1 Solving linear systems $\boldsymbol{A}\vec{x}=\vec{b}$

Direct methods (Gauss elimination, LU decomposition, Cholesky decomposition). Useful for dense or banded (e.g., tridiagonal) matrices.

Iterative methods. Useful for large sparse matrices.

In general, $\vec{x}^{(k+1)}=\boldsymbol{B}^{(k)}\vec{x}^{(k)}+\vec{g}^{(k)}$ $\qquad \vec{x}=\boldsymbol{B}^{(k)}\vec{x}+\vec{g}^{(k)}$

Hopefully, $\vec{x}^{(k)}\to\vec{x},\ k\to\infty$

For the methods we have considered, $\boldsymbol{A}=\boldsymbol{A}_1+\boldsymbol{A}_2,$ $\qquad \boldsymbol{A}_1\vec{x}=\vec{b}-\boldsymbol{A}_2\vec{x}$

$$\vec{x}=\boldsymbol{A}_1^{-1}\vec{b}-\boldsymbol{A}_1^{-1}\boldsymbol{A}_2\vec{x} \ \Rightarrow\ \vec{x}^{(k+1)}=\boldsymbol{A}_1^{-1}\vec{b}-\boldsymbol{A}_1^{-1}\boldsymbol{A}_2\vec{x}^{(k)}$$

$\boldsymbol{B}=-\boldsymbol{A}_1^{-1}\boldsymbol{A}_2;\ \ \vec{g}=\boldsymbol{A}_1^{-1}\vec{b}$ Both are *k*-independent.

Stationary iterative methods. $\vec{x}^{(k+1)}=\boldsymbol{B}\vec{x}^{(k)}+\vec{g}$

Consider the error $\vec{e}^{(k)}=\vec{x}^{(k)}-\vec{x}.$

$$\vec{e}^{(k+1)}=\vec{x}^{(k+1)}-\vec{x}=\boldsymbol{B}^{(k)}\vec{x}^{(k)}+\vec{g}^{(k)}-\boldsymbol{B}^{(k)}\vec{x}-\vec{g}^{(k)}=\boldsymbol{B}^{(k)}\vec{e}^{(k)}$$

$$\vec{e}^{(k+1)} = \boldsymbol{B}^{(k)} \vec{e}^{(k)}$$

$$\|\vec{e}^{(k+1)}\| = \|\boldsymbol{B}^{(k)} \vec{e}^{(k)}\| \le |\lambda|_{\max} \|\vec{e}^{(k)}\| = \rho(\boldsymbol{B}) \|\vec{e}^{(k)}\|$$

$\rho(\boldsymbol{B})$ is called the spectral radius of matrix $\boldsymbol{B}$. $\|\vec{e}^{(k)}\| \le [\rho(\boldsymbol{B})]^k \|\vec{e}^{(0)}\|$

For convergence, need $\rho(\boldsymbol{B}) < 1.$ If we want the final error to be at most ε times the initial error, the minimum number of iterations satisfies $[\rho(\boldsymbol{B})]^{k_{\min}} \le \epsilon$

The smaller $\rho(\boldsymbol{B})$, the fewer iterations are needed.

$\boldsymbol{A} = \boldsymbol{A}_1 + \boldsymbol{A}_2$ $\vec{x}^{(k+1)} = \boldsymbol{A}_1^{-1} \vec{b} - \boldsymbol{A}_1^{-1} \boldsymbol{A}_2 \vec{x}^{(k+1)}$ $\boldsymbol{B} = -\boldsymbol{A}_1^{-1} \boldsymbol{A}_2$

For the Jacobi method, $\boldsymbol{A}_1 = \boldsymbol{D}$, $\boldsymbol{A}_2 = \boldsymbol{L} + \boldsymbol{U}$, $\boldsymbol{B} = -\boldsymbol{D}^{-1}(\boldsymbol{L} + \boldsymbol{U})$

$$b_{ij} = \begin{cases} -\dfrac{a_{ij}}{a_{ii}}, & i \ne j, \\ 0, & i = j. \end{cases}$$

$$x_i^{(k+1)} = -\frac{1}{a_{ii}} \sum_{j \ne i} a_{ij} x_j^{(k)} + \frac{1}{a_{ii}} b_i$$

Consider an arbitrary eigenvalue of $\boldsymbol{B}$, λ, and the corresponding eigenvector $\vec{y}$. Assume, without loss of generality, that y_i is the largest component by absolute value and $|y_i| = 1.$ Then

$$|\lambda| = |\lambda||x_i| = \left|\Sigma_{j \ne i} b_{ij} x_j\right| \le \Sigma_{j \ne i} |b_{ij}| = \Sigma_{j \ne i} \frac{|a_{ij}|}{|a_{ii}|}$$

$$|\lambda| \le \sum_{j \neq i} \frac{|a_{ij}|}{|a_{ii}|}$$ To ensure convergence, need $|\lambda| < 1$

This will definitely be the case, if $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$. Diagonal dominance.

