unptrack User Guide

Version 3.0

Philip Gillibrand <philip.gillibrand@mowi.com>

Mowi Scotland Ltd.

June 8, 2022

An offline lagrangian particle tracking model for simulating the dispersion of particulate and soluble wastes and the dispersal of parasites from marine fish farms using flow fields from unstructured grid hydrodynamic models.

Table of Contents

Contents

1 Introduction

unptrack (unstructured mesh particle tracking) is a lagrangian particle-tracking model designed to simulate the transport pathways of pelagic biota or chemical contaminants using flow fields generated by unstructured mesh hydrodynamic (HD) models. **unptrack** was developed from an earlier particle-tracking model that used hydrodynamic flow fields from regular grid models; the earlier version has been used to simulate the transport and dispersion of solute veterinary medicines (Willis et al., 2005) and dispersal of pelagic organisms, including sea lice larvae (Gillibrand and Willis, 2007) and harmful algal blooms (Gillibrand et al., 2016). In 2017-2018, the original "ptrack" model was developed to use flow fields from the unstructured mesh hydrodynamic models that were increasingly being used in coastal environments, to become "**unptrack**".

The model runs offline; velocity data to drive the model can be obtained from current meter observations or from hydrodynamic model simulations. In the case of the latter, the particle-tracking model will use the same numerical grid as the hydrodynamic model, so that output files from the HD model can be used directly. If measured current data are used, a numerical grid needs to be constructed to cover the area of the simulated dispersion, and the observed current data is applied at each of the grid cells; in this case, the velocity field experienced by the numerical particles is spatially non-varying in the horizontal, although vertical shear can be present if multiple current meters, or multiple bins from an ADCP deployment, are used. In both cases, realistic bathymetry can be used.

Advection can be treated using either a fourth-order Runge-Kutta algorithm or a simple Euler approach. A random walk model is used to simulate horizontal and vertical eddy diffusion. Various aspects of biological development (e.g. temperature-dependent stage development, mortality) and behaviour (e.g. vertical migration, low salinity avoidance) can be simulated. For chemical contaminants, a decay half-life can be simulated. The basic advection, diffusion and biological algorithms in the model have been described by Gillibrand and Willis (2007) and Gillibrand et al. (2016).From version 3.0, a sediment bed model is included to simulate deposition and erosion of settling particles.

The program is written in Fortran 90 and utilises openMP routines on shared memory processors. This user guide provides details on how to set up the necessary input files and the keywords used to specify particle characteristics and behaviours. There are also short sections on the output file formats and results from basic model tests.

2 Model Description

2.1 Mathematical Framework

Numerical particles are advected by the velocity field and mixed by horizontal and vertical eddy diffusion, simulating the physical transport and dispersion of pollutants and organisms in the marine environment. The mathematical framework of the model follows standard methodology for advection and diffusion of particles in the marine environment (Allen, 1982; Hunter et al., 1993; Ross and Sharples, 2004; Visser, 1997), whereby the location $X_P^{t+\Delta t} = X_P^{t+\Delta t}$(x,y,z) of particle P at time $t + \Delta t$, can be expressed as:

$$\mathbf{X}_P^{t+\Delta t} = \mathbf{X}_P^t + \Delta t[\vec{\mathbf{U}}_P + \vec{\mathbf{w}}_P] + \delta_H + \delta_V \tag{1}$$

where $\vec{\mathbf{U}}_P$ is the 3D model velocity vector at the particle location, $\vec{\mathbf{w}}p$ is an additional vertical motion term due to, for example, particle settling or vertical migration, and Δt is the model time step. Particle advection is treated using either a fourth-order Runge-Kutta algorithm or a forward Euler method. Horizontal and vertical eddy diffusion are represented in the model by the “random walk” displacements δ_H and δ_V respectively, given by (Proctor et al, 1994):

$$\delta_H = R[6.K_H.\Delta t]^{1/2} \tag{2}$$

$$\delta_V = R[6.K_V.\Delta t]^{1/2} \tag{3}$$

where R is a real random number, uniformly distributed over the range -1 $\leqslant$ R $\leqslant$ 1 (R must have a mean of zero and standard deviation of $1/\sqrt{3}$), and K_H and K_V are the horizontal and vertical eddy diffusivities respectively.

If the vertical diffusivity is variable in space (i.e. VERTICALDIFF < 0, see Section 5), then a correction must be made to Equation 3 to prevent particles accumulating in regions of low diffusivity (Hunter et al., 1993; Visser, 1997). In this case, Equation 3 becomes:

$$\delta_V = K_V'\Delta t + R[6.K_V(z_p + 0.5K_V'\Delta t)\Delta t]^{1/2} \tag{4}$$

where K_V' is the vertical gradient of the vertical diffusivity calculated at the particle location (depth z_p), and the vertical diffusivity used in the second term on the right-hand-side of Equation 4 is calculated at a depth shifted by $0.5K_V'\Delta t$ from the particle depth z_p. The necessity of this correction for spatially variable vertical diffusivity is demonstrated in the well-mixed condition test (Section 8.3).

In Equation 1 for particulate or biotic substances, $\vec{\mathbf{w}}p$ may represent additional vertical motion of the particle. For particulates, for example, $\vec{\mathbf{w}}p$ may represent a settling velocity. For living organisms,$\vec{\mathbf{w}}p$ may be used to allow vertical migration behaviour; upward and downward swimming speeds can be specified separately. For solute substances, the vertical velocity term

$\vec{\mathbf{w}}p$ may represent additional vertical motion due to, for example, buoyancy.

For settling particles, **unptrack** can simulate sediment deposition and resuspension using either a full bed model or a simplified particle resuspension scheme. In the simpler scheme, individual particles that settle on the seabed can be lifted back into suspension in the water column where they are subject again to advection and diffusion until they resettle. This deposition-resuspension-deposition cycle continues indefinitely. Resuspension is triggered when the calculated bed stress at the settled particle location exceeds a critical erosion stress threshold. In the simpler approach, a stochastic approach is taken to particle resuspension, where a probability of resuspension is calculated for each settled particle. This prevents all particles at a location being resuspended immediately the critical erosion stress is exceeded. A number of algorithms are available to specify the resuspension probability, with options incorporating the age of the particle, with probability of resuspension reducing with particle age (on the basis that older particles are more likely to have been consolidated into the seabed), or the local deposited mass with larger mass reducing the probability of resuspension (on the basis that more sediment mass leads to more rapid consolidation). The advantage of the simpler approach is that it can be applied over the whole model domain, although run times can increase significantly as a result.

From version 3.0, **unptrack** also incorporates a bed model to simulation consolidation and erosion of settled particulates. The bed model framework follows Sandford (2008), though the emphasis here is on muds rather than mixed sediments. Particles settling onto the seabed are incorporated into the bed model. As the mass at each bed model grid cell increases, sediment layering develops. Each layer consists of a specified mass of sediment. Deposition and resuspension occur to and from the surface layer of the bed. The critical erosion thresholds for layers beneath the surface layer increase with depth. When the mass of sediment in the surface layer exceeds the specified layer mass, a new surface layer is formed. Conversely, when the surface layer is eroded away, the number of layers is reduced. The critical erosion and deposition stress thresholds relax towards equilibrium values with user-specified consolidation and expansion time scales. The objective of the bed model is to simulate sediment consolidation and erosion more accurately than the stochastic approach described above. The limitation is that the bed model utilises a regular (not unstructured) grid and must be implemented over relatively small domains in order to avoid excessive computational demands; a bed model domain that is a sub-area of the full model domain can be specified by the user in the run input file.

2.2 Particle Sources

Particles can be released from any number of discrete sources. For each source, a number of source characteristics are required. These are:

x0, y0, z0 the location of the source, in space coordinates (e.g. Easting, Northing, depth)

xrange, yrange, zrange the range by which initial particle locations may differ from the source location. So in the x-dimension, the initial location for particle i may be x0-xrange $\leqslant x_i \leqslant$ x0+xrange. If a source is circular, then xrange sets the radius of the circle, and yrange should be set to zero, otherwise if both xrange and yrange are greater than zero, the source is rectangular.

start stop the start and stop times of the release from this source. Times are in hours relative to the start of the simulation. Individual particle release times from the source are evenly distributed in time between the start and stop time.

Mass or Concentration the total mass of material to be released from the source. This is a positive value in kilograms. Alternatively, a concentration can be specified (as a negative value); in this case, the mass released is calculated from the volume of the source (calculated from xrange, yrange and zrange) and the concentration.

Settling Velocity the settling velocity of particles released from this source (in ms^{-1}). Different particle types, with different settling velocities, released from the same location can be specified as separate sources.

2.3 Particle Characteristics

Each particle may have particular characteristics, which are tracked throughout its lifetime. The age of every particle from release is stored, its source, and a particle mass or, for living organisms, the number of organisms that the numerical particle represents.

