(h) Implementierung in Matlab

- Triangulierung:

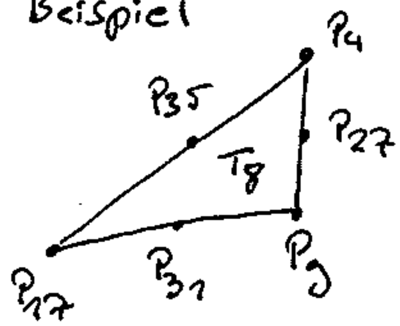
Für quadratische Finite Elemente müssen die Seitenmittelpunkte der Dreiecke als neue Knoten definiert werden. Das liefert die Funktion

$$[T2, P2, EdgesDir2, EdgesNeu2] = trieg\_lin2quad(T, P, EdgesDir, EdgesNeu)$$

die von uns bereitgestellt wird.

T2 hat sechs Spalten: Indizes der Ecken und Seitenmittelpunkte

Beispiel



$$\longrightarrow T2(8,:) = [\underbrace{17 \;\; 9 \;\; 4}_{\text{Ecken}} \;\; 31 \;\; 27 \;\; 35]$$

Mittelpunkt von $[P_{17}, P_9]$

$[P_9, P_4]$

$[P_4, P_{17}]$

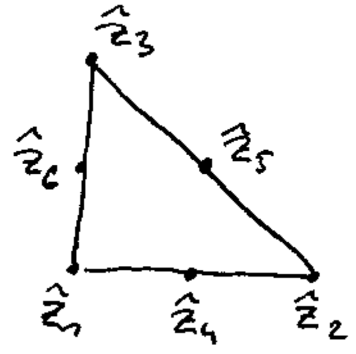
- Berechnung der Lastmatrix und des Steifigkeitsvektors:

Definiere zuerst die Quadraturpunkte (ξ_n, η_n), $n = 1, 2, 3$, und berechne

$$\hat{\varphi}_j(\xi_n, \eta_n), \quad \partial_r \hat{\varphi}_j(\xi_n, \eta_n), \quad \partial_s \hat{\varphi}_j(\xi_n, \eta_n), \quad j = 1, \dots, 6$$

Speichere die Werte in drei 3×6-Matrizen ab.

Achtung! Die Nummerierung der $\hat{\varphi}_j$ muss der Nummerierung der Knoten der Dreiecke entsprechen:



$$\hat{\varphi}_i(\hat{z}_j) = \begin{cases} 1 & \text{für } i = j \\ 0 & \text{sonst} \end{cases}$$

Ich hatte in (c) eine andere Nummerierung verwendet. Das war nicht falsch, aber ungünstig.

Gehe vor wie bei linearen Elementen (Aufgabe 2 bzw. 3).
Auf dem Dreieck T_k werden nun

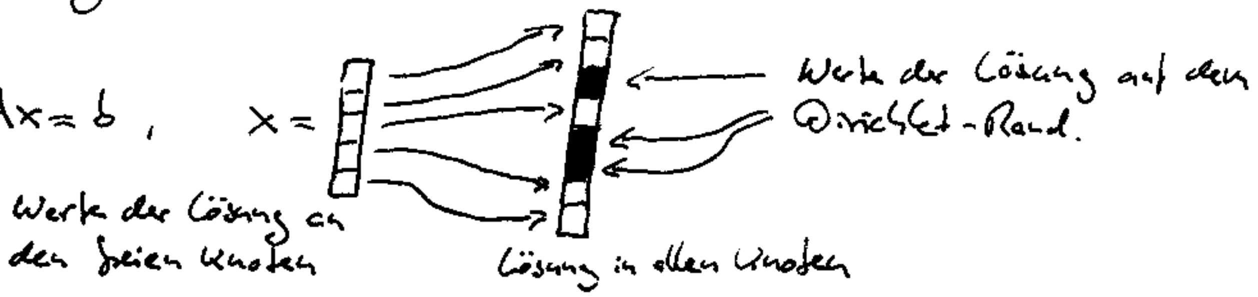
$$a_{T_k}(\varphi_i^k, \varphi_j^k) \quad \text{und} \quad \ell_{T_k}(\varphi_j^k) \qquad \text{für } i,j = 1,\dots,6$$

berechnet und zu den entsprechenden Einträgen in A und b dazuaddiert.

Entferne am Ende Zeilen / Spalten von Knoten, die auf dem Dirichlet-Rand liegen.

Löse LGS und füge danach Werte auf dem Dirichlet-Rand zur Lösung hinzu.

