Data Science (Prof. Neff)   Abgabedatum 15.12.2020

Gruppe 6: Benjamin Hamm (2060696), Jan Klotter (2060690),

Anna Kuhn (2051063), Michael Schulze (2061282)

Labor zur Prozesskontrolle

```
clear
```

1) Auswahl des Prozesses

Lesegeschwindigkeit eines USB2.0 Sticks

Gemessen wird mit CrystalDiskMark, 10mal von jedem Gruppenteilnehmer (1x 1GiB Zufallsdaten Read SEQ1M Q8T1 in MB/s)

Die Toleranz sollte eigentlich von der Spezifikation vorgegeben werden, aber man kann nicht die Spezifikation von USB2.0 nehmen, da es theoretische Werte sind und diese abhängig vom verwendeten Rechner und des verbauten USB-Chips sind. Wir haben uns auf eine untere Spezifikation von 16 MB/s festgelegt (unterhalb des langsamsten Wertes unserer Messugen, damit wir die Analyse überhaupt durchführen können). Die obere Spezifikation haben wir auf 100 MB/s gesetzt, da für die Geschwindigkeit nach oben in unserem Anwendungsfall "Lesen eines USB-Sticks" keine Grenzen gesetzt sind.

```
%Spezifikationsgrenzen
specLow = 16
```

```
specLow = 16
```

```
specHigh = 100
```

```
specHigh = 100
```

2) Datensätze auf Normalverteilung prüfen

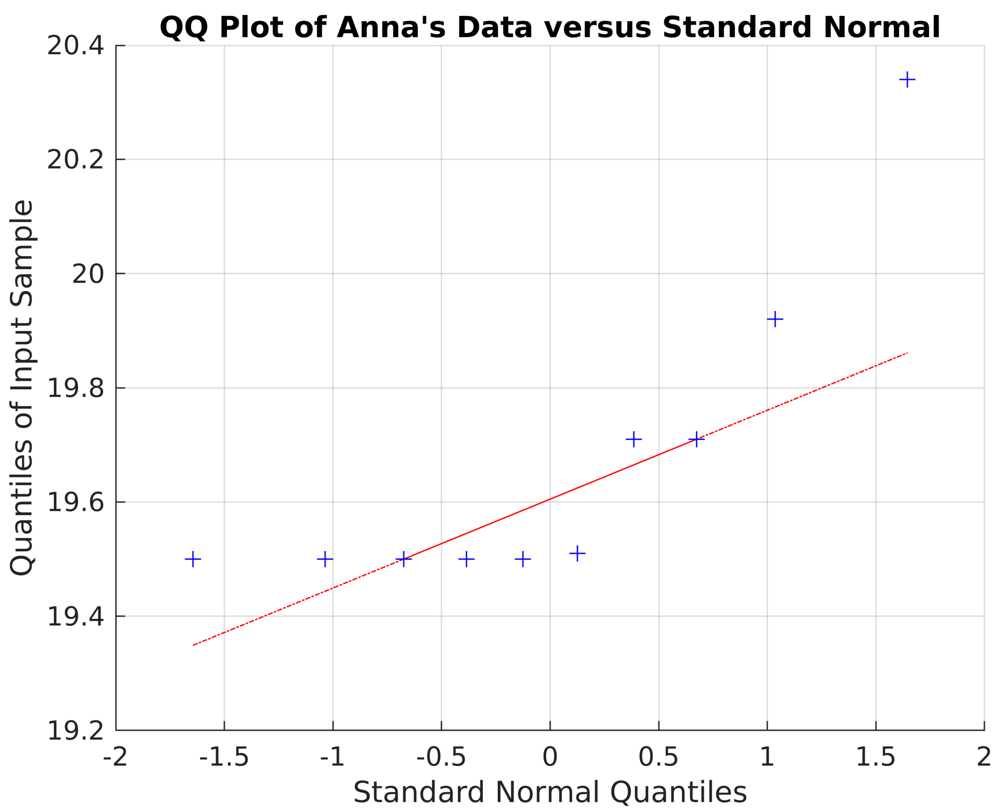
```
%Daten einlesen
data=readtable('USB.xlsx')
```

data = 10×5 table

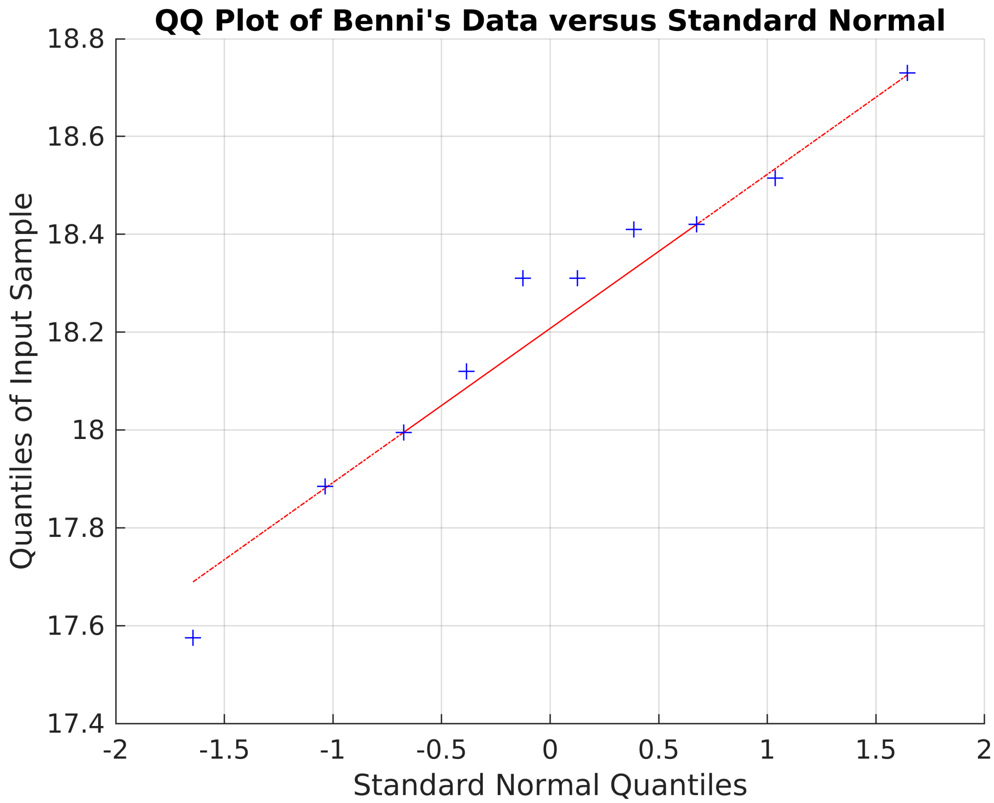
	Var1	Anna	Benni	Michael	Jan
1	1	19.7100	17.9950	17.3870	18.4500
2	2	19.9200	18.4100	16.9600	18.8800
3	3	19.5000	18.7300	17.6390	18.8800
4	4	19.5000	18.4200	16.6480	18.8800
5	5	20.3400	18.3100	18.5410	18.8700
6	6	19.5000	18.3100	18.0240	18.6600

