Weak Form & LiveLink for MATLAB Based Modified Uzawa Method for Solving Steady Navier-Stokes Equation

Huashan Sheng[*1], Shuai Zhu[1]
1 Department of Mathematics Shanghai Jiao Tong University
* Dongchuan Rd. No. 800, Minhang, Shanghai, China, shs3701001@sjtu.edu.cn

Abstract: COMSOL Multiphysics can solve nonlinear PDEs easily by the built-in Newton method, and one typical example is the Navier-Stokes equation. But, sometimes we can't figure out a good initial guess for Newton method, or maybe we have some other better algorithms. In these cases, The Laminar Flow interface may not so suitable. So in this paper, we test the modified Uzawa iterative algorithm[1] for Navier-Stokes equation by using weak form and LiveLink for MATLAB in COMSOL. We get the same solution as other FEM packages (FreeFEM++ & FEniCs) even at each iterative step.

Keywords: Navier-Stokes equation, Modified Uzawa Method, Weak Form PDE, LiveLink for MATLAB

1. Introduction

This paper is concerned with using the modified Uzawa iterative algorithm for solving the steady incompressible Navier-Stokes equation numerically in COMSOL Multiphysics**.**

1.1 Navier-Stokes Equation Theory

Steady incompressible Navier-Stokes equation:

$$\begin{cases} -\mu\Delta\boldsymbol{u} + (\boldsymbol{u}\cdot\nabla)\boldsymbol{u} + \nabla p = \boldsymbol{f} & \text{in } \Omega, \\ \nabla\cdot\boldsymbol{u} = 0 & \text{in } \Omega. \end{cases} \quad (1\text{-}1)$$

Where the domain $\Omega \subset \mathrm{R}^{\mathrm{d}}$ ($d = 2$ or 3) is a bounded domain with Lipschitz boundary $\partial\Omega$, $\mu = 1/Re$ means the viscosity coefficient, and Re denoting the Reynolds number; the right hand side $\boldsymbol{f}$ is the prescribed body force; $\boldsymbol{u}$ and p are the corresponding velocity field and the pressure field, respectively. The above PDEs are the well-known mathematical model describing the steady flow of an incompressible Newtonian fluid, such as air or water, which are frequently encountered in engineering applications, from airplane design to the construction of power stations. Sometimes, to simplify, we impose the non-slip condition for Navier-Stokes Equation:

$$\boldsymbol{u} = 0 \quad \text{on} \quad \partial\Omega \quad (1\text{-}2)$$

To ensure the uniqueness of pressure field p we always assume that:

$$p \in L_0^2(\Omega) := \left\{ q \in L^2(\Omega); \int_\Omega q \mathrm{d}\Omega = 0 \right\}$$

We introduce some notation for later use. Let $(\cdot,\cdot)$ denote the scalar product over $L_2(\Omega)$. Given a non-negative integer s, let $H_s(\Omega)$ be the usual Sobolev space consisting of all functions $v \in L_2(\Omega)$ whose generalized derivatives with the total degree no more than s are still $L_2(\Omega)$ integrable. With the known number s, let $H_s(\Omega)$ with the standard norm $\|\cdot\|_s$ and seminorm $|\cdot|_s$. Furthermore, let $\boldsymbol{V} := \boldsymbol{H}_0^1(\Omega)$ $P := L_0^2(\Omega)$ and define a trilinear form over $\boldsymbol{V}^3$ as follows:

$$a_1(\boldsymbol{u};\boldsymbol{v},\boldsymbol{w}) = \int_\Omega (\boldsymbol{u}\cdot\nabla)\boldsymbol{v}\cdot\boldsymbol{w}\mathrm{d}x \quad (1\text{-}3)$$

The weak form of Navier-Stokes Equation (1-1): Find $(\boldsymbol{u}, p) \in \boldsymbol{V}\times P$ such that

$$\begin{cases} a_1(\boldsymbol{u};\boldsymbol{u},\boldsymbol{v}) + \mu(\nabla\boldsymbol{u},\nabla\boldsymbol{v}) - (p, \operatorname{div}\boldsymbol{v}) = \langle \boldsymbol{f},\boldsymbol{v}\rangle & \forall \boldsymbol{v}\in\boldsymbol{V}, \\ (\operatorname{div}\boldsymbol{u}, q) = 0 & \forall q \in P. \end{cases} \quad (1\text{-}4)$$

