~

Iterative Solver convergence

Scientific Computing Winter 2016/2017

Lecture 9

With material from Y. Saad "Iterative Methods for Sparse Linear Systems"

Jürgen Fuhrmann

juergen.fuhrmann@wias-berlin.de

$\sim$

Homework analysis

Machine epsilon

Sample solution: `/net/wir/examples/part3/macheps.cxx`

```
T eps=1.0;
T one=1.0;
T epsnew=1.0;
T result=0.0;
do
{
    eps=epsnew;
    epsnew=eps/2.0;
    result=one+epsnew;
} while (result>one);
```

Common errors:

- In exact math it is true that from $1 + \varepsilon = 1$ it follows that $0 + \varepsilon = 0$ and vice versa. In floating point computations *this is not true*
- Many of you used the right algorithm and used the first value or which $1 + \varepsilon = 1$ as the result. This is half the desired quantity.
- Some did not divide by 2 but by other numbers. Division by 2 is a mantissa shift and essentially exact. 2 itself is also represented exactly in floating point arithmetic.

Machine epsilon values

```
   Calculated: 1.1920928955078125e-07
From <limits>: 1.1920928955078125e-07

   Calculated: 2.2204460492503130808472633361 8e-16
From <limits>: 2.22044604925031308084726333618e-16

   Calculated: 1.08420217248550443400745280087e-19
From <limits>: 1.08420217248550443400745280087e-19
```

Summation

$\sum_{n=1}^{N} \frac{1}{n^2} \approx \frac{\pi^2}{6}$

Intended answer: sum in reverse order. Start with adding up many small values which would be cancelled out if added to an already large sum value.

Sample solution: `/net/wir/examples/part3/basel.cxx`

Here are the results for float

n	forward sum	forward sum error	reverse sum	reverse sum error
10	1.5497677326202392e+00	9.5166444778442382 8e-02	1.54976773262023925e+00	9.5166444778442382 8e-02
100	1.6349840164184570e+00	9.9501609802246093 7e-03	1.63498389720916748e+00	9.95028018951416015e-03
1000	1.6439348459243774e+00	9.9933147430419921 8e-04	1.64393448829650878e+00	9.99689102172851562e-04
10000	1.6447253227233886e+00	2.08854675292968750e-04	1.64483404159545898e+00	1.00135803222656250e-04
100000	1.6447253227233886e+00	2.08854675292968750e-04	1.64492404460906982e+00	1.01327896118164062e-05
1000000	1.6447253227233886e+00	2.08854675292968750e-04	1.64493298530578613e+00	1.19209289550781250e-06
10000000	1.6447253227233886e+00	2.08854675292968750e-04	1.64493393898010253e+00	2.38418579101562500e-07
100000000	1.6447253227233886e+00	2.08854675292968750e-04	1.64493405818939208e+00	1.19209289550781250e-07

Summation: Unexpected highlight answer I

by Minh Huyen Ly Le

In order to improve the accuracy of the approximation of the limit, one can use the *Euler-Maclaurin-Summation Formula*, just as Euler did to approximate the series of the Baseler Problem. With this formula the convergence of the partial sums is accelerated.
The Asymptotic Expansion of sums: For $a, b \in \mathbb{N}$ and $B_k,\ k \in \mathbb{N}$ Bernoulli-numbers we have:

$$\sum_{n=a}^{b} f(n) \sim \int_a^b f(x)dx + \frac{f(a)+f(b)}{2} + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} \left\{ f^{(2k-1)}(b) - f^{(2k-1)}(a) \right\}$$

Therefore, with $f(x) = \frac{1}{x^2}$, $f^{(n)}(x) = (-1)^n (n+1)! x^{-(n+2)}$ we have on the one hand

$$\frac{\pi^2}{6} = \sum_{n=1}^{\infty} \frac{1}{n^2} \sim \int_1^{\infty} \frac{1}{x^2} dx + \frac{1}{2} + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!} \left\{ 0 - (-1)^{2k-1} (2k)! 1^{-(2k+1)} \right\}$$
$$= 1 + \frac{1}{2} + \sum_{k=1}^{\infty} B_{2k} =: C$$

