Mathe II für Informatiker — Sommersemester 05
Walter Oevel — 19. 4. 2005

Übungsblatt 2

Mit $*$ gekennzeichnete Aufgaben können zum Sammeln von Bonuspunkten verwendet werden. Lösungen sind per Web-Formular unter `http://math-www.upb.de/~walter` ($\longrightarrow$ Lehre SS 05 $\longrightarrow$ Übungen) bis spätestens Do, 28.4.05, abzuliefern.

Aufgabe 7: (Schnelle komplexe Multiplikation)
In einer Software-Umgebung stehe reelle Arithmetik zur Verfügung. Die Kosten der Addition/Subtraktion reeller Zahlen sei gegenüber den Kosten einer Multiplikation vernachlässigbar. Es soll in dieser Umgebung die Arithmetik für komplexe Zahlen implementiert werden, welche als Listen $[x, y]$ von Real- und Imaginärteil dargestellt werden. Die Definition

$$[x_1, y_1] \cdot [x_2, y_2] = [x_1 \cdot x_2 - y_1 \cdot y_2, x_1 \cdot y_2 + y_1 \cdot x_2]$$

der komplexen Multiplikation $(x_1 + i \cdot y_1) \cdot (x_2 + i \cdot y_2)$ benötigt vier reelle Multiplikationen. Finde einen Weg, das komplexe Produkt über geeignete Zwischenergebnisse mit nur 3 reellen Multiplikationen zu berechnen.
Hinweis: Betrachte $(x_1 + y_1) \cdot (x_2 + y_2)$.

Musterlösung:
Wir betrachten das Produkt

$$\underbrace{[x_3, y_3]}_{x_3 + i \cdot y_3} = \underbrace{[x_1, y_1]}_{x_1 + i \cdot y_1} \cdot \underbrace{[x_2, y_2]}_{x_2 + i \cdot y_2}.$$

Der gegebene Hinweis

$$(x_1 + y_1) \cdot (x_2 + y_2) = x_1 \cdot y_2 + y_1 \cdot x_2 + x_1 \cdot x_2 + y_1 \cdot y_2.$$

liefert

$$x_1 \cdot y_2 + y_1 \cdot x_2 = (x_1 + y_1) \cdot (x_2 + y_2) - x_1 \cdot x_2 - y_1 \cdot y_2.$$

Die linke Seite ist der gesuchte Imaginärteil des Produktes, die rechte Seite besteht aus der Differenz von drei Hilfsgrößen, die jeweils mit nur einer reellen Multiplikation berechnet werden können. Beachte, dass auch der gesuchte Realteil des Produkts

$$x_1 \cdot x_2 - y_1 \cdot y_2$$

mit Hilfe zweier dieser Größen ausgedrückt werden kann. Damit liefert der Algorithmus

$$\begin{aligned}
t_1 &= x_1 \cdot x_2; \\
t_2 &= y_1 \cdot y_2; \\
t_3 &= (x_1 + y_1) \cdot (x_2 + y_2); \\
x_3 &= t_1 - t_2; \\
y_3 &= t_3 - t_1 - t_2.
\end{aligned}$$

den Real- und den Imaginärteil des komplexen Produktes mit 3 reellen Multiplikationen und insgesamt 5 reellen Additionen/Subtraktionen. Die ursprüngliche Definition des Produkts braucht 4 reelle Multiplikationen und 2 reelle Additionen/Subtraktionen.

Aufgabe 8: (Faktorpolynome, Horner–Schema)
Betrachte das Polynom $p(x) = x^n - 1$. Für jedes $n \in \mathbb{N}$ läßt sich der Linearfaktor $x - 1$ abspalten (warum?). Bestimme für beliebiges n die explizite Form des Faktorpolynoms $p(x)/(x - 1)$.
Hinweis: Durchlaufe das Horner–Schema.

