Numerical Methods in Physics

Numerische Methoden in der Physik, 515.421.

Instructor: Ass. Prof. Dr. Lilia Boeri
Room: PH 03 090
Tel: +43-316-873 8191
Email Address: l.boeri@tugraz.at

Room: TDK Seminarraum

Time: 8:30-10 a.m.

Exercises: Computer Room, PH EG 004 F

http://itp.tugraz.at/LV/boeri/NUM_METH/index.html
(Lecture slides, Script, Exercises, etc).

TOPICS (this year):

- **Chapter 1: Introduction.**
- **Chapter 2: Numerical methods for the solution of linear inhomogeneous systems.**
- Chapter 3: Interpolation of point sets.
- **Chapter 4: Least-Squares Approximation.**
- **Chapter 5: Numerical solution of transcendental equations.**
- Chapter 6: Numerical Integration.
- Chapter 7: Eigenvalues and Eigenvectors of real matrices.
- **Chapter 8: Numerical Methods for the solution of ordinary differential equations: initial value problems.**
- Chapter 9: Numerical Methods for the solution of ordinary differential equations: marginal value problems.

Last Week (15/10/2013)

- **Linear Systems:** Definition and applications.
- Solution: **Direct** vs iterative methods.
- Gaussian methods: **LU decomposition**.
- A **practical** example: 3x3 square matrix (step-by-step).
- Strategies to reduce the error: **partial pivoting**.
- Stability of the solution: Condition numbers.
- **Programs** for the LU decomposition: **LUDCMP** and **LUBKSB**.
- How to **use** the programs for LU decompositions (inhomogeneous systems, matrix inversion, determinant of a matrix).

This week(22/10/2013)

- **Linear Systems:** Direct (**LU decomposition**) vs indirect methods (**Gauss-Seidel**).
- Direct methods**, iterative improvement** of the solution (reduce roundoff).
- Direct methods, special cases: **tridiagonal matrices**.
- **Indirect methods**: **iterative** solution for **sparse matrices**.
- Indirect methods: **Gauss-Seidel iteration rule**.
- **Band matrices**: definition and properties.
- **Gauss-Seidel** iteration for **band** matrices, storing information and efficiency**.**
- **Gauss-Seidel** method with over and under-**relaxation**.

Linear Systems

Definitions:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$
$$\cdots$$
$$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n$$

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

with:

$$|b_1| + |b_2| + \ldots |b_n| \neq 0$$

Inhomogeneous **linear** system of equations.

$$\det(\hat{A}) \neq 0$$

The problem is non-singular and admits a solution.

Methods of Solution (numerical):

- **Direct Methods:**
- **No methodological** error, BUT computationally expensive; **roundoff** errors can be **large**.

LU decomposition (Doolittle and Crout) (*Gaussian Elimination*).

- **Iterative Methods:**
- **Simple algorithms; roundoff is easily controlled. The solution is approximate (truncation).**

Gauss-Seidel method.

Direct Methods:

$$\hat{A}\mathbf{x} = \mathbf{b} \quad \Rightarrow \quad \hat{U}\mathbf{x} = \mathbf{y}$$

Two-steps procedure:

1) LU decomposition (*Doolittle and Crout*): Reformulation of the Gaussian elimination. A real matrix can always be represented as the product of two real triangular matrices **L** and **U**, *i.e.*

$$\hat{A} = \hat{L} \cdot \hat{U}$$

2) The two auxiliary systems are solved through **back** and **forward** substitution:

$$\hat{U} \cdot \mathbf{x} = \mathbf{y}$$

Back

$$\hat{L} \cdot \mathbf{y} = \mathbf{b}$$

Forward

Iterative Improvement of the Solution:

Direct methods do not introduce any methodological error. But the effect of ***roundoff*** can be severe. There is the possibility of ***improving iteratively*** the solution:

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

$\mathbf{x}$ **Real Solution** $A \cdot \mathbf{x} = \mathbf{b}$ $\mathbf{x}' = \mathbf{x} + \delta\mathbf{x}$ **Approximate Solution**