Summation: Unexpected highlight answer II

On the other hand, we have for $K \in \mathbb{N}$

$$\begin{aligned}\sum_{n=1}^{K} \frac{1}{n^2} &\sim \int_1^K \frac{1}{x^2}dx + \frac{1}{2} + \frac{1}{2K^2} - \sum_{k=1}^{\infty} B_{2k} K^{-(2k+1)} + \sum_{k=1}^{\infty} B_{2k} \\ &= 1 - \frac{1}{K} + \frac{1}{2} + \frac{1}{2K^2} - \sum_{k=1}^{\infty} B_{2k} K^{-(2k+1)} + \sum_{k=1}^{\infty} B_{2k} \\ &= C \underbrace{- \frac{1}{K} + \frac{1}{2K^2} - \frac{1}{6K^3} + \frac{1}{30K^5} - \frac{1}{42K^7} + \frac{1}{30K^9} \cdots}_{(RHS)}\end{aligned}$$

For the approximation, let us look at an example for $K = 100$ and truncate the Right-Hand-Side (RHS) from above after the K^9-term. (See Output above)
(LHS) $= \sum_{n=1}^{K} \frac{1}{n^2} = 1.63498390018489$
(RHS) $= -\frac{1}{K} + \frac{1}{2K^2} - \frac{1}{6K^3} + \frac{1}{30K^5} - \frac{1}{42K^7} + \frac{1}{30K^9} = -0.00995016666333357$
C = LHS-RHS = 1.64493406684823 $\sim \frac{\pi^2}{6}$ and we therefore get an accuracy for at least 8 digits!

```
Improvement with EMSF, e.g. K = 100:
K=100: LHS=1.63498390018489
K=100: RHS=-0.00995016666333357
K=100: C = LHS-RHS =1.64493406684823
```

- So, yes, you can beat the computer with good math...

$\sim$

Recap from last time

Sparse direct solvers: solution steps (Saad Ch. 3.6)

1. Pre-ordering
   - The amount of non-zero elements generated by fill-in can be decreases by re-ordering of the matrix
   - Several, graph theory based heuristic algorithms exist
2. Symbolic factorization
   - If pivoting is ignored, the indices of the non-zero elements are calculated and stored
   - Most expensive step wrt. computation time
3. Numerical factorization
   - Calculation of the numerical values of the nonzero entries
   - Not very expensive, once the symbolic factors are available
4. Upper/lower triangular system solution
   - Fairly quick in comparison to the other steps

- Separation of steps 2 and 3 allows to save computational costs for problems where the sparsity structure remains unchanged, e.g. time dependent problems on fixed computational grids
- With pivoting, steps 2 and 3 have to be performed together
- Instead of pivoting, *iterative refinement* may be used in order to maintain accuracy of the solution

Interfacing UMFPACK from C++ (numcxx)

(shortened version of the code)

```
#include <suitesparse/umfpack.h>

// Calculate LU factorization
template<> inline void TSolverUMFPACK<double>::update()
{
    pMatrix->flush(); // Update matrix, adding newly created elements
    int n=pMatrix->shape(0);
    double *control=nullptr;

    //Calculate symbolic factorization only if matrix patter
    //has changed
    if (pMatrix->pattern_changed())
    {
       umfpack_di_symbolic (n, n, pMatrix->pIA->data(), pMatrix->pJA->data(), pMatrix->pA->data(),
       &Symbolic, 0, 0);
    }

    umfpack_di_numeric (pMatrix->pIA->data(), pMatrix->pJA->data(), pMatrix->pA->data(),
    Symbolic, &Numeric, control, 0) ;

   pMatrix->pattern_changed(false);
}

// Solve LU factorized system
template<> inline void TSolverUMFPACK<double>::solve( TArray<T> & Sol,  const TArray<T> & Rhs)
{
    umfpack_di_solve (UMFPACK_At,pMatrix->pIA->data(), pMatrix->pJA->data(), pMatrix->pA->data(),
                      Sol.data(), Rhs.data(),
                      Numeric, control, 0 ) ;
}
```

Example code

- Copy files, creating subdirectory part3
  - the `.` denotes the current directory

```
$ cp -r /net/wir/examples/part3 .
```

- Compile sources (for each of the .cxx files)

```
$ g++ --std=c++11 -I/net/wir/include -o executable source.cxx
   -llapack -lblas -L/net/wir/lib -lumfpack -lamd -lcolamd -lcholmod
```

More compiler flags

(see Makefile)

```
| -o name              | Name of output file                          |
| -g                   | Generate debugging instructions              |
| -O0, -O1, -O2, -O3   | Optimization levels                          |
| -c                   | Avoid linking                                |
| -I<path>             | Add <path> to include search path            |
| -D<symbol>           | Define preprocessor symbol                   |
| -std=c++11           | Use C++11 standard                           |
| -lname               | Link with libname.a or libname.so from system |
| -Lpath               | Search for libraries in path                 |
```

How to use ?

```
#include <numcxx/numcxx.h>
auto pM=numcxx::DSparseMatrix::create(n,n);
auto pF=numcxx::DArray1::create(n);
auto pU=numcxx::DArray1::create(n);

auto &M=*pM;
auto &F=*pF;
auto &U=*pU;

F=1.0;
for (int i=0;i<n;i++)
{
    M(i,i)=3.0;
    if (i>0) M(i,i-1)=-1;
    if (i<n-1) M(i,i+1)=-1;
}

auto pUmfpack=numcxx::DSolverUMFPACK::create(pM);
pUmfpack->solve(U,F);
```

Elements of iterative methods (Saad Ch.4)

Solve $Au = b$ iteratively

- Preconditioner: a matrix $M \approx A$ "approximating" the matrix A but with the property that the system $Mv = f$ is easy to solve
- Iteration scheme: algorithmic sequence using M and A which updates the solution step by step

Simple iteration with preconditioning

Idea: $A\hat{u} = b \Rightarrow$

$$\hat{u} = \hat{u} - M^{-1}(A\hat{u} - b)$$

$\Rightarrow$ iterative scheme

$$u_{k+1} = u_k - M^{-1}(Au_k - b) \quad (k = 0, 1 \dots)$$

1. Choose initial value u_0, tolerance ε, set $k = 0$
2. Calculate *residuum* $r_k = Au_k - b$
3. Test convergence: if $||r_k|| < \varepsilon$ set $u = u_k$, finish
4. Calculate *update*: solve $Mv_k = r_k$
5. Update solution: $u_{k+1} = u_k - v_k$, set $k = i + 1$, repeat with step 2.

The Jacobi method

- Let $A = D - E - F$, where D: main diagonal, E: negative lower triangular part F: negative upper triangular part
- Jacobi: $M = D$, where D is the main diagonal of A.

$$u_{k+1,i} = u_{k,i} - \frac{1}{a_{ii}} \left(\sum_{j=1\dots n} a_{ij} u_{k,j} - b_i \right) \quad (i = 1 \dots n)$$

$$a_{ii} u_{k+1,i} + \sum_{j=1\dots n, j\neq i} a_{ij} u_{k,j} = b_i \quad (i = 1 \dots n)$$

- Alternative formulation:

$$u_{k+1} = D^{-1}(E + F)u_k + D^{-1}b$$

- Essentially, solve for main diagonal element row by row
- Already calculated results not taken into account
- Variable ordering does not matter

Use in numcxx

```
auto pM=numcxx::DSparseMatrix::create(n,n);
auto pF=numcxx::DArray1::create(n);
auto pU=numcxx::DArray1::create(n);
auto pR=numcxx::DArray1::create(n);
auto pV=numcxx::DArray1::create(n);

auto &M=*pM;
auto &F=*pF;
auto &U=*pU;
auto &V=*pV;
auto &R=*pR;

F=1.0;
for (int i=0;i<n;i++)
{
    M(i,i)=3;
    if (i>0) M(i,i-1)=-1;
    if (i<n-1) M(i,i+1)=-1;
}
pM->flush();
auto pJacobi=numcxx::DPreconJacobi::create(pM);
pJacobi->update();
double residual_norm=0.0;
U=0.0;
int niter=1000;
for (int i=0;i<niter;i++)
{
    R=M*U-F;
    residual_norm=normi(R);
    if (residual_norm<1.0e-15) break;
    pJacobi->solve(V,R);
    U-=V;
}
std::cout << "residual:" << residual_norm << std::endl;
```

