3 stycznia 2024 r.

Sebastian Pergała
Grupa 3
nr indeksu: 327301

Projekt 2:
Metoda SOR

Temat nr 1.5

1 Opis metody

Metoda SOR (ang. **S**uccessive **O**ver**R**elaxation), nazywana też metodą nadrelaksacji, jest uogólnieniem metody Gaussa-Seidla.

W metodzie tej występuje parametr $\omega \in \mathbb{R}$, zwany parametrem relaksacji.

Algorytm (metoda SOR)

$x^{(0)} = (x_1^{(0)}, \ldots, x_n^{(0)})^T$ – przybliżenie początkowe

for $k = 0, 1, \ldots, n$ (dopóki nie będzie spełniony wybrany warunek stopu)
  for $i = 1, 2, \ldots, n$

$$x_i^{k+1} = (1-\omega)x_i^{(k)} + \omega(b_i - \sum_{j=1, j<i}^{n} a_{ij}x_j^{(k+1)} - \sum_{j=1, j>i}^{n} a_{ij}x_j^{(k)})/a_{ii}$$

  end
end

Zauważmy, że gdy $\omega = 1$, metoda SOR staje się metodą Gaussa-Seidla. Metodę SOR można wyprowadzić zapisując macierz A jako

$$A = L + D + U$$

podobnie, jak miało to miejsce przy metodach Jacobiego i Gaussa-Seidla. Mnożąc powyższe równanie stronami przez $\omega \neq 0$, a następnie dodając stronami macierz D i odpowiednio grupując wyrazy, otrzymamy:

$$\omega A = (D + \omega L) - ((1 - \omega)D - \omega U).$$

Podstawiając powyższą zależność do równania $\omega Ax = b$ (które jest równoważne równaniu $Ax = b$, otrzymamy:

$$(D + \omega L)x - ((1 - \omega)D - \omega U)x = \omega b,$$

a dalej

$$x = (D + \omega L)^{-1}((1 - \omega)D - \omega U)x + \omega(D + \omega L)^{-1}b.$$

Wzór iteracyjny ma zatem postać:

$$x^{k+1} = B_{SOR}x^{(k)} + c_{SOR},$$

gdzie

$$B_{SOR} = (D + \omega L)^{-1}((1 - \omega)D - \omega U)$$

oraz

$$c_{SOR} = \omega(D + \omega L)^{-1}b.$$

2 Opis programu obliczeniowego

Funkcja *SOR* przyjmuje następujące argumenty:
- A - macierz kwadratowa, $A \in \mathbb{R}^{n \times n}$,
- b - wektor, $b \in \mathbb{R}^n$,
- *liczbaIteracji* - maksymalna liczba iteracji, którą wykona algorytm,
- w - parametr relaksacji w metodzie *SOR*
- *dokladnosc* - program kończy działanie, gdy spełniony jest następujący warunek:

$$max_{i=1,2,\ldots,n} \left(|x_i^{(k)} - x_i^{(k-1)}| \right) \leq dokladnosc, \tag{1}$$

gdzie $x_i^{(k)}$ jest i-tą współrzędną wektora wynikowego w k-tej iteracji

i zwraca:
- X - wektor wynikowy, w teorii będący przybliżeniem rozwiązania równania $Ax = b$, $x \in \mathbb{R}^n$,
- *iloscWykonanychIteracji* - ilość wykonanych iteracji.

Funkcja wyrzuca błąd z informacją "Błędna ilość argumentów.", jeśli podana liczba argumentów nie jest równa 5, następnie sprawdza, czy rozmiary macierzy A i wektora b są poprawne. Jeśli tak nie jest, to rzucany jest błąd z komunikatem ofpowiednio: "Macierz A nie jest macierzą kwadratową."i "Nieodpowiedni rozmiar wektora b.".

Po sprawdzeniu poprawności danych algorytm oblicza elementy potrzebne do metody *SOR* według wzorów podanych w opisie metody (sekcja 1.). Jeśli się okaże, że promień spektralny macierzy iteracji jest większy lub równy 1, to funkcja kończy działanie zwracając X jako wektor zerowy i *iloscWykonanychIteracji* jako 0. Dodatkowo na konsoli jest wypisywana informacja "Promień spektralny jest większy lub równy 1.".

Jeśli argument *dokladnosc* jest większy od 0, to jest wykonywana część kodu, która w każdej iteracji sprawdza, czy warunek (1) jest spełniony. Jeśli argument *dokladnosc* nie jest większy od 0, to jest wykonywany kod niezawierający wcześniejszego warunku. To rozdzielenie zostało zaimplementowane, aby podczas działania funkcja wykonała minimalną liczbę operacji. Wszystkie operacje obliczeniowe są zwektoryzowane, zapweniając tym samym efektywne działanie programu.

