Wstęp do metod numerycznych

Metody iteracyjne

Algebraiczna metoda gradientów sprzężonych

P. F. Góra

http://th-www.if.uj.edu.pl/zfs/gora/

2017

Metody iteracyjne

Rozwiązanie układu równań liniowych, uzyskane za pomocą którejś z dotąd poznanych metod, byłoby dokładne (ścisłe), gdyby nie błędy zaokrąglenia (które, dodajmy, dla układów źle uwarunkowanych mogą być *znaczne*). Dlatego metody te nazywa się *metodami dokładnymi*.

W metodach iteracyjnych rozwiązanie dokładne otrzymuje się, teoretycznie, w granicy nieskończenie wielu kroków — w praktyce liczymy na to, że po skończonej (i niewielkiej) liczbie kroków zbliżymy się do wyniku ścisłego w granicach błędu zaokrąglenia.

Rozpatrzmy układ równań:

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 = b_1 \quad (1a)$$
$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 = b_2 \quad (1b)$$
$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 = b_3 \quad (1c)$$

Przepiszmy ten układ w postaci

$$x_1 = (b_1 - a_{12}x_2 - a_{13}x_3)/a_{11} \quad (2a)$$
$$x_2 = (b_2 - a_{21}x_1 - a_{23}x_3)/a_{22} \quad (2b)$$
$$x_3 = (b_3 - a_{31}x_1 - a_{32}x_2)/a_{33} \quad (2c)$$

Gdyby po prawej stronie (2) były “stare” elementy x_j, a po lewej “nowe”, dostalibyśmy metodę iteracyjną

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} \right) \Big/ a_{ii} \qquad (3)$$

Górny indeks $x^{(k)}$ oznacza, ze jest to przybliżenie w k-tym kroku. Jest to tak zwana *metoda Jacobiego*.

Zauważmy, że w metodzie (3) nie wykorzystuje się najnowszych przybliżeń: Powiedzmy, obliczając $x_2^{(k+1)}$ korzystamy z $x_1^{(k)}$, mimo iż znane jest już wówczas $x_1^{(k+1)}$. **Za to metodę tę łatwo można zrównoleglić.** Sugeruje to następujące ulepszenie:

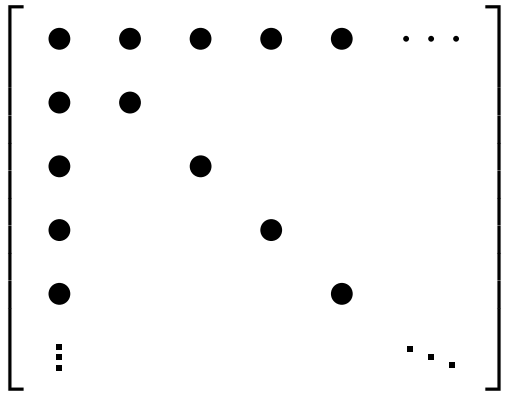
$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{N} a_{ij} x_j^{(k)} \right) \Big/ a_{ii} \qquad (4)$$

Jest to tak zwana metoda Gaussa-Seidela.

Jeżeli macierz $\mathbf{A} = \{a_{ij}\}$ jest rzadka, obie te metody iteracyjne będą efektywne *tylko i wyłącznie* wówczas, gdy we wzorach (3), (4) uwzględni się ich strukturę, to jest uniknie redundantnych mnożeń przez zera.

Powtórzmy: Dla numerycznej efektywności metod iteracyjnych jest **niesłychanie** ważne, aby metodę zaprogramować w ten sposób, aby uwzględniać strukturę macierzy rzadkiej.

Przykład: Niech macierz $\mathbf{A} \in \mathbb{R}^{N \times N}$ ma strukturę



(5)

Metoda Gaussa-Seidela dla macierzy o strukturze (5) ma postać

$$\begin{aligned}
x_1^{(k+1)} &= \left(b_1 - \sum_{j=2}^{N} a_{1j} x_j^{(k)}\right) / a_{11} \qquad (6)\\
x_2^{(k+1)} &= \left(b_2 - a_{21} x_1^{(k+1)}\right) / a_{22}\\
x_3^{(k+1)} &= \left(b_3 - a_{31} x_1^{(k+1)}\right) / a_{33}
\end{aligned}$$

....

$$x_N^{(k+1)} = \left(b_N - a_{N1} x_1^{(k+1)}\right) / a_{NN}$$

Widać, że jedek krok (*sweep*) algorytmu (6) odbywa się w czasie proporcjonalnym do N.

