# Anwendung des Mikrocomputers im Mathematik-Unterricht

Helmut Brunner, Mödling

Für die Kollegen aus der AHS sei vorausgeschickt, daß alle Handelsakademien in Österreich in den Schuljahren 1979/80 und 1980/81 mit Mikrocomputern für den Einsatz in den Gegenständen Datenverarbeitung, Rechnungswesen und Textverarbeitung ausgestattet wurden. Die Computer sind mit Bildschirm und Floppy-Disk-Laufwerken versehen und derzeit sind je fünf Computer an einen gemeinsamen Drucker angeschlossen.

Natürlich bedeutet eine derartige Ausstattung auch eine Herausforderung zum Einsatz im Mathematik-Unterricht. Für den Mathematiklehrerselbst bietet der Computer die Möglichkeit, einen Vorrat von Übungs- und Schularbeitsbeispielen aus allen Gebieten der Schulmathematik ausarbeiten zu lassen. Bei der Anwendung im Unterricht selbst sind zwei Aspekte zu beachten: einerseits die Erlernung des algorithmischen Denkens bei der Ausarbeitung von Programmen und andererseits die experimentelle Gestaltung von Teilen des Mathematikunterrichtes (z.B. Wahrscheinlichkeitsrechnung).

Es soll nun an Hand von typischen Beispielen gezeigt werden, wie eine Realisierung im Unterricht möglich ist. Vorher aber noch einige Bemerkungen für jene Kollegen, die selbst noch nicht (in BASIC) programmiert haben:

1. Jeder Befehl (Anweisung) beginnt mit einer Zeilennummer.

2. Eine Gleichung X = X+H ist so zu lesen:

   Neuer Wert von X = Alter Wert von X plus Wert von H oder:

   H = H/10 : Neuer Wert von H = Alter Wert von H dividiert durch 10.

3. Es sei folgende Kette von Befehlen gegeben:

```
30      IF 10* H> E THEN 100
40      X = X-H
50      ........
.       ........
.       ........
100     X = X+H
```

   Zeile 30 bedeutet: Wenn der Wert von 10H größer ist als der Wert von E, dann wird das Programm bei Zeile 100 fortgesetzt. (Bedingter Sprung). Wenn dagegen 10H = E oder 10H< E, dann wird bei Zeile 40 fortgesetzt.

4. INPUT A, B, H

   bedeutet Eingabe der Zahlenwerte von A, B, H in dieser Reihenfolge.

5. PRINT A, B, H

   bedeutet Ausgabe (Bildschirm) des letzten Wertes von A, B, H in dieser Reihenfolge.

6. DEF FNF (X) = X* EXP (-X)

   Überall, wo im Programm FNF(X) vorkommt, ist dies der jeweilige Wert, der auf der rechten Seite stehenden Funktion, hier also der jeweilige Wert von $f(x) = x \cdot e^{-x}$

## 7. "FOR-Schleifen"

**Beispiel:**

```
.  ................
.  ................
.  ................
40  X = A
50  FOR J = 1  TO  N
60  X = X+H
70  PRINT X, FNF(X)
80  NEXT J
.  ................
.  ................
```

J durchläuft nacheinander die Werte 1, 2, 3, ... bis N und dabei wird jedesmal X um einen Betrag H erhöht, dieser Wert X und der zugehörige Funktionswert FNF(X) nebeneinander ausgedrückt (Tabelle) solange, bis J den Wert N erreicht hat. Dann wird bei Zeile 90 im Programm fortgesetzt.

Nach dieser Vorbereitung nun zu den Beispielen.

1. $f(x)$ sei in $[a,b]$ stetig und $f(a) < 0$, $f(b) > 0$.
   Dann existiert mindestens eine Nullstelle in $[a,b]$.

   Wir starten in x = a; von hier gehen wir mit der Schrittlänge h solange nach rechts, bis erstmals $f(x) > 0$; dann gehen wir einen Schritt zurück, ersetzen h durch h/10 und wiederholen den Vorgang solange, bis die gesuchte Nullstelle in einem Intervall liegt, dessen Länge kleiner als die vorgegebene Fehlerschranke E ist.

