HAL: Hydrodynamics-based **A**lgorithm for **L**agrangian simulations

1. Introduction

HAL is a Lagrangian model for calculating the transport of zooplankton such as copepods, the egg and larval phases of marine fish, and the early life stages of crustaceans such as krill and lobsters. In addition to passive transport, it includes various options for more complex representations of species-specific biology:

- growth and development, based on temperature-dependent relationships, with default values for plaice
- egg buoyancy, specified as either a vertical velocity or as a particle density, from which a vertical velocity is calculated based on a comparison of the particle and local water densities
- diel vertical migration (DVM), specified as either nocturnal or diurnal (reverse) upward migration
- directed horizontal swimming, relative to currents or bathymetric gradients
- transport with sea ice
- maximum and minimum depths of vertical distribution
- seasonal diapause behaviour with separate vertical distribution limits for the surface and diapause phases

In addition, the model has been adapted for the modelling of the fate of oil spills and includes options for buoyant oil droplets and beaching. If no behaviours are chosen, the model simulates the passive transport of any neutrally-buoyant particle.

1.1 Numerical methods

The movement of particles within HAL is calculated using supplied 3D (with an option for 2D horizontal) velocity fields from an ocean circulation model. HAL assumes horizontal velocities are oriented eastward and northward and vertical velocities are true vertical; velocities from model grids with an alternative configuration must be rotated by pre-processing. The distance each particle travels is calculated using a 2nd order Runge-Kutta scheme with a user-supplied timestep; the latter should be carefully chosen to satisfy the horizontal and vertical Courant-Friedrichs-Lewy (CFL) stability criteria. Specifically, the timestep must be short enough such that a particle cannot 'skip over' grid cells in the horizontal, or grid levels in the vertical. For example, considering horizontal motion, an upper limit on the timestep can be estimated with *dt < min(dx)/max(U),* where *dt* is the model timestep, *min(dx)* is the minimum horizontal grid spacing (in m), and *max(U)* is the maximum horizontal current speed (in m s^{-1}).

Forcing ocean model data are generally stored at coarser temporal resolution (e.g. 5-day means) than the HAL timestep, and are gridded spatially, so temporal and spatial interpolation of water properties (e.g. currents, temperature, salinity) to the model timestep and particle locations is required. At HAL model timesteps between those of the ocean forcing data files, water properties are linearly interpolated to the model time from the two nearest ocean model outputs. Forcing data are spatially interpolated to the particle positions using trilinear interpolation.

In addition to advection, HAL includes an option to add random horizontal and vertical diffusion to particle movement, which allows representation of sub-grid scale turbulent motion and/or random swimming. This is simulated using a random-walk approach, detailed by Dyke (2001). HAL includes an option to input spatially-varying diffusion coefficients from the oceanographic model (POLCOMS and POM only). However, the default option comprises an assumption of isotropic diffusion with user-supplied horizontal and vertical diffusion coefficients. Appropriate diffusion coefficients should be chosen by considering observed diffusive processes in the model region, and/or the empirical length scales detailed by Okubo (1971).

2. Downloading and installing HAL

The HAL software is stored in a Github repository at https://github.com/emmafyoung/HAL. In addition to the source code in directory /*Source*, example input files and Matlab code for generating input files are provided in */Inputfiles,* basic Matlab programs for visualisation of the model output are included in directory */Postprocessing* (note, the m_map package is required), and an example setup for submitting a model simulation is provided in */Root*. Large input files required to run the test case *SGtest* are available at the linked Open Science Framework (OSF) project (DOI 10.17605/OSF.IO/SZ2PQ). HAL is written in FORTRAN and is designed to run in a Linux environment as a serial job on a dedicated processor; the following instructions assume a basic understanding of Linux commands.

- Create a home directory *HAL*
- Download the directory *Source* to your home directory
- Load Intel compilers and netCDF and HDF5 libraries; the current version of HAL is tested with the following:
  - netcdf/intel/4.4.1.1
  - hdf5/intel/1.8.19
  - intel/oneapi/2021.1.0.2659
- Navigate to the *Source* directory and compile the code with the command *make*; the resulting executable is called *ibm*.
- Note, if you edit the source code, it will need recompiling. If so, first use the command '*./make_clean*' in the *Source* directory to remove all existing object files and executables before recompiling.

