EE 381V: Large Scale Optimization	**Fall 2014**
Lecture 9 — September 25	
Lecturer: Sanghavi	*Scribe: Avradeep Bhowmik & Shalmali Joshi*

(Note that the main reference for this lecture is Nocedal & Wright (Chapter 5 and 6))

9.1 Topics Covered Last Time

- Newton's Method
- Self-Concordant functions
- Interior Point Methods

We have seen in previous lectures that gradient descent methods have simpler updates which depend only on the gradient $g_f(x)$ but require more iterations for convergence, while Newton's method gives better guarantees for rate of convergence but both requires the knowledge of as well as involves an inversion of the $n \times n$ Hessian matrix $H_f(\cdot)$. Let us review what Newton's method does.

Recall that the Newton step, Δx_{nt}, is a solution to the minimization of a quadratic approximating function $\hat{f}(x + \Delta x_{\text{nt}})$:

$$\begin{aligned}\hat{f}(x + \Delta x_{\text{nt}}) &\triangleq f(x) + g_f(x)^T \Delta x_{\text{nt}} + \Delta x_{\text{nt}}^T H_f(x) \Delta x_{\text{nt}} \\ &= f(x) + g_f(x)^T \Delta x_{\text{nt}} + \|\Delta x_{\text{nt}}\|_{H_f(x)}\end{aligned}$$

which is the solution to the linear system:

$$H_f(x)\Delta x_{\text{nt}} = g_f(x).$$

In case the function $f(x)$ is a quadratic function, that is $f(x) = \frac{1}{2}x^\top Ax - bx$, the approximation is exact and hence Newton's method with step size = 1 converges in a single step as follows

$$\begin{aligned}x^* = x^+ &\leftarrow x - H_f^{-1} g_f(x) \\ &= x - A^{-1}(Ax - b) \\ &= A^{-1}b\end{aligned}$$

The idea behind adapting Newton's Method comes from steepest descent, where the step direction is determined using a different norm, $||\cdot||_{**}$. If the norm chosen is a quadratic norm $||\cdot||_{**} = \|\cdot\|_B$ (defined as $\|v\|_B = v^\top Bv$ where B is a positive definite matrix), the

steepest descent step direction is given by $\Delta x_{\text{sd}} = -B^{-1} g_f(x)$. [1] The Newton update therefore chooses a different norm at every step as $\|\cdot\|_B = \|\cdot\|_{H_f(x)}$, which is the choice of B that results in faster convergence close to the optimum. [2] But this choice requires computing and inverting the Hessian; is there a better choice that converges faster than a fixed-norm approach, but is computationally cheaper than an optimal approach? Essentially, we would like a method which avoids the computational cost in Newton's method of inverting an $n \times n$ matrix at every step while at the same time has faster convergence than generic gradient descent methods.

There are many variable metric methods for solving this problem. These use a different norm at each step, to avoid calculation of second derivatives and simplify the calculation of the search direction. The basic idea of variable metric methods is to iteratively construct a good approximation to the inverse Hessian by building a sequence of matrices. The conjugate gradient methods are a set of methods which build upon this idea.

9.2 The Conjugate Gradient Method

The conjugate gradient method has a very natural motivation when the function to be optimised is quadratic. Consider the following optimisation problem

$$\min_x \phi(x) = \frac{1}{2} x^\top A x - bx \tag{9.1}$$

where A is a *positive definite* matrix. Since at any x_k we have $\nabla\phi(x_k) = Ax_k - b$ which is the residual r_k at x_k, solving this optimisation problem is equivalent to finding a solution of the system of linear equations

$$\nabla\phi(x^*) = 0 \equiv Ax^* = b \tag{9.2}$$

The solution to this is, of course, $x^* = A^{-1}b$. As we have seen previously, the Newton update applied to this problem results in converging to the optimum in a single step. However, the objective here is to get to the optimal x^* without actually computing the inverse of A.

