
PC-lin-D	5.92e-3	-1.60e-3	4.03e-2	-4.55e-3	2.38e-2	-1.41e-3
PC-bil-D	3.71e-5	-7.87e-4	1.04e-4	-2.20e-4	3.08e-4	-1.39e-3

Table 13: Test case 3A. Relative horizontal displacement (in meters) between particle pairs at end time of simulation.

direction after having passed the mound. The x-displacement, which should be close to zero in this case, show that there is clear improvement when using (bil) bi-linear interpolation as opposed to (lin) linear interpolation. It is interesting that the effect of using higher order interpolation only becomes apparent when analyzing horizontal displacement of particle pairs (Table 13), but not vertical displacement of individual particles (Table 12). The reason may be that there is a slight drift in the background flow, which would influence the absolute value of the vertical displacement, but not the relative distance between particles .

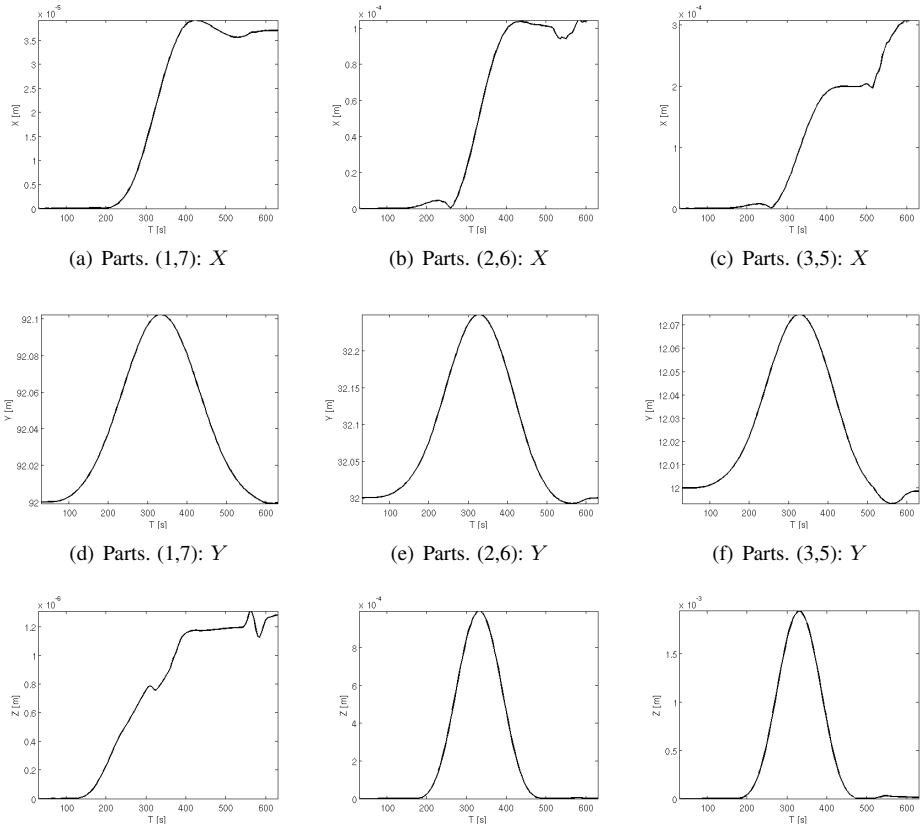
Figure 13 shows the time evolution of particle pair displacements for particle pairs (1,7), (2,6), and (3,5) for the EF-bil-D test run. As can reasonably be expected, the Y displacement (Figs. 13(d)-13(f)) has a maximum corresponding to the mound location, indicating that the particles are displaced towards the channel walls when encountering the mound, move towards the channel center line downstream of the mound. There is a similar characteristic feature also in two of the Z displacement plots (Figs. 13(h) and 13(i)), which should probably not be present in a completely symmetrical problem formulation. It can be assumed that these features are due either to inaccuracies related to the interpolation method, or inaccuracies in the bathymetry data.

4.3.2 Test case 3B: non-dispersive particle tracking with bottom friction

A simulation with bottom friction was performed with simulation parameters EF-bil-D. Figure 14 show a vertical profile of the particle tracks, indicating the upward drift due to the gradual thickening of the bottom boundary layer. Table 14 show the horizontal and vertical displacement of individual particles. Both the upward drift and the along channel velocity is slightly stronger near the center line of the channel than at the channel walls, and the motion is symmetrical in the cross-channel direction. It is not clear, however, that the difference between the horizontal velocity near the channel center line and channel wall can be attributed to the presence of bottom friction (see Section 4.3.3).

Coord.	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
X	140.46	140.57	140.64	140.65	140.64	140.57	140.46
Z	1.81e-1	1.88e-1	1.91e-1	1.91e-1	1.91e-1	1.88e-1	1.81e-1

Table 14: Test case 3B. Horizontal and vertical displacement (in meters) of particles from start to end time.



(a) Parts. (1,7): X (b) Parts. (2,6): X (c) Parts. (3,5): X

(d) Parts. (1,7): Y (e) Parts. (2,6): Y (f) Parts. (3,5): Y

(g) Parts. (1,7): Z (h) Parts. (2,6): Z (i) Parts. (3,5): Z

Figure 13: Test case 3A, case parameters EF-bil-D: Particle pair displacement

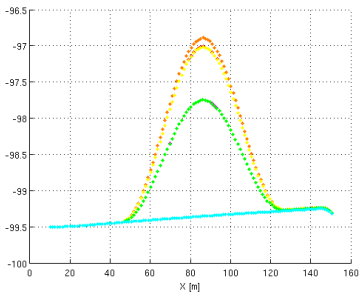


Figure 14: Test case 3B: Particle tracks with bottom friction.

Table 15 show the particle pair displacement in X and Y direction, for particle pairs (1,7), (2,6), and (3,5). When comparing to the results in the previous section (Table 13), we see that the inclusion of bottom friction has little effect on the along-channel displacement X, but increases the tendency in the Y displacement of moving particles toward the center line of the channel.