2.1 The run environment

Before model experiments with HAL can be run, the necessary directory structure and input files must be prepared. The input files are described in detail in section 3; this section summarises the directory structure and locations of key files. Throughout this document, * is used where it is common to append a run-specific identifier to a filename.

The required directory structure is shown in Figure 1. The root directory is *HAL*, within which are 3 subdirectories, *Source*, *Inputfiles* and *Outputfiles*. The root directory also contains the executable (*ibm*), a *filenames.** file containing the names of all input and output files, and a script for running the program. An example script for running HAL on a high-performance computing system with a slurm job manager is provided (*slurm_runibm.SGtest*).

The *Source* directory contains all the program Fortran code, plus an included file *comblk_ibm.h*, and the *makefile*, and *make_clean* files mentioned in the previous section. HAL is compiled from within this directory; the resultant executable (*ibm*) should be copied to the root directory. The *comblk_ibm.h* file contains flags for turning some options on/off and information on the underpinning ocean model. If any changes are made to this file, HAL needs to be recompiled.

The *Inputfiles* directory contains model grid information files from the forcing ocean model, a *forcingfiles.** file listing the names of the ocean forcing data files (including the directory path), and three input files specific to each experiment (*model_spec.dat.**, *particle_spec.dat.** and *biology_spec.dat.**), which are described in more detail in section 3. The *Inputfiles* directory should be created and populated with relevant files after installing and compiling HAL. Example input files are provided in the Github repository with large files (e.g. oceanographic forcing data) available through the linked OSF project (DOI 10.17605/OSF.IO/SZ2PQ).

The *Outputfiles* directory should be created after installing and compiling HAL. This is where HAL writes the output file; this file is created automatically by the program and cannot be overwritten by the program. A new output file should be specified in the *filenames.** file for each model experiment.; if the output file already exists the program will stop.

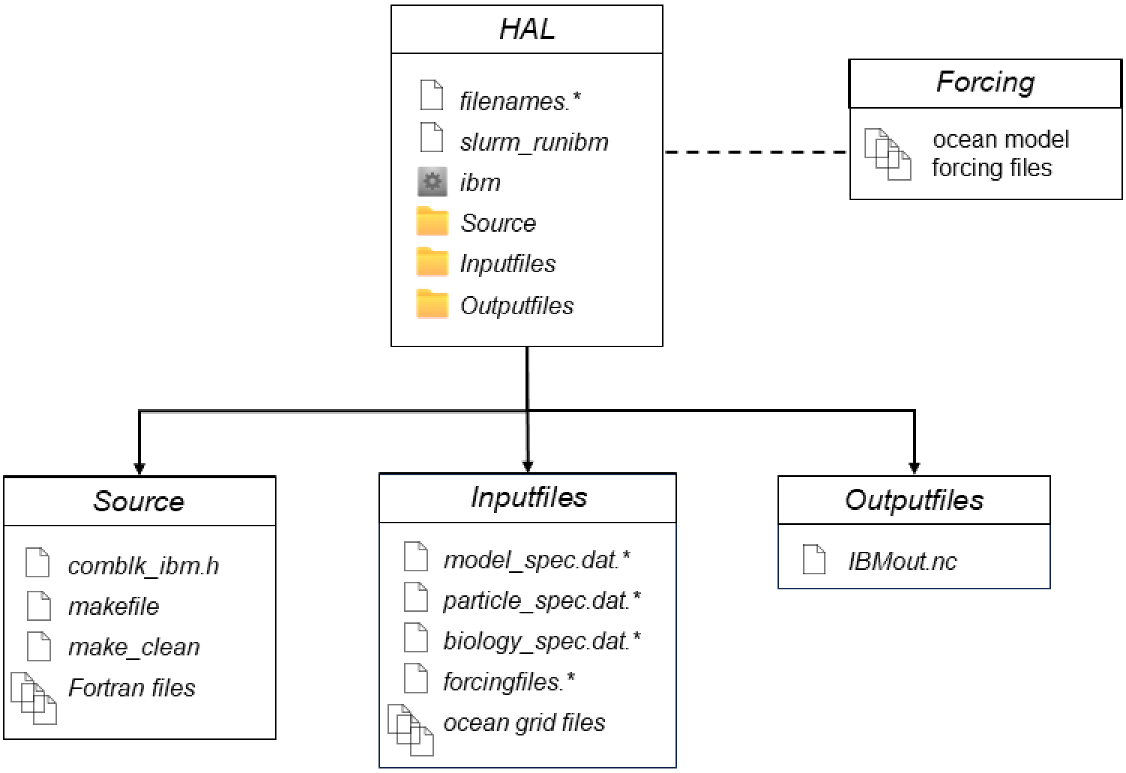

Figure 1: Diagram illustrating the required directory structure and location of key files for running HAL. The dashed line indicates a path to the directory where ocean model output data are stored.