	Var1	Anna	Benni	Michael	Jan
7	7	19.7100	18.5150	18.5530	18.8700
8	8	19.5000	18.1200	17.6750	19.0800
9	9	19.5100	17.8850	18.1690	19.0800
10	10	19.5000	17.5750	16.7470	19.0900

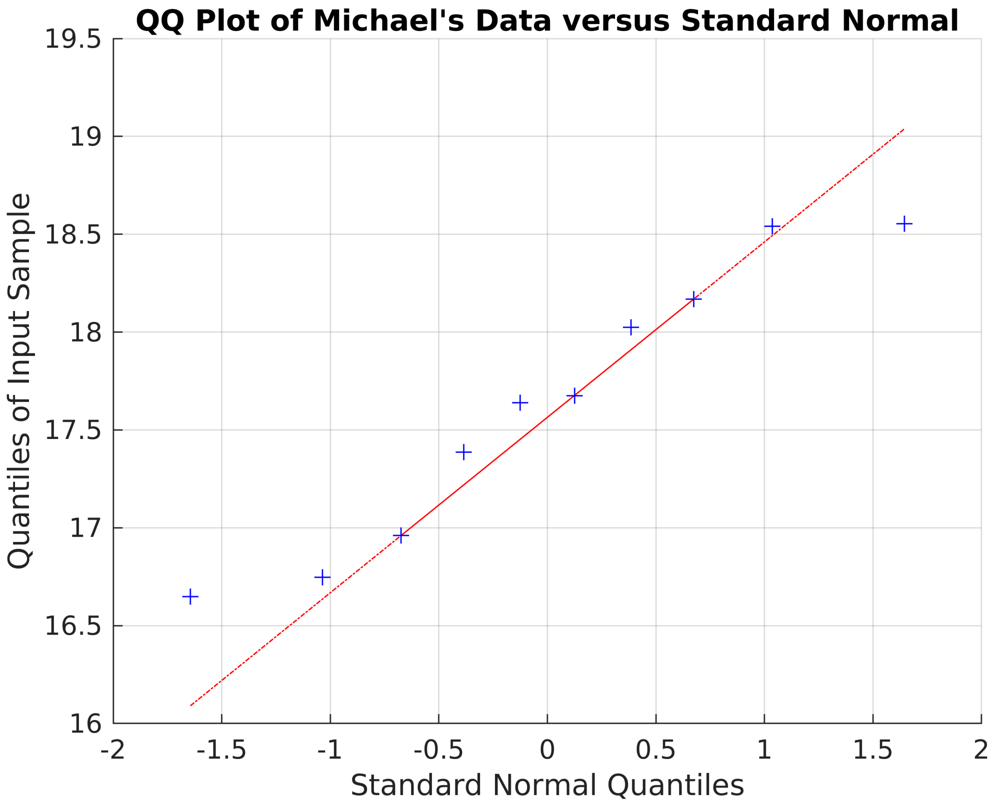
```
%qq-Plot
figure
qqplot(data(:,2).Variables) % Anna
title('QQ Plot of Anna''s Data versus Standard Normal')
grid on
```