The approximate solution can be reinserted into the original equation:

$$A \cdot \mathbf{x}' = A \cdot (\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$$

$$(A\mathbf{x} - \mathbf{b}) + A \cdot \delta\mathbf{x} = \delta\mathbf{b}$$

The error on the solution can be estimated from:

$$A \cdot \delta\mathbf{x} = \delta\mathbf{b} \quad \Rightarrow \quad \mathbf{x} = \mathbf{x}' - \delta\mathbf{x}$$

Residual Vector **Corrected Solution**

Iterative method: The corrected solution **x**' can then be re-inserted into the original system, until the method converges, i.e. until there is "no difference" between the result at at n^{th} and $(n+1)^{th}$ iteration.

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

$$\delta\mathbf{b} = (A \cdot \mathbf{x}' - \mathbf{b})$$

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

$$\mathbf{x} = \mathbf{x}' - \delta\mathbf{x}$$

$$\mathbf{x} \rightarrow \mathbf{x}'$$

Practical Implementation

(using LUDCMP and LUBKSB)

1) Save the original (A) matrix in an auxiliary array (LUDCMP overwrites the original matrix).

2) Solve the system A**x**=**b** using LU decomposition(LUDCMP + LUBKSB); the solution is **x**'.

3) Find the residual vector δ**b**=-b+A**x**'.

4) Find the solution of the system Aδ**x**=δ**b**.

5) **x**=**x**'-δ**x**

6) Iterate points 3)-5) with **x** as the new **x**' until the exit condition is met (typically, |δ**x|** or |δ**x|/x**' smaller than a given threshold).

```
I=1(1)N
    J=1(1)N
        ASP(I,J):=A(I,J)
LUDCMP (A,N,INDX,D,KHAD)
LUBKSB (A,N,INDX,B,X)
    I=1(1)N
        SUMDP:=-B(I)
        J=1(1)N
            SUMDP:=SUMDP + DBLE(ASP(I,J))*DBLE(X(J))
        RES(I):=SUMDP
    LUBKSB (A,N,INDX,RES,DELX)
    I=1(1)N
        X(I):=X(I)-DELX(I)
(Condition, under which the iterative method should stop)
(Result and return)
```

Save the original (A) matrix in an auxiliary array

Solve the system Ax=b using LU decomposition.

Find the residual vector δb=-b+Ax'.

Find the solution of Aδx=δb.

x=x'-δx

More on direct methods

(Effect of roundoff error in ill-conditioned systems)

Example (5x5 matrix):

0.10000E+01	0.50000E+00	0.33333E+00	0.25000E+00	0.20000E+00	0.228333E+01
0.50000E+00	0.33333E+00	0.25000E+00	0.20000E+00	0.16667E+00	0.145000E+01
0.33333E+00	0.25000E+00	0.20000E+00	0.16667E+00	0.14286E+00	0.109286E+01
0.25000E+00	0.20000E+00	0.16667E+00	0.14286E+00	0.12500E+00	0.884530E+00
0.20000E+00	0.16667E+00	0.14286E+00	0.12500E+00	0.11111E+00	0.745640E+00

Eigenvalues

$$a_{ij} = \frac{1}{i+j-1}$$

$$x_1 = x_2 = x_3 = x_4 = x_5 = 1.0$$

Exact solution

Hadamard's condition number (K_H):

$$K_H(A) = \frac{|\det(A)|}{\alpha_1 \alpha_2 \ldots \alpha_n}, \qquad \alpha_i = \sqrt{a_{i1}^2 + a_{i2}^2 + \ldots + a_{in}^2}$$

For this problem: $K_H(A)=0.55 \times 10^{-10}$.

$$\begin{cases} K_H(A) < 0.01 & \text{Ill-conditioned} \\ K_H(A) > 0.1 & \text{Well-conditioned} \end{cases}$$

Numerical Solution:

x (exact)	$\bar{x}$ (numerical)	$A \cdot \bar{x}$
1.	0.9999459E+00	0.2283330E+01
1.	0.1000955E+01	0.1450000E+01
1.	0.9960009E+00	0.1092860E+01
1.	0.1005933E+01	0.8845300E+00
1.	0.9971328E+00	0.7456400E+00

0.228333E+01

0.145000E+01

0.109286E+01

0.884530E+00

0.745640E+00

The **eigenvector** is off, although the **eigenvalues** are correct!!!