$Ax = b$, $\quad x =$



90

- Berechnung der L^2-Norm des Fehlers

$$\|u - u_h\|^2_{L^2(\Omega)} = \sum_k \int_{T_k} |u(x,y) - u_h(x,y)|^2 \, d(x,y)$$

Zur Approximation des Integrals könnte man die gleiche Quadraturformel verwenden wie zur Berechnung von $a_{T_k}(\varphi_i^k, \varphi_j)$ bzw. $\ell_{T_k}(\varphi_j^k)$.
Dazu müsste man aber die numerische Approximation u_h in den Punkten $\Psi_k(\xi_n, \eta_n)$ auswerten $\to$ zusätzlicher Aufwand.
Einfacher: Verwende stattdessen die Quadraturformel

$$\int_{T_k} F(x,y) \, d(x,y) \approx \frac{|T_k|}{3} \left(F(P_4^k) + F(P_5^k) + F(P_6^k) \right)$$

wobei P_4^k, P_5^k, P_6^k die Seitenmittelpunkte des Dreiecks sind.
Vorteil: An diesen Stellen wurden Werte von u_h berechnet.

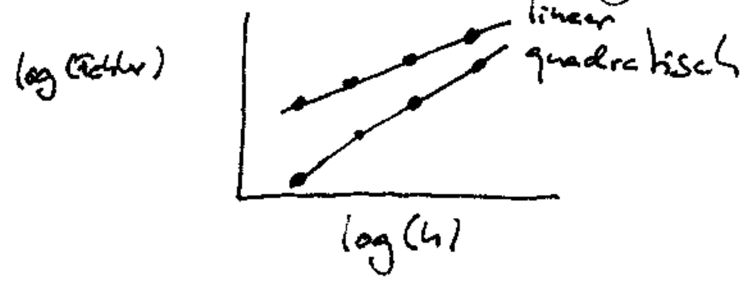
• Visualisierung der Approximation

Bei quadratischen Finiten Elementen ist $u_h|_{T_k}$ ein Polynom zweiter Ordnung, das die Werte in den Knoten interpoliert.
Leider gibt es scheinbar keine Matlab-Funktion für die Visualisierung.
Man kann die Lösung mit

```
trimesh(T2(:,1:3), P2(:,1), P2(:,2), ...)
```

plotten, aber dies liefert nur eine <u>lineare</u> Interpolation durch die Ecken.

• Teste die Konvergenzordnung



Ordnung = Steigung der Geraden

Erstelle zusätzlich eine Tabelle der folgenden Form:

h	Fehler linear	Steigung	Fehler quadratisch	Steigung
⋮	⋮	⋮	⋮	⋮

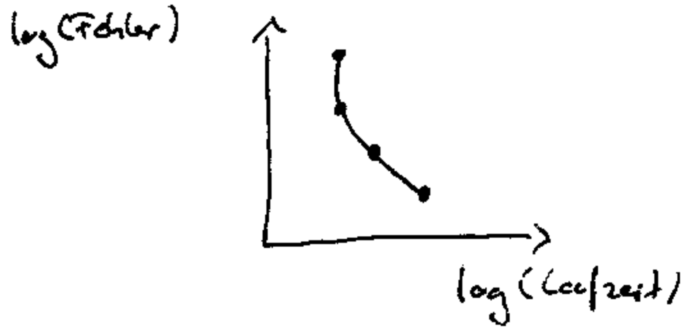
Sei ε_n L^2-Fehler auf dem n-ten Gitter, d.h. mit $h = h_n$.
(längste Kante)

$$\text{Steigung} = \frac{\log(\varepsilon_{n+1}) - \log(\varepsilon_n)}{\log(h_{n+1}) - \log(h_n)} = \text{geschätzte Ordnung}$$

Matlab-Befehl: fprintf

- Work-Precision-Diagramm:

Stoppe die Laufzeit der beiden Verfahren für jede Triangulierung und plotte den L^2-Fehler in Abhängigkeit von der Laufzeit:



Matlab-Befehle: tic, toc

(iii) Fehlerabschätzungen

Annahmen: $\operatorname{span}\{\hat{\varphi}_1, \hat{\varphi}_2, \ldots\}$ enthält alle Polynome vom Grad $\leq k$
(k=1 für lineare Elemente, k=2 für quadratische)
Exakte Lösung $u \in H^{k+1}(\Omega)$

Dann gilt

$$\|u-u_h\|_{H^1(\Omega)} \leq C h^k |u|_{k+1}$$

$$\|u-u_h\|_{L^2(\Omega)} \leq C h^{k+1} |u|_{k+1}$$

mit der Sobolev-Halbnorm

$$|u|_{k+1} = \left(\sum_{i+j=k+1} \|\partial_x^i \partial_y^j u\|^2_{L^2(\Omega)} \right)^{1/2}$$

Details: Knabner & Angermann, Kap. 3.4, Theorem 3.29 und 3.37

Brauche für größeres k immer mehr Basisfunktionen und Knoten: