Rozwiązanie zadania 4N z zestawu 2

Rozwiązać równanie **Ax** = **e**, gdzie **A** jest macierzą (5) z zad 3, natomiast **e** jest wektorem, którego wszystkie składowe są równe 1, za pomocą
(a) metody Gaussa-Seidela
(b) metody gradientów sprzężonych

Metoda Gaussa-Seidela:

```
#include <iostream>
#include <cmath>
#include <iomanip>
using namespace std;

const int wymiar = 128;
const int iteracje = 40;

int main() {
	double A[wymiar][wymiar] = {0};
	double x[wymiar]; //rozwiazania
	double e[wymiar]; //wyraz wolny
	double norma[wymiar];
	double nowanorma[wymiar];

	for(int i=0; i<wymiar; i++){
			x[i] = e[i] = norma[i] = 1;
			for(int j=0; j<wymiar; j++) {
				if(i==j) {
					A[i][j] = 4;
					if(j<wymiar-1) A[i+1][j] = A[i][j+1] = 1;
					 if(j<wymiar-4) A[i+4][j] = A[i][j+4] = 1;
				}
			}
	}

	for(int i=0; i<iteracje; i++) {
		x[0] = (e[0]-A[0][1]*x[1]-A[0][4]*x[4])/4;
		x[1] = (e[1]-A[1][0]*x[0]-A[1][2]*x[2]-A[1][5]*x[5])/4;
		x[2] = (e[2]-A[2][1]*x[1]-A[2][3]*x[3]-A[2][6]*x[6])/4;
		x[3] = (e[3]-A[3][2]*x[2]-A[3][4]*x[4]-A[3][7]*x[7])/4;

		for(int j=4; j<wymiar-4; j++) x[j] = (e[j]-A[j][j-4]*x[j-4]-A[j][j-1]*x[j-1]-A[j][j+1]*x[j+1]-
								A[j][j+4]*x[j+4])/4;

		x[wymiar-4] = (e[wymiar-4]-A[wymiar-4][wymiar-8]*x[wymiar-8]-A[wymiar-4][wymiar-5]*
				x[wymiar- 5]-A[wymiar-4][wymiar-3]*x[wymiar-3])/4;
		x[wymiar-3] = (e[wymiar-3]-A[wymiar-3][wymiar-7]*x[wymiar-7]-A[wymiar-3][wymiar-4]*
				x[wymiar-4]-A[wymiar-3][wymiar-2]*x[wymiar-2])/4;
		x[wymiar-2] = (e[wymiar-2]-A[wymiar-2][wymiar-6]*x[wymiar-6]-A[wymiar-2][wymiar-3]*
				x[wymiar-3]-A[wymiar-2][wymiar-1]*x[wymiar-1])/4;
		x[wymiar-1] = (e[wymiar-1]-A[wymiar-1][wymiar-5]*x[wymiar-5]-A[wymiar-1][wymiar-2]*
				x[wymiar-2])/4;
```

```
        for(int j=0; j<wymiar; j++) nowanorma[j] = x[j];

        double norm= 0;
        for(int j=0; j<wymiar; j++) norm = norm+(nowanorma[j]-norma[j])*(nowanorma[j]-norma[j]);
        norm = sqrt(norm);

        for(int j=0; j<wymiar; j++) norma[j] = nowanorma[j];

        if(i>=1) cout << setprecision(12) << fixed << "||xk - x(k-1)|| = " << norm << endl;
    }

    for(int i=0; i<wymiar; i++) cout << "x" << i << " = " << x[i] << endl;
}
```

