Folgen, Mengen, Listen und ihre Graphen

Prof. Dr. Dörte Haftendorn: Mathematik mit MuPAD 4, 3.4.99, Jan 03 Update Jan 08
http://haftendorn.uni-lueneburg.de www.mathematik-verstehen.de

++

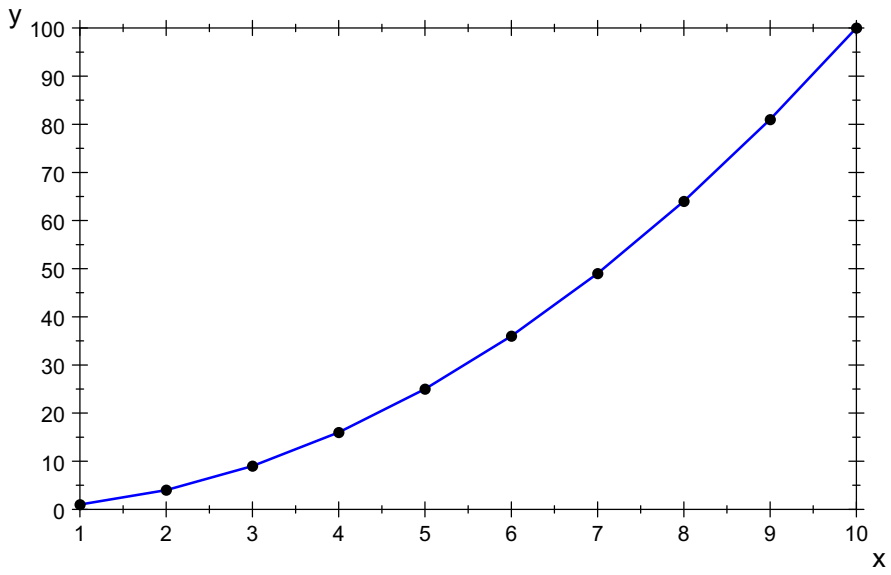
########## 1. Erzeugen und Zeichnen von Folgen #######