Parts. (1,7)		Parts. (2,6)		Parts. (3,5)	
Δx	Δy	Δx	Δy	Δx	Δy
3.97e-5	-1.75e-2	1.11e-4	-5.11e-2	1.94e-4	-3.36e-2

Table 15: Test case 3B. Relative horizontal displacement (in meters) between particle pairs at end time of simulation.

4.3.3 Test case 3C: non-hydrostatic simulation results

Coord.	Part. 1	Part. 2	Part. 3	Part. 4	Part. 5	Part. 6	Part. 7
X	150.26	150.45	150.53	150.54	150.53	150.45	150.26
Z	8.13e-6	-9.50e-6	5.50e-6	6.19e-6	6.74e-6	3.60e-6	-7.01e-6

Table 16: Test case 3C. Horizontal and vertical displacement (in meters) of particles from start to end time.

Parts. (1,7)		Parts. (2,6)		Parts. (3,5)	
Δx	Δy	Δx	Δy	Δx	Δy
9.30e-5	-1.15e-3	1.74e-4	-4.04e-4	2.04e-4	-1.42e-3

Table 17: Test case 3C. Relative horizontal displacement (in meters) between particle pairs at end time of simulation.

Results for a non-hydrostatic simulation without bottom friction are summarized in Tables 16 and 17. The vertical displacement for individual particles are usually one or two orders of magnitude less than the comparable hydrostatic simulation results (comparing Tables 12 and 16). The horizontal displacement result from Table 16 also indicate a higher along-channel velocity along the center of the channel than near the channel wall. This effect is therefore most likely due to higher horizontal velocity over the top of the mound compared to the surrounding area. The particle pair results in Table 17 are similar to the results presented in Table 13 for the hydrostatic case.

4.3.4 Test case 3D: Tracer concentration and particle distribution with Smagorinsky and Mellor-Yamada 2.5

The test case compares the transport of a passive tracer field with transport of particles. The simulation is hydrostatic, without bottom friction. The passive tracer and particles are initially located within the same region; a box defined by $30.0\text{m} \leq x \leq 50.0\text{m}; 34.0\text{m} \leq y \leq 78.0\text{m}; -99.5\text{m} \leq z \leq -95.0\text{m}$. A total of 20,000 particles were used for this simulation. Horizontal diffusion coefficients are calculated using Smagorinsky, and vertical diffusion coefficients are calculated using the Mellor-Yamada 2.5 method.

A particle concentration has been calculated based on particle distribution by taking the sum of all particles within each 3D grid cell divided by the volume of the cell, and dividing by the maximum concentration in a single cell.

$$C_{PART}(i,j,k) = \frac{\frac{\sum \text{particles in cell}(i,j,k)}{\text{volume of cell}(i,j,k)}}{\max\{C_{PART}(i,j,k)\}}$$

The resulting concentration is somewhat artificial, since the particle concentration will always be 1 (i.e. fully saturated) for at least one cell, but it is sufficient for the comparison of particle dispersion vs. tracer diffusion, which is the purpose of this test case.

The figure panels in Fig. 15 show a comparison of the tracer field and particle patch at times $T = 0$s, 60s, 180s, and 420s. Panels 15(a), 15(c), 15(f), and 15(i) show an iso-surface plot for the iso-surface of 5% particle concentration, i.e. 95% of the particles are contained within the red volume. A similar sequence of plots of the 5% iso-surface for the tracer field are not shown here, since the development of the tracer and particle patches are quite similar. The remaining figure panels show differences

$$C_{DIFF} = C_{TRACER} - C_{PART}$$

between the tracer and particle concentrations; Figs. 15(b), 15(e), and 15(h) show vertically averaged horizontal concentration differences, and Figs. 15(d), 15(g), and 15(j) show vertical concentration differences averaged over the y-axis.

These figures illustrate that the particle and tracer patches mostly occupy the same region throughout the simulation time. However, the tracer concentration show signs of increased horizontal diffusion when compared to the horizontal particle dispersion. It is generally the case that it is difficult to maintain large gradients in a tracer concentration field. This is not an issue for passive particles, as long as the particles are essentially independent.

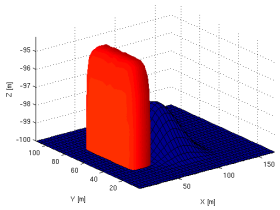
5 Topics for further research

5.1 Parallelization algorithm for particle tracking

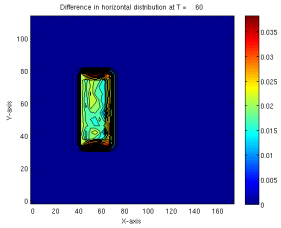
The parallelization method described in Section 3 is efficient if the particles are uniformly distributed. An alternative method would be to distribute the particles among the processors in such a way that each processor was responsible for approximately the same number of particles. This is probably a more efficient method if a large number of particles are concentrated within a small sub-domain in the model area. The drawback of this method is that the processors would need to exchange velocity data for regions surrounding the particles, so ensuring an equal load balance among the processors may increase the cost in terms of data exchange. This method has not been implemented in the PTM, but it may be a topic for further study.

5.2 Time stepping for the particle tracking model

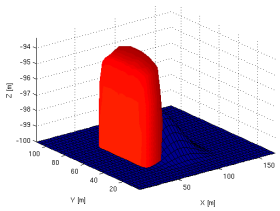
The current version of the PTM updates the position of the particles at each 3D time step with BOM. This ensures that the particles are updated at the same time intervals as the salinity, temperature, and passive tracer fields. However, it is not clear that this is the optimal choice for all cases. If a simulation is dominated by strong tidal forcing, it may be more appropriate to update the particles according to the barotropic mode instead of the baroclinic mode. It may also be beneficial to separate the particle advection and particle dispersion parts, since the advection process typically depends on smaller time scales than dispersion.