Direct Methods

Special cases

Tridiagonal matrices: In linear algebra, a tridiagonal matrix is a matrix that has nonzero elements only on the main diagonal, the first diagonal below this, and the first diagonal above the main diagonal.

$$T = \begin{pmatrix} b_1 & c_1 & 0 & 0 & \dots & 0 \\ a_2 & b_2 & c_2 & 0 & \dots & 0 \\ 0 & a_3 & b_3 & c_3 & \dots & 0 \\ . & & & & & . \\ . & & & & & c_{n-1} \\ 0 & 0 & \dots & & \dots & b_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ x_n \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ . \\ . \\ . \\ r_n \end{pmatrix}$$

The tridiagonal matrix of coefficients can be rewritten in terms of three vectors:

b (main diagonal), **a** and **c** (lower and upper secondary diagonal).

The **determinant** of a **tridiagonal** matrix is given by a **continuant** of its elements.

Simple continuant of a series of n numbers $a_1, \ldots, a_n$:

$$K(0) = 1$$
$$K(1) = a_1$$
$$\ldots$$
$$K(n) = a_n K(n-1) + K(n-2)$$

Extended continuant of three sequences a_n, b_n, c_n:

$$K(0) = 1$$
$$K(1) = a_1$$
$$\ldots$$
$$K(n) = a_n K(n-1) - b_{n-1} c_{n-1} K(n-2)$$

Solve through **LU decomposition:**

$$\underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ m_2 & 1 & 0 & 0 & \dots & 0 \\ 0 & m_3 & 1 & & & 0 \\ . & & & & & . \\ . & & & & & 0 \\ 0 & 0 & \dots & & \dots & 1 \end{pmatrix}}_{\mathbf{L}} \underbrace{\begin{pmatrix} u_1 & c_1 & 0 & 0 & \dots & 0 \\ 0 & u_2 & c_2 & 0 & \dots & 0 \\ . & & & & \dots & 0 \\ . & & & & & . \\ . & & & & & c_{n-1} \\ 0 & 0 & 0 & & \dots & u_n \end{pmatrix}}_{\mathbf{U}} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ x_n \end{pmatrix}}_{\equiv \mathbf{y}} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ . \\ . \\ . \\ r_n \end{pmatrix}$$

$$u_1 = b_1$$

$$y_1 = r_1$$

$$m_j = a_j / u_{j-1}$$

$$u_j = b_j - m_j \cdot c_{j-1} \qquad j = 2, \dots, n$$

$$y_j = r_j - m_j \cdot y_{j-1}$$

Structure chart 6 — TRID(A,B,C,R,N,X)

```
Y(1):=R(1)
U(1):=B(1)
U(1) = 0.0
Y                                N
(Exit with error message!)   ......
J=2(1)N
    M:=A(J)/U(J-1)
    U(J):=B(J)-M*C(J-1)
    U(J) = 0.0
    Y                            N
    (Exit with error message!)   ......
    Y(J):=R(J)-M*Y(J-1)
X(N):=Y(N)/U(N)
J=N-1(-1)1
    X(J):=(Y(J)-C(J)*X(J+1))/U(J)
(return)
```

$$u_1 = b_1$$
$$y_1 = r_1$$

$$m_j = a_j / u_{j-1}$$
$$u_j = b_j - m_j \cdot c_{j-1}$$
$$j = 2,\ldots,n$$
$$y_j = r_j - m_j \cdot y_{j-1}$$

$$x_n = y_n / u_n$$
$$x_j = (y_j - c_j \cdot x_{j+1}) / u_j$$
$$j = n-1,\ldots,1$$