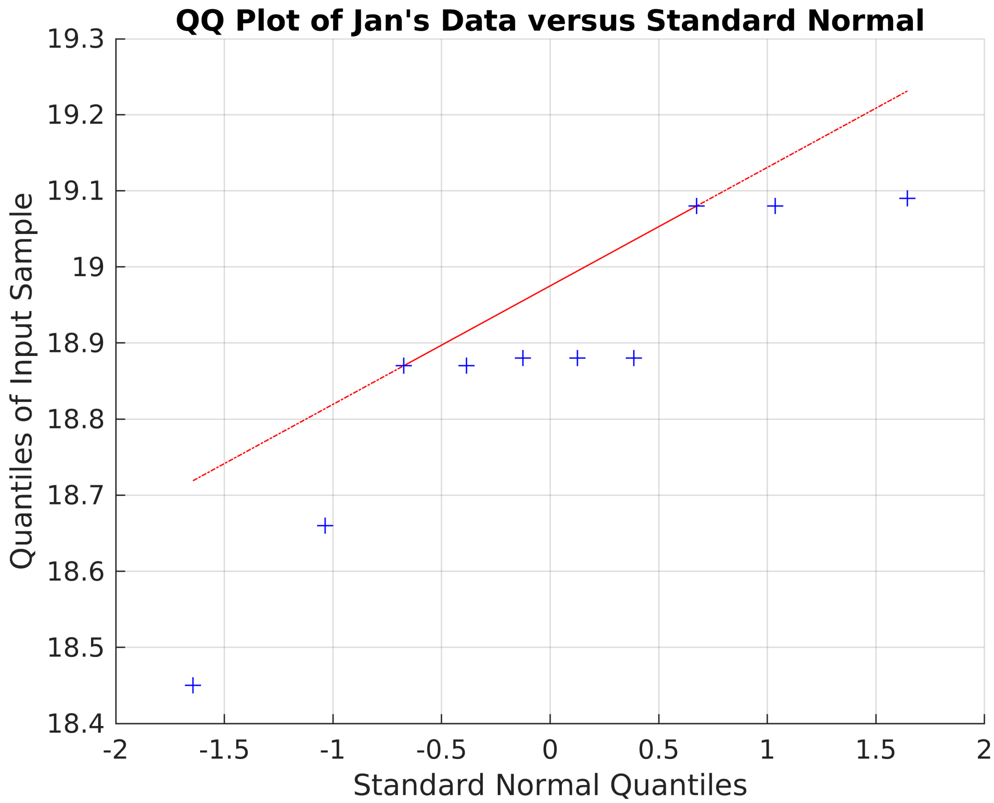

```
figure
qqplot(data(:,3).Variables) % Benni
title('QQ Plot of Benni''s Data versus Standard Normal')
grid on
```




```
figure
qqplot(data(:,4).Variables) % Michael
title('QQ Plot of Michael''s Data versus Standard Normal')
grid on
```




```
figure
qqplot(data(:,5).Variables) % Jan
title('QQ Plot of Jan''s Data versus Standard Normal')
grid on
```




```
% Die qq-Plots lassen erahnen, dass Benni's und Michael's Daten
% normalverteilt sind und die Daten von Anna und Jan nicht.
```

```
% Anderson Darlington Test auf Normalverteiung
% H0 = Die Daten stammen aus einer Normalverteilung
% (h0 = 1 heißt, dass die Daten nicht normalverteilt sind)
[h_anna,p_anna]=adtest(data(:,2).Variables)
```

```
h_anna = logical
   1
p_anna = 9.5840e-04
```

```
[h_benni,p_benni]=adtest(data(:,3).Variables)
```

```
h_benni = logical
   0
p_benni = 0.7575
```

```
[h_michael,p_michael]=adtest(data(:,4).Variables)
```

```
h_michael = logical
   0
p_michael = 0.7106
```

```
[h_jan,p_jan]=adtest(data(:,5).Variables)
```

```
h_jan = logical
```

```
   1
p_jan = 0.0457
```

```
% Bei Anna und Jan sagt der Anderson Darlington Test, dass
% die Daten nicht normalverteilt sind, bei Benni und Michael
% kann eine Normalverteilung angenommen werden.

% Beide Verfahren (qq-Plot und Anderson Darlington Test) haben das
% gleiche Ergebnis, bei 2 Stichproben kann eine Normalverteilung
% angenommen werden und bei 2 Stichproben nicht.
```

3) Prozesskennzahlen und Standardabweichung

```
cp_Anna=capability(data(:,2).Variables,[specLow specHigh])
```

```
cp_Anna = struct with fields:
       mu: 19.6690
    sigma: 0.2758
        P: 1
       Pl: 1.1245e-40
       Pu: 0
       Cp: 50.7577
      Cpl: 4.4341
      Cpu: 97.0814
      Cpk: 4.4341
```

```
cp_Benni=capability(data(:,3).Variables,[specLow specHigh])
```

```
cp_Benni = struct with fields:
       mu: 18.2270
    sigma: 0.3379
        P: 1.0000
       Pl: 2.1992e-11
       Pu: 0
       Cp: 41.4279
      Cpl: 2.1967
      Cpu: 80.6591
      Cpk: 2.1967
```

```
cp_Michael=capability(data(:,4).Variables,[specLow specHigh])
```

```
cp_Michael = struct with fields:
       mu: 17.6343
    sigma: 0.6977
        P: 0.9904
       Pl: 0.0096
       Pu: 0
       Cp: 20.0650
      Cpl: 0.7808
      Cpu: 39.3492
      Cpk: 0.7808
```

```
cp_Jan=capability(data(:,5).Variables,[specLow specHigh])
```

```
cp_Jan = struct with fields:
       mu: 18.8740
    sigma: 0.1992
        P: 1
       Pl: 1.7908e-47
       Pu: 0
```

```
        Cp: 70.2699
       Cpl: 4.8085
       Cpu: 135.7313
       Cpk: 4.8085
```

```
std_Anna = std(data.Anna)
```

```
std_Anna = 0.2758
```

```
std_Benni = std(data.Benni)
```

```
std_Benni = 0.3379
```

```
std_Michael = std(data.Michael)
```

```
std_Michael = 0.6977
```

```
std_Jan = std(data.Jan)
```

```
std_Jan = 0.1992
```

Ist der Prozess für die Anwendung ausreichend fähig?

```
% cpk = 4.4(Anna), 2.1(Benni), 0.7(Michael), 4.8(Jan).
% Alle Prozesse sind beherrscht (da innerhalb von UCL und OCL).
% In Halbleiterindustrie (HLI) muss cpk >= 1,3 sein.
% Anna's, Benni's und Jan's Prozess sind für die HLI fähig, bei Michael ist
% der cpk zu klein.
```

