Thesis Title

Institution Name

Author Name

Day Month Year

Innhold

Kapittel 1

Verification and Validation

During the last decade, the amount of reaserch regarding simulations of physical problems has grown vast. Even though computers have changed our ways of solving real world problems, thrusting blindly numbers generated from a computer code has proven to be naive. It doesn't take a lot of coding experience before one realizes the many things that can brake down and produce unwanted and even suprisingly unexpected results. With this in mind, computer scientists and engineers need some common ground to check if a computer code works as expected. And it is here the framework of verification and validation plays and important role.

An elegant and simple definition found throughout the litterature of verification and validation framwork, used by Roache [3], states *verification* as "solving the equations right", and *validiation* as "solving the right equations". "Solving the right equations"is rather vaguely, a measurement is needed. We will in this thesis use the more detailed description found in [3].

> The code author defines precisely what continuum partial differential equations and continuum boundary conditions are being solved, and convincingly demonstrates that they are solved correctly, i.e., usually with some order of accuracy, and always consistently, so that as some measure of discretization (e.g. the mesh increments) $\nabla \to 0$, the code produces a solution to the continuum equations; this is Verification.
>
> — *Roache, P.J.*

Roach [2], further distinguish between the verification of *code* and *calculation*. Verification of code is seen as achieving the expected order of accuracy of the implementation, while verification of calculation is the measure of error against a known solution. Of these .. has proven to
The goal of this chapter is to verify our implementations using the method of manufactured solution (MMS).

1.1 Verification of Code

For scientists exploring physical phenomena,systems of partial differential equations (PDE´s) are often encountered. For their application it is important that these equations are implemented and solved numerically the right way. Therefore insurence of right implemention is crucial.

Let a partial differential equation of interest be on the form

$$\mathbf{L}(\mathbf{u}) = \mathbf{f}$$

Here **L** is a differential operator, **u** is variable the of interest, and **f** is some sourceterm.
In the method of manufactured solution, one first manufactures a **u**, which is differentiated with **L** which yields a sourceterm **f**. The sourceterm **f** with respect to the selected solution **u** is then used as input in the implementation, yielding a numerical solution. Verification of code and calculation is then performed on the numerical solution against the manifactured solution **u**.

The beauty of such an approach as mentioned by Roache [2], is that our exact solution can be constructed without any physical reasoning. As such, code verificaion is purly a mathematical exercise were we are only interested if we are solving our equation right. These sentral ideas have existed for some time, but the concept of combining manufactured exact solution in in partnership with refinement studies of of computational mesh has been absent. One of the earliest was Steinberg and Roache [5] using these principles deliberately for *verification of code* (estimate order of convergence)

To deeply verify the robustness of the method of manufactured solution, a report regarding code verification using this approach was published by Salari and Knupp [4]. This thorough work applied the method for both compressible and incompressible time-dependent Navier-Stokes equation. To prove its robustness the authors delibritary implemented code errors in a verified Navier-Stokes solver by MMS presented in the report. In total 21 blind testcases where implemented, where different approaches of verifcation frameworks were tested. Of these 10 coding mistakes that reduces the observed order-of-accuracy was implemented. Here the method of manufactured solution captured all of them.

For the purpose of verification of calculation we need to calculate the error of our numerical simulation. Let $\mathbf{u}_h$ denote our numerical solution and $\mathbf{u}$ be our exact solution. By letting $||\cdot||$ be the L^2 norm, we define the error as

$$E = ||\mathbf{u} - \mathbf{u}_h||$$

Assuming our computational mesh is constructed by equilateral triangles, and that our simulations are solved with a constant timestep, the total error contribution from the temporal and spatial discretized PDE can be written as

$$E = A\delta x^l + B\delta t^k$$

Where A and B are constants, and l and k denote the expected convergencerate... FYLL INN REF FRA ANNET KAP OM EXPECTED CONVERGENCERATE
. In order to evalute properties of either the spatial or temporal discretization, we must reduce the numerical error contribution of the discretization not of interest. Say we would like to evaluate the convergencerate of the spatial discretization, then the temporal error must be reduced in order to not poute..
Even though the method of MMS a certain freedom in the construction of a manufactured solution, certain guidelines have been proposed ([5], [4], [2]).

- To ensure theoretical order-of-accuracy, the manufactured solution should be constructed of polynomials, exponential or trigonometric functions to construct smooth solutions.
- The solution should be utilized by every term in the PDE of interest, such that no term yields zero. (få frem at en løsning må velges slik at ingen differentials blir 0
- Certain degree to be able to calculate expected order of convergence (Få frem at må ha grad noktil å kunne regne convergencerate)

Fluid structure interaction consists of several buildingblocks of fluid and structure equations describing forces exerted from one another. With this in mind a verification of the full FSI code can be tedious as implementation errors yielding non-desired results can be hard to find. We will therefore provide verification of each buildingblock until we reach the total system of equations.

For construction of the sourceterm **f** the Unified Form Language (UFL) [1] provided in FEniCS Project will be used. UFL provides a simple yet powerfull method of declaration for finite element forms. An example will be provided in the Fluid Problem section.

1.1.1 Fluid Problem

Recall from Chapter ??? the ALE formulation of the Navier Stokes equation.

$$\rho_f \hat{\mathrm{J}} \frac{\partial \hat{\mathbf{u}}}{\partial t} + \hat{\mathrm{J}} \hat{\mathbf{F}}^{-1}(\hat{\mathbf{u}} - \hat{\mathbf{w}}) \cdot \nabla \hat{\mathbf{u}} - \nabla \cdot \hat{\mathrm{J}} \sigma \hat{\mathbf{F}}^{-T} = f$$

```
f = rho*diff(u_vec, t_) + rho*dot(grad(u_vec), (u_vec - w_vec)) - div(sigma_f(p_c
```

We will on the basis of the presented guidelines define the manufactured solution.

$$\begin{aligned} u &= sin(x+y+t)^2 \\ v &= cos(x+y+t)^2 \\ p &= cos(x+y+t) \end{aligned}$$

1.2 Turek flag

Bibliografi

[1] Fenics Project. Unified Form Language (UFL) Documentation. 2016.

[2] Patrick J. Roache. Code Verification by the Method of Manufactured Solutions. *Journal of Fluids Engineering*, 124(1):4, 2002.

[3] P.J. Roache. *Verification and Validation in Computational Science and Engineering*. Computing in Science Engineering, Hermosa Publishers, 1998, 8-9, 1998.

[4] Kambiz Salari and Patrick Knupp. Code Verification by the Method of Manufactured Solution. Technical report, Sandia National Laboratories, 2000.

[5] Stanly Steinberg and Patrick J. Roache. Symbolic manipulation and computational fluid dynamics. *Journal of Computational Physics*, 57(2):251–284, 1985.