Preconditioning

Karl Meerbergen

KU Leuven

October 9, 2013

Literature

- James Demmel, Applied numerical Linear Algebra, SIAM, 1997.

  Gives a good overview of preconditioning techniques.

- Anne Greenbaum, Iterative methods for solving linear systems, SIAM, 1997.

  Gives an excellent overview of Krylov methods. The book also briefly discusses preconditioning techniques.

1D Poisson equation

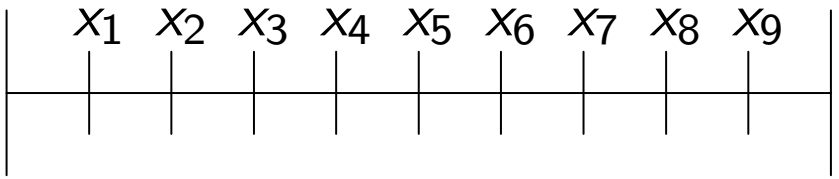
- The equation:

$$\begin{aligned} -\frac{d^2u(x)}{dx^2} &= f(x) \quad , \quad 0 < x < 1 \\ u(0) = u(1) &= 0 \end{aligned}$$

- Discretization: compute the solution in points

$$x_j = hj \quad , \quad j = 1, \ldots, N$$

with $h = (N+1)^{-1}$.



1D Poisson equation

- Finite difference

$$\frac{d^2u}{dx^2}(x_j) \simeq \frac{u(x_{j+1}) - 2u(x_j) + u(x_{j-1})}{h^2}$$

- Filling in the finite difference in

$$\begin{aligned} -\frac{d^2u(x)}{dx^2} &= f(x) \quad , \quad 0 < x < 1 \\ u(0) = u(1) &= 0 \end{aligned}$$

and evaluation in N points:

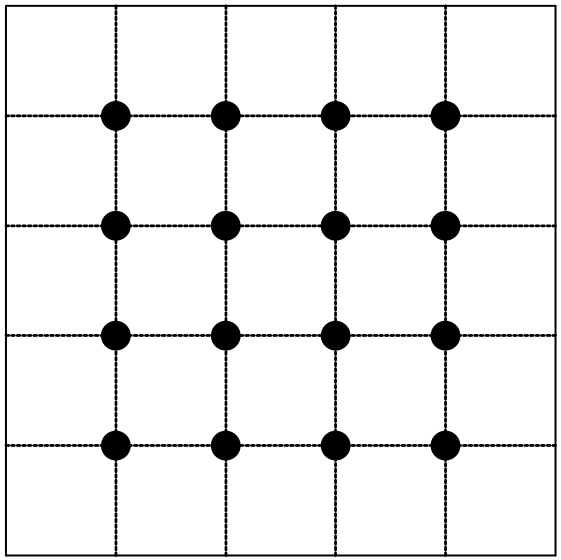
$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix}\begin{pmatrix} u_1 \\ \vdots \\ \vdots \\ u_N \end{pmatrix} = \begin{pmatrix} f_1 \\ \vdots \\ \vdots \\ f_N \end{pmatrix}$$

2D Poisson equation

- The equation:

$$\begin{aligned} -\frac{\partial^2 u(x,y)}{\partial x^2} - \frac{\partial^2 u(x,y)}{\partial y^2} &= f(x,y), \quad 0 < x, y < 1 \\ u(0,y) = u(1,y) = u(x,0) = u(x,1) &= 0, \quad 0 < x, y < 1 \ . \end{aligned}$$

- Discretization: N points in x direction and y direction: $u(x_i, y_j)$, $i, j = 0, \ldots, N+1$
  with $u(x_0, y_j) = u(x_{N+1}, y_j) = u(x_i, y_0) = u(x_i, y_{N+1}) = 0$.



2D Poisson equation

- Finite differences:

$$\frac{\partial^2 u}{\partial x^2}(x_j, y_i) \simeq \frac{u(x_{j+1}, y_i) - 2u(x_j, y_i) + u(x_{j-1}, y_i)}{h^2}$$
$$\frac{\partial^2 u}{\partial y^2}(x_j, y_i) \simeq \frac{u(x_j, y_{i+1}) - 2u(x_j, y_i) + u(x_j, y_{i-1})}{h^2}$$

- Filling-in in equation (lexicographical ordering):

$$\left[\begin{array}{cccc|cccc|cccc|cccc}
4 & -1 & & & -1 & & & & & & & & & & & \\
-1 & 4 & -1 & & & -1 & & & & & & & & & & \\
 & -1 & 4 & -1 & & & -1 & & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & & \\
\hline
-1 & & & & 4 & -1 & & & -1 & & & & & & & \\
 & -1 & & & -1 & 4 & -1 & & & -1 & & & & & & \\
 & & -1 & & & -1 & 4 & -1 & & & -1 & & & & & \\
 & & & -1 & & & -1 & 4 & & & & -1 & & & & \\
\hline
 & & & & -1 & & & & 4 & -1 & & & -1 & & & \\
 & & & & & -1 & & & -1 & 4 & -1 & & & -1 & & \\
 & & & & & & -1 & & & -1 & 4 & -1 & & & -1 & \\
 & & & & & & & -1 & & & -1 & 4 & & & & -1 \\
\hline
 & & & & & & & & -1 & & & & 4 & -1 & & \\
 & & & & & & & & & -1 & & & -1 & 4 & -1 & \\
 & & & & & & & & & & -1 & & & -1 & 4 & -1 \\
 & & & & & & & & & & & -1 & & & -1 & 4
\end{array}\right]
\left(\begin{array}{c}
u_{11} \\ u_{12} \\ u_{13} \\ u_{14} \\ \hline
u_{21} \\ u_{22} \\ u_{23} \\ u_{24} \\ \hline
u_{31} \\ u_{32} \\ u_{33} \\ u_{34} \\ \hline
u_{41} \\ u_{42} \\ u_{43} \\ u_{44}
\end{array}\right)$$

2D Poisson equation

- The linear system can also be denoted as follows.
  Let L be the 1D Laplacian matrix (tridiagonal), then the 2D equation becomes
  $$(I \otimes L + L \otimes I)u = f$$
  where $\otimes$ denotes the Kronecker product.
- If we write u f as matrices U and F (rows correspond to y coordinates and columns to x coordinates), then we have
  $$LU + UL = F$$
  This is called a Lyapunov/Sylvester equation.

3D Poisson equation

- Domain: $\Omega = \{(x, y, z) : 0 < x, y, z < 1\}$
- Laplace operator: $\nabla^2 = -\frac{\partial^2 u(x,y,z)}{\partial x^2} - \frac{\partial^2 u(x,y,z)}{\partial x^2} - \frac{\partial^2 u(x,y,z)}{\partial z^2}$
- The equation:

$$\begin{aligned} -\nabla^2 &= f(x, y, z), \quad (x, y, z) \in \Omega \\ u(x, y, z) = &= 0, (x, y, z) \in \partial\Omega \end{aligned}$$

- Discretization similar to 1D/2D: N points in each direction: N^3 unknowns.
- Linear system:

$$(I \otimes I \otimes L + I \otimes L \otimes I + L \otimes I \otimes I)u = f$$

Example

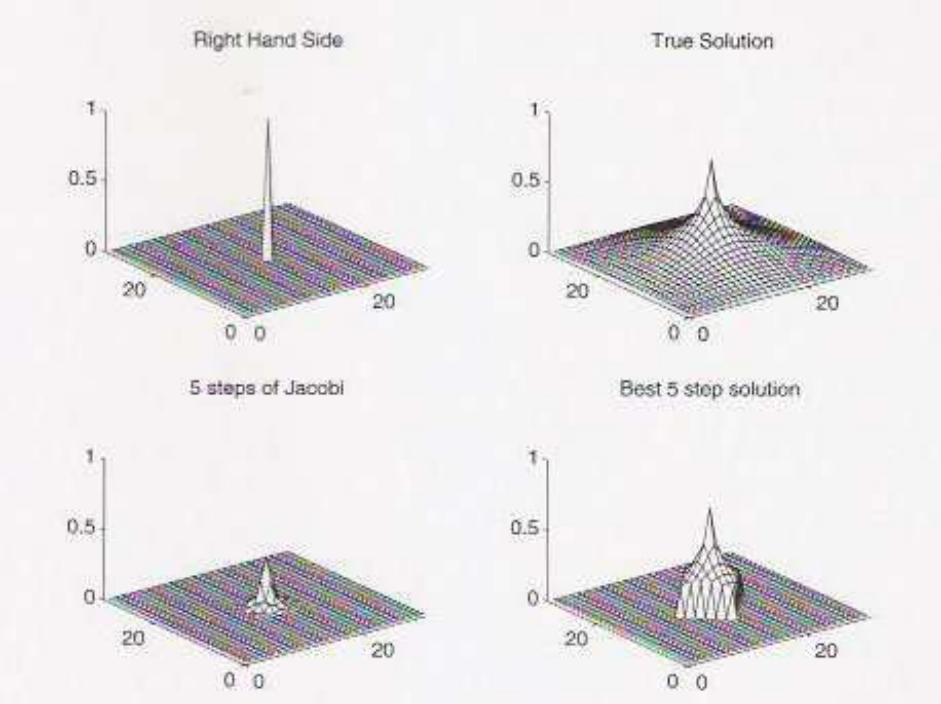
- Comparison of finite difference discretizations for 1D, 2D and 3D Poisson equation.

Table: Results for the solution of Poisson's equation by a multi-frontal direct method (MUMPS). Timings obtained in 2007 on a Dell Dimension 4550 using GCC.

	N	n	core memory (MB)	wall clock time (s)
1D	250000	250000	72	3
2D	500	250000	195	6
3D	63	250047	2020	108

Iterative solver

- Krylov method performs one matrix vector product on each iteration.
- For Poisson-equation: matrix vector product is a local operation:



- Conclusion: N iterations are needed to transport information from one side of the mesh to the other.
- Solving linear systems arising from PDE's should therefore be solved by methods that transport information in a much faster way.

What is preconditioning?

1. 'Krylov methods converge slower when the condition number is large'.
   Preconditioning improves the condition number
   This definition is far too pessimistic.
2. The preconditioner M is an approximation to A, so that

$$M^{-1}A \approx I$$

3. M maps the eigenvalues of $M^{-1}A$ close to one.

where $M^{-1}A \cdot x$ is easy/cheap to compute and $A^{-1} \cdot x$ is very hard.

Definition of preconditioner

A preconditioner M is a matrix so that a linear system with $M^{-1}A$ is easier to solve by a Krylov method than a linear system with A.

Types of preconditioning

- Preconditioning

$$(M^{-1}A)x = (M^{-1}b)$$

- Postconditioning

$$(AM^{-1})y = b \quad \text{and} \quad x = M^{-1}y$$

- Both

$$(M_1^{-1}AM_2^{-1})y = (M_1^{-1}b) \quad \text{and} \quad x = M_2^{-1}y$$

We only consider preconditioning.

Choice of Krylov methods

- Symmetric positive definite linear systems:
  - Conjugate gradients
- Symmetric (Hermitian) systems:
  - MINRES
- Unsymmetric (general) systems:
  - BiCGStab: good, reliable method
  - IDR(s): Newest promising Krylov method
  - GMRES: expensive storage, restart needed (Gram-Schmidt is expensive; for hard problems, use Gram-Schmidt with reorthogonalization, as for the Arnoldi eigenvalue solver)

Overview of preconditioners

- Splitting methods (Jacobi, Gauss-Seidel, SOR, SSOR)
- Incomplete factorization
- Approximate inverse
- Multigrid
- Domain decomposition
- Application dependent: shifted preconditioners, KKT structured preconditioners, Fast Fourier Transform based.

Often, preconditioners work as such without Krylov 'accelerators' (This is called Richardson iteration)

Reference application

- 1D Laplacian (Poisson)

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix}$$

  with $h = (n+1)^{-1}$.
- Eigenvalues are

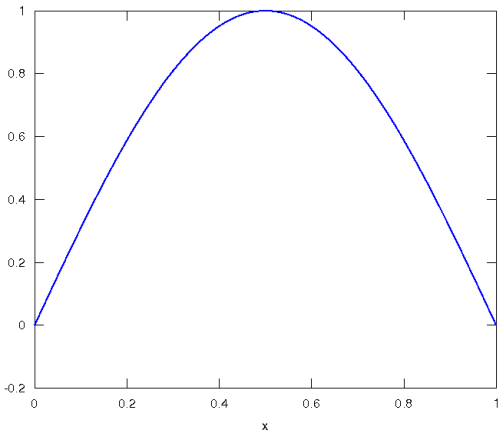
$$\lambda_j = 2h^{-2}(1 - \cos(\pi h j)) \quad , \quad j = 1, \ldots, n$$

- For small j, $\lambda_j \simeq \pi^2 j^2$
- For large n, $\lambda_n \simeq 4h^{-2}$
- Krylov methods converge slowly since the spectrum is spread out over $[\pi^2, 4h^{-2}]$, especially hard for small h.
- Eigenvector is discretization of the function

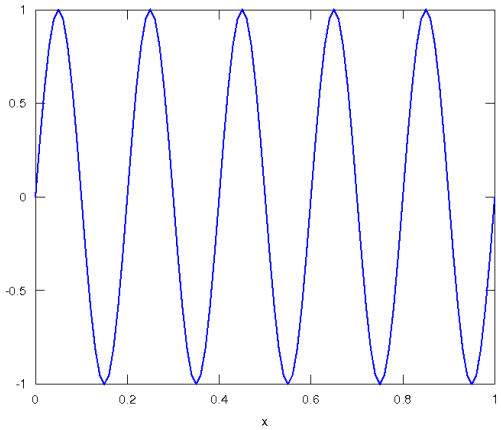
$$u_j = \sin(\pi j x) \quad , \quad 0 < x < 1$$

Reference application

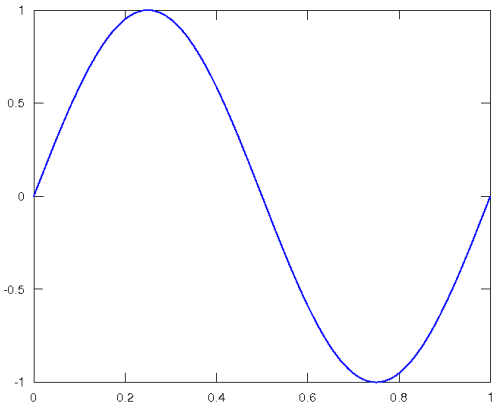
- Eigenvalue
  $\lambda_1 = 2h^{-2}(1 - \cos \pi h) \simeq \pi^2$



- Eigenvalue
  $\lambda_{10} = 2h^{-2}(1 - \cos 10\pi h)$



- Eigenvalue $\lambda_2 =$
  $2h^{-2}(1 - \cos 2\pi h) \simeq 4\pi^2$



Preliminaries

- Spectral radius $\rho(A) = \max |\lambda|$, maximum taken over all eigenvalues of A.
- $\rho(A) \leq \|A\|$ for any norm.

Jacobi iteration

- Iterative method:
  - Given x
  - Compute next iterate $\widetilde{x}$
- Jacobi iteration: solve ith element of x from the ith equation by freezing the values of the other equations:

$$\begin{cases} a_{1,1}\widetilde{x}_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \rightarrow \widetilde{x}_1 \\ \vdots \\ a_{i,1}x_1 + \cdots + a_{i,i}\widetilde{x}_i + \cdots + a_{i,n}x_n = b_i \rightarrow \widetilde{x}_i \\ \vdots \\ a_{n,1}x_1 + \cdots + a_{n,n-1}x_{n-1} + a_{n,n}\widetilde{x}_n = b_n \rightarrow \widetilde{x}_n \end{cases}$$

- Jacobi is a substitution method.
- This process is repeated until convergence

Jacobi iteration

for $i = 1, \ldots, n$ **do**

$$\widetilde{x}_i = \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j \right) / a_{ii}$$

end for

- Cost = cost of matrix vector multiply
- All variables can be updated in any order
- This means that the method is easy to parallelize.

Jacobi iteration

- Let $A = L + D + U$ with L the strictly lower triangular part of A and U the strictly upper triangular part of A.
- Jacobi iteration:

$$D\tilde{x} = b - (L + U)x$$

- From $Ax = b$, we have

$$Dx = b - (L + U)x$$

- Using the idea of a substitution method, we arrive at the method of Jacobi.