Consider the simplest case when A is the scaled identity matrix $a_0 I$- the solution is trivial to obtain by setting each coordinate as $x_i^* = \frac{b_i}{a_0}$. Similarly, in case A is a diagonal matrix we can still find the solution easily by sequentially setting the coordinates as $x_i^* = \frac{b_i}{A_{ii}}$. Note that for these simple case, the level sets of $\phi(x)$ are ellipsoids [3] with the common centre at x^* with the axes of each ellipsoid aligned parallel to the coordinate axes. Therefore starting from any point, we can reach the optimal point x^* by minimising along each axis exactly once- essentially, we are projecting the gradient $Ax - b = A(x - x^*)$ (appropriately scaled)

[1] Recall that the dual norm of $\|\cdot\|_B$ for a positive definite B is given by $\|\cdot\|_{B^{-1}}$

[2] Refer to section 9.4.4 in Boyd & Vandenberghe for a more detailed explanation

[3] Strictly speaking, it is the boundary of the level sets that are ellipsoids, not the level sets themselves

along the direction of the axes of the ellipsoid which happen to be aligned in this case to the coordinate axes, which is equivalent to setting each coordinate sequentially as

$$\begin{aligned} x_i^+ &= x_i - \frac{(Ax-b)^\top e_i}{e_i^\top A e_i} e_i \\ &= x_i - \frac{r_i^\top e_i}{A_{ii}} e_i \\ &= \frac{b_i}{A_{ii}} \end{aligned}$$

For a more general positive definite matrix, the level sets are still ellipsoids but now their axes are no longer aligned parallel to the coordinate axes, hence the updates are no longer coordinate-wise. However, if we know the directions we can still obtain one-step minimisation along each direction. This idea is formalised in the conjugate gradient method.

Definition 1. Conjugate vectors : *A set of nonzero vectors $\{p_0, p_1, \cdots, p_n\}$ are said to be conjugate with respect to the symmetric positive definite matrix A if*

$$p_i^T A p_j = 0, \quad i \neq j. \tag{9.3}$$

Note that conjugate vectors are, therefore, orthogonal with respect to the transformation $x \to A^{1/2}x$ which maps $\mathbb{R}^n$ to a different inner product space [4] where the inner product is defined as $\langle x, y \rangle_A = x^\top A y$ and the set of (appropriately scaled) conjugate vectors in $\mathbb{R}^n$ correspond to the unit vectors along the coordinate axes in the new space. The following facts provide more intuition about conjugate vectors.

Proposition 1. *The set of eigenvectors (equivalently, singular vectors) of the matrix A are conjugate with respect to A.*

Proof: Consider the singular value decomposition of A as $A = U\Sigma\ U^\top$. Say u_i, u_j are columns of U, we have, since U is unitary, $u_i^\top u_j = 0$, $u_i^\top u_i = u_j^\top u_j = 1$. Therefore,

$$u_i^\top A u_j = u_i^\top U \Sigma\ U^\top u_j = e_i^\top \Sigma\ e_j = \Sigma_{ij}$$

which is 0 for $i \neq j$ since all the off-diagonal entries of Σ are 0. □

Note that since the singular value decomposition is not necessarily unique as far as the actual singular vectors are concerned (consider, for example, when two of the singular values are identical), the set of conjugate vectors for a given positive definite matrix are also not necessarily unique. The extreme case of this is when $A = I$ in which case any orthonormal basis on $\mathbb{R}^n$ is a set of conjugate vectors with respect to A.

However, the subspace mapped by the set of singular vectors is always unique. The following proposition shows that the subspace spanned by a set of n conjugate vectors is actually all of $\mathbb{R}^n$.

[4] In particular, a hypersphere in this new space maps to an ellipsoid in $\mathbb{R}^n$ and vice versa

Proposition 2. *A set of vectors which are conjugate with respect to some positive definite matrix are also linearly independent.*

Proof: Suppose this were not the case, say $\sum_{i\in\mathcal{I}} \nu_{ik} p_i = p_k$, then $\forall i \neq k$ and $i, k \in \mathcal{I}$ we have

$$0 = p_i^\top A p_k = -\nu_{ik} p_i^\top A p_i$$

Since A is positive definite, $p_i^\top A p_i > 0$ for all i, and therefore, we must have $\nu_{ik} = 0 \ \ \forall i \neq k$ and $\nu_{kk} = 1$. □

Thus, given a positive definite matrix A, a set of conjugate vectors with respect to A can act as a basis for the $\mathbb{R}^n$ space. In fact, it can be shown that they span a very special basis with respect to A. As we saw earlier, conjugate vectors in $\mathbb{R}^n$ map to the directions along the coordinate axes in a new space where the inner product in $\mathbb{R}^n$ corresponds to the quadratic inner product $\langle \cdot, \cdot \rangle_A$, and where the level sets of $\phi(x)$ are ellipsoids with principal axes aligned along the coordinate directions. It is, therefore, desirable to harness the useful properties of this new space to solve the optimisation problem (**??**) by making use of the given set of conjugate vectors.