Gauss-Seidel. $$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(-\sum_{j<i} a_{ij} x_j^{(k+1)} - \sum_{j>i} a_{ij} x_j^{(k)} + \frac{1}{a_{ii}} b_i \right)$$

$$\vec{x}^{(k+1)} = \boldsymbol{A}_1^{-1} \vec{b} - \boldsymbol{A}_1^{-1} \boldsymbol{A}_2 \vec{x}^{(k)} \qquad \boldsymbol{A}_1 = \boldsymbol{D} + \boldsymbol{L} \qquad \boldsymbol{A}_2 = \boldsymbol{U}$$

$\vec{x}^{(k+1)} = \boldsymbol{B}\,\vec{x}^{(k)} + \vec{g}$ It can be proved that for symmetric ***A*** iterations converge if and only if ***A*** is also positive definite (all eigenvalues > 0).

E.g., for tridiagonal matrices exactly twice as fast as Jacobi.

Note the result depends on the ordering of the components.

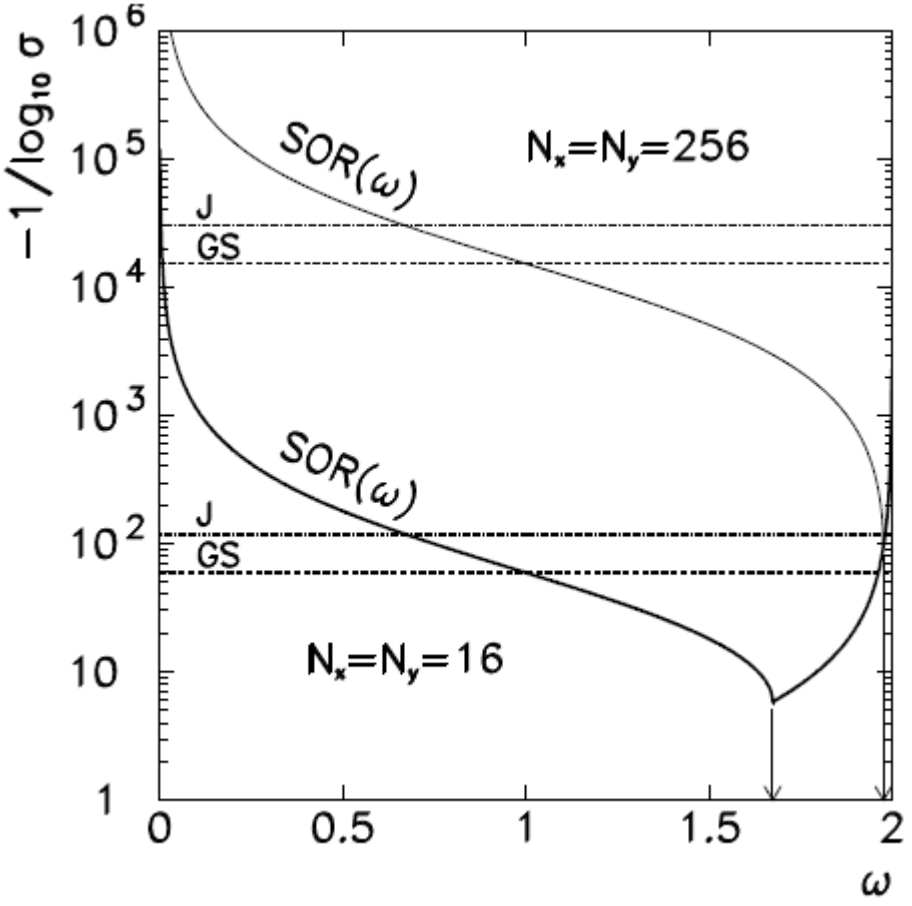
Successive overrelaxation (SOR)

$$x_i^{(n+1)} = (1-\omega) x^{(n)} + \frac{\omega}{a_{ii}} \left(-\sum_{j<i} x_j^{(n+1)} - \sum_{j>i} x_j^{(n)} + \frac{1}{a_{ii}} b_i \right)$$

Converges whenever Gauss-Seidel converges for $0 < \omega < 2$.

Overrelaxation for $1 < \omega < 2$.

Inverse of the convergence rate for matrices arising from discretization of the Poisson equation in 2D.



Optimal convergence rate for a particular ω. The larger the matrix, the larger the gain. Optimal ω is known for matrices arising in some problems, but not in general. There are ways to estimate it, but not easy.

Non-stationary approaches

$$\vec{x}^{(k+1)} = \boldsymbol{B}^{(k)} \vec{x}^{(k)} + \vec{g}^{(k)}$$

Conjugate gradient method

Only works for symmetric positive definite matrices (can be generalized to Hermitian matrices)

Solving a linear system as a minimization problem: consider

$$F(\vec{x}) = \frac{1}{2} \vec{x} \cdot \boldsymbol{A} \cdot \vec{x} - \vec{b} \cdot \vec{x} = \frac{1}{2} a_{ij} x_i x_j - b_i x_i$$

$$\frac{\partial F}{\partial x_i} = \frac{1}{2} (a_{ij} x_j + a_{ji} x_j) - b_i = a_{ij} x_j - b_i = 0$$

Minimum, since the matrix is positive definite.