Musterlösung:
Da $x = 1$ für jedes n eine Nullstelle ist, läßt sich stets der Linearfaktor $x - 1$ abspalten. Die explizite Darstellung

$$\frac{x^n - 1}{x - 1} = b_0 \cdot x^{n-1} + b_1 \cdot x^{n-2} + \cdots + b_{n-1}$$

wird durch das Horner-Schema 1.13 mit $x^* = 1$ und den Polynomkoeffizienten $(a_0, \ldots, a_n) = (-1, 0, \ldots, 0, 1)$ geliefert:

$$\begin{aligned} b_0 &= a_n = 1; \\ b_k &= b_{k-1} \cdot x^* + a_{n-k} = b_{k-1} = 1 \quad (k = 1, 2, \ldots, n-1), \\ b_n &= b_{n-1} \cdot x^* + a_0 = b_{n-1} - 1 = 0 = p(1). \end{aligned}$$

Also:

$$\boxed{x^n - 1 = (x - 1) \cdot \Big(x^{n-1} + x^{n-2} + \cdots + x + 1\Big).}$$

Anmerkung:

$$1 + x + x^2 + \cdots + x^{n-1} = \frac{x^n - 1}{x - 1}$$

ist die wohlbekannte (auf den jungen Gauß zurückgehende) Summenformel für geometrische Reihen.

Aufgabe 9: (MuPAD)
Lies die MuPAD–Hilfeseiten zu `factor` und `expand`. Faktorisiere die Polynome $x^n - 1$ für $n = 2, 3, \ldots, 10$. Benutze `expand`, um die in Aufgabe 8 gefragte Form des Faktorpolynoms $p(x)/(x - 1)$ zu verifizieren.

Musterlösung:
Wir realisieren die 3 Werte $n = 2, \ldots, 10$ durch eine `for`-Schleife:

```
>> for n from 2 to 10 do
&>   print(n, factor(x^n - 1));
&> end_for:
                           2, (x - 1) (x + 1)
                                             2
                        3, (x - 1) (x + x  + 1)
                                                2
                      4, (x - 1) (x + 1) (x  + 1)
                                    2    3    4
                   5, (x - 1) (x + x  + x  + x  + 1)
                                       2              2
             6, (x - 1) (x + 1) (x + x  + 1) (- x + x  + 1)
```

```
                              2    3    4    5    6
         7, (x - 1) (x + x  + x  + x  + x  + x  + 1)
                                      2         4
             8, (x - 1) (x + 1) (x  + 1) (x  + 1)
                                 2         3    6
            9, (x - 1) (x + x  + 1) (x  + x  + 1)
                           2    3    4              2    3    4
  10, (x - 1) (x + 1) (x + x  + x  + x  + 1) (- x + x  - x  + x  + 1)
```

Für $n = 4, 6, 8, 9, 10$ wird eine weitergehende Faktorisierung gefunden. (Für gerades n ist dies kein Wunder, da dann auch $x = -1$ eine Wurzel ist und der Linearfaktor $x + 1$ stets abgespalten werden kann.) Die folgende Schleife expandiert das Produkt aller Faktoren außer dem Linearfaktor $x - 1$ und liefert damit die in Aufgabe 8 gefragte Form von $p(x)/(x - 1)$:

```
>> for n from 2 to 10 do
&>   print(n, expand(factor(x^n - 1)/(x - 1)));
&> end_for:
                              2, x + 1
                                   2
                           3, x + x  + 1
                                 2    3
                         4, x + x  + x  + 1
                              2    3    4
                      5, x + x  + x  + x  + 1
                            2    3    4    5
                    6, x + x  + x  + x  + x  + 1
                         2    3    4    5    6
                 7, x + x  + x  + x  + x  + x  + 1
                       2    3    4    5    6    7
               8, x + x  + x  + x  + x  + x  + x  + 1
                    2    3    4    5    6    7    8
            9, x + x  + x  + x  + x  + x  + x  + x  + 1
                  2    3    4    5    6    7    8    9
          10, x + x  + x  + x  + x  + x  + x  + x  + x  + 1
```