Metoda gradientów sprzężonych:

```
#include <iostream>
#include <cmath>
#include <iomanip>
using namespace std;

const int wymiar = 128;
const int iteracje = 40;

int main() {
    double A[wymiar][wymiar] = {0};
    double x[wymiar]; //rozwiazania
    double e[wymiar]; //wyraz wolny
    double p[wymiar]; //wektor p ze wzoru na gradienty sprzezone
    double r[wymiar]; //tez ze wzoru
    double Ap[wymiar];
    double rn[wymiar]; //ze wzoru r(k+1)
    double pn[wymiar]; // p(k+1)
    double norma[wymiar];
    double nowanorma[wymiar];

    for(int i=0; i<wymiar; i++){
        x[i] = e[i] = norma[i] = 1;

        for(int j=0; j<wymiar; j++) {
            if(i==j) {
                A[i][j] = 4;
                if(j<wymiar-1) A[i+1][j] = A[i][j+1] = 1;
                if(j<wymiar-4) A[i+4][j] = A[i][j+4] = 1;
            }
        }
    }

    r[0] = e[0]+(-A[0][0]*x[0]-A[0][1]*x[1]-A[0][4]*x[4]);
    r[1] = e[1]+(-A[1][0]*x[0]-A[1][1]*x[1]-A[1][2]*x[2]-A[1][5]*x[5]);
```

```
r[2] = e[2]+(-A[2][1]*x[1]-A[2][2]*x[2]-A[2][3]*x[3]-A[2][6]*x[6]);
r[3] = e[3]+(-A[3][2]*x[2]-A[3][3]*x[3]-A[3][4]*x[4]-A[3][7]*x[7]);

for(int j=4; j<wymiar-4; j++) r[j] = e[j]-(A[j][j-4]*x[j-4]+A[j][j-1]* x[j-1]+A[j][j]*
                                        x[j]+A[j][j+1]*x[j+1]+A[j][j+4]*x[j+4]);

r[wymiar-4] = e[wymiar-4]+(-A[wymiar-4][wymiar-8]*x[wymiar-8]-A[wymiar-4][wymiar-5]*
              x[wymiar-5]-A[wymiar-4][wymiar-4]*x[wymiar-4]-A[wymiar-4][wymiar-3]*x[wymiar-3]);
r[wymiar-3] = e[wymiar-3]+(-A[wymiar-3][wymiar-7]*x[wymiar-7]-A[wymiar-3][wymiar-4]*
              x[wymiar-4]-A[wymiar-3][wymiar-3]*x[wymiar-3]-A[wymiar-3][wymiar-2]*x[wymiar-2]);
r[wymiar-2] = e[wymiar-2]+(-A[wymiar-2][wymiar-6]*x[wymiar-6]-A[wymiar-2][wymiar-3]*
              x[wymiar-3]-A[wymiar-2][wymiar-2]*x[wymiar-2]-A[wymiar-2][wymiar-1]*x[wymiar-1]);
r[wymiar-1] = e[wymiar-1]+(-A[wymiar-1][wymiar-5]*x[wymiar-5]-A[wymiar-1][wymiar-2]*
              x[wymiar-2]-A[wymiar-1][wymiar-1]*x[wymiar-1]);

for(int i=0; i<wymiar; i++) p[i] = r[i];

for(int i=0; i<iteracje; i++) {
    Ap[0] = A[0][0]*p[0]+A[0][1]*p[1]+A[0][4]*p[4];
    Ap[1] = A[1][0]*p[0]+A[1][1]*p[1]+A[1][2]*p[2]+A[1][5]*p[5];
    Ap[2] = A[2][1]*p[1]+A[2][2]*p[2]+A[2][3]*p[3]+A[2][6]*p[6];
    Ap[3] = A[3][2]*p[2]+A[3][3]*p[3]+A[3][4]*p[4]+A[3][7]*p[7];

    for(int j=4; j<wymiar-4; j++) Ap[j] = A[j][j-4]*p[j-4]+A[j][j-1]*p[j-1]+A[j][j]*p[j]+A[j][j+1]*
                                          p[j+1]+A[j][j+4]*p[j+4];

    Ap[wymiar-4] = A[wymiar-4][wymiar-8]*p[wymiar-8]+A[wymiar-4][wymiar-5]*
                   p[wymiar-5]+A[wymiar-4][wymiar-4]*p[wymiar-4]+A[wymiar-4][wymiar-3]*
                   p[wymiar-3];
    Ap[wymiar-3] = A[wymiar-3][wymiar-7]*p[wymiar-7]+A[wymiar-3][wymiar-4]*
                   p[wymiar-4]+A[wymiar-3][wymiar-3]*p[wymiar-3]+A[wymiar-3][wymiar-2]*
                   p[wymiar-2];
    Ap[wymiar-2] = A[wymiar-2][wymiar-6]*p[wymiar-6]+A[wymiar-2][wymiar-3]*
                   p[wymiar-3]+A[wymiar-2][wymiar-2]*p[wymiar-2]+A[wymiar-2][wymiar-1]*
                   p[wymiar-1];
    Ap[wymiar-1] = A[wymiar-1][wymiar-5]*p[wymiar-5]+A[wymiar-1][wymiar-2]*
                   p[wymiar-2]+A[wymiar-1][wymiar-1]*p[wymiar-1];


    double licznik, mianownik, alfa, beta;
    licznik = mianownik = alfa = beta = 0;

    for(int j=0; j<wymiar; j++) {
        licznik = licznik + r[j]*r[j];
        mianownik = mianownik + p[j]*Ap[j];
    }
    alfa = licznik/mianownik;

    for(int j=0; j<wymiar; j++) {
        rn[j] = r[j] - alfa*Ap[j];
        x[j] = x[j] + alfa*p[j];
    }

    mianownik = licznik;
    licznik = 0;
    for(int j=0; j<wymiar; j++) licznik = licznik + rn[j]*rn[j];
```

```
            beta = licznik/mianownik;

            for(int j=0; j<wymiar; j++) pn[j] = rn[j] + beta*p[j];

            for(int j=0; j<wymiar; j++) {
                p[j] = pn[j];
                r[j] = rn[j];
            }

            for(int j=0; j<wymiar; j++) nowanorma[j] = x[j];

            double norm = 0;
            for(int j=0; j<wymiar; j++) norm = norm+(nowanorma[j]-norma[j])*(nowanorma[j]-norma[j]);
            norm = sqrt(norm);

            for(int j=0; j<wymiar; j++) norma[j] = nowanorma[j];

            if(i>=1) cout << setprecision(12) << fixed << "||xk - x(k-1)|| = " << norm << endl;
        }

        for(int i=0; i<wymiar; i++) cout << "x" << i << " = " << x[i] << endl;
    }
```

