MOwNiT – Układy równań liniowych - metody bezpośrednie

Przygotował:
Szymon Budziak

Problem 1:

Elementy macierzy A o wymiarze $n\,x\,n$ są określone wzorem:

$$\begin{cases} a_{1j} = 1 \\ a_{ij} = \dfrac{1}{i+j-1} \quad dla \;\; i \neq 1 \end{cases} \qquad i, j = 1, \ldots, n$$

Przyjmij wektor x jako dowolną n−elementową permutację ze zbioru $\{1, \; -1\}$ i oblicz wektor **b**. Następnie metodą eliminacji Gaussa rozwiąż układ równań liniowych **Ax=b** (przyjmując jako niewiadomą wektor **x**). Przyjmij różną precyzję dla znanych wartości macierzy **A** i wektora **b**. Sprawdź, jak błędy zaokrągleń zaburzają rozwiązanie dla różnych rozmiarów układu (porównaj – zgodnie z wybraną normą – wektory **x** obliczony z **x** zadany). Przeprowadź eksperymenty dla różnych rozmiarów układu.

Rozmiary układu, które zostały przetestowane w tym zadaniu to: 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 30, 50, 70, 100, 150, 200, 300, 500. Przyjęta prezycja to float32, float64, float128 z biblioteki numpy.

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \begin{matrix} \\ \frac{a_{21}}{a_{11}} \} - \\ \end{matrix} \begin{matrix} \\ \\ \frac{a_{31}}{a_{11}} \} - \end{matrix}$$

Aby to zadziałało $a_{11} \neq 0$.
Po pierwszym etapie mamy:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(2)} \end{pmatrix}$$

$$a_{22}^{(2)} = a_{22} - \frac{a_{21}}{a_{11}} \cdot a_{12}, a_{23}^{(2)} = \ldots,$$

$$a_{32}^{(2)} = a_{32} - \frac{a_{31}}{a_{11}} \cdot a_{12}, a_{33}^{(2)} = \ldots,$$

$$b_2^{(2)} = b_2 - \frac{a_{21}}{a_{11}} \cdot b_1, \; b_3^{(2)} = b_3 - \frac{a_{31}}{a_{11}} \cdot b_1,$$

W wyniku następnego etapu otrzymamy:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & a_{32}^{(2)} & a_{33}^{(2)} \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(2)} \end{pmatrix} \begin{matrix} \\ \\ \frac{a_{32}^{(2)}}{a_{22}^{(2)}} \} - \end{matrix}$$

$$U = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} \\ 0 & 0 & a_{33}^{(3)} \end{pmatrix}, \; c = \begin{pmatrix} b_1 \\ b_2^{(2)} \\ b_3^{(3)} \end{pmatrix}$$

$$a_{33}^{(3)} = a_{33}^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} \cdot a_{23}^{(2)}$$

$$b_3^{(3)} = b_3^{(2)} - \frac{a_{32}^{(2)}}{a_{22}^{(2)}} \cdot b_2^{(2)}$$

Ogólnie po k etapach otrzymujemy:
$A^{(k+1)} \cdot x = b^{(k+1)}$

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \cdot a_{kj}^{(k)}, i, j > k \quad \boxed{1a}$$

$$b_{i}^{(k+1)} = b_{i}^{(k)} - \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \cdot b_{k}^{(k)}, i > k \quad \boxed{1b}$$

przy założeniu, że $a_{kk}^{(k)} \neq 0$
Jeśli $a_{kk}^{(k)} = 0$ należy przestawic wiersze.

W efekcie $A^{(n)} = U$ staje się macierzą trójkątną górną (upper triangular)

Elimination

for i := 1 to n-1 do
    begin
      p :=smallest integer $in[i,\ n] : a_{pi} \neq 0$;
      if no p then no unique solution exist!; STOP;
      if $p \neq i$ then $E_p \leftrightarrow E_i$ /przestawienie/
      for j := i + 1 to n do $(E_j - \frac{a_{ji}}{a_{ii}} E_i \rightarrow E_j)$
    end
if $a_{nn} = 0$ then no unique solution exist!; STOP;

Backward substitution

$x_n = b'_n / a'_{nn}$;

for i $:=$ n $-$ 1 downto 1 do $x_i = (b_i - \sum_{j=i+1}^{n} a_{ij} x_j)/a_{ii}$;