Particles may have up to two active life stages: the first is generally passive, whilst the second may exhibit specified behaviours. Modelling solute tracers uses only the first stage, while modelling some living organisms e.g. sea lice larvae, (Gillibrand and Willis, 2007) may use both stages. The duration of each stage is specified by the user in units of days or degree-days. The latter recognises the temperature-dependence of stage development in some living organisms. Behaviours that can be specified for particles include upward and downward swimming, low salinity avoidance and depth limitation (Gillibrand and Willis, 2007). If an upward swimming speed is specified, this is generally applied during daylight hours only (diurnally, specified by default as between 06:00 and 18:00 hours each day, but these times can be set by the user), whereas downward swimming (which can be zero) occurs during night-time (default: 18:00 - 06:00). A phototactic response can be modelled by setting an upward swimming speed greater than zero while setting the downward swimming speed to zero. Different swimming speeds can be specified for the two stages. Swimming of Stage 1 particles (nauplii) can also by thermally driven, whereby particles move upwards or downwards towards warmer water. The low salinity avoidance response is triggered when the local salinity at the particle location (taken from the hydrodynamic model) is less than the trigger value specified by the keyword S_{AVOID} (see below). Since salinity avoidance is assumed to be a rapid swimming response, the upward swimming speed is used but applied downwards until the local salinity becomes greater than the trigger value. The low salinity avoidance response overrides any specified vertical migration swimming behaviour.

Biological mortality is modelled as a reduction in the number of biological organisms, N_P, represented by each numerical particle. Given a mortality rate γ (hr^{-1}):

$$N_P = N_0 e^{-\gamma t_p} \tag{5}$$

where N_0 is the initial number of organisms per particle and t_p is the age of the particle since release. If a mortality rate is specified, it is applied to both biological stages.

For solutes, chemical decay can also be simulated by varying the particle properties. At the time of release, each numerical particle represents a mass, M_0, of the discharged substance, and the chemical mass, M_P, represented by each particle evolves according to:

$$M_P = M_0 e^{-\gamma t_p} \tag{6}$$

where $\gamma = ln(2)/T_{1/2}$ and $T_{1/2}$ is the half-life of the chemical decay. The mass M_P of every particle is stored in each output file.

3 Installation

The source code is in a git repository. The first step is to install git (a revision control program) on the target system. This is freely available software and may be obtained at

http://git-scm.com/downloads

Once git is installed, create a base directory in your user space for the **unptrack** source e.g. /. Change directory to the directory that has just been created. Then type

```
git clone https://github.com/gillibrandpa/unptrack.git
```

which will create a directory ./unptrack and install the code there in ./unptrack/src. Some test cases are provided (in ./unptrack/inputs) and some pre- and post-processing Matlab® scripts. In the next sections, specific instructions are given for building the program with different operating systems.

3.1 Model Build

The model is written in Fortran 90 and has been tested with Intel Fortran (v.19) and GNU gfortran on linux Red Hat Enterprise Linux (RHEL7) and Ubuntu 18.04.4 LTS platforms. A Makefile is supplied with the source code, and can be editted to set the fortran compiler, openMP, optimisation and debug options.
A number of targets are available in the Makefile. These include:

unptrack default, includes openMP

unptrack-dbg includes debugging compiler options

unptrack-serial no openMP

unptrack-verbose includes additional write statement during execution

3.2 Model Execution

uptrack is run using the following syntax:

$unptrackPATH/*program inputs/inputfile.dat*

Depending on the model required, the program executable can be:

unptrack the default version of the model, using openMP and limited text output.

unptrack_dbg the debug version of the code.

unptrack_serial a single processor version of the code with no openMP.

unptrack_verb the verbose version of the code with extra text output during the simulation.

Input files are expected to be in the *inputs* subdirectory (see next section). Output files are written to the *results* subdirectory.

3.3 Directory Structure

A sample **unptrack** working directory can be created as shown here. The following describes the contents of each of the folders in this directory tree.

.. /project name
The main directory for a particular project.

.. /project name/inputs
Contains input files:

grid-input.txt key grid and particle parameters, used for memory allocation
grid-data.dat grid file containing node information and connectivity (element) list.
run-input[optional-tag .dat] control run file
***.nc** netCDF file containing flow fields from hydrodynamic model.
or
***.dat** observed flow data from current meter deployment.

.. /project name/results
Contains the output files:

model_parameters.log parameter values and hourly particle count
particleCharacteristics.dat particle information
grid_coordinates.dat model grid
node-elements-out.dat list of elements surroundin each node
***.nc** particle locations and characteristics at each output time step.

The following files may also be generated depending on the output options selected:
EQScompliance.csv peak concentrations, area exceeding EQS and total mass
Connectivity_stage*.dat connectivity matrix for stage 1 or 2 particles.

pdepos*.rgr	particle deposition calculated on a regular (square) grid.
pdnsty*.dat	particle density calculated on a regular (square) grid.

The netCDF, deposition and density output files have a suffix (*) of the output time in seconds (see Section 6).

4 Pre-Processing

The file *grid-data.dat* contains details of the source unstructured mesh of the hydrodynamicflow fields. The first line is a header containing the number of nodes (vertices) and the number of elements (triangles) in the mesh. The following lines, each of 3 values, provide the x- and y-coordinates (assumed to be in units of metres) of the grid nodes together with the water depth at each node. The node data is followed by the connectivity (element) list, detailing the nodes that make up each triangular element. The mesh format follows standard protocols for unstructurd mesh hydrodynamic models such as FVCOM (Chen et al., 2003) and RiCOM (Walters and Casulli, 1998; Walters, 2006).

The file *grid-input.txt* (Figure 1) contains grid and particle metrics used for memory allocation, including the number of nodes, the numbers of elements, the maximum number of neighbouring nodes, a constant water depth (optional), a minimum water depth (optional), the maximum number of particles to be released during a simulation, and the number of vertical layers of hydrodynamic flow data. The latter value is followed by a line containing the sigma depths of the flow data.

The flow data is typically output from a hydrodynamic model such as FVCOM and can consist of either cell-centred or node-based velocity data (*u, v*) and optionally surface elevation data (*zeta*).

Alternatively, velocity from a single location, as recorded by a profiling current meter or a single point current meter, can be used. In this case, the velocity does not vary spatially across the model domain but does vary temporally. Note that a triangular mesh is still required, covering the area where modelling outputs are required.

The type of velocity data used must be specified by the *utype* variable in the *grid-input.txt* file. The following codes apply:

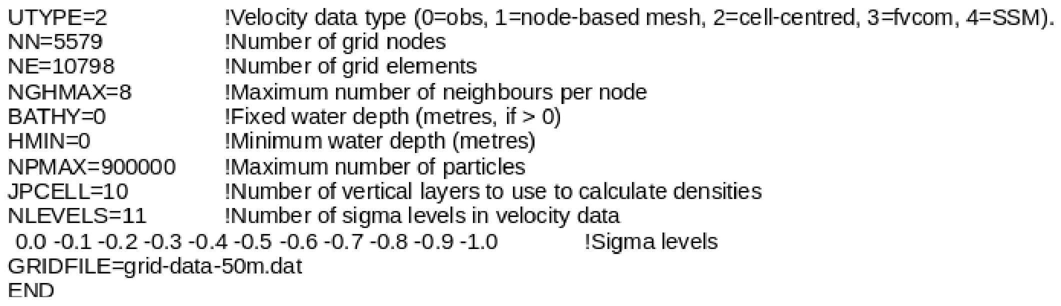
utype = 0 for current meter observations from a single location

utype = 1 for an unstructured mesh with data defined at the nodes

utype = 2 for an unstructured mesh with all data defined at the cell centres

utype = 3 for an FVCOM-style mesh with velocity data defined at the cell centres and scalars at the nodes

utype = 4 for the Scottish Shelf Model (SSM) and sub-models, and WestCOMS (Aleynik et al., 2016).



```
UTYPE=2          !Velocity data type (0=obs, 1=node-based mesh, 2=cell-centred, 3=fvcom, 4=SSM).
NN=5579          !Number of grid nodes
NE=10798         !Number of grid elements
NGHMAX=8         !Maximum number of neighbours per node
BATHY=0          !Fixed water depth (metres, if > 0)
HMIN=0           !Minimum water depth (metres)
NPMAX=900000     !Maximum number of particles
JPCELL=10        !Number of vertical layers to use to calculate densities
NLEVELS=11       !Number of sigma levels in velocity data
 0.0 -0.1 -0.2 -0.3 -0.4 -0.5 -0.6 -0.7 -0.8 -0.9 -1.0        !Sigma levels
GRIDFILE=grid-data-50m.dat
END
```

Figure 1: Example grid input file.

