School of Engineering
Winterthur
Zürcher Hochschule für Angewandte Wissenschaften

PDE Toolbox zur Lösung von PDEs **MNEU**

Im ersten Schritt berechnen wir die Lösung folgendes Problems:

Es sei $\Omega = [0,1] \times [0,0.1] \subset \mathbb{R}^2$. Gesucht ist $u: \ \Omega \to \mathbb{R}$ mit

$$\begin{aligned} u - \Delta u &= (1+\pi^2)\,\sin(\pi\,x) && \text{in } \Omega \\ u &= \sin(\pi\,x) && \text{auf } [0,1]\times\{0\} \cup [0,1]\times\{0.1\} \\ u &= 0 && \text{auf } \{0\}\times[0,0.1] \cup \{1\}\times[0,0.1]\,. \end{aligned}$$

Das Problem ist so gestellt, dass die Lösung der von Aufgabe 7 auf Blatt 7 in der Datei `07_NumDiff.pdf` entspricht. Das Modell ist unabhängig von y und lässt sich demzufolge für jedes feste y wie das d1-Problem stellen.

Step 1

Rufen Sie die PDE Toolbox in der Kommandozeile auf.

```
>> pdetool
```

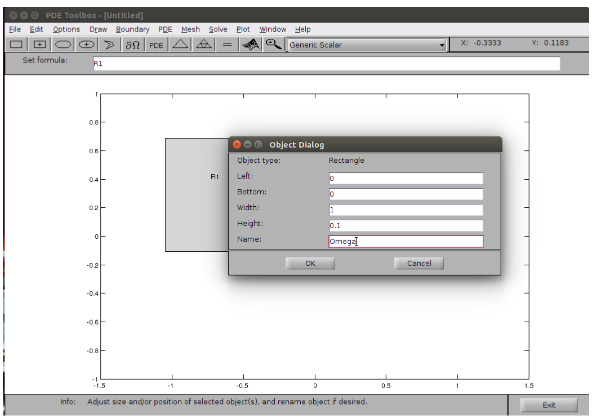
Step 2: In der PDE Toolbox

(a) Gebietsdefinition:

(i) Aktivieren Sie `Options > Grid`. (optional)

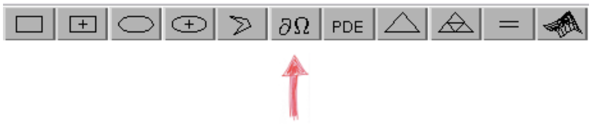
(ii) Setzen Sie ein Viereck mit der Maus.

Doppelklicken Sie in das Viereck und wählen die Koordinaten für linke untere Ecke $(0,0)$ und rechte obere Ecke $(1,0.1)$. Der Name des Gebietes ist Omega (Ω).

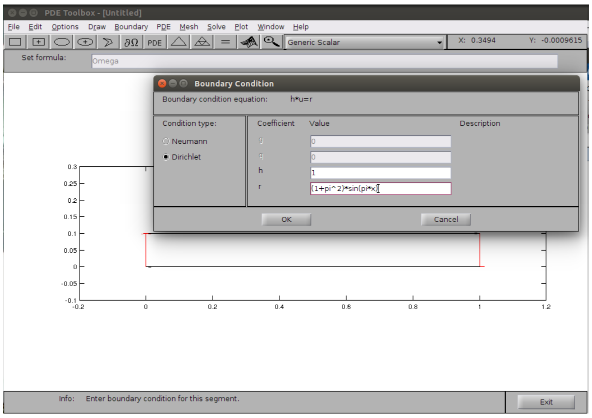


(iii) `Options > Axes Limits: auto` passt die Ansichtsgrsöße von Ω an.