Wyniki działania programów

Metoda Gaussa-Seidela:

```
||xk - x(k-1)|| = 4.347526282501
||xk - x(k-1)|| = 1.430748983160
||xk - x(k-1)|| = 0.471072861006
||xk - x(k-1)|| = 0.154869041634
||xk - x(k-1)|| = 0.051372333179
||xk - x(k-1)|| = 0.017263732423
||xk - x(k-1)|| = 0.006242350825
||xk - x(k-1)|| = 0.002642480878
||xk - x(k-1)|| = 0.001439319880
||xk - x(k-1)|| = 0.000915965548
||xk - x(k-1)|| = 0.000625614674
||xk - x(k-1)|| = 0.000437571586
||xk - x(k-1)|| = 0.000310315435
||xk - x(k-1)|| = 0.000222012213
||xk - x(k-1)|| = 0.000160001310
||xk - x(k-1)|| = 0.000116009217
||xk - x(k-1)|| = 0.000084551844
||xk - x(k-1)|| = 0.000061903004
||xk - x(k-1)|| = 0.000045499954
||xk - x(k-1)|| = 0.000033559499
||xk - x(k-1)|| = 0.000024828561
```

Metoda gradientów sprzężonych:

```
||xk - x(k-1)|| = 0.332773183960
||xk - x(k-1)|| = 0.074073759768
||xk - x(k-1)|| = 0.023176288362
||xk - x(k-1)|| = 0.010205400914
||xk - x(k-1)|| = 0.004870100518
||xk - x(k-1)|| = 0.002793505377
||xk - x(k-1)|| = 0.002110760773
||xk - x(k-1)|| = 0.001485902085
||xk - x(k-1)|| = 0.000884308742
||xk - x(k-1)|| = 0.000429385951
||xk - x(k-1)|| = 0.000211693992
||xk - x(k-1)|| = 0.000117530404
||xk - x(k-1)|| = 0.000073097644
||xk - x(k-1)|| = 0.000049650735
||xk - x(k-1)|| = 0.000030263434
||xk - x(k-1)|| = 0.000016614641
||xk - x(k-1)|| = 0.000008596052
||xk - x(k-1)|| = 0.000004698531
||xk - x(k-1)|| = 0.000002881458
||xk - x(k-1)|| = 0.000001584908
||xk - x(k-1)|| = 0.000000554669
```