The Gauss-Seidel method

- Solve for main diagonal element row by row
- Take already calculated results into account

$$a_{ii}u_{k+1,i} + \sum_{j<i} a_{ij}u_{k+1,j} + \sum_{j>i} a_{ij}u_{k,j} = b_i \qquad (i = 1 \dots n)$$

$$(D - E)u_{k+1} - Fu_k = b$$

$$u_{k+1} = (D - E)^{-1}Fu_k + (D - E)^{-1}b$$

- May be it is faster
- Variable order probably matters
- The preconditioner is $M = D - E$
- Backward Gauss-Seidel: $M = D - F$
- Splitting formulation: $A = M - N$, then

$$u_{k+1} = M^{-1}Nu_k + M^{-1}b$$

SOR and SSOR

- SOR: Successive overrelaxation: solve $\omega A = \omega B$ and use splitting

$$
\begin{aligned}
\omega A &= (D - \omega E) - (\omega F + (1 - \omega D)) \\
M &= \frac{1}{\omega}(D - \omega E)
\end{aligned}
$$

  leading to

$$
(D - \omega E)u_{k+1} = (\omega F + (1 - \omega D)u_k + \omega b
$$

- SSOR: Symmetric successive overrelaxation

$$
\begin{aligned}
(D - \omega E)u_{k+\frac{1}{2}} &= (\omega F + (1 - \omega D)u_k + \omega b \\
(D - \omega F)u_{k+1} &= (\omega E + (1 - \omega D)u_{k+\frac{1}{2}} + \omega b
\end{aligned}
$$

$$
M = \frac{1}{\omega(2 - \omega)}(D - \omega E)D^{-1}(D - \omega F)
$$

- Gauss-Seidel and symmetric Gauss-Seidel are special cases for $\omega = 1$.

Block methods

- Jacobi, Gauss-Seidel, (S)SOR methods can as well be used block-wise, based on a partition of the system matrix into larger blocks,
- The blocks on the diagonal should be square matrices, and invertible
- Interesting variant for systems of partial differential equations, where multiple species interact with each other

Convergence

Let $\hat{u}$ be the solution of $Au = b$.

$$\begin{aligned} u_{k+1} &= u_k - M^{-1}(Au_k - b) \\ &= (I - M^{-1}A)u_k + M^{-1}b \\ u_{k+1} - \hat{u} &= u_k - \hat{u} - M^{-1}(Au_k - A\hat{u}) \\ &= (I - M^{-1}A)(u_k - \hat{u}) \\ &= (I - M^{-1}A)^k(u_0 - \hat{u}) \end{aligned}$$

So when does $(I - M^{-1}A)^k$ converge to zero for $k \to \infty$?

Spectral radius and convergence

- λ_i $(i = 1 \dots p)$: eigenvalues of A
- $\sigma(A) = \{\lambda_1 \dots \lambda_p\}$: spectrum of A
- $\rho(A) = \max_{\lambda \in \sigma(A)} |\lambda|$: spectral radius

Theorem (Saad, Th. 1.10) $\lim_{k\to\infty} A^k = 0 \Leftrightarrow \rho(A) < 1$.

Theorem (Saad, Th. 1.12) $\lim_{k\to\infty} ||A^k||^{\frac{1}{k}} = \rho(A)$

$\Rightarrow$ Sufficient condition for convergence: $\rho(I - M^{-1}A) < 1$.

$\Rightarrow$ At the same time, $\rho(A)$ is the worst case estimate for the asymptotic convergence factor:

$$\lim_{k\to\infty} \left(\max_{u_0} \frac{||(I - M^{-1}A)^k (u_0 - \hat{u})||}{||u_0 - \hat{u}||} \right)^{\frac{1}{k}} \le \rho(A)$$

Richardson iteration

$M = \frac{1}{\alpha}$, $I - M^{-1}A = I - \alpha A$. Assume for the eigenvalues of A: $\lambda_{min} \leq \lambda_i \leq \lambda_{max}$.

Then for the eigenvalues μ_i of $I - \alpha A$ one has $1 - \alpha\lambda_{max} \leq \lambda_i \leq 1 - \alpha\lambda_{min}$.

If $\lambda_{min} < 0$ and $\lambda_{max} < 0$, at least one $\mu_i > 1$.

So, assume $\lambda_{min} > 0$. Then we must have

$1 - \alpha\lambda_{max} > -1, 1 - \alpha\lambda_{min} < 1 \Rightarrow$
$0 < \alpha < \frac{2}{\lambda_{max}}$.

$\rho = \max(|1 - \alpha\lambda_{max}|, |1 - \alpha\lambda_{min}|)$

$\alpha_{opt} = \frac{2}{\lambda_{min} + \lambda_{max}}$

$\rho_{opt} = \frac{\lambda_{max} - \lambda_{min}}{\lambda_{max} + \lambda_{min}}$

1.10 Nonnegative Matrices, M-Matrices

Nonnegative matrices play a crucial role in the theory of matrices. They are important in the study of convergence of iterative methods and arise in many applications including economics, queuing theory, and chemical engineering.

A *nonnegative matrix* is simply a matrix whose entries are nonnegative. More generally, a partial order relation can be defined on the set of matrices.

Definition 1.23 *Let A and B be two $n \times m$ matrices. Then*

$$A \le B$$

if by definition, $a_{ij} \le b_{ij}$ for $1 \le i \le n$, $1 \le j \le m$. If O denotes the $n \times m$ zero matrix, then A is nonnegative if $A \ge O$, and positive if $A > O$. Similar definitions hold in which "positive" is replaced by "negative".

The binary relation "$\le$" imposes only a *partial* order on $\mathbb{R}^{n \times m}$ since two arbitrary matrices in $\mathbb{R}^{n \times m}$ are not necessarily comparable by this relation. For the remainder of this section, we now assume that only square matrices are involved. The next proposition lists a number of rather trivial properties regarding the partial order relation just defined.

Properties of $\leq$ for matrices

Proposition 1.24 *The following properties hold.*

1. *The relation $\leq$ for matrices is reflexive ($A \leq A$), antisymmetric (if $A \leq B$ and $B \leq A$, then $A = B$), and transitive (if $A \leq B$ and $B \leq C$, then $A \leq C$).*

2. *If A and B are nonnegative, then so is their product AB and their sum $A + B$.*

3. *If A is nonnegative, then so is A^k.*

4. *If $A \leq B$, then $A^T \leq B^T$.*

5. *If $O \leq A \leq B$, then $\|A\|_1 \leq \|B\|_1$ and similarly $\|A\|_\infty \leq \|B\|_\infty$.*

Irreducible matrices

A is *irreducible* if there is a permutation matrix P such that PAP^T is upper block triangular.

Perron-Frobenius Theorem

Theorem (Saad Th.1.25) Let A be a real $n \times n$ nonnegative irreducible martrix. Then:

- The spectral radius $\rho(A)$ is a simple eigenvalue of A.
- There exists an eigenvector u associated wit $\rho(A)$ which has positive elements

Proof: see e.g. Varga, "Matrix Iterative Analysis"

$\sim$

Consequences of Perron-Frobenius for iterative method convergence

Comparison of products of nonnegative matrices

Proposition 1.26 *Let A, B, C be nonnegative matrices, with $A \leq B$. Then*

$$AC \leq BC \quad \text{and} \quad CA \leq CB.$$

Proof. Consider the first inequality only, since the proof for the second is identical. The result that is claimed translates into

$$\sum_{k=1}^{n} a_{ik}c_{kj} \leq \sum_{k=1}^{n} b_{ik}c_{kj}, \quad 1 \leq i, j \leq n,$$

which is clearly true by the assumptions. □

Corollary 1.27 *Let A and B be two nonnegative matrices, with $A \leq B$. Then*

$$A^k \leq B^k, \quad \forall\, k \geq 0. \tag{1.42}$$

Proof. The proof is by induction. The inequality is clearly true for $k = 0$. Assume that (1.42) is true for k. According to the previous proposition, multiplying (1.42) from the left by A results in

$$A^{k+1} \leq AB^k. \tag{1.43}$$

Now, it is clear that if $B \geq 0$, then also $B^k \geq 0$, by Proposition 1.24. We now multiply both sides of the inequality $A \leq B$ by B^k to the right, and obtain

$$AB^k \leq B^{k+1}. \tag{1.44}$$

The inequalities (1.43) and (1.44) show that $A^{k+1} \leq B^{k+1}$, which completes the induction proof. ☐