Indirect Methods

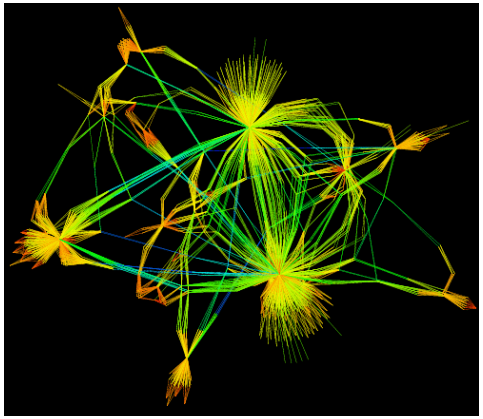
Gauss-Seidel Method

Gauss-Seidel Method:

iterative method for linear inhomogeneous sets of equations.

Advantages: simplicity; the matrix of coefficients is not changed during the iteration. Very **efficient** for systems with a **sparse** matrix of coefficients.

Definition: a **sparse** matrix is a matrix populated primarily with zeros. By contrast, if a larger number of elements differ from zero, then it is common to refer to the matrix as a **dense** matrix. The fraction of zero elements (non-zero elements) in a matrix is called the **sparsity** (density).



Sparse matrices represent **weakly connected** systems;

Many **applications** in different fields (network theory, graph theory, data analysis).

Given a **linear** set of equations:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$

$$\cdots$$

$$a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = b_n$$

If **all** the elements on the **main diagonal** are **non-zero**, we can write:

$$x_1 = -\frac{1}{a_{11}}\left(a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n - b_1\right)$$

$$x_2 = -\frac{1}{a_{22}}\left(a_{21}x_1 + a_{23}x_3 + \cdots + a_{2n}x_n - b_2\right)$$

$$\boxed{x_i = -\frac{1}{a_{ii}}\left(\sum_{j=1(j\neq i)}^{n} a_{ij}x_j - b_i\right)}$$

General Formula

$$\boxed{x_i = -\frac{1}{a_{ii}}\left(\sum_{j=1(j\neq i)}^{n} a_{ij}x_j - b_i\right)}$$

$$\boxed{\mathbf{x} = \hat{C}\cdot\mathbf{x} + \mathbf{f}} \qquad \hat{C} = \{c_{ij}\} \qquad c_{ij} = \begin{cases} -a_{ij}/a_{ii} & i \neq j \\ = 0 & i = j \end{cases} \qquad f_i = \frac{b_i}{a_{ii}}$$

The **Gauss-Seidel Iteration rule** derives from the following expression:

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

Gauss-Seidel Iteration rule:

$$x_i^{(t+1)} = x_i^{(t)} - \Delta x_i^{(t)} \qquad (i = 1, \ldots, n)$$

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

Starting from an initial vector $\mathbf{x}_0$ (***starting vector***) one obtains a sequence of vectors $\mathbf{x}^{(t)}$,

which **converges** to the **exact** solution:

$$\boxed{\lim_{t \to \infty} \mathbf{x}^{(t)} \to \mathbf{x}}$$

Exit conditions:

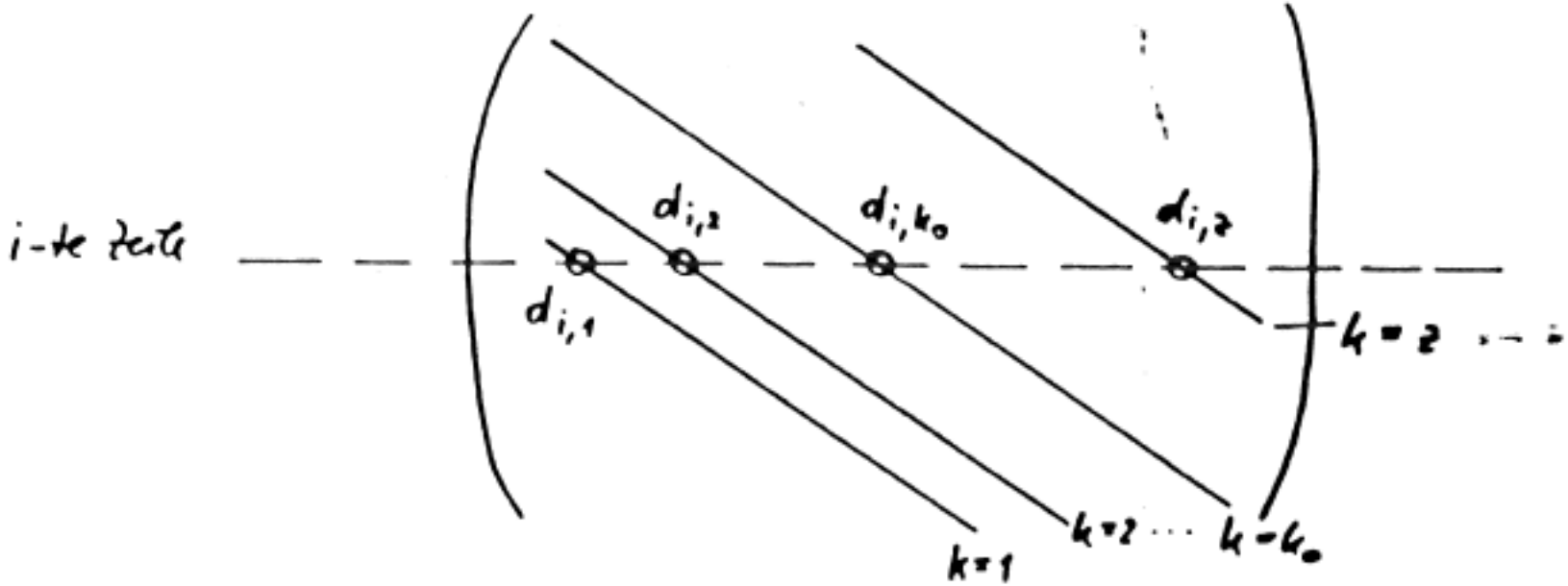
1) The desired **precision** is reached ($x^t - x^{t-1} < \varepsilon$)

2) The **maximum number** of iterations is reached.

Band Matrices: a band matrix is a **sparse matrix** whose non-zero entries are confined to a diagonal band, comprising the main diagonal and zero or more diagonals on either side.

$$\begin{pmatrix} a_{11} & a_{12} & 0 & a_{14} & 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 & a_{25} & 0 & 0 & 0 \\ a_{31} & a_{32} & a_{33} & a_{34} & 0 & a_{36} & 0 & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} & \cdots & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \\ . & & & & \cdots & \cdots & & \\ . & & & & & \cdots & & \\ 0 & & & & & a_{nn-2} & a_{nn-1} & a_{nn} \end{pmatrix}$$

Storage: The most economical way to store a band matrix is to treat it as a 2xz array (z is the number of diagonals with non-zero elements).



d_{ik}: *i*=row; *k*=diagonal.

$s(k)$= vector with **relative positions**.

Indexing: Besides the dik array, one defines an indexing vector *s(k)*, to keep track of the **position** of each sub-diagonal with respect to the main diagonal.

$$s(k) = \begin{cases} 0 & \textbf{Main diagonal} \\ +(-)1 & \textbf{First lower (upper) diagonal} \\ +(-)t & \textbf{t}^{\textbf{th}}\ \textbf{lower (upper) diagonal} \end{cases}$$