Trochę teorii

Metody Jacobiego i Gaussa-Seidela należą do ogólnej kategorii

$$\mathbf{M}\mathbf{x}^{(k+1)} = \mathbf{N}\mathbf{x}^{(k)} + \mathbf{b} \tag{7}$$

gdzie $\mathbf{A} = \mathbf{M} - \mathbf{N}$ jest *podziałem* (*splitting*) macierzy. Dla metody Jacobiego $\mathbf{M} = \mathbf{D}$ (część diagonalna), $\mathbf{N} = -(\mathbf{L} + \mathbf{U})$ (części pod- i ponaddiagonalne, bez przekątnej). Dla metody Gaussa-Seidela $\mathbf{M} = \mathbf{D} + \mathbf{L}$, $\mathbf{N} = -\mathbf{U}$. Rozwiązanie równania $\mathbf{A}\mathbf{x} = \mathbf{b}$ jest punktem stałym iteracji (7).

Twierdzenie 1. *Iteracja* (7) *jest zbieżna jeśli* $\det \mathbf{M} \neq 0$ *oraz* $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$*, gdzie* $\rho(\bullet)$ *oznacza promień spektralny macierzy..*

Dowód. Przy tych założeniach iteracja (7) jest odwzorowaniem zwężającym. □

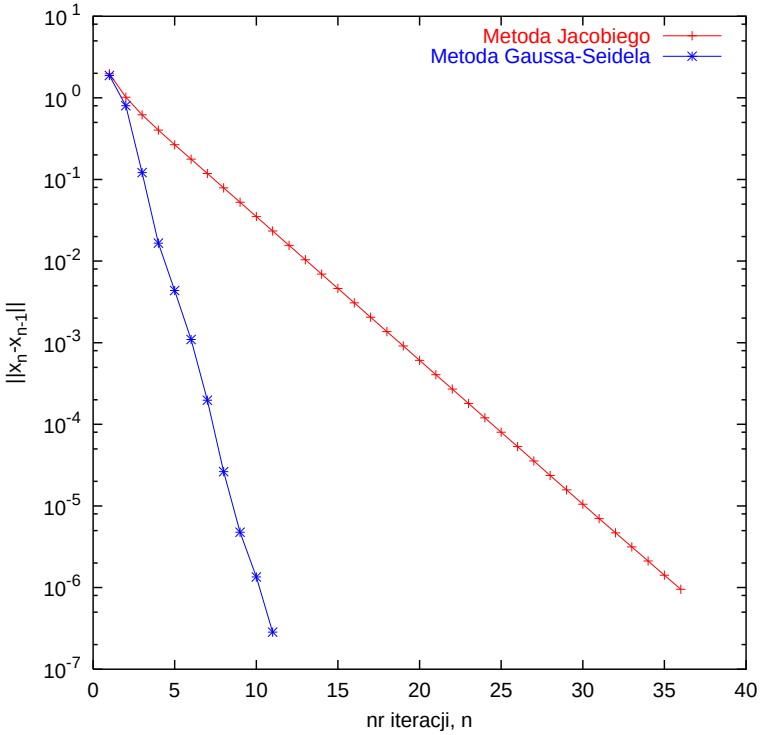
Twierdzenie 2. *Metoda Jacobiego jest zbieżna, jeśli macierz* $\mathbf{A}$ *jest silnie diagonalnie dominująca, to znaczy jeśli wartości bezwzględne elementów na głównej przekątnej są większe od sumy wartości bezwzględnych pozostałych elementów w danym wierszu.*

Twierdzenie 3. *Metoda Gaussa-Seidela jest zbieżna, jeśli macierz* $\mathbf{A}$ *jest symetryczna i dodatnio określona.*

Przykład

Rozwiązujemy układ równań:

$$\begin{aligned} 3x &+ y &+ z &= 1 \\ x &+ 3y &+ z &= 1 \\ x &+ y &+ 3z &= 1 \end{aligned}$$



Inny przykład

Dla macierzy o wymiarach 128×128

$$\begin{bmatrix} 128 & 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & & & & \\ 1 & & 2 & & & \\ 1 & & & 2 & & \\ \vdots & & & & \ddots & \\ 1 & & & & & 2 \end{bmatrix} \tag{8}$$

(niezaznaczone elementy są zerami)

zbieżność z dokładnością do 10^{-12} w metodzie Gaussa-Seidela, według algorytmu (6), uzyskuje się w ~ 42 iteracjach.

Metoda gradientów sprzężonych — motywacja

Rozważmy funcję $f : \mathbb{R}^N \to \mathbb{R}$

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x} + c\,, \tag{9}$$

gdzie $\mathbf{x}, \mathbf{b} \in \mathbb{R}^N$, $c \in \mathbb{R}$, $\mathbf{A} = \mathbf{A}^T \in \mathbb{R}^{N\times N}$ jest *symetryczna i dodatnio określona*. Przy tych założeniach, funkcja (9) ma dokładnie jedno minimum, będące zarazem minimum globalnym. Szukanie minimów dodatnio określonych form kwadratowych jest (względnie) łatwe i z praktycznego punktu widzenia ważne. Minimum to leży w punkcie spełniającym

