Homework # 1

NAME:_____

Signature:_____

STD. NUM: _____

1 Regularized linear regression

1.1 Data standardization

The goal of standardization is to make the d input attributes of the data comparable. If we were predicting the price or real estate in Vancouver using as input attributes the number of rooms and the price of the property 3 years ago ($d = 2$), then we would run into trouble because the number of rooms is typically in the range 1 to 4, while the price 3 years ago is probably in the range 500,000 to 5,000,000. We need to place these two attributes on the same scale so that applying the same regularizer to their weights makes sense.

To achieve this, we standardize the input data by ensuring it has zero mean and unit variance. We can do this by computing

$$x_{ij} \;=\; \frac{x_{ij} - \overline{x}_j}{\sigma_j}, \tag{1}$$

where $\overline{x}_j = \frac{1}{n} \sum_{i=1}^{n} x_{ij}$ is the empirical mean of the j'th attribute, and $\sigma_j^2 = \frac{1}{n} \sum_{i=1}^{n} (x_{ij} - \overline{x}_j)^2$ is the empirical variance. Here, $i = 1, \ldots, n$ denotes the index over the data and $j = 1, \ldots, d$ is the index over input attributes.

The subtraction of the mean is not necessary for datasets that are already known to be zero mean, such as acoustic time series. Likewise, for gray-scale images we know that each attribute (pixel) has the same range of possible values, so we can skip the variance normalization. If the attributes do not have the same range of values, then the variance rescaling can prove to be very useful. For example if the first attribute is the number of people who entered a building, the second attribute the temperature of the building, and the third attribute the number of rooms in the building, then these three numbers are likely not to be in the same scale. Variance rescaling places them on a reasonable comparative scale.

The parameters $\overline{x}_j$ and σ_j^2 should be considered part of the model. So at test time, we should center with respect to $\overline{x}_j$ and scale with respect to σ_j, rather than centering and scaling the test set with respect to the mean and standard deviation of the test set. To see why, imagine the test set consisted of a single example; then scaling by the test set standard deviation, instead of the value derived from the training set, would clearly be wrong, since the standard deviation of a single data point is not defined. All this will be made more clear in the following section.

1.2 Load and standardize the data

Download the prostate cancer dataset from the course website. In this prostate cancer study 9 variables – including age, log weight, log cancer volume, etc. – were measured for 97 patients. We will now construct a model to predict the 9th variable a linear combination of the other 8. A description of this dataset appears in the textbook of Hastie et al, freely available on the course website:

"*The data for this example come from a study by Stamey et al. (1989) that examined the correlation between the level of prostate specific antigen (PSA) and a number of clinical measures, in 97 men who were about to receive a radical prostatectomy. The goal is to predict the log of PSA (lpsa) from a number of measurements including log cancer volume (lcavol), log prostate weight lweight, age, log of benign prostatic*

hyperplasia amount lbph, seminal vesicle invasion svi, log of capsular penetration lcp, Gleason score gleason, and percent of Gleason scores 4 or 5 pgg45. "

1. First load the data and split it into a response vector (y) and a matrix of attributes (X),

```
X = np.loadtxt('prostate.data', skiprows=1)
y = X[:,-1]
X = X[:,0:-1]
```

2. Choose the first 50 patients as the training data. The remaining patients will be the test data.

```
ytrain, ytest = y[0:50], y[50:]
Xtrain, Xtest = X[0:50], X[50:]
```

3. Set both variables to have zero mean and standardize the input variables to have unit variance.

```
Xbar = np.mean(Xtrain, axis=0)
Xstd = np.std(Xtrain, axis=0)
ybar = np.mean(ytrain)
ytrain = ytrain - ybar
Xtrain = (Xtrain - Xbar) / Xstd
```

Note: here we are using the "broadcasting" feature of *numpy*, which allows summation of a vector and a matrix. For example:

```
>>> a
array([[1, 2, 3],
       [4, 5, 6]])
>>> c
array([7, 8, 9])
>>> a+c
array([[ 8, 10, 12],
       [11, 13, 15]])
```

You will have to be careful when you do this in *numpy* in order to make the shapes match up as they should. For more details on how to do this properly look up "*numpy* broadcasting".

Important detail: In the training step, we will learn the bias θ_0 separately. We do this because it makes not sense to apply regularization to the bias θ_0. Recall that the purpose of regularization is to get rid of unwanted input attributes.

