MITberg V3.0c

A Dynamic-Thermodynamic Iceberg Model for MITgcm

USER MANUAL

Dr. Alan Condron

Woods Hole Oceanographic Institution

acondron@whoi.edu

Updated: Nov 2nd, 2018

1. INTRODUCTION

MITberg is a dynamic-thermodynamic iceberg model capable of simulating the melt and drift of 10,000's of icebergs in the ocean. The model is coded in parallel FORTRAN and is being coupled to the Massachusetts Institute of Technology general circulation model (MITgcm). MITberg can be used to study a variety of ice-related problems from iceberg forecasting, ice-ocean interactions, climate change, and iceberg towing. Icebergs are assumed to be rectangular, with a user defined width (*W*) to length (*L*) ratio (set by default) to 1:1.62, and drift with their long axis parallel to the direction of travel to reflect observations (e.g. Dowdeswell et al., 1992). As ice is less dense than seawater, ~8/9 of each iceberg is below the waterline. The subaqueous part is referred to as the *keel* and the keel's thickness as *draft* (*D*) while the subaerial portion is called the *sail* and the sails height above the sea surface, *freeboard* (*Fb*). In the model, the keel thickness and freeboard height are derived from the total iceberg thickness, *T*, and the ratio of the density of ice, ρ_i, to seawater, ρ_w (Table 1). Finally, icebergs that ground on the sea floor (when keel depth > sea floor depth) remain stationary until they melt sufficiently to begin drifting again.

1.1 Dynamics

Icebergs are considered as lagrangian particles with their horizontal acceleration calculated from the equation of motion for an iceberg (e.g. Robe, 1980; Smith et al 1993; Bigg et al., 1997) as:

$$m\frac{d\vec{\boldsymbol{v}}_i}{dt} = -mf\hat{z}\times\vec{\boldsymbol{v}}_i + \vec{\boldsymbol{F}}_a + \vec{\boldsymbol{F}}_w + \vec{\boldsymbol{F}}_s + \vec{\boldsymbol{F}}_r + \vec{\boldsymbol{F}}_p$$

where m is the mass of the iceberg, $\vec{\boldsymbol{v}}_i$ is iceberg velocity, *t* is time, and the six terms on the right-hand-side represent the various forces (units: kg.m.s^{-2}) exerted on each iceberg: the Coriolis force $-mf\hat{z}\times\vec{\boldsymbol{v}}_i$, where f is the Coriolis parameter and $\hat{z}$ is the vertical unit vector, $\vec{\boldsymbol{F}}_a$ the wind drag, $\vec{\boldsymbol{F}}_w$ the water drag, $\vec{\boldsymbol{F}}_s$ the sea ice drag, $\vec{\boldsymbol{F}}_r$ is wave radiation, and $\vec{\boldsymbol{F}}_p$ the horizontal pressure gradient.

The drag force produced by the wind acting on both the vertical side walls above the water line (form drag) and the horizontal surface plane (skin drag) of each icebergs are described (Smith and Banke, 1983) as:

$$\vec{\boldsymbol{F}}_a = \left[\frac{1}{2}\rho_a C_{av}A_{av} + \rho_a C_{ah}A_{ah}\right]|\vec{\boldsymbol{v}}_a - \vec{\boldsymbol{v}}_i|(\vec{\boldsymbol{v}}_a - \vec{\boldsymbol{v}}_i)$$

where ρ_a is air density, $\vec{\boldsymbol{v}}_a$ surface wind velocity, A_{av} and A_{av} are the vertical and horizontal cross sectional areas of the iceberg, and parameters C_{av} and C_{ah} are the form and skin drag coefficients, respectively (Table 1). The drag force from the ocean is calculated in a similar way but accounts for changes in ocean velocity with depth by summing the drag force at each vertical ocean model level the iceberg keel penetrates (Keghouche et al., 2009; Turnball et al., 2015) as:

$$\vec{\mathbf{F}}_{\mathrm{w}} = \sum_{i=1}^{n}\left[\frac{1}{2}\rho_w C_{wv}A_{wv}(i)|\vec{\boldsymbol{v}}_w(i) - \vec{\boldsymbol{v}}_i|(\vec{\boldsymbol{v}}_w(i) - \vec{\boldsymbol{v}}_i)\right] + \rho_w C_{wh}A_{wh}(n)|\vec{\boldsymbol{v}}_w(n) - \vec{\boldsymbol{v}}_i|(\vec{\boldsymbol{v}}_w(n) - \vec{\boldsymbol{v}}_i)$$

where i is the vertical ocean model level, $\vec{\boldsymbol{v}}_w$ *(i)* is the water velocity at each vertical model level, $A_{wv}(i)$ and A_{wh} *(n)* are the vertical and horizontal cross sectional areas of the iceberg at each model level and at the base of the iceberg, and parameters C_{wv} and C_{wh} are the vertical form drag and horizontal skin drag coefficients, respectively. While changes in ocean drag with depth are considered by default, the model does provides the user with the option to derive the total ocean drag force from only the surface ocean velocity (i=1) as in, for example, Bigg et al., (1997) and Martin and Adcroft, (2010).

The drag force exerted by sea ice acts only on the sidewalls of the iceberg, and only on the part of the keel that is in the surface level of the model, as:

$$\vec{\boldsymbol{F}}_s = \frac{1}{2}\rho_s C_{sv} L_\perp T_s |\vec{\boldsymbol{v}}_s - \vec{\boldsymbol{v}}_i|(\vec{\boldsymbol{v}}_s - \vec{\boldsymbol{v}}_i)$$

where ρ_s is the density of sea ice, C_{sv} is the sea ice form drag coefficient, $L_\perp$is the length of the iceberg normal to the stressing force at the surface level (i.e. width or length, where i=1), T_s the sea ice thickness, and $\vec{\boldsymbol{v}}_s$ is sea ice velocity. In addition, the sea ice drag force is only considered when the concentration of sea ice exceeds 15% and in high (>90%) concentrations of sea ice icebergs drift with the pack ice (i.e. $\vec{\boldsymbol{v}}_i = \vec{\boldsymbol{v}}_s$) until the ice concentration sufficiently reduces (Lichey and Hellmer, 2001; Keghouche et al., 2009; Morison and Goldberg, 2012).

The drag force from ocean waves is calculated as in Savage (1999) as:

$$\vec{\boldsymbol{F}}_r = \frac{1}{2}\rho_w C_r g a \min(a, Fb)\, L_\perp \frac{\vec{\boldsymbol{v}}_a}{|\vec{\boldsymbol{v}}_a|}$$

where C_r is the wave force coefficient and g is gravitational acceleration. The wave amplitude, *a,* is empirically related to wind speed and dependent on both sea ice area and freeboard height to avoid producing erroneously large wave drag forces (Gladstone et al., 2001; Martin and Adcroft, 2010) as

$$a = 0.010125|\vec{\boldsymbol{v}}_a - \vec{\boldsymbol{v}}_w|^2 \times \frac{1}{2}(1 + \cos[\pi {A_s}^3])$$

where A_s is the fractional area of the grid cell occupied by sea ice (0 – 1). Lastly, the pressure gradient force is calculated directly from the sea surface height, η, of the ocean model's nonlinear free surface as:

$$\vec{\boldsymbol{F}}_p = -mg\vec{\nabla}\eta$$

At this point in the development of MITberg, the model will not work without a nonlinear free surface.

1.2. Thermodynamics

Melting of icebergs occurs both above (subaerial) and below (subaqueous) the waterline. In the model, subaerial melt occurs due to radiative heating, sensible heating (i.e. forced convection), while melt below the waterline occurs due to forced convection, buoyant vertical convection*,* and wave erosion (as in El-Tahan et al., 1987; Savage et al., 2001; Kubat et al., 2007). As iceberg melt tends to occur laterally much faster than vertically, icebergs become unstable and roll-over. Consistent with Mugford and Doweswell (2010) icebergs are assumed to be unstable and roll onto their side when the length to thickness ratio is less than 0.7, (e.g. $L/T < 0.7$). In this case, L and T are instantaneously swapped.

