

Równania różniczkowe cząstkowe (RRCz)

→ równanie eliptyczne

→ równanie Poissona

1. Klasyfikacja RRCz, przykłady
2. Metody numerycznego rozwiązywania równania Poissona
 - a) FFT (met. bezpośrednia)
 - b) metoda różnic skończonych
 - c) układ równań z jawnie zdefiniowaną macierzą
 - d) metody iteracyjne bez konstrukcji macierzy:
 - metoda Jakobiego, Gaussa-Seidla, zbieżność
 - całka działania
 - relaksacja, nadrelaksacja
 - metoda zagęszczania siatki
 - relaksacja wielosiatkowa

Równania różniczkowe cząstkowe – klasyfikacja równań

Ogólna postać RRCz 2 rzędu zaleźnego od n zmiennych

$$\mathbf{z} \in R^n, \quad \text{np.: } \mathbf{z} = (x, y, z, t)^T$$

$$-\sum_{i,j=1}^n \frac{\partial}{\partial z_i} \left(a_{i,j} \frac{\partial u}{\partial z_j} \right) + \sum_{i=1}^n \left(\frac{\partial b_i u}{\partial z_i} + c_i \frac{\partial u}{\partial z_i} \right) + a_0 u = f$$

i zakładamy

$$\begin{aligned} u &= u(\mathbf{z}) \in C^2 & c_i &= c_i(\mathbf{z}) \in C^1 \\ a_{i,j} &= a_{i,j}(\mathbf{z}) \in C^1 & a_0 &= a_0(\mathbf{z}) \in C \\ b_i &= b_i(\mathbf{z}) \in C^1 & f &= f(\mathbf{z}) \in C \end{aligned}$$

Zakładając, że: $\mathbf{v} \in R^n, \quad \mathbf{v} \neq 0$ wówczas RRCz jest:

- **eliptyczne** jeśli $v^T A v > 0$ (A jest dodatniookreślona)
- **paraboliczne** jeśli: $v^T A v \geq 0 \quad \wedge \quad \text{rank}\{A, b, c\} = n$
- **hiperboliczne** jeśli wartości własne A:

$$\lambda_1, \lambda_2, \dots, \lambda_{n-1} > 0 \quad \wedge \quad \lambda_n < 0$$

Eliptyczne - równanie Poissona

$$-\nabla^2 = f$$

$$\mathbf{z} = (x, y, z)^T \quad - \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \right) = f \quad A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Opisuje rozwiązania **stacjonarne**. Gdzie się pojawia?

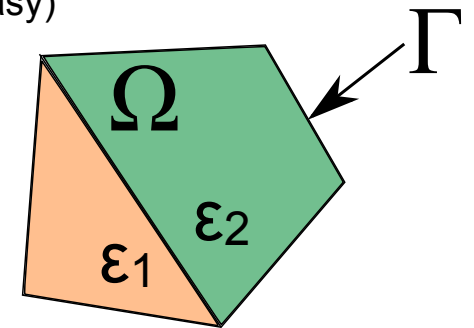
- Rozkład potencjału elektrycznego (grawitacyjnego)

$$-\nabla \cdot \varepsilon \nabla u = f$$

ε - „stała” dielektryczna (w określonych podobszarach Ω), f - gęstość ładunku (masy)

- stacjonarne rozwiązania równania dyfuzji (s – źródło ciepła)

$$\frac{\partial T}{\partial t} = \nabla^2 T - S \quad \implies \quad \frac{\partial T}{\partial t} = 0$$



- Jedno z równań Naviera-Stokesa (stacjonarny przepływ cieczy lepkiej)

$$\nabla^2 \Psi = \zeta$$

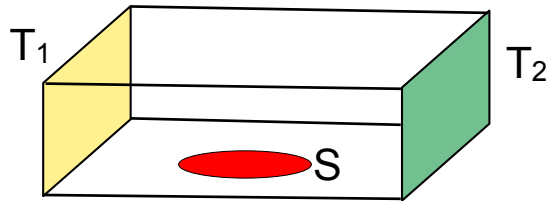
$$\mu \nabla^2 \zeta = \rho \left(\frac{\partial \psi}{\partial y} \frac{\partial \zeta}{\partial x} - \frac{\partial \psi}{\partial x} \frac{\partial \zeta}{\partial y} \right)$$

to równaniem Poissona nie jest

Paraboliczne – równanie dyfuzji ciepła

(k – wsp. przewodności cieplnej, S – źródło ciepła)

$$\frac{\partial T}{\partial t} - \nabla \cdot k \nabla T = -S$$



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

4 zmienne niezależne - ale mamy tylko 3 pochodne 2 rzędu w równaniu

Hiperboliczne – równanie falowe

$$\frac{\partial^2 u}{\partial t^2} - \frac{1}{v^2} \nabla^2 u = \beta u + f$$

Opisują drgania mechaniczne np. membrany w 2D (f – wymuszenie, β - tłumienie)

lub rozchodzenie się fal elektromagnetycznych np. w światłowodzie

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

$$\lambda_{1,2,3} = 1$$

$$\lambda_4 = -1$$

Eliptyczne RRCz na przykładzie równania Poissona

Aby rozwiązanie równania eliptycznego było jednoznaczne musi ono spełniać określone warunki na brzegu obszaru

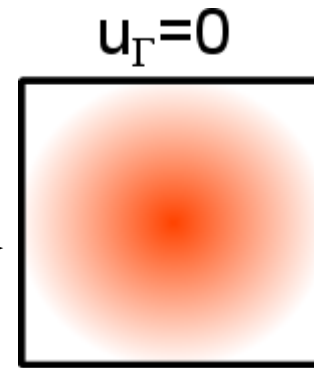
– rozwiązujemy tzw. **problem brzegowy** RRCz.

Typy warunków brzegowych (Γ - brzeg):

1. Dirichleta

- jednorodne

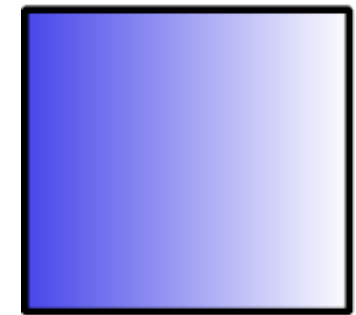
$$u|_{\Gamma} = 0$$



- niejednorodne

$$u|_{\Gamma} = C(\mathbf{r})$$

$$u_{\Gamma} = C(\mathbf{r})$$



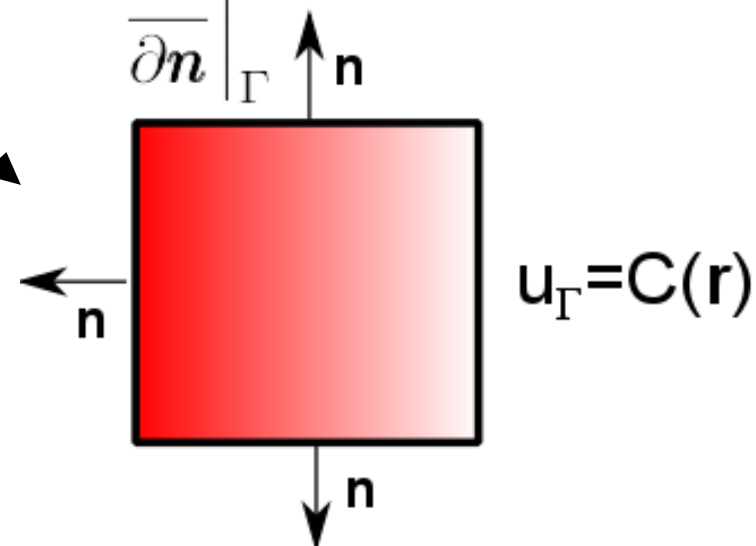
2. Neumanna

$$\left. \frac{\partial u}{\partial \mathbf{n}} \right|_{\Gamma} = D(\mathbf{r})$$

3. mieszane (Robina)

$$\left(u + \frac{\partial u}{\partial \mathbf{n}} \right)_{\Gamma} = W(\mathbf{r})$$

$$\left. \frac{\partial u}{\partial \mathbf{n}} \right|_{\Gamma}$$

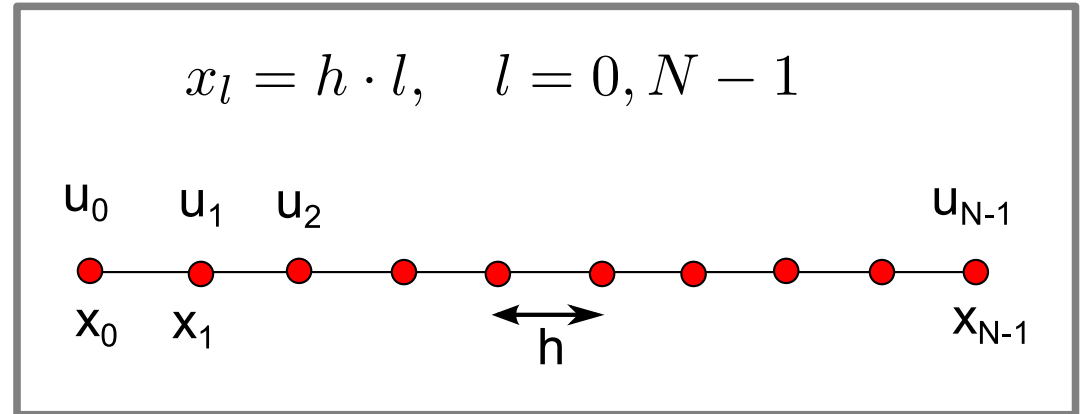


Metoda różnic skończonych w 1D/2D

Wprowadzamy siatkę przestrzenną z równoodległymi węzłami

Rozwiązania $u(x)$ poszukujemy w położeniach węzłowych.