||xk - x(k-1)|| = 0.000018419295
||xk - x(k-1)|| = 0.000013697926
||xk - x(k-1)|| = 0.000010209164
||xk - x(k-1)|| = 0.000007624073
||xk - x(k-1)|| = 0.000005703819
||xk - x(k-1)|| = 0.000004274222
||xk - x(k-1)|| = 0.000003207749
||xk - x(k-1)|| = 0.000002410698
||xk - x(k-1)|| = 0.000001813999
||xk - x(k-1)|| = 0.000001366602
||xk - x(k-1)|| = 0.000001030672
||xk - x(k-1)|| = 0.000000778106
||xk - x(k-1)|| = 0.000000587987
||xk - x(k-1)|| = 0.000000444713
||xk - x(k-1)|| = 0.000000336629
||xk - x(k-1)|| = 0.000000255011
||xk - x(k-1)|| = 0.000000193322
||xk - x(k-1)|| = 0.000000146656
x0 = 0.194276803143
x1 = 0.130930197375
x2 = 0.146794905195
x3 = 0.162311327101
x4 = 0.091962599350
x5 = 0.135207496702
x6 = 0.119578852038
x7 = 0.111997198113
x8 = 0.140353962114
x9 = 0.116698372525
x10 = 0.127684994343
x11 = 0.129767053859
x12 = 0.117925997315
x13 = 0.129960046367
x14 = 0.123215743391
x15 = 0.123323606434
x16 = 0.128214932301
x17 = 0.122319713141
x18 = 0.126168377920
x19 = 0.125507835221
x20 = 0.123570968833
x21 = 0.126377778548
x22 = 0.124283199980
x23 = 0.124905712643
x24 = 0.125615565858
x25 = 0.124315015689
x26 = 0.125415325979
x27 = 0.124970544514
x28 = 0.124746049459
x29 = 0.125331256117
x30 = 0.124769941940
x31 = 0.125050735124
x32 = 0.125098428297
x33 = 0.124843977875
x34 = 0.125122908307
x35 = 0.124958145590
x36 = 0.124965529538

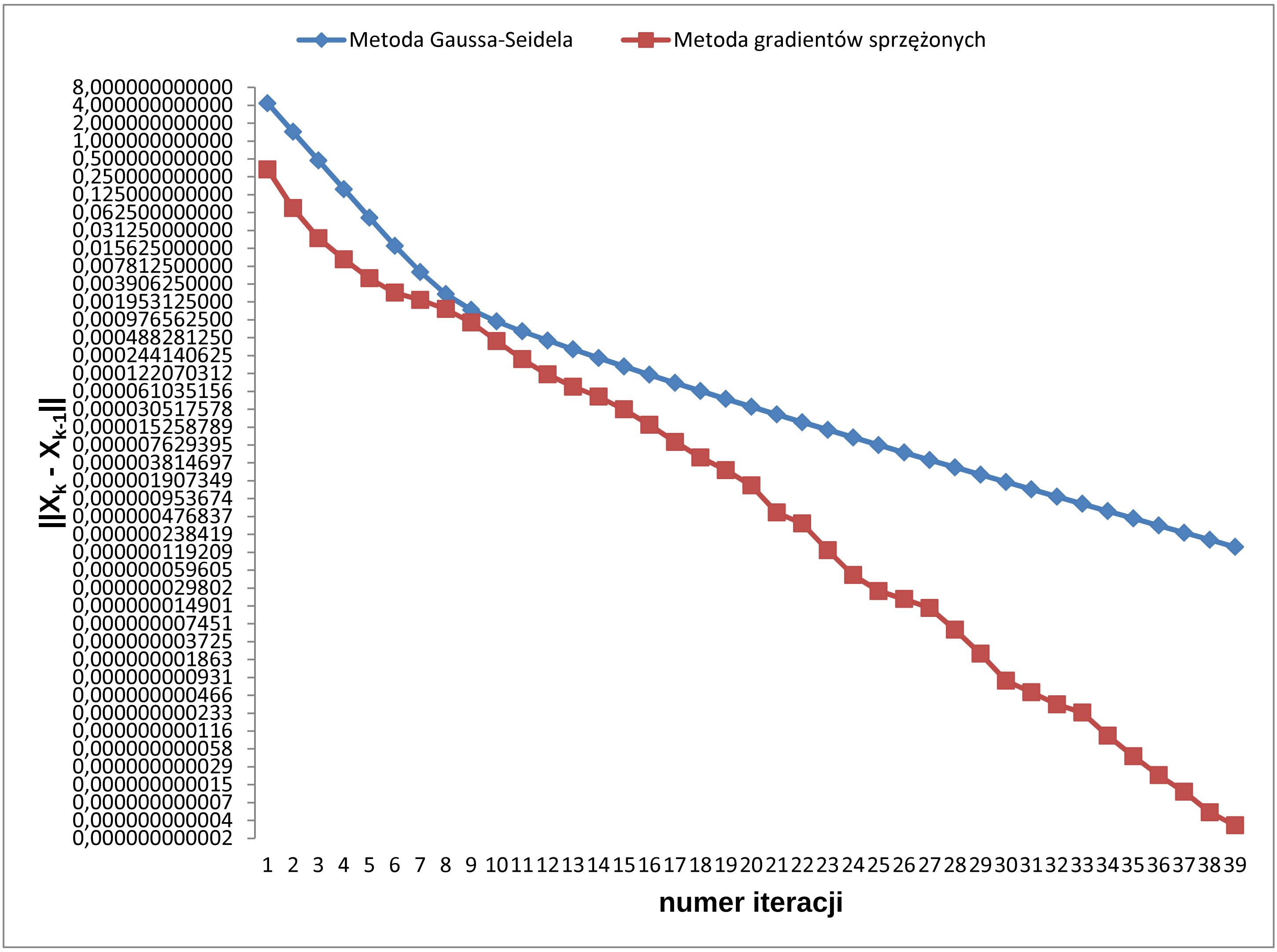
||xk - x(k-1)|| = 0.000000363768
||xk - x(k-1)|| = 0.000000129328
||xk - x(k-1)|| = 0.000000049167
||xk - x(k-1)|| = 0.000000026565
||xk - x(k-1)|| = 0.000000019393
||xk - x(k-1)|| = 0.000000013728
||xk - x(k-1)|| = 0.000000005886
||xk - x(k-1)|| = 0.000000002334
||xk - x(k-1)|| = 0.000000000821
||xk - x(k-1)|| = 0.000000000521
||xk - x(k-1)|| = 0.000000000328
||xk - x(k-1)|| = 0.000000000238
||xk - x(k-1)|| = 0.000000000097
||xk - x(k-1)|| = 0.000000000044
||xk - x(k-1)|| = 0.000000000021
||xk - x(k-1)|| = 0.000000000011
||xk - x(k-1)|| = 0.000000000005
||xk - x(k-1)|| = 0.000000000003
x0 = 0.194276795444
x1 = 0.130930202478
x2 = 0.146794908343
x3 = 0.162311315286
x4 = 0.091962615745
x5 = 0.135207486300
x6 = 0.119578848862
x7 = 0.111997214768
x8 = 0.140353939990
x9 = 0.116698387716
x10 = 0.127684995141
x11 = 0.129767036788
x12 = 0.117926021811
x13 = 0.129960027706
x14 = 0.123215746070
x15 = 0.123323621125
x16 = 0.128214908273
x17 = 0.122319733580
x18 = 0.126168371747
x19 = 0.125507824368
x20 = 0.123570990391
x21 = 0.126377757954
x22 = 0.124283208997
x23 = 0.124905719266
x24 = 0.125615547841
x25 = 0.124315035216
x26 = 0.125415315047
x27 = 0.124970541729
x28 = 0.124746063762
x29 = 0.125331238295
x30 = 0.124769953871
x31 = 0.125050735008
x32 = 0.125098417087
x33 = 0.124843993972
x34 = 0.125122896169
x35 = 0.124958147283
x36 = 0.124965538910

