Name: Anna Huang

AMATH 515 Homework Set 3

Due: Wednesday, March 3rd by midnight.

Let f be a closed proper convex function. The convex conjugate of f, called f^*, is defined by

$$f^*(z) = \sup_x \left\{ z^T x - f(x) \right\}.$$

(1) Compute the conjugates of the following functions.

(a) $f(x) = \delta_{\mathbb{B}_\infty}(x)$.

Answer to 1a: $f(x) = \delta_{\mathbb{B}_\infty}(x)$ is defined below:

$$\delta_{\mathbb{B}_\infty}(x) = \begin{cases} 0 & x \in [-1,1]^n \\ \infty & \text{o.w.} \end{cases}$$

When $\delta_{\mathbb{B}_\infty}(x) = 0$ is when $x \in [-1,1]^n$.

$$f^*(z) = \sup_x \left\{ z^T x - f(x) \right\}$$

$$f^*(z) = \sup_x \left\{ z^T x - \delta_{\mathbb{B}_\infty}(x) \right\}$$

$$f^*(z) = \sup_x \left\{ z^T x \right\}, x \in [-1,1]^n$$

This results in the following:

$$f^*(z) = \begin{cases} x = 1 & z \geq 0 \\ x = -1 & z \leq 0 \end{cases}$$

$$f^*(z) = \begin{cases} z & z \geq 0 \\ -z & z < 0 \end{cases}$$

$$f^*(z) = ||z||_1$$

(b) $f(x) = \delta_{\mathbb{B}_2}(x)$.

Answer to 1b: $f(x) = \delta_{\mathbb{B}_2}(x)$ is defined below:

$$\delta_{\mathbb{B}_2}(x) = \begin{cases} ||x||_2 \leq 1 & 0 \\ \text{o.w.} & \infty \end{cases}$$

When $\delta_{\mathbb{B}_\infty}(2) = 0$ is when $||x||_2 \leq 1$.

$$f^*(z) = \sup_x \left\{ z^T x - f(x) \right\}$$

$$f^*(z) = \sup_x \left\{ z^T x - \delta_{\mathbb{B}_2}(x) \right\}$$

$$f^*(z) = \sup_x \left\{ z^T x \right\}, ||x||_2 \leq 1$$

The max is when $||x||_2 = 1$, so therefore:

$$f^*(z) = ||z||_2$$

(c) $f(x) = \exp(x)$.
Answer to 1c:

$$f^*(z) = \sup_x \left\{z^T x - f(x)\right\}$$
$$f^*(z) = \sup_x \left\{z^T x - \exp(x)\right\}$$

To find the supremum of $z^T x - \exp(x)$, we find the derivative and set it equal to zero.

$$\frac{d}{dx}(z^T x - \exp(x)) = 0$$
$$z - \exp(x) = 0$$
$$x = ln(z)$$

Plugging x back into the equation,

$$\begin{aligned} f^*(z) &= z(\ln(z)) - e^{(\ln(z))} \\ &= z\ln(z) - z \\ &= z(\ln(z) - 1) \\ &= \begin{cases} z(\ln(z) - z & z > 0 \\ 0 & z = 0 \\ \infty & z < 0 \end{cases} \end{aligned}$$

(d) $f(x) = \log(1 + \exp(x))$
Answer to 1d:

$$f^*(z) = \sup_x \left\{z^T x - f(x)\right\}$$
$$f^*(z) = \sup_x \left\{z^T x - \log(1 + \exp(x))\right\}$$

To find the supremum of $z^T x - \log(1 + \exp(x))$, we find the derivative and set it equal to zero.

$$\frac{d}{dx}(z^T x - \log(1 + \exp(x))) = 0$$
$$z - \frac{e^x}{1 + e^x} = 0$$

To find x:

$$\begin{aligned} z &= \frac{e^x}{1+e^x} \\ \ln(z) &= \ln(e^x) - \ln(1+e^x) \\ \ln(z) &= x - \ln(1+e^x) \\ \ln(z) &= x - f(x) \\ x &= \ln(z) + f(x) \end{aligned}$$