Dyskretyzujemy rów. Poissona zastępując pochodne ciągłe ilorazami różnicowymi (np. 2 pochodna \rightarrow iloraz trójpunktowy symetryczny)



1D

$$u(x) = u(x_l) = u_l$$

$$\rho(x) = \rho(x_l) = \rho_l$$

$$\nabla^2 u(x) = \frac{\partial^2 u}{\partial x^2} = -\rho(x)$$

$$\frac{u_{l-1} - 2 \cdot u_l + u_{l+1}}{h^2} = -\rho_l$$

$$\nabla^2 u(x) = -\rho(x)$$

2D

$$u(x, y) = u(x_l, y_j) = u_{l,j}$$

$$\rho(x, y) = \rho(x_l, y_j) = \rho_{l,j}$$

$$\nabla^2 u(x) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -\rho(x, y)$$

$$\frac{u_{l-1,j} - 2 \cdot u_{l,j} + u_{l+1,j}}{\Delta x^2} + \frac{u_{l,j-1} - 2 \cdot u_{l,j} + u_{l,j+1}}{\Delta y^2} = -\rho_{l,j}$$

Rozwiązywanie równania Poissona przy użyciu FFT (fast Poisson solver)

Przypomnienie → FFT w 1D:

u_l - wartości potencjału w węzłach sieci prostej

\hat{u}_l - wartości potencjału w węzłach sieci odwrotnej (**transformata**)

$$k_m = \frac{2\pi}{N \cdot h} m \quad x_l = h \cdot l \quad l, n = 0, 1, \dots, N - 1$$
$$y_j = h \cdot j \quad j, m = 0, 1, \dots, M - 1$$

$$1D \implies u_l = \frac{1}{N} \sum_{n=0}^{N-1} \hat{u}_n e^{ik_n x_l} = \frac{1}{N} \sum_{n=0}^{N-1} \hat{u}_n e^{i \frac{n \cdot 2\pi}{N h} h l} = \frac{1}{N} \sum_{n=0}^{N-1} \hat{u}_n e^{i \frac{2\pi n l}{N}}$$

$$2D \implies u_{l,j} = \frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \hat{u}_{n,m} e^{i \frac{2\pi n l}{N}} e^{i \frac{2\pi m j}{M}}$$

$$\rho_{l,j} = \frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \hat{\rho}_{n,m} e^{i \frac{2\pi n l}{N}} e^{i \frac{2\pi m j}{M}}$$

Po wstawieniu transformat odwrotnych do zdyskretyzowanego równania Poissona:

$$\frac{u_{l-1,j} - 2 \cdot u_{l,j} + u_{l+1,j}}{\Delta x^2} + \frac{u_{l,j-1} - 2 \cdot u_{l,j} + u_{l,j+1}}{\Delta y^2} = -\rho_{l,j}$$

dla $\Delta x = \Delta y = h$ otrzymujemy

$$\hat{u}_{n,m} \left(e^{i \frac{2\pi n(-1)}{N}} - 2 + e^{i \frac{2\pi n(+1)}{N}} + e^{i \frac{2\pi m(-1)}{M}} - 2 + e^{i \frac{2\pi m(+1)}{M}} \right) = \hat{\rho}_{n,m} \cdot h^2$$

korzystając z relacji $e^{ix} + e^{-ix} = 2\cos(x)$

otrzymamy

$$\hat{u}_{nm} = \frac{\hat{\rho}_{n,m} h^2}{2 \left(\cos \frac{2\pi n}{N} + \cos \frac{2\pi m}{M} - 2 \right)}$$

→ rozwiązanie uzyskamy wykonując transformację odwrotną (poprzedni slajd)

→ jest ono periodyczne

$$u_{l,j} = u_{l+N,j} = u_{l,j+M}$$

→ czy istnieje problem dla $m=n=0$? (zero w mianowniku) –

nie ponieważ przyczynek od $k_m = k_n = 0$ pochodzi od funkcji stałej,

którą możemy dodać do naszego rozwiązania (tak aby spełniało np. WB)

Warunki brzegowe:

→ **jednorodne WB Dirichleta** $u_{l,j}\Gamma = 0$ wówczas stosujemy transformację sinusową

$$\hat{\rho}_{n,m} = \sum_{j=0}^{N-1} \sum_{l=0}^{M-1} \rho_{j,l} \sin\left(\frac{\pi nl}{N}\right) \sin\left(\frac{\pi mj}{M}\right)$$

Stosując zdyskretyzowaną postać rów. Poissona korzystamy z relacji

$$\sin(x+y) + \sin(x-y) = 2\sin(x)\cos(y)$$

co prowadzi do wyrażenia

$$\hat{u}_{nm} = \frac{\hat{\rho}_{n,m} h^2}{2 \left(\cos \frac{\pi n}{N} + \cos \frac{\pi m}{M} - 2 \right)}$$

Po obliczeniu wszystkich $\hat{u}_{n,m}$ dokonujemy transformacji odwrotnej potencjału

$$u_{l,j} = \frac{2}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \hat{u}_{n,m} \sin\left(\frac{\pi nl}{N}\right) \sin\left(\frac{\pi mj}{M}\right)$$

Warunki brzegowe cd.:

→ **niejednorodne Dirichleta** np. $u(x_{max}, y) = u^b(y)$ i jednorodne w pozostałej części brzegu

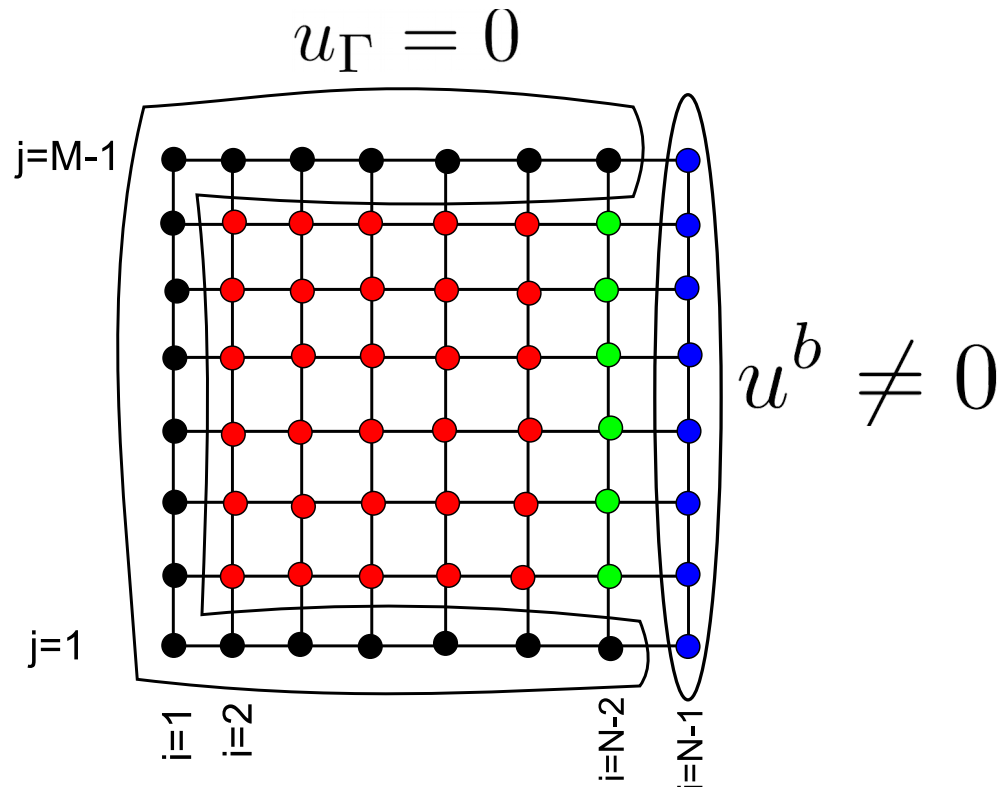
Rozwiązanie można zapisać w postaci (zasada superpozycji)

$$u = u' + u^b$$

Z warunkami

$$u'_\Gamma = 0 \quad u^b_\Gamma = u^b(y)|_{x=x_{max}}$$

$$u'_{\Omega} \neq 0 \quad u'_{\Omega} = 0$$



Po wstawieniu u do zdyskretyzowanego r. Poissona dostaniemy

$$\nabla^2 u' = -\nabla^2 u^b - \rho$$

$$\begin{aligned} u'_{l-1,j} - 2u'_{l,j} + u'_{l+1,j} + u'_{l,j-1} - 2u'_{l,j} + u'_{l,j+1} &= \dots \\ \dots &= -\left(u^b_{l-1,j} - 2u^b_{l,j} + u^b_{l+1,j} + u^b_{l,j-1} - 2u^b_{l,j} + u^b_{l,j+1}\right) - h^2 \rho_{l,j} \end{aligned}$$

Wkład od u_b pochodzi tylko od węzłów $i=N-1$ leżących na prawym brzegu
(zakładamy że poza brzegiem $u_b=0$)

$$u^b_{l,j} = u^b_j \cdot \delta_{l,N-1}$$

$$u'_{l-1,j} + u'_{l+1,j} + u'_{l,j-1} + u'_{l,j+1} - 4u'_{l,j} = \underbrace{4u^b_j \cdot \delta_{l,N-1} - h^2 \rho_{l,j}}_{f'_{l,j}}$$

Czyli modyfikujemy tylko prawą stronę równania (gęstość) – dalej postępujemy jak w przypadku warunków jednorodnych Dirichleta.