x37 = 0.125071487401
x38 = 0.124936308568
x39 = 0.125028242411
x40 = 0.125009816653
x41 = 0.124968242394
x42 = 0.125032120003
x43 = 0.124982762309
x44 = 0.124998722958
x45 = 0.125013597920
x46 = 0.124984214534
x47 = 0.125009863339
x48 = 0.124998925855
x49 = 0.124994438122
x50 = 0.125007553103
x51 = 0.124994643745
x52 = 0.125001280596
x53 = 0.125002159325
x54 = 0.124996497314
x55 = 0.125002732188
x56 = 0.124999136052
x57 = 0.124999159324
x58 = 0.125001562962
x59 = 0.124998784148
x60 = 0.125000301168
x61 = 0.125000491374
x62 = 0.124999305590
x63 = 0.125000284379
x64 = 0.125000361480
x65 = 0.124999280122
x66 = 0.125000447690
x67 = 0.125000385643
x68 = 0.124998714759
x69 = 0.125001570478
x70 = 0.124999220377
x71 = 0.124999044921
x72 = 0.125002794116
x73 = 0.124996506330
x74 = 0.125002082721
x75 = 0.125001377485
x76 = 0.124994588335
x77 = 0.125007530068
x78 = 0.124994527652
x79 = 0.124998824174
x80 = 0.125009913805
x81 = 0.124984248180
x82 = 0.125013498730
x83 = 0.124998828368
x84 = 0.124982714825
x85 = 0.125032079743
x86 = 0.124968347493
x87 = 0.125009708804
x88 = 0.125028288914
x89 = 0.124936351228
x90 = 0.125071380468
x91 = 0.124965638166