Additionally,

$$\begin{aligned} z &= \frac{e^x}{1+e^x} \\ 1-z &= 1 - \frac{e^x}{1+e^x} \\ 1-z &= \frac{1}{1+e^x} \\ \ln(1-z) &= \ln(\frac{1}{1+e^x}) \\ &= \ln(1) - \ln(1+e^x) \\ &= -\ln(1+e^x) \\ &= -f(x) \end{aligned}$$

Plugging x back into the equation:

$$\begin{aligned} f^*(z) &= z(\ln(z) + f(x)) - f(x) \\ &= z\ln(z) + z[-\ln(1-z)] + \ln(1-z) \\ &= z\ln(z) - z(\ln(1-z)) + \ln(1-z) \\ &= z\ln(z) + \ln(1-z)(1-z) \\ &= z\log(z) + \log(1-z)(1-z) \end{aligned}$$

(e) $f(x) = x\log(x)$

Answer to 1e:

$$\begin{aligned} f^*(z) &= \sup_x \left\{ z^T x - f(x) \right\} \\ f^*(z) &= \sup_x \left\{ z^T x - x\log(x) \right\} \end{aligned}$$

To find the supremum of $z^T x - x\log(x)$, we find the derivative and set it equal to zero and find x:

$$\begin{aligned} \frac{d}{dx}(z^T x - x\log(x)) &= 0 \\ z - log(x) - 1 &= 0 \\ x &= e^{z-1} \end{aligned}$$

Plugging x back into the equation:

$$f^*(z) = z(e^{z-1}) - (e^{z-1})(z-1)$$
$$f^*(z) = e^{z-1}$$

(2) Let g be any convex function; f is formed using g. Compute f^* in terms of g^*.

(a) $f(x) = \lambda g(x)$.

Answer to 2a: To compute $f^*(z)$:

$$\begin{aligned} f^*(z) &= \sup_x \left\{ z^T x - \lambda g(x) \right\} \\ &= \sup_x \left\{ \lambda \frac{z^T x}{\lambda} - g(x) \right\} \\ &= \lambda g^*(\frac{z}{\lambda}) \end{aligned}$$

(b) $f(x) = g(x-a) + \langle x, b \rangle$.

Answer to 2b: To compute $f^*(z)$:

$$\begin{aligned} f^*(z) &= \sup_x \left\{ z^T x - [g(x-a) + < x, b >] \right\} \\ &= \sup_x \left\{ x^T z - g(x-a) - x^T b \right\} \end{aligned}$$

Setting $w = x - a$ and so $x = w + a$,

$$\begin{aligned} f^*(z) &= \sup_w \left\{ (w+a)^T z - g(w) - (w+a)^T b \right\} \\ &= \sup_w \left\{ w^T z + a^T z - g(w) - w^T b - a^T b \right\} \\ &= \sup_w \left\{ w^T (z-b) - g(w) \right\} + a^T (z-b) \\ &= g^*(z-b) + a^T (z-b) \end{aligned}$$

(c) $f(x) = \inf_z \{g(x,z)\}$.

Answer to 2c: To compute $f^*(z)$ and assuming $g(x,z)$ as $g(x,y)$ to avoid confusion:

$$\begin{aligned} f^*(z) &= \sup_x \left\{ z^T x - \inf_z \{g(x,y)\} \right\} \\ &= \sup_x \left\{ z^T x + \sup_z \{-g(x,y)\} \right\} \\ &= \sup_x \sup_z \left\{ z^T x - g(x,y) \right\} \} \\ &= \sup_x \sup_z \left\{ z^T x + 0^T y - g(x,y) \right\} \} \\ &= g^*(z, 0) \end{aligned}$$

