Introduction to Scientific Computing

(Lecture 6: Linear system of equations)

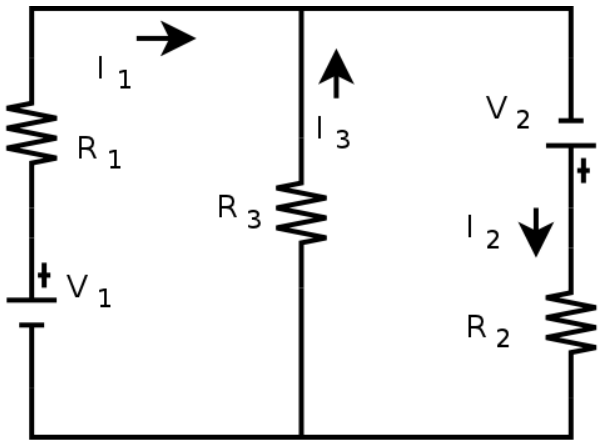
Bojana Rosić

Institute of Scientific Computing

December 6, 2016

Motivation

Let us resolve the problem scheme by using Kirchhoff's laws:



$$-I_1 + I_2 - I_3 = 0$$
$$-I_1 R_1 + I_3 R_3 = -V_1$$
$$-I_2 R_2 - I_3 R_3 = -V_2$$

- the algebraic sum of all the currents flowing toward a node is equal to zero.

$$\sum I = 0$$

- In any closed circuit, the algebraic sum of all the voltages around the loop is equal to zero

$$\sum V = \sum IR$$

Motivation

Hence, we end up with the system of equations

$$-I_1 + I_2 - I_3 = 0$$

$$-I_1 R_1 + I_3 R_3 = -V_1$$
$$-I_2 R_2 - I_3 R_3 = -V_2$$

which is easy to solve algebraically by elimination

$$I_1 = I_2 + I_3$$

and substitution

$$-(I_2 + I_3)R_1 + I_3 r_3 = -V_1 \Rightarrow I_2 = (V_1 + I_3(R_3 - R_1))R_1^{-1}$$

and so on...

What to do in this case?

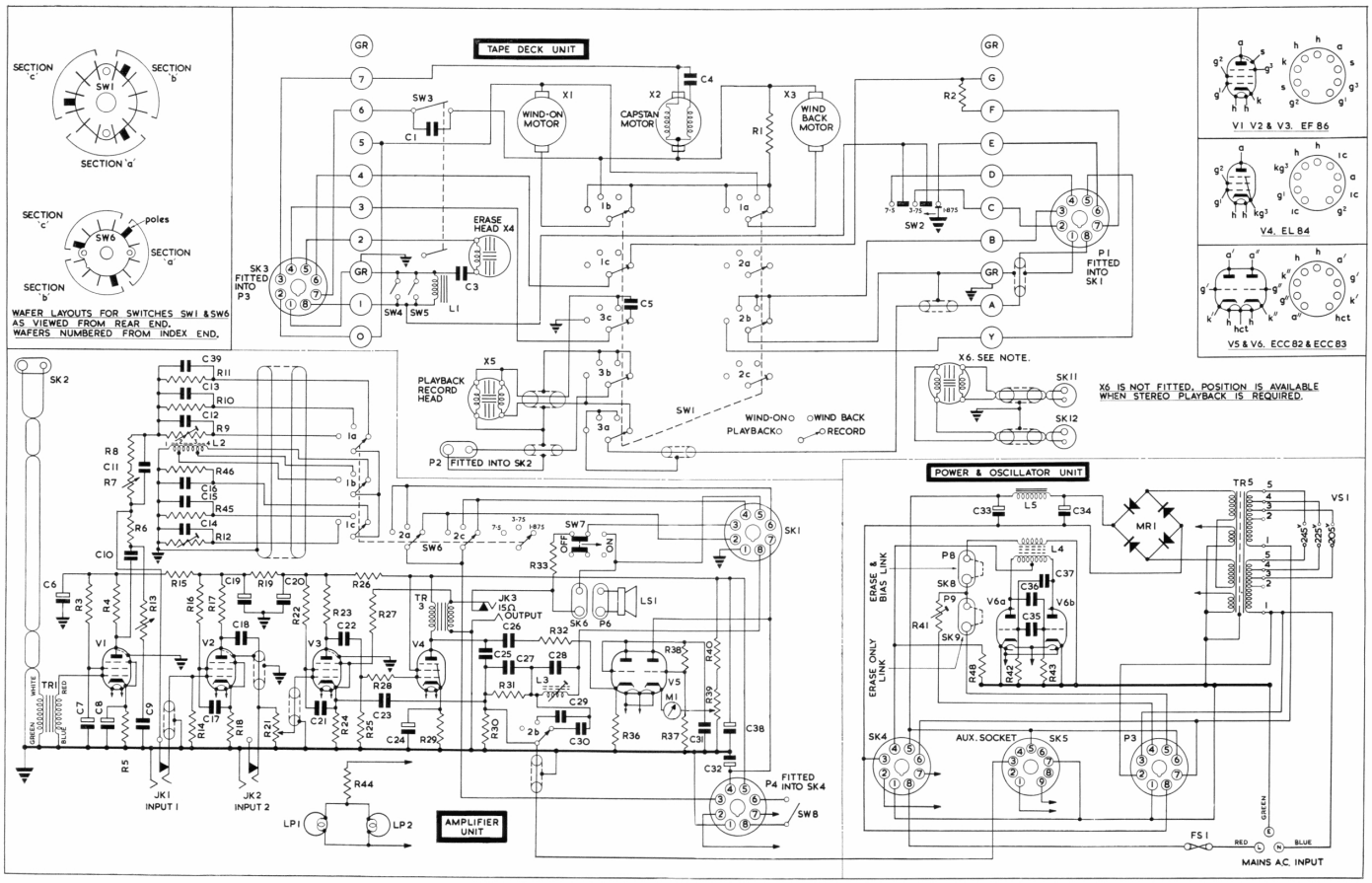
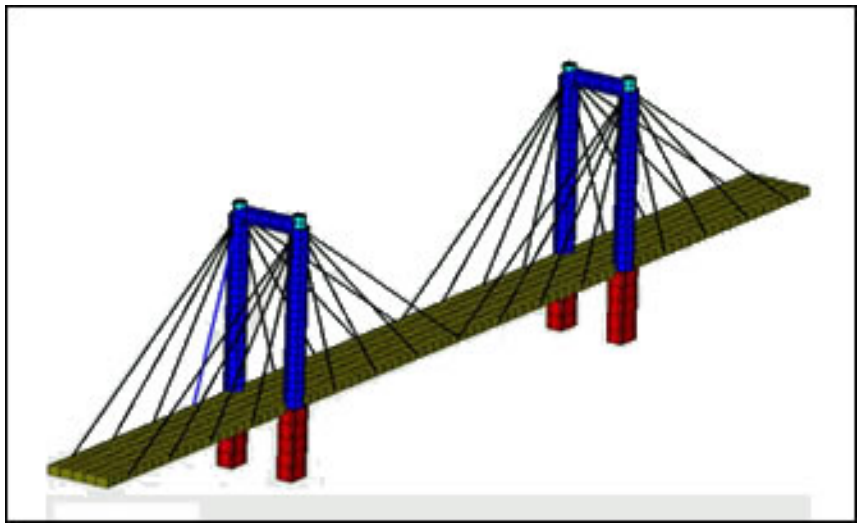


FIG. 20 CIRCUIT DIAGRAM.

Algebraically is difficult to solve.

But, engineering goal

is to solve large-scale (realistic) systems



This usually matches with solving

$$\mathbf{x} = \mathbf{F}(\mathbf{x})$$

in which x is the system state.

Special case

Special case are systems in a linear form

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

in which matrix $\mathbf{A}$ can be of dimensions larger than 10. Think about

$$\mathbf{A} \in \mathbb{R}^{1000 \times 1000}$$

$$\mathbf{A} \in \mathbb{R}^{1000000 \times 1000000}$$

Examples are: solving linear partial differential equations (elasticity, heat equation etc.)

In these cases one cannot compute

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

as one would have problem with memory or computation time.

Linear systems

There are many methods one can use to solve the system

$$\mathbf{Ax} = \mathbf{b}$$

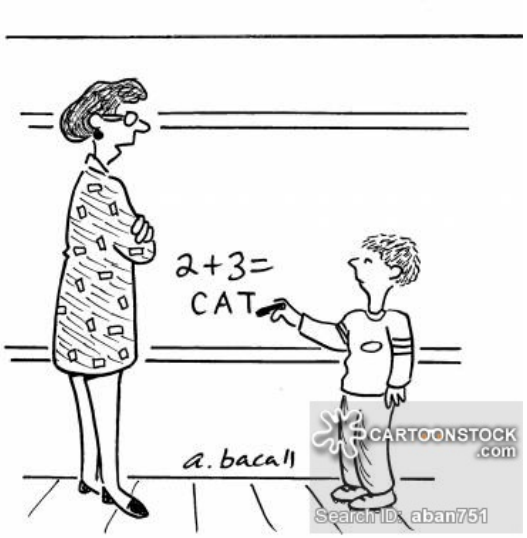
such as:

- **Stationary iterative methods** (approximate operator)
  - Gauss-Seidel (GS)
  - Jacobi (JM)
  - Successive over-relaxation (SOR)
- **Krylov subspace methods**
  - Conjugate Gradient (CG)
  - Biconjugate gradient method (BiCG)
  - generalized minimal residual method (GMRES)
- **combination**

Solving system

Remembering fixed point iteration, we may try something similar. Let us guess the solution by putting

$$\mathbf{x} = \mathbf{x}^{(k)}$$



"There's supposed to be a fine line between a guess and an educated guess."

Since we do not know if our guess is correct, we may check its accuracy by evaluating

$$\mathbf{A}\mathbf{x}^{(k)}$$

and check if the value matches $\mathbf{b}$. Usually our guess will not be correct, and hence we may compute the error

$$\mathbf{d}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}$$

Now we may have educated guess by drawing conclusions from the error value.

What would be our next guess?

Thus, our next guess $\mathbf{x}^{(k+1)}$ will be smaller or larger than $\mathbf{x}^{(k)}$ depending on the error $\mathbf{d}^{(k)}$. Hence,

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{v}^{(k)}$$

in which $\mathbf{v}^{(k)}$ denotes correction.

Our goal is now to find the best correction!

Naturally, the best correction $\mathbf{v}^{(k)}$ would be the correction which satisfies the equation

$$\mathbf{x}^{(k)} + \mathbf{v}^{(k)} = \mathbf{x}.$$

The only problem is that we do not know $\mathbf{x}$!!

