Iterative Methods for Linear Systems of Equations

Projection methods (2)

ITMAN PhD-course DTU 20-10-08 till 24-10-08

Martin van Gijzen

Overview day 3

- Arnoldi's method
- Computation of an approximate solution
- Solution methods:
  - FOM
  - GMRES
- Convergence theory
- Truncation and flexible methods: GCR and GMRESR

Introduction

Yesterday we saw that an orthogonal basis for the Krylov subspace $K^k(A; r_0)$ can be constructed using Arnoldi's method.

In the symmetric case Arnoldi's method is very efficient: new basis vectors can be computed using a three-term recursion; involving only two old basis vectors. This feature makes it possible to construct very efficient iterative solvers: CG and CR (MINRES). These methods combine short recurrences with an optimality condition for the error.

Today we will start looking at iterative solvers for nonsymmetric systems. We start with methods that use Arnoldi to construct a basis for $K^k(A; r_0)$. Since we make no assumptions on symmetry, all previously computed basis vectors are needed to compute a new one.

Arnoldi's method

Choose a starting vector q_1 with $\|q_1\|_2 = 1$.

FOR	$k = 1, \cdots$ DO	iteration
	$v = Aq_k$	expansion
	FOR $i = 1, ..., k$	orthogonalisation
	$h_{i,k} = v^T q_i$	
	$v = v - h_{i,k} q_i$	
	END FOR	
	$h_{k+1,k} = \|v\|_2$	
	IF $h_{k+1,k} = 0$ STOP	invariant subspace spanned
	$q_{k+1} = v / h_{k+1,k}$	new basis vector
END	FOR	

The Arnoldi relation (1)

Recall that Arnoldi's method can be summarised in a compact way. Let

$$H_k = \begin{bmatrix} h_{1,1} & \cdots & \cdots & h_{1,k} \\ h_{2,1} & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ O & & h_{k,k-1} & h_{k,k} \end{bmatrix}$$

and $Q_k = [q_1 \; q_2 \cdots q_k]$ then

$$AQ_k = Q_k H_k + h_{k+1,k} q_{k+1} e_k^T$$

Here e_k is the $k-th$ canonical basis vector in $\mathbb{R}^k$.

The Arnoldi relation (2)

By defining

$$\underline{H}_k = \begin{bmatrix} h_{1,1} & \cdots & \cdots & h_{1,k} \\ h_{2,1} & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ & & h_{k,k-1} & h_{k,k} \\ O & & & h_{k+1,k} \end{bmatrix}$$

the Arnoldi relation can also be written as

$$AQ_k = Q_{k+1}\underline{H}_k$$

Optimal approximations (1)

Arnoldi's method provides an orthogonal basis for the Krylov subspace $K^k(A; r_0)$. Our approximations can be written as

$$x_k = x_0 + Q_k y_k$$

Yesterday we computed y_k so that either the error

$$f(x_k) = \|x_k - x\|_A^2 = (x_k - x)^T A(x_k - x)$$

is minimised in A-norm, or that

$$g(x_k) = \|A(x_k - x)\|_2^2 = r_k^T r_k,$$

i.e. the norm of the residual is minimised.

Optimal approximations (2)

If A is not SPD, the A-norm is not a proper norm. However, we also saw that in the SPD-case, minimising the A-norm of the error yields residuals that are orthogonal to Q_k.

Imposing that the residual is orthogonal to Q_k gives

$$Q_k^T(r_0 - AQ_k y_k) = 0 \Rightarrow \|r_0\|e_1 - H_k y_k = 0$$

with e_1 the first canonical basis vector of dimension k.

Solving the latter (small) system gives

$$y_k = \|r_0\|H_k^{-1}e_1 \quad x_k = x_0 + Q_k y_k$$

This method is called Full Orthogonalisation Method or FOM.

The Full Orthogonalisation Method

FOM is equivalent to CG if A is SPD. Unfortunately, it has some important disadvantages:

- FOM is not memory-efficient: Q_k has to be stored completely. Every iteration, a new basis vector has to be computed and stored. Orthogonalisation of new basis vectors also becomes increasingly more expensive with k.
- FOM does not have an optimality property
- The method is finite, but this is only of theoretical importance. To reach this point a complete set of at most n basis vectors has to be computed and stored.
- FOM is not robust. H_k can be singular and hence not invertible.