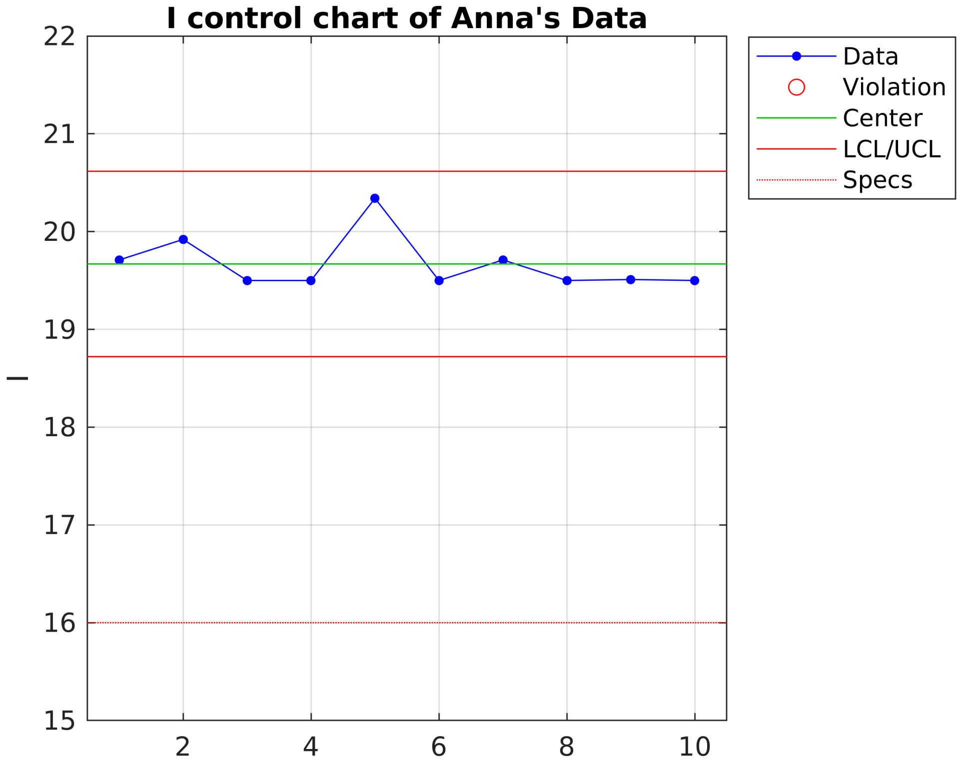
4a) Einzelne Regelkarten

```
%Anna
[stats_i2,plot_i2]=controlchart(data(:,2:2).Variables,data.Var1, ...
    'charttype','i','specs',[specLow specHigh])
```

```
stats_i2 = struct with fields:
        n: [10×1 double]
     mean: [10×1 double]
        i: [10×1 double]
       mr: [10×1 double]
       mu: 19.6690
    sigma: 0.3160
plot_i2 = struct with fields:
    pts: [10×1 double]
     cl: [10×1 double]
    lcl: [10×1 double]
    ucl: [10×1 double]
     se: [10×1 double]
      n: [10×1 double]
    ooc: [10×1 logical]
```

```
ylim([15,22])
title('I control chart of Anna''s Data')
```

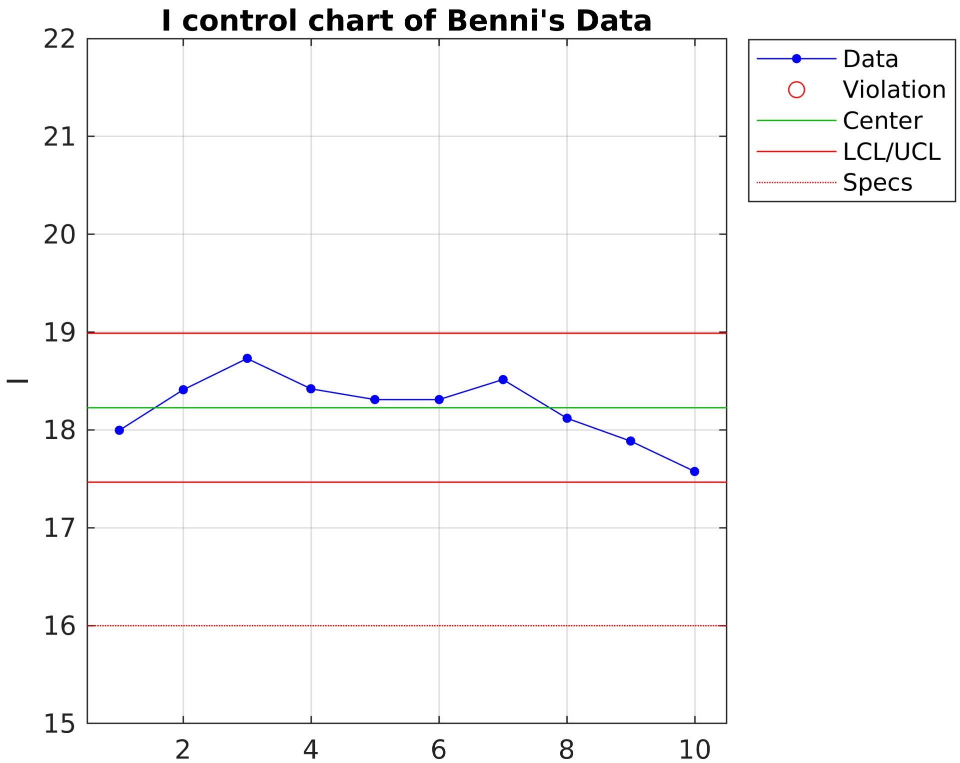
```
grid on
```




```
%Benni
[stats_i2,plot_i2]=controlchart(data(:,3:3).Variables,data.Var1, ...
    'charttype','i','specs',[specLow specHigh])
```

```
stats_i2 = struct with fields:
        n: [10×1 double]
     mean: [10×1 double]
        i: [10×1 double]
       mr: [10×1 double]
       mu: 18.2270
    sigma: 0.2537
plot_i2 = struct with fields:
    pts: [10×1 double]
     cl: [10×1 double]
    lcl: [10×1 double]
    ucl: [10×1 double]
     se: [10×1 double]
      n: [10×1 double]
    ooc: [10×1 logical]
```