Złożoność obliczeniowa $O(n^3)$

Wyniki z problemu pierwszego

n	float32	float64	float128
3	0	0	0
4	$6.646e^{-15}$	$3.018e^{-13}$	0
5	$7.282e^{-13}$	$9.229e^{-12}$	$4.068e^{-12}$
6	$5.440e^{-11}$	$3.637e^{-10}$	$1.085e^{-11}$
7	$3.695e^{-09}$	$1.360e^{-08}$	$1.808e^{-09}$
8	$6.747e^{-09}$	$1.203e^{-07}$	$6.036e^{-09}$
9	$4.038e^{-08}$	$5.400e^{-07}$	$1.625e^{-06}$
10	$6.730e^{-09}$	$1.662e^{-04}$	$2.235e^{-05}$
11	$8.556e^{-09}$	$1.223e^{-02}$	$1.518e^{-03}$
12	$1.100e^{-08}$	1.213	$5.231e^{-02}$
13	$2.396e^{-08}$	$2.121e^{+01}$	$5.625e^{-01}$
14	$1.814e^{-08}$	$2.111e^{+01}$	$3.727e^{-01}$
15	$2.391e^{-08}$	$1.504e^{+01}$	$6.504e^{-01}$
20	$1.779e^{-08}$	$8.714e^{+02}$	8.780

30	$1.573e^{-08}$	$1.892e^{+02}$	6.828
50	$1.471e^{-07}$	$2.460e^{+02}$	7.419
70	$8.482e^{-08}$	$9.681e^{+02}$	$6.868e^{+01}$
100	$2.777e^{-06}$	$3.777e^{+03}$	$1.635e^{+02}$
150	$2.460e^{-07}$	$4.671e^{+03}$	$3.104e^{+02}$
200	$3.305e^{-07}$	$2.755e^{+03}$	$5.193e^{+01}$
300	$5.895e^{-07}$	$1.537e^{+04}$	$5.079e^{+01}$
500	$3.644e^{-06}$	$3.688e^{+04}$	$4.168e^{+01}$

Tabela 1: Błędy otrzymane w problemie pierwszym dla różnych precyzji

Wnioski

Możemy zauważyć, że błędy zwiększają się wraz z rosnącym rozmiarem układu. Również można zaobserwować, że błąd jest różny dla różnych precyzji. Dla typów float32 oraz float64 błędy te są znacznie większe niż błędy dla typu float128.

Problem 2:

Powtórz eksperyment dla macierzy zadanej wzorem:

$$\begin{cases} a_{ij} = \dfrac{2i}{j} \quad dla \quad j \geq i \\ a_{ij} = a_{ji} \quad dla \quad j < i \end{cases} \qquad i, j = 1, \ldots, n$$

Porównaj wyniki z tym, co otrzymano w przypadku układu z punktu 1). Spróbuj uzasadnić, skąd biorą się różnice w wynikach. Sprawdź uwarunkowanie obu układów.

Rozmiary układu, które zostały przetestowane w tym zadaniu to: 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 30, 50, 70, 100, 150, 200, 300, 500. Przyjęta prezycja to float32, float64, float128 z biblioteki numpy.

Wyniki z problemu drugiego

n	float32	float64	float128
3	0	$3.140e^{-16}$	$3.140e^{-16}$
4	0	$2.482e^{-16}$	$5.661e^{-16}$
5	$2.482e^{-16}$	$4.154e^{-16}$	$4.965e^{-16}$
6	$3.140e^{-16}$	$9.742e^{-16}$	$6.473e^{-16}$
7	$2.482e^{-16}$	$1.694e^{-15}$	$7.771e^{-16}$
8	$6.080e^{-16}$	$4.672e^{-15}$	$2.294e^{-15}$
9	$1.481e^{-15}$	$3.310e^{-15}$	$1.426e^{-15}$
10	$2.991e^{-15}$	$3.082e^{-15}$	$5.304e^{-15}$

11	$2.983e^{-15}$	$4.421e^{-15}$	$6.666e^{-15}$
12	$3.638e^{-15}$	$1.980e^{-14}$	$1.083e^{-14}$
13	$4.764e^{-15}$	$2.200e^{-14}$	$9.176e^{-15}$
14	$4.025e^{-15}$	$2.276e^{-14}$	$1.175e^{-14}$
15	$3.612e^{-15}$	$2.836e^{-14}$	$1.404e^{-14}$
20	$1.971e^{-14}$	$3.809e^{-14}$	$1.559e^{-14}$
30	$7.746e^{-14}$	$9.935e^{-14}$	$6.379e^{-14}$
50	$1.362e^{-13}$	$3.460e^{-13}$	$1.671e^{-13}$
70	$3.046e^{-13}$	$9.250e^{-13}$	$8.040e^{-13}$
100	$1.210e^{-12}$	$2.287e^{-12}$	$1.556e^{-12}$
150	$3.111e^{-12}$	$9.784e^{-12}$	$3.334e^{-12}$
200	$5.853e^{-12}$	$3.118e^{-11}$	$6.725e^{-12}$
300	$1.731e^{-11}$	$8.961e^{-11}$	$2.144e^{-11}$
500	$6.425e^{-11}$	$3.533e^{-10}$	$7.574e^{-11}$