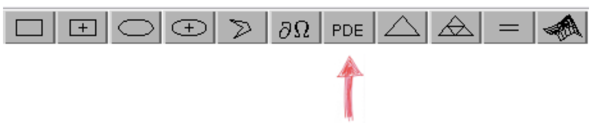
(b) Festlegung der Randwerte:



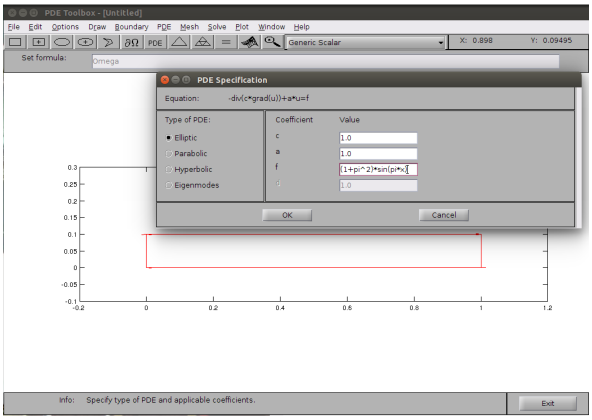
Klicken Sie auf das Symbol $\partial\Omega$. Klicken Sie auf ein Randstück. Es wird nun dargestellt welche Möglichkeiten Sie haben, Randwerte zu setzen. Unser Problem hat am linken ($\{0\} \times [0, 0.1]$) und rechten Rand ($\{1\} \times [1, 0.1]$) Dirichlet-Nullrandwerte. Da das die Standardeinstellung in der Toolbox ist brauchen wir an diesen Stellen nichts weiter zu tun. Am oberen und unteren Rand wählen wir ebenfalls Dirichlet Randwerte aber diesmal nicht Null sondern so, dass die exakte Lösung aus dem 1d-Problem als Lösung überhaupt möglich ist. Markieren Sie beide Ränder ($[0, 1] \times \{0\} \cup [0, 1] \times \{0.1\}$) durch Anklicken und gleichzeitiges Halten der Shift-Taste. Doppelklick auf den markierten Rand öffnet das passende Menuefenster:



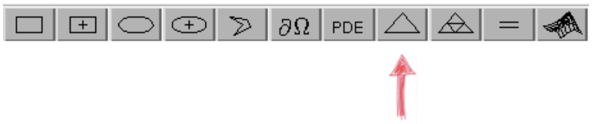
(c) Bestimmung des Modells:



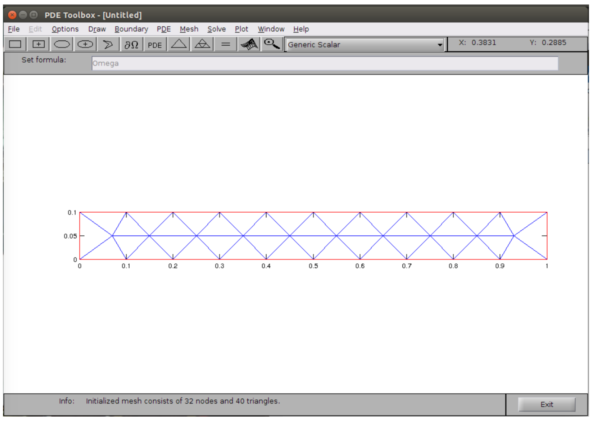
Klicken Sie das Symbol PDE in der Symbolleiste. Im geöffneten Fenster können Sie Ihr Modell zusammenstellen:



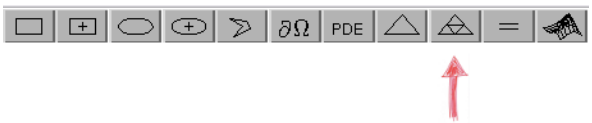
(d) Makrotriangulierung des Gebiets:



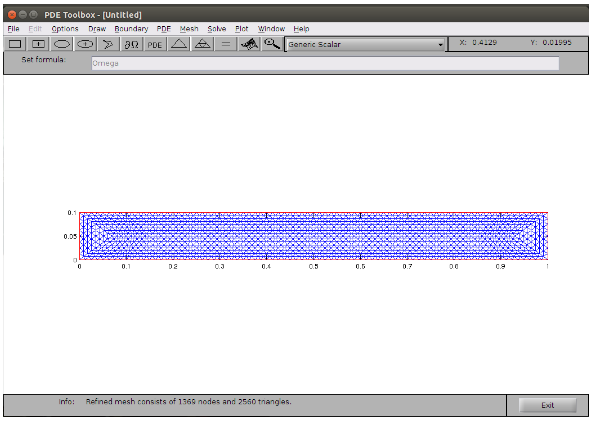
Es sieht dann ungefähr so aus:



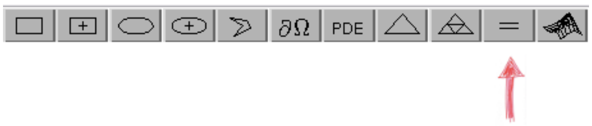
(e) Verfeinern Sie das Gitter:



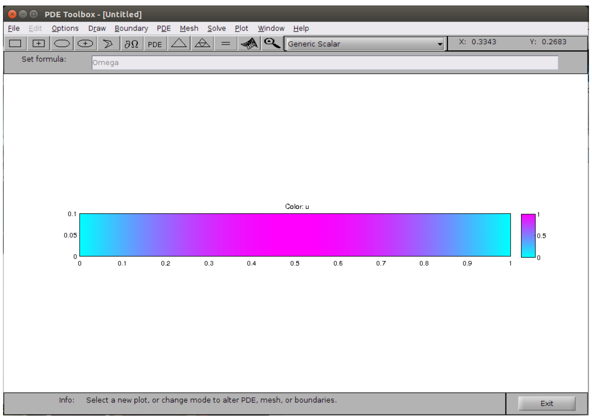
Klicken Sie das Dreieckssymbol. Jeder Klick drittelt alle Dreiecke ein Mal. Nach dreimaligem Klicken erhalten Sie dieses Rechengitter:



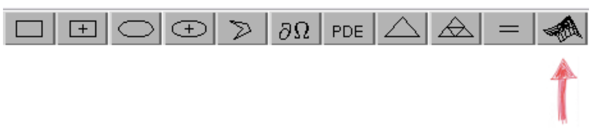
(f) Durch Anklicken des "="-Symbols in der Menueleiste aktivieren wir in der Löser des Modells:



Sie sehen anschließende eine graphische Darstellung der Lösung. Die Interpretation können Sie an der rechtsstehenden Legende ablesen:



(g) Es gibt verschiedene Darstellungmöglichkeiten und somit Möglichkeiten einer qualitativen Beurteilung des Ergebnisses. Klicken Sie dazu das rechte Symbol auf der Menueleiste:



(h) Speichern aller Informationen: `File > Save as ...` Geben Sie den Namen `MyFirstPDE.m` und achten Sie darauf, dass die Datei im root-Verzeichnis `Numerik` gespeichert wird. Sie können jetzt die PDE Toolbox schließen.

(i) Durch Aufruf von

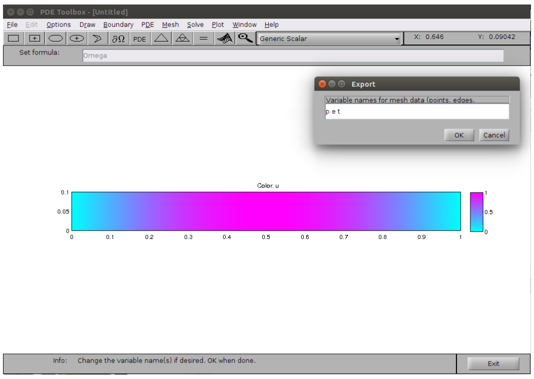
```
>> MyFirstPDE
```

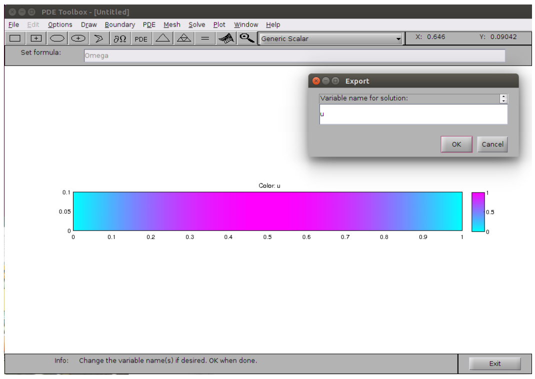
wird alles wieder ausgeführt und dargestellt, wie Sie es in der PDE Toolbox veranlasst hatten.