But,

we do know that the best correction $\mathbf{v}^{(k)}$) has to satisfy

$$\mathbf{A}\mathbf{x} = \mathbf{A}(\mathbf{x}^{(k)} + \mathbf{v}^{(k)}) = \mathbf{b}$$

This in turn gives us equation for the correction

$$\mathbf{A}\mathbf{v}^{(k)} = \mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} = \mathbf{d}^{(k)}$$

which further yields

$$\mathbf{v}^{(k)} = \mathbf{A}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}) = \mathbf{A}^{-1}\mathbf{d}^{(k)}$$

Problem

However, thhe process of solving

$$\mathbf{v}^{(k)} = \mathbf{A}^{-1}\mathbf{d}^{(k)}$$

requires inversion of $\mathbf{A}$ which may be very expensive.

size	time [s]
10×10	7.6000e-05
100×100	8.4500e-04
$10^3 \times 10^3$	0.394
$10^4 \times 10^4$	415.64

Solution: pose the problem differently

Do not invert matrix $\mathbf{A}$, but find some simpler matrix $\mathbf{C} \approx \mathbf{A}$ which is easy to invert!

Then the problem

$$\mathbf{v}^{(k)} = \mathbf{A}^{-1}\mathbf{d}^{(k)}$$

reduces to

$$\mathbf{v}^{(k)} = \mathbf{C}^{-1}\mathbf{d}^{(k)}$$

i.e.

$$\mathbf{C}\mathbf{v}^{(k)} = \mathbf{d}^{(k)}$$

which takes only **small computational time**.

Linear iteration scheme

Then we can formulate the general linear iteration scheme

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \underbrace{\mathbf{C}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right)}_{\mathbf{v}^{(k)}}.$$

in which $\mathbf{C}$ can be chosen in different ways. With respect to the type of approximation one may distinguish:

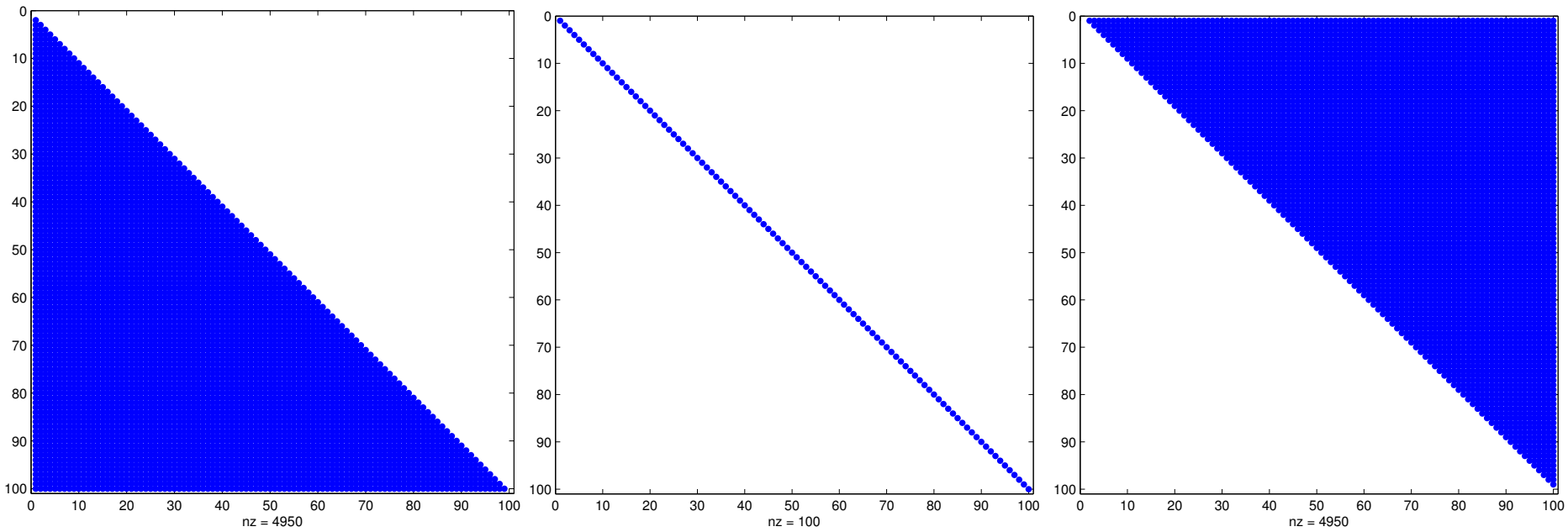
- Jacobi method
- Gauss-Seidel method
- The successive over relaxation method, etc.

These methods choose $\mathbf{C}$ from LDU decomposition of a matrix $\mathbf{A}$.

LDU decomposition

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U},$$

where **L** is a lower triangular matrix with main diagonal equals 0, D is a diagonal matrix und **U** is an upper triangular matrix with main diagonal equals zero. Decomposition of **A**:



$$L_{ij} = 0, j \geq i \quad \text{diag}(a_{ii}) \quad U_{ij} = 0, j \leq i$$

Jacobi method

Let us take that $\mathbf{C}$ is approximated only by diagonal

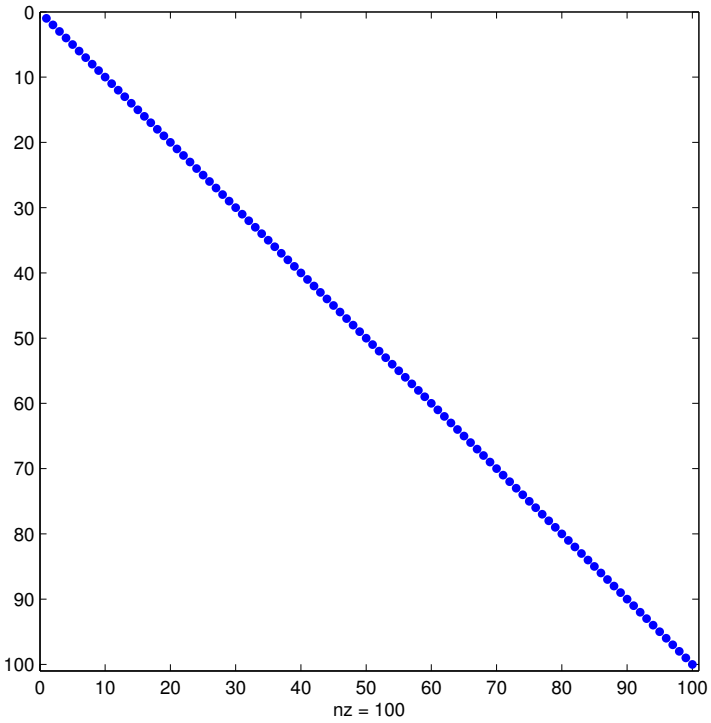
$$\mathbf{C} := \mathbf{D} = \mathrm{diag}(a_{ii}).$$

Then it follows

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \underbrace{\mathbf{D}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right)}_{\mathbf{v}^{(k)}}.$$

With $\mathbf{D}^{-1} = \mathrm{diag}\left(a_{ii}^{-1}\right)$ and

$$\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right)_i = b_i - \sum_{j=1}^{n} a_{ij} x_j^{(k)}$$



Jacobi method

The iterative procedure in element-wise form reads:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{\substack{j=1 \\ i \neq j}}^{n} a_{ij} x_j^{(k)} \right), \qquad i = 1, \ldots, n.$$

Hence, the values a_{ii} must be different than zero.

Exercise

Let us use Jacobi method to solve

$$\begin{aligned} 2x + \ \ y \ \ &= 11 \\ 5x + 7y &= 13 \end{aligned}$$

by starting at $\mathbf{x}^{(0)} = (1; 1)^T$. Hence,

$$\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 5 & 7 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 11 \\ 13 \end{pmatrix}$$

Exercise

The first iteration reads

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{D}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}\right)$$

where $\mathbf{D}$ is the diagonal part of the matrix $\mathbf{A}$:

$$\mathbf{D} = \begin{pmatrix} 2 & 0 \\ 0 & 7 \end{pmatrix}$$

with the inverse

$$\mathbf{D}^{-1} = \begin{pmatrix} 1/2 & 0 \\ 0 & 1/7 \end{pmatrix}.$$

Exercise

Hence,

$$\mathbf{x}^{(1)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1/2 & 0 \\ 0 & 1/7 \end{pmatrix} (\begin{pmatrix} 11 \\ 13 \end{pmatrix} - \begin{pmatrix} 2 & 1 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix})$$

$$\mathbf{x}^{(1)} = \begin{pmatrix} 5.0000 \\ 1.1429 \end{pmatrix}$$

After 22 iterations one obtains:

$$\mathbf{x}^{(22)} = \begin{pmatrix} 7.1110 \\ -3.2222 \end{pmatrix}$$

Exercise

Let us check solution

$$\mathbf{A}\mathbf{x}^{(22)} = \begin{pmatrix} 2 & 1 \\ 5 & 7 \end{pmatrix} \begin{pmatrix} 7.1110 \\ -3.2222 \end{pmatrix} = \begin{pmatrix} 11 \\ 13 \end{pmatrix}$$

Exercise

Let us use Jacobi method to solve

$$\begin{aligned} 3x + 2y - z &= 1 \\ 2x - 2y + 4z &= -2 \\ -x + \tfrac{1}{2}y - z &= 0 \end{aligned}$$

Hence,

$$\mathbf{A} = \begin{pmatrix} 3 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0.5 & -1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix}$$

Exercise

The diagonal part of the matrix is

$$\mathbf{D} = \begin{pmatrix} 3 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & -1 \end{pmatrix} \Rightarrow$$

$$\mathbf{D}^{-1} = \begin{pmatrix} 1/3 & 0 & 0 \\ 0 & -1/2 & 0 \\ 0 & 0 & -1/1 \end{pmatrix}$$

and the Jacobi method reads:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \underbrace{\mathbf{D}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right)}_{\mathbf{v}^{(k)}}.$$

Exercise

Start with

$$\mathbf{x}^{(0)} = \mathbf{0} = (0, 0, 0)^T$$

such that

$$\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \mathbf{D}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(0)}\right)$$

$$\mathbf{x}^{(1)} = \mathbf{0} + \mathbf{D}^{-1}\left(\begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix} - \begin{pmatrix} 3 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0.5 & -1 \end{pmatrix}\mathbf{0}\right)$$

holds.

