

# A numerical analysis of the Navier Stokes Equations

Anders Håfreager

December 5, 2012

## 1 Derivation

There are many ways to derive the Navier Stokes Equations (NSE). Most of them are equivalent because they simply assume conservation of mass and momentum. This is also the approach we will do here. NSE describes the two most important properties in a fluid, namely the pressure  $p$  and velocity field  $\mathbf{u}$ . We will use the notation of the material derivative

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + \mathbf{u} \cdot \nabla,$$

and the integral theorems known from vector calculus.

### 1.1 Conservation laws

Consider a property  $L$  that is measurable within a finite, arbitrary volume  $\Omega$  with boundary  $\partial\Omega$ . The rate of change equals the amount that is created/consumed inside the volume or what flows through the boundary. This can be expressed as

$$\frac{d}{dt} \int_{\Omega} L dV = - \int_{\partial\Omega} L \mathbf{u} \cdot \mathbf{n} dA - \int_{\Omega} Q dV,$$

where  $\mathbf{n}$  is the normal vector to the boundary pointing outwards,  $\mathbf{u}$  is the fluid velocity and  $Q$  represents the sources or sinks inside the fluid. If we apply the divergence theorem, this becomes

$$\frac{d}{dt} \int_{\Omega} L dV = - \int_{\Omega} \nabla \cdot (L \mathbf{u}) dV - \int_{\Omega} Q dV,$$

or simply

$$\int_{\Omega} \left( \frac{\partial L}{\partial t} + \nabla \cdot (L \mathbf{u}) + Q \right) dV = 0.$$

Since the volume  $\Omega$  is arbitrary chosen, the integrand must be zero

$$\frac{\partial L}{\partial t} + \nabla \cdot (L \mathbf{u}) + Q = 0. \quad (1)$$

This is the differential form of conservation laws that easily can be applied to any scalar field. It can also be used for vector fields by using dyadic tensors. This will be used to use the conservation law on the momentum  $\rho \mathbf{u}$ .

### 1.2 Conservation of mass

By assuming that no mass is created or destroyed, and applying the conservation equation on the mass, we get

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0.$$

If the fluid is incompressible, we also have  $\rho$  as a constant, reducing the above equation to

$$\nabla \cdot \mathbf{u} = 0, \quad (2)$$

which will play an important part of choice of finite element formulation later on.

### 1.3 Conservation of momentum

We apply the conservation law on the momentum per unit volume  $\rho \mathbf{u}$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) + \mathbf{Q} = 0.$$

The  $\mathbf{Q}$  term, the source or sink, is simply the body force, i.e. external forces that we denote by  $\mathbf{b}$ . The term with  $\mathbf{u} \mathbf{u}$  is a dyad, i.e. a second rank tensor. This then becomes

$$\mathbf{u} \frac{\partial \rho}{\partial t} + \rho \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}(\mathbf{u} \cdot \nabla \rho) + \rho \mathbf{u}(\nabla \cdot \mathbf{u}) + \rho(\mathbf{u} \cdot \nabla) \mathbf{u} = \mathbf{b}.$$

Rearranging

$$\begin{aligned} \mathbf{b} &= \mathbf{u} \left( \frac{\partial \rho}{\partial t} + \mathbf{u} \cdot \nabla \rho + \rho \nabla \cdot \mathbf{u} \right) + \rho \left( \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) \\ &= \mathbf{u} \left( \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) \right) + \rho \left( \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right). \end{aligned}$$

We recognize the first term as the mass conservation equation, which equals to zero, so conservation of momentum reduces to

$$\mathbf{b} = \rho \left( \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) = \rho \frac{D\mathbf{u}}{Dt},$$

where we have the material derivative of the fluid velocity multiplied with the density.