3. Input files

3.1 Forcing data and associated files

3.1.1 *forcingfiles.**

This file lists the names of the input oceanographic forcing fields to be used, in chronological order. An annotated example is shown in Appendix 1. Only the root of the filenames is included (including the directory path); HAL automatically appends this with the relevant variable identifier (i.e. U, V, W, T, I). The representative date of the first data file (e.g. mid-point of a 5-day mean field) should match the date specified in the file *model_spec.dat.** (see section 3.3.1). HAL includes an option for 'warm starts', which allows a long model simulation to be split into multiple legs if computer resources are limited. If using warm starts, the relevant leg number is specified in this file to allow HAL to correctly adjust the model time; if no warm starts are required the leg number is set to 1.

3.1.2 Ocean model grid files

These files contain information on the grid of the oceanographic model used to create the forcing fields. HAL includes options to read grid information and forcing fields from 4 different source oceanographic models: NEMO, POLCOMS, POM, and OCCAM. The latter 3 models are rarely used and are not detailed in this documentation; for information on how to run HAL with these models please contact the author (Emma Young; eyoung@bas.ac.uk).

For NEMO-generated fields, 3 input files are required:

a. *mesh_hgr.nc*: horizontal grid information
b. *mesh_zgr.nc*: vertical grid information
c. *mask.nc*: masks for land and bathymetry

NEMO versions more recent than v4.0 only produce one mesh_mask.nc file when the oceanographic model grid is generated; in this scenario, 3 copies of the mesh_mask.nc file should be created and renamed as above.

3.1.3 Ocean model forcing files

The names of the oceanographic forcing fields to be used to drive HAL are specified in the *forcingfiles.** file. NEMO produces multiple output netcdf files, each containing different variables; the specific files required for each simulation depends on the options set in the included *comblk_ibm.h* file (section 3.2.1). All simulations require the 3 files containing the horizontal and vertical components of velocity (*U*, *V*, *W*). A file containing ocean temperature and salinity (*T*) is required if temperature- or density-dependent behaviours are included in the simulation, and a file containing sea-ice information (*I*) is required if sea ice-dependent behaviour is included. These forcing files should be stored in a directory accessible from the processor on which the model is to be run, with the full directory path included in the *forcingfiles.** file.

3.2 Pre-compilation input file

3.2.1 *comblk_ibm.h*

The *comblk_ibm.h* file is an included file required for compilation of HAL and is located in the */Source* directory. It defines scalars, variables, arrays and common blocks used by the program, and sets various logicals and array sizes. This file would need editing if an alternative oceanographic source model was to be used, or if additional forcing files (*T* or *I*) or alternative model physics were required for the choices of particle behaviour. Available options are located in the second block of the *comblk_ibm.h* file, as illustrated in Appendix 2.

3.3 Other input files

3.3.1 ***model_spec.dat.****

This file sets various model parameters, including horizontal and vertical diffusivities, integration time step, output frequency, information on the input flow fields, and logicals to specify how initial particle information will be defined (and hence, the format of input file *particle_spec.dat*). An example for a standard set-up is shown in Appendix 3, together with information on available options.

3.3.2 ***particle_spec.dat.****

This file defines the initial distribution and release times of particles to be simulated. Each particle is also assigned a 'weight', which is the number of individuals each particle represents (i.e. each particle may be considered as representing a cohort of individuals: a 'super-individual'). The weight term is only used if mortality is included in the simulation. HAL does not remove particles through mortality but instead reduces the particle weight. This allows the model to capture all potential transport pathways, including those with small numbers of survivors, which is important for studies of genetic connectivity. The format of *particle_spec.dat.** depends on the logicals set in the *model_spec.dat.** file; two examples are shown in Appendix 4. Example A is for the default options in Appendix 3 and is appropriate for the release of multiple particles with individually-specified release locations and times. With multiple particles, this file can become very large and is best created using an external program such as the example Matlab program provided in */Inputfiles* (see section 5). Example B in Appendix 4 is for multiple releases of particles in an ellipse formation around a central point, with the particle depths and the ellipse characteristics specified in the *model_spec.dat.** file.