Start at $\vec{x}^{(0)}$. Move in the steepest descent direction

$$\vec{p}^{(0)} = -\nabla F = \vec{b} - \boldsymbol{A} \vec{x} = \vec{r}^{(0)}. \qquad \vec{x}^{(1)} = \vec{x}^{(0)} + \alpha_0 \vec{p}^{(0)}.$$

$$F(\vec{x}^{(1)}) = \frac{1}{2} (\vec{x}^{(0)} + \alpha_0 \vec{p}^{(0)}) \cdot \boldsymbol{A} \cdot (\vec{x}^{(0)} + \alpha_0 \vec{p}^{(0)}) - \vec{b} \cdot (\vec{x}^{(0)} + \alpha_0 \vec{p}^{(0)})$$

$$\frac{dF}{d\alpha_0} = \alpha_0 \vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(0)} + \vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{x}^{(0)} - \vec{b} \cdot p^{(0)} = \alpha_0 \vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(0)} - \vec{r}^{(0)} \cdot \vec{p}^{(0)} = 0$$

$$\frac{dF}{d\,\alpha_0}=\alpha_0\,\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}-\vec{r}^{(0)}\cdot\vec{p}^{(0)}=0\;\Rightarrow\;\alpha_0=\frac{\vec{r}^{(0)}\cdot\vec{p}^{(0)}}{\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}}=\frac{\vec{r}^{(0)}\cdot\vec{r}^{(0)}}{\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}}$$

$$\vec{x}^{(1)}=\vec{x}^{(0)}+\alpha_0\,\vec{p}^{(0)}=\vec{x}^{(0)}+\frac{\vec{r}^{(0)}\cdot\vec{r}^{(0)}}{\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}}\,\vec{p}^{(0)}$$

$$\vec{r}^{(1)}=\vec{b}-\boldsymbol{A}\,\vec{x}^{(1)}=\vec{b}-\boldsymbol{A}\,\vec{x}^{(0)}-\alpha_0\,\boldsymbol{A}\,\vec{p}^{(0)}=\vec{r}^{(0)}-\alpha_0\,\boldsymbol{A}\,\vec{p}^{(0)}=\vec{r}^{(0)}-\frac{\vec{r}^{(0)}\cdot\vec{r}^{(0)}}{\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}}\,\boldsymbol{A}\,\vec{p}^{(0)}$$

$$\vec{r}^{(1)}\cdot\vec{r}^{(0)}=\vec{r}^{(0)}\cdot\vec{r}^{(0)}-\frac{\vec{r}^{(0)}\cdot\vec{r}^{(0)}}{\vec{p}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}}\,\vec{r}^{(0)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}=0$$

Quite logical: the gradient at the minimum should be orthogonal to the trajectory. Could have used the orthogonality condition instead of the minimization condition to determine α_0.

What do we do now? Steepest descent again? Not optimal.

$\vec{p}^{(1)}=\vec{r}^{(1)}+\beta_1\,\vec{p}^{(0)}$ Choose β_1 so that reach minimum in the space spanned by $\vec{r}^{(0)}$ and $\vec{r}^{(1)}$. $\vec{r}^{(2)}$ should be $\perp\;\vec{r}^{(0)},\vec{r}^{(1)}$

$$\vec{p}^{(1)} = \vec{r}^{(1)} + \beta_1 \vec{p}^{(0)}$$

$$\vec{x}^{(2)} = \vec{x}^{(1)} + \alpha_1 \vec{p}^{(1)} \qquad \vec{r}^{(2)} = \vec{b} - \boldsymbol{A}\vec{x}^{(2)} = \vec{b} - \boldsymbol{A}\vec{x}^{(1)} - \alpha_1 \boldsymbol{A}\vec{p}^{(1)} = \vec{r}^{(1)} - \alpha_1 \boldsymbol{A}\vec{p}^{(1)}$$

$$\vec{r}^{(2)} \cdot \vec{r}^{(0)} = -\alpha_1 \vec{r}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(1)} = -\alpha_1 \vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(1)} = 0$$

So the requirement is

$$\vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(1)} = 0 \;\Rightarrow\; \vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \left(\vec{r}^{(1)} + \beta_1 \vec{p}^{(0)}\right) = 0 \;\Rightarrow\; \beta_1 = -\frac{\vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{r}^{(1)}}{\vec{p}^{(0)} \cdot \boldsymbol{A} \cdot \vec{p}^{(0)}}$$

New direction is conjugate to the previous one – hence the name.

Next iteration: want to reach the minimum in space spanned by $\vec{r}^{(0)}$, $\vec{r}^{(1)}$, $\vec{r}^{(2)}$.