(d) $f(x) = \inf_z \left\{ \frac{1}{2}\|x - z\|^2 + g(z) \right\}$

Answer to 2d: To compute $f^*(z)$, first define $G(x, z) = \frac{1}{2}\|x - z\|^2 + g(z)$ and using what we solved from 2c:

$$\begin{aligned} f^*(z) &= \sup_x \left\{ z^T x - \inf_z \left\{ \frac{1}{2}\|x - z\|^2 + g(z) \right\} \right\} \\ &= \sup_x \left\{ z^T x - \inf_z \{G(x, z)\} \right\} \\ &= G^*(z, 0) \end{aligned}$$

To calculate $G^*(z, 0)$, we know that $G^*(z, w) = \sup_x \left\{ x^T w - G(x, z) \right\}$:

$$\begin{aligned} G^*(z, 0) &= \sup_x \left\{ x^T(0) - G(x, z) \right\} \\ &= \sup_x \left\{ x^T(0) - \frac{1}{2}\|x - z\|^2 - g(z) \right\} \\ &= \sup_x \left\{ x^T(0) - \frac{1}{2}\|x - z\|^2 \right\} - g(z) \end{aligned}$$

From completing the square, we know that $\frac{1}{2}\|x - z\|^2 = \frac{1}{2} < x, x > - < x, z > + \frac{1}{2} < z, z >$ so therefore:

$$G^*(z, 0) = \sup_x \left\{ < x, 0 > -\frac{1}{2} < x, x > + < x, z > -\frac{1}{2} < z, z > \right\} - g(z)$$

$$G^*(z, 0) = \sup_x \left\{ < x, 0 > -\frac{1}{2} < x, x > + < x, z > \right\} - g(z) - \frac{1}{2} < z, z >$$

(3) Moreau Identities.

(a) Derive the Moreau Identity:

$$\text{prox}_f(z) + \text{prox}_{f^*}(z) = z.$$

Answer to 3a: The proximal operator is defined as:

$$\text{prox}_{tf}(y) = \arg\min_x \frac{1}{2t}||x-y||^2 + f(y)$$

and the convex conjugate of the proximal is:

$$\text{prox}_{f^*}(z) = \arg\min_y \frac{1}{2}||y-z||^2 + f^*(y)$$

Taking the optimality conditions in regards to the $\text{prox}_{f^*}(x)$, we assume $x^* = \text{prox}_{f^*}(x)$:

$$\begin{aligned} 0 &\in \partial(\text{prox}_{f^*}(x^*)) \\ 0 &\in \partial[\frac{1}{2}||y-z||^2 + f^*(x^*)] \\ 0 &\in (x^*-z) + \partial f^*(x^*) \\ (z-x^*) &\in \partial f^*(x^*) \\ x^* &\in \partial(z-x^*) \end{aligned}$$

From the conditions above, we can assume that the minimizer is x^*. Assuming the minimizer of the proximal operator $\text{prox}_{tf}(y) = \arg\min_x \frac{1}{2t}||x-y||^2 + f(y)$ is x^*, then we have the following:

$$\text{prox}_f(z) + \text{prox}_{f^*}(z) = x^* + (z-x^*) = z$$

(b) Use the Moreau identity and 1a, 1b to check your formulas for

$$\text{prox}_{\|\cdot\|_1}, \quad \text{prox}_{\|\cdot\|_2}$$

from last week's homework.

Answer to 3b: To verify the above Moreau Identity, the $\text{prox}_{\|\cdot\|_1}$ is the following:

$$\text{prox}_{\|\cdot\|_1} = \text{prox}_{\mathbb{B}_\infty}(z) + \text{prox}_{||z||_1}(z)$$

$$\text{prox}_{\|\cdot\|_1} = \max(\min(z_i, 1), -1) + \begin{cases} z_i - 1 & z_i \geq 1 \\ 0 & z_i \in [-1, 1] \\ z_i + 1 & z_i \leq -1 \end{cases}$$

$$\text{prox}_{\|\cdot\|_1} = \max(\min(z_i, 1), -1) + \begin{cases} 1 + z_i - 1 & z_i \geq 1 \\ z_i + 0 & z_i \in [-1, 1] \\ -1 + z_i + 1 & z_i \leq -1 \end{cases}$$

$$\text{prox}_{\|\cdot\|_1} = z$$

To verify the above Moreau Identity, the $\text{prox}_{\|\cdot\|_2}$ is the following:

$$\text{prox}_{\|\cdot\|_2} = \text{prox}_{\mathbb{B}_2}(z) + \text{prox}_{||z||_2}(z)$$

$$\text{prox}_{\|\cdot\|_2} = \begin{cases} z_i & z_i \in B_2 \\ \frac{z}{||z_i||_2} & z_i \text{not} \in B_2 \end{cases} + \begin{cases} 0 & z_i \in B_2 \\ z(\frac{||z||_2 - 1}{||z_i||_2}) & z_i \text{not} \in B_2 \end{cases}$$

$$\text{prox}_{\|\cdot\|_2} = \begin{cases} z_i + 0 & z_i \in B_2 \\ \frac{z}{||z_i||_2} + z(\frac{||z||_2 - 1}{||z_i||_2}) & z_i \text{not} \in B_2 \end{cases}$$

$$\text{prox}_{\|\cdot\|_2} = z$$

(4) Duals of regularized GLM. Consider the Generalized Linear Model family:

$$\min_x \sum_{i=1}^{n} g(\langle a_i, x\rangle) - b^T Ax + R(x),$$

Where g is convex and R is any regularizer.

(a) Write down the dual obtained by dualizing g.

Answer to 4a: To find the dual, we find the conjugate of $g(\langle a_i, x\rangle)$:

$$\sum_{i=1}^{n} g(a_i^T x) = \sum_{i=1}^{n} \sup_{wi} w_i(a_i^T x) - g^*(w_i)$$

and then plug in back into the GLM equation:

$$\begin{aligned}
&\inf_x \sup_w \sum_{i=1}^{n} \left\{w_i(a_i^T x) - g^*(w_i)\right\} - b^T Ax + R(x) \\
&= \inf_x \sup_w \sum_{i=1}^{n} \{-g^*(w_i)\} + w^T Ax - b^T Ax + R(x) \\
&= \sup_w \inf_x \sum_{i=1}^{n} \{-g^*(w_i)\} + w^T Ax - b^T Ax + R(x) \\
&= \sup_w - \sup_x \sum_{i=1}^{n} \{g^*(w_i)\} - w^T Ax + b^T Ax - R(x) \\
&= - \sup_w \sup_x \sum_{i=1}^{n} \{g^*(w_i)\} - w^T Ax + b^T Ax - R(x) \\
&= - \sup_w \sup_x (-w^T A + b^T A)x - R(x) + \sum_{i=1}^{n} \{g^*(w_i)\}
\end{aligned}$$

The $\sup_x(-w^T A + b^T A)x - R(x)$ equals to $R^*((b-w)^T A)$ so then the dual is:

$$- \sup_w \left\{ R^*((b-w)^T A) + \sum_{i=1}^{n} \{g^*(w_i)\} \right\}$$

(b) Specify your formula to Ridge-regularized logistic regression:

$$\min_x \sum_{i=1}^{n} \log(1 + \exp(\langle a_i, x\rangle)) - b^T Ax + \frac{\lambda}{2}\|x\|^2.$$

Answer to 4b: Using the dual from 4a, we have the following:

$$- \sup_w \left\{ R^*((b-w)^T A) + \sum_{i=1}^{n} \{g^*(w_i)\} \right\}$$

and $R(x) = \frac{\lambda}{2}||x||^2$ and $g(x) = \log(1+\exp(\langle a_i, x\rangle))$. $R(x)$ is self conjugate and we solved the conjugate of $g(x)$ from 1d. Therefore the dual will be:

$$-\sup_w \left\{ \frac{\lambda}{2}||\frac{(b-w)^T A}{\lambda}||^2 + \sum_{i=1}^{n} \{w_i \log(w_i) + \log(1-w_i)(1-w_i)\} \right\}$$

