DTU

Boundary conditions

FVM bulk flow equation solution

Table of Contents

Inlet

$$u_b = \frac{\dot{m}}{A}$$

$$v_b = v_i = \text{constant}$$

$$p_b = p_i - 0.5(1 + \xi_i)u^2$$

p_i is the inlet total pressure (fixed)

```
area = np.sqrt(sf[i, 0] ** 2 + sf[i, 1] ** 2) * hf[i]
self.ubc[idx] = phi[i] * (cn[i, 0] / np.abs(cn[i, 0])) / area
self.pbc[idx] = self.p_i - 0.5 * (1.0 + self.xi_in) * np.abs(self.ubc[idx]) ** 2
```

Outlet

$$u_b = \frac{\dot{m}}{A}$$

$$v_b \leftarrow \text{extrapolate from interior}$$

$$p_b = p_e - 0.5(1 - \xi_e)u^2$$

p_e is the exit total pressure (fixed)

```
area = np.sqrt(sf[i, 0] ** 2 + sf[i, 1] ** 2) * hf[i]
self.ubc[idx] =  phi[i] * (cn[i, 0] / np.abs(cn[i, 0])) / area
self.vbc[idx] = self.v[p]
self.pbc[idx] = self.p_e - 0.5 * (1.0 - self.xi_exit) * np.abs(self.ubc[idx]) ** 2
```

Momentum equation

Inlet treated like prescribed velocity / mass flux bc

$$b_u \leftarrow -\dot{m}u_b$$
$$b_v \leftarrow -\dot{m}v_b$$

```
self.bu[p] += - phi[i] * self.ubc[idx]   # convective flux
self.bv[p] += - phi[i] * self.vbc[idx]  # convective flux
```

Outlet is extrapolated

$$\dot{m}u_b \approx \dot{m}u_p$$

```
self.A[p, p] +=  phi[i]
```

Pressure-correction equation

Both inlet and outlet contribution mass fluxes to source terms. Boundary values used to evaluate coefficients.

$$a_f \approx \rho_f D_f h_f \left(\frac{S_x^f}{x_f - x_P} + \frac{S_y^f}{y_f - y_P} \right)$$

```
if self.bc_type[idx] == 1:  # inflow
  fluxp = rhof[i] * Df[i] * hf[i] * (div(sf[i, 0], cn[i, 0]) + div(sf[i, 1], cn[i, 1]))
  self.Ap[p, p] += fluxp
  self.bp[p] += - phi[i]
if self.bc_type[idx] == 2:  # outflow, specified pressure
  fluxp = rhof[i] * Df[i] * hf[i] * (div(sf[i, 0], cn[i, 0]) + div(sf[i, 1], cn[i, 1]))
  self.Ap[p, p] += fluxp
  self.bp[p] += - phi[i]
```

Table of Contents

Inlet

The following does not work...

$$u_b = \frac{\dot{m}}{A}$$
$$v_b = 0.0$$
$$p_b = -(1+\xi_i)|u_0|u_1$$

Replaced with...

$$u_b \leftarrow \text{extrapolate from interior}$$
$$v_b = 0.0$$
$$p_b = (1+\xi_i)|u_0|u_1$$

```
if self.bc_type[idx] == 1:  # total pressure inlet
```

Implementation

```
if self.bc_type[idx] == 1:  # total pressure inlet
  area = np.sqrt(sf[i, 0] ** 2 + sf[i, 1] ** 2) * hf[i]
  #self.u1bc[idx] = phi1[i] * (cn[i, 0] / np.abs(cn[i, 0])) / area
  self.u1bc[idx] = self.u1[p]
  #self.p1bc[idx] = - (1.0 + self.xi_in) * np.abs(self.ubc[idx]) * (np.abs(np.real(self.u1bc[
      idx])) + np.abs(np.imag(self.u1bc[idx]))*1j)
  self.p1bc[idx] = (1.0 + self.xi_in) * np.abs(self.ubc[idx]) * self.u1bc[idx]
if self.bc_type[idx] == 2:  # outlet
  area = np.sqrt(sf[i, 0] ** 2 + sf[i, 1] ** 2) * hf[i]
  #self.u1bc[idx] =  phi1[i] * (cn[i, 0] / np.abs(cn[i, 0])) / area
  self.u1bc[idx] = self.u1[p]
  self.v1bc[idx] = self.v1[p]
  self.p1bc[idx] = (1.0 - self.xi_exit) * np.abs(self.ubc[idx]) * self.u1bc[idx]
```

Sample results

Kanki 1984, long seal, see test04b.py

Original

```
---Values (model | exp)---
leakage [cm^3/s] : 4673.79 | 4634
K_xx [MN/m]      : 6.14143 | 3.59
K_yx [MN/m]      : 9.81498 | 10.8
D_xx [kN.s/m]    : 112.72 | 147
D_yx [kN.s/m]    : -52.4166 | 55.3
M_xx [kg]        : 294.34 | 221.5
---Relative errors---
leakage [%] : 0.858638
K_xx [%]    : 71.0704
K_yx [%]    : -9.12054
D_xx [%]    : -23.32
D_yx [%]    : -194.786
M_xx [%]    : 32.8848
```

Corrected

```
---Values (model | exp)---
leakage [cm^3/s] : 4673.79 | 4634
K_xx [MN/m]      : 3.74828 | 3.59
K_yx [MN/m]      : 10.894 | 10.8
D_xx [kN.s/m]    : 140.536 | 147
D_yx [kN.s/m]    : -53.784 | 55.3
M_xx [kg]        : 302.827 | 221.5
---Relative errors---
leakage [%] : 0.858638
K_xx [%]    : 4.409
K_yx [%]    : 0.870695
D_xx [%]    : -4.39696
D_yx [%]    : -197.259
M_xx [%]    : 36.7165
```