Funkcje *SORInformacje*, *GSInformacje* i *JInformacje* podają informacje o macierzy iteracji dla metody kolejno: *SOR*, *Gaussa-Seidla* i *Jacobiego*. Przyjmują następujące argumenty:

- A - macierz kwadratowa, $A \in \mathbb{R}^{n \times n}$,
- *dokladnosc* - oczekiwana dokładność wyniku

i zwracają:

- *promienSpektralny* - promień spektralny macierzy iteracji w danej metodzie,
- *iloscPotrzebnychIteracji* - szacowaną jedynie na podstawie promienia spektralnego liczbę itereacji potrzebną do osiągnięcia danej dokładności.

Te algorytmy pełnią rolę pomocniczą w programie, usprawniając pozyskiwanie informacji o danych macierzach w konkretnych metodach. Jeśli wyliczona liczba *iloscPotrzebnychIteracji* jest mniejsza od 0, to *iloscPotrzebnychIteracji* przyjmuje wartość *"Inf"*, co ma sygnalizować, że metoda jest rozbieżna dla danej macierzy A.

Dodatkowo program zawiera też funkcje *przykład1*, *przykład2*, ..., których celem jest zebranie kodu użytego do generowania przykładów obliczeniowych w sekcji 3., porządkując tym samym kod.

Skrypt *main.m* został użyty, do wywoływania funkcji *przykład*$\{i\}$, a skrypt *skryptTestujacy.m* zawiera wyniki testów, na przykładowych macierzach, wykonanych za pomocą funkcji *SOR*, *SORInformacje*, *GSInformacje* i *JInformacje*. Część tych wyników została umieszczona w sekcji 4, dotyczącej analizy wyników.

3 Przykłady obliczeniowe

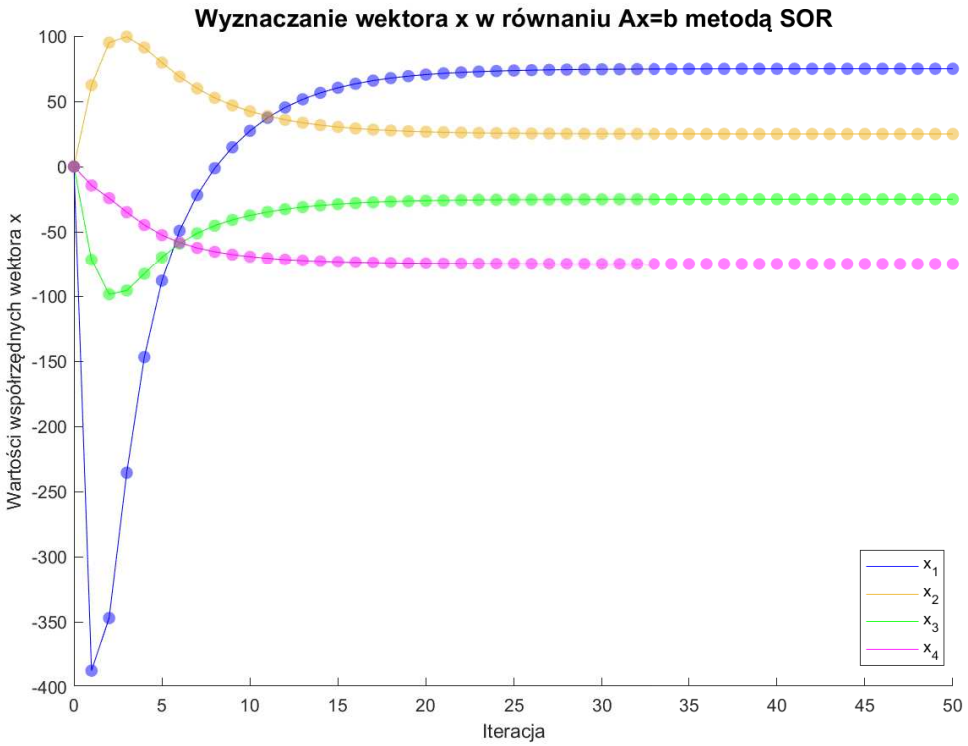
Przykład 1.

Metoda rozwiązywania układów równań liniowych SOR jest motodą iteracyjną. Przy konkretnych założeniach wynikowy wektor x w teorii jest coraz bliższy dokładnemu wynikowi równania $Ax = b$. Poniższy wykres ilustruje działanie metody SOR dla danych

$$A = \begin{pmatrix} 0.5 & 2 & 4 & 5 \\ 4 & 15 & 2 & 4 \\ 5 & 4 & -17 & 2 \\ 0.5 & 0.4 & 0.3 & 5 \end{pmatrix}, \quad b = \begin{pmatrix} -387.5 \\ 325 \\ 750 \\ -335 \end{pmatrix}, \quad \omega = 0.5.$$

Na osi x jest numer iteracji, a na osi y znajdują się wartości współrzędnych wektora $x = (x_1, x_2, x_3, x_4)^T$ W tym przypadku metoda SOR jest zbieżna do prawidłowego wyniku, czyli

$$x = \begin{pmatrix} 75 \\ 25 \\ -25 \\ -75 \end{pmatrix}$$

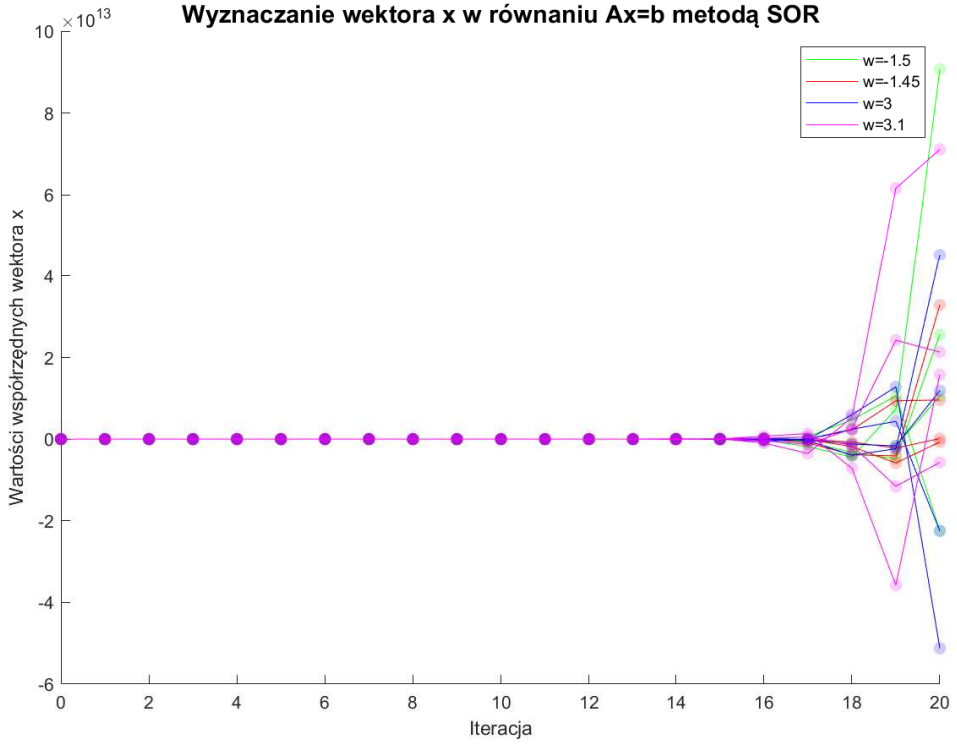


Rysunek 1: Przykład zbieżności metody *SOR*

Przykład 2.

Jeśli $\omega \notin (0, 2)$, to metoda *SOR* nie jest zbieżna (*wniosek 6.1* z wykładu). Poniżej jest wykres ilustrujący ten fakt dla

$$A = \begin{pmatrix} 0.5 & 2 & 4 & 5 \\ 4 & 15 & 2 & 4 \\ 5 & 4 & -17 & 2 \\ 0.5 & 0.4 & 0.3 & 5 \end{pmatrix}, \quad b = \begin{pmatrix} -387.5 \\ 325 \\ 750 \\ -335 \end{pmatrix}, \quad \text{a poprawnym wynikiem jest} x = \begin{pmatrix} 75 \\ 25 \\ -25 \\ -75 \end{pmatrix}.$$



Rysunek 2: Przykład rozbieżności metody *SOR*

Dodatkowo, jeśli $A \in \mathbb{R}^{n \times n}$ jest macierzą symetryczną i dodatnio określoną, a $b \in \mathbb{R}^n$, to metoda *SOR* jest zbieżna globalnie wtedy i tylko wtedy, gdy $0 < \omega < 2$ (*twierdzenie 6.8* z wykładu).

Jeśli promień spektralny macierzy iteracji jest większy od 1, to metoda *SOR* nie jest zbieżna. W pokazanym przypadku promienie spektralne mają następujące wartości:

Tabela 1: Wartości promienia spektralnego macierzy iteracji dla danych parametrów relaksacji ω