(c) Specify your formula to 1-norm regularized Poisson regression:

$$\min_x \sum_{i=1}^{n} \exp(\langle a_i, x\rangle) - b^T Ax + \lambda \|x\|_1.$$

Answer to 4c: Using the dual from 4a and $R(x) = \lambda||x||_1$ and $g(x) = \exp(\langle a_i, x\rangle)$. We have the conjugate of $g(x)$ from 1c. The conjugate of $R(x)$ is

$$f^*(x) = \delta_{\mathbb{B}_\infty}(x)$$

Therefore the dual will be the following:

$$-\sup_w \left\{ \delta_{\mathbb{B}_\infty}(\frac{(b-w+)^T A}{\lambda}) + \sum_{i=1}^{n} \{w_i \ln(w_i) - w_i)\} \right\}$$

Coding Assignment

Please download `515Hw3_Coding.ipynb` and `proxes.py` to complete problem (5).

(5) In this problem you will write a routine to project onto the capped simplex.
The Capped Simplex Δ_k is defined as follows:

$$\Delta_k := \{x : 1^T x = k, \quad 0 \le x_i \le 1 \quad \forall i.\}$$

This is the intersection of the k-simplex with the unit box.
The projection problem is given by

$$\text{proj}_{\Delta_k}(z) = \arg\min_{x \in \Delta_k} \frac{1}{2}\|x - z\|^2.$$

(a) Derive the (1-dimensional) dual problem by focusing on the $1^T x = k$ constraint.
Answer to 5a: Knowing that $\delta_0(1^T x - k) = \sup_\lambda \lambda(1^T x - k)$ and the dual is the following:

$$\begin{aligned}
&\sup_\lambda \text{argmin}_x \frac{1}{2}||x - z||^2 + \lambda(1^T x - k) + \delta_{[0,1]^n}(x) \\
&\sup_\lambda \min_x \frac{1}{2}||x - z||^2 + \lambda(1^T x - k) + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2}||x - z||^2 + \lambda(1^T x - k) + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2} < x, x > - < x, z > + \frac{1}{2} < z, z > + \lambda(1^T x - k) + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2} < x, x > - < x, z > + \frac{1}{2} < z, z > + \lambda(1^T x - k) + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2} < x, x > - < x, z > + \lambda(1^T x - k) + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2} < x, x > - < z^T - \lambda 1^T, x > + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \min_x \frac{1}{2}||x - (z - \lambda 1^T)||^2 + \delta_{B_{[0,1]^n}}(x) \\
&\sup_\lambda \text{proj}_{[0,1]^n}(z - \lambda) \\
&\sup_\lambda \max(\min(1, z - \lambda), 0))
\end{aligned}$$

(b) Implement a routine to solve this dual. It's a scalar root finding problem, so you can use the root-finding algorithm provided in the code.
Answer to 5b: From the earlier constraint $1^T x = k$, the root-finding problem becomes:

$$1^T \hat{x} = k$$

$$1^T \max(\min(1, z_i - \lambda), 0)) - k = 0$$

In the root finding problem, we will bisect the λ.

(c) Using the dual solution, write down a closed form formula for the projection. Use this formula, along with your dual solver, to implement the projection. You can use the unit test provided to check if your code id working correctly.
Answer to 5c: From problem 5a, the closed form formula for the projection is below:

$$\text{proj}_{[0,1]^n}(z - \lambda) = \max(\min(1, z - \lambda), 0))$$

(6) In this problem you will learn apply proximal operators to matrices to perform matrix completion. You'll find that you can recover most of the information in a low rank matrix despite only seeing a small percentage of the entries. At the end of this problem, you'll see something that is, in my opinion, **truly remarkable**.