Warunki brzegowe cd.:

→ **warunki von Neumanna**

$$\left. \frac{\partial u}{\partial \mathbf{n}} \right|_{\Gamma} = C_{\Gamma}(\mathbf{r})$$

→ dla **jednorodnych warunków v.N.**
stosujemy transformację kosinusową,
która automatycznie spełnia WB

$$\left. \frac{\partial u}{\partial \mathbf{n}} \right|_{\Gamma} = 0$$

$$\hat{u}_{nm} = \frac{\hat{\rho}_{n,m} h^2}{2 \left(\cos \frac{2\pi n}{N} + \cos \frac{2\pi m}{M} - 2 \right)}$$

$$u_{l,j} = \frac{2}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \hat{u}_{n,m} \cos \left(\frac{\pi n l}{N} \right) \cos \left(\frac{\pi m j}{M} \right) \cdot \boxed{C_n \cdot C_m}$$

$$C_p = \begin{cases} 1 & p \in \Omega \\ \frac{1}{2} & p \in \Gamma \end{cases}$$

→ **niejednorodne WB. Neumana**

$$\frac{\partial u}{\partial n} \Big|_{\Gamma} \neq 0$$

np. na prawym brzegu

$$\frac{\partial u}{\partial x} \Big|_{x_{N-1}} = C(y)$$

Znowu zakładamy postać rozwiązania na brzegu

$$u = u' + u^B$$

z warunkami:

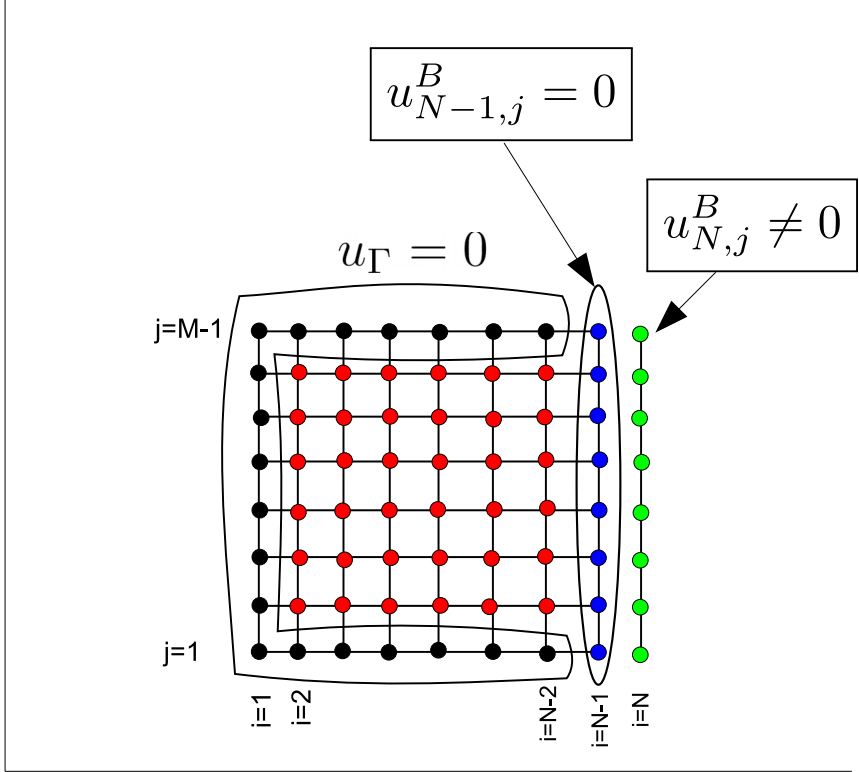
$$\begin{cases} \frac{\partial u'}{\partial x} \Big|_{x_{N-1}} = 0 \\ u'_{N-1} \neq 0 \end{cases}$$

niezerowy wkład do wartości
+ znikanie pochodnej
= jednorodne WB v. Neumanna
= **transformacja kosinusowa**

Znika wartość na brzegu, ale daje wkład do pochodnej

$$\begin{aligned} \frac{\partial u^B}{\partial x} \Big|_{x_{N-1}} &= C_j \\ u^B_N &\neq 0 \\ u^B_{N-2}, u^B_{N-1} &= 0 \end{aligned}$$

$$\frac{u^B_{N,j} - u^B_{N-2,j}}{2h} = \frac{u^B_{N,j}}{2h} \rightarrow u^B_{N,j} = 2hC_j$$



Do równania różnicowego

$$u'_{l-1,j} - 2u'_{l,j} + u'_{l+1,j} + u'_{l,j-1} - 2u'_{l,j} + u'_{l,j+1} = \dots$$

$$\dots = - (u^b_{l-1,j} - 2u^b_{l,j} + u^b_{l+1,j} + u^b_{l,j-1} - 2u^b_{l,j} + u^b_{l,j+1}) - h^2 \rho_{l,j}$$

wstawiamy dla $l=N-1$
(pozostałe wyrazy =0)

$$u^B_{N,j} = 2hC_j$$

I otrzymujemy równanie ze zmodyfikowaną prawą stroną
dla **transformacji kosinusowej**:

$$u'_{l-1,j} + u'_{l+1,j} + u'_{l,j-1} + u'_{l,j+1} - 4u'_{l,j} = \underbrace{2hC_j \delta_{N-1,l}}_{f'_{l,j}} - h^2 \rho_{l,j}$$

Zalety:

- metoda szybka (jedynie FMG może być szybsza) i bezpośrednia (daje wynik po skończonej liczbie operacji)
- dobrze działa na siatce prostokątnej
- transformacje wykonywane na jednej tablicy potencjału (potrzebujemy jeszcze 2 tablicę dla prawej strony RRCz)

Wady:

- zastosowanie tylko do obszarów o regularnych kształtach: prostopadłościan, sfera, cylinder
- nie uwzględnia powierzchni brzegowej w środku np. elektrody włożone do obszaru, w którym szukamy potencjału

Biblioteki: **Math Kernel Library** (Fortran, C/C++) → Fast Poisson Solver
(wykorzystuje pakiet FFTW)

Jeszcze o metodzie bezpośredniej rozwiązywania r. Poissona.

Skorzystajmy ze zdyskretyzowanej postaci równania w 1D

$$\frac{u_{l-1} - 2 \cdot u_l + u_{l+1}}{h^2} = -\rho_l$$

→ generuje ono jedno równanie dla ustalonej wartości l .

Zapisując je dla $l=0,2,3,\dots,N-1$ otrzymamy układ równań

przykład

$$A\mathbf{u} = \mathbf{b}$$

$$\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} \Delta^2 \rho_1 \\ \Delta^2 \rho_2 \\ \Delta^2 \rho_3 \\ \Delta^2 \rho_4 \end{bmatrix}$$

→ Jak wprowadzić warunki brzegowe?

Uwzględniamy warunki brzegowe:

$$u_i = C_i$$

→ **Dirichleta** (3 kroki)

$$1) a_{i,j} = 0, \text{ dla } j = 1, \dots, N$$

$$2) a_{i,i} = 1$$

$$3) b_i = C_i$$

→ **von Neumanna**

np.: na prawym brzegu

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} C_1 \\ \Delta^2 \rho_2 \\ \Delta^2 \rho_3 \\ h D_N \end{bmatrix}$$

$$\left. \frac{du}{dx} \right|_{x=x_N} = \frac{u_N - u_{N-1}}{h} = D_N$$

$$-u_{N-1} + u_N = h D_N$$

Postać macierzowa operatora Laplace'a w 2D i 3D

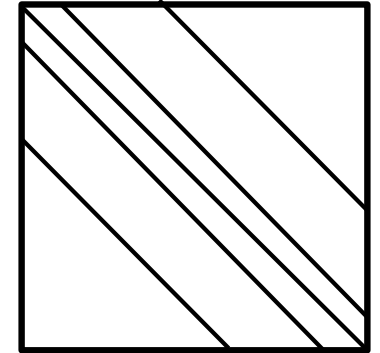
$$\nabla^2 u(x, y) = -\rho(x, y)$$

2D

$$x = x_i, \quad i = 1, 2, \dots, n_x$$

$$y = y_j, \quad j = 1, 2, \dots, n_y$$

A=



$$u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} = -\Delta^2 \rho_{i,j}$$

reindeksacja węzłów

$$l = i + (j - 1) \cdot n_x, \quad l = 1, 2, \dots, N, \quad N = n_x \cdot n_y$$

5 przekątnych:

$$u_{l-n_x} + u_{l-1} - 4u_l + u_{l+1} + u_{l+n_x} = -\Delta^2 \rho_l$$

3D (analogicznie)

$$z = z_k, \quad k = 1, 2, \dots, n_z$$

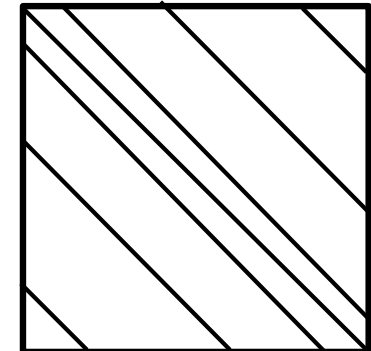
$$l = i + (j - 1) \cdot n_x + (k - 1) \cdot n_x \cdot n_y, \quad l = 1, 2, \dots, N, \quad N = n_x \cdot n_y \cdot n_z$$

7 przekątnych:

$$u_{l-n_x n_y} + u_{l-n_x} + u_{l-1} - 6u_l + u_{l+1} +$$

$$+ u_{l+n_x} + u_{l+n_x n_y} = -\Delta^2 \rho_l$$

A=



Kiedy opłaca się rozwiązywać rów. Poissona znajdując rozwiązanie $Ax=b$?

- tylko wtedy gdy używamy algorytmów dla macierzy rzadkich
- w 1D UARL rozwiążemy przy użyciu LU dla macierzy trójdzielnych z nakładem $\sim O(n)$
- w 2D UARL też rozwiążemy stosując LU ale dla macierzy pięcioprzekatniowych
- w 3D można stosować
 - i) LU jeśli liczba węzłów jest mała
 - ii) iLU jako preconditioner dla metod iteracyjnych (BiCGStab, GMRES, TFQMR itp.)

