Technische Universität Ilmenau
Fakultät für Mathematik
und Naturwissenschaften
Institut für Mathematik
http://www.tu-ilmenau.de/fakmn/math.html

Postfach 10 05 65
D - 98684 Ilmenau
Germany
Tel.: 03677/69 3267
Fax: 03677/69 3272
Telex: 33 84 23 tuil d.
email: werner.neundorf@tu-ilmenau.de

Preprint No. M 16/07

Kondition eines Problems sowie Gleitpunktarithmetik in den CAS Maple, MATLAB und in höheren Programmiersprachen

Werner Neundorf

November 2007

[‡]MSC (2000): 65-01, 65-04, 65F05, 65F10, 65Y99, 68U99, 68Q25

Zusammenfassung

Es gibt viele Möglichkeiten, mathematische und andere Aufgabenstellungen mit Computeralgebrasystemen wie Maple, MATLAB, *Mathematica* oder Derive zu behandeln und zu lösen. Dazu findet der Leser eine immense Fülle von Informationen und Literatur sowie Demonstrationen und Hilfen in den Systemen selbst.
Trotzdem zeigt sich, dass zu spezifischen Fragestellungen eine detaillierte Aufbereitung der Sachverhalte sowie ihre Darstellung und Erläuterung sehr nützlich erscheint. In einem einführendem Abschnitt wird der Umgang mit großen Datenmengen bei ihrer Ausgabe erläutert. Der Schwerpunkt in dieser Arbeit wird jedoch die Untersuchung der Eigenschaften der Gleitpunktarithmetik *double* in den CAS Maple und MATLAB sein, mit vergleichenden Betrachtungen anhand von Beispielen.

Inhaltsverzeichnis

Kapitel 1

Rechengenauigkeit

1.1 Einleitung

Naturgemäß sind Publikationen, die auf Software Bezug nehmen und diese für spezielle Aufgabenstellungen verwenden, einer ständigen Gefahr ausgesetzt. Upgrades oder Updates von Software sowie unterschiedliche Versionen für verschiedene Rechnerplattformen führen unter Umständen dazu, dass die in den Arbeiten dargestellten und mit der Software zusammenhängenden Sachverhalten nicht mit der Version übereinstimmen, die dem Leser zur Verfügung steht.
Die CAS erfahren eine ständige Weiterentwicklung und werden immer größer und leistungsfähiger. Programme, Files und Arbeitsblätter, die unter älteren Versionen entstanden und gelaufen sind, sollte man bezüglich der Möglichkeiten und Veränderungen in neuen Versionen stets kritisch begutachten und eventuell anpassen. Das betrifft auch die breite Palette der Menüfunktionen.
Alles dies trifft auch auf die CAS Maple und MATLAB zu. Manche von früheren Versionen erzeugte Resultate haben sich in Darstellung und Inhalt geändert. Darüber hinaus ist es ein ständiges Anliegen der Hersteller, Unzulänglichkeiten und Fehler in den Programmen und ihren Entwicklungsumgebungen zu beseitigen.

Es sollen zunächst nur einige Aspekte und Probleme zu Maple 9.5 und MATLAB 7.2.0.232 (R2006a) angeschnitten werden. Dazu werden auch Vergleiche mit älteren Versionen der CAS sowie mit Implementationen in höheren Programmiersprachen wie Pascal herangezogen.
Detaillierte Untersuchungen zu Lösungsansätzen und ihrer Genauigkeit in der genutzten Gleitpunktarithmetik (GPA) bzw. Gleitpunktformaten (GPF) erfolgen in den weiteren Abschnitten.

1.2 Auswertung von Ausdrücken

Dass Ungenauigkeiten bei der numerischen Berechnung von Ausdrücken mit Wurzeln sowie numerischen Auswertung von Funktionen/Kommandos entstehen, ist verbunden mit der benutzten GPA und dem Rundungsregime. Dabei erwartet man von neueren Versionen von CAS im Allgemeinen bessere Lösungen.
Beim Aufruf der Befehle oder in den Paketen verweisen die Hersteller oft explizit auf solche Veränderungen. Der Nutzer sollte sich im Zweifelsfalle der Mühe unterziehen, das zu kontrollieren. An drei Beispielen wollen wir das Spektrum der Möglichkeiten anhand von Maple demonstrieren.

Eigentlich erwartet man in Maple bei der Rechnung mit `Digits:=10` (entspricht dem *real*-Format und ist Standard), dass die letzte Ziffer der dargestellten dezimalen Mantisse aus einem vernünftigen Rundungseffekt zur nächsten Ziffer entsteht. Dass dies in Maple V schon funktioniert hat, aber in der neueren Version Maple 8, 9.5 anders, aber eigentlich schlechter gelöst worden ist, kann einen Nutzer eher schockieren als zufriedenstellen. Es ist oft so, dass bei der Gleitpunktdarstellung (GP-Darstellung) von exakten Ergebnissen die erste Dezimale außerhalb der angezeigten Mantisse einfach abgeschnitten wird.
Als eines von zahlreichen Beispielen soll die Auswertung des Ausdrucks mit Bruch und Wurzel $(1+\sqrt{5})/2$ demonstriert werden.
MATLAB erzeugt in seinem *double*-Format bei 16–stelliger Mantisse den Wert `1.618033988749895e+000`.
Bei Maple muss man von vornherein bemerken, dass es allgemein bei der Wurzelberechnung "seine Probleme hat".

Rechnungen in Maple (Datei: *calc1.mws*)

Maple V: vernünftige Rundung

```
> t:=(1+sqrt(5))/2; evalf(t);
  Digits:=8:  evalf(t);      # letzte Mantissenstelle gerundet
  Digits:=9:  evalf(t);
  Digits:=10: evalf(t);
  Digits:=11: evalf(t);
  Digits:=12: evalf(t);
  Digits:=16: evalf(t);
  Digits:=20: evalf(t); Digits:=10:
```

$$t := \frac{1}{2} + \frac{1}{2}\sqrt{5}$$

1.618033989

1.6180340

1.61803399

1.618033989

1.6180339888

1.61803398875

1.618033988749895

1.6180339887498948482

Der Maple-Wert bei `Digits=16` entspricht dem MATLAB-Ergebnis.
Aber bei bestimmten Genauigkeiten besteht eine Gefahr durch den Rundungseffekt bei Weiterverarbeitung der Größen.

```
> z:=(1+sqrt(5))/2;
  Digits:=10:
  z10:=evalf((1+sqrt(5))/2);
  Digits:=20:
  evalf(z-z10);
  1000*evalf(z-z10); # gefaehrlich, da gerundete Stelle in der Mantisse
                     # nach vorne rutscht
```

$$z := \frac{1}{2} + \frac{1}{2}\sqrt{5}$$

$$z10 := 1.618033989$$

$$-0.2501051518\ 10^{-9}$$

$$-0.2501051518000\ 10^{-6}$$

Maple 8, 9.5, Classic Maple 9.5
Hier stellt sich die Frage, warum an einer Stelle, nämlich bei `Digits=10`, eine Verschlechterung gegenüber Maple V auftritt?

```
> t:=(1+sqrt(5))/2;
  evalf(t);
  Digits:=8:  evalf(t);
  Digits:=9:  evalf(t);
  Digits:=10: evalf(t);   # letzte Mantissenstelle falsch
  Digits:=11: evalf(t);
  Digits:=12: evalf(t);
  Digits:=15: evalf(t);
  Digits:=16: evalf(t);
  Digits:=17: evalf(t);
  Digits:=20: evalf(t);
  Digits:=10:
```

$$t := \frac{1}{2} + \frac{\sqrt{5}}{2}$$

$$1.618033988$$

$$1.6180340$$

$$1.61803399$$

$$1.6180339888$$

$$1.61803398875$$

$$1.61803398874990$$

$$1.618033988749895$$

$$1.6180339887498948$$

$$1.6180339887498948482$$

Größere Gefahr des Rundungseffekts bei Weiterverarbeitung

```
> z:=(1+sqrt(5))/2;
  Digits:=10:
  z10:=evalf((1+sqrt(5))/2);
  Digits:=20:
  evalf(z-z10);
  1000*evalf(z-z10); # gefaehrlich, da falsche Stelle in der Mantisse
                     # nach vorne rutscht
```

$$z := \frac{1}{2} + \frac{\sqrt{5}}{2}$$

$$z10 := 1.618033988$$

$$0.7498948482\ 10^{-9} \quad \text{bzw.} \quad 7.498948482\ 10^{-10}$$

$$0.7498948482000\ 10^{-6} \quad \text{bzw.} \quad 7.498948482000\ 10^{-7}$$

1.3 Termumformung und mathematische Äquivalenz von Formeln

Empfiehlt man dem Betrachter für die Berechnung eines Wertes die mathematisch äquivalenten Formeln

$$(1) \quad \frac{1}{(3+\sqrt{10})^4} = 6.93481\,60951\,23042\,52272...\text{E}-4,$$

$$(2) \quad (3-\sqrt{10})^4,$$

$$(3) \quad (19-6\sqrt{10})^2,$$

$$(4) \quad 721-228\sqrt{10},$$

$$(5) \quad \frac{1}{(19+6\sqrt{10})^2},$$

$$(6) \quad \frac{1}{721+228\sqrt{10}},$$

so wird er auf den ersten Blick vom ästhetischen Standpunkt aus sich für die Verwendung der Formel (3) entscheiden. Dass er damit fast den numerisch schlechtesten Ausdruck gewählt hat, wird verursacht durch das so genannte Subtraktionsdilemma. Im folgenden Turbo-Pascal-Programm mit wahlweiser "künstlicher" Festlegung der Mantissenlänge auf 2...5 Byte kann man die "Güte" der Formeln schnell herausfinden.

```
program Aequivalenz;
{ Aequivalenz von Quadratwurzel-Bruch-Termen bei kuenstlich
  gekuerzter Mantisse }
{ equiv_2.pas}

uses crt;
```

```
type float = real;
var  arr   : array[0..5] of byte;
     x     : float absolute arr;
     n,m   : integer;

{kuenstliches Abschneiden der Mantisse auf n-1 Bytes}
procedure clear;
var i:integer;
begin
  for i:=1 to 6-n do arr[i]:=0
end;

begin
  clrscr;
  writeln('Aequivalenz von Termen');
  writeln;
  writeln(
   'Byteanzahl der GKZ/Laenge der Mantisse/gueltige Dez.stellen');
  writeln('  n = 3 / 2 / 4                ');
  writeln('      4 / 3 / 7   Format single');
  writeln('      5 / 4 / 9                ');
  writeln('      6 / 5 / 11  Format real  ');
  writeln;
  write('  n = ');
  readln(n);
  writeln;
  case n of  { m --> Format m+6,  6 = Vorzeichen & Punkt & E+00 }
    3 : m:=4;
    4 : m:=7;
    5 : m:=9;
    6 : m:=11
  end;
  {Formel (1), Formeln (2),...,(6) analog}

  x:=10;      clear;
  x:=sqrt(x); clear;
  x:=3+x;     clear;
  x:=sqr(x);  clear;
  x:=sqr(x);  clear;
  x:=1/x;     clear;
  write(' (1)  ',x:m+6);
  x:=1/sqr(sqr(3+sqrt(10)));
  writeln('    real  : ',x);
  writeln;
  writeln(' Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)');
  writeln;
  writeln('     exakt :  6.93481609512304252272...E-04');
  readln
end.
```

Hier sind die Ergebnisse zur numerischen Äquivalenz von Termen mit obigen Turbo-Pascal-Programm (TP-Programm) im Vergleich der GPF *single* und *real*.

```
Aequivalenz von Termen
Byteanzahl der GKZ/Laenge der Mantisse/gueltige Dez.stellen
  n = 3 / 2 / 4
      4 / 3 / 7   Format single
      5 / 4 / 9
      6 / 5 / 11  Format real
  n = 4
 (1)   6.934818E-04      real  :  6.9348160951E-04
 (2)   6.934781E-04      real  :  6.9348160952E-04
 (3)   6.936202E-04      real  :  6.9348160973E-04
 (4)   7.934570E-04      real  :  6.9348141551E-04
 (5)   6.934817E-04      real  :  6.9348160951E-04
 (6)   6.934816E-04      real  :  6.9348160951E-04
 Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)
      exakt :  6.93481609512304252272...E-04
```

Mit wachsender Mantisse können auch mit den problematischen Formeln (2), (3), (4) bessere Ergebnisse erzielt werden, wie bei TP mit GPF *double* und *extended*.

Führen wir mit einem weiteren TP-Programm analoge Rechnungen mit den GPF *single*, *real*, *double* oder *extended* bei den Compilerdirektiven N+, E+ durch.

```
{$N+,E+}
program Equivalent_1;
{ Aequivalenz von Quadratwurzel-Bruch-Termen }
{ equiv_1.pas}

uses crt;
type float=real; (* single (7 MSt.), real (11 MSt.),
                    double (15 MSt.), extended (18 MSt.)
                    bei Compilerdirektiven N+,E+         *)
var  x:float;
begin
 clrscr;
 writeln('Aequivalenz von Termen in TP7.0/BP7.0');
 writeln('Format real = 6Byte => 11stellige Mantisse');
 writeln;
 x:=1/sqr(sqr(3+sqrt(10)));
 writeln(' (1) : ',x);
 x:=sqr(sqr(3-sqrt(10)));
 writeln(' (2) : ',x);
 x:=sqr(19-6*sqrt(10));
 writeln(' (3) : ',x);
```

```
 x:=721-228*sqrt(10);
 writeln(' (4) : ',x);
 x:=1/sqr(19+6*sqrt(10));
 writeln(' (5) : ',x);
 x:=1/(721+228*sqrt(10));
 writeln(' (6) : ',x);
 writeln;
 writeln(' Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)');
 (* writeln(' Guete : (6) >= (5) >= (1) >= (2) > (3) >> (4)'); *)
 writeln;
 writeln('     exakt :  6.9348160951230425272E-04');
 readln
end.
```

Dabei ist jedoch folgendes zu beachten. Arithmetische Ausdrücke werden bei ihrer Berechnung mit der Stellenzahl des *extended*-Formats behandelt (Rechnung jedoch nicht unbedingt in dieser Genauigkeit) und erst durch Zuordnung eines Wertes zu einer Variablen kommt es zur Stellenabschneidung.

```
Aequivalenz von Termen in TP7.0/BP7.0

Format single = 4Byte => 7stellige Mantisse
 (1) :  6.93481590133160E-0004
 (2) :  6.93481590133160E-0004
 alle gleich

-----------------------------------------------------
Format real = 6Byte => 11stellige Mantisse
 (1) :  6.93481609512325E-0004
 (2) :  6.93481609512325E-0004
 alle gleich

-----------------------------------------------------
Format double = 8Byte => 15stellige Mantisse
 (1) :  6.93481609512304E-0004
 (2) :  6.93481609512304E-0004
 (3) :  6.93481609512304E-0004
 (4) :  6.93481609512325E-0004
 (5) :  6.93481609512304E-0004
 (6) :  6.93481609512304E-0004

-----------------------------------------------------
```

```
Format extended = 10Byte => 18stellige Mantisse
                  (Ausgabeformat maximal vergroessert)

 (1) :  6.93481609512304252E-0004
 (2) :  6.93481609512304252E-0004
 (3) :  6.93481609512304262E-0004
 (4) :  6.93481609512325292E-0004
 (5) :  6.93481609512304252E-0004
 (6) :  6.93481609512304252E-0004

 Guete : (6) >= (5) >= (1) >= (2) > (3) >> (4)
      exakt :  6.934816095123042522 72...E-04
```

Man sieht, dass im Format *double* die Rechnungen wie mit dem Format *extended* durchgeführt werden und erst bei der Ausgabe das Abschneiden der Mantisse erfolgt.
Zum Vergleich sei noch eine Rechnung mit Pascal-XSC mit dem GPF *real*, angegeben, das in Wirklichkeit dem *double*-Format entspricht.

```
Aequivalenz von Termen in Pascal-XSC
Format real = 8Byte => 15-16stellige Mantisse

 (1) :  6.934816095123040E-004
 (2) :  6.934816095123075E-004
 (3) :  6.934816095122907E-004
 (4) :  6.934816094599228E-004
 (5) :  6.934816095123043E-004
 (6) :  6.934816095123043E-004

 Guete : (6) >= (5) >= (1) > (2) > (3) >> (4)
      exakt :  6.934816095123042522 72...E-04
```

Bei komplizierten Formeln müsste man also auch "zwischendurch" immer wieder die Mantisse kürzen (siehe TP-Programm `Aequivalenz`).
Es ist also ein Unterschied zwischen den ausgegebenen Größen bei folgender Anweisungsfolge

```
var x:single;
...
x:=721-228*sqrt(10);
writeln(x);                    { --> 6.93481590.........E-004 }

writeln(721-228*sqrt(10):26);  { --> 6.93481609512325292E-004 }
...
```

Rechnungen in Maple (Datei *aequiv2.mws*)

Termumformung und mathematische Äquivalenz von Formeln

```
> Digits:=30;
  t1 := 1/(3+sqrt(10))^4;
  evalf(t1);
```

$$Digits := 30$$

$$t1 := \frac{1}{(3+\sqrt{10})^4}$$

$$0.00069348160951230425227186934017 6$$

```
> Digits := 7:
  t1 := 1/(3+sqrt(10.0))^4;
  t2 := (3-sqrt(10.0))^4;
  t3 := (19-6*sqrt(10.0))^2;
  t4 := 721-228*sqrt(10.0);
  t5 := 1/(19+6*sqrt(10.0))^2;
  t6 := 1/(721+228*sqrt(10.0));
```

$$t1 := 0.0006934815$$
$$t2 := 0.0006934874$$
$$t3 := 0.0006932689$$
$$t4 := 0.0006$$
$$t5 := 0.0006934815$$
$$t6 := 0.0006934818$$

```
> i:='i':
  n:=17:
  printf(‘Tabelle mit Genauigkeiten  Digits=1..17,
          keine 2 Spalten sind gleich‘):
  printf(‘ ‘):
  printf(‘  i             t1                    t2                    t3‘||
   ‘                   t4                    t5                    t6\n‘):

  for i from 1 to n do
   Digits:=i:
   t1 := 1/(3+sqrt(10.0))^4;
   t2 := (3-sqrt(10.0))^4;
   t3 := (19-6*sqrt(10.0))^2;
   t4 := 721-228*sqrt(10.0);
   t5 := 1/(19+6*sqrt(10.0))^2;
   t6 := 1/(721+228*sqrt(10.0));
   printf(‘ %2d   %23.20f %23.20f %23.20f %24.20f %23.20f %23.20f\n‘,
          i,t1,t2,t3,t4,t5,t6):
  end do:

  Digits:=30:
  evalf(1/(3+sqrt(10))^4);
  Digits:=10:
```

Tabelle mit Genauigkeiten `Digits=i=1..17`, keine 2 Spalten sind gleich.
Das Format der Zahlendarstellung wurde wegen Platzeinsparung verändert.

```
 i         t1                  t2                  t3                  t4                  t5                  t6

 1 8.0000000000000000E-4 0.0000000000000000E+0 0.0000000000000000E+0  1.0000000000000000E+2 6.0000000000000000E-4 1.0000000000000000E-3
 2 6.8000000000000000E-4 1.6000000000000000E-3 0.0000000000000000E+0 -2.0000000000000000E+1 6.9000000000000000E-4 6.7000000000000000E-4
 3 6.9400000000000000E-4 6.5500000000000000E-4 0.0000000000000000E+0  1.0000000000000000E+0 6.9300000000000000E-4 6.9400000000000000E-4
 4 6.9360000000000000E-4 6.8870000000000000E-4 9.0000000000000000E-4  1.0000000000000000E-1 6.9360000000000000E-4 6.9350000000000000E-4
 5 6.9347000000000000E-4 6.9386000000000000E-4 6.7600000000000000E-4  0.0000000000000000E+0 6.9347000000000000E-4 6.9348000000000000E-4
 6 6.9348100000000000E-4 6.9352200000000000E-4 6.9169000000000000E-4  0.0000000000000000E+0 6.9348000000000000E-4 6.9348100000000000E-4
 7 6.9348150000000000E-4 6.9348740000000000E-4 6.9326890000000000E-4  6.0000000000000000E-4 6.9348150000000000E-4 6.9348180000000000E-4
 8 6.9348159000000000E-4 6.9348229000000000E-4 6.9347956000000000E-4  6.8000000000000000E-4 6.9348161000000000E-4 6.9348161000000000E-4
 9 6.9348161000000000E-4 6.9348160700000000E-4 6.9347955600000000E-4  6.9400000000000000E-4 6.9348160800000000E-4 6.9348160800000000E-4
10 6.9348160960000000E-4 6.9348160660000000E-4 6.9348166270000000E-4  6.9350000000000000E-4 6.9348160960000000E-4 6.9348160980000000E-4
11 6.9348160950000000E-4 6.9348161005000000E-4 6.9348161005000000E-4  6.9347000000000000E-4 6.9348160951000000E-4 6.9348160952000000E-4
12 6.9348160951200000E-4 6.9348160954000000E-4 6.9348161005400000E-4  6.9348100000000000E-4 6.9348160951300000E-4 6.9348160951200000E-4
13 6.9348160951250000E-4 6.9348160950580000E-4 6.9348160952680000E-4  6.9348170000000000E-4 6.9348160951230000E-4 6.9348160951250000E-4
14 6.9348160951229000E-4 6.9348160951266000E-4 6.9348160952684000E-4  6.9348160000000000E-4 6.9348160951231000E-4 6.9348160951230000E-4
15 6.9348160951230400E-4 6.9348160951231600E-4 6.9348160951104000E-4  6.9348160900000000E-4 6.9348160951230300E-4 6.9348160951230400E-4
16 6.9348160951230440E-4 6.9348160951229860E-4 6.9348160951261980E-4  6.9348160960000000E-4 6.9348160951230450E-4 6.9348160951230450E-4
17 6.9348160951230427E-4 6.9348160951230371E-4 6.9348160951230383E-4  6.9348160952000000E-4 6.9348160951230425E-4 6.9348160951230425E-4
```

6.93481609512304252271869340176E-4

Rechnungen in MATLAB

Termumformung und mathematische Äquivalenz von Formeln (Datei *aequiv2.m*)

```
Varianten (1)-(6)
 x = 6.934816095123039e-004
 x = 6.934816095123075e-004
 x = 6.934816095122907e-004
 x = 6.934816094599228e-004
 x = 6.934816095123043e-004
 x = 6.934816095123043e-004

 Guete : (6) >= (5) > (1) > (2) > (3) >> (4)

 exakt :  6.93481609512304252272...E-04
```

MATLAB hat fast genau das Ergebnis wie Pascal-XSC.

MATLAB ist qualitativ vergleichbar mit Maple bei `Digits=16`. Die quantitativen Unterschiede rühren eher von den Maple-Schwächen bei der Berechnung der Quadratwurzel her.

1.4 Auswertung von Funktionen und Kommandos

(1) Maple-Kommando `linsolve` zur Lösung von Gleichungssystemen

Wir verwenden das Kommando für ein linearen Gleichungssystemen (LGS) mit der Hilbert-Matrix H als Koeffizientenmatrix. Diese ist zwar symmetrisch und positiv definit, aber schlecht konditioniert. So genügt es, ein kleines System zu betrachten. Wenn die Rechengenauigkeit `Digits=...` eingestellt ist und die Kondition $\text{cond}(H) = \mathcal{O}(10^t)$ beträgt, dann muss man in der Lösung des LGS einen Genauigkeitsverlust von ca. t Mantissenstellen einkalkulieren.
Maple 8, 9.5 liefert dabei rund eine Dezimale mehr Genauigkeit als Maple V.

(2) Maple-Kommandos `norm` und `cond` aus dem Paket `linalg`

Diese beiden Kommandos waren in Maple V eine Schwachstelle und sind in der neueren Version korrigiert worden. Zum Test kann man die symmetrische und positiv definite Hilbert-Matrix verwenden [16].

(3) Lange Listen und Folgen

Man kann in Maple mit beliebig langen Listen arbeiten. Aber die Indizierung der Listenelemente, wie dies bei Vektorkomponenten üblich ist, erfährt an der Grenze 100 andere Eigenschaften.
Die Abfrage von Listenelementen, auch mit einem Index größer als 100, ist zwar möglich. Aber eine Belegung/Zuordnung zu beliebigen Listenelementen - nicht nur im "oberen" Indexbereich - mittels Ergibtanweisung führt zu einer Fehlermeldung mit dem Hinweis, auf Arrays auszuweichen [16].

(4) Felder

Es gibt verschiedene Möglichkeiten der Definition von Feldern in Maple.
Dabei erweisen sich die zahlreichen Kommandos/Funktionen in den Paketen `linalg` und `LinearAlgebra`, speziell zur linearen Algebra, als sehr nützlich.
Ein- und mehrdimensionale Felder werden mit dem Kommando `array` erzeugt.
Matrizen `matrix` und Vektoren `vector` sind spezielle Felder.
Der Index ist aus einem festem Integer-Bereich zu wählen und es erfolgt eine Kontrolle der Zulässigkeit der Indizes.
Ein Feld kann dann initialisiert werden, entweder mit einer Liste von Daten oder ohne.

Man beachte, dass der Wert eines Feldes A der Bezeichner des Feldes A selbst ist, weil auch oft die Feldkomponenten noch keine konkreten Werte haben. Es wird also mit Adressen gearbeitet und es werden Adressen übergeben.
Es gibt spezielle Kommandos zur Berechnung/Verarbeitung von Feldern, wo es um ihren Inhalt, also um die Feldkomponenten geht. Wenn man Zuweisungen, Ausgaben oder Prozedurrückgabe von Feldern machen will, dann benutze man das Kommando `print` oder eine der Berechnungsfunktionen `eval, evalm, evalf, ...`.

Dabei wird unterschieden, ob das Argument ein `array` oder eine `matrix` bzw. `vector` ist. Insbesondere, wenn ein `array` die unteren Indexgrenzen 1 hat, dann ist es auch eine `matrix`. Diese Doppeldeutung hat Auswirkung auf die Verwendung der Berechnungsfunktionen `eval, evalm`.

Aus Datei *genarr2.mws*

```
> A:=hilbert(3);       # A:=linalg[hilbert](3);  #  nicht hilbert(m,n)
  whattype(hilbert), whattype(A);
  type(A,matrix), type(A,array);
  A;
  evalm(A);            # wie eval(A) bzw. print(A)
  evalf(A);
  evalf(evalm(A));     # Digits=10
```

$$A := \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

$$symbol, symbol$$

$$true, true$$

$$A$$

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

$$A$$

$$\begin{bmatrix} 1. & 0.5000000000 & 0.3333333333 \\ 0.5000000000 & 0.3333333333 & 0.2500000000 \\ 0.3333333333 & 0.2500000000 & 0.2000000000 \end{bmatrix}$$

Traditionelle mathematische Notation mit Bezeichner und Gleichheitszeichen

```
> A:
  %=evalm(%);
```

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

Will man Matrizen und Vektoren umspeichern und unter anderen Namen verwenden, dann benutzt man das Kommando `evalm`.
Dabei sollte man lieber einmal mehr als zu wenig dieses Kommando notieren.

Zahlreiche Möglichkeiten der Generierung von Matrizen und Vektoren, verbunden auch mit praktischen Anwendungsbereichen, sind in [16] aufgezeigt.

(5) LU-Faktorisierung einer quadratischen Matrix

In der linearen Algebra treten sehr häufig unterschiedliche Faktorisierungen von Matrizen auf, die LU-Faktorisierung (auch LU-Zerlegung genannt) ist eine davon. Sie ist direkt verbunden mit dem Gaußschen Eliminationsverfahren (Gauß-Algorithmus, GA) zur Lösung von LGS.

Betrachten wir die Realisierung der **vollständigen LU-Faktorisierung** ohne Pivotisierung von
$A = A(n, n) = (a_{ij})$ in eine untere bzw. obere Dreiecksmatrix gemäß $A = LU$.

Die Durchfürbarkeit ohne Pivotisierung entspricht der Diagonalstrategie und ist möglich, wenn in allen Schritten das Diagonalelement nicht Null wird. Natürlich ist es günstiger eine andere Pivotstrategie zu verfolgen, denn auch Divisionen durch betragskleine Werte beeinträchtigen die numerische Stabilität der Faktorisierung.
Als erste Verbesserung betrachten wir nur die Spaltenpivotisierung mit Zeilenvertauschung. Dabei gibt es einige kleine Modifikationen der Wahl des sogenannten Pivotelements in der jeweiligen Spalte. Zum Pivotelement gehört seine Pivotzeile.

- Wenn ein Diagonalelement verschwindet, dann sucht man darunter in derselben Spalte
  – das erste nicht verschwindende Element oder
  – das betragsgrößte Element
  und vertauscht diese Zeile mit der Pivotzeile.

- Generell sucht man von Anfang an in der ersten Spalte eines jeden Teiltableaus das betragsgrößte Element und tauscht gegebenfalls die zugehörige Zeile mit der Diagonalzeile.

Dabei merkt man sich die Zeilenvertauschungen in einem Permutationsvektor
$p = (p_1, p_2, ..., p_n)$, dessen Initialisierung $p = (1, 2, ..., n)$ ist.
Damit ergeben sich die Einträge in der orthogonalen Permutationsmatrix P.
p_i bedeutet in der i-ten Zeile und p_i-ten Spalte eine Eins, sonst Nullen in der Zeile.
Praktisch heißt das die Bildung der zeilenvertauschten Matrix $\widetilde{A} = PA$ und ihre Faktorisierung $\widetilde{A} = PA = LU$.
So kann man auch die Formel $A = P^T LU = \widetilde{L}U,\ \widetilde{L} = P^T L$, notieren.

Bei der LU-Faktorisierung mit den CAS Maple und MATLAB sollte man auf die "feinen" Unterschiede und die Möglichkeiten der Ergebnisdarstellung achten.
Dabei soll vorausgeschickt werden, das die CAS bezüglich der dazu verwendeten Algorithmen und Strategien einem ständigen Wandel unterliegen. Das heißt nicht unbedingt, dass die neuere Version in jedem Fall besser ist, angefangen von der Verständlichkeit hin bis zu algorithmischen Aspekten und letztendlich auch bei der Lösungsdarstellung.
Manchmal wird der Nutzer von der Implementation nicht nur in Erstaunen versetzt, sondern fast schockiert.

Zunächst das Maple-Kommando `LUdecomp` im Paket `linalg`.
In der Online-Hilfe wird dazu eine relativ diffuse Erläuterung gegeben. Genauer ist:

```
- Symbolische Rechnung
  Falls alle Pivotlemente nicht Null sind, dann keine Pivotstrategie und
  A=LU, P=I.
  Falls Pivotelement=0, wird in der Spalte das naechste NNE gesucht und
  die beiden Zeilen vertauscht, dabei Bildung der Permutationsmatrix P,
  so dass A=PLU.
- Numerische Rechnung
  Generell Anwendung der Spaltenpivotstrategie mit Suche des
  betragsgroessten Elements in der Spalte und dann gegebenenfalls
  Zeilenvertauschung.
```

Bei der symmetrischen positiv definiten (spd) Hilbert-Matrix verläuft die symbolische Rechnung der LU-Faktorisierung verständlicherweise ohne Pivotstrategie.

Aus Datei *lu2.mws*

```
> A:=hilbert(3);
```

$$A := \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{bmatrix}$$

Kurzform

```
> LUdecomp(A); # Ergebnis ist U
```

$$\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & \frac{1}{12} & \frac{1}{12} \\ 0 & 0 & \frac{1}{180} \end{bmatrix}$$

Langform

```
> LUdecomp(A,L='l',U='u',P='p'):  # Ergebnisse sind U,L,P
  U:=evalm(u);
  L:=evalm(l);
  P:=evalm(p):   # P=I
```

$$U := \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & \frac{1}{12} & \frac{1}{12} \\ 0 & 0 & \frac{1}{180} \end{bmatrix}$$

$$L := \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{3} & 1 & 1 \end{bmatrix}$$

Bei der numerische Rechnung (GP-Zahlen) mit Pivotstrategie passiert folgendes.

Wenn wir den ersten Zwischenschritt bei der LU-Faktorisierung mit dem Resttableau nehmen, so erkennen wir in der 2. Spalte an der Diagonalstelle und eine Position darunter die gleichen Werte $\frac{1}{12}$. Bei numerischen Rechnungen stehen dort die GP-Zahlen der Berechnungen, die bei Berücksichtigung von zusätzlichen Rundungen in der GPA nahe dem exakten Wert 0.083333333333333... liegen. Aber es kann dann natürlich passieren, dass der numerische Wert in der 3. Zeile größer ist, und somit eine Vertauschung von 2. und 3. Zeile stattfindet.
Damit gibt es ein abweichendes Verhalten von der symbolischen Rechnung und das Ergebnis ist die dann notierte Faktorisierung mit der Zeilenvertauschung

$$A = P_1 L_1 U_1 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{1}{2} & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ 0 & \frac{1}{12} & \frac{4}{45} \\ 0 & 0 & \frac{-1}{180} \end{pmatrix}.$$

An dieser Stelle machen wir Vergleichsrechnungen mit Maple V und den 2 Varianten von Maple 8 bzw. 9.5.

Der "alte" numerische Algorithmus in `LUdecomp` von Maple V wurde zwar in Maple 7/8/9.5 ausgetauscht (Ersetzung durch eine bessere FORTRAN-Routine), aber es ist schon eigenartig, dass z. B. in Maple 8 die Matrixdefinitionen `A:=hilbert(3)` und `A:=evalm(hilbert(3))` zu unterschiedlicher Behandlung führen.

Maple V

```
> # A:=hilbert(3):     oder
  A:=matrix(3,3,[[1,  1/2,1/3],
                 [1/2,1/3,1/4],
                 [1/3,1/4,1/5]] );
> A[1,1]:=1.0:  # numerische Rechnung
> Digits:=10:

> LUdecomp(A,L='l',U='u',P='p');  # Ausgabe von U
  U:=evalm(u);
  L:=evalm(l);
  P:=evalm(p);
```

Maple 8, D1

```
> # 1. Definition
  A:=hilbert(3):
> A[1,1]:=1.0:
> Digits:=10:

> LUdecomp(A,L='l',U='u',P='p');  # Ausgabe von U
  U:=evalm(u);
  L:=evalm(l);
  P:=evalm(p);
```

Maple 8, D2

```
> # 2. Definition
  A:=evalm(hilbert(3)):
  # oder
  # A:=matrix(3,3,[[1,1/2,1/3],[1/2,1/3,1/4],[1/3,1/4,1/5]]);
> A[1,1]:=1.0:
> Digits:=10:

> LUdecomp(A,L='l',U='u',P='p');  # Ausgabe von U
  U:=evalm(u);
  L:=evalm(l);
  P:=evalm(p);
```

Maple 9.5, D1/2

```
> # 1. bzw. 2. Definition mit gleicher Behandlung
  A:=hilbert(3):   # bzw. A:=evalm(hilbert(3)): oder
  # A:=matrix(3,3,[[1,1/2,1/3],[1/2,1/3,1/4],[1/3,1/4,1/5]]);
> A[1,1]:=1.0:
> Digits:=10:

> LUdecomp(A,L='l',U='u',P='p');  # Ausgabe von U
  U:=evalm(u);
  L:=evalm(l);
  P:=evalm(p);
```

Die Ergebnisse sind in der nachfolgenden Tabelle eingetragen und verlangen einen Kommentar.

Bemerkungen

– Man sieht, dass sich Maple V und Maple 8, 9.5 in der Pivotstrategie nach dem 1. Zwischenschritt anders verhalten. Wie ist das bei anderen Genauigkeiten?
– Erkennbar sind Unterschiede in den letzten Dezimalen der Elemente der Matrizen.
– Unverständlich ist die Ausgabe der oberen Dreiecksmatrix U bei `LUdecomp` in Maple 8 in der 2. Matrixdefinition `A:=evalm(hilbert(3))` bzw. Maple 9.5. bei beiden Varianten.
– Warum stehen solche langen Dezimalzahlen, obwohl wegen der eingestellten Genauigkeit nur die ersten Stellen nach dem Punkt richtig sind und man die weiteren Stellen gar nicht verwerten kann?
– Zusätzlich stellt sich die Frage, wie die Darstellung bei anderen Genauigkeiten aussieht. Man sollte auf alles gefasst sein.

Die Zeilenvertauschung ist bei den einzelnen Versionen nicht das eigentliche Problem, auch wenn sie von der Computerrechnung dem Nutzer einfach aufgezwungen wird.

Auffälliger ist, welche Auswirkung die Genauigkeitseinstellung `Digits=...` hat. Diese führt in manchen Situationen dazu, dass auch alle Zwischenrechnungen auf die gültige Stellenanzahl reduziert/gerundet werden. Bei jüngeren Maple-Versionen scheint es so, dass in den Rechnungen eine zusätzliche Stelle mitgeführt wird und das letzte Ergebnis dann in der vorgegebenen Stellenanzahl erscheint.
Detaillierte Ausführungen findet man in [16].

	Maple V	Maple 8, D1
U	$\begin{bmatrix} 1.0 & .5000000000 & .3333333333 \\ 0 & .0833333333 & .0833333333 \\ 0 & 0 & .00555555560 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.5000000000 & 0.3333333333 \\ 0 & 0.0833333334 & 0.0888888889 \\ 0 & 0 & -0.00555555539 \end{bmatrix}$
U	$\begin{bmatrix} 1.0 & .5000000000 & .3333333333 \\ 0 & .0833333333 & .0833333333 \\ 0 & 0 & .00555555560 \end{bmatrix}$	$\begin{bmatrix} 1.0 & 0.5000000000 & 0.3333333333 \\ 0 & 0.0833333334 & 0.0888888889 \\ 0 & 0 & -0.00555555539 \end{bmatrix}$
L	$\begin{bmatrix} 1 & 0 & 0 \\ .5000000000 & 1 & 0 \\ .3333333333 & 1.000000000 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0.3333333333 & 1 & 0 \\ 0.5000000000 & 0.9999999988 & 1 \end{bmatrix}$
P	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
		Maple 8, D2 bzw. Maple 9.5, D1/2
U		$\begin{bmatrix} 1. & 0.500000000000000000 & 0.333333333299999979 \\ 0. & 0.0833333333500000107 & 0.0888888889111111304 \\ 0. & 0. & -0.00555555550777775410 \end{bmatrix}$
U		$\begin{bmatrix} 1. & 0.5000000000 & 0.3333333333 \\ 0. & 0.08333333335 & 0.08888888891 \\ 0. & 0. & -0.005555555508 \end{bmatrix}$
L		$\begin{bmatrix} 1. & 0. & 0. \\ 0.3333333333 & 1. & 0. \\ 0.5000000000 & 0.9999999994 & 1. \end{bmatrix}$
P		$\begin{bmatrix} 1. & 0. & 0. \\ 0. & 0. & 1. \\ 0. & 1. & 0. \end{bmatrix}$

Tab. 1.1 Ergebnisse der 5 Maple-Versionen bezüglich LU-Faktorisierung in der GPA `Digits=10`

Nun zum MATLAB-Kommando `lu`.

Die LU-Faktorisierung ohne Zeilenvertauschung liefert $A = LU$ mit den entsprechenden Dreiecksmatrizen L, $l_{ii} = 1$, und U. Zeilenvertauschungen werden entweder in die untere Dreiecksmatrix L einbezogen (psychologically lower triangular matrix) oder als Permutationsmatrix in Form eines zusätzlichen Ergebnisparameters angegeben.

`[L,U] = lu(A)`	$A = LU$,	in L stehen die untere Dreiecksmatrix und Permutationsmatrix, U obere Dreiecksmatrix
`[L,U,P] = lu(A)`	$PA = LU$,	L, U untere bzw. obere Dreiecksmatrix, P Permutationsmatrix

Wir erinnern uns, die Faktorisierung mit dem Maple-Kommando `LUdecomp` ergab bei Pivotisierung $A = PLU$.

`lu(A)` alleine ergibt ein Tableau (eine Ergebnismatrix) $A1$ mit 2 Dreiecksmatrizen.

Werden keine Zeilenvertauschungen vorgenommen, so gilt folgendes.
In der älteren Version MATLAB 4.2 hat $A1$ die Form $C\backslash B$, welche beim verketteten Gauß-Algorithmus (VGA) entsteht, so dass $A = -CB = LU$ ist (output from LINPACK'S ZGEFA routine).
Es ist dann

```
U = triu(lu(A))
L = eye(n)-tril(lu(A),-1)  % n ist Dimension
```

In den neuen Versionen Matlab 6.$*$ erhält man eine Tableau mit L und U (output from LAPACK'S DGETRF or ZGETRF routine), so dass

```
U = triu(lu(A))
L = eye(n)+tril(lu(A),-1)
```

folgen.

Aber die LU-Faktorisierung verwendet generell eine Pivotstrategie, und zwar die Spaltenpivotisierung mit Zeilenvertauschung. Dabei wird in der jeweiligen Spalte nach dem betragsgrößten Element gesucht.
Die Zeilenvertauschung ist aus der Kurzform `lu(A)` nicht ohne Weiteres erkennbar. Das obere Dreieck von $A1$ mit Diagonale bildet die Matrix U, die Zeilen von L ergeben sich aus den Zeilen unterhalb der Diagonalen von $A1$, aber eventuell in getauschter Reihenfolge, sowie mit $l_{ii} = 1$. Deshalb ist die Verwendung der Kommandos `[L,U] = lu(A)` oder `[L,U,P] = lu(A)` sinnvoll.

Betrachten wir wieder die spd Hilbert-Matrix `A=hilb(3)`. Die symbolische Rechnung und Faktorisierung in Maple läuft auch ohne Zeilenvertauschung ab und ergibt die erwartete Lösung.

```
A  = [1    0  0                  # Maple
      1/2  1  0
      1/3  1  1]*[1  1/2   1/3
                     0  1/12  1/12
                     0  0     1/180]
```

Das MATLAB-Kommando `lu` hingegen führt im 2. Schritt eine Vertauschung von 2. und 3. Zeile durch.

Aus Datei *luf2.m*

```
% MATLAB 4.2
% Tableau wie beim verketteten Gauss-Algorithmus, A=-CB

format rat      % display rational approximations
lu(A)
ans =
       1      1/2      1/3
    -1/2      1/12     4/45
    -1/3     -1       -1/180
```

Daraus ergibt sich die obere Dreiecksmatrix

$$U = \begin{pmatrix} 1 & 1/2 & 1/3 \\ 0 & 1/12 & 4/45 \\ 0 & 0 & -1/180 \end{pmatrix}.$$

Aber bezüglich der unteren Dreiecksmatrix ist daraus keine verwertbare Information zu entnehmen, denn die möglichen Zeilen für L würden $(1,0,0)$, $(1/2,1,0)$, $(1/3,1,1)$ sein, und das ist falsch.

MATLAB 6.$*$ und MATLAB R2006a haben diese irritierende Darstellung durch die Verwendung des Resttableau-Algorithmus korrigiert.

```
% MATLAB 6.*, MATLAB R2006a
% Tableau wie beim Resttableau-Algorithmus, A=LU
format rat
lu(A)
ans =
       1      1/2      1/3
      1/3     1/12     4/45
      1/2      1      -1/180
```

So sind zumindest die Zeilen der Matrix L als $(1,0,0)$, $(1/3,1,0)$, $(1/2,1,1)$ ablesbar, auch wenn wegen Zeilenvertauschungen ihre Reihenfolge noch nicht klar ist.

Die Pivotstrategie bewirkt, dass im 2. Schritt beim Vergleich von 1/12=0.08333... in der 2. Zeile mit 1/12 darunter die 3. Zeile bevorzugt wird.