Das entsprechende BASIC Programm lautet:

```
10 DEF FNF(X) = X* X-3* X-1   (Wir betrachten eine Nullstelle
                               von f(x) = x^2 - 3x - 1)
20 INPUT A, H, E              (Intervallanfang, Schrittweite,
                               Fehlerschranke)
30 X = A
40 X = X+H
50 IF FNF(X)> 0 THEN 70
60 GOTO 40
70 X = X-H
80 H = H/10
90 IF 10* H> E THEN 40
100 PRINT "Nullstelle ="; X
110 END
```

Wenn wir mit dem Befehl RUN das Programm starten und für A den Wert 2, für H den Wert 0.1 und für E den Wert 0.001 eingeben, dann erhält man als Ausgabe auf dem Bildschirm: Nullstelle = 3.302. Da $f(x) = x^2 - 3x - 1$ an der Stelle a = 2 negativ ist: f(2) = -3, dagegen an der Stelle b = 4 positiv: f(4) = 3 haben wir die dazwischen liegende Nullstelle mit einem Fehler von weniger als 0.001 berechnet.

Falls f(a)> 0 und f(b)< 0, braucht man in Zeile 10 an Stelle von f(x) nur -f(x) einzugeben.

Um einen Überblick zu erhalten, wo innerhalb eines gegebenen Intervalls [a,b] Nullstellen von f(x) liegen könnten, ist es zweckmäßig, sich eine Funktionstabelle ausdrucken zu lassen, indem man [a,b] in n gleiche Teile teilt. Dies leistet das Programm:

```
10 DEF FNF(X) = X* X - 3* X - 1
20 INPUT A, B, N
30 H = (B - A)/N
40 X = A - H
50 FOR J = 0 TO N
60 X = X + H
70 PRINT X, FNF(X)
80 NEXT J
90 END
```

Setzt man a = -1, b = 4, N = 10, so wird in Zeile 30 : h = 0.5 berechnet und innerhalb der FOR-Schleife (Zeile 50 bis Zeile 80) erfolgt ein Ausdruck folgender Tabelle:

| | |
|---|---|
| -1 | 3 |
| -0.5 | 0.75 |
| 0 | -1 |
| 0.5 | -2.25 |
| 1 | -3 |
| 1.5 | -3.25 |
| 2 | -3 |
| 2.5 | -2.25 |
| 3 | -1 |
| 3.5 | 0.75 |
| 4 | 3! |

Aus der Tabelle sieht man sofort, daß eine Nullstelle im Intervall [-0.5/0] und eine weitere im Intervall [3/3.5] liegt.

2. f(x) stetig in [a,b] und nicht monoton fallend. Berechnet soll das erste, rechts von a liegende Maximum von f(x) werden. Wir starten in x = a und gehen mit der Schrittweite h solange nach rechts bis erstmals $f(x+h) \leq f(x)$ gilt. Dann gehen wir einen Schritt zurück, ersetzen h durch h/10 und wiederholen den Vorgang solange, bis die Maximumstelle in einem Intervall liegt, das kleiner als die gegebene Fehlerschranke ε ist:

```
10 DEF FNF(X) = X* EXP(-X)
20 INPUT A, B, H, E
30 X = A
40 IF FNF(X+H)<= FNF(X) THEN 100
50 X = X+H
60 IF X< B THEN 40
70 PRINT B, FNF(B) ... in diesem Fall war f(x) in [a,b] monoton
                       steigend
80 GOTO 20

100 H = H/10
110 IF 10* H = E THEN 40
120 PRINT X, FNF(X)
130 END
```

Gibt man hier a = 0, b = 2, h = 0.1, ε = 0.001 ein, so erhält man
x = 0.999 f(x) = 0.367879

3. <u>Vor.</u> $n \in \mathbb{N}$, $a \in \mathbb{R}^+$

Es soll $\sqrt[n]{a}$ mit Hilfe einer Intervallschachtelung und einem Maximalfehler ε berechnet werden.

Es gilt:

$$a^{\frac{1}{n}} = x \Leftrightarrow x^n = a$$

Wir gehen von einem Näherungswert $x_o^n < a$ aus, gehen dann mit der Schrittweite h solange nach rechts, bis erstmals $x^n > a$; dann gehen wir einen Schritt zurück, ersetzen h durch h/10 und wiederholen den Vorgang solange, bis ein Intervall entsteht, das kleiner als e ist:

```
10 INPUT A, N, X0, E
20 X = X0
30 X = X + H
40 IF X^ N> A THEN 60
50 GOTO 30
60 X = X - H
70 H = H/10
80 IF 10* H> E THEN 30
90 PRINT "Wurzel = ": X
100 END
```