Obacht: Die Datei `MyFirstPDE.m`, die Matlab erstellt hat, sollten Sie nicht editieren. Es sei denn Sie sind sich absolut sicher, dass Sie wissen was Sie tun. Sie dient im Wesentlichen dazu, alles was Sie getan haben für die Nachwelt oder eben den nächsten Arbeitstag aufzubewahren, damit Sie nicht jedes Mal alles neu eingeben müssen.

Step 3: Weiterverarbeitung der Ergebnisdaten, EOC

(a) `Mesh > Export Mesh ...` und `Solve > Export Solution ...`





speichert alle Gitterinformationen in die Felder `p e t` (point/edge/triangle)

```
>> whos
  Name          Size                  Bytes  Class     Attributes

  e             7x176                  9856  double
  p             2x1369                21904  double
  t             4x2560                81920  double
  u          1369x1                   10952  double
```

(b) Die exportierten Felder beinhalten die Information

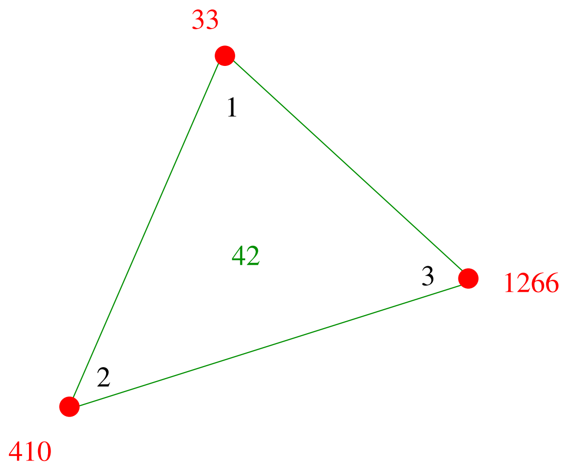
```
>> Nt = length(t);
>> Np = length(p);
```

das heißt es gibt $N_t = 2560$ Dreiecke im Gitter und $N_p = 1369$ Freiheitsgrade, also Dreiecksecken. Jedes Dreieck und jeder Knoten wird mit einer Nummer identifiziert. Jeder Knoten besitzt bezüglich eines Dreiecks auch eine lokale Nummer.

Nehmen wir zum Beispiel das Dreieck mit der Nummer 42:

```
>> t(:,42)
ans =
          33
         410
        1266
```

(Die vierte Komponente ignorieren wir.)



Dreieck 42 hat also die Ecken 33, 410 und 1266 (die lokale Nummerierung verläuft immer gegen den Uhrzeigersinn; darauf kann man sich verlassen). Die Koordinaten der Eckknoten sind im Feld p gepseichert:

```
>> p(:,33)              >> p(:,410)              >> p(:,1266)

ans =                   ans =                    ans =

    0.0500                  0.0375                   0.0462
    0.1000                  0.1000                   0.0938
```

Es wird also das Dreieck mit der Nummer 42 aufgespannt durch die Punkte

$$P_1 = (x_{33}/y_{33}) = (0.05/0.1)\,,$$
$$P_2 = (x_{410}/y_{410}) = (0.0375/0.1)$$

und

$$P_3 = (x_{1266}/y_{1266}) = (0.0462/0.0938)\,.$$

Nun liegt es auf der Hand, dass u in der i-ten Komponente den Wert der Lösung bei der i-ten Koordinate (x_i, y_i) beinhaltet:

$$\texttt{u(i)} = u(x_i, y_i)$$

(c) Jetzt eine kleine Aufgabe:

(i) Erstellen Sie ein Verzeichnis `PDE`.