Comparison of spectral radii of nonnegative matrices

Theorem 1.28 *Let A and B be two square matrices that satisfy the inequalities*

$$O \le A \le B. \tag{1.45}$$

Then

$$\rho(A) \le \rho(B). \tag{1.46}$$

Proof. The proof is based on the following equality stated in Theorem 1.12

$$\rho(X) = \lim_{k\to\infty} \|X^k\|^{1/k}$$

for any matrix norm. Choosing the $1-$norm, for example, we have from the last property in Proposition 1.24

$$\rho(A) = \lim_{k\to\infty} \|A^k\|_1^{1/k} \le \lim_{k\to\infty} \|B^k\|_1^{1/k} = \rho(B)$$

which completes the proof. □

Nonnegative matrices in iterations

Theorem 1.29 *Let B be a nonnegative matrix. Then $\rho(B) < 1$ if and only if $I - B$ is nonsingular and $(I - B)^{-1}$ is nonnegative.*

Proof. Define $C = I - B$. If it is assumed that $\rho(B) < 1$, then by Theorem 1.11, $C = I - B$ is nonsingular and

$$C^{-1} = (I - B)^{-1} = \sum_{i=0}^{\infty} B^i. \tag{1.47}$$

In addition, since $B \geq 0$, all the powers of B as well as their sum in (1.47) are also nonnegative.

To prove the sufficient condition, assume that C is nonsingular and that its inverse is nonnegative. By the Perron-Frobenius theorem, there is a nonnegative eigenvector u associated with $\rho(B)$, which is an eigenvalue, i.e.,

$$Bu = \rho(B)u$$

or, equivalently,

$$C^{-1}u = \frac{1}{1 - \rho(B)}u.$$

Since u and C^{-1} are nonnegative, and $I - B$ is nonsingular, this shows that $1 - \rho(B) > 0$, which is the desired result. □

M-Matrices

Definition 1.30 *A matrix is said to be an M-matrix if it satisfies the following four properties:*

1. $a_{i,i} > 0$ *for* $i = 1, \ldots, n$.
2. $a_{i,j} \leq 0$ *for* $i \neq j,\ i, j = 1, \ldots, n$.
3. A *is nonsingular.*
4. $A^{-1} \geq 0$.

- This matrix property plays an important role for discrtized PDEs:
  - convergence of iterative methods
  - nonnegativity of discrete solutions (e.g concentrations)
  - prevention of unphysical oscillations

Equivalent definition

Theorem 1.31 *Let a matrix A be given such that*

1. $a_{i,i} > 0$ *for* $i = 1, \ldots, n$.
2. $a_{i,j} \leq 0$ *for* $i \neq j,\ i, j = 1, \ldots, n$.

Then A is an M-matrix if and only if

3. $\rho(B) < 1$, *where* $B = I - D^{-1}A$.

Proof. From the above argument, an immediate application of Theorem 1.29 shows that properties (3) and (4) of the above definition are equivalent to $\rho(B) < 1$, where $B = I - C$ and $C = D^{-1}A$. In addition, C is nonsingular iff A is and C^{-1} is nonnegative iff A is. □

Equivalent definition

Theorem 1.32 *Let a matrix A be given such that*

1. $a_{i,j} \leq 0$ *for* $i \neq j,\ i, j = 1, \ldots, n$.
2. A *is nonsingular.*
3. $A^{-1} \geq 0$.

Then

4. $a_{i,i} > 0$ *for* $i = 1, \ldots, n$, *i.e.,* A *is an* M*-matrix.*
5. $\rho(B) < 1$ *where* $B = I - D^{-1}A$.

Proof. Define $C \equiv A^{-1}$. Writing that $(AC)_{ii} = 1$ yields

$$\sum_{k=1}^{n} a_{ik}c_{ki} = 1$$

which gives

$$a_{ii}c_{ii} = 1 - \sum_{\substack{k=1 \\ k \neq i}}^{n} a_{ik}c_{ki}.$$

Since $a_{ik}c_{ki} \leq 0$ for all k, the right-hand side is ≥ 1 and since $c_{ii} \geq 0$, then $a_{ii} > 0$. The second part of the result now follows immediately from an application of the previous theorem. $\square$

Comparison criterion

Theorem 1.33 *Let A, B be two matrices which satisfy*

1. $A \leq B$.
2. $b_{ij} \leq 0$ *for all* $i \neq j$.

