Conjugate Gradient Method

Problem: Linear system $Ax = b$ with $A \in \mathbb{C}^{n \times n}$ hermition and positive-definite.

Solving this is equivalent to minimizing the function

$$f(y) = \frac{1}{2} y^T A y - y^T b$$

since $\nabla f = Ay - b$. The positive-definiteness of A guarantees the uniqueness of the minimizer x.

Idea to find x: Gradient descent with the added constraint that the subsequent search directions p_i are conjugate wrt. A, i.e.

$$(p_i, p_j) := p_i^H A p_j = 0 \quad \text{for } i \neq j.$$

The algorithm works as follows:

<u>Init</u> First guess x_0 & starting direction $p_0 = b - Ax_0$

$\longrightarrow$ next guess: $x_{k+1} = x_k + \alpha_k p_k$ w/ $\alpha_k = \frac{p_k^H r_k}{(p_k, p_k)}$

(the latter follows from $\frac{d}{d\alpha_k} f(x_{k+1}) \stackrel{!}{=} 0$)

$\longrightarrow$ new residual: $r_{k+1} = b - Ax_{k+1} = r_k - \alpha_k A p_k$

$\longrightarrow$ new search direction: $p_{k+1} = r_{k+1} - \sum_{i \leq k} \frac{p_i^H A r_{k+1}}{(p_i, p_i)} p_i$

(comp. of r_k conjugate to all previous p_i)

$\longrightarrow$ repeat until $\|r_{k+1}\| < \varepsilon$

The algorithm can be simplified using the fact that the r_k are pairwise orthogonal. This can be proven inductively:

Hypothesis: $r_j^H r_k = 0$ for all $j < k$ $\quad (*)$

$\boxed{k=1}$

$$r_0^H r_1 = r_0^H (r_0 - \alpha_0 A p_0) = r_0^H r_0 - r_0^H r_0 = 0 \checkmark$$

$\boxed{k \longrightarrow k+1}$ $(j < k+1)$

$$r_j^H r_{k+1} = r_j^H (r_k - \alpha_k A p_k) = r_j^H r_k - \alpha_k r_j^H A p_k$$

$$= r_j^H r_k - \alpha_k \Big(p_j + \sum_{i \leq j-1} \frac{p_i^H A r_j}{(p_i, p_i)} p_i \Big)^H A p_k$$

$$= r_j^H r_k - \alpha_k \Big((p_j, p_k) + \sum_{i<j} \frac{p_i^H A r_j}{(p_i, p_i)} \underbrace{(p_i, p_k)}_{=0 \text{ since } i<j\leq k} \Big)$$

$$= r_j^H r_k - \alpha_k (p_j, p_k)$$

If $j<k$, $r_j^H r_k = 0$ (using ⊛) and $(p_j, p_k)=0$ by construction. If $j=k$:

$$r_j^H r_{k+1} = r_k^H r_k - \underbrace{p_k^H}_{\text{Lin. combination of } \{r_i\}_{i=0,\dots,k}\,!} r_k \overset{⊛}{=} r_k^H r_k - r_k^H r_k = 0 \quad \blacksquare$$

From this property it follows that

$$\alpha_k = \frac{p_k^H r_k}{(p_k, p_k)} \overset{⊛}{=} \frac{r_k^H r_k}{(p_k, p_k)} \quad \text{and}$$

$$p_{k+1} = r_{k+1} - \sum_{i\leq k} \frac{p_i^H A r_{k+1}}{(p_i, p_i)} p_i = r_{k+1} - \sum_{i\leq k} \alpha_i \frac{p_i^H A r_{k+1}}{r_i^H r_i} p_i$$

$$\overset{A=A^H}{=} r_{k+1} - \sum_{i\leq k} \frac{(\alpha_i A p_i)^H r_{k+1}}{r_i^H r_i} p_i$$

$$= r_{k+1} - \sum_{i\leq k} \frac{(r_i - r_{i+1})^H r_{k+1}}{r_i^H r_i} p_i$$

$$\overset{⊛}{=} r_{k+1} + \frac{r_{k+1}^H r_{k+1}}{r_k^H r_k} p_k \,.$$

Thus the algorithm simplifies to:

<u>Init</u> First guess x_0 & starting direction $p_0 = b - Ax_0$
$\longrightarrow x_{k+1} = x_k + \alpha_k p_k$ w/ $\alpha_k = \frac{r_k^H r_k}{(p_k, p_k)}$
$\longrightarrow r_{k+1} = r_k - \alpha_k A p_k$
$\longrightarrow p_{k+1} = r_{k+1} + \frac{r_{k+1}^H r_{k+1}}{r_k^H r_k} p_k$
$\longrightarrow$ repeat until $\lVert r_{k+1} \rVert < \varepsilon$