Numerical Linear Algebra

Basic iterative methods

Gerard Sleijpen and Martin van Gijzen

October 5, 2016

Program Lecture 4

- Basic methods for eigenproblems.
  - Power method
  - Shift-and-invert Power method
  - QR algorithm
- Basic iterative methods for linear systems
  - Richardson's method
  - Jacobi, Gauss-Seidel and SOR
  - Iterative refinement
- Steepest decent and the Minimal residual method

Basic methods for eigenproblems

The eigenvalue problem

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$$

can not be solved in a direct way for problems of order > 4, since the eigenvalues are the roots of the characteristic equation

$$\det(\mathbf{A} - \lambda\mathbf{I}) = 0.$$

Today we will discuss two *iterative* methods for solving the eigenproblem.

The Power method

The Power method is the classical method to compute in modulus largest eigenvalue and associated eigenvector of a matrix.

Multiplying with a matrix amplifies strongest the eigendirection corresponding to the in modulus largest eigenvalues.

Successively multiplying and scaling (to avoid overflow or underflow) yields a vector in which the direction of the largest eigenvector becomes more and more dominant.

Algorithm

The Power method for an $n \times n$ matrix $\mathbf{A}$.

$\mathbf{u}_0 \in \mathbb{C}^n$ is given
for $k = 1, 2, ...$
$\quad \widetilde{\mathbf{u}}_k = \mathbf{A}\mathbf{u}_{k-1}$
$\quad \mathbf{u}_k = \widetilde{\mathbf{u}}_k / \|\widetilde{\mathbf{u}}_k\|_2$
$\quad \lambda^{(k)} = \mathbf{u}_{k-1}^* \widetilde{\mathbf{u}}_k$
end for

If $\mathbf{u}_k$ is an eigenvector corresponding to λ_j, then

$$\lambda^{(k+1)} = \mathbf{u}_k^* \mathbf{A}\mathbf{u}_k = \lambda_j \, \mathbf{u}_k^* \mathbf{u}_k = \lambda_j \, \|\mathbf{u}_k\|_2^2 = \lambda_j.$$

Convergence (1)

Let the n eigenvalues λ_i with eigenvectors $\mathbf{v}_i$, $\mathbf{A}\mathbf{v}_i = \lambda_i \mathbf{v}_i$, be ordered such that $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$.

- Assume the eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ form a basis.
- Assume $|\lambda_1| > |\lambda_2|$.

Each arbitrary starting vector $\mathbf{u}_0$ can be written as:

$$\mathbf{u}_0 = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + ... + \alpha_n \mathbf{v}_n$$

and if $\alpha_1 \neq 0$, then it follows that

$$\mathbf{A}^k \, \mathbf{u}_0 = \alpha_1 \lambda_1^k \left(\mathbf{v}_1 + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \left(\frac{\lambda_j}{\lambda_1} \right)^k \mathbf{v}_j \right) .$$

Convergence (2)

Using this equality we conclude that

$$|\lambda_1 - \lambda^{(k)}| = \mathcal{O}\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right) \qquad (k \to \infty)$$

and also that $\mathbf{u}_k$ **directionally converges** to $\mathbf{v}_1$:

the angle between $\mathbf{u}_k$ and $\mathbf{v}_1$ is of order $\left| \frac{\lambda_2}{\lambda_1} \right|^k$.

If $|\lambda_1| > |\lambda_j|$ for all $j > 1$, then we call

λ_1 the **dominant eigenvalue** and

$\mathbf{v}_1$ the **dominant eigenvector**.

Convergence (3)

• If the component $\alpha_1 \mathbf{v}_1$ in $\mathbf{u}_0$ is small as compared to, say $\alpha_2 \mathbf{v}_2$, i.e. $|\alpha_1| \ll |\alpha_2|$, then *initially* convergence may seem to be dominated by λ_2 (until $|\alpha_2 \lambda_2^k| < |\alpha_1 \lambda_1^k|$).