### 1.4 The body force

The source of momentum inside a volume is caused by body forces  $\mathbf{b}$ , but this again can be divided into different kinds of forces, distant forces like gravity or electrostatic forces, and a stress term (friction and normal forces, i.e. stress forces). By using the stress tensor, the body force can be written as

$$\mathbf{b} = \nabla \cdot \sigma + \mathbf{f},$$

which inserted in the momentum conservation looks like

$$\rho \frac{D\mathbf{u}}{Dt} = \nabla \cdot \sigma + \mathbf{f},$$

where  $\sigma$  is the stress tensor, and  $\mathbf{f}$  is the sum of distant forces. The stress tensor is defined in the usual way

$$\begin{aligned} \sigma &= \begin{pmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_y & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_z \end{pmatrix} \\ &= -p\mathbb{1} + \mathbb{T}, \end{aligned}$$

where  $\mathbb{1}$  is the identity matrix,  $p$  is the average pressure

$$p = -\frac{1}{3}(\sigma_x + \sigma_y + \sigma_z),$$

and  $\mathbb{T}$  is the deviatoric stress tensor

$$\mathbb{T} = \begin{pmatrix} \sigma_x + p & \tau_{xy} & \tau_{xz} \\ \tau_{yx} & \sigma_y + p & \tau_{yz} \\ \tau_{zx} & \tau_{zy} & \sigma_z + p \end{pmatrix}.$$

The conservation equation now looks like

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \nabla \cdot \mathbb{T} + \mathbf{f}.$$

This is the Navier Stokes equation in its most general form. There are many degrees of freedom since  $\mathbb{T}$  can be defined in many different ways depending on the fluid properties. A usual simplification is to assume that the divergence of  $\mathbb{T}$  is proportional to  $\nabla^2 \mathbf{u}$  with proportionality constant  $\mu$

$$\nabla \cdot \mathbb{T} = \mu \nabla^2 \mathbf{u},$$

where  $\mu$  is called viscosity, a sort of friction constant. NSE then looks like

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla p + \mu \nabla^2 \mathbf{u} + \mathbf{f}. \quad (3)$$

## 2 Numerical solution of the incompressible equation

In order to solve this with a finite element method, we need to discretize the equation. The only time dependent part of the equation is in the material derivative. We will analyze this with different schemes. We rewrite NSE to

$$\frac{\partial \mathbf{u}}{\partial t} = \nu \nabla^2 \mathbf{u} - \frac{1}{\rho} \nabla p + \mathbf{f} - (\mathbf{u} \cdot \nabla) \mathbf{u},$$

where we have defined the kinematic viscosity  $\nu = \mu/\rho$ , redifined  $\mathbf{f} \rightarrow \mathbf{f}/\rho$ , force per unit volume. We will use a scheme where we can exploit the fact that  $\nabla \cdot \mathbf{u} = 0$  and manage to calculate  $p^{n+1}$ .

### 2.1 The classical splitting method

We use the Forward Euler method to find an expression for  $\mathbf{u}^{n+1}$ , but we evaluate the pressure also at the time step  $n + 1$ . This will give an extra degree of freedom that we can use to make sure that  $\mathbf{u}$  behaves as we know it should

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \nu \nabla^2 \mathbf{u}^n - \frac{\Delta t}{\rho} \nabla p^{n+1} + \Delta t \mathbf{f} - \Delta t (\mathbf{u}^n \cdot \nabla) \mathbf{u}^n.$$

It is now possible to choose  $\nabla p^{n+1}$  so that  $\nabla \cdot \mathbf{u}^{n+1} = 0$ . We will do this by using a temporary  $\mathbf{u}^*$  which we define as

$$\mathbf{u}^* = \mathbf{u}^n + \Delta t \nu \nabla^2 \mathbf{u}^n - \beta \frac{\Delta t}{\rho} \nabla p^n + \Delta t \mathbf{f} - \Delta t (\mathbf{u}^n \cdot \nabla) \mathbf{u}^n, \quad (4)$$

note the factor  $\beta$  in the pressure gradient term. We choose a  $\delta \mathbf{u}$  so that