Convergence of Jacobi iteration

- $Ax = b$, iteration i: $x^{(i)}$.
- Error

$$\begin{aligned} e^{(i)} &= x - x^{(i)} \\ &= x - D^{-1}(b - (L+U)x^{(i-1)}) \\ &= -D^{-1}(L+U)e^{(i-1)} \end{aligned}$$

  (Note that $x = D^{-1}(b - (L+U)x)$.)
- Iteration matrix

$$\begin{aligned} G &= -D^{-1}(L+U) \\ &= I - D^{-1}A \end{aligned}$$

- Convergence:

$$e^{(i)} = G^i e^{(0)}$$

  converges to zero, when spectral radius of G, $\rho(G)$, is smaller than one.

Convergence of Jacobi iteration

- Convergence:

$$e^{(i)} = G^i e^{(0)}$$

converges to zero, when $\rho(G)$ is smaller than one.

Theorem (Sufficient condition for convergence)

When A is diagonally dominant, $\rho(G) < 1$.

The proof follows from the Gerschgorin theorem.

Theorem (Sufficient condition for convergence)

When A is weakly diagonally dominant and irreducible[a], $\rho(G) < 1$.

[a] block upper triangular form by row permutations

Gauss-Seidel iteration

- Iterative method:
  - Given x
  - Compute next iterate $\widetilde{x}$
- Gauss-Seidel iteration: solve ith element of x from the ith equation by using the most recent values of x for the other unknowns

$$\begin{cases} a_{1,1}\widetilde{x}_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n = b_1 \rightarrow \widetilde{x}_1 \\ \vdots \\ a_{i,1}\widetilde{x}_1 + \cdots + a_{i,i}\widetilde{x}_i + \cdots + a_{i,n}x_n = b_i \rightarrow \widetilde{x}_i \\ \vdots \\ a_{n,1}\widetilde{x}_1 + \cdots + a_{n,n-1}\widetilde{x}_{n-1} + a_{n,n}\widetilde{x}_n = b_n \rightarrow \widetilde{x}_n \end{cases}$$

- Gauss-Seidel is a substitution method.
- This process is repeated until convergence

Gauss-Seidel iteration

Let $\widetilde{x} = x$.
for $i = 1, \ldots, n$ **do**

$$\widetilde{x}_i = \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} \widetilde{x}_j \right) / a_{ii}$$

end for

- Cost = cost of matrix vector multiply
- The order of the update of variables depends on the ordering of the rows/columns of the matrix.
- This algorithm is sequential: hard to parallelize.

Gauss-Seidel iteration

- Let $A = L + D + U$ with L the strictly lower triangular part of A and U the strictly upper triangular part of A.
- Gauss-Seidel iteration:

$$(L + D)\tilde{x} = b - Ux$$

- From $Ax = b$, we have

$$(L + D)x = b - Ux$$

- Using the idea of a substitution method, we arrive at the method of Gauss-Seidel.

Convergence of Gauss-Seidel iteration

- $Ax = b$, iteration i: $x^{(i)}$.
- Error

$$\begin{aligned} e^{(i)} &= x - x^{(i)} \\ &= x - (L+D)^{-1}(b - Ux^{(i-1)}) \\ &= -(L+D)^{-1}Ue^{(i-1)} \end{aligned}$$

  (Note that $x = (L+D)^{-1}(b - Ux)$.)
- Iteration matrix

$$\begin{aligned} G &= -(L+D)^{-1}U \\ &= I - (L+D)^{-1}A \end{aligned}$$

- Convergence:

$$e^{(i)} = G^i e^{(0)}$$

  converges to zero, when spectral radius of G, $\rho(G)$, is smaller than one.

Convergence of Gauss-Seidel iteration

- Convergence:

$$e^{(i)} = G^i e^{(0)}$$

converges to zero, when $\rho(G)$ is smaller than one.

Theorem (Sufficient condition for convergence)

When A is diagonally dominant, $\|G_{GS}\|_\infty \leq \|G_J\|_\infty < 1$.

Theorem (Sufficient condition for convergence)

When A is weakly diagonally dominant and irreducible, $\rho(G_{GS}) < \rho(G_J) < 1$.