Residual $\vec{r}^{(3)}$ will need to be orthogonal to $\vec{r}^{(0)}$, $\vec{r}^{(1)}$, $\vec{r}^{(2)}$. It would seem reasonable to assume that we would now need to choose

$$\vec{p}^{(2)} = \vec{r}^{(2)} + \beta_2 \vec{p}^{(1)} + \gamma_2 \vec{p}^{(0)} \qquad \vec{x}^{(3)} = \vec{x}^{(2)} + \alpha_2 \vec{p}^{(2)} \;\Rightarrow\; \vec{r}^{(3)} = \vec{r}^{(2)} - \alpha_2 \boldsymbol{A}\vec{p}^{(2)}$$

This way, we would have 3 constants to adjust to satisfy 3 orthogonality conditions. And then we'd have 4 constants, 5, 6, ...

A remarkable thing is that this is not the case! $\gamma_2 = 0$!!!

$\vec{p}^{(2)}=\vec{r}^{(2)}+\beta_2\vec{p}^{(1)}$ $\qquad\qquad$ $\vec{x}^{(3)}=\vec{x}^{(2)}+\alpha_2\vec{p}^{(2)} \;\Rightarrow\; \vec{r}^{(3)}=\vec{r}^{(2)}-\alpha_2\boldsymbol{A}\vec{p}^{(2)}$

Adjusting α_2 and β_2 to satisfy $\vec{r}^{(3)}\cdot\vec{r}^{(2)}=0$ and $\vec{p}^{(2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(1)}$, we automatically satisfy $\vec{r}^{(3)}\cdot\vec{r}^{(1)}=0$, $\vec{r}^{(3)}\cdot\vec{r}^{(0)}=0$ and $\vec{p}^{(2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(0)}$.

Moreover, this is true for all subsequent iterations as well: if

$\vec{r}^{(j)}\cdot\vec{r}^{(i)}=0\;\forall\, i,\, j<k\;\;(i\neq j)$ and $\vec{p}^{(j)}\cdot\boldsymbol{A}\cdot\vec{p}^{(i)}=0\;\forall\, i,\, j<k-1\;\;(i\neq j)$,

then, defining $\vec{p}^{(k-1)}=\vec{r}^{(k-1)}+\beta_{k-1}\vec{p}^{(k-2)}$ and $\vec{x}^{(k)}=\vec{x}^{(k-1)}+\alpha_{k-1}\vec{p}^{(k-1)}$,

it is sufficient to ensure $\vec{r}^{(k)}\cdot\vec{r}^{(k-1)}=0$ and $\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}=0$ for

$\vec{r}^{(k)}\cdot\vec{r}^{(j)}=0,\;\; j<k-1$ and $\left(\vec{p}^{(k-1)}\right)^T\boldsymbol{A}\,\vec{p}^{(j)}=0,\;\; j<k-2$ to be satisfied

automatically. Indeed, using $\vec{r}^{(k)}=\vec{b}-\boldsymbol{A}\left(\vec{x}^{(k-1)}+\alpha_{k-1}\vec{p}^{(k-1)}\right)=\vec{r}^{(k-1)}-\alpha_{k-1}\boldsymbol{A}\,\vec{p}^{(k-1)}$,

$$\vec{r}^{(k)}\cdot\vec{r}^{(j)}=\left(\vec{r}^{(k-1)}-\alpha_{k-1}\boldsymbol{A}\,\vec{p}^{(k-1)}\right)\cdot\vec{r}^{(j)}=-\alpha_{k-1}\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{r}^{(j)}$$

$$=-\alpha_{k-1}\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\left(\vec{p}^{(j)}-\beta_j\vec{p}_{j-1}\right)=0,\;\; j<k-1,$$

$$\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(j)}=\left(\vec{r}^{(k-1)}+\beta_{k-1}\vec{p}^{(k-2)}\right)\cdot\boldsymbol{A}\cdot\vec{p}^{(j)}=\vec{r}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(j)}$$

$$=\vec{r}^{(k-1)}\cdot\left(\vec{r}^{(j)}-\vec{r}^{(j+1)}\right)/\alpha_j=0,\;\; j<k-2.$$

$$\vec{p}^{(k-1)}=\vec{r}^{(k-1)}+\beta_{k-1}\vec{p}^{(k-2)},\quad \vec{r}^{(k)}=\vec{r}^{(k-1)}-\alpha_{k-1}\boldsymbol{A}\vec{p}^{(k-1)}$$

$$\vec{r}^{(k-1)}\cdot\vec{r}^{(k)}=\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}-\alpha_{k-1}\vec{r}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-1)}$$

$$=\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}-\alpha_{k-1}\left(\vec{p}^{(k-1)}-\beta_{k-1}\vec{p}^{(k-2)}\right)\cdot\boldsymbol{A}\cdot\vec{p}^{(k-1)}$$

$$=\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}-\alpha_{k-1}\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-1)}=0$$