Tabela 2: Błędy otrzymane w problemie drugim dla różnych precyzji

Porównanie wyników z problemu pierwszego z wynikami z problemu drugiego

n	ex 1 float32	ex 2 float32	ex 1 float64	ex 2 float64	ex 1 float128	ex 2 float128
3	0	0	0	$3.140e^{-16}$	0	$3.140e^{-16}$
4	$6.646e^{-15}$	0	$3.018e^{-13}$	$2.482e^{-16}$	0	$5.661e^{-16}$
5	$7.282e^{-13}$	$2.482e^{-16}$	$9.229e^{-12}$	$4.154e^{-16}$	$4.068e^{-12}$	$4.965e^{-16}$
6	$5.440e^{-11}$	$3.140e^{-16}$	$3.637e^{-10}$	$9.742e^{-16}$	$1.085e^{-11}$	$6.473e^{-16}$
7	$3.695e^{-09}$	$2.482e^{-16}$	$1.360e^{-08}$	$1.694e^{-15}$	$1.808e^{-09}$	$7.771e^{-16}$
8	$6.747e^{-09}$	$6.080e^{-16}$	$1.203e^{-07}$	$4.672e^{-15}$	$6.036e^{-09}$	$2.294e^{-15}$
9	$4.038e^{-08}$	$1.481e^{-15}$	$5.400e^{-07}$	$3.310e^{-15}$	$1.625e^{-06}$	$1.426e^{-15}$
10	$6.730e^{-09}$	$2.991e^{-15}$	$1.662e^{-04}$	$3.082e^{-15}$	$2.235e^{-05}$	$5.304e^{-15}$
11	$8.556e^{-09}$	$2.983e^{-15}$	$1.223e^{-02}$	$4.421e^{-15}$	$1.518e^{-03}$	$6.666e^{-15}$
12	$1.100e^{-08}$	$3.638e^{-15}$	1.213	$1.980e^{-14}$	$5.231e^{-02}$	$1.083e^{-14}$
13	$2.396e^{-08}$	$4.764e^{-15}$	$2.121e^{+01}$	$2.200e^{-14}$	$5.625e^{-01}$	$9.176e^{-15}$
14	$1.814e^{-08}$	$4.025e^{-15}$	$2.111e^{+01}$	$2.276e^{-14}$	$3.727e^{-01}$	$1.175e^{-14}$
15	$2.391e^{-08}$	$3.612e^{-15}$	$1.504e^{+01}$	$2.836e^{-14}$	$6.504e^{-01}$	$1.404e^{-14}$
20	$1.779e^{-08}$	$1.971e^{-14}$	$8.714e^{+02}$	$3.809e^{-14}$	8.780	$1.559e^{-14}$
30	$1.573e^{-08}$	$7.746e^{-14}$	$1.892e^{+02}$	$9.935e^{-14}$	6.828	$6.379e^{-14}$
50	$1.471e^{-07}$	$1.362e^{-13}$	$2.460e^{+02}$	$3.460e^{-13}$	7.419	$1.671e^{-13}$
70	$8.482e^{-08}$	$3.046e^{-13}$	$9.681e^{+02}$	$9.250e^{-13}$	$6.868e^{+01}$	$8.040e^{-13}$
100	$2.777e^{-06}$	$1.210e^{-12}$	$3.777e^{+03}$	$2.287e^{-12}$	$1.635e^{+02}$	$1.556e^{-12}$

150	2.460e-07	3.111e-12	4.671e+03	9.784e-12	3.104e+02	3.334e-12
200	3.305e-07	5.853e-12	2.755e+03	3.118e-11	5.193e+01	6.725e-12
300	5.895e-07	1.731e-11	1.537e+04	8.961e-11	5.079e+01	2.144e-11
500	3.644e-06	6.425e-11	3.688e+04	3.533e-10	4.168e+01	7.574e-11

Tabela 3: Porównanie błędów z problemu pierwszego oraz drugiego dla różnych precyzji

