IcePlume: a subglacial runoff parameterisation for MITgcm (documentation v03)

This package is described in the paper (and supplementary material):

Cowton T, D Slater, A Sole, D Goldberg, P Nienow (2015). Modeling the impact of glacial runoff on fjord circulation and submarine melt rate using a new subgrid-scale parameterization for glacial plumes. Journal of Geophysical Research - Oceans.

Overview

The purpose of this package is to permit the efficient incorporation of glaciers – and specifically glacial runoff – in simulations of fjord systems. The unusual feature of this system is that freshwater runoff enters the fjord at depth, rising as a buoyant plume adjacent to the glacier front. In the existing marine-terminating glacier package ('Icefront' – Xu *et al.* 2012, *Ann. Glac.*) this runoff is input to the model at depth, allowing this plume to be simulated in MITgcm. While advantageous for examining detailed circulation at the glacier front, this requires a high-resolution (c. 10m), non-hydrostatic simulation, which is not computationally practical for modelling fjord systems (which may be hundreds of square kilometres in area). For this reason, IcePlume instead uses a theoretical plume model to parameterise vertical transport in the plume adjacent to the glacier front. It is therefore suitable for use in coarser resolution and hydrostatic simulations, permitting larger domains and longer model runs.

Key routines

iceplume_calc.F - The main routine. This locates cells with runoff input, obtains profiles of ambient water properties, calls the plume appropriate plume model, calculates the vertical distribution of plume entrainment and outflow, calls the routine to calculate background meltrate, calculates the tendency and addmass terms, and sends information to the diagnostics fill routines.

iceplume_plume_model.F – This contains the main loop for the plume model and several subroutines describing the individual plumes models.

Inputs

*indicates more detail is given below

Option	Unit	Default	Description
ICEPLUME PARM01			
plumeMaskFile*	filename		xy mask of ice/plume types
runoffVelFile*	filename		xyt mask of runoff velocity
runoffRadFile*	filename		xyt mask of runoff thickness/radius
usePlumeDiagnostics*	Logical	.FALSE.	Option to write out plume properties as MITgcm diagnostics
conserveMass*	Logical	.FALSE.	Adjust plume outflow to prevent net addition of mass due to runoff and melting
backgroundVel*	ms^{-1}	0.001	Unresolved velocity at ice-ocean interface (appropriate range of 0 –

			0.1 m/s)
E_0*		0.1	Entrainment parameter in plume model.
T_sg	°C	0.001	Potential temperature of subglacial runoff.
S_sg	psu	0.001	Salinity of subglacial runoff.
iceTemp	°C	0	Temperature of ice in contact with ocean.
rho_ref	kg m^{-3}	1020	Reference density
g	m s^{-1}	9.81	Gravitational acceleration
c_w	J kg^{-1}°C^{-1}	3994	Heat capacity of water
c_i	J kg^{-1}°C^{-1}	2009	Heat capacity of ice
L	J kg^{-1}	334000	Latent heat of melting
lambda1	°C psu^{-1}	0.0573	Freezing point slope
lambda2	°C	0.0832	Freezing point offset
lambda3	°C m^{-1}	0.00076	Freezing point depth
GamT		0.022	Thermal turbulent transfer coefficient
GamS		0.00062	Salt turbulent transfer coefficient
Cd		0.0025	Ice-plume drag coefficient
RTOL		1×10^{-5}	Relative tolerance parameter (ODE solver)
ATOL		1×10^{-5}	Absolute tolerance parameter (ODE solver)
ICEPLUME_PARM02			
applyIcefrontTendT	Logical	.TRUE.	Option to cool cells in response to calculated melting
applyIcefrontTendS	Logical	.TRUE.	Option to freshen cells in response to calculated melting
ICEFRONTlatentHeat	J kg^{-1}	334000	As for ICEFRONT
ICEFRONTHeatCapacity_Cp		2000	As for ICEFRONT
ICEPLUME_PARM03			
useInputPtracers*	Logical	.FALSE.	Option to add a quantity of ptracer to the plume output, corresponding to the volume of subglacial runoff (requires pkg/ptracers)
ptracerMaskFile*	filename		Nx by ny by n_ptracers mask giving the concentration of a ptracer in the subglacial runoff in that location.