Fresh water from melting icebergs is released into the surface layer of the MITgcm ocean model with a temperature of 0°C and 0 psu so that melting icebergs cool and freshen the ocean. Both the temperature and salinity of iceberg meltwater is set in data.iceberg.

2. SET-UP INSTRUCTIONS

As MITberg is not officially checked-in as a package for MITgcm you will need to add the following code to several scripts

In folder **/MITgcm/model/inc/** add the following code to **PARAMS.h**
add LOGICAL useICEBERG after LOGICAL useFLT

And then in COMMON /PARM_PACKAGES/ add useICEBERG to the list

In folder /**MITgcm/model/src/**
Add the following code to the following routines

do_oceanic_phys.F
After the call to SHELFICE add:

```
#ifdef ALLOW_ICEBERG
     IF (useICEBERG) THEN
      CALL TIMER_START('ICEBERG    [DO_OCEANIC_PHYS]', myThid)
      CALL ICEBERG_MODEL ( myTime, myIter, myThid)
      CALL TIMER_STOP ('ICEBERG    [DO_OCEANIC_PHYS]', myThid)
     ENDIF
#endif
```

do_the_model_io.F
After the call to SHELFICE add:

```
#ifdef ALLOW_ICEBERG
     IF ( useICEBERG ) THEN
       CALL ICEBERG_OUTPUT( myTime, myIter, myThid )
       CALL ICEBERG_WRITE_PICKUP ( myTime, myIter, myThid)
     ENDIF
#endif  /* ALLOW_ICEBERG */
```

external_forcing_surf.F
Near the top of the script under the "Global variables" add:

```
#ifdef ALLOW_ICEBERG
#include "ICEBERG.h"
#endif /* ALLOW_ICEBERG */
```

After **#ifdef ALLOW_SALT_PLUME** add:

```
#ifdef ALLOW_ICEBERG
      IF (useICEBERG) THEN
       CALL ICEBERG_FORCING_SURF(
     &        bi, bj, iMin, iMax, jMin, jMax,
     &        myTime, myIter, myThid)
      ENDIF
#endif /* ALLOW_ICEBERG */
```

packages_boot.F

Under "NAMELIST /PACKAGES/" add:

```
     &          useICEBERG,
```

Under "C-- Default package configuration" add

```
          useICEBERG      =.FALSE.
```

packages_init_fixed.F

```
#ifdef ALLOW_ICEBERG
      IF (useICEBERG) CALL ICEBERG_INIT_FIXED( myThid )
#endif
```

Note: I added this code after the call to the #ifdef ALLOW_FLT

packages_init_variables.F

```
#ifdef ALLOW_ICEBERG
      IF (useICEBERG ) THEN
       CALL ICEBERG_INIT_VARIA( myThid )
      ENDIF
#endif /* ALLOW_ICEBERG */
```

Again, I added this code after the call to #ifdef ALLOW_SHELFICE

packages_readparms.F

```
#ifdef ALLOW_ICEBERG
      IF (useICEBERG) CALL ICEBERG_READPARMS( myThid )
#endif /* ALLOW_ICEBERG */
```

This code was added after call to #ifdef ALLOW_SHELFICE

Once you've added the above bits of code, create a new folder in MITgcm/pkg called 'iceberg'. Copy the file iceberg.tar to this folder and untar it. You then need to copy file ICEBERG_OPTIONS.h to your verification/{EXPname}/code/ folder, i.e. the one where you'll be running your experiments. You can edit this file to turn on/off various aspects of the iceberg model such as multilevel drag, keel model, calving scheme etc. The file should be fairly self-explanatory.

Now add the line "iceberg" to your list of packages in code/packages.conf

In data.pkg you will need to write:

```
useICEBERG     = .TRUE.,
```

This will allow you to turn on/off MITberg once the model has compiled using "make"

Finally copy data.iceberg to the folder where you'll run your experiments.

You can now try compiling MITgcm.

3. RUNNING THE MODEL

First you must create a file with the initial positions, size, and velocity of the icebergs you wish to simulate. In my example, 'iceberg_locations.txt' there are 3 icebergs. You can have up to 15000 listed here. The file format is currently set as I6, I3, 6F9.2 with the 8 input fields as follows:

ID number	Face # (cube-sphere)	i-location	j-location	Width (m)	Thickness (m)	u-velocity (m.s-1)	v-velocity (m.s.-1)

The face number relates to the cube-sphere grid (1-6) where, for example, 3 is the Arctic face, the i and j positions refer to the grid point locations of your icebergs on the ocean model domain (not lon and lat), which if you are using the cube-sphere will be the i and j points of that particular face; the model then works out which tile each iceberg is located on if you're running in parallel using MPI. In the example included with the code the two icebergs are on face 3 of the cube-sphere grid, the first near the north pole and the second in the center of the Labrador Sea.

To enable MITberg to read this file set niter0=0 in 'data' with the name of this file specified in data.iceberg using 'IcebergLocationFile'. Make sure calving is turned off in code/ICEBERG_OPTIONS.h otherwise the iceberg model will look for the Calving file and ignore this one. If you wish to start your model runs from an ocean pickup rather than initial conditions but don't have an iceberg pickup file then you can set useIcebergPickup = .FALSE. so that the model will read in the IcebergLocationFile rather than an iceberg pickup.

3.1 data.iceberg

Many of the variables in data.iceberg deal with calving, and initially they can be commented out (as I've done in my example). Most of the parameters should be quite self-explanatory as they deal with density and iceberg drag coefficients. There are a couple of things in here to stop the model becoming unstable, one of which is 'min_size' which sets the minimum size an iceberg can be before it is assumed to melt. This is included to stop the model becoming unstable with very small icebergs. The other parameter is 'ibMaxV' which sets the maximum absolute velocity (ms^{-1}) that an iceberg can move at but this is more important in areas of high ocean velocity when using coarser grids and longer time steps. At the bottom of data.iceberg you will see a list a scalable parameters. Again these are probably not going to be all that useful but they can be used, for example, to locally increase wind speeds, air temperature etc at the iceberg location.

3.2 Iceberg model time step

The iceberg model timestep is set using 'deltaT_ice' in data.iceberg. Rather than being the actual timestep in seconds it is the number of times the iceberg code will loop through the dynamics routine (iceberg_adv.F) for every 1 ocean model time step. For example, if your ocean model time step is 300s, then setting 'deltaT_ice' to 10 would make the iceberg advection loop 10 times for every 1 model time step, so essentially the advection time step is 30 seconds (300/10), setting to 20 would make the iceberg adevection time step 15 seconds, etc. Note that the iceberg thermodynamics routine has the same time step as the ocean model.

3.3 Output data

MITberg writes two output files in binary. The main one is ICEBERG.XXXXXXXXXX (where X's refer to the model time). This file provides a snapshot of where all the icebergs in your domain are at any one time, how big they are, how fast they are moving etc. The file is written out at a user defined frequency (in seconds) by setting 'ib_write_freq' in data.iceberg. The file is binary, REAL8, and 15000 rows by 12 columns. The columns are:

ID	Tile	Face	i-pos	j-pos	Width	thickness	u-vel	v-vel	Flag	scale	source

ID:	identification number of the iceberg
Tile:	tile iceberg is located on (when using MPI)
Face:	Used only with cube-sphere configure which has 6 faces
i-pos:	x position on global model domain
j-pos:	y position on global model domain
width:	iceberg width (meters)
thickness:	iceberg thickness (meters)
u-vel:	iceberg u velocity (m s^{-1})
v-vel:	iceberg v velocity (m s^{-1})
Flag:	0 = iceberg freely drifting, 1 = drifting with sea ice, 2 = grounded on seafloor
Scale:	Used with calving scheme to allow 1 particle to represent N icebergs
Source:	Location/ice stream id number that iceberg originally calved from

The other main output file is ibFWflx.XXXXXXXXXX.data. This is a 2-dimensional time average of the volume of fresh water released to the ocean model from melting icebergs (units: kg m^{-2} s^{-1}). The file structure is binary (real*4) and has the same dimensions as the ocean model diagnostics written out (i.e. dimensions of the ocean model domain). The output frequency is set by changing 'IcebergTaveFreq' in data.iceberg.

3.4 Reading output files

Two simple example matlab routines are included to read in a single ICEBERG file and ibFWflx field, and called read_iceberg_data.M and read_FWforcing.F, respectively.

3.5 Pickup Files

Iceberg model pickup files are written out at the same interval as the ocean model (set as pChkptFreq in data). There are two pickup files written out at each interval: pickup_iceberg.XXXXXXXXXX.data and pickup_calving.XXXXXXXXXX.data, the latter only being written if the iceberg calving scheme is turned on. The files are both binary and real*8. Files labeled 'pickup_iceberg' are 15000 rows by 13 columns and are therefore quite similar in structure to the ICEBERG.XXXXXXXX files:

ID	Tile	Face	calve_slab_ counter	i-pos	j-pos	Width	thickness	u-vel	v-vel	Flag	scale	source

where calve_slab_counter is the number of seconds since a slab of overhanging ice was calved from an iceberg.

3.6 Finally, some things to be consider:

- Iceberg length (L) is always a multiple of iceberg width (W), and is user defined in data.iceberg. By default L = 1.62 x W.
- If iceberg keels touch bottom then iceberg velocity = 0
- Icebergs are not aware of each other. They will not collide and 2 icebergs could, in theory, occupy the same space.
- There is some code to allow icebergs to move along the coasts if they hit land (but are too small to ground on the sea floor). In this case the model will only allow the iceberg to move parallel or away from the coast. Turn this on in data.iceberg.
- If sea ice cover > 0.9 then icebergs drift with sea ice model.
- Meltwater is released to the surface layer of the ocean model.

4. MITBERG: MODEL DEVELOPMENT FOR THE FUTURE

- Allow icebergs to collide with fjord walls and each other and 'bounce off'.
- Release meltwater at different vertical levels in the ocean model.

- Allow user defined iceberg shapes based on observed iceberg profiles to be read in at niter0=0
- Add a dye tracer to iceberg meltwater to track the fresh water released from individual icebergs.

5. REFEREENCE

6. TABLES

Table 1: A list of the main dynamics coefficients used in the model.

Coefficient	Description	Units	Default Value
ρ_i	density of iceberg	kg. m^{-3}	917
ρ_w	density of water	kg.m^{-3}	1025
ρ_a	density of air	kg.m^{-3}	1.2
ρ_s	density of sea ice	kg.m^{-3}	910
C_{wv}	vertical drag coefficient for water	Dimensionless	1.0
C_{av}	vertical drag coefficient for air	Dimensionless	0.8
C_{iv}	vertical drag coefficient for sea ice	Dimensionless	1.0
C_{wh}	horizontal (skin) drag coefficient for water	Dimensionless	0.0012
C_{ah}	horizontal (skin) drag coefficient for air	Dimensionless	0.0055
g	Gravity	m.s^{-2}	9.8

Table 2: A list of the main coefficients used in the iceberg thermodynamics model.

Coefficient	Description	Units	Value
Γ_i	latent heat of fusion of ice	J kg^{-1}	$3.33\text{x}10^5$
α	Iceberg albedo	dimensionless	0.7
k_a	Thermal conductivity of air (at 10°C)	J s^{-1} m^{-1} K^{-1}	0.0249
k_w	Thermal conductivity of water (at 0°C)	J s^{-1} m^{-1} K^{-1}	0.563
v_a	kinematic viscosity of air (at 10°C)	m^2 s^{-1}	1.46 x 10^{-5}
v_w	kinematic viscosity of water (at 0°)	m^2 s^{-1}	1.83 x 10^{-6}
D_a	thermal diffusivity air (at 0°C)	m^2 s^{-1}	2.16 x 10^{-5}
D_w	thermal diffusivity water (at 0°C)	m^2 s^{-1}	1.37 x 10^{-7}
R	Roughness height of the iceberg	m	0.01
W_p	Wave period	s	10