The mapping between the original a_{ij}'s and the new storage scheme is given by:

$$\boxed{a_{ij} = d_{ik}} \qquad \boxed{j = s(k) + i}$$

$$i = 1, \ldots, n \qquad k = 1, \ldots, z$$

Gauss-Seidel Iteration rule for band matrices:

$$x_i^{(t+1)} = x_i^{(t)} - \Delta x_i^{(t)} \qquad (i = 1, \ldots, n)$$

$$\Delta x_i^{(t)} = x_i^{(t)} + \frac{1}{d_{i,k_0}} \left[\sum_{k=1(k \neq k_0)}^{z} d_{ik} x_{s(k)+i}^{(t)} - b_i \right]$$

The sum contains only **a few terms** for which:

$$0 < s(k) + 1 \leq n$$

Convergence criterion (empirical)**:** All linear systems in which the ***elements on the main diagonal*** *dominate* the other matrix elements have a good chance of converging with the G-S method.

Precision: Since there is **no accumulation** of roundoff errors upon successive iterations, the GS method is **more accurate** than direct methods.

Practical Implementation (GAUSEI):

INPUT parameters:

N: Order of the system.

NDIAG: Number of non-zero diagonals in the band matrix.

S( ): INTEGER array containing the relative positions of the diagonals.

DIAG(,): Array with the matrix elements: the first index specifies the matrix row, the second the diagonal.

F( ): Inhomogeneous vector of the system.

TMAX: Maximum number of iteration steps.

W: Relaxation parameter (see section 2.7.6).

IREL: IREL $\neq 1$: *absolute* error tolerance
IREL $= 1$: *relative* error tolerance

TOL: Absolute or relative error which has to be reached during the iteration.

Practical Implementation (GAUSEI):

OUTPUT parameters:

SOL( ): Solution vector.

T: Number of iteration steps performed by GAUSEI.

ERROR: Logical variable for error diagnostic: After the execution of GAUSEI ERROR is 'false' if the required precision has been reached, and 'true' if

- not all the elements on the main diagonal are different from zero.
- the required precision has not been reached within TMAX iteration steps.

important internal variables:

K0: Index of the main diagonal.

DX: Iteration-correction value according to Eq.(2.24).

ISCH: Control variable for the precision.

How does it work?