Consider a matrix X, (for example, where it is a matrix of ratings, and $X_{i,j}$ is the rating that user i gave to item j). However, you only observed the entries $(i, j) \in \Omega$ where Ω is a subset of the entries $\{(i, j)\}_{i,j=1}^n$. In order to solve this, we assume that the matrix X is low rank, and we will use this to try prior knowledge to try and recover it. Similar to how the lasso or ℓ_1 penalty promotes sparsity in regression, the nuclear norm $\|\cdot\|_*$ promotes low rank matrices.[1] We will implement and experiment with a handful of approaches to the matrix completion problem.

(a) One natural approach to setting up an optimization problem that models this situation is to assume that X is rank k, and solve the optimization problem

$$\text{minimize } \|P \odot (Y - X)\|_F^2$$

$$\text{subject to } \text{rank}(Y) \leq k$$

where $\odot$ denotes the elementwise product, and P is a matrix with $P_{i,j} = 1$ when $(i, j) \in \Omega$ and 0 if $(i, j) \notin \Omega$ (essentially a matrix of which entries are measured). **Is this problem convex? Why or why not?**

[1] One interpretation of this assumption is that a low rank model is equivalent to a latent factor model with low dimensional latent factors. Assume that X is rank d. This is like saying that $X_{i,j} = \langle u_i, v_j \rangle$ for some set of vectors $\{u_i\}$ representing the rows (or users), and $\{v_j\}$, representing the columns (or items), with $u, v \in \mathbb{R}^d$

Answer to 6a: No, this problem is not convex because the constraint subject to $\text{rank}(Y) \leq k$ is not convex. Below are two 2x2 matrix J and P, rank 1 matrices:

$$J = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \; P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Taking both matrices of rank 1, a convex combination of J and P would be a rank 2 matrix and hence the set of rank 1 matrix is not convex. The rank constraint counts the number of nonzero singular values making the average of both J and P to have rank 2.

(b) We relax the constraint above into a nuclear norm constraint

$$\text{minimize } \|P \odot (Y - X)\|_F^2$$
$$\text{subject to } \|Y\|_* \leq K$$

Is this problem convex? Why or why not? Rather than the problem above, we replace this with a penalized version,

$$\text{minimize } \|P \odot (Y - X)\|_F^2 + \lambda \|Y\|_*$$

because the projection onto the ℓ_1 ball requires a little bit of work to write (though it can be done exactly in $O(n \log(n))$, using a sort+O(n) operations to collect the result, and is faster if most of the entries are zero. This is even better for us since we're already computing an SVD so the singular values will come sorted.) This approach is known as soft-impute in the literature.

Answer to 6b: This is a convex problem because all norms are convex, including the nuclear norm, so adding this constraint to the minimizing problem would still be convex. The minimization problem is convex because it is quadratic, and its epigraph is set above the graph of the function.

(c) Derive formulas for (or procedures for computing) the following proximal and projection operators

$$\text{prox}_{t\|\cdot\|_*}(Y) = \arg\min_M \frac{1}{2t}\|Y - M\|_2^2 + \|M\|_*$$

$$\text{proj}_{\text{rank}_k}(Y) = \arg\min_{\text{rank}(M)\leq k} \frac{1}{2}\|Y - M\|_2^2$$

$$\text{proj}_{*K}(Y) = \arg\min_{\|Y\|_*\leq K} \frac{1}{2}\|Y - M\|_2^2$$

For the third one, you may write the answer in terms of an ℓ_1 ball projection

$$\text{proj}_{\ell_1\leq R}(x) = \arg\min_{\|y\|_1\leq R} \frac{1}{2}\|x - y\|_2^2$$

