INSTITUTE OF SCIENTIFIC COMPUTING

TECHNISCHE UNIVERSITÃĎT DRESDEN

MODELING AND SIMULATION SEMINAR

GROUP-SHAPE OPTIMIZATION

Modeling of Thin membranes

Anurag Trivedi, anurag.trivedi@mailbox.tu-dresden.de

Mohammed Parvez, Parvez.Mohammed@mailbox.tu-dresden.de

July 27, 2020

1 Abstract

This work aims to show that a simple, structured, and efficient shape optimization framework can be written by using finite element software AMDiS. This is illustrated by the implementation of a classical technique called geometrical optimization for such type of problem. we modeled theoretical and computational work for 2D Problems to develop robust technique. Our computational Result is very useful for engineers or researchers.

2 Introduction

The goal is to Optimize the shape of a thin Membrane subjected to certain load and boundary conditions. We explore Quadrature with the finite element method. In PDE-constrained optimization, we have two choices for discretization whether to discretize-then-optimize or optimize-then-discretize. we have chosen a discretize-then-optimize approach. because most optimization problems are naturally self ad-joint and by this choice, we can avoid non-symmetry due to discretization that can arise with the optimize-then-discretize approach. So that we are able to use the iterative method is the projection Gradient method. However, it would be impractical to write the code for the entire finite element method from scratch. This project aims to use AMDiS, open-source software for FEM, to deal with the nuances of FEM such as mesh generation, calculation of numerical solutions of PDEs, etc then write simple shape optimization routines.

2.1 Shape Optimization of Structure

The problem of Shape Optimization of structures is defined by three ingredients

1. A Mathematical model to evaluate the mechanical behavior of a structure.
2. an Objective function which has to be minimized or maximized, or sometimes several objectives (also called cost-functions or criteria).
3. Set of admissible thicknesses which precisely defines the optimization variables, including possible constraints.

2.1.1 Mathematical Model

In our mathematical model, we consider our membrane to be fixed at the boundaries and subject to some vertical force $f : \Omega \to R$. Due to very small deformations, the membrane can be modeled by its vertical displacement $u : \Omega \to R$, the solution to PDE.

$$-div(h\nabla u) = f in\Omega$$
$$u = 0 on\partial\Omega \tag{2.1}$$

The thickness h is our optimization variable. It is bounded by some given minimum and maximum values:

$$0 < h_{min} \le h(x) \le h_{max} < \infty$$

2.1.2 Objective function

The optimization of the structure can be linked to some mechanical properties.

$$J(h) = \int_\Omega j(u)dx$$

It could be the optimization to achieve a target displacement

$$j(u) = |u - u_0|^2$$

It could also be to minimize the max stress-induced.

$$J(h) = sup_{x\in\Omega}|\sigma(x)|$$

We, however, in this project are optimizing for compliance(-rigidity)

$$j(u) = fu$$

2.1.3 Set of admissible thicknesses

Volume constraint

Many structural optimization problems feature a volume constraint on the admissible open sets.

$U_{ad} = \{h \in L^\infty(\Omega)$:0< h$_{min} \le h(x) \le h_{max} < \infty$ in Ω, $\int_\Omega h(x)dx = h_0|\Omega|\}$

In this project, we have considered our thickness function to be piece-wise constant on a coarse grid other than the discretization grid that is used in the FEM formulation by AMDiS.

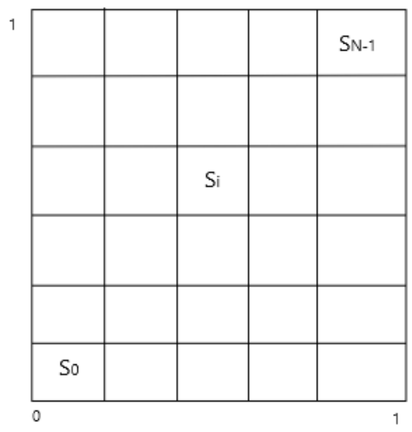


Fig 1-Grid

Hence our admissible open set is

$$U_{ad} = \{h(x) = \sum_{i=0}^{N-1} h_i I_{s_i}(x); 0 < 0.1 \leq h_i(x) \leq 5, \sum_{i=0}^{N-1} h_i(x) = N\}$$

Where N is the total number of grid cells and I_{si} is the indicator function for $i'th$ rectangular grid cell.