We will need the terms (Xbar, Xstd, ybar) when we do predictions. Mathematically, what we are saying is that the bias term will be computed separately as follows:

$$\theta_0 = \bar{y} - \bar{\mathbf{x}}^T \widehat{\boldsymbol{\theta}}$$

where $\bar{y}$ is the mean of the elements of the **training data** vector $\mathbf{y}$ and $\bar{\mathbf{x}}^T$ is the vector of 8 means for the input attributes. Note that in this case the 8-dimensional parameter vector $\widehat{\boldsymbol{\theta}}$ includes all the parameters other than the bias term that have been learned with ridge regression. That is, we first learn $\widehat{\boldsymbol{\theta}}$ using standardized data and then proceed to learn θ_0.

When we encounter a new input $\mathbf{x}^\star$ in the test set, we need to standardize it before making a prediction. The actual prediction should be:

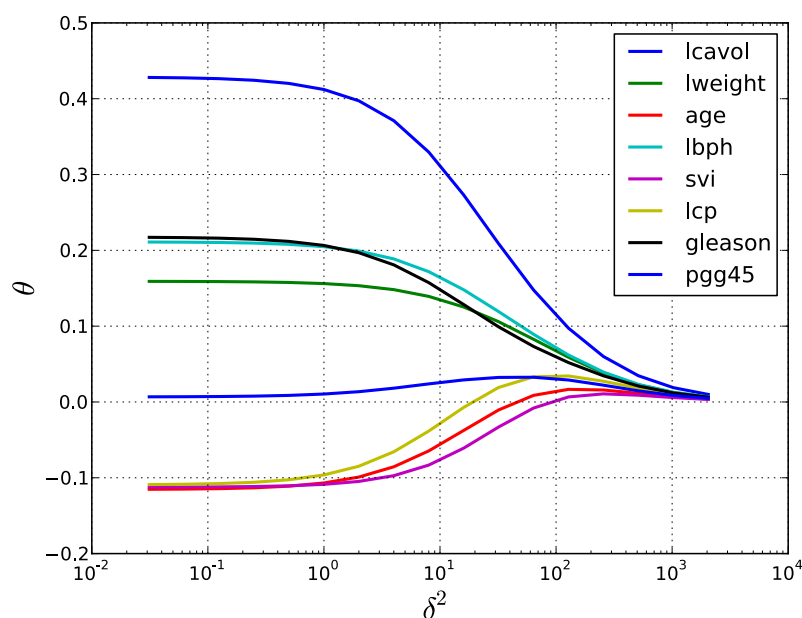$$\widehat{y} = \bar{y} + \sum_{j=1}^{8} \frac{x_j^\star - \bar{x}_j}{\sigma_j} \widehat{\theta}_j$$

Figure 1: Regularization path for ridge regression.

where $\bar{x}_j$ and σ_j are the mean and standard deviation of the j-th attribute **obtained from the training data.**

One reason for standardizing the inputs is that we want them to be comparable. Had we had an input much bigger than the other, we would have wanted to apply a different regularizer to it. By standardizing the inputs first, we only need a single scalar regularization coefficient δ^2.

1.3 Ridge regression

We will now construct a model using ridge regression to predict the 9th variable as a linear combination of the other 8.

1. Write code for ridge regression starting from the following skeleton:

```
def ridge(X, y, d2):
    ???
    return theta
```

Compute the ridge regression solutions for a range of regularizers (δ^2). Plot the values of each $\boldsymbol{\theta}$ in the y-axis against δ^2 in the x-axis. This set of plotted values is known as a regularization path. Your plot should look like Figure 1. *Hand in your version of this plot, along with the code you used to generate it.*

2. For each computed value of $\boldsymbol{\theta}$, compute the train and test error. Remember, you will have to standardize your test data with the same means and standard deviations as above (`Xbar, Xstd, ybar`) before you can make a prediction and compute your test error. In other words, to make a prediction do:

```
yhat = ybar + numpy.dot((Xtest - Xbar) / Xstd, theta)
```
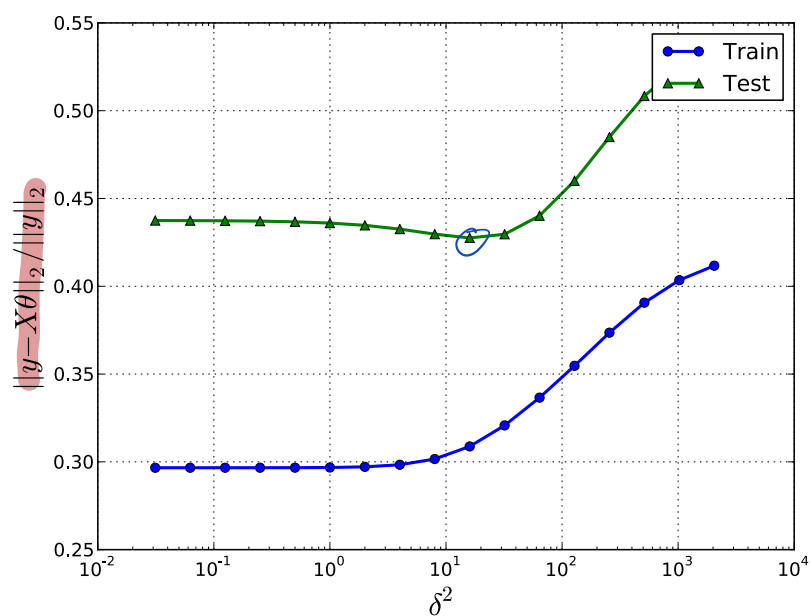
Figure 2: Relative error of the ridge estimator against regularization parameter δ^2.