• If the basis of eigenvectors is ill-conditioned, then some eigenvector components in $\mathbf{u}_0$ may be large even if $\mathbf{u}_0$ is modest and *initially* convergence may seem to be dominated by non-dominant eigenvalues.

• $\mathbf{u}_0 = \mathbf{V}\mathbf{a}$, where $\mathbf{V} \equiv [\mathbf{v}_1, \ldots, \mathbf{v}_n]$, $\mathbf{a} \equiv (\alpha_1, \ldots, \alpha_n)^T$. Hence, $\|\mathbf{u}_0\| \leq \|\mathbf{V}\| \, \|\mathbf{a}\|$ and $\|\mathbf{a}\| = \|\mathbf{V}^{-1}\mathbf{u}_0\| \leq \|\mathbf{V}^{-1}\| \, \|\mathbf{u}_0\|$: the constant in the '$\mathcal{O}$-term' may depend on the conditioning of the basis of eigenvectors.

Convergence (4)

Note that there is a problem if $|\lambda_1| = |\lambda_2|$, which is the case for instance if $\lambda_1 = \overline{\lambda}_2$.
A vector $\mathbf{u}_0$ can be written as

$$\mathbf{u}_0 = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \sum_{j=3}^{n} \alpha_j \mathbf{v}_j \ .$$

The component in the direction of $\mathbf{v}_3, \ldots, \mathbf{v}_n$ will vanish in the Power method if $|\lambda_2| > |\lambda_3|$, but $\mathbf{u}_k$ will not tend to a limit if $\mathbf{u}_0$ has nonzero components in $\mathbf{v}_1$ and $\mathbf{v}_2$ and $|\lambda_1| = |\lambda_2|$, $\lambda_1 \neq \lambda_2$.

Shifting

Clearly, the (asymptotic) convergence depends on $|\frac{\lambda_2}{\lambda_1}|$.
To speed-up convergence the Power method can also be applied to the **shifted problem**

$$(\mathbf{A} - \sigma \mathbf{I})\mathbf{v} = (\lambda - \sigma)\mathbf{v}$$

The asymptotic rate of convergence now becomes

$$\left| \frac{\lambda_2 - \sigma}{\lambda_1 - \sigma} \right|$$

Moreover, by choosing a suitable *shift* σ (*how?*) convergence can be forced towards the smallest eigenvalue of $\mathbf{A}$.

Shift-and-invert

Another way to speed-up convergence is to apply the Power method to the **shifted and inverted problem**

$$(\mathbf{A} - \sigma \mathbf{I})^{-1}\mathbf{v} = \mu \mathbf{v}, \qquad \lambda = \frac{1}{\mu} + \sigma.$$

This technique allows us to compute eigenvalues near the shift. However, for this the solution of a system is required in every iteration!

Assignment. Show that the shifted and inverted problem and the original problem share the same eigenvectors.

QR-factorisation, power method

Consider the QR-decomposition $\mathbf{A} = \mathbf{QR}$
with $\mathbf{Q} = [\mathbf{q}_1, \ldots, \mathbf{q}_n]$ unitary and $\mathbf{R} = (r_{ij})$ upper triangular.

Observations.

1) $\mathbf{A}\mathbf{e}_1 = r_{11}\,\mathbf{q}_1$.
2) Since $\mathbf{A}^*\mathbf{Q} = \mathbf{R}^*$, we also have $\mathbf{A}^*\mathbf{q}_n = \overline{r_{nn}}\,\mathbf{e}_n$.

The QR-decomposition incorporates one step of the power method with $\mathbf{A}$ in the first column of $\mathbf{Q}$ and with $(\mathbf{A}^*)^{-1}$ in the last column of $\mathbf{Q}$ (without inverting $\mathbf{A}$!).
To continu, 'rotate' the basis: instead of $\mathbf{e}_1, \ldots, \mathbf{e}_n$,
take $\mathbf{q}_1, \ldots, \mathbf{q}_n$ as new basis in domain and image space of $\mathbf{A}$.
$\mathbf{A}_1 \equiv \mathbf{Q}^*\mathbf{A}\mathbf{Q} = \mathbf{R}\mathbf{Q}$ is the matrix of $\mathbf{A}$ w.r.t. this rotated basis.
In the new basis $\mathbf{q}_1$ and $\mathbf{q}_n$ are represented by $\mathbf{e}_1$ and $\mathbf{e}_n$, resp..