Then if A is an M-matrix, so is the matrix B.

Proof. Assume that A is an M-matrix and let D_X denote the diagonal of a matrix X. The matrix D_B is positive because

$$D_B \geq D_A > 0.$$

Consider now the matrix $I - D_B^{-1}B$. Since $A \leq B$, then

$$D_A - A \geq D_B - B \geq O$$

which, upon multiplying through by D_A^{-1}, yields

$$I - D_A^{-1}A \geq D_A^{-1}(D_B - B) \geq D_B^{-1}(D_B - B) = I - D_B^{-1}B \geq O.$$

Since the matrices $I - D_B^{-1}B$ and $I - D_A^{-1}A$ are nonnegative, Theorems 1.28 and 1.31 imply that

$$\rho(I - D_B^{-1}B) \leq \rho(I - D_A^{-1}A) < 1.$$

This establishes the result by using Theorem 1.31 once again. ◻

Regular splittings

- $A = M - N$ is a regular splitting if
  - M is nonsingular
  - M^{-1}, N are nonnegative, i.e. have nonnegative entries
- Regard the iteration $u_{k+1} = M^{-1}Nu_k + M^{-1}b$.
- We have \$I-M^{-1}A= $M^{-1}N$.

When does it converge ?

Convergence of iterations based on regular splittings

Theorem 4.4 *Let M, N be a regular splitting of a matrix A. Then $\rho(M^{-1}N) < 1$ if and only if A is nonsingular and A^{-1} is nonnegative.*

Proof. Define $G = M^{-1}N$. From the fact that $\rho(G) < 1$, and the relation

$$A = M(I - G) \tag{4.35}$$

it follows that A is nonsingular. The assumptions of Theorem 1.29 are satisfied for the matrix G since $G = M^{-1}N$ is nonnegative and $\rho(G) < 1$. Therefore, $(I - G)^{-1}$ is nonnegative as is $A^{-1} = (I - G)^{-1} M^{-1}$.

To prove the sufficient condition, assume that A is nonsingular and that its inverse is nonnegative. Since A and M are nonsingular, the relation (4.35) shows again that $I - G$ is nonsingular and in addition,

$$\begin{aligned} A^{-1}N &= \left(M(I - M^{-1}N)\right)^{-1} N \\ &= (I - M^{-1}N)^{-1}M^{-1}N \\ &= (I - G)^{-1}G. \end{aligned} \tag{4.36}$$

Clearly, $G = M^{-1}N$ is nonnegative by the assumptions, and as a result of the Perron-Frobenius theorem, there is a nonnegative eigenvector x associated with $\rho(G)$ which is an eigenvalue, such that

$$Gx = \rho(G)x.$$

Convergence of iterations based on regular splittings II

From this and by virtue of (4.36), it follows that

$$A^{-1}Nx = \frac{\rho(G)}{1-\rho(G)}x.$$

Since x and $A^{-1}N$ are nonnegative, this shows that

$$\frac{\rho(G)}{1-\rho(G)} \geq 0$$

and this can be true only when $0 \leq \rho(G) \leq 1$. Since $I - G$ is nonsingular, then $\rho(G) \neq 1$, which implies that $\rho(G) < 1$. □

This theorem establishes that the iteration (4.34) always converges, if M, N is a regular splitting and A is an M-matrix.

Regular splittings: example

- Jacobi
- Gauss-Seidel

Further methods for establishing convergence

- Theory for diagonally dominant matrices
- Theory for symmetric, positive definite matrices

Installation on MacOSX

1. Install Xcode from the App-Store
2. Trigger installaion of Command line developer tools in the terminal via

```
$ gcc
```

A dialogue window should pop up, click on install Dann im erscheinenden Dialogfenster “Install” klicken.

3. Check with

```
$ xcode-select -p
/Library/Developer/CommandLineTools
```

4. Install Homebrew + Cakebrew GUI
   `http://brew.sh/index.html`
   `https://www.cakebrew.com/`
5. Install via homebrew
   `make`, `cmake suite-sparse` from science tree

To link with lapack/blas: use `-framework Accelerate` instead of -lblas -llapack