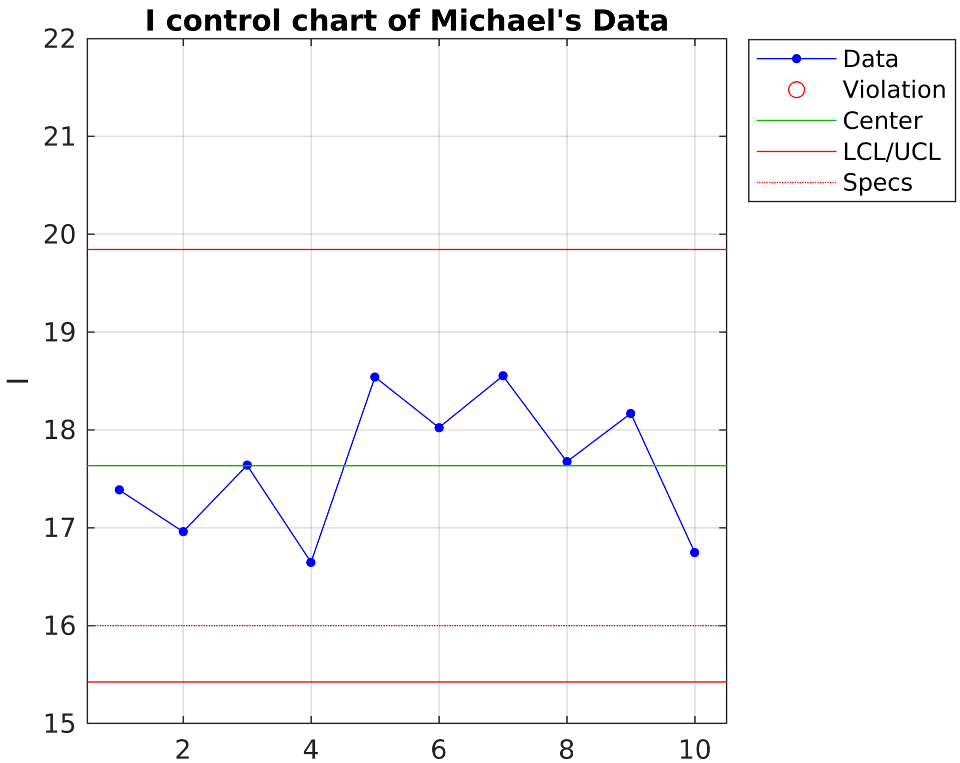
```
ylim([15,22])
title('I control chart of Benni''s Data')
grid on
```




```
%Michael
[stats_i2,plot_i2]=controlchart(data(:,4:4).Variables,data.Var1, ...
    'charttype','i','specs',[specLow specHigh])
```

```
stats_i2 = struct with fields:
        n: [10×1 double]
     mean: [10×1 double]
        i: [10×1 double]
       mr: [10×1 double]
       mu: 17.6343
    sigma: 0.7366
plot_i2 = struct with fields:
    pts: [10×1 double]
     cl: [10×1 double]
    lcl: [10×1 double]
    ucl: [10×1 double]
     se: [10×1 double]
      n: [10×1 double]
    ooc: [10×1 logical]
```

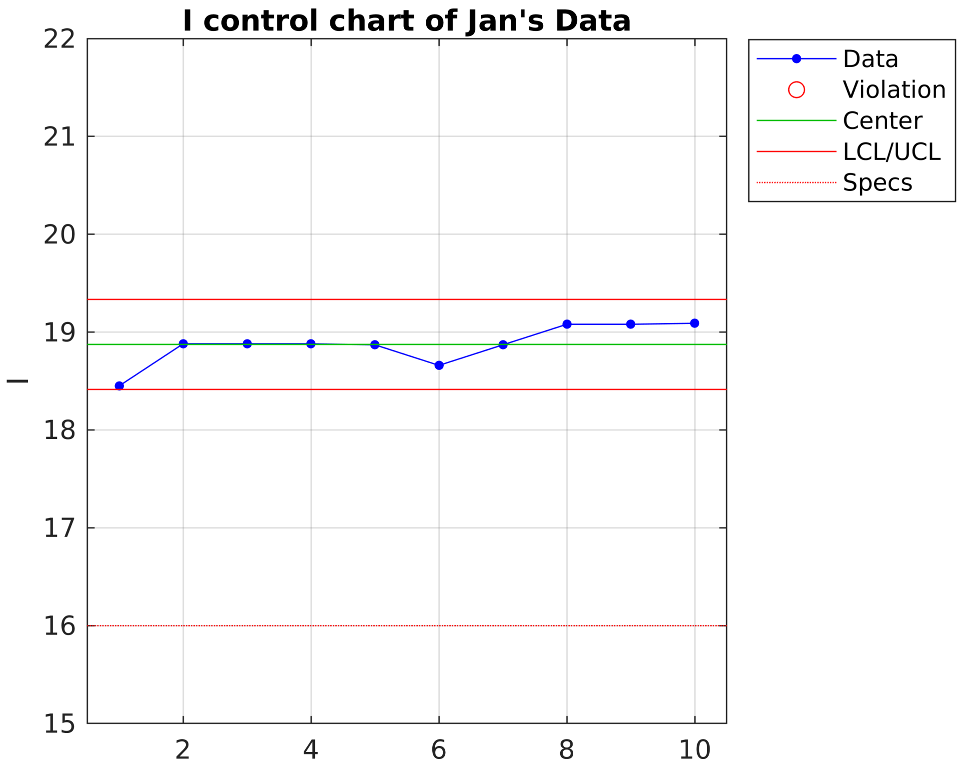
```
ylim([15,22])
title('I control chart of Michael''s Data')
grid on
```




```
%Jan
[stats_i2,plot_i2]=controlchart(data(:,5:5).Variables,data.Var1, ...
    'charttype','i','specs',[specLow specHigh])
```

```
stats_i2 = struct with fields:
        n: [10×1 double]
     mean: [10×1 double]
        i: [10×1 double]
       mr: [10×1 double]
       mu: 18.8740
    sigma: 0.1533
plot_i2 = struct with fields:
    pts: [10×1 double]
     cl: [10×1 double]
    lcl: [10×1 double]
    ucl: [10×1 double]
     se: [10×1 double]
      n: [10×1 double]
    ooc: [10×1 logical]
```

```
ylim([15,22])
title('I control chart of Jan''s Data')
grid on
```



4b) XBAR-Chart

Bieten Regelkarten zusätzliche Informationen? Z. B. hat ein Datensatz eine größere Streuung?

Man kann beispielsweise erkennen welcher Messprozess größere Streuung besitzen, als andere.

