1 13th of December 2018 — F. Poloni

1.1 Lanczos algorithm

Do you recall?

In last lecture we saw how to factorize a matrix $A \in M(m, \mathbb{R})$ with Arnoldi (i.e. $AQ_n = Q_{n+1}\underline{H}_n = Q_n\underline{H}_n + h_{n+1,n}q_{n+1}{e_1}^T$).

If A is **symmetric**, something special happens: $\underline{H}_n = Q_m^T A Q_m$ is symmetric as well, so it is a **tridiagonal** matrix. This improves the complexity of the Arnoldi process, because many iterations of the for loop (`j = 1 : n`) are not needed anymore, we need only two iterations.

We term the symmetric Arnoldi Lanczos algorithm, and such algorithm reduces the cost to n matrix products + $O(mn)$.

Suppose $A = A^T$ is positive definite. Then, we can find the solution to $Ax = b$ by minimizing the (strictly convex) function $f(x) = \frac{1}{2}x^T Ax - b^T x$.

Surprisingly, conjugate gradient on this problem can be interpreted as a Krylov subspace method.

The pseudocode of such algorithm can be found in Algorithm 1.1, where x_k is the current iterate, $r_k = b - Ax_k = -\nabla f(x_k)$ is the residual and d_k is the search direction.

Algorithm 1.1 Pseudocode for the conjugate gradient method.

```
1:  procedure CG_ITERATION
2:      $x_0 \leftarrow 0$;
3:      $r_0 \leftarrow b$;
4:      $d_0 \leftarrow b$;
5:      for k = 1:n do
6:          $\alpha_k \leftarrow ({r_{k-1}}^T r_{k-1})/({d_{k-1}}^T A d_{k-1})$;
7:          $x_k \leftarrow x_{k-1} + \alpha_k d_{k-1}$;
8:          $r_k \leftarrow r_{k-1} - \alpha_k A d_{k-1}$;
9:          $\beta_k \leftarrow ({r_k}^T r_k)/({r_{k-1}}^T r_{k-1})$;
10:         $d_k \leftarrow r_k + \beta_k d_{k-1}$;
11:     end for
12: end procedure
```

Notice that the search direction (line 10) is modified adding a multiple of the previous search direction to the residual and β_k is chosen such that d_k and d_{k-1} are A-orthogonal (formally, ${d_k}^T A d_{k-1} = 0$).

Conversely, the next point is chosen in order to minimize the objective function $f(x_{k-1}\alpha_k d_{k-1})$.

As far as the complexity is concerned, the spacial complexity is constant (three vectors) and the time complexity is O(mn).

Theorem 1.1. $K_k(A,b) = span(x_1, x_2, \ldots, x_k) = span(d_0, d_1, \ldots, d_{k-1}) = span(r_0, r_1, \ldots, r_{k-1})$.

Theorem 1.2. *The residuals are orthogonal and the search directions are A-orthogonal. Formally,* $r_j{}^T r_k = d_i{}^T A d_k = 0, \ \forall \ i < k$.

Proof. By induction: Let us assume we proved the thesis $r_j{}^T r_k = 0$ for $k-1, k-2, \ldots, 0$. Since $x_k = x_{k-1}\alpha_k d_{k-1}$, the residual $r_k = b - Ax_k = b - A(x_{k-1} + \alpha_k d_{k-1}) = b - Ax_{k-1} - \alpha_k A d_{k-1} = r_{k-1} - \alpha_k A d_{k-1}$.

Let us compute $r_j{}^T r_k = r_j{}^T \cdot (r_{k-1} - \alpha_k A d_{k-1})$.

- If $i < k-1$ then $r_j{}^T r_{k-1} - r_j{}^T \alpha_k A d_{k-1} = 0$, because the first term is 0 from induction hypothesis and $r_j{}^T \alpha_k A d_{k-1} = 0$, because $r_j \in K_{k-1}(A,b) = span(d_0, d_1, \ldots, d_{k-2})$.
- If $i = k-1$, $0 = r_{k-1}{}^T \cdot (r_{k-1} - \alpha_k A d_{k-1}) = r_{k-1}{}^T r_{k-1} - \alpha_k r_{k-1}{}^T A d_{k-1}$ holds if $\alpha_k = \frac{r_{k-1}^T r_{k-1}}{r_{k-1}^T A d_{k-1}}$. We are left with proving that $\alpha_k = \frac{r_{k-1}^T r_{k-1}}{r_{k-1}^T A d_{k-1}} = \alpha_k = \frac{r_{k-1}^T r_{k-1}}{d_{k-1}^T A d_{k-1}}$.