ω	-1.5	-1.45	3	3.1
Wartość promienia spektralnego	3.3173	3.6080	3.5527	3.7467

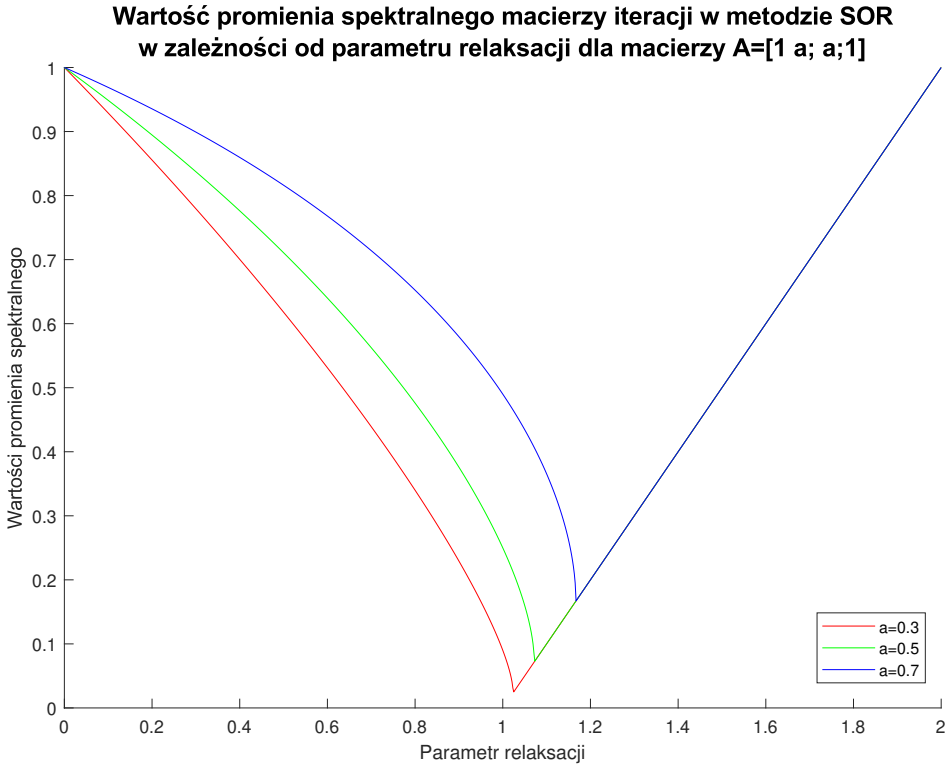
Przykład 3.

Promień spektralny macierzy iteracji B_{SOR} dla ustalonej macierzy A w rozważanym równaniu $Ax = b$ zależy od promienia spektralnego macierzy iteracyjnej w metodzie *Jacobiego* ($B_J = D : (D - A)$, gdzie $D = diag(A)$) i współczynnika relaksacji ω. Niech μ będzie promieniem spektralnym macierzy iteracji B_J w metodzie Jacobiego. Wtedy wartość promienia spektralnego macierzy iteracji metody *SOR* można wyrazić następująco:

$$\rho(B_\omega) = \begin{cases} \frac{1}{4}(\omega\mu + \sqrt{\omega^2\mu^2 - 4(\omega - 1)})^2 & \text{dla} \quad 0 < \omega < \omega_{opt}, \\ \omega - 1 & \text{dla} \quad \omega_{opt} < \omega < 2, \end{cases} \tag{2}$$

gdzie ω_{opt} określone jest wzorem:

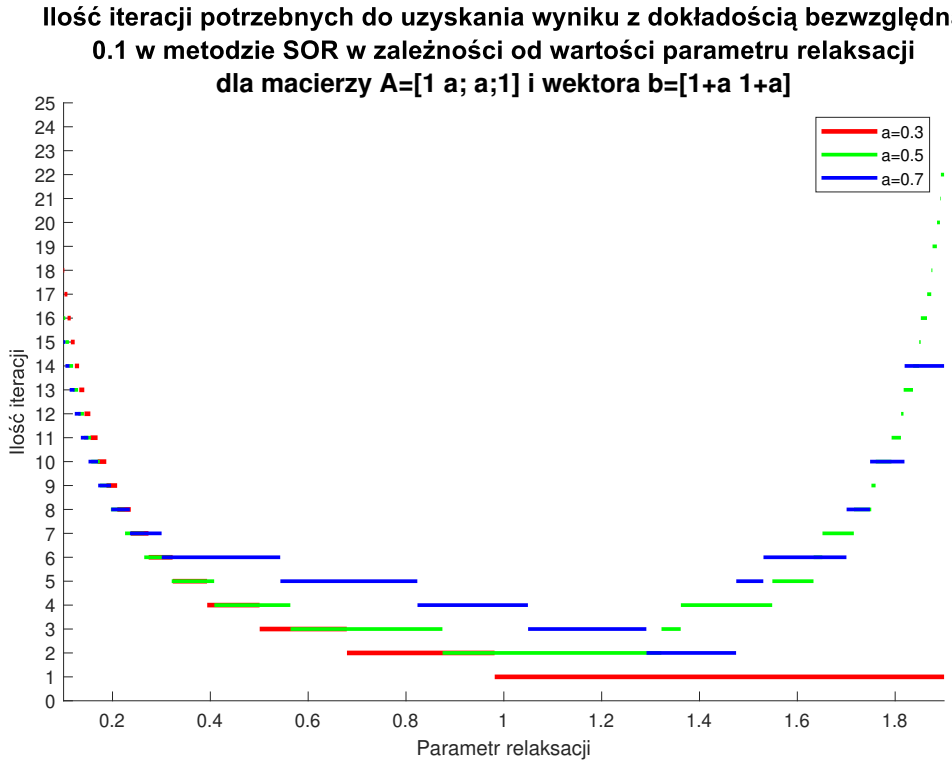
$$\omega_{opt} := 1 + \left(\frac{\mu}{1 + \sqrt{1 - \mu^2}} \right)^2 \tag{3}$$