Halten Sie die Regelkarte für diese Prozess für nützlich?

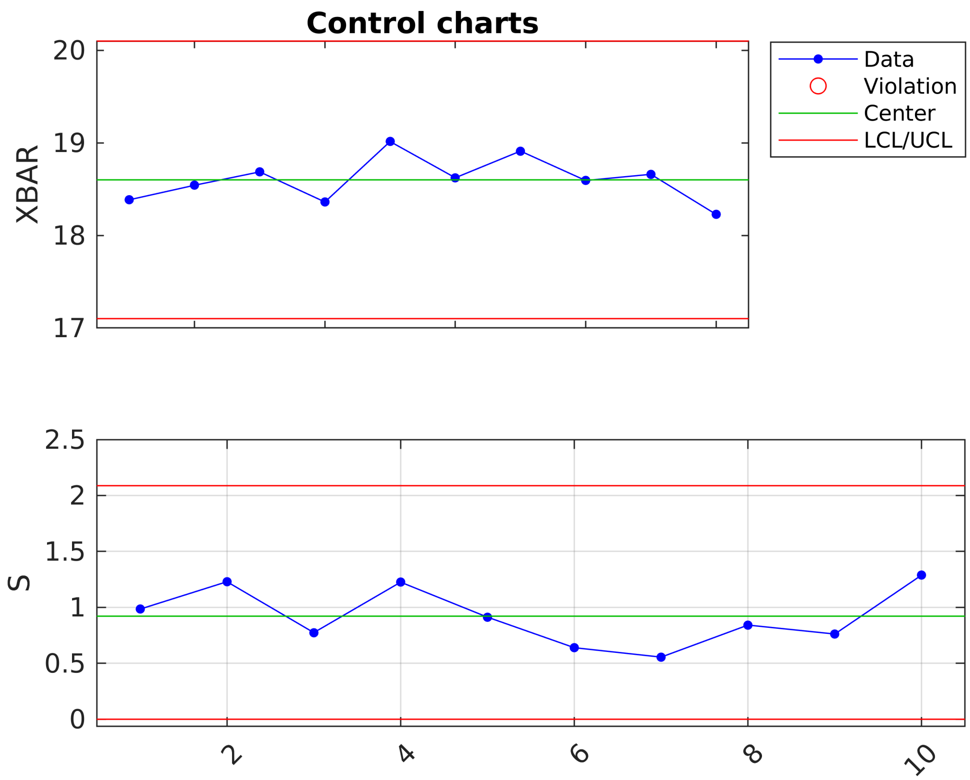
Für diesen Prozess halten wir Regelkarte nicht sinnvoll, da jeder seinen eigenen Prozess mit seinen eigenen Umweltparametern hatte.

```
%XBAR-Chart mit s und R
[stats_xs,plot_xs]=controlchart(data(:,2:5).Variables,data.Var1, ...
    'charttype',{'xbar' 's'})
```

```
stats_xs = struct with fields:
     mean: [10×1 double]
      std: [10×1 double]
        n: [10×1 double]
    range: [10×1 double]
       mu: 18.6011
    sigma: 1.0000
plot_xs = 1×2 struct
```

Fields	pts	cl	lcl	ucl	se	n	ooc
1	[18.3855...	[18.6011...	[17.1011...	[20.1010...	[0.5000;...	[4;4;4;4...	10×1 logical
2	[0.9845;...	[0.9213;...	[0;0;0;0...	[2.0877;...	[0.3888;...	[4;4;4;4...	10×1 logical

```
% aus XBAR s kann man erkennen, zu welchem Zeitpunkt die größte Streuung war
grid on
axes=gca;
axes.XTickLabelRotation=45;
```

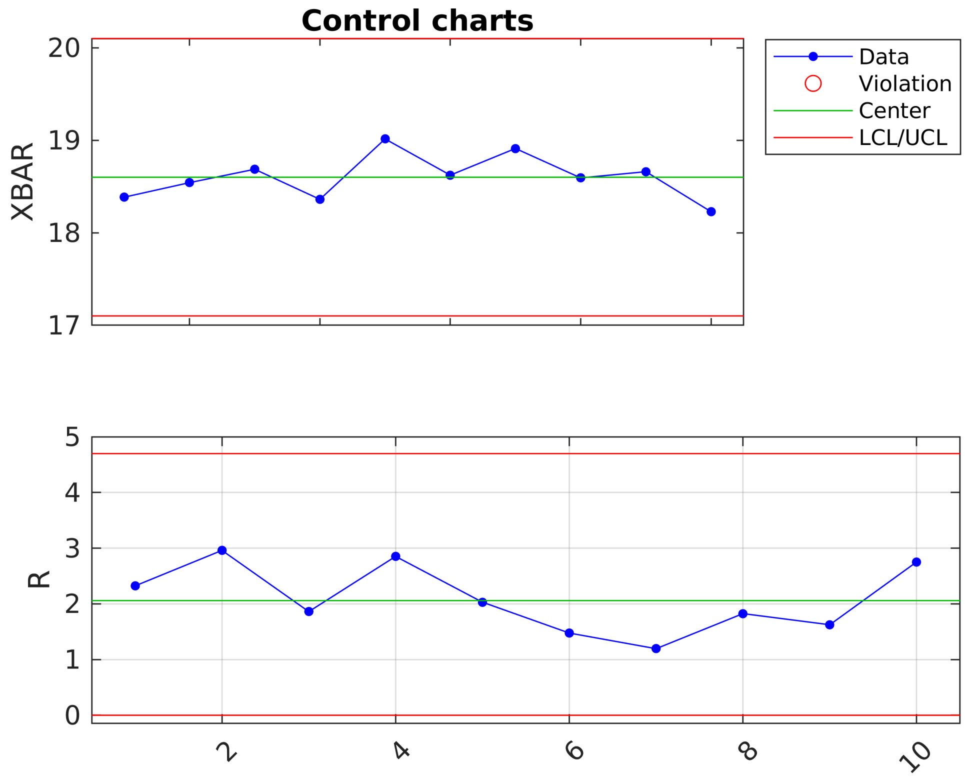



```
[stats_xr,plot_xr]=controlchart(data(:,2:5).Variables,data.Var1, ...
    'charttype',{'xbar' 'r'})
```

```
stats_xr = struct with fields:
     mean: [10×1 double]
      std: [10×1 double]
        n: [10×1 double]
    range: [10×1 double]
       mu: 18.6011
    sigma: 1.0000
plot_xr = 1×2 struct
```