$$\alpha_{k-1}=\frac{\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}}{\vec{p}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-1)}}$$

$$\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-1)}=\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\left(\vec{r}^{(k-1)}+\beta_{k-1}\vec{p}^{(k-2)}\right)=0 \qquad \beta_{k-1}=-\frac{\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{r}^{(k-1)}}{\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}}$$

$$\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{r}^{(k-1)}=\vec{r}^{(k-1)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}=\vec{r}^{(k-1)}\cdot\left(\vec{r}^{(k-2)}-\vec{r}^{(k-1)}\right)/\alpha_{k-2}$$

$$=-\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}/\alpha_{k-2}$$

$$\vec{p}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}=\left(\vec{r}^{(k-2)}+\beta_{k-2}\vec{p}^{(k-3)}\right)\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}=\vec{r}^{(k-2)}\cdot\boldsymbol{A}\cdot\vec{p}^{(k-2)}$$

$$=\vec{r}^{(k-2)}\cdot\left(\vec{r}^{(k-2)}-\vec{r}^{(k-1)}\right)/\alpha_{k-2}=\vec{r}^{(k-2)}\cdot\vec{r}^{(k-2)}/\alpha_{k-2}$$

$$\beta_{k-1}=\frac{\vec{r}^{(k-1)}\cdot\vec{r}^{(k-1)}}{\vec{r}^{(k-2)}\cdot\vec{r}^{(k-2)}}$$

Complete algorithm:

Start with initial guess $\vec{x}^{(0)}$. Calculate the residual $\vec{r}^{(0)} = \vec{b} - \boldsymbol{A}\vec{x}^{(0)}$.

$k = 0$

while $\|\vec{r}^{(k)}\| > \text{tolerance} \times \|\vec{b}\|$

if $k = 0$, $\vec{p}^{(0)} = \vec{r}^{(0)}$

else

$$\beta_k = \frac{\vec{r}^{(k)} \cdot \vec{r}^{(k)}}{\vec{r}^{(k-1)} \cdot \vec{r}^{(k-1)}}$$

$$\vec{p}^{(k)} = \vec{r}^{(k)} + \beta_k \vec{p}^{(k-1)}$$

end

$$\alpha_k = \frac{\vec{r}^{(k)} \cdot \vec{r}^{(k)}}{\vec{p}^{(k)} \cdot \boldsymbol{A} \cdot \vec{p}^{(k)}}$$

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} + \alpha_k \vec{p}^{(k)},$$

$$\vec{r}^{(k+1)} = \vec{r}^{(k)} - \alpha_k \boldsymbol{A} \vec{p}^{(k)},$$

$k = k + 1$

end

After every iteration, the residual gets restricted to space of a smaller dimensionality.

Should vanish completely after N iterations (actually, as many as there are distinct eigenvalues). In this sense, it should behave like an exact method.

Unfortunately, only true in exact arithmetic.

But is useful as an iterative method.

11 Let's look again at the residuals.

$$\vec{r}^{(1)} = \vec{r}^{(0)} - \alpha_0 \boldsymbol{A}\,\vec{p}^{(0)} = \vec{r}^{(0)} - \alpha_0 \boldsymbol{A}\,\vec{r}^{(0)}$$ A linear combination of $\vec{r}^{(0)}$ and $\boldsymbol{A}\,\vec{r}^{(0)}$

Space spanned by $\vec{r}^{(0)}$ and $\vec{r}^{(1)}$ is also space spanned by $\vec{r}^{(0)}$ and $\boldsymbol{A}\,\vec{r}^{(0)}$

$$\vec{p}^{(1)} = \vec{r}^{(1)} + \beta_1 \vec{p}^{(0)} = \vec{r}^{(1)} + \beta_1 \vec{r}^{(0)}$$

$$\vec{r}^{(2)} = \vec{r}^{(1)} - \alpha_1 \boldsymbol{A}\,\vec{p}^{(1)} = \vec{r}^{(0)} - \alpha_0 \boldsymbol{A}\,\vec{r}^{(0)} - \alpha_1 \boldsymbol{A}\,\vec{r}^{(1)} - \alpha_1 \beta_1 \boldsymbol{A}\,\vec{r}^{(0)}$$

$$= \vec{r}^{(0)} - \alpha_0 \boldsymbol{A}\,\vec{r}^{(0)} - \alpha_1 \boldsymbol{A}\,\vec{r}^{(0)} + \alpha_0 \alpha_1 \boldsymbol{A}^2 \vec{r}^{(0)} - \alpha_1 \beta_1 \boldsymbol{A}\,\vec{r}^{(0)}$$

A linear combination of $\vec{r}^{(0)},\ \boldsymbol{A}\,\vec{r}^{(0)},\ \boldsymbol{A}^2 \vec{r}^{(0)}$

Linear subspace spanned by $\vec{u},\ \boldsymbol{B}\,\vec{u},\ \boldsymbol{B}^2 \vec{u},\ ...,\ \boldsymbol{B}^{s-1} \vec{u}$ is called order-s Krylov subspace.