3.3.3 ***biology_spec.dat.****

This file sets the biological options and parameterisations, including stage-dependent growth, mortality, egg buoyancy, directed swimming, and vertical migration. An example for a standard set-up for passive particles is shown in Appendix 5, together with information on available options. There are two key variables that define the species/behaviour attributes of the particles:

species: this variable has two options: *passive* or *active*. If it is set to *passive*, particles are assumed to have no behaviour and are advected by the 3-dimensional velocity fields as passive particles (with diffusion if non-zero values for *Ah* and/or *Az* are set in *model_spec.dat*). If *species* is set to *active*, particles can be assigned different behavioural characteristics according to the options detailed in Appendix 5 and summarised below.

stage_spec: this variable defines the species or life stage that is represented by each model particle and determines the available behaviour options. There are five available options for *stage_spec*: *e* (fish eggs), *l* (fish larvae), *k* (krill), *c* (copepods), and *n* (none – no species-specific behaviour). The available particle behaviours for each of these, and the associated adjustable parameters, are listed in table 1.

stage_spec	*Available behaviours*	*Adjustable parameters*
e	mortalityYN growthYN buoytyp	- hatchlength buoyancy, eggdiam
l	mortalityYN growthYN migrationYN swimming	- settlelength VMtyp, VMvel, buoyancy swim_base
k	migrationYN swimming	VMtyp, VMvel swim_base
c	diapauseYN	diapause_cue, dia_down, dia_up, VMvel
n	-	-

Table 1: Available behaviours and associated adjustable parameters for the range of species/life stages in HAL.

In addition to the parameters listed in table 1, for options *e* and *l*, the variable *biocalcfreq*, which specifies the frequency of updating computationally-demanding biological processes (e.g. growth and mortality), can be set to a longer timestep (e.g. daily) than the model timestep to improve model efficiency.

For all options, *mind* and *maxd* specify the depth range to which particle depths are restricted. If modelling krill (option *k*), the maximum depth becomes *maxdi* if *Iopt* in the *comblk_ibm.h* file is set to true. If modelling copepods (option *c*), *mind* and *maxd* specify the depth range of particles during the surface phase; once in diapause, the depth range becomes *mind_dia* and *maxd_dia*.

3.3.4 *filenames.**

This file specifies the names and locations of the input and output files. If the restart option is selected in *model_spec.dat*, a *filenames.** file is required for each leg of the simulation. An annotated example is shown in Appendix 6.

4. Output files

4.1 *IBMout.**

This is the main output file, usually written in NetCDF format, containing lists of simulated particle locations, depths and particle information (e.g. on/off flag used to define if a particle is still ‘live’) at the output frequency specified in *model_spec.dat*.

4.2 *restart.**

This is the restart file produced at the end of each simulation leg, and used to initialise the next leg in multi-leg runs.

5. Creating the input files

Most of the required input files can be created by editing the existing examples. However, *particle_spec.dat*, which provides the particle release information, can be a large file, and is more easily generated using an external program. An example Matlab script that generates a *particle_spec.dat* file for particle releases within a specified area of the South Georgia shelf, *createIBMstartpts_SG_example.m,* is provided in */Inputfiles*. This program generates particle release locations based on a consideration of a defined area and depth range within the context of the bathymetry of the source oceanographic model. The program generates an input file with the format described in Appendix 4 and produces a plot showing the particle release points.

6. Run procedure

The maximum number of particles that can be simulated is limited by the available computer resources and will require assessment on different architectures; for example, no more than 250,000 is recommended for the current British Antarctic Survey (BAS) high performance computing (HPC) facility.

1) Complete the download and installation procedure, and set up the environment structure, as described in section 2.

2) Check all input files are in the correct directories and edit as required to ensure they are appropriate for the new application.