Exercise

Hence,

$$\mathbf{x}^{(1)} = \begin{pmatrix} 1/3 & 0 & 0 \\ 0 & -1/2 & 0 \\ 0 & 0 & -1/1 \end{pmatrix} \begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/3 \\ 1 \\ 0 \end{pmatrix}$$

Similarly, one may write

$$\mathbf{x}^{(2)} = \begin{pmatrix} 1/3 \\ 1 \\ 0 \end{pmatrix} + \mathbf{D}^{-1} \left(\begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix} - \begin{pmatrix} 3 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0.5 & -1 \end{pmatrix} \begin{pmatrix} 1/3 \\ 1 \\ 0 \end{pmatrix} \right)$$

$$\mathbf{x}^{(2)} = \begin{pmatrix} -0.3333 \\ 1.3333 \\ 0.1667 \end{pmatrix}$$

Exercise

Iter	x_1	x_2	x_3
1	0.33	1.00	0.00
2	-0.33	1.33	0.17
3	-0.50	1.00	1.00
100	-0.43e6	1.94e6	1.29e6
1000	-0.29e59	1.3e59	0.87e59

However, by iterating one may notice that the solution does not converge. This happens even by changing the initial condition. The question is why this happens?

Convergence

From

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{D}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right) = F(\mathbf{x}^{(k)})$$

we may conclude that this is one kind of fixed point iteration scheme with the Lipschitz constant

$$F'(\mathbf{x}) = \mathbf{1} - \mathbf{D}^{-1}\mathbf{A}$$

Since

$$\mathbf{A} = \mathbf{D} + \mathbf{R}$$

in which $\mathbf{R}$ is the rest of the matrix ("residual ")

$$F'(\mathbf{x}) = \mathbf{1} - \mathbf{D}^{-1}(\mathbf{D} + \mathbf{R}) = -\mathbf{D}^{-1}\mathbf{R}$$

Convergence

To cheek if the method is convergent, one has to check contractivity

$$q = \sup \|F'\| = \sup \|\mathbf{D}^{-1}\mathbf{R}\|_2 < 1.$$

The last relation is equivalent to the condition for *the spectral radius*

$$\rho \stackrel{\text{def}}{=} \max_i(|\lambda_i|) < 1$$

where λ_i are the eigenvalues of $\mathbf{D}^{-1}\mathbf{R}$.

Exercise

In the first example one has

$$\mathbf{D}^{-1}\mathbf{R} = \begin{pmatrix} 0 & 0.5000 \\ 0.7143 & 0 \end{pmatrix}$$

Hence,

$$\rho = 0.5976 < 1$$

Exercise

In the second example one has

$$\mathbf{D}^{-1}\mathbf{R} = \begin{pmatrix} 0 & 0.6667 & -0.3333 \\ -1.0000 & 0 & -2.0000 \\ 1.0000 & -0.5000 & 0 \end{pmatrix}$$

Hence,

$$\rho = 1.14 > 1$$

Can we make this conclusion apriori?

In most of cases when the matrix **A** is **strictly or irreducibly diagonally dominant**, Jacobi method will converge. Strict row diagonal dominance means that for each row, the absolute value of the diagonal term is greater than the sum of absolute values of other terms:

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|.$$

Gauss-Seidel method

Going back to the general scheme

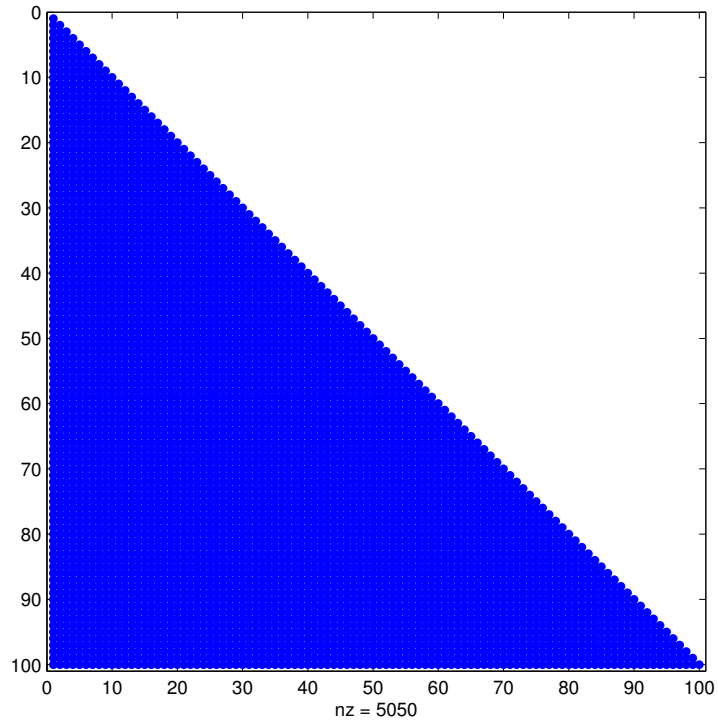
$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{C}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right).$$

we may take the approximation for $\mathbf{C}$ to be equal

$$\mathbf{C} := \mathbf{L} + \mathbf{D}$$

Then it follows

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + (\mathbf{L} + D)^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right).$$



Gauss-Seidel method

However, **we said before that inversion is avoided in numerical schemes**. Due to this reason, let us multiply both sides of equation by $(\mathbf{L}+\mathbf{D})$:

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

Having that $\mathbf{A} = \mathbf{L}+\mathbf{D}+\mathbf{U}$ we obtain

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

Splitting the left side of equation, one obtains

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

Gauss-Seidel method

In component form one obtains

$$a_{ii}x_i^{(k+1)} = b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)}, \qquad i = 1, \ldots, n.$$

and

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k)} \right] \qquad i = 1, \ldots, n.$$

Exercise

Let us use Gauss-Seidel method to solve

$$\begin{aligned} 2x + \ \ y \ \ &= 11 \\ 5x + 7y &= 13 \end{aligned}$$

by starting at $\mathbf{x}^{(0)} = (1; 1)^T$. Hence,

$$\mathbf{A} = \begin{pmatrix} 2 & 1 \\ 5 & 7 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 11 \\ 13 \end{pmatrix}$$

Exercise

The first iteration reads

$$x_i^{(1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(0)} \right] \qquad i = 1, \ldots, n.$$

$$x_1^{(1)} = \frac{1}{a_{11}} [b_1 - \sum_{j=2}^{2} a_{1j} x_j^{(0)}] = 5$$

$$x_2^{(1)} = \frac{1}{a_{22}} [b_2 - \sum_{j=1}^{1} a_{2j} x_j^{(1)}] = -1.7143$$

It takes approximately the same number of iterations as Jacobi to get the solution.

Exercise

Let us use Gauss-Seidel method to solve

$$\begin{aligned} 3x + 2y - z &= 1 \\ 2x - 2y + 4z &= -2 \\ -x + \tfrac{1}{2}y - z &= 0 \end{aligned}$$

Hence,

$$\mathbf{A} = \begin{pmatrix} 3 & 2 & -1 \\ 2 & -2 & 4 \\ -1 & 0.5 & -1 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ -2 \\ 0 \end{pmatrix}$$

Exercise

The first iteration reads

$$x_i^{(1)} = \frac{1}{a_{ii}} \left[b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(0)} \right]$$

$$x_1^{(1)} = \frac{1}{a_{11}} [b_1 - \sum_{j=2}^{3} a_{1j} x_j^{(0)}] = 0.333$$

$$x_2^{(1)} = \frac{1}{a_{22}} [b_2 - \sum_{j=1}^{1} a_{2j} x_j^{(1)} - \sum_{j=3}^{3} a_{2j} x_j^{(0)}] = -1.333$$

$$x_3^{(1)} = \frac{1}{a_{33}} [b_3 - \sum_{j=1}^{2} a_{3j} x_j^{(1)}] = 0.333$$

Exercise

Similarly to the Jacobi method, the Gauss-Seidel does not converge for this system of equations.

Convergence

From

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + (\mathbf{L} + \mathbf{D})^{-1} \left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)} \right).$$

we may conclude that this is one kind of fixed point iteration scheme with the Lipschitz constant

$$F'(\mathbf{x}) = \mathbf{1} - (\mathbf{L} + \mathbf{D})^{-1}\mathbf{A}$$

Since

$$\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$$

one has

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

Convergence

Thus, the method is convergent if the spectral radius of

$$\rho((\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}) < 1$$

The convergence properties of the Gauss–Seidel method are dependent on the matrix A. Namely, the procedure is known to converge if either:

- A is symmetric positive-definite (symmetric $A = A^T$, positive definite matrix is a symmetric matrix A for which all eigenvalues are positive).
- A is strictly or irreducibly diagonally dominant.

Convergence

In the first example one has that

$$\rho((\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}) = 0.3571 < 1$$

and in the second example

$$\rho((\mathbf{L}+\mathbf{D})^{-1}\mathbf{U}) = 1.24 > 1$$

Successive over relaxation method (SOR)

The SOR-scheme tries to improve the convergence properties of the Gauß-Seidel method. The idea is to introduce a relaxation parameter $\omega > 0$ such that

$$\mathbf{C} := \frac{1}{\omega}\mathbf{D} + \mathbf{L}$$

Inserting yields

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \left(\frac{1}{\omega}\mathbf{D} + \mathbf{L}\right)^{-1} \left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right).$$

Multiplying by $\frac{1}{\omega}\mathbf{D} + \mathbf{L}$, and using $A = \mathbf{L} + \mathbf{D} + \mathbf{U}$ one obtains

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

SOR method

Then

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

and

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

The representation with components is now given by

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

and

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

SOR method

Note that when $\omega = 1$ the method

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

becomes

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

which is Gauß-Seidel method.

SOR: how to choose ω?

With an optimal choice of ω we can improve the convergence of the SOR-method in comparison with the Gauß-Seidel method. But the optimal value for ω depends on the problem, i.e. in general one has to try several values. One should start with larger values, i.e. $\omega = 1.7$, then $\omega = 1.3$ ... and take a look on the convergence behaviour.

Convergence

From

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

we may conclude that this is one kind of fixed point iteration scheme with the Lipschitz constant

$$F'(\mathbf{x}) = \mathbf{1} - \left(\frac{1}{\omega}\mathbf{D} + \mathbf{L}\right)^{-1}\mathbf{A}$$

and hence the spectral radius must satisfy

$$\rho(\mathbf{1} - \left(\frac{1}{\omega}\mathbf{D} + \mathbf{L}\right)^{-1}\mathbf{A}) < 1$$

to get convergence.

