0368-3248-01-Algorithms in Data Mining **Fall 2011**

Lecture 9: The power method

Lecturer: Edo Liberty

Warning: *This note may contain typos and other inaccuracies which are usually discussed during class. Please do not cite this note as a reliable source. If you find mistakes, please inform me.*

We give a simple algorithm for computing the Singular Value Decomposition of a matrix $A \in \mathbb{R}^{m \times n}$. We start by computing the first singular value σ_1 and left and right singular vectors u_1 and v_1 of A, for which $min_{i<j} \log(\sigma_i/\sigma_j) \geq \lambda$:

1. Generate x_0 such that $x_0(i) \sim \mathcal{N}(0, 1)$.
2. $s \leftarrow \log(4 \log(2n/\delta)/\varepsilon\delta)/2\lambda$
3. for i in $[1, \ldots, s]$:
4. $\quad x_i \leftarrow A^T A x_{i-1}$
5. $v_1 \leftarrow x_i/\|x_i\|$
6. $\sigma_1 \leftarrow \|Av_1\|$
7. $u_1 \leftarrow Av_1/\sigma_1$
8. return (σ_1, u_1, v_1)

Let us prove the correctness of this algorithm. First, write each vector x_i as a linear combination of the right singular values of A i.e. $x_i = \sum_j \alpha_j^i v_j$. From the fact that v_j are the eigenvectors of $A^T A$ corresponding to eigenvalues σ_j^2 we get that $\alpha_j^i = \alpha_j^{i-1}\sigma_j^2$. Thus, $\alpha_j^s = \alpha_j^0 \sigma_j^{2s}$. Looking at the ratio between the coefficients of v_1 and v_i for x_s we get that:

$$\frac{| < x_s, v_1 > |}{| < x_s, v_i > |} = \frac{|\alpha_1^0|}{|\alpha_i^0|} \left(\frac{\sigma_1}{\sigma_i}\right)^{2s}$$

Demanding that the error in the estimation of σ_1 is less than ε gives the requirement on s.

$$\frac{|\alpha_1^0|}{|\alpha_i^0|} \left(\frac{\sigma_1}{\sigma_i}\right)^{2s} \geq \frac{n}{\varepsilon} \tag{1}$$

$$s \geq \frac{\log(n|\alpha_i^0|/\varepsilon|\alpha^0|_1)}{2\log(\sigma_1/\sigma_i)} \tag{2}$$

From the two-stability of the gaussian distribution we have that $\alpha_i^0 \sim \mathcal{N}(0, 1)$. Therefore, $\Pr[\alpha_i^0 > t] \leq e^{-t^2}$ which gives that with probability at least $1-\delta/2$ we have for all i, $|\alpha_i^0| \leq \sqrt{\log(2n/\delta)}$. Also, $\Pr[|\alpha_1^0| \leq \delta/4] \leq \delta/2$ (this is because $\Pr[|z| < t] \leq max_r \Psi_z(r) \cdot 2t$ for any distribution and the normal distribution function at zero takes it maximal value which is less than 2) Thus, with probability at least $1 - \delta$ we have that for all i, $\frac{|\alpha_1^0|}{|\alpha_i^0|} \leq \frac{\sqrt{\log(2n/\delta)}}{\delta/4}$. Combining all of the above we get that it is sufficient to set $s = \log(4n \log(2n/\delta)/\varepsilon\delta)/2\lambda = O(\log(n/\varepsilon\delta)/\lambda)$ in order to get ε precision with probability at least $1 - \delta$.

We now describe how to extend this to a full SVD of A. Since we have computed (σ_1, u_1, v_1), we can repeat this procedure for $A - \sigma_1 u_1 v_1^T = \sum_{i=2}^n \sigma_i u_i v_i^T$. The top singular value and vectors of which are (σ_2, u_2, v_2). Thus, computing the rank-k approximation of A requires $O(mnks) = O(mnk\log(n/\varepsilon\delta))/\lambda)$

operations. This is because computing $A^T Ax$ requires $O(mn)$ operations and for each of the first k singular values and vectors this is performed s times.

The main problem with this algorithm is that its running time is heavily influenced by the value of λ. Other variants of this algorithm are much less sensitive to the value of this parameter, but are out of the scope of this class.