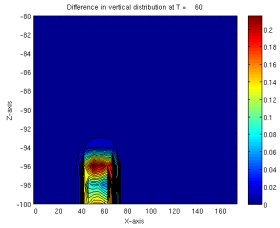
(a) Part. conc.: $T = 0$ s



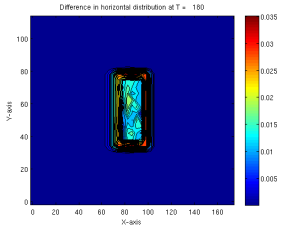
(b) Hor. diff.: $T = 60$ s



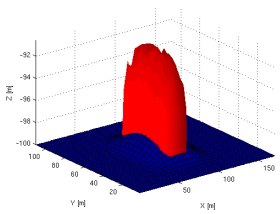
(c) Part. conc.: $T = 60$ s



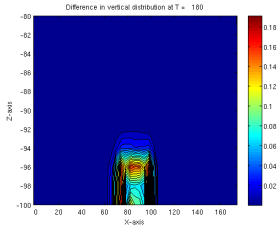
(d) Vert. diff.: $T = 60$ s



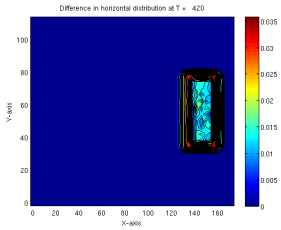
(e) Hor. diff.: $T = 180$ s



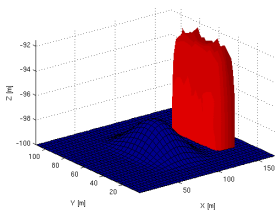
(f) Part. conc.: $T = 180$ s



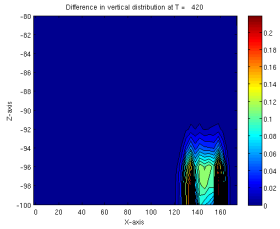
(g) Vert. diff.: $T = 180$ s



(h) Hor. diff.: $T = 420$ s



(i) Part. conc.: $T = 420$ s



(j) Vert. diff.: $T = 420$ s

Figure 15: Iso-surface plot for the 5% level of particle concentration and tracer concentration

Appendices

A Review of probability theory

This presentation closely follows the presentation in the book by Taylor and Karlin [1998]. Similar material should be available from any introductory text on probability theory.

A.1 Random variables

A *random variable* can loosely be defined as a variable that takes on its value by chance. The expression $\{X \leq x\}$ is the event that the random variable X assumes a value that is less or equal to the real number x, and the probability that this event occurs is denoted as $\Pr\{X \leq x\}$. By allowing x to vary over the range of all real numbers, we can define the *distribution function*

$$F(x) = \Pr\{X \leq x\}\,, \qquad -\infty < x < \infty\,, \tag{A.1}$$

of the random variable X. The random variable X is defined to be *continuous* if

$$\Pr\{X = x\} = 0$$

for every value of x, and in this case the distribution function $F(x)$ is a continuous function of x. If $F(x)$ is differentiable in x, then X has a *probability density function* (PDF) given by

$$f(x) = \frac{d}{dx}F(x) = F'(x)\,, \qquad -\infty < x < \infty\,. \tag{A.2}$$

A.2 Moments, expected values, and variance

If X is a continuous random variable with a probability density function $f(x)$, then its *m*th moment is given by

$$E[X^m] = \int_{-\infty}^{\infty} x^m f(x)\; dx\,, \tag{A.3}$$

provided that this integral converges absolutely. The *first moment* μ_X, given by

$$\mu_X = \int_{-\infty}^{\infty} x f(x)\; dx\,, \tag{A.4}$$

is commonly called the *mean* or *expected value* of X. The *m*th *central moment* of X is defined as the *m*th moment of the random variable $X - \mu_X$, provided that μ_X exists. The first central moment is zero, since

$$E[X - \mu_X] = E[X] - E[\mu_X] = \mu_X - \mu_X = 0\,.$$

The second central moment σ_X^2 is called the *variance* of X, and is given by

$$\text{Var}\,[X] \equiv \sigma_X^2 = E[(X - \mu_X)^2] = E[X^2] - 2\mu_X E[X] + \mu_X^2 = E[X^2] - \mu_X^2\,. \tag{A.5}$$

A.3 Joint distribution functions

Given a pair (X, Y) of random variables, their *joint distribution function* F_{XY} is a function of two real variables given by

$$F_{XY}(x, y) = \Pr\{(X \leq x) \cap (Y \leq y)\}\,, \qquad -\infty < x < \infty\,; -\infty < y < \infty\,. \tag{A.6}$$

The joint distribution function F_{XY} is said to possess a probability density function if there exist a function f_{XY} such that

$$F_{XY}(x,y) = \int_{-\infty}^{x}\int_{-\infty}^{y} f_{XY}(\xi,\eta)\; d\eta\; d\xi\,, \qquad \text{for all } x,\; y.$$

The *marginal distribution function* of X and of Y are defined as

$$F_X = \lim_{y\to\infty} F_{XY}(x,y) \qquad \text{and} \qquad F_Y = \lim_{x\to\infty} F_{XY}(x,y)\,, \tag{A.7}$$

respectively. If the joint distribution function possesses the joint density function f_{XY}, the marginal density functions for X and Y, are given, respectively, by

$$f_X(x) = \int_{-\infty}^{\infty} f_{XY}(x,y)\; dy \qquad \text{and} \qquad f_Y(y) = \int_{-\infty}^{\infty} f_{XY}(x,y)\; dx\,.$$

If X and Y are jointly distributed, then

$$\begin{aligned} E[X+Y] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x f_{XY}(x,y) + y f_{XY}(x,y)\; dy\; dx \\ &= \int_{-\infty}^{\infty} x f_X(x)\; dx + \int_{-\infty}^{\infty} y f_Y(y)\; dy \\ &= E[X] + E[Y]\,, \end{aligned}$$

provided all these moments exist.