Convergence

For $\omega = 1.7$ the spectral radius in the previous two examples becomes

$$\rho(\mathbf{1} - \left(\frac{1}{\omega}\mathbf{D} + \mathbf{L}\right)^{-1} A) = 0.7 < 1$$

and in the second example

$$\rho(\mathbf{1} - \left(\frac{1}{\omega}\mathbf{D} + \mathbf{L}\right)^{-1} \mathbf{A}) = 3.008 > 1$$

General convergence results

The equation

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{C}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right).$$

for the general iteration method can be transformed into

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{C}^{-1}\left(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}\right) = \mathbf{C}^{-1}\mathbf{b} + (I - \mathbf{C}^{-1}\mathbf{A})\mathbf{x}^{(k)}$$

i.e.

$$\mathbf{x}^{(k+1)} = \mathbf{M}\mathbf{x}^{(k)} + \mathbf{g}$$

in which $\mathbf{M} = I - \mathbf{C}^{-1}\mathbf{A}$ and $\mathbf{g} = \mathbf{C}^{-1}\mathbf{b}$. This equation is often called normal form. Every linear iteration method can be written in normal form. The matrix $\mathbf{M}$ is called iteration matrix of the method.

General convergence results

For the derivation of convergence criteria we need the spectral radius of a matrix which can be defined in the following way:

Definition

For a given matrix $\mathbf{M} \in \mathbb{R}^{n,n}$ *let* λ_i, $i = 1, \ldots, n$ *be the eigenvalues of* $\mathbf{M}$. *Then*

$$\rho(\mathbf{M}) := \max_{i=1,\ldots,n} |\lambda_i|$$

is called the spectral radius of $\mathbf{M}$.

General convergence results

Theorem

The linear iteration method with the iteration matrix $\mathbf{M}$ *converges for an arbitrary start vector* $\mathbf{x}^{(0)}$ *if and only if* $\rho(\mathbf{M}) < 1$ *and if in a dedicated matrix norm* $\|\cdot\|$, *the condition* $\|\mathbf{M}\| \leq 1$ *is satisfied. In this case the following error estimates are true:*

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}_*\right\| \leq \frac{\|\mathbf{M}\|^k}{1 - \|\mathbf{M}\|} \left\|\mathbf{x}^{(1)} - \mathbf{x}^{(0)}\right\|, \tag{1}$$

$$\left\|\mathbf{x}^{(k)} - \mathbf{x}_*\right\| \leq \frac{\|\mathbf{M}\|}{1 - \|\mathbf{M}\|} \left\|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right\|. \tag{2}$$

Inequality (1) *is called a priori estimate and* (2) *is called a posteriori estimate.*

Literature

- C. T. Kelley, Iterative Methods for Linear and Nonlinear Equations
- Yousef Saad, Iterative methods for sparse linear systems
- Martin Stoll, Solving Linear Systems using the Adjoint

Conjugate Gradient Method

In order to solve

$$Ax = b \tag{3}$$

let us observe the function

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c$$

and find its derivative (Jacobian)

$$f'(x) = Ax - b. \tag{4}$$

By comparing Eq. (1) and Eq. (2) we may conclude that they are the same in case when

$$f'(x) = 0$$

Conjugate gradient method

The condition

$$f'(x) = 0$$

is actually equivalent to finding the maximum or minimum of a function $f(x)$. Assuming that A is positive definite (this means that the term $x^T Ax$ is always positive no matter how we choose the non-zero vector x) and symmetric, one may prove that x satisfying

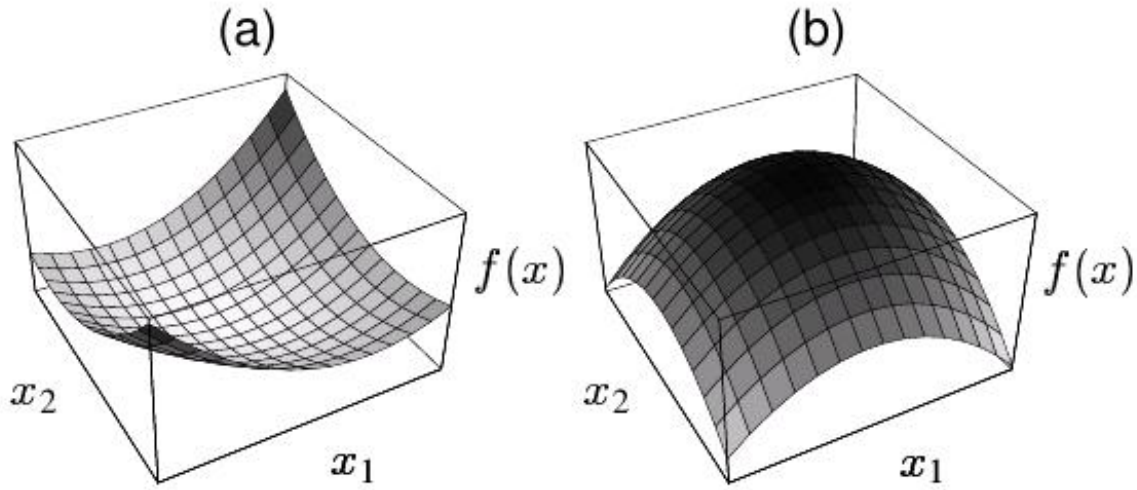
$$f'(x) = 0$$

is the minimum of a function $f(x)$, i.e.

$$x = \min_x \left(\frac{1}{2} x^T Ax - b^T x + c \right)$$

Conjugate gradient method

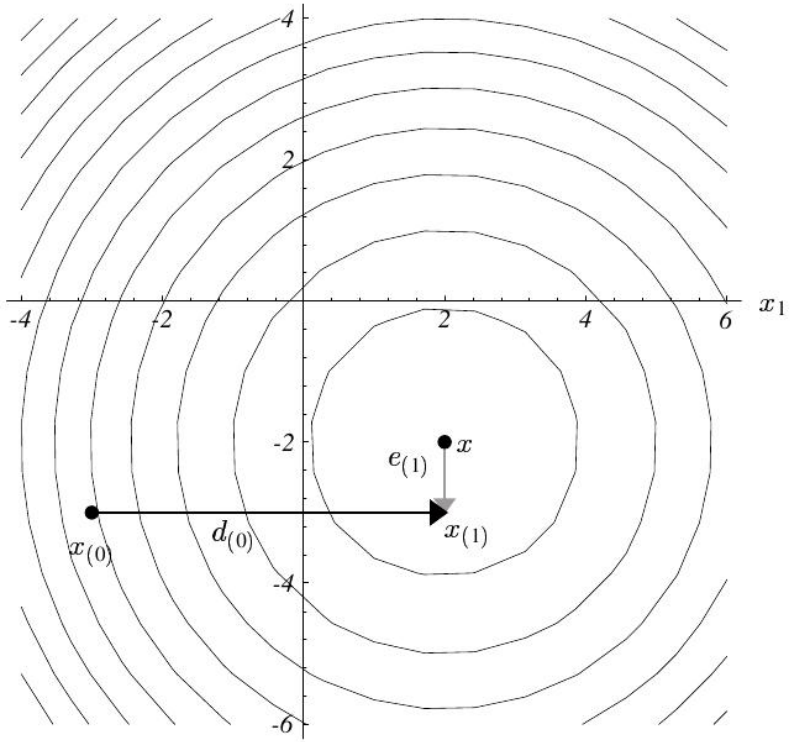
Graph a) represents $f(x)$ for positive definite matrix A and graph b) $f(x)$ for negative definite matrix A.



The fact that $f(x)$ is paraboloid helps us to understand why x is minimum of the function $f(x)$

Conjugate gradient method

The idea of this algorithm is to find minimum by starting from some point $x_{(0)}$ and then move one step in first search direction $d_{(0)}$, then one step in **orthogonal** search direction $d_{(1)}$ and so on until we hit the minimum.



Conjugate gradient method

Thus, we move according to the following rule

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

where $\alpha_{(i)}$ is the step length and $d_{(i)}$ are orthogonal directions.After moving in iteration (i), one may compute the error between the new position $x_{(i)}$ and the real minimum x:

$$e_{(i)} = x_{(i)} - x.$$

Similarly, the error between the iteration $(i+1)$ and the real minimum x is

$$e_{(i+1)} = x_{(i+1)} - x = x_{(i)} + \alpha_{(i)} d_{(i)} - x = e_{(i)} + \alpha_{(i)} d_{(i)}$$

Conjugate gradient method

Hence, for now we have two important equations:

- change of position

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \tag{5}$$

- error propagation (measures how far are we from the real position)

$$e_{(i+1)} = e_{(i)} + \alpha_{(i)} d_{(i)} \tag{6}$$

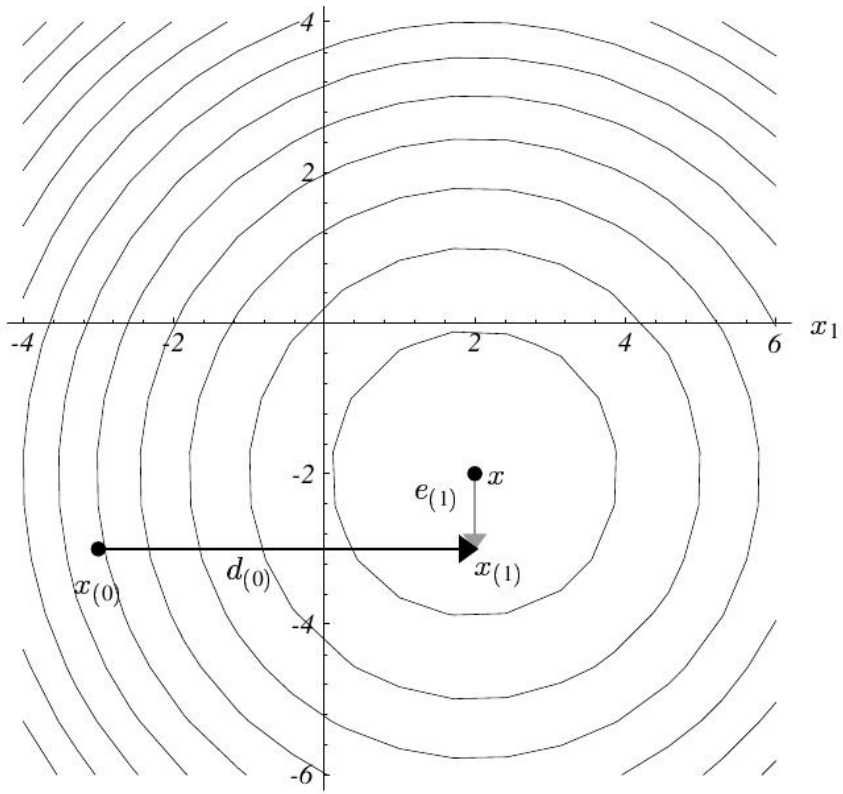
Conjugate gradient method

Because we move in orthogonal directions our error $e_{(i+1)}$ is orthogonal to the direction $d_{(i)}$, i.e.