This constraint is imposed by introducing the Lagrange multiplier $'l'$ in the formulation. Hence, the descent direction is now computed from the shape derivative of the Lagrangian

$$J'(\Omega) + lV'(\Omega)$$

where $V(\Omega)$ is volume of Ω, $'l'$ is Lagrange multiplier.
The Lagrange multiplier is updated in each iteration by

$$l = 2\left(\frac{\int_\Omega h^{\tilde{n}+1} dx - 1}{N}\right)$$

Where $\tilde{h_n}$ is the thickness that is obtained from gradient descent without projection(see section 3.1)

3 Numerical algorithm

We are presented numerical schemes for solving these kinds of the minimization problem. The iteration schemes starting from giving initial value h^0 we construct a sequence $(h^n)_{n\in N}$, which can be shown to converge to solution u of our consider minimization problem.

3.1 Gradient method

Gradient descent is one of the most simplest and popular algorithm to perform optimization by far the most common way to use when the parameter cannot be calculated analytically and must be searched for by an optimization algorithm. We included some more detailed.

Suppose that $V = R^N$ (a Hilbert space with its dual V'). We consider the minimizing problem without non-constrained case:

$$inf_{v\in V} J(v) \tag{3.1}$$

1- pick on Initial point $h^0 \in V$

2- Loop until stopping condition is meet.

(a)-Descent direction: pick the descent direction as $\rightarrow \nabla J'(u(h^n)$

(b)-Step Size: μ is a positive parameter that we choose involving the optimal choice of $\mu = \mu^n$

(c)- Iterative update: $h^{\tilde{n+1}} = h^n - \mu J'(u(h^n))$

(d)- Now the gradient of J is given by

$$\nabla J(u) = \nabla u . \nabla p$$

Where 'u' is the solution to equation(2.1) with $h = h^n$

Find 'p' is the ad-joint state given by the solution to

$$\begin{aligned} \text{-div}(\text{h}^n \nabla p) &= j'(u) in\, \Omega \\ p &= 0 on \text{d}\Omega \end{aligned} \tag{3.2}$$

Here we optimize for compliance so the PDE is reduced to

$$\begin{aligned} \text{-div}(\text{h}^n \nabla p) &= -f in\, \Omega \\ p &= 0 on \text{d}\Omega \end{aligned} \tag{3.3}$$

$\rightarrow$ p = -u, Therefore, we don't have to solve two PDE's to calculate the gradient and It could be easily calculated from the previous solution u^n by the formula.

$$\text{J'(u)} = -\nabla u * \nabla u$$

3.2 Projection Gradient method

Let J is real valued strictly convex differentiable functional defined on a non-empty closed convex subset K of the Hilbert space V. The set K represents the imposed constraints. We consider the minimizing problem with constrained case.

$$inf_{v \in K} J(v) \tag{3.4}$$

Let $P_k : V \to K$ denoted the projection operator onto the convex subset K. Then h^{n+1} is the orthogonal projection of $(h^n - \mu J'(h^n))$ onto K.

1- pick on Initial point $h^0 \in V$

2- Loop until stopping condition is meet.