Langformen des Kommandos zur Kontrolle mit der Permutationsmatrix

```
[L,U,P] = lu(A)
L =
      1      0      0
     1/3     1      0
     1/2     1      1
U =
      1     1/2    1/3
      0     1/12   4/45
      0      0    -1/180
P =
      1      0      0
      0      0      1
      0      1      0
P*A-L*U;  % 0

[Ls,U] = lu(A)  % Zeilenvertauschung in L zu erkennen
Ls =
      1      0      0
     1/2     1      1
     1/3     1      0
U =
      1     1/2    1/3
      0     1/12   4/45
      0      0    -1/180

format long
[L,U,P] = lu(A)
L =
    1.00000000000000                   0                   0
    0.33333333333333   1.00000000000000                   0
    0.50000000000000   1.00000000000000   1.00000000000000
U =
    1.00000000000000   0.50000000000000   0.33333333333333
                   0   0.08333333333333   0.08888888888889
                   0                   0  -0.00555555555556
P =
      1      0      0
      0      0      1
      0      1      0
P*A-L*U;  % 0
```

Kapitel 2

Verarbeitung und Varianten zum Anhalten der Ausgabe von Daten

2.1 Vektoren und Matrizen

Es wurde schon darauf hingewiesen, dass man in Maple mit beliebig langen Listen arbeiten kann, aber die Indizierung der Listenelemente, wie dies bei Vektorkomponenten üblich ist, an der Grenze 100 andere Eigenschaften erfährt.

Also nutzt man für die Speicherung großer Datenmengen zunächst Felder (Vektoren, Matrizen, ...), im Weiteren natürlich auch ihre "Auslagerung" auf Speichermedien in Form von Dateien.

Maple will sich in der Benutzung von Vektoren als Spalten- oder Zeilenvektoren nicht festlegen, sondern sich nach Bedarf und Stellung eines Vektors in der Formel entscheiden. Das hat zur Folge, dass eigentlich Verwirrungen vorprogrammiert sind. MATLAB ist diesbezüglich konsequenter.

Wir fassen wichtige Erkenntnisse von unterschiedlichen Implementationen in Maple zusammen (Rechnungen in Datei *matvek1.mws*, [16]).

1. So wird das Kommando des Vektortransposition `transpose(x)` nicht explizit ausgeführt.

2. Im Matrix-Vektor-Produkt `A&*x` ist x ein Spaltenvektor, während x in `x&*A` als Zeilenvektor genommen wird. Stehen jedoch nur Vektoren in der Formel, wie beim Skalarprodukt `transpose(x)&*x` oder dyadischen Produkt `x&*transpose(x)`, dann wird x als Spaltenvektor interpretiert. Also ist die Deutung des links stehenden Vektors x in den Produkten `x&*A` und `x&*transpose(x)` verschieden.

3. Eine Mischung der Situationen von Skalar- und Matrix-Vektor-Produkt ergibt sich bei der Betrachtung der quadratischen Form $x^T Ax$.
   Wegen `x&*A` und `A&*x` ist einerseits die Darstellung `x&*A&*x` möglich.

Aber genauso kann man wegen `A&*x` und `transpose(x)&*x` den Ausdruck `transpose(x)&*A&*x` notieren.

4. Sobald in größeren Formeln Ausdrücke mit Feldern stehen, z. B. Differenzen von Vektoren oder Matrizen, ist zumeist die Einbeziehung der Berechnungsfunktion `evalm` zu empfehlen, manchmal ist es zur Vermeidung von syntaktischen Fehlern sogar notwendig.

5. Bei Differenzen von Matrizen haben wir die Situation, dass `A-A` nicht die Nullmatrix ist, sondern einfach der Wert 0, so dass dann das Ergebnis auch keine Dimensionsabfragen mit `rowdim( )` bzw. `coldim( )` erlaubt.
   Ist A jedoch inhaltlich mit der Einheitsmatrix I identisch, so ist die Differenz `A-I` eine Nullmatrix mit gegebener Dimension.

6. Bei Potenzen von Matrizen haben wir die Situation, dass `A^0` nicht die Einheitsmatrix ist, sondern einfach der Wert 1, so dass dann das Ergebnis auch keine Dimensionsabfragen mit `rowdim( )` bzw. `coldim( )` erlaubt.
   Bildet man jedoch `A^0-I` mit der Einheitsmatrix I, so erinnert sich Maple daran, dass A^0 eigentlich mehr als die Eins darstellt und berechnet richtig als Differenz die Nullmatrix mit gegebener Dimension.

2.2 Erzeugung und Behandlung von Feldern

Es gibt viele Varianten der Eingabe, Generierung und Manipulation von Matrizen und Vektoren in Maple und MATLAB. Dabei erweisen sich die zahlreichen Kommandos/Funktionen speziell zur linearen Algebra, als sehr nützlich.

Ausgewählte Möglichkeiten
– Explizite Angabe
– Mit Funktionsdefinition
– Spezielle Matrizen wie band, diag, rand, hilbert, toeplitz, ...
– Matrizen mit Bildungsfunktion/Optionen
– Einheitsmatrix
– Blockdiagonalmatrix
– Elementweise Eingabe einer Matrix
– Elementweise Definition einer Matrix mit Doppelschleife
– Matrixelemente aus Gleichungen
– Aneinandersetzen/Zusammenfügen von Matrizen und Vektoren, Blockstrukturen
– ...

Implementationen in Maple findet man z. B. in der Datei *genarr2.mws*.

2.3 Varianten zum Anhalten der Datenausgabe

In MATLAB steht das einfache Kommando `pause` zur Verfügung, um allgemein die Abarbeitung der Befehlsfolge eines m-Files anzuhalten (Fortsetzung mit Tastenbetätigung) oder speziell im Rahmen von Schleifen die Erstellung und Ausgabe von Ergebnissen/Daten bzw. Dateien zu unterbrechen.

In Pascal erzielt man analoge Effekte mit den Anweisungen `readkey` oder `readln, readln(...)`.

Betrachten wir eine TP-Prozedur, die das Extrapolationsverfahren zur numerischen Integration, nämlich das Romberg-Verfahren, umsetzt [34]. Das Romberg-Schema bedeutet die zeilenweise Berechnung eines zweidimensionalen Tableaus, und dies als unteres linkes Dreieck. Die Anzahl der Zeilen kann man entweder vorgeben oder beschränken und dabei noch ein weiteres Abbruchkriterium hinzunehmen. Das Tableau könnte man als Matrix (zusätzlicher Prozedurparameter) speichern.

```
procedure Romberg2(zw:boolean;a,b,eps:float;var IWert:float;var iter:integer);  { var T:feld }
{ Romberg-Schema Spalte 0,1,..,k,..  bis Toleranz eps erreicht  oder  k>=mmax,
   Vektor p[k..0] ist letzte Zeile T[k,0],T[k,1],...,T[k,k] des Tableaus  T[0..k,0..k]  }

const mmax=50;
var   h,ha,ah,s,q:float;
      i,k,n:integer;
      p:array[0..mmax] of float;
begin
 h:=b-a;
 n:=1;
 p[0]:=0.5*h*(f(a)+f(b));                { T[0,0]:=p[0]; }
 assign(tt,'tt1.dat');
 rewrite(tt);
 if zw then writeln(p[0]:14:10);
 writeln(tt,p[0]:14:10);
 readln;
 k:=0;
 repeat                                  { for k:=1 to m do }
   k:=k+1;
   s:=0;
   ha:=h; h:=0.5*h;
   ah:=a-h;
   n:=2*n;
   q:=1;
   for i:=1 to n div 2 do s:=s+f(ah+i*ha);
   p[k]:=0.5*p[k-1]+s*h;                 { T[k,0]:=p[k]; }
   if zw then write(p[k]:14:10,' ' );
   write(tt,p[k]:14:10);
   for i:=k-1 downto 0 do
    begin
     q:=q*4;
     p[i]:=p[i+1]+(p[i+1]-p[i])/(q-1);   { T[k,k-i]:=p[i]; }
     if zw then write(p[i]:14:10,' ' );
     write(tt,p[i]:14:10);
    end;
    writeln(tt);
    readln;
 until (abs(p[0]-p[1])<eps) or (k>=mmax);

 IWert:=p[0];
 iter:=k;
end;
```

Die Variablentypen sind aus dem Zusammenhang erkennbar. Alle wichtigen Informationen und der gewünschte Näherungswert des Integrals sind jedoch in der jeweils letzten Zeile des Tableaus enthalten.
Die Ausgabe erfolgt gegebenenfalls auf Monitor und in eine globale Text-Datei. Nach jeder Zeile des Tableaus wird mittels `readln` angehalten.

Berechnen wir nun näherungsweise das bestimmte Integral

$$I(f) = \int_0^1 \sqrt{t}\, e^{-t}\, dt = 0.378\,944\,691\,640\,984\,703\,803...$$

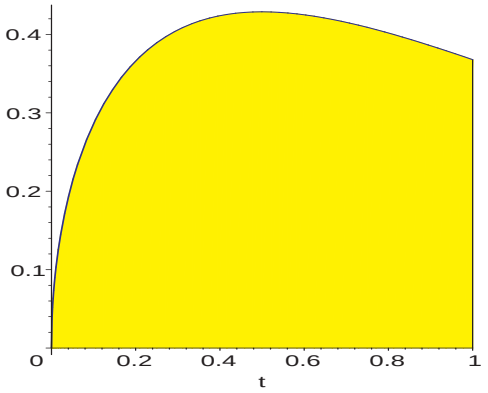

Abb. 2.1 Funktion $f(t) = \sqrt{t}\, e^{-t},\ t \in [0,1]$, und bestimmtes Integral als Fläche

Dazu liefert das Romberg-Verfahren bei einem 6-zeiligen Schema die Datei `tt1.dat` mit folgender Gestalt. Damit wird jedoch noch keine allzu große Genauigkeit erzielt, obwohl $|T[5,4] - T[5,5]| < 10^{-6}$ ist.

```
0.1839397206
0.3064108315   0.3472345351
0.3528258723   0.3682975526   0.3697017538
0.3696612874   0.3752730924   0.3757381284   0.3758339439
0.3756617255   0.3776618715   0.3778211234   0.3778541868   0.3778621093
0.3777857243   0.3784937239   0.3785491807   0.3785607372   0.3785635080   0.3785641936
```

Berechnungen in Maple mit `Digits=10` (Datei *romberg2.mws*)

```
> f:=unapply(sqrt(t)*exp(-t),t);
  Int(f(t),t=0..1)=int(f(t),t=0..1);
  Int(f(t),t=0..1)=evalf(int(f(t),t=0..1));
  Iex:=rhs(%);
```

$$f := t \to \sqrt{t}\ \mathrm{e}^{(-t)}$$

$$\int_0^1 \sqrt{t}\,\mathrm{e}^{(-t)}\, dt = -\mathrm{e}^{(-1)} + \frac{1}{2}\sqrt{\pi}\ \mathrm{erf}(1)$$

$$\int_0^1 \sqrt{t}\,\mathrm{e}^{(-t)}\, dt = 0.3789446918$$

$$Iex := 0.3789446918$$

In der Auswertung ist die letzte Mantissenstelle i. Allg. fehlerbehaftet.

Romberg-Verfahren

Wir illustrieren im Zusammenhang mit Dateiarbeit mehrere Varianten zum Anhalten der Datenausgabe.

Datei für unteres linkes Dreieck des Romberg-Schemas

```
> pfad:=‘C:/D/Neundorf/Maple5/‘:
  name0:=‘tt0.txt‘:
  datei0:=cat(pfad,name0):    # physischer Dateiname
```

Variante 1

```
> mmax:=20;
  feld:=array(sparse,0..mmax,0..mmax); # Nullmatrix
  eval(feld):                          # umfangreiche Ausgabe unterdrueckt
  vektor:=array(sparse,0..mmax);
  eval(vektor);
```

$$mmax := 20$$
$$feld := \mathrm{array}(sparse, 0..20, 0..20, [])$$
$$vektor := \mathrm{array}(sparse, 0..20, [])$$
$$\mathrm{array}(\mathrm{sparse},\ 0\ ..\ 20,\ [(0) = 0,(1) = 0,(2) = 0,...,(20) = 0])$$

Prozeduren mit/ohne Zwischenhalt

```
> Romberg0_m:=proc(zw::boolean,m::posint,a::numeric,b::numeric,
                   IWert::evaln)
  local h,ha,ah,s,q,    # numeric
        i,k,n,          # integer
        p,T,            # array
        td;             # Textdatei
  global f,             # Integrand
         datei0;

  td:=fopen(datei0,WRITE,TEXT):   # Dateivariable

  p:=array(sparse,0..m);
  T:=array(sparse,0..m,0..m);

  h:=b-a;
  n:=1;
  p[0]:=evalf(0.5*h*(f(a)+f(b)));
  T[0,0]:=p[0];
  if zw then fprintf(default,‘%14.10f\n‘,p[0]); end if;
  fprintf(td,‘%14.10f\n‘,p[0]);
  readline();

  for k from 1 to m do
   s:=0;
   ha:=h; h:=0.5*h;
   ah:=a-h;
   n:=2*n;
```

```
 q:=1;
 for i from 1 to n/2 do s:=s+f(ah+i*ha); end do;
 p[k]:=evalf(0.5*p[k-1]+s*h);
 T[k,0]:=p[k];
 if zw then fprintf(default,‘%14.10f ‘,p[k]); end if;
 fprintf(td,‘%14.10f ‘,p[k]);
 for i from k-1 by -1 to 0 do
   q:=q*4;
   p[i]:=evalf(p[i+1]+(p[i+1]-p[i])/(q-1));
   T[k,k-i]:=p[i];
   if zw then fprintf(default,‘%14.10f ‘,p[i]); end if;
   fprintf(td,‘%14.10f ‘,p[i]);
 end do;
 fprintf(default,‘\n‘);
 fprintf(td,‘\n‘);
 readline();
end do;

fclose(td);
lprint(‘<Ausgabe Ende>‘);
IWert:=p[0];
eval(T);
end proc:

> Romberg0_o:=proc(zw::boolean,m::posint,a::numeric,b::numeric,
                   IWert::evaln)
  # bez. des Ergebnisparameters IWert sind zulaessig
  # ... :=proc(zw::boolean,m::posint,a::numeric,b::numeric,IWert::name)
  # ... :=proc(zw::boolean,m::posint,a::numeric,b::numeric,IWert)

  local h,ha,ah,s,q,    # numeric
        i,k,n,          # integer
        p,T,            # array
        td;             # Textdatei
  global f,             # Integrand
         datei0;

  td:=fopen(datei0,WRITE,TEXT):   # Dateivariable

  p:=array(sparse,0..m);
  T:=array(sparse,0..m,0..m);

  h:=b-a;
  n:=1;
  p[0]:=evalf(0.5*h*(f(a)+f(b)));
  T[0,0]:=p[0];
  if zw then fprintf(default,‘%14.10f\n‘,p[0]); end if;
  fprintf(td,‘%14.10f\n‘,p[0]);
  # readline();

  for k from 1 to m do
   s:=0;
   ha:=h; h:=0.5*h;
   ah:=a-h;
```

```
    n:=2*n;
    q:=1;
    for i from 1 to n/2 do s:=s+f(ah+i*ha); end do;
    p[k]:=evalf(0.5*p[k-1]+s*h);
    T[k,0]:=p[k];
    if zw then fprintf(default,‘%14.10f ‘,p[k]); end if;
    fprintf(td,‘%14.10f ‘,p[k]);
    for i from k-1 by -1 to 0 do
      q:=q*4;
      p[i]:=evalf(p[i+1]+(p[i+1]-p[i])/(q-1));
      T[k,k-i]:=p[i];
      if zw then fprintf(default,‘%14.10f ‘,p[i]); end if;
      fprintf(td,‘%14.10f ‘,p[i]);
    end do;
    fprintf(default,‘\n‘);
    fprintf(td,‘\n‘);
    # readline();
  end do;

  fclose(td);
  # lprint(‘<Ausgabe Ende>‘);
  IWert:=p[0];
  eval(T);
  end proc:
```

Parameter

```
> m:=10:
  a:=0:
  b:=1:
```

Im Aufruf der **Prozedur mit Ausgabehalt** bemerkt man einen Unterschied zwischen Maple 9.5 und Classic Maple 9.5.

```
> # Maple 9.5 mit Ausgabehalt, keine leeren Maple-Zellen vorbereiten
  lprint(‘Bei Ausgabehalt <Enter>‘);
  TT:=Romberg0_m(true,m,a,b,IWert):
```

Anhalten bedeutet das Öffnen eines Dialog-Fensters `Read a Line` mit der Aufforderung `Enter a line of text` in einem Eingabebalken. Quittiert man dies einfach mit OK (leerer Text), so wird die Abarbeitung fortgesetzt. Das Ergebnistableau erscheint dann in der folgenden kompakten Darstellung (Abstand zwischen den Spalten im Nachhinein verkleinert).

```
‘Bei Ausgabehalt <Enter>‘

  0.1839397206
  0.3064108315 0.3472345351
  0.3528258723 0.3682975526 0.3697017538
  0.3696612874 0.3752730924 0.3757381284 0.3758339439
  0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
  0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.3785635080 0.3785641936
  0.3785357016 0.3787856940 0.3788051587 0.3788092218 0.3788101962 0.3788104373 0.3788104974
  0.3788003417 0.3788885551 0.3788954125 0.3788968451 0.3788971887 0.3788972737 0.3788972949 0.3788973002
  0.3788937288 0.3789248578 0.3789272780 0.3789277838 0.3789279051 0.3789279351 0.3789279426 0.3789279445 0.3789279450
  0.3789266935 0.3789376817 0.3789385366 0.3789387153 0.3789387582 0.3789387688 0.3789387714 0.3789387721 0.3789387723 0.3789387723
  0.3789383338 0.3789422139 0.3789425160 0.3789425792 0.3789425944 0.3789425981 0.3789425990 0.3789425992 0.3789425993 0.3789425993 0.3789425993
‘<Ausgabe Ende>‘
```

```
# Classic Maple mit Ausgabehalt, m+1 leere Maple-Zellen (Execution Group)
  lprint(‘Bei Ausgabehalt <Enter>‘);
  TT:=Romberg0_m(true,m,a,b,IWert);
```

Bevor die Prozedur aufgerufen wird, muss man $m+1$ leere Maple-Zellen im Anschluss an die Maple-Zelle mit dem Prozeduraufruf vorbereiten.
Die Ausführung der Prozedur beginnt mit

```
‘Bei Ausgabehalt <Enter>‘
  0.1839397206
```

Die Abarbeitung wird nun gestoppt und in der nächsten leeren Maple-Zelle wartet der Cursor auf die Betätigung der <Enter>-Taste. Erfolgt das, wird erst dann in dieser Maple-Zelle die 2. Zeile des Romberg-Schemas ausgegeben. Der Cursor springt danach in die nächste leere Maple-Zelle usw.
Das Ergebnistableau erscheint somit in der folgenden “aufgelockerten“ Darstellung (Abstand zwischen den Spalten im Nachhinein verkleinert).

```
‘Bei Ausgabehalt <Enter>‘
  0.1839397206
>
  0.3064108315 0.3472345351
>
  0.3528258723 0.3682975526 0.3697017538
>
  0.3696612874 0.3752730924 0.3757381284 0.3758339439
>
  0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
>
  0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.3785635080 0.3785641936
>
  0.3785357016 0.3787856940 0.3788051587 0.3788092218 0.3788101962 0.3788104373 0.3788104974
>
  0.3788003417 0.3788885551 0.3788954125 0.3788968451 0.3788971887 0.3788972737 0.3788972949 0.3788973002
>
  0.3788937288 0.3789248578 0.3789272780 0.3789277838 0.3789279051 0.3789279351 0.3789279426 0.3789279445 0.3789279450
>
  0.3789266935 0.3789376817 0.3789385366 0.3789387153 0.3789387582 0.3789387688 0.3789387714 0.3789387721 0.3789387723 0.3789387723
>
  0.3789383338 0.3789422139 0.3789425160 0.3789425792 0.3789425944 0.3789425981 0.3789425990 0.3789425992 0.3789425993 0.3789425993 0.3789425993
>
‘<Ausgabe Ende>‘
```

Da der Aufruf mit Semikolon abschließt, wird noch einmal das Ergebnistableau als `array` in folgender Form ausgegeben (nicht alle Elemente notiert).

$TT :=$ array($sparse$, 0..10, 0..10,
[(0, 0) = 0.1839397206,(0, 1) = 0,(0, 2) = 0,...,(0, 9) = 0,(0, 10) = 0,
(1, 0) = 0.3064108315,(1, 1) = 0.3472345351,(1, 2) = 0,...,(1, 9) = 0,(1, 10) = 0,
(2, 0) = 0.3528258723,(2, 1) = 0.3682975526,(2, 2) = 0.3697017538,...,(2, 9) = 0,(2, 10) = 0,
...
(9, 0) = 0.3789266935,(9, 1) = 0.3789376817,(9, 2) = 0.3789385366,...,(9, 9) = 0.3789387723,(9, 10) = 0,
(10, 0) = 0.3789383338,(10, 1) = 0.3789422139,(10, 2) = 0.3789425160,..., (10, 9) = 0.3789425993,(10, 10) = 0.3789425993])

```
> IWert;
  TT;           # Ausgabe des Variablennamens
  evalm(TT);    #    "
  eval(TT):     # Ausgabe aller Elemente in Langform wie oben
                # untereinander (Classic Maple 9.5) bzw.
                # nebeneinander (Maple 9.5)
```

$$0.3789425993$$

$$TT$$

$$TT$$

Test der Prozedur ohne Ausgabehalt
Dabei wird noch der Einfluss der Wahl des Typs vom formalen Ergebnisparameter `IWert` sowie seine Ersetzung durch den aktuellen Parameter untersucht.

```
> # Prozedur ohne Ausgabehalt
  IWert:='IWert':
  TT:=Romberg0_o(true,m,a,b,IWert):
      # moeglich bei formalen Parameter  IWert::evaln, name  bzw. IWert
  IWert;

  IWert:='IWert':
  TT:=Romberg0_o(true,m,a,b,'IWert'):
      # nicht moeglich bei formalen Parameter  IWert::evaln
      # moeglich bei formalen Parameter  IWert::name  bzw. IWert
  IWert;
```

```
0.1839397206
0.3064108315 0.3472345351
0.3528258723 0.3682975526 0.3697017538
0.3696612874 0.3752730924 0.3757381284 0.3758339439
0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.3785635080 0.3785641936
0.3785357016 0.3787856940 0.3788051587 0.3788092218 0.3788101962 0.3788104373 0.3788104974
0.3788003417 0.3788885551 0.3788954125 0.3788968451 0.3788971887 0.3788972737 0.3788972949 0.3788973002
0.3788937288 0.3789248578 0.3789272780 0.3789277838 0.3789279051 0.3789279351 0.3789279426 0.3789279445 0.3789279450
0.3789266935 0.3789376817 0.3789385366 0.3789387153 0.3789387582 0.3789387688 0.3789387714 0.3789387721 0.3789387723 0.3789387723
0.3789383338 0.3789422139 0.3789425160 0.3789425792 0.3789425944 0.3789425981 0.3789425990 0.3789425992 0.3789425993 0.3789425993 0.3789425993
```

$$0.3789425993$$

```
Error, illegal use of an object as a name
```

$$IWert$$

Als nächstes werden wir aus dem Tableau eine Matrix erzeugen.
Zwei Arten der Transformation `array TT -> matrix TT1` führen auf das gleiche Ergebnis. Die sparse Belegung geht dabei verloren.

```
> # einfache Variante
  TT1:=convert(TT,matrix);

> # andere Variante
  TT1:=matrix(m+1,m+1,[]):
  for j from 0 to m do
   for k from 0 to m do
    TT1[j+1,k+1]:=evalf(TT[j,k]);
   end do;
  end do:
  TT1:=evalm(TT1);
```

$TT1 :=$

$$\begin{bmatrix}
0.1839397206 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.3064108315 & 0.3472345351 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.3528258723 & 0.3682975526 & 0.3697017538 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.3696612874 & 0.3752730924 & 0.3757381284 & 0.3758339439 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.3756617255 & 0.3776618715 & 0.3778211234 & 0.3778541868 & 0.3778621093 & 0 & 0 & 0 & 0 & 0 & 0 \\
0.3777857243 & 0.3784937239 & 0.3785491807 & 0.3785607372 & 0.3785635080 & 0.3785641936 & 0 & 0 & 0 & 0 & 0 \\
0.3785357016 & 0.3787856940 & 0.3788051587 & 0.3788092218 & 0.3788101962 & 0.3788104373 & 0.3788104974 & 0 & 0 & 0 & 0 \\
0.3788003417 & 0.3788885551 & 0.3788954125 & 0.3788968451 & 0.3788971887 & 0.3788972737 & 0.3788972949 & 0.3788973002 & 0 & 0 & 0 \\
0.3788937288 & 0.3789248578 & 0.3789272780 & 0.3789277838 & 0.3789279051 & 0.3789279351 & 0.3789279426 & 0.3789279445 & 0.3789279450 & 0 & 0 \\
0.3789266935 & 0.3789376817 & 0.3789385366 & 0.3789387153 & 0.3789387582 & 0.3789387688 & 0.3789387714 & 0.3789387721 & 0.3789387723 & 0.3789387723 & 0 \\
0.3789383338 & 0.3789422139 & 0.3789425160 & 0.3789425792 & 0.3789425944 & 0.3789425981 & 0.3789425990 & 0.3789425992 & 0.3789425993 & 0.3789425993 & 0.3789425993
\end{bmatrix}$$

In Maple 9.5 wird die Ausgabe bei einer Matrixdimension ≥ 10 umgeschaltet auf die zeilenweise Darstellung.

Nun speichern wir Tableau und Matrix als Dateien ab.
Wir verwenden die physischen Dateinamen `C:/D/Neundorf/Maple5/tableau*.txt`.
Tableau `TT` speichern und lesen

```
> name1:=‘tableau1.txt‘:
  datei1:=cat(pfad,name1):     # physischer Dateiname
  save TT,datei1;  # als array mit Namen TT speichern,
                   # Abspeichern der Nichtnull-Elemente nebeneinander
  read datei1;     # Ausgabe aller Elemente in Langform
                   # untereinander (Classic Maple 9.5) gemaess
                   #            TT:=array(sparse,0..10,0..10,[
                   #            (0,0)=0.1839397206
                   #            ...
                   # bzw. nebeneinander (Maple 9.5)
  print(TT):       # Ausgabe der Elemente untereinander/nebeneinander
```

Die Datei `tableau1.txt` hat folgenden Aufbau. Man beachte, dass wegen der Option `sparse` wirklich nur die Nichtnull-Elemente gespeichert werden. Ihre Reihenfolge ist ungeordnet und für die verschiedenen Maple-Versionen unterschiedlich.

```
TT := array(sparse,0 .. 10, 0 .. 10,[(7, 0)=.3788003417,(10, 1)=.3789422139,(
10, 9)=.3789425993,(6, 2)=.3788051587,(9, 9)=.3789387723,(10, 0)=.3789383338,(
9, 7)=.3789387721,(6, 5)=.3788104373,(6, 0)=.3785357016,(7, 1)=.3788885551,(8,
3)=.3789277838,(9, 0)=.3789266935,(7, 3)=.3788968451,(10, 2)=.3789425160,(4, 4
)=.3778621093,(8, 1)=.3789248578,(9, 3)=.3789387153,(9, 1)=.3789376817,(1, 0)=
.3064108315,(10, 6)=.3789425990,(5, 1)=.3784937239,(8, 7)=.3789279445,(3, 3)=.\
3758339439,(7, 7)=.3788973002,(5, 0)=.3777857243,(4, 2)=.3778211234,(3, 2)=.37\
57381284,(10, 10)=.3789425993,(8, 0)=.3788937288,(6, 4)=.3788101962,(9, 4)=.37\
89387582,(3, 0)=.3696612874,(9, 6)=.3789387714,(5, 2)=.3785491807,(5, 3)=.3785\
607372,(5, 5)=.3785641936,(10, 5)=.3789425981,(9, 2)=.3789385366,(4, 3)=.37785\
41868,(5, 4)=.3785635080,(3, 1)=.3752730924,(8, 2)=.3789272780,(7, 6)=.3788972\
949,(2, 2)=.3697017538,(10, 3)=.3789425792,(6, 3)=.3788092218,(7, 5)=.37889727\
37,(1, 1)=.3472345351,(4, 1)=.3776618715,(10, 8)=.3789425993,(8, 4)=.378927905\
1,(2, 0)=.3528258723,(8, 6)=.3789279426,(8, 5)=.3789279351,(7, 2)=.3788954125,
(6, 1)=.3787856940,(7, 4)=.3788971887,(2, 1)=.3682975526,(4, 0)=.3756617255,(9
, 5)=.3789387688,(9, 8)=.3789387723,(10, 7)=.3789425992,(10, 4)=.3789425944,(0
, 0)=.1839397206,(6, 6)=.3788104974,(8, 8)=.3789279450]);
```

Damit macht das Kommando `readdata` zum Lesen der jeweiligen anfänglichen Spalten der Datei keinen Sinn und ist nicht nachvollziehbar. Zur Illustration, der Wert 0.3788972949 steht z. B. im Element (7,6).

```
> readdata(datei1,1);    # read the first column of the data
  readdata(datei1,2);    # read the first two columns of the data
```

$$[0., 0.3788972949\ 10^{10}, 0.3777857243\ 10^{10}, 0.97017538\ 10^{8}, 0.9425993\ 10^{7}, 1962., 551., 16., 9., 4., 0.]$$

$$[[0.], [0.3788972949\ 10^{10}], [0.3777857243\ 10^{10}], [0.97017538\ 10^{8}], [0.9425993\ 10^{7}], [1962.], [551.], [16.], [9.], [4.], [0.]]$$

Matrix TT1 speichern und lesen

```
> name2:=‘tableau2.txt‘:
  datei2:=cat(pfad,name2):
  save TT1,datei2; # als matrix mit Namen TT1 speichern
                   # (wie bei array(1.. ,1..))
                   # Abspeichern aller Elemente nebeneinander
  read datei2;     # Ausgabe von Matrixname und allen Elementen als Matrix
  print(TT1):      # Ausgabe aller Elemente als Matrix
```

Die Ausgabe der Matrix TT1 als übliche Form haben wir mit dem Kommando TT1:=evalm(TT1) in Classic Maple 9.5 schon vorgeführt.
In Maple 9.5 erhält man wegen $m = 10$ (damit Matrixdimension ≥ 10) eine zeilenweise Darstellung.

$TT1$:= [[0.1839397206, 0., 0., 0., 0., 0., 0., 0., 0., 0.], [.3064108315, .3472345351, 0., 0., 0., 0., 0., 0., 0., 0.],
[0.3528258723, 0.3682975526, 0.3697017538, 0., 0., 0., 0., 0., 0., 0.],
[0.3696612874, 0.3752730924, 0.3757381284, 0.3758339439, 0., 0., 0., 0., 0., 0.],
[0.3756617255, 0.3776618715, 0.3778211234, 0.3778541868, 0.3778621093, 0., 0., 0., 0., 0.],
[0.3777857243, 0.3784937239, 0.3785491807, 0.3785607372, 0.3785635080, 0.3785641936, 0., 0., 0., 0.],
[0.3785357016, 0.3787856940, 0.3788051587, 0.3788092218, 0.3788101962, 0.3788104373, 0.3788104974, 0., 0., 0.],
[0.3788003417, 0.3788885551, 0.3788954125, 0.3788968451, 0.3788971887, 0.3788972737, 0.3788972949, 0.3788973002, 0., 0.],
[0.3788937288, 0.3789248578, 0.3789272780, 0.3789277838, 0.3789279051, 0.3789279351, 0.3789279426, 0.3789279445, 0.3789279450, 0., 0.],
[0.3789266935, 0.3789376817, 0.3789385366, 0.3789387153, 0.3789387582, 0.3789387688, 0.3789387714, 0.3789387721, 0.3789387723,
0.3789387723, 0.],
[0.3789383338, 0.3789422139, 0.3789425160, 0.3789425792, 0.3789425944, 0.3789425981, 0.3789425990, 0.3789425992, 0.3789425993,
0.3789425993, 0.3789425993]]

Die Datei tableau2.txt hat folgenden Aufbau. Man beachte, dass alle Elemente gespeichert werden. Ihre Reihenfolge ist ungeordnet und für die verschiedenen Maple-Versionen unterschiedlich.

```
TT1 := array(1 .. 11, 1 .. 11,[(2, 5)=0.,(6, 3)=.3785491807,(11, 3)=.378942516\
0,(5, 1)=.3756617255,(5, 6)=0.,(2, 6)=0.,(1, 10)=0.,(3, 8)=0.,(9, 8)=.37892794\
45,(4, 7)=0.,(4, 2)=.3752730924,(10, 1)=.3789266935,(11, 8)=.3789425992,(2, 4)
=0.,(9, 11)=0.,(11, 7)=.3789425990,(10, 6)=.3789387688,(11, 4)=.3789425792,(8,
9)=0.,(6, 10)=0.,(7, 8)=0.,(3, 10)=0.,(9, 5)=.3789279051,(10, 7)=.3789387714,(
8, 11)=0.,(11, 1)=.3789383338,(7, 10)=0.,(8, 10)=0.,(10, 4)=.3789387153,(2, 9)
=0.,(8, 3)=.3788954125,(10, 3)=.3789385366,(8, 5)=.3788971887,(11, 6)=.3789425\
981,(1, 3)=0.,(4, 3)=.3757381284,(11, 10)=.3789425993,(7, 5)=.3788101962,(11,
11)=.3789425993,(1, 2)=0.,(4, 11)=0.,(8, 6)=.3788972737,(8, 4)=.3788968451,(5,
10)=0.,(5, 9)=0.,(4, 9)=0.,(3, 6)=0.,(9, 3)=.3789272780,(7, 2)=.3787856940,(5,
11)=0.,(11, 9)=.3789425993,(5, 7)=0.,(10, 10)=.3789387723,(2, 3)=0.,(8, 2)=.37\
88885551,(3, 5)=0.,(2, 11)=0.,(5, 5)=.3778621093,(6, 8)=0.,(6, 11)=0.,(4, 8)=0\
.,(8, 7)=.3788972949,(1, 9)=0.,(4, 4)=.3758339439,(5, 8)=0.,(7, 4)=.3788092218
,(10, 11)=0.,(6, 7)=0.,(7, 11)=0.,(3, 2)=.3682975526,(1, 7)=0.,(3, 1)=.3528258\
723,(4, 5)=0.,(5, 4)=.3778541868,(5, 2)=.3776618715,(7, 9)=0.,(10, 8)=.3789387\
721,(7, 6)=.3788104373,(9, 4)=.3789277838,(2, 1)=.3064108315,(7, 3)=.378805158\
7,(3, 11)=0.,(6, 6)=.3785641936,(1, 4)=0.,(4, 10)=0.,(3, 3)=.3697017538,(3, 4)
=0.,(6, 4)=.3785607372,(6, 1)=.3777857243,(6, 9)=0.,(7, 7)=.3788104974,(11, 2)
=.3789422139,(9, 1)=.3788937288,(2, 8)=0.,(2, 10)=0.,(7, 1)=.3785357016,(10, 5
)=.3789387582,(9, 7)=.3789279426,(5, 3)=.3778211234,(6, 5)=.3785635080,(1, 8)=
0.,(3, 9)=0.,(8, 1)=.3788003417,(10, 2)=.3789376817,(1, 1)=.1839397206,(11, 5)
=.3789425944,(1, 11)=0.,(8, 8)=.3788973002,(6, 2)=.3784937239,(9, 10)=0.,(9, 9
)=.3789279450,(1, 6)=0.,(3, 7)=0.,(9, 6)=.3789279351,(2, 7)=0.,(4, 1)=.3696612\
874,(1, 5)=0.,(10, 9)=.3789387723,(9, 2)=.3789248578,(2, 2)=.3472345351,(4, 6)
=0.]);
```

Damit macht auch hier das Kommando `readdata` zum Lesen der jeweiligen anfänglichen Spalten der Datei keinen Sinn und ist nicht nutzbar.

```
> readdata(datei2,1);    # read the first column of the data
  readdata(datei2,2);    # read the first two columns of the data
```

$$[8., 5., 2., 962., 4373., 7., 937239., 3., 885551., 4., 0.3789272780\ 10^{10}, 0.9277838\ 10^{7}, 0.789387153\ 10^{9}, 5366.]$$
$$[[8.], [5.], [2.], [962.], [4373.], [7.], [937239.], [3.], [885551.], [4.], [0.3789272780\ 10^{10}], [0.9277838\ 10^{7}], [0.789387153\ 10^{9}], [5366.]]$$

Zu empfehlen ist die Verwendung einer Dateivariablen in Ergänzung zum physischen Dateinamen für die Matrix `TT1`.

```
> name3:=‘tableau3.txt‘:
  datei3:=cat(pfad,name3):
  td3:=fopen(datei3,WRITE,TEXT): # File variable, file descriptor
                                 # Explicitly open the file of type TEXT
  writedata(td3,TT1);            # Write the matrix in the file
  fclose(td3);                   # Close the file

  writedata(terminal,TT1);       # Printed on the user’s terminal
```

Das Ergebnis der Kontrollausgabe `writedata` auf dem Bildschirm ist

```
0.1839397206 0 0 0 0 0 0 0 0 0
0.3064108315 0.3472345351 0 0 0 0 0 0 0 0
0.3528258723 0.3682975526 0.3697017538 0 0 0 0 0 0 0
0.3696612874 0.3752730924 0.3757381284 0.3758339439 0 0 0 0 0 0
0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093 0 0 0 0 0
0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.378563508  0.3785641936 0 0 0 0
0.3785357016 0.378785694  0.3788051587 0.3788092218 0.3788101962 0.3788104373 0.3788104974 0 0 0
0.3788003417 0.3788885551 0.3788954125 0.3788968451 0.3788971887 0.3788972737 0.3788972949 0.3788973002 0 0 0
0.3788937288 0.3789248578 0.378927278  0.3789277838 0.3789279051 0.3789279351 0.3789279426 0.3789279445 0.378927945  0 0
0.3789266935 0.3789376817 0.3789385366 0.3789387153 0.3789387582 0.3789387688 0.3789387714 0.3789387721 0.3789387723 0.3789387723 0
0.3789383338 0.3789422139 0.378942516  0.3789425792 0.3789425944 0.3789425981 0.378942599  0.3789425992 0.3789425993 0.3789425993 0.3789425993
```

Die abgespeicherte Datei `tableau3.txt` hat eine analoge Struktur, nur dass zwischen den Spalten etwas größere Abstände auftreten können.

Die Verwendung von Variablennamen für die Datei gestattet dann den sinvollen Umgang mit den Kommandos `readdata` und `fscanf`.
Zunächst können wir das Kommando `readdata` und den physischen Dateinamen zum Lesen der jeweiligen anfänglichen Spalten der Datei `tableau3.txt` nutzen.

```
> readdata(datei3);     # read the first column of the data
  readdata(datei3,1);   # read the first column of the data
  readdata(datei3,2);   # read the first 2 columns of the data
  readdata(datei3,3);   # read the first 3 columns of the data
```

$$[\ 0.1839397206, 0.3064108315, 0.3528258723, 0.3696612874, 0.3756617255, 0.3777857243, 0.3785357016, 0.3788003417, 0.3788937288, 0.3789266935, 0.3789383338\ ]$$
$$[\ 0.1839397206, 0.3064108315, 0.3528258723, 0.3696612874, 0.3756617255, 0.3777857243, 0.3785357016, 0.3788003417, 0.3788937288, 0.3789266935, 0.3789383338\ ]$$
$$[[\ 0.1839397206, 0.\ ], [0.3064108315, 0.3472345351\ ], [0.3528258723, 0.3682975526\ ], [0.3696612874, 0.3752730924\ ], [0.3756617255, 0.3776618715\ ], [0.3777857243, 0.3784937239\ ], [0.3785357016, 0.378785694\ ], [0.3788003417, 0.3788885551\ ], [0.3788937288, 0.3789248578\ ], [0.3789266935, 0.3789376817\ ], [0.3789383338, 0.3789422139\ ]]$$
$$[[\ 0.1839397206, 0., 0.\ ], [0.3064108315, 0.3472345351, 0.\ ], [\ 0.3528258723, 0.3682975526, 0.3697017538\ ], [\ 0.3696612874, 0.3752730924, 0.3757381284\ ], [\ 0.3756617255, 0.3776618715, 0.3778211234\ ], [\ 0.3777857243, 0.3784937239, 0.3785491807\ ], [\ 0.3785357016, 0.378785694, 0.3788051587\ ], [\ 0.3788003417, 0.3788885551, 0.3788954125\ ], [\ 0.3788937288, 0.3789248578, 0.378927278\ ], [\ 0.3789266935, 0.3789376817, 0.3789385366\ ], [\ 0.3789383338, 0.3789422139, 0.378942516\ ]]$$

Bei Verwendung von Variablennamen der Datei ist die normale Situation wie folgt: Datei öffnen, Daten lesen, Datei schließen.

```
> td31:=fopen(datei3,READ,TEXT):
  readdata(td31);     # read the first column of the data
                      # wie readdata(td31,1)
  fclose(td31);

> td32:=fopen(datei3,READ,TEXT):
  readdata(td32,2);   # read the first two column of the data
  fclose(td32);

> td33:=fopen(datei3,READ):
  readdata(td33,float,3);   # read the first three column of the data
                            # as floating-point numbers
  fclose(td33);
```

[0.1839397206, 0.3064108315, 0.3528258723, 0.3696612874, 0.3756617255, 0.3777857243, 0.3785357016, 0.3788003417, 0.3788937288, 0.3789266935, 0.3789383338]

[[0.1839397206, 0.], [0.3064108315, 0.3472345351], [0.3528258723, 0.3682975526], [0.3696612874, 0.3752730924], [0.3756617255, 0.3776618715], [0.3777857243, 0.3784937239], [0.3785357016, 0.378785694], [0.3788003417, 0.3788885551], [0.3788937288, 0.3789248578], [0.3789266935, 0.3789376817], [0.3789383338, 0.3789422139]]

[[0.1839397206, 0., 0.], [0.3064108315, 0.3472345351, 0.], [0.3528258723, 0.3682975526, 0.3697017538], [0.3696612874, 0.3752730924, 0.3757381284], [0.3756617255, 0.3776618715, 0.3778211234], [0.3777857243, 0.3784937239, 0.3785491807], [0.3785357016, 0.378785694, 0.3788051587], [0.3788003417, 0.3788885551, 0.3788954125], [0.3788937288, 0.3789248578, 0.378927278], [0.3789266935, 0.3789376817, 0.3789385366], [0.3789383338, 0.3789422139, 0.378942516]]

Wir machen noch einen Test bei wiederholtem Aufruf von `readdata`, wobei dann die obige Reihenfolge der Dateiarbeit nicht eingehalten worden ist.
Die nachfolgende Bemerkung in der Maple-Hilfe ist nicht korrekt, weil nach `readdata` die Datei nicht geschlossen wird, sondern der Dateizeiger am Dateiende steht und weiteres Lesen die Position des Dateizeigers nicht ändert und "ins Leere" führt.

```
> td3:=fopen(datei3,READ,TEXT):
  readdata(td3);    # read the first column of the data
                    # Maple-Hilfe:
                    # Furthermore, if a file name is specified,
                    # the file is closed when readdata returns.
  readdata(td3);    # Inhalt leer und Dateizeiger weiterhin am Dateiende
  fclose(td3);

> readdata(td3);    # Error, (in readline) file descriptor not in use
                    # Datei nicht geoeffnet
```

[0.1839397206, 0.3064108315, 0.3528258723, 0.3696612874, 0.3756617255, 0.3777857243, 0.3785357016, 0.3788003417, 0.3788937288, 0.3789266935, 0.3789383338]

[ ]

```
Error, (in readline) file descriptor not in use
```

Etwas komplizierter wird es, wenn ausgewählte Spalten der Datei zu lesen sind, etwa alle ungeraden Spalten. Dafür und natürlich auch für die ganz normale Situation eignet sich das Kommando `fscanf`.