3) If you have edited the source code or *comblk_ibm.h* file, recompile the model code: first use the command ‘*./make_clean*’ in the *Source* directory to remove all existing object files and executables before recompiling using the command *make*. Copy the resulting executable (*ibm*) to the root directory.

4) Submit the model run to an appropriate processor or hpc queue using the relevant job scheduling system. For example, for a system using slurm, edit the supplied slurm script (*slurm_runibm.SGtest)* and use the command ***sbatch slurm_runibm.SGtest*** from within the root directory.

7. Postprocessing

Two example Matlab programs are provided in directory /*Postprocessing* for displaying the output of the HAL test simulation.

7.1 *plot_st_end_SGtest.m*

This program plots particle locations at pre-defined times from the example SGtest run. It assumes that HAL was run as one leg (no restarts), with source flow fields from the South Georgia regional application of NEMO (https://github.com/emmafyoung/South-Georgia) available at the linked OSF project (DOI 10.17605/OSF.IO/SZ2PQ). The program calculates the locations of the particles at a set time after release (variable *nplank*; set to 90 days in the example). It then produces a plot with particle start points marked as red filled circles, and their locations after *nplank* days as blue filled circles. The underlying bathymetry is from the source oceanographic model.

7.2 *plot_traject_SGtest.m*

This program plots particle trajectories from the example SGtest run. It assumes that HAL was run as one leg (no restarts), with source flow fields from the South Georgia regional application of NEMO (https://github.com/emmafyoung/South-Georgia). The program plots particle trajectories for a specified period after release (*nplank*: set to 180 days in the example) overlaid on the source oceanographic model bathymetry. If variable *linecol* is set to 0, particle tracks are all plotted in red; if *linecol* is set to 1, each particle trajectory is a different colour.

References

Dyke, P. (2001) Coastal and shelf sea modelling. Kluwer Academic, Boston, MA.

Okubo, A. (1971). Oceanic diffusion diagrams. *Deep-Sea Research and Oceanographic Abstracts*, *18*(8), 789–802. https://doi.org/10.1016/0011-7471(71)90046-5

Appendix 1

Example *forcingfiles.** file

```
2                                              ! Number of forcing file groups to read
/Forcing/SG_5d_20000101_20001231_grid_         ! Group 1
/Forcing/SG_5d_20010101_20011231_grid_         ! Group 2
1                                              ! Leg number
```

Each group of forcing files corresponds to the same output time period of the forcing oceanographic model and may include a single temporal output (e.g. one 5-day mean) or multiple outputs (e.g. a year of 5-day means). The temporal characteristics of the forcing data are specified in *model_spec.dat* (Appendix 3).

The specific files in each forcing group depends on the chosen model options. At a minimum, it includes horizontal and vertical velocities (U, V, W); additional files may include temperature and salinity (T), and sea ice (I).

In this example, the forcing data are stored in linked directory *Forcing* and the filenames of group 1 are:

SG_5d_20000101_20001231_grid_U
SG_5d_20000101_20001231_grid_V
etc.

Note: when running a simulation with multiple legs (see section 3.1.1), it is common practice to design the simulation such that the length of each leg matches the time period encompassed by each forcing data file group. Then, nearly all the required oceanographic information is contained in the first velocity file. However, due to temporal interpolation within the model, the first set of data in the second group of files will also be required. This second group will then become Group 1 in the next leg of the simulation.

Appendix 2

Extract from an example *comblk_ibm.h* file illustrating key user-defined options with definitions below.

```
C***********************************************************************
C
C       Set size and components of model here:
C
        parameter
     $ (im=380, jm=400, kb=75)
C
        parameter
     $ (mx2=im*jm, maxp=250000, nlim=1, maxterms=4,
     $ nstage=10, mx2u=im*jm, curvi=2)
C
        parameter
     $ (polcoms=.false., occam=.false., pom=.false., nemo=.true.,
     $ circumpolar=.false., slevel=.false., newname=.false.,
     $ TSopt=.false., Iopt=.false., beaching=.false.,
     $ oil=.false.)
C
C***********************************************************************
```

- The dimensions of the oceanographic model grid are defined by *im* (number of columns), *jm* (number of rows), and *kb* (number of vertical levels).
- The maximum number of particles that can be used in the simulation is set with *maxp*.
- *nlim* and *maxterms* should only be adjusted if the POM model is used.
- *nstage* is the maximum number of development stages for fish eggs and larvae.
- The value chosen for *curvi* depends on the type of oceanographic model grid and impacts the internal numerics of HAL:
  - 0 for a regular Cartesian grid.
  - 1 for a curvilinear spherical grid, with irregular spacing in both latitude and longitude.
  - 2 for a curvilinear spherical grid, with regular grid spacing in longitude and either regular or irregular spacing in latitude.

