DEPARTMENT MATHEMATIK
UND INFORMATIK
DER UNIVERSITÄT ZU KÖLN

Sommersemester 2019

Prof. Dr. A. Klawonn
J. Knepper, M. Sc.

1. Mai 2019

5. Übung zu Wissenschaftliches Rechnen II

Aufgabe 1: (5 Punkte)
Sei $u \in \left(H_0^1(\Omega)\right)^3$ Lösung des Verschiebungs- oder Hellinger-Prange-Reissner-Ansatzes mit homogenen Dirichletrandwerten. Zeigen Sie, dass dann

$$\int_\Omega \operatorname{tr}(\sigma(u))\,\mathrm{d}x = 0$$

gilt.

Programmieraufgabe 1 (Ebener Spannungs- und Verzerrungszustand): (10 Punkte)
Nutzen Sie für alle Teilaufgaben $\mathcal{P}_1$-Elemente zur Diskretisierung.

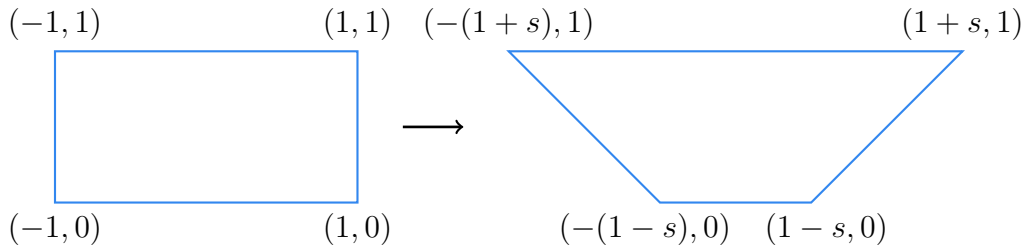
- Implementieren Sie für Programmieraufgabe 1 des letzten Übungsblattes den ebenen Spannungs- und Verzerrungszustand. Nutzen Sie dieselbe Gitterauflösung in die entsprechenden Richtungen wie für die 3D-Simulation.
- Modifizieren Sie zudem Ihre 3D-Simulation und lassen Sie keine Verschiebung in die Richtung der Tiefe des Trägers zu. Sofern die Tiefe bei Ihnen durch die y-Koordinate kodiert ist, können Sie dies mit dem folgenden Befehl tun:

```
bc2 = DirichletBC(V.sub(1),Constant(0),lambda x,b:True)
```

  Ist in `bc1` die andere Dirichletbedingung (Haftbedingung links und rechts) gespeichert, so wird das zusammengesetzte Objekt über `bc = [bc1,bc2]` definiert.
- Stellen Sie die Ergebnisse zur maximalen Verschiebung der 4 Simulationen gegenüber. Geben Sie zur Einordnung auch die Größe der FEM-Matrix mit an.
- Kommentieren Sie, inwiefern die jeweiligen Modellannahmen zum Problem passen und welches Modell der Realität am nächsten kommt.

Programmieraufgabe 2 (Locking): (15 Punkte)

a) Stellen Sie ein Variationsproblem zur linearen Elastizität auf, sodass durch dessen Lösung die Deformation von $[-1,1]\times[0,1]$ einen symmetrischen Trapez ergibt; siehe Abbildung. Die Lamé-Parameter μ, λ seien dabei beliebig. Starrkörperbewegungen sollen durch das Festhalten der Kante $(0,y)$, $y \in [0,0.04]$, aus dem Kern des Problems entfernt werden. Sei $0 \leq s < 1$.



Überprüfen Sie, ob die Verschiebungslösung eine zulässige Deformation ergibt.

b) Implementieren Sie die Diskretisierung des Variationsproblems in FEniCS und nutzen Sie $2 \cdot 50^2$ viele $\mathcal{P}_1$-Dreiecke. Setzen Sie $s = 0.01$ für die Deformation und $E = 1$. Testen Sie $\nu \in \{0.4, 0.49, 0.499, 0.4999\}$ und vergleichen Sie die Verschiebung der Theorie mit der numerischen Lösung in den Punkten $(1, 0)$ und $(1, 1)$ in x-Richtung. Geben Sie zusätzlich den relativen L^2-Fehler an. Sofern `uExact` die Funktion der exakten Lösung in FEniCS angibt, so können Sie zur Fehlerberechnung den folgenden Code verwenden:

```
uExactL2 = sqrt(assemble(uExact**2*dx(mesh)))
errRel = errornorm(uExact,u,'L2')/uExactL2
```

Hinweis: Stellen Sie die Funktion g zur Flächenkraft nicht als `UserExpression` auf, sondern nutzen Sie eine Matrix-Vektormultiplikation und `FacetNormal(mesh)`, um Normalenvektoren zu bestimmen. Sei `mat` $= \sigma(u)$, wobei u die exakte Lösung ist. Sie können `mat` wie folgt aufstellen:

```
mat = Expression(
       (
              ('a','b'),
              ('c','d')
       ),
       s=s,lambda_=lambda_,mu=mu,degree=1)
```

Anschließend stellen Sie das Objekt zu den Normalen mit `nf = FacetNormal(mesh)` auf und können es mit `dot(mat,nf)` in das lineare Funktional integrieren.

c) Verdoppeln Sie nun den Polynomgrad und vergleichen Sie die Ergebnisse dafür mit denen einer Verdoppelung der Auflösung.

Tipp: Mit `dofs = len(V.dofmap().dofs())` können Sie die Anzahl Freiheitsgrade für den Finite-Elemente-Raum `V` ermitteln.

d) Implementieren Sie nun mit Hilfe der Funktion `AdaptiveLinearVariationalSolver` für $\mathcal{P}_1$-Elemente und einer Startauflösung von $2 \cdot 50^2$ Dreiecken eine adaptive Gitterverfeinerung. Nutzen Sie als Zielfunktion

```
u = Function(V)
u1, u2 = split(u)
M = (u1+u2)*dx()
```

Wählen Sie als Toleranz zur Zielfunktion 10^{-5} und zur Lösung der Gleichungssysteme einen direkten Löser. Plotten Sie das adaptiv verfeinerte Gitter zu $\nu = 0.4$ mit

```
import matplotlib.pyplot as plt
fig = plt.figure()
plot(u.leaf_node().function_space().mesh())
```

Geben Sie zudem die Anzahl Freiheitsgrade des adaptiv verfeinerten Gitters an:

```
len(u.leaf_node().function_space().dofmap().dofs())
```

Vergleichen Sie die Ergebnisse mit den vorherigen.

Hinweis: Die Lösung `u.leaf_node()` ist nicht mehr vom Typ `Function`. Zum Umwandeln muss die Lösung in den FE-Raum interpoliert werden: `interpolate(u.leaf_node(),V)`.

Stellen Sie Ihre Ergebnisse (inkl. Plots) übersichtlich dar ($\rightarrow$ Latex).

Abgabe: Bis Mittwoch, 8. Mai 2019 , 14:00 Uhr, im entsprechenden Kasten in Raum 3.01 des Mathematischen Instituts.