Specifically, we make use of the orthogonality of the conjugate vectors in the new space by choosing them as the directions over which to perform the minimisation at each step. Consider for now that we have an oracle which gives us the set of conjugate vectors $\{p_i\}$ for any given positive definite matrix A. Given a starting point x_0, we perform updates in the directions p_i by choosing the step size using exact line search.

Therefore, our updates are of the form

$$x_{k+1} = x_k + \alpha_k p_k$$

where the step size α_k is chosen as

$$\alpha_k = \arg\min_{\alpha} \phi(x_k + \alpha p_k)$$

Optimising over α amounts to solving a quadratic minimisation problem in one dimension, which is relatively easy, and we therefore get a closed form solution as

$$\alpha_k = -\frac{r_k^\top p_k}{p_k^\top A p_k}. \tag{9.4}$$

where, as earlier, $r_k = Ax_k - b$ is the residual at the k^{th} step.

Algorithm 9.1 : Baby Conjugate Gradient

1. Input: $A \succ 0$, $b \in \mathbb{R}^n$, conjugate vectors $\{p_0, p_1, \cdots p_{n-1}\}$
2. Initialise: $x_0 \in \mathbb{R}^n$, $r_0 = Ax_0 - b$, $k = 0$
3. while $r_k \neq 0$ do:
   - $\alpha_k = -\frac{r_k^T p_k}{p_k^T A p_k}$
   - $x_{k+1} = x_k - \alpha_k p_k$
   - $r_{k+1} = Ax_{k+1} - b$
   - $k \leftarrow k + 1$
4. Output : x^*

We refer to this version of the algorithm as the "baby conjugate gradient" method, or B-CG. As we mentioned earlier, the point of choosing $\{p_i\}$ as the minimisation directions is so that we only need to perform a single minimisation step along each of the chosen directions to reach the global minimum. This claim is formalised in the following theorem.

Theorem 9.1. *For any $x_0 \in R^n$ the sequence $\{x_k\}$ generated by Algorithm 9.1 converges to the solution x^* of the optimisation problem (**??**) in at most n steps.*

Proof: Since the directions p_i are linearly independent, they must span the whole space $\mathbb{R}^n$ which includes the initial error vector $x^* - x_0$. Hence, we have

$$x^* - x_0 = \sigma_0 p_0 + \sigma_1 p_1 + \cdots + \sigma_{n-1} p_{n-1},$$

for some scalars σ_k. Computing the scalars can be done sequentially in the standard procedure by taking the respective inner products $\langle p_k, x^* - x_0 \rangle_A = p_k^\top A(x^* - x_0) = \sum_j \sigma_j p_k^\top A p_j$ Since the $\{p_j\}$ are conjugate, the inner product terms $p_k^\top A p_j$ disappear for $j \neq k$ and we obtain

$$\sigma_k = \frac{p_k^T A(x^* - x_0)}{p_k^T A p_k} \tag{9.5}$$

Since x^* is a solution to the system $Ax - b$, we have $Ax^* = b$ and therefore, we obtain the coefficients as

$$\begin{aligned} \sigma_k &= \frac{p_k^\top (b - Ax_0)}{p_k^\top A p_k} \\ &= -\frac{p_k^\top r_k}{p_k^\top A p_k} \\ &= \alpha_k \qquad \text{(from equation **??**)} \end{aligned}$$

Therefore, the coefficients σ_k coincide with the step length α_k generated by the formula in equation (**??**). In fact, at the k^{th} step. we have

$$x^* - x_k = \sigma_k p_k + \sigma_{k+1} p_{k+1} + \cdots + \sigma_{n-1} p_{n-1}, \tag{9.6}$$

Therefore, $x_n = x^*$ and we can see that we need at most n steps to reach the optimum. □

The last part in the proof for the above theorem suggests the following result.

Theorem 9.2. *For the B-CG algorithm as outlined above, we have*

1. $r_k^\top p_i = 0$ *for all* $i = 0, 1, \cdots k-1$
2. x_k *is the minimiser of* $\phi(x) = \frac{1}{2}x^\top Ax - bx$ *over the set* $x_0 + span\{p_0, p_1, \cdots p_{k-1}\}$

Proof: The first part is easy to prove by premultiplying equation **??** by $p_i^\top A$ for $i = 0, 1, \cdots k-1$ and using the expression for $-r_k = b - Ax_k = A(x^* - x_k)$, and the facts that p_i for $i = 0, 1, \cdots k-1$ is conjugate with every p_j for $j = k, k+1, \cdots n-1$. Writing it out, we have $r_k^\top p_i = 0$ for all $i = 0, 1, \cdots k-1$, which proves the first part of the theorem.

The second part of the proof comes from basic KKT conditions. Recall that y^* is the minimiser of a function $f(y)$ over the set C if and only if $\nabla f(y^*) \in \mathcal{N}_C(y^*)$ where $\mathcal{N}_C(y^*)$ is the normal cone of C at y^*. It can be shown that for $C = x_0 + span\{p_0, p_1, \cdots p_{k-1}\}$, the normal cone is exactly $span\{Ap_k, Ap_{k+1}, \cdots Ap_n\}$ (proof sketch: the tangent cone at any point in the affine subspace C is $span\{p_0, p_1, \cdots p_{k-1}\}$ and every vector in the tangent cone is perpendicular to every vector in $span\{Ap_k, Ap_{k+1}, \cdots Ap_n\}$- moreover any vector perpendicular to any vector in $span\{Ap_k, Ap_{k+1}, \cdots Ap_n\}$ are conjugate to $\{p_k \cdots p_n\}$ and therefore can only have components in the subspace $span\{p_0, p_1, \cdots p_{k-1}\}$, therefore they are dual cones- in fact, they are orthogonal subspaces). Note that the gradient of $\phi(x)$ at x_k is $\nabla\phi(x_k) = Ax_k - b = A(x_k - x^*) = r_k$ which lies in $span\{Ap_k, Ap_{k+1}, \cdots Ap_n\}$ by equation **??**, hence we have our result.

All of the above can also be proved using the result in the first part. Because the gradient of $\phi(x)$ at x_k is $\nabla\phi(x_k) = Ax_k - b = r_k$, we have that the directional derivative [5] of $\phi(x)$ at x_k in the direction of p_i is proportional to $r_k^\top p_i$. Since every direction in the affine subspace $x_0 + span\{p_0, p_1, \cdots p_{k-1}\}$ is spanned by the set $\{p_0, p_1, \cdots p_{k-1}\}$, and we have that $r_k^\top p_i = 0$ for all $i = 0, 1, \cdots k-1$, the convex, differentiable function $\phi(x)$ at x_k is already at a local minimum with respect to the given affine subspace. Since a local minimum of a convex function in an affine subspace (which is a convex set) is is also the global minimum within that subspace, therefore we have that x_k is the minimiser of $\phi(x) = \frac{1}{2}x^\top Ax - bx$ over the set $x_0 + span\{p_0, p_1, \cdots p_{k-1}\}$. □

[5] Recall that the directional derivative of a function $f(y)$ at $y = y_0$ in the direction v (where $\|v\| = 1$) is equal to $\langle \nabla f(y_0), v \rangle$

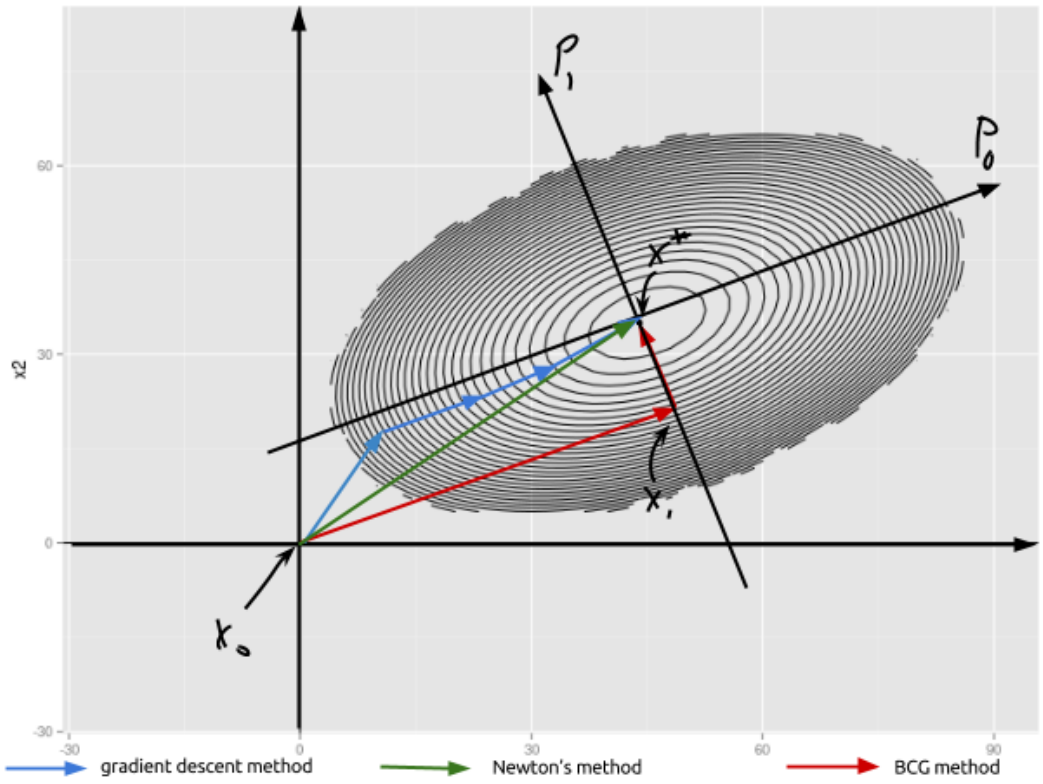

Figure 9.1. Comparison of CG with GD and Newton's method

The figure **??** shows the updates for the iterations of the conjugate gradient descent as compared to gradient descent and Newton's method. Note that Newton's method points directly towards the minimum, which is what Newton's method does by specifying the optimal direction via the inverse Hessian. Gradient descent does not choose an overall optimal direction but simply chooses the best direction at every step. Conjugate gradient descent chooses the best directions a priori such that the minimisation along each direction needs to be done exactly once.

Remark 1. As mentioned earlier, *the B-CG algorithm in $\mathbb{R}^n$ is equivalent to performing coordinate descent with exact line search in a different space defined by the transformation $x \to T^{-1}x$ where T is the $n \times n$ matrix with $\{p_i\}$ as its columns.*

Using the new variable as $\hat{x} = T^{-1}x$ the quadratic function to be minimised ϕ becomes

$$\hat{\phi}(\hat{x}) = \phi(T\hat{x}) = \frac{1}{2}\hat{x}^\top (T^\top AT)\hat{x} - (T^\top b)^\top \hat{x}.$$

By the conjugacy property **??**, the matrix $T^\top AT$ is diagonal. Hence we can find the minimizing value of $\hat{\phi}$ by performing n one-dimensional minimization steps along the coordinate

directions of $\hat{x}$, which correspond to the directions p_i in our original space. Therefore, B-CG method in $\mathbb{R}^n$ is equivalent to a coordinate descent with exact line search in the new coordinate system.

Remark 2. *We have seen that given the conjugate vectors, we can get optimal descent directions so that the algorithm takes at most n steps to converge.* In general, however, the conjugate vectors need not be known beforehand. We have seen that the set of singular vectors for the A matrix are conjugate with respect to A- however, computing the singular vectors can often be at least as expensive as computing the matrix inverse, which was what we were trying to avoid all this time.

It would be, therefore, immensely useful if we had a (relatively) inexpensive way of computing conjugate vectors on the fly. The following modification to the B-CG algorithm does just that. Everything remains the same except that instead of having a known set of conjugate vectors at the initialisation stage as the input, we start with one conjugate vector and compute the remaining conjugate vectors iteratively at each step in the following way.

First, we start with $p_0 = -r_0 = b - Ax_0$. For every subsequent step, we generate a new conjugate gradient $p_k,\ k = 0, 1, 2, \cdots n-1$ as

$$p_{k+1} = -r_{k+1} + \beta_{k+1} p_k \tag{9.7}$$

where the scalar β_k is chosen such that p_k and p_{k-1} are conjugate with respect to A. By premultiplying both sides of equation (**??**) by $p_{k-1}^\top A$ and setting the left hand side to 0, we can find the value of β_k as

$$\beta_{k+1} = \frac{r_{k+1}^\top A p_k}{p_k^\top A p_k} \tag{9.8}$$

This algorithm only seems to ensure that p_k and p_{k+1} are conjugate to each other- it is not obvious that conjugacy is maintained throughout the set of p_i vectors which are computed at every step. We will now show that the above set of updates is actually sufficient to ensure that p_{k+1} are conjugate to all $p_i,\ i \leq k$.

Algorithm 9.2 : Conjugate Gradient Method

1. Input: $A \succ 0$, $b \in \mathbb{R}^n$
2. Initialise: $x_0 \in \mathbb{R}^n$, $r_0 = Ax_0 - b$, $p_0 = -r_0$, $k = 0$
3. while $r_k \neq 0$ do:
   - $\alpha_k = -\frac{r_k^\top p_k}{p_k^\top A p_k}$
   - $x_{k+1} = x_k - \alpha_k p_k$
   - $r_{k+1} = Ax_{k+1} - b$
   - $\beta_{k+1} = \frac{r_{k+1}^\top A p_k}{p_k^\top A p_k}$
   - $p_{k+1} = -r_{k+1} + \beta_{k+1} p_k$
   - $k \leftarrow k + 1$
4. Output : x^*

Theorem 9.3. *Suppose $r_{k+1} \neq 0$, then $r_{k+1} \perp r_i$, i.e., $r_{k+1}^\top r_i = 0$, for $i = 0, 1, \cdots, k$. Furthermore, $\{p_i : i = 1, 2, \cdots k+1\}$ are conjugate with respect to A, i.e., $p_j^\top A p_i = 0$, for $i, j \in \{0, 1, \cdots, k+1\}$, $i \neq j$.*

Proof: The proof is by induction and proceeds in several steps. By our update rule, we have p_0 and p_1 are conjugate to each other. Assume that till step k, all p_i are conjugate to each other for $i = 0, 1, \cdots k$. We start off by proving the first part of the theorem.

We have by our induction hypothesis that p_i are conjugate to each other for $i = 0, 1, \cdots k$, therefore, in a manner identical [6] to Theorem **??**, $r_{k+1}^\top p_i = 0$ for all $i = 0, 1, \cdots, k$.
By rearranging **??**, we find that $p_i = -r_i + \beta_i p_{i-1}$, so that $r_i \in \text{span}\{p_i, p_{i-1}\}$ for all $i = 1, 2, \cdots, k$. Hence, we conclude that $r_{k+1}^\top r_i = 0$ for all $i = 1, \cdots, k$. This completes the proof for the first part of the theorem.

For the proof of the next part of the theorem, we shall need the following results.

Lemma **??***-1*: $\text{span}\{r_0, r_1, \cdots, r_k\} = \text{span}\{r_0, Ar_0, \cdots, A^k r_0\}$

Lemma **??***-2*: $\text{span}\{p_0, p_1, \cdots, p_k\} = \text{span}\{r_0, Ar_0, \cdots, A^k r_0\}$

Proof of Lemmata The proofs are by induction. Lemmata **??**-1 and **??**-2 hold for $k = 0$. We

[6] Note that while it is not immediately obvious from the way we proved Theorem **??**-1 that it holds even when we have only assumed conjugacy upto p_k, we still have the result from Theorem **??**-2 since the minimisations over the directions $p_i, i \leq k$ only depend on the the fact that they are conjugate. Therefore, it follows that Theorem **??**-1 holds as well by the exact argument that was used to prove Theorem **??**-2

show that assuming these properties are true for some $k = j$, they hold for $k = j + 1$. First, we show that each set on the left hand side is included in the corresponding set on the right hand side side. By our induction hypothesis,

$$r_j \in \text{span}\{r_0, Ar_0, A^2r_0, \cdots, A^jr_0\}$$
$$p_j \in \text{span}\{r_0, Ar_0, A^2r_0, \cdots, A^jr_0\}$$

By multiplying the second expression by A,

$$Ap_j \in \text{span}\{Ar_0, A^2r_0, A^3r_0, \cdots, A^{j+1}r_0\}$$

We have $r_{j+1} = Ax_{j+1} - b = A(x_j - \alpha_j p_j) - b = r_j - \alpha_j Ap_j$. Combining this with the result for Ap_j above, we have $r_{j+1} \in \text{span}\{r_0, Ar_0, A^2r_0, \cdots, A^{j+1}r_0\}$. Therefore, since by our update rule $p_{j+1} \in \text{span}\{r_{j+1}, p_j\}$, we also have $p_{j+1} \in \text{span}\{r_0, Ar_0, A^2, r_0, \cdots, A^{j+1}r_0\}$. Therefore, the set on the left hand side is included in the set on the right hand side.

For the reverse inclusion, note that by our induction hypothesis for Lemma **??**-2, we have $A^{j+1}r_0 = A.A^jr_0 \in A(\text{span}\{p_0, p_1, \cdots, p_j\}) = \text{span}\{Ap_0, Ap_1, \cdots, Ap_j\}$. Since $Ap_i = \frac{r_{i+1}-r_i}{\alpha_i} \in \text{span}\{r_i, r_{i+1}\}$, we have $A^{j+1}r_0 \in \text{span}\{Ap_0, Ap_1, \cdots, Ap_j\} \subseteq \text{span}\{r_0, r_1, \cdots, r_{j+1}\}$. Thus we have equality for $\text{span}\{r_0, r_1, \cdots, r_{j+1}\} = \text{span}\{r_0, Ar_0, \cdots, A^{j+1}r_0\}$ as well which completes the proof for the Lemma **??**-1.

We can prove Lemma **??**-2 by the following argument:

$$\begin{aligned}
\text{span}\{p_0, p_1, \cdots, p_j, p_{j+1}\} &= \text{span}\{p_0, p_1, \cdots, p_j, r_{j+1}\} && \text{(since } p_{j+1} \in \text{span}\{r_{j+1}, p_j\}\text{)} \\
&= \text{span}\{r_0, Ar_0, \cdots, A^jr_0, r_{j+1}\} && \text{(by induction hypothesis)} \\
&= \text{span}\{r_0, r_1, \cdots, r_j, r_{j+1}\} && \text{(by induction hypothesis)} \\
&= \text{span}\{r_0, Ar_0, \cdots, A^{j+1}r_0\}.
\end{aligned}$$

This completes the proof for Lemma **??**-2.

We now prove the statement of the theorem. By our induction hypothesis we already have that p_i is conjugate to p_j for all $i, j < k + 1$, and from the way we defined β_{k+1}, p_k is also conjugate to p_{k+1}. We now show that p_{k+1} is also conjugate to all $p_i, i = 1, 2, \cdots k - 1$.

Multiplying **??** by $p_i^\top A, i = 0, 1, \cdots, k - 1$ we have

$$p_i^\top Ap_{k+1} = -p_i^\top Ar_{k+1} + \beta_{k+1} p_i^\top Ap_k \tag{9.9}$$

From our induction hypothesis, $p_i^\top Ap_k = 0$, so the second term vanishes in the right hand side of the above equation. Also, we have already seen that $r_{k+1}^\top p_i = 0$ for $i = 0, 1, \cdots, k$. By repeatedly applying Lemma **??**-3, we can find that

$$Ap_i \in \text{span}\{Ar_0, A^2r_0, \cdots, A^{i+1}r_0\} \subseteq \text{span}\{p_0, p_1, \cdots, p_{i+1}\}.$$

Therefore, we can deduce that $r_{k+1}^T Ap_i = 0$ for $i = 0, 1, \cdots, k - 1$. Consequently, the first and second term in the right side of **??** vanishes, and we conclude that $p_k^T Ap_i = 0$, for $i = 0, 1, \cdots, k - 1$ as well. This completes the proof. □

Note that the algorithm 9.2 requires only matrix vector multiplications and no inversions, hence it is still relatively inexpensive. In the next lecture we shall look at further properties of this algorithm and generalise it to other cases.

Summary:

1. Conjugate Gradient method applied to a quadratic function results in one-step minimisation along each direction specified by a set of conjugate vectors
2. The algorithm corresponds to coordinate descent with exact line search in a transformed space
3. For any positive definite matrix A-
   - Eigenvectors of A are conjugate with respect to A
   - The set of conjugate vectors are linear independent
4. The gradient at each step (correspondingly, the residual of the linear system) is orthogonal to every direction that has already been optimised over, $r_k^\top p_i = 0, i < k$
   - In particular, the x_k at every step is the global minimiser for the objective function over the affine subspace $x_0 + span\{p_0, \cdots p_k\}$
5. Conjugate vectors need not be specified in advance- they can be computed at each step using only matrix-vector computations which are relatively inexpensive
6. Each new conjugate vector is computed as a linear combination of the gradient at the particular step and the previous conjugate vector- while the linear combination is chosen to only ensure conjugacy between the computed vectors at successive iterations, the procedure automatically guarantees that the entire set of computed vectors are conjugate with each other