Rysunek 3: Wartość promienia spektralnego w zależności od parametru relaksacji

Przykład 4.

Szybkość zbieżności metody *SOR* zależy od wartości promienia spektralnego macierzy iteracji B_{SOR}, która z kolei zależy od wartości parametru relaksacji ω (jak było pokazanie w przykładzie 3.). Im promień spektralny macierzy iteracji ma mniejszą wartośc, tym szybkość zbiegania do prawidłowego wyniku jest szybsza (*uwaga 6.7* z wykładu). Według wzoru (2) promień spektralny jest najmniejszy dla parametru relaksacji równego ω_{opt} (wzór (3)). Poniższy wykres pokazuje ilość iteracji, które wykonał program, aby $max_{i=1,2,\ldots,n}(|x - x^{(k)}|) \leq tol$, gdzie $x_i \in \mathbb{R}^n$ jest *i-tą* współrzędną wektora x, będącego dokładnym wynikiem równania $Ax = b$, $x_i^{(k)} \in \mathbb{R}^n$ jest *i-tą* współrzędną wektora $x^{(k)}$, będącego wynikiem przybliżonym metodą *SOR* w *k-tej* iteracji, *tol* to liczba ustalana przez użytkownika, a macierz $A \in \mathbb{R}^{n\times n}$ i wektor $b \in \mathbb{R}^n$ są dane.

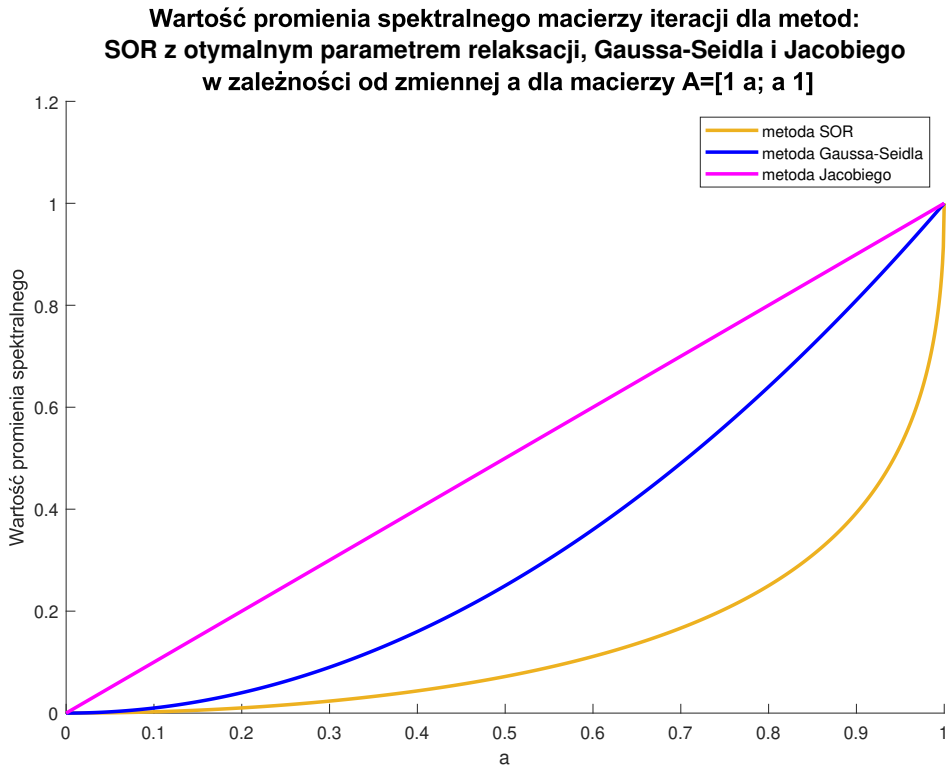


Rysunek 4: Liczba iteracji potrzebnych do uzyskania wyniku z dokładnością bezwzględną 0.1

Przykład 5.

Porównanie metody SOR z GS i Jakobego dla w optymalnego.