Folge der Quadratzahlen

```
quadr:=i^2 $ i=1..10
```

1, 4, 9, 16, 25, 36, 49, 64, 81, 100

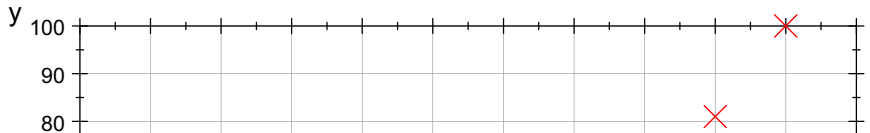
```
plot(plot::Listplot([quadr]));
```

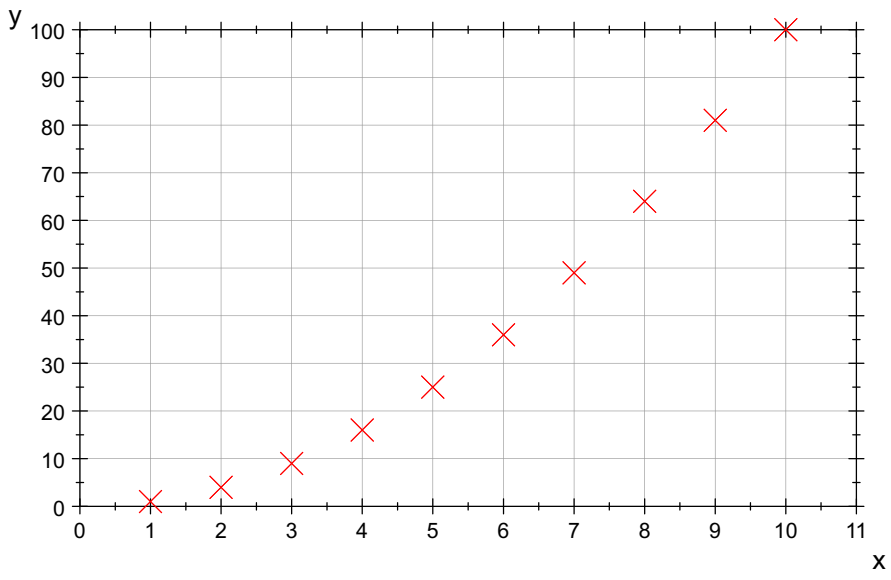


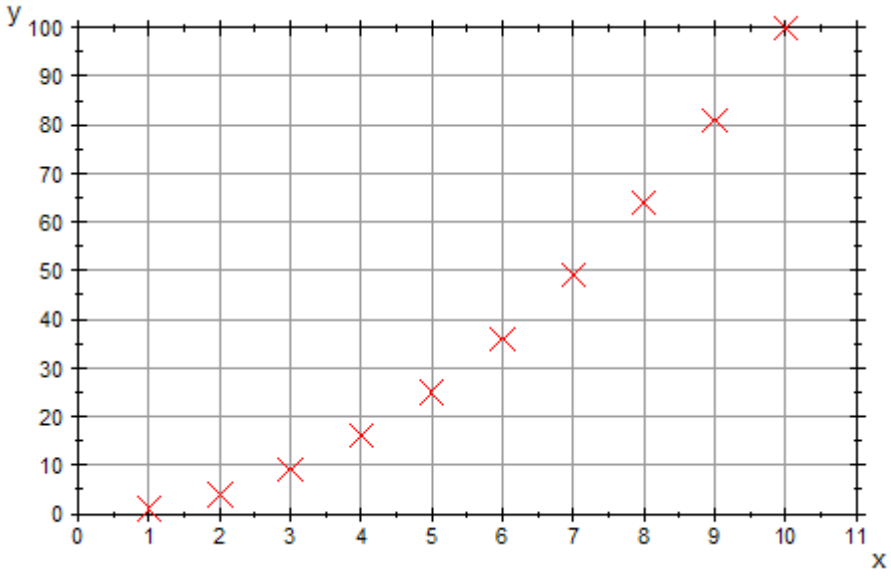
Die [...]-Klammern machen aus der Folge eine Liste

Hier können noch reichhaltig Optionen verändert werden.
Siehe hierzu Graphen-Lernen Level 1 und 2. Erst interaktiv mit Doppelklick
auf die Graphik, dann Option ausprobieren, dann hier eintragen.

```
plot(plot::Listplot([quadr],
   LinesVisible=FALSE, PointSize=3,PointStyle=XCrosses,
   PointColor=RGB::Red),GridVisible=TRUE,
 ViewingBox=[0..11,0..100]);
```