The value of *utype* must be consistent with the VELOCITYDATA keyword in the run control file (see next section).

5 The Run Control File

Details of specific simulations are supplied through the run control file. This is a text file editted using any standard text editor. Examples of run control files are included in the inputs folder.

The list of keywords are grouped into four approximate themes, describing simulation details, biological parameters, physical processes and the bed model. A brief description of each keyword, with available options, follows. Units and/or format are given in parentheses, default values are given in square brackets.

#SIMULATION DETAILS

DELTAT= Time step of the particle model (seconds)

NSUBSTEP= Maximum number of sub-time-stepping iterations to perform when the advection or diffusion schemes lose track of particles with the full time step. Each iteration halves the model time step.

DT= Time step of modelled or observed flow data (seconds)

STARTDAY= Start day of simulation, relative to the start of the flow data [STARTDAY=1]

STARTTIME= Time of start of simulation on STARTDAY (hhmmss)

DURATION= Length of simulation (hours)

OUTPUTSTART= Start of output (hours)

OUTPUTFREQ= Output interval (seconds)

VELOCITYDATA= Type/source of velocity data. Options include: *obs*, *mesh*, *nccc* or *fvcom*. These correspond to values of **utype** of 0, 1, 2 and 3 respectively. *mesh* is used for an unstructured mesh with data defined at the nodes, *nccc* is used for a netCDF file with all data defined at cell centres, *fvcom* is used for an FVCOM-style netCDF file with velocities specified at cell centres and scalars defined at the nodes. **unptrack** can also use flow fields from the Scottish Shelf Model (SSM, Marine Scotland (2016)), its sub-models (e.g. ECLH, WLLS etc) or the WestCOMS model developed by the Scottish Association for Marine Science (Aleynik et al., 2016); in this case the SSM or WestCOMS acronyms are used i.e. VELOCITYDATA=SSM or VELOCITYDATA=WestCOMS, and **utype** must be 4.

filename For options **utype** = 0, 1, 2 and 3, *filename* is the name of a file containing the velocity (and optionally, sea surface height, temperature, salinity and vertical diffusivity) data. This can be observational data (**utype** = 0) or an output file from a hydrodynamic model (**utype** = 1, 2, 3). For the Scottish Shelf Model and sub-models (**utype** = 4), *filename* is the root of the data file names e.g. 'SSM/Climatology_swona' where 'SSM' is a symbolic link in the inputs directory to the directory containing the climatology flow files 'Climatology_swona_*.nc'. For WestCOMS (**utype** = 4), *filename* is the name of a text file which contains the path and filenames of the WestCOMS output files to be used.

ADV_SCHEME= Advection scheme, Euler forward or Runge-Kutta order 4 (euler or rk4)

BBL= Bottom boundary layer type, used in the deposition algorithm, defining the shape of the velocity profile below the lowest velocity data point, either uniform or logarithmic (uniform or log).

OUTPUT_FORMAT= Output format, netCDF, binary, or ASCII ([NC], BIN or DAT)

OUTPUT_FULL= Output all particles, whether active or passive (T/[F]). If false, only positions and characteristics of active particles are saved.

OUTPUT_PARTICLES= Output particle positions ([T]/F)

OUTPUT_PDENSITY= Output particle densities (T/[F])

CALC_DENSITY= Frequency (in time steps) of particle density calculation

OUTPUTUNITS= Units for deposition (g, mg, ug) per m^2 or concentration output per m^3. Only applies to deposition, PDENSITY or PEAK output. Default is kg.

OUTPUT_PEAK= Output peak concentration and associated variables (T/[F])

EQS= Environmental quality standard (EQS) if used (units in OUTPUTUNITS) and depth over which it is applied.

CALC_CONNECTIVITY= Calculate connectivity between source locations and a list of destination locations ([0] = no, 1 or 2 = stage)

inputs/destination_file_name.dat

#BIOLOGICAL PARAMETERS

PASSIVESTAGE= Duration of passive stage. A positive value indicates a fixed duration, a negative value indicates temperature-dependent development (+days or -degree-days).

LIFESPAN= Total lifespan of the larval stage. A positive value indicates a fixed duration, a negative value indicates temperature-dependent development (+days or -degree-days).

NAUPLIISWIMSPEEDUP= Upward swimming speed for Stage 1 particles (metres per hour)

NAUPLIISWIMSPEEDDOWN= Downward swimming speed for Stage 1 particles (metres per hour)

NAUPLIISWIMSTRATEGY= Swimming strategy for Stage 1 particles (DIURNAL, THERMAL, or NONE)

SWIMSPEEDUP= Upward swimming speed for Stage 2 particles (metres per hour)

SWIMSPEEDDOWN= Downward swimming speed for Stage 2 particles (metres per hour)

STARTSWIMUP= Time of day that upward swimming begins (HH)

STARTSWIMDOWN= Time of day that downward swimming begins (HH)

S_AVOID= Salinity trigger for avoidance behaviour, if > 0. If lower salinity is detected, the particle "swims" downward. Note that this is considered "burst" swimming, and the upward swim speed is used.

CELLGROWTH= Cell growth switch, for phytoplankton simulations (1 = yes, 0 = no)

CELLMORTALITY= Cell mortality switch (1 = yes, 0 = no)

MORTALITYCONST= Constant in mortality expression (per hour)

HALFLIFE= Half-life of chemical decay (hours)

MINSHEAR= Minimum velocity shear for phytoplankton behaviour (per second)

MASSPERPARTICLE= Mass of chemical or number of organisms per virtual particle

PARTICLERELEASE= Release rate of particles per hour per source. If used, mass per particle is calculated from the total mass per source and the total number of particles to be released from that source, overriding the MASSPERPARTICLE value.

CHLPERCELL= Amount of chlorophyll per cell, for phytoplankton simulations (micrograms chl per cell)

DEPTHLIMIT= Maximum particle depth permitted (metres)

PINITIAL=n Initial particle distribution. If $n > 0$, n particles are released in each grid cell at the start of the simulation. If $n < 0$, initial concentrations are read in from an input file and converted to particle numbers using MASSPERPARTICLE. If $n = 0$, initial particle release is zero.

SOURCETYPE= Type of source, either constant (CONSTANT) or time-varying (VARIABLE). If SOURCETYPE=VARIABLE, a filename containing the time-varying source data must be supplied on the following line.

NSOURCE= Number of particle sources, N. If N $>$ 0, source details are read from the following N lines. If N $<$ 0, the following line should be the name of a text file containing the source details (in the same format as described below). In both cases, particles are released at a steady rate from the start time (in hours) to the stop time (hours).

#Source parameter format: x0, y0, z0, x-range, y-range, z-range, start, stop, +mass or -concentration, settling velocity

#PHYSICAL FORCING PARAMETERS

SEAWATERDENSITY= Density of seawater (kgm^{-3}). Default = 1025.0

HORIZONTALDIFFTYPE= Method for calculating horizontal diffusion. Options are: CONSTANT|OKUBO|SMAGORINSKY|COMPUTED. For CONSTANT, the value of HORIZONTALDIFF (below) gives the horizontal diffusivity applied everywhere. For OKUBO, the horizontal diffusivity is calculated based on the model mesh, with $K_H = 0.0103l^{1.15}$ (Okubo , 1971), where $l = (2A)^{0.5}$ is a length scale related to the area A of each grid cell. The OKUBO method is invariant with time and does not use the HORIZONALDIFF value. The SMAGORINSKY option calculates horizontal diffusivity based on the widely used algoritm by Smagorinsky (1963), which uses the length scale l squared and the local horizontal velocity shear; here, the value of HORIZONTALDIFF is used as the so-called Smagorinsky coefficient. Finally, the COMPUTED option uses values of horizontal diffusivity calculated by the hydrodynamic model; in this case, the hydrodynamic model output files must contain a variable called 'viscofh'.

HORIZONTALDIFF= Value for horizontal diffusion (m^2s^{-1}) if HORIZONTALDIFFTYPE= CONSTANT or the Smagorinsky coefficient if HORIZONTALDIFFTYPE=SMAGORINSKY.

VERTICALDIFF= Value for vertical diffusion (m^2s^{-1}). If VERTICALDIFF $<$ 0, **unptrack** uses the temporally- and spatially-varying vertical eddy diffusivity from the hydrodynamic model. A variable "kh" must be present in the netCDF flow data file.

USEW= Run in 3D mode using vertical velocity from the hydrodynamic model ([0]/1). If USEW = 1, a vertical velocity field ("w" or "ww") is expected in the flow data file.

USESSH= Use sea surface height (SSH) from the hydrodynamic model ([0]/1). If USESSH = 1, a surface elevation field ("zeta" or "eta") is expected in the flow data file.