1.2 Mixed Element Method

With the partition Th, we associate a stable pair of finite element spaces $(\boldsymbol{V}_h, P_h)$, or equivalently, $\boldsymbol{V}_h$ (resp. P_h) is a subspace of $\boldsymbol{V}$ (resp. P), which admit inf–sup(B.B) condition. We always use the MINI element, the Girault–Raviart element. In this paper we use the $\mathrm{P}_2, \mathrm{P}_1$ element which is also called the Taylor–Hood element. The mixed element method for (1-4) is described as follows: Find $(\boldsymbol{u}_h, p_h) \in \boldsymbol{V}_h \times P_h$ such that

$$\begin{cases} a_1(\boldsymbol{u}_h;\boldsymbol{u}_h,\boldsymbol{v}) + \mu(\nabla\boldsymbol{u}_h,\nabla\boldsymbol{v}) - (p_h, \operatorname{div}\boldsymbol{v}) \\ \qquad = \langle \boldsymbol{f},\boldsymbol{v}\rangle \quad \forall \boldsymbol{v}\in\boldsymbol{V}_h, \\ (\operatorname{div}\boldsymbol{u}_h, q) = 0 \quad \forall q \in P_h. \end{cases} \quad (1\text{-}5)$$

1.3 The Modified Uzawa Method

Given two initial functions $\boldsymbol{u}_h^0$ and p_h^0, which are given by zeros initial guess or the solution of the discrete Stokes equations, and given a positive number $\rho > 0$, run the following iteration until we get the desired accuracy:

$$\begin{cases} a_1(\boldsymbol{u}_h^n;\boldsymbol{u}_h^{n+1},\boldsymbol{v})+\mu(\nabla\boldsymbol{u}_h^{n+1},\nabla\boldsymbol{v}) \\ \qquad -(p_h^n,\operatorname{div}\boldsymbol{v})=\langle\boldsymbol{f},\boldsymbol{v}\rangle \quad \forall\boldsymbol{v}\in\boldsymbol{V}_h, \\ (p_h^{n+1},q)=(p_h^n,q)-\rho(\operatorname{div}\boldsymbol{u}_h^{n+1},q) \\ \qquad \forall q\in P_h. \end{cases} \tag{1-6}$$

In paper [1], we have proved that the modified Uzawa methods converge geometrically with a contraction number independent of the finite element mesh size h.

For understanding convenience, detailed steps of algorithm are given as follows:

Step 0: Set initial value

$\boldsymbol{u}^k \leftarrow \boldsymbol{0}, p^k \leftarrow 0,$

iterative step n=0, give ρ, eps

Step 1: Backup $\boldsymbol{u}^k$ *and* p^k

$\boldsymbol{u}_{bk}^k \leftarrow \boldsymbol{u}^k, p_{bk}^k \leftarrow p^k$

Step 2: Solve velocity field $\boldsymbol{u}^{k+1}$ *by (1-6)*

$a_1(\boldsymbol{u}^k;\boldsymbol{u}^{k+1},\boldsymbol{v})+\mu(\nabla\boldsymbol{u}^{k+1},\nabla\boldsymbol{v})$

$-(p^k, div\ \boldsymbol{v})=\langle\boldsymbol{f},\boldsymbol{v}\rangle$

Step 3: Solve pressure field p^{k+1} *by (1-6)*

$(p^{k+1},q)=(p^k,q)-\rho(div\ \boldsymbol{u}^{k+1},q)$

Step 4: Update velocity and pressure field

$\boldsymbol{u}^k \leftarrow \boldsymbol{u}^{k+1}, p^k \leftarrow p^{k+1}$

Step 5: Break condition

$ErrU = \| \boldsymbol{u}^k - \boldsymbol{u}_{bk}^k \|_\infty / \| \boldsymbol{u}_{bk}^k \|_\infty;$

$ErrP = \| p^k - p_{bk}^k \|_\infty / \| p_{bk}^k \|_\infty;$

if $max(ErrU, ErrP) < eps$

break;

Go to Step 6;

else $n \leftarrow n+1$;

Go to Step 1;

end

Step 6: Output solution, plot and error analysis

Algorithm 1. The modified Uzawa Method

2. COMSOL Multiphysics Settings

The Navier-Stokes equation can be solved conveniently by the laminar flow interface with built-in Newton method, but it's difficult to change its built-in algorithms. Fortunately, the Weak Form PDE interface in the Mathematics branch can solve this problem perfectly. The steps for setting are given as follows (2D problems):

First, start COMSOL Multiphysics, and build a new mph file with 2D domain. Define some parameters and variables in Global Definitions branch, and draw the domain for solving in Geometry branch.