Metody iteracyjne

Punktem wyjścia jest układ równań
(jeszcze w postaci macierzowej)

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

Który rozwiązywać będziemy iteracyjnie

$$\mathbf{x}^{k+1} = M\mathbf{x}^k + \mathbf{c}$$

M jest macierzą iteracji, która posiada wartości i wektory własne

$$M\mathbf{v}_i = \lambda_i\mathbf{v}_i$$

$$\rho(M) = \max\{|\lambda_1|, |\lambda_2|, \dots, |\lambda_n|\}$$

Metoda iteracyjna jest zbieżna wtedy i tylko wtedy,
gdy promień spektralny spełnia warunek

$$\rho(M) < 1$$

Co dają metody iteracyjne:

- jeśli M jest macierzą rzadką to koszt jednej iteracji jest rzędu $O(n)$, dla pełnej macierzy $O(n^2)$
- jeśli rozwiązanie startowe jest „bliskie” dokładnemu to ilość iteracji może być mała (rel. wielosiatkowe)
- zazwyczaj prosta konstrukcja macierzy M, a **w przypadku rów. Poissona nie trzeba jej nawet tworzyć** (zysk w postaci ograniczenia zajętej pamięci)
- jak w każdej procedurze iteracyjnej mogą jednak wystąpić problemy ze zbieżnością, zbieżność silnie zależy od **postaci macierzy iterującej/schematu relaksacyjnego**

Zbieżność metody iteracyjnej

Założenia:

$$A\mathbf{x}_{dok} = \mathbf{b}$$

$$\mathbf{x}^{k+1} = M\mathbf{x}^k + \mathbf{c}$$

$$\rho(M) < 1$$

Dla przepisu iteracyjnego mamy

$$\mathbf{x}^k = \mathbf{x}_{dok} + \mathbf{e}^k \leftarrow \text{błąd w } k\text{-tej iteracji}$$

$$\mathbf{x}^{k+1} = \mathbf{x}_{dok} + \mathbf{e}^{k+1} = \underbrace{M(\mathbf{x}_{dok} + \mathbf{e}^k) + \mathbf{c}}_{(M\mathbf{x}_{dok} + \mathbf{c})} + M\mathbf{e}^k$$

$$\mathbf{e}^{k+1} = M\mathbf{e}^k$$

$$\mathbf{e}^{k+1} = M^{k+1}\mathbf{e}^0$$

Mamy dostępną bazę w postaci wektorów własnych M:

$$M\mathbf{v}_i = \lambda_i\mathbf{v}_i$$

$$\mathbf{e}^0 = \sum_i c_i\mathbf{v}_i$$

$$\mathbf{e}^{k+1} = M^{k+1}\mathbf{e}^0 = \sum_i c_i\lambda_i^{k+1}\mathbf{v}_i \implies \lim_{k \rightarrow \infty} \mathbf{e}^{k+1} = \mathbf{0}$$

Metoda powinna być zbieżna, ale jaka będzie szybkość zbieżności do rozwiązania dokładnego? → **zależy od postaci M**

Konstrukcja macierzy iterującej

- Założenia: 1) $Ax = b \rightarrow x^{k+1} = Mx^k + c$
2) zbieżność: $\rho(M) < 1$
3) tempo zbieżności $R_\infty = -\log_{10}(\rho(M))$

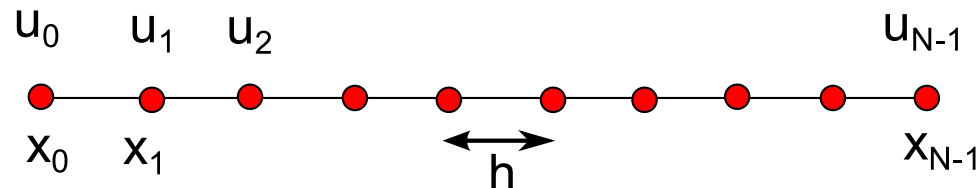
$$\begin{aligned} A &= B + C \\ (B + C)x &= b \\ Bx &= b - Cx \quad B^{-1} \cdot / \\ x &:= -B^{-1}Cx + B^{-1}b \\ M &= -B^{-1}C, \quad c = B^{-1}b \end{aligned}$$

Metoda Jakobiego: $A = D + L + U$

diagonala macierz trójkątna dolna macierz trójkątna górna

$$B = D, \quad C = L + U$$

$$u_{i-1} - 2 \cdot u_i + u_{i+1} = -\rho_i \cdot h^2$$



$$A_{6 \times 6} = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

$$M = -D^{-1}(L + U) = \begin{bmatrix} 0 & 1/2 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 1/2 & 0 \end{bmatrix}$$

$$B = D^{-1} = \begin{bmatrix} -\frac{1}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

Przepis iteracyjny (macierzowy):

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

Przepis dla pojedynczego węzła siatki
→ to **wzór relaksacyjny**:

$$x_i^{k+1} = \frac{1}{2} (x_{i-1}^k + x_{i+1}^k) + \frac{1}{2} \rho_i \cdot h^2$$

$$C = L + U = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

Otrzymaliśmy wzór relaksacyjny dla metody Jakobiego

$$x_i^{k+1} = \frac{1}{2} (x_{i-1}^k + x_{i+1}^k) + \frac{1}{2} \rho_i \cdot h^2$$

→ nie musimy już konstruować macierzy
(ta ma prostą konstrukcję, a jej współczynniki są zawarte w równaniu)

→ w obliczeniach **musimy** użyć dwóch tablic/wektorów,
dla starego i nowego rozwiązania,
metoda Jakobiego to relaksacja globalna

→ jaka jest zbieżność tej metody?

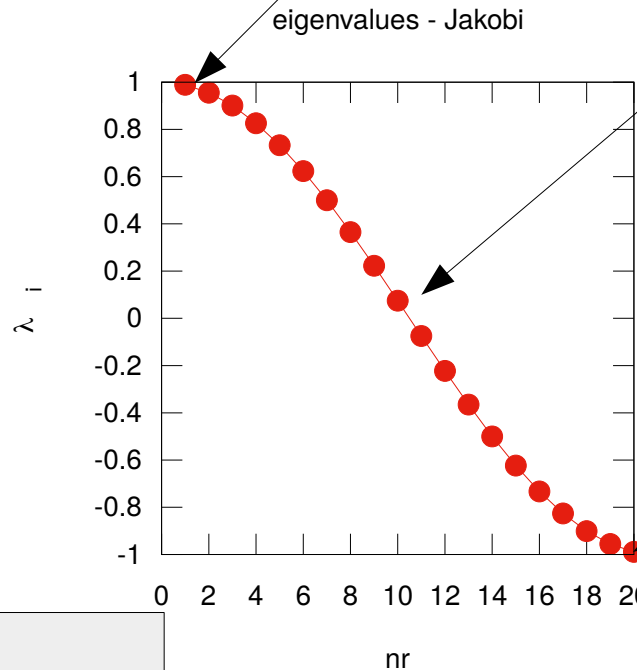
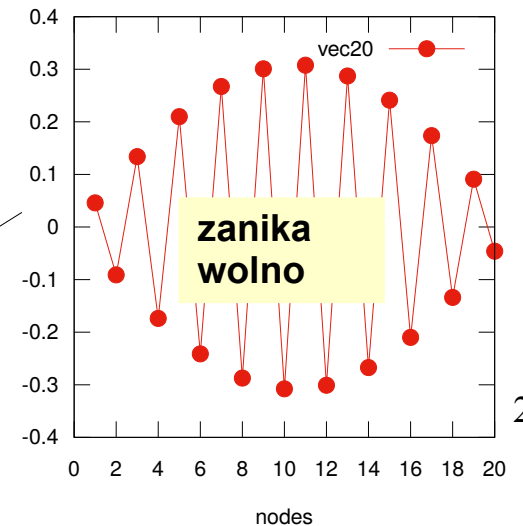
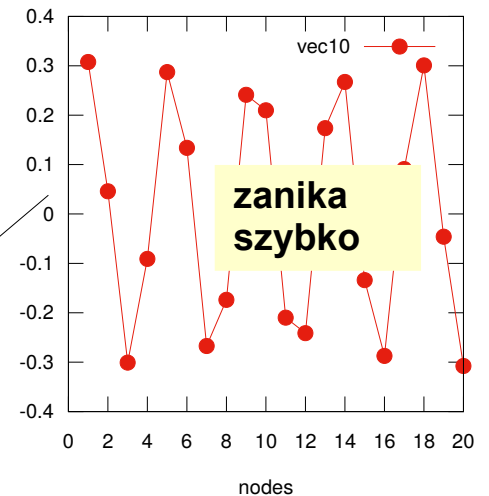
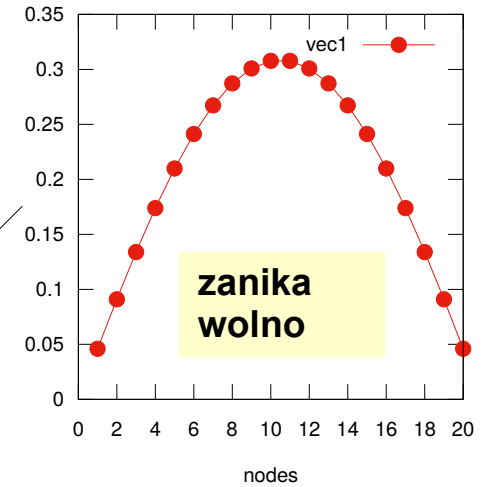
Przykład:

20 węzłów, $i=1,2,\dots,20$

$$\lambda_k = \cos \left(\frac{k\pi}{n+1} \right)$$

$$v_k(i) = \sin \left(\frac{ik\pi}{n+1} \right)$$

- i) metoda Jakobiego jest zbieżna
- ii) są wartości własne o modułach bliskich 1
→ metoda jest wolno zbieżna



Zbieżność metody Jakobiego

Tempo zbieżności (eliminacji błędu)

$$\mathbf{e}^{k+1} = \sum_i c_i \lambda_i^{k+1} \mathbf{v}_i$$

zależy od promienia spektralnego M

$$k = 1 \implies \rho(M) = \cos\left(\frac{\pi}{N+1}\right) \approx 1 - \left(\frac{\pi}{N+1}\right)^2 + \dots$$

→ im więcej węzłów N tym bardziej promień spektralny zbliża się do 1

→ zagęszczanie siatki spowoduje spowolnienie działania metody Jakobiego

Algorytm

for $k = 1$ *to* K_{max} *do*

for $i = 2$ *to* $N - 1$ *do*

$$x_i^{k+1} = \frac{1}{2}(x_{i-1}^k + x_{i+1}^k) + \frac{1}{2}\rho_i h^2$$

end do

$$\mathbf{x}^k \leftarrow \mathbf{x}^{k+1}$$

end do

Metoda Gaussa-Seidla

$$B = L + D \quad C = U$$

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

$$\mathbf{x}^{k+1} = -D^{-1}U\mathbf{x}^k - D^{-1}L\mathbf{x}^{k+1} + D^{-1}\mathbf{b}$$

$$M = -(L + D)^{-1}U$$

Dla laplasjanu (po wymnożeniu przez D^{-1}) otrzymamy:

$$\mathbf{x}^{k+1} = \frac{1}{2}U\mathbf{x}^k + \frac{1}{2}L\mathbf{x}^{k+1} - \frac{1}{2}\mathbf{b}$$

$$\begin{pmatrix} x_1^{k+1} \\ x_2^{k+1} \\ x_3^{k+1} \\ x_4^{k+1} \\ x_5^{k+1} \\ x_6^{k+1} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1^k \\ x_2^k \\ x_3^k \\ x_4^k \\ x_5^k \\ x_6^k \end{pmatrix} + \frac{1}{2} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1^{k+1} \\ x_2^{k+1} \\ x_3^{k+1} \\ x_4^{k+1} \\ x_5^{k+1} \\ x_6^{k+1} \end{pmatrix} + \frac{h^2}{2} \begin{pmatrix} \rho_1 \\ \rho_2 \\ \rho_3 \\ \rho_4 \\ \rho_5 \\ \rho_6 \end{pmatrix}$$

Średnia arytmetyczna:
sąsiada z prawej strony (**k**)
oraz z lewej strony (**k+1**)

Wzór iteracyjny w metodzie GS

$$\mathbf{x}^{k+1} = \frac{1}{2}U\mathbf{x}^k + \frac{1}{2}L\mathbf{x}^{k+1} - \frac{1}{2}\mathbf{b}$$

→ Ze względu na jednoprzekątniową postać L i U wzór działa jak podstawienie. Pozbywając się macierzy, dostaniemy wzór relaksacyjny.

→ wyliczając nową wartość w węźle i-tym korzystamy z wartości w węźle (i-1) już zmienionym w bieżącej iteracji

→ **metoda GS to relaksacja lokalna**
(rozwiązanie znajdziemy dysponując tylko jednym wektorem rozwiązań)

Algorytm GS

for $k = 1$ *to* K_{max} *do*

for $i = 2$ *to* $N - 1$ *do*

$$x_i = \frac{1}{2}(x_{i-1} + x_{i+1}) + \frac{1}{2}\rho_i h^2$$

end do

end do

Przykład. Macierz iteracji w metodzie GS

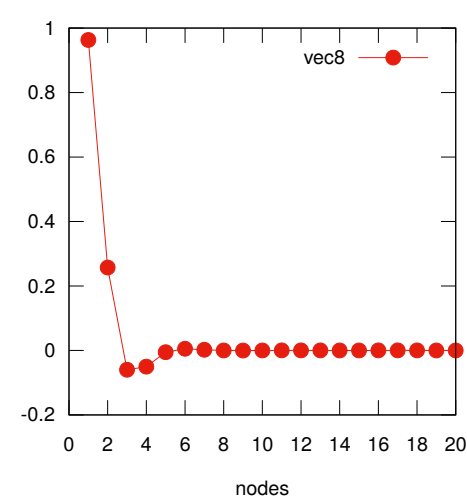
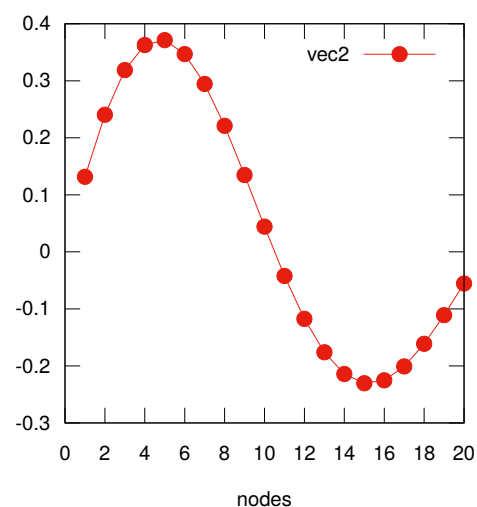
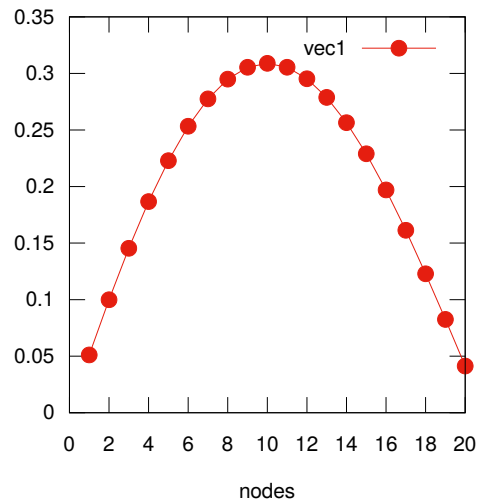
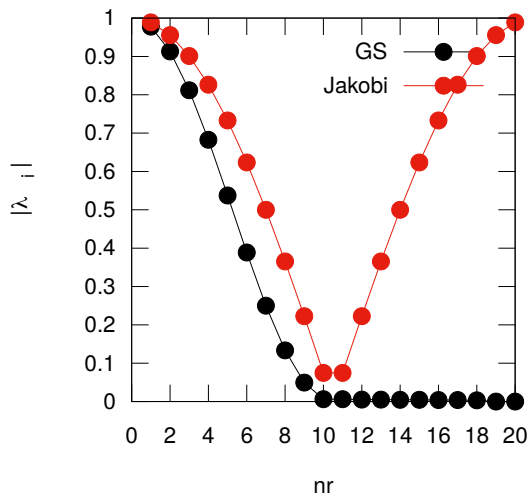
$$M = -(L + D)^{-1}U$$

$$M_{6 \times 6} = \begin{pmatrix} 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{4} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{16} & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} & 0 \\ 0 & \frac{1}{32} & \frac{1}{16} & \frac{1}{8} & \frac{1}{4} & \frac{1}{2} \\ 0 & \frac{1}{64} & \frac{1}{32} & \frac{1}{16} & \frac{1}{8} & \frac{1}{4} \end{pmatrix}$$

Jakie ma wartości i wektory własne?

Wartości i wektory własne macierzy iteracji M w metodzie Guassa-Seidla (dla laplasjanu w 1D), n=20

eigenvalues - GS-Jakobi



→ wartości własne są nieujemne

→ wektory własne odpowiadające najwyższym wartościom własnym są wolnozmiennie (w m. Jakobiego wektory te były szybko- i wolno-zmienne)

→ **m. Gaussa-Seidla ma własności wygładzające błąd (cechę tę wykorzystujemy w relaksacji wielosiatkowej)**

→ promień spektralny M w metodzie GS jest nieznacznie mniejszy niż w m. Jakobiego, ale dla k=1000

$$\rho_J = 0.98883, \quad \rho_J^{1000} = 1.3 \cdot 10^{-5}$$

$$\rho_{GS} = 0.97779, \quad \rho_{GS}^{1000} = 1.76 \cdot 10^{-10}$$

Co oznacza zwrot: „własności wygładzające” w praktyce?

Przykład.

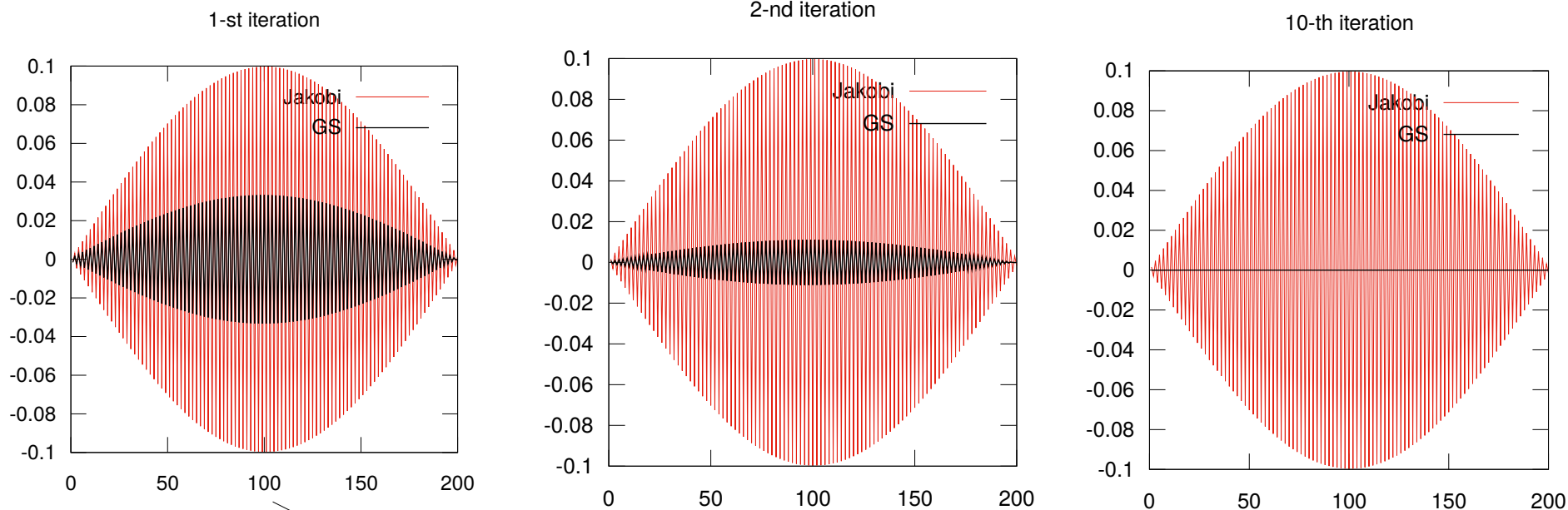
Porównanie szybkości tłumienia błędu w metodach GS i Jakobiego w przypadku równania Laplace'a (brak niejednorodności).