WINDFORCING= Switch for wind forcing applied directly to particles ([0]/1)

STOKESDRIFT= Switch for Stokes Drift ([0]/1). If STOKESDRIFT = 1, **unptrack** uses the temporally- and spatially-varying surface velocity fields derived from a wave model. The wave data must be on the same model grid as the hydrodynamic data. The wave velocities are added to the hydrodynamic model velocity. Variables ”usd”, ”vsd” and ”k”, representing the east and north drift velocity components and wavenumber respectively, must be present in the netCDF data file.

stokesVelocityDataFile.nc startday name of netCDF file containing Stokes Drift velocity data (including variables ”usd”, ”vsd”, ”k”), and the day of the data on which the simulation starts.

BEACHING= Allow beaching of particles on coastline ([0]/1)

BUOYANCY= Switch for buoyant particles ([0]/1)

#BED MODEL PARAMETERS

BED_MODEL= Run bed model (T/[F])

#Xmin Xmax Ymin Ymax dx Limits of the bed model grid and the grid spacing (dx).

RESUSPENSION= Particle resuspension for settled particles [0 - 5].

BEDROUGHNESS= Bed roughness length scale, z_0 (m) [0.01]

EROSIONCRITICALSTRESS= Critical stress (Pa) for erosion at sediment surface, τ_c [0.02]

DEPOSITIONCRITICALSTRESS= Critical stress (Pa) for deposition (if <0, the Bagnold formula is used), τ_s [0.02]

SEDIMENTSETTLINGVELOCITY= Settling velocity (m.s) of sediment [0.0057]

EROSIONCOEFFICIENT= Coefficient in erosion mass calculation, c_p [0.031]

EROSIONEXPONENT= Exponent in erosion mass calculation, β [1]

TCONSOLIDATION= Time scale (s) for bed layer contraction, λ [0]. If a value of zero is applied, the consolidation time scale is infinite.

TSWELLING= Time scale (s) for bed layer expansion [0.] If a value of zero is applied, the expansion time scale is infinite.

BEDLAYERMASS= Mass (kg) per bed layer per regular grid cell [100]

SEDIMENTDENSITY= Density of sediment (kg/m^3) [1400]

BEDFRICTIONANGLE= Bed friction angle (degrees) for internal critical erosion stress [23]

BEDPARTICLERELEASERATE= Number of bed particles released per cell per time step [1]

BEDRELEASEHEIGHT= Height (m) of bed particle release in grid cell [0.35]

BEDRELEASELOCATION= Location of bed particle release in grid cell ([CENTRE]/RANDOM)

BEDVERTICALDIFF= Value for vertical diffusion (m^2s^{-1}) for resuspended particles in the bed model.

Resuspension of settled particles is activated by setting RESUSPENSION to a value greater than zero. The simpler stochastic resuspension scheme can be implemented by invoking resuspension but not the bed model (RESUSPENSION $>$ 0 and BED_MODEL='F'), whereas implementing the bed model needs both parameters to be invoked (RESUSPENSION $>$ 0 and BED_MODEL='T'). In the latter case, the value of RESUSPENSION is not significant, as long as it is greater than zero.

For the simpler stochastic resuspension approach, the choice of resuspension algorithm is set by the value of RESUSPENSION:

RESUSPENSION = 0 : no resuspension

RESUSPENSION = 1 : if calculated bed stress is greater than the critical erosion stress, τ_c, the particle is resuspended.

RESUSPENSION = 2 : if calculated bed stress is greater than the critical erosion stress, τ_c, then the resuspension probability is a function of the excess stress, given by:

$$P = c_p(\tau - \tau_c)^\lambda \tag{7}$$

RESUSPENSION = 3 : if calculated bed stress is greater than the critical erosion stress, τ_c, then the resuspension probability is an exponential function of the excess stress and particle age, given by:

$$P = c_p(\tau - \tau_c)e^{(-t_p/\lambda)} \tag{8}$$

Here, the probability of erosion is assumed to reduce with particle age, scaled by λ, as it becomes more likely that with age the particle is consolidated into the seabed sediment.

RESUSPENSION = 4 : if calculated bed stress is greater than the critical erosion stress, τ_c, then the resuspension probability is an inverse function of the excess stress and particle age, given by:

$$P = c_p(\tau - \tau_c)[1 + t_p/\lambda]^{-\beta} \tag{9}$$

Here, the probability of erosion is assumed to reduce with particle age, scaled by λ, as it becomes more likely that with age the particle is consolidated into the seabed sediment.

RESUSPENSION = 5 : if calculated bed stress is greater than the critical erosion stress, τ_c, then the resuspension probability is an inverse function of the excess stress and local deposition, d ($kg\ m^{-2}$), given by:

$$P = c_p(\tau - \tau_c)[1 + d/\lambda]^{-\beta} \tag{10}$$

Here, the deposition, d, in each grid cell is calculated by **unptrack** at a frequency specified by the keyword CALC_DENSITY. As local deposition increases, scaled by λ, Equation 10 reduces the probability of erosion as it becomes more likely that the particle is consolidated into the seabed sediment in areas of heavy deposition.

In options 2 - 5 above, the values of c_p, λ, τ_c and β are free parameters that can be used to tune and calibrate the deposition model. The calculation of bed shear stress is also dependent on another free parameter, z_0, if a log boundary layer is specified (keyword BBL). The critical settlement stress, τ_s, can be specified directly or the calculation given by Bagnold (1966) can be invoked by setting $\tau_s < 0$. In both cases, particles only settle when the local bed stress is less than the critical settlement stress i.e. $\tau < \tau_s$.

Settled particles are assumed to quickly break down and take on characteristics of fine sediment particulates. The principal change is that the settling velocity of settled particles is changed to $w_s\ m\ s^{-1}$ for the remainder of the simulation.

6 Output Files

Output files are written to sub-directory ./results. At present, a single netCDF file is written for every output time: a simulation of 8760 hours (1 year), with output every 3 hours, would produce 2921 files, each containing results pertinent to the specific time step. Various types of output are possible:

OUTPUT_PARTICLES This is the standard output, providing the location (x, y, sigma-coordinate, z) of every particle, together with particle characteristics (mass, age, stage, source, particle number), where the mass is the quantity of modelled substance represented by each particle (after decay, mortality etc). If OUTPUT_FULL=T, data for all particles are written, including those not yet released and those that have been lost or become inactive. If OUTPUT_FULL=F [default], only data for active particles (Stages 1 and 2) are written. The output file names are ptrack-ddddddddd.nc, where ddddddddd is the time step in seconds.

OUTPUT_PDENSITY If true, (OUTPUT_PDENSITY=T), the density of particles in each grid cell is calculated at each time step. This is only recommended if the particle density is required for a process in the model e.g. density-dependent mortality, otherwise post-processing is recommended. The output files are ASCII text files, with names following pdnsty-ddddddddd.dat, where ddddddddd is the time step in seconds.

OUTPUT_PEAK If true, (OUTPUT_PEAK=T), the peak concentration of particles (i.e. modelled substance) over the model domain is determined at each output step and written to an output file "EQScompliance.csv". The concentration units are specified by OUTPUTUNITS. The area (in km^2) of the domain where an Environmental Quality Standard (specified by the EQS keyword) is exceeded is also determined and written to the file. Finally, the total mass of substance represented by active particles in the model is calculated and written to the file at each output step.

BED_MODEL If true, (BED_MODEL=T), the deposition of particles on the seabed is calculated and output every time step. A rectangular sub-grid is created, with the limits of the grid along the x- and y-axes specified in the following line (Xmin, Xmax, Ymin, Ymax). The output files are ASCII text files, with names following pdepos-ddddddddd.rgr, where ddddddddd is the time step in seconds.

7 Post-Processing

Two key scripts for post-processing model results are plot_particle_loc.m and plot_particle_conc.m. The former plots particles on the model mesh, the latter calculates and plots concentrations. Usage of the two scripts is as follows:

[pxysz, pass, varargout] = plot_particle_loc(pTime,pStage,pSource,plotYN,varargin);
where

pTime is the time (in seconds) of the data to plot. pTime can be a vector, in which case an animation of particle locations is created.

pStage identifies the particle stage to plot (1 or 2). If pStage=0, both active stages are plotted, distinguished by color. If pStage ¡ 0, particles from each source are plotted in different colours (but only one stage can be plotted).

pSource specifies whether to plot particles from a specific source only (pSource > 0) or from all sources (pSource = 0).

plotYN determines whether particle locations are plotted ('Y') or not ('N'). If plotYN='movie', a movie file ParticlePlayBack.avi is created.

pxysz contains location data for each particle: x-coordinate, y-coordinate, sigma-coordinate, z-coordinate

pass contains particle characteristics: particle mass, particle age, particle stage, source number and particle number.

Further input and output options are available and are described in the script header.