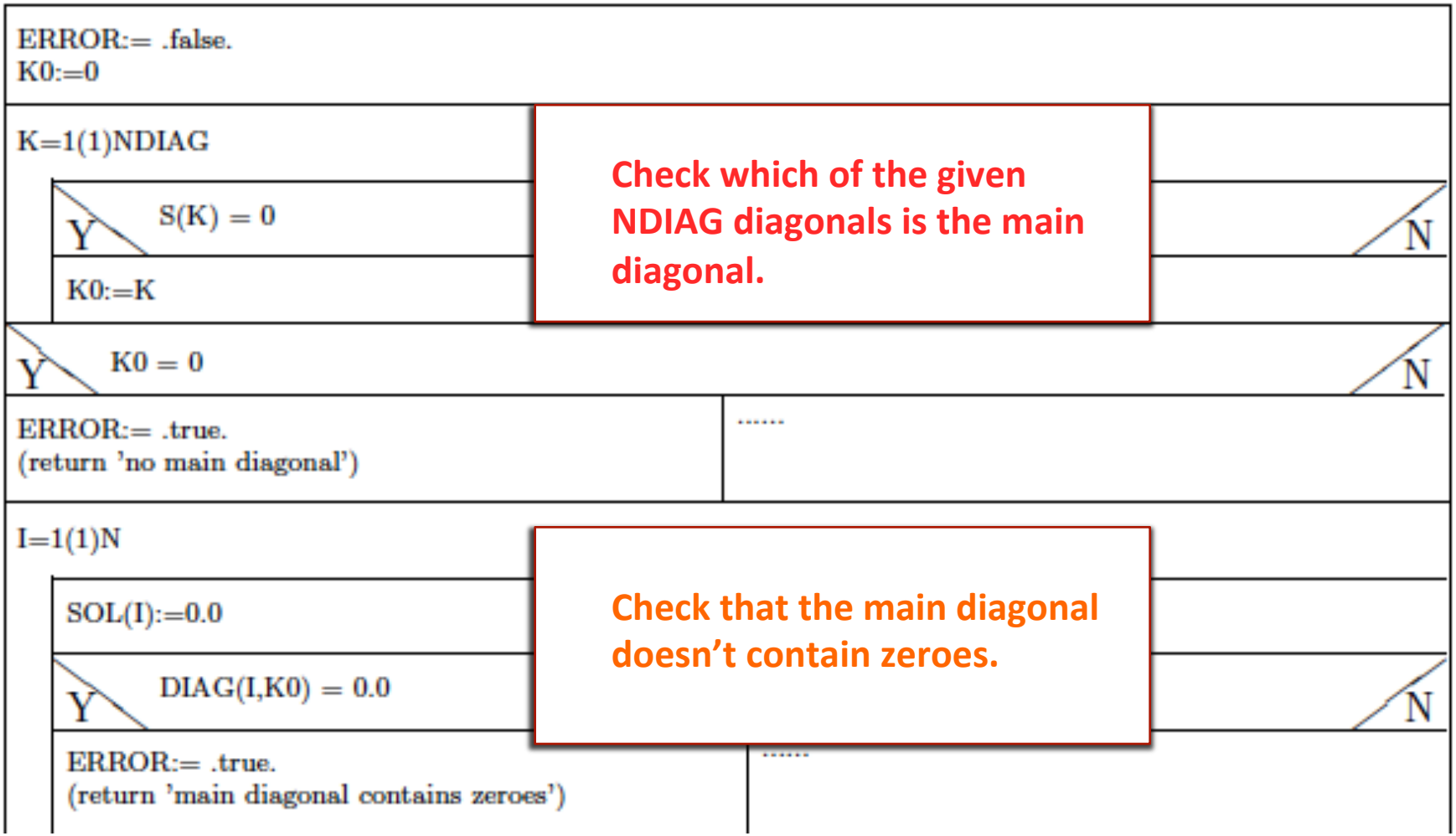
1) Check which of the given NDIAG diagonals is the main diagonal, using the definition of S(K). Save the k0 index.

2) Check if the main diagonal contains zero (if this is the case, exit the program).

3) Perform the G-S iteration $\mathbf{x}^{t+1}=C\mathbf{x}^{t}+\mathbf{f}$, until the exit condition is met.

$$x_i^{(t+1)} = x_i^{(t)} - \Delta x_i^{(t)} \qquad (i = 1,\ldots,n)$$

$$\Delta x_i^{(t)} = x_i^{(t)} + \frac{1}{d_{i,k_0}}\left[\sum_{k=1(k\neq k_0)}^{z} d_{ik} x_{s(k)+i}^{(t)} - b_i\right]$$



Check which of the given NDIAG diagonals is the main diagonal.

Check that the main diagonal doesn't contain zeroes.

```
T:=0
  ISCH:=0
  I=1(1)N
    S1:=0.0
    K=1(1)NDIAG
      J:=S(K)+I
      Y  K ≠ K0 .and. J > 0 .and. J ≤ N
      S1:=S1+DIAG(I,K)*SOL(J)        ......
    DX:=W*((S1-F(I))/DIAG(I,K0)+LOES(I)
    SOL(I):=SOL(I)-DX
    Y  IREL = 1                                   N
    ERR:=| DX/SOL(I) |              ERR:= | DX |
    Y  ERR > TOL                                  N
    ISCH:=1                         ......
  T:=T+1
T > TMAX .or. ISCH=0
Y  ISCH=0                                         N
print: 'No convergence'             ......
ERROR:= .true.
(return)
```

$$x_i^{(t+1)} = x_i^{(t)} - \Delta x_i^{(t)} \qquad (i = 1, \ldots, n)$$

$$\Delta x_i^{(t)} = x_i^{(t)} + \frac{1}{d_{i,k_0}} \left[\sum_{k=1(k \neq k_0)}^{z} d_{ik} x_{s(k)+i}^{(t)} - b_i \right]$$

Exit condition

Relaxation parameter in the GS method:

The usual G-S iteration:

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \Delta\mathbf{x}^{(t)}$$

can be modified introducing a **relaxation parameter** ω:

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \omega \cdot \Delta\mathbf{x}^{(t)}$$

$$\omega \begin{cases} < 1 & \textbf{Under-relaxation.} \\ = 1 & \textbf{Standard Gauss-Seidel.} \\ > 1 & \textbf{Over-relaxation.} \end{cases}$$

Which oftens speeds up convergence.