Then add the physics field in Component branch, which is the most important. As step 1 in Alg. 1, add Coefficient Form PDE with two dependent variables named uk_bk, vk_bk, and choose the shape function type to be quadratic Lagrange. Let all the coefficients equal to 0 except the absorption coefficient $a = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and the source term $f = \begin{pmatrix} uk \\ vk \end{pmatrix}$. Add another Coefficient Form PDE with one dependent variable named p_bk, and choose linear Lagrange shape function type. Similarly, let all the coefficients equal to 0 except the absorption coefficient $a = 1$ and the source term $f = pk$. As step 2, add Weak Form PDE with two dependent variables named u, v, and choose quadratic Lagrange shape function type. Write the weak expressions as follows:

$$-(uk * ux + vk * uy) * test(u)$$
$$-mu * \big(ux * test(ux) + uy * test(uy)\big)$$
$$+pk * test(ux) + f1(x,y) * test(u)$$
$$-(uk * vx + vk * vy) * test(v)$$
$$-mu * \big(vx * test(vx) + vy * test(vy)\big)$$
$$+pk * test(vy) + f2(x,y) * test(v)$$

After input the weak expressions, add some Dirichlet boundary conditions. As step 3, add Weak Form PDE with one dependent variable named p, and choose linear Lagrange shape function type. Write the weak expressions similarly as follows:

$$pk * test(p) - rho * (ux + vy) * test(p)$$
$$-p * test(p)$$

Notice that step 4 and step 1 in Alg. 1 are almost the same, add Coefficient Form PDE with two dependent variables named uk, vk, and choose the shape function type to be quadratic Lagrange. Let all the coefficients equal to 0 except the absorption coefficient $a = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ and the source term $f = \begin{pmatrix} u \\ v \end{pmatrix}$. Add another Coefficient Form PDE with one dependent variable named p_bk, and choose linear Lagrange shape function type. Similarly, let all the coefficients equal to 0 except the absorption coefficient $a = 1$ and the source term $f = p$.

Next, add three stationary study steps, and let first step only solve for the first two Coefficient Form PDEs, and second step only solve for the two Weak Form PDEs, and last step only solve for the last two Coefficient Form PDEs as shown in Fig. 1. "Values of variables not solved for"

should be checked in all study steps as shown in Fig. 2.

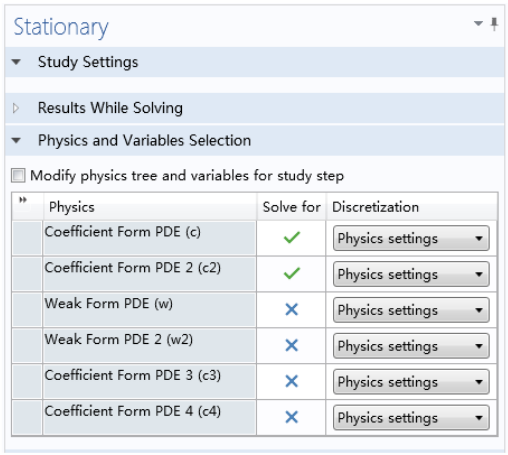


Figure 1. Details for setting in Study Step 1

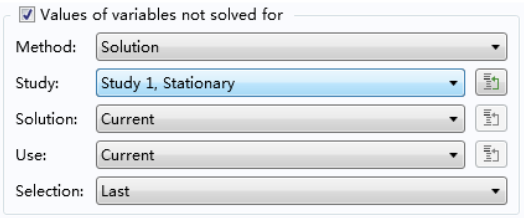


Figure 2. Details for setting in all Study Steps

Lastly, partition the solving domain with the size you need, and save the model without computing. Notice that it's no need to compute the model in COMSOL, and once click "compute", the initial value of $\boldsymbol{u}$ and p will be changed. The "tree" structure is shown in Fig. 3