(a)-Descent direction: pick the descent direction as $\to \nabla J'(h^n)$

(b)-Step Size: μ is a positive parameter that we choose involving the optimal choice of $\mu = \mu^n$

(c)- Iterative update: $h^{n+1} = P_k(h^n - \mu J'(h^n))$

(d)- Projection $h^{n+1} = argmin_{h \in V} \| h^{n+1} - \tilde{h}^{n+1} \|^2_{L^2}$

3- we compute projection by the formula

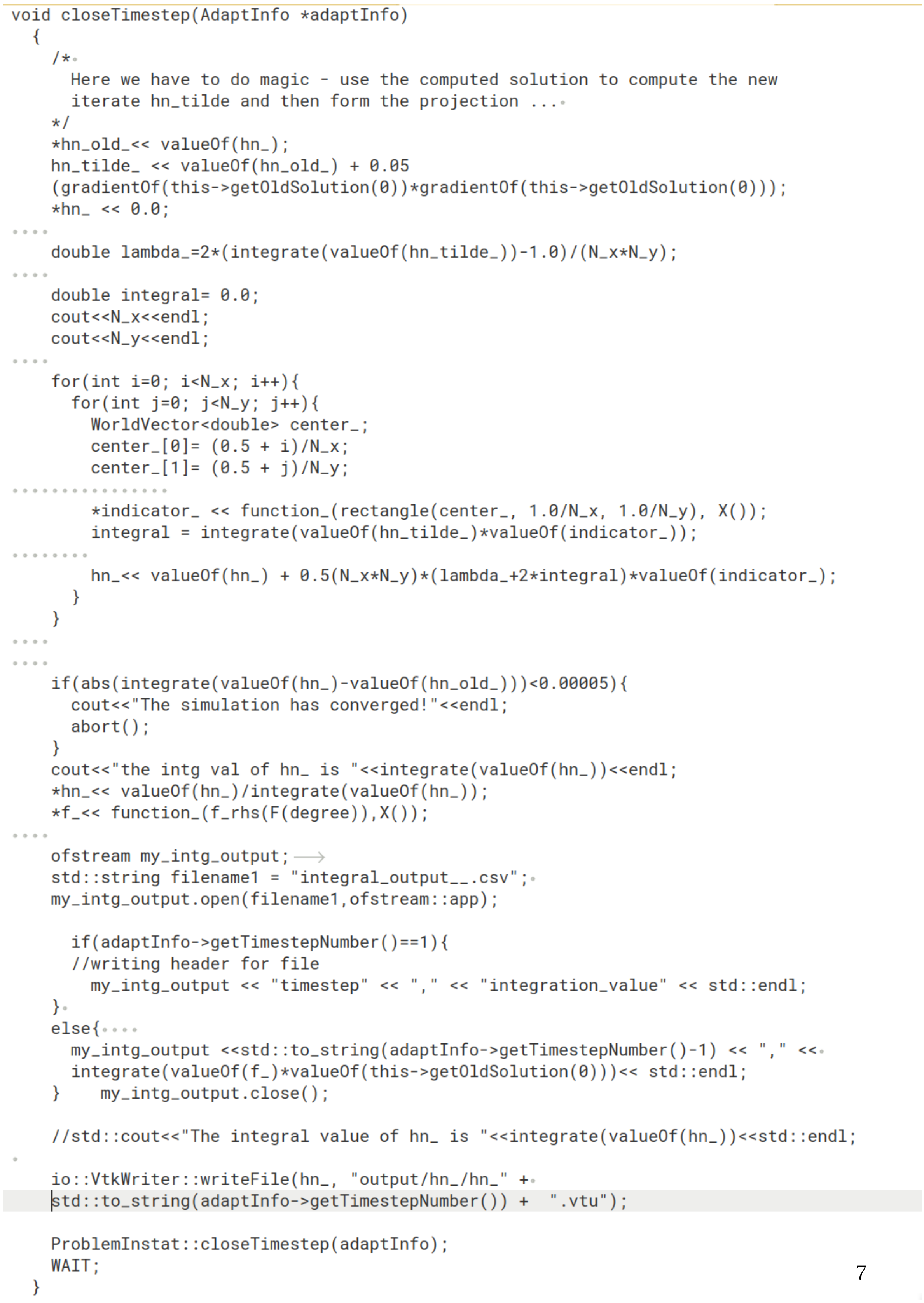
$$h_i^{n+1} = P_{k_i} = \frac{N}{2}(l + 2\int_{S_i} h^{\tilde{n+1}})$$

where P_{K_i} is projection for the thickness in the $i'th$ rectangular cell.