Fields	pts	cl	lcl	ucl	se	n	ooc
1	[18.3855...	[18.6011...	[17.1011...	[20.1010...	[0.5000;...	[4;4;4;4...	10×1 logical
2	[2.3230;...	[2.0587;...	[0;0;0;0...	[4.6981;...	[0.8798;...	[4;4;4;4...	10×1 logical

```
grid on
axes=gca;
axes.XTickLabelRotation=45;
```



5) Datensätze vergleichen

```
%Darstellung Histogramme und qq-Plots zu Mean und sigma bzw. Varianz
sigma=std(data(:,2:5).Variables')
```

```
sigma = 1×10
    0.9845    1.2293    0.7742    1.2254    0.9126    0.6396    0.5553    0.8419 ⋯
```

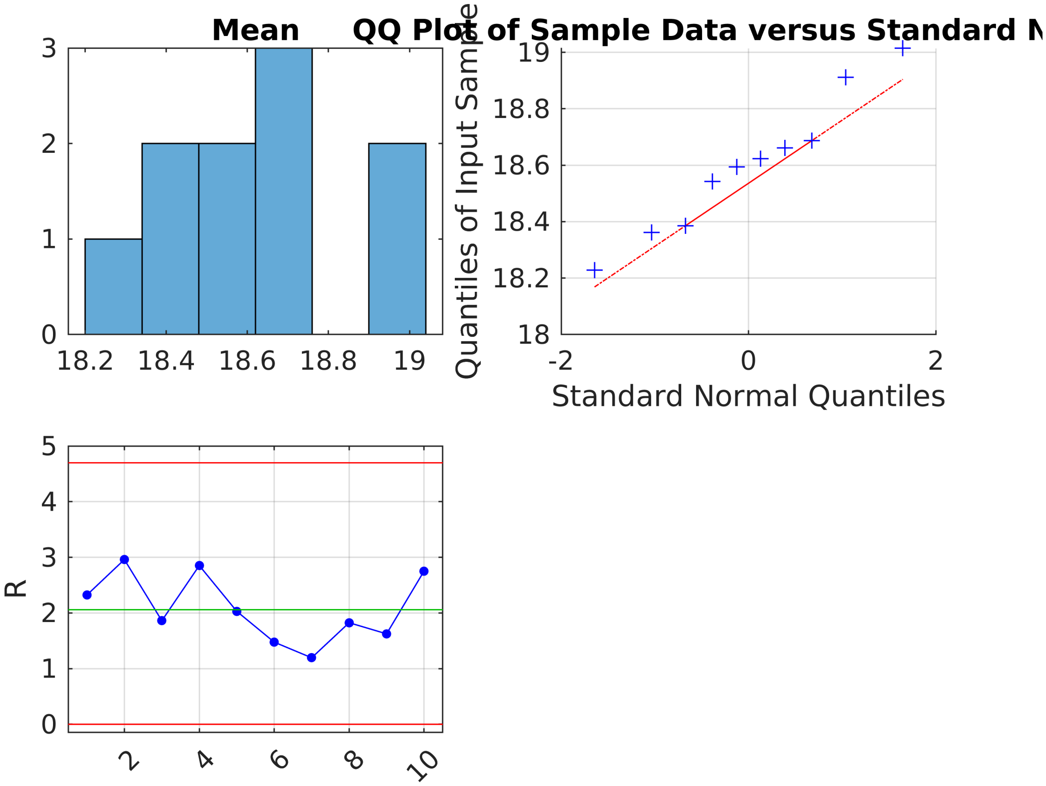
```
variance=var(data(:,2:5).Variables')
```

```
variance = 1×10
    0.9693    1.5111    0.5995    1.5015    0.8328    0.4091    0.3084    0.7088 ⋯
```