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^* + \delta \mathbf{u} \\ \delta \mathbf{u} &= \mathbf{u}^{n+1} - \mathbf{u}^* \\ &= -\frac{\Delta t}{\rho} (\nabla p^{n+1} - \beta \nabla p^n) = -\frac{\Delta t}{\rho} \nabla \Phi, \end{aligned}$$

where  $\Phi = p^{n+1} - \beta p^n$ . The incompressibility constraint can now be written as

$$\nabla \cdot \mathbf{u}^{n+1} = \nabla \cdot (\mathbf{u}^* + \delta \mathbf{u}) = 0,$$

or

$$\nabla \cdot \mathbf{u}^* = -\nabla \cdot \delta \mathbf{u}.$$

We use this on our expression for  $\delta \mathbf{u}$

$$\nabla^2 \Phi = \frac{\rho}{\Delta t} \nabla \cdot \mathbf{u}^*,$$

and recognize this as a Poisson equation for  $\Phi$  that can be solved since we know  $\mathbf{u}^*$ . We now have everything we need to evolve the system in time where the velocity is given as

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \frac{\Delta t}{\rho} \nabla \Phi, \quad (5)$$

and the pressure

$$\mathbf{p}^{n+1} = \Phi + \beta p^n. \quad (6)$$

### 3 A Finite Element Method

A popular way to solve the NSE is by using a finite element method. We will work through the details by looking at one part at the time. Assume now that we have given the initial pressure field  $p$  and the initial velocity field  $\mathbf{u}_0$ . Following the recipe described above, we start by finding the  $\mathbf{u}^*$ .

#### 3.1 Variational form for $\mathbf{u}^*$

We introduce basis vectors living in the test space  $V^{(u)}$ , and seek  $\mathbf{u}^*, \mathbf{u}^{n+1} \in V^{(u)}$ . We create a variational form of (4) by multiplying by a test vector  $v \in V^{(u)}$  and integrate

$$\begin{aligned} \frac{1}{\Delta t} \int_{\Omega} (\mathbf{u}^* - \mathbf{u}^n) \cdot \mathbf{v}^{(u)} d\Omega - \nu \int_{\Omega} (\nabla^2 \mathbf{u}^n) \cdot \mathbf{v}^{(u)} d\Omega \\ + \frac{\beta}{\rho} \int_{\Omega} \nabla p^n \cdot \mathbf{v}^{(u)} d\Omega - \int_{\Omega} \mathbf{f} \cdot \mathbf{v}^{(u)} d\Omega + \int_{\Omega} (\mathbf{u}^n \cdot \nabla) \mathbf{u}^n \cdot \mathbf{v}^{(u)} d\Omega = 0. \end{aligned}$$

By integrating the Laplace term and the pressure term by parts, we get

$$\begin{aligned} \frac{1}{\Delta t} \int_{\Omega} (\mathbf{u}^* - \mathbf{u}^n) \cdot \mathbf{v}^{(u)} d\Omega + \nu \int_{\Omega} \nabla \mathbf{u}^n \cdot \nabla \mathbf{v}^{(u)} d\Omega \\ - \frac{\beta}{\rho} \int_{\Omega} p^n \nabla \cdot \mathbf{v}^{(u)} d\Omega - \int_{\Omega} \mathbf{f} \cdot \mathbf{v}^{(u)} d\Omega + \int_{\Omega} (\mathbf{u}^n \cdot \nabla) \mathbf{u}^n \cdot \mathbf{v}^{(u)} d\Omega \\ = \int_{\partial\Omega_{N,u}} \left( \nu \frac{\partial \mathbf{u}}{\partial n} - p^n \mathbf{n} \right) \cdot \mathbf{v}^{(u)} ds \end{aligned}$$

#### 3.2 Variational form for $\Phi$

To solve the Poisson equation for  $\Phi$ , we multiply by a test function  $v^{(\Phi)} \in V^{(\Phi)}$  and integrate

$$\begin{aligned} \int_{\Omega} \nabla^2 \Phi v^{(\Phi)} d\Omega &= \frac{\rho}{\Delta t} \int_{\Omega} \nabla \cdot \mathbf{u}^* v^{(\Phi)} d\Omega \\ \int_{\Omega} \nabla \Phi \cdot \nabla v^{(\Phi)} d\Omega &= -\frac{\rho}{\Delta t} \int_{\Omega} \nabla \cdot \mathbf{u}^* v^{(\Phi)} d\Omega + \int_{\partial\Omega_{n,\Phi}} \frac{\partial \Phi}{\partial n} v^{(\Phi)} ds \end{aligned}$$

