Matlab Tutorials

Das *LiveScript* zu dem Tutorial 7.

Lösung: Aufgabe aus Tutorial 6

```
U = @(R,I) R*I
R = @(U,I) U/I
I = @(U,R) U/R
```

Tutorial 7 - Anwendung

In dem folgenden Tutorial soll es um eine Datenverarbeitungsaufgabe gehen. Hierfür verwenden wir Wetterdaten vom April 2016, die auf dem Uni Server bereitgestellt werden. Wir gehen die folgen 5 Schritte der Reihe nach durch:

1. Einlesen der Daten
2. Darstellung in Tabelle
3. Analyse der Daten
4. Grafische Darstellung
5. Optimierung des Plots

1. Zunächst werden die Daten vom Uni Server heruntergeladen.

```
clear
close all
clc

data_raw = urlread('http://www.uni-oldenburg.de/dez4/wetter/ausgabe.php?datei=wetter1604.txt')
data_raw = strrep(data_raw, ',', '.');
data = textscan(data_raw, '%s %s %f %f %f %*d %*f %*f %*d %f %*d %*d %*d %*f %*f %*d %*d %*d %
```

2. Anschließend werden die Daten richtig formatiert und in einer Tabelle abgespeichert.

```
% Quelle - Uni Server:
% http://www.uni-oldenburg.de/dez4/wetter/
names = {...
    'Datum', ...
    'Windgeschwindigkeit', ...
    'Temperatur', ...
    'Temperatur_gefuehlt', ...
    'Niederschlag'};

% Datum: Strings zu "datetime" konvertieren
dates = datetime([char(data{1}) char(data{2})], ...
    'InputFormat', 'dd.MM.yyyyHH:mm:ss',...
    'Format', 'dd.MM.yy HH:mm');

% Erstellen einer Tabelle
t = table(dates, data{3}, data{4}, data{5}, data{6}, 'VariableNames', names)
```

```
% Speichern der Tabelle für beliebige andere Programme
writetable(t,['Klima_' datestr(t.Datum(1), 'mmmmyyyy') '.csv']);
```

3. Analyse der Daten mit Hilfe von der Mittelwert- und Summenfunktion.

```
% get mean temp data
idx_noon = find(t.Datum.Hour == 12 & t.Datum.Minute == 0);

% Preallocate für Tagesdaten (Temperatur und Niederschlag)
vTemp_mean = NaN(max(t.Datum.Day), 1);
vRain_sum  = NaN(max(t.Datum.Day), 1);
for iDay = min(t.Datum.Day):max(t.Datum.Day)
    idx = t.Datum.Day == iDay;

    % speicher Mittelwert der Temperatur
    vTemp_mean(iDay) = mean(t.Temperatur(idx));
    % speicher Summer des Niederschlags
    vRain_sum(iDay) = sum(t.Niederschlag(idx));
end
```

4. Nun werden die Daten anschaulich dargestellt.

```
% Plots (linker Axis)
yyaxis('left')
plot(idx_noon, vTemp_mean, 'r', 'LineWidth', 2)
hold('on')
plot(t.Temperatur, 'r--', 'LineWidth', 0.5)
hold('off')
ha              = gca;
ha.YColor       = 'r';
ha.YLabel.String= 'Temperatur in °C';
YTick_left      = ha.YTick;

% Plots (rechte Axis)
yyaxis('right')
plot(idx_noon, vRain_sum, 'b', 'LineWidth', 2)
ha          = gca;
ha.YColor   = 'b';
```

5. Im letzen Schritt werden noch ein paar optische Verbesserungen durchgeführt.

```
legend('Temperatur (Mittelwert)', 'Temperatur', 'Niederschlag')

ha.XTick        = idx_noon;
ha.XTickLabel   = datestr(t.Datum(idx_noon), 'dd.mm.');
ha.XTickLabelRotation = 60;
ha.YGrid        = 'on';
ha.GridColor    = 'k';
ha.Title.String = ['Temperatur Übersicht: ' datestr(t.Datum(1), 'mmmm yyyy')];
ha.XLabel.String= 'Datum';
ha.YLabel.String= 'Niederschlag in mm';
ha.YTick        = YTick_left*2;
ha.XLim         = [0 size(data{1}, 1)];
```

```
ha.YLim         = [YTick_left(1)*2 YTick_left(end)*2];
```

Vielen Erfolg beim Programmieren mit Matlab!