Aufgabe 10*: (Polynomwurzeln. 10 Bonuspunkte)

Seien M_1, M_2 Zahlen (die von `hex` zufällig gewählt werden). Das Polynom

$$p(x) = x^4 - 2 \cdot M_1 \cdot x^3 + (2 - M_2^2) \cdot x^2 - 4 \cdot M_1 \cdot x - 2 \cdot M_2^2$$

hat stets die komplexe Nullstelle $\sqrt{2} \cdot i$. Bestimme alle Nullstellen!
Hinweis: Satz 1.19 liefert eine zweite Nullstelle. Nach Abspalten zweier Linearfaktoren sind nur noch die Nullstellen eines quadratischen Faktorpolynoms zu suchen.
Relevante MuPAD-Funktionen: `factor, solve`.

Musterlösung:
Hier nur die mit MuPAD ermittelten Endergebnisse:

```
>> p:= x^4 - 2*M1*x^3 + (2 - M2^2)*x^2 - 4*M1*x - 2*M2^2
>> factor(p);
```

```
                         2                  2     2
                    - (x   + 2) (2 x M1 - x   + M2 )

>> solve(p, x)
           2     2 1/2           2     2 1/2        1/2     1/2
 {M1 + (M1   + M2 )   , M1 - (M1   + M2 )   , - I 2   , I 2   }
```

Aufgabe 11: (Zum Horner-Schema)
Das Horner-Schema ist für die Auswertung „dicht besetzter" Polynome (fast alle Koeffizienten sind ungleich 0) ein guter und schneller Algorithmus. Diskutiere die Situation für „dünn besetzte" Polynome (fast alle Koeffizienten sind 0) anhand des Beispiels $x^n - 1$.

Musterlösung:
Für $p(x) = x^n - 1$ braucht i.W. nur x^n berechnet zu werden. Per „repeated squaring" ist die Anzahl der dafür benötigten Multiplikationen in der Größenordnung $\log_2(n)$. Im Vergleich dazu ist das Horner-Schema mit n Multiplikationen langsam.

Allgemein: Sind nur k Terme des Polynoms (vom Grad n) besetzt und berechnet man beispielsweise alle Terme einzeln per „repeated squaring", so ergibt sich ein Aufwand, der auf jeden Fall geringer ist als $k \cdot \log_2(n)$. Sind also weniger als $n/\log_2(n)$ Terme im Polynom vorhanden, ist das „repeated squaring" auf jeden Fall schneller.

Aufgabe 12: (Vietascher Wurzelsatz)
a) Seien $x_1, \ldots, x_k$ die unterschiedlichen Wurzeln des Polynoms $p(x) = a_n \cdot x^n + \cdots + a_1 \cdot x + a_0$ mit den Vielfachheiten $n_1, \ldots, n_k$, wobei $n = n_1 + \cdots + n_k$ und $a_n \neq 0$ gelte. Zeige:

$$x_1^{n_1} \cdot x_2^{n_2} \cdot \ldots \cdot x_k^{n_k} = (-1)^n \cdot \frac{a_0}{a_n}.$$

b) Betrachte $p(x) = 3 \cdot x^3 - 51 \cdot x^2 - 3\,x + 51$. Die Werte $x_{1,2} = \pm 1$ sind Nullstellen. Finde die dritte Wurzel ohne Polynomdivision.