$$d_{(i)}^T e_{(i+1)} = 0$$

By taking $e_{(i+1)} = e_{(i)} + \alpha_{(i)} d_{(i)}$ the last realtion becomes

$$d_{(i)}^T (e_{(i)} + \alpha_{(i)} d_{(i)}) = 0.$$



From this one we then may compute the step length

$$\alpha_{(i)} = -\frac{d_{(i)}^T e_{(i)}}{d_{(i)}^T d_{(i)}}$$

Conjugate gradient method

Looking at

$$\alpha_{(i)} = -\frac{d_{(i)}^T e_{(i)}}{d_{(i)}^T d_{(i)}}$$

we may notice one thing:

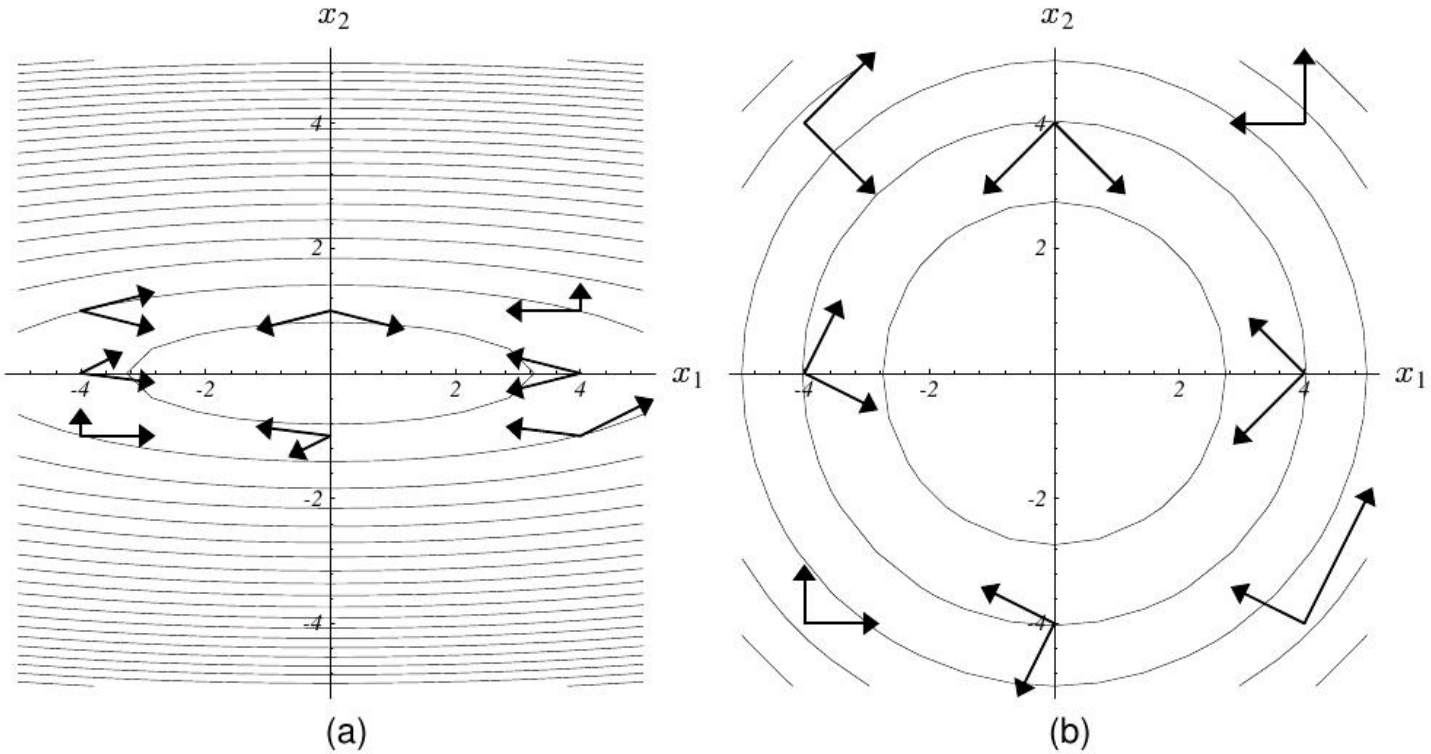
We do not know $e_{(i)} = x_{(i)} - x$ because x is something we want to compute!!

Solution: do not take orthogonal directions, but A-**orthogonal directions (also known as conjugate)**.

Conjugate gradient method

A-orthogonal (conjugate) directions $d_{(i)}$ and $d_{(j)}$ are the directions which satisfy

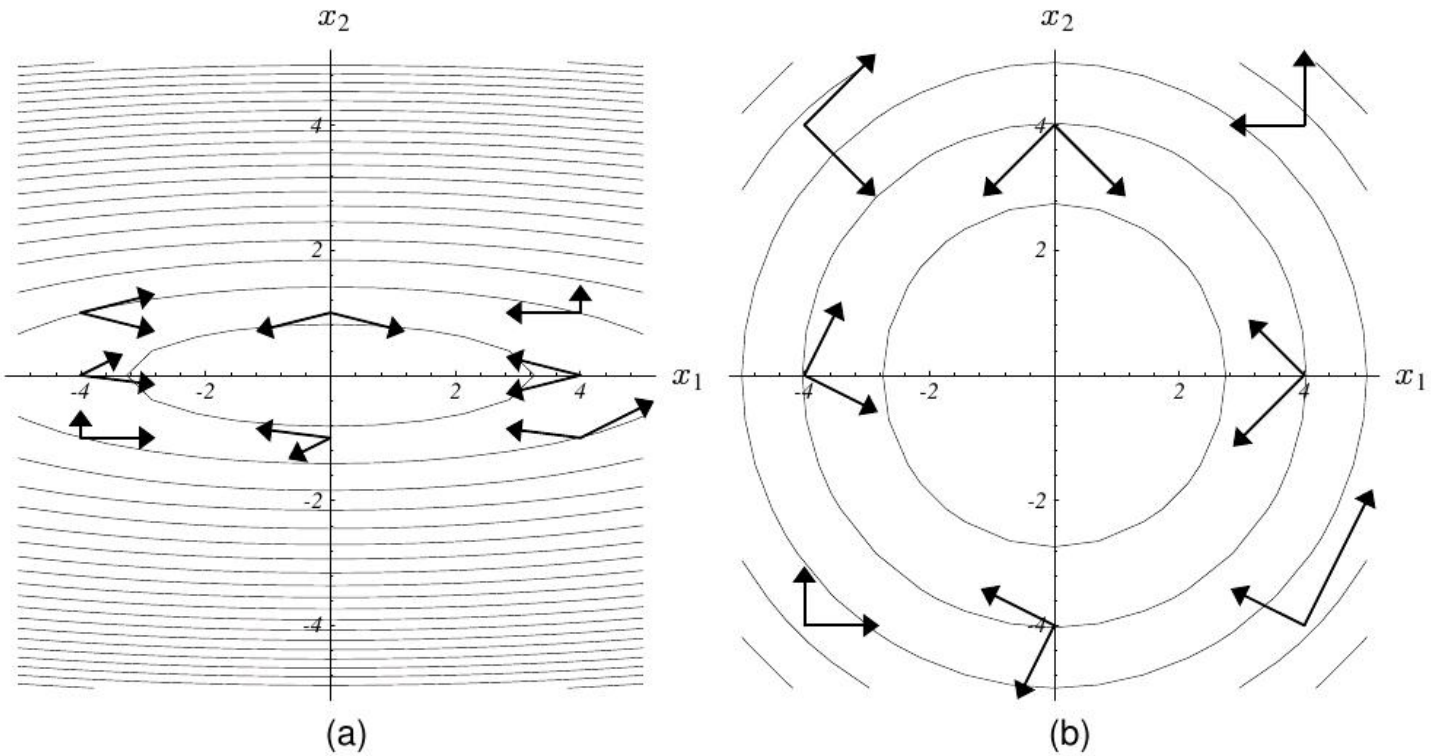
$$\langle d_{(i)}, d_{(j)} \rangle_A = d_{(i)}^T A d_{(j)} = 0$$



Conjugate gradient method

Similarly, it must hold that

$$d_{(i)}^T A e_{(i+1)} = 0$$



Conjugate gradient method

Now from $d_{(i)}^T A e_{(i+1)} = 0$ one has

$$d_{(i)}^T A(e_{(i)} + \alpha_{(i)} d_{(i)}) = 0$$

which leads to

$$\alpha_{(i)} = -\frac{d_{(i)}^T A e_{(i)}}{d_{(i)}^T A d_{(i)}}$$

The term

$$Ae_{(i)} = A(x_{(i)} - x) = Ax_{(i)} - Ax = Ax_{(i)} - b = -r_{(i)}$$

leads us to the **residual**

$$r_{(i)} = b - Ax_{(i)}$$

Conjugate gradient method

Finally, one has

- step length

$$\alpha_{(i)} = \frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}} \tag{7}$$

- change of position

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)} \tag{8}$$

- error propagation

$$e_{(i+1)} = e_{(i)} + \alpha_{(i)} d_{(i)} \tag{9}$$

But, the problem is that we also do not have A-orthogonal directions $d_{(i)}$.

Conjugate gradient method

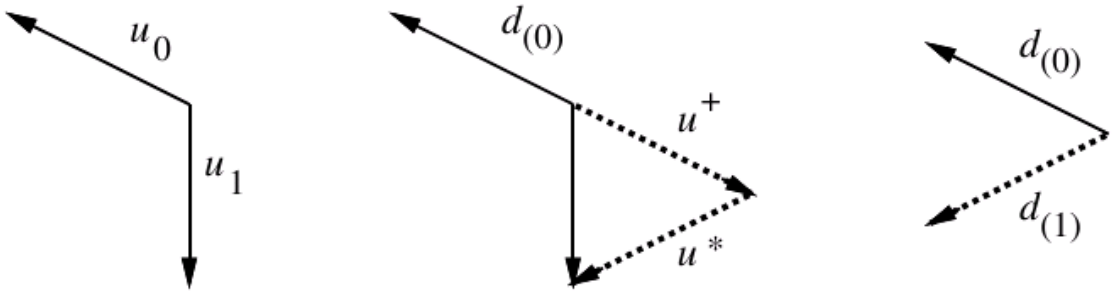
The question is how to choose $d_{(i)}$'s?

Conjugate Gram-Schmidt process

Suppose that $u_{(0)}, \ldots u_{(m)}$ form a basis in a vector space. The Gram-Schmidt process means orthogonalisation of the previous basis. The procedure starts by choosing:

$$d_{(0)} = u_{(0)}.$$

Then the next vector $u_{(1)}$ is decomposed into the vector u^+ parallel to $d_{(0)}$ and the vector u^* A-orthogonal to it.



