Simple MAC

1.0

Generated by Doxygen 1.8.1.1

Wed Dec 12 2012 00:14:32

Contents

1 PySimpleMAC Main Solver Documentation

Here you will find the the documentation for the Fortran backend of the PySimpleMAC solver.

See also

dim
solver

2 Todo List

Subprogram solver::parpoisson ()

Parallel Poisson solver does not currently work due to F2Py bug with OpenMP barriers. Use the function poisson()

3 Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

4 File Index

4.1 File List

Here is a list of all files with brief descriptions:

5 Data Type Documentation

5.1 dim Module Reference

This module holds parameter data for the solver.

Public Attributes

- integer, parameter nx = 90

  Number of cells in x-direction.
- integer, parameter ny = 90

  Number of cells in y-direction.
- double precision, parameter dx = 1.0d0/(nx-1.0d0)

  Calculated spatial step size, Δx.
- double precision, parameter dy = 1.0d0/(ny-1.0d0)

  Calculated spatial step size, Δy.
- double precision, parameter wconst = 1.0d0/(4.0d0$*$dx)

  Carried out constant from the calculation of vorticity.
- integer t

  Current timestep value.
- double precision dt

  Current Δt for timestepping.
- double precision re

  Reynolds number of flow, Re.
- double precision r

  Additional convergence criteria for CFL condition.

5.1.1 Detailed Description

This module holds parameter data for the solver.

We store all data not specific to the conserved variables in this module. The variables may be set through subroutines in the solver code.

Definition at line 12 of file mac.f90.

5.1.2 Member Data Documentation

5.1.2.1 double precision dim::dt

Current Δt for timestepping.

Definition at line 25 of file mac.f90.

5.1.2.2 double precision, parameter dim::dx = 1.0d0/(nx-1.0d0)

Calculated spatial step size, Δx.

Definition at line 19 of file mac.f90.

5.1.2.3 double precision, parameter dim::dy = 1.0d0/(ny-1.0d0)

Calculated spatial step size, Δy.

Definition at line 20 of file mac.f90.

5.1.2.4 integer, parameter dim::nx = 90

Number of cells in x-direction.

Definition at line 16 of file mac.f90.

5.1.2.5 integer, parameter dim::ny = 90

Number of cells in y-direction.

Definition at line 17 of file mac.f90.

5.1.2.6 double precision dim::r

Additional convergence criteria for CFL condition.

Definition at line 28 of file mac.f90.

5.1.2.7 double precision dim::re

Reynolds number of flow, Re.

Definition at line 27 of file mac.f90.

5.1.2.8 integer dim::t

Current timestep value.

Definition at line 24 of file mac.f90.

5.1.2.9 double precision, parameter dim::wconst = 1.0d0/(4.0d0∗dx)

Carried out constant from the calculation of vorticity.

Definition at line 22 of file mac.f90.

The documentation for this module was generated from the following file:

- /Users/laptop/Desktop/newProject3/wrapping/mac.f90

5.2 solver Module Reference

The MAC Method solver and it's persistent data.

Public Member Functions

- subroutine initzero ()

  Initialize all variables with zero values.
- subroutine returnu (array)

  Accessor for U-Velocity.
- subroutine returnv (array)

  Accessor for V-Velocity.
- subroutine returnp (array)

  Accessor for Pressure.
- subroutine ghostcondition ()

  Applies ghost cell boundary conditions at the walls.
- subroutine lidcondition ()

  Applies a moving lid to the top of the cavity, and a no slip condition at every other wall.
- subroutine calctstep ()

  Calculates are maximum allowable timestep, Δt, based on our stability conditions.
- subroutine calcfngn ()

  Calculates intermediate step of F_n and G_n. Reference Harlow & Welch (1965) for explanation.
- subroutine calcqn ()

  Q_n is calculated as a source for the RHS of the pressure Poisson equation.
- subroutine calcvel ()

  This subroutine calculates velocity of the field and stores the data in the module.
- subroutine vort (w)

  Calculates vorticity based on velocity field.
- subroutine poisson ()

  Poisson solver for the pressure field.
- subroutine parpoisson ()

  Parallelized Poisson solver.