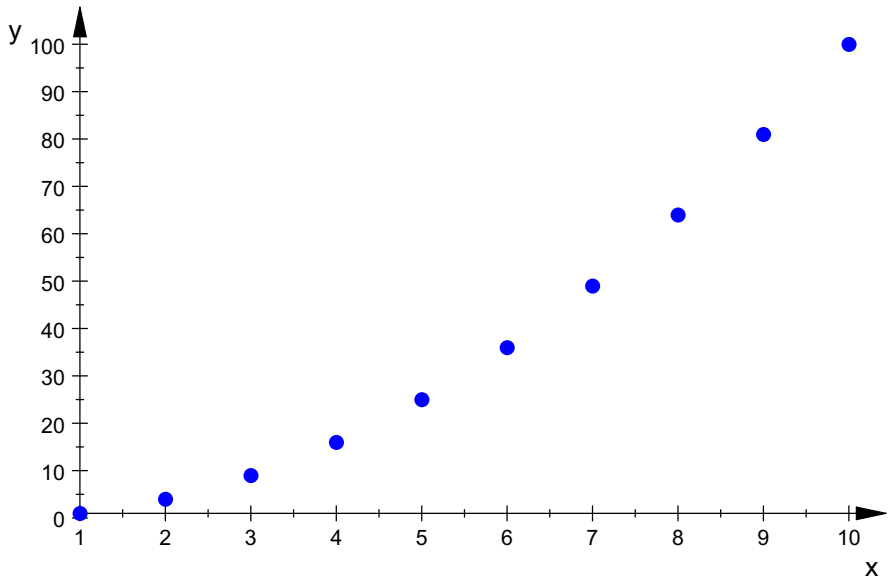







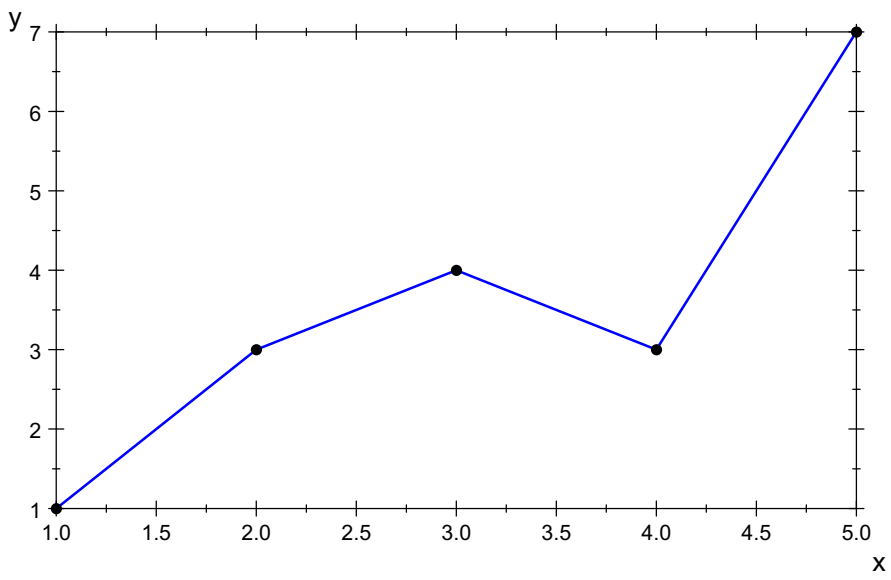
Alternativ kann man sich vom nächsten Befehl Berechnung und Zeichnung gemeinsam erledigen lassen.

```
plot(plot::Sequence(i^2,i=1..10))
```

Hierfür muss einfach der Befehl die die Erzeugung der Folge wiederholt werden.
Schulisch ist diese Version wegen der vorgewählten günstigen Optionen angenehm.
Allerdings wird der Name einer Folge oder auch eine ausgedachte Folge wie 1,3,4,3,7 nicht akzeptiert,
Das kann nur Listplot

```
plot(plot::Listplot([1,3,4,3,7]))
```



2. Folgen, Listen, Mengen, Matrizen... und ihr Verbinden

Folgen haben nur Kommata zwischen den Folgenelementen

```
nummern:= (i $i=1..6);
```

$$1, 2, 3, 4, 5, 6$$

Listen haben Eckige Kammern [...] um Folgen herum.

```
[nummern]
```

$$[1, 2, 3, 4, 5, 6]$$

Mengen haben geschweifte Klammern

```
menge:={1,3,2,3,3,1}
```

$$\{1, 2, 3\}$$

Doppelte Elemente kann es in Mengen nicht geben, daher werden weniger Elemente angegeben. Die Reihenfolge ist nicht relevant.

Dagegen bleibt bei Listen alles erhalten.

```
myliste:=[1,3,2,3,3,1]
```

$$[1, 3, 2, 3, 3, 1]$$

```
ma:=matrix(myliste)
```

$$\begin{pmatrix} 1 \\ 3 \\ 2 \\ 3 \\ 3 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 3 \\ 2 \\ 3 \\ 3 \\ 1 \end{pmatrix}$$

```
folge1:= 1,3,4,3,7;
folge2:= 2,2,6,8,5;
```

$$1, 3, 4, 3, 7$$

$$2, 2, 6, 8, 5$$

Verbinden von zwei Folgen

```
folge1,folge2
```

$$1, 3, 4, 3, 7, 2, 2, 6, 8, 5$$

Verbinden von zwei Listen

```
liste1:=[folge1]:
liste2:=[folge2]:
```

Mit Komma wird es also eine Folge von Listen erzeugt, ganz logisch!

```
liste1,liste2;
```

$$[1, 3, 4, 3, 7], [2, 2, 6, 8, 5]$$

Mit einem Punkt dazwischen kann man Listen verschmelzen.