The random variables X and Y are said to be *independent* if

$$F_{XY}(x,y) = F_X(x)F_Y(y) \qquad \text{for all choices of } x, y. \tag{A.8}$$

If X and Y are independent and possesses a joint density function $f_{XY}(x,y)$, then it follows that

$$f_{XY}(x,y) = f_X(x)f_Y(y) \qquad \text{for all choices of } x, y.$$

Given jointly distributed random variables X and Y with means μ_X and μ_Y and finite variances, the *covariance* of X and Y, written σ_{XY} or $\mathrm{Cov}[X,Y]$, is the product moment

$$\sigma_{XY} = E[(X-\mu_X)(Y-\mu_Y)] = E[XY] - \mu_X\mu_Y\,, \tag{A.9}$$

and X and Y are said to be *uncorrelated* if their covariance is zero. Independent random variables having finite variances are uncorrelated. However, uncorrelated random variables are not necessarily independent. The *correlation coefficient* is defined by

$$\rho = \frac{\sigma_{XY}}{\sigma_X\sigma_Y}\,, \tag{A.10}$$

which is defined on the range $-1 \le \rho \le 1$.

A.4 The normal distribution

The *normal distribution* with mean μ and variance $\sigma^2 > 0$ is defined by the probability density function

$$\phi(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}\,, \qquad -\infty < x < \infty\,. \tag{A.11}$$

In the case with $\mu = 0$ and $\sigma^2 = 1$, eq. (A.11) is called the *standard normal distribution*. If X is normally distributed with mean μ and variance σ^2, then $Z = (X-\mu)/\sigma$ has a standard normal distribution. The standard normal probability density function and distribution function are given respectively by

$$\phi(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\,, \qquad -\infty < x < \infty\,, \tag{A.12}$$

and

$$\Phi(x) = \int_{-\infty}^{x} \phi(\xi)\, d\xi\,, \qquad -\infty < x < \infty\,. \tag{A.13}$$

A.4.1 The central limit theorem

Given the sum $S_n = \xi_1 + \xi_2 + \cdots + \xi_n$ of *independent* and *identically distributed* random variables $\xi_1, \xi_2, \ldots$, having finite means $\mu = E[\xi_k]$ and finite variances $\sigma^2 = E[\xi_k^2] - \mu^2$. The *central limit theorem* asserts that

$$\lim_{n\to\infty} \Pr\left\{ \frac{S_n - n\mu}{\sigma\sqrt{n}} \le x \right\} = \Phi(x) \qquad \text{for all } x. \tag{A.14}$$

In other words, for large values of n, the sum S_n of independent, identically distributed random variables is approximately normally distributed with mean $n\mu$ and variance $n\sigma^2$.

A.5 Conditional probability

The conditional probability $\Pr\{A|B\}$ of an event A given the event B is defined by

$$\Pr\{A|B\} = \frac{\Pr\{A \cap B\}}{\Pr\{B\}} \qquad \text{if } \Pr\{B\} > 0, \tag{A.15}$$

and is not defined, or is assigned an arbitrary value, if $\Pr\{B\} = 0$. Assuming the random variables X and Y are discrete, the *conditional probability mass function* $p_{X|Y}(x|y)$ of X given $Y = y$ is defined by

$$p_{X|Y}(x, y) = \frac{\Pr\{(X = x) \cap (Y = y)\}}{\Pr\{Y = y\}} \qquad \text{if } \Pr\{Y = y\} > 0, \tag{A.16}$$

and is not defined, or is assigned an arbitrary value, if $\Pr\{Y = y\} = 0$. In terms of the joint and marginal probability mass functions

$$p_{XY}(x, y) \qquad \text{and} \qquad p_Y(y) = \sum_x p_{XY}(x, y),$$

respectively, the definition is

$$p_{X|Y}(x, y) = \frac{p_{XY}(x, y)}{p_Y(y)} \qquad \text{if } p_Y(y) > 0. \tag{A.17}$$

A.5.1 Conditioning on a continuous random variable

Assume X and Y are jointly distributed continuous random variables with a joint probability density function $f_{XY}(x, y)$. The *conditional probability density function* $f_{X|Y}(x|y)$ of X given $Y = y$ is defined by

$$f_{X|Y}(x, y) = \frac{f_{XY}(x, y)}{f_Y(y)} \qquad \text{if } f_Y(y) > 0, \tag{A.18}$$

and the conditional density is not defined at values of y where $f_Y(y) = 0$. The definition given in eq. (A.18) extends the original definition eq. (A.15) because it allows the calculation of a conditional probability density function conditioning on events having zero probability. Given a function g for which $E[|g(X)|] < \infty$, the *conditional expectation* of $g(X)$ given $Y = y$ is defined to be

$$E[g(X)|Y = y] = \int g(x) f_{X|Y}(x, y)\, dx \qquad \text{if } f_Y(y) > 0. \tag{A.19}$$

B Stochastic processes and modeling advection-diffusion

This section aims to give a short introduction to stochastic processes, and Markov processes in particular, with specific emphasis on subjects relevant for modeling the advection-diffusion process. The presentation is based mostly on background material from Gardiner [1985]. Other useful text books on this subject include: Taylor and Karlin [1998]; Pope [2000]; Bennett [2006].

B.1 Stochastic processes

A *stochastic process* can loosely be described as a system which evolves probabilistically in time, i.e. a system in which a certain time-dependent random variable $\mathbf{X}(t)$ exist. The values $\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots$ of $\mathbf{X}(t)$ can be measured at times $t_1, t_2, t_3, \ldots$, and we assume that the system is completely described by a set of joint probability densities

$$f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2; \mathbf{x}_3, t_3; \ldots) .$$

Based on these joint probability density functions, it is possible also to define conditional probability density functions

$$f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2; \ldots | \mathbf{y}_1, s_1; \mathbf{y}_2, s_2; \ldots) = \frac{f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2; \ldots; \mathbf{y}_1, s_1; \mathbf{y}_2, s_2; \ldots)}{f(\mathbf{y}_1, s_1; \mathbf{y}_2, s_2; \ldots)} . \quad \text{(B.1)}$$

When we study the evolution of the system it is natural to consider conditional probabilities as predictions of future values of $\mathbf{X}(t)$ (i.e. $\mathbf{x}_1, \mathbf{x}_2, \ldots$ at times $t_1, t_2, \ldots$) given the knowledge of past states (the values $\mathbf{y}_1, \mathbf{y}_2, \ldots$ at times $s_1, s_2, \ldots$). For the conditional probability density function (B.1), we assume that the times are ordered by

$$t_1 \geq t_2 \geq \ldots \geq s_1 \geq s_2 \geq \ldots$$

following the convention of Gardiner [1985] to order terms with increasing time from right to left.