Public Attributes

- double precision, dimension(nx, ny) u

  Instantaneous velocity in x-direction, $u_{i,j}^{n+1}$.
- double precision, dimension(nx, ny) v

  Instantaneous velocity in y-direction, $v_{i,j}^{n+1}$.
- double precision, dimension(nx, ny) p

  Scalar pressure at cell center $P_{i,j}$.
- double precision, dimension(nx, ny) fn

  See Harlow & Welch 1965.
- double precision, dimension(nx, ny) gn

  See Harlow & Welch 1965.
- double precision, dimension(nx, ny) q

  Pressure source term for Poisson.

5.2.1 Detailed Description

The MAC Method solver and it's persistent data.

Definition at line 33 of file mac.f90.

5.2.2 Member Function/Subroutine Documentation

5.2.2.1 subroutine solver::calcfngn ( )

Calculates intermediate step of F_n and G_n. Reference Harlow & Welch (1965) for explanation.

These variables are calculated based on the previous time-step's velocity. F_n is:

$$F^n_{i+\frac{1}{2},j} = u_{i+\frac{1}{2},j} + \Delta t \left[\frac{u^n_{i+\frac{3}{2},j} - 2u^n_{i+\frac{1}{2},j} + u^n_{i-\frac{1}{2},j}}{Re(\Delta x)^2} + \frac{u^n_{i+\frac{1}{2},j-1} - 2u^n_{i+\frac{1}{2},j} + u^n_{i+\frac{1}{2},j-1}}{Re(\Delta y)^2} - \frac{(uv)^n_{i+\frac{1}{2},j+\frac{1}{2}} - (uv)^n_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} \right]$$

G_n is defined as:

$$G^n_{i,j+\frac{1}{2}} = v_{i,j+\frac{1}{2}} + \Delta t \left[\frac{v^n_{i+1,j+\frac{1}{2}} - 2v^n_{i,j+\frac{1}{2}} + v^n_{i-1,j+\frac{1}{2}}}{Re(\Delta x)^2} + \frac{v^n_{i,j+\frac{3}{2}} - 2v^n_{i,j+\frac{1}{2}} + u^n_{i,j-\frac{1}{2}}}{Re(\Delta y)^2} - \frac{(uv)^n_{i+\frac{1}{2},j+\frac{1}{2}} - (uv)^n_{i-\frac{1}{2},j-\frac{1}{2}}}{\Delta y} - \frac{(v^n_{i,j+1})^2 - (v^n_{i,j})^2}{\Delta y} \right]$$

Definition at line 197 of file mac.f90.

5.2.2.2 subroutine solver::calcqn ( )

Q_n is calculated as a source for the RHS of the pressure Poisson equation.

It is calculated based on the values of F_n and Q_n calculated in the routine calcfngn().

$$Q^n_{i,j} = \left[\frac{P_{i-1,j} - 2P_{i+1,j}}{(\Delta x)^2} + \frac{P_{i,j-1} - 2P_{i,j} + P_{i,j+1}}{(\Delta y)^2} \right]^{n+1}$$

which may be discretized to show:

$$Q^n_{i,j} = \frac{1}{\Delta t} \left[\frac{F^n_{i+\frac{1}{2},j} - F^n_{i-\frac{1}{2},j}}{\Delta x} + \frac{G^n_{i,j+\frac{1}{2}} - G^n_{i,j-\frac{1}{2}}}{\Delta y} \right].$$

Definition at line 235 of file mac.f90.

5.2.2.3 subroutine solver::calctstep ( )

Calculates are maximum allowable timestep, Δt, based on our stability conditions.

The first stability condition to be evaluated is:

$$\Delta t = Re \Delta x \Delta y * r$$

Where r must be less than 0.25. The second stability condition, called the CFL condition, is shown to be:

$$\Delta t = \frac{4}{Re(|u| + |v|)^2}$$

We take the minimum of these two conditions to ensure stability.

Definition at line 143 of file mac.f90.

5.2.2.4 subroutine solver::calcvel ( )

This subroutine calculates velocity of the field and stores the data in the module.

Values of the velocity may be accessed in Python from the routines returnu() and returnv(). The velocity is calculated by the equations:

$$u^{n+1}_{i+\frac{1}{2},j} = F^n_{i+\frac{1}{2},j} - \frac{\Delta t}{\Delta x}(P^{n+1}_{i+1,j} - P^{n+1}_{i,j})$$

$$v^{n+1}_{i+,j+\frac{1}{2}} = G^n_{i,j+\frac{1}{2}} - \frac{\Delta t}{\Delta x}(P^{n+1}_{i,j+1} - P^{n+1}_{i,j})$$

Definition at line 261 of file mac.f90.