Zunächst mit `fscanf` die normale Situation: anfängliche Spalten lesen.
Dazu benötigen ein sogenanntes Leseformat (Schablone).

```
> # Lesen der ersten 2 Spalten der Datei tableau3.txt
  ls12:=[seq([0,0],i=1..m+1)]:  # Ziel der Speicherung

     # The optional * in the format string indicates
     # that the object is to be scanned, but not returned
     # as part of the result (that is, it is discarded).

  fh1:=`%*f`:  # Leseformat einstellen
  fh:=fh1:
  for i from 1 to m-2 do fh:=cat(fh,fh1); end do:
  fh:=cat(`%f%f`,fh,`\n`):
              # Spalten 1,2 werden gelesen und 3,4,...,m+1 uebergangen

  td4:=fopen(datei3,READ):
  for i from 0 to m do
   ls12[i+1]:=fscanf(td4,fh);
  end do:

  # Kontrolle
  ls12[1];
  ls12[m+1];
  ls12;
  fclose(td4);
```

[0.1839397206, 0.]

[0.3789383338, 0.3789422139]

[[0.1839397206, 0.], [0.3064108315, 0.3472345351], [0.3528258723, 0.3682975526], [0.3696612874, 0.3752730924], [0.3756617255, 0.3776618715], [0.3777857243, 0.3784937239], [0.3785357016, 0.378785694], [0.3788003417, 0.3788885551], [0.3788937288, 0.3789248578], [0.3789266935, 0.3789376817], [0.3789383338, 0.3789422139]]

```
> # Lesen der ungeraden Spalten der Datei tableau3.txt
  anz:=1+iquo(m,2);
  lsu:=[seq([0$anz],i=1..m+1)]:  # Ziel der Speicherung

  fh1:=`%f%*f`:   # Leseformat einstellen
  fh:=fh1:
  for i from 1 to anz-2 do fh:=cat(fh,fh1); end do:
  fh:=cat(fh,`%f\n`):
                  # Spalten 1,3,5,7,9,11 werden gelesen
                  # und 2,4,6,8,10 uebergangen

  td5:=fopen(datei3,READ):
  for i from 0 to m do
   lsu[i+1]:=fscanf(td5,fh);
  end do:

  # Kontrolle
  lsu;
  fclose(td5);
```

In Maple 9.5 haben wir beim Aufruf der Prozedur `Romberg0_m(true,m,a,b,IWert)` bemerkt, dass das enthaltene Kommando `readline()` zum Anhalten der Abarbeitung (Stopp der Ausgabe) führt. Gleichzeitig bedeutete es das Öffnen eines Dialog-Fensters `Read a Line` mit der Aufforderung `Enter a line of text` in einem Eingabebalken. Bei einfacher Quittierung mit OK (leerer Text) wurde die Abarbeitung fortgesetzt.

Eine mit dieser Situation vergleichbare Möglichkeit bietet die Fenster-Eingabe in der Java-Umgebung. Dazu gibt es das Maple-Paket `Maplets Package` für Fenster, Dialoge und visuelle Interfaces. Das Maplets package nutzt die Technology von Java [TM] Runtime Environment.
Wir beschränken uns hier auf unseren konkreten Anwendungsbereich mit entsprechenden Illustrationen und verweisen weiterführend auf die Informationen aus der Maple-Hilfe.
Zunächst testen wir die Vorgehensweise mit dieser Fenstereingabe. Es öffnet sich ein Dialog-Fenster `Input Request` mit der Aufforderung `Enter an integer` in einem Eingabebalken. Bei einfacher Quittierung z. B. mit 1 & OK bzw. 1 & <Enter> wird die Arbeit fortgesetzt. Es folgen noch einige Kontrollausgaben.

```
> with(Maplets[Elements]):
> maplet_integer:=Maplet(
     InputDialog['ID1']("Enter an integer",
        'onapprove'=Shutdown(['ID1']),
        'oncancel'=Shutdown()
     )):

> z:=Maplets[Display](maplet_integer);
  whattype(z);
  op(z);
  whattype(op(z));
  z1:=convert(op(z),name);
```

```
Initializing Java runtime environment.
```

$$z := [\text{“1“}]$$
$$\textit{list}$$
$$\text{“1“}$$
$$\textit{string}$$
$$\textit{z1} := \textit{1}$$

Mit dem Ergebnis *z1* als Zeichenkette (hier ein Zeichen lang) kann man numerisch jedoch nicht rechnen. Der Befehl `evalf(sin(z1))` liefert `sin(`*1*`)` und keinen numerischen Wert.
Ähnlich ist *z2* in den folgenden Zeilen keine numerische Größe, obwohl der Befehl `IsAlphaNumeric(z2)` mit *true* beantwortet wird.

```
> z2:=convert(op(z),rational,1);
  op(evalf(z2));
  convert(z2,float);
```

Analog öffnen sich Dialog-Fenster `Input Request` mit der Aufforderung `Enter an string` bzw. `Continue by Enter` in einem Eingabebalken.
Bei einfacher Quittierung z. B. mit abc & OK, abc & <Enter> bzw. nur <Enter> wird die Arbeit fortgesetzt.

```
> maplet_string:=Maplet(
    InputDialog['ID1']("Enter a string",    # kann auch leere ZK sein
       'onapprove'=Shutdown(['ID1']),
       'oncancel'=Shutdown()
    )):

> z:=Maplets[Display](maplet_string);
  whattype(z);
  op(z);
  whattype(op(z));
  z1:=convert(op(z),name);
```

$$z := [\text{“}abc\text{“}]$$
$$list$$
$$\text{“}abc\text{“}$$
$$string$$
$$z1 := abc$$

```
> maplet_enter:=Maplet(
    InputDialog['ID1']("Continue by Enter",    # kann auch ZK sein
       'onapprove'=Shutdown(['ID1']),
       'oncancel'=Shutdown()
    )):

> z:=Maplets[Display](maplet_enter);
  whattype(z);
  op(z);
  whattype(op(z));
  z1:=convert(op(z),name);

> # Kurzform
  Maplets[Display](maplet_enter):
```

$$z := [\text{“”}]$$
$$list$$
$$\text{“”}$$
$$string$$
$$z1 :=$$

Die Kurzform `Maplets[Display](maplet_enter)` des Anhaltens der Abarbeitung mit Fenstereingabe soll im Romberg-Verfahren verwendet werden.
Dabei sind die Ergebnisse in einem Vektor gemerkt, können aber auch noch als Tableau $T[0..m, 0..m]$ abgelegt werden. Zusätzlich erfolgt die Speicherung in einer Datei. Somit enstehen unter Berücksichtigung der eingefügten Kommentare mehrere Versionen der entsprechenden Prozedur.

```
> Romberg0_e:=proc(zw::boolean,m::posint,a::numeric,b::numeric,
                   # tol::numeric,iter::evaln,
                   IWert::evaln)
  local h,ha,ah,s,q,    # numeric
        i,k,n,          # integer
        p,T,            # array
        td;             # Textdatei
  global f,             # Integrand
         datei0,
         maplet_enter;

  td:=fopen(datei0,WRITE,TEXT):

  p:=array(sparse,0..m);
  T:=array(sparse,0..m,0..m);       # ev. ohne Tableau

  h:=b-a;
  n:=1;
  p[0]:=evalf(0.5*h*(f(a)+f(b)));
  T[0,0]:=p[0];
  if zw then fprintf(default,`%14.10f\n`,p[0]); end if;
  fprintf(td,`%14.10f\n`,p[0]);

  Maplets[Display](maplet_enter);   # readline();

  for k from 1 to m do
    s:=0;
    ha:=h; h:=0.5*h;
    ah:=a-h;
    n:=2*n;
    q:=1;
    for i from 1 to n/2 do s:=s+f(ah+i*ha); end do;
    p[k]:=evalf(0.5*p[k-1]+s*h);
    T[k,0]:=p[k];
    if zw then fprintf(default,`%14.10f `,p[k]); end if;
    fprintf(td,`%14.10f `,p[k]);
    for i from k-1 by -1 to 0 do
      q:=q*4;
      p[i]:=evalf(p[i+1]+(p[i+1]-p[i])/(q-1));
      T[k,k-i]:=p[i];
      if zw then fprintf(default,`%14.10f `,p[i]); end if;
      fprintf(td,`%14.10f `,p[i]);
    end do;
    fprintf(default,`\n`);
    fprintf(td,`\n`);

    Maplets[Display](maplet_enter); # readline();
                                    # if abs(p[0]-p[1])<tol then break; end if;
  end do;

  fclose(td);
  # lprint(`<Ausgabe Ende>`);
                                    # iter:=k;
  IWert:=p[0];
  T;                                # ev. p; bzw. Liste [seq(p[i],i=0..k)];
 end proc:
```

Wir rufen diese Prozedur auf.

```
> # mit Ausgabehalt und Eingabefenster
  lprint(‘Bei Ausgabehalt <Eingabefenster>‘);
  TT:=Romberg0_e(true,m,a,b,IWert):

> IWert;
  eval(TT):   # lange Ausgabe unterdrueckt
```

So erhalten wir die schon erwähnte Datei `tt0.dat` sowie die Ausgabe

```
‘Bei Ausgabehalt <Eingabefenster>‘
0.1839397206
0.3064108315 0.3472345351
0.3528258723 0.3682975526 0.3697017538
0.3696612874 0.3752730924 0.3757381284 0.3758339439
0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.378563508  0.3785641936
0.3785357016 0.378785694  0.3788051587 0.3788092218 0.3788101962 0.3788104373 0.3788104974
0.3788003417 0.3788885551 0.3788954125 0.3788968451 0.3788971887 0.3788972737 0.3788972949 0.3788973002
0.3788937288 0.3789248578 0.378927278  0.3789277838 0.3789279051 0.3789279351 0.3789279426 0.3789279445 0.378927945
0.3789266935 0.3789376817 0.3789385366 0.3789387153 0.3789387582 0.3789387688 0.3789387714 0.3789387721 0.3789387723 0.3789387723
0.3789383338 0.3789422139 0.378942516  0.3789425792 0.3789425944 0.3789425981 0.378942599  0.3789425992 0.3789425993 0.3789425993 0.3789425993
```

$$0.3789425993$$

Bei einer modifizierten Variante der Prozedur mit maximaler Zeilenanzahl des Romberg-Schemas und zusätzlicher Abbruchbedingung bei Unterschreitung einer Toleranz 10^{-6}

```
> # ohne Ausgabehalt, ohne Tableau
  tol:=1e-6:
  pp:=Romberg2_o(true,m,a,b,tol,iter,IWert):

> iter;
  IWert;
  eval(pp):  # letzte Zeile des Romberg-Schemas als Liste
             # in umgekehrter Reihenfolge
```

werden nur 6 Zeilen des Schemas gerechnet.
Man erhält die Datei `tt2.dat` sowie die Ausgabe

```
0.1839397206
0.3064108315 0.3472345351
0.3528258723 0.3682975526 0.3697017538
0.3696612874 0.3752730924 0.3757381284 0.3758339439
0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.378563508  0.3785641936
```

$$5$$

$$0.3785641936$$

$$[\ 0.3785641936,\ 0.3785635080,\ 0.3785607372,\ 0.3785491807,0.3784937239,\ 0.3777857243\ ]$$

Das Kommando `readline` ist eigentlich zum zeilenweisen Lesen von Textdateien vorgesehen.
Der normale Fall wäre die folgende Prozedur mit entsprechendem Aufruf (Verwendung der Datei `tt2.dat`) und Ergebnis.

```
> printtext:=proc(x::string)
   local line;
   line:=readline(x):
   while line<>0 do
    printf("%s\n",line):
    line:=readline(x):
   end do:
  end proc:

> printtext("C:/D/Neundorf/Maple5/tt2.txt"):
```

```
0.1839397206
0.3064108315 0.3472345351
0.3528258723 0.3682975526 0.3697017538
0.3696612874 0.3752730924 0.3757381284 0.3758339439
0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.378563508  0.3785641936
```

Zum Schluss machen wir noch einige Bemerkungen zum Kommando `readstat`.
Es dient allgemein zum Einlesen von Anweisungen bei entsprechender Aufforderung, kann aber auch zum Anhalten der Abarbeitung genutzt werden. Zu letzterem sollte der Dialog natürlich möglichst sparsam sein.

Beginnen wir mit der einfachsten Version ohne Eingabeaufforderung.

```
> readstat();           # Eingabe mit ; oder : abschliessen
```

Der Befehl erwartet eine Eingabe. Damit ist die Abarbeitung angehalten.
Nach dieser Maple-Zelle sollte eine leere Zelle zur Verfügung stehen. Dort gibt man ; & <Enter> bzw. : & <Enter> ein. Und die Abarbeitung wird fortgesetzt.

Etwas eleganter ist die Version mit Eingabeaufforderung (`prompt`).
Auch hier ist ein nachfolgende leere Maple-Zelle erforderlich.

```
> readstat("z="):
```

In der nachfolgenden Maple-Zelle kommt die Aufforderung `z=`.
Man gibt z. B. eine Zahl (hier 11) abgeschlossen mit ; oder : ein.
Steht nach `readstat("z=")` ein Semikolon statt Doppelpunkt, so ist das Ergebnis und die Ausgabe

11

Sicher brauchbarer ist die Variante mit Variablenzuweisung und Eingabeaufforderung, wiederum bei nachfolgender leerer Maple-Zelle.

```
> z:=readstat("z=");
```

In der nachfolgenden Maple-Zelle kommt die Aufforderung `z=`.
Man gibt z. B. eine Zahl (hier 22) abgeschlossen mit ; oder : ein. Das Ergebnis/Ausgabe ist

$$z{:=}22$$

so dass man mit der Variablen z auch weiter arbeiten kann.

Die Eingabe kann auch ein Ausdruck sein, was abschließend notiert werden soll.

```
> s:=readstat("Summe s=");
```

In der nachfolgenden Maple-Zelle kommt die Aufforderung `Summe s=`.
Man gibt z. B. einen Ausdruck ein, hier `1+sqrt(7)+sin(1)` abgeschlossen mit Semikolon oder Doppelpunkt. Das Ergebnis/Ausgabe ist

$$s := 1 + \sqrt{7} + \sin(1)$$

Die Auswertung `evalf(s)` liefert den Zahlenwert 4.487222296.

Hat man viele Ausgabehalts wie bei der Ausgabe von Zeilen eines Tableaus, muss man dafür sorgen, dass entsprechend viele leere Maple-Zellen vorhanden sind.
Die Darstellung selbst ist natürlich auch nicht so "angenehm", weil sie durch die Eingabe von Zeilen mit Semikolon bzw. Doppelpunkt unterbrochen ist.

```
> Romberg2_e:=proc(zw::boolean,m::posint,a::numeric,b::numeric,
                   tol::numeric,iter::evaln,
                   IWert::evaln)
   local h,ha,ah,s,q,    # numeric
         i,k,n,          # integer
         p,              # array
         td;             # Textdatei
   global f,             # Integrand
          datei0;

   td:=fopen(datei0,WRITE,TEXT):

   p:=array(sparse,0..m);

   h:=b-a;
   n:=1;
   p[0]:=evalf(0.5*h*(f(a)+f(b)));
   if zw then fprintf(default,`%14.10f\n`,p[0]); end if;
   fprintf(td,`%14.10f\n`,p[0]);

   readstat():

   for k from 1 to m do
     s:=0;
     ha:=h; h:=0.5*h;
     ah:=a-h;
     n:=2*n;
     q:=1;
     for i from 1 to n/2 do s:=s+f(ah+i*ha); end do;
     p[k]:=evalf(0.5*p[k-1]+s*h);
     if zw then fprintf(default,`%14.10f `,p[k]); end if;
     fprintf(td,`%14.10f `,p[k]);
     for i from k-1 by -1 to 0 do
       q:=q*4;
       p[i]:=evalf(p[i+1]+(p[i+1]-p[i])/(q-1));
       if zw then fprintf(default,`%14.10f `,p[i]); end if;
       fprintf(td,`%14.10f `,p[i]);
     end do;
     fprintf(default,`\n`);
     fprintf(td,`\n`);
```

```
    readstat():

    if abs(p[0]-p[1])<tol then break; end if;
  end do;

  fclose(td);
  # lprint(‘<Ausgabe Ende>‘);

  iter:=k;
  IWert:=p[0];
  [seq(p[i],i=0..k)];
 end proc:
```

Nun der Aufruf der Prozedur.

```
  > # mit Ausgabehalt, hinreichend viele leere Zellen vorbereiten,
    # Eingabeabschluss mit ; bzw. :
    tol:=1e-6:
    lprint(‘Bei Ausgabehalt :<Enter> bzw. ;<Enter>‘);
    pp:=Romberg2_e(true,m,a,b,tol,iter,IWert):
```

```
‘Bei Ausgabehalt :<Enter> bzw. ;<Enter>‘
  0.1839397206
;
  0.3064108315 0.3472345351
;
  0.3528258723 0.3682975526 0.3697017538
;
  0.3696612874 0.3752730924 0.3757381284 0.3758339439
:
  0.3756617255 0.3776618715 0.3778211234 0.3778541868 0.3778621093
:
  0.3777857243 0.3784937239 0.3785491807 0.3785607372 0.3785635080 0.3785641936
:
```

Das erinnert uns an die ähnliche Situation in Classic Maple mit Ausgabehalt wegen `readline()` und $m + 1$ leeren Maple-Zellen.

Kapitel 3

Testmatrizen mit Eigenschaften

3.1 Matrix 1

Gegeben sei die Matrix

$$A = \begin{pmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{pmatrix} = \begin{pmatrix} \frac{39}{50} & \frac{563}{1000} \\ \frac{913}{1000} & \frac{659}{1000} \end{pmatrix}.$$

Ihre Eigenwerte (EW) sind

$$\lambda_{1,2} = \frac{1439}{2000} \pm \frac{\sqrt{2070717}}{2000} = 1.4389993050726317415,\ 0.69492736825854 \cdot 10^{-6} > 0,$$

die zugehörigen linear unabhängigen skalierten EV

$$\begin{aligned} \tilde{v}_1 &= \Big(1, -\frac{121}{1126} + \frac{\sqrt{2070717}}{1126}\Big)^T = (1,\ 1.1705138633616904822)^T, \\ \tilde{v}_2 &= \Big(1, -\frac{121}{1126} - \frac{\sqrt{2070717}}{1126}\Big)^T = (1, -1.3854339344096478534)^T. \end{aligned}$$

Nach Normierung mittels euklidischer Norm sind die Eigenvektoren (EV)

$$\begin{aligned} v_1 &= c_1\Big(1, -\frac{121}{1126} + \frac{\sqrt{2070717}}{1126}\Big)^T = \begin{pmatrix} 0.64955572831439685710 \\ 0.76031398501800126831 \end{pmatrix}, \\ & \qquad c_1 = \Big[1 + \Big(-\frac{121}{1126} + \frac{\sqrt{2070717}}{1126}\Big)^2\Big]^{-1/2}, \\ v_2 &= c_2\Big(1, -\frac{121}{1126} - \frac{\sqrt{2070717}}{1126}\Big)^T = \begin{pmatrix} 0.58526314402966098506 \\ -0.81084342029797362184 \end{pmatrix}, \\ & \qquad c_2 = \Big[1 + \Big(-\frac{121}{1126} - \frac{\sqrt{2070717}}{1126}\Big)^2\Big]^{-1/2}. \end{aligned}$$

Da die EV bis auf einen konstanten Faktor eindeutig bestimmt sind, kann das Ergebnis von Maple mit `eigenvectors` auch folgendes sein.

$$\begin{aligned}\hat{v}_1 &= \Big(\frac{11}{166}+\frac{\sqrt{2070717}}{1826},1\Big)^T = (\ 0.85432563534789895011,1)^T,\\ \hat{v}_2 &= \Big(\frac{11}{166}-\frac{\sqrt{2070717}}{1826},1\Big)^T = (-0.72179551486597123927,1)^T.\end{aligned}$$

Nach Normierung mittels euklidischer Norm sind hier die EV

$$\begin{aligned}v_1 &= c_1\Big(\frac{11}{166}+\frac{\sqrt{2070717}}{1826},1\Big)^T = \begin{pmatrix} 0.64955572831439685713\\ 0.76031398501800126832\end{pmatrix},\\ c_1 &= \Big[1+\Big(\frac{11}{166}+\frac{\sqrt{2070717}}{1826}\Big)^2\Big]^{-1/2},\\ v_2 &= c_2\Big(\frac{11}{166}-\frac{\sqrt{2070717}}{1826},1\Big)^T = \begin{pmatrix} -0.58526314402966098506\\ 0.81084342029797362183\end{pmatrix},\\ c_2 &= \Big[1+\Big(\frac{11}{166}-\frac{\sqrt{2070717}}{1826}\Big)^2\Big]^{-1/2}.\end{aligned}$$

Wir machen nun die Orthogonaltransformation von A auf eine obere Dreiecksmatrix R gemäß

$$Q^TAQ = R,\ \ Q=(q_1,q_2),\ Q^TQ=I,\ R=\begin{pmatrix} r_{11} & r_{12}\\ 0 & r_{22}\end{pmatrix}.$$

I ist dabei die Einheitsmatrix. Wegen der Ähnlichkeit von A und R ist $r_{11}=\lambda_1$ und $r_{22}=\lambda_2$.
Wegen $(q_1,q_2)R = A(q_1,q_2)$ kann man die erste Spalte von Q zu $q_1=v_1$ wählen.
Als Vektor $q_2 \perp q_1$ nimmt man einfach $q_2=(q_{1,2},-q_{1,1})^T,\ \|q_2\|_2=1$.
Aus $r_{12}q_1+r_{22}q_2 = Aq_2$ und $q_1^Tq_2=0$ folgt zunächst $r_{12}=q_1^TAq_2=0.35$, gleichzeitig aber auch $r_{22}=q_2^TAq_2=q_2^TA^Tq_2$, was zur Kontrolle der Übereinstimmung mit λ_2 dienen kann. Somit erhält man

$$\begin{aligned}Q &= \begin{pmatrix} 0.64955572831439685710 & 0.76031398501800126831\\ 0.76031398501800126831 & -0.64955572831439685710\end{pmatrix},\\ R &= \begin{pmatrix} 1.4389993050726317415 & 0.35\\ 0 & 0.69492736825854\cdot 10^{-6}\end{pmatrix}.\end{aligned}$$

Die Matrix A hat eine schlechte Kondition. Kennzeichen dafür sind unter anderem:

- Die Matrix A ist fast singulär.
- Die Determinante von A ist nahe Null, denn $\det(A)=10^{-6}$.

– Wenn die Matrix A Elemente der Größenordnung $\mathcal{O}(1)$ besitzt, dann hat die inverse Matrix A^{-1} betragsmäßig große Elemente, hier

$$A^{-1} = \begin{pmatrix} 659000 & -563000 \\ -913000 & 780000 \end{pmatrix}, \quad \det(A^{-1}) = 10^6.$$

– Das Spektrum der Eigenwerte von A ist sehr "breit". Es liegen Größenordnungen zwischen dem betragsmäßig kleinsten ($\neq 0$) und größten Eigenwert, denn sie betragen hier

$$\begin{aligned} \lambda_1 &= 0.000\,000\,694\,927\,368, \\ \lambda_2 &= 1.438\,999\,305\,072\,632. \end{aligned}$$

– Eine einfache geometrische Interpretation ist die Charakterisierung der Lösung des LGS $Ax = b$ als Schnittpunkt zweier Geraden, die sich hier in einem extrem spitzen Winkel schneiden.

Mit der Zeilensummennorm $\|A\|_\infty$ beträgt die Kondition

$$\mathrm{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty = 1.572 \cdot 1693000 \approx 2.661 \cdot 10^6,$$

mit der Spektralnorm $\|A\|_2$ beträgt die Kondition

$$\mathrm{cond}_2(A) = \|A\|_2 \|A^{-1}\|_2 = 1.480952059 \cdot 1480952.059 \approx 2.193218999999 \cdot 10^6.$$

Der Exponent $t = 6$ in der Näherung des Konditionswerts charakterisiert im Groben die Reduzierung der gültigen Mantissenstellen der in Rechnungen benutzten GPA bei Lösung des LGS $Ax = b$.

Die Matrix A ist nicht positiv definit, denn mit $x = (-1, 1)^T$ gilt $x^T Ax = -0.037 < 0$. A ist indefinit.

Einige Rechnungen in Maple mit der Matrix A (Datei *matrix2x2.mws*)

```
> restart:
  with(linalg):
```

Definition des LGS mit Matrix A (`float`-Zahlen bzw. symbolisch)

```
> Digits:=10:
  II:=evalm(array(identity,1..2,1..2)):
  A:=matrix(2,2,[0.780,0.563,0.913,0.659]);
  AA:=matrix(2,2,(i,j)->convert(A[i,j],rational));
  rank(A);
  det(A);
  norm(A);
  inverse(A);
  inverse(AA);

  b:=vector([0.217,0.254]);
  bb:=vector(n,[217/1000,254/1000]):
  xs:=vector([1,-1]);
 ‘Ax*-b‘=evalm(A&*xs-b);
```

$$A := \begin{bmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{bmatrix}$$

$$AA := \begin{bmatrix} \frac{39}{50} & \frac{563}{1000} \\ \frac{913}{1000} & \frac{659}{1000} \end{bmatrix}$$

$$2$$

$$0.1\,10^{-5}$$

$$1.572$$

$$\begin{bmatrix} 659000.0000 & -563000.0000 \\ -913000.0000 & 780000.0000 \end{bmatrix}$$

$$\begin{bmatrix} 659000 & -563000 \\ -913000 & 780000 \end{bmatrix}$$

$$b := [0.217,\ 0.254]$$

$$xs := [1,\ -1]$$

$$Ax^* - b = [0.,\ 0.]$$

Rechnung bei schwacher GPA mit unbefriedigenden Ergebnissen

```
> Digits:=5:
  evalm(A);
  rank(A);
  det(A);
  norm(A);
  inverse(A);
```

$$\begin{bmatrix} 0.780 & 0.563 \\ 0.913 & 0.659 \end{bmatrix}$$

$$1$$

$$0.$$

$$1.572$$

```
Error, (in inverse) singular matrix

> Digits:=10:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 659000.0000 & -563000.0000 \\ -913000.0000 & 780000.0000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

```
> Digits:=16:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 659000.0000000000 & -563000.0000000000 \\ -913000.0000000000 & 780000.0000000000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

Die inverse Matrix ist exakt in der geforderten Genauigkeit der Mantissenlänge.

Test von zwei Lösungskandidaten und Berechnung der Residuen dazu

```
> Digits:=16;
  xq:=vector([0.341,-0.087]);
  xd:=vector([0.999,-1.001]);
  multiply(A,xq);
  rq:=evalm(b-multiply(A,xq));
  rd:=evalm(b-A&*xd);
```

$$Digits := 16$$
$$xq := [0.341,\ -0.087]$$
$$xd := [0.999,\ -1.001]$$
$$[0.216999,\ 0.254000]$$
$$rq := [0.1\,10^{-5},\ 0.]$$
$$rd := [0.001343,\ 0.001572]$$

Lösung des LGS mit GPA bei unterschiedlicher Mantissenlänge

```
> Digits:=16:
  erg_exakt:=linsolve(AA,bb);
  erg:=linsolve(A,b);
  erg[1]; erg[2];
```

```
> i:='i':
  printf(`Digits         x[1]                    x[2]\\n`):
  for i from 5 to 25 do
   Digits:=i:
   erg:=linsolve(A,b):
   if rank(A)<2 then
    printf(`%2d       Matrix A singulaer\n`,i)
   else
    printf(`%2d       %28.25f  %28.25f\n`,i,erg[1],erg[2])
   end if:
  end do:
```

$$erg\_exakt := [1,\ -1]$$
$$erg := [1.000000000000000,\ -1.000000000000000]$$
$$1.000000000000000$$
$$-1.000000000000000$$

```
Digits                x[1]                              x[2]
 5       Matrix A singulaer
 6       Matrix A singulaer
 7       Matrix A singulaer
 8        0.974221610000000000000000  -0.964285710000000000000000
 9        1.000659780000000000000000  -1.000914080000000000000000
10        0.999934106700000000000000  -0.999908709200000000000000
11        1.000013180100000000000000  -1.000018260200000000000000
12        0.999998023006000000000000  -0.999997261004000000000000
13        0.999999868200200000000000  -0.999999817400000000000000
14        0.999999980230040000000000  -0.999999972610000000000000
15        0.999999999341002000000000  -0.999999999087000000000000
```

```
 16        1.00000000000000000000000000  -1.00000000000000000000000000
 17        1.00000000013180000000000000  -1.00000000001826000000000000
 18        1.00000000000065900000000000  -1.00000000000091300000000000
 19        0.99999999999993410010000000  -0.99999999999990870000000000
 20        0.99999999999998023004000000  -0.99999999999997261000000000
 21        1.00000000000000197700000000  -1.00000000000000273900000000
 22        1.00000000000000013180000000  -1.00000000000000018260000000
 23        1.00000000000000001977000000  -1.00000000000000002739000000
 24        0.99999999999999999868200300  -0.99999999999999999817400000
 25        1.00000000000000000019770000  -1.00000000000000000027390000
```

Es ist bemerkenswert, wie sich die Genauigkeit `Digits=16` deutlich gegenüber den anderen hervorhebt. Es ist anzunehmen, dass in diesem Fall die Ausführung der arithmetischen Operationen im Prozessor besonders effizient und genau erfolgt. In der Nachbarschaft `Digits=15,17` sind nur 9–10 Nachkommastellen der Näherungslösung genau.

Wir führen nun analoge Anweisungen in MATLAB mit seiner GPA *double* aus.
Die Ergebnisse sollen dann mit Maple-Rechnungen in der Genauigkeit `Digits=16`, gegebenenfalls auch `Digits=10`, verglichen werden.

```
% matrix2x2.m
% Test zu schlecht konditionierten Matrizen
diary matrix2x2.txt
diary off
echo off
clear
clc

format compact
format long
n=2;           % n=input('n=')
II=eye(n);     % eye(n,n)

% Matrix 1
disp('Matrix 1')
A =[0.780 0.563; 0.913 0.659]
AA=[39/50 563/1000; 913/1000 659/1000]  % wie A
b =[0.217 0.254]'
xs=[1,-1]'    % exakte Loesung
A*xs-b         % Kontrolle
det(A)                              % =det(AA)
detA=A(1,1)*A(2,2)-A(2,1)*A(1,2) % genauer bzw. =det(A)

invA=inv(A)
A*invA
II-A*invA
% Inverse per Hand
invAs=[A(2,2) -A(1,2); -A(2,1) A(1,1)]/det(A)
A*invAs
II-A*invAs
% etwas besser wegen genauerem detA
```

```
invAs_=[A(2,2) -A(1,2); -A(2,1) A(1,1)]/detA
A*invAs_
II-A*invAs_

% Norm, Kondition
norm2_A=norm(A)        % Spektralnorm, norm(A,2)
cond2_A=cond(A)
% Test von zwei Loesungskandidaten und Berechnung der Residuen dazu:
% schlechter Kandidat hat kleineres Residuum
xq=[0.341 -0.087]'        % schlechter Kandidat
xd=[0.999 -1.001]'
rq=A*xq-b
rd=A*xd-b
% LGS
lgs_erg1=A\b
lgs_erg2=linsolve(A,b)         % ungenauer als A\b
lgs_erg3=maple('linsolve',A,b) % exakt
pause
```

Zunächst die Ergebnisse.

```
Matrix 1
A =
   0.78000000000000   0.56300000000000
   0.91300000000000   0.65900000000000
AA =
   0.78000000000000   0.56300000000000
   0.91300000000000   0.65900000000000
b =
   0.21700000000000
   0.25400000000000
xs =
     1
    -1
ans =
  1.0e-016 *
   0.83266726846887
                  0
ans =
    1.000000000122681e-006
detA =
    1.000000000028756e-006
invA =
  1.0e+005 *
   6.58999999919154  -5.62999999930931
  -9.12999999887993   7.79999999904309
ans =
   0.99999999994179                  0
                  0   0.99999999988358
ans =
  1.0e-009 *
   0.05820766091347                  0
                  0   0.11641532182693
invAs =
  1.0e+005 *
   6.58999999919154  -5.62999999930931
  -9.12999999887993   7.79999999904309
ans =
   0.99999999994179                  0
                  0   0.99999999988358
ans =
```

```
  1.0e-009 *
   0.05820766091347                  0
                  0   0.11641532182693
invAs_ =
  1.0e+005 *
   6.58999999981050  -5.62999999983810
  -9.12999999973746   7.79999999977571
ans =
   1.00000000000000                  0
                  0   1.00000000005821
ans =
  1.0e-010 *
                  0                  0
                  0  -0.58207660913467
norm2_A =
   1.48095205864320
cond2_A =
    2.193218999650770e+006
xq =
   0.34100000000000
  -0.08700000000000
xd =
   0.99900000000000
  -1.00100000000000
rq =
  1.0e-006 *
  -0.99999999994549
                  0
rd =
  -0.00134300000000
  -0.00157200000000

lgs_erg1 =
   0.99999999998171
  -0.99999999997466
lgs_erg2 =
   0.99999999989025
  -0.99999999984795
lgs_erg3 =
matrix([[1], [-1]])
```

Einige Bemerkungen

– Bei numerischen Rechnungen in MATLAB werden von den 16 angezeigten Mantissenstellen wegen der schlechten Matrixkondition die letzten 4–5 Stellen im Allgemeinen falsch.

– Der Wert der Spektralkondition der Matrix `cond2_A=2.193218999650770e+006` hat nur 10 genaue Mantissenstellen.

– Die Handrechnung der Determinante für diese kleine Matrix ist eine Dezimalstelle genauer als der Befehl `det(A)`. Damit kann dann auch die Inverse etwas genauer werden und die Fehlermatrix `II-A*invAs_` ist "kleiner".

– Beim Test der beiden Lösungskandidaten für das LGS ist der Genauigkeitsverlust zumindest bei der ersten Komponente von `rq` sichtbar.

– Die Lösung des LGS in den 3 Varianten zeigt den Vorteil des Kommandos `A\b`. Dass der in MATLAB aufgerufene Maple-Befehl `linsolve` die exakte Lösung ausgibt, mag damit zusammenhängen, dass die Maple-Rechnung bei `Digits=16` so genau ist.

3.2 Matrix 2

Gegeben sei die Matrix

$$A = \begin{pmatrix} -29\,998 & -39\,996 \\ 29\,997/2 & 19\,997 \end{pmatrix}.$$

Sie ist die Matrix eines steifen linearen Systems gewöhnlicher Differentialgleichungen. Das Anfangswertproblem dazu für $\mathbf{x} = (x, y)^T$ ist

$$\mathbf{x}'(t) = A\,\mathbf{x}(t), \quad \mathbf{x}(0) = (1, 1)^T, \ t \geq 0.$$

Die allgemeine Lösung ist

$$x(t) = C_1\,e^{-t} + C_2\,e^{-10000\,t}, \quad y(t) = -\frac{3}{4}C_1\,e^{-t} - \frac{1}{2}C_2\,e^{-10000\,t},$$

die des Anfangswertproblems mit den gegebenen Anfangsbedingungen

$$x(t) = -6\,e^{-t} + 7\,e^{-10000\,t}, \quad y(t) = \frac{9}{2}\,e^{-t} - \frac{7}{2}\,e^{-10000\,t}.$$

Ihre EW sind also

$$\lambda_1 = -1, \quad \lambda_2 = -10\,000,$$

die zugehörigen linear unabhängigen skalierten bzw. normierten EV können sein

$$\begin{aligned} v_1 &= (-4/3, 1)^T, \quad (1, -3/4)^T, \quad (4/5, -3/5)^T, \\ v_2 &= (-2, 1)^T, \quad\ \ (1, -1/2)^T, \quad (2/\sqrt{5}, -1/\sqrt{5})^T. \end{aligned}$$

Die Determinante von A ist $\det(A) = \prod_{i=1}^{2} \lambda_i = 10^4$.
Da die Matrix A Elemente der Größenordnung $\mathcal{O}(10^4)$ besitzt, dann hat die inverse Matrix A^{-1} betragsmäßig moderate Elemente, hier

$$A^{-1} = \begin{pmatrix} 1.9997 & 3.9996 \\ -1.49985 & -2.9998 \end{pmatrix}, \quad \det(A^{-1}) = 10^{-4}.$$

Mit der Zeilensummennorm $\|A\|_\infty$ beträgt die Kondition

$$\mathrm{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty = 69994 \cdot 5.9993 \approx 4.199 \cdot 10^5,$$

mit der Spektralnorm $\|A\|_2$ beträgt die Kondition

$$\mathrm{cond}_2(A) = \|A\|_2 \|A^{-1}\|_2 = 55896.28817 \cdot 5.589628817 \approx 3.124 \cdot 10^5$$

Zahlreiche Eigenschaften der Matrix, wie sie in Abschnitt 3.1 genannt wurden, sind auch hier feststellbar.

Einige Rechnungen in Maple mit der Matrix A (Datei *matrix2x2.mws*)

```
> restart:
  with(linalg):
```

Definition des LGS mit Matrix A (`float`-Zahlen bzw. symbolisch)

```
> Digits:=10:
  A:=matrix(2,2,[-29998.0, -39996.0, 14998.5, 19997.0]);
  AA:=matrix(2,2,[-29998, -39996, 29997/2, 19997]);

  rank(A);
  det(A);
  norm(A);
  inverse(A);
  inverse(AA);

  b:=vector([9998,-4998.5]);
  bb:=vector([9998,-9997/2]);
  xs:=vector([1,-1]);
 `Ax*-b`=evalm(A&*xs-b);
```

$$A := \begin{bmatrix} -29998.0 & -39996.0 \\ 14998.50000 & 19997.0 \end{bmatrix}$$

$$AA := \begin{bmatrix} -29998 & -39996 \\ \frac{29997}{2} & 19997 \end{bmatrix}$$

$$2$$

$$10000.0$$

$$69994.0$$

$$\begin{bmatrix} 1.999700000 & 3.999600000 \\ -1.499850000 & -2.999800000 \end{bmatrix}$$

$$\begin{bmatrix} \frac{19997}{10000} & \frac{9999}{2500} \\ \frac{-29997}{20000} & \frac{-14999}{5000} \end{bmatrix}$$

$$b := [9998,\ -4998.5]$$

$$bb := \left[9998,\ \frac{-9997}{2}\right]$$

$$xs := [1,\ -1]$$

$$Ax^* - b = [0.,\ 0.]$$

Rechnung bei schwacher GPA mit unbefriedigenden Ergebnissen

```
> Digits:=5:
  evalm(A);
  rank(A);
  det(A);
```

```
norm(A);
inverse(A);
det(inverse(A));
```

$$\begin{bmatrix} -29998. & -39996. \\ 14998. & 19997. \end{bmatrix}$$

$$1$$

$$-10000.$$

$$69994.$$

$$\begin{bmatrix} -1.9997 & -3.9996 \\ 1.4998 & 2.9998 \end{bmatrix}$$

$$-0.0001$$

```
> Digits:=10:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 1.999700000 & 3.999600000 \\ -1.499850000 & -2.999800000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

```
> Digits:=16:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 1.999700000000000 & 3.999600000000000 \\ -1.499850000000000 & -2.999800000000000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

Die inverse Matrix ist exakt in der geforderten Genauigkeit der Mantissenlänge.

Test von zwei Lösungskandidaten und Berechnung der Residuen dazu

```
> Digits:=16;
  xq:=vector([0,-0.249975]);
  xd:=vector([0.999,-1.001]);
  multiply(A,xq);
  rq:=evalm(b-multiply(A,xq));
  rd:=evalm(b-A&*xd);
```

$$Digits := 16$$

$$xq := [0,\ -0.249975]$$

$$xd := [0.999,\ -1.001]$$

$$[9998.0001000,\ -4998.7500750]$$

$$rq := [0.0001000,\ -0.2500750]$$

$$rd := [69.9940,\ -34.99550000]$$

Lösung des LGS mit GPA bei unterschiedlicher Mantissenlänge

```
> Digits:=16:
  erg_exakt:=linsolve(AA,bb);
  erg:=linsolve(A,b);
  erg[1]; erg[2];

> i:='i':
  printf(`Digits         x[1]                    x[2]\\n`):
  for i from 5 to 25 do
   Digits:=i:
   erg:=linsolve(A,b):
   if rank(A)<2 then
    printf(`%2d       Matrix A singulaer\n`,i)
   else
    printf(`%2d       %28.25f  %28.25f\n`,i,erg[1],erg[2])
   end if:
  end do:
```

$$erg\_exakt := [1, -1]$$

$$erg := [1.000000000004000, -1.000000000003000]$$

$$1.000000000004000$$

$$-1.000000000003000$$

```
Digits                x[1]                              x[2]
 5       Matrix A singulaer
 6       Matrix A singulaer
 7        1.000000000000000000000000  -1.000000000000000000000000
 8        1.000400000000000000000000  -1.000300000000000000000000
 9        0.999680128000000000000000  -0.999760086000000000000000
10        0.999980002500000000000000  -0.999985001400000000000000
11        1.000000000000000000000000  -1.000000000000000000000000
12        1.000000400000000000000000  -1.000000300000000000000000
13        0.999999968003200000000000  -0.999999976001400000000000
14        0.999999998000200000000000  -0.999999998500100000000000
15        1.000000000000000000000000  -1.000000000000000000000000
16        1.000000000040000000000000  -1.000000000030000000000000
17        0.999999999968003200000000  -0.999999999976001400000000
18        0.999999999998000200000000  -0.999999999998500100000000
19        1.000000000000000000000000  -1.000000000000000000000000
20        1.000000000000004000000000  -1.000000000000003000000000
21        0.999999999999996800320000  -0.999999999999997600140000
22        0.999999999999999800020000  -0.999999999999999850010000
23        1.000000000000000000000000  -1.000000000000000000000000
24        1.000000000000000000400000  -1.000000000000000000300000
25        0.999999999999999999680032  -0.999999999999999999760014
26        0.999999999999999999980002  -0.999999999999999999985001
27        1.000000000000000000000000  -1.000000000000000000000000
28        1.000000000000000000000040  -1.000000000000000000000030
29        0.999999999999999999999968  -0.999999999999999999999976
30        0.999999999999999999999998  -0.999999999999999999999998
```

Es ist schon bemerkenswert, wie sich die Genauigkeiten `Digits=7,11,15,19,23,27` deutlich gegenüber den anderen hervorhebt. Dabei ist sogar ein fester Abstand zwischen den Formaten sichtbar.

Es ist anzunehmen, dass in diesem Fall die Ausführung der arithmetischen Operationen im Prozessor besonders effizient und genau erfolgt.
In der Nachbarschaft (davor und dahinter) von `Digits=i`, $i = 11, 15, 19, 23, 27$ sind nur ca. $i - 7$ bzw. $i - 3$ Nachkommastellen der Näherungslösung genau.