Example: Use of relaxation:

$$\begin{cases} x + 2y = 3 \\ x - 4y = -3 \end{cases}$$

ω	t
0.65	20
0.70	18
0.75	15
0.8	14
0.85	12
0.9	12
0.95	21
1.0	31
1.05	48

There is an ideal value of ω: **0.85 <ω_{ideal}< 0.9**.

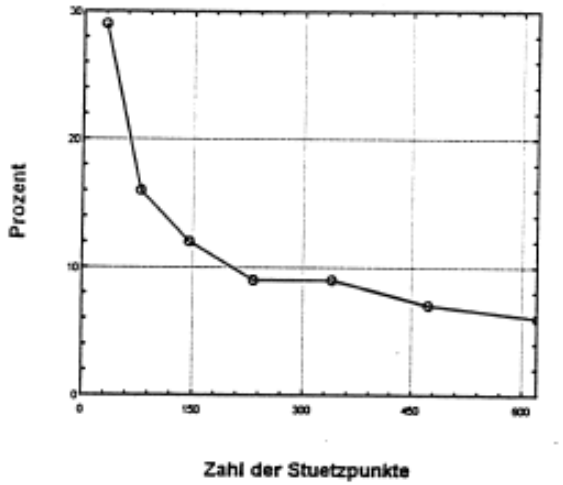
Empirical rules:

- ω must be smaller than 2.
- For many important systems $1<\omega_{ideal}<2$.
- In these cases:

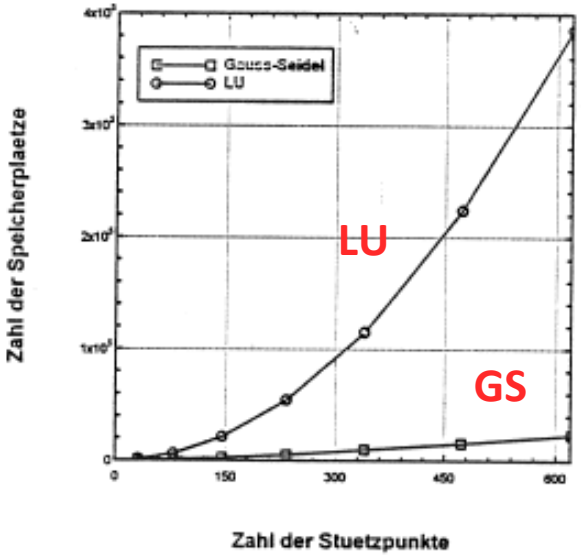
$$\omega_{ideal} = \frac{2}{1+\sqrt{1-\boxed{\lambda_1^2}}}$$

Largest eigenvalue of C.

Efficiency of the Gauss-Seidel Method:



Occupation of the matrix of coefficients for a Laplace equation.



Efficiency of the G-S method compared to LU decomposition (memory storage).

This week(22/10/2013)

- **Linear Systems:** Direct (**LU decomposition**) vs indirect methods (**Gauss-Seidel**).
- Direct methods**, iterative improvement** of the solution (reduce roundoff).
- Direct methods, special cases: **tridiagonal matrices**.
- **Indirect methods**: **iterative** solution for **sparse matrices**.
- Indirect methods: **Gauss-Seidel iteration rule**.
- **Band matrices**: definition and properties.
- **Gauss-Seidel** iteration for **band** matrices, storing information and efficiency**.**
- **Gauss-Seidel** method with over and under-**relaxation**.