The QR method (1)

This leads to the **QR method**, a popular technique in particular to solve small or dense eigenvalue problems.
The method repeatedly uses QR-factorisation.

The method starts with the matrix $\mathbf{A}_0 \equiv \mathbf{A}$,
factors it into $\mathbf{A}_0 = \mathbf{Q}_0 \mathbf{R}_0$,
and then reverses the factors: $\mathbf{A}_1 = \mathbf{R}_0 \mathbf{Q}_0$.

Assignment. Show that $\mathbf{A}_0$ and $\mathbf{A}_1$ are similar (share the same eigenvalues).

The QR method (2)

And repeats these steps:

$$\text{factor} \quad \mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k, \quad \text{multiply} \quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k.$$

Hence, $\mathbf{A}_k \, \mathbf{Q}_k = \mathbf{Q}_k \, \mathbf{A}_{k+1}$ and

$$\mathbf{A}_0 \, (\mathbf{Q}_0 \mathbf{Q}_1 \cdot \ldots \cdot \mathbf{Q}_{k-1}) = (\mathbf{Q}_0 \mathbf{Q}_1 \cdot \ldots \cdot \mathbf{Q}_{k-1}) \, \mathbf{A}_k$$

$\mathbf{U}_k \equiv \mathbf{Q}_0 \mathbf{Q}_1 \cdot \ldots \cdot \mathbf{Q}_{k-1}$ is unitary,
$\mathbf{A}_0 \mathbf{U}_k = \mathbf{U}_k \mathbf{A}_k$: $\mathbf{A}_0$ and $\mathbf{A}_k$ are similar.

The QR method (2)

And repeats these steps:

$$\text{factor} \quad \mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k, \quad \text{multiply} \quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k.$$

Hence, with $\mathbf{U}_k \equiv \mathbf{Q}_0 \mathbf{Q}_1 \cdots \mathbf{Q}_{k-1}$,

$$\mathbf{A} \, \mathbf{U}_k = \mathbf{U}_k \, \mathbf{A}_k = \mathbf{U}_k \mathbf{Q}_k \, \mathbf{R}_k = \mathbf{U}_{k+1} \, \mathbf{R}_k.$$

In particular, $\mathbf{A}\mathbf{u}_1^{(k)} = \tau \, \mathbf{u}_1^{(k+1)}$ with τ the $(1,1)$-entry of $\mathbf{R}_k$:
the first columns $\mathbf{u}_1^{(k)}$ of the $\mathbf{U}_k$ represent the *power method*.
Here, we used that $\mathbf{R}_k$ is upper triangular.

The QR method (2)

And repeats these steps:

$$\text{factor} \quad \mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k, \quad \text{multiply} \quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k.$$

Hence, $\mathbf{U}_k \equiv \mathbf{Q}_0 \mathbf{Q}_1 \cdot \ldots \cdot \mathbf{Q}_{k-1}$ is unitary and

$$\mathbf{A}^* \, \mathbf{U}_{k+1} = \mathbf{U}_k \, \mathbf{R}_k^*.$$

In particular, $\mathbf{A}^* \, \mathbf{u}_n^{(k+1)} = \tau \, \mathbf{u}_n^{(k)}$, now with τ the (n,n)-entry of $\mathbf{R}_k^*$: the last column $\mathbf{u}_n^{(k)}$ of $\mathbf{U}_k$ incorporates the *inverse* power method. Here, we used that $\mathbf{R}_k^*$ is lower triangular.