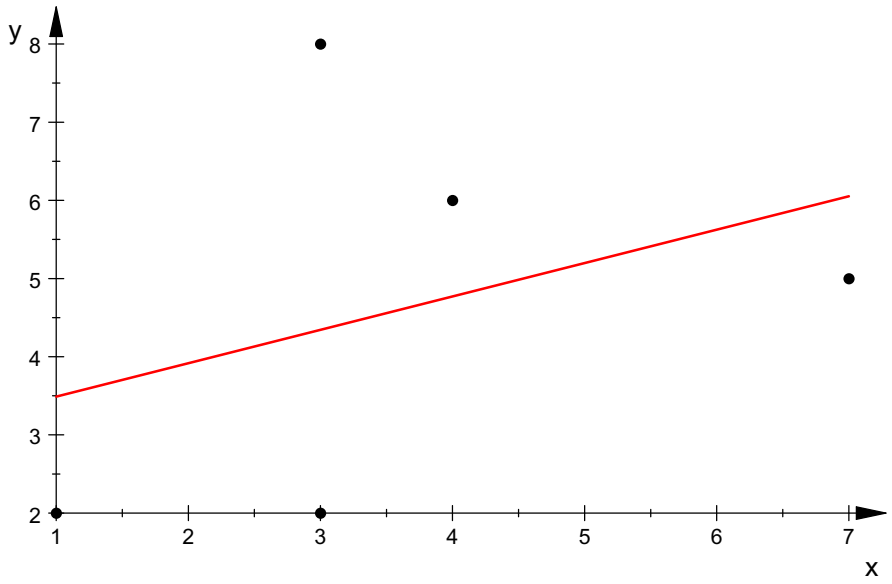
```
liste1.liste2
```

$$[1, 3, 4, 3, 7, 2, 2, 6, 8, 5]$$

3. Erzeugen und Zeichnen von (Daten-)Punkten ####################

```
plot(plot::Scatterplot([liste1,liste2]))
```

```
plot(plot::Scatterplot([liste1,liste2]))
```



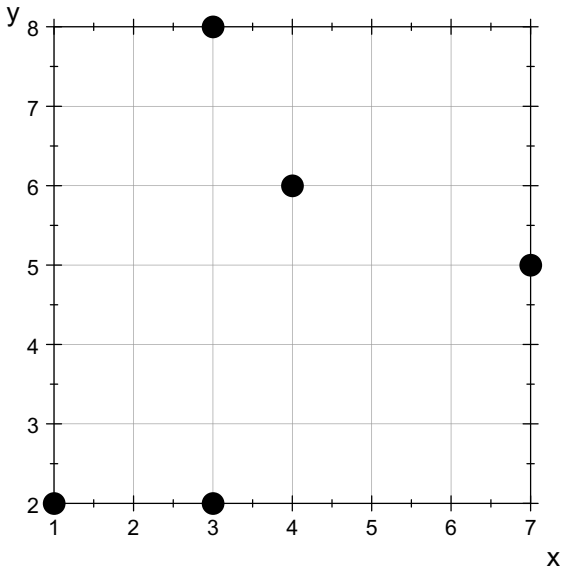
Hier wird gleich die Regressionsgerade eingefügt. Das ist für die Schule nicht so schöne.

