# 1 PCA Generalizations

## 1.1 Robust PCA

If you run `python main -t 1` the code will load a dataset $X$ where each row contains the pixels from a single frame of a video of a highway. The demo applies PCA to this dataset and then uses this to reconstruct the original image. It then shows the following 3 images for each frame:

1. The original frame.
2. The reconstruction based on PCA.
3. A binary image showing locations where the reconstruction error is non-trivial.

Robust PCA is a variation on PCA where we replace the L2-norm with the L1-norm,

$$f(Z, W) = \sum_{i=1}^{n} \sum_{j=1}^{d} |\langle w^j, z_i \rangle - x_{ij}|,$$

and it has recently been proposed as a more effective model for background subtraction.

**"multi-quadric" approximation:**

$$|\alpha| \approx \sqrt{\alpha^2 + \epsilon},$$

where $\epsilon$ controls the accuracy of the approximation (a typical value of $\epsilon$ is 0.0001).

**Objective Function and Derivatives**

$$f(Z, W) = \sum_{i=1}^{n} \sum_{j=1}^{d} |\langle w^j, z_i \rangle - x_{ij}|,$$

$$\approx \sum_{i=1}^{n} \sum_{j=1}^{d} \sqrt{\left(\langle w^j, z_i \rangle - x_{ij}\right)^2 + \epsilon}$$

Let $l$ and $m$ be $\{l \in \mathbb{N} \mid 1 \leq l \leq n\}$ and $\{m \in \mathbb{N} \mid 1 \leq m \leq k\}$. Since,

$$\frac{\partial}{\partial Z_{lm}} \langle w^j, z_i \rangle = \frac{\partial}{\partial Z_{lm}} \sum_{p=1}^{k} z_{ip} \cdot w_{pj} = \begin{cases} w_{mj} & \text{if} \quad i = l, \\ 0 & \text{otherwise.} \end{cases}$$

Therefore,

$$\frac{\partial f(Z, W)}{\partial Z_{lm}} = \frac{\partial}{\partial Z_{lm}} \sum_{i=1}^{n} \sum_{j=1}^{d} \sqrt{\left(\langle w^j, z_i \rangle - x_{ij}\right)^2 + \epsilon}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{d} \frac{1}{2} \cdot \frac{1}{\sqrt{\left(\langle w^j, z_i \rangle - x_{ij}\right)^2 + \epsilon}} \cdot 2 \cdot \left(\langle w^j, z_i \rangle - x_{ij}\right) \cdot \frac{\partial}{\partial Z_{lm}} \langle w^j, z_i \rangle$$

$$= \sum_{j=1}^{d} \cdot \frac{\langle w^j, z_l \rangle - x_{lj}}{\sqrt{\left(\langle w^j, z_l \rangle - x_{lj}\right)^2 + \epsilon}} \cdot w_{mj}$$

$$= \langle w_m,\ \texttt{np.multiply(R[l,:],1/f\_mat[l,:])} \rangle$$

`np.multiply()` is element-wise multiplication of matrices. `f_mat` is the objective function before summation (matrix).

So,

$$\frac{\partial f(Z,W)}{\partial Z} = \texttt{np.multiply(R,1/f\_mat)@W.T}$$

`np.multiply(R,1/f_mat)` is a $(n \times d)$ matrix and `W.T` is a $(d \times k)$ matrix. So we obtain a gradient matrix with the same size as $Z$.

As for $\dfrac{\partial f(Z,W)}{\partial W}$, let $p$ and $q$ be $\{p \in \mathbb{N} \mid 1 \le p \le k\}$ and $\{q \in \mathbb{N} \mid 1 \le q \le d\}$,

$$\frac{\partial}{\partial W_{pq}} \langle w^j, z_i \rangle = \frac{\partial}{\partial W_{pq}} \sum_{r=1}^{k} z_{ir} \cdot w_{rj} = \begin{cases} z_{ip} & \text{if} \quad j = q, \\ 0 & \text{otherwise.} \end{cases}$$

$$\frac{\partial f(Z,W)}{\partial W_{pq}} = \frac{\partial}{\partial W_{pq}} \sum_{i=1}^{n} \sum_{j=1}^{d} \sqrt{\left(\langle w^j, z_i \rangle - x_{ij}\right)^2 + \epsilon}$$

$$= \sum_{i=1}^{n} \sum_{j=1}^{d} \frac{1}{2} \cdot \frac{1}{\sqrt{\left(\langle w^j, z_i \rangle - x_{ij}\right)^2 + \epsilon}} \cdot 2 \cdot \left(\langle w^j, z_i \rangle - x_{ij}\right) \cdot \frac{\partial}{\partial W_{pq}} \langle w^j, z_i \rangle$$

$$= \sum_{i=1}^{n} \cdot \frac{\langle w^q, z_i \rangle - x_{iq}}{\sqrt{\left(\langle w^q, z_i \rangle - x_{iq}\right)^2 + \epsilon}} \cdot z_{ip}$$

In the same way,

$$\frac{\partial f(Z,W)}{\partial W} = \texttt{Z.T@np.multiply(R,1/f\_mat)}$$

`Z.T` is a $(k \times n)$ matrix, `np.multiply(R,1/f_mat)` is a $(n \times d)$ matrix, so $\frac{\partial f(Z,W)}{\partial W}$ is a $(k \times d)$ matrix like we wanted.