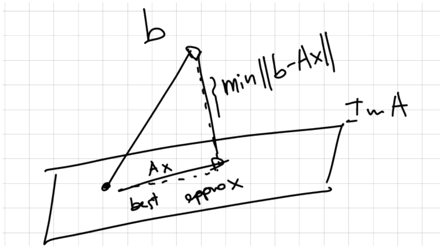
  This is true, since $d_{k-1} = r_{k-1} + \beta_{k-1} d_{k-2}$, so $d_{k-1}{}^T A d_{k-1} = (r_{k-1} + \beta_{k-1} d_{k-2})^T A d_{k-1} = r_{k-1}{}^T A d_{k-1} + \beta_{k-1} d_{k-2} A d_{k-1}{}^T$ and the second part is equal to 0 by induction.

□

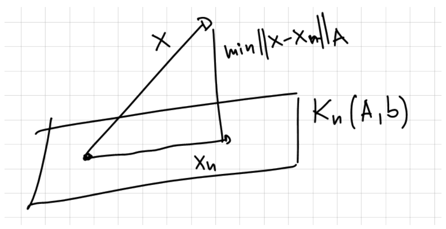
Notice that this base is not orthonormal, we need to rescale it to obtain an orthonormal one, moreover, $\frac{1}{\|r_i\|} r_i$ coincides (up to a sign) with the q_i obtained with Arnoldi.

We are left with writing the equation we need to solve at each iteration, namely we need to ensure that $r_k = b - Ax_k$ is orthogonal to all vectors of $K_k(A,b)$ which is equivalent to requiring $Q_k{}^T \cdot (b - Ax_k) = 0$ or, equivalently, $\|b\| \cdot e_1 = H_k y_k$.

In figure Figure 1.1 we can see a comparison between Arnoldi algorithm and the conjugate gradient.



(A) Arnoldi



(B) Conjugate gradient

FIGURE 1.1: Traditional orthogonality (Arnoldi) leads to the minimization of the 2-norm, while in the conjugate gradient we impose A-orthogonality and we get a good approximation in several norms.

As far as convergence speed is concerned,

Theorem 1.3. *x_k is the best approximation of the exact (and unknown) solution x to $Ax = b$ in $K_k(A,b)$ in the A-norm, i.e.* $x_k = arg \min_{z \in K_k(A,b)} (x-z)^T A (x-z)$

Theorem 1.4. *Let $\lambda_{\max}$, $\lambda_{\min}$ be the maximum/minimum eigenvalue of A; then, CG converges with rate*

$$\|x - x_k\| \leq \left(\frac{\sqrt{\lambda_{\max}} - \sqrt{\lambda_{\min}}}{\sqrt{\lambda_{\max}} + \sqrt{\lambda_{\min}}} \right)^k \|x - x_0\| .$$

We can rewrite it in terms of a more familiar quantity: for a positive definite matrix, eigenvalues and singular values coincide, hence

$$\frac{\sqrt{\lambda_{\max}} - \sqrt{\lambda_{\min}}}{\sqrt{\lambda_{\max}} + \sqrt{\lambda_{\min}}} = \frac{\sqrt{\sigma_1} - \sqrt{\sigma_m}}{\sqrt{\sigma_1} + \sqrt{\sigma_m}} = \frac{\sqrt{\frac{\sigma_1}{\sigma_m}} - 1}{\sqrt{\frac{\sigma_1}{\sigma_m}} + 1} = \frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1}.$$

For large values of $\kappa(A)$, this is approximately $1 - \frac{2}{\sqrt{\kappa(A)}}$, while if $\kappa(A) \approx 1$ the convergence speed is very high.

As for GMRES, if A has only n different eigenvalues, then this minimum reaches 0 after n steps. If the eigenvalues of A are 'clustered', one can construct polynomials such that $q(\lambda)$ is small for each λ then fast convergence is implied.