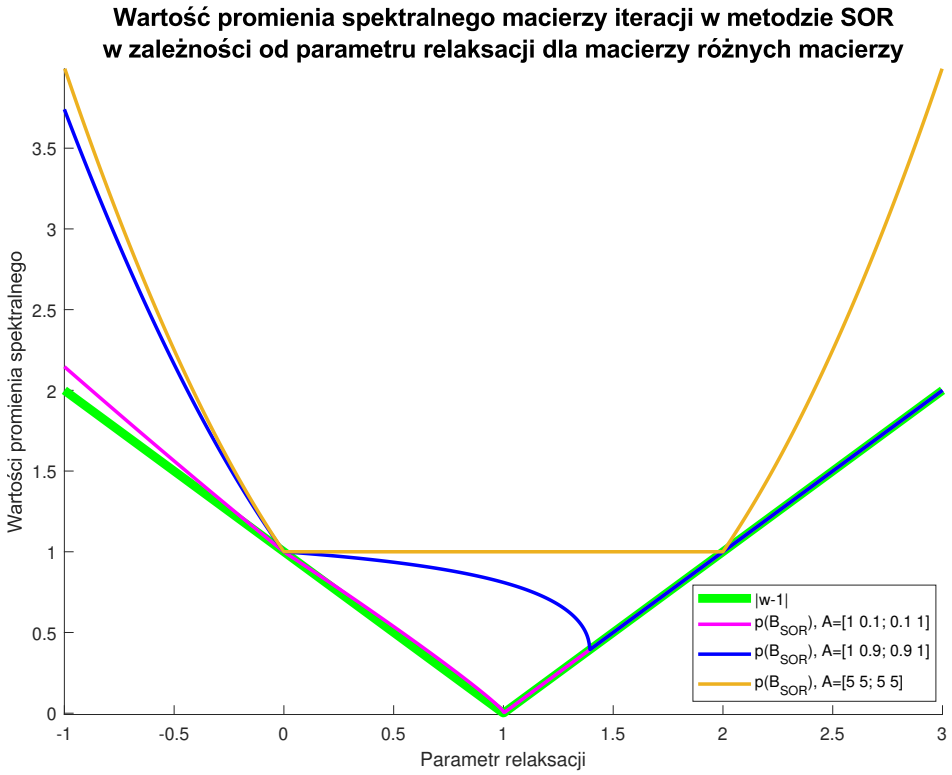
Dla parametru relaksacji $\omega = 1$ metoda *SOR* jest równoważna metodzie *Gaussa-Seidla*. Wartość promienia spektralnego macierzy iteracji w metodzie *Gaussa-Seidla* (B_{GS}) można obliczyć korzystając ze wozru (2). Wtedy $\rho(B_\omega) = \rho(B_1) = \mu^2$, gdzie μ jest promieniem spektralnym macierzy iteracji w metodzie *Jacobiego*. Dla parametru relaksacji ω_{opt} (wzór (3)), promień spektralny, zgodnie ze wzorem (2), jest równy $\rho(C_{\omega_{opt}}) = \frac{1-\sqrt{1-\mu^2}}{1+\sqrt{1-\mu^2}} = \frac{\mu^2}{4} + O(\mu^3)$, co oznacza, że metoda *SOR* jest około 4 razy szybciej zbieżna od metody *Gaussa-Seidla*, przy wyborze optymalnego parametru relaksacji. Ta własność jest gwarantowana, jeśli $A \in \mathbb{R}^{n\times n}$ jest macierzą symetryczną i dodatnio określoną (pokazane w przykładzie 2.).



Rysunek 5: Warotść promienia spektralnego dla metody *SOR*, *Gaussa-Seidla* i *Jacobiego* dla danych macierzy

Przykład 6.

Promień spektralny $\rho(B_{SOR})$ macierzy iteracji w metodzie SOR spełnia zależność: $\rho(B_{SOR}) \geq |\omega - 1|$ (*twierdzenie 6.7* z wykładu).



Rysunek 6: Zilustrowanie faktu: $\rho(B_{SOR}) \geq |\omega - 1|$

4 Analiza wyników

Metoda *SOR* jest iteracyjną metodą wyznaczania rozwiązań układów równań liniowych postaci $Ax = b$, gdzie $A \in \mathbb{R}^{n\times n}$ i $b \in \mathbb{R}^n$ są dane. Metoda ta faktycznie jest w stanie przybliżyć dokładny wynik, ale nie zawsze jest zbieżna. Jeśli $A \in \mathbb{R}^{n\times n}$ jest macierzą symetryczną i dodatnio określoną, a $b \in \mathbb{R}^n$, to metoda *SOR* jest zbieżna globalnie wtedy i tylko wtedy, gdy $0 < \omega < 2$. Dla macierzy symetrycznych i dodatnio określonych zachodzi wzór (2):