(ii) Schreiben Sie eine Funktion `MaxGridWidth(p,t)` im Verzeichnis `PDE`, (s. Abbildung 1)

(ii) Schreiben Sie ein Programm `MyPDEPostProcessing`, welches `MaxGridWidth` aufruft und die Gitterweite `h` ausgibt:

```
>> MaxGridWidth(p,t)
ans =
     0.0125
```

(iii) Erstellen Sie ein Feld `uex` mit

$$\texttt{uex(i)} = \sin(\pi\, x_i)\,.$$

Berechnen Sie den Fehler in der ∞-Norm

$$\|u - u_{\text{ex}}\|_\infty = \texttt{norm(u-uex',inf)}\,.$$

Ausgabe (Programmcode s. Abbildung 2):

```
>> norm(u-uex',inf)
ans =
     2.2117e-05
```

(iv) Verfeinern Sie das Gitter 2,3,4,5-Mal und berechnen Sie jeweils die maximale Gitterweite und den zugehörigen Fehler. Speichern Sie die Werte (per Hand) in die Felder `Err` und `Gri`:

```
Err=[6.91e-05 2.21e-05 6.75e-06 1.99e-06];
Gri=[2.50e-02 1.25e-02 6.25e-03 3.13e-03];
```

Anschließend ermitteln Sie die Konvergenzordnung mit der Funktion `EOCh`:

```
EOCh(Err,Gri)
```

Ausgabe:

```
Konvergenzordnung p(1)=1.64e+00
Konvergenzordnung p(2)=1.71e+00
Konvergenzordnung p(3)=1.77e+00
```

Es gilt also für das Programm:

$$\|u - u_{\text{ex}}\|_2 = \mathcal{O}(h^{1.77})\,.$$

```
function hmax=MaxGridWidth(p,t)

N = length(p);

hmax = 0;
for k=1:N
      node = t(1:3,k);
      h(1) = norm(p(:,node(1))-p(:,node(2)));
      h(2) = norm(p(:,node(2))-p(:,node(3)));
      h(3) = norm(p(:,node(1))-p(:,node(3)));
      hm = max(h);
      if hm>hmax hmax=hm; end
end
end
```

Abbildung 1: `PDE/MaxGridWidth.m`

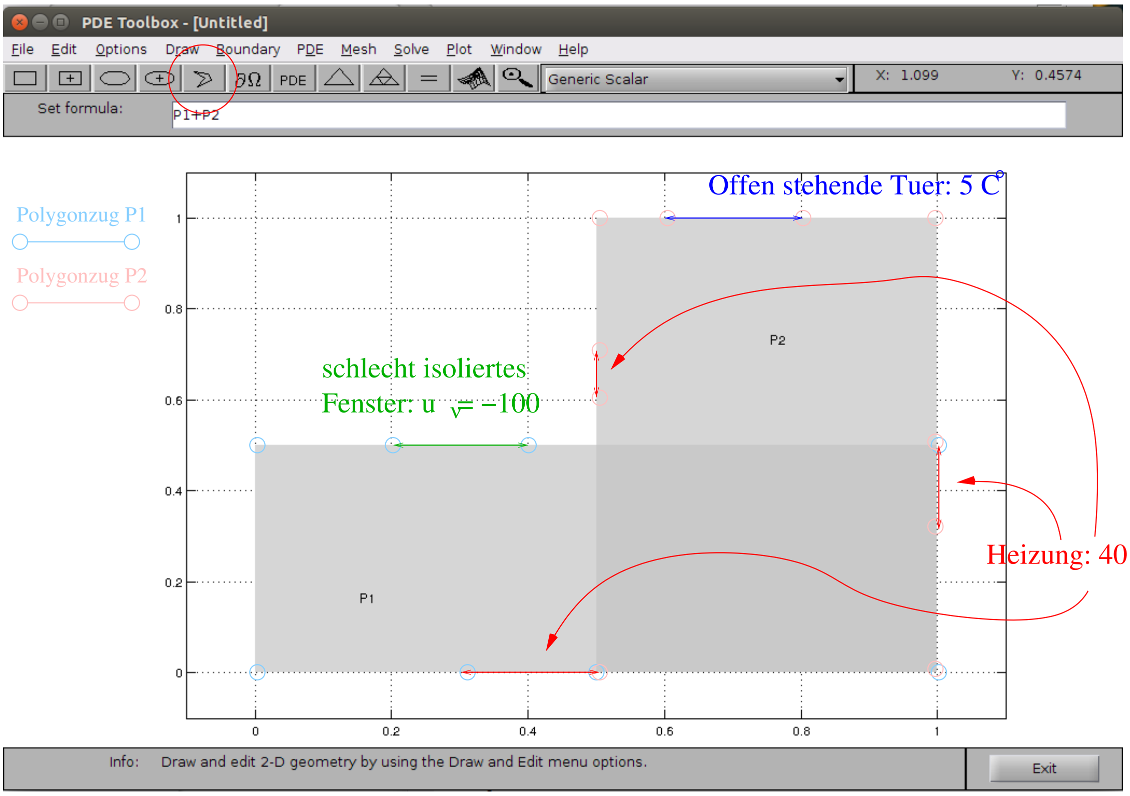
```
close all
% clear all: ACHTUNG: dieser Befehl würde Ihre exportierten
Felder p,t und u wieder löschen !!!

addpath('PDE')

h=MaxGridWidth(p,t);
uex=sin(pi*p(1,:));
Error=norm(u-uex',inf);

fprintf('%.2e %.2e\n',h,Error);
```