  For NEMO, option 1 or 2 (default) should be used.
- Specify the type of forcing oceanographic model by setting one of *polcoms*, *occam*, *pom*, *nemo* to true.
- Set *circumpolar* to true if using a circumpolar forcing extract from the global 1/4° NEMO model. Note, this option does not currently work for an extract from any other circumpolar model.
- *slevel* is only relevant to the POLCOMS model; if set to true the model uses s-coordinates in the vertical instead of sigma coordinates.
- *newname* should be set to true if using output from NEMO v4.0 or more recent.
- Set *TSopt* to true if ocean temperature and salinity forcing data are required.
- Set *Iopt* to true if sea ice forcing data are required.

- Set *beaching* to true if particles should beach when they hit the coast (meaning their trajectories are not simulated beyond that timestep), otherwise particles are reflected away from the coast and their transport continues to be simulated.
- Set *oil* to true if particles should be treated as oil droplets. Note, simulating oil droplets also requires appropriate options to be chosen in *biology_spec.dat* (see Appendix 5).

Appendix 3

Example *model_spec.dat* file

1800	! Output frequency of simulated particle locations, *Outdt* (seconds)
5.000	! Horizontal diffusion coefficient, *Ah* (m^2 s^{-1}; negative if using imported model fields (only POLCOMS or POM))
0.00010	! Vertical diffusion coefficient, *Az* (m^2 s^{-1}; negative if using imported model fields (only available for POLCOMS or POM))
-13	! Random number seed, *Iseed*
300	! HAL model timestep, *Dt* (seconds)
432000	! Source oceanographic field temporal resolution, *Dt2* (seconds)
3 12.0	! Start time (day of year and hours) of first velocity file, *Start_d*, *Start_h* (note, this should be the temporal mid-point of the first record for time-averaged forcing data)
365	! Time period of source oceanographic data files (days), *slength*
n	! Type of velocities used, *veltyp* (*y* = depth-mean, *n* = 3D)
n	! Option to start from a restart file if running the simulation as a series of legs (*y* = restart option)
y	! Method of specifying particle locations, *Named* (*n* = ellipse formulation using parameters below, *y* = all particle locations individually specified in *particle_spec.dat.**)
y	! Method of specifying particle release times, *Namet* (*n* = all released at same time, *y* = all particle release times individually specified in *particle_spec.dat.**)
y	! Method of specifying particle depths, *Nameh* (*n* = all released at same depth (*deppart* below), *y* = all initial particle depths individually specified in *particle_spec.dat.**)
50.000	! Specified start depth of particles (m), *deppart* (particle depths if *Nameh* = *n*)
1	! Number of release periods; in a staggered release scenario particles will be released this number of times; this number must match the number of release events in *particle_spec.dat.**.
5.0 5.0 45.0	! Ellipse parameters, *aell* (semi-major axis length, km), *bell* (semi-minor axis length, km), *dell* (orientation with respect to east, positive anticlockwise, degrees). Only used if *Named* = *n*.

Note, HAL reads this input file using a formatted read statement, so any changes must preserve the formatting of the example file provided (*model_spec.dat.SGtest*).

Appendix 4

Example *particle_spec.dat* files

(A) All particles individually specified

Format appropriate to choices of *Named* = *y*, *Namet* = *y*, *Nameh* = *y* and a single release period set in *model_spec.dat* file. For multiple release periods, the following block of particle information is repeated with incremental increases in the release period number.

```
1                    ! Release period number
3                    ! Number of particles in this release period
-55 28.500000        ! Particle 1: Initial particle location, latitude (degrees, minutes)
-35 39.000000        ! Particle 1: Initial particle location, longitude (degrees, minutes)
3  12.000000         ! Particle 1: Particle release time (day of year and decimal hours)
183 12.000000        ! Particle 1: Time at which to end particle tracking (day of year and
                     decimal hours)
50.000000            ! Particle 1: Initial particle depth (m)
10                   ! Particle 1: Particle weighting (non-dimensional)
-55 28.500000        ! Particle 2: Initial particle location, latitude (degrees, minutes)
-35 36.000000        ! Particle 2: Initial particle location, longitude (degrees, minutes)
3 12.000000          ! Particle 2: Particle release time (day of year and decimal hours)
183 12.000000        ! Particle 2: Time at which to end particle tracking (day of year and
                     decimal hours)
50.000000            ! Particle 2: Initial particle depth (m)
10                   ! Particle 2: Particle weighting (non-dimensional)
-55 28.500000        ! Particle 3: as above
-35 33.000000
3 12.000000
183 12.000000
50.000000
10
```

(B) Particles released in an ellipse with multiple release periods

Format appropriate to choices of *Named* = *n, Namet* = *n, Nameh* = *n* and 3 release periods in *model_spec.dat* file.

```
1                    ! Release number 1
1000                 ! Number of particles in this release period
-55 28.500000        ! Centre of ellipse latitude (degrees, minutes)
-35 39.000000        ! Centre of ellipse longitude (degrees, minutes)
31 12.000000         ! Release time of all particles in this release (day of year and decimal
                     hours)
51 12.000000         ! Time at which to end tracking of all particles in this release (day of
                     year and decimal hours)
10                   ! Particle weighting (non-dimensional)
2                    ! Release number 2
```

```
1000                ! Number of particles in this release period
-55 28.500000       ! Centre of ellipse latitude (degrees, minutes)
-35 39.000000       ! Centre of ellipse longitude (degrees, minutes)
32 12.000000        ! Release time of all particles in this release (day of year and decimal
                    hours)
52 12.000000        ! Time at which to end tracking of all particles in this release (day of
                    year and decimal hours)
10                  ! Particle weighting (non-dimensional)
3                   ! Release number 3
1000                ! As above
-55 28.500000
-35 39.000000
33 12.000000
53 12.000000
10
```

Appendix 5

Example *biology_spec.dat* file

The following parameter choices are for simulating the transport of passive neutrally-buoyant particles in the upper 100 m of the water column.

1	! Number of early life stages (*true_no_stages*); multiple stages only used for egg and larvae options
passive	! Species (*species*: *passive* or *active*)
n	! Temperature dependent growth (*growthYN*)
n	! Temperature dependent mortality (*mortalityYN*)
n	! Vertical migration (*migrationYN*)
n	! Seasonal vertical migration (diapause: *diapauseYN*)
n	! Directed horizontal swimming (*swimming*: *n* for none, *r* for relative to background ocean velocity, *a* for absolute speed)
1.00	! Baseline swim speed (*swim_base*) (cm/s (absolute swimming) or proportion of passive speed (0 to 1; relative swimming))
8.0	! Length of larvae at hatching in mm (*hatchlength*)
11.0	! Length of larvae at settlement in mm (*settlelength*)
1.00	! Minimum particle depth in m (*mind*); particles are restricted to the depth range *mind* to *maxd* (*maxdi* under ice)
100.00	! Maximum particle depth in m (*maxd*)
30.00	! Maximum particle depth under ice in m (*maxdi*)
500.00	! Minimum particle depth during diapause in m (*mind_dia*); particles are restricted to the depth range *mind_dia* to *maxd_dia* during diapause
750.00	! Maximum particle depth during diapause in m (*maxd_dia*)
n	! Type of distribution at depth (*dia_distr*: *n* for random (default), *y* for weighted sin^2 distribution)
1	! Stage number for egg/larval development (*istage*); single stage only if *growthYN*=*n*. *Following lines are duplicated for multiple stages.
n	! Species/life stage (*stage_spec*: *e* for egg, *l* for larvae, *k* for krill, *c* for copepods, *n* for none)
1825.00	! Stage duration at 10C in days (*stagelen*); set to longer than the model run length unless using multiple stages with *stage_spec* set to *e* or *l* and *growthYN* set to *y*.
1800	! Frequency of biological updates in seconds (*biocalcfreq*) – this can be less frequent than the physical transport calculation
v	! Buoyancy calculation type (*buoytyp*: *v*=fixed buoyant velocity, *d*=density, *c*=constant depth)
0.00500	! Value for calculation of egg/oil buoyancy and non-DVM larval vertical movement (*buoyancy*: density (g/cm^3) or velocity (cm/s), positive is upwards)
1.50	! Egg/oil droplet diameter for buoyancy calc (mm) (*eggdiam*)
d	! Type of DVM (*VMtyp*: *d*=surface during day, *n*=surface at night)
0.00500	! Vertical migration speed in cm/s (for either DVM or seasonal migration: *VMvel*)
j	! Cue for diapause start and end dates (*diapause_cue*: *j* for day of year, *d* for daylength (in hours))