See Figure 1 for illustration of the input fields.

- *plumeMaskFile(nx,ny)*
  - 2D matrix giving the type and location of proglacial plumes in the domain (Figure 1)

- In each grid location, specify:
  - 0 = no glacier ice or runoff
  - 1 = vertical glacier ice (permitting melting), but no input of meltwater runoff
  - 2 = 'sheet plume' (Jenkins 2011)
  - 3 = 'half-conical plume' (Cowton et al 2015)
  - 4 = both sheet and half-conical plume (NOT IMPLEMENTED)
  - 5 = detaching conical plume (must be permitted in ICEPLUME_OPTIONS.h)
- Options 0-3 can be specified in different locations in the same domain. 4 is not presently available. 5 modifies the size of some fields, and so must be used in isolation (I think).
- In cases 1-5, it is assumed that there is a vertical wall of ice spanning the full depth of the water column and full width of the cell. Unlike in the ICEFRONT package, this does not act as a physical barrier to flow (the glacier should instead be defined as land in the model bathymetry, and adjacent cells masked appropriately using the values above). The purpose of the virtual ice wall in IcePlume is to permit the calculation of the melt rate, and to modify the temperature and salinity of the cell accordingly. To do this, it is necessary to define the orientation of the virtual ice front, so that the impact of horizontal currents on melt rate can be calculated. This is done by selecting the sign of the mask values – a positive value defines the icefront as being orientated along a north-south axis, while a negative value defines an east-west orientation.

- *runoffVelFile; runoffRadFile;*
  - Combined, these two files define the strength of the runoff input (Figure 1)
  - *runoffVelFile (nx,ny,nt)* – specify the velocity at which runoff enters the domain in any x,y location. In the case of the sheet plume (2) and half-conical plume (3), this velocity is by necessity in a vertical direction. In the case of the detatching plume, the direction of input can be specified (4). As for other forcings / boundary conditions in MITgcm, this forcing can change over time by specifying multiple time layers in the third dimension (see *periodicExternalForcing* in *data*, or the EXF package).
  - *runoffRadFile (nx,ny,nt)* – specify the radius or thickness of the runoff input. In the case of the half-conical or detatching plume, this value is the initial plume radius, and can be thought of as the radius of the subglacial channel. In the case of the sheet plume, specify the thickness of the initial sheet in the direction perpendicular the icefront (the width of the sheet parallel to the icefront is assumed to span the width of the cell). As for runoff velocity, this value can vary over time.
  - Except for the detatching plume, the exact values of velocity and radius are not important, so long as they combine to give the desired discharge. For example, it is convenient to set runoff velocity to 1 m/s, and modify the runoff radius so that runoff is equal to the discharge of meltwater from that channel.
  - Note that if the specified plume mask value in a location includes runoff (i.e. types 2-5), there must also be a value of runoff velocity and runoff or else the plume will not generate. Likewise, specifying a value of runoff velocity or radius without an appropriate value in the plume mask file will have no effect.
- *usePlumeDiagnostics*

- o If this options is selected, the model with output several additional diagnostics relating to IcePlume.
- o Requires pkg/diagnostics to be compiled and enabled in data.pkg
- o The frequency of output can be set in data.diagnostics.
- o The diagnostics are output as a 4D matrix (nx,ny,nz,nt). Values will be zero except for those locations given in the plumeMaskFile
- o The available diagnostics are
  - icefrntW – plume vertical velocity (m s^{-1})
  - icefrntT – plume temperature (°C)
  - icefrntS – plume salinity (psu)
  - icefrntM – plume melt rate (m d^{-1})
  - icefrntA – cell average melt rate (m d^{-1})

- *conserveMass*
  - o Without this option, the volume of water output from the plume is equal to the volume of subglacial runoff added plus the volume of ambient water entrained into the plume. The entrained water is neutral with respect to the volume of the domain, but the addition of the subglacial runoff results in a small net volume gain. This becomes problematic if it cannot under the constraints of the scenario be balanced by a net outflow across an open boundary. If this is the case, *conserveMass*, can be used scale down the output from the plume such that it this net input is eliminated. This generally only results in a small decrease in plume output (<2%), because the majority of the plume is comprised of entrained fjord waters.

- *backgroundVel*
  - o Currents at the icefront – particularly melt driven convection – may by driven by processes too fine to resolve in the model set up (e.g. requiring a metre scale grid). These currents are however important as they strongly influence the melt rate. In this case, *backgroundVel* can be used to specify a constant minimum background velocity which is applied in the melt parameterisation in the absence of stronger currents. From laboratory, field and modelling studies, it is expected that melt driven convection should be relatively consistent over time, and of the order of 0.01 – 0.1 m/s.

- *E_0*
  - o The is the entrainment coefficient in the plume model. A value of 1 is commonly used, but it may vary between 0.07 – 0.16 along a continuum from forced jets to buoyant plumes (Kaminski *et al.* 2005). Proglacial plumes are buoyancy dominated, and so values in the upper half of this range may be appropriate.

- *useInputPtracers*
  - o This option, in conjunction with pkg/ptracers, marks the subglacial runoff with a passive tracer. The concentration of this tracer represents the proportion of water in a cell that was originally glacial runoff (e.g. 1 means the cell is filled entirely with subglacial runoff). This enables the flow of these waters to be tracked. This volume represents only the subglacial discharge, not the total volume of the plume (which is mainly entrained fjord water).
  - o Whether or not this option is selected, the plume model will transport any existing ptracers that become entrained in the plume

- When using iceplume in conjunction with ptracers, the modified version of ptracers_apply_forcing.F must be included in the local code directory
- *ptracerMaskFile*
  - If *useInputPtracers* = .TRUE., then this file must be used to give the concentration of the different ptracers in the subglacial runoff.
  - The mask has dimension nx by ny by n_ptracers, where n_ptracers corresponds to PTRACERS_numInUse in data.ptracers.
  - A value should be given for each ptracer for each plume location, as specified in *plumeMaskFile.* If a value of zero is given, that tracer will not be added in that location.
  - If a ptracer value is assigned in a non-plume location, it will not trigger a warning; it will simply have no effect.
  - The ptracer is added to the plume as subglacial_discharge * ptracerMask value. For example, if the runoff fraction is the quantity of interest, the ptracerMask value should be set as 1.

Compiling

At present, the package is not incorporated into MITgcm. To compile MITgcm with IcePlume, follow these steps:

1) Copy the contents of the directory iceplume/pkg to MITgcm/pkg/iceplume.
2) Copy the contents of the direction iceplume/mods to your local code directory. Note that the modified version of ptracers_apply_forcing.F should be included if the ptracers package is enabled, but should not be included otherwise (it will generate a compiling error).
3) Compile MITgcm in the usual way. Note that the external library ODEPACK (included in the directory iceplume/pkg, and used to solve the plume equations) relies on implicit classifying of variables, and will not compile if this is disallowed (e.g. if the -fimplicit-none is used in gfortran).

Known restrictions and issues

For the sake of simplicity, partial cells (hFac<1) are not permitted in ice front locations (i.e. where the plume mask does not equal 0). Doing so will cause the model to terminate. It is therefore necessary to ensure that the seafloor depth is equal to a full cell thickness in ice front locations.

- Development – in the newer code, the plume will now start at the cell boundary above the sea bed if the sea bed does not coincide with a cell boundary.

If a plume is placed in a dry cell, this will now trigger an error.

Specifying a runoff velocity or radius of zero is problematic. In some circumstances it seems permissible and will not generate an error (or a plume); however, a transition (using periodicExternalForcing) from a nonzero value to a zero value seems to cause the model to crash without warning or error message.

The package has been most thoroughly tested with the Portland Group compilers. Bugs may exist that show up when using alternative compilers.

Rather than compiling ODEPACK directly into the model, it may be preferable to compile it as an external library linked to MITgcm.

Example

The example simulates the circulation in a small, rectangular cuboid fjord.