Abbildung 2: `MyPDEPostProcessing.m`

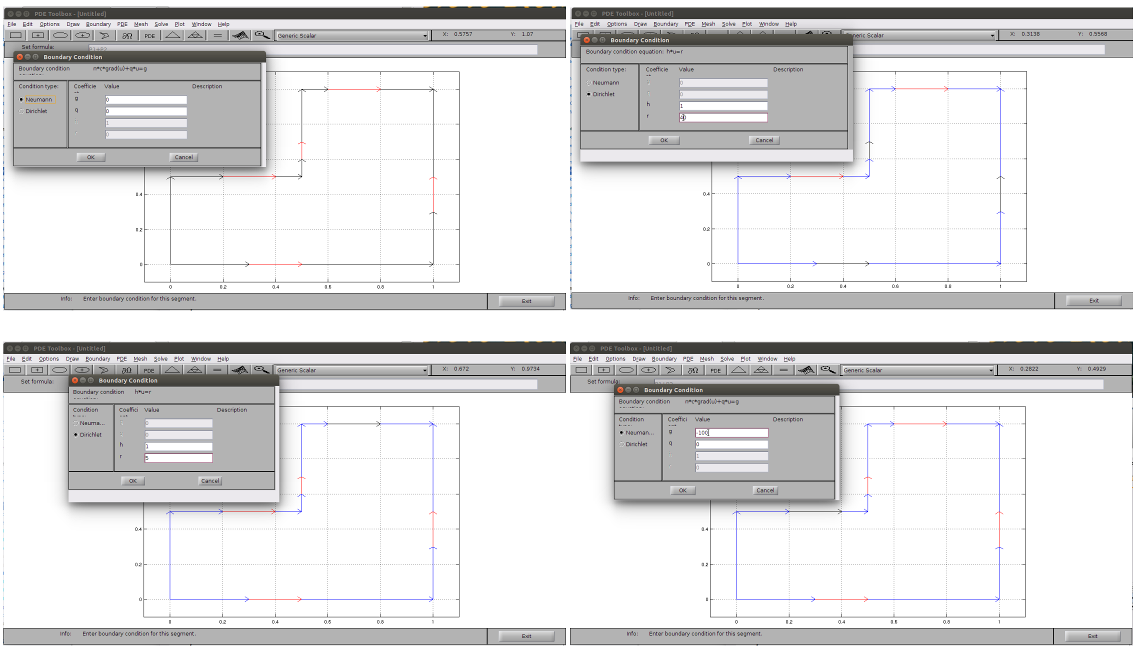
Step 4: Eine Aufgabe

Ihre Firma richtet ein neues Großraumbüro ein, in welchem auch Sie einen neuen Arbeitsplatz zugewiesen bekommen. Per Zufall ist der Architekt, der die Räumlichkeiten renoviert ein Tischtenniskollege Ihres KFZ-Mechanikers, der Ihnen noch etwas schuldig ist. Auf diesem Weg erhalten Sie die Pläne des Raumes.