Musterlösung:
a) Der Fundamentalsatz

$$p(x) = a_n \cdot x^n + \cdots + a_1 \cdot x + a_0 = a_n \cdot (x - x_1)^{n_1} \cdot \ldots \cdot (x - x_k)^{n_k}$$

liefert bei Auswertung an der Stelle $x = 0$:

$$p(0) = a_0 = a_n \cdot (-x_1)^{n_1} \cdot \ldots (-x_k)^{n_k} = a_n \cdot (-1)^{n_1 + \ldots + n_k} \cdot x_1^{n_1} \cdot \ldots \cdot x_k^{n_k}.$$

b) Nach a) muss gelten

$$x_1 \cdot x_2 \cdot x_3 = -\frac{51}{3} = -17 \quad \Rightarrow \quad x_3 = -\frac{17}{x_1 \cdot x_2} = 17.$$

Aufgabe 13: (Schranken für Polynomwurzeln. Für Ehrgeizige, anspruchsvoll!)
Sei z eine (eventuell komplexe) Wurzel des Polynoms $p(x) = a_n \cdot x^n + \cdots + a_1 \cdot x + a_0$, $a_n \neq 0$.
Zeige

$$|z| \leq \max \left\{ \left| \frac{a_0}{a_n} \right|, 1 + \left| \frac{a_1}{a_n} \right|, 1 + \left| \frac{a_2}{a_n} \right|, \ldots, 1 + \left| \frac{a_{n-1}}{a_n} \right| \right\}.$$

Anleitung: Zeige, dass

$$|a_n \cdot z^n| \overset{(*)}{>} |a_{n-1}| \cdot |z^{n-1}| + \cdots + |a_1| \cdot |z| + |a_0| \quad \Big(\geq |a_{n-1} \cdot z^{n-1} + \cdots + a_1 \cdot z + a_0| \Big)$$

gilt, wenn $|z|$ größer als die angegebene Schranke ist (damit kann z keine Nullstelle sein).
Zeige $(*)$ per Induktion nach j:

$$|a_n| \cdot |z^j| > |a_{j-1}| \cdot |z^{j-1}| + \cdots + |a_1| \cdot |z| + |a_0| \quad \text{für alle} \quad j = 1, \ldots, n.$$

Musterlösung:
Die Grundidee ist:

$$p(z) = a_n \cdot z^n + \cdots + a_0 = a_n \cdot z^n \cdot \Big(1 + \frac{a_{n-1}}{a_n \cdot z} + \cdots + \frac{a_0}{a_n \cdot z^n} \Big).$$

Für hinreichend großes z ist der Faktor $(1 + \sum ...)$ ungefähr 1, so dass $p(z) \neq 0$ gelten muss.

Sei $|z|$ größer als die angegebene Schranke. Wir zeigen

$$|a_n| \cdot |z^j| > |a_{j-1}| \cdot |z^{j-1}| + \cdots + |a_1| \cdot |z| + |a_0|$$

für alle $j = 1, \ldots, n$. Für $j = 1$ ist als Induktionsstart

$$|a_n| \cdot |z| > |a_0|$$

zu zeigen. Dies ist nach der Voraussetzung „$|z|$ sei größer als die angegebene Schranke“ aber erfüllt.
Induktionsschritt $j \to j + 1$: die Induktionsvoraussetzung liefert

$$|a_j| \cdot |z^j| + \underbrace{|a_{j-1}| \cdot |z^{j-1}| + \cdots + |a_1| \cdot |z| + |a_0|}_{<|a_n| \cdot |z|^j} < |a_j| \cdot |z^j| + |a_n| \cdot |z^j| = (|a_j| + |a_n|) \cdot |z^j|.$$

Die Voraussetzung „$|z|$ sei größer als die angegebene Schranke“ liefert

$$|z| > 1 + \frac{|a_j|}{|a_n|}, \quad \text{also} \quad |a_j| + |a_n| < |a_n| \cdot |z|.$$

Macht zusammen:

$$|a_j| \cdot |z^j| + |a_{j-1}| \cdot |z^{j-1}| + \cdots + |a_1| \cdot |z| + |a_0| < |a_n| \cdot |z| \cdot |z^j| = |a_n| \cdot |z^{j+1}|.$$