Fasst man die erste Liste als x-Werte, die zweite als y-Werte auf, so kann man die Punkte so bekommen:

```
datenPunkte:=stats::unzipCol([liste1,liste2]);
```

$$[1, 2], [3, 2], [4, 6], [3, 8], [7, 5]$$

```
graphDatenPunkte:=plot::Listplot([datenPunkte],
   LinesVisible=FALSE, PointSize=3,Scaling=Constrained,
    GridVisible=TRUE):
plot(graphDatenPunkte)
```



4. Datenpunkte aus Folgen und Funktionen

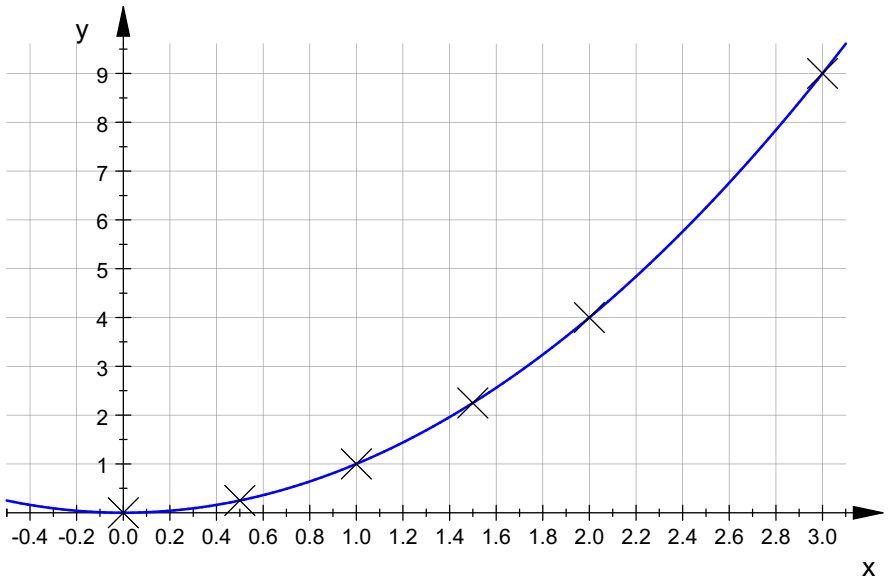
Hier wird der Term x^2 an vorgegebenen Stellen ausgewertet, wie es in Schulbüchern vorkommt.

```
pkte:=[x,x^2] $ x in [0,1/2,1,1.5,2,3]
```

```
pkte:=[x,x^2] $ x in [0,1/2,1,1.5,2,3]
```

$$[0, 0], \left[\frac{1}{2}, \frac{1}{4}\right], [1, 1], [1.5, 2.25], [2, 4], [3, 9]$$

```
graphPkte:=plot::Listplot([pkte],
LinesVisible=FALSE,PointStyle=XCrosses, PointSize=4):
graphFkt:=plot::Function2d(x^2,x=-0.5..3.1, Axes=Origin):
plot(graphFkt,graphPkte,GridVisible=TRUE)
```



5. Anfügen und Herausgreifen

Durch eckige Klammern und Nummern greift man auf Elemente zu.
Daber verändert man die Folge nicht.
Genauso geht es für Listen.

```
folge1;
liste1;
folge1[3];
liste1[3];
folge1[2..4];
folge1;
```

$$1, 3, 4, 3, 7$$

$$[1, 3, 4, 3, 7]$$

$$4$$

$$4$$

$$3, 4, 3$$

$$1, 3, 4, 3, 7$$

Will man aus eine Liste wieder eine Folge machen, so eignet sich

```
op(liste1)
```

```
op(liste1)
```

```
 1, 3, 4, 3, 7
```

op ist eine wichtige sehr allgemeine Funktion,
mit der man an die Operanden eines Ausdrucks herankommt.

```
op(a+b,0)
```

```
 _plus
```

Anzahl der Elemente eine Folge oder Liste

```
nops(folge1);
nops(liste1)
```

```
 5

 5
```

Alles über Listen erfährt man in der "Hilfe" mit dem **Fernglas-Button.**
Tragen Sie Listen bei "Suchen", nicht bei "Komandos" ein