Choose a value of δ^2 using cross-validation. What is this value? Show all your intermediate cross-validation steps and the criterion you used to choose δ^2. Plot the train and test errors as a function of δ^2. Your plot should look like Figure 2. *Hand in your version of this plot, along with the code used to generate it.*

2   Bayesian Linear Regression

Let $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\mathbf{y} \in \mathbb{R}^{n \times 1}$. Assume the likelihood is Gaussian:

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}, \boldsymbol{\Sigma}) = |2\pi\boldsymbol{\Sigma}|^{-1/2} e^{-\frac{1}{2}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})^T \boldsymbol{\Sigma}^{-1}(\mathbf{y}-\mathbf{X}\boldsymbol{\theta})}.$$

$\theta \in R^{d \times 1}$

$\Sigma \in R^{n \times n}$

Assume that the prior for $\boldsymbol{\theta}$ is also Gaussian:

$$p(\boldsymbol{\theta}) = |2\pi\boldsymbol{\Delta}|^{-1/2} e^{-\frac{1}{2}\boldsymbol{\theta}^T \boldsymbol{\Delta}^{-1} \boldsymbol{\theta}}.$$

$\Delta \in R^{d \times d}$

1. Using Bayes rule and completing squares, derive an expression for the posterior distribution of $\boldsymbol{\theta}$. In this part, assume that the covariance $\boldsymbol{\Sigma}$ is given. State clearly what the mean and variance of the posterior are. Also, state the conditions under which the posterior mean would be equivalent to the ridge and maximum likelihood estimators.

We know when prior and likelihood are Gaussians, the posterior is also Gaussian.

$$P(\theta|X,y,\Sigma) = \frac{P(y|X,\theta,\Sigma)\,P(\theta)}{P(y|X,\Sigma)} \propto P(y|X,\theta,\Sigma)\cdot P(\theta)$$

($P(y|X,\Sigma)$ is normalizing constant w.r.t. θ)

$$P(y|X,\theta,\Sigma)\,P(\theta)$$

$$= |2\pi\Sigma|^{-\frac{1}{2}}|2\pi\Delta|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}\left[(y-X\theta)^T\Sigma^{-1}(y-X\theta)+\theta^T\Delta^{-1}\theta\right]\right\} \quad \left(\text{Note } \theta^TX^T\Sigma^{-1}y = y^T\Sigma^{-1}X\theta\right.$$
$$\text{because it is a number.}\left.\right)$$

$$= |2\pi\Sigma|^{-\frac{1}{2}}|2\pi\Delta|^{-\frac{1}{2}}\exp\left\{-\frac{1}{2}\left[y^T\Sigma^{-1}y - 2\theta^TX^T\Sigma^{-1}y + \theta^T(X^T\Sigma^{-1}X + \Delta^{-1})\theta\right]\right\}$$

We want the exponent in the form of $-\frac{1}{2}(\theta-\theta_n)^T V_n^{-1}(\theta-\theta_n)$

Therefore the variance V_n and mean θ_n of the posterior are:

$$V_n^{-1} = X^T\Sigma^{-1}X + \Delta^{-1}$$

$$\theta_n = V_n X^T\Sigma^{-1}y = \left(X^T\Sigma^{-1}X + \Delta^{-1}\right)^{-1}X^T\Sigma^{-1}y.$$

when $\underline{\Sigma^{-1}$ is identity matrix $I}$, the mean θ_n becomes:

$$\theta_n = (X^TX + \Delta^{-1})^{-1}X^Ty \qquad \text{it is ridge regression when } \Delta^{-1} = \sigma^2 I$$
$$(\sigma^2 \in R)$$

when $\Delta^{-1} = 0$, i.e. the prior reduces to a vector

$$\theta_n = (X^TX)^{-1}X^Ty \qquad \text{this is the maximum likelihood estimator.}$$

2. Derive an analytical, closed-form expression for the maximum likelihood estimate of Σ. You will need the following properties of matrix derivatives (for a symmetric matrix A):

$$\frac{\partial \log |A|}{\partial A} = A^{-1} \qquad and \qquad \frac{\partial trace(x^T A x)}{\partial A} = xx^T \qquad trace(x^T A x) \in \mathbb{R}$$

Hint: Take derivatives of the log-likelihood with respect to Σ^{-1}.