Hier kann man auch experimentell den Grenzwert der Folge $\langle a^{\frac{1}{n}} \rangle$ für $n \to \infty$ ermitteln, indem man für n der Reihe nach eine steigende Folge natürlicher Zahlen eingibt. Man sieht sehr deutlich, wie mit zunehmendem n die Wurzel (und zwar fallend) gegen 1 strebt. Wenn zu Beginn davon die Rede war, daß der Computer auch zur Ausführung "mathematischer Experimente" herangezogen werden kann, so soll dies hier durch ein Beispiel belegt werden:

Simulation eines Würfelspiels

Durch den Befehl RANDOMIZE wird ein Zufallszahlengenerator eingeschaltet, der beim Befehl RND eine reelle Zufallszahl aus dem Intervall (0,1) erzeugt. Will man den Wurf eines Würfels simulieren,

so geschieht dies durch den Befehl INT (6* RND) + 1 durch den eine Zufallszahl aus der Menge {1,2,...,6} erzeugt wird. Der Zufallszahlengenerator ist so ausgelegt, daß er gleichverteilte Zufallszahlen erzeugt, was man übrigens wieder mit einem Programm überprüfen kann, in welches man die Chi-Quadrat-Verteilung einbaut.

Das folgende Programm soll nun Würfe mit 2 Würfeln simulieren und 2 fiktive Spieler A,B einführen, wobei A gewinnt, wenn die Augensumme $Z \leq 7$, während B sonst gewinnen soll. Das Spiel ist nicht gerecht, da die Wahrscheinlichkeit für einen Gewinn von A gleich 21/36 beträgt, während sie für einen Gewinn von B nur 15/36 ist. Genau dies soll aber auch bei Spielen mit dem Computer entdeckt werden. Wir bezeichnen mit A die Anzahl der Gewinne von A, analog mit B die Anzahl der Gewinne von B und denken uns N Spiele durchgeführt. Die theoretische Häufigkeit für A wäre dann 21.N/36, für B 15N/36. Und nun das Programm:

```
10 RANDOMIZE
20 INPUT "Anzahl der Spiele"; N
   A = 0, B = 0
   FOR J = 1 TO N
   X = INT (6* RND) + 1
   Y = INT (6* RND) + 1
   Z = X + Y
   IF Z< = 7 THEN 120
100 B = B + 1
110 GOTO 130
120 A = A + 1
130 NEXT J
   PRINT "Anzahl d. Gewinne:"; A,B
   PRINT "Theoret. Anzahl:"; 21 * N/36, 15* N/36
   GOTO 20
```

Es folgen die Ergebnisse einiger Simulationen:

| | | | |
|---|---|---|---|
| N = 36 | A = 23 | , B = 13 | Theoretisch: 21, 15 |
| N = 36 | A = 20 | , B = 16 | 21, 15 |
| N = 72 | A = 45 | , B = 27 | 42, 30 |
| N = 72 | A = 41 | , B = 31 | 42, 30 |
| N =360 | A =194 | , B =166 | 210,150 |
| N =360 | A =219 | , B =141 | 210,150 |

Als zweites Beispiel soll eine Brown'sche Bewegung in der x,y-Ebene, ausgehend vom Nullpunkt mit dem Computer simuliert werden.
Pro Spiel sollen M Schritte mit einer zufälligen Länge S aus (0,1) und einem zufälligen Winkel W aus $(0,2\pi)$ betrachtet werden. Es sollen N Spiele durchgeführt und für diese die mittlere quadratische Entfernung E(Q) des Teilchens berechnet werden. Wir erzeugen die Zufallszahlen S und W mit den Befehlen S = RND und W = 2* PI* RND. Dann wird die Position des Teilchens nach jedem Schritt durch die Koordinaten X,Y festgelegt:

$$X = X + S * \mathrm{COS}(W)$$

$$Y = Y + S * \mathrm{SIN}(W)$$

Nach M Schritten berechnen wir die quadratische Entfernung vom Nullpunkt: Q = X*X* Y
Nach N Wiederholungen wird $E(Q) = \sum_{i=1}^{N} (x_i^2 + y_i^2)/N$ berechnet und ausgedruckt.

**Das Programm lautet:**

```
10 RANDOMIZE
20 INPUT N,M
   E = Ø
   FOR J = 1 TO N
   X = Ø
   Y = Ø
   FOR K = 1 TO M
   W = 2* PI * RND
   S = RND
   X = X + S* COS(W)
   Y = Y + S* SIN(W)
   NEXT K
   Q = X* X + Y* Y
   E = E + Q
   NEXT J
   PRINT "Mittl. quadr. Abstand ="; E/N
   GOTO 2Ø
```

HR Dir.Dr. Helmut Brunner
HAK Mödling
Maria Theresien-Gasse 25
2340 Mödling