In the conjugate gradient method by construction after s iterations the residual vanishes in order-s Krylov subspace. Such methods are called Krylov subspace methods.

Convergence rate of the conjugate gradient method

$$\|\vec{x}-\vec{x}^{(k)}\|_A \le 2\|\vec{x}-\vec{x}^{(0)}\|_A \left(\frac{\sqrt{K(\boldsymbol{A})}-1}{\sqrt{K(\boldsymbol{A})}+1}\right)^k,$$

where $\|\vec{w}\|_A = \sqrt{\vec{w}\cdot\boldsymbol{A}\cdot\vec{w}}$ and $K(\boldsymbol{A})$ is the spectral condition number (the ratio of the largest and smallest eigenvalues). Slow convergence if this ratio is large (ill-conditioned matrix). Sometimes not as bad, e.g., when the extreme eigenvalues are well-separated from the rest. But still, very often an issue, since many matrices are ill-conditioned.

Consider a chain of N beads connected with springs; x_i are displacements from equilibrium. Assume all masses are 1.

$\ddot{\vec{x}} = -\boldsymbol{D}\,\vec{x}$ Eigenvalues of D are squares of the frequencies of the modes.

$\boldsymbol{D}\,\vec{x} = \vec{F}$ $\omega \sim c/\lambda$ $K(\boldsymbol{D}) \sim N^2$

Preconditioning

Preconditioning

$$A\vec{x}=\vec{b} \qquad C^{-1}AC^{-1}C\vec{x}=C^{-1}\vec{b} \qquad \tilde{A}\tilde{\vec{x}}=\tilde{\vec{b}},$$

where $\tilde{A}=C^{-1}AC^{-1},\ \tilde{\vec{x}}=C\vec{x},\ \tilde{\vec{b}}=C^{-1}\vec{b}.$

If C is symmetric and positive definite, the same is true about $C^{-1}AC^{-1}$,

so we can apply conjugate gradient to it. Just add tildes.

$$\tilde{\vec{r}}=\tilde{\vec{b}}-\tilde{A}\tilde{\vec{x}}=C^{-1}\vec{b}-C^{-1}AC^{-1}C\vec{x}=C^{-1}\vec{r}$$

Let us define $\vec{\pi}^{(k)}$ such that $\tilde{\vec{p}}^{(k)}=C\vec{\pi}^{(k)}$, $M=C^2$, $\vec{z}^{(k)}$ such that $M\vec{z}^{(k)}=\vec{r}^{(k)}$

$$\tilde{\vec{p}}^{(0)}=\tilde{\vec{r}}^{(0)} \ \Rightarrow\ C\vec{\pi}^{(0)}=C^{-1}\vec{r}^{(0)} \ \Rightarrow\ \vec{\pi}^{(0)}=M^{-1}\vec{r}^{(0)}=\vec{z}^{(0)}$$

$$\beta_k=\frac{\tilde{\vec{r}}^{(k)}\cdot\tilde{\vec{r}}^{(k)}}{\tilde{\vec{r}}^{(k-1)}\cdot\tilde{\vec{r}}^{(k-1)}}=\frac{\vec{r}^{(k)}C^{-1}C^{-1}\vec{r}^{(k)}}{\vec{r}^{(k-1)}C^{-1}C^{-1}\vec{r}^{(k-1)}}=\frac{\vec{r}^{(k)}\vec{z}^{(k)}}{\vec{r}^{(k-1)}\vec{z}^{(k-1)}}$$

$$\tilde{\vec{p}}^{(k)}=\tilde{\vec{r}}^{(k)}+\beta_k\tilde{\vec{p}}^{(k-1)} \ \Rightarrow\ C\vec{\pi}^{(k)}=C^{-1}\vec{r}^{(k)}+\beta_k C\vec{\pi}^{(k-1)} \ \Rightarrow\ \vec{\pi}^{(k)}=\vec{z}^{(k)}+\beta_k\vec{\pi}^{(k-1)}$$

$$\alpha_k=\frac{\tilde{\vec{r}}^{(k)}\cdot\tilde{\vec{r}}^{(k)}}{\tilde{\vec{p}}^{(k)}\cdot\tilde{A}\cdot\tilde{\vec{p}}^{(k)}}=\frac{\vec{r}^{(k)}C^{-1}C^{-1}\vec{r}^{(k)}}{\vec{\pi}^{(k)}CC^{-1}AC^{-1}C\vec{\pi}^{(k)}}=\frac{\vec{r}^{(k)}\cdot\vec{z}^{(k)}}{\vec{\pi}^{(k)}A\vec{\pi}^{(k)}}$$

$$\tilde{\vec{x}}^{(k+1)} = \tilde{\vec{x}}^{(k)} + \alpha_k \tilde{\vec{p}}^{(k)} \;\Rightarrow\; \boldsymbol{C}\vec{x}^{(k+1)} = \boldsymbol{C}\vec{x}^{(k)} + \alpha_k \boldsymbol{C}\vec{\pi}^{(k)} \;\Rightarrow\; \vec{x}^{(k+1)} = \vec{x}^{(k)} + \alpha_k \vec{\pi}^{(k)}$$