$$\nabla f = 0\,. \tag{10}$$

Obliczmy

$$\frac{\partial f}{\partial x_i} = \frac{1}{2}\frac{\partial}{\partial x_i}\sum_{j,k} A_{jk}x_j x_k - \frac{\partial}{\partial x_i}\sum_j b_j x_j + \underbrace{\frac{\partial c}{\partial x_i}}_{0}$$

$$= \frac{1}{2}\sum_{j,k} A_{jk}\left(\underbrace{\frac{\partial x_j}{\partial x_i}}_{\delta_{ij}} x_k + x_j \underbrace{\frac{\partial x_k}{\partial x_i}}_{\delta_{ik}}\right) - \sum_j b_j \underbrace{\frac{\partial x_j}{\partial x_i}}_{\delta_{ij}}$$

$$= \frac{1}{2}\sum_k A_{ik}x_k + \frac{1}{2}\sum_j A_{ji}x_j - b_i = \frac{1}{2}\sum_k A_{ik}x_k + \frac{1}{2}\sum_j A_{ij}x_j - b_i$$

$$= (\mathbf{Ax} - \mathbf{b})_i \ . \tag{11}$$

Widzimy zatem, że funkcja (9) osiąga minimum w punkcie, w którym zachodzi

$$\mathbf{A}\mathbf{x} - \mathbf{b} = 0 \Leftrightarrow \mathbf{A}\mathbf{x} = \mathbf{b}. \tag{12}$$

Rozwiązywanie układu równań liniowych (12) z macierzą symetryczną, dodatnio określoną jest równoważne poszukiwaniu minimum dodatnio określonej formy kwadratowej.

Przypuśćmy, że macierz $\mathbf{A}$ jest przy tym *rzadka* i duża (lub co najmniej średnio-duża). Wówczas metoda gradientów sprzężonych jest godną uwagi metodą rozwiązywania (12)

Metoda gradientów sprzężonych, *Conjugate Gradients*, CG

$\mathbf{A} \in \mathbb{R}^{N \times N}$ symetryczna, dodatnio określona, $\mathbf{x}_1$ — początkowe przybliżenie rozwiązania równania (12), $0 < \varepsilon \ll 1$.

$$
\begin{aligned}
&\mathbf{r}_1 = \mathbf{b} - \mathbf{A}\mathbf{x}_1,\ \mathbf{p}_1 = \mathbf{r}_1 \\
&\textbf{while } \|\mathbf{r}_k\| > \varepsilon \\
&\qquad \alpha_k = \frac{\mathbf{r}_k^T \mathbf{r}_k}{\mathbf{p}_k^T \mathbf{A}\mathbf{p}_k} \\
&\qquad \mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k \\
&\qquad \beta_k = \frac{\mathbf{r}_{k+1}^T \mathbf{r}_{k+1}}{\mathbf{r}_k^T \mathbf{r}_k} \\
&\qquad \mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k \\
&\qquad \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
&\textbf{end}
\end{aligned}
\tag{13}
$$

Wówczas zachodzą twierdzenia:

Twierdzenie 4. *Ciągi wektorów* $\{\mathbf{r}_k\}$, $\{\mathbf{p}_k\}$ *spełniają następujące zależności:*

$$\mathbf{r}_i^T \mathbf{r}_j = 0\,, \quad i > j\,, \tag{14a}$$

$$\mathbf{r}_i^T \mathbf{p}_j = 0\,, \quad i > j\,, \tag{14b}$$

$$\mathbf{p}_i^T \mathbf{A} \mathbf{p}_j = 0\,, \quad i > j\,. \tag{14c}$$

Twierdzenie 5. *Jeżeli* $\mathbf{r}_M = 0$, *to* $\mathbf{x}_M$ *jest ścisłym rozwiązaniem równania* (12).

Dowód. Oba (sic!) dowody przebiegają indukcyjnie. □

Ciąg $\{\mathbf{x}_k\}$ jest w gruncie rzeczy "pomocniczy", nie bierze udziału w iteracjach, służy tylko do konstruowania kolejnych przybliżeń rozwiązania.

Istotą algorytmu jest konstruowanie dwu ciągów wektorów spełniających zależności (14). Wektory $\{\mathbf{r}_k\}$ są wzajemnie prostopadłe, a zatem *w arytmetyce dokładnej* $\mathbf{r}_{N+1} = 0$, wobec czego $\mathbf{x}_{N+1}$ jest poszukiwanym ścisłym rozwiązaniem.

Zauważmy, że ponieważ $\mathbf{A}$ jest symetryczna, dodatnio określona, warunek (14c) oznacza, że wektory $\{\mathbf{p}_k\}$ są wzajemnie prostopadłe w metryce zadanej przez $\mathbf{A}$. Ten właśnie warunek nazywa się warunkiem *sprzężenia względem* $\mathbf{A}$, co daje nazwę całej metodzie.

Ten wariant metody gradientów sprzężonych nazywamy "algebraicznym", gdyż przy założeniu, że *znamy* macierz $\mathbf{A}$ oraz wektor $\mathbf{x}_1$, możemy skonstruować ciągi $\{\mathbf{r}_k, \mathbf{p}_k, \mathbf{x}_k\}$ metodami algebraicznymi.