When we have found  $\Phi$  and  $\mathbf{u}^*$ , we can use a variational form of (5)

$$\int_{\Omega} \mathbf{u}^{n+1} \cdot \mathbf{v}^{(u)} d\Omega = \int_{\Omega} \left( \mathbf{u}^* - \frac{\Delta t}{\rho} \nabla \Phi \right) \cdot \mathbf{v}^{(u)} d\Omega,$$

similarly for the pressure

$$\int_{\Omega} p^{n+1} v^{(\Phi)} d\Omega = \int_{\Omega} (\Phi + \beta p^n) v^{(\Phi)} d\Omega.$$

### 4 Numerical implementation

#### 4.1 Stability criterion

The numerical solution is based on a discretized time derivative, and the stability criterion is given by [1]

$$\Delta t \leq \frac{h^2}{2\nu + Uh},$$

where  $h$  is the minimum element size and  $U$  is a characteristic size of the velocity. For water and air, the viscosity is less than  $10^{-4}$ , so the stability criterion goes like

$$\Delta t \leq \frac{h}{U},$$

which we will study in more detail later in this report.

## 5 Verification

So far, we've only looked at finite element theory and how to formulate NSE into a variational form. We will now look at two problems that have analytical solutions that we can compare with.

### 5.1 2 dimensional Hagen-Poiseuille flow

We will now look at flow in a 2 dimensional box of length  $L$  ( $x$ -direction) and height  $d$  where we assume that  $\mathbf{u} = (u_x, 0, 0)$ , i.e. the only non zero component is in the  $x$ -direction. The boundaries are not moving, and we will use  $u = u_x$  for simple notation. If we assume that  $\partial_x u_x = 0$ , the NSE looks like

$$\rho \frac{\partial u}{\partial t} = -\frac{\partial p}{\partial x} + \mu \frac{\partial^2 u}{\partial y^2} = -\beta + \mu \frac{\partial^2 u}{\partial y^2}$$

where  $\beta = \partial_x p$ . If we also assume stationary flow, this reduces to

$$\frac{\partial^2 u}{\partial y^2} = \frac{\beta}{\mu}$$

which has the solution

$$u(y) = \frac{\beta}{2\mu} y(d - y).$$

The maximum velocity is found in the middle of the box  $y = d/2$

$$u_{max} = u(d/2) = \frac{\beta}{2\mu} \frac{d^2}{4}. \tag{7}$$

The pressure can be assumed to be linear, so that the gradient is just a constant

$$\beta = \frac{P_A - P_B}{L},$$

which inserted in (7) gives

$$u_{max} = \frac{(P_A - P_B)}{8\mu L} d^2.$$

## 6 Numerical results

We will compare the theoretical maximum velocity  $u_{max}$  with the numerical solution. Our adjustable parameters are

- $\Delta t$  - time step
- $\Delta P$  - pressure difference
- $\mu$  - viscosity
- $L$  - distance between  $A$  and  $B$  in pressure difference
- $d$  - box height

We will use a uniform mesh, a rectangle, of size  $L \times d$  with 10 elements in each direction. A FEniCS implementation of the Hagen-Poiseuille flow can be found in Appendix A. We let FEniCS run until we reach a steady and compare the thermal velocity to the theoretical max given by (7). The results can be found in table 1. Note that the different parameters carry units, but we only look at the numerical values.

#	$\Delta p$	$\mu$	$L$	$h$	Analytical	FEniCS	Error
1	1	1	1	1	0.125	0.125	0.000
2	2	1	1	1	0.250	0.250	0.000
3	1	1	1	2	0.500	0.500	0.000

Table 1: Comparison of the analytical and the numerical solution of the pressure driven stationary system.