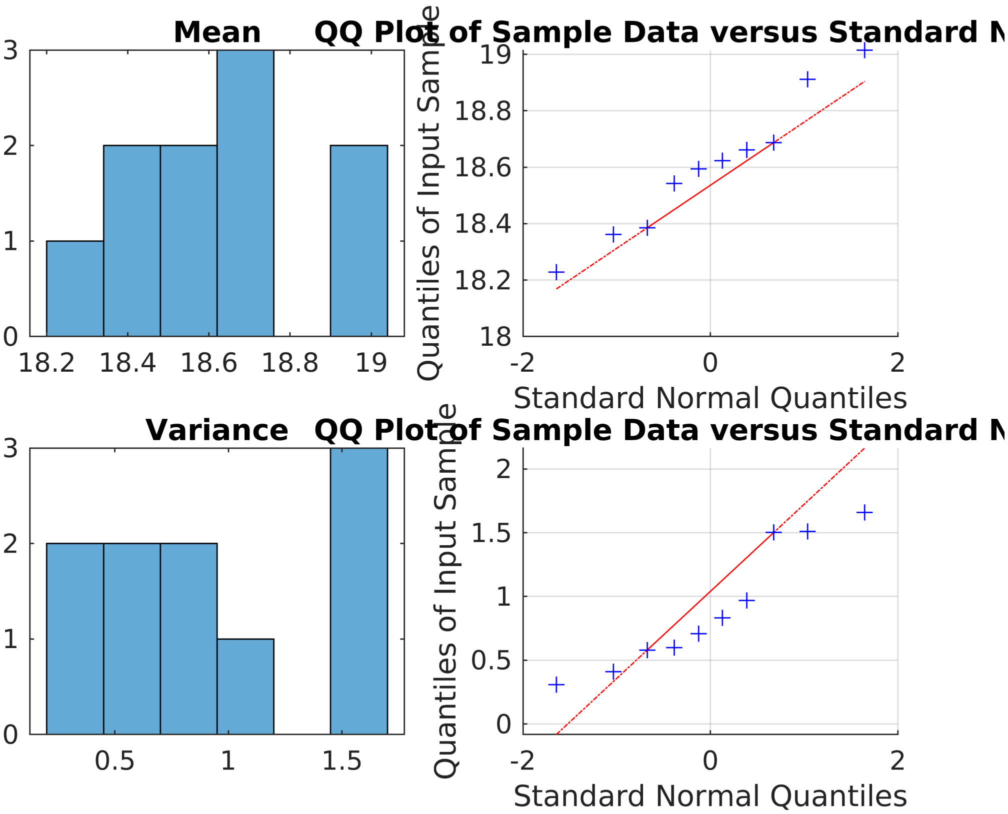
```
mean_daten=mean(data(:,2:5).Variables')
```

```
mean_daten = 1×10
   18.3855   18.5425   18.6872   18.3620   19.0153   18.6235   18.9120   18.5938 ⋯
```

```
subplot(2,2,1)
histogram(mean_daten,round(2*sqrt(10),0))
title("Mean")
subplot(2,2,2)
qqplot(mean_daten)
grid on
```




```
subplot(2,2,3)
histogram(variance,round(2*sqrt(10),0))
title("Variance")
%histogram(sqrt(variance),round(2*sqrt(21),0))
%title("sigma")
subplot(2,2,4)
qqplot(variance)
grid on
```




```
[h_mean,p_mean]=adtest(mean_daten)
```

```
h_mean = logical
   0
p_mean = 0.8092
```

```
[h_var,p_var]=adtest(variance)
```

```
h_var = logical
   0
p_var = 0.2044
```

```
% Anderson Darlington Test sagt, dass H0 angenommen wird und somit Daten
% normalverteilt sind
```

```
%ANOVA

% In Punkt 2 haben wir bereits ermittelt, dass nur für 2 der 4 Stichproben
% eine Normalverteilung angenommen werden kann. Wir werden im folgenden trotzdem
% die anova1-Funktion verwenden,obwohl einige Stichproben nicht normalverteilt sind,
% da es in Matlab zu aufwändig ist, Varianzanalysen mit anderen Verteilungen
% durchzuführen.

%ANOVA nur machen, wenn Normalverteilt
[p,tbl,stats] = anova1(data(:,2:5).Variables,{'Anna', 'Benni', 'Michael', 'Jan'})
```

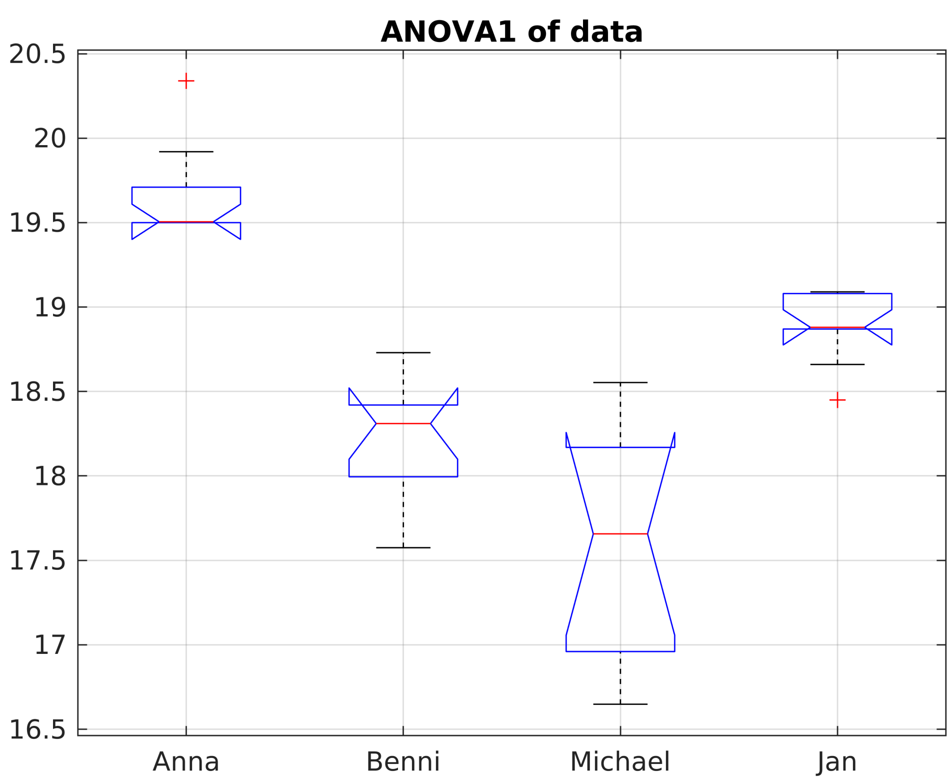
```
                         ANOVA Table
Source        SS        df       MS          F          Prob>F
-----------------------------------------------------------------
Columns     22.8954      3    7.63179    42.59    6.24855e-12
Error        6.4512     36    0.1792
Total       29.3466     39
```

```
p = 6.2486e-12
tbl = 4×6 cell
```

	1	2	3	4	5	6
1	'Source'	'SS'	'df'	'MS'	'F'	'Prob>F'
2	'Columns'	22.8954	3	7.6318	42.5879	6.2486e-12
3	'Error'	6.4512	36	0.1792	[]	[]
4	'Total'	29.3466	39	[]	[]	[]

```
stats = struct with fields:
    gnames: {4×1 cell}
         n: [10 10 10 10]
    source: 'anova1'
     means: [19.6690 18.2270 17.6343 18.8740]
        df: 36
         s: 0.4233
```

```
grid on
title("ANOVA1 of data")
```




```
%Mittelwerte sind nicht gleich zwischen den Messreihen
```

```
%Streuung bei Michael ist besonders groß
```