$\nabla_{\Sigma^{-1}} \log P(y|X, \theta, \Sigma)$

$= \nabla_{\Sigma^{-1}} \left[\frac{1}{2} \log |\Sigma^{-1}| - \frac{1}{2}(y-X\theta)^T \Sigma^{-1}(y-X\theta) \right]$ drop constant, and $|\Sigma|^{-1} = |\Sigma^{-1}|$, $(y-X\theta)^T \Sigma^{-1}(y-X\theta) \in \mathbb{R}$

$= \frac{1}{2}\Sigma - \frac{1}{2}(y-X\theta)(y-X\theta)^T$

Set to zero, we have the maximum likelihood estimate of Σ:

$\Sigma = (y-X\theta)(y-X\theta)^T$

3. Given an *inverse Wishart prior* (Wikipedia has a nice description of the inverse Wishart distribution!) on Σ, with fixed known parameters α and Σ^*,

$$p(\Sigma|\alpha, \Sigma^*) \propto |\Sigma|^{-(\alpha+n+1)/2} e^{-\frac{1}{2}trace(\Sigma^*\Sigma^{-1})},$$

show by conjugate analyisis that the posterior distribution of Σ is also an inverse Wishart distribution. What are the new parameters of the posterior? **Hint:** $trace(z^T A z) = trace(zz^T A)$. Note that $\mathbb{E}(\Sigma) = \frac{\Sigma^*}{\alpha-n-1}$. Hence, under which conditions are the maximum likelihood and posterior mean estimates equivalent?

$P(\Sigma|X, y, \theta, \alpha, \Sigma^*) = \dfrac{P(y|X, \theta, \Sigma) P(\Sigma|\alpha, \Sigma^*)}{P(y|X, \theta, \alpha, \Sigma^*)}$

$P(y|X, \theta, \alpha, \Sigma^*)$ is constant with respect to Σ. The posterior of Σ:

$P(\Sigma|X, y, \theta, \alpha, \Sigma^*) \propto P(y|X, \theta, \Sigma) P(\Sigma|\alpha, \Sigma^*)$

$P(y|X, \theta, \Sigma) P(\Sigma|\alpha, \Sigma^*)$

$\propto |2\pi\Sigma|^{-\frac{1}{2}} |\Sigma|^{-(\alpha+n+1)/2} \exp\left\{-\frac{1}{2}\left[(y-X\theta)^T \Sigma^{-1}(y-X\theta) + trace(\Sigma^*\Sigma^{-1})\right]\right\}$

Since $|2\pi\Sigma|^{-\frac{1}{2}} = (2\pi)^{-\frac{n}{2}} |\Sigma|^{-\frac{1}{2}}$,

and $(y-X\theta)^T \Sigma^{-1} (y-X\theta)$ is a number, Therefore

$$(y-X\theta)^T \Sigma^{-1} (y-X\theta) = tr\left[(y-X\theta)^T \Sigma^{-1} (y-X\theta)\right]$$
$$= tr\left[(y-X\theta)(y-X\theta)^T \Sigma^{-1}\right]$$

We have

$$P(y|X,\theta,\Sigma) P(\Sigma|\alpha,\Sigma^*)$$
$$\propto 2\pi^{-\frac{n}{2}} |\Sigma|^{-\frac{(\alpha+n+2)}{2}} \exp\left\{-\frac{1}{2}tr\left[\left((y-X\theta)(y-X\theta)^T + \Sigma^*\right)\Sigma^{-1}\right]\right\}$$

The new parameters of the posterior inverse Wishart are

$$\alpha_{posterior} = \alpha + 1$$

$$\Sigma^*_{posterior} = (y-X\theta)(y-X\theta)^T + \Sigma^*.$$

$$E\left(\Sigma_{posterior}\right) = \frac{(y-X\theta)(y-X\theta)^T + \Sigma^*}{\alpha - n}$$

Σ^* and α are known constant.