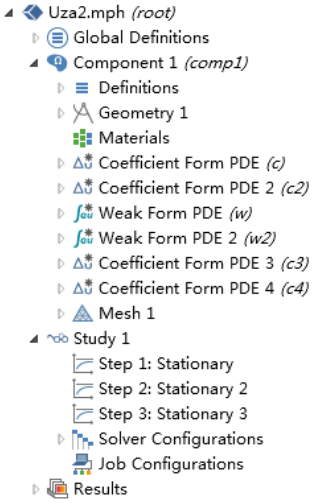


Figure 3. "Tree" structure of model

3. LiveLink for MATLAB

LiveLink for MATLAB is a powerful function in COMSOL for designing iterative algorithm. We will let MATLAB do the "while" loop in Alg. 1. The way starting COMSOL with MATLAB on Windows / Mac OSX / Linux can be found in help document. Once they have been connected, the mph model can be handled in MATLAB so conveniently as using inline function.

To realize the "while" loop, step 5 and step 6 in Alg. 1, first change the current MATLAB fold to the location of the COMSOL file. Then run these codes as follows:

```
model = mphload('Uza2.mph');
ERROR = 1;
COUNT = 1;
ERRUVEC = [];
ERRUPRE = [];
while ERROR > 1e-6 && COUNT <1000
    model.sol('sol1').run;
    Q1 = mphmax(model,...
         'abs(u-uk_bk)','surface');
    Q2 = mphmax(model,...
         'abs(v-vk_bk)','surface');
    Q3 = mphmax(model,...
         'abs(p-pk_bk)','surface');
    B1 = mphmax(model,...
         'abs(uk_bk)','surface');
    B2 = mphmax(model,...
         'abs(vk_bk)','surface');
    B3 = mphmax(model,...
         'abs(pk_bk)','surface');
    ErrU = max(Q1/B1,Q2/B2);
    ErrP = Q3/B3;
    fprintf('Itr.Stp:%d\t\n',COUNT);
    fprintf('Vec.Er:%.2e\t\n',ErrU);
    fprintf('Pre.Er:%.2e\t\n',ErrP);
    ERROR = max(ErrP,ErrU);
    COUNT = COUNT+1;
    ERRUVEC = [ERRUVEC;ErrU];
    ERRUPRE = [ERRUPRE;ErrP];
end
fprintf('---------OK!-------- \n ');
%Plot the solution and Err via step.
subplot(2,2,1);
mphplot(model,'pg1');
subplot(2,2,2);
mphplot(model,'pg2');
subplot(2,2,3)
semilogy(ERRUVEC);
grid on;
subplot(2,2,4);
semilogy(ERRUPRE);
grid on;
```

4. Numerical Experiment

The only other change in different models is the Component branch. For example, the domain is changed or the inlet velocity is larger, nothing needs to do except changing the Geometry branch and the Dirichlet boundary condition. In this part, four typical examples will be tested.

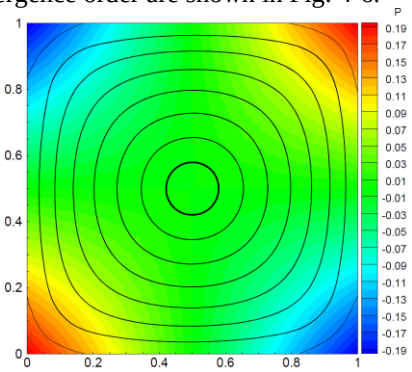
4.1 Flows in a Confined Cavity

We take the domain as $\Omega := (0,1) \times (0,1)$, set the viscosity coefficient $\mu = 0.1$ and choose the right side function $\boldsymbol{f}$ in (1-1) such that the PDE has the following solution:

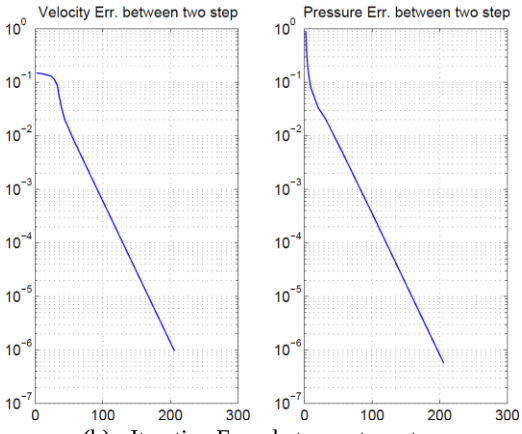
$$\begin{cases} u(x,y) = 0.2\phi(x)\phi'(y) \\ v(x,y) = -0.2\phi'(x)\phi(y) \\ p(x,y) = 0.2(2x-1)(2y-1) \end{cases}$$

$\phi(t) = t^2\,(t-1)^2$

Here $\boldsymbol{u}(x,y) := \big(u(x,y), v(x,y)\big)^T$. We choose the parameter $\rho = \mu(1 - 0.56019)$, and let the mesh size $h = 1/8 \sim 1/128$. Solution and FEM convergence order are shown in Fig. 4-6.