Galerkin and Petrov-Galerkin

The FOM-method to find x_k is part of a family of techniques to extract an approximate solution from a *search space* Q_k by making the residual orthogonal to a *test space* W_k. Formally, this can be formulated as:

Let $x_k = x_0 + Q_k y_k$. Find y_k such that

$$W_k^T(r_0 - AQ_k y_k) = 0$$

Such conditions are called Petrov-Galerkin conditions.
If $W_k = Q_k$ we call it a Galerkin condition.

Optimal approximations (3)

We now look at the second way to obtain in some sense optimal approximations: minimising the norm of the residual. Clearly minimising

$$g(x_k) = \|A(x_k - x)\|_2^2 = r_k^T r_k$$

is a well-defined problem, also if A is nonsymmetric.

Minimal residuals

The problem is: find $x_k = x_0 + Q_k y_k$ such that $\|r_k\|$ is minimal.

$$r_k = b - Ax_k = r_0 - AQ_k y_k = \|r_0\| q_1 - AQ_k y_k$$

hence

$$\begin{aligned} \|r_k\| &= \|\|r_0\| q_1 - AQ_k y_k\| \\ &= \|\|r_0\| Q_{k+1} e_1 - Q_{k+1} \underline{H}_k y_k\| \\ &= \|\|r_0\| e_1 - \underline{H}_k y_k\| \end{aligned}$$

GMRES (1)

Solving the overdetermined system

$$\underline{H}_k y_k = \|r_0\| e_1$$

provides us with iterates

$$x_k = x_0 + Q_k y_k$$

that minimise the residual. The resulting algorithm is called GMRES.

GMRES, which was proposed by Saad and Schultz, is one of the most popular methods for solving nonsymmetric systems.

GMRES (2)

GMRES is equivalent to MINRES if A is symmetric. Other important features are:

- GMRES is not limited-memory: Q_k has to be stored completely. Every iteration, a new basis vector has to be computed and stored. Orthogonalisation of new basis vectors also becomes increasingly more expensive with k.
- GMRES minimises the norm of the residual.
- The method is finite, but this is only of theoretical importance. To reach this point a complete set of at most n basis vectors has to be computed and stored .
- GMRES is robust: $\underline{H}_k y_k = \|r_0\| e_1$ always has a least-squares solution.

Another way of deriving GMRES

GMRES can also be derived by imposing a Petrov-Galerkin condition with $W_k = AQ_k$:

$$W_k^T r_k = Q_k^T A^T r_k = 0.$$

Using $r_k = \|r_0\| q_1 - AQ_k y_k$ gives

$$\|r_0\| Q_k^T A^T q_1 - Q_k^T A^T A Q_k y_k = 0.$$

With $AQ_k = Q_{k+1} \underline{H_k}$ we get

$$\|r_0\| \underline{H_k}^T e_1 - \underline{H_k}^T \underline{H_k} y_k = 0$$

which are exactly the normal equations that correspond to the overdetermined system $\underline{H_k} y_k = \|r_0\| e_1$.

Optimality and orthogonality

We already saw that for A SPD

$$\min_{x_k} \|x_k - x\|_A \text{ with } x_k \in \{x_0 \cup K^k(A; r_0)\}$$

implies that $Q_k^T r_k = 0$, and hence that

$$r_k \perp K^k(A; r_0) \ .$$

Now, we have also seen that

$$\min_{x_k} \|b - Ax_k\|_2 \text{ with } x_k \in \{x_0 \cup K^k(A; r_0)\}$$

implies that $Q_k^T A^T r_k = 0$ and hence that

$$r_k \perp AK^k(A; r_0) = K^k(A; Ar_0) \ .$$

Convergence

Like CG and MINRES, FOM and GMRES show superlinear convergence.

Since GMRES minimises the residual norm, it decreases monotonically and convergence is smooth.

FOM on the other hand does not minimise anything which results in more erratic convergence.