The second script plots modelled concentrations:

[pconc, mesh, varargout] = plot_particle_conc(gridFileName,pTime,depth,pStage,pSource,plotYN,units,figur
where

gridFileName is the name of a model grid file. The grid output file "grid_coordinates.dat" can be used here. Alternatively, the grid can be read in advance using e.g. mesh = read_**unptrack**_grid('grid_coordinates.dat',1) and the structured array "mesh" input as the first argument.

depth specifies the limits of the layer to plot e.g. depth = [10 20] plots particle concentrations between 10 and 20 m. If depth has a single value, z, the surface layer from 0 - z m is calculated. If depth = 999, the depth-integrated concentration over the whole water column is calculated.

units Specify the units to use for concentrations. Options are: 'ng/L', 'ug/m2', 'ug/kg', 'ug/L'), 'mg/m2', 'mg/L', 'g/m2'. If not specified, the default units are 'kg/m3'.

figureprops is a structured array that contains figure properties. See the script for details.

pconc calculated concentrations of the horizontal grid over the depth range defined by **depth**.

Further input and output options are available and are described in the script header.

Note that plot_particle_conc.m uses the particle location files ("ptrack-*********.nc") to calculate concentrations. It does not use the output files generated by OUTPUT_PDENSITY='T' or OUTPUT_DEPOSITION='T'. Other scripts are required to read those files.

To plot test results, go to the results folder in Matlab. Load in the *.mat file for the appropriate test run. For example, following the advection test case:

unptrack inputs/run-input-advection.dat

```
>> load plotAdvection.mat

>> plot_particle_loc(pTime,pStage,pSource,'y','grid_coordinates.dat',figureprops);
```

For the concentration simulation:
unptrack inputs/run-input-bath.dat

```
>> load plotBath.mat

>> plot_particle_conc(mesh,pTime,depth,pStage,pSource,'y',units,figureprops);
```

The *.mat files contain appropriate parameter values to produce basic plots and animations.

8 Model Tests

8.1 Advection

The dispersion model has been subjected to various tests, including the standard Brickman test (Brickman et al., 2009) to ensure advection is treated accurately in spatially-varying flow fields. The test releases particles into a steady flow field which is derived from the analytical solution for flow round an obstacle (Figure 2). Particles are released near the left-hand boundary (x = 3000 m) and allowed to advect for 24 hours. Diffusion is set to zero. After 24 hours, the particles should have advected toward the right-hand boundary; however, since the particles toward the bottom half of the domain (y < 16000 m) have to traverse around the obstacle midway along the domain, they trail the particles in the northern half of the domain (Figure 3). Increasing the time step with the Euler Forward advection scheme from 60 s to 600 s leads to increasing inaccuracy in the final particle locations. The distributions of particles for the RK4 scheme and Euler scheme with $\Delta t = 60$ s match those presented by (Brickman et al., 2009), giving confidence in the advection algorithm in **unptrack**.

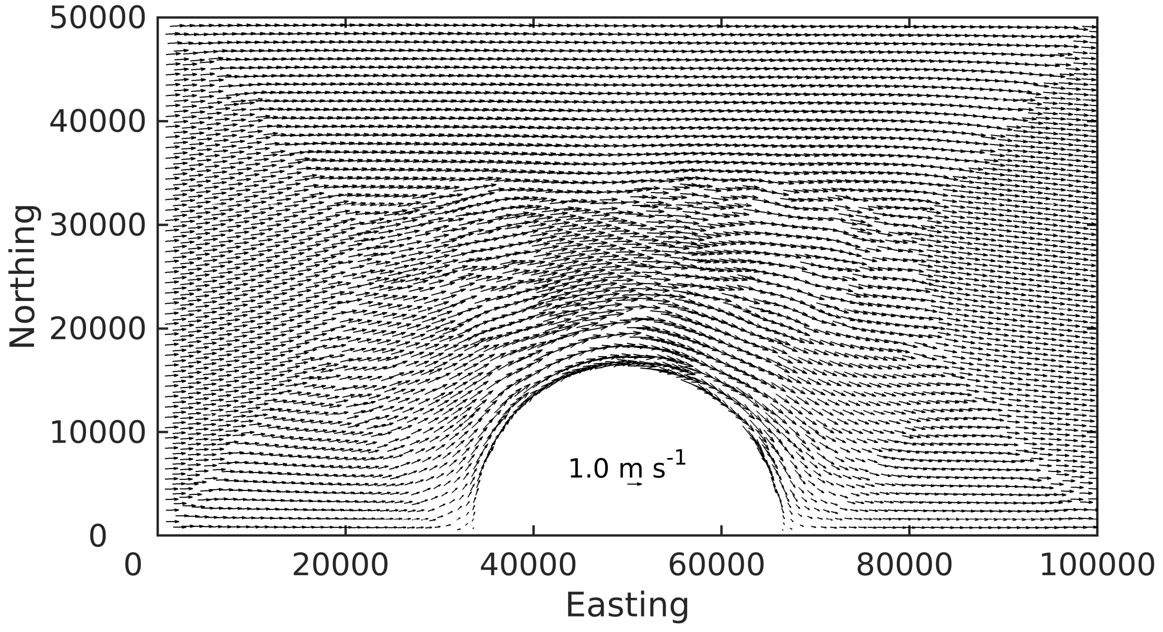


Figure 2: Flow vectors for the Brickman test. Flow at the left-hand boundary is 1.0 ms^{-1}.

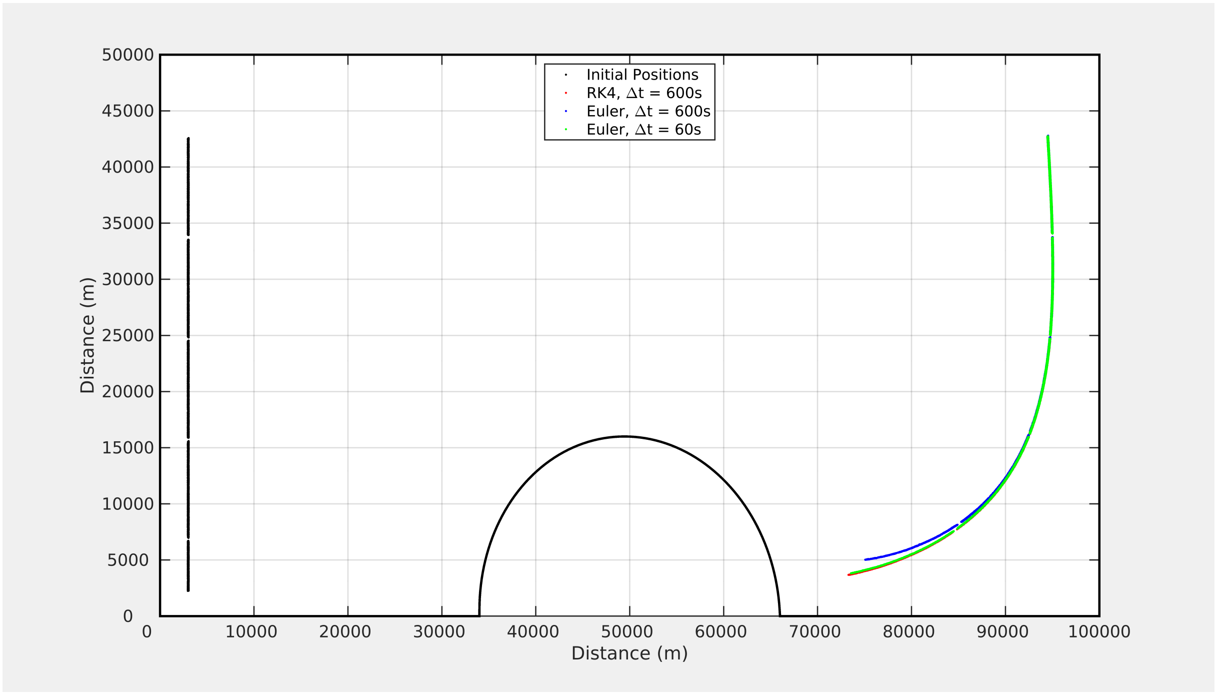


Figure 3: Brickman test results: particle locations after 24 hours advection. Particle locations for the 4th order Runge-Kutta (RK4) advection scheme with $\Delta t = 600$s, and the Euler scheme with $\Delta t = 600$s and 60s are shown. The particles for the Euler scheme with $\Delta T = 60$s overlay the particles from the RK4 scheme.

8.2 Diffusion

For the diffusion test case, the Brickman model domain was again used, but with the velocity everywhere set to zero. Particles were released from five locations across the domain, and allowed to diffuse horizontally and vertically for 72 hours (Figure 4). By determining the variance of particle locations from each source, the modelled diffusion can be compared to the theoretical diffusion predicted by Fickian theory. The random walk algorithm correctly simulated the increase in particle variance with specified horizontal dispersion coefficients of $K_X = K_Y = 0.1$ m^2s^{-1} and 1.0 m^2s^{-1} (Figure 5).