W przyszłości poznamy wariant metody gradientów sprzężonych, w którym wszystkich kroków nie uda się w ten sposób wykonać.

Koszt metody

W arytmetyce dokładnej metoda zbiega się po N krokach, zatem jej koszt wynosi $O(N \cdot \text{koszt\_jednego\_kroku})$. Koszt jednego kroku zdominowany jest przez obliczanie iloczynu $\mathbf{A}\mathbf{p}_k$. Jeśli macierz $\mathbf{A}$ jest pełna, jest to $O(N^2)$, a zatem całkowity koszt wynosi $O(N^3)$, czyli tyle, ile dla metod dokładnych. Jeżeli jednak $\mathbf{A}$ jest rzadka, koszt obliczania iloczynu jest mniejszy, **o ile obliczenie to jest odpowiednio zaprogramowane**. Jeśli $\mathbf{A}$ jest pasmowa o szerokości pasma $M \ll N$, całkowity koszt wynosi $O(M \cdot N^2)$.

Przykład

Dla macierzy o wymiarach 128×128

$$\begin{bmatrix} 128 & 1 & 1 & 1 & \dots & 1 \\ 1 & 2 & & & & \\ 1 & & 2 & & & \\ 1 & & & 2 & & \\ \vdots & & & & \ddots & \\ 1 & & & & & 2 \end{bmatrix} \tag{15}$$

(niezaznaczone elementy są zerami)

zbieżność z dokładnością do 10^{-12} w algebraicznej metodzie gradientów sprzężonych uzyskuje się po 4 (*sic!*) iteracjach (w metodzie Gaussa-Seidela były to 42 iteracje; w obu wypadkach znacznie poniżej rozmiaru macierzy).

Problem!

W arytmetyce o skończonej dokładności kolejne generowane wektory nie są *ściśle* ortogonalne do swoich poprzedników — na skutek akumulującego się błędu zaokrąglenia rzut na poprzednie wektory może stać się z czasem znaczny. Powoduje to istotne spowolnienie metody.

Twierdzenie 6. *Jeżeli* $\mathbf{x}$ *jest ścisłym rozwiązaniem równania* (12)*,* $\mathbf{x}_k$ *są generowane w metodzie gradientów sprzężonych, zachodzi*

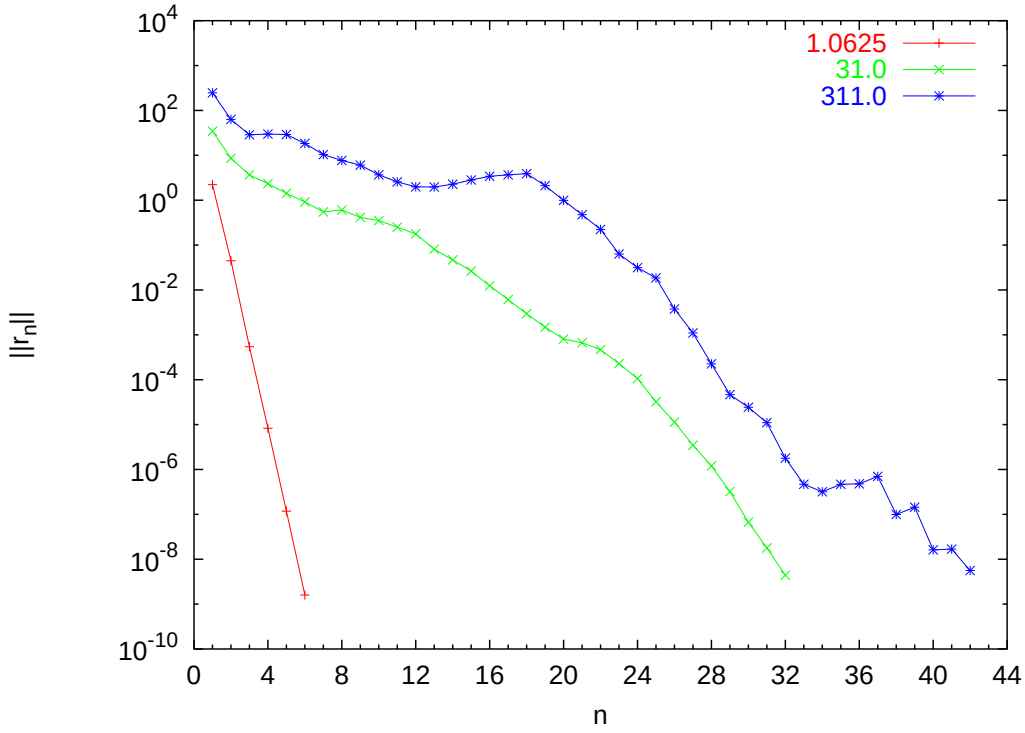
$$\|\mathbf{x}-\mathbf{x}_k\| \leqslant 2\|\mathbf{x}-\mathbf{x}_1\| \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{k-1}, \tag{16}$$