Wir führen nun analoge Anweisungen in MATLAB mit seiner GPA *double* aus und vergleichen dann mit Maple-Rechnungen in der Genauigkeit `Digits=16`.

```
% matrix2x2.m

% Matrix 2
disp('Matrix 2')
% Test zur Systemmatrix fuer steifes System gDGL
A =[-29998 -39996; 14998.5 19997]
AA=[-29998 -39996; 29997/2 19997];
b =[9998 -4998.5]'
xs=[1,-1]'          % exakte Loesung
A*xs-b              % Kontrolle
...
```

Zunächst die Ergebnisse.

```
Matrix 2
A =
  1.0e+004 *
  -2.99980000000000  -3.99960000000000
   1.49985000000000   1.99970000000000
b =
  1.0e+003 *
   9.99800000000000
  -4.99850000000000
xs =
     1
    -1
ans =
     0
     0
ans =
    9.999999999943349e+003
detA =
       10000

invA =
   1.99970000001133   3.99960000002266
  -1.49985000000850  -2.99980000001699
ans =
   0.99999999999272                  0
   0.00000000000728   1.00000000000728
ans =
  1.0e-011 *
   0.72759576141834                  0
  -0.72759576141834  -0.72759576141834

invAs =
   1.99970000001133   3.99960000002266
  -1.49985000000850  -2.99980000001699
ans =
   1.00000000000000                  0
   0.00000000000364   1.00000000000728
```

```
ans =
  1.0e-011 *
                  0                   0
  -0.36379788070917  -0.72759576141834

invAs_ =
   1.99970000000000   3.99960000000000
  -1.49985000000000  -2.99980000000000
ans =
     1     0
     0     1
ans =
     0     0
     0     0

normA =
    5.589628817030694e+004
condA =
    3.124395031180386e+005

xq =
                  0
  -0.24997500000000
xd =
   0.99900000000000
  -1.00100000000000
rq =
   0.00009999999929
  -0.25007499999992
rd =
  69.99399999999878
 -34.99549999999908

lgs_erg1 =
   1.00000000000728
  -1.00000000000546
lgs_erg2 =
   1.00000000000728
  -1.00000000000546
lgs_erg3 =
matrix([[1], [-1]])
```

Einige Bemerkungen

– Bei numerischen Rechnungen in MATLAB werden von den 16 angezeigten Mantissenstellen wegen der Matrixkondition die letzten 3–5 Stellen i. Allg. falsch.

– Der Wert der Spektralkondition der Matrix `cond2_A=3.124395031180386e+005` hat nur 11 genaue Mantissenstellen.

– Die Handrechnung der Determinante für diese kleine Matrix ist exakt und damit 5 Dezimalstellen genauer als der Befehl `det(A)`. Damit wird auch die Inverse genau und die Fehlermatrix `II-A*invAs_=0`.

– Beim Test der beiden Lösungskandidaten für das LGS ist der Genauigkeitsverlust nicht ganz so drastisch (Abweichungen in den letzten 1–3 Stellen).

– Die Lösung des LGS in den 3 Varianten zeigt `A\b=linsolve(A,b)`.
Dass der in MATLAB aufgerufene Maple-Befehl `linsolve` die exakte Lösung ausgibt, mag damit zusammenhängen, dass die Maple-Rechnung bei `Digits=15` genau ist, obwohl in der Nachbarschaft `Digits=14,16` Abweichungen auftreten.

3.3 Matrix 3

Gegeben sei die Matrix

$$A = \begin{pmatrix} 2 & 6 \\ 2 & 6.00001 \end{pmatrix}.$$

Ihre EW sind

$$\lambda_1 = \frac{800001}{200000} + \frac{640000800001}{200000} = 8.000007500002343748535,$$

$$\lambda_2 = \frac{800001}{200000} - \frac{640000800001}{200000} = 0.000002499997656251465.$$

Die zugehörigen linear unabhängigen skalierten EV sind

$$\begin{aligned} v_1 &= \left(1, \frac{400001 + \sqrt{640000800001}}{1200000}\right)^T = (1,\ 1.000001250000390624756)^T, \\ v_2 &= \left(1, \frac{400001 - \sqrt{640000800001}}{1200000}\right)^T = (1, -0.333332916667057291423)^T. \end{aligned}$$

Die Determinante von A ist $\det(A) = \prod_{i=1}^{2} \lambda_i = 0.00002$.
Da die Matrix A Elemente der Größenordnung $\mathcal{O}(1)$ besitzt, dann hat die inverse Matrix A^{-1} betragsmäßig große Elemente, hier

$$A^{-1} = \begin{pmatrix} 300000.5 & -300000 \\ -100000 & 100000 \end{pmatrix}, \quad \det(A^{-1}) = 50000.$$

Mit der Zeilensummennorm $\|A\|_\infty$ beträgt die Kondition

$$\mathrm{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty = 8.00001 \cdot 600000.5 \approx 4.80001 \cdot 10^6,$$

mit der Spektralnorm $\|A\|_2$ beträgt die Kondition

$$\mathrm{cond}_2(A) = \|A\|_2 \|A^{-1}\|_2 = 8.94428 \cdot 447213.9309 \approx 4.000006001 \cdot 10^6$$

Für die Matrix

$$C = AA^T = \begin{pmatrix} 40 & \frac{2000003}{50000} \\ \frac{2000003}{50000} & \frac{400001200001}{10000000000} \end{pmatrix} = \begin{pmatrix} 40 & 40.000006 \\ 40.000006 & 40.0001200001 \end{pmatrix}$$

liefert Maple mit der Genauigkeit `Digits=10` das Spektrum $\sigma(C) = \{0.,\ 80.00012001\}$. D. h. jedoch, dass das Format nicht ausreicht, denn der EW Null ist natürlich eine numerische Näherung. Mit `Digits=20` erhält man das Spektrum

$$\sigma(C) = \{0.4999993 \cdot 10^{-11},\ 80.000120000095000009\}.$$

Wir haben also die Situation einer fast singulären Matrix A mit einem EW fast Null. Durch das Matrixprodukt $C = (c_{ij}) = AA^T$ wird die Kondition eher schlechter. Der EW nahe Null wird noch kleiner. Ein weiterer EW liegt aber in der Größenordnung von $\sum_j |c_{ij}|$.
Bei ausgewogenen Matrixelementen von A ist ihre schlechte Kondition daran zu erkennen, dass die Elemente der dazu inversen Matrix betragsmäßig groß werden.
Die äquilibrierte Form von A muss man schon voraussetzen, denn für die mit einem großen Faktor durchmultiplizierte Matrix

$$\widetilde{A} = \begin{pmatrix} 2 \cdot 10^5 & 6 \cdot 10^5 \\ 2 \cdot 10^5 & 6.00001 \cdot 10^5 \end{pmatrix}$$

würde man

$$\widetilde{A}^{-1} = (a'_{ij}) = \begin{pmatrix} 3.000005 & -3 \\ -3 & 1 \end{pmatrix}$$

mit $a'_{ij} = \mathcal{O}(1)$ erhalten.

Zahlreiche Eigenschaften der Matrix, wie sie in Abschnitt 3.1 genannt wurden, sind auch hier feststellbar.

Einige Rechnungen in Maple mit der Matrix A (Datei *matrix2x2.mws*)

```
> restart:
  with(linalg):
```

Definition des LGS mit Matrix A (`float`-Zahlen bzw. symbolisch)

```
> Digits:=10:
  A:=matrix(2,2,[2.0, 6.0, 2.0, 6.00001]);
  AA:=matrix(2,2,[2, 6, 2, 600001/100000]);

  rank(A);
  det(A);
  norm(A);
  inverse(A);
  inverse(AA);

  b:=vector([8, 8.00001]);
  bb:=vector([8, 800001/100000]);
  xs:=vector([1,1]);
 `Ax*-b`=evalm(A&*xs-b);
```

$$A := \begin{bmatrix} 2.0 & 6.0 \\ 2.0 & 6.00001 \end{bmatrix}$$

$$AA := \begin{bmatrix} 2 & 6 \\ 2 & \frac{600001}{100000} \end{bmatrix}$$

$$2$$

$$0.000020$$

$$8.00001$$

$$\begin{bmatrix} 300000.5000 & -300000.0000 \\ -100000.0000 & 100000.0000 \end{bmatrix}$$

$$\begin{bmatrix} \frac{600000}{2} & -300000 \\ -100000 & 100000 \end{bmatrix}$$

$$b := [8,\ 8.00001]$$

$$bb := \left[8,\ \frac{800001}{100000}\right]$$

$$xs := [1,\ 1]$$

$$Ax^* - b = [0.,\ 0.]$$

Rechnung bei schwacher GPA mit unbefriedigenden Ergebnissen

```
> Digits:=5;
  evalm(A);
  rank(A);
  det(A);
  norm(A);
  inverse(A);
```

$$Digits := 5$$

$$\begin{bmatrix} 2.0 & 6.0 \\ 2.0 & 6.0000 \end{bmatrix}$$

$$1$$

$$0.$$

$$8.$$

```
Error, (in inverse) singular matrix

> Digits:=10:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 300000.5000 & -300000.0000 \\ -100000.0000 & 100000.0000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

```
> Digits:=16:
  inverse(A);
  evalm(II-A&*inverse(A));
```

$$\begin{bmatrix} 300000.5000000000 & -300000.0000000000 \\ -100000.0000000000 & 100000.0000000000 \end{bmatrix}$$

$$\begin{bmatrix} 0. & 0. \\ 0. & 0. \end{bmatrix}$$

Die inverse Matrix ist exakt in der geforderten Genauigkeit der Mantissenlänge.

Test von zwei Lösungskandidaten und Berechnung der Residuen dazu

```
> Digits:=16;
  xq:=vector([0,1.333333]);
  xd:=vector([0.999,1.001]);
  multiply(A,xq);
  rq:=evalm(b-multiply(A,xq));
  rd:=evalm(b-A&*xd);
```

$$Digits := 16$$

$$xq := [0,\ 1.333333]$$

$$xd := [0.999,\ 1.001]$$

$$[7.9999980,\ 8.00001133333]$$

$$rq := [-0.20\ 10^{-5},\ 0.133333\ 10^{-5}]$$

$$rd := [0.0040,\ 0.00400001]$$

Lösung des LGS mit GPA bei unterschiedlicher Mantissenlänge

```
> Digits:=16:
  erg_exakt:=linsolve(AA,bb);
  erg:=linsolve(A,b);
  erg[1]; erg[2];

> i:='i':
  printf(`Digits         x[1]                    x[2]\\n`):
  for i from 5 to 25 do
   Digits:=i:
   erg:=linsolve(A,b):
   if rank(A)<2 then
    printf(`%2d       Matrix A singulaer\n`,i)
   else
    printf(`%2d       %28.25f  %28.25f\n`,i,erg[1],erg[2])
   end if:
  end do:
```

$$erg\_exakt := [1,\ 1]$$

$$erg := [0.9999999996999996,\ 1.000000000100000]$$

$$0.9999999996999996$$

$$1.000000000100000$$

```
Digits                x[1]                              x[2]
 5       Matrix A singulaer
 6       Matrix A singulaer
 7       Matrix A singulaer
 8        0.969696960000000000000000   1.010101000000000000000000
 9        0.996996996000000000000000   1.001001000000000000000000
10        0.999699970000000000000000   1.000100010000000000000000
11        0.999969999700000000000000   1.000010000100000000000000
12        0.999996999996000000000000   1.000001000000000000000000
13        0.999999699999500000000000   1.000000100000000000000000
14        1.000000000000000000000000   1.000000000000000000000000
15        1.000000006000010000000000   0.999999997999996000000000
16        0.999999996999996000000000   1.000000001000000000000000
17        1.000000000000000000000000   1.000000000000000000000000
18        1.000000000030000000000000   0.999999999990000000000000
19        1.000000000003000010000000   0.999999999998999996000000
20        0.999999999999399999100000   1.000000000000200000000000
21        1.000000000000000000000000   1.000000000000000000000000
22        1.000000000000006000010000   0.999999999999997999996000
23        0.999999999999999999999900   1.000000000000000000000000
24        1.000000000000000120000200   0.999999999999999959999940
25        1.000000000000000006000010   0.999999999999999997999996
26        1.000000000000000000000000   1.000000000000000000000000
27        1.000000000000000000000000   1.000000000000000000000000
28        1.000000000000000000003000   0.999999999999999999999000
29        0.999999999999999999999700   1.000000000000000000000100
30        1.000000000000000000000030   0.999999999999999999999990
31        1.000000000000000000000012   0.999999999999999999999996
32        1.000000000000000000000001   1.000000000000000000000000
33        1.000000000000000000000000   1.000000000000000000000000
34        1.000000000000000000000000   1.000000000000000000000000
```

Es ist zu bemerken, wie sich die Genauigkeiten `Digits=14,17,21,26,27,33,34,...` deutlich gegenüber den anderen hervorhebt. Eine Systematik bei diesen Formaten ist nicht erkennbar. Es ist anzunehmen, dass in diesem Fall die Ausführung der arithmetischen Operationen im Prozessor genauer erfolgt.

Wir führen nun analoge Anweisungen in MATLAB mit seiner GPA *double* aus und vergleichen dann mit Maple-Rechnungen in der Genauigkeit `Digits=16`.

```
% matrix2x2.m

% Matrix 3
disp('Matrix 3')
% Test fuer sehr schlecht konditionierte Matrix
A =[2 6; 2 6.00001]
AA=[2 6; 2 600001/100000];
b =[8 8.00001]'
xs=[1,1]'         % exakte Loesung
A*xs-b             % Kontrolle
...
```

Zunächst die Ergebnisse.

```
Matrix 3
A =
   2.00000000000000   6.00000000000000
   2.00000000000000   6.00001000000000
b =
   8.00000000000000
   8.00001000000000
xs =
     1
     1
ans =
     0
     0

ans =
    1.999999999924285e-005
detA =
    1.999999999924285e-005

invA =
  1.0e+005 *
   3.00000500011357  -3.00000000011357
  -1.00000000003786   1.00000000003786
ans =
     1     0
     0     1
ans =
     0     0
     0     0

% invA  und  invAs_ analog

normA =
   8.94427861820589
condA =
    4.000006000334953e+006

xq =
                  0
   1.33333300000000
xd =
   0.99900000000000
   1.00100000000000
rq =
  1.0e-005 *
  -0.19999999993914
   0.13333300010743
rd =
   0.00400000000000
   0.00400001000000

lgs_erg1 =
     1
     1
lgs_erg2 =
     1
     1
lgs_erg3 =
matrix([[1], [1]])
```

Einige Bemerkungen

– Bei numerischen Rechnungen in MATLAB werden von den 16 angezeigten Mantissenstellen wegen der Matrixkondition die letzten 4–5 Stellen i. Allg. falsch.
– Der Wert der Spektralkondition der Matrix `cond2_A=4.000006000334953e+006` hat nur 10 genaue Mantissenstellen.
– Die Handrechnung der Determinante für die Matrix ist wie `det(A)` und beide haben die letzten 5 Dezimalstellen falsch. Das trifft auch auf die Inverse zu. Trotzdem ist die Fehlermatrix `II-A*invAs_=0`.
– Beim Test der beiden Lösungskandidaten für das LGS ist der Genauigkeitsverlust zumindest bei den Komponenten von `rq` sichtbar .
– Die Lösung des LGS ist in allen 3 Varianten exakt.
Dass der in MATLAB aufgerufene Maple-Befehl `linsolve` die exakte Lösung ausgibt, mag damit zusammenhängen, dass die Maple-Rechnung bei `Digits=14,17` genau ist, obwohl in der Nachbarschaft Abweichungen auftreten.

3.4 Matrix 4

Die Matrix [17]

$$A \;=\; A(n,n) \;=\; \begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & 0 \\ -1 & 2 & -1 & \cdots & 0 & 0 \\ 0 & -1 & 2 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 2 & -1 \\ 0 & 0 & 0 & \cdots & -1 & 2 \end{pmatrix} = \text{tridiag}(-1,2,-1)$$

ist eine Matrix mit zahlreichen typischen Merkmalen wie

– schwach besetzt,
– tridiagonal und damit Bandstruktur,
– schwach diagonaldominant,
– irreduzibel diagonaldominant,
– symmetrisch und positiv definit, positive relle EW,
– diagonalisierbar,
– Eigenschaft A,
– konsistent geordnet (Eigenschaft wird später erläutert),
– L-Matrix, Stieltjes-Matrix,
– M-Matrix, monoton.

Dazu notieren wir die symmetrische inverse Matrix (nur oberes Dreieck)

$$A^{-1} = (a'_{ij}) = \begin{pmatrix} n\frac{1}{n+1} & (n-1)\frac{1}{n+1} & (n-2)\frac{1}{n+1} & \cdots & 2\frac{1}{n+1} & \frac{1}{n+1} \\ & (n-1)\frac{2}{n+1} & (n-2)\frac{2}{n+1} & \cdots & 2\frac{2}{n+1} & \frac{2}{n+1} \\ & & (n-2)\frac{3}{n+1} & \cdots & 2\frac{3}{n+1} & \frac{3}{n+1} \\ & & & \ddots & \vdots & \vdots \\ & & & & 2\frac{n-1}{n+1} & \frac{n-1}{n+1} \\ & & & & & \frac{n}{n+1} \end{pmatrix}$$

mit ihren nicht negativen Elementen

$$a'_{ij} = \frac{i\,(n+1-j)}{(n+1)}, \quad 1 \le i \le j \le n, \text{ und } \; a'_{ii} > 0.$$

Hier sind sogar alle Einträge positiv.

Wegen $A = A^T$ sind Zeilen- und Spaltensummennorm gleich und betragen

$$\|A\|_\infty = \|A\|_1 = 4$$

sowie mit der Gaußschen Klammer $[\,]$

$$\|A^{-1}\|_\infty = \|A^{-1}\|_1 = \frac{1}{2}\left[\frac{n+1}{2}\right]\left[\frac{n+2}{2}\right] = \begin{cases} \frac{(n+1)^2}{8}, & \text{falls } n \text{ ungerade,} \\ \frac{n(n+2)}{8}, & \text{falls } n \text{ gerade.} \end{cases}$$

Somit gilt

$$\mathrm{cond}_\infty(A) = \mathrm{cond}_1(A) = 2\left[\frac{n+1}{2}\right]\left[\frac{n+2}{2}\right] = \frac{n^2}{2} + \mathcal{O}(n).$$

Für $n = 99$ erhält man $\mathrm{cond}_\infty(A) = 5\,000$, für $n = 999$ den Wert $\mathrm{cond}_\infty(A) = 500\,000$.
Es ist typisch, dass bei schlecht konditionierten Matrizen mit moderaten Elementen die Inverse dazu betragsgroße Einträge besitzt. Der bzw. die beiden größten Koeffizienten von A^{-1} stehen in der "Mitte" auf der Diagonalen und haben den Wert

$$\begin{aligned} a'_{\frac{n+1}{2},\frac{n+1}{2}} &= \frac{(n+1)^2}{4(n+1)} = \frac{n+1}{4}, \quad n \text{ ungerade,} \\ a'_{\frac{n}{2},\frac{n}{2}} = a'_{\frac{n}{2}+1,\frac{n}{2}+1} &= \frac{n(n+2)}{4(n+1)}, \quad n \text{ gerade.} \end{aligned}$$

Mit der regulären Zerlegung $A = D - C,\ D = \mathrm{diag}(A) = 2I$, erhält man die Matrix

$$J \;=\; D^{-1}C \;=\; \frac{1}{2}\begin{pmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 1 & 0 & 1 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \;=\; \mathrm{tridiag}(d, 0, d),\ d = \frac{1}{2}.$$

Ihr Spektralradius ist $\rho(J) < 1$.
Wir notieren auch die EW $\tilde{\lambda}_i$ und EV $\tilde{v}^{(i)}$ von J.

$$\begin{aligned} \tilde{\lambda}_i &= \cos(i\pi/(n+1)), \quad i = 1, 2, ..., n, \\ &\quad -1 < \tilde{\lambda}_n < \tilde{\lambda}_{n-1} < .... < \tilde{\lambda}_2 < \tilde{\lambda}_1 = \cos(\pi/(n+1)) < 1, \\ \tilde{\lambda}_{n+1-i} &= -\tilde{\lambda}_i, \quad i = 1, 2, ..., [\tfrac{n}{2}], \quad \text{falls } n \text{ ungerade, dann } \tilde{\lambda}_{\frac{n+1}{2}} = 0, \\ \tilde{v}^{(i)} &= (\tilde{v}_1^{(i)}, \tilde{v}_2^{(i)}, ..., \tilde{v}_n^{(i)})^T \text{ mit } \tilde{v}_j^{(i)} = \sin(ij\pi/(n+1)). \end{aligned}$$

Die reellen positiven EW der spd und irreduzibel diagonaldominanten Matrix A sind

$$\lambda_i = 2[1-\cos(i\pi/(n+1))] = 4\sin^2(i\pi/(2(n+1))), \quad i = 1,2,...,n,$$

die EV

$$v^{(i)} = (v_1^{(i)}, v_2^{(i)}, ..., v_n^{(i)})^T \text{ mit } v_j^{(i)} = \sin(ij\pi/(n+1)).$$

Wegen $A = 2(I-J)$ gilt $\lambda_i = 2(1-\tilde{\lambda}_i)$ und $v^{(i)} = \tilde{v}^{(i)}$.
Mit $h = 1/(n+1)$, $\cos(x) = 1-2\sin^2(x/2)$, gelten die Beziehungen

$$\lambda_i = 4\sin^2(i\pi h/2),$$

$$v^{(i)}, \quad v_j^{(i)} = \sin(ij\pi h),$$

$$0 < 2(1-\cos(\pi h)) = 4\sin^2(\pi h/2) = \lambda_{min} = \lambda_1 < \lambda_2 < ... < \lambda_n = \lambda_{max} =$$

$$= 4\sin^2(n\pi h/2) = 4[1-\sin^2(\pi h/2)] = 4\cos^2(\pi h/2) = 4 - \lambda_{min} < 4,$$

und die Abschätzungen

$$\lambda_1 \approx \pi^2 h^2, \qquad \lim_{h\to 0} \lambda_1 = 0,$$

$$\lambda_n \approx 4 - \pi^2 h^2, \quad \lim_{h\to 0} \lambda_n = 4,$$

$$\lambda_{\frac{n+1}{2}} = 2, \quad \text{falls } n \text{ ungerade},$$

$$\sigma(A) \in (0,4).$$

Leider ist die Kondition der Matrix A sehr schlecht, was bei der Berechnung von $\text{cond}_1(A)$ schon bemerkt wurde. Mit der Spektralnorm haben wir

$$\begin{aligned} \|A\|_2 &= \sqrt{\max_{i=1(1)n} \mu_i}, \quad 0 \le \mu_i \in \sigma(A^TA), \\ &= \sqrt{\rho(A^TA)}, \\ &\qquad \sigma(A^TA) = \sigma(AA^T) = \{\mu_i(A^TA),\ i = 1,2,...,n\} \text{ Spektrum}, \end{aligned}$$

wegen $A = A^T > 0$ die spektrale Kondition

$$\begin{aligned} \kappa(A) &= \text{cond}_2(A) = \|A\|_2 \, \|A^{-1}\|_2 \\ &= \frac{\lambda_{max}}{\lambda_{min}} = \frac{4-\lambda_{min}}{\lambda_{min}} = \frac{4}{\lambda_{min}} - 1 \\ &\approx \frac{4}{4\sin^2(\pi h/2)} \approx \frac{1}{(\pi h/2)^2} \\ &\approx \frac{4(n+1)^2}{\pi^2} \gg 1 \text{ für } n \gg 1. \end{aligned}$$

Die Matrix A hat eine einfache LU-Faktorisierung, wobei die Bandstruktur sich auf die beiden Dreiecksmatrizen überträgt. Diese Zerlegung entsteht auch beim GA ohne Pivotstrategie.

$$A = LU = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -\frac{1}{2} & 1 & 0 & \cdots & 0 & 0 \\ 0 & -\frac{2}{3} & 1 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & -\frac{n-1}{n} & 1 \end{pmatrix} \begin{pmatrix} \frac{2}{1} & -1 & 0 & \cdots & 0 & 0 \\ 0 & \frac{3}{2} & -1 & \cdots & 0 & 0 \\ 0 & 0 & \frac{4}{3} & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & \frac{n}{n-1} & -1 \\ 0 & 0 & 0 & \cdots & 0 & \frac{n+1}{n} \end{pmatrix}$$

Damit ist $L = \text{tridiag}(l, 1, 0)$ und $U = \text{tridiag}(0, u, -1)$.
Aus der Produktdarstellung ist sofort die Determinante von A erkennbar, nämlich

$$\det(A) = \det(L)\det(U) = 1 \cdot (n+1) = n+1.$$

Damit ist $\det(A^{-1}) = 1/(n+1)$.

Um bei der Lösung des LGS $Ax = b$ eine akzeptable Näherungslösung $\tilde{x}$ zur exakten Lösung x zu erhalten, ist eine starke GPA notwendig.
Die Fehlerakkumulation beim GA mit $A(n, n)$ und t Dualstellen der Mantisse des GPF führt bei $n \to \infty$ zum absoluten Fehler

$$\|\Delta x\| = \|\tilde{x} - x\| = 2^{-t}\,\text{cond}(A)\,K(n),$$

wobei für den verkürzten GA ("chase method", "metod progonka") ohne Pivotisierung bei $A = \text{tridiag}()$ und diagonaldominant oder $A = A^T > 0$ die Beziehung

$$K(n) = \mathcal{O}(\sqrt{n})$$

gilt.
Für $t = 64$ (*extended*-Format), $t = 53$ (*double*-Format) oder $t = 40$ (*real*-Format) kann man die gültigen Dezimalstellen der Näherungslösung ermitteln.

Betrachten wir die Lösung eines LGS etwas detaillierter.
Die exakte Lösung des LGS $Ax = b$ mit $b = (1, 0, 0, ..., 0, 1)^T$ ist der Einsvektor $x^* = (1, 1, ..., 1)^T$.

Zur Bestimmung seiner Computerlösung $\tilde{x}$ machen wir zwei hauptsächliche Schritte.

1. Direkte Lösung mittels verkürztem GA.
2. Iterative Lösung (Iterationsverfahren, IV) mittels Jacobi- (GSV) und Gauß-Seidel-Verfahren (ESV).

Zu 1.: Wegen $A = A^T > 0$ kann der verkürzte Gauß-Algorithmus ohne Pivotstrategie durchgeführt werden. Dabei erhält man auch die LU-Zerlegung von A.
Die allgemeine Fehlerakkumulation bei der numerischen Lösung von $Ax = b$ mittels GA mit $A(n, n)$ und t Dualstellen der Mantisse des GPF führt bei $n \to \infty$ zum absoluten Fehler ([27], Kap. 5.3) der Näherungslösung x gemäß

$$\|e(x)\| = \|x - x^*\| = 2^{-t}\,\mathrm{cond}(A)\; K(n),$$

wobei x^* exakte Lösung und

$$K(n) = \begin{cases} \mathcal{O}(\sqrt{n}) & \text{für verkürzten GA ohne Pivotisierung bei tridiagonaler ,} \\ & \text{Matrix } A\text{, dazu } A \text{ diag.dominant oder } A = A^T > 0, \\ \mathcal{O}(n) & \text{für GA ohne Pivotisierung, } A \text{ diag. dom. oder } A = A^T > 0, \\ \mathcal{O}(2^n) & \text{für GA mit Spaltenpivotisierung,} \\ \mathcal{O}(n^2) & \text{für GA mit vollständiger Pivotisierung.} \end{cases}$$

Den Einfluss der Kondition der Matrix A auf den Fehler $e(x)$ kann man sich auch durch folgende Überlegung veranschaulichen.
Es ist $Ax^* = b$ und sei x als Näherungslösung die exakte Lösung des gestörten LGS $(A + \Delta A)x = b$. Dann folgen aus der Differenz beider die Beziehungen

$$\begin{aligned} A(x^* - x) &= -\Delta A x, \\ x^* - x &= -A^{-1}\Delta A x. \end{aligned}$$

Also werden sich betragsgroße Elemente in A^{-1}, d. h. eine große Konditionszahl von A, ungünstig auf den Fehler auswirken. Damit wird eine gewisse von der Dimension n abhängige Anzahl von letzten Mantisssenstellen der Näherungslösung ungültig.
Für das *double*-Format ist $t = 53$ (53 Bit Mantisse bei 8 Byte Wortlänge).
Der Akkumulationsfehler zum GA berechnet sich bei schlechter Kondition von A als

$$\begin{aligned} \|e(x)\| &= c\,2^{-t}\,\mathrm{cond}(A)\; n, \quad c > 0 \\ &= \hat{c}\,2^{-53}\; n^3, \quad \hat{c} > 0 \\ &= \tilde{c}\,10^{-16}\; n^3, \quad \tilde{c} \approx 10^{-4} > 0 \\ &\approx 10^{-20}\; n^3 \end{aligned}$$

Der Akkumulationsfehler beim verkürzten GA für obige Matrix A beträgt

$$\begin{aligned} \|e(x)\| &= c\,2^{-t}\,\mathrm{cond}(A)\; \sqrt{n}, \quad c > 0 \\ &= \tilde{c}\,10^{-16}\; n^2\sqrt{n}, \quad \tilde{c} \approx 10^{-3} > 0 \\ &\approx 10^{-19}\; n^2\sqrt{n}. \end{aligned}$$

Die jeweilige Wahl und "grobe Anpassung" des Faktors $\tilde{c}$ erfolgte aus einigen Testrechnungen mit der Maximumnorm und dem Format *double* zum gegebenen Beispiel.

Damit erhält man für letzteres zu den verschiedenen Dimensionen n die folgende Tabelle.

n	$\|e(x)\| \approx 10^{-19}\ n^2\sqrt{n}$
9	$2 \cdot 10^{-17}$
19	$3 \cdot 10^{-16}$
99	10^{-14}
999	$3 \cdot 10^{-12}$
4999	$2 \cdot 10^{-11}$
7999	$6 \cdot 10^{-10}$

Für das *real*-Format ist $t = 40$ (40 Bit Mantisse bei 6 Byte Wortlänge). Dann berechnet sich der Akkumulationsfehler beim verkürzten GA als $\|e(x)\| \approx 10^{-15}\ n^2\sqrt{n}$. Man sollte also das LGS mit entsprechend starker GPA behandeln, damit die Computerzwischenlösung x akzeptabel ist.
Wir berechnen diese mit einem Pascal-Programm (TP, Borland Pascal bzw. DevPascal) für die vier GPF *single, real, double* und *extended*. Die Anzahl der dabei verwendeten Dezimalziffern der Mantisse beträgt entsprechend 7–8, 11–12, 15–16 und 19–20.
Die größten Abweichungen von der exakten Lösung x^* sind in oder nahe der Intervallmitte zu erwarten, also

$$\max_{1\le i\le n} |x_i - x_i^*| \approx |x_{[n/2]} - x^*_{[n/2]}|.$$

	max|x(i)-x^*(i)|			
n	single	real	double	extended
9	1.19E-7	4.55E-12	1.11E-16	1.16E-19
19	1.79E-7	2.73E-11	2.22E-16	3.79E-19
99	2.50E-6	6.11E-10	1.22E-14	1.04E-17
999	3.63E-4	5.73E-08	2.88E-13	9.32E-17
1999	1.33E-3	2.27E-07	5.52E-13	3.27E-16
2999	2.88E-3	5.18E-07	2.36E-12	4.69E-16
3999	1.56E-2	9.16E-07	4.67E-12	4.38E-16
4999	7.88E-2	1.43E-06	5.12E-12	5.49E-16
5999	1.63E-1	2.04E-06	5.42E-12	6.44E-16
6999	2.52E-1	2.77E-06	6.40E-12	1.76E-15
7999	3.39E-1	3.61E-06	8.40E-12	1.24E-15
8999	4.21E-1	4.58E-06	1.14E-11	1.04E-15
9999	4.95E-1	5.67E-06	1.28E-11	1.19E-15
15999	7.88E-1		2.79E-11	1.65E-15

Zu 2.: Das GSV

$$x^{(k+1)} = J\,x^{(k)} + c, \quad k = 0, 1, ...,$$

mit der üblichen Matrixzerlegung $A = A_L + A_D + A_R$ und der Iterationsmatrix

$$J = \text{tridiag}(\tfrac{1}{2},0,\tfrac{1}{2}) = I - A_D^{-1}A = I - \tfrac{1}{2}A$$

ist für beliebigen Startvektor $x^{(0)}$ konvergent. Die EW von J sind

$$\lambda_i(J) = \cos(i\pi/(n+1)) = 1 - \tfrac{1}{2}\lambda_i(A), \quad i = 1, 2, ..., n,$$

und damit betragsmäßig kleiner als Eins.
Jedoch wird die lineare Konvergenz äußerst langsam sein, denn der Spektralradius

$$\rho(J) = \max_i |\lambda_i(J)| = \cos(\pi/(n+1)) = 1 - 2\sin^2(\pi/(2n+2)) \approx 1 - \frac{\pi^2}{2(n+1)^2} < 1$$

liegt sehr nahe der Eins. Für $n = 99$ erhält man $\rho(J) = 0.999\,506\,560$, für $n = 999$ den Wert $\rho(J) = 0.999\,995\,065$.
Als Startvektoren testen wir bei gegebenen n zwei Versionen.
Zuerst nehmen wir als $x^{(0)}$ das Ergebnis des 1. Schritts. Der weitere Verlauf ist wie folgt, wobei nur der Fehler $\max\limits_{1\le i\le n} |x_i^{(k)} - x_i^*|$ an einigen Iterationsschritten ausgewertet wird. Das Kürzel o.V. heißt "ohne Veränderung" des bisherigen Fehlers, genauso der Iterierten.

n=999

	max |x^(k)(i)-x^*(i)|		
k	single	real	double	extended
0	3.626346E-4	5.725178E-08	2.882138E-13	9.324138E-17
1	o.V.	5.725087E-08	2.882138E-13	o.V.
2	o.V.	5.725087E-08	2.881028E-13	o.V.
3	o.V.	5.724996E-08	2.879918E-13	o.V.
4	o.V.	5.724996E-08	o.V.	o.V.
1000	o.V.	5.68E-08	o.V.	o.V.
2000	o.V.	5.63E-08	o.V.	o.V.
5000	o.V.	5.50E-08	o.V.	o.V.
10000	o.V.	5.27E-08	o.V.	o.V.
20000	o.V.	4.82E-08	o.V.	o.V.
50000	o.V.	3.45E-08	o.V.	o.V.
100000	o.V.	1.18E-08	o.V.	o.V.
120000	o.V.	2.68E-09	o.V.	o.V.
125000	o.V.	4.08E-10	o.V.	o.V.
126000	o.V.	0	o.V.	o.V.

Man sieht, dass die Nachiteration mit dem GSV bis auf den Fall des Turbo Pascal-typischen Formats *real* keine Verbesserung der Iterierten bringt. Warum ist das so? Trotz erfüllter Konvergenzbedingung ist auch hier die Kondition von J wichtig.
Die Matrix J erbt diese sozusagen von der Matrix A. Für ungerades n ist J singulär, denn der mittlere EW ist Null, und die Konditionszahl ist ∞. Für gerades n ist J regulär, denn alle EW sind ungleich Null. Die beiden mittleren EW sind
$\cos(\frac{n}{n+1}\frac{\pi}{2}) > 0$ und $\cos(\frac{n+2}{n+1}\frac{\pi}{2}) = -\cos(\frac{n}{n+1}\frac{\pi}{2}) < 0$.
Die spektrale Konditionszahl beträgt für $n \gg 1$

$$\frac{\cos(\frac{\pi}{n+1})}{\cos(\frac{n}{n+1}\frac{\pi}{2})} = \frac{1-2\sin^2(\frac{\pi}{2(n+1)})}{\cos(\frac{n}{n+1}\frac{\pi}{2})} \approx \frac{1-2(\frac{\pi}{2(n+1)})^2}{\frac{\pi}{2}-\frac{n}{n+1}\frac{\pi}{2}} = \frac{2}{\pi}\left(n+1-\frac{\pi^2}{2(n+1)}\right) \approx \frac{2}{\pi}(n+1).$$

Wir machen deshalb dieselbe Überlegung wie vorher.
x^* ist die exakte Lösung der Fixpunktgleichung $x = J\,x + c$, $\tilde{x}$ sei die letzte Iterierte und als Näherungslösung die exakte Lösung des gestörten Systems $x = (J+\Delta J)\,x+c$.
Damit erhalten wir die Beziehungen

$$\begin{aligned} x^* &= J\,x^* + c, \\ \tilde{x} &= (J+\Delta J)\,\tilde{x} + c, \\ \tilde{x} - x^* &= J(\tilde{x} - x^*) + \Delta J\,\tilde{x}, \\ (I-J)(\tilde{x} - x^*) &= \Delta J\,\tilde{x}, \\ \tilde{x} - x^* &= (I-J)^{-1}\Delta J\,\tilde{x}. \end{aligned}$$

Die Größen $\lambda(I-J) = 1-\lambda(J)$ mit

$$0 < \lambda_{min}(I-J) = 1-\cos(\tfrac{\pi}{n+1}) < 1-\cos(\tfrac{2\pi}{n+1}) < \ldots < 1+\cos(\tfrac{\pi}{n+1}) = \lambda_{max}(I-J) < 2$$

sind die EW von $I-J$ und die spektrale Konditionszahl zu $I-J$ wie auch $(I-J)^{-1}$ beträgt

$$\frac{1+\cos(\frac{\pi}{n+1})}{1-\cos(\frac{\pi}{n+1})} \approx \frac{2}{1-(1-2\sin^2(\frac{\pi}{2(n+1)}))} \approx \frac{4(n+1)^2}{\pi^2}.$$

Die Matrixkondition ist die Fehlerquelle. Damit wird in der gegebenen GPA der Iterationsprozess vorzeitig beendet, ohne dass man genau zur Lösung gelangt.

Aber es macht Sinn, die Iteration bei einem Startvektor zu beginnen, der aus dem ersten Schritt mit ungenauerer Arithmetik entsteht. Das heißt, z. B. den GA in der GPA *double* auszuführen und das GSV als Nachiteration mit *extended* weiter zu rechnen.

Nun nehmen wir das GSV von Beginn an mit dem Startvektor $x^{(0)} = 0$ und kontrollieren, wie sich der genannte Fehler entwickelt. Da die Dimension etwas kleiner ist, ist die Kondition der entsprechenden Matrizen A und J nicht so schlecht. Aber das IV stagniert ebenfalls, wenn auch "später" und bei kleinerem Fehler.

n=99

iter	single	max|x(i)-x^*(i)| real	double	extendend
0	1	1	1	1
1000	7.72E-1	7.72E-01	7.72E-01	7.72E-01
2000	4.74E-1	4.74E-01	4.74E-01	4.74E-01
4000	1.77E-1	1.77E-01	1.77E-01	1.77E-01
6000	6.59E-2	6.59E-02	6.59E-02	6.59E-02
8000	2.45E-2	2.45E-02	2.45E-02	2.45E-02
10000	9.15E-3	9.15E-03	9.15E-03	9.15E-03
20000	6.91E-5	6.57E-05	6.57E-05	6.57E-05
21000	4.86E-5	4.01E-05	4.01E-05	4.01E-05
25000	o.V.	5.57E-06	5.57E-06	5.57E-06
35000	o.V.	3.95E-08	4.00E-08	4.00E-08
43000	o.V.	3.81E-10	7.72E-10	7.72E-10
44000	o.V.	0	4.71E-10	4.71E-10
50000	o.V.	0	2.44E-11	2.44E-11
60000	o.V.	0	1.88E-13	1.75E-13
62000	o.V.	0	9.06E-14	6.53E-14
75000	o.V.	0	o.V.	1.16E-16
77000	o.V.	0	o.V.	4.43E-17
78000	o.V.	0	o.V.	4.42E-17
79000	o.V.	0	o.V.	o.V.

Der Fehler genügt der rekursiven Berechnung

$$e^{(k+1)} = Je^{(k)} = ... = J^{k+1}e^{(0)}$$

und erfüllt mit der euklidischen Vektornorm und spektralen Matrixnorm die Beziehung

$$\|e^{(k+1)}\|_2 \leq \|J\|_2 \, \|e^{(k)}\|_2 = r\|e^{(k)}\|_2 \leq r^{k+1}\|e^{(0)}\|_2,$$

wobei $r = \rho(J) = \cos(\frac{\pi}{n+1}) \lessapprox 1$.

Wir sehen in den zwei implementierten Fällen, wie langsam die Potenzen r^k für $r \lessapprox 1$ abnehmen und gegen Null tendieren.

k	n=999 r=0.9999 9506 r^k	n=99 r=0.9995 0656 r^k
1000	0.9950 7734 5854	0.6104 4845 7516
5000	0.9756 2786 4558	0.0847 7055 0500
10000	0.9518 4973 0103	0.0071 8604 6232
20000	0.9060 1790 8697	0.0000 5163 9260
50000	0.7813 4341 3421	0.0000 0000 0019
100000	0.6104 9752 9697	0.3671E-021
500000	0.0848 0462 8271	0.6675E-107
1000000	0.0071 9182 4976	0.4456E-214

Das ESV

$$x^{(k+1)} = H_1\, x^{(k)} + c, \quad k = 0, 1, ...,$$

mit der Iterationsmatrix

$$H_1 = \begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{2^2} & \frac{1}{2} & 0 & \cdots & 0 \\ 0 & \frac{1}{2^3} & \frac{1}{2^2} & \frac{1}{2} & \ddots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots & 0 \\ 0 & \frac{1}{2^{n-1}} & \frac{1}{2^{n-2}} & \frac{1}{2^{n-3}} & \cdots & \frac{1}{2} \\ 0 & \frac{1}{2^n} & \frac{1}{2^{n-1}} & \frac{1}{2^{n-2}} & \cdots & \frac{1}{2^2} \end{pmatrix} = I - (A_D + A_L)^{-1} A$$

ist für beliebigen Startvektor $x^{(0)}$ konvergent.
Die EW von H_1 sind reell oder komplex und betragsmäßig kleiner als Eins, mindestens einer ist Null. Im Spektrum $\sigma(H_1)$ befinden sich auch die Werte $\lambda(J)^2$.
Jedoch wird die lineare Konvergenz langsam sein, denn der Spektralradius

$$\rho(H_1) = \max_i |\lambda_i(H_1)| = \cos^2(\pi/(n+1)) \approx 1 - \frac{\pi^2}{(n+1)^2} < \rho(J) < 1$$

liegt sehr nahe der Eins. Gleichzeitig gilt $\rho(H_1) = \rho(J)^2 < 1$, so dass das ESV schneller konvergiert und ca. halb soviel Iterationen wie das GSV braucht.

Wir rechnen dieselben Beispiele wie zum GSV.
Für $n = 999$ ist das Ergebnis des GA so gut, dass das ESV mit dem Startvektor als Ergebnis des GA fast den gleichen Iterationsverlauf wie das GSV zeigt.
Die etwas schnellere Konvergenz bei halber Iterationsanzahl der Nachiteration mit dem ESV ist am ehesten bei den Formaten *real* und *double* zu erkennen.