A bound on the GMRES-residual norm

Many bounds on the GMRES-residual norm are known. We mention the classical bound by Elman:

Suppose A is real and $\frac{1}{2}(A + A^T)$ positive definite. Let

$$\theta = \lambda_{max}(\frac{1}{2}(A + A^T)).$$

Then, the GMRES-residual norm after k iterations satisfies

$$\|r^k\|/\|r^0\| \leq (1 - \frac{\theta^2}{\|A\|^2})^{k/2}.$$

In practice, this bound is useful, but very pessimistic.

GMRES(m)

If many iterations have to be performed GMRES can become prohibitively expensive, with respect to both memory and computing time.

An easy solution is to restart the process after a cycle of m iterations, where m is a parameter chosen by the user.

This procedure can have a very strong negative effect on the rate of convergence. Typically, convergence becomes linear, and superlinear convergence is lost.

Note that the one-step minimal residual method we discussed on monday is equivalent to GMRES(1).

GCR (1)

A method that explicitly constructs residuals that satisfy

$$r_k \perp AK^k(A; r_0) \ .$$

is the Generalised Conjugate Residual method. Clearly GCR is equivalent to GMRES.

GCR computes two sets of basis vectors, and memory requirements are twice as high as for GMRES. GCR, however, also has important advantages.

The GCR method (2)

choose x_0, compute $r_0 = b - Ax_0$

FOR $k = 1, 2, \ldots$ DO

$s_k = r_{k-1}$,

$v_k = As_k$,

FOR $j = 1, \ldots, k-1$ DO

$\alpha = v_k^T v_j$,

$v_k := v_k - \alpha v_j$, $s_k := s_k - \alpha s_j$,

END FOR

$v_k := v_k / \|v_k\|_2$, $s_k := s_k / \|v_k\|_2$

$x_k := x_{k-1} + (r_{k-1}^T v_k) s_k$;

$r_k := r_{k-1} - (r_{k-1}^T v_k) v_k$;

END FOR

The GCR method (3)

GCR generates two sets of basis vectors:

$$s_1, s_2, \ldots, s_k$$

which form a basis for $K^k(A; r_0)$, and

$$v_1, v_2, \ldots, v_k$$

which form an orthogonal basis for $AK^k(A; r_0)$.

The GCR method (3)

Although storing two sets of vectors is a considerable overhead, it gives two big advantages:

- GCR can be truncated, which can be much more efficient than restarting. For example, if A is symmetric, truncating after one vector (i.e. keeping only the vectors s_{k-1} and v_{k-1} from the previous iterations) yields the CR method.
- x_k and r_k can always be computed in a consistent way, independent of how s_k is constructed. This makes it possible to apply another iterative technique as a (variable) preconditioner. The method that results if GMRES is used as preconditioner is called GMRESR (by Vuik and Van der Vorst).

GMRESR (1)

choose x_0, m, compute $r_0 = b - Ax_0$

FOR $k = 1, 2, \ldots$ DO

$s_k = P_m(A) r_{k-1}$,

$v_k = A s_k$,

for $j = 1, \ldots, k-1$ do

$\alpha = v_k^T v_j$,

$v_k := v_k - \alpha v_j$, $s_k := s_k - \alpha s_j$,

END FOR

$v_k := v_k / \|v_k\|_2$, $s_k := s_k / \|v_k\|_2$

$x_k := x_{k-1} + (r_{k-1}, v_k) s_k$;

$r_k := r_{k-1} - (r_{k-1}, v_k) v_k$;

END FOR

GMRESR (2)

In the algorithm $P_m(A)r$ denotes 'apply m GMRES iterations to the residual'.

GMRESR is an example of a so-called flexible method, which allows the application of a preconditioner that varies from iteration to iteration. A flexible variant of GMRES also exists.

By tuning well the work in the outer GCR loop and the inner GMRES loop, it is possible to optimise for the computational work and/or for memory requirements.

Concluding remarks

Today, we have discussed GMRES-type methods for nonsymmetric problems. The advantage of these methods is that they minimise the residual. The disadvantage is that they use long recurrences, using many vectors.

Tomorrow, we will concentrate on iterative solvers for nonsymmetric problems that use short recurrences.