The QR method (2)

And repeats these steps:

$$\text{factor} \quad \mathbf{A}_k = \mathbf{Q}_k \mathbf{R}_k, \quad \text{multiply} \quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k.$$

Hence, $\mathbf{U}_k \equiv \mathbf{Q}_0 \mathbf{Q}_1 \cdot \ldots \cdot \mathbf{Q}_{k-1}$ is unitary,
$\mathbf{U}_k$ converges to an unitary matrix $\mathbf{U}$,
$\mathbf{R}_k$ converges to an upper triangular matrix $\mathbf{S}$, and

$$\mathbf{A}\mathbf{U}_k = \mathbf{U}_{k+1}\mathbf{R}_k \quad \rightarrow \quad \mathbf{A}\mathbf{U} = \mathbf{U}\mathbf{S},$$

which is a so-called **Schur decomposition** of $\mathbf{A}$.

The eigenvalues of $\mathbf{A}$ are on the main diagonal of $\mathbf{S}$
(They appear on the main diagonal of $\mathbf{A}_k$ and of $\mathbf{R}_k$).

The QR method (3)

Normally the algorithm is used with shifts

- $$\mathbf{A}_k - \sigma_k \mathbf{I} = \mathbf{Q}_k \mathbf{R}_k, \qquad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k + \sigma_k \mathbf{I}$$

Check that $\mathbf{A}_{k+1}$ is similar to $\mathbf{A}_k$.

- The process incorporates shift-and-invert iteration
(in the last column of $\mathbf{U}_k \equiv \mathbf{Q}_0 \cdot \ldots \cdot \mathbf{Q}_{k-1}$).

The shifted algorithm (with proper shift) converges quadratically.

An eigenvalue of the 2×2 right lower block of $\mathbf{A}_k$ is such a proper shift (**Wilkinson's shift**).

The QR method (4)

Other ingredients of an effective algorithm of the QR method:

- **Deflation** is used, that is,
converged columns and rows (the last ones) are removed.

Theorem. $\mathbf{A}_{k+1}$ is Hessenberg if $\mathbf{A}_k$ is Hessenberg:

- Select $\mathbf{A}_0 = \mathbf{U}_0^* \mathbf{A} \mathbf{U}_0$ to be upper Hessenberg.

Costs to compute all eigenvalues (do not compute $\mathbf{U}$)
of the $n \times n$ matrix $\mathbf{A}$ to full accuracy: $\approx 12n^3$ flop.
Stability is optimal
(order of $n\mathbf{u}\times$ stability of the eigenvalue problem of $\mathbf{A}$).

The QR method for eigenvalues

Ingredients (summary):

1) Bring $\mathbf{A}$ to upper Hessenberg form
2) Select an appropriate shift strategy
3) Repeat: shift, factor, reverse factors & multiply, de-shift
4) Deflate upon convergence

Find all eigenvalues λ_j on the diagonal of $\mathbf{S}$.

Costs $\approx 12n^3$ flop.
Discard one of the ingredients $\rightsquigarrow$ costs $\mathcal{O}(n^4)$ or higher.

$n = 10^3$*: Matlab needs a few seconds. What about* $n = 10^4$*?*

The QR method: concluding remarks

- The QR method is the method of choice for dense systems of size n with n up to a few thousend.
- Usually, for large values of n, one is only interested in a few eigenpairs or a part of the spectrum. The QR-method computes all eigenvalues. The order in which the method detects the eigenvalues can not be pre-described. Therefore, all eigenvalues are computed and the wanted ones are selected.
- For larger values of n, methods are used (to be discussed in a following lectures) that project the eigenvalue problem onto low dimensional spaces, where the QR method is used.
- The method of choice for computing zeros of polynomials is also the QR method (applied to the companion system).

Iterative methods for linear systems

Iterative methods construct successive approximations $\mathbf{x}_k$ to the solution of the linear systems $\mathbf{A}\mathbf{x} = \mathbf{b}$. Here k is the iteration number, and the approximation $\mathbf{x}_k$ is also called the **iterate**.
The vector $\mathbf{e}_k \equiv \mathbf{x}_k - \mathbf{x}$ is the **error**,
$\mathbf{r}_k \equiv \mathbf{b} - \mathbf{A}\mathbf{x}_k \ (= -\mathbf{A}\mathbf{e}_k)$ is the **residual**.