Answer to 6c: In order to calculate the nuclear norm $\text{prox}_{t\|\cdot\|_*}(Y) = \arg\min_M \frac{1}{2t}\|Y - M\|_2^2 + \|M\|_*$, we first apply the SVD on M to get the diagonal σ as shown below:

$$\text{prox}_{t\|\cdot\|_*}(Y) = \arg\min_M \frac{1}{2t}\|Y - M\|_2^2 + \|M\|_*$$
$$Y = U\Sigma V^T$$

to get the singular values of σ. Then apply the prox of the one norm on σ. Afterwards, we apply the singular value decomposition to get the prox of the nuclear norm:

$$Y = U\text{diag}(\text{prox}_{t\|\cdot\|_1}(\sigma))V^T$$

For the rank projection, we first apply the SVD on M to get the σ as shown below:

$$\text{proj}_{\text{rank}_k}(Y) = \arg\min_{\text{rank}(M)\leq k} \frac{1}{2}\|Y - M\|_2^2$$
$$Y = U\Sigma^* V^T$$

where Σ^* is the largest k σ elements when the $\text{rank}(M) \leq k$.

For the nuclear norm projection with a constraint is to first apply the SVD on M as shown below:

$$\text{proj}_{*K}(Y) = \arg\min_{\|Y\|_*\leq K} \frac{1}{2}\|Y - M\|_2^2 Y \quad = U\Sigma V^T$$

to get the singular values of σ. Then apply the prox of the one norm on σ. Afterwards, we apply the singular value decomposition to get the prox of the nuclear norm:

$$Y = U\text{diag}(\text{proj}_{\ell_1\leq R}(\sigma))V^T$$

(d) Suppose we want to solve the optimization problem in (b) for many values of λ to give ourselves the best chance of finding a good recovery (performing some kind of cross validation or scoring to pick the best one. In the coding section, we'll just compare the true reconstruction error $\|X - Y\|_F^2$ for simplicity in order to understand how changing the regularization parameter changes the performance of the model. This data wouldn't be available in practice since we only observe $P \odot X$ but there are ways to approximate that quantity. However, as you'll see, even for small examples, this problem can be somewhat time consuming. **What could we do to speed this up? Do you expect a larger value of λ to make the problem converge faster or slower?**

Answer to 6d: Instead of using a different lambda every time, we could try computing it using an iterative process where the λs are sorted. Then use the solution of the previous solution as the input to calculate the next solution. I would expect a larger value of λ to converge faster.

(e) Suppose that we rather than knowing the observed entries of X exactly, there were some large corrupted entries. **What could we change to make our problem robust to such corruptions?**

Answer to 6e: We could try to use different methods to promote more sparsity, such as adding another λ. Thus the minimization problem will become: minimize $|P \odot (Y - X)|_F + \lambda 1|Y| * + \lambda_2 |Y|_1$.

(f) Fill in Problem 6 in the jupyter notebook, the nuclear norm and rank projection in proxes.py
Answer to 6: Complete

There are a couple of other tricks to make this scalable to real world big data sets which we're not making you do in this class. The biggest speed up would come from using a "partial" or "truncated" SVD at each step since we only care about the top components, along the lines of the algorithm in https://arxiv.org/pdf/1607.03463.pdf. Additionally, because they apply an iterative algorithm, you can both warm start the SVD itself with the previous SVD, and stop computing additional components once you know they will be "proxed" down to zero (either with the nonconvex projection or with nuclear norm prox).

For **extra credit** implement the LazySVD algorithm in arxiv.org/pdf/1607.03463.pdf with a warm start for the nuclear norm regularized problem, and adaptively quit computing components when you are sure that the rest will be sent to zero by the prox operator! You may have to write a custom accelerated proximal gradient descent function for this, as you'll need to save the decompositions between iterations!

Ultimately, for the best performance (assuming very low rank and very few observed entries), you would want to avoid multiplying the matrices together, in the SVD, making each matrix vector product $O(kn + |\Omega|)$, where the kn comes from the two factor matrices, and $|\Omega|$ comes from the sparsity of the gradient (the gradient of the smooth part is zero except for where you observed the entries of X).