4 Convergence Criteria

A typical convergence criterion for stopping the optimization loop would be to check that the gradient of the cost function is small. However, due to numerical discretionary errors this criterion is hopeless. One possible solution to this would be fixed some number of iterations. However, here we have plotted the cost function $J(u)$ w.r.t the number of iterations and choose the h^n that yielded in the least cost function.

5 Computational Task



```cpp
void closeTimestep(AdaptInfo *adaptInfo)
  {
    /*
      Here we have to do magic - use the computed solution to compute the new
      iterate hn_tilde and then form the projection ...
    */
    *hn_old_<< valueOf(hn_);
    hn_tilde_ << valueOf(hn_old_) + 0.05
    (gradientOf(this->getOldSolution(0))*gradientOf(this->getOldSolution(0)));
    *hn_ << 0.0;

    double lambda_=2*(integrate(valueOf(hn_tilde_))-1.0)/(N_x*N_y);

    double integral= 0.0;
    cout<<N_x<<endl;
    cout<<N_y<<endl;

    for(int i=0; i<N_x; i++){
      for(int j=0; j<N_y; j++){
        WorldVector<double> center_;
        center_[0]= (0.5 + i)/N_x;
        center_[1]= (0.5 + j)/N_y;

        *indicator_ << function_(rectangle(center_, 1.0/N_x, 1.0/N_y), X());
        integral = integrate(valueOf(hn_tilde_)*valueOf(indicator_));

        hn_<< valueOf(hn_) + 0.5(N_x*N_y)*(lambda_+2*integral)*valueOf(indicator_);
      }
    }


    if(abs(integrate(valueOf(hn_)-valueOf(hn_old_)))<0.00005){
      cout<<"The simulation has converged!"<<endl;
      abort();
    }
    cout<<"the intg val of hn_ is "<<integrate(valueOf(hn_))<<endl;
    *hn_<< valueOf(hn_)/integrate(valueOf(hn_));
    *f_<< function_(f_rhs(F(degree)),X());

    ofstream my_intg_output;
    std::string filename1 = "integral_output__.csv";
    my_intg_output.open(filename1,ofstream::app);

      if(adaptInfo->getTimestepNumber()==1){
      //writing header for file
        my_intg_output << "timestep" << "," << "integration_value" << std::endl;
    }
    else{
      my_intg_output <<std::to_string(adaptInfo->getTimestepNumber()-1) << "," <<
      integrate(valueOf(f_)*valueOf(this->getOldSolution(0)))<< std::endl;
    }    my_intg_output.close();

    //std::cout<<"The integral value of hn_ is "<<integrate(valueOf(hn_))<<std::endl;

    io::VtkWriter::writeFile(hn_, "output/hn_/hn_" +
    std::to_string(adaptInfo->getTimestepNumber()) +  ".vtu");

    ProblemInstat::closeTimestep(adaptInfo);
    WAIT;
  }
```