5.2.2.5 subroutine solver::ghostcondition ( )

Applies ghost cell boundary conditions at the walls.

For a rectanglular shape, we see that:

$$u|_{L_x=0} = 0$$
$$u|_{L_x=1} = 0$$
$$v|_{L_y=0} = 0$$
$$v|_{L_y=1} = 0$$

This mathematically states that a fluid element cannot pass through a wall.

Definition at line 83 of file mac.f90.

5.2.2.6 subroutine solver::initzero ( )

Initialize all variables with zero values.

Definition at line 46 of file mac.f90.

5.2.2.7 subroutine solver::lidcondition ( )

Applies a moving lid to the top of the cavity, and a no slip condition at every other wall.

The no slip condition discretized for MAC Method is posed as such:

$$u|_{L_y=0} = 0$$
$$u|_{L_y=1} = 1$$
$$v|_{L_x=0} = 0$$
$$v|_{L_x=1} = 0$$

Definition at line 114 of file mac.f90.

5.2.2.8 subroutine solver::parpoisson ( )

Parallelized Poisson solver.

Todo Parallel Poisson solver does not currently work due to F2Py bug with OpenMP barriers. Use the function poisson()

Definition at line 373 of file mac.f90.

5.2.2.9 subroutine solver::poisson ( )

Poisson solver for the pressure field.

The equation for the pressure field is defined to be:

$$\nabla^2 P = Q_n$$

For two dimensions the equation becomes

$$\frac{\partial P}{\partial x} + \frac{\partial P}{\partial y} - Q_n = 0$$

In discretized form, using an explicit Gauss-Seidel method, we see

$$P_{i,j}^{n+1} = \frac{1}{4}\left[P_{i-1,j}^{n+1} + P_{i,j-1}^{n+1} + P_{i+1,j}^{n} + P_{i,j+1}^{n} - (\Delta x)^2 Q_{i,j}^{n}\right]$$

Definition at line 319 of file mac.f90.

5.2.2.10 subroutine solver::returnp (double precision, dimension(nx,ny), intent(out) *array*)

Accessor for Pressure.

Parameters

out	*array*	Function returns a Numpy array of P

Definition at line 68 of file mac.f90.

5.2.2.11 subroutine solver::returnu (double precision, dimension(nx,ny), intent(out) *array*)

Accessor for U-Velocity.

Parameters

out	*array*	Function returns a Numpy array of U

Definition at line 56 of file mac.f90.

5.2.2.12 subroutine solver::returnv (double precision, dimension(nx,ny), intent(out) *array*)

Accessor for V-Velocity.

Parameters

out	*array*	Function returns a Numpy array of V

Definition at line 62 of file mac.f90.

5.2.2.13 subroutine solver::vort (double precision, dimension(nx,ny), intent(out) *w*)

Calculates vorticity based on velocity field.

Vorticity is mathematically defined as half the curl of the velocity field.

$$\omega = \frac{1}{2}\nabla \times \bar{U}$$

Definition at line 290 of file mac.f90.

5.2.3 Member Data Documentation

5.2.3.1 double precision, dimension(nx,ny) solver::fn

See Harlow & Welch 1965.

Definition at line 39 of file mac.f90.

5.2.3.2 double precision, dimension(nx,ny) solver::gn

See Harlow & Welch 1965.

Definition at line 40 of file mac.f90.

5.2.3.3 double precision, dimension(nx,ny) solver::p

Scalar pressure at cell center $P_{i,j}$.

Definition at line 38 of file mac.f90.

5.2.3.4 double precision, dimension(nx,ny) solver::q

Pressure source term for Poisson.

Definition at line 41 of file mac.f90.

5.2.3.5 double precision, dimension(nx,ny) solver::u

Instantaneous velocity in x-direction, $u_{i,j}^{n+1}$.

Definition at line 36 of file mac.f90.

5.2.3.6 double precision, dimension(nx,ny) solver::v

Instantaneous velocity in y-direction, $v_{i,j}^{n+1}$.

Definition at line 37 of file mac.f90.

The documentation for this module was generated from the following file:

- /Users/laptop/Desktop/newProject3/wrapping/mac.f90

6 File Documentation

6.1 /Users/laptop/Desktop/newProject3/wrapping/mac.f90 File Reference

Data Types

- module dim

  This module holds parameter data for the solver.
- module solver

  The MAC Method solver and it's persistent data.

Index