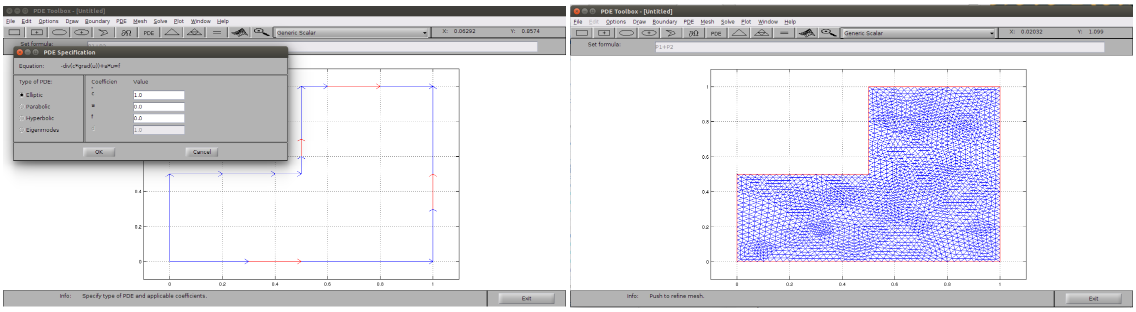
Sie arbeiten am liebsten bei lauschigen 25° C. Unter Zuhilfenahme der Raumpläne des Architekten wollen Sie errechnen welchen Arbeitsplatz Sie bevorzugen, damit Sie möglichst rasch einen Wunsch äußern können, sobald Ihr Chef Sie danach fragt.



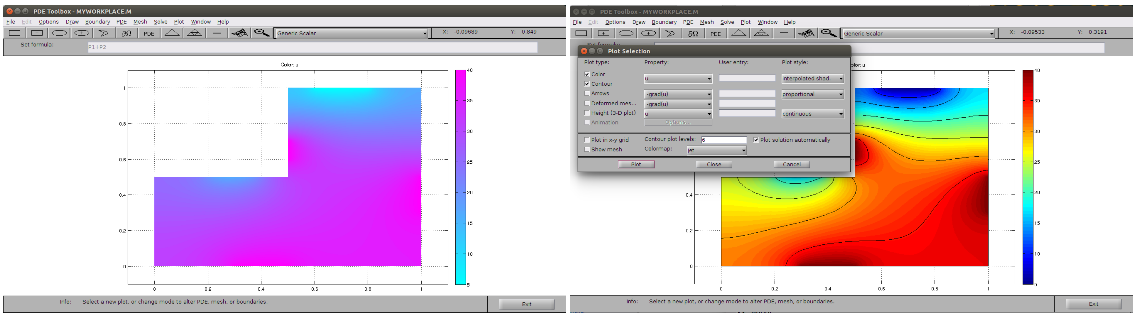
(a) Randwerte setzen:



(b) Modell definieren und Gebiet triangulieren:



(c) Lösung plotten:



(d) Postprocessing:

Daten exportieren und in einem Programm nach die gesuchten Lokalitäten berechnen:

```
close all

% Solve > Export Solution
[umin ii]=min(abs(u-25));

fprintf('Der beste Platz befindet sich bei (%.2f,%.2f) und hat %.2f Grad\n', ...
    p(1,ii),p(2,ii),u(ii));
```

liefert die Ausgabe:

```
Der beste Platz befindet sich bei (0.94,0.72) und hat 25.03 Grad
```

Na! Da lässt es sich sicher vorzüglich arbeiten. ;-)