The domain is 50 cells long, 11 cells wide and 50 cells deep, at a resolution of 200x200x10 m. There are dry cells along the northern and southern boundaries (fjord walls) and western boundary (glacier). At the eastern boundary there is a sponge layer. At the western end of the fjord, a glacier front (melting but no runoff) is defined in the cells adjacent to the dry cells. In the centre of the glacier front, there is a subglacial input of 50 m^3s^{-1} from a discrete channel (i.e. forming a half-conical plume).

The data files are contained in the input directory. To generate the binary input files, run the script gendata.m in matlab.

The code folder contains the files from iceplume/mods (mentioned above). These are necessary for IcePlume to sit within the MITgcm framework.

Compile MITgcm in the build directory (see notes above). Copy the mitgcmuv executable and the contents of the input folder into the run directory. Run the model as usual.

If the model runs correctly, a warm, down-fjord current should form at approximately 100 m depth at the centre of the fjord adjacent to the glacier. This is a mix of runoff and entrained fjord waters, output from the plume as it reaches neutral buoyancy.
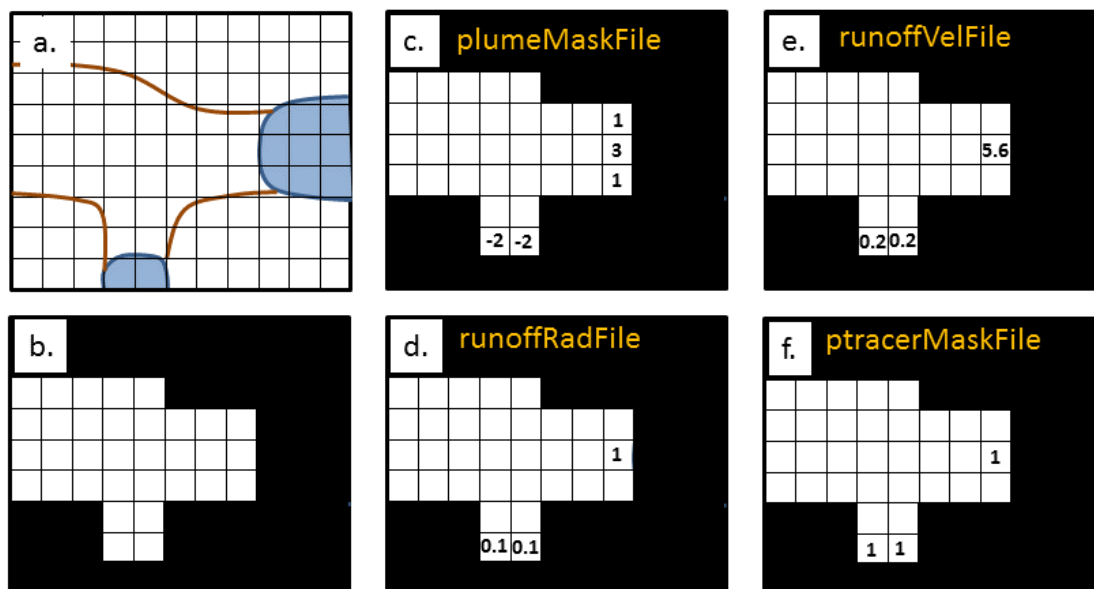
Figures

Figure 1. A simplified example set up for IcePlume. a. The scenario contains two glaciers draining into a small fjord system. b. Mask out the land and glaciers by setting bathymetry to equal zero. c. The plumeMaskFile values around the glacier fronts describe areas in which there is ice but no subglacial runoff (1), a sheet plume (distrubted runoff input; 2) or half-conical plume (discrete runoff input; 3). d-e. The runoffRadFile and runoffVelFile values combine to give a

discrete input of 50 m³s⁻¹ at the eastern glacier and a distrubted input of 20 m³a⁻¹ along the southern glacier (given a grid resolution of 500 m). f. Finally, the ptacerMaskFile specified that the add runoff should have a ptracer(1) concentration of 1 in all plume locations. If more than one ptracer is used, the ptracerMask will must have the same number of layers as ptracers. In c-f., grid cell values not shown are equal to zero. Note that this figure is for illustrative purposes only – IT DOES NOT show the set up for the example experiment.