The iterative methods are composed of only a few different basic operations:

- Products with the matrix $\mathbf{A}$
- Vector operations (updates and inner product operations)
- *Preconditioning operations*

Preconditioning

Usually iterative methods are applied not to the original system

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

but to the **preconditioned system**

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

where the **preconditioner** $\mathbf{M}$ is chosen such that:

- Preconditioning operations (operations with $\mathbf{M}^{-1}$, i.e., solves $\mathbf{M}\mathbf{w} = \mathbf{r}$ for $\mathbf{w}$) are cheap;
- The iterative method converges much faster for the preconditioned system with appropriate preconditioner.

Basic iterative methods

The first iterative methods we will discuss are the **basic iterative methods**. Basic iterative methods only use information of the previous iteration.

Until the 70's they were quite popular. Some are still used but as preconditioners in combination with an acceleration technique. They also still play a role in multigrid techniques where they are used as smoothers.

Basic iterative methods (2)

Basic iterative methods are usually constructed using a **splitting** of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{M} - \mathbf{R}.$$

Successive approximations are then computed using the iterative process

$$\mathbf{M}\mathbf{x}_{k+1} = \mathbf{R}\mathbf{x}_k + \mathbf{b}$$

which is equivalent too

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{M}^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_k) = \mathbf{x}_k + \mathbf{M}^{-1}\mathbf{r}_k$$

The next few slides we look at $\mathbf{M} = \mathbf{I}$.

Richardson's method

The choice $\mathbf{M} = \mathbf{I}$, $\mathbf{R} = \mathbf{I} - \mathbf{A}$ gives **Richardson's method**, which is the most simple iterative method possible.

The iterative process becomes

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (\mathbf{b} - \mathbf{A}\mathbf{x}_k) = \mathbf{b} + (\mathbf{I} - \mathbf{A})\mathbf{x}_k$$

Richardson's method (2)

This process yields the following iterates:
Initial guess $\mathbf{x}_0 = \mathbf{0}$

$$\mathbf{x}_1 = \mathbf{b}$$

$$\mathbf{x}_2 = \mathbf{b} + (\mathbf{I} - \mathbf{A})\mathbf{x}_1 = \mathbf{b} + (\mathbf{I} - \mathbf{A})\mathbf{b}$$

$$\mathbf{x}_3 = \mathbf{b} + (\mathbf{I} - \mathbf{A})\mathbf{x}_2 = \mathbf{b} + (\mathbf{I} - \mathbf{A})\mathbf{b} + (\mathbf{I} - \mathbf{A})^2\mathbf{b}$$

Repeating this gives

$$\mathbf{x}_{k+1} = \sum_{i=0}^{k} (\mathbf{I} - \mathbf{A})^i \mathbf{b}$$

Richardson's method (3)

So Richardson's method 'generates' the series expansion for $(\mathbf{I} - \mathbf{Z})^{-1}$ with $\mathbf{Z} = \mathbf{I} - \mathbf{A}$. If this series converges we have

$$\sum_{i=0}^{\infty} (\mathbf{I} - \mathbf{A})^i = \mathbf{A}^{-1}.$$

The series expansion for $\frac{1}{1-z}$ $(z \in \mathbb{C})$ converges if $|z| < 1$.
The series $\sum_i (\mathbf{I} - \mathbf{A})^i$ converges if

$$|1 - \lambda| < 1 \quad \text{all eigenvalues } \lambda \text{ of } \mathbf{A}.$$

Richardson's method (3)

So Richardson's method 'generates' the series expansion for $(\mathbf{I}-\mathbf{Z})^{-1}$ with $\mathbf{Z}=\mathbf{I}-\mathbf{A}$. If this series converges we have

$$\sum_{i=0}^{\infty}(\mathbf{I}-\mathbf{A})^i = \mathbf{A}^{-1}.$$

The series expansion for $\frac{1}{1-z}$ $(z \in \mathbb{C})$ converges if $|z|<1$.