Dies ist aber genau die zu beweisende Aussage für $j+1$. Für $j = n$ folgt über die Dreiecksungleichung

$$|\underbrace{a_{n-1} \cdot z^{n-1} + \cdots + a_1 \cdot z + a_0}_{X}| \leq |a_{n-1}| \cdot |z^{n-1}| + \cdots + |a_1| \cdot |z| + |a_0| < |a_n| \cdot |z^n| = |\underbrace{a_n \cdot z^n}_{Y}|.$$

Aus $|X| < |Y|$ folgt $|Y + X| > 0$, denn die Dreiecksungleichung liefert

$$|Y| = |Y + X - X| \leq |Y + X| + |-X| = |Y + X| + |X| \quad \Rightarrow \quad 0 < |Y| - |X| \leq |Y + X|.$$

Insgesamt ergibt dies

$$|\underbrace{a_n \cdot z^n}_{Y} + \underbrace{a_{n-1} \cdot z^{n-1} + \cdots + a_1 \cdot z + a_0}_{X}| = |p(z)| > 0,$$

d.h., z kann keine Nullstelle sein, wenn $|z|$ größer als die angegebene Schranke ist.

Aufgabe 14*: (Matrixdiagonalisierung. 10 Bonuspunkte. Zum Knobeln. Rechenintensiv!) Seien M_1, M_2 Zahlen (die von `hex` zufällig gewählt werden). Finde drei unterschiedliche 3×3 Matrizen, deren Quadrat die Matrix A mit den folgenden Einträgen ergibt:

$$\begin{aligned}
A_{11} &= 4 + (M_1 - 1) \cdot (M_1 - 5) \cdot (M_1 + 3) \cdot M_2,\\
A_{12} &= M_2 \cdot (1 - M_1) \cdot (M_1 - 5),\\
A_{13} &= (M_1 - 1) \cdot (M_1 - 5),\\
A_{21} &= (M_1 + 3) \cdot (M_1 + 2) \cdot (M_1 + 6) \cdot (3 \cdot M_2 + M_1 \cdot M_2 - 1),\\
A_{22} &= 16 + M_1 \cdot (8 + M_1) - M_2 \cdot (M_1 + 2) \cdot (M_1 + 3) \cdot (M_1 + 6),\\
A_{23} &= (M_1 + 2) \cdot (M_1 + 3) \cdot (M_1 + 6),\\
A_{31} &= 7 \cdot M_2 \cdot (M_1 + 3) \cdot (2 \cdot M_1 + 1) \cdot (3 \cdot M_2 + M_1 \cdot M_2 - 1),\\
A_{32} &= 7 \cdot M_2 \cdot (2 \cdot M_1 + 1) \cdot (1 - 3 \cdot M_2 - M_1 \cdot M_2),\\
A_{33} &= (M_1 - 3)^2 + 7 \cdot M_2 \cdot (M_1 + 3) \cdot (2 \cdot M_1 + 1).
\end{aligned}$$

Anleitung: Die Aufgabe ist leicht zu lösen, wäre A eine Diagonalmatrix. Also: diagonalisiere A. (Die Diagonalisierung ist rechenintensiv. Es bietet sich an, dies mit MuPAD mittels der Funktion `linalg::eigenvectors` zu tun.) Zur Kontrolle: A hat stets nur ganzzahlige Eigenwerte und ist diagonalisierbar.
Löse die Aufgabe für die Diagonalmatrix der Eigenwerte von A. Wie lassen sich aus den „Wurzeln" der Diagonalmatrix die „Wurzeln" von A konstruieren?
Relevante MuPAD-Funktionen: `matrix, linalg::eigenvectors`. Matrizen werden mittels `*` multipliziert. Eine Matrix T wird mittels `1/T` oder `T^(-1)` invertiert.