Splitting methods

- General splitting methods:
- $A = M - N$ is a splitting iff M invertible
- Solution:

$$\begin{aligned} x^{(i+1)} &= M^{-1}(b + Nx^{(i)}) \\ &= x^{(i)} + M^{-1}(b - Ax^{(i)}) \end{aligned}$$

- Error:

$$\begin{aligned} e^{(i+1)} &= x - x^{(i+1)} \\ &= M^{-1}(Mx - b - Nx^{(i)}) \\ &= M^{-1}(Mx - Mx + Nx - Nx^{(i)}) \\ &= M^{-1}Ne^{(i)} \\ &= Ge^{(i)} \end{aligned}$$

- Iteration matrix $G = M^{-1}N = I - M^{-1}A$.
- Sufficient condition for convergence: $\rho(G) < 1$
- Jacobi: $M = D$, $N = -L - U$
- Gauss-Seidel $M = D + L$, $N = -U$

Example: 1D Poisson

$$\frac{1}{h^2}\begin{bmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{bmatrix}$$

- Laplacian matrix has $h^{-2}2$ on the main diagonal and maximum two times the value h^{-2} on the off diagonal elements.
- The matrix is therefore not diagonally dominant.
- Jacobi iteration:

$$u^{(i+1)} = u^{(i)} + \frac{h^2}{2}(b - Au^{(i)})$$

- For the error we have

$$e^{(i+1)} = e^{(i)} - \frac{h^2}{2}Ae^{(i)} = \left(I - \frac{h^2}{2}A\right)e^{(i)}$$

Example: 1D Poisson

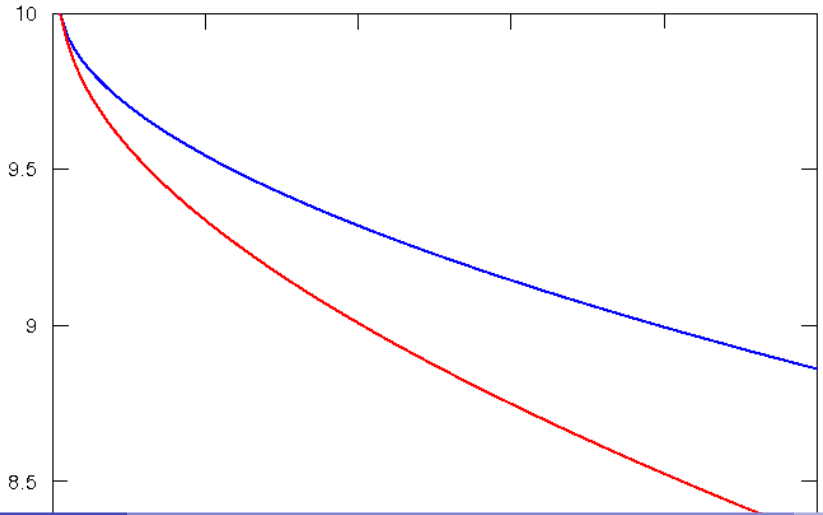
- The eigenvalues of $G = I - h^2 A/2$ are

$$1 - h^2 \lambda_j / 2 = \cos(\pi h j)$$

- Hence $\rho(G) = \cos(\pi h) \approx 1 - \frac{\pi^2}{2(n+1)^2} < 1$.
- Asymtotic convergence for k iterations:

$$\rho(G)^k \approx 1 - k \frac{\pi^2}{2(n+1)^2}$$

  so, convergence is very slow.
- Convergence is not linear: we will come back to this later



Successive overrelaxation (SOR)

- Splitting method with a parameter:

$$M = \omega^{-1} D + L, N = -U - (1 - \omega^{-1}) D$$

hence

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

- Convergence speed depends on ω.
  - $\omega = 1$: Gauss-Seidel
- Determination of optimal ω for *consistently ordered matrices*[1]:

$$\begin{aligned} \omega_{\text{opt}} &= \frac{2}{1 + \sqrt{1 - \mu^2}} \\ \rho_{G_{SOR}} &= \omega_{\text{opt}} - 1 \end{aligned}$$

[1] $\forall z \in \mathbb{R} : \det(zI - L - U) = \det(zI - \gamma L - \gamma^{-1} U) \quad \forall \gamma \in \mathbb{R} \setminus \{0\}$.

Limitations of averaging neighbours