gdzie κ *jest współczynnikiem uwarunkowania macierzy* $\mathbf{A}$*.*

Jeżeli $\kappa \gg 1$*, zbieżność może być bardzo wolna.*

Przykład

Rozwiązujemy układy równań z *małymi* (32×32) macierzami symetrycznymi, rzeczywistymi, dodatnio określonymi, o różnych współczynnikach uwarunkowania. Poniższy rysunek pokazuje normy kolejnych wektorów $\mathbf{r}_n$. Iteracje zatrzymywano, gdy $\|\mathbf{r}_n\| \leqslant 10^{-8}$. W arytmetyce dokładnej $\|\mathbf{r}_{n>32}\| \equiv 0$.



“Prewarunkowana” (*preconditioned*) metoda gradientów sprzężonych

Spróbujmy przyspieszyć zbieżność odpowiednio modyfikując równanie (12) i algorytm (13), jednak tak, aby

- nie zmienić rozwiązania,
- macierz zmodyfikowanego układu pozostała symetryczna i dodatnio określona, aby można było zastosować metodę gradientów sprzężonych,
- macierz zmodyfikowanego układu pozostała rzadka, aby jeden krok iteracji był numerycznie tani,
- macierz zmodyfikowanego układu miała niski współczynnik uwarunkowania.

Czy to się w ogóle da zrobić? Okazuje się, że tak!

Postępujemy następująco: Niech $\mathbf{C} \in \mathbb{R}^{N\times N}$ będzie odwracalną macierzą symetryczną, rzeczywistą, dodatnio określoną. Wówczas $\widetilde{\mathbf{A}} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1}$ też jest symetryczna, rzeczywista, dodatnio określona.

$$\mathbf{C}^{-1}\mathbf{A}\underbrace{\mathbf{C}^{-1}\mathbf{C}}_{\mathbb{I}}\mathbf{x} = \mathbf{C}^{-1}\mathbf{b}\,, \tag{17a}$$

$$\widetilde{\mathbf{A}}\widetilde{\mathbf{x}} = \widetilde{\mathbf{b}}\,, \tag{17b}$$

gdzie $\widetilde{\mathbf{x}} = \mathbf{C}\mathbf{x}$, $\widetilde{\mathbf{b}} = \mathbf{C}^{-1}\mathbf{b}$. Do równania (17b) stosujemy teraz metodę gradientów sprzężonych.

W każdym kroku iteracji musimy obliczyć (tyldy, bo odnosi się to do “tyldowanego” układu (17b))

$$\alpha_k = \frac{\widetilde{\mathbf{r}}_k^T \widetilde{\mathbf{r}}_k}{\widetilde{\mathbf{p}}_k^T \widetilde{\mathbf{A}} \widetilde{\mathbf{p}}_k} = \frac{\widetilde{\mathbf{r}}_k^T \widetilde{\mathbf{r}}_k}{\widetilde{\mathbf{p}}_k^T \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1} \widetilde{\mathbf{p}}_k}\,, \tag{18a}$$

$$\widetilde{\mathbf{r}}_{k+1} = \widetilde{\mathbf{r}}_k - \alpha_k \widetilde{\mathbf{A}} \widetilde{\mathbf{p}}_k = \widetilde{\mathbf{r}}_k - \alpha_k \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1} \widetilde{\mathbf{p}}_k\,, \tag{18b}$$

$$\beta_k = \frac{\widetilde{\mathbf{r}}_{k+1}^T \widetilde{\mathbf{r}}_{k+1}}{\widetilde{\mathbf{r}}_k^T \widetilde{\mathbf{r}}_k}\,, \tag{18c}$$

$$\widetilde{\mathbf{p}}_{k+1} = \widetilde{\mathbf{r}}_{k+1} + \beta_k \widetilde{\mathbf{p}}_k\,, \tag{18d}$$

$$\widetilde{\mathbf{x}}_{k+1} = \widetilde{\mathbf{x}}_k + \alpha_k \widetilde{\mathbf{p}}_k\,. \tag{18e}$$

Równania (18) zawierają jawne odniesienia do macierzy $\mathbf{C}^{-1}$, co nie jest zbyt wygodne. Łatwo się przekonać, iż za pomocą prostych przekształceń macierz tę można „usunąć”, tak, iż pozostaje tylko jedno jej nietrywialne wystąpienie. Zdefiniujmy mianowicie

$$\widetilde{\mathbf{r}}_k = \mathbf{C}^{-1}\mathbf{r}_k\,, \quad \widetilde{\mathbf{p}}_k = \mathbf{C}\mathbf{p}_k\,, \quad \widetilde{\mathbf{x}}_k = \mathbf{C}\mathbf{x}_k\,. \tag{19}$$

W tej sytuacji $\widetilde{\mathbf{r}}_k^T\widetilde{\mathbf{r}}_k = (\mathbf{C}^{-1}\mathbf{r}_k)^T\mathbf{C}^{-1}\mathbf{r}_k = \mathbf{r}_k^T(\mathbf{C}^{-1})^T\mathbf{C}^{-1}\mathbf{r}_k = \mathbf{r}_k^T\mathbf{C}^{-1}\mathbf{C}^{-1}\mathbf{r}_k = \mathbf{r}_k^T(\mathbf{C}^{-1})^2\mathbf{r}_k$ etc.