Projection of a vector onto vector

The parallel vector can be obtained by projecting $u_{(1)}$ onto $d_{(0)}$ such that

$$u^+ = proj_{d_{(0)}} u_{(1)} = |u_{(1)} \cos\theta| \hat{d}_{(0)}$$

in which θ is the angle between $u_{(1)}$ and $d_{(0)}$, and $\hat{d}_{(0)}$ is the unit vector. From scalar product of vectors

$$\langle u_{(1)}, d_{(0)} \rangle = |u_{(1)}||d_{(0)}| \cos\theta, \quad \hat{d}_{(0)} = \frac{d_{(0)}}{|d_{(0)}|}$$

one obtains the formula

$$u^+ = proj_{d_{(0)}} u_{(1)} = \frac{\langle u_{(1)}, d_{(0)} \rangle}{\langle d_{(0)}, d_{(0)} \rangle} d_{(0)}$$

Conjugate Gram-Schmidt process

However, in case of conjugate Gram-Schmidt process we search for A-orthogonal projection and hence the last relation

$$u^+ = \frac{\langle u_{(1)}, d_{(0)} \rangle}{\langle d_{(0)}, d_{(0)} \rangle} d_{(0)}$$

transforms to

$$u^+ = \frac{\langle u_{(1)}, d_{(0)} \rangle_A}{\langle d_{(0)}, d_{(0)} \rangle_A} d_{(0)}$$

in which

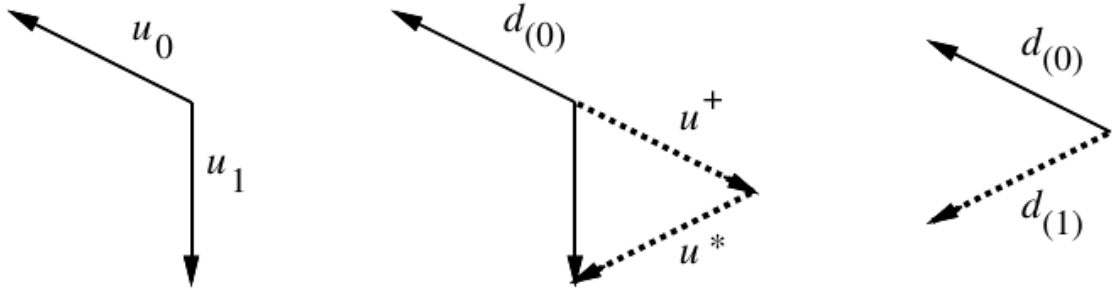
$$\langle u_{(1)}, d_{(0)} \rangle_A = u_{(1)}^T A d_{(0)}, \quad \langle d_{(0)}, d_{(0)} \rangle_A = d_{(0)}^T A d_{(0)}$$

Conjugate Gram-Schmidt process

Once we have u^+ we may compute u^* by substruction

$$u^* = u_{(1)} - u^+ = u_{(1)} - \frac{u_{(1)}^T A d_{(0)}}{d_{(0)}^T A d_{(0)}} d_{(0)} =: d_{(1)}$$

The last relation is exactly our next search direction $d_{(1)}$.



Conjugate Gram-Schmidt process

Furthermore, we may compute the search direction $d_{(2)}$ by taking $u_{(2)}$ and projecting both $d_{(0)}$ and $d_{(1)}$ onto $u_{(2)}$ such that

$$u^* = u_{(2)} - u^+_{(0)} - u^+_{(1)}$$

holds. Here, $u^+_{(0)}$ is A-orthogonal projection of $u_{(2)}$ onto $d_{(0)}$ and $u^+_{(1)}$ is the A-orthogonal projection of $u_{(2)}$ onto $d_{(1)}$. Hence,

$$d_{(2)} = u_{(2)} - \frac{u^T_{(2)} A d_{(0)}}{d^T_{(0)} A d_{(0)}} d_{(0)} - \frac{u^T_{(2)} A d_{(1)}}{d^T_{(1)} A d_{(1)}} d_{(1)}$$

Conjugate Gram-Schmidt process

Finally, we may write general formula for computation of A-orthogonal directions

$$d_{(i)} = u_{(i)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)}$$

in which

$$\beta_{ik} = -\frac{u_{(i)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}}, \quad i > k$$

Note that this is expencive operation ($\mathcal{O}(n^3)$) since one has to keep all the old search vectors to construct the new one.

Conjugate gradient method

The conjugate gradient method uses conjugate Gram-Schmidt process by taking for the vector set $u_{(0)}, ..., u_{(m)}$ the set of residuals $r_{(0)}, ...r_{(m)}$. Then, we walk in the solution space by

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

where

$$\alpha_{(i)} = \frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}}, \quad r_{(i)} = b - Ax_{(i)}$$

and

$$d_{(i)} = r_{(i)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)}, \quad \beta_{ik} = -\frac{r_{(i)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}}, \quad i > k$$

.

Conjugate gradient method

From

$$d_{(i)}^T A e_{(i+1)} = 0.$$

and $r_{(i+1)} = -Ae_{(i+1)}$ one concludes

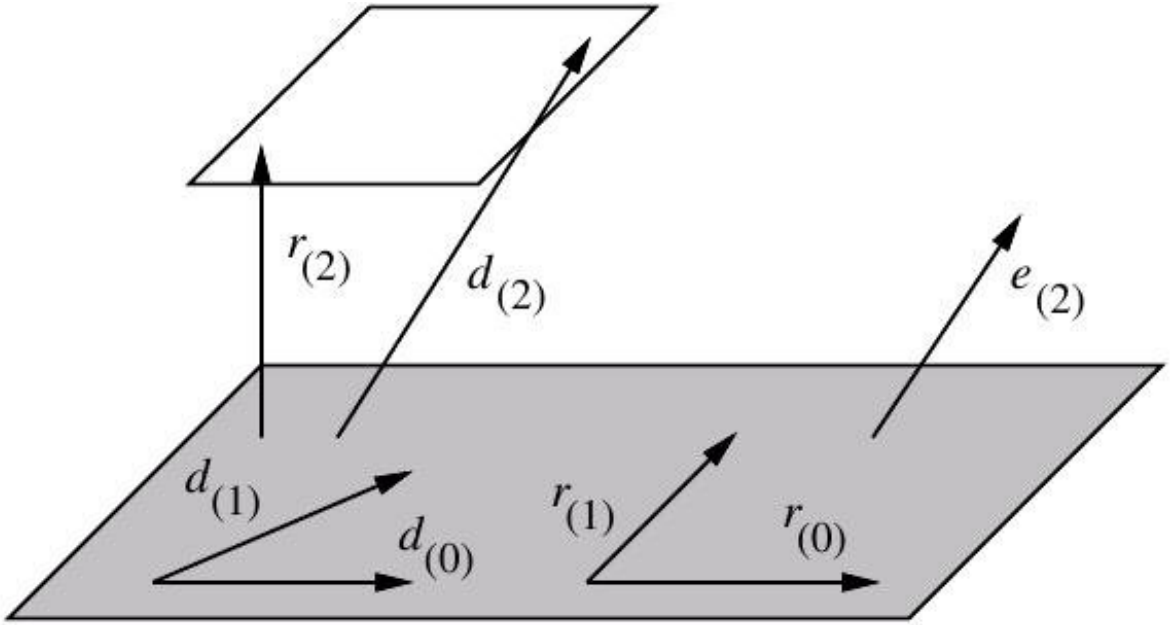
$$d_{(i)}^T r_{(i+1)} = 0$$

or more generally

$$d_{(i)}^T r_{(j)} = 0, \quad j > i$$

Hence, the residuals are **orthogonal** (not that there is no prefix A) to the seacrh directions.

Conjugate gradient method



Conjugate gradient method

Furthermore, since $r_{(i)}$ are taken to be $u_{(i)}$ one may write

$$d_{(i)} = r_{(i)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)}$$

Taking the inner product of the last one with $r_{(j)}$ one obtains

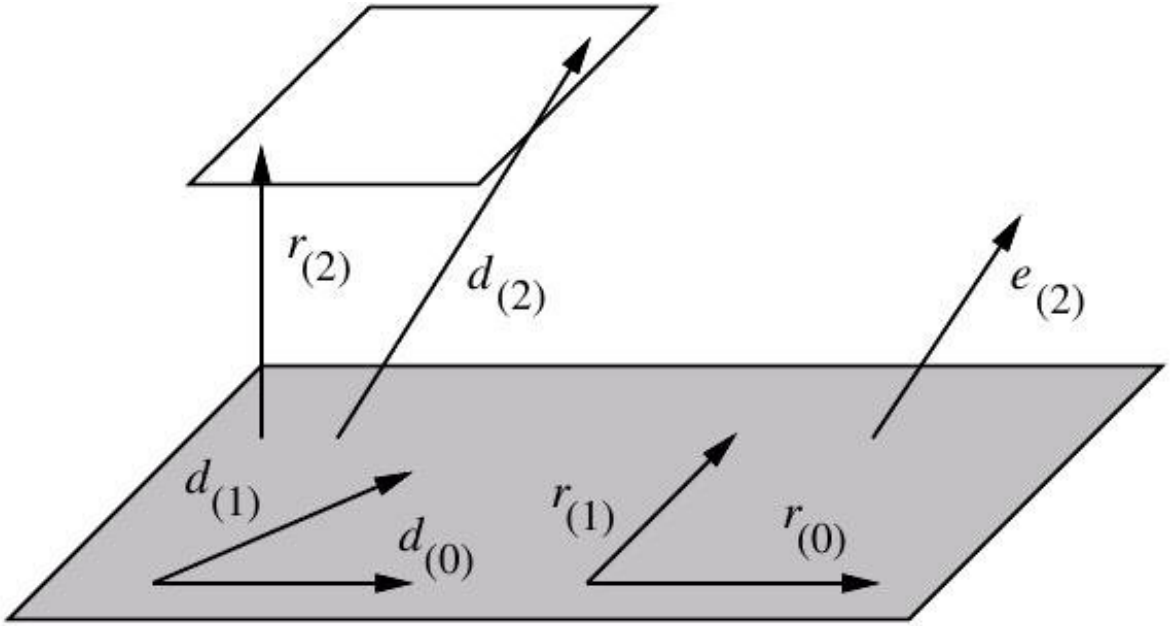
$$d_{(i)}^T r_{(j)} = r_{(i)}^T r_{(j)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)}^T r_{(j)}$$

and using orthogonality condition $d_{(i)}^T r_{(j)} = 0, j > i$ one obtains

$$r_{(i)}^T r_{(j)} = 0, \quad j > i, \quad \text{as well as} \quad d_{(i)}^T r_{(j)} = r_{(i)}^T r_{(j)}.$$

Conjugate gradient method

Thus, each new residual is made to be orthogonal to the all previous residuals and search directions, and each new search direction is made from residual to be A-orthogonal to all previous residuals and search directions.