Generujemy wektor szybko-zmienny (najmniejsza wartość własna macierzy iterującej Jakobiego), które stanowi rozwiązanie początkowe

$$\nabla^2 x = 0$$

$$n = 0, \quad WB : x_0 = x_{N+1} = 0$$

$$v_k(i) = \sin\left(\frac{ik\pi}{N+1}\right) \implies \boxed{\mathbf{x}^0 = \mathbf{v}^N}$$



→ widać że m. GS bardzo szybko tłumí wysokozmienny błąd
→ M Jakobiego tej własności nie posiada

Tempo zbieżności relaksacji GS możemy kontrolować dodając do równania

$$\mathbf{x}^{k+1} = \frac{1}{2}U\mathbf{x}^k + \frac{1}{2}L\mathbf{x}^{k+1} - \frac{1}{2}\mathbf{b}$$

parametr zbieżności $\omega \in (0, 2)$

$$\mathbf{x}^{k+1} = (1 - \omega) \cdot \mathbf{x}^k + \omega \cdot \left(\frac{1}{2}U\mathbf{x}^k + \frac{1}{2}L\mathbf{x}^{k+1} - \frac{1}{2}\mathbf{b} \right)$$

Inaczej: mieszamy stare i nowe rozwiązanie

→ $\omega < 1$ **podrelaksacja** (obniżamy tempo zbieżności)

→ $\omega = 1$ zwykła relaksacja

→ $\omega > 1$ **nadrelaksacja** (tu powinniśmy uzyskać przyśpieszenie obliczeń)

(dowód zbieżności metody GS dla $\omega \in (0, 2)$ pokazany na wykładzie z Metod Numerycznych dla układu równań liniowych)

Pytanie: kiedy zatrzymać relaksację?

Kiedy zatrzymać relaksację?

Jakość rozwiązania równania Poissona możemy sprawdzić na dwa sposoby:

1) badając normę euklidesową wektora reszt

$$Ax = b$$
$$r = b - Ax$$

2) licząc całkę działania dla pola elektrycznego

$$S = \int_{\Omega} \frac{1}{2} (\nabla u)^2 - \rho \cdot u dx$$

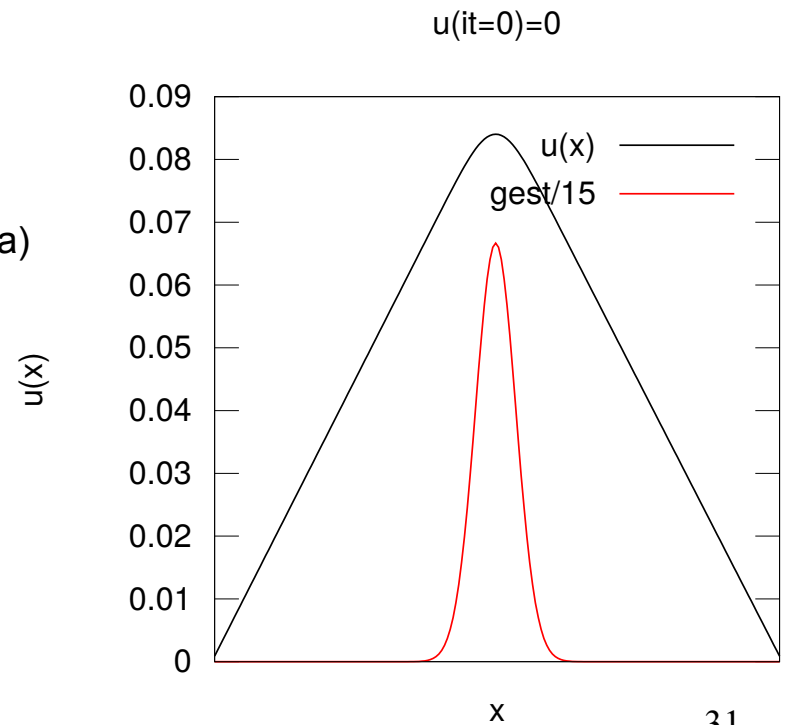
→ **całka działania osiąga minimum dla rozwiązania dokładnego (zasada wariacyjna Rayleigha-Ritza)**

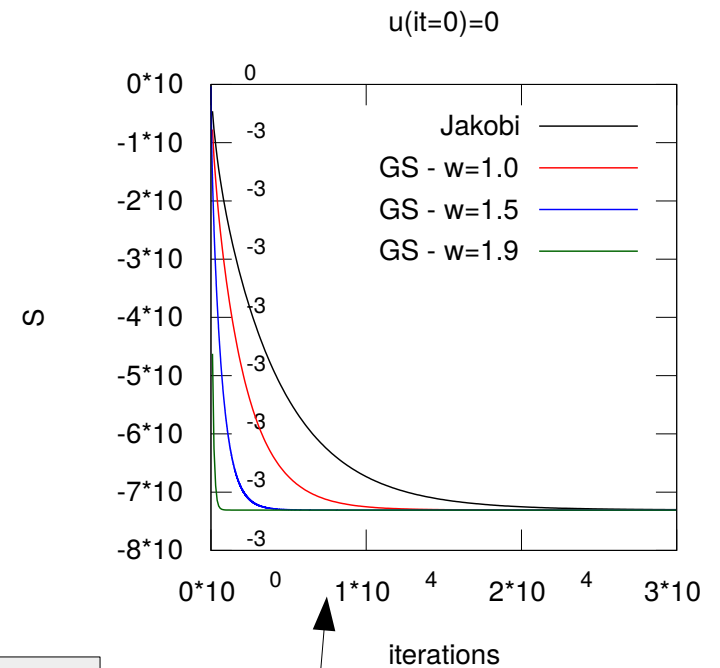
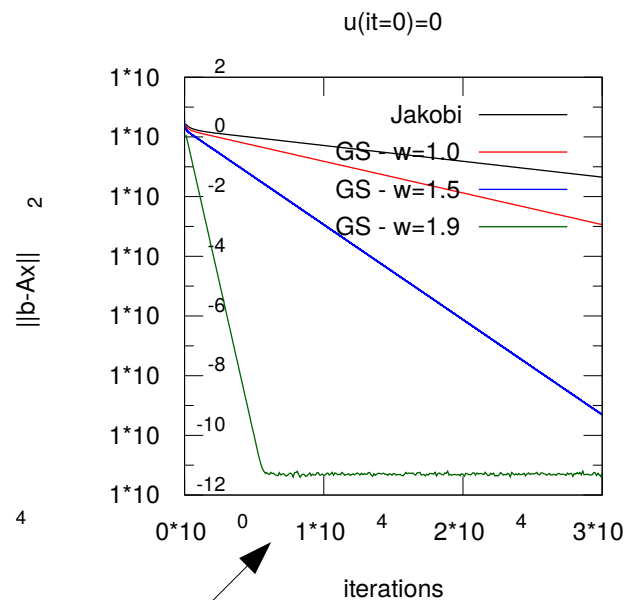
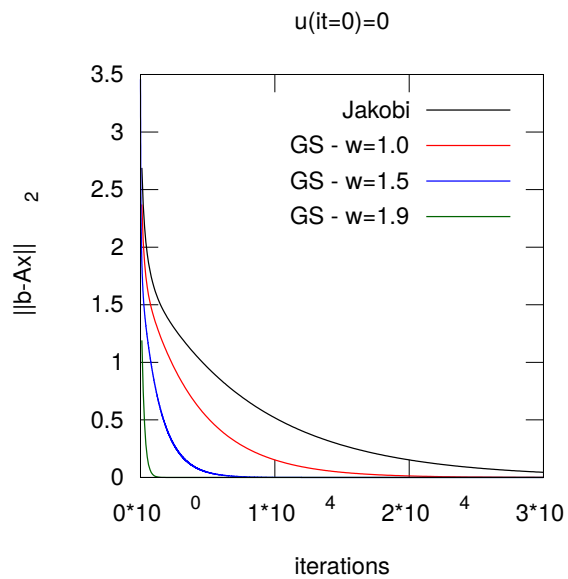
Przykład. Rozwiązujemy metodą relaksacji (Jakobi, GS, nadrelaksacja) równanie Poissona w 1D

$$\nabla^2 u(x) = -\rho(x), \quad x \in [-1, 1]$$

$$WB. : \quad u(-1) = u(1) = 0$$

$$\rho(x) = \exp^{-100 x^2}$$





Dla normy wektora reszt pojawia się problem, który jest wynikiem dokładności rozwiązania numerycznego i precyzji obliczeń
 → **trudno określić warunek zakończenia relaksacji**

Jeśli cała działka osiąga minimum to w kolejnych iteracjach już się nie zmienia
 → **to wygodny warunek zatrzymania relaksacji.**

→ m. Jakobiego najwolniejsza

→ nadrelaksacja tym szybsza im większa wartość parametru zbieżności (ale nie jest to regułą – zależy często od WB)

→ co lepsze do warunku zakończenia?: $\|r\|_2$ czy całka działania?

Jak jeszcze można przyspieszyć (zoptymalizować) proces relaksacji?

Np. stosując:

1) metodę zagęszczania siatki

2) relaksację wielosiatkową

Zagęszczanie siatki – najprostsze.