Wówczas równania (18) przechodzą w

$$\alpha_k = \frac{\mathbf{r}_k^T \left(\mathbf{C}^{-1}\right)^2 \mathbf{r}_k}{\mathbf{p}_k^T \mathbf{A} \mathbf{p}_k}\,, \tag{20a}$$

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k\,, \tag{20b}$$

$$\beta_k = \frac{\mathbf{r}_{k+1}^T \left(\mathbf{C}^{-1}\right)^2 \mathbf{r}_{k+1}}{\mathbf{r}_k^T \left(\mathbf{C}^{-1}\right)^2 \mathbf{r}_k}\,, \tag{20c}$$

$$\mathbf{p}_{k+1} = \left(\mathbf{C}^{-1}\right)^2 \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k\,, \tag{20d}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k\,. \tag{20e}$$

W powyższych równaniach rola macierzy C sprowadza się do obliczenia — *jeden raz w każdym kroku iteracji* — wyrażenia $\left(\mathrm{C}^{-1}\right)^2 \mathbf{r}_k$, co, jak wiadomo, robi się rozwiązując odpowiedni układ równań. Zdefiniujmy

$$\mathrm{M} = \mathrm{C}^2 \,. \tag{21}$$

Macierz M należy rzecz jasna dobrać tak, aby równanie $\mathrm{M}\mathbf{z} = \mathbf{r}$ można było szybko rozwiązać.

Ostatecznie otrzymujemy następujący algorytm:

$$
\begin{aligned}
&\mathbf{r}_1 = \mathbf{b} - \mathbf{A}\mathbf{x}_1 \\
&\text{rozwiąż } \mathbf{M}\mathbf{z}_1 = \mathbf{r}_1 \\
&\mathbf{p}_1 = \mathbf{z}_1 \\
&\textbf{while } \|\mathbf{r}_k\| > \varepsilon \\
&\qquad \alpha_k = \frac{\mathbf{r}_k^T \mathbf{z}_k}{\mathbf{p}_k^T \mathbf{A}\mathbf{p}_k} \\
&\qquad \mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k \\
&\qquad \text{rozwiąż } \mathbf{M}\mathbf{z}_{k+1} = \mathbf{r}_{k+1} \\
&\qquad \beta_k = \frac{\mathbf{r}_{k+1}^T \mathbf{z}_{k+1}}{\mathbf{r}_k^T \mathbf{z}_k} \\
&\qquad \mathbf{p}_{k+1} = \mathbf{z}_{k+1} + \beta_k \mathbf{p}_k \\
&\qquad \mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k \\
&\textbf{end}
\end{aligned}
\tag{22}
$$

Incomplete Cholesky preconditioner

Niech rozkład QR macierzy $\mathbf{C}$ ma postać $\mathbf{C} = \mathbf{Q}\mathbf{H}^T$, gdzie $\mathbf{Q}$ jest macierzą ortogonalną, $\mathbf{H}^T$ jest macierzą trójkątną górną. Zauważmy, że

$$\mathbf{M} = \mathbf{C}^2 = \mathbf{C}^T\mathbf{C} = \left(\mathbf{Q}\mathbf{H}^T\right)^T \mathbf{Q}\mathbf{H}^T = \mathbf{H}\mathbf{Q}^T\mathbf{Q}\mathbf{H}^T = \mathbf{H}\mathbf{H}^T, \qquad (23)$$

a więc macierz $\mathbf{H}$ jest czynnikiem Cholesky'ego macierzy $\mathbf{M}$. Niech rozkład Cholesky'ego macierzy $\mathbf{A}$ ma postać $\mathbf{A} = \mathbf{G}\mathbf{G}^T$. *Przypuśćmy, iż* $\mathbf{H} \simeq \mathbf{G}$.

Wówczas

$$\widetilde{\mathbf{A}} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-1} = \left(\mathbf{C}^{T}\right)^{-1}\mathbf{A}\mathbf{C}^{-1} = \left(\left(\mathbf{Q}\mathbf{H}^{T}\right)^{T}\right)^{-1}\mathbf{A}\left(\mathbf{Q}\mathbf{H}^{T}\right)^{-1} = \left(\mathbf{H}\mathbf{Q}^{T}\right)^{-1}\mathbf{A}\left(\mathbf{H}^{T}\right)^{-1}\mathbf{Q}^{T} = \mathbf{Q}\underbrace{\mathbf{H}^{-1}\mathbf{G}}_{\simeq\mathbb{I}}\underbrace{\mathbf{G}^{T}\left(\mathbf{H}^{T}\right)^{-1}}_{\simeq\mathbb{I}}\mathbf{Q}^{T} \simeq \mathbf{Q}\mathbf{Q}^{T} = \mathbb{I}\,. \tag{24}$$

Ponieważ $\widetilde{\mathbf{A}} \simeq \mathbb{I}$, współczynnik uwarunkowania tej macierzy powinien być bliski jedności.