The most simple stochastic process is one where the probability densities are completely independent

$$f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2; \mathbf{x}_3, t_3; \ldots) = \prod_i f(\mathbf{x}_i, t_i)$$

which means that the value of $\mathbf{X}$ at any given time t is completely independent of all past and future states of the system. The special process where also $f(\mathbf{x}_i, t_i)$ is independent of t_i, i.e. the same probability law governs the process at all times, is called *Bernoulli trials*, in which case the same probabilistic process is repeated at successive times.

B.2 Markov processes

If we assume that successive states of the system are not completely independent, the *Markov process* represent the simples possible model. A *Markov process* $\{X_t\}$ is a stochastic process with the property that, given the value of X_t, the values of X_u for $u > t$ are not influenced by the values of X_s for $s < t$. In other words, if the state of the Markov process is known at a time t, then the probability of any particular future behavior depends only on the current state, and is not altered by past behavior.

B.2.1 Markov chains

A *discrete-time Markov chain* is a Markov process whose state space is a finite and countable set, and whose (time) index set is $T = (0, 1, 2, \ldots)$. The Markov property is that

$$\Pr\{X_0 = i | X_1 = j, \ldots, X_{n-1} = j_{n-1}, X_n = j_n\} = \Pr\{X_0 = i | X_1 = j\} \quad \text{(B.2)}$$

for all time points n and for all states $i, j, j_2, \ldots, j_{n-1}, j_n$. The probability of X_0 being in state i given that X_1 is is state j is called the *one-step transition probability*, and is denoted by

$$P_{ij}^{0,1} = \Pr\{X_0 = i | X_1 = j\}. \tag{B.3}$$

In general the transition probabilities are functions of the time n of the transition, as well as the initial and final states. However, in many processes the transitional probabilities are independent of time, and in this case we say that the Markov chain has *stationary transition probabilities*, in which case we can simplify the notation because $P_{ij}^{n-1,n} = P_{ij}$ for all n. A Markov process is completely defined once the initial state X_n and all transitional probabilities have been specified. Assuming $\Pr\{X_n = i\} = p_i$, we can apply eqs. (A.17), (B.2), and (B.3) in successive steps to write the probability of a specific sequence of events as

$$\begin{aligned}
&\Pr\{X_0 = i_0, X_i = i_1, , X_2 = i_2, \ldots, X_n = i\} \\
&\quad = \Pr\{X_i = i_1, , X_2 = i_2, \ldots, X_n = i\} \times \\
&\qquad\quad \Pr\{X_0 = i_0 | X_1 = i_1, X_2 = i_2, \ldots, X_n = i\} \\
&\quad = \Pr\{X_2 = i_2, \ldots, X_n = i\} \times \\
&\qquad\quad \Pr\{X_1 = i_1 | X_2 = i_2, \ldots, X_n = i\} \times \\
&\qquad\quad \Pr\{X_0 = i_0 | X_1 = i_1\} \\
&\quad \ldots \\
&\quad = P_{i_0 i_1} P_{i_1 i_2} \cdots P_{i_{n-1} i_n} p_i.
\end{aligned}$$

B.3 The Chapman-Kolmogorov equation

From the discussion on joint distribution functions (Section A.3) combined with the definition of the conditional probability density function (A.18), we get the general result

$$f_X(x) = \int_{-\infty}^{\infty} f_{XY}(x, y)\, dy = \int_{-\infty}^{\infty} f_{X|Y}(x, y) f_Y(y)\, dy. \tag{B.4}$$

Applying eq. (B.4) for a stochastic process we find

$$f(\mathbf{x}_1, t_1) = \int_{-\infty}^{\infty} f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2)\, d\mathbf{x}_2 = \int_{-\infty}^{\infty} f(\mathbf{x}_1, t_1 | \mathbf{x}_2, t_2) f(\mathbf{x}_2, t_2)\, d\mathbf{x}_2. \tag{B.5}$$

From eq. (B.5) we can construct the conditional PDF for $\mathbf{x}_1, t_1$ with respect to a third event $\mathbf{x}_3, t_3$

$$\begin{aligned}
f(\mathbf{x}_1, t_1 | \mathbf{x}_3, t_3) &= \int_{-\infty}^{\infty} f(\mathbf{x}_1, t_1; \mathbf{x}_2, t_2 | \mathbf{x}_3, t_3)\, d\mathbf{x}_2 \\
&= \int_{-\infty}^{\infty} f(\mathbf{x}_1, t_1 | \mathbf{x}_2, t_2; \mathbf{x}_3, t_3) f(\mathbf{x}_2, t_2 | \mathbf{x}_3, t_3)\, d\mathbf{x}_2,
\end{aligned} \tag{B.6}$$

which is a general identity valid for all stochastic processes. If we assume the stochastic process is a Markov process, with times $t_1 \geq t_2 \geq t_3$, we can drop the t_3 dependence in the doubly conditioned probability, in which case we get the *Chapman-Kolmogorov equation*

$$f(\mathbf{x}_1, t_1 | \mathbf{x}_3, t_3) = \int_{-\infty}^{\infty} f(\mathbf{x}_1, t_1 | \mathbf{x}_2, t_2) f(\mathbf{x}_2, t_2 | \mathbf{x}_3, t_3)\, d\mathbf{x}_2. \tag{B.7}$$

B.4 The differential Chapman-Kolmogorov equation

According to Gardiner [1985], it can be shown that the sample paths for a Markov process are continuous functions of t, if for any $\epsilon > 0$ we have

$$\lim_{\Delta t \to 0} \frac{1}{\Delta t} \int_{|\mathbf{x}-\mathbf{z}|>\epsilon} f(\mathbf{x}, t + \Delta t | \mathbf{z}, t)\, d\mathbf{x} = 0 \tag{B.8}$$

uniformly in $\mathbf{z}$, t and Δt. This means that the probability of finding the end position $\mathbf{x}$ to be finitely different from the start position $\mathbf{z}$ goes to zero *faster* than Δt, as Δt goes to zero.