The series $\sum_i(\mathbf{I}-\mathbf{A})^i$ converges if

$$\lambda \in \{\zeta \in \mathbb{C} \mid |1-\zeta|<1\} \quad \text{all eigenvalues } \lambda \text{ of } \mathbf{A}.$$

For λ real this means that $0<\lambda<2$.

Richardson's method (4)

In order to increase the radius of convergence and to speed up the convergence, one can introduce a parameter α:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha(\mathbf{b}-\mathbf{A}\mathbf{x}_k) = \alpha\mathbf{b} + (\mathbf{I}-\alpha\mathbf{A})\mathbf{x}_k$$

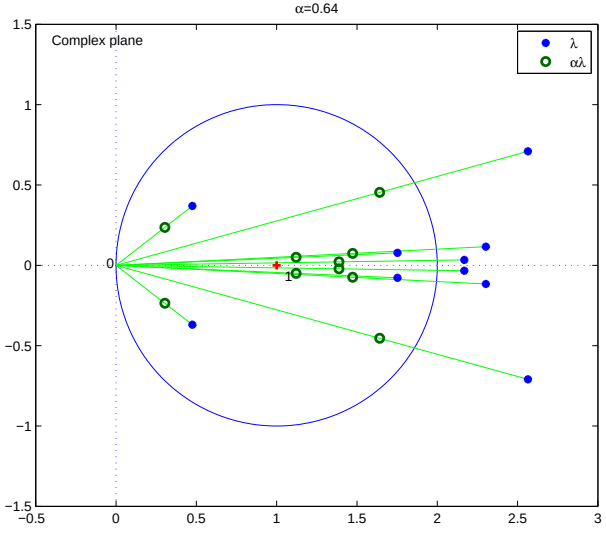
It is easy to verify that if all eigenvalues are real and positive the optimal α is given by

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

If all eigenvalues are in right half of the complex plane, i.e., $\mathrm{Re}(\lambda)>0$ all eigs. λ of $\mathbf{A}$, then, for some α

$$|1-\alpha\lambda|<1 \quad \text{all eigenvalues } \lambda \text{ of } \mathbf{A}.$$

Richardson's method (4)



Initial guess

Before, we assumed for the initial guess $\mathbf{x}_0=\mathbf{0}$.
Starting with another initial guess $\mathbf{x}_0$ only
means that we have to solve a "shifted" system

$$\mathbf{A}(\mathbf{y}+\mathbf{x}_0)=\mathbf{b} \quad \Leftrightarrow \quad \mathbf{A}\mathbf{y}=\mathbf{b}-\mathbf{A}\mathbf{x}_0=\mathbf{r}_0$$

So the results obtained before remain valid, irrespective of the initial guess.

Stopping criterion

We want to stop once the error $\|\mathbf{x}_k - \mathbf{x}\| < \epsilon$, with ϵ some prescribed tolerance. Unfortunately we do not know $\mathbf{x}$, so this criterion does not work in practice.
Alternatives are:

- $\|\mathbf{r}_k\| = \|\mathbf{b} - \mathbf{A}\mathbf{x}_k\| = \|\mathbf{A}\mathbf{x} - \mathbf{A}\mathbf{x}_k\| < \epsilon$
  Disadvantage: criterion not scaling invariant
- $\|\mathbf{r}_k\|/\|\mathbf{r}_0\| < \epsilon$
  Disadvantage: good initial guess does not reduce the number of iterations
- $\|\mathbf{r}_k\|/\|\mathbf{b}\| < \epsilon$
  Seems best (fits the idea of a small backward error).