Niepełny rozkład Cholesky'ego — algorytm w wersji *GAXPY*

for $k = 1{:}N$
  $H_{kk} = A_{kk}$
  for $j = 1{:}k{-}1$
    $H_{kk} = H_{kk} - H_{kj}^2$
  end
  $H_{kk} = \sqrt{H_{kk}}$
  for $l = k{+}1{:}N$
    $H_{lk} = A_{lk}$
    if $A_{lk} \neq 0$
      for $j = 1{:}k{-}1$
        $H_{lk} = H_{lk} - H_{lj}H_{kj}$
      end
      $H_{lk} = H_{lk}/H_{kk}$
    endif
  end
end

Uwagi

- Ponieważ $\mathbf{A}$ jest rzadka, powyższy algorytm na obliczanie przybliżonego czynnika Cholesky'ego wykonuje się szybko. Wykonuje się go *tylko raz*.
- Równanie $\mathbf{Mz} = \mathbf{r}$ rozwiązuje się szybko, gdyż znamy czynnik Cholesky'ego $\mathbf{M} = \mathbf{H}\mathbf{H}^T$.
- Obliczone $\mathbf{H}$ jest rzadkie, a zatem równanie $\mathbf{Mz} = \mathbf{r}$ rozwiązuje się szczególnie szybko.
- Mamy nadzieję, że macierz $\tilde{\mathbf{A}}$ ma współczynnik uwarunkowania bliski jedności, a zatem nie potrzeba wielu iteracji (22).

Przykład — macierz pasmowa z pustymi diagonalami

Rozważmy macierz o następującej strukturze:

$$\mathbf{A} = \begin{bmatrix} a_1 & 0 & b_3 & 0 & c_5 & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & a_2 & 0 & b_4 & 0 & c_6 & 0 & 0 & 0 & 0 & \dots \\ b_3 & 0 & a_3 & 0 & b_5 & 0 & c_7 & 0 & 0 & 0 & \dots \\ 0 & b_4 & 0 & a_4 & 0 & b_6 & 0 & c_8 & 0 & 0 & \dots \\ c_5 & 0 & b_5 & 0 & a_5 & 0 & b_7 & 0 & c_9 & 0 & \dots \\ 0 & c_6 & 0 & b_6 & 0 & a_6 & 0 & b_8 & 0 & c_{10} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \tag{25}$$

Macierz ta jest symetryczna, zakładamy też, że jest dodatnio określona.

Niepełny czynnik Cholesky'ego macierzy (25) ma postać

$$\mathbf{H} = \begin{bmatrix} p_1 & & & & & & \\ 0 & p_2 & & & & & \\ q_3 & 0 & p_3 & & & & \\ 0 & q_4 & 0 & p_4 & & & \\ r_5 & 0 & q_5 & 0 & p_5 & & \\ 0 & r_6 & 0 & q_6 & 0 & p_6 & \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \tag{26}$$

(W pełnym czynniku Cholesky'ego macierzy (25) zera leżące w (26) *pomiędzy* diagonalą "p" a diagonalną "r" znikłyby — w ogólności mogłyby tam znajdować się jakieś niezerowe liczby.)

Zgodnie z podanym algorytmem, elementy ciągów $\{p_k\}$, $\{q_k\}$, $\{r_k\}$ wyliczamy z następujących wzorów:

$$
\begin{array}{rcl@{\qquad}rcl}
p_1 &=& \sqrt{a_1}, & p_2 &=& \sqrt{a_2}, \\
q_3 &=& b_3/p_1, & q_4 &=& b_4/p_2, \\
r_5 &=& c_5/p_1, & r_6 &=& c_6/p_2, \\
p_3 &=& \sqrt{a_3 - q_3^2}, & p_4 &=& \sqrt{a_4 - q_4^2}, \\
q_5 &=& (b_5 - r_5 q_3)/p_3, & q_6 &=& (b_6 - r_6 q_4)/p_4, \\
r_7 &=& c_7/p_3, & r_8 &=& c_7/p_4, \\
p_5 &=& \sqrt{a_5 - q_5^2 - r_5^2}, & p_6 &=& \sqrt{a_6 - q_5^6 - r_6^2}, \\
q_7 &=& (b_7 - r_7 q_5)/p_5, & q_8 &=& (b_8 - r_8 q_6)/p_6, \\
r_9 &=& c_9/p_5, & r_{10} &=& c_{10}/p_6, \\
p_7 &=& \sqrt{a_7 - q_7^2 - r_7^2}, & p_8 &=& \sqrt{a_8 - q_8^6 - r_8^2}, \\
q_9 &=& (b_9 - r_9 q_7)/p_7, & q_{10} &=& (b_{10} - r_{10} q_8)/p_8, \\
r_{11} &=& c_{11}/p_7, & r_{12} &=& c_{12}/p_8, \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots
\end{array}
$$