Conjugate gradient method

Furthermore, from

$$r_{(i+1)} = -Ae_{i+1} = -A(e_{(i)} + \alpha_{(i)} d_{(i)})$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$$

one concludes that the residual $r_{(i+1)}$ is only linear combination of the previous residual $r_{(i)}$ and $Ad_{(i)}$. Hence, our search space is Krylov subspace

$$\mathcal{K} = \text{span} \quad \{d_{(0)}, Ad_{(0)}, ..., A^i d_{(0)}\}$$

Having that $r_{(0)} = d_{(0)}$ this is the same as

$$\mathcal{K} = \text{span} \quad \{r_{(0)}, Ar_{(0)}, ..., A^i r_{(0)}\}$$

Conjugate gradient method

To simplify formulas

$$x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(i)}$$

where

$$\alpha_{(i)} = -\frac{d_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}}, \quad r_{(i)} = b - A x_{(i)}$$

and

$$d_{(i)} = r_{(i)} + \sum_{k=0}^{i-1} \beta_{ik} d_{(k)}, \quad \beta_{ik} = -\frac{r_{(i)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}}, \quad i > k$$

.

Conjugate gradient method

one may take the inner product of

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$$

and $r_{(j)}$ such that

$$r_{(j)}^T r_{(i+1)} = r_{(j)}^T r_{(i)} + \alpha_{(i)} r_{(j)}^T A d_{(i)}$$

holds. This then leads to

$$r_{(j)}^T A d_{(i)} = \frac{1}{\alpha_{(i)}} \left[r_{(j)}^T r_{(i)} - r_{(j)}^T r_{(i+1)} \right]$$

The right hand side will be different from zero only when $i = j$ or $i + 1 = j$ (due to orthogonality of residuals).

Conjugate gradient method

This simplifies the coefficients

$$\beta_{i,i-1} = -\frac{r_{(i)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}}, \quad i > k$$

because their number drastically reduces to

$$\beta_{i,i-1} = -\frac{r_{(i)}^T A d_{(i-1)}}{d_{(i-1)}^T A d_{(i-1)}} = \frac{1}{\alpha_{(i-1)}} \frac{r_{(i)}^T r_{(i)}}{d_{(i-1)}^T A d_{(i-1)}}$$

In addition, from $d_{(i)}^T r_{(j)} = r_{(i)}^T r_{(j)}$ and $r_{(i)} = r_{(i-1)} - \alpha_{(i-1)} A d_{(i-1)}$ one obtains

$$\beta_{(i)} := \beta_{i,i-1} = \frac{r_{(i)}^T r_{(i)}}{r_{(i-1)}^T r_{(i-1)}}$$

Conjugate gradient method

Finally, the CG algorithm looks like this

$$d_{(0)} = r_{(0)} = b - Ax_{(0)}$$

$$\alpha_{(i)} = \frac{r_{(i)}^T r_{(i)}}{d_{(i)}^T A d_{(i)}}$$

$$x_{(i+1)} = x_{(i)} + \alpha_i d_{(i)}$$

$$r_{(i+1)} = r_{(i)} - \alpha_{(i)} A d_{(i)}$$

$$\beta_{i+1} = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}}$$

$$d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)}$$

Conjugate gradient method

The number of iterations neccessary to obtain the error ϵ is given as

$$k \leq \frac{1}{2}\kappa ln(\frac{1}{\epsilon})$$

where κ is condition number of matrix A (associated with the linear equation $Ax = b$ the condition number gives a bound on how inaccurate the solution x will be after approximation).

$$\kappa(A) = \left\|A^{-1}\right\| \cdot \|A\| .$$

Hence, the larger condition number the slower algorithm is.

Preconditioned conjugate gradient method

To speed up convergence, one may try to altrenate the condition number of a matrix A by premultiplying the system of equations

$$Ax = b$$

by **preconditioner** M: a positive-definite matrix that approximates A but is easier to invert. Then,

$$M^{-1}(Ax - b) = 0$$

in which t$\kappa(M^{-1}A) << \kappa(A)$. This may lead us to faster convergence, but the problem is that $M^{-1}A$ is not neccessarily symmeteric an positive definite.

Preconditioned conjugate gradient method

To overcome this issue, one may perform the Cholesky decomposition of a matrix M

$$M = QQ^T$$

such that one solves the problem

$$Q^{-1}AQ^{-T}\hat{x} = Q^{-1}b =: \hat{b}, \quad \hat{x} = Q^T x$$

This system is characterised by a matrix

$$\hat{A} = Q^{-1}AQ^{-T}$$

which is symmetric and positive definite.

Preconditioned conjugate gradient method

Some of possible choices of perconditioner are

- diagonal : $M = \operatorname{diag} A$ (also known as Jacobi preconditioner)
- incomplete Cholesky preconditioner (The Cholesky factorization of a positive definite matrix A is $A = LL^*$ where L is a lower triangular matrix. An incomplete Cholesky factorization is given by a sparse lower triangular matrix K that is in some sense close to L. The corresponding preconditioner is KK^*.)

Exercise

Let us solve by CG method $Ax = b$ where

$$A = \begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad x_{(0)} = 0$$

Hence, we start with

$$d_{(0)} = r_{(0)} = b - Ax_{(0)} = b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Then we compute

$$\alpha_{(0)} = \frac{r_{(0)}^T r_{(0)}}{d_{(0)}^T A d_{(0)}} = \frac{1}{2}, \quad x_{(1)} = x_{(0)} + \alpha_0 d_{(0)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \frac{1}{2}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0.5 \\ 0 \end{pmatrix}$$

Exercise

Let us check if $x_{(1)}$ is the solution? Compute

$$r_{(1)} = r_{(0)} - \alpha_0 A d_{(0)} = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix}$$

which is the same as

$$r_{(1)} = b - Ax_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ -0.5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix}$$

Check if $r_{(0)}$ and $r_{(1)}$ are orthogonal: $r_{(1)}^T r_{(0)} = 0$. Also check if r_1 and $d_{(0)}$ are orthogonal $d_{(0)}^T r_1 = 0$. Since both hold for now we are fine.

Exercise

Let us now compute the other seacrh direction

$$d_{(1)} = r_{(1)} + \beta_{(1)} d_{(0)}$$

where

$$\beta_1 = \frac{r_{(1)}^T r_{(1)}}{r_{(0)}^T r_{(0)}} = \frac{1}{4}$$

$$d_{(1)} = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix} + \frac{1}{4} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1/4 \\ 0.5 \end{pmatrix}$$

Check if $d_{(1)}$ si A-orthogonal to $d_{(0)}$:

$$d_{(1)}^T A d_{(0)} = 0$$

Exercise

Now compute

$$\alpha_{(1)} = \frac{r_{(1)}^T r_{(1)}}{d_{(1)}^T A d_{(1)}} = \frac{2}{3}$$

and new solution

$$x_{(2)} = x_{(1)} + \alpha_1 d_{(1)} = \begin{pmatrix} 0.5 \\ 0 \end{pmatrix} + \frac{2}{3} \begin{pmatrix} 1/4 \\ 1/2 \end{pmatrix} = \begin{pmatrix} 2/3 \\ 1/3 \end{pmatrix}$$

and residual

$$r_{(2)} = r_{(1)} - \alpha_1 A d_{(1)} = \begin{pmatrix} 0 \\ 0.5 \end{pmatrix} - \begin{pmatrix} 0 \\ 0.5 \end{pmatrix}) = 0$$

Hence, we got correct solution.

Exercise

Let us solve by PCG method $Ax = b$ where

$$A = \begin{pmatrix} 4 & 1 \\ 1 & 3 \end{pmatrix}, \quad b = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, \quad x_{(0)} = (2 \quad 1)^T$$

Hence, we may assume the diagonal preconditioner:

$$M = \begin{pmatrix} 4 & 0 \\ 0 & 3 \end{pmatrix}$$

The transformed system of equations becomes $\hat{A}x = M^{-1}Ax = M^{-1}b = \hat{b}$

Exercise

The new system $\hat{A}x = \hat{b}$ has matrix

$$\hat{A} = \begin{pmatrix} 1.0000 & 0.2500 \\ 0.3333 & 1.0000 \end{pmatrix}$$

Its condition number is $\kappa_{\hat{A}} = 1.82$, whereas the condition number of A is $\kappa_A = 1.94$. Hence, we get slight imporvement. This shows that we can suggest another preconditioner (but about this some other time).

In this particular case the matrix $\hat{A}$ is positive-definite, and one does not need to do Cholesky decomposition.

Exercise

The process of solving then follows the previous example.
Hence, we start with

$$d_{(0)} = r_{(0)} = \hat{b} - \hat{A}x_{(0)}$$

$$= \begin{pmatrix} 1.5 \\ 2.33 \end{pmatrix} - \begin{pmatrix} 1.0000 & 0.2500 \\ 0.3333 & 1.0000 \end{pmatrix} \begin{pmatrix} 2 \\ 1 \end{pmatrix} = \begin{pmatrix} -0.7500 \\ 0.6667 \end{pmatrix}$$

$$\alpha_{(0)} = \frac{r_{(0)}^T r_{(0)}}{d_{(0)}^T \hat{A} d_{(0)}}$$

and so on.