Under certain conditions, which are closely related to the continuity condition (B.8), a differential for of the Chapman-Kolmogorov equation can be derived. The following conditions should be satisfied for all $\epsilon > 0$:

$$(i) \lim_{\Delta t \to 0} \frac{1}{\Delta t} f(\mathbf{x}, t + \Delta t | \mathbf{z}, t) = Q(\mathbf{x}|\mathbf{z}, t) \quad \text{uniformly in } \mathbf{x}, \mathbf{z}, t \text{ for } |\mathbf{x} - \mathbf{z}| \geq \epsilon \tag{B.9}$$

$$(ii) \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int\limits_{|\mathbf{x}-\mathbf{z}|<\epsilon} (x_i - z_i) f(\mathbf{x}, t + \Delta t | \mathbf{z}, t)\, d\mathbf{x} = A_i(\mathbf{z}, t) + \mathcal{O}(\epsilon) \tag{B.10}$$

$$(iii) \lim_{\Delta t \to 0} \frac{1}{\Delta t} \int\limits_{|\mathbf{x}-\mathbf{z}|<\epsilon} (x_i - z_i)(x_j - z_j) f(\mathbf{x}, t + \Delta t | \mathbf{z}, t)\, d\mathbf{x} = B_{ij}(\mathbf{z}, t) + \mathcal{O}(\epsilon) \tag{B.11}$$

where conditions (ii) and (iii) should hold uniformly in $\mathbf{z}$, ϵ and t. It can be shown (see Gardiner [1985]) that all higher-order terms of the type (B.10) and (B.11) must vanish, i.e. the right hand side must be at most $\mathcal{O}(\epsilon)$. A derivation of the *differential Chapman-Kolmogorov equation* from eq. (B.7) is presented by Gardiner [1985], and the end result is

$$\begin{aligned} \frac{\partial}{\partial t} f(\mathbf{z}, t|\mathbf{y}, t') = &- \sum_i \frac{\partial}{\partial z_i} \left[A_i(\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] \\ &+ \sum_{i,j} \frac{1}{2} \frac{\partial^2}{\partial z_i \partial z_j} \left[B_{ij}(\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] \\ &+ \int \left[Q(\mathbf{z}|\mathbf{x}, t) f(\mathbf{x}, t|\mathbf{y}, t') - Q(\mathbf{x}|\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] d\mathbf{x} \end{aligned} \tag{B.12}$$

The different conditions (B.9) - (B.11), and the corresponding terms in eq. (B.12) can be related to different processes. Condition (B.9) is associated with a *jump process*, condition (B.10) is associated with a *drift process*, and (B.11) is associated with a *diffusion process*.

B.5 The Fokker-Planck equation

If the sample paths for a Markov process is continuous, i.e that condition (B.8) is satisfied, it follows that we must have $Q(\mathbf{x}|\mathbf{z}, t) \equiv 0$, and the differential Chapman-Kolmogorov equation (B.12) is reduced to

$$\begin{aligned} \frac{\partial}{\partial t} f(\mathbf{z}, t|\mathbf{y}, t') = &- \sum_i \frac{\partial}{\partial z_i} \left[A_i(\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] \\ &+ \sum_{i,j} \frac{1}{2} \frac{\partial^2}{\partial z_i \partial z_j} \left[B_{ij}(\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] \end{aligned} \tag{B.13}$$

which is called the *Fokker-Planck equation* (eq. (B.13) is also called the *forward Kolmogorov equation* or the *Smoluchowski equation*).

B.5.1 Deterministic processes and Liouville's equation

In the case that $B_{ij} \equiv 0$, the Fokker-Planck equation is reduced to the *Liouville equation*

$$\frac{\partial}{\partial t} f(\mathbf{z}, t|\mathbf{y}, t') = - \sum_i \frac{\partial}{\partial z_i} \left[A_i(\mathbf{z}, t) f(\mathbf{z}, t|\mathbf{y}, t') \right] \tag{B.14}$$

which describes a completely deterministic motion. If $\mathbf{x}(t)$ is the solution of the initial value problem

$$\frac{\partial \mathbf{x}(t)}{dt} = A(\mathbf{x}(t), t) \qquad \text{with } \mathbf{x}(t') = \mathbf{y} \tag{B.15}$$

then the solution of (B.14) with initial condition

$$f(\mathbf{z}, t'|\mathbf{y}, t') = \delta(\mathbf{z} - \mathbf{y})$$

is

$$f(\mathbf{z}, t|\mathbf{y}, t') = \delta(\mathbf{z} - \mathbf{x}(t)) \,. \tag{B.16}$$

This can easily be verified by direct substitution in eq. (B.13), which gives, for the left hand side

$$\frac{\partial}{\partial t}\delta(\mathbf{z} - \mathbf{x}(t)) = -\sum_i \frac{\partial}{\partial z_i}\delta(\mathbf{z} - \mathbf{x}(t))\frac{d\mathbf{x}(t)}{dt}$$

and for the right hand side

$$\begin{aligned} -\sum_i \frac{\partial}{\partial z_i}\left[A_i(\mathbf{z}, t)\delta(\mathbf{z} - \mathbf{x}(t))\right] &= -\sum_i \frac{\partial}{\partial z_i}\left[A_i(\mathbf{x}(t), t)\delta(\mathbf{z} - \mathbf{x}(t))\right] \\ &= -\sum_i A_i(\mathbf{x}(t), t)\left[\frac{\partial}{\partial z_i}\delta(\mathbf{z} - \mathbf{x}(t))\right] \end{aligned}$$

which are equal under the condition given by eq. (B.15). Therefore, if the particle is in the position $\mathbf{y}$ at time t', it remains on the trajectory obtained by solving the initial value problem (B.15).