```
n=999
                        max |x^(k)(i)-x^*(i)|
  iter |  single        real           double         extended
-------------------------------------------------------------------
     0 | 3.626346E-4   5.725178E-08   2.882138E-13   9.324138E-17
     1 |  o.V.         5.725087E-08   2.879918E-13    o.V.
     2 |  o.V.         5.724996E-08    o.V.           o.V.
  1000 |  o.V.         5.63E-08        o.V.           o.V.
  2000 |  o.V.         5.54E-08        o.V.           o.V.
  5000 |  o.V.         5.27E-08        o.V.           o.V.
 10000 |  o.V.         4.82E-08        o.V.           o.V.
 20000 |  o.V.         3.91E-08        o.V.           o.V.
 50000 |  o.V.         1.18E-08        o.V.           o.V.
```

60000	o.V.	2.68E-09	o.V.	o.V.
62000	o.V.	8.63E-10	o.V.	o.V.
63000	o.V.	o.V.	o.V.	o.V.

Nun nehmen wir das ESV von Beginn an mit dem Startvektor $x^{(0)} = 0$ und kontrollieren, wie sich der genannte Fehler entwickelt. Da die Dimension etwas kleiner ist, ist die Kondition der entsprechenden Matrix A nicht so schlecht. H_1 hat jedoch die Konditionszahl ∞. Im Vergleich zum GSV ergibt sich die halbe Iterationsanzahl. Aber der Iterationsprozess stagniert ebenfalls, wenn auch "später" und bei kleinerem Fehler.

n=99

	max|x(i)-x^*(i)|		
iter | single	real	double	extendend
0 | 1	1	1	1
1000 | 4.75E-1	4.75E-01	4.75E-01	4.75E-01
2000 | 1.77E-1	1.77E-01	1.77E-01	1.77E-01
4000 | 2.46E-2	2.46E-02	2.46E-02	2.46E-02
6000 | 3.41E-3	3.41E-03	3.41E-03	3.41E-03
8000 | 4.79E-4	4.74E-04	4.74E-04	4.74E-04
10000 | 6.92E-5	6.58E-05	6.58E-05	6.58E-05
11000 | 4.86E-5	2.45E-05	2.45E-05	2.45E-05
12000 | o.V.	9.13E-06	9.13E-06	9.13E-06
15000 | o.V.	4.72E-07	4.73E-07	4.73E-07
20000 | o.V.	2.94E-09	3.40E-09	3.40E-09
21000 | o.V.	8.37E-10	1.27E-09	1.27E-09
22000 | o.V.	0	4.72E-10	4.72E-10
25000 | o.V.	0	2.44E-11	2.44E-11
30000 | o.V.	0	1.89E-13	1.75E-13
31000 | o.V.	0	9.06E-14	6.54E-14
35000 | o.V.	0	o.V.	1.26E-15
38000 | o.V.	0	o.V.	6.83E-17
39000 | o.V.	0	o.V.	4.42E-17
40000 | o.V.	0	o.V.	o.V.

Zum Schluss rechnen wir für $n = 99$ den GA im Format *single* und setzen dessen Ergebnis mit den beiden IV fort, und zwar stufenweise in den Genauigkeiten *single, real, double, extended*, wobei die letzte Iterierte einer Stufe der Startvektor für die nächste Stufe ist. Die erste Nachiteration, nach dem GA mit dem Format *single* ausgeführt, bringen keinen relevanten Genauigkeitszuwachs.
Wir sehen noch einmal deutlich die halbe Iterationsanzahl vom ESV gegenüber dem GSV und die sinnvolle Nachiteration bei ungenauem Startvektor.
Die Angaben sind auf 13 Mantissenstellen abgeschnitten.

```
n=99
                              max|x(i)-x^*(i)|

  single       GA |  2.5033 9508 0566E-06
  ------------------------------------------------------------
              iter |          GSV                      ESV
  single         0 |  2.5033 9508 0566E-06   2.5033 9508 0566E-06
                 1 |   o.V.                  2.3841 8579 1015E-06
                 2 |                          o.V.
  ------------------------------------------------------------
  real           0 |  2.5033 9508 0566E-06   2.3841 8579 1015E-06
                 1 |  2.4437 9043 5791E-06   2.3841 6396 3142E-06
              6876 |  3.0735 4639 5171E-08   2.2300 8100 8747E-09
              6877 |  3.0706 3601 2125E-08   2.2282 6201 9343E-09
              6878 |  3.0706 3601 2125E-08    o.V.
             13686 |  2.2300 8100 8747E-09
             13687 |  2.2282 6201 9343E-09
             13688 |   o.V.
  ------------------------------------------------------------
  double         0 |  2.2282 6201 9343E-09   2.2282 6201 9343E-09
                 1 |  2.2282 6201 9343E-09   2.2273 5252 4641E-09
              9685 |  1.9395 5962 4020E-11   1.8141 0442 2237E-13
              9686 |  1.9385 8262 7758E-11   1.8118 8397 6188E-13
              9687 |  1.9376 5004 0417E-11    o.V.
             19371 |  1.8141 0442 2237E-13
             19372 |  1.8118 8397 6188E-13
             19373 |   o.V.
  ------------------------------------------------------------
  extendend      0 |  1.8118 8397 6188E-13   1.8118 8397 6188E-13
                 1 |  1.8118 8397 6188E-13   1.8112 4971 7917E-13
              7869 |  3.8664 4817 8751E-15   8.8579 3174 9206E-17
              7870 |  3.8644 2180 3390E-15   8.8470 8972 7481E-17
              7871 |  3.8626 8707 9914E-15    o.V.
             15738 |  8.8579 3174 9206E-17
             15739 |  8.8470 8972 7481E-17
             15740 |   o.V.
  ------------------------------------------------------------
```

Wir notieren noch das zugehörige Pascal Progamm, wobei die Dateiarbeit für das "Stufenprogramm" weggelassen wurde.

```
{$N+} program GA_Progonka_IV;
{(C)  W.Neundorf IfMath TU Ilmenau
      Verkuerzter GA und IV=GSV/ESV fuer Ax=b, A tridiagonal
      PROGONK3.PAS                                          }

uses  dos,crt;
const nmax=2000;   { bei BPW 64KByte-Grenze beachten }
                   { bei DEV-Pascal keine Grenze     }
type  float=real;  { single; real; double; extended; }
                   { bei DEV-Pascal real=double      }
      Vektor=array[1..nmax] of float;
      PVektor=^Vektor;
var   n,i,ie,nmin1,nmin2,nmin3,iter,itermax : longint;
      a,c,g,b,l,u,v,x,y,x0 : PVektor;
```

```
      h,Th,err,err1,errmin1,errmin2,errmin3 : float;
      ch:char;

{exakte Loesung}
function Tex(x:float):float;
begin
  Tex:=1;
end;

begin
  clrscr;
  writeln('Verkuerzter GA und IV=GSV/ESV fuer Tridiagonalsysteme');
  {Eingabe n,itermax}
  write('n='); readln(n);
  write('itermax='); readln(itermax);

  new(a); new(b); new(c); new(g); new(x); new(y); new(x0);
  new(l); new(u); new(v);
  nmin1:=n; errmin1:=1;
  nmin2:=n; errmin2:=1;
  nmin3:=n; errmin3:=1;
  while  n<=0  do
  begin
   n:=n+1000;
   h:=1.0/(n+1);

   { Definition A,b }
   for i:=1 to n do
   begin
    a^[i]:=2; c^[i]:=-1; g^[i]:=-1; b^[i]:=0;
   end;
   b^[1]:=1; b^[n]:=1; g^[n]:=0; v^[n]:=0;

   { GA, LU-Zerlegung }
   for i:=1 to n-1 do v^[i]:=g^[i];
   u^[1]:=a^[1];
   for i:=1 to n-1 do
   begin
    l^[i]:=c^[i]/u^[i];
    u^[i+1]:=a^[i+1]-l^[i]*v^[i];
   end;
   { Vorwaertsrechnung }
   y^[1]:=b^[1];
   for i:=2 to n do y^[i]:=b^[i]-l^[i-1]*y^[i-1];
   { Rueckwaertsrechnung }
   x^[n]:=y^[n]/u^[n];
   for i:=n-1 downto 1 do x^[i]:=(y^[i]-v^[i]*x^[i+1])/u^[i];

   { Ergebnisse }
   { writeln('u');
   for i:=1 to n do
   begin
    writeln(u^[i]);
    if i mod 20 = 0 then readln;
   end;
```

```
 readln;
 (* analog v,l,y,x *)
 }
 { writeln; writeln('x                              T(x)'); }
 err:=0;
 for i:=1 to n do
  begin
   Th:=Tex(i*h);
   err1:=abs(x^[i]-Th);
   if err1>err then
    begin
     err:=err1; ie:=i;
    end;
 { writeln(x^[i],'  ', Th); if i mod 20 = 0 then readln; }
  end;
 { writeln;   }
 writeln('GA:   n=',n:4,'  err=',err,'  i=',ie);
 readln;
 if err<errmin1 then
 begin
  errmin1:=err;
  nmin1:=n;
 end;

{IV}
{ Startvektor  (x0^[1],...,x0^[n]) }
for i:=1 to n do x0^[i]:=x^[i];
{for i:=1 to n do x0^[i]:=0; }

{ GSV }
for i:=1 to n do x^[i]:=x0^[i];
for iter:=1 to itermax do
 begin
  y^[1]:=(b^[1]-g^[1]*x^[2])/a^[1];
  for i:=2 to n-1 do y^[i]:=(b^[i]-c^[i]*x^[i-1]-g^[i]*x^[i+1])/a^[i];
  y^[n]:=(b^[n]-c^[n-1]*x^[n-1])/a^[n];
  err:=0;
  for i:=1 to n do
   begin
    Th:=Tex(i*h);
    err1:=abs(y^[i]-Th);
    if err1>err then
    begin
     err:=err1; ie:=i;
    end;
   end;
  x^:=y^;

  { Zwischenausgabe }
  if iter mod 1000 = 0 then
   begin
    writeln('GSV:  n=',n:4,'  iter=',iter,'  err=',err,'  i=',ie);
    writeln('P,p->Ausgabe x, Esc->Halt, sonst weiter');
    ch:=readkey;
    case ch of
```

```
      'P','p' : begin
                 writeln('x');
                 for i:=1 to n do
                  begin
                   writeln(x^[i]);
                   if i mod 20 = 0 then readln;
                  end;
                end;
      #27 : halt;
     end;
    end;
   if err<errmin2 then
    begin
     errmin2:=err;
     nmin2:=n;
    end;
  end; {iter}

{ ESV }
for i:=1 to n do x^[i]:=x0^[i];
for iter:=1 to itermax do
 begin
  y^[1]:=(b^[1]-g^[1]*x^[2])/a^[1];
  for i:=2 to n-1 do y^[i]:=(b^[i]-c^[i]*y^[i-1]-g^[i]*x^[i+1])/a^[i];
  y^[n]:=(b^[n]-c^[n-1]*y^[n-1])/a^[n];
  err:=0;
  for i:=1 to n do
   begin
    Th:=Tex(i*h);
    err1:=abs(y^[i]-Th);
    if err1>err then
    begin
     err:=err1;
     ie:=i;
    end;
   end;
  x^:=y^;

  { Zwischenausgabe }
  if iter mod 1000 = 0 then
   begin
    writeln('ESV:  n=',n:4,'  iter=',iter,'  err=',err,'  i=',ie);
    writeln('P,p->Ausgabe x, Esc->Halt, sonst weiter');
    ch:=readkey;
    case ch of
     'P','p' : begin
                writeln('x');
                for i:=1 to n do
                 begin
                  writeln(x^[i]);
                  if i mod 20 = 0 then readln;
                 end;
               end;
     #27 : halt;
   end;
```

```
    end;
    if err<errmin3 then
     begin
      errmin3:=err;
      nmin3:=n;
     end;
   end; {iter}
  end; {n}

  writeln('GA : errmin1=',errmin1,'  nmin1=',nmin1);
  writeln('GSV: errmin2=',errmin2,'  nmin2=',nmin2);
  writeln('ESV: errmin3=',errmin3,'  nmin2=',nmin3);
  dispose(a); dispose(b); dispose(c); dispose(g);
  dispose(x); dispose(y); dispose(x0);
  dispose(l); dispose(u); dispose(v);
  readln;
end.
```

Spd Tridiagonalmatrix

Für $n = 5$ erhält man

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{pmatrix}, \quad A^{-1} = \begin{pmatrix} \frac{5}{6} & \frac{4}{6} & \frac{3}{6} & \frac{2}{6} & \frac{1}{6} \\ \frac{4}{6} & \frac{8}{6} & \frac{6}{6} & \frac{4}{6} & \frac{2}{6} \\ \frac{3}{6} & \frac{6}{6} & \frac{9}{6} & \frac{6}{6} & \frac{3}{6} \\ \frac{2}{6} & \frac{4}{6} & \frac{6}{6} & \frac{8}{6} & \frac{4}{6} \\ \frac{1}{6} & \frac{2}{6} & \frac{3}{6} & \frac{4}{6} & \frac{5}{6} \end{pmatrix}.$$

Rechnungen für die spd Tridiagonalmatrix in Maple

Determinante, Inverse, Norm, Kondition und EW/EV (Datei *matr_trid1.mws*)

Zunächst noch einige Varianten zur Erzeugung der Tridiagonalmatrix

```
> A:=matrix(n,n,(i,j)->if i=j then 2
            elif abs(j-i)=1 then -1 else 0 end if);

  # linalg
  A:=band([-1,2,-1],n);           # A:=linalg[band]([-1,2,-1],n);
                                  # nicht A:=band([-1,2,-1],n,n+1);

  # Symmetrie wird mit oberem Dreieck erzeugt
  A:=array(symmetric,1..n,1..n,[]):
  for i from 1 to n do
   for j from i to n do
     if i=j then A[i,j]:=2
     elif j-i=1 then A[i,j]:=-1 else A[i,j]:=0 end if;
   end do:
  end do:
  A:=evalm(A);

  # falsch
  A:=array(symmetric,1..n,1..n,(i,j)->if i=j then 2
           elif j-i=1 then -1 else 0 end if);
```

Die ersten 3 Varianten liefern das gleiche Ergebnis

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}.$$

Variante 4 ist fehlerhaft.

```
Error, invalid parameters for creation of table or array
```

Weitere Möglichkeiten

```
> # Symmetrie wird mit oberem Dreieck erzeugt
  A:=Matrix(n,(i,j)->if i=j then 2
              elif abs(j-i)=1 then -1 else 0 end if,shape=symmetric);
  # genauso
  A:=Matrix(n,(i,j)->if i=j then 2
              elif j-i=1 then -1 else 0 end if,shape=symmetric):

  # Symmetrie wird mit oberem Dreieck erzeugt, deshalb hier Diagonalform
  A:=Matrix(n,(i,j)->if i=j then 2
              elif i-j=1 then -1 else 0 end if,shape=symmetric);
```

$$A := \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

$$A := \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

```
> # LinearAlgebra
  BandMatrix([-1,2,-1],1,n);  # LinearAlgebra[BandMatrix]([-1,2,-1],1,n);
  BandMatrix([-1,2,-1],1,n,n+1);
```

$$\begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 2 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \end{bmatrix}$$

```
> Digits:=16:
  i:='i': j:='j':
  n:=5:
  # A:=band([-1,2,-1],n);
  A:=matrix(n,n,(i,j)->if i=j then 2
                       elif abs(i-j)=1 then -1 else 0 end if):
  `A`=evalm(A);
  `rank(A)`=rank(A);
  `det(A)`=det(A);
  `inv(A)`=inverse(A);

  # Normen
  `norm(A,2)`=norm(A,2);                          # Spektralnorm
  evalf(%);
  `norm(A,1)`=norm(A,1);                          # Spaltensummennorm
  evalf(%);
  `norm(A)=norm(A,infinity)`=norm(A);             # Zeilensummennorm = SSN
  evalf(%);
  `norm(A,frobenius)`=norm(A,frobenius);          # Frobenius-Norm
  evalf(%);
  `norm(invA)=norm(invA,infinity)`=norm(invA); # ZSN der Inversen
  evalf(%);

  # Konditionen dazu, auch Kommando cond(A,*)
  `cond(A,2)`=evalf(norm(A,2)*norm(inverse(A),2));
  `cond(A,1)`=evalf(norm(A,1)*norm(inverse(A),1));
  `cond(A)=cond(A,infinity)`=evalf(norm(A)*norm(inverse(A)));
  `cond(A,frobenius)`=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
```

$$A = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

$$rank(A) = 5$$

$$det(A) = 6$$

$$inv(A) = \begin{bmatrix} \frac{5}{6} & \frac{2}{3} & \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ \frac{2}{3} & \frac{4}{3} & 1 & \frac{2}{3} & \frac{1}{3} \\ \frac{1}{2} & 1 & \frac{3}{2} & 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{2}{3} & 1 & \frac{4}{3} & \frac{2}{3} \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{2} & \frac{2}{3} & \frac{5}{6} \end{bmatrix}$$

$$norm(A, 2) = 2 + \sqrt{3}$$

$$norm(A, 2) = 3.732050807568877$$

$$norm(A, 1) = 4$$

$$norm(A, 1) = 4.$$

$$norm(A) = norm(A, infinity) = 4$$

$$norm(A) = norm(A, infinity) = 4.$$
$$norm(A, frobenius) = 2\sqrt{7}$$
$$norm(A, frobenius) = 5.291502622129182$$
$$norm(invA) = norm(invA, infinity) = \frac{9}{2}$$
$$norm(invA) = norm(invA, infinity) = 4.500000000000000$$
$$cond(A, 2) = 13.92820323027551$$
$$cond(A, 1) = 18.$$
$$cond(A) = cond(A, infinity) = 18.$$
$$cond(A, frobenius) = 20.73912030706970$$

Bei numerischer Rechnung erhalten wir für diese kleine Matrixdimension $n = 5$ bei den meisten Kommandos dieselben Ergebnisse. Für deutliche Abweichungen muss n wesentlich größer sein.

```
> Digits:=16:
  # A:=band([-1.0,2.0,-1.0],n);
  A:=matrix(n,n,(i,j)->if i=j then 2.0
                       elif abs(i-j)=1 then -1.0 else 0.0 end if):
  ...
```

$$rank(A) = 5$$
$$det(A) = 6.00000$$

$$inv(A) =$$
$$\begin{bmatrix} 0.8333333333333335 & 0.6666666666666670 & 0.5000000000000000 & 0.3333333333333337 & 0.1666666666666667 \\ 0.6666666666666667 & 1.333333333333334 & 1.000000000000000 & 0.6666666666666674 & 0.3333333333333334 \\ 0.5000000000000002 & 1.000000000000001 & 1.500000000000000 & 1.000000000000001 & 0.5000000000000001 \\ 0.3333333333333334 & 0.6666666666666670 & 1.000000000000000 & 1.333333333333334 & 0.6666666666666666 \\ 0.1666666666666667 & 0.3333333333333335 & 0.5000000000000001 & 0.6666666666666666 & 0.8333333333333333 \end{bmatrix}$$

$$norm(A, 2) = 3.732050807568877$$
$$norm(A, 1) = 4.0$$
$$norm(A) = norm(A, infinity) = 4.0$$
$$norm(A, frobenius) = 5.291502622129181$$
$$norm(invA) = norm(invA, infinity) = 4.500000000000002$$
$$cond(A, 2) = 13.92820323027552$$
$$cond(A, 1) = 18.00000000000000$$
$$cond(A) = cond(A, infinity) = 18.00000000000001$$
$$cond(A, frobenius) = 20.73912030706971$$

Berechnung der EW (Informationen aus der Maple-Hilfe)

- **Eigenvals(A)** returns an array of the eigenvalues of A. The function Eigenvals itself is inert. To actually compute the eigenvalues and eigenvectors, the user must evaluate the inert function in the floating point domain, by **evalf(Eigenvals(A))**.
- The call **eigenvalues(A)** returns for a symbolic case a sequence of the eigenvalues of A computed by solving the characteristic polynomial $\det(\lambda I - A) = 0$ or for larger dimension (greater than four) the eigenvalues are expressed using Maple's **RootOf** notation for algebraic extensions. If A contains floating-point numbers, a numerical method is used where all arithmetic is done at the precision specified by **Digits**.

```
> A:=matrix(n,n,(i,j)->if i=j then 2
                       elif abs(i-j)=1 then -1 else 0 end if):
> charpoly(A,lambda);
  Eigenvals(A);
  evalf(Eigenvals(A));
  eigenvals(A);
  evalf(eigenvals(A));
```

$$\lambda^5 - 10\,\lambda^4 + 36\,\lambda^3 - 56\,\lambda^2 + 35\,\lambda - 6$$

$$\text{Eigenvals}\left(\begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}\right)$$

[0.2679491924311242, 1.000000000000001, 2.000000000000000, 3.000000000000000, 3.732050807568877]

$$1,\ 2,\ 3,\ 2+\sqrt{3},\ 2-\sqrt{3}$$

1., 2., 3., 3.732050807568877, 0.267949192431123

Prozedur für die Berechnung der Spektralnorm über EW von AA^T bzw. A^TA

```
> norm2:=proc(A::matrix)
   local n,i,B,EVB,seq_EVB;
   B:=evalm(A&*transpose(A));
   n:=linalg[rowdim](B);
   EVB:=evalf(Eigenvals(B));
   seq_EVB:=seq(EVB[i],i=1..n);
   sqrt(max(seq_EVB));
  end:

> norm2(A);
  evalf(norm(A,2));   # =norm2(A)
```

3.732050807568877
3.732050807568877

Die Kondition der Tridiagonalmatrix A ist abhängig von ihrer Dimension n und verschlechtert sich langsam mit wachsendem n.

Spektrale Kondition der Tridiagonalmatrix

Rechnet man "symbolisch", dann dauern die damit verbundenen exakten Berechnungen sehr lange. Erst bei der Bestimmung von `erg` schleichen sich in der letzten Mantissenstelle wegen `Digits=...` dann kleine Rundungsfehler ein.

```
> Digits:=16:
  i:='i': j:='j':
  printf(` n          cond(T(n,n))\n`):
  for n from 1 by 10 to 101 do
   A:=matrix(n,n,(i,j)->if i=j then 2
                        elif abs(i-j)=1 then -1 else 0 end if):
   erg:=evalf(norm(A,2)*norm(inverse(A),2));
   printf(`%2d    %.15e  \n`,n,erg);
  end do:
```

```
 n          cond(T(n,n))
  1    1.000000000000000 e+00
 11    5.76954805409810 3e+01
 21    1.954914850025216 e+02
 31    4.143450622319016 e+02
 41    7.14255698384607 3e+02
 51    1.09522331645103 2e+03
 61    1.557247895814529 e+03
 71    2.100329429074228 e+03
 81    2.724467913042699 e+03
 91    3.429663346163696 e+03
101    4.21591572760405 2e+03
```

Bei numerischer Rechnung mit `A:=band([-1.0,2.0,-1.0],n);` gibt es mit wachsendem n zunehmende Abweichungen. Außerdem ist im untersuchten Breich der Dimension n das monotone Verhalten der Folge der Konditionswerte zu erkennen.
Die exakten Werte lassen sich am schnellsten mittels numerischer Rechnung bei großer Genauigkeit, z. B. mit `Digits=20`, ermitteln.
In der ersten Spalte der Tabelle ist die Vergrößerung der Anzahl der ungenauen letzten Mantissenstellen, wenn möglich, durch einen Abstand in der Ziffernfolge der Zahl deutlich gemacht.

```
 n          cond(T(n,n))          cond(T(n,n)) exakt
  1    1.000000000000000 e+00    1.000000000000000e+00
 11    5.76954805409810 3e+01    5.769548054098104e+01
 21    1.95491485002521 3e+02    1.954914850025216e+02
 31    4.1434506223190 04e+02    4.143450622319016e+02
 41    7.1425569838460 35e+02    7.142556983846071e+02
 51    1.09522331645103 0e+03    1.095223316451031e+03
 61    1.5572478958145 39e+03    1.557247895814529e+03
 71    2.1003294290742 47e+03    2.100329429074228e+03
 81    2.724467913042 727e+03    2.724467913042699e+03
 91    3.429663346163 766e+03    3.429663346163696e+03
101    4.215915727604 155e+03    4.215915727604050e+03
```

Um hier Maple mit MATLAB zu vergleichen, muss man natürlich seine numerische Rechnung verwenden. Wir betrachten die Durchführung des VGA für die Ermittlung der Inversen und die Berechnung der Determinanten $\det(A)$, $\det(A^{-1})$ bei wachsender Dimension n. Dabei kann die Pivotstrategie mit den Diagonalelementen arbeiten (Diagonalstrategie). Zu Grunde liegen Rechnungen in MATLAB (*double* Präzision) und Maple.

Generell wird sich bei numerischen Rechnungen die Anzahl der genauen Stellen im Ergebnis proportional zur wachsenden Dimension n langsam verringern.
Es werden Vergleiche mit der exakten Auswertung in Maple gemacht.

Die Konditionszahl $\text{cond}_\infty(A)$ kann aus der folgenden Tabelle ermittelt werden.

Programm	n	11	21	31	41	51
Maple	$\det(A)$	12.0000000000000	21.9999999999986	31.9999999999955	41.9999999999912	51.9999999999826
Digits=15	$\det(A^{-1})$ (a)	0.0833333333333288	0.0454545454545470	0.0312499999999926	0.0238095238095227	0.0192307692307882
	(b)	0.0833333333333352	0.0454545454545418	0.0312499999999965	0.0238095238095173	0.0192307692307561
	$\|A\|_\infty$	4.0	4.0	4.0	4.0	4.0
	$\|A^{-1}\|_\infty$ (a)	18.0000000000000	60.5000000000034	128.000000000011	220.500000000033	338.000000000067
	(b)	18.0000000000000	60.5000000000000	128.000000000000	220.500000000000	338.000000000000
Maple	$\det(A)$	12.00000000000000	22.00000000000006	32.00000000000016	42.00000000000022	51.99999999999987
Digits=16	$\det(A^{-1})$ (a)	0.0833333333333335	0.0454545454545568	0.0312499999999873	0.0238095238095362	0.0192307692307214
	(b)	0.0833333333333390	0.0454545454545549	0.0312499999999919	0.0238095238095377	0.0192307692307064
	$\|A\|_\infty$	4.0	4.0	4.0	4.0	4.0
	$\|A^{-1}\|_\infty$ (a)	18.00000000000000	60.49999999999992	127.9999999999996	220.4999999999988	337.9999999999994
	(b)	18.00000000000000	60.50000000000000	128.0000000000000	220.5000000000000	338.0000000000000
Maple	$\det(A)$	12.000000000000000	21.999999999999992	31.999999999999973	41.999999999999964	51.999999999999990
Digits=17	$\det(A^{-1})$ (a)	0.083333333333333372	0.045454545454545477	0.031250000000000042	0.023809523809523798	0.019230769230768982
	(b)	0.083333333333333343	0.045454545454545476	0.031250000000000037	0.023809523809523707	0.019230769230769435
	$\|A\|_\infty$	4.0	4.0	4.0	4.0	4.0
	$\|A^{-1}\|_\infty$ (a)	17.999999999999999	60.500000000000024	128.00000000000007	220.50000000000014	338.00000000000010
	(b)	18.000000000000000	60.500000000000000	128.00000000000000	220.50000000000000	338.00000000000000
Maple	$\det(A)$	12	22	32	42	52
exakt	$\det(A^{-1})$	$\frac{1}{12}$=0.0833...	$\frac{1}{22}$=0.04545...	$\frac{1}{32}$=0.03125	$\frac{1}{42}$=0.0238095238095...	$\frac{1}{52}$=0.01923076923076...
	$\|A\|_\infty$	4	4	4	4	4
	$\|A^{-1}\|_\infty$	18	60.5	128	220.5	338
MATLAB	$\det(A)$	12	22	32	42	52
double	$\det(A^{-1})$ (a)	0.08333333333333	0.04545454545455	0.03125000000000	0.02380952380952	0.01923076923077
	$\|A\|_\infty$	4	4	4	4	4
	$\|A^{-1}\|_\infty$ (a)	18	60.50000000000000	127.9999999999997	220.4999999999986	337.9999999999973
	(b)	18.00000000000001	60.50000000000013	128.0000000000005	220.5000000000011	338.0000000000013

Tab. 3.1 Berechnungen für die Tridiagonalmatrix mit Maple und MATLAB, wobei die Inverse numerisch berechnet wird und zur Ermittlung der Inversen 2 Varianten genommen werden

Maple
(a) `inverse(A)`
(b) `linsolve(A,I)`

MATLAB
(a) `inv(A)`
(b) `A\I`

Die Determinantenberechnung $\det(A^{-1})$ mit MATLAB-Variante (b) hat das gleiche Ergebnis wie bei Variante (a).
MATLAB (Datei *matr_trid1.m*) liefert in den meisten unserer Fälle etwas bessere Ergebnisse als Maple bei `Digits=16` und `inverse(A)`, aber ist schon ungenauer als Maple bei `Digits=16` und `linsolve(A,I)`. Es kann nicht ganz konkurrieren mit Maple bei `Digits=17`. Maple bei `Digits=15` zeigt schon größere Ungenauigkeiten.
Betreffend der Variantenwahl bei der Inversen in Maple ist die Variante mit `linsolve(A,I)` etwas genauer als `inverse(A)`.
Bezüglich der Variantenwahl bei der Matrixinversen in MATLAB, die hier in der Normberechnung enthalten ist, gibt es keinen endgültigen Aufschluss über Vorteile/Nachteile, denn für unterschiedliche Dimensionen n scheint mal `inv(A)`, mal `A\I` günstiger zu sein.

3.5 Matrix 5

Sei $A = A(n,n) = (a_{ij})$ die Hilbert-Matrix. Ihre Elemente sind

$$a_{ij} = \frac{1}{i+j-1}.$$

Sie ist spd. Ihre Inverse $A^{-1} = (a'_{ij})$ hat die ganzzahligen Elemente

$$a'_{ij} = \frac{(-1)^{i+j}}{i+j-1}\gamma_i\gamma_j, \quad \gamma_i = \frac{(n+i-1)!}{(i-1)!^2(n-i)!}, \quad i,j = 1,2,...,n.$$

Für $n = 5$ erhält man

$$A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{pmatrix},$$

$$A^{-1} = \begin{pmatrix} 25 & -300 & 1\,050 & -1\,400 & 630 \\ -300 & 4\,800 & -18\,900 & 26\,880 & -12\,600 \\ 1\,050 & -18\,900 & 79\,380 & -117\,600 & 56\,700 \\ -1\,400 & 26\,880 & -117\,600 & 179\,200 & -88\,200 \\ 630 & -12\,600 & 56\,700 & -88\,200 & 44\,100 \end{pmatrix}.$$

Rechnungen für die Hilbert-Matrix in Maple
Determinante, Inverse, Norm, Kondition und EW/EV (Datei *matr_hilb1.mws*)

```
> Digits:=16:
  i:='i': j:='j':
  n:=5:
  A:=matrix(n,n,(i,j)->1/(i+j-1)):              # hilbert(n)
  `A`=evalm(A);
  `rank(A)`=rank(A);
  `det(A)`=det(A);
  `inv(A)`=inverse(A);

  # Normen
  `norm(A,2)`=norm(A,2);                        # Spektralnorm
  evalf(%);
  `norm(A,1)`=norm(A,1);                        # Spaltensummennorm
  evalf(%);
  `norm(A)=norm(A,infinity)`=norm(A);           # Zeilensummennorm = SSN
  evalf(%);
  `norm(A,frobenius)`=norm(A,frobenius);        # Frobenius-Norm
  evalf(%);
  `norm(invA)=norm(invA,infinity)`=norm(invA); # Zeilensummennorm
```

```
evalf(%);

# Konditionen dazu, auch Kommando cond(A,*)
‘cond(A,2)‘=evalf(norm(A,2)*norm(inverse(A),2));
‘cond(A,1)‘=evalf(norm(A,1)*norm(inverse(A),1));
‘cond(A)=cond(A,infinity)‘=evalf(norm(A)*norm(inverse(A)));
‘cond(A,frobenius)‘=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
```

$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}$$

$$rank(A) = 5$$

$$det(A) = \frac{1}{266716800000}$$

$$inv(A) = \begin{bmatrix} 25 & -300 & 1050 & -1400 & 630 \\ -300 & 4800 & -18900 & 26880 & -12600 \\ 1050 & -18900 & 79380 & -117600 & 56700 \\ -1400 & 26880 & -117600 & 179200 & -88200 \\ 630 & -12600 & 56700 & -88200 & 44100 \end{bmatrix}$$

$$norm(A, 2) =$$

$$\frac{1}{5}\text{RootOf}(-1 + 3700542505\,\_Z - 1582832489513760\,\_Z^2 + 487666609069973760\,\_Z^3$$
$$-455148325561466880\,\_Z^4 + 7284515983589376\,\_Z^5,\ index = 5)^{1/2}$$

$$norm(A, 2) = 1.567050691098231$$

$$norm(A, 1) = \frac{137}{60}$$

$$norm(A, 1) = 2.283333333333333$$

$$norm(A) = norm(A, infinity) = \frac{137}{60}$$

$$norm(A) = norm(A, infinity) = 2.283333333333333$$

$$norm(A, frobenius) = \frac{\sqrt{15871330}}{2520}$$

$$norm(A, frobenius) = 1.580906263272022$$

$$norm(invA) = norm(invA, infinity) = 413280$$

$$norm(invA) = norm(invA, infinity) = 413280.$$

$$cond(A, 2) = 476607.2502425608$$

$$cond(A, 1) = 943656.$$

$$cond(A) = cond(A, infinity) = 943656.$$

$$cond(A, frobenius) = 480849.1169947188$$

Bei numerischer Rechnung erhalten wir für diese kleine Matrixdimension 5 nur bei einigen Kommandos abweichende Ergebnisse.

```
> Digits:=16:
  A:=matrix(n,n,(i,j)->1.0/(i+j-1)):              # hilbert(n)
  ...
```

$$rank(A) = 5$$

$$det(A) = 0.37492951261350\ 10^{-1}$$

$$inv(A) =$$

$$\begin{bmatrix} 25.00000000002770 & -300.0000000003670 & 1050.000000001230 & -1400.000000001490 & 630.0000000006000 \\ -300.0000000003569 & 4800.000000003932 & -18900.00000001032 & 26880.00000000898 & -12600.00000000208 \\ 1050.000000001140 & -18900.00000000969 & 79380.00000001349 & -117599.9999999923 & 56699.99999998649 \\ -1400.000000001326 & 26880.00000000740 & -117599.9999999896 & 179199.9999999424 & -88199.99999995733 \\ 630.0000000005108 & -12600.00000000110 & 56699.99999998421 & -88199.99999995584 & 44099.99999997138 \end{bmatrix}$$

$$norm(A, 2) = 1.567050691098230$$

$$norm(A, 1) = 2.283333333333333$$

$$norm(A) = norm(A, infinity) = 2.283333333333333$$

$$norm(A, frobenius) = 1.580906263272022$$

$$norm(invA) = norm(invA, infinity) = 413279.9999998980$$

$$cond(A, 2) = 476607.2502424522$$

$$cond(A, 1) = 943655.9999997738$$

$$cond(A) = cond(A, infinity) = 943655.9999997670$$

$$cond(A, frobenius) = 480849.1169946093$$

Berechnung der EW

```
> charpoly(A,lambda);
  Eigenvals(A);
  evalf(Eigenvals(A));
  eigenvals(A);
  evalf(eigenvals(A));
```

$$\lambda^5 - \frac{563}{315}\lambda^4 + \frac{735781}{2116800}\lambda^3 - \frac{852401}{222264000}\lambda^2 + \frac{61501}{53343360000}\lambda - \frac{1}{266716800000}$$

$$\text{Eigenvals}\left(\begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} \end{bmatrix}\right)$$

$$[0.3287928771915000\,10^{-5},\ 0.0003058980401511738,\ 0.01140749162341977,$$
$$0.2085342186110133,\ 1.567050691098231]$$

$$\%1 := -1 + 61501\,\_Z - 40915248\,\_Z^2 + 741667248\,\_Z^3 - 762725376\,\_Z^4 + 85349376\,\_Z^5$$

$$\frac{1}{5}\,\text{RootOf}(\%1,\ index = 1),\ \frac{1}{5}\,\text{RootOf}(\%1,\ index = 2),\ \frac{1}{5}\,\text{RootOf}(\%1,\ index = 3),$$

$$\frac{1}{5}\,\mathrm{RootOf}(\%1,\ index=4),\ \frac{1}{5}\,\mathrm{RootOf}(\%1,\ index=5)$$
$$0.3287928772171862\,10^{-5},\ 0.0003058980401511918,\ 0.01140749162341981,$$
$$0.2085342186110134,\ 1.567050691098231$$

Verwendung der Prozedur `norm2` für die Berechnung der Spektralnorm über EW von AA^T bzw. A^TA wie bei Matrix 4

```
> norm2(A);
  evalf(norm(A,2));   # geringfuegig genauer
```

$$1.567050691098230$$
$$1.567050691098231$$

Die Kondition der Hilbert-Matrix A ist abhängig von ihrer Dimension n und verschlechtert sich mit wachsendem n.

Spektrale Kondition der Hilbert-Matrix

Rechnet man "symbolisch", dann dauern die damit verbundenen exakten Berechnungen sehr lange. Erst bei der Bestimmung von `erg` schleichen sich in der letzten Mantissenstelle wegen `Digits=...` dann Rundungsfehler ein. Dabei ist nicht jeder Rundungseffekt plausibel, denn in der folgenden Tabelle hätte man bei $n = 7$ eigentlich den Wert `4.753673549881790e+08` erwartet.

```
> Digits:=16:
  i:='i': j:='j':
  printf(` n          cond(H(n,n))\n`):
  for n from 1 to 19 do
   A:=matrix(n,n,(i,j)->1/(i+j-1)):
   erg:=evalf(norm(A,2)*norm(inverse(A),2));
   printf(`%2d    %.15e  \n`,n,erg);
  end do:
```

```
 n          cond(H(n,n))              Rundungseffekt
 1    1.000000000000000 e+00
 2    1.928147006790397 e+01
 3    5.24056777586060 8e+02
 4    1.55137387389325 9e+04   <-  1.551373873893258 5328...e+04
 5    4.766072502425608 e+05
 6    1.495105864013122 e+07
 7    4.75367354988178 9e+08   <-  4.753673549881789 7237...e+08
 8    1.525757574164694 e+10
 9    4.93154926971542 1e+11
10    1.602628687021688 e+13
11    5.23067739242940 9e+14
12    1.71322890469700 5e+16
13    5.627942373760076 e+17
14    1.85338170234715 0e+19
15    6.11656579161984 1e+20
16    2.02234591767453 0e+22
17    6.69743898056063 1e+23
18    2.22119003943386 5e+25
19    7.37595118050884 2e+26
```

Bei numerischer Rechnung mit `A:=matrix(n,n,(i,j)->1.0/(i+j-1))` gibt es mit wachsendem n deutliche Abweichungen. Außerdem ist das monotone Verhalten der Folge der Konditionswerte verletzt.
Die exakten Werte lassen sich am schnellsten mittels numerischer Rechnung bei sehr großer Genauigkeit, z. B. mit `Digits=50`, ermitteln.
In der ersten Spalte der Tabelle ist die Vergrößerung der Anzahl der ungenauen letzten Mantissenstellen, wenn möglich, durch einen Abstand in der Ziffernfolge der Zahl deutlich gemacht.