Musterlösung:
Hier nur die mit MuPAD berechneten Zwischenergebnisse.

```
>> A:= matrix(3, 3):
>> A[1,1]:= 4 + (M1 - 1)*(M1 - 5)*(M1 + 3)*M2;
>> A[1,2]:= M2*(1 - M1)*(M1 - 5);
>> A[1,3]:= (M1 - 1)*(M1  - 5);
>> A[2,1]:= (M1 + 3)*(M1 + 2)*(M1 + 6)*(3*M2 + M1*M2- 1);
>> A[2,2]:= 16 + M1*(8 + M1) - M2*(M1 + 2)*(M1 + 3)*(M1 + 6);
>> A[2,3]:= (M1 + 2)*(M1 + 3)*(M1 + 6);
>> A[3,1]:= 7*M2*(M1 + 3)*(2*M1 + 1)*(3*M2 + M1*M2 - 1);
>> A[3,2]:= 7*M2*(2*M1 + 1)*(1 - 3*M2- M1*M2);
>> A[3,3]:= (M1 - 3)^2 + 7*M2*(M1 + 3)*(2*M1 + 1);
```

Eigenwerte mit ihrer Vielfachheit samt zugehörigen Eigenvektoren werden von der MuPAD-Funktion `linalg::eigenvectors` geliefert:

```
>> linalg::eigenvectors(A);
-- --       -- +-        -+ -- --  --
|  |        |  |     1    |  |  |  |
|  |        |  |  ------  |  |  |  |
|  |        |  |  M1 + 3  |  |  |  |     2
|  |  4, 1, |  |          |  |  |, |  M1  - 6 M1 + 9, 1,
|  |        |  |     1    |  |  |  |
|  |        |  |          |  |  |  |
|  |        |  |     0    |  |  |  |
-- --       -- +-        -+ -- --  --
   -- +-                       -+ -- --  --                                    -- +-     -+ -- -- --
   |  |            1            |  |  |  |                                    |  |  0   |  |  |  |
   |  |  - ----------------     |  |  |  |                                    |  |      |  |  |  |
   |  |    3 M2 + M1 M2 - 1     |  |  |  |                2                   |  |  1   |  |  |  |
   |  |                         |  |  |, |  8 M1 + M1  + 16, 1,              |  |  --  |  |  |  |
   |  |           0             |  |  |  |                                    |  |  M2  |  |  |  |
   |  |                         |  |  |  |                                    |  |      |  |  |  |
   |  |           1             |  |  |  |                                    |  |  1   |  |  |  |
   -- +-                       -+ -- --  --                                    -- +-     -+ -- -- --
```

Alle Eigenwerte sind Quadrate ganzer Zahlen:

$$4 = 2^2, \quad M_1^2 - 6\,M_1 + 9 = (M_1 - 3)^2, \quad M_1^2 + 8\,M_1 + 16 = (M_1 + 4)^2.$$

Es gilt damit

$$A = T\,D_2\,T^{-1} \quad \text{mit} \quad D_2 = \mathrm{diag}(2^2, (M_1 - 3)^2, (M_1 + 4)^2)$$

und

$$T = \begin{pmatrix} 1 & -1 & 0 \\ M_1 + 3 & 0 & 1 \\ 0 & 3 \cdot M_2 + M_1 \cdot M_2 - 1 & M_2 \end{pmatrix}.$$

Die Matrix T wurde dabei spaltenweise aus den von MuPAD gefundenen Eigenvektoren aufgebaut, wobei die gefundenen Eigenvektoren jeweils mit $M_1 + 3$ bzw. $3 \cdot M_2 + M_1 \cdot M_2 - 1$ bzw. M_2 multipliziert wurden, damit T schöne ganzzahlige Einträge hat (dies ist reine Kosmetik):

```
>> D2:= matrix([[4,      0      ,   0      ],
                [0, (M1 - 3)^2,   0      ],
                [0,      0      ,(M1 + 4)^2]]):

>> T:= matrix([[  1 , -1       , 0],
               [M1+3,  0   ,  1  ],
               [  0 , 3*M2 + M1*M2 -1, M2]]):
```