In the vertical, the particles were released at 25m depth, in a total water depth of 50m. The final vertical distribution of particles after 72 hours is shown in Figure 6. The modelled vertical variance agrees well with Fiackian diffusion theory.

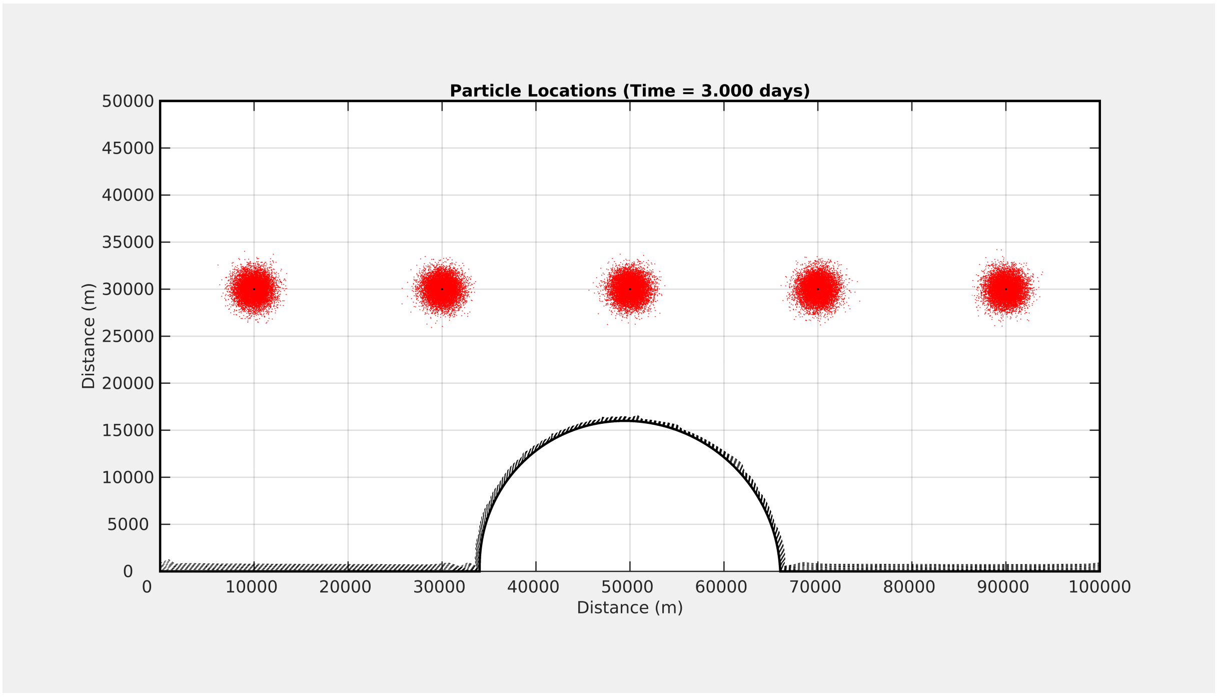


Figure 4: Diffusion test results: particle locations after 72 hours diffusion with a horizontal diffusion coefficient of 2.0 m^2s^{-1}.The black spots indicate the five release locations. The red dots indicate the final particle locations.

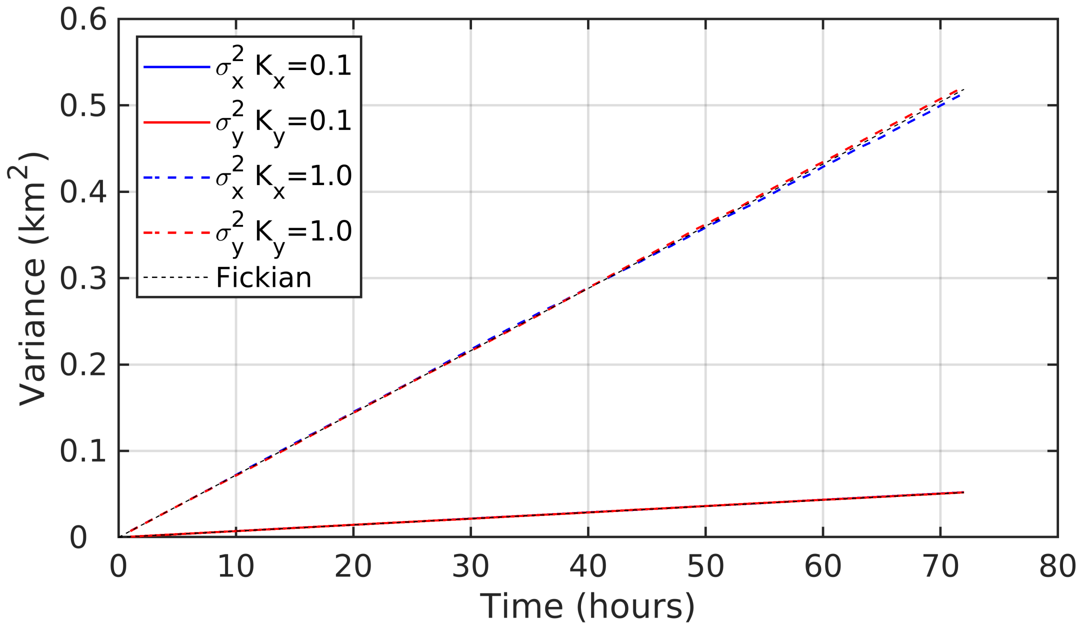


Figure 5: Diffusion test results: modelled horizontal variance over time of particle locations from a single source. Cases with horizontal diffusivities of $K_X = K_Y = 0.1\ m^2s^{-1}$ and $1.0\ m^2s^{-1}$ are shown. The theoretical variance due to Fickian diffusion is shown for comparison in each case.

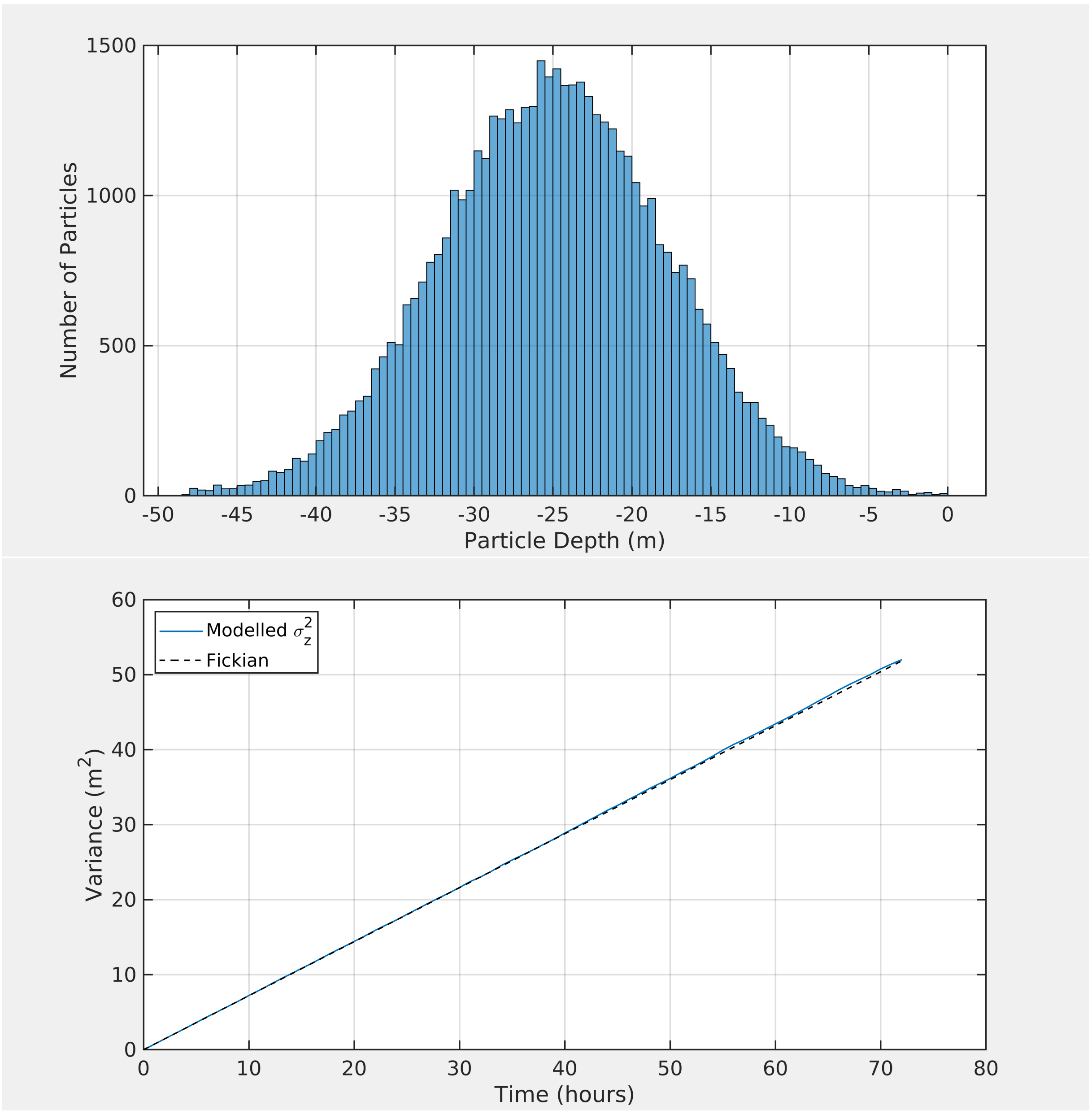


Figure 6: Diffusion test results: modelled vertical distribution of particles after 72 hours (top) and the variance over time of particle locations from a single source (bottom). The release depth was -25 m and the vertical diffusivity was $K_Z = 0.0001\ m^2 s^{-1}$. The theoretical variance due to Fickian diffusion is shown for comparison.