$$\tilde{\vec{r}}^{(k+1)} = \tilde{\vec{r}}^{(k)} - \alpha_k \tilde{\boldsymbol{A}} \tilde{\vec{p}}^{(k)} \;\Rightarrow\; \boldsymbol{C}^{-1}\vec{r}^{(k+1)} = \boldsymbol{C}^{-1}\vec{r}^{(k)} - \alpha_k \boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{C}\vec{\pi}^{(k)}$$

$$\Rightarrow\; \vec{r}^{(k+1)} = \vec{r}^{(k)} - \alpha_k \boldsymbol{A}\vec{\pi}^{(k)}$$

Preconditioned algorithm:

Start with initial guess $\vec{x}^{(0)}$.

Calculate the residual $\vec{r}^{(0)} = \vec{b} - \boldsymbol{A}\vec{x}^{(0)}$.

$k = 0$

while $\|\vec{r}^{(k)}\| > \text{tolerance} \times \|\vec{b}\|$

Solve $\boldsymbol{M}\vec{z}^{(k)} = \vec{r}^{(k)}$

if $k = 0$, $\vec{p}^{(0)} = \vec{z}^{(0)}$

else

$$\beta_k = \frac{\vec{r}^{(k)}\vec{z}^{(k)}}{\vec{r}^{(k-1)}\vec{z}^{(k-1)}}$$

$$\vec{\pi}^{(k)} = \vec{z}^{(k)} + \beta_k \vec{\pi}^{(k-1)}$$

end

$$\alpha_k = \frac{\vec{r}^{(k)} \cdot \vec{z}^{(k)}}{\vec{\pi}^{(k)}\boldsymbol{A}\vec{\pi}^{(k)}}$$

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} + \alpha_k \vec{\pi}^{(k)}$$

$$\vec{r}^{(k+1)} = \vec{r}^{(k)} - \alpha_k \boldsymbol{A}\vec{\pi}^{(k)}$$

$$k = k + 1$$

end

15 By construction, equivalent to solving an equation with matrix $\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}^{-1}$.

Need to choose $\boldsymbol{C}$ such that $\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C}^{-1}$ is better conditioned. Note $\boldsymbol{C}$ does not enter explicitly, only $\boldsymbol{M}=\boldsymbol{C}^2$.

Need to solve a system with matrix M at every step. Obviously, should be easy to solve.

There are many different preconditioners used in practice differing by the amount of work needed to set it up and cost per iteration.

Simplest choice: Jacobi preconditioner – diagonal matrix

$$m_{ij}=\begin{cases} a_{ii},\ i=j \\ 0 \text{ otherwise} \end{cases}$$

Block Jacobi – a block-diagonal matrix. E.g., multiple variables per node, or spatially, or according to division of variables between processors.

Incomplete factorization preconditioners. E.g., Cholesky decomposition keeping zero all matrix elements equal to zero in $\boldsymbol{A}$.

Indefinite and/or non-symmetric systems

Conjugate gradient is not directly applicable. For non-symmetric systems, there is no short recurrence where only a few recent directions need to be used. For symmetric, but indefinite systems, short recurrences are possible, but the solution no longer corresponds to a minimum. No universally good approach.

Simplest approach: conjugate gradient for normal equations.

$\boldsymbol{A}\,\vec{x} = \vec{b}$ Multiply by $\boldsymbol{A}^T$: $\boldsymbol{A}^T\boldsymbol{A}\,\vec{x} = \boldsymbol{A}^T\,\vec{x}$

Matrix $\boldsymbol{A}^T\boldsymbol{A}$ is both symmetric and positive definite (unless $\boldsymbol{A}$ is singular). CG can be applied to it.

Problem: the condition number is the square of that of matrix $\boldsymbol{A}$, so convergence can be rather slow.

MINRES and SYMMLQ: for symmetric but indefinite matrices. Minimizes the norm of the residual. A sequence of orthogonal vectors in the Krylov subspace can still be generated easily.

GMRES: a similar approach, but for non-symmetric systems. No short recurrences, needs to store all vectors generated so far and orthogonalize with respect to all of them. In practice, restarts are done. Reliable but costly in time and memory if many iterations are required.