B.5.2 The Wiener process - Brownian motion

In 1827 the English botanist Robert Brown observed that microscopic pollen grains suspended in a drop of water moved constantly in haphazard zigzag trajectories. Similar *Brownian motion* phenomena are apparent whenever very small particles are suspended in a fluid medium, such as smoke particles in the air. The mathematical foundation for describing Brownian motion was developed by Norbert Wiener in 1923, and the process is also frequently called the *Wiener process* or *Wiener-Einstein process*.

A multivariate Wiener process can be defined as a continuous-time, continuous state-space Markov process

$$\mathbf{W}(t) = [W_1(t), W_2(t), \ldots, W_n(t)]$$

with zero drift and diffusion coefficient 1. The process therefore satisfies the multivariate Fokker-Planck equation

$$\frac{\partial}{\partial t}f(\mathbf{w}, t|\mathbf{w}_0, t_0) = \sum_i \frac{1}{2}\frac{\partial^2}{\partial w_i^2}f(\mathbf{w}, t|\mathbf{w}_0, t_0) \,, \tag{B.17}$$

with the initial condition

$$f(\mathbf{w}, t|\mathbf{w}_0, t_0) = \delta(\mathbf{w} - \mathbf{w}_0) \,. \tag{B.18}$$

The general solution of eq. (B.17) is

$$f(\mathbf{w}, t|\mathbf{w}_0, t_0) = [2\pi(t - t_0)]^{-n/2} \exp\left[\frac{(\mathbf{w} - \mathbf{w}_0)^2}{2(t - t_0)}\right] \quad (t > t_0) \tag{B.19}$$

which correspond to the Gaussian distribution with mean

$$E[\mathbf{W}(t)] = \mathbf{w}_0 \,, \tag{B.20}$$

and variance

$$\text{Var}\,[\mathbf{W}(t)] = E[(W_i(t) - w_{0i})(W_j(t) - w_{0j})] = (t - t_0)\delta_{ij} \,. \tag{B.21}$$

C Short user guide for the particle tracking model

The entire code for the particle tracking model (PTM) is contained within the single module file *mod_particle.f90*. The PTM imports the BOM structure and main variables through the *state* module, which then is available for all subroutines and functions in PTM. In addition, the setup part of PTM use the *setup2* and *util* modules.

At present, the PTM is made available in BOM by modifying the two subroutines *user_init* and *user_step*. Users also need to specify model parameters in a file called *particle_setup.dat*, and provide the initial data for the particles in a separate ascii or binary file.

Example of a *user_init.f90* file with particle tracking:

```
subroutine user_init()
!! Empty shell routine meant as a placeholder for any extra
!! initializations not provided by the init system
  USE MOD_PARTICLE
!
! Initialize particle data
!
  call setup_particle

end subroutine user_init
```

Example of a *user_step.f90* file with particle tracking:

```
subroutine user_step()
!! called after computation of u,v,s,t etc are complete in
!! a time step, users may add calls to submodels here.
  USE MOD_PARTICLE
!
! Move particle through advection and diffusion
!
  call move_particle

end subroutine user_step
```

C.1 The *particle_setup.dat* file

The *particle_setup.dat* file specifies the PTM parameters using a <keyword>:<value> system adopted from *setup2* in BOM. A small *particle_setup.dat* file can be specified as follows

```
! Setup file for particles
part_activate      t
part_random_step   f
part_init_data     2   fort.33   20000
part_bound_east    3   1.0
part_bound_west    3   1.0
part_capture_h     t
part_capture_eta   t
part_capture_land  t

partplot_timeseries      t
partplot_ts_write      600
```

The rules for *particle_setup.dat* are the same as for the *setupfile2.dat* file: (1) The first column is a keyword that, generally, tells the PTM what variable or feature to set (keyword must be in lower case); (2) Blank lines are allowed, and a "!" means to discard the rest of the line as a comment; (3) The order of the assignments does not matter. Input data for the particle tracking model with dimensions (length in meters, time in seconds, velocity in m/s), except for output routines where the intervals for writing to file is defined as a certain number of timesteps. Logical variables are declared as *false*, and integer switches are declared as the lowest permissible value, by default.