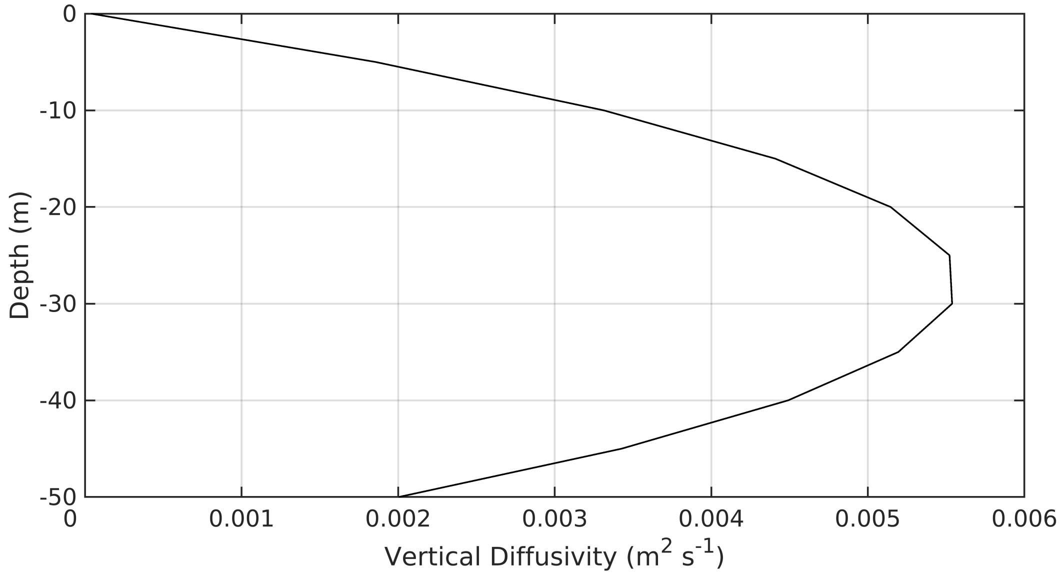


Figure 7: Well mixed condition test:the profile of vertical diffusivity used in the test.

8.3 Well Mixed Condition Test

A fundamental property of a lagrangian particle tracking model is that it maintains an initially uniform distribution of particles uniform for all time. This is known as the well-mixed condition (Brickman and Smith, 2002; North et al., 2006). Where the vertical diffusivity varies in space, a correction to the vertical random walk (Equation 4) must be made to maintain well-mixed conditions. Here we test this using a simple test on the Brickman model domain. The water depth is 50m uniformly across the domain. The vertical diffusivity varied with depth uniformly across the domain (Figure 7). Particles were released uniformly (Figure 8) over the 50m deep water column at five locations.

The model was run for 72 hours. No flow was applied, so particles were subject only to horizontal and vertical diffusion. The model was run first with the simple random walk method (Equation 3) and the final vertical distribution of particles after 72 hours is shown in Figure 9. The migration of particles to the regions of low diffusivity near the surface and seabed is evident.

With the corrected random walk (Equation 4), the well mixed condition is maintained (Figure 10). The correction is applied automatically within **unptrack** when the vertical diffusivity in read from the hydrodynamic model output file (VERTICALDIFF < 0).

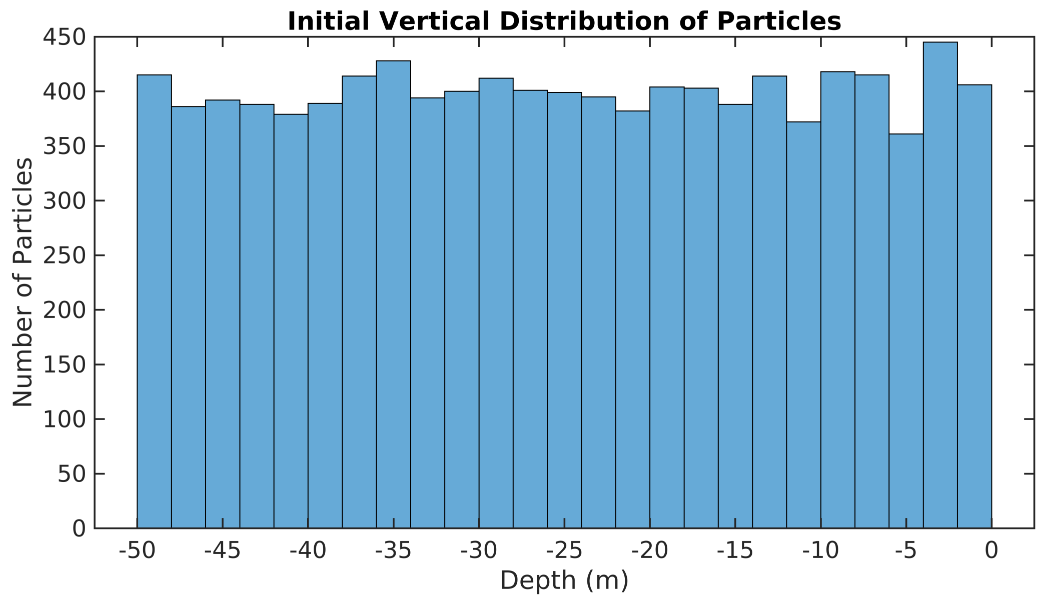


Figure 8: Well mixed condition test:the initial vertical distribution of particles.

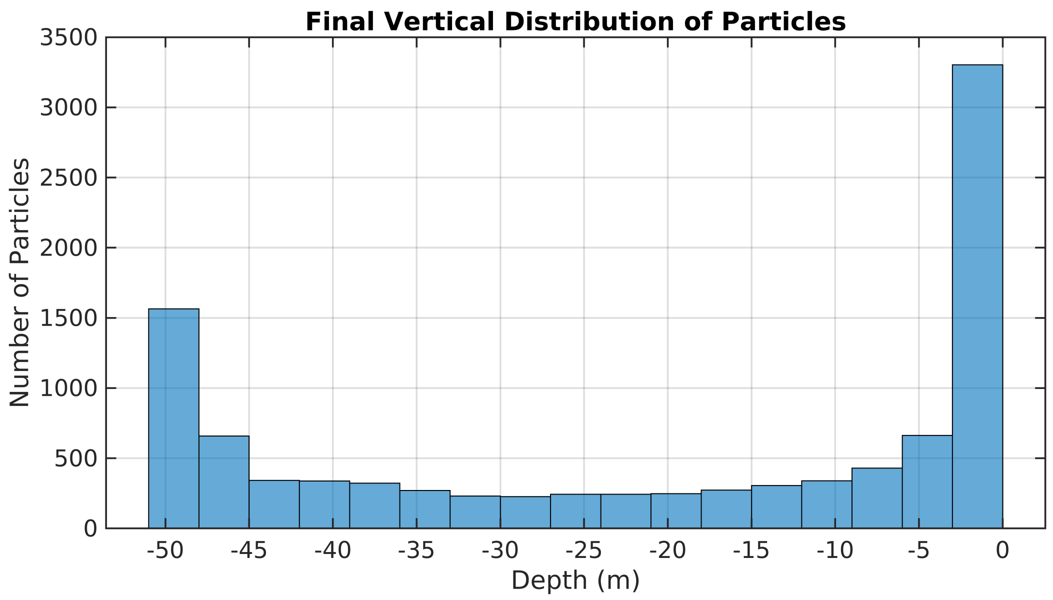


Figure 9: Well mixed condition test:the final vertical distribution of particles after 72 hours using the simple random walk method (Equation 3).