Convergence

To investigate the convergence of Basic Iterative Methods in general, we look again at the formula

$$\mathbf{M}\mathbf{x}_{k+1} = \mathbf{R}\mathbf{x}_k + \mathbf{b}.$$

Remember that $\mathbf{A} = \mathbf{M} - \mathbf{R}$. If we subtract $\mathbf{M}\mathbf{x} = \mathbf{R}\mathbf{x} + \mathbf{b}$ from this equation we get a recursion for the error $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}$:

$$\mathbf{M}\mathbf{e}_{k+1} = \mathbf{R}\mathbf{e}_k$$

Convergence (2)

We can also write this as

$$\mathbf{e}_{k+1} = \mathbf{M}^{-1}\mathbf{R}\mathbf{e}_k$$

This is a power iteration and hence the error will ultimately point in the direction of the dominant eigenvector of $\mathbf{M}^{-1}\mathbf{R}$.
The rate of convergence is determined by
the *spectral radius* $\rho(\mathbf{M}^{-1}\mathbf{R})$ of $\mathbf{M}^{-1}\mathbf{R}$ $(= \mathbf{I} - \mathbf{M}^{-1}\mathbf{A})$:

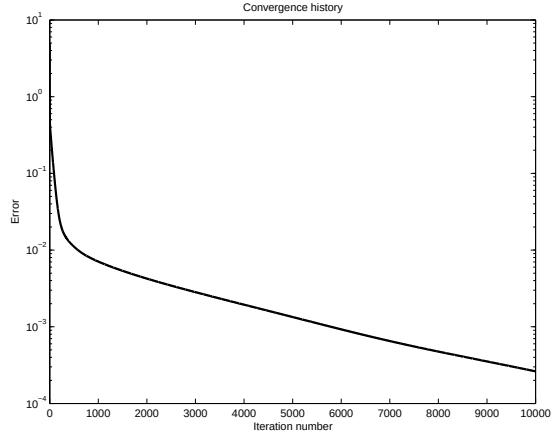
$$\begin{aligned} \rho(\mathbf{M}^{-1}\mathbf{R}) &\equiv \max\{|\lambda| \mid \lambda \text{ eigenvalue } \mathbf{M}^{-1}\mathbf{R}\} \\ &= \max\{|1-\lambda| \mid \lambda \text{ eigenvalue } \mathbf{M}^{-1}\mathbf{A}\} \end{aligned}$$

For convergence we must have that

$$\rho(\mathbf{M}^{-1}\mathbf{R}) < 1.$$

Linear convergence

Ultimately, we have $\|\mathbf{e}_{k+1}\| \approx \rho(\mathbf{M}^{-1}\mathbf{R})\,\|\mathbf{e}_k\|$, which means that we have linear convergence.



The vertical axis displays the size $\|\mathbf{e}_k\|_2$ of the error on $\log$-scale.

Classical Basic Iterative Methods

We will now briefly discuss the three best known basic iterative methods

- Jacobi's method
- The method of Gauss-Seidel
- Successive overrelaxation

These methods can be seen as Richardson's method applied to the preconditioned system

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

Jacobi's method

We first write $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$, with

$\mathbf{L}$ the strictly lower triangular part of $\mathbf{A}$,
$\mathbf{D}$ the main diagonal, and
$\mathbf{U}$ the strictly upper triangular part.

Jacobi's method is defined by the choice

$$\mathbf{M} \equiv \mathbf{D} \quad \Rightarrow \quad \mathbf{R} = -\mathbf{L} - \mathbf{U}.$$

The process is given by

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

or, equivalently, by

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

The Gauss-Seidel method

We write again $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$.

The **Gauss-Seidel** method is defined by the choice

$$\mathbf{M} \equiv \mathbf{L} + \mathbf{D} \quad \Rightarrow \quad \mathbf{R} = -\mathbf{U}.$$

The process is given by

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

or, equivalently, by

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

Successive overrelaxation (SOR)

We write again $\mathbf{A} = \mathbf{L} + \mathbf{D} + \mathbf{U}$.

The **SOR method** is defined by the choice

$$\mathbf{M} \equiv \frac{1}{\omega}\mathbf{D} + \mathbf{L} \quad \Rightarrow \quad \mathbf{R} = (\frac{1}{\omega} - 1)\mathbf{D} - \mathbf{U}.$$

The parameter ω is called the **relaxation parameter**.