45.00	! day of year or daylight hours for start of diapause (descent; *dia_down*)
187.00	! day of year or daylight hours for end of diapause (ascent; *dia_up*)

Note, HAL reads this input file using a formatted read statement, so any changes must preserve the formatting of the example file provided (*biology_spec.dat.SGtest*).

Example parameter choices for selected applications

(A) Mackerel icefish larvae

Young, E.F. et al. (2012). ‘Physical and behavioural influences on larval fish retention: contrasting patterns in two Antarctic fishes’. Marine Ecology Progress Series 465, 201-215, https://doi.org/10.3354/meps09908.

The following parameter choices are relevant for the modelling of late-stage mackerel icefish larvae with diel vertical migration. If not explicitly defined, values are as in the default *biology_spec.dat* example above.

species: active
migrationYN: y
stage_spec: l
biocalcfreq: 300
VMvel: 0.5

(B) Antarctic krill

Young, E.F. et al. (2024). ‘Environmental and behavioural drivers of Antarctic krill distribution at the South Orkney Islands: a regional perspective’. Journal of Marine Systems, 241, https://doi.org/10.1016/j.jmarsys.2023.103920.

The following parameter choices are relevant for the modelling of krill with sea ice-associated behaviour, and with diel vertical migration or modified diel vertical migration under sea ice. Note, simulations with sea ice-associated behaviour also require Iopt to be set to true in the *comblk_ibm.h* file. If not explicitly defined, values are as in the default *biology_spec.dat* example above.

species: active
migrationYN: y
stage_spec: k
biocalcfreq: 300
**VMtyp*: n for DVM , y for modified DVM under sea ice

VMvel: 0.5

(C) Oil spill

Bell, J. et al. (2021). ‘Gough Island: Environmental Desk Based Risk Assessment: MFV Geo Searcher wreck, sank October 2020’. Blue Belt Technical Report CR166.

The following parameter choices are illustrative; buoyancy and droplet diameter (eggdiam) should be chosen to best represent the oil characteristics. Note, oil and TSopt should also be set to true in the *comblk_ibm.h* file. If not explicitly defined, values are as in the default *biology_spec.dat* example above.

species: active
mind: 0
maxd: 500
stage_spec: e
biocalcfreq: 300
buoytyp: d
buoyancy: 0.843
eggdiam: 1.0

Appendix 6

Example *filenames.** file

Description

The *filenames.** files specify the location (directory), number, and names of all the input and output files, and the format of the files. For a standard model run with oceanographic forcing from NEMO, there are 7 input files for a single leg simulation or the first leg of a multi-leg run (no restart file; 8 input files for subsequent legs with a restart file), and 2 output files. The numbers before each file name indicate the file format, as follows:

‘0’: an unformatted file in the input or output directory specified immediately below ‘INPUT’ or ‘OUTPUT’
‘1’: a formatted file in the input or output directory specified immediately below ‘INPUT’ or ‘OUTPUT’
‘2’: a formatted file in the directory from which the program is run
‘3’: an unformatted file in the directory from which the program is run
‘4’: a netCDF file in the input or output directory specified immediately below ‘INPUT’ or ‘OUTPUT’
‘5’: a netCDF file in the directory from which the program is run

Example file

```
INPUT                              ! Next section refers to input files
/Inputdir                          ! Full path of directory containing input files
7                                  ! Number of input files
1 model_spec.dat
1 particle_spec.dat
1 biology_spec.dat
1 forcingfiles.SGtest
4 SG_mesh_hgr.nc
4 SG_mesh_zgr.nc
4 SG_mask.nc
OUTPUT                             ! Next section refers to output files
2                                  ! Number of output files
/Outputdir                         ! Directory containing output files
 4 IBMout.nc
 2 restart.SGtest
```