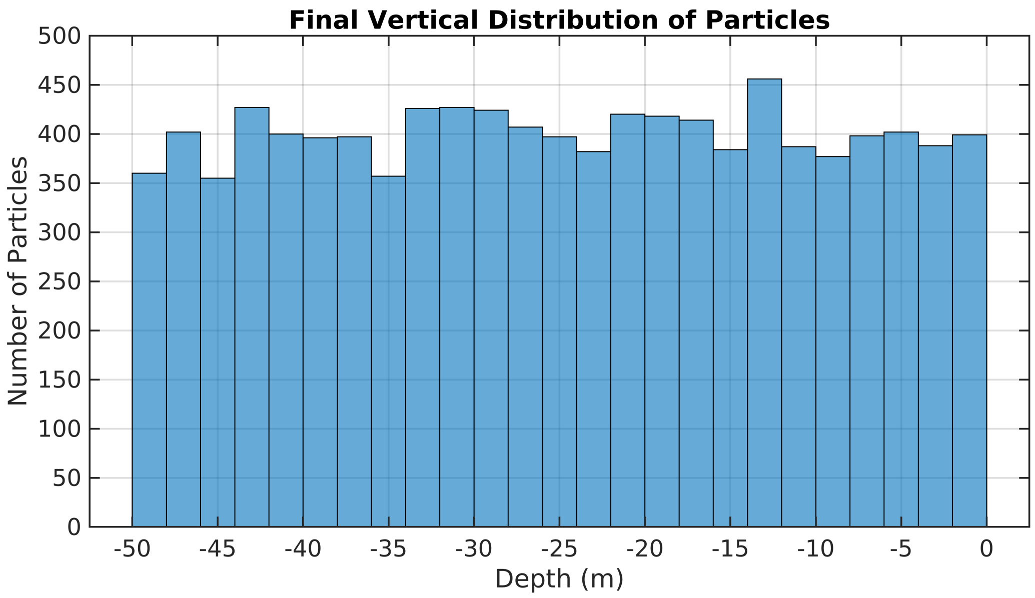


Figure 10: Well mixed condition test:the final vertical distribution of particles after 72 hours using the corrected random walk method (Equation 4).

8.4 Chemical Decay

The diffusion test was repeated with several different half-lives for chemical decay. The mass of material released at each of the five sources was 1 kg. The half-lives tested were $T_{1/2} = \infty$, 213.6 h, 134.4 h and 55.2 h. The total active mass in the simulation for the different half-lives was calculated and accurately tracks the decaying mass (Figure 11).

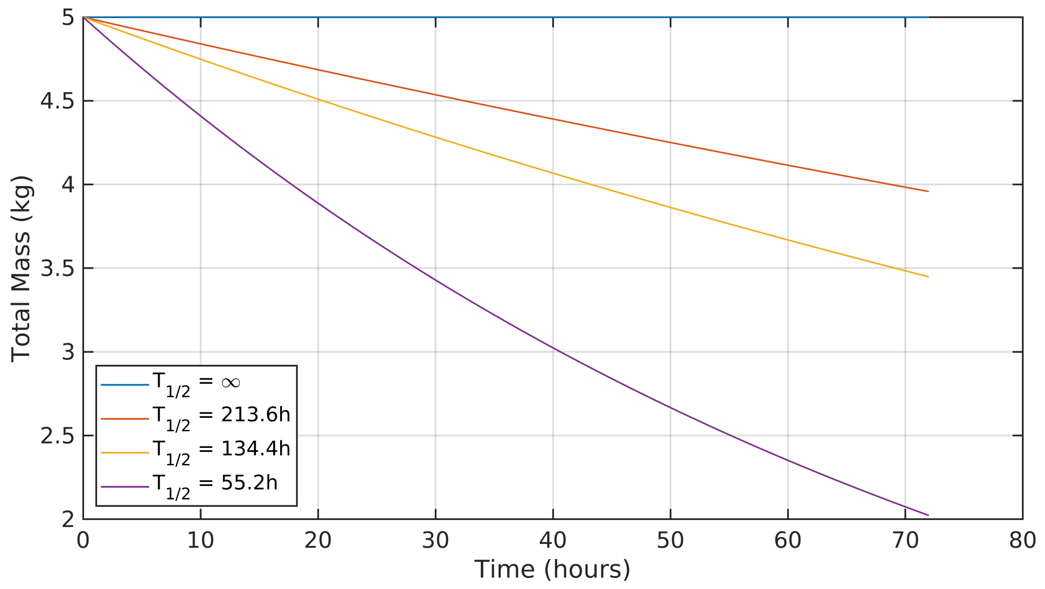


Figure 11: Chemical decay test results: the total mass of material over the test simulation for different half-lives ($T_{1/2}$).

References

Aleynik, D., Davidson, K., Dale A.C. and Porter, M. A high resolution hydrodynamic model system suitable for novel harmful algal bloom modelling in areas of complex coastline and topography. Harmful Algae, 53(3), 102–117, 2016.

Allen, C.M. Numerical simulation of contaminant dispersion in estuary flows. *Proc. Royal. Soc. London (A)*, 381, 179–194, 1982.

Bagnold, R.A. An approach to the sediment transport problem from general physics. *US Geological Survey Professional Paper*, 422-I, 37pp.

Brickman, D., Ådlandsvik, B., Thygesen, U.H., Parada, C., Rose, K., Hermann, A.J. and Edwards, K. Particle Tracking. *In: Manual of Recommended Practices for Modelling Physical – Biological Interactions during Fish Early Life*, Ed. by E. W. North, A. Gallego, and P. Petitgas. ICES Cooperative Research Report No. 295, 27–42.

Brickman, D., and P.C. Smith Lagrangian stochastic modeling in coastal oceanography. *J. Atmos. Ocean Tech.*, 19, 83–99, 2002.

Chen, C., H. Liu, and R.C. Beardsley. An unstructured, finite-volume, three-dimensional, primitive equation ocean model: Application to coastal ocean and estuaries. *J. Atmos. Ocean. Tech.*, 20, 159–186, 2003.

Gillibrand, P.A., B. Siemering, P.I. Miller and K. Davidson. Individual-based modelling of the development and transport of a Karenia mikimotoi bloom on the North-west European continental shelf. *Harmful Algae*, 53,118–134, 2016.

Gillibrand, P.A. and K.J. Willis. Dispersal of sea lice larvae from salmon farms: A model study of the influence of environmental conditions and larval behaviour. *Aquatic Biology*, 1,63–75, 2007.

Hunter J.R., P.D. Craig and H.E. Phillips. On the use of random walk models with spatially variable diffusivity. *J. Comput. Phys.*, 106, 366–376, 1993.

Marine Scotland. The Scottish Shelf Model. *http://marine.gov.scot/themes/scottish-shelf-model*

North, E.W., R.R. Hood, S.-Y. Chao and L.P. Sanford. Using a random displacement model to simulate turbulent particle motion in a baroclinic frontal zone: A new implementation scheme and model performance tests. *J. Mar. Sys.*, 60, 365–380, 2006.

Okubo, A. Ocean diffusion diagrams. *Deep Sea Research*, 18, 789–802, 1971.

Proctor, R., R.A. Flather and A.J. Elliott Modelling tides and surface drift in the Arabian Gulf—application to the Gulf oil spill. *Continental Shelf Research*, 14, 531–545, 1994.

Ross, O.N. and J. Sharples. Recipe for 1-D Lagrangian particle tracking models in space-varying diffusivity. *Limnology and Oceanography: Methods*, 2, 289–302, 2004.

Sanford, L.P. Modeling a dynamically varying mixed sediment bed with erosion, deposition, bioturbation, consolidation and armoring. *Computers & Geosciences*, 34, 1263–1283, 2008.

Samgorinsky, J. General circulation experiments with the primitive equations. *Monthly Weather Review*, 91 (3), 99–152, 1963.

Visser, A.W. Using random walk models to simulate the vertical distribution of particles in a turbulent water column. *Mar. Ecol. Prog. Ser.*, 158, 275–281, 1997.

Walters, R.A. Design considerations for a finite element coastal ocean model. *Ocean Model.*, 15, 90–100, 2006.

Walters, R.A. and Casulli, V. A robust, finite element model for hydrostatic surface water flows. *Comm. Numer. Meth. Eng.*, 14,931–940, 1998.

Walters, R.A., Gillibrand, P.A., Bell, R., Lane, E.M. A Study of Tides and Currents in Cook Strait, New Zealand. *Ocean Dyn.*, 60,1559-1580, 2010.

Willis, K.J, P.A. Gillibrand, C.J. Cromey and K.D. Black. Sea lice treatments on salmon farms have no adverse effect on zooplankton communities: A case study. *Marine Pollution Bulletin*, 50, 806 – 816, 2005.