Vienna University of Technology

Course Title - VO/VU/UE/SE

Institute where profs from course are from

Document Title

Authors:
Camilo Tello Fachin
12127084

Second Author
Mat.Nr

Supervisor:
Prof. Dr. Super Visor

October 31, 2022

Zusammenfassung

Zusammenfassung in Deutsch!

Abstract

Abstract in English! We have already established that Faustmann [3] is a tough MF!

Contents

To Do's

- What's still to you do make your supervisor/prof happy?

Chapter 1

Chapter 1

Lieber Wappler, vergiss net die pdf metadata anzupassen! des is im `02_settings.tex` file drinne!

1.1 Chapter 2

1.1.1 Pseudo Code Environment Example

Algorithm PDE Constrained Shape Optimization in NGSolve

```
1: resetDeformation
2: initializeParameters $\alpha$, $\beta$, $\gamma$    ▷ Aug. Lag. weights for
       $\mathrm{vol}(\Omega_\mathrm{i})$, $\mathrm{bc}_x(\Omega_\mathrm{i})$, $\mathrm{bc}_y(\Omega_\mathrm{i})$
3: for i < $\mathrm{iter}_{\max}$ do
4:     SolveStokes()    ▷ Solve Stokes on $\Omega_i$
5:     SolveDeformationEquation()    ▷ Solve Auxiliary Problem on $\Omega_i$, yields $X$
6:     Evaluate gfxbndnorm = $||X||_{\mathrm{L}^2(\Gamma_{\infty,\mathrm{i}})}$
7:     Evaluate ScalingParamter = $\frac{0.01}{||X||_{\mathrm{L}^2(\Omega_\mathrm{i})}}$
8:     if gfxbndnorm < $\varepsilon$ then
9:         Increase $\alpha$, $\beta$, $\gamma$
10:        if parametersTooBig then
11:            break
12:        end if
13:    end if
14:    Set $\Omega_{i+1} = \Omega_i - X\cdot$ ScalingParameter    ▷ Gradient Descent Step
15: end for
```

1.2 Chapter 3

1.2.1 Proof and Theorem Environment Example

As explained in the introduction, the existence of the shape derivative needs to be shown. The perturbations of the shape Ω are described by the transformation: $\Omega_t := (Id + tX)(\Omega)$ For small perturbations and $t > 0$, the shape derivative is [2]

$$DJ(\Omega)(X) := \left(\frac{\partial}{\partial t} J(\Omega_t)\right)\Big|_{t=0} = \lim_{t\to 0} \frac{J(\Omega_t) - J(\Omega)}{t}. \tag{1.1}$$

This notion of the shape derivative is used in this chapter in the context of differentiability. The functional $J(\Omega)$ that returns a scalar quantity represetative of the energy dissipation is shown here where $:$ is the Frobenius product and $\mathrm{D}\mathbf{u}$ is the Jacobi matrix of $\mathbf{u}$.

$$J(\Omega) = \frac{1}{2}\int_\Omega \mathrm{D}\mathbf{u} : \mathrm{D}\mathbf{u}\,\mathrm{dx}. \tag{1.2}$$

Sturm et. al. [1] proposed the following shape derivative.

Theorem. *The Shape Derivative of J at Ω in direction $X \in [C^{0,1}(\bar{\Omega})]^2$ is given by:*

$$\mathrm{d}J(\Omega)(X) = \int_\Omega \mathrm{S}_1 : \mathrm{D}X\,\mathrm{dx}, \tag{1.3}$$

$$\mathrm{S}_1 = \left(\frac{1}{2}\mathrm{D}\mathbf{u} : \mathrm{D}\mathbf{u} - p\,\mathrm{div}(\mathbf{u})\right)\mathrm{I}_2 + \mathrm{D}\mathbf{u}^\top p - \mathrm{D}\mathbf{u}^\top\mathrm{D}\mathbf{u}. \tag{1.4}$$

*where $(\mathbf{u}, p)$ solve (**??**)*

Proof. *let $X \in [C^{0,1}(\bar{\Omega})]^2$ with $X|_{\Gamma_\infty} = 0$ be a given vectorfield.*
*Set $\mathrm{T}_t(.) := \mathrm{id} + tX$, with $t \in \mathbb{R}$ and $\Omega_t := \mathrm{T}_t(\Omega)$, where $(\mathbf{u}_t, p_t)$ solve (**??**) and Ω is replaced by Ω_t s.t. $p_t \in \mathrm{L}^2(\Omega_t), \int_{\Omega_t} p_t\,\mathrm{dx} = 0$ and $\mathrm{u}_t \in [H^1(\Omega_t)]^2$. Then there holds*

$$\int_{\Omega_t} \mathrm{D}\mathbf{u}_t : \mathrm{D}\mathbf{v} + \mathrm{div}(\mathbf{v})\,p_t + \mathrm{div}(\mathbf{u}_t)\,q\,dx = 0 \quad \forall (v, q) \in [H_0^1(\Omega_t)]^d \times L^2(\Omega_t). \tag{1.5}$$

Introduction of change of variables shows that $(\mathbf{u}^t, p^t) := (\mathbf{u}_t \circ \mathrm{T}_t, p_t \circ \mathrm{T}_t)$ satisfy

$$\begin{gathered}\int_\Omega \det(\mathrm{DT}_t)\left(\mathrm{DT}_t^{-1}\mathrm{D}\mathbf{u}^t : \mathrm{DT}_t^{-1}\mathrm{D}\mathbf{v} - p\,\mathrm{tr}(\mathrm{D}\mathbf{v}\mathrm{DT}_t^{-1}) + q\,\mathrm{tr}(\mathrm{D}\mathbf{u}\mathrm{DT}_t^{-1})\right)\mathrm{dx}, \\ \forall (v, q) \in [H^1(\Omega)]^2 \times L^2(\Omega),\end{gathered} \tag{1.6}$$

Used in equation (1.6)

$$\begin{aligned}\mathrm{D}\mathbf{v} \circ \mathrm{T}_t &= \mathrm{D}(\mathbf{v} \circ \mathrm{T}_t), \\ \mathrm{div}(\mathbf{v}) &= \mathrm{tr}\left(\mathrm{D}(\mathbf{v} \circ \mathrm{T}_t)(\mathrm{DT}_t^{-1})\right).\end{aligned}$$

The functional $J(\Omega, \mathbf{u})$ is now reduced to the functional $J(\Omega)$, since the change of the quantities $(\mathbf{u}, p)$ is taken into account by the transformation theorem. The minimum of (1.2) satisfies the saddlepoint problem (1.6). It can be obtained with the Lagrange Multiplier method, see Faustmann [3]. The corresponding Lagrangian which can be used to minimize (1.2) is

$$\begin{aligned}\mathcal{L}(t, \mathbf{v}, q) = \frac{1}{2}\int_\Omega &\det(\mathrm{DT}_t)\mathrm{D}\mathbf{v}(\mathrm{DT}_t)^{-1} : \mathrm{D}\mathbf{v}(\mathrm{DT}_t)^{-1}\,\mathrm{dx},\\ -\int_\Omega &\det(\mathrm{DT}_t) q \,\mathrm{tr}\left(\mathrm{D}\mathbf{v}(\mathrm{DT}_t)^{-1}\right).\end{aligned} \tag{1.7}$$

To find the shape derivative, one can now derive this parametrized Lagrangian, for details on the derivation of parametrized Lagrangians, see K. Ito et. al. **lagrangian˙derivative**. *With the derivative of the Lagrangian obtained, it holds true that*

$$\mathrm{d}J(\Omega)(X) = \frac{\mathrm{d}}{\mathrm{d}t}\mathcal{L}(t, \mathbf{u}^t, 0)\big|_{t=0} = \frac{\partial}{\partial t}\mathcal{L}(0, \mathbf{u}, p) = \int_\Omega \mathrm{S}_1 : \mathrm{D}X\,\mathrm{dx}. \tag{1.8}$$

□

1.3 Chapter 4

1.4 Chapter 5

1.5 Chapter 6

1.6 Chapter 7

1.7 Results and Conclusion

Bibliography

[1] J. Iglesias, K. Sturm, and F. Wechsung, “Two-Dimensional Shape Optimization with Nearly Conformal Transformations,” *SIAM Journal on Scientific Computing*, vol. 40, A3807–A3830, Jan. 2018. DOI: https://doi.org/10.1137/17M1152711.