x37 = 0.125071472559
x38 = 0.124936320200
x39 = 0.125028240784
x40 = 0.125009807432
x41 = 0.124968256682
x42 = 0.125032109690
x43 = 0.124982761950
x44 = 0.124998733895
x45 = 0.125013583591
x46 = 0.124984222410
x47 = 0.125009867832
x48 = 0.124998911447
x49 = 0.124994452650
x50 = 0.125007549247
x51 = 0.124994632863
x52 = 0.125001299834
x53 = 0.125002145113
x54 = 0.124996495087
x55 = 0.125002751634
x56 = 0.124999111241
x57 = 0.124999171975
x58 = 0.125001573657
x59 = 0.124998754274
x60 = 0.125000331594
x61 = 0.125000482090
x62 = 0.124999284036
x63 = 0.125000326019
x64 = 0.125000326019
x65 = 0.124999284036
x66 = 0.125000482090
x67 = 0.125000331594
x68 = 0.124998754274
x69 = 0.125001573657
x70 = 0.124999171975
x71 = 0.124999111241
x72 = 0.125002751634
x73 = 0.124996495087
x74 = 0.125002145113
x75 = 0.125001299834
x76 = 0.124994632863
x77 = 0.125007549247
x78 = 0.124994452650
x79 = 0.124998911447
x80 = 0.125009867832
x81 = 0.124984222410
x82 = 0.125013583591
x83 = 0.124998733895
x84 = 0.124982761950
x85 = 0.125032109690
x86 = 0.124968256682
x87 = 0.125009807432
x88 = 0.125028240784
x89 = 0.124936320200
x90 = 0.125071472559
x91 = 0.124965538910

x92 = 0.124958098410	x92 = 0.124958147283
x93 = 0.125122867283	x93 = 0.125122896169
x94 = 0.124844082450	x94 = 0.124843993972
x95 = 0.125098321003	x95 = 0.125098417087
x96 = 0.125050783992	x96 = 0.125050735008
x97 = 0.124769977848	x97 = 0.124769953871
x98 = 0.125331157945	x98 = 0.125331238295
x99 = 0.124746152896	x99 = 0.124746063762
x100 = 0.124970493785	x100 = 0.124970541729
x101 = 0.125415297794	x101 = 0.125415315047
x102 = 0.124315103869	x102 = 0.124315035216
x103 = 0.125615469051	x103 = 0.125615547841
x104 = 0.124905764516	x104 = 0.124905719266
x105 = 0.124283218952	x105 = 0.124283208997
x106 = 0.126377703190	x106 = 0.126377757954
x107 = 0.123571056211	x107 = 0.123570990391
x108 = 0.125507783772	x108 = 0.125507824368
x109 = 0.126168368391	x109 = 0.126168371747
x110 = 0.122319773860	x110 = 0.122319733580
x111 = 0.128214856955	x111 = 0.128214908273
x112 = 0.123323655134	x112 = 0.123323621125
x113 = 0.123215744559	x113 = 0.123215746070
x114 = 0.129960000943	x114 = 0.129960027706
x115 = 0.117926058396	x115 = 0.117926021811
x116 = 0.129767010886	x116 = 0.129767036788
x117 = 0.127684999186	x117 = 0.127684995141
x118 = 0.116698403211	x118 = 0.116698387716
x119 = 0.140353917045	x119 = 0.140353939990
x120 = 0.111997231780	x120 = 0.111997214768
x121 = 0.119578844670	x121 = 0.119578848862
x122 = 0.135207479020	x122 = 0.135207486300
x123 = 0.091962627321	x123 = 0.091962615745
x124 = 0.162311307093	x124 = 0.162311315286
x125 = 0.146794910818	x125 = 0.146794908343
x126 = 0.130930204793	x126 = 0.130930202478
x127 = 0.194276791972	x127 = 0.194276795444

Graficzne tempo zbieżności:



Dla macierzy z naszego zadania złożoność obliczeniowa dla rozkładu Cholesky'ego wynosi $5n^2$. W wypadku metody Gaussa-Seidela mamy złożoność m*5n (m – liczba iteracji, n – wymiar macierzy), natomiast dla zapełnionej niezerowymi wyrazami macierzy metoda ta kosztuje $m*n^2$. Tak samo jest w przypadku metody gradientów sprzężonych.