1) Znajdujemy rozwiązanie na najrzadszej siatce (czerwone węzły, krok na siatce $k=4$)

stosując wzór relaksacyjny

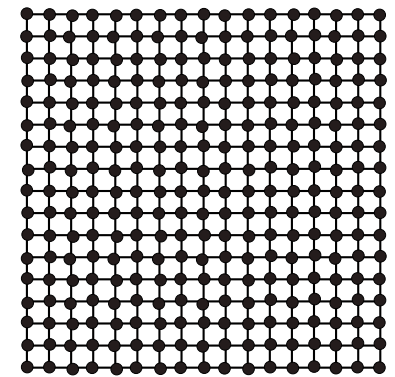
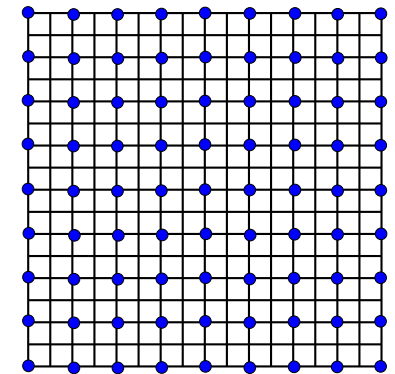
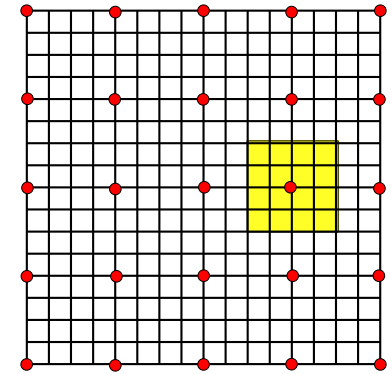
$$x_{i,j} = (1 - \omega)x_{i,j} + \frac{\omega}{4} (x_{i+k,j} + x_{i-k,j} + x_{i,j+k} + x_{i,j-k} + k^2 \cdot h^2 \cdot \tilde{\rho}_{i,j})$$

oraz uśredniając gęstość w komórce otaczającej aktualny węzeł (**żółty obszar**)

$$c_\alpha = \begin{cases} 0.5, & \alpha = \pm k/2 \\ 1, & -k/2 < \alpha < k/2 \end{cases}$$

$$w = \sum_{\alpha=-k/2}^{k/2} \sum_{\beta=-k/2}^{k/2} c_\alpha c_\beta$$

$$\tilde{\rho}_{i,j} = \frac{1}{w} \sum_{\alpha=-k/2}^{k/2} \sum_{\beta=-k/2}^{k/2} c_\alpha c_\beta \rho_{i+\alpha, j+\beta}$$



2) Przechodzimy na siatkę o dwukrotnie mniejszym oczku siatki, wartości w nowych węzłach (niebieskie) interpolujemy liniowo wartościami z węzłów sąsiednich

Siatka rzadka: **czerwone węzły**

Siatka gęsta: **niebieskie węzły**

$$u_{i,j}$$

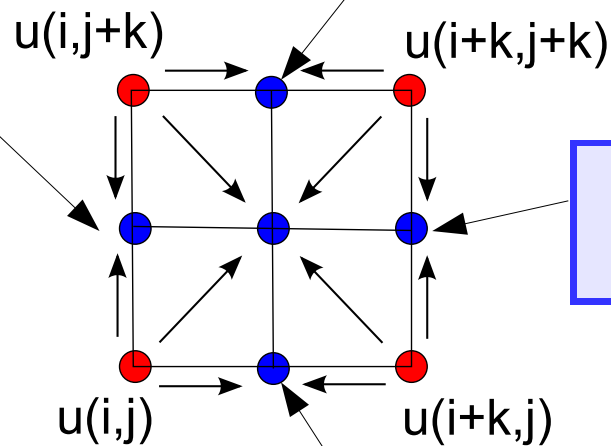
$$u_{i+k,j}$$

$$u_{i,j+k}$$

$$u_{i+k,j+k}$$

$$u_{i,j+\frac{k}{2}} = \frac{1}{2} (u_{i,j} + u_{i,j+k})$$

$$u_{i+\frac{k}{2},j+k} = \frac{1}{2} (u_{i,j+k} + u_{i+k,j+k})$$



$$u_{i+k,j+\frac{k}{2}} = \frac{1}{2} (u_{i+k,j} + u_{i+k,j+k})$$

$$u_{i+\frac{k}{2},j} = \frac{1}{2} (u_{i,j} + u_{i+k,j})$$

węzeł środkowy:

$$u_{i+\frac{k}{2},j+\frac{k}{2}} = \frac{1}{4} (u_{i,j} + u_{i+k,j} + u_{i,j+k} + u_{i+k,j+k})$$

3) na rzadszej siatce relaksujemy równanie

4) powtarzamy operacje (2) i (3) na kolejnych gęstszych siatkach

Przykład.

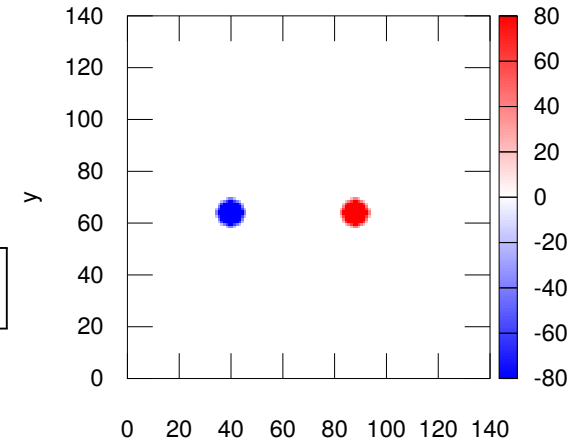
$$\nabla^2 u(x, y) = -\rho(x, y)$$

$$u_{0,j} = +0.1, u_{N,j} = -0.1, u(i, 0) = 0, u_{i,N} = 0$$

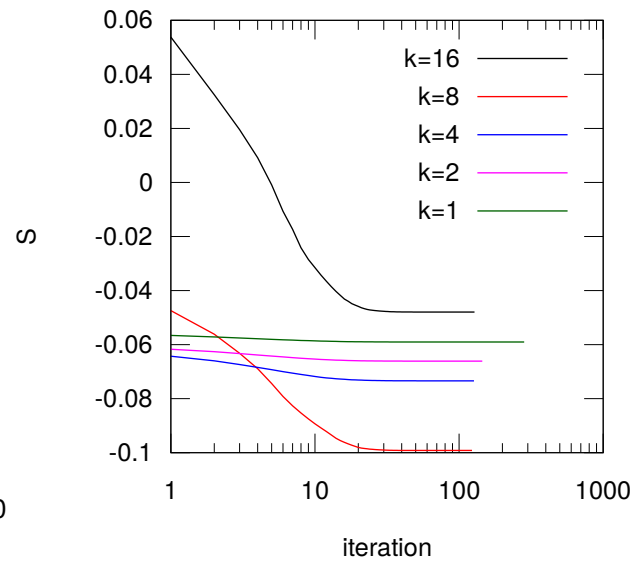
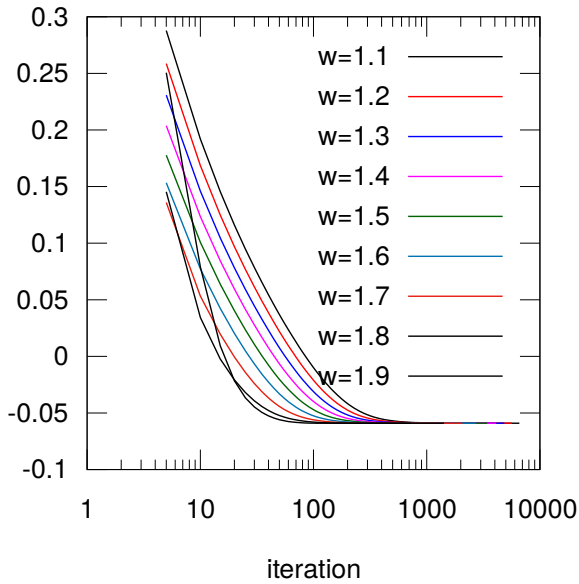
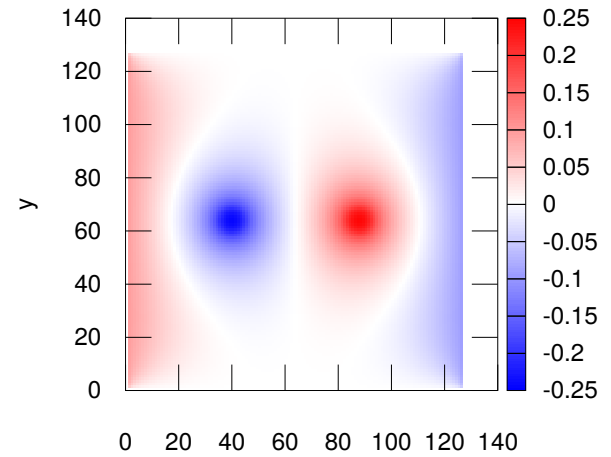
Overrelaxation