$$\rho(B_\omega) = \begin{cases} \frac{1}{4}(\omega\mu + \sqrt{\omega^2\mu^2 - 4(\omega-1)})^2 & \text{dla} \quad 0 < \omega < \omega_{opt}, \\ \omega - 1 & \text{dla} \quad \omega_{opt} < \omega < 2, \end{cases}$$

gdzie ω_{opt} określone jest wzorem:

$$\omega_{opt} := 1 + \left(\frac{\mu}{1 + \sqrt{1-\mu^2}}\right)^2.$$

Szybkość zbieżności metody zależy od promienia spektralnego macierzy iteracji - im wartość jest mniejsza, tym metoda szybciej zbiega do prawidłowego wyniku. Jeśli promień spektralny jest większy od 1, to metoda jest rozbieżna. Metoda *SOR* jest uogólnieniem metody *Gaussa-Seidla*, a dokładniej dla $\omega = 1$ metoda *Gaussa-Seidla* jest równoważna metodzie *SOR*. Z reguły macierz iteracji w metodzie *Jacobiego* ma większy promień spektralny niż w metodzie *Gaussa-Seidla*; $\rho(B_{GS}) = \rho(B_J)^2$. Z kolei promień spektralny macierzy iteracji w metodzie nadrelaksacji jest około cztery razy mniejszy, niż w metodzie *Gaussa-Seidla*. Prawdziwy jest również fakt, że promień spektralny $\rho(B_{SOR})$ macierzy iteracji w metodzie *SOR* spełnia zależność: $\rho(B_{SOR}) \geq |\omega - 1|$.

Wyniki testów dla wybranych macierzy:

$$A_1 = \begin{pmatrix} 1 & 0.8 \\ 0.8 & 1 \end{pmatrix}, \quad b_1 = \begin{pmatrix} 1.8 \\ 1.8 \end{pmatrix}, \quad x_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Macierz jest symetryczna i dodatnio określona. Wybierając optymalny parametr relaksacji (ω_{opt}=1.25) metoda *SOR* uzyskuje szybszą zbieżność niż metoda *Gaussa-Seidla* ($\omega = 1$).

Tabela 2: Błędy metody *SOR* dla $A_1, b_1, \omega = 1$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	1.0245	0.72443	0.8	0.8
2	0.65568	0.46364	0.512	0.512
3	0.41964	0.29673	0.32768	0.32768
4	0.26857	0.18991	0.20972	0.20972
5	0.17188	0.12154	0.13422	0.13422
6	0.11	0.077785	0.085899	0.085899
7	0.070403	0.049783	0.054976	0.054976
8	0.045058	0.031861	0.035184	0.035184
9	0.028837	0.020391	0.022518	0.022518
10	0.018456	0.01305	0.014412	0.014412

Tabela 3: Błędy metody *SOR* dla $A_1, b_1, \omega = 1.25$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	1.6008	1.1319	1.25	1.25
2	0.8149	0.57622	0.6875	0.6875
3	0.30817	0.21791	0.26563	0.26563
4	0.1032	0.072975	0.089844	0.089844
5	0.032345	0.022871	0.02832	0.02832
6	0.0097228	0.0068751	0.0085449	0.0085449
7	0.0028399	0.0020081	0.0025024	0.0025024
8	0.00081231	0.00057439	0.00071716	0.00071716
9	0.00022866	0.00016169	0.00020218	0.00020218
10	6.3561e-05	4.4944e-05	5.6267e-05	5.6267e-05

$$A_2 = \begin{pmatrix} 0.5 & 2 & 4 & 5 \\ 4 & 15 & 2 & 4 \\ 5 & 4 & -17 & 2 \\ 0.5 & 0.4 & 0.3 & 5 \end{pmatrix}, \quad b_2 = \begin{pmatrix} -387.5 \\ 325 \\ 750 \\ -335 \end{pmatrix}, \quad x_2 = \begin{pmatrix} 75 \\ 25 \\ -25 \\ -75 \end{pmatrix}$$

Dobierając nieodpowiedni parametr relaksacji ($\omega = 1.5$) metoda *SOR* zbiega wolniej niż *Gaussa-Seidla*.