```
 n          cond(H(n,n))          cond(H(n,n)) exakt
 1    1.000000000000000 e+00    1.000000000000000e+00
 2    1.928147006790397 e+01    1.928147006790397e+01
 3    5.240567775860 722e+02    5.240567775860607e+02
 4    1.5513738738 84959e+04    1.551373873893258e+04
 5    4.76607250242 4522e+05    4.766072502425608e+05
 6    1.49510586 3908718e+07    1.495105864013122e+07
 7    4.7536735 58150294e+08    4.753673549881790e+08
 8    1.52575757 2620145e+10    1.525757574164694e+10
 9    4.93155 0264460760e+11    4.931549269715423e+11
10    1.6026 48177259556e+13    1.602628687021688e+13
11    5.230 219007896473e+14    5.230677392429410e+14
12    1. 697041441020638e+16    1.713228904697006e+16
13     4.650924592519341e+17    5.627942373760076e+17
14     8.552342219037117e+17    1.853381702347149e+19
15     9.092568079649491e+17    6.116565791619842e+20
16     5.439364263014374e+17    2.022345917674529e+22
17     4.877383693269593e+18    6.697438980560632e+23
18     2.918211913751011e+18    2.221190039433866e+25
19     2.928278176299190e+18    7.375951180508844e+26
```

Um hier Maple mit MATLAB (Datei *matr_hilb1.m*) oder TP zu vergleichen, muss man natürlich seine numerische Rechnung verwenden.
Wir betrachten die Durchführung des verketteten Gauß-Algorithmus (VGA) mit Spaltenpivotisierung für die Ermittlung der näherungsweisen Inversen $\widetilde{A}^{-1}$ und der Determinante $\det(A)$ bei wachsender Dimension n. Dabei tendieren die Pivotelemente gegen Null.
Zu Grunde liegen Implementierungen in TP mit den GPF *double* (64 Binärstellen, 15–16 Dezimalstellen der Mantisse) und *extended* (80 Binärstellen, 19–20 Dezimalstellen der Mantisse) sowie Rechnungen in MATLAB (*double* Präzision) und Maple.

Generell wird sich bei numerischen Rechnungen die Anzahl der genauen Stellen im Ergebnis proportional zur wachsenden Dimension n verringern.
So ist z. B. bei der Berechnung von $\det(A)$ mit TP *double* Präzision bei $n = 13$ höchsten noch die Größenordnung des Wertes verlässlich, bei $n = 16$ nicht einmal diese, was der Vergleich mit der exakten Auswertung in Maple zeigt.
Die Konditionszahl $\mathrm{cond}_\infty(A)$ kann aus den folgenden Tabellen ermittelt werden.

Programm	n	7	10	13	16	19
TP	$\det(A)$	4.836E–25	2.164E–53	2.616E–92	1.843E–135	-3.854E–180
double	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.800E+8	1.207E+13	2.303E+17	5.724E+17	5.889E+17
TP	$\det(A)$	4.836E–25	2.164E–53	1.444E–92	-3.193E–141	1.050E–192
extended	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$	3.800E+8	1.207E+13	4.161E+17	7.324E+20	7.494E+20
MATLAB	$\det(A)$	4.836E–25	2.164E–53	4.448E–92	2.424E–135	-2.232E–180
double	$\det(A^{-1})$ (a)	2.068E+24	4.620E+52	2.519E+91	4.257E+134	-1.100E+180
	$\|A\|_\infty$	2.593	2.929	3.180	3.381	3.548
	$\|A^{-1}\|_\infty$ (a)	3.800E+8	1.207E+13	1.345E+17	6.838E+17	1.942E+18
	(b)	3.800E+8	1.207E+13	3.876E+18	6.838E+17	1.942E+18

Tab. 3.2 Berechnungen für die Hilbert-Matrix mit TP und MATLAB, wobei die Inverse numerisch berechnet wird und bei MATLAB 2 Varianten der Ermittlung der Inversen genommen werden
(a) `inv(A)` (ab $n \geq 12$ mit Warnungen),
(b) `A\I`

Für die Dimensionen $n = 16$ und 19 sind die Berechnungen der Determinante und Kondition der Matrix A in TP bzw. MATLAB mit der vorliegenden GPA nicht vertretbar.

Bei MATLAB erkennt man noch eine besondere Situation für $n = 13$.
Deshalb soll die Zeilensummennorm der inversen Matrix in dieser Umgebung noch betrachtet werden.

n	11	12	13	14	15	16
$\|A^{-1}\|_\infty$ (a)	4.074E+14	1.224E+16	1.345E+17	1.833E+18	2.420E+17	6.838E+17
(b)	4.084E+14	1.311E+16	3.876E+18	1.833E+18	2.420E+17	6.838E+17
(c)	4.085E+14	1.326E+16	4.165E+17	1.396E+19	4.639E+20	1.498E+22
exakt	4.085E+14	1.326E+16	4.165E+17	1.396E+19	4.639E+20	1.498E+22

Tab. 3.3 Zeilensummennorm für die inverse Hilbert-Matrix mit MATLAB, wobei
(a) `inv(A)` (ab $n \geq 12$ mit Warnungen)
(b) `A\I`
(c) `invhilb(n)`

Gemäß MATLAB-HIlfe erzeugt das Kommando `invhilb(n)` die exakte ganzzahlige inverse Hilbert-Matrix bis zur Dimension ungefähr $n = 15$. Rechnungen ergeben, dass die Auswertung der ∞-Norm der Näherungsinversen $\widetilde{A}^{-1}$ gemäß `invhilb(n)` auch noch für die nächsten Werte n sehr gut ist.
Man sollte jedoch hier bei der Verwendung von `invhilb(n)` nicht einer Illusion unterliegen. Für $n = 11$ enthält $\widetilde{A}^{-1} = (a'_{ij})$ =`invhilb(n)` ganzzahlige Elemente (Zahlenwerte) mit maximal 15-stelliger Mantisse. Diese Zahlen sind im *double*-Format noch richtig erfassbar. Aber für $n = 15$ ist die Mantisse schon 20-stellig, obwohl die letzten Dezimalstellen Nullen sind. So ist bei $\widetilde{A}^{-1}(15, 15)$ das Element $a'_{11,11} = 11\,470\,898\,792\,429\,076\,000$, es sind also 17 signifikante Mantissenstellen. Sobald also mit solchen Matrizen weitere numerische Rechnungen erfolgen, z. B. Determinantenberechnung, Multiplikation mit Matrix oder Vektor, werden natürlich Rundungsfehler und Genauigkeitsprobleme auftreten.
Im nächsten Kapitel werden diese Aspekte nochmal untersucht.

Programm	n	7	10	13
Maple	$\det(A)$	4.835802 52616001E–25	2.16 675736755201E–53	-5.87256512146775E–91
Digits=15	$\det(A^{-1})$ (a)	2.067909 12288620E+24	4.6 1629649910000E+52	-2.40645333085835E+90
	(b)	2.067909 29297571E+24	4.6 1520203510628E+52	4.34941183746083E+90
	$\|A\|_\infty$	2.59285714285714	2.92896825396825	3.18013375513375
	$\|A^{-1}\|_\infty$ (a)	3.7996497 7713027E+8	1.20 572705449516E+13	1.25046677424904e17E+16
	(b)	3.7996497 9984073E+8	1.20 667670830567E+13	2.10900728593740e17E+16
Maple	$\det(A)$	4.835802615694910E–25	2.1641 53786793289E–53	1. 760285512154180E–92
Digits=16	$\det(A^{-1})$ (a)	2.0679090 55655154E+24	4.62 1756491407878E+52	7.581245893324645E+91
	(b)	2.06790904 1363235E+24	4.62 2105554445228E+52	-6.441938922422012E+91
	$\|A\|_\infty$	2.59285714285714 3	2.928968253968254	3.180133755133755
	$\|A^{-1}\|_\infty$ (a)	3.79964970 6587486E+8	1.2071 78345000572E+13	3.438677627898522E+17
	(b)	3.7996497000 70217E+8	1.2071 52634107370E+13	3.925461294783808E+17
Maple	$\det(A)$	4.8358026223374353E–25	2.164 2025787149194E–53	1. 7641442571635657E–92
Digits=17	$\det(A^{-1})$ (a)	2.06790904 84893457E+24	4.6206 006842351238E+52	6. 4274780121754340E+91
	(b)	2.067909047 0725904E+24	4.620 5755764020611E+52	1.0428968398910485E+92
	$\|A\|_\infty$	2.592857142857142 9	2.9289682539682540	3.1801337551337551
	$\|A^{-1}\|_\infty$ (a)	3.79964970 12524995E+8	1.2071 506035957616E+13	3.4105050502292530E+17
	(b)	3.799649700 5554529E+8	1.20716 46123663961E+13	4. 8120354053904484E+17
Maple	$\det(A)$	4.8358026239261169E–25	2.1641792264314919E–53	1.4428965187911365E–92
exakt	$\det(A^{-1})$	2.0679090479257706496E+24	4.6206893947914691E+52	6.9305039341130527E+91
	$\|A\|_\infty$	$\frac{363}{140}$ =2.59285714285714...	$\frac{7381}{2520}$ =2.928968253968253...	$\frac{1145993}{360360}$ =3.180133755133755...
	$\|A^{-1}\|_\infty$	379964970	1.207163621664E+13	4.1646330343706088E+17
MATLAB	$\det(A)$	4.8358026 08198562E–25	2.164 405264621389E–53	4.448044215086199E–92
double	$\det(A^{-1})$ (a)	2.0679090 52374171E+24	4.620 187171197534E+52	2.518875103143657E+91
	(b)	2.0679090 53818976E+24	4.620 759775107971E+52	-5.439678636373364E+93
	(c)	2.06790904 8252736E+24	4.6206893 67315927E+52	-1.032669986022261E+91
	$\|A\|_\infty$	2.59285714285714	2.92896825396825	3.18013375513376
	$\|A^{-1}\|_\infty$ (a)	3.7996497 12418148E+8	1.207 036795494732E+13	1.344582864279075E+17
	(b)	3.7996497 19219860E+8	1.207 201456347934E+13	3.875544667200439E+18
	(c)	379964970	1.207163621664000E+13	4.164633034370609E+17

Programm	n	16	19
Maple	$\det(A)$	1.190E–132	-5.405E-174
Digits=15	$\det(A^{-1})$ (a)	3.237E+132	-1.058E+174
	(b)	-1.395E+133	1.259E+173
	$\|A\|_\infty$	3.38072899322899	3.54773965714368
	$\|A^{-1}\|_\infty$ (a)	4.213E+17	6.647E+16
	(b)	8.005E+17	3.434E+16
Maple	$\det(A)$	7.111E–135	2.230E–179
Digits=16	$\det(A^{-1})$ (a)	2.189E+134	-5.356E+179
	(b)	2.642E+134	-1.075E+178
	$\|A\|_\infty$	3.380728993228993	3.547739657143682
	$\|A^{-1}\|_\infty$ (a)	4.307E+17	2.778E+18
	(b)	3.346E+17	4.466E+17
Maple	$\det(A)$	-1.368E–136	-1.306E-183
Digits=17	$\det(A^{-1})$ (a)	-2.584E+134	-3.487E+183
	(b)	-1.052E+137	-5.373E+183
	$\|A\|_\infty$	3.3807289932289932	3.5477396571436819
	$\|A^{-1}\|_\infty$ (a)	1.129E+18	5.515E+18
	(b)	2.879E+18	1.267E+19
Maple	$\det(A)$	1.419E–142	2.049E–203
exakt	$\det(A^{-1})$	7.047E+141	4.880E+202
	$\|A\|_\infty$	$\frac{2436559}{720720}$ =3.3807289932289932...	$\frac{275295799}{77597520}$ =3.5477396571436
	$\|A^{-1}\|_\infty$	1.4975393761665516488 4E+22	5.428161213438534455207232E+26
MATLAB	$\det(A)$	2.424E–135	-2.232E–180
double	$\det(A^{-1})$ (a)	4.257E+134	-1.100E+180
	(b)	5.575E+134	1.229E+180
	(c)	-5.240E+148	1.401E+226
	$\|A\|_\infty$	3.38072899322899	3.54773965714368
	$\|A^{-1}\|_\infty$ (a)	6.838E+17	1.942E+18
	(b)	6.838E+17	1.942E+18
	(c)	1.497539376166551E+22	5.428161213438534E+26

Tab. 3.4 Berechnungen für die Hilbert-Matrix mit Maple und MATLAB, wobei die Inverse numerisch berechnet wird und zur Ermittlung der Inversen 2 Varianten genommen werden

	Maple	MATLAB
(a)	`inverse(A)`	`inv(A)`
(b)	`linsolve(A,I)`	`A\I`
(c)		`invhilb(n)`

MATLAB liegt mit seiner Genauigkeit i. Allg. bei Maple mit `Digits=16,17`. Bei $n = 13$ liefert Maple mit `Digits=16,17` bessere Ergebnisse als MATLAB und TP mit dem GPF *double*. Für die Dimensionen $n = 16$ und 19 sind die Werte von Determinante und Kondition der Matrix A in MATLAB und Maple mit der vorliegenden Genauigkeit `Digits=...` nicht akzeptabel. Ausnahme sind die unproblematische Normberechnung $\|A\|_\infty$ sowie MATLAB mit der Normberechnung $\|A^{-1}\|_\infty$ in dem Fall, wo die Inverse mit der speziellen Variante `invhilb(A)` für die betrachteten Dimensionen n ermittelt wird.

3.6 Matrix 6

Gegeben sei die nicht symmetrische ganzzahlige Boothroyd/Dekker-Matrix $A(n,n) = (a_{ij})$ mit

$$a_{ij} = \binom{n+i-1}{i-1}\binom{n-1}{n-j}\frac{n}{i+j-1} = \frac{(n+i-1)!}{(n-j)!\,(i-1)!\,(j-1)!\,(i+j-1)}.$$

Es gilt $\det(A) = 1$ und ihre Inverse $A^{-1} = (a'_{ij})$ hat die Elemente

$$a'_{ij} = (-1)^{i+j}a_{ij}, \quad i,j = 1,2,...,n.$$

Für $n = 5$ erhält man

$$A = \begin{pmatrix} 5 & 10 & 10 & 5 & 1 \\ 15 & 40 & 45 & 24 & 5 \\ 35 & 105 & 126 & 70 & 15 \\ 70 & 224 & 280 & 160 & 35 \\ 126 & 420 & 540 & 315 & 70 \end{pmatrix},$$

$$A^{-1} = \begin{pmatrix} 5 & -10 & 10 & -5 & 1 \\ -15 & 40 & -45 & 24 & -5 \\ 35 & -105 & 126 & -70 & 15 \\ -70 & 224 & -280 & 160 & -35 \\ 126 & -420 & 540 & -315 & 70 \end{pmatrix}.$$

Damit haben die Matrizen A und A^{-1} die gleichen Normen und Konditionen.

Rechnungen für die Boothroyd/Dekker-Matrix in Maple
Determinante, Inverse, Norm, Kondition und EW/EV (Datei *matr_booth1.mws*)

```
> Digits:=16:
  i:='i': j:='j':
  n:=5:
  A:=matrix(n,n):
  for i from 1 to n do
   for j from 1 to n do
    A[i,j]:=binomial(n+i-1,i-1)*binomial(n-1,n-j)*n/(i+j-1)
   end do:
  end do:
  # analog
  for i from 1 to n do
   for j from 1 to n do
    A[i,j]:=(n+i-1)!/((n-j)!*(i-1)!*(j-1)!*(i+j-1));
   end do:
  end do:
  `A`=evalm(A);

  `rank(A)`=rank(A);
  `det(A)`=det(A);
  `inv(A)`=inverse(A);
```

```
# Normen
‘norm(A,2)‘=norm(A,2);                          # Spektralnorm
evalf(%);
‘norm(A,1)‘=norm(A,1);                          # Spaltensummennorm
evalf(%):
‘norm(A)=norm(A,infinity)‘=norm(A);             # Zeilensummennorm
evalf(%):
‘norm(A,frobenius)‘=norm(A,frobenius);          # Frobenius-Norm
evalf(%);
‘norm(invA)=norm(invA,infinity)‘=norm(invA); # Zeilensummennorm, =norm(A)
evalf(%):

# Konditionen dazu, auch Kommando cond(A,*)
‘cond(A,2)‘=evalf(norm(A,2)*norm(inverse(A),2));
‘cond(A,1)‘=evalf(norm(A,1)*norm(inverse(A),1));
‘cond(A)=cond(A,infinity)‘=evalf(norm(A)*norm(inverse(A)));
‘cond(A,frobenius)‘=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
```

$$A = \begin{bmatrix} 5 & 10 & 10 & 5 & 1 \\ 15 & 40 & 45 & 24 & 5 \\ 35 & 105 & 126 & 70 & 15 \\ 70 & 224 & 280 & 160 & 35 \\ 126 & 420 & 540 & 315 & 70 \end{bmatrix}$$

$$rank(A) = 5$$

$$det(A) = 1$$

$$inv(A) = \begin{bmatrix} 5 & -10 & 10 & -5 & 1 \\ -15 & 40 & -45 & 24 & -5 \\ 35 & -105 & 126 & -70 & 15 \\ -70 & 224 & -280 & 160 & -35 \\ 126 & -420 & 540 & -315 & 70 \end{bmatrix}$$

$$norm(A,2) = \sqrt{\mathrm{RootOf}(\_Z^4 - 786254\,\_Z^3 + 132041256\,\_Z^2 - 786254\,\_Z + 1,\ index = 4)}$$

$$norm(A,2) = 886.6149259774036$$

$$norm(A,1) = 1001$$

$$norm(A) = norm(A, infinity) = 1471$$

$$norm(A, frobenius) = \sqrt{786255}$$

$$norm(A, frobenius) = 886.7102119632998$$

$$norm(invA) = norm(invA, infinity) = 1471$$

$$cond(A,2) = 786086.0269659169$$

$$cond(A,1) = 1.002001\ 10^6$$

$$cond(A) = cond(A, infinity) = 2.163841\ 10^6$$

$$cond(A, frobenius) = 786255.$$

Bei numerischer Rechnung erhalten wir für diese kleine Matrixdimension 5 nur bei einigen Kommandos abweichende Ergebnisse.

```
> Digits:=16:
  A:=matrix(n,n,(i,j)->binomial(n+i-1,i-1)*binomial(n-1,n-j)*n/(i+j-1.0)):
  ...
```

$$rank(A) = 5.$$

$$det(A) = 1.0000000000000$$

$$inv(A) =$$

$$\begin{bmatrix} 4.999999999991222 & -9.999999999973650 & 9.999999999967857 & -4.999999999982143 & 0.9999999999961666 \\ -14.99999999996482 & 39.99999999989277 & -44.99999999986993 & 23.99999999992742 & -4.999999999984400 \\ 34.99999999990764 & -104.9999999997175 & 125.9999999996572 & -69.99999999980846 & 14.99999999995883 \\ -69.99999999980350 & 223.9999999993973 & -279.9999999992680 & 159.9999999995907 & -34.99999999991200 \\ 125.9999999996328 & -419.9999999988712 & 539.9999999986278 & -314.9999999992324 & 69.99999999983490 \end{bmatrix}$$

$$norm(A, 2) = 886.6149259774036$$

$$norm(A, 1) = 1001.000000000000$$

$$norm(A) = norm(A, infinity) = 1471.000000000000$$

$$norm(A, frobenius) = 886.7102119632998$$

$$norm(invA) = norm(invA, infinity) = 1470.999999996199$$

$$cond(A, 2) = 786086.0269638766$$

$$cond(A, 1) = 1.002000999997388\ 10^6$$

$$cond(A) = cond(A, infinity) = 2.163840999994409\ 10^6$$

$$cond(A, frobenius) = 786254.9999979597$$

Berechnung der EW

```
> charpoly(A,lambda);
  Eigenvals(A);
  evalf(Eigenvals(A));
  eigenvals(A);
  evalf(eigenvals(A));
```

$$\lambda^5 - 401\,\lambda^4 + 4498\,\lambda^3 - 4498\,\lambda^2 + 401\,\lambda - 1$$

$$\text{Eigenvals}\left(\begin{bmatrix} 5 & 10 & 10 & 5 & 1 \\ 15 & 40 & 45 & 24 & 5 \\ 35 & 105 & 126 & 70 & 15 \\ 70 & 224 & 280 & 160 & 35 \\ 126 & 420 & 540 & 315 & 70 \end{bmatrix}\right)$$

$$[389.4809409975025,\ 10.42052704685171,\ 0.09596443591611827,$$
$$0.002567519728805225,\ 0.9999999999999948]$$

$$\%1 := 1 - 400\,\_Z + 4098\,\_Z^2 - 400\,\_Z^3 + \_Z^4$$

$$1,\ \text{RootOf}(\%1,\ index = 1),\ \text{RootOf}(\%1,\ index = 2),\ \text{RootOf}(\%1,\ index = 3),\ \text{RootOf}(\%1,\ index = 4)$$

$$1.,\ 0.002567519728793122,\ 0.09596443591613984,\ 10.42052704685168,\ 389.4809409975034$$

Verwendung der Prozedur `norm2` für die Berechnung der Spektralnorm über EW von AA^T bzw. A^TA wie bei Matrix 4

```
> norm2(A);
  evalf(norm(A,2));   # =norm2(A)
```

886.6149259774036
886.6149259774036

Die Kondition der Boothroyd/Dekker-Matrix A ist abhängig von ihrer Dimension n und verschlechtert sich mit wachsendem n.

Spektrale Kondition der Boothroyd/Dekker-Matrix

Rechnet man "symbolisch", dann dauern die damit verbundenen exakten Berechnungen sehr lange. Erst bei der Bestimmung von `erg` können sich in der letzten Mantissenstelle wegen `Digits=...` dann Rundungsfehler einschleichen.

```
> Digits:=16:
  i:='i': j:='j':
  printf(` n          cond(B(n,n))\n`):
  for n from 1 to 19 do
   A:=matrix(n,n,(i,j)->binomial(n+i-1,i-1)*binomial(n-1,n-j)*n/(i+j-1)):
   erg:=evalf(norm(A,2)*norm(inverse(A),2));
   printf(`%2d    %.15e  \n`,n,erg);
  end do:
```

```
 n          cond(B(n,n))
 1    1.000000000000000 e+00
 2    1.794427190999916 e+01
 3    4.879979508110673 e+02
 4    1.820120835388574 e+04
 5    7.86086026965916 9e+05
 6    3.687493479922402 e+07
 7    1.825083259346978 e+09
 8    9.38053920373871 0e+10
 9    4.958529644288436 e+12
10    2.678482366677917 e+14
11    1.472028569123609 e+16
12    8.20445562727671 1e+17
13    4.62653872741545 3e+19
14    2.634780213053048 e+21
15    1.51319552059205 9e+23
16    8.75415555808713 9e+24
17    5.09684922102875 2e+26
18    2.984205687512471 e+28
19    1.755988370979872 e+30
```

Bei numerischer Rechnung mit
`A:=matrix(n,n,(i,j)->binomial(n+i-1,i-1)*binomial(n-1,n-j)*n/(i+j-1.0))` gibt es mit wachsendem n deutliche Abweichungen. Außerdem ist das monotone Verhalten der Folge der Konditionswerte verletzt.

Die "exakten" Werte lassen sich am schnellsten mittels numerischer Rechnung bei sehr großer Genauigkeit, z. B. mit `Digits=50`, ermitteln.
In der ersten Spalte der Tabelle ist die Vergrößerung der Anzahl der ungenauen letzten Mantissenstellen wenn möglich durch einen Abstand in der Ziffernfolge der Zahl deutlich gemacht.

```
 n          cond(B(n,n))          cond(B(n,n)) exakt
 1    1.000000000000000 e+00    1.000000000000000e+00
 2    1.794427190999916 e+01    1.794427190999916e+01
 3    4.879979508110673 e+02    4.879979508110673e+02
 4    1.820120835388574 e+04    1.820120835388574e+04
 5    7.8608602696 38764e+05    7.860860269659167e+05
 6    3.6874934799 59106e+07    3.687493479922402e+07
 7    1.82508325 0143127e+09    1.825083259346978e+09
 8    9.380539 469972827e+10    9.380539203738713e+10
 9    4.95852 3200936041e+12    4.958529644288436e+12
10    2.678 095652210775e+14    2.678482366677917e+14
11    1.47 3006292412061e+16    1.472028569123609e+16
12    8. 064470412197736e+17    8.204455627276710e+17
13     8.709625000721411e+19    4.626538727415452e+19
14     7.586159929154813e+19    2.634780213053048e+21
15     1.258244856873877e+21    1.513195520592060e+23
16     7.176402631381276e+21    8.754155558087140e+24
17     2.296941780939168e+22    5.096849221028753e+26
18     2.581414396929671e+24    2.984205687512471e+28
19     1.068510051436661e+24    1.755988370979872e+30
```

Um hier Maple mit MATLAB (Datei *matr_booth1.m*) zu vergleichen, muss man natürlich seine numerische Rechnung verwenden.
Wir betrachten die Durchführung des VGA für die Ermittlung der näherungsweisen Inversen und die Berechnung der Determinante $\det(A)$ bei wachsender Dimension n. Dabei tendieren die Pivotelemente gegen Null.
Zu Grunde liegen Rechnungen in MATLAB (*double* Präzision) und Maple.

Generell wird sich bei numerischen Rechnungen die Anzahl der genauen Stellen im Ergebnis proportional zur wachsenden Dimension n verringern.
Es werden Vergleiche mit der exakten Auswertung in Maple gemacht, wo $\det(A) = \det(A^{-1})$ und $\|A\|_\infty = \|A^{-1}\|_\infty$ gelten.
Die Konditionszahl $\mathrm{cond}_\infty(A)$ kann aus der folgenden Tabelle ermittelt werden.

Programm	n	7	10	13	16	19
Maple	$\det(A)$	0.999999968110124	0.999451798364661	13.6249626527649	-2.135E+09	-1.239E+28
Digits=15	$\det(A^{-1})$	1.00000000613210	0.999823553678780	-0.390214525895609	6.575E-09	-6.640E-28
	$\|A\|_\infty$	78079.0000000000	32978945.0000000	14698053631.0000	6751535300609.00	3162019821780 99E+01
	$\|A^{-1}\|_\infty$	78079.0025978169	32996831.4933737	322091121.449972	2297564474.04069	814569492.655672
Maple	$\det(A)$	1.000000004937126	1.000143016118538	0.5062992104024677	40691101.32298157	1.150E+22
Digits=16	$\det(A^{-1})$	1.000000002234593	1.000126951225985	15.68557892921648	0.000000573313984	9.078E-20
	$\|A\|_\infty$	78079.00000000000	32978945.00000000	14698053631.00000	6751535300609.000	3162019821780991.
	$\|A^{-1}\|_\infty$	78078.99960363582	32974169.90492817	27596266698.34065	5334644304.537208	1757980846.178551
Maple	$\det(A)$	0.99999999998589404	0.99998628433584601	1.0352910937271376	23937.890228639031	-4.055E+17
Digits=17	$\det(A^{-1})$	0.99999999968326915	1.0000021360240729	0.8903001619462903	-0.001682451098551	-3.322E-18
	$\|A\|_\infty$	78079.000000000000	32978945.000000000	14698053631.000000	6751535300609.0000	3162019821780991.0
	$\|A^{-1}\|_\infty$	78079.000001153833	32979402.652158409	14227925264.680704	19901806328.728246	27121245287.204463
Maple	$\det(A)$	1	1	1	1	1
exakt	$\|A\|_\infty$	78079	32978945	14698053631	6751535300609	3162019821780991
MATLAB	$\det(A)$	1	1	1.25863956768224	-9374031.375107423	-2.009E+23
double	$\det(A^{-1})$ (a)	0.9999999996020	0.99998266829921	-2.09883035095195	-0.000000875884472	-1.139E-23
	(b)	0.99999999945585	1.00001764482417	-0.14571991408237	0.000000473532539	-6.603E-24
	$\|A\|_\infty$	78079	32978945	14698053631.00000	6751535300609.000	3162019821780991
	$\|A^{-1}\|_\infty$ (a)	78078.99992812875	32978898.22200207	11521467855.98685	5869992038.349004	572977390.3903831
	(b)	78078.99992812873	32978898.22200207	11521467855.98685	5869992038.349005	572977390.3903831

Tab. 3.5 Berechnungen für die Boothroyd/Dekker-Matrix mit MATLAB und Maple, wobei die Inverse numerisch berechnet wird und bei MATLAB 2 Varianten der Ermittlung der Inversen genommen werden
(a) `inv(A)` (ab $n \geq 13$ mit Warnungen),
(b) `A\I`

Die Determinantenberechnung $\det(A^{-1})$ hat verschiedene Resultate mit den MATLAB-Varianten (a) und (b).
MATLAB liefert in den meisten unserer Fälle etwas bessere Ergebnisse als Maple bei `Digits=16`. Es kann nicht ganz konkurrieren mit Maple bei `Digits=17`. Mit Maple bei `Digits=15` werden die Rechnungen schon ungenauer und ab $n = 13$ unakzeptabel, genauso auch die anderen Programme ab $n = 16$. Grund ist die extrem schlechte Matrixkondition.
Bezüglich der Variantenwahl `inv(A)` oder `A\I` bei der Matrixinversen in MATLAB, die hier in der Normberechnung enthalten ist, gibt es keinen endgültigen Aufschluss über Vorteile/Nachteile in Abhängigkeit von der Dimensionen n.

Kapitel 4

Maple und MATLAB für spezielle Matrixeigenschaften

Wir betrachten die eingeführten Testmatrizen $A(n,n) = (a_{ij})$ aus dem Kapitel 3.
Um die Abhängigkeit der Ergebnisse von der Matrixdimension n zu untersuchen, werden wir uns auf die Matrizen 4, 5 und 6 beschränken.
Zu diesen Matrizen sind die exakten inversen Matrizen $A^{-1} = (a'_{ij})$ verfügbar, und natürlich mit den CAS auch die Näherungsinversen $\widetilde{A}^{-1} = (\tilde{a}'_{ij})$. Dabei betrachten wir in MATLAB wiederum 2 bzw. 3 Varianten der Berechnung der Näherungsinversen gemäß `inv(A)` und `A\I` sowie im besonderen Fall `invhilb(n)`.
Wir interessieren uns insbesondere für die numerische Auswertung der folgenden Größen.

$$
\begin{aligned}
&|a'_{kl}| = \max_{i,j} |a'_{ij}|, \quad \max_{i,j} a'_{ij}, \\
&|\tilde{a}'_{kl}| = \max_{i,j} |\tilde{a}'_{ij}|, \quad \max_{i,j} \tilde{a}'_{ij}, \quad \tilde{a}'_{ij} = \mathcal{O}(\,), \\
&AA^{-1}, \quad I - AA^{-1}, \quad \|I - AA^{-1}\|_F, \\
&A\widetilde{A}^{-1}, \quad I - A\widetilde{A}^{-1}, \quad \|I - A\widetilde{A}^{-1}\|_F, \\
&|\tilde{a}'_{i_{max},j_{max}} - a'_{i_{max},j_{max}}| = \max_{i,j} |\tilde{a}'_{ij} - a'_{ij}|.
\end{aligned}
$$

Der Ablauf in den Programmen mit MATLAB und Maple ist zu den drei Matrizen ziemlich ähnlich. Deshalb wird nur für die erste Tridiagonalmatrix der Programmkode angegeben. Kleine Unterschiede werden durch die Nutzung von CAS-spezifischen Kommandos erzeugt, wie z.B. den für Matrizen nützlichen MATLAB-Befehl `max` oder den in MATLAB möglichen Befehl `invhilb(n)` zur exakten Berechnung der inversen Hilbert-Matrix für kleine Matrixdimensionen.
Die Rechnungen enthalten zu Beginn noch Möglichkeiten der Matrixgenerierung und sind oft etwas umfangreicher. In den Tabellen kommt nur ein Teil der Ergebnisse zur Auswertung.

4.1 Matrix 4

Spd Tridiagonalmatrix $A = A(n,n) = \text{tridiag}(-1,2,-1)$ mit der symmetrischen inversen Matrix $A^{-1} = (a'_{ij})$.
Ihr positiven Elemente sind

$$a'_{ij} = \frac{i\,(n+1-j)}{n+1}, \quad a'_{ji} = a'_{ij}, \quad 1 \le i \le j \le n.$$

Berechnungen in MATLAB (Datei *matr_trid2.m*)

```
% MATLAB
% matr_trid2.m
% Matrix, Eigenschaften, Inverse, Genauigkeit
%
diary matr_trid2.txt
diary off
echo off
% echo on
clear
clc
% clf

format long e
format compact

% Tridiagonalmatrix
disp('Generierung der spd Tridiagonalmatrix')
n=11            % 21,31,...
A=zeros(n);
for i=1:n
 for j=1:n
  if (i==j) A(i,j)=2;
   elseif (abs(i-j)==1) A(i,j)=-1;
   else  A(i,j)=0;
  end;
 end;
end;
A;
%
A=3*eye(n)-triu(tril(ones(n),1),-1);
pause

% Eigenschaften
disp('Eigenschaften')
II=eye(n);
A=3*eye(n)-triu(tril(ones(n),1),-1);
rank(A)
det_A=det(A)
norminf_A=norm(A,inf)           % Zeilensummennorm
norm1_A=norm(A,1)               % SSN = ZSN
norm2_A=norm(A)                 % Spektralnorm, norm(A,2)
cond2_A=cond(A)                 % Spektralkondition

% Exakte inverse Matrix
% Elemente der symmetrischen Inversen:
```

```
% a'(i,j)=i(n+1-j)/(n+1), 1<=i<=j<=n,   a'(i,j)>0
invA=zeros(n);
for i=1:n
 for j=i:n
  h=i*(n+1-j)/(n+1);
  invA(i,j)=h;
  invA(j,i)=h;
 end;
end;
invA;
det_invA=det(invA)

% Naeherungsweise inverse Matrix
% 2 Varianten
% (a)
invAs1=inv(A);
det_invAs1=det(invAs1)
norminf_invAs1=norm(invAs1,inf)  % Zeilensummennorm
norm1_invAs1=norm(invAs1,1)      % SSN = ZSN
norm2_invAs1=norm(invAs1)        % Spektralnorm, norm(invAs1,2)
cond2_invAs1=cond(invAs1)        % Spektralkondition
% (b)
invAs2=A\II;
det_invAs2=det(invAs2)
norminf_invAs2=norm(invAs2,inf)  % Zeilensummennorm
norm1_invAs2=norm(invAs2,1)      % SSN =ZSN
norm2_invAs2=norm(invAs2)        % Spektralnorm, norm(invAs2,2)
cond2_invAs2=cond(invAs2)        % Spektralkondition
pause

% Test
disp('Test der Genauigkeit der Naehrungsinversen')
II=eye(n);
A;
cond2_A=cond(A);                 % Spektralkondition

% Exakte inverse Matrix
IT=A*invA;
'||II-A*invA||_F=',norm(II-IT,'fro')

% Bei Naeherungsinversen kann fuer grosse Dimensionen n
% die Symmetrie verletzt sein.
ITs1=A*invAs1;
'||II-A*invAs1||_F=',norm(II-ITs1,'fro')

ITs2=A*invAs2;
'||II-A*invAs2||_F=',norm(II-ITs2,'fro')
pause

% In A^(-1) betragsgroesstes und groesstes Element mit Indizes
% Da die exakte Inverse theoretisch nur Elemente >0 enthaelt, gilt
% betragsgroesstes=groesstes Element.
bgr=0;
gr=-1e+100;
kb=1; lb=1;
k=1; l=1;
for i=1:n
```

```
 for j=1:n
  hh=invA(i,j);
  ahh=abs(hh);
  if (ahh>bgr) bgr=ahh; kb=i; lb=j; end;
  if (hh>gr) gr=hh; k=i; l=j; end;
 end;
end;

Bgr_ex=bgr
Kb_ex=kb
Lb_ex=lb
Gr_ex=gr
K_ex=k
L_ex=l

% Kurzversionen
[Y,II]=max(abs(invA));
[abw,JJ]=max(Y);
Bgr_ex=abw
IJmax=[II(JJ),JJ];
Kb_ex=IJmax(1)
Lb_ex=IJmax(2)

[Y,II]=max(invA);
[abw,JJ]=max(Y);
Gr_ex=abw
IJmax=[II(JJ),JJ];
K_ex=IJmax(1)
L_ex=IJmax(2)
pause

% In Naeherungsinverse betragsgroesstes und groesstes Element mit Indizes
% 2 Varianten
% (a)
bgr=0;
gr=-1e+100;
kb=1; lb=1;
k=1; l=1;
for i=1:n
 for j=1:n
  hh=invAs1(i,j);
  ahh=abs(hh);
  if (ahh>bgr) bgr=ahh; kb=i; lb=j; end;
  if (hh>gr) gr=hh; k=i; l=j; end;
 end;
end;
Bgr1=bgr
Kb1=kb
Lb1=lb
Gr1=gr
K1=k
L1=l
% (b)
bgr=0;
gr=-1e+100;
kb=1; lb=1;
k=1; l=1;
```

```
for i=1:n
 for j=1:n
  hh=invAs2(i,j);
  ahh=abs(hh);
  if (ahh>bgr) bgr=ahh; kb=i; lb=j; end;
  if (hh>gr) gr=hh; k=i; l=j; end;
 end;
end;
Bgr2=bgr
Kb2=kb
Lb2=lb
Gr2=gr
K2=k
L2=l

% Kurzversionen
[Y,II]=max(abs(invAs1));
[abw,JJ]=max(Y);
Bgr1=abw
IJmax=[II(JJ),JJ];
Kb1=IJmax(1)
Lb1=IJmax(2)
[Y,II]=max(invAs1);
[abw,JJ]=max(Y);
Gr1=abw
IJmax=[II(JJ),JJ];
K1=IJmax(1)
L1=IJmax(2)

[Y,II]=max(abs(invAs2));
[abw,JJ]=max(Y);
Bgr2=abw
IJmax=[II(JJ),JJ];
Kb2=IJmax(1)
Lb2=IJmax(2)
[Y,II]=max(invAs2);
[abw,JJ]=max(Y);
Gr2=abw
IJmax=[II(JJ),JJ];
K2=IJmax(1)
L2=IJmax(2)
pause

% Naeherungsinverse - Inverse, dazu Indizes
% 2 Varianten
% (a)
abw=0;
imax=1;
jmax=1;
for i=1:n
 for j=1:n
  hh=abs(invAs1(i,j)-invA(i,j));
  if (hh>abw) abw=hh; imax=i; jmax=j; end;
 end;
end;
Abw1=abw
Imax1=imax
```

```
Jmax1=jmax
% (b)
abw=0;
imax=1;
jmax=1;
for i=1:n
 for j=1:n
  hh=abs(invAs2(i,j)-invA(i,j));
  if (hh>abw) abw=hh; imax=i; jmax=j; end;
 end;
end;
Abw2=abw
Imax2=imax
Jmax2=jmax

% Kurzversionen
[Y,II]=max(abs(invAs1-invA));
[abw,JJ]=max(Y);
Abw1=abw
rel_Abw1=abw/max(max(invAs1))
IJmax=[II(JJ),JJ];
Imax1=IJmax(1)
Jmax1=IJmax(2)
%
[Y,II]=max(abs(invAs2-invA));
[abw,JJ]=max(Y);
Abw2=abw
rel_Abw2=abw/max(max(invAs2))
IJmax=[II(JJ),JJ];
Imax2=IJmax(1)
Jmax2=IJmax(2)
pause

format
diary off
echo off
```

Berechnungen in Maple (Datei *matr_trid2.mws*)
Nur wenige Ergebnisse werden notiert.

Bandmatrix, Tridiagonalmatrix, spd Matrix

```
> restart: with(linalg): with(LinearAlgebra):
> n:=5:
  i:='i': j:='j':
  A:=matrix(n,n,(i,j)->if i=j then 2         # A:=band([-1,2,-1],n);
            elif abs(j-i)=1 then -1 else 0 end if);
```

$$A := \begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

Eigenschaften der Tridiagonalmatrix
Exakte (symbolische) Rechnung

```
> `rank(A)`=rank(A);
  `det(A)`=det(A);
  invA:=inverse(A):
  `inv(A)`=evalm(invA);
  `det(invA)`=det(invA);

  # Normen
  `norm(A,2)`=norm(A,2);                          # Spektralnorm
  evalf(%);
  `norm(A,1)`=norm(A,1);                          # Spaltensummennorm
  `norm(A)=norm(A,infinity)`=norm(A);             # Zeilensummennorm = SSN
  `norm(A,frobenius)`=norm(A,frobenius);          # Frobenius-Norm
  evalf(%);
  `norm(invA)=norm(invA,infinity)`=norm(invA); # ZSN der Inversen
  evalf(%);

  # Konditionen dazu, auch Kommando cond(A,*)
  `cond(A,2)`=evalf(norm(A,2)*norm(inverse(A),2));
  `cond(A,1)`=evalf(norm(A,1)*norm(inverse(A),1));
  `cond(A)=cond(A,infinity)`=evalf(norm(A)*norm(inverse(A)));
  `cond(A,frobenius)`=evalf(norm(A,frobenius)*norm(inverse(A),frobenius));
```

$$rank(A) = 5$$

$$det(A) = 6$$

$$inv(A) = \begin{bmatrix} \frac{5}{6} & \frac{2}{3} & \frac{1}{2} & \frac{1}{3} & \frac{1}{6} \\ \frac{2}{3} & \frac{4}{3} & 1 & \frac{2}{3} & \frac{1}{3} \\ \frac{1}{2} & 1 & \frac{3}{2} & 1 & \frac{1}{2} \\ \frac{1}{3} & \frac{2}{3} & 1 & \frac{4}{3} & \frac{2}{3} \\ \frac{1}{6} & \frac{1}{3} & \frac{1}{2} & \frac{2}{3} & \frac{5}{6} \end{bmatrix}$$

$$norm(invA) = norm(invA, infinity) = \frac{9}{2}$$

$$det(invA) = \frac{1}{6}$$

$$norm(A, 2) = 3.732050807568877$$

$$norm(A, 1) = 4$$

$$norm(A) = norm(A, infinity) = 4$$

$$norm(A, frobenius) = 2\sqrt{7}$$

$$norm(A, frobenius) = 5.291502622129182$$

$$norm(invA) = norm(invA, infinity) = \frac{9}{2}$$

$$norm(invA) = norm(invA, infinity) = 4.500000000000000$$

$$cond(A, 2) = 13.92820323027551$$

$$cond(A, 1) = 18.$$

$$cond(A) = cond(A, infinity) = 18.$$

$$cond(A, frobenius) = 20.73912030706970$$

Exakte inverse Matrix
Elemente der symmetrischen Inversen:
$a'(i,j) = i(n+1-j)/(n+1),\ 1 <= i <= j <= n,\ \ a'(i,j) > 0$

```
> invA:=matrix(n,n):
  for i from 1 to n do
   for j from i to n do
    h:=i*(n+1-j)/(n+1);
    invA[i,j]:=h;
    invA[j,i]:=h;
   end do:
  end do:
  invA:=evalm(invA);

> # Symmetrie wird mit oberem Dreieck erzeugt
  invA:=array(symmetric,1..n,1..n,[]):
  for i from 1 to n do
   for j from i to n do
    invA[i,j]:=i*(n+1-j)/(n+1);
   end do:
  end do:
  invA:=evalm(invA);

> invA:=Matrix(n,(i,j)->i*(n+1-j)/(n+1),shape=symmetric);

> # falsch
  invA:=matrix(n,n,(i,j)->i*(n+1-j)/(n+1));
```

Test der Genauigkeit der Nährungsinversen

```
> Digits:=15:    # 16,17

> n:=11:         # 21,31,...
  II:=evalm(array(identity,1..n,1..n)):
  A:=evalf(BandMatrix([-1,2,-1],1,n)):
  cond(A,2);             # Spektralkondition
```

Exakte inverse Matrix mit Fehlernorm

```
> invA:=Matrix(n,(i,j)->i*(n+1-j)/(n+1),shape=symmetric):
  evalm(invA):
  IT:=evalm(A&*invA):
  norm(II-IT,frobenius);
```

In A^{-1} betragsgrößtes und größtes Element mit Indizes.
Da die exakte Inverse theoretisch nur Elemente > 0 enthält, gilt betragsgrößtes = größtes Element.