The process is given by

$$(\tfrac{1}{\omega}\mathbf{D} + \mathbf{L})\mathbf{x}_{k+1} = ((\tfrac{1}{\omega} - 1)\mathbf{D} - \mathbf{U})\mathbf{x}_k + \mathbf{b}$$

or as

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (\tfrac{1}{\omega}\mathbf{D} + \mathbf{L})^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_k)$$

With $\omega = 1$ we get the method of Gauss-Seidel back.
In general the optimal value of ω is not known.

Iterative refinement

Two weeks ago, we saw direct methods. For numerical stability it is necessary to perform partial pivoting. However, this goes at the expense of the efficiency.

If the LU-factors are inaccurate, such that $\mathbf{A} = \mathbf{LU} - \Delta_A$, they might still be usable as *preconditioner* for the process

$$\mathbf{x}_{k+1} = \mathbf{x}_k + (\mathbf{LU})^{-1}(\mathbf{b} - \mathbf{A}\mathbf{x}_k)$$

This is called **iterative refinement** and is used to improve the accuracy of the direct solution.

One-step projection methods

The convergence of Richardson's method is not guaranteed and if the method converges, convergence is often very slow.

We now introduce two methods that are guaranteed to converge for wide classes of matrices. The two methods take special linear combinations of the vectors $\mathbf{r}_k$ and $\mathbf{A}\mathbf{r}_k$ to construct a new iterate $\mathbf{x}_{k+1}$ that satisfies a local optimality property.

Steepest descent

Let $\mathbf{A}$ be Hermitian positive definite. Define the function

$$f(\mathbf{x}_k) \equiv \|\mathbf{x}_k - \mathbf{x}\|_A^2 = (\mathbf{x}_k - \mathbf{x})^* \mathbf{A} (\mathbf{x}_k - \mathbf{x})$$

Let $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{r}_k$. Then the choice

$$\alpha_k = \frac{\mathbf{r}_k^* \mathbf{r}_k}{\mathbf{r}_k^* \mathbf{A} \mathbf{r}_k}$$

minimizes $f(\mathbf{x}_{k+1})$ (given $\mathbf{x}_k$ and $\mathbf{r}_k$).

Theorem. Steepest decent converges
if $\mathbf{A}$ is Hermitian positive definite.

Convergence is still usually very slow.

(Local) Minimal residual

Let $\mathbf{A}$ be general square. Define the function

$$g(\mathbf{x}_k) \equiv \|\mathbf{b} - \mathbf{A}\mathbf{x}_k\|_2^2 = \mathbf{r}_k^* \mathbf{r}_k$$

Let $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{r}_k$. Then the choice

$$\alpha_k = \frac{\mathbf{r}_k^* \mathbf{A} \mathbf{r}_k}{\mathbf{r}_k^* \mathbf{A}^* \mathbf{A} \mathbf{r}_k}$$

minimizes $g(\mathbf{x}_{k+1})$ (given $\mathbf{x}_k$ and $\mathbf{r}_k$).

Theorem. The local minimal residual method converges
if $\mathrm{Re}(\lambda) > 0$ for all eigenvalues λ of $\mathbf{A}$.

Convergence is still usually very slow.

Orthogonality properties

The optimality properties of the steepest descent method and the minimal residual method are equivalent with the following orthogonality properties:

For **steepest descent**

$$\alpha_k = \frac{\mathbf{r}_k^* \mathbf{r}_k}{\mathbf{r}_k^* \mathbf{A} \mathbf{r}_k} \quad \Leftrightarrow \quad \mathbf{r}_{k+1} \perp \mathbf{r}_k.$$

For the (local) **minimal residual method**

$$\alpha_k = \frac{\mathbf{r}_k^* \mathbf{A} \mathbf{r}_k}{\mathbf{r}_k^* \mathbf{A}^* \mathbf{A} \mathbf{r}_k} \quad \Leftrightarrow \quad \mathbf{r}_{k+1} \perp \mathbf{A} \mathbf{r}_k \ .$$

Concluding remarks

During the next lessons, the steepest decent method and the minimal residual method will be generalised.

This will ultimately give rise to a class of optimal iterative methods.

Moreover, we will see that these methods are closely linked to eigenvalue method (as the simple iterative methods are to the Power method).