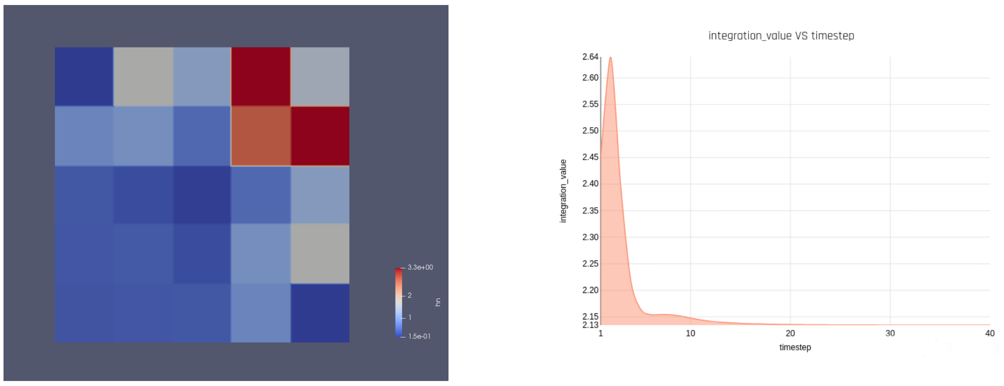


Figure 6.1: Example-1: left (a) Visualization of thickness, right (b)plot of convergence

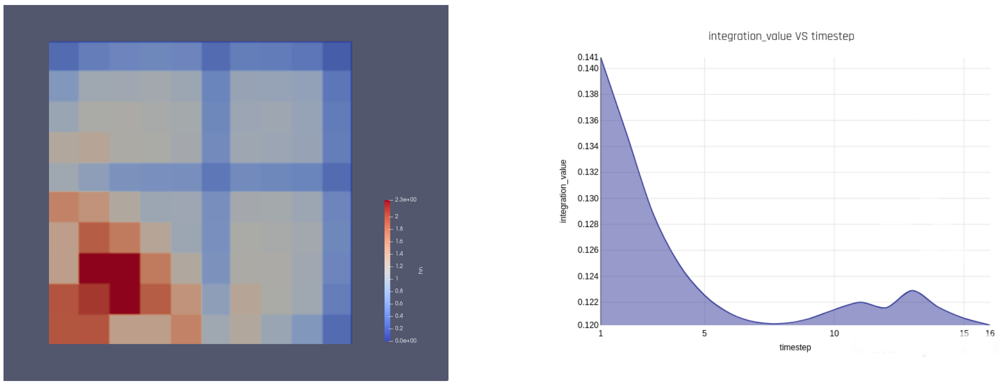


Figure 6.2: Example-2: left(a) Visualization of thickness, right (b)plot of convergence

6 Results

We illustrate our method with some different examples. We consider simple stationary problem as an in-stationary one with Dirichlet boundary condition.

In Example-1.Figure 6.1(a) we consider force $f = 10 * x * y$ with grid spacing $N = N_x * N_y$= 5*5 = 25.(b) plot of our optimal thickness level.

In Example-2.Figure 6.2(a) we consider force $f = -(400x^2 - 20d)e^{-}10x^2$ with grid spacing $N = N_x * N_y$= 10*10 = 100 (b) plot of our optimal thickness level.

The result is quite intuitive as the result have a higher thickness in the regions with strong gradients.

7 Future Scope

To be honest, there is quite a big room for improvement. Some of the improvements that can be made are.

1. We have just optimized for rigidity. This can be extended to optimize for maximum stress-induced or to achieve a target displacement.

2. In this project, we have considered the thickness to be a piece-wise constant. A continuous function in L-infinity is also possible.

3. An analysis could be done to see the scaling behavior of the model.

4. Adaptive step length could also be used in the gradient method.

5. In all our tests, we have initialized our thickness as h=1. We could test different starting points.

6. Linearization could be done to improve the computational time.

Acknowledgment

This work is done under the guidance of Prof. Dr. Stefan Neukamm, Mario Varga, and Dennis Wenzel. We thank all of you for providing information and resources.

References

[1] Allaire, G., Cavallina, L., Miyake, N., Oka, T., Yachimura, T. (2019). *The homogenization method for topology optimization of structures: old and new.* Interdisciplinary Information Sciences, 25(2), 75-146. (visit Allaireetal)

[2] Braess, Dietrich. Finite elements. *Theory, fast solvers, and applications in-solid mechanics..* Cambridge University Press, 2007