The recognized keywords are:

```
keyword             value  description
------------------------------------------------------------------
part_activate       <t/f>  Turn particle tracking on/off
part_restart        <t/f>  True if restarting from a checkpoint
part_random_step    <t/f>  Random perturbation in advection
                           of particles
part_random_seed    <t/f>  Set new seed for pseudorandom
                           number generator
part_random_type    <int>  Type of random walk model
                           0=Random walk with default horizontal
                             and vertical diffusivity
                           1=Random walk with constant horizontal
                             and vertical diffusivity
part_ah             <real> Constant horizontal diffusivity
part_kh             <real> Constant vertical diffusivity
part_runge_kutta4   <t/f>  Use trapeziodal method in calculation
                           of advection
part_bilinear       <t/f>  Use bilinear method in calculation
                           of advection
part_h_is_positive <t/f>  Particle depth specified as positive
                           numbers
part_capture_land   <t/f>  Particles captured when hitting land
part_capture_h      <t/f>  Particles captured when hitting bottom
part_capture_eta    <t/f>  Particles captured when hitting surface
part_minh           <real> Min distance the particle can be above
                           the bottom
part_mineta         <real> Min distance the particle can be close
                           to eta
part_init_minh      <real> Min initial distance the particle can
                           be above the bottom
part_init_mineta    <real> Min initial distance the particle can
                           be close to eta
part_minland        <real> Distance to bump back particles that
                           enter land cells.
part_exchange_size <int>  Max number of particles that can be
                           exchanged between subdomains
part_init_data      <int>  <string>  <int>
                           column 2: 1=data from ascii file
                                     2=data from binary file
                                     3=data from binary file,
                                       stream access
                           column 3: filename
                           column 4: number of particles
```

```
keyword            value  description
-------------------------------------------------------------------
part_bound_east    <int>  <real>
                          Particle boundary condition (east)
                          column 2: 1=capture particles
                                    2=bump back particles
                                    3=periodic boundary cond.
                          column 3: Offset distance (index space)
                                    between BOM and particle
                                    boundaries.
part_bound_west    <int>  <real>
                          Particle boundary condition (west)
part_bound_north   <int>  <real>
                          Particle boundary condition (north)
part_bound_south   <int>  <real>
                          Particle boundary condition (south)
part_check_status  <t/f>  Check particle status in output routine
part_outputlevel   <int>  How much writing to stdout
                          0=silent
                          1=normal
                          2=debug
part_write_time    <int>  Write output at timestep intervals
                          part_write_time
partplot_timeseries
                   <t/f>  Write time series of tracks to file
partplot_ts_write  <int>  Write time series output at timestep
                          intervals partplot_ts_write
partplot_distribution     Write distribution of all active
                   <t/f>  particles to file
partplot_dist_write       Write distribution output at timestep
                   <int>  intervals partplot_dist_write
part_use_pgplot    <t/f>  <int>
                          column 2: Plot particles with pgplot
                          column 3: Number of plots to be made
                                    by pgplot (default = -1)
pgpart_plot_time   <int>  Time for updating fields used in pgplot
pgpart_ntime_pts   <int>  Number of time levels in history record
                          for particle positions
pgpart_set_plot    <int>  <int>  <int>  <int>  <int>
                          Define parameters for pgpart_object:
                          column 2: ind
                          column 3: type
                          column 4: style
                          column 5: color
                          column 6: line width
```

C.2 Initial particle data file

Initial data for the particles are expected to be provided in a separate data file. The format of the file, which can be either ascii or binary should be a list of four columns giving the initial time (in seconds simulated time) and the location in (physical) Cartesian coordinates. The following example

```
60.0   10.0   10.0  -99.5
60.0   10.0   40.0  -99.5
60.0   10.0   50.0  -99.5
60.0   10.0   55.0  -99.5
60.0   10.0   60.0  -99.5
60.0   10.0   70.0  -99.5
60.0   10.0  100.0  -99.5
```

shows the initiation file *particle_init.dat* for 7 particles seeded after 60 seconds simulated time, at $x = 10$m, $z = -99.5$m, and over 7 different values of y. To use this file, the *particle_setup.dat* file should contain the line

```
part_init_data  1  particle_init.dat   7
```

C.2.1 Generating initial particle data from a fortran program

A stand-alone fortran program is often used to generate input files for BOM. The same program can also be used to generate initial particle data in binary format. Assume the depth matrix h is defined in a program *init.f90*, and we wish to seed 20,000 particles randomly within a box defined by 30 m $\leq x \leq$ 50 m; 34 m $\leq y \leq$ 78 m, with minimum distance 0,5 m above the bottom and 95 m below the free surface boundary. An example is shown on the next page, which writes output to a binary file *fort.33*. To use this file, the *particle_setup.dat* file should contain the line

```
part_init_data  2  fort.33   20000
```

```
!! Random particle seeding example
!!=================================================
program init
...
... define variables
... define depth matrix h
...

parts = 20000
allocate(part_data(parts,4))
minh = 0.5
mineta = 95.0
box = (/30., 50., 34., 78./)

do start=1,parts
   ! time
   part_data(start,1) = 0.

   ! x-coordinate
   call random_number(ran)
   part_data(start,2) = box(1) + ran*(box(2)-box(1))

   ! y-coordinate
   call random_number(ran)
   part_data(start,3) = box(3) + ran*(box(4)-box(3))

   ! grid indices
   ipos = nint(part_data(start,2)/dx + 1.5)
   jpos = nint(part_data(start,3)/dy + 1.5)

   ! vertical position
   call random_number(ran)
   part_data(start,4) = - (mineta +  &
                       ran*(h(ipos,jpos)-mineta-minh))
end do
write(33) part_data
end program
```

References

A. Bennett. *Lagrangian Fluid Dynamics*. Cambridge Monographs on Mechanics. Cambridge University Press, 2006.

J. Berntsen. USERS GUIDE for a modesplit σ-coordinate numerical ocean model. Technical Report 135, Dept. of Applied Mathematics, University of Bergen, Johs. Bruns gt.12, N-5008 Bergen, Norway, 2000. 48p.

A.F. Blumberg, D.J. Dunning, H.H. Li, D. Heimbuch, and W.R. Geyer. Use of a particle-tracking model for predicting entrainment at power plants on the hudson river. *Estuaries*, 27(3):515–526, JUN 2004. ISSN 0160-8347.

D. Brickman and P.C. Smith. Lagrangian stochastic modeling in coastal oceanography. *J. Atmos. and Oceanic Technology*, 84:83–99, 2002. ISSN 0739-0572.

K.N. Dimou and E.E. Adams. A random-walk, particle tracking model for well-mixed estuaries and coastal waters. *Estuarine Coastal and Shelf Science*, 37(1):99–110, JUL 1993. ISSN 0272-7714.

C.W. Gardiner. *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*. Springer Series in Synergetics. Springer-Verlag, 2nd edition, 1985.

S.B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

H.M. Taylor and S. Karlin. *An Introduction To Stochastic Modeling*. Academic Press, 3rd edition, 1998. ISBN 0-12-684887-4.

A.F.B. Tompson and L.W. Gelhar. Numerical-simulation of solute transport in 3-dimensional, randomly heterogeneous porous-media. *Water Resources Research*, 26 (10):2541–2562, OCT 1990. ISSN 0043-1397.