ASEN 5331 - HW4

James Wright

December 11, 2019

0.1 Meaning of `n...`

Term	Definition	Source/Relevant Reference
`nsd`	number of spacial dimensions	`common/common.h#343`
`nflow`	number of flow variables (ie. size of $\boldsymbol{Y}$)	?
`nshape`	number of interior element shape functions	`common/common.h#444`
`ngauss`	number of interior element integration points	`common/common.h#447`
`npro`	number of elements processed in a single call of `e3.f`	Jansen lecture
`npro`	number of virtual processors for the current block	`common/common/h#586`
`nen`	maximum number of element nodes	`common/common.h#341`
`nQpt`	number of quadrature points per element	`common/shp4t.f#14`
`nshl`	number of shape functions per element	`common/genblkPosix.f#70`
`nshg`	global number of shape functions	`common/common.h#354`
`nenl`	number of element nodes for current block	`common/common.h#382`
`nedof`	total number of degrees of freedom	`common/e3.f#35,344`

1 Essential Boundary Conditions

1.1 Setting BC Values

In `compressible/itrbc.f`
$\boldsymbol{Y} =$ `y`
$g(x) =$ `BC` on a per node basis

The essential boundary conditions are set in `/compressible/itrbc.f#59-198`. The `iBC` variable contains bit-wise information on what specific boundary conditions are going to be set. `BC` contains the BC data ($g(x)$ in the notes) for each individual node. `iBC` is set in `common/genibc.f` and `BC` is set in `common/genbc.f`, which takes `iBC` as an input.

Essentially, the code checks `iBC` for which values of $\boldsymbol{Y}$ should be set. This logical check is done via the `ibits()` function. If a given $\boldsymbol{Y}$ chosen, then $\boldsymbol{Y}$ is set to the corresponding value in `BC`.

So `common/gendat.f->gendat()` calls `common/genibc.f->geniBC()` to create the `iBC` vector. `common/gendat.f` then calls `common/genbc.f->genBC()` to create the `BC` vector which contains the values that should then be substituted into the `y` array in `compressible/itrbc.f`.

1.2 Applying $\boldsymbol{S}$ Matrices

The application of the $\boldsymbol{S}$ matrices applied in two different locations: `compressible/b3res.f` and `compressible/b3lhs.f`, which apply $\boldsymbol{S}$ to the residual (`res`, RHS) and mass matrix (`EGMass`, LHS) respectively.

For the residual, the values of `res` are replaced based on the logical output from the same `ibits()` operation on `iBC` as before, similar to when `BC` values were set in `common/genbc.f->genBC()`. This process occurs in `compressible/b3res.f#28-163`.

Applying $\boldsymbol{S}$ on the LHS operates in a similar way, with the primary difference being the differences in how $\boldsymbol{S}$ is applied analytically (ie. $\boldsymbol{S}^T\boldsymbol{G}$ vs. $\boldsymbol{S}^T\boldsymbol{M}\boldsymbol{S}$). Since the $\boldsymbol{S}^T\boldsymbol{M}\boldsymbol{S}$ operation acts on the applicable row and column of the $\boldsymbol{M}$ matrix at once via `do` loops.

2 Natural Boundary Conditions

There are 6 different flux components: normal flux h^m, pressure flux h^p, viscous stress/traction vector h^v_j, and heat flux h^h.

`iBCB` is the equivalent of `IBC` for natural boundary conditions; it stores, bit-wise, which fluxes are to be set.

`BCB` stores the h^j values. The last index of the array represents the 6 flux components:

Index	Variable	Description
1	h^m	mass/normal flux
2	h^p	pressure flux
3	h^v_1	viscous flux in x1 direction
4	h^v_2	viscous flux in x2 direction
5	h^v_3	viscous flux in x3 direction
6	h^h	heat flux

`compressible/e3bvar.f#85-128` first computes the flow variable values at the quadrature points, `pres, T, u1, u2, u3, rho, ei, rk`. These values are used to set the "floating" fluxes (ie. fluxes on $\Gamma - \Gamma^j_h$). This also computes the user defined flux values h^j at quadrature points and puts them into `rou, p, Fv1, Fv2, Fv3, heat` for the components of `BCB`.

`compressible/e3b.f` sets a series of flux vectors `F1, F2, F3, F4, F5` which correspond to the 5 fluxes of the primitive equation (density flux, 3 velocity fluxes, and a temperature flux). These fluxes are either set to their "floating" values or the user specified flux values, as determined by the value of `iBCB`. The `e3b()` routine also defines `Fv2, Fv3, Fv4` and `Fv5` as parts of the viscous flux, which are then used for creating other fluxes (for example, part of the energy flux is simply $u_i\tau_{ij}$).

Once the `F1, F2, F3, F4, F5` variables are set, they are then put into the residual `rl` in `compressible/e3b.f#266-292`.