Biconjugate gradient (BiCG): generates two mutually orthogonal sequences of vectors.

$$\vec{r}^{(i)} = \vec{r}^{(i-1)} - \alpha_i \boldsymbol{A}\, \vec{p}^{(i)}, \qquad \vec{r}\,'^{(i)} = \vec{r}\,'^{(i-1)} - \alpha_i \boldsymbol{A}\, \vec{p}\,'^{(i)},$$

$$\vec{p}^{(i)} = \vec{r}^{(i-1)} + \beta_{i-1}\, \vec{p}^{(i-1)}, \qquad \vec{p}\,'^{(i)} = \vec{r}\,'^{(i-1)} + \beta_{i-1}\, \vec{p}\,'^{(i-1)},$$

$$\alpha_i = \frac{\vec{r}\,'^{(i-1)} \cdot \vec{r}^{(i-1)}}{\vec{p}\,'^{(i)} \cdot \boldsymbol{A} \cdot \vec{p}^{(i)}} \qquad \beta_i = \frac{\vec{r}\,'^{(i)} \cdot \vec{r}^{(i)}}{\vec{r}\,'^{(i-1)} \cdot \vec{r}^{(i-1)}} \qquad \vec{r}\,'^{(i)} \cdot \vec{r}^{(j)} = \vec{p}\,'^{(i)} \cdot \boldsymbol{A} \cdot \vec{p}^{(j)} = 0, \quad i \neq j$$

Quasi-minimal residual (QMR), conjugate gradient squared (CGS), biconjugate gradient stabilized (BiCGStab).

Iterative improvement

$\boldsymbol{A}\vec{x}=\vec{b}$ Suppose the solution $\tilde{\vec{x}}$ obtained with a direct method is approximate due to round-off error. Define $\delta\vec{x}=\tilde{\vec{x}}-\vec{x}$

$$\vec{r}=\vec{b}-\boldsymbol{A}\tilde{\vec{x}}=\boldsymbol{A}\vec{x}-\boldsymbol{A}\tilde{\vec{x}}=-\boldsymbol{A}\delta\vec{x}$$

Solve for $\delta\vec{x}$. Can be repeated, in which case decomposition of A should be done.

The eigenvalue problem

$$\boldsymbol{A}\,\vec{x} = \lambda\,\vec{x}$$

In principle, eigenvalues are the roots of the characteristic polynomial:

$$\det\left(\boldsymbol{A} - \lambda\,\boldsymbol{I}\right) = 0$$

Can use a root-finding procedure. Not a good idea, except perhaps for very small matrices and some special cases. Coefficients are subject to roundoff error and the roots are very sensitive to the values of the coefficients. One important observation, though:

$$\begin{pmatrix} 0-\lambda & 1 & 0 & \ldots & 0 \\ 0 & 0-\lambda & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 1 \\ -a_0 & -a_1 & -a_2 & \ldots & -a_{n-1}-\lambda \end{pmatrix}$$

The determinant of this is (up to a sign)

$$a_0 + a_1\lambda + a_2\lambda^2 + \ldots + a_{n-1}\lambda^{n-1} + \lambda^n$$

Thus, for any polynomial there is an eigenvalue problem with the same roots. But there is no exact universal formula for order > 4. Thus, universal direct methods for eigenvalues cannot exist either – iterations only.

Just like for solving linear systems, we can ask the question of “condition” or sensitivity to perturbations in the matrix elements. The relevant condition number is very different from that for solving linear equations. In fact, for symmetric or Hermitian matrices the condition number is 1, so these problems are well-conditioned.

Different methods serve different purposes: first of all, some are preferable when only one or a few eigenvalues are desired (say, the largest or the closest to a particular value) and others are designed to find all of them. Type of matrix matters, too.

Power method

The basic approach finds one eigenvalue (largest by absolute value).

Start with an arbitrary vector, multiply with the matrix repeatedly.

Suppose the eigenvalues are λ_i and the corresponding eigenvectors are $\vec{u}_i$.

Suppose also that λ_i is the largest one by absolute value.

Start with vector $\vec{v}^{(0)}$. Expand it: $\vec{v}^{(0)} = \sum_{i=1}^{N} c_i \vec{u}_i$

After k multiplications,

$$\vec{v}^{(k)} = \boldsymbol{A}^k \vec{v}^{(0)} = \boldsymbol{A}^k \sum_{i=1}^{N} c_i \vec{u}_i = \sum_{i=1}^{N} \lambda_i^k c_i \vec{u}_i = \lambda_1^k \left(c_1 \vec{u}_1 + \sum_{i=2}^{N} c_i \left(\lambda_i / \lambda_1 \right)^k \vec{u}_i \right)$$

So the relative weight of eigenvectors other than the first one decays with *k*

The ratio of the values of a given component from one iteration to the next converges to the eigenvalue.

What can go wrong with this approach?

The starting vector may have no component in the dominant eigenvector. Usually not a problem, because even if this is the case initially, it will be introduced due to round-off errors.

What if there are several vectors with the same largest absolute value of the eigenvalue? Let's suppose there are two. Keep just these terms and neglect everything else.

$$\vec{v}^{(k)} \approx \lambda_1^k \left(c_1 \vec{u}_1 + c_2 \left(\lambda_2 / \lambda_1 \right)^k \vec{u}_2 \right)$$

If $\lambda_1 = \lambda_2$, then this will converge to some linear combination of the vectors. But if they are different (only equal by absolute value), there will be never-ending oscillations.