```
> bgr:=0:
  gr:=-1e+100:
  kb:=1: lb:=1:
  k:=1: l:=1:
  i:='i':j:='j':
```

```
for i from 1 to n do
 for j from 1 to n do
  hh:=invA[i,j];
  ahh:=abs(hh);
  if ahh>bgr then bgr:=ahh; kb:=i; lb:=j; end if;
  if hh>gr then gr:=hh; k:=i; l:=j; end if;
 end do:
end do:

‘Bgr_ex‘=bgr;
‘Kb_ex‘=kb;
‘Lb_ex‘=lb;
‘Gr_ex‘=gr;
‘K_ex‘=k;
‘L_ex‘=l;
```

Bei Näherungsinversen kann für grosse Dimensionen n die Symmetrie verletzt sein.
2 Varianten
(a)

```
> invAs1:=inverse(A):
  ITs1:=evalm(A&*invAs1):
  norm(II-ITs1,frobenius);
```

In Näherungsinverse betragsgrößtes und größtes Element mit Indizes

```
> bgr:=0:
  gr:=-1e+100:
  kb:=1: lb:=1:
  k:=1: l:=1:
  i:='i':j:='j':
  for i from 1 to n do
   for j from 1 to n do
    hh:=invAs1[i,j];
    ahh:=abs(hh);
    if ahh>bgr then bgr:=ahh; kb:=i; lb:=j; end if;
    if hh>gr then gr:=hh; k:=i; l:=j; end if;
   end do:
  end do:

  ‘Bgr1‘=bgr;
  ‘Kb1‘=kb;
  ‘Lb1‘=lb;
  ‘Gr1‘=gr;
  ‘K1‘=k;
  ‘L1‘=l;
```

Näherungsinverse - Inverse

```
> abw:=0:
  imax:=1:
  jmax:=1:
  i:='i':j:='j':
  for i from 1 to n do
   for j from 1 to n do
```

```
    hh:=abs(invAs1[i,j]-invA[i,j]);
    if hh>abw then abw:=hh; imax:=i; jmax:=j; end if;
   end do:
  end do:
  ‘Abw1‘=abw;
  ‘Imax1‘=imax;
  ‘Jmax1‘=jmax;
```

(b)

```
> invAs2:=linsolve(A,II):
  ITs2:=evalm(A&*invAs2):
  norm(II-ITs2,frobenius);
```

In Näherungsinverse betragsgrößtes und größtes Element mit Indizes

```
> bgr:=0:
  gr:=-1e+100:
  kb:=1: lb:=1:
  k:=1: l:=1:
  i:='i':j:='j':
  for i from 1 to n do
   for j from 1 to n do
    hh:=invAs2[i,j];
    ahh:=abs(hh);
    if ahh>bgr then bgr:=ahh; kb:=i; lb:=j; end if;
    if hh>gr then gr:=hh; k:=i; l:=j; end if;
   end do:
  end do:

  ‘Bgr2‘=bgr;
  ‘Kb2‘=kb;
  ‘Lb2‘=lb;
  ‘Gr2‘=gr;
  ‘K2‘=k;
  ‘L2‘=l;
```

Näherungsinverse - Inverse

```
> abw:=0:
  imax:=1:
  jmax:=1:
  i:='i':j:='j':
  for i from 1 to n do
   for j from 1 to n do
    hh:=abs(invAs2[i,j]-invA[i,j]);
    if hh>abw then abw:=hh; imax:=i; jmax:=j; end if;
   end do:
  end do:
  ‘Abw2‘=abw;
  ‘Imax2‘=imax;
  ‘Jmax2‘=jmax;
```

Analoge Rechnungen für andere Matrixdimensionen und mit `Digits=16,17`.

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\approx \frac{4}{\pi^2}(n+1)^2$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k \approx \frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$I - A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I - AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max\limits_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max}, j_{max})
11	5.770 10^1 (a) (b)	10^0 $\tilde{a}'_{66}=3$ $\tilde{a}'_{66}=3.0$ $a'_{66}=3$	4.016 10^{-15} 3.444 10^{-15} 1.683 10^{-15}	8.9 10^{-16} (5,5) 2.7 10^{-15} (8,8)
21	1.955 10^1 (a) (b)	10^1 $\tilde{a}'_{11,11}=5.5$ $\tilde{a}'_{11,11}=5.5$ $a'_{11,11}=5.5$	1.816 10^{-14} 1.078 10^{-14} 7.444 10^{-15}	2.7 10^{-15} (13,14) 1.3 10^{-14} (9,9)
31	4.143 10^2 (a) (b)	10^1 $\tilde{a}'_{16,16}=8.0$ $\tilde{a}'_{16,16}=8.0$ $a'_{16,16}=8.0$	3.765 10^{-14} 2.311 10^{-14} 0	2.5 10^{-14} (20,19) 2.8 10^{-14} (15,15)
41	7.143 10^2 (a) (b)	10^1 $\tilde{a}'_{21,21}=10.5$ $\tilde{a}'_{21,21}=10.5$ $a'_{21,21}=10.5$	8.359 10^{-14} 4.049 10^{-14} 2.419 10^{-14}	6.0 10^{-14} (20,20) 5.0 10^{-14} (22,23)
51	1.095 10^3 (a) (b)	10^1 $\tilde{a}'_{26,26}=13.0$ $\tilde{a}'_{26,26}=13.0$ $a'_{26,26}=13$	1.518 10^{-13} 5.658 10^{-14} 3.456 10^{-14}	9.1 10^{-14} (21,21) 5.9 10^{-14} (16,19)
61	1.557 10^3 (a) (b)	10^1 $\tilde{a}'_{31,31}=15.5$ $\tilde{a}'_{31,31}=15.5$ $a'_{31,31}=15.5$	2.171 10^{-13} 7.863 10^{-14} 5.084 10^{-14}	1.3 10^{-13} (29,27) 1.1 10^{-13} (43,44)
71	2.100 10^3 (a) (b)	10^1 $\tilde{a}'_{36,36}=18.0$ $\tilde{a}'_{36,36}=18.0$ $a'_{36,36}=18$	2.666 10^{-13} 1.116 10^{-13} 7.016 10^{-14}	1.5 10^{-13} (29,26) 2.2 10^{-13} (44,44)
81	2.724 10^3 (a) (b)	10^1 $\tilde{a}'_{41,41}=20.5$ $\tilde{a}'_{41,41}=20.5$ $a'_{41,41}=20.5$	4.494 10^{-13} 1.471 10^{-13} 9.170 10^{-14}	1.8 10^{-13} (30,31) 3.3 10^{-13} (44,44)
91	3.430 10^3 (a) (b)	10^1 $\tilde{a}'_{46,46}=23.0$ $\tilde{a}'_{46,46}=23.0$ $a'_{46,46}=23$	6.376 10^{-13} 1.847 10^{-13} 1.160 10^{-13}	2.0 10^{-13} (30,29) 4.2 10^{-13} (44,44)
101	4.216 10^3 (a) (b)	10^1 $\tilde{a}'_{51,51}=25.5$ $\tilde{a}'_{51,51}=25.5$ $a'_{51,51}=25.5$	7.847 10^{-13} 2.233 10^{-13} 1.496 10^{-13}	2.2 10^{-13} (29,28) 5.2 10^{-13} (45,46)

n	Spektral-kondition $\text{cond}_2(A) \approx \frac{4}{\pi^2}(n+1)^2$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k \approx \frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$I - A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I - AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max}, j_{max})
151	9.363 10^3 (a) (b)	10^1 $\tilde{a}'_{76,76}=38.0$ $\tilde{a}'_{76,76}=38.0$ $a'_{76,76}=38$	2.121 10^{-12} 4.942 10^{-13} 3.260 10^{-13}	4.9 10^{-13} (89,83) 9.7 10^{-13} (94,94)
201	1.654 10^4 (a) (b)	10^2 $\tilde{a}'_{101,101}=50.5$ $\tilde{a}'_{101,101}=50.5$ $a'_{101,101}=50.5$	4.508 10^{-12} 8.857 10^{-13} 5.662 10^{-13}	7.7 10^{-13} (138,141) 1.8 10^{-12} (95,96)
251	2.574 10^4 (a) (b)	10^2 $\tilde{a}'_{126,126}=63.0$ $\tilde{a}'_{126,126}=63.0$ $a'_{126,126}=63$	6.589 10^{-12} 1.346 10^{-12} 8.352 10^{-13}	2.1 10^{-12} (147,148) 2.2 10^{-12} (95,94)
301	3.696 10^4 (a) (b)	10^2 $\tilde{a}'_{151,151}=75.5$ $\tilde{a}'_{151,151}=75.5$ $a'_{151,151}=75.5$	1.058 10^{-11} 1.983 10^{-12} 1.142 10^{-12}	3.8 10^{-12} (154,152) 2.7 10^{-12} (94,94)
351	5.022 10^4 (a) (b)	10^2 $\tilde{a}'_{176,176}=88.0$ $\tilde{a}'_{176,176}=88.0$ $a'_{176,176}=88$	1.814 10^{-11} 2.752 10^{-12} 1.792 10^{-12}	5.3 10^{-12} (154,154) 4.7 10^{-12} (259,259)
401	6.549 10^4 (a) (b)	10^2 $\tilde{a}'_{201,201}=100.5$ $\tilde{a}'_{201,201}=100.5$ $a'_{201,201}=100.5$	2.723 10^{-11} 3.563 10^{-12} 2.261 10^{-12}	6.8 10^{-12} (193,191) 9.9 10^{-12} (263,265)
451	8.280 10^4 (a) (b)	10^2 $\tilde{a}'_{226,226}=113.0$ $\tilde{a}'_{226,226}=113.0$ $a'_{226,226}=113.0$	3.222 10^{-11} 4.418 10^{-12} 2.748 10^{-12}	8.3 10^{-12} (189,193) 1.6 10^{-11} (264,265)
501	1.021 10^5 (a) (b)	10^2 $\tilde{a}'_{251,251}=125.5$ $\tilde{a}'_{251,251}=125.5$ $a'_{251,251}=125.5$	4.336 10^{-11} 5.333 10^{-12} 3.291 10^{-12}	9.8 10^{-12} (194,192) 2.3 10^{-11} (265,267)
601	1.469 10^5 (a) (b)	10^2 $\tilde{a}'_{301,301}=150.5$ $\tilde{a}'_{301,301}=150.5$ $a'_{301,301}=150.5$	6.477 10^{-11} 7.721 10^{-12} 4.922 10^{-12}	1.2 10^{-11} (491,489) 3.4 10^{-11} (289,291)
701	1.997 10^5 (a) (b)	10^2 $\tilde{a}'_{351,351}=175.5$ $\tilde{a}'_{351,351}=175.5$ $a'_{351,351}=175.5$	1.101 10^{-10} 1.052 10^{-11} 6.916 10^{-12}	3.6 10^{-11} (494,493) 3.9 10^{-11} (289,290)

n	Spektral-kondition $\text{cond}_2(A) \approx \frac{4}{\pi^2}(n+1)^2$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k \approx \frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max}, j_{max})
801	2.607 10^5 (a) (b)	10^2 $\tilde{a}'_{401,401}=200.5$ $\tilde{a}'_{401,401}=200.5$ $a'_{401,401}=200.5$	1.445 10^{-10} 1.369 10^{-11} 8.929 10^{-12}	6.4 10^{-11} (497,495) 4.0 10^{-11} (529,529)
901	3.297 10^5 (a) (b)	10^2 $\tilde{a}'_{451,451}=225.5$ $\tilde{a}'_{451,451}=225.5$ $a'_{451,451}=225.5$	1.716 10^{-10} 1.707 10^{-11} 1.104 10^{-11}	8.6 10^{-11} (493,496) 7.0 10^{-11} (551,548)
1001	4.069 10^5 (a) (b)	10^2 $\tilde{a}'_{501,501}=250.5$ $\tilde{a}'_{501,501}=250.5$ $a'_{501,501}=250.5$	2.396 10^{-10} 2.074 10^{-11} 1.314 10^{-11}	1.0 10^{-10} (493,496) 1.1 10^{-10} (551,554)
1501	9.143 10^5 (a) (b)	10^2 $\tilde{a}'_{751,751}=375.5$ $\tilde{a}'_{751,751}=375.5$ $a'_{751,751}=375.5$	6.881 10^{-10} 4.754 10^{-11} 3.165 10^{-11}	1.2 10^{-10} (497,495) 5.7 10^{-10} (816,815)
2001	1.624 10^6 (a) (b)	10^3 $\tilde{a}'_{1001,1001}=500.5$ $\tilde{a}'_{1001,1001}=500.5$ $a'_{1001,1001}=500.5$	1.467 10^{-9} 8.326 10^{-11} 5.235 10^{-11}	2.6 10^{-10} (944,942) 1.0 10^{-9} (893,895)

Tab. 4.1 Berechnungen für die Tridiagonalmatrix mit MATLAB (*double* Präzision)
`A=3*eye(n)-triu(tril(ones(n),1),-1)`
2 Varianten der Ermittlung der Näherungsinversen $\widetilde{A}^{-1}$
(a) `inv(A)`
(b) `A\I`

Bei MATLAB treten in den numerischen Rechnungen für $\tilde{a}'_{ij}$ Ungenauigkeiten in den letzten Mantissenstellen auf, damit auch die Abweichungen bei den angegebenen Normen. Die Anzahl dieser Stellen nimmt mit der Dimension n zu und entspricht ungefähr den Werten $t-1$ bzw. t des Exponenten in der GP-Darstellung der Spektralkondition $\text{cond}_2(A) = m\,10^t$.

Vergleicht man die Ergebnisse mit Maple, stellt man ähnliche Eigenschaften fest.
Wegen der deutlich längeren Rechenzeiten kann man nicht allzu große Dimensionen testen.
MATLAB liegt bez. der Qualität der Ergebnisse in der Nähe von Maple bei `Digits=16`.
Mit Maple-Variante `linsolve(A,I)` fallen die Kontrollgrößen $\max_{i,j}|\tilde{a}'_{ij}-a'_{ij}|$ und $\max_{i,j}\tilde{a}'_{ij}$ der numerischen Rechnung bei allen `Digits=15,16,17` sehr günstig aus.
Im Vergleich der hier verwendeten Berechnungsvarianten für die Näherungsinverse hat das Maple-Kommando `inverse(A)` "gewisse kleine numerische Schwachstellen". Zumindest kann es mit den anderen Befehlen für die Testmatrizen meistens nicht ganz konkurrieren.

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\approx \frac{4}{\pi^2}(n+1)^2$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
11	$5.770\ 10^1$ (a) 10^0 (b)	15 $\tilde{a}'_{66}=3.0$ $\tilde{a}'_{66}=3$ $a'_{66}=3$	$6.115\ 10^{-14}$ $5.162\ 10^{-14}$ $1.342\ 10^{-14}$	$2.0\ 10^{-14}$ (7,8) 0
	(a) (b)	16 $\tilde{a}'_{66}=3.0$ $\tilde{a}'_{66}=3$ $a'_{66}=3$	$5.620\ 10^{-15}$ $5.162\ 10^{-15}$ $1.342\ 10^{-15}$	$1.0\ 10^{-15}$ (3,2) 0
	(a) (b)	17 $\tilde{a}'_{66}=3.0$ $\tilde{a}'_{66}=3$ $a'_{66}=3$	$5.943\ 10^{-16}$ $5.162\ 10^{-16}$ $1.342\ 10^{-16}$	$3.0\ 10^{-16}$ (5,6) 0
21	$1.955\ 10^2$ (a) 10^1 (b)	15 $\tilde{a}'_{11,11}=5.5$ $\tilde{a}'_{11,11}=5.5$ $a'_{11,11}=\frac{11}{2}$	$1.988\ 10^{-13}$ $1.399\ 10^{-13}$ $1.106\ 10^{-13}$	$3.0\ 10^{-13}$ (12,12) 0
	(a) (b)	16 $\tilde{a}'_{11,11}=5.5$ $\tilde{a}'_{11,11}=5.5$ $a'_{11,11}=\frac{11}{2}$	$1.814\ 10^{-14}$ $1.399\ 10^{-14}$ $1.106\ 10^{-14}$	$1.3\ 10^{-14}$ (13,13) 0
	(a) (b)	17 $\tilde{a}'_{11,11}=5.5$ $\tilde{a}'_{11,11}=5.5$ $a'_{11,11}=\frac{11}{2}$	$1.790\ 10^{-15}$ $1.399\ 10^{-15}$ $1.106\ 10^{-15}$	$2.1\ 10^{-15}$ (14,15) 0
31	$4.143\ 10^2$ (a) 10^1 (b)	15 $\tilde{a}'_{16,16}=8.0$ $\tilde{a}'_{16,16}=8$ $a'_{16,16}=8$	$5.209\ 10^{-13}$ 0. 0.	$7.8\ 10^{-13}$ (13,13) 0
	(a) (b)	16 $\tilde{a}'_{16,16}=8.0$ $\tilde{a}'_{16,16}=8$ $a'_{16,16}=8$	$5.407\ 10^{-14}$ $5.407\ 10^{-14}$ 0.	$3.3\ 10^{-14}$ (14,14) 0
	(a) (b)	17 $\tilde{a}'_{16,16}=8.0$ $\tilde{a}'_{16,16}=8$ $a'_{16,16}=8$	$5.342\ 10^{-15}$ 0. 0.	$5.9\ 10^{-15}$ (15,15) 0
41	$7.143\ 10^2$ (a) 10^1 (b)	15 $\tilde{a}'_{21,21}=10.5$ $\tilde{a}'_{21,21}=10.5$ $a'_{21,21}=\frac{21}{2}$	$8.548\ 10^{-13}$ $6.581\ 10^{-13}$ $6.389\ 10^{-13}$	$1.4\ 10^{-12}$ (19,19) 0
	(a) (b)	16 $\tilde{a}'_{21,21}=10.5$ $\tilde{a}'_{21,21}=10.5$ $a'_{21,21}=\frac{21}{2}$	$9.444\ 10^{-14}$ $7.198\ 10^{-14}$ $6.631\ 10^{-14}$	$6.2\ 10^{-14}$ (19,18) 0
	(a) (b)	17 $\tilde{a}'_{21,21}=10.5$ $\tilde{a}'_{21,21}=10.5$ $a'_{21,21}=\frac{21}{2}$	$8.622\ 10^{-15}$ $6.999\ 10^{-15}$ $6.703\ 10^{-15}$	$8.3\ 10^{-15}$ (15,15) 0

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\approx \frac{4}{\pi^2}(n+1)^2$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$		$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes Digits	$\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max\limits_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
51	$1.095\ 10^3$	(a)	15	$\tilde{a}'_{26,26}=13.0$	$1.411\ 10^{-12}$	$2.3\ 10^{-12}$ (25,25)
	10^1	(b)		$\tilde{a}'_{26,26}=13$	$1.378\ 10^{-12}$	0
				$a'_{26,26}=13$	$1.093\ 10^{-12}$	
		(a)	16	$\tilde{a}'_{26,26}=13.0$	$1.479\ 10^{-13}$	$9.0\ 10^{-14}$ (36,37)
		(b)		$\tilde{a}'_{26,26}=13$	$1.384\ 10^{-13}$	0
				$a'_{26,26}=13$	$1.085\ 10^{-13}$	
		(a)	17	$\tilde{a}'_{26,26}=13.0$	$1.408\ 10^{-14}$	$1.0\ 10^{-14}$ (15,15)
		(b)		$\tilde{a}'_{26,26}=13$	$1.351\ 10^{-14}$	0
				$a'_{26,26}=13$	$1.081\ 10^{-14}$	
61	$1.557\ 10^3$	(a)	15	$\tilde{a}'_{31,31}=15.5$	$1.985\ 10^{-12}$	$3.4\ 10^{-12}$ (28,24)
	10^1	(b)		$\tilde{a}'_{31,31}=15.5$	$2.070\ 10^{-12}$	0
				$a'_{31,31}=\frac{31}{2}$	$1.642\ 10^{-12}$	
		(a)	16	$\tilde{a}'_{31,31}=15.5$	$2.047\ 10^{-13}$	$2.0\ 10^{-13}$ (37,35)
		(b)		$\tilde{a}'_{31,31}=15.5$	$2.036\ 10^{-13}$	0
				$a'_{31,31}=\frac{31}{2}$	$1.570\ 10^{-13}$	
		(a)	17	$\tilde{a}'_{31,31}=15.5$	$1.996\ 10^{-14}$	$1.9\ 10^{-14}$ (43,43)
		(b)		$\tilde{a}'_{31,31}=15.5$	$2.021\ 10^{-14}$	0
				$a'_{31,31}=\frac{31}{2}$	$1.570\ 10^{-14}$	
71	$2.100\ 10^3$	(a)	15	$\tilde{a}'_{36,36}=18.0$	$2.655\ 10^{-12}$	$4.9\ 10^{-12}$ (43,43)
	10^1	(b)		$\tilde{a}'_{36,36}=18$	$2.580\ 10^{-12}$	0
				$a'_{36,36}=18$	$1.653\ 10^{-12}$	
		(a)	16	$\tilde{a}'_{36,36}=18.0$	$2.746\ 10^{-13}$	$3.1\ 10^{-13}$ (39,39)
		(b)		$\tilde{a}'_{36,36}=18$	$2.580\ 10^{-13}$	0
				$a'_{36,36}=18$	$1.653\ 10^{-13}$	
		(a)	17	$\tilde{a}'_{36,36}=18.0$	$2.644\ 10^{-14}$	$3.7\ 10^{-14}$ (43,45)
		(b)		$\tilde{a}'_{36,36}=18$	$2.580\ 10^{-14}$	0
				$a'_{36,36}=18$	$1.653\ 10^{-14}$	
81	$2.724\ 10^3$	(a)	15	$\tilde{a}'_{41,41}=20.5$	$3.308\ 10^{-12}$	$7.2\ 10^{-12}$ (42,44)
	10^1	(b)		$\tilde{a}'_{41,41}=20.5$	$3.343\ 10^{-12}$	0
				$a'_{41,41}=\frac{41}{2}$	$2.502\ 10^{-12}$	
		(a)	16	$\tilde{a}'_{41,41}=20.5$	$3.335\ 10^{-13}$	$3.9\ 10^{-13}$ (39,41)
		(b)		$\tilde{a}'_{41,41}=20.5$	$3.271\ 10^{-13}$	0
				$a'_{41,41}=\frac{41}{2}$	$2.511\ 10^{-13}$	
		(a)	17	$\tilde{a}'_{41,41}=20.5$	$3.435\ 10^{-14}$	$6.1\ 10^{-14}$ (45,46)
		(b)		$\tilde{a}'_{41,41}=20.5$	$3.333\ 10^{-14}$	0
				$a'_{41,41}=\frac{41}{2}$	$2.426\ 10^{-14}$	

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\approx \frac{4}{\pi^2}(n+1)^2$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
91	$3.430\ 10^3$ (a) 10^1 (b)	15 $\tilde{a}'_{46,46}=23.0$ $\tilde{a}'_{46,46}=23$ $a'_{46,46}=23$	$4.043\ 10^{-12}$ $4.017\ 10^{-12}$ $2.916\ 10^{-12}$	$9.0\ 10^{-12}$ (43,47) 0
	(a) (b)	16 $\tilde{a}'_{46,46}=23.0$ $\tilde{a}'_{46,46}=23$ $a'_{46,46}=23$	$3.984\ 10^{-13}$ $3.968\ 10^{-13}$ $2.916\ 10^{-13}$	$5.0\ 10^{-13}$ (39,41) 0
	(a) (b)	17 $\tilde{a}'_{46,46}=23.0$ $\tilde{a}'_{46,46}=23$ $a'_{46,46}=23$	$4.054\ 10^{-14}$ $3.951\ 10^{-14}$ $2.899\ 10^{-14}$	$7.9\ 10^{-14}$ (48,50) 0
101	$4.216\ 10^3$ (a) 10^1 (b)	15 $\tilde{a}'_{51,51}=25.5$ $\tilde{a}'_{51,51}=25.5$ $a'_{51,51}=\frac{51}{2}$	$4.757\ 10^{-12}$ $4.661\ 10^{-12}$ $3.398\ 10^{-12}$	$1.1\ 10^{-11}$ (43,44) 0
	(a) (b)	16 $\tilde{a}'_{51,51}=25.5$ $\tilde{a}'_{51,51}=25.5$ $a'_{51,51}=\frac{51}{2}$	$4.677\ 10^{-13}$ $4.675\ 10^{-13}$ $3.411\ 10^{-13}$	$6.6\ 10^{-13}$ (40,41) 0
	(a) (b)	17 $\tilde{a}'_{51,51}=25.5$ $\tilde{a}'_{51,51}=25.5$ $a'_{51,51}=\frac{51}{2}$	$4.859\ 10^{-14}$ $4.632\ 10^{-14}$ $3.436\ 10^{-14}$	$9.7\ 10^{-14}$ (48,53) 0
151	$9.363\ 10^3$ (a) 10^1 (b)	15 $\tilde{a}'_{76,76}=38.0$ $\tilde{a}'_{76,76}=38$ $a'_{76,76}=38$	$8.977\ 10^{-12}$ $7.977\ 10^{-12}$ $5.615\ 10^{-12}$	$2.2\ 10^{-11}$ (61,60) 0
	(a) (b)	16 $\tilde{a}'_{76,76}=38.0$ $\tilde{a}'_{76,76}=38$ $a'_{76,76}=38$	$9.083\ 10^{-13}$ $8.002\ 10^{-13}$ $5.606\ 10^{-13}$	$2.1\ 10^{-12}$ (72,72) 0
	(a) (b)	17 $\tilde{a}'_{76,76}=38.0$ $\tilde{a}'_{76,76}=38$ $a'_{76,76}=38$	$9.382\ 10^{-14}$ $7.946\ 10^{-14}$ $5.628\ 10^{-14}$	$1.6\ 10^{-13}$ (106,108) 0
201	$1.654\ 10^4$ (a) 10^2 (b)	15 $\tilde{a}'_{101,101}=50.5$ $\tilde{a}'_{101,101}=50.5$ $a'_{101,101}=\frac{101}{2}$	$1.461\ 10^{-11}$ $1.143\ 10^{-11}$ $8.378\ 10^{-12}$	$3.4\ 10^{-11}$ (88,89) 0
	(a) (b)	16 $\tilde{a}'_{101,101}=50.5$ $\tilde{a}'_{101,101}=50.5$ $a'_{101,101}=\frac{101}{2}$	$1.532\ 10^{-12}$ $1.150\ 10^{-12}$ $8.011\ 10^{-13}$	$3.8\ 10^{-12}$ (90,90) 0
	(a) (b)	17 $\tilde{a}'_{101,101}=50.5$ $\tilde{a}'_{101,101}=50.5$ $a'_{101,101}=\frac{101}{2}$	$1.508\ 10^{-13}$ $1.143\ 10^{-13}$ $8.378\ 10^{-14}$	$6.7\ 10^{-13}$ (114,115) 0

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\approx \frac{4}{\pi^2}(n+1)^2$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
301	3.696 10^4 (a) 10^2 (b)	15 $\tilde{a}'_{151,151}=75.5$ $\tilde{a}'_{151,151}=75.5$ $a'_{151,151}=\frac{151}{2}$	4.417 10^{-11} 3.691 10^{-11} 3.518 10^{-11}	1.5 10^{-10} (189,188) 0
	(a) (b)	16 $\tilde{a}'_{151,151}=75.5$ $\tilde{a}'_{151,151}=75.5$ $a'_{151,151}=\frac{151}{2}$	4.601 10^{-12} 3.679 10^{-12} 3.505 10^{-12}	1.1 10^{-11} (210,212) 0
	(a) (b)	17 $\tilde{a}'_{151,151}=75.5$ $\tilde{a}'_{151,151}=75.5$ $a'_{151,151}=\frac{151}{2}$	4.423 10^{-13} 3.671 10^{-13} 3.495 10^{-13}	1.7 10^{-12} (132,131) 0
401	6.549 10^4 (a) 10^2 (b)	15 $\tilde{a}'_{201,201}=100.5$ $\tilde{a}'_{201,201}=100.5$ $a'_{201,201}=\frac{201}{2}$	7.865 10^{-11} 6.403 10^{-11} 6.245 10^{-11}	3.5 10^{-10} (211,211) 0
	(a) (b)	16 $\tilde{a}'_{201,201}=100.5$ $\tilde{a}'_{201,201}=100.5$ $a'_{201,201}=\frac{201}{2}$	8.177 10^{-12} 6.421 10^{-12} 6.268 10^{-12}	2.4 10^{-11} (212,213) 0
	(a) (b)	17 $\tilde{a}'_{201,201}=100.5$ $\tilde{a}'_{201,201}=100.5$ $a'_{201,201}=\frac{201}{2}$	7.721 10^{-13} 6.404 10^{-13} 6.252 10^{-13}	2.7 10^{-12} (144,145) 0
501	1.021 10^5 (a) 10^2 (b)	15 $\tilde{a}'_{251,251}=125.5$ $\tilde{a}'_{251,251}=125.5$ $a'_{251,251}=\frac{251}{2}$	1.288 10^{-10} 1.228 10^{-10} 1.043 10^{-10}	4.5 10^{-10} (210,212) 0
	(a) (b)	16 $\tilde{a}'_{251,251}=125.5$ $\tilde{a}'_{251,251}=125.5$ $a'_{251,251}=\frac{251}{2}$	1.316 10^{-11} 1.232 10^{-11} 1.034 10^{-11}	2.9 10^{-11} (212,213) 0
	(a) (b)	17 $\tilde{a}'_{251,251}=125.5$ $\tilde{a}'_{251,251}=125.5$ $a'_{251,251}=\frac{251}{2}$	1.246 10^{-12} 1.154 10^{-12} 1.028 10^{-12}	8.0 10^{-12} (336,338) 0

Tab. 4.2 Berechnungen für die Tridiagonalmatrix mit Maple, `Digits=15,16,17`
`A:=band([-1,2,-1],n)`
2 Varianten der Ermittlung der Näherungsinversen $\widetilde{A}^{-1}$
(a) `inverse(A)`
(b) `linsolve(A,I)`

4.2 Matrix 5

Hilbert-Matrix $A = A(n,n) = (a_{ij})$ mit

$$a_{ij} = \frac{1}{i+j-1}.$$

Sie ist symmetrisch, positiv definit und ihre Inverse $A^{-1} = (a'_{ij})$ hat die ganzzahligen Elemente

$$a'_{ij} = \frac{(-1)^{i+j}}{i+j-1}\gamma_i\gamma_j, \quad \gamma_i = \frac{(n+i-1)!}{(i-1)!^2(n-i)!}, \quad i,j = 1,2,...,n.$$

Das größte Element a'_{ij} ist zugleich betragsgrößtes und steht auf der Diagonalen.

n	Spektral-kondition $\text{cond}_2(A)$	$A^{-1} = (a'_{ij})$, $\widetilde{A}^{-1} = (\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k = n, n-1, n-2, ...$ betragsgrößtes & größtes $\tilde{a}'_{ij} = \mathcal{O}(\ )$	$\tilde{a}'_{ij}$ / a'_{ij}	$I - A\widetilde{A}^{-1}$ / $\|I-A\widetilde{A}^{-1}\|_F$ / $I - AA^{-1}$ / $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max}, j_{max})
2	$1.928\ 10^1$ (a)	10^1	$\tilde{a}'_{22} = 12.00000000000000$	0	$3.553\ 10^{-15}$ (2,2)
	(b)		$\tilde{a}'_{22} = 12.00000000000000$	$6.280\ 10^{-16}$	$1.776\ 10^{-15}$ (2,2)
	(c)		$\tilde{a}'_{22} = 12$	0	0
			$a'_{22} = 12$	0	
3	$5.241\ 10^2$ (a)	10^2	$1.920000000000007\ 10^2$	$1.787\ 10^{-14}$	$6.821\ 10^{-13}$ (2,2)
	(b)		$1.920000000000007\ 10^2$	$1.013\ 10^{-14}$	$7.105\ 10^{-13}$ (2,2)
	(c)		$1.92\ 10^2$	0	0
			$1.92\ 10^2$	0	
4	$1.551\ 10^4$ (a)	10^3	$6.479999999999403\ 10^3$	$6.332\ 10^{-13}$	$5.966\ 10^{-10}$ (3,3)
	(b)		$6.479999999999434\ 10^3$	$3.061\ 10^{-13}$	$5.657\ 10^{-10}$ (3,3)
	(c)		$6.48\ 10^3$	$5.684\ 10^{-14}$	0
			$6.48\ 10^3$	$5.684\ 10^{-14}$	
5	$4.766\ 10^5$ (a)	10^5	$1.791999999999916\ 10^5$	$1.539\ 10^{-11}$	$1.438\ 10^{-8}$ (4,3)
	(b)		$1.791999999994423\ 10^5$	$6.531\ 10^{-12}$	$5.577\ 10^{-7}$ (4,4)
	(c)		$1.792\ 10^5$	$3.638\ 10^{-12}$	0
			$1.792\ 10^5$	$3.638\ 10^{-12}$	
6	$1.495\ 10^7$ (a)	10^6	$4.410000000447804\ 10^6$	$2.536\ 10^{-9}$	$4.478\ 10^{-4}$ (5,5)
	(b)		$4.410000000176569\ 10^6$	$2.121\ 10^{-10}$	$1.766\ 10^{-4}$ (5,5)
	(c)		$4.41\ 10^6$	$1.455\ 10^{-11}$	$9.095\ 10^{-13}$ (1,5)
			$4.41\ 10^6$	$1.455\ 10^{-11}$	

n	Spektral-kondition $\mathrm{cond}_2(A)$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k=n,n-1,n-2,...$ betragsgrößtes & größtes $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$\tilde{a}'_{ij}$ / a'_{ij}	$I-A\widetilde{A}^{-1}$ / $\|I-A\widetilde{A}^{-1}\|_F$ / $I-AA^{-1}$ / $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
7	$4.754\ 10^8$ (a)	10^8	$1.334025004360048\ 10^8$	$1.019\ 10^{-7}$	$4.360\ 10^{-1}$ (5,5)
	(b)		$1.334025006743926\ 10^8$	$5.341\ 10^{-9}$	$6.744\ 10^{-1}$ (5,5)
	(c)		$1.334025\ 10^8$	$5.209\ 10^{-10}$	$7.451\ 10^{-9}$ (4,6)
			$1.334025\ 10^8$	$3.883\ 10^{-9}$	
8	$1.526\ 10^{10}$ (a)	10^9	$4.249941660559195\ 10^9$	$3.165\ 10^{-6}$	$3.544\ 10^1$ (6,6)
	(b)		$4.249942391421277\ 10^9$	$3.350\ 10^{-7}$	$6.954\ 10^2$ (6,6)
	(c)		$4.249941696\ 10^9$	$6.667\ 10^{-8}$	$4.768\ 10^{-7}$ (6,5)
			$4.249941696\ 10^9$	$1.738\ 10^{-7}$	
9	$4.932\ 10^{11}$ (a)	10^{11}	$1.223670500662108\ 10^{11}$	$7.788\ 10^{-5}$	$3.951\ 10^5$ (7,7)
	(b)		$1.223678174141008\ 10^{11}$	$1.263\ 10^{-5}$	$3.722\ 10^5$ (7,7)
	(c)		$1.223674452\ 10^{11}$	$2.731\ 10^{-7}$	$1.907\ 10^{-6}$ (9,6)
			$1.223674452\ 10^{11}$	$5.870\ 10^{-7}$	
10	$1.603\ 10^{13}$ (a)	10^{12}	$3.480307538227295\ 10^{12}$	$2.631\ 10^{-3}$	$3.657\ 10^8$ (7,7)
	(b)		$3.479990271992993\ 10^{12}$	$2.259\ 10^{-4}$	$1.093\ 10^8$ (7,7)
	(c)		$3.4806739968\ 10^{12}$	$2.865\ 10^{-5}$	$4.883\ 10^{-4}$ (8,7)
			$3.4806739968\ 10^{12}$	$1.545\ 10^{-4}$	
11	$5.225\ 10^{14}$ (a)	10^{14}	$1.173253413258971\ 10^{14}$	$6.990\ 10^{-2}$	$3.177\ 10^{11}$ (8,8)
	(b)		$1.176028150914772\ 10^{14}$	$7.982\ 10^{-3}$	$4.020\ 10^{10}$ (8,8)
	(c)		$1.1764301193216\ 10^{14}$	$1.632\ 10^{-3}$	$4.883\ 10^{-4}$ (6,5)
			$1.1764301193216\ 10^{14}$	$1.631\ 10^{-3}$	
12	$1.735\ 10^{16}$ (a)	10^{15}	$3.377682208218081\ 10^{15}$	$1.177\ 10^1$	$2.818\ 10^{14}$ (9,9)
	(b)		$3.617578335565005\ 10^{15}$	$2.329\ 10^{-1}$	$4.187\ 10^{13}$ (9,9)
	(c)		$3.65944915908\ 10^{15}$	$5.955\ 10^{-2}$	$5.000\ 10^{-1}$ (10,9)
			$3.65944915908\ 10^{15}$	$1.289\ 10^{-1}$	

Tab. 4.3 Berechnungen für die Hilbert-Matrix mit MATLAB (*double* Präzision)
`A=hilb(n)`, 3 Varianten der Ermittlung der Näherungsinversen $\widetilde{A}^{-1}$
(a) `inv(A)`
(b) `A\I`
(c) `invhilb(n)`

Die MATLAB-Inversen-Variante `A\I` ist mit einer Ausnahme bei $n=2$ für alle Dimensionen bezüglich der Fehlergröße $\|I-A\widetilde{A}^{-1}\|_F$ günstiger. Das überträgt sich aber nicht zwingend auf die Abweichung $\max_{i,j}|\tilde{a}'_{ij}-a'_{ij}|$. Diese kann im Vergleich mit der Variante `inv(A)` besser oder schlechter sein.

Bechten wir noch folgendes Kuriosum.
Das Kommando `invhilb(n)` erzeugt die exakte Hilbert-Matrix bis zur Dimension von ca $n = 15$ (Information aus MATLAB-Online-Hilfe).
Die eigentlich richtige Berechnungsformel der ganzzahligen Elemente der Inversen $A^{-1} = (a'_{ij})$ beginnt bei $n = 6$ wegen der großen Werte von Fakultäten und der numerischen Rechnung mit den Quotienten aus Fakultäten, in der letzten Mantissenstelle kleine Fehler zu erzeugen. Das wird erstens sichtbar daran, dass der Wert $\|I - A\widehat{A}^{-1}\|_F$ mit $\widehat{A}^{-1} = (\hat{a}'_{ij})$=`invhilb(n)`, nachdem er anfangs noch mit dem Wert $\|I - AA^{-1}\|_F$ übereinstimmt, dann aber bei wachsendem $n \geq 6$ meist kleiner ist als dieser.
Zweitens wird für $\widehat{A}^{-1} = (\hat{a}'_{ij})$ eine nicht verschwindende Größe $\max_{i,j} |\hat{a}'_{ij} - a'_{ij}|$ durch die fehlerhafte numerische Berechnung der sehr groß werdenden Elemente a'_{ij} von A^{-1} verursacht, nicht durch die Elemente $\hat{a}'_{ij}$ von $\widehat{A}^{-1}$.
Die Auswirkungen dieser beginnenden Ungenauigkeit sind jedoch noch nicht so gravierend. Es gilt weiterhin für die beiden anderen Näherungsinversen $\max_{i,j} |\tilde{a}'_{ij} - a'_{ij}| = \max_{i,j} |\tilde{a}'_{ij} - \hat{a}'_{ij}|$.
Wir betrachten für $n = 6, 7, ..., 12, ..., 16$ die Abweichungen zwischen den Matrizen A^{-1} und $\widehat{A}^{-1}$. Die Beträge der Elemente der Differenzmatrix sind größenordnungsmäßig angegeben. Für $n = 6$ sind zusätzlich die beiden Matrizen A^{-1} und $\widehat{A}^{-1}$ als Ergebnis und Ausgabe in einem MATLAB-File notiert. Es ist eine Stelle (1,5) in der Matrix mit der Abweichung $\approx 10^{-12}$ erkennbar. Für $n > 6$ ist die Anzahl solcher Stellen größer.

```
invA =
 Columns 1 through 3
   3.600000000000000e+001   -6.300000000000000e+002    3.360000000000000e+003
  -6.300000000000000e+002    1.470000000000000e+004   -8.820000000000000e+004
   3.360000000000000e+003   -8.820000000000000e+004    5.644800000000000e+005
  -7.560000000000000e+003    2.116800000000000e+005   -1.411200000000000e+006
   7.560000000000000e+003   -2.205000000000000e+005    1.512000000000000e+006
  -2.772000000000000e+003    8.316000000000000e+004   -5.821200000000000e+005
 Columns 4 through 6
  -7.560000000000000e+003    7.560000000000001e+003   -2.772000000000000e+003
   2.116800000000000e+005   -2.205000000000000e+005    8.316000000000000e+004
  -1.411200000000000e+006    1.512000000000000e+006   -5.821200000000000e+005
   3.628800000000000e+006   -3.969000000000000e+006    1.552320000000000e+006
  -3.969000000000000e+006    4.410000000000000e+006   -1.746360000000000e+006
   1.552320000000000e+006   -1.746360000000000e+006    6.985440000000000e+005