(a) Streamlines and Pressure Value



(b) Iterative Error between two step

Figure 4. Solution of Cavity Flow at $h = 1/16$

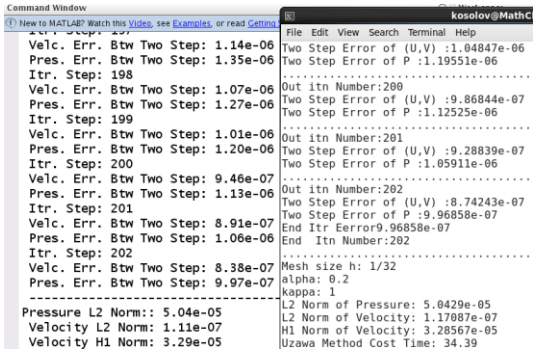


Figure 5. Compare with FreeFEM++ (Right part)

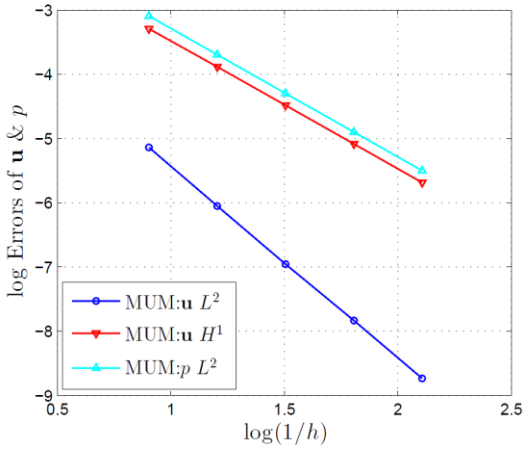
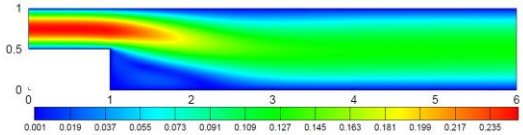


Figure 6. FEM Convergence Order of P2-P1 Element

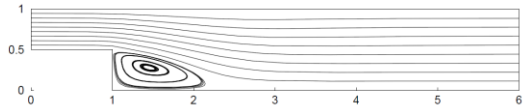
4.2 Back-Step Problem

The second classic example is the back-step flow. Let domain $\Omega := (0,6) \times (0,1) \backslash (0,1) \times (0,0.5)$, and set the viscosity coefficient $\mu = 1/500$ and choose the right side function $\boldsymbol{f} = \boldsymbol{0}$. The left side of the "back-step" is the inlet boundary, and the flow velocity is $u = 4(y-0.5)(1-y)$ perpendicular to the inlet boundary. The right side of the "back-step" is the outlet boundary with do-nothing condition, and this boundary is Neumann boundary. The up side and down side of the "back-step" are the Dirichlet boundary with $\boldsymbol{u} = \boldsymbol{0}$.

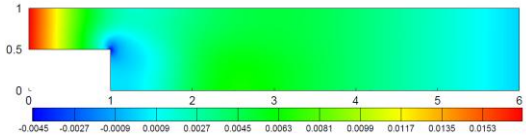
The solution and iterative error with mesh size $h \approx 0.05$ and parameter $\rho = \mu$ are shown in Fig. 7.