From the last question, $\Sigma_{ML} = (y-X\theta)(y-X\theta)^T$

when $\alpha - n = 1$, and $\Sigma^* = 0$

$$E\left(\Sigma_{posterior}\right) = \Sigma_{ML}.$$

3 Dual Form of Ridge Regression

The ridge method is a regularized version of least squares

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \|\mathbf{y} - \mathbf{X}\boldsymbol{\theta}\|_2^2 + \delta^2 \|\boldsymbol{\theta}\|_2^2$$

where the input matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the output vector $\mathbf{y} \in \mathbb{R}^n$. The solution is obtained by differentiating the above cost function:

$$(\mathbf{X}^T \mathbf{X} + \delta^2 \mathbf{I}_d)\boldsymbol{\theta} = \mathbf{X}^T \mathbf{y}$$

where $\mathbf{I}_d$ is the $d \times d$ identity matrix and δ is a scalar.

1. Show that the solution can be written as $\boldsymbol{\theta} = \mathbf{X}^T \boldsymbol{\alpha}$, where $\boldsymbol{\alpha} = \delta^{-2}(\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$.

$$(X^T X + \delta^2 I_d)\theta = X^T y$$

$$\Rightarrow \quad X^T X \theta + \delta^2 \theta = X^T y.$$

$$\Rightarrow \quad \delta^2 \theta = X^T(y - X\theta)$$

$$\Rightarrow \quad \theta = X^T \frac{y - X\theta}{\delta^2} = X^T \alpha.$$

where $\quad \alpha = \frac{y - X\theta}{\delta^2}$

2. Show that $\boldsymbol{\alpha}$ can also be written as follows: $\boldsymbol{\alpha} = (\mathbf{X}\mathbf{X}^T + \delta^2 \mathbf{I}_n)^{-1}\mathbf{y}$. (This is an awesome trick because if $n = 20$ patients with $d = 10,000$ gene measurements, the computation of $\boldsymbol{\alpha}$ only requires inverting the $n \times n$ matrix, while the direct computation of $\boldsymbol{\theta}$ would have required the inversion of a $d \times d$ matrix.)

We know $\alpha = \frac{y - X\theta}{\delta^2}$ and want to remove θ from the expression.

replace θ with $(X^T X + \delta^2 I)^{-1} X^T y$, we have:

$$\alpha = \frac{1}{\delta^2}\left[I - X(X^T X + \delta^2 I)^{-1} X^T \right] y$$

$$= \frac{1}{\delta^2}\left[I - (XX^T + \delta^2 I)^{-1} XX^T \right] y \qquad \left(by \ (BA + \lambda I)^{-1} B = B(AB + \lambda I)^{-1} \right)$$

$$= \frac{1}{\delta^2}\left[(XX^T + \delta^2 I)^{-1}(XX^T + \delta^2 I - XX^T) \right] y$$

$$= (XX^T + \delta^2 I)^{-1} y.$$

$$\theta = (X^T X + \delta^2 I_d)^{-1} X^T y \qquad \text{use when } n > d$$

$$= X^T(XX^T + \delta^2 I_n)^{-1} y. \qquad \text{use when } d > n.$$

4 Collaborative Filtering for Movie Recommendation

This exercise is based on the Netflix Competition. The goal is to recommend movies to users based on their votes. Here we will derive a key technique by the guys who won the million dollar prize. Note that although we are focusing on movies here, we could use this technique for recommending any items in a social network. Finally, we don't get to implement the technique yet in this homework, but we will do so in the next homework. The actual reference is:

Yifan Hu, Yehuda Koren and Chris Volinsky. **Collaborative Filtering for Implicit Feedback Datasets**. *IEEE International Conference on Data Mining*, pages 263–272, 2008.

We assume that user u has indicated preference for item i via the variable $p_{ui} \in \{0, 1\}$, where $u = 1, \ldots, m$ and $i = 1, \ldots, n$. For example, a 1 could mean that the user gave the item a thumbs up. These votes are part of a matrix $\mathbf{P} \in \mathbb{R}^{m \times n}$ that has many *missing entries*. (The machine learning method here will take care of filling in the missing entries!).

Yifan Hu and collaborators applied this technique to movies, where users provide ratings r_{ui} in the form of stars. They convert these ratings to our preference variables as follows:

$$p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0, \end{cases}$$

and r_{ui} is the vote of user u for item i.

We also assume that we have an additional confidence signal c_{ui} that measures the extent of the interaction (attraction, etc.) of user u to item i. For example if user u gives a movie i a thumbs up and finishes watching it; or if the user is an expert in the genre of the movie, we would use a large value of c_{ui}. Yifan Hu use c_{ui} as the "confidence" of user u on item i:

$$c_{ui} = 1 