## 6.1

# 7 Appendices

## A Poiseuilles law in FEniCS

```

from dolfin import *
import numpy as np

#mesh = UnitSquare(10,10)
mesh = Rectangle(0,0,2,1,20,20,'left')

# Define function spaces (P2-P1)
V = VectorFunctionSpace(mesh, "CG", 2)
Q = FunctionSpace(mesh, "CG", 1)

# Define trial and test functions
u = TrialFunction(V) # Trial function for u
p = TrialFunction(Q) # Trial function for p

v = TestFunction(V) # Test function for u
q = TestFunction(Q) # Test function for p

# Normal vector
n = FacetNormal(mesh)

# Set parameter values
dt = 0.01 # Timestep
T = 40    # Final time
nu = 0.5 # Kinematic viscosity
rho = 1.0 # Density

p_in = Constant(1.0) # Pressure at x=0
#p_in = Expression("sin(2*pi*t)", t=0.0)
p_out = Constant(0.0) # Pressure at x=1
# No slip makes sure that the velocity field is zero at the boundaries
noslip_1 = DirichletBC(V, (0, 0),
                        "on_boundary && \
x[1] < DOLFIN_EPS")

noslip_2 = DirichletBC(V, (0, 0),
                        "on_boundary && \
x[1] > 1 - DOLFIN_EPS")

# We apply a pressure at x=0, no pressure at
inflow = DirichletBC(Q, p_in, "x[0] < DOLFIN_EPS")
outflow = DirichletBC(Q, p_out, "x[0] > 2.0 - DOLFIN_EPS")

bcu = [noslip_1, noslip_2]
bcp = [inflow, outflow]

# Create functions
u0 = Function(V)
u1 = Function(V)
p0 = Function(Q)
p1 = Function(Q)

# Define coefficients
k = Constant(dt)
f = Constant((0, 0))

# Tentative velocity step
U = 0.5*(u0 + u)

```

```

F1 = (1/k)*inner(u - u0, v)*dx \
+ nu*inner(grad(U), grad(v))*dx \
- p0*div(v)*dx \
- inner(f, v)*dx \
+ inner(grad(u0)*u0, v)*dx \
+ inner(p0*n,v)*ds \

a1 = lhs(F1)
L1 = rhs(F1)

# Calculation of phi
a2 = inner(grad(p), grad(q))*dx
L2 = inner(grad(p0), grad(q)) *dx \
- (1.0/k)*div(u1)*q*dx

# Velocity update
a3 = inner(u, v)*dx
L3 = inner(u1, v)*dx - k*inner(grad(p1 - p0), v)*dx

# Assemble matrices
A1 = assemble(a1)
A2 = assemble(a2)
A3 = assemble(a3)

# Create files for storing solution
ufile = File("results/velocity.pvd")
pfile = File("results/pressure.pvd")

# Time-stepping
t = dt
while t < T + DOLFIN_EPS:
    # Update pressure boundary condition
    p_in.t = t

    # Compute tentative velocity step
    b1 = assemble(L1)
    [bc.apply(A1, b1) for bc in bcu]
    solve(A1, u1.vector(), b1, "gmres", "default")

    # Pressure correction
    b2 = assemble(L2)
    [bc.apply(A2, b2) for bc in bcp]
    solve(A2, p1.vector(), b2, "gmres", "amg")

    # Velocity correction
    b3 = assemble(L3)
    [bc.apply(A3, b3) for bc in bcu]
    solve(A3, u1.vector(), b3, "gmres", "default")

    plot(u1, title="Velocity", rescale=True)

    # Save to file
    #ufile << u1
    #pfile << p1

    # Move to next time step
    u0.assign(u1)
    p0.assign(p1)
    t += dt

    u_vec = u1.vector()
    u_array = u_vec.array()
    print "t =", t
    print "max: ", u_array.max()

# Hold plot
interactive()

```

## References

- [1] Prof. Hans Petter Langtangen. *Numerical methods for the Navier-Stokes equations*. URL: [http://hplgit.github.com/INF5620/doc/notes/main\\_ns.pdf](http://hplgit.github.com/INF5620/doc/notes/main_ns.pdf).