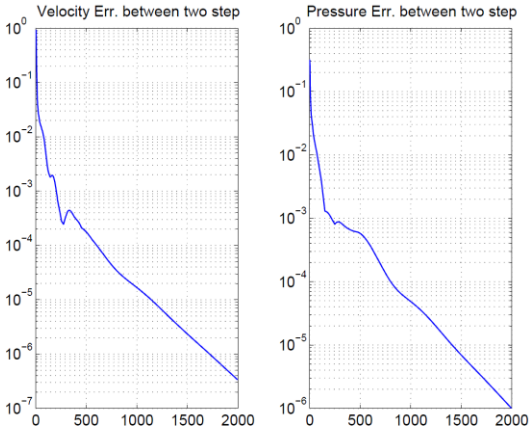
(a) Velocity magnitude



(b) Streamlines



(c) Pressure value

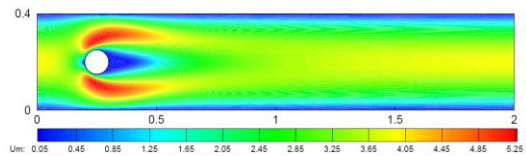


(d) Iterative Error between two step

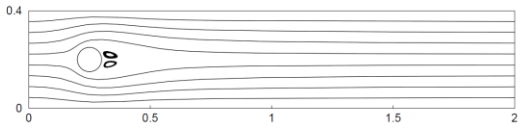
Figure 7. Solution of back step flow

4.3 Channel Flow past a Circular Cylinder

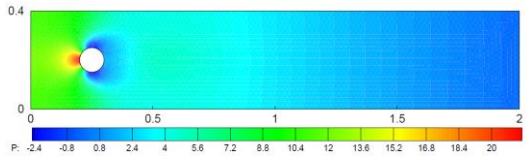
The third example is the channel flow past a circular cylinder. Let the solving domain $\Omega := (0,1) \times (0,0.4) \backslash R(0.25,0.2,0.05)$, and set the viscosity coefficient $\mu = 1/80$ and choose the right side function $\boldsymbol{f} = \boldsymbol{0}$. The left side of the "channel" is the inlet boundary, and the flow velocity is $u = 100y(0.4 - y)$ perpendicular to the inlet boundary. The right side is the flow-out boundary with do-nothing condition, and this boundary is Neumann boundary. The other sides are the entire Dirichlet boundary with $\boldsymbol{u} = \boldsymbol{0}$.
The solution and iterative error with extremely fine physics-controlled mesh, and parameter $\rho = \mu$ are shown in Fig. 8.



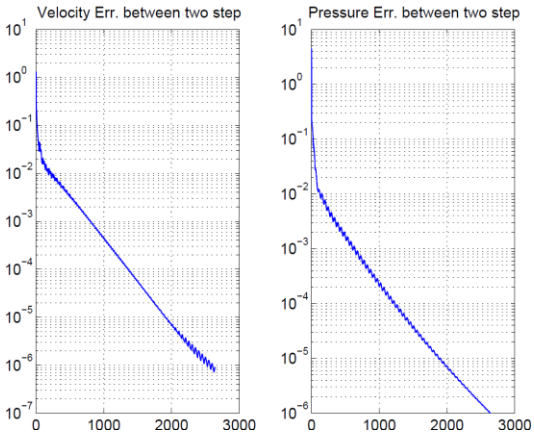
(a) Velocity magnitude



(b) Streamlines



(c) Velocity magnitude



(d) Iterative Error between two step

Figure 8. Solution of channel flow past a circular cylinder

4.4 Aneurysm Simulation

The last example is the channel flow in the blood vessel with aneurysm. The mesh is contributed by Kent-Andre Mardal, and we change it from VMTK type into the mphtxt type for COMSOL. The inlet and outlet boundary can be easily distinguished from "Walls" in Fig 9.

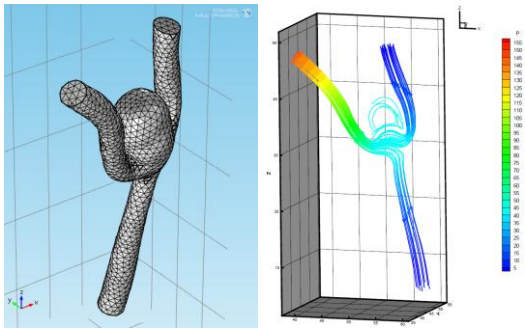


Figure 9. Aneurysm Simulation

5. Conclusion

The weak form PDE and LiveLink for MATLAB in COMSOL give a perfect performance during testing the three examples by modified Uzawa method. This attempt gives algorithm researchers a hint or a method to test their algorithm, and gives engineers another way to compute their models especially those cannot find appropriate built-in interface in COMSOL.

6. References

1. P. Chen, J. Huang, H. Sheng, Some Uzawa methods for steady incompressible Navier–Stokes equations discretized by mixed element methods, *Journal of Computational and Applied Mathematics*, **273**, 313-325 (2015)