Kontrolle:

```
>> expand(A - T*D2*T^(-1))
                            +-         -+
                            |  0, 0, 0  |
                            |           |
                            |  0, 0, 0  |
                            |           |
                            |  0, 0, 0  |
                            +-         -+
```

Finden wir eine Matrix D_1 mit mit $D_1^2 = D_2$, so folgt

$$A = T\,D_2\,T^{-1} = T\,D_1^2\,T^{-1} = T\,D_1\,T^{-1}T\,D_1T^{-1} = (T\,D_1\,T^{-1})^2,$$

und wir haben eine Matrix $T\,D_1\,T^{-1}$ gefunden, deren Quadrat A ergibt. Man findet sofort 8 Matrizen D_1 mit $D_1^2 = D_2$:

$$D_1 = \mathrm{diag}(\pm 2, \pm(M_1 - 3), \pm(M_1 + 4))$$

wobei die Vorzeichen beliebig gewählt werden können. Eine der so konstruierten Matrizen ist (es werden lauter positive Vorzeichen gewählt):

```
>> D1:= matrix(3, 3, [2, M1 - 3 , M1 + 4], Diagonal);
                               +-                      -+
                               |  2,     0,        0    |
                               |                        |
                               |  0, M1 - 3,       0    |
                               |                        |
                               |  0,     0,    M1 + 4   |
                               +-                      -+

>> map(T*D1*T^(-1), factor)
                 array(1..3, 1..3,
                                                  2
                   (1, 1) = - 15 M2 - 2 M1 M2 + M1  M2 + 2,
                   (1, 2) = M2 (- M1 + 5),
                   (1, 3) = M1 - 5,
                   (2, 1) = (M1 + 2) (M1 + 3) (3 M2 + M1 M2 - 1),
                                                    2
                   (2, 2) = M1 - 6 M2 - 5 M1 M2 - M1  M2 + 4,
                   (2, 3) = (M1 + 2) (M1 + 3),
                   (3, 1) = 7 M2 (M1 + 3) (3 M2 + M1 M2 - 1),
                   (3, 2) = -7 M2 (3 M2 + M1 M2 - 1),
                   (3, 3) = M1 + 21 M2 + 7 M1 M2 - 3
                 )
```

(Die Vereinfachung der Einträge per Faktorisierung `map(..., factor)` ist nur hier in dieser Musterlösung interessant, wo M_1 und M_2 nicht-numerische Symbole sind.) Eine andere Matrix, deren Quadrat A ergibt:

```
>> D1:= matrix(3, 3, [2, 3 - M1 , M1 + 4], Diagonal);

                              +-                       -+
                              |  2,      0,        0    |
                              |                         |
                              |  0, - M1 + 3,      0    |
                              |                         |
                              |  0,      0,    M1 + 4   |
                              +-                       -+

>> map(T*D1*T^(-1), factor)
              array(1..3, 1..3,
                                             2
                (1, 1) = 3 M2 - 2 M1 M2 - M1  M2 + 2,
```

```
  (1, 2) = M2 (M1 - 1),
  (1, 3) = - M1 + 1,
  (2, 1) = (M1 + 2) (M1 + 3) (3 M2 + M1 M2 - 1),
                                   2
  (2, 2) = M1 - 6 M2 - 5 M1 M2 - M1  M2 + 4,
  (2, 3) = (M1 + 2) (M1 + 3),
  (3, 1) = M2 (M1 + 3) (2 M1 + 1) (3 M2 + M1 M2 - 1),
  (3, 2) = M2 (2 M1 + 1) (- 3 M2 - M1 M2 + 1),
                                       2
  (3, 3) = - M1 + 3 M2 + 7 M1 M2 + 2 M1  M2 + 3
)
```