Wskaźnik uwarunkowania

Dodatkowo. możemy również obliczyć wskaźnik uwarunkowania macierzy. Wartość ta jest miarą jak bardzo zmieni się rozwiązanie **x** układu równań w stosunku do zmiany **b**. Jeżeli wskaźnik macierzy jest duży, to nawet mały błąd **b** może spowodować duże błędy w **x**. Wskaźnik uwarunkowania macierzy obliczamy ze wzoru:

$\kappa = ||A^{-1}|| \cdot ||A||$

n	ex 1 condition number	ex 2 condition number
3	2.160e+02	1.444444
4	2.880e+03	1.833333
5	2.800e+04	2.233333
6	2.268e+05	2.644444
7	1.629e+06	3.031746
8	1.286e+07	3.448413
9	1.120e+08	3.849206
10	8.841e+08	4.249206
11	6.473e+09	4.659428
12	4.407e+10	5.055219

13	$1.347e^{+11}$	5.465475
14	$2.459e^{+11}$	5.868898
15	$1.733e^{+11}$	6.268898
20	$4.003e^{+11}$	8.289565
30	$5.932e^{+11}$	12.331882
50	$2.434e^{+12}$	20.420510
70	$8.602e^{+13}$	28.508027
100	$6.974e^{+14}$	40.638622
150	$1.536e^{+15}$	60.855672
200	$1.378e^{+16}$	81.073053
300	$3.584e^{+18}$	121.508544
500	$2.156e^{+18}$	202.379076

Tabela 4: Wskaźnik uwarunkowania dla problemu 1 oraz problemu 2

Z powyższej tabeli możemy zauważyć, że wskaźnik uwarunkowania dla macierzy z problemu 1 jest znacznie większy od problemu 2. Oznacza to, że niewielki błąd znacznie wpływa na wynik.

Wnioski

Dzięki tabelom możemy zauważyć, że jedną z przyczyn złych wyników dla układu równań z pierwszego problemu jest metoda wybierania elementu wiodącego. W implementacji z Problemu 1 algorytm eliminacji Gaussa jako zawsze jako element wiodący wybiera kolejne elementy przekątnej macierzy. W przypadku macierzy z Problemu 1, elementy na przekątnej redukują się na tyle, że osiągają wartość rzędu nawet 10^{-18}. W przypadku macierzy z Problemu 2 jest to wielkość rzędu 10^{-2}, więc jest tutaj znaczna różnica. Mała wartość w przypadku Problemu 2 stanowi problem, ponieważ poszczególne wiersze w metodzie eliminacji Gaussa są dzielone

przez element wiodący, to znaczy mnożone przez jego odwrotność. Jeżeli element ten ma małą wartość (< 1), to wiersze mnożone są przez duża wartość , więc błędy stają się duże w stosunku do współczynników oryginalnej macierzy. Możemy zauważyć, że wskaźniki uwarunkowania dla macierzy w problemie 1 są znacznie większe od wskaźników uwarunkowania w problemie 2. Oznacza to, że niewielki błąd znacznie wpływa na wyniki.

Problem 3:

Powtórz eksperyment dla jednej z macierzy zadanej wzorem poniżej (macierz i parametry podane w zadaniu indywidualnym). Następnie rozwiąż układ metodą przeznaczoną do rozwiązywania układów z macierzą trójdiagonalną. Porównaj wyniki otrzymane dwoma metodami (czas, dokładność obliczeń i zajętość pamięci) dla różnych rozmiarów układu. Przy porównywaniu czasów należy pominąć czas tworzenia układu. Opisz, jak w metodzie dla układów z macierzą trójdiagonalną przechowywano i wykorzystywano macierz A.

$(m,k$ - parametry zadania):

$$\begin{cases} a_{i,i} = -m \cdot i - k \\ a_{i,i+1} = i \\ a_{i,i-1} = \dfrac{m}{i} \quad dla \;\; i > 1 \\ a_{i,j} = 0 \qquad dla \;\; j < i-1 \quad oraz \quad j > i+1 \end{cases} \qquad i, j = 1, \ldots, n$$

parametry zadania: **k = 8, m = 3**.

Rozmiary układu, które zostały przetestowane w tym zadaniu to: 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 20, 30, 50, 70, 100, 150, 200, 300, 500. Przyjęta prezycja to domyślna prezycja w Pythonie czyli float64.