ans =
         36        -630        3360       -7560        7560       -2772
       -630       14700       -88200      211680     -220500       83160
       3360       -88200       564480    -1411200     1512000     -582120
      -7560       211680     -1411200     3628800    -3969000     1552320
       7560      -220500      1512000    -3969000     4410000    -1746360
      -2772        83160      -582120     1552320    -1746360      698544
```

$$|A^{-1} - \widehat{A}^{-1}|(6,6) = \begin{pmatrix} 0 & 0 & 0 & 0 & 9\ 10^{-13} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$|A^{-1}-\widehat{A}^{-1}|(7,7)=\begin{pmatrix} 0 & 0 & 2\ 10^{-12} & 0 & 7\ 10^{-12} & 7\ 10^{-12} & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 7\ 10^{-9} & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 9\ 10^{-10} & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$|A^{-1}-\widehat{A}^{-1}|(8,8)=\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 9\ 10^{-10} & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 5\ 10^{-7} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$|A^{-1}-\widehat{A}^{-1}|(9,9)=\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 2\ 10^{-10} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 1\ 10^{-8} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 2\ 10^{-6} & 0 & 0 & 0 \end{pmatrix},$$

$$|A^{-1}-\widehat{A}^{-1}|(10,10)=\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 9\ 10^{-10} & 2\ 10^{-9} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 6\ 10^{-8} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 5\ 10^{-4} & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 2\ 10^{-7} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$|A^{-1}-\widehat{A}^{-1}|(11,11)=\begin{pmatrix} 0 & 0 & 0 & 0 & 9\ 10^{-10} & 0 & 0 & 0 & 7\ 10^{-9} & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 5\ 10^{-4} & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5\ 10^{-4} & 0 \end{pmatrix},$$

$$|A^{-1}-\widehat{A}^{-1}|(12,12)=\begin{pmatrix} 0 & 0 & 0 & 0 & 4\ 10^{-9} & 0 & 0 & 0 & 0 & 3\ 10^{-8} & 0 & 2\ 10^{-9}\\ 0 & 0 & 0 & 6\ 10^{-8} & 0 & 0 & 0 & 0 & 8\ 10^{-6} & 0 & 0 & 2\ 10^{-7}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2\ 10^{-5}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\ 10^{-4}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\ 10^{-3}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6\ 10^{-2} & 0 & 4\ 10^{-3}\\ 0 & 0 & 0 & 2\ 10^{-3} & 0 & 0 & 0 & 0 & 3\ 10^{-1} & 0 & 0 & 2\ 10^{-2}\\ 0 & 0 & 0 & 0 & 0 & 0 & 3\ 10^{-1} & 0 & 0 & 0 & 0 & 3\ 10^{-2}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3\ 10^{-2}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5\ 10^{-1} & 0 & 0 & 3\ 10^{-2}\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1\ 10^{-1} & 0 & 1\ 10^{-1} & 0 & 8\ 10^{-3}\\ 0 & 5\ 10^{-7} & 0 & 1\ 10^{-4} & 1\ 10^{-3} & 4\ 10^{-3} & 0 & 0 & 3\ 10^{-2} & 0 & 2\ 10^{-2} & 4\ 10^{-3} \end{pmatrix}.$$

$n=13:\quad \max_{ij}|A^{-1}-\widehat{A}^{-1}|=32, \qquad n=14:\quad \max_{ij}|A^{-1}-\widehat{A}^{-1}|=1024$

$n=15:\quad \max_{ij}|A^{-1}-\widehat{A}^{-1}|=32768, \qquad n=16:\quad \max_{ij}|A^{-1}-\widehat{A}^{-1}|=524288$

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk},\ k=n,n-1,n-2,...$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
2	$1.928\ 10^1$ (a) 10^1 (b)	15 $\tilde{a}'_{22}=12.0000000000000$ $\tilde{a}'_{22}=12.0000000000000$ $a'_{22}=12$	$2.236\ 10^{-14}$ $2.236\ 10^{-14}$ 0	$2.0\ 10^{-14}$ (1,2) $2.0\ 10^{-14}$ (1,2)
	(a) (b)	16 $\tilde{a}'_{22}=12.00000000000000$ $\tilde{a}'_{22}=12.00000000000000$ $a'_{22}=12$	$2.236\ 10^{-15}$ $2.236\ 10^{-15}$ 0	$2.0\ 10^{-15}$ (1,2) $2.0\ 10^{-15}$ (1,2)
	(a) (b)	17 $\tilde{a}'_{22}=12.000000000000000$ $\tilde{a}'_{22}=12.000000000000000$ $a'_{22}=12$	$2.236\ 10^{-16}$ $2.236\ 10^{-16}$ 0	$2.0\ 10^{-16}$ (1,2) $2.0\ 10^{-16}$ (1,2)
3	$5.241\ 10^2$ (a) 10^2 (b)	15 $1.92000000000043\ 10^2$ $1.92000000000017\ 10^2$ $1.92\ 10^2$	$9.489\ 10^{-13}$ $4.823\ 10^{-13}$ $2.002\ 10^{-13}$	$4.3\ 10^{-11}$ (2,2) $1.7\ 10^{-11}$ (2,2)
	(a) (b)	16 $1.920000000000042\ 10^2$ $1.920000000000017\ 10^2$ $1.92\ 10^2$	$9.216\ 10^{-14}$ $4.823\ 10^{-14}$ $2.002\ 10^{-14}$	$4.2\ 10^{-12}$ (2,2) $1.7\ 10^{-12}$ (2,2)
	(a) (b)	17 $1.9200000000000040\ 10^2$ $1.9200000000000017\ 10^2$ $1.92\ 10^2$	$1.270\ 10^{-14}$ $4.823\ 10^{-15}$ $2.002\ 10^{-15}$	$4.0\ 10^{-13}$ (2,2) $1.7\ 10^{-13}$ (2,2)
4	$1.551\ 10^4$ (a) 10^3 (b)	15 $6.48000000013342\ 10^3$ $6.48000000001909\ 10^3$ $6.48\ 10^3$	$1.462\ 10^{-10}$ $6.933\ 10^{-12}$ $3.005\ 10^{-12}$	$1.3\ 10^{-7}$ (3,3) $1.9\ 10^{-8}$ (3,3)
	(a) (b)	16 $6.479999999965381\ 10^3$ $6.480000000001405\ 10^3$ $6.48\ 10^3$	$3.202\ 10^{-11}$ $5.925\ 10^{-13}$ $3.165\ 10^{-13}$	$3.5\ 10^{-8}$ (3,3) $1.4\ 10^{-9}$ (3,3)
	(a) (b)	17 $6.4799999999991155\ 10^3$ $6.4800000000001657\ 10^3$ $6.48\ 10^3$	$1.777\ 10^{-12}$ $1.418\ 10^{-13}$ $2.834\ 10^{-14}$	$8.8\ 10^{-10}$ (3,3) $1.7\ 10^{-10}$ (3,3)
5	$4.766\ 10^5$ (a) 10^5 (b)	15 $1.79200000014538\ 10^5$ $1.79200000009294\ 10^5$ $1.792\ 10^5$	$2.294\ 10^{-10}$ $2.377\ 10^{-10}$ $4.473\ 10^{-11}$	$1.5\ 10^{-5}$ (4,4) $9.3\ 10^{-6}$ (4,4)
	(a) (b)	16 $1.791999999999424\ 10^5$ $1.791999999994192\ 10^5$ $1.792\ 10^5$	$2.015\ 10^{-11}$ $2.177\ 10^{-11}$ $1.755\ 10^{-11}$	$5.8\ 10^{-8}$ (4,4) $5.8\ 10^{-7}$ (4,4)
	(a) (b)	17 $1.7920000000007003\ 10^5$ $1.7920000000001743\ 10^5$ $1.792\ 10^5$	$2.202\ 10^{-12}$ $2.069\ 10^{-12}$ $1.123\ 10^{-12}$	$7.0\ 10^{-8}$ (4,4) $1.7\ 10^{-8}$ (4,4)

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk},\ k=n,n-1,n-2,...$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
6	$1.495\ 10^7$ (a) 10^6 (b)	15 $4.41000000543852\ 10^6$ $4.41000000414500\ 10^6$ $4.41\ 10^6$	$3.211\ 10^{-9}$ $4.951\ 10^{-9}$ $3.271\ 10^{-9}$	$5.4\ 10^{-3}$ (5,5) $4.1\ 10^{-3}$ (5,5)
	(a) (b)	16 $4.409999999696157\ 10^6$ $4.409999999696157\ 10^6$ $4.41\ 10^6$	$2.759\ 10^{-10}$ $2.759\ 10^{-10}$ $4.220\ 10^{-10}$	$3.0\ 10^{-4}$ (5,5) $4.4\ 10^{-4}$ (5,5)
	(a) (b)	17 $4.4100000000163323\ 10^6$ $4.4099999999987944\ 10^6$ $4.41\ 10^6$	$3.381\ 10^{-11}$ $5.012\ 10^{-11}$ $3.218\ 10^{-11}$	$1.6\ 10^{-5}$ (5,5) $1.2\ 10^{-6}$ (5,5)
7	$4.754\ 10^8$ (a) 10^8 (b)	15 $1.33402502707438\ 10^8$ $1.33402503505121\ 10^8$ $1.334025\ 10^8$	$2.351\ 10^{-7}$ $2.235\ 10^{-7}$ $3.292\ 10^{-8}$	$2.7\ 10^{0}$ (5,5) $3.5\ 10^{0}$ (5,5)
	(a) (b)	16 $1.334025002310484\ 10^8$ $1.334025000023068\ 10^8$ $1.334025\ 10^8$	$1.799\ 10^{-8}$ $2.481\ 10^{-8}$ $1.147\ 10^{-8}$	$2.3\ 10^{-1}$ (5,5) $6.2\ 10^{-3}$ (6,6)
	(a) (b)	17 $1.3340250004395298\ 10^8$ $1.3340250001949108\ 10^8$ $1.334025\ 10^8$	$2.310\ 10^{-9}$ $1.962\ 10^{-9}$ $4.318\ 10^{-10}$	$4.4\ 10^{-2}$ (5,5) $1.9\ 10^{-2}$ (5,5)
8	$1.526\ 10^{10}$ (a) 10^9 (b)	15 $4.24994113467916\ 10^9$ $4.24994321990471\ 10^9$ $4.249941696\ 10^9$	$3.965\ 10^{-6}$ $3.963\ 10^{-6}$ $2.360\ 10^{-6}$	$5.6\ 10^{2}$ (6,6) $1.5\ 10^{3}$ (6,6)
	(a) (b)	16 $4.249941691625708\ 10^9$ $4.249941739956877\ 10^9$ $4.249941696\ 10^9$	$3.537\ 10^{-7}$ $2.839\ 10^{-7}$ $2.015\ 10^{-7}$	$4.4\ 10^{0}$ (6,6) $4.4\ 10^{1}$ (6,6)
	(a) (b)	17 $4.2499417234344538\ 10^9$ $4.2499417087123128\ 10^9$ $4.249941696\ 10^9$	$4.234\ 10^{-8}$ $3.793\ 10^{-8}$ $1.789\ 10^{-8}$	$2.7\ 10^{1}$ (6,6) $1.3\ 10^{1}$ (6,6)
9	$4.932\ 10^{11}$ (a) 10^{11} (b)	15 $1.22364051141666\ 10^{11}$ $1.22366977400762\ 10^{11}$ $1.223674452\ 10^{11}$	$1.967\ 10^{-4}$ $2.317\ 10^{-4}$ $5.432\ 10^{-5}$	$3.4\ 10^{6}$ (7,7) $4.7\ 10^{5}$ (7,7)
	(a) (b)	16 $1.223674699766789\ 10^{11}$ $1.223674571141203\ 10^{11}$ $1.223674452\ 10^{11}$	$1.463\ 10^{-5}$ $1.380\ 10^{-5}$ $1.170\ 10^{-5}$	$2.5\ 10^{4}$ (7,7) $1.2\ 10^{4}$ (7,7)
	(a) (b)	17 $1.2236743339251794\ 10^{11}$ $1.2236745060287829\ 10^{11}$ $1.223674452\ 10^{11}$	$1.099\ 10^{-6}$ $3.065\ 10^{-6}$ $5.228\ 10^{-7}$	$1.2\ 10^{4}$ (7,7) $5.4\ 10^{3}$ (7,7)

n	Spektral-kondition $\text{cond}_2(A)$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$		$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{kk}$, $k=n,n-1,n-2,...$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
10	$1.603\ 10^{13}$	(a)	15 $3.47653487719802\ 10^{12}$	$5.354\ 10^{-3}$	$4.1\ 10^{9}$ (7,7)
	10^{12}	(b)	$3.47927125559676\ 10^{12}$	$4.785\ 10^{-3}$	$1.4\ 10^{9}$ (7,7)
			$3.4806739968\ 10^{12}$	$1.582\ 10^{-3}$	
		(a)	16 $3.480716443744591\ 10^{12}$	$4.775\ 10^{-4}$	$4.2\ 10^{7}$ (7,7)
		(b)	$3.480642393846225\ 10^{12}$	$5.822\ 10^{-4}$	$3.2\ 10^{7}$ (7,7)
			$3.4806739968\ 10^{12}$	$2.315\ 10^{-4}$	
		(a)	17 $3.4806365106147272\ 10^{12}$	$4.905\ 10^{-5}$	$3.7\ 10^{7}$ (7,7)
		(b)	$3.4806768553268085\ 10^{12}$	$6.171\ 10^{-5}$	$2.9\ 10^{6}$ (7,7)
			$3.4806739968\ 10^{12}$	$2.374\ 10^{-5}$	
11	$5.225\ 10^{14}$	(a)	15 $1.14577323837776\ 10^{14}$	$1.005\ 10^{-1}$	$3.1\ 10^{12}$ (8,8)
	10^{14}	(b)	$1.16564181793996\ 10^{14}$	$1.481\ 10^{-1}$	$1.1\ 10^{12}$ (8,8)
			$1.1764301193216\ 10^{14}$	$4.019\ 10^{-2}$	
		(a)	16 $1.176327278727867\ 10^{14}$	$8.812\ 10^{-3}$	$1.0\ 10^{10}$ (8,8)
		(b)	$1.175577806909240\ 10^{14}$	$1.773\ 10^{-2}$	$8.5\ 10^{10}$ (8,8)
			$1.1764301193216\ 10^{14}$	$4.383\ 10^{-3}$	
		(a)	17 $1.1759370428559068\ 10^{14}$	$1.174\ 10^{-3}$	$4.9\ 10^{10}$ (8,8)
		(b)	$1.1764536070926620\ 10^{14}$	$1.148\ 10^{-3}$	$2.3\ 10^{9}$ (8,8)
			$1.1764301193216\ 10^{14}$	$4.260\ 10^{-4}$	
12	$1.735\ 10^{16}$	(a)	15 $4.15301240808476\ 10^{15}$	$4.396\ 10^{1}$	$4.9\ 10^{14}$ (9,9)
	10^{15}	(b)	$4.40133786946088\ 10^{15}$	$4.784\ 10^{1}$	$7.4\ 10^{14}$ (9,9)
			$3.65944915908\ 10^{15}$	$3.631\ 10^{-1}$	
		(a)	16 $3.624788866216343\ 10^{15}$	$5.322\ 10^{-1}$	$3.5\ 10^{13}$ (9,9)
		(b)	$3.604291780815086\ 10^{15}$	$6.444\ 10^{-1}$	$5.5\ 10^{13}$ (9,9)
			$3.65944915908\ 10^{15}$	$1.771\ 10^{-1}$	
		(a)	17 $3.6204566466454548\ 10^{15}$	$5.000\ 10^{-2}$	$3.9\ 10^{13}$ (9,9)
		(b)	$3.6667277301253598\ 10^{15}$	$4.635\ 10^{-2}$	$7.3\ 10^{12}$ (9,9)
			$3.65944915908\ 10^{15}$	$1.473\ 10^{-2}$	

Tab. 4.4 Berechnungen für die Hilbert-Matrix mit Maple, `Digits=15,16,17`
`A:=hilbert(n)`
2 Varianten der Ermittlung der Näherungsinversen $\widetilde{A}^{-1}$
(a) `inverse(A)`
(b) `linsolve(A,I)`

Einige Bemerkungen zum Vergleich der Rechnungen.

MATLAB liegt bezüglich der Güte der Ergebnisse in der Nähe der numerischen Rechnung mit Maple bei `Digits=16`.

Die Maple-Inversen-Variante `linsolve(A,I)` ist für die meisten Dimensionen bezüg-lich der Fehlergröße $\|I - A\widetilde{A}^{-1}\|_F$ günstiger als `inverse(A)`. Das überträgt sich aber nicht zwingend auf die Abweichung $\max_{i,j} |\tilde{a}'_{ij} - a'_{ij}|$. Diese kann im Vergleich mit der von der Variante `inverse(A)` besser oder schlechter sein.

Erstaunlich ist, dass in beiden CAS trotz betragsgroßer Matrixelemente $\tilde{a}'_{ij}$, a'_{ij} und erheblicher Abweichungen zwischen $\widetilde{A}^{-1}$ und A^{-1}, das heißt $\max_{i,j} |\tilde{a}'_{ij} - a'_{ij}| \gg 1$, im Matrixprodukt $A\widetilde{A}^{-1}$ Fehler kompensiert werden und es nahe der Einheitmatrix liegt, was hier mit der Norm $\|I - A\widetilde{A}^{-1}\|_F \ll 1$ unterstrichen wird.

4.3 Matrix 6

Nicht symmetrische ganzzahlige Boothroyd/Dekker-Matrix

$$A = (a_{ij}), \quad a_{ij} = \binom{n+i-1}{i-1}\binom{n-1}{n-j}\frac{n}{i+j-1} = \frac{(n+i-1)!}{(n-j)!\,(i-1)!\,(j-1)!\,(i+j-1)}$$

mit $\det(A) = 1$ sowie der Inversen $A^{-1} = (a'_{ij})$ und ihren Elementen

$$a'_{ij} = (-1)^{i+j} a_{ij}, \quad i,j = 1,2,...,n.$$

Bemerkungen zum betragsgrößten und größten Element mit Indizes in A^{-1}.
Die exakte Inverse enthält Elemente abwechselnd > 0 und < 0. Das betragsgrößte Element steht immer in der letzten Zeile sowie dort in der Mitte, falls n ungerade ist, vor der Mitte, falls n gerade ist. Es ist postiv oder negativ je nach Teilbarkeit von n bzw. $n+1$ durch 2 oder 4. Falls das betragsgrößte Element der inversen Matrix negativ ist, steht das größte - es ist dann positiv - links oder rechts daneben.

n	Spektral- kondition $\mathrm{cond}_2(A)$	$A^{-1} = (a'_{ij}), \quad \widetilde{A}^{-1} = (\tilde{a}'_{ij})$ $\tilde{a}'_{nk}, \ k \approx \frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}$=$\mathcal{O}(\ )$ $\quad \tilde{a}'_{ij}$ a'_{ij}=$\mathcal{O}(\ )$ $\quad a'_{ij}$	$I - A\widetilde{A}^{-1}$ $\|I - A\widetilde{A}^{-1}\|_F$ $I - AA^{-1}$ $\|I - AA^{-1}\|_F$	$\max\limits_{i,j}\|\tilde{a}'_{ij} - a'_{ij}\|$ Stelle (i_{max}, j_{max})
7	1.825 10^9 (a)	10^4 $\tilde{a}'_{7,4}$ = -2.402399997803248 10^4 $\tilde{a}'_{7,3}$ = 2.001999998130650 10^4	3.430 10^{-8}	2.2 10^{-5} (7,4)
	(b)	$\tilde{a}'_{7,4}$ = -2.402399997803248 10^4 $\tilde{a}'_{7,3}$ = 2.001999998130650 10^4	7.092 10^{-9}	2.2 10^{-5} (7,4)
		10^4 $a'_{7,4}$ = -24024 $a'_{7,3}$ = 20020	0	
10	2.678 10^{14} (a)	10^7 $\tilde{a}'_{10,5}$ = -8.314008208578656 10^6 $\tilde{a}'_{10,6}$ = 7.759740958680756 10^6	6.916 10^{-3}	1.2 10^1 (10,5)
	(b)	$\tilde{a}'_{10,5}$ = -8.314008208578656 10^6 $\tilde{a}'_{10,6}$ = 7.759740958680756 10^6	6.471 10^{-4}	1.2 10^1 (10,5)
		10^7 $a'_{10,5}$ = -8314020 $a'_{10,6}$ = 7759752	0	
11	1.472 10^{16} (a)	10^8 $\tilde{a}'_{11,6}$ = -6.102139250095709 10^7 $\tilde{a}'_{11,5}$ = 5.424006627350676 10^7	3.085 10^0	8.7 10^4 (11,6)
	(b)	$\tilde{a}'_{11,6}$ = -6.102139250095708 10^7 $\tilde{a}'_{11,5}$ = 5.424006627350676 10^7	4.531 10^{-2}	8.7 10^4 (11,6)
		10^8 $a'_{11,6}$ = -61108047 $a'_{11,5}$ = 54318264	0	

n	Spektral-kondition $\mathrm{cond}_2(A)$	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{nk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}=\mathcal{O}(\ )$ $\quad$ $\tilde{a}'_{ij}$ $a'_{ij}=\mathcal{O}(\ )$ $\quad$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max\limits_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
12	8.204 10^{17} (a)	10^8 $\tilde{a}'_{12,6}=$ 4.521989154202473 10^8 $\tilde{a}'_{12,6}=$ 4.521989154202473 10^8	2.546 10^2	1.1 10^7 (12,6)
	(b)	$\tilde{a}'_{12,6}=$ 4.521989154202473 10^8 $\tilde{a}'_{12,6}=$ 4.521989154202473 10^8	2.047 10^0	1.1 10^7 (12,6)
		10^8 $a'_{12,6}=$ 440936496 $a'_{12,6}=$ 440936496	2.404 10^0	
13	4.627 10^{19} (a)	10^9 $\tilde{a}'_{13,7}=$ 2.582618743887998 10^9 $\tilde{a}'_{13,7}=$ 2.582618743887998 10^9	1.032 10^4	7.1 10^8 (13,7)
	(b)	$\tilde{a}'_{13,7}=$ 2.582618743887998 10^9 $\tilde{a}'_{13,7}=$ 2.582618743887998 10^9	1.002 10^2	7.1 10^8 (13,7)
		10^9 $a'_{13,7}=$ 3287684400 $a'_{13,7}=$ 3287684400	4.485 10^1	
16	8.754 10^{24} (a)	10^9 $\tilde{a}'_{16,6}=$ 1.406404323527516 10^9 $\tilde{a}'_{16,6}=$ 1.406404323527516 10^9	1.128 10^5	1.3 10^{12} (16,8)
	(b)	$\tilde{a}'_{16,6}=$ 1.406404323527516 10^9 $\tilde{a}'_{16,6}=$ 1.406404323527516 10^9	4.831 10^2	1.3 10^{12} (16,8)
		10^{12} $a'_{16,8}=$ 1345374716400 $a'_{16,8}=$ 1345374716400	3.578 10^6	
19	1.756 10^{30} (a)	10^8 $\tilde{a}'_{19,\ 4}=$ 1.110921315852490 10^8 $\tilde{a}'_{19,\ 4}=$ 1.110921315852490 10^8	9.375 10^7	5.8 10^{14} (19,10)
	(b)	$\tilde{a}'_{19,\ 4}=$ 1.110921315852490 10^8 $\tilde{a}'_{19,\ 4}=$ 1.110921315852490 10^8	1.776 10^4	5.8 10^{14} (19,10)
		10^{15} $a'_{19,10}=$ -583057996306500 $a'_{19,\ 9}=$ 544187463219400	3.034 10^{11}	

Tab. 4.5 Berechnungen für die Boothroyd/Dekker-Matrix mit MATLAB (*double* Präzision)

```
A=(a(i,j)), a(i,j)=factorial(n+i-1)/(factorial(n-j)*
                    factorial(i-1)*factorial(j-1)*(i+j-1))
```

2 Varianten der Ermittlung der Näherungsinversen $\widetilde{A}^{-1}$

(a) `inv(A)`

(b) `A\I`

Die MATLAB-Inversen-Variante `A\I` ist für alle Dimensionen n bezüglich der Fehlergröße $\|I-A\widetilde{A}^{-1}\|_F$ günstiger. Das überträgt sich aber nicht auf die Abweichung $\max_{i,j}|\tilde{a}'_{ij}-a'_{ij}|$. Beide Varianten `inv(A)` und `A\I` zeigen diesbezüglich sowie bei den betragsgrößten bzw. größten Elemente keine Unterschiede.
Dabei übersieht man jedoch, dass die "Dominanz" dieser Elemente andere Abweichungen überspielen kann, denn die inversen Matrizen A_1=`inv(A)` und A_2=`A\I` sind natürlich i. Allg. nicht gleich. Ab $n = 13$ beginnen in beiden Inversen-Varianten deutliche Unterschiede (in Größenordnungen) zur exakten Inversen.

Wir berechnen für einige Dimensionen n den Unterschied $D_n = \max_{i,j}|\tilde{a}'_{1_{ij}}-\tilde{a}'_{2_{ij}}|$ zwischen den beiden Näherungsinversen A_1 und A_2 sowie die realisierenden Indizes.

n	D_n	(i,j)
4	1.865174681370263e-014	(1,3)
7	2.570033075244282e-011	(1,3)
10	9.793998856366670e-008	(1,5)
11	9.406556955582346e-007	(1,6)
12	1.592369113723180e-005	(1,8)
13	7.230216124298750e-005	(1,7)
16	1.581774070018582e-004	(1,6)
19	1.037275374340396e-004	(1,8)
22	1.440906242464735e-004	(1,5)

Die maximalen Abweichungen entstehen also fern von den betragsgrößten Elementen der inversen Matrizen.
Betrachten wir auch beispielhaft für die Dimension $n = 12$ die 6. Spalte und für die Dimension $n = 13$ die 7. Spalte der Inversen sowie der beiden Näherungsinversen. Dort stimmen zumindest noch die Größenordnung der Matrixelemente überein. Bei den nächsten Werten n ist das nicht so.

n=12

A^(-1)	inv(A)	A\I
-924	-9.424289586165644e+002	-9.424289548134138e+002
10296	1.051213286678806e+004	1.051213286884418e+004
-63063	-6.444000090316839e+004	-6.444000090196565e+004
280280	2.865949975834668e+005	2.865949975835494e+005
-1009008	-1.032331825904341e+006	-1.032331825904108e+006
3118752	3.192398659229155e+006	3.192398659229037e+006
-8576568	-8.782788173066258e+006	-8.782788173066394e+006
21488544	2.201329816511807e+007	2.201329816511814e+007
-49884120	-5.111879391577423e+007	-5.111879391577417e+007
108636528	1.113572475107842e+008	1.113572475107841e+008
-224062839	-2.297330598804583e+008	-2.297330598804583e+008
440936496	4.521989154202473e+008	4.521989154202473e+008

```
n=13

    A^(-1)              inv(A)                      A\I
-------------------------------------------------------------------
       1716     1.388467732310838e+003    1.388467804612999e+003
     -21021    -1.692405075335563e+004   -1.692405074531063e+004
     140140     1.123596341455393e+005    1.123596341368700e+005
    -672672    -5.374406940344826e+005   -5.374406940276994e+005
    2598960     2.070272567659576e+006    2.070272567655773e+006
   -8576568    -6.814273317412782e+006   -6.814273317412554e+006
   25069968     1.987383791186230e+007    1.987383791186184e+007
  -66512160    -5.262222003727139e+007   -5.262222003727179e+007
  162954792     1.286985407755439e+008    1.286985407755439e+008
 -373438065    -2.944732848269159e+008   -2.944732848269160e+008
  808383576     6.365543556965237e+008    6.365543556965236e+008
-1665760096    -1.310028835859093e+009   -1.310028835859093e+009
 3287684400     2.582618743887998e+009    2.582618743887998e+009
```

Dass bei $n = 13$ die kritische Situation beginnt, erkennt man am Verhalten der Determinanten. Theoretisch ist zwar $\det(A) = \det(A^{-1}) = 1$, aber in der numerischen Auswertung verursachen die wachsenden Fakultäten in den exakten Formeln für die Matrixelemente zunehmende Fehler.
So gilt

```
   n        det(A)          det(A^(-1))      det(inv(A))       det(A\I)
-------------------------------------------------------------------------
   7    1                 1                 0.999999999602    0.999999999456
  10    1                 1                 0.999982668299    1.000017644824
  11    1                 1                 0.999071365173    1.001547713746
  12    0.975756666110    0.975756666110    0.904307627535    0.994311302268
  13    1.258639567682    1.258639567682   -2.098830350952   -0.145719914082
  14  -67.258897094742  -67.258897094742   -0.021306833023   -0.062351987274
```

Bei numerischer Rechnung werden mit wachsender Dimension n sowohl A und A^{-1} als auch beide Varianten der Näherungsinversen gleichermaßen schnell "unbrauchbar".
Mit den großen Einträgen in den Matrizen A und A^{-1} wird das Matrizenprodukt AA^{-1} zunehmend "ungenauer" und damit auch die Kontrollgröße $\|I - AA^{-1}\|_F$.

Nun zu Rechnungen in Maple und Vergleichen mit MATLAB.

Für die anfänglichen Dimensionen $n = 7, 10, 11, 12, 13$ ist die Inversenberechnung mittels (b) `linsolve(A,I)` sehr genau. Sie ist so genau, dass sie sich bez. der Fehlergrößen nicht von denen mit A^{-1} unterscheidet. Hingegen verursacht die Variante (a) `inverse(A)` von Anfang an Ungenauigkeiten, die dann noch schnell anwachsen. Die MATLAB-Kommandos können also in diesem Bereich mit `linsolve(A,I)` nicht konkurrieren.

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$ $a'_{ij}=\mathcal{O}(\ )$		$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{nk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes Digits $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\|\tilde{a}'_{ij}-a'_{ij}\|$ Stelle (i_{max},j_{max})
7	1.825 10^9	(a)	15 $\tilde{a}'_{7,4}$ = -24024.0007876353	2.527 10^{-7}	7.9 10^{-4} (7,4)
	10^4		$\tilde{a}'_{7,3}$ = 20020.0006837867		
		(b)	$\tilde{a}'_{7,4}$ = -24024.	0	0
			$\tilde{a}'_{7,3}$ = 20020.		
		(a)	16 $\tilde{a}'_{7,4}$ = -24023.99987835507	2.025 10^{-8}	1.2 10^{-4} (7,4)
			$\tilde{a}'_{7,3}$ = 20019.99989769968		
		(b)	$\tilde{a}'_{7,4}$ = -24024.	0	0
			$\tilde{a}'_{7,3}$ = 20020.		
		(a)	17 $\tilde{a}'_{7,4}$ = -24024.000000307290	1.164 10^{-9}	3.6 10^{-7} (7,3)
			$\tilde{a}'_{7,3}$ = 20020.000000361726		
		(b)	$\tilde{a}'_{7,4}$ = -24024.	0	0
			$\tilde{a}'_{7,3}$ = 20020.		
	10^4		$a'_{7,4}$ = -24024	0	
			$a'_{7,3}$ = 20020		
10	2.678 10^{14}	(a)	15 $\tilde{a}'_{10,5}$ = -8.31853308542213 10^6	1.450 10^{-2}	4.5 10^3 (10,5)
	10^7		$\tilde{a}'_{10,6}$ = 7.76397071308267 10^6		
		(b)	$\tilde{a}'_{10,5}$ = -8.314020 10^6	0	0
			$\tilde{a}'_{10,6}$ = 7.759752 10^6		
		(a)	16 $\tilde{a}'_{10,5}$ = -8.312814291485192 10^6	1.948 10^{-3}	1.2 10^3 (10,5)
			$\tilde{a}'_{10,6}$ = 7.758637309154823 10^6		
		(b)	$\tilde{a}'_{10,5}$ = -8.314020 10^6	0	0
			$\tilde{a}'_{10,6}$ = 7.759752 10^6		
		(a)	17 $\tilde{a}'_{10,5}$ = -8.3141355559725266 10^6	1.210 10^{-4}	1.2 10^2 (10,5)
			$\tilde{a}'_{10,6}$ = 7.7598590807933210 10^6		
		(b)	$\tilde{a}'_{10,5}$ = -8.314020 10^6	0	0
			$\tilde{a}'_{10,6}$ = 7.759752 10^6		
	10^7		$a'_{10,5}$ = -8314020	0	
			$a'_{10,6}$ = 7759752		

n	Spektral-kondition $\mathrm{cond}_2(A)$ $\tilde{a}'_{ij}=\mathcal{O}(\ )$ $a'_{ij}=\mathcal{O}(\ )$	Digits	$A^{-1}=(a'_{ij})$, $\widetilde{A}^{-1}=(\tilde{a}'_{ij})$ $\tilde{a}'_{nk}$, $k\approx\frac{n}{2}$ betragsgrößtes & größtes $\tilde{a}'_{ij}$ a'_{ij}	$I-A\widetilde{A}^{-1}$ $\|I-A\widetilde{A}^{-1}\|_F$ $I-AA^{-1}$ $\|I-AA^{-1}\|_F$	$\max_{i,j}\lvert\tilde{a}'_{ij}-a'_{ij}\rvert$ Stelle (i_{max},j_{max})
11	1.472 10^{16} (a)	15	$\tilde{a}'_{11,6}=$ -5.84962761361236 10^7	6.560 10^{-1}	2.6 10^6 (11,6)
	10^8		$\tilde{a}'_{11,5}=$ 5.19613926819347 10^7		
	(b)		$\tilde{a}'_{11,6}=$ -61108047.	0	0
			$\tilde{a}'_{11,5}=$ 54318264.		
	(a)	16	$\tilde{a}'_{11,6}=$ -6.114881246756908 10^7	6.051 10^{-2}	4.1 10^4 (11,6)
			$\tilde{a}'_{11,5}=$ 5.435423374977763 10^7		
	(b)		$\tilde{a}'_{11,6}=$ -61108047.	0	0
			$\tilde{a}'_{11,5}=$ 54318264.		
	(a)	17	$\tilde{a}'_{11,6}=$ -6.1078205356652086 10^7	8.706 10^{-3}	3.0 10^4 (11,6)
			$\tilde{a}'_{11,5}=$ 5.4291380619478465 10^7		
	(b)		$\tilde{a}'_{11,6}=$ -61108047.	0	0
			$\tilde{a}'_{11,5}=$ 54318264.		
	10^8	15..17	$a'_{11,6}=$ -61108047	0	
			$a'_{11,5}=$ 54318264		
12	8.204 10^{17} (a)	15	$\tilde{a}'_{12,6}=$ 2.22784403569253 10^8	1.126 10^1	2.2 10^8 (12,6)
	10^8		$\tilde{a}'_{12,6}=$ 2.22784403569253 10^8		
	(b)		$\tilde{a}'_{12,6}=$ 440936496.	1.327 10^1	0
			$\tilde{a}'_{12,6}=$ 440936496.		
	10^8		$a'_{12,6}=$ 440936496	1.327 10^1	
			$a'_{12,6}=$ 440936496		
	(a)	16	$\tilde{a}'_{12,6}=$ 4.332872377787031 10^8	1.908 10^0	7.6 10^6 (12,6)
			$\tilde{a}'_{12,6}=$ 4.332872377787031 10^8		
	(b)		$\tilde{a}'_{12,6}=$ 440936496.	0	0
			$\tilde{a}'_{12,6}=$ 440936496.		
			$a'_{12,6}=$ 440936496	0	
			$a'_{12,6}=$ 440936496		
	(a)	17	$\tilde{a}'_{12,6}=$ 4.4153799102441881 10^8	2.026 10^{-1}	6.0 10^5 (12,6)
			$\tilde{a}'_{12,6}=$ 4.4153799102441881 10^8		
	(b)		$\tilde{a}'_{12,6}=$ 440936496.	0	0
			$\tilde{a}'_{12,6}=$ 440936496.		
			$a'_{12,6}=$ 440936496	0	
			$a'_{12,6}=$ 440936496		

13	4.627 10^{19} (a)	15	$\tilde{a}'_{13,8}$ = -7.52156448289628 10^7	6.878 10^1	3.2 10^9 (13,7)
	10^8		$\tilde{a}'_{13,9}$ = 6.16995818203720 10^7		
	(b)		$\tilde{a}'_{13,7}$ = 3287684400.	1.740 10^3	0
			$\tilde{a}'_{13,7}$ = 3287684400.		
	10^9		$a'_{13,7}$ = 3287684400	1.740 10^3	
			$a'_{13,7}$ = 3287684400		
	(a)	16	$\tilde{a}'_{13,7}$ = 6.183665695017593 10^9	2.697 10^2	2.9 10^9 (13,7)
	10^{10}		$\tilde{a}'_{13,7}$ = 6.183665695017593 10^9		
	(b)		$\tilde{a}'_{13,7}$ = 3287684400.	6.106 10^1	0
			$\tilde{a}'_{13,7}$ = 3287684400.		
			$a'_{13,7}$ = 3287684400	6.106 10^1	
			$a'_{13,7}$ = 3287684400		
	(a)	17	$\tilde{a}'_{13,6}$ = 3.1814031343324433 10^9	1.302 10^1	1.1 10^8 (13,7)
	10^9		$\tilde{a}'_{13,6}$ = 3.1814031343324433 10^9		
	(b)		$\tilde{a}'_{13,6}$ = 3287684400.	4.000 10^0	0
			$\tilde{a}'_{13,6}$ = 3287684400.		
			$a'_{13,7}$ = 3287684400	4.000 10^0	
			$a'_{13,7}$ = 3287684400		
16	8.754 10^{24} (a)	15	$\tilde{a}'_{16,6}$ = -5.71401222611683 10^8	2.619 10^3	1.3 10^{12} (16,8)
	10^9		$\tilde{a}'_{16,5}$ = 5.49516830436296 10^8		
	(b)		$\tilde{a}'_{16,8}$ = 1345374716400.	1.349 10^8	0
			$\tilde{a}'_{16,8}$ = 1345374716400.		
	10^{12}		$a'_{16,8}$ = 1345374716400	1.349 10^8	
			$a'_{16,8}$ = 1345374716400		
	(a)	16	$\tilde{a}'_{16,6}$ = -1.300313089058713 10^9	1.896 10^3	1.3 10^{12} (16,8)
	10^9		$\tilde{a}'_{16,7}$ = 1.151693019387510 10^9		
	(b)		$\tilde{a}'_{16,8}$ = 1345374716400.	1.504 10^7	0
			$\tilde{a}'_{16,8}$ = 1345374716400.		
			$a'_{16,8}$ = 1345374716400	1.504 10^7	
			$a'_{16,8}$ = 1345374716400		
	(a)	17	$\tilde{a}'_{16,7}$ = -4.5522159042902287 10^9	1.951 10^3	1.3 10^{12} (16,8)
	10^9		$\tilde{a}'_{16,6}$ = 4.0935611206208382 10^9		
	(b)		$\tilde{a}'_{16,8}$ = 1345374716400.	6.900 10^5	0
			$\tilde{a}'_{16,8}$ = 1345374716400.		
			$a'_{16,8}$ = 1345374716400	6.900 10^5	
			$a'_{16,8}$ = 1345374716400		

19	1.756 10^{30} (a)	15	$\tilde{a}'_{19,5}$ = -2.15432653995046 10^8	1.371 10^5	5.8 10^{14} (19,10)
	10^8		$\tilde{a}'_{19,4}$ = 1.73754414039790 10^8		
	(b)		$\tilde{a}'_{19,10}$ = -583057996306500.	9.216 10^{12}	0
			$\tilde{a}'_{19,9}$ = 544187463219400.		
	10^{15}		$a'_{19,10}$ = -583057996306500	9.216 10^{12}	
			$a'_{19,9}$ = 544187463219400		
	(a)	16	$\tilde{a}'_{19,7}$ = -4.317372702222652 $* 10^8$	2.761 10^5	5.8 10^{14} (19,10)
	10^8		$\tilde{a}'_{19,8}$ = 3.977086961303914 10^8		
	(b)		$\tilde{a}'_{19,10}$ = -583057996306500.	8.117 10^{11}	0
			$\tilde{a}'_{19,9}$ = 544187463219400.		
			$a'_{19,10}$ = -583057996306500	8.117 10^{11}	
			$a'_{19,9}$ = 544187463219400		
	(a)	17	$\tilde{a}'_{19,6}$ = -5.7111748424555925 10^9	6.319 10^4	5.8 10^{14} (19,10)
	10^{10}		$\tilde{a}'_{19,7}$ = 5.5924230927773507 10^9		
	(b)		$\tilde{a}'_{19,10}$ = -583057996306500.	4.184 10^{10}	0
			$\tilde{a}'_{19,9}$ = 544187463219400.		
			$a'_{19,10}$ = -583057996306500	4.184 10^{10}	
			$a'_{19,9}$ = 544187463219400		

Tab. 4.6 Berechnungen für die Boothroyd/Dekker-Matrix mit Maple, `Digits=15,16,17`
`A=(a(i,j)), a(i,j)=(n+i-1)!/((n-j)!*(i-1)!*(j-1)!*(i+j-1))`
2 Varianten der Ermittlung der Näherungsinversen $\widetilde{A^{-1}}$
(a) `inverse(A)`
(b) `linsolve(A,I)`

Wir berechnen für einige Dimensionen n den Unterschied $D_n = \max_{i,j} |\tilde{a}'_{1_{ij}} - \tilde{a}'_{2_{ij}}|$ zwischen den beiden Näherungsinversen A_1=`inverse(A)` und A_2=`linsolve(A,I)` sowie die realisierenden Indizes. Der Grund für die deutlichen Abweichungen liegt i. Allg. in der ungenauen Berechnung der Inversen mit dem Kommando `inverse(A)`, bei großem n dann mit beiden Varianten wegen zu schwacher GPA bei sehr schlechter Matrixkondition.

```
     Digits=15              Digits=16              Digits=17
 n      D_n      (i,j)         D_n      (i,j)         D_n      (i,j)
-------------------------------------------------------------------
 4   0                      0                      0
 7   7.876e-04  ( 7, 4)    1.216e-04  ( 7, 4)    3.617e-07  ( 7, 3)
10   4.513e+03  (10, 5)    1.206e+03  (10, 5)    1.156e+02  (10, 5)
11   2.612e+06  (11, 6)    4.077e+04  (11, 6)    2.984e+04  (11, 6)
12   2.182e+08  (12, 6)    7.649e+06  (12, 6)    6.015e+05  (12, 6)
13   3.235e+09  (13, 7)    2.896e+09  (13, 7)    1.063e+08  (13, 7)
16   1.346e+12  (16, 8)    1.346e+12  (16, 8)    1.342e+12  (16, 8)
19   5.831e+14  (19,10)    5.831e+14  (19,10)    5.831e+14  (19,10)
22   9.807e+10  (22, 7)    5.647e+13  (22, 8)    4.201e+15  (22,10)
```

4.4 Was leistet der Computer?

An solchen Untersuchungen und Beispielen zeigt sich, was Computer sowie Computersoftware schon und noch nicht leisten können. Der wirklich kreative Anteil am Problemlösungsprozess bleibt beim Menschen. Der Phantasie sind aber keine Grenzen gesetzt. Natürlich ist es zumeist nicht einfach, komplizierte Sachverhalte methodisch geschickt und didaktisch wirksam für Präsentationen vor einem Hörerkreis aufzubereiten. Das ist besonders schwierig, wenn Detailwissen gefragt ist und sehr spezifische Aspekte behandelt werden.
Der Komfort des Werkzeugs Computer einschließlich der Software kann sich aber nur "entfalten" durch den an Ideen reichen und kreativen Nutzer in der Einheit mit seiner gezielten Nutzung, gründlichen Analyse und ständigen Weiterentwicklung.

4.5 Dateien, Arbeitsblätter und Programme

Zu den einzelnen Kapiteln gibt es Arbeitsblätter in den CAS, Programme und diverse Dateien. Man findet diese auf der Seite *www.tu-ilmenau.de/site/math/neundorf.html* unter **Sonstiges → Maple → Preprints Maple 5**.

1. Rechengenauigkeit, Auswertung von Ausdrücken, Funktionen und Kommandos
   calc1.mws
2. Termumformung und mathematische Äquivalenz von Ausdrücken
   equiv_2.pas, equiv_1.pas, aequiv2.mws, aequiv2.m
3. Matrixgenerierung
   genarr2.mws
4. *LU*-Faktorisierung von Matrizen
   lu2.mws, lu1.m, luf2.m
5. Produkte, Differenzen und Potenzen mit Matrizen und Vektoren
   matvek1.mws
6. Integration, Romberg-Verfahren, Anhalten der Verarbeitung und Datenausgabe
   romberg2.mws
7. (2×2)-Matrizen, Eigenschaften
   matrix2x2.mws, matrix2x2.m
8. LGS, Matrixinverse
   lgs_inv2.pas, progonka3.pas
9. Determinante, Inverse, Norm, Kondition und EW/EV von Matrizen
   spd Tridiagonalmatrix *matr_trid1.mws, matr_trid1.m*
   Hilbert-Matrix *matr_hilb1.mws, matr_hilb1.m*
   Boothroyd/Dekker-Matrix *matr_booth1.mws, matr_booth1.m*
10. Spezielle Matrixeigenschaften, Genauigkeit der inversen Matrix
    spd Tridiagonalmatrix *matr_trid2.mws, matr_trid2.m*
    Hilbert-Matrix *matr_hilb2.mws, matr_hilb2.m*
    Boothroyd/Dekker-Matrix *matr_booth2.mws, matr_booth2.m*

Literaturverzeichnis

Maple

[1] Blachman, N. und Mossinghoff M. J.: *Maple griffbereit*. Vieweg Verlag Braunschweig 1996.

[2] Heal, K. M. et al.: *Handbuch Waterloo Maple*. Maple V - Learning Guide, Springer-Verlag New York 1998.

[3] Heck, A.: *Introduction to Maple*. 2nd Ed. Springer-Verlag New York 1996.

[4] Krawietz, A.: *Maple V für das Ingenieurstudium*. Springer-Verlag New York 1997.

[5] Klimek, G. und M. Klimek: *Discovering Curves and Surfaces with Maple*. Springer-Verlag New York 1997.

[6] Kofler, M.: *Maple V Release 4*. Addison Wesley Bonn 1996.

[7] Monagan, M.: *Programming in Maple: The Basics*. Institut für Wissenschaftliches Rechnen ETH-Zentrum, CH-8092 Zürich.

[8] Monagan, M. et al.: *Maple V Programming Guide*. Springer-Verlag New York 1996.

[9] Neundorf, W.: *Programming in Maple V Release 5*. Extended Basics. Preprint M 07/99 IfMath der TU Ilmenau, Februar 1999.

[10] Neundorf, W. und B. Walther: *Grafik, Animation und Ausgabeformate in Maple V Release 5*. Preprint M 12/00 IfMath der TU Ilmenau, Juni 2000.

[11] Nicolaides, R. A. und N. J. Walkington: *Maple: A Comprehensive Introduction*. Cambridge University Press 1996.

[12] Walz, A.: *Maple V. Rechnen und Programmieren mit Release 4*. R. Oldenbourg Verlag München 1998.

[13] Werner, W.: *Mathematik lernen mit Maple*. Bd. 1,2. Ein Lehr-und Arbeitsbuch für das Grundstudium. dpunkt Heidelberg 1996, 1998 (CD).

[14] Westermann, T.: *Mathematik für Ingenieure mit Maple*. Springer-Verlag 1996.

[15] Neundorf, W.: *Lösungsmethoden mit Maple. Betrachtung von Fehlern und Kondition sowie Darstellungsmöglichkeiten*. Preprint M 08/03 IfMath der TU Ilmenau, April 2003.

[16] Neundorf, W.: *Spezielle Aspekte zu CAS Maple und MATLAB. Rechengenauigkeit, Listen, Felder, Faktorisierungen, Dateiarbeit, TP → Maple, Maple → MATLAB mit Beispielen*. Preprint M 10/03 IfMath der TU Ilmenau, Juni 2003.

[17] Neundorf, W.: *Grundlagen der numerischen linearen Algebra*. Preprint M 04/04 IfMath der TU Ilmenau, Februar 2004.

Matlab

[18] Gramlich, G. und W. Werner: *Numerische Mathematik mit Matlab*. 1. Auflage. dpunkt.verlag Heidelberg 2000.

[19] Neundorf, W.: *MATLAB - Teil I: - Vektoren, Matrizen, lineare Gleichungssysteme*. Preprint M 20/99 IfMath der TU Ilmenau, Juli 1999.

[20] Neundorf, W.: *MATLAB - Teil II: - Speicheraspekte, spezielle LGS, SDV, EWP, Graphik, NLG, NLGS*. Preprint M 23/99 TUI, September 1999.

[21] Neundorf, W.: *MATLAB - Teil III: - Komplexe LGS, Interpolation, Splines*. Preprint M 10/00 IfMath der TU Ilmenau, Mai 2000.

[22] Neundorf, W.: *MATLAB - Teil IV: - Approximation, Numerische Intergration*. Preprint M 11/00 IfMath der TU Ilmenau, Mai 2000.

Numerik

[23] Deuflhard, P. und H. Hohmann: *Numerische Mathematik*. 1: Eine algorithmisch orientierte Einführung. 3. überarbeitete und erweiterte Auflage, Lehrbuch. Walter de Gruyter Berlin 2002.

[24] Hanke-Bourgeois, M.: *Grundlagen der Numerischen Mathematik und des Wissenschaftlichen Rechnens*. Mathematische Leitfäden. B. G. Teubner GmbH, Stuttgart 2002.

[25] Hermann, M.: *Numerische Mathematik*. R. Oldenbourg Verlag München 2001.

[26] Stoer, J. und R. Bulirsch: *Numerical mathematics 2*. An Introduction - under consideration of lectures by F. L. Bauer. 4. neu bearbeitete und erweiterte Auflage. Springer-Verlag Berlin 2000.

[27] Kiełbasiński, A. und H. Schwetlick: *Numerische lineare Algebra*. Mathematik für Naturwissenschaft und Technik Band 18, DVW, Berlin 1988.

[28] Hackbusch, W.: *Iterative Lösung großer schwach besetzter Gleichungssysteme*. Leitfäden der angewandten Mathematik und Mechanik Band 69. B. G. Teubner Stuttgart 1991.

[29] Maess, G.: *Vorlesungen über numerische Mathematik*. Band 1, 2. Akademie-Verlag Berlin 1984, 1988.

[30] Meister, A.: *Numerik linearer Gleichungssysteme*. Eine Einführung in moderne Verfahren. Friedr. Vieweg & Sohn VG mbH, Braunschweig 1999.

[31] Schwarz, H. R.: *Numerische Mathematik*. B. G. Teubner Stuttgart 1988.

[32] Schaback, R. und H. Werner: *Numerische Mathematik*. Springer-Verlag, Berlin 1992.

[33] Süli, E. und D. Mayers *An Introduction to Numerical Analysis*. Cambridge Textbooks. Cambridge University Press 2003.

[34] Neundorf, W.: *Numerische Mathematik*. Vorlesungen, Übungen, Algorithmen und Programme. Berichte aus der Mathematik. Shaker Verlag Aachen 2002.

[35] Neundorf, W.: *Wissenschaftliches Rechnen - Matrizen und lineare Gleichungssysteme*. Vorlesungsskript IfMath der TU Ilmenau, August 2002.

Anschrift:

Dr. rer. nat. habil. Werner Neundorf
Technische Universität Ilmenau, Institut für Mathematik
PF 10 05 65
D - 98684 Ilmenau

E-mail: werner.neundorf@tu-ilmenau.de
Homepage: http://www.tu-ilmenau.de/fakmn/neundorf.html
http://www.tu-ilmenau.de/site/math/neundorf.html