[2] P. Gangl, K. Sturm, M. Neunteufel, and J. Schöberl, “Fully and Semi-automated Shape Differentiation in NGSolve,” *Structural and multidisciplinary optimization*, vol. 63, no. 3, pp. 1579–1607, 2021. DOI: https://doi.org/10.1007/s00158-020-02742-w.

[3] M. Faustmann and J. Schoeberl, “Lecture notes for Numerical Methods for PDE’s - TU Vienna ASC,” Jun. 2022.

.1 Python Code Listing

Here is an example of a python listing, you can change appearance of comments, strings, numbering, known commands and variables in the package settings in packages.tex. You can obviously use the listings environment in the rest of the document. The same procedure applies for listings in other languages.

Python Listing Title

```
  ]
# Python Script, API Version = V18

import math

#    DELETE EVERYTHING ------------------------------

 ClearAll ()

#    PARAMETERS ---------------------------------------

w = float(Parameters.w)       # side length of one element or half of a unit cell
e = float(Parameters.e)       # rectangle ratio e
b = w/(1+e)
rho = float(Parameters.rho)   # relative density
f = float(Parameters.f)       # number of layers = folds+1
h = 2*w/f          # layer height = size of a unit cell divided by the number of layers
f = int(Parameters.f)

# Calculation of wall thickness t
t1 = ((math.sqrt(1-rho)+1)*math.sqrt(2)*w)/2
t2 = -((math.sqrt(1-rho)-1)*math.sqrt(2)*w)/2
 if t1 <t2:
   t=t1
else:
    t=t2

# auxiliary variable to build up rectangle
m = math.sqrt(pow(t,2)*2)/2
```

.2 XMl Code Listing

Here is an example for XML code listing.

XML Listing Title

```xml
<extension version="1" name="EnergyIntegral" loadasdefault="True">
  <guid shortid="EnergyIntegral">8005c624-8869-4c74-b32b-97ac59c200b2</guid>
  <script src="energy_integral.py" />
  <interface context="Mechanical">
```

.3 MATLAB Code Listing

Here is an example for MATLAB code listing

MATLAB Listing Title

```
%% Linear model Poly44 from MATLAB Curve Fit App:

%Polynomial Coefficients (with 95\% confidence bounds):
       p00 =       13.79;  %(13.22, 14.36)
       p10 =      -2.897;  %(-3.454, -2.34)
       p01 =       3.752;  %(3.163, 4.34)
       p20 =       3.279;  %(2.231, 4.327)
       p11 =      0.5404;  %(-0.2001, 1.281)
       p02 =      0.8638;  %(-0.4624, 2.19)
       p30 =       0.299;  %(0.01281, 0.5851)
       p21 =     -0.5091;  %(-0.7299, -0.2884)
       p12 =      0.4973;  %(0.2716, 0.7229)
       p03 =      0.3595;  %(0.04484, 0.6741)
       p40 =     -0.8495;  %(-1.291, -0.4084)
       p31 =    -0.02258;  %(-0.3136, 0.2685)
       p22 =     -0.2819;  %(-0.5502, -0.01351)
       p13 =      0.2674;  %(-0.05265, 0.5874)
       p04 =      0.2019;  %(-0.3968, 0.8006)

    f(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p30*x^3 + p21*x^2*y
    + p12*x*y^2 + p03*y^3 + p40*x^4 + p31*x^3*y + p22*x^2*y^2
    + p13*x*y^3 + p04*y^4

  %Goodness of fit:
  %SSE: 3.189
  %R-square: 0.9949
  %Adjusted R-square: 0.9902
  %RMSE: 0.4611
```