Macierze niesymetryczne

Jeżeli w równaniu

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{27}$$

macierz $\mathbf{A}$ nie jest symetryczna i dodatnio określona, sytuacja się komplikuje. Zakładając, że $\det \mathbf{A} \neq 0$, równanie (27) możemy "zsymetryzować" na dwa sposoby.
CGNR:

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}\,, \tag{28}$$

lub CGNE:

$$\begin{aligned} \mathbf{A}\mathbf{A}^T\mathbf{y} &= \mathbf{b}\,, && \text{(29a)} \\ \mathbf{x} &= \mathbf{A}^T\mathbf{y}\,. && \text{(29b)} \end{aligned}$$

Do dwu powyższych równań formalnie rzecz biorąc *można* używać metody gradientów sprzężonych. Trzeba jednak pamiętać, że nawet jeśli macierz $\mathbf{A}$ jest rzadka, macierze $\mathbf{A}^T\mathbf{A}$, $\mathbf{A}\mathbf{A}^T$ nie muszą być rzadkie, a co gorsza, ich współczynnik uwarunkowania jest kwadratem współczynnika uwarunkowania macierzy wyjściowej.

Alternatywnie, zamiast "symetryzować" macierz, można zmodyfikować algorytm, tak aby zamiast dwu, generował on *cztery* ciągi wektorów. Należy jednak pamiętać, że dla wielu typów macierzy taki algorytm bywa bardzo wolno zbieżny, a niekiedy nawet dochodzi do kompletnej stagnacji przed uzyskaniem rozwiązania:

Metoda gradientów bi-sprzężonych (*Bi-Conjugate Gradients*, Bi-CG)

$\mathbf{r}_1 = \mathbf{b} - \mathbf{A}\mathbf{x}_1$, $\mathbf{p}_1 = \mathbf{r}_1$, $\bar{\mathbf{r}}_1 \neq 0$ dowolny, $\bar{\mathbf{p}}_1 = \bar{\mathbf{r}}_1$

while $\|\mathbf{r}_k\| > \varepsilon$

$$
\begin{aligned}
\alpha_k &= \frac{\bar{\mathbf{r}}_k^T \mathbf{r}_k}{\bar{\mathbf{p}}_k^T \mathbf{A}\mathbf{p}_k} \\
\mathbf{r}_{k+1} &= \mathbf{r}_k - \alpha_k \mathbf{A}\mathbf{p}_k \\
\bar{\mathbf{r}}_{k+1} &= \bar{\mathbf{r}}_k - \alpha_k \mathbf{A}^T \bar{\mathbf{p}}_k \\
\beta_k &= \frac{\bar{\mathbf{r}}_{k+1}^T \mathbf{r}_{k+1}}{\bar{\mathbf{r}}_k^T \mathbf{r}_k} \\
\mathbf{p}_{k+1} &= \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k \\
\bar{\mathbf{p}}_{k+1} &= \bar{\mathbf{r}}_{k+1} + \beta_k \bar{\mathbf{p}}_k \\
\mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k
\end{aligned}
\tag{30}
$$

end

Wektory wygenerowane w algorytmie (30) spełniają następujące relacje:

$$\bar{\mathbf{r}}_i^T \mathbf{r}_j = \mathbf{r}_i^T \bar{\mathbf{r}}_j = 0\,,\ i > j\,, \tag{31a}$$
$$\bar{\mathbf{r}}_i^T \mathbf{p}_j = \mathbf{r}_i^T \bar{\mathbf{p}}_j = 0\,,\ i > j\,, \tag{31b}$$
$$\bar{\mathbf{p}}_i^T \mathbf{A}\mathbf{p}_j = \mathbf{p}_i^T \mathbf{A}^T \bar{\mathbf{p}}_j = 0\,,\ i > j\,. \tag{31c}$$

Jeżeli w algorytmie (30) weźmiemy $\bar{\mathbf{r}}_1 = \mathbf{A}\mathbf{r}_1$, we wszystkich krokach zachodzić będzie $\bar{\mathbf{r}}_k = \mathbf{A}\mathbf{r}_k$ oraz $\bar{\mathbf{p}}_k = \mathbf{A}\mathbf{p}_k$. Jest to wersja przydatna dla rozwiązywania układów równań z macierzami symetrycznymi, ale nieokreślonymi dodatnio. Jest to przy okazji szczególny wariant algorytmu GMRES (*generalised minimum residual*), formalnie odpowiadającego minimalizacj funkcjonału

$$\Phi(\mathbf{x}) = \frac{1}{2}\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2\,. \tag{32}$$