Tabela 4: Błędy metody *SOR* dla $A_2, b_2, \omega = 1$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	898.71	8.0383	850	11.333
2	74.115	0.6629	70	0.93333
3	107.6	0.96244	101.34	1.3512
4	16.539	0.14793	15.616	0.20822
5	13.936	0.12465	13.13	0.17507
6	2.9935	0.026775	2.826	0.03768
7	1.8702	0.016728	1.7627	0.023503
8	0.49342	0.0044133	0.46568	0.0062091
9	0.25786	0.0023064	0.2431	0.0032413
10	0.077574	0.00069384	0.073197	0.00097596

Tabela 5: Błędy metody *SOR* dla $A_2, b_2, \omega = 1.5$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	1383.3	12.373	1237.5	16.5
2	656.46	5.8715	442.93	5.9057
3	474.85	4.2472	471.46	6.2862
4	581.85	5.2043	507.64	6.7685
5	144.13	1.2891	109.54	1.4605
6	264.06	2.3619	254.46	3.3928
7	194.55	1.7401	158.9	2.1187
8	52.934	0.47346	46.792	0.62389
9	120.05	1.0738	110.49	1.4732
10	53.48	0.47834	36.395	0.48526

$$A_3 = \begin{pmatrix} 4 & -1 & -6 & 0 \\ -5 & -4 & 10 & 8 \\ 0 & 9 & 4 & -2 \\ 1 & 0 & -7 & 5 \end{pmatrix}, \quad b_3 = \begin{pmatrix} 2 \\ 21 \\ -12 \\ -6 \end{pmatrix}, \quad x_3 = \begin{pmatrix} 3 \\ -2 \\ 2 \\ 1 \end{pmatrix}$$

W tym przypadku, po dobraniu odpowiedniego parametru relaksacji ($\omega = 0.5$), metoda *SOR* jest zbieżna, choć metoda *Gaussa-Seidla* nie jest. Stałe wartości błędów wynikają ze sposobu implementacji metody (opisanego w sekcji 2.) - algorytm przerywa działanie zwracając wektor zerowy, gdy promień spektralny macierzy iteracji jest większy lub równy 1 (metoda nie jest zbieżna).

Tabela 6: Błędy metody *SOR* dla $A_3, b_3, \omega = 1$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	4.2426	1	3	1
2	4.2426	1	3	1
⋮	⋮	⋮	⋮	⋮
50	4.2426	1	3	1

Tabela 7: Błędy metody *SOR* dla $A_3, b_3, \omega = 0.5$

Iteracja	Błąd bezwzględny norma 2	Błąd względny norma 2	Błąd bezwzględny norma ∞	Błąd względny norma ∞
1	2.9233	0.68902	2.75	0.91667
2	1.7705	0.41732	1.751	0.58366
3	1.0943	0.25794	0.92952	0.30984
4	0.76259	0.17974	0.73061	0.24354
5	0.47729	0.1125	0.47127	0.15709
6	0.2949	0.06951	0.26938	0.089793
7	0.20278	0.047796	0.19552	0.065172
8	0.12924	0.030462	0.12707	0.042358
9	0.080805	0.019046	0.076136	0.025379
10	0.054621	0.012874	0.05278	0.017593
⋮	⋮	⋮	⋮	⋮
50	1.6707e-09	3.9378e-10	1.6187e-09	5.3958e-10

5 Przykłady zastosowania zaimplementowanej metody w praktycznym użyciu

Czas jest cennym surowcem. Jeśli chodzi o rozwiązywanie układów równań, czasem lepiej jest uzyskać szybciej przybliżone rozwiązanie, niż wolniej dokładne. Standardowa metoda uzyskiwania dokładnych rozwiązań, np. eliminacja Gaussa, wymaga ilości operacji rzędu n^3, co staje się czasochłonne przy dużych wartościach n. Metoda *SOR* natomiast, chociaż dostarcza jedynie przybliżenie, to robi to znacznie szybciej niż eliminacja Gaussa.

Metoda *SOR* to jedna z najważniejszych metod rozwiązywania dużych układów równań liniowych. Znajduje zastosowanie w

- programowaniu matematycznym,

- teorii sprężystości,

- przepływach chemicznie reagujących mieszanin ściśliwych - badanie istnienia słabych rozwiązań dla układów równań Naviera-Stokesa uzupełnionych równaniami reakcji-dyfuzji poszczególnych składników,

- przetwarzaniu obrazów - rozwiązywanie równania Poissona w przetwarzaniu obrazów w celu wygładzania i redukcji szumów na obrazach,

- symulacja obwodów - rozwiązywanie równań węzłowych w symulacji obwodów w celu określenia napięcia i natężenia prądu w każdym węźle obwodu,

- robotyce - problem kinematyki odwrotnej dla ramienia robota,

- uczeniu maszynowym itp.

Przykłady zastosowań *SOR* w dynamice obejmują: badanie stałego przewodnictwa ciepła i dynamiki płynów - rozwiązywanie równań opisujących przewodzenie ciepła i przepływ cieczy w różnych zastosowaniach inżynieryjnych, takich jak symulacja wymienników ciepła, przepływ cieczy w ośrodkach porowatych, swobodna konwekcja, turbulencje i przepływy warstw granicznych.

Z tych powodu metoda *SOR* jest istotna dla celów praktycznych i badawczych.