MECH 570C Code Project 1

February 17, 2024

1 main

Crd has the id of each points on the entire domian,the second colume is the x-coord of each point and the thrid is the y-coord.

Cnn defines every elements in the entire domain, for instance, 6->173->1238->464 is the first element.

BCCyl/Top/bot are not boundary conditions, they are the id of elements on the corresponding boundary.

$C = unique(A)$ returns the same data as in A, but with no repetitions. C is in **sorted** order.

Sol.u has number of elements as row and 2 columns, 1 is the x direction and 2 is the y direction.

pmc is the time integration parameters.

idx2(:,1) means which elements has the indicated number as the first coord. Eg: the 4249th element has the 4th points as the first coord which is on the boundary of the circle.

So each local elements/poins shown in BCCyl is found in the global cnn matrix with global id. Then form new cnnCyl by local id and global 4 coords.

cnnCylnew reorders the elements on the cylinder

2 IO

ismember: $[Lia, Locb] = ismember()$ also returns an array, Locb, using any of the previous syntaxes. Generally, Locb contains the lowest index in B for each value in A that is a member of B. Values of 0 indicate where A is not a member of B.
If the 'rows' option is specified, then Locb contains the lowest index in B for each row in A that is also a row in B. Values of 0 indicate where A is not a row of B.
If A and B are tables or timetables, then Locb contains the lowest index in B for each row in A that is also a row in B. Values of 0 indicate where A is not a row of B.

3 N-S

Function Definition

$\rho u_t + \rho u.divu - div(mu * gradu) + gradp = fin\Omega$, $divu = 0$ in Ω, Dirichlet boundary condition $u = g_D on \Gamma_D$, Neumann boundary condition $du/dn - np = g_N on \Gamma_N$.

- **Function Name:** `navierStokes`
- **Inputs:**
  - `solver`, `fluid`, `pmc`: Structures containing various solver settings, fluid properties, and numerical method parameters.
  - `Sol`: Structure holding the solution vectors.
  - `cnn`, `crd`: Connectivity and coordinate matrices for the mesh.
  - `elemType`, `ndof`, `nen`, `nElem`, `BCCyl`: Parameters defining the type of elements used, degrees of freedom, number of nodes per element, total number of elements, and boundary condition information.

Quadrature Rules

- Sets up quadrature points (`gP`) and weights (`gW`) based on the element type (triangular or quadrilateral). These are used for numerical integration.
- Defines shape functions (`N`) and their derivatives (`Nx`, `Ny`) for the finite elements.

Boundary Conditions

- Applies Dirichlet and Neumann boundary conditions to the solution vector `Sol.u`.

Interpolation for Alpha Values

- Interpolates values for the generalized-alpha method, a numerical technique for time integration in transient problems.

Navier-Stokes Equations

- Prepares variables (`xxf`, `yyf`, `ux`, `uy`, etc.) for assembling the finite element matrices. These include coordinates, velocities, pressure, and additional variables.

Assembly of Galerkin and Petrov-Galerkin Terms

- Calls functions to form the left-hand side (LHS) and right-hand side (RHS) of the Navier-Stokes equations. This involves complex operations based on the finite element method.
- The Galerkin method is used for discretizing the problem, while Petrov-Galerkin is an enhanced approach for stability and accuracy.

Solving the Linear System

- Determines the free nodes not constrained by boundary conditions.
- Solves the linear system for the unknowns (`Increment`) using the assembled LHS and RHS.

Update and Output

- Updates the solution vectors (`Sol.u`, `Sol.uDot`, `Sol.p`) with the new increments.
- Calculates a norm (`NSnormIndicator`) to indicate the convergence or error of the current iteration.
- Outputs the updated solution structure `Sol` and the convergence/error indicator.

Summary of Data Flow

1. **Input Processing:** Takes in initial conditions, mesh information, and solver parameters.
2. **Setup:** Establishes quadrature rules and boundary conditions.
3. **Equation Assembly:** Forms the Navier-Stokes equations using finite element discretization.
4. **Solution Update:** Solves for increments and updates the solution.
5. **Output:** Returns the updated solution and a convergence/error metric.

4 Integraded Output

Function Definition

- **Function Name:** `IntegratedOutput`
- **Inputs:**

- `Sol`: A structure containing the solution vectors (velocity, pressure, etc.).
- `crd`: The coordinates of the mesh nodes.
- `BCCyl`: Boundary condition data.
- `fluid`: A structure containing fluid properties.
- `cnn`**: Connectivity matrix for the mesh elements.**

Initialization

- Sets the number of element nodes (`nen`) for a 2D element (4 for a quadrilateral element).
- Swaps the columns of `BCCyl` for further processing.
- Determines the number of elements (`nElem`) and degrees of freedom (`ndof`) involved in the boundary condition.

Boundary Layer Elements

- Identifies elements corresponding to the first layer of the boundary using `cnn` and `BCCyl`.
- Reorders the element points for reduced integration.

Quadrature Integration Setup

- Defines Gauss points (`gP`) and weights (`gW`) for numerical integration.
- Sets up shape functions (`N`) and their derivatives (`Nx`, `Ny`).

Localizing Data

- Extracts local coordinates (`xxf`, `yyf`), velocities (`ux`, `uy`), and pressure (`pres`) for each element.

Integration Process

- Loops over quadrature points to:
  - Calculate the Jacobian matrix `J` for coordinate transformation.
  - Compute the volume and normal vectors for each element.
  - Evaluate pressure and velocity gradients (`locgradUx`, `locgradUy`, `locgradVx`, `locgradVy`).
  - Compute the length/area (`A0`) of the line/surface integral.

Calculation of Forces

- The loop sums up the contributions from all quadrature points to calculate the integrated force over the element surface.
- The force components (X and Y directions) are to be computed within the loop (currently not implemented in the provided code).

Output

- Returns the total length of the integrated area (`Length`) and the force components (`Force`).