The forward sweep consists of the computation of new coefficients as follows, denoting the new coefficients with primes:

$$c'_i = \begin{cases} \dfrac{c_i}{b_i}, & i = 1, \\ \dfrac{c_i}{b_i - a_i c'_{i-1}}, & i = 2, 3, \ldots, n-1 \end{cases}$$

and

$$d'_i = \begin{cases} \dfrac{d_i}{b_i}, & i = 1, \\ \dfrac{d_i - a_i d'_{i-1}}{b_i - a_i c'_{i-1}}, & i = 2, 3, \ldots, n. \end{cases}$$

The solution is then obtained by back substitution:

$$x_n = d'_n,$$
$$x_i = d'_i - c'_i x_{i+1}, \quad i = n-1, n-2, \ldots, 1.$$

The method above does not modify the original coefficient vectors, but must also keep track of the new coefficients. If the coefficient vectors may be modified, then an algorithm with less bookkeeping is:

For $i = 2, 3, \ldots, n$, do

$$w = \frac{a_i}{b_{i-1}},$$
$$b_i := b_i - w c_{i-1},$$
$$d_i := d_i - w d_{i-1},$$

followed by the back substitution

$$x_n = \frac{d_n}{b_n},$$
$$x_i = \frac{d_i - c_i x_{i+1}}{b_i} \quad \text{for } i = n-1, n-2, \ldots, 1.$$

Wyniki z problemu trzeciego

n	gaussian norm	thomas norm	gaussian time [s]	thomas time [s]
3	$2.220e^{-16}$	$2.220e^{-16}$	0.000202	0.000027
4	$2.220e^{-16}$	$2.220e^{-16}$	0.001041	0.000031
5	$2.220e^{-16}$	$2.220e^{-16}$	0.000665	0.000034
6	$2.220e^{-16}$	$2.220e^{-16}$	0.000381	0.000034
7	$2.482e^{-16}$	$2.482e^{-16}$	0.000459	0.000037
8	$2.220e^{-16}$	$2.220e^{-16}$	0.000450	0.000041
9	$2.220e^{-16}$	$2.220e^{-16}$	0.000512	0.000044
10	$2.220e^{-16}$	$2.220e^{-16}$	0.000643	0.000048

11	$2.220e^{-16}$	$2.220e^{-16}$	0.000726	0.000050
12	$2.220e^{-16}$	$2.220e^{-16}$	0.000866	0.000055
13	$2.220e^{-16}$	$2.220e^{-16}$	0.000999	0.000078
14	$2.220e^{-16}$	$2.220e^{-16}$	0.000797	0.000063
15	$3.140e^{-16}$	$3.140e^{-16}$	0.000885	0.000073
20	$4.154e^{-16}$	$4.154e^{-16}$	0.002333	0.000088
30	$4.839e^{-16}$	$4.839e^{-16}$	0.004216	0.000125
50	$8.382e^{-16}$	$8.382e^{-16}$	0.008047	0.000162
70	$1.023e^{-15}$	$1.023e^{-15}$	0.011174	0.000160
100	$1.164e^{-15}$	$1.164e^{-15}$	0.030569	0.000201
150	$1.456e^{-15}$	$1.456e^{-15}$	0.040810	0.000319
200	1.716e-15	$1.716e^{-15}$	0.073106	0.000383
300	$2.192e^{-15}$	$2.192e^{-15}$	0.161571	0.000519
500	$2.793e^{-15}$	$2.793e^{-15}$	0.471024	0.001004

Tabela 5: Błędy oraz czasy otrzymane w problemie trzecim dla algorytmu Gaussa oraz Thomasa

Wnioski

Z Tabeli 4 możemy zaobserwować, że niezależnie której metody użyjemy to błędy (norm) są takie same. Wyniki te są przewidywalne, ponieważ metody Gaussa i Thomasa są bardzo podobne, a w niektórych miejscach identyczne. Metoda Thomasa ogranicza się tylko do działania na elementach trójdiagonali. Jeśli chodzi o czasy działania to metoda Thomasa jest znacznie szybsza od metody eliminacji Gaussa. Nie jest to jednak zaskakujące, ponieważ wykonuje ona znacznie mniej operacji, ograniczając się tylko do trójdiagonali macierzy. Metoda gaussa jest skutecznym i prostym sposobem na rozwiązywanie układów równań

liniowych. W niektórych przypadkach jest ona jednak wolna z powodu błędów dokładności, które są spowodowane słabym uwarunkowaniem układów wejściowych lub sposobu wybierania elementu wejściowego. W przypadku, gdy macierz jest trójdiagonalna, to warto stosować zamiast metody eliminacji Gaussa metodę Thomasa. Jest to uproszczona wersja algorytmu Gaussa, która daje wyniki o tej samej dokładności jednak działa znacznie szybciej.

Literatura

- Wykład nr 8 dr Rycerz z przedmiotu MOwNiT
- Wikipedia na temat algorytmu Gaussa oraz algorytmu Thomasa