overrelaxation - multigrid

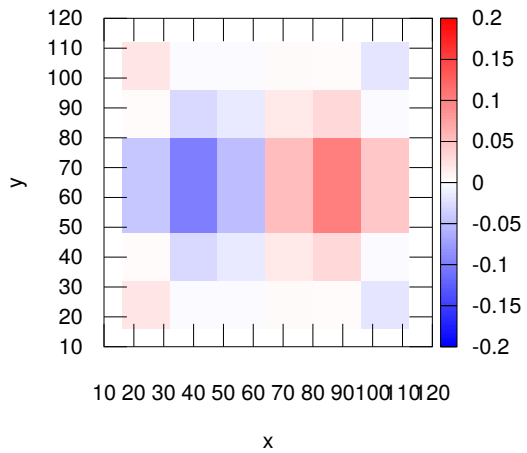
gestosc



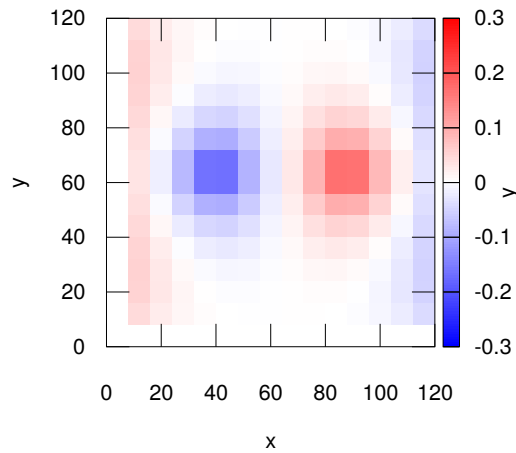
x
k=1



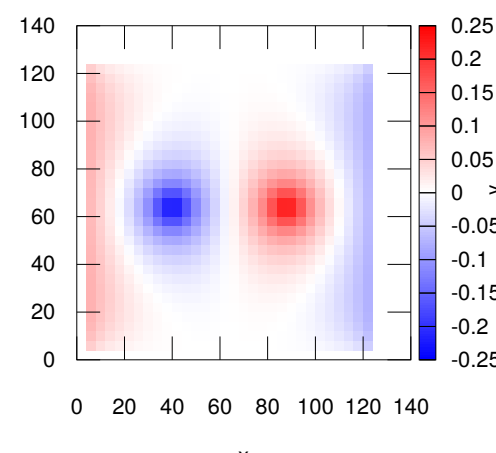
k=16



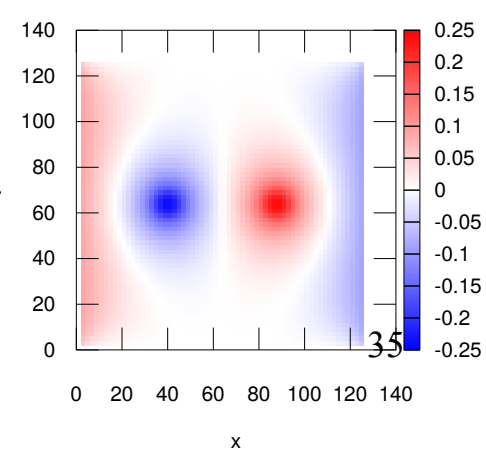
k=8



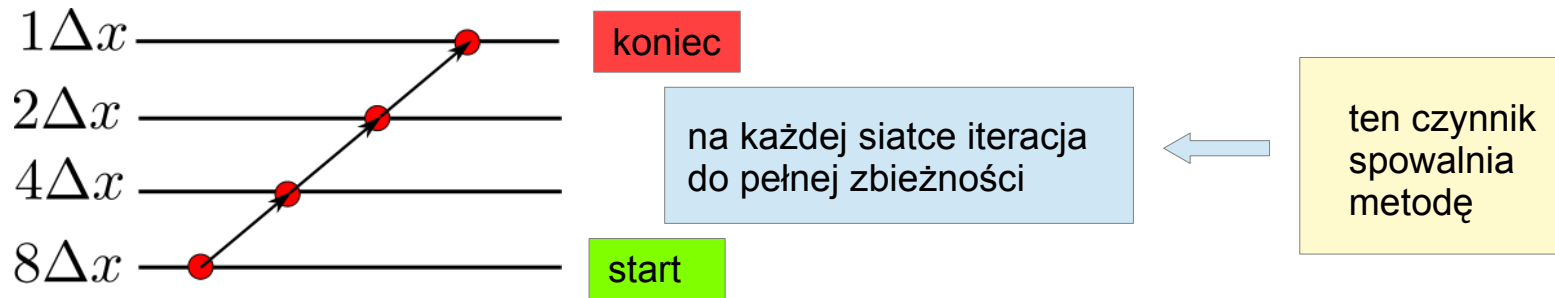
k=4



x
k=2



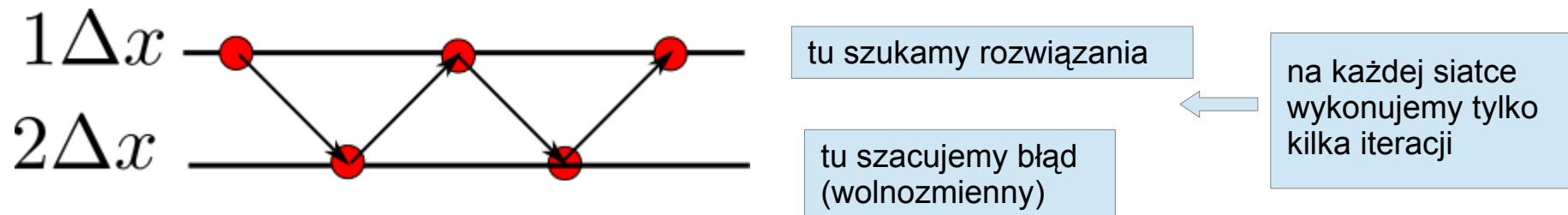
schemat prostego zagęszczania siatki



Relaksacja wielosiatkowa

Zamiast jednokierunkowego zagęszczania siatki możemy użyć schematu wielosiatkowego (*multigrid method*)

np. najprostszego dwusiatkowego



Procedura iteracji dwusiatkowej dla równania

$$Ax = b \implies x^{k+1} = Mx^k + c$$

1. na siatce (Δx) wykonujemy n_1 iteracji, dostajemy:

$$x(\Delta x) = x_{dok}(\Delta x) + e(\Delta x)$$

2. liczymy wektor reszt (Δx):

$$r(\Delta x) = Ax(\Delta x) - b = \underbrace{(Ax_{dok} - b)}_{=0} + Ae(\Delta x)$$



$$Ae(\Delta x) = r(\Delta x)$$

Uwaga: wektor błędów $e(\Delta x)$ zawiera składowe wolnozmiennne - znajdziemy go na siatce ($2\Delta x$)

3. rzutujemy wektor reszt na rzadszą siatkę używając (macierzowego) operatora restrykcji

$$r(2\Delta x) = R(\Delta x \rightarrow 2\Delta x)r(\Delta x)$$

4. teraz znajdujemy przybliżone rozwiązanie dla $e(2\Delta x)$ tj. eliminujemy wpływ błędów wolnozmiennnych

$$Ae(2\Delta x) = r(2\Delta x)$$

- wykonujemy tylko n_2 iteracji

5. Wektor błędów $e(2\Delta x)$ rzutujemy na siatkę gęstszą używając (macierzowego) operatora prolongacji

$$r(\Delta x) = P(2\Delta x \rightarrow \Delta x)r(2\Delta x)$$

6. poprawiamy przybliżone rozwiązanie na siatce (Δx)

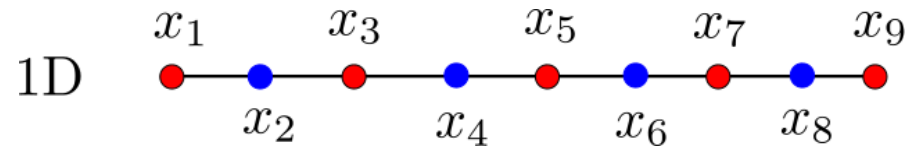
$$\tilde{x}(\Delta x) = x(\Delta x) - e(\Delta x)$$

Uwaga: wektory $x(\Delta x)$ oraz $e(\Delta x)$ stanowią tylko przybliżenia ich dokładnych odpowiedników dlatego procedurę (kroki 1-6) powtarzamy aż do uzyskania zbieżności np. funkcjonału dla pola elektrycznego

Operator restrykcji/prolongacji – jak je skonstruować?

Najpierw zobaczmy co chcemy uzyskać

Przykład dla siatki 1D



restrykcja:

zostawiamy tylko
węzły niebieskie

$$\begin{bmatrix} x_2 \\ x_4 \\ x_6 \\ x_8 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix}$$

prolongacja:

generujemy węzły czerwone x_3 , x_5 i x_7

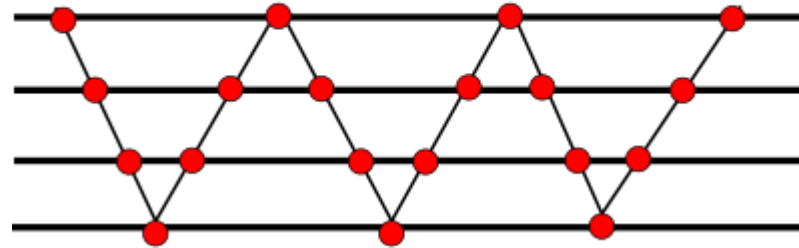
(węzły brzegowe x_1 i x_9 – bez zmian
np. ze względu na WB Dirichleta)

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_4 \\ x_6 \\ x_8 \\ x_9 \end{bmatrix}$$

W relaksacji wielosiatkowej możemy stosować różne schematy

V - cycle

4-siatkowa relaksacja
typu V-cycle



Pełna relaksacja wielosiatkowa

- szukamy rozwiązania na siatce (Δx^*k) [1]
- rozwiązania rzutujemy je na siatkę $(\Delta x^*k/2)$ [2] i wykonujemy $N(k/2)$ iteracji
- szukamy błędu na siatce (Δx^*k) [3]
- poprawiamy rozwiązanie na siatce $(\Delta x^*k/2)$ [4]
- rzutujemy rozwiązanie na siatkę $(\Delta x^*k/4)$ [5] i wykonujemy $N(k/4)$ iteracji
- szukamy błędu na siatce $(\Delta x^*k/2)$ [6] oraz (Δx^*k) [7]
- poprawiamy błąd na siatce $(\Delta x^*k/2)$ [8] oraz rozwiązanie na siatce $(\Delta x^*k/4)$ [9]
- rzutujemy rozwiązanie na siatkę $(\Delta x^*k/8)$ [10] i relaksujemy je $N(k/8)$ razy
- wykonujemy jeden do kilku pełnych V-cykli

full multigrid