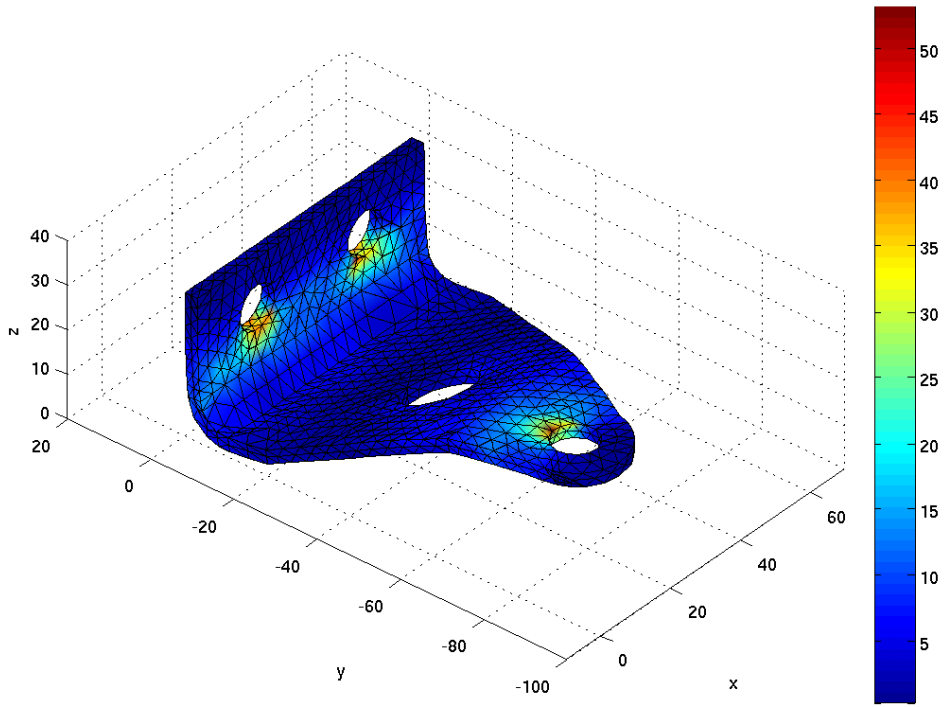
In this particular case the matrix $\hat{A}$ is positive-definite, and one does not need to do Cholesky decomposition.

Solving finite element system

If we do linear elastic analysis of an iron piece of hardware (fixed at two holes and pulled at the third hole in z direction) we end up with the solving the following system of equations

$$Ax = b$$

in which A is the stiffness matrix and b is the force vector (external and internal forces).



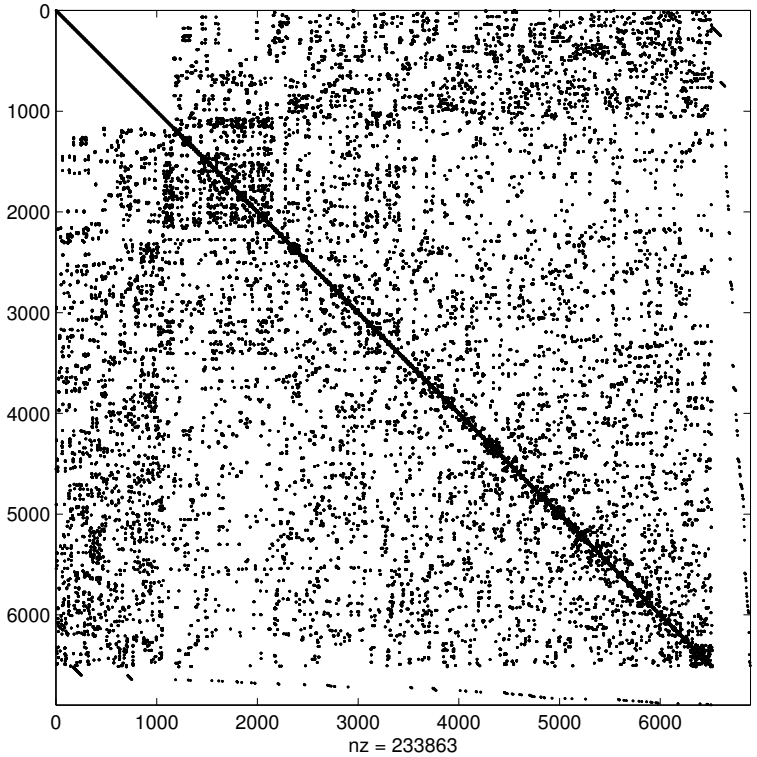
@Carstensen

Solving finite element system

The system has 6894 degrees of freedom (size of matrix A). The total number of elements in A is 47527236, but only 233863 of them are non-zero. Hence, the matrix is sparse. Without knowing this we may solve the system directly by inverting the matrix A

$$x = A^{-1}b = \text{inv}\ (A)b$$

The time neccessary to do that is 179.214366 seconds.

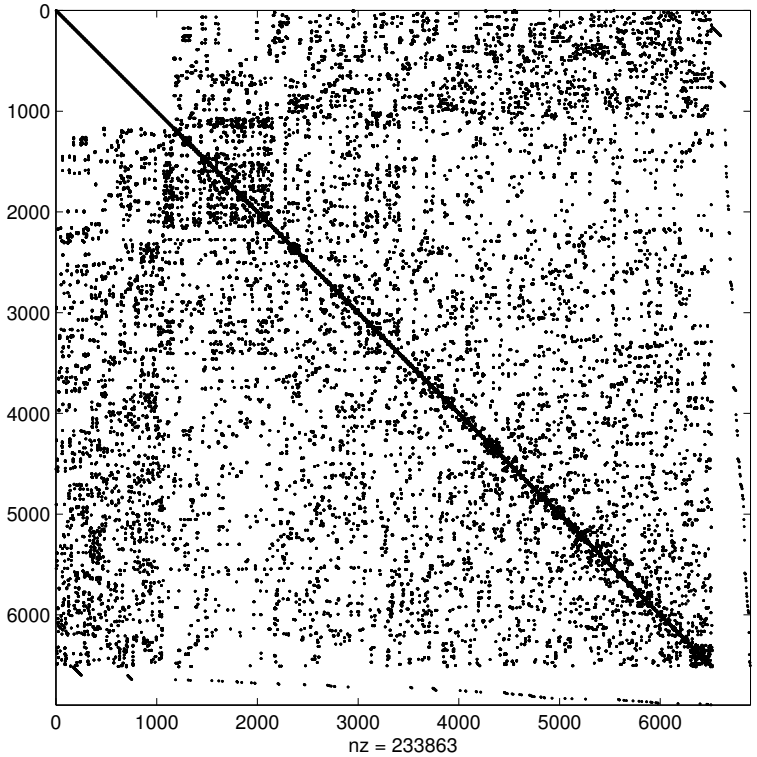


Solving finite element system

In case that you know that the matrix is sparse then you may use the command

$$x = A^{-1}b = A \setminus b$$

which is multifrontal method for solving the sparse system of equations. The computation time is 0.584555 seconds. However, what to do if the matrix is not sparse? In that case one may use the CG and PCG method (also Jacobi, Gauss-Seidel etc.).

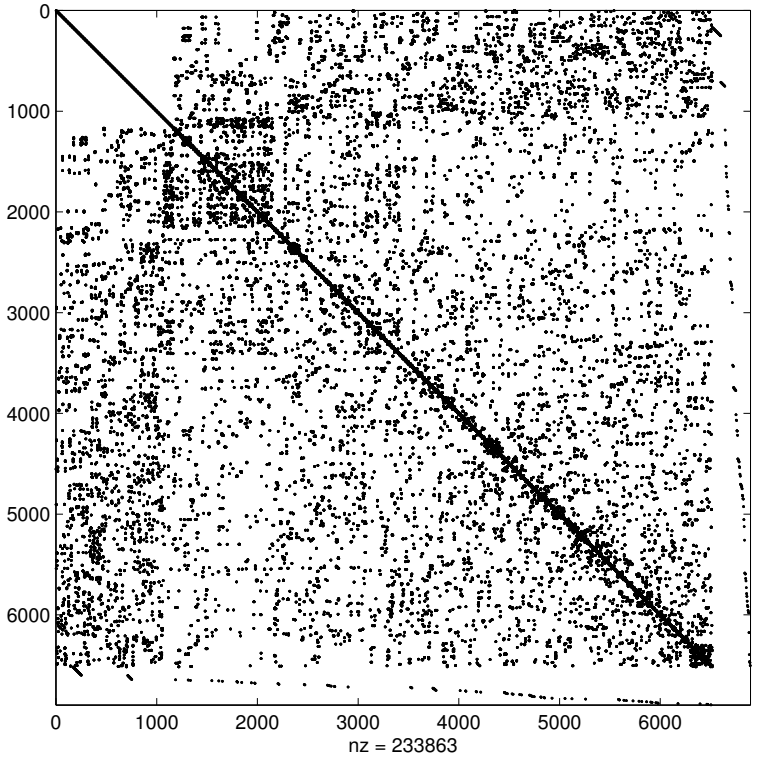


Solving finite element system

To use CG or PCG method the matrix A has to be positive definite and symmetric . To check if matrix is symmetric it is enough to check if $A = A^T$ holds. On the other side, the positive definitness can be checked by cheking the Cholesky decomposition of a matrix.

$$[\ , p] = \text{chol}\ (A)$$

if $p = 0$ then the matrix is positive definite.

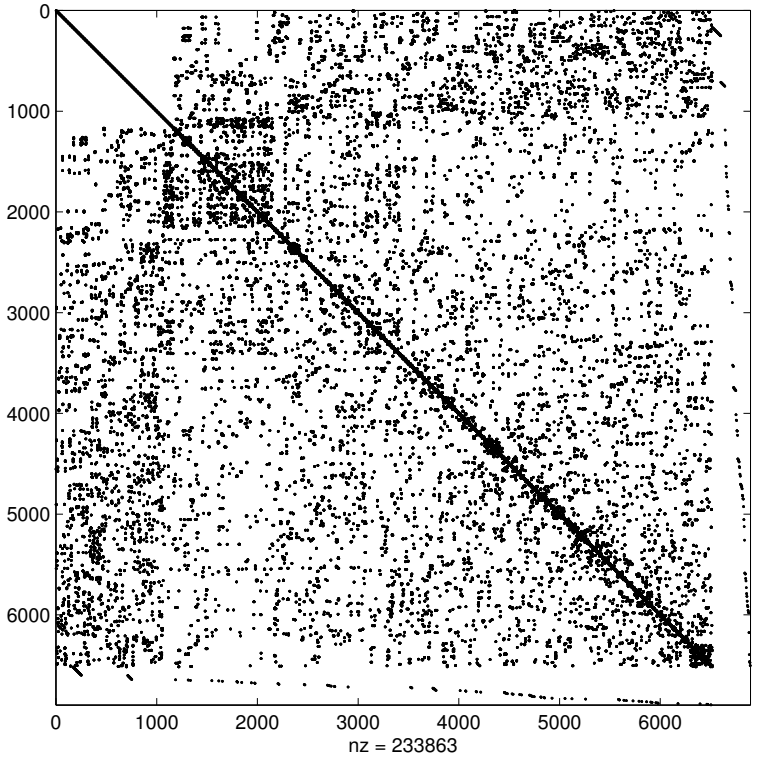


Solving finite element system

In case that the matrix is not positive definite, then one may solve the system

$$A^T A x = A^T b$$

because the matrix $A^T A$ will be positive definite.



References

This lecture is based on the following two scripts

1. Jonathan Richard Shewchuk, `http://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf`
2. Lieven Vandenberghe, `http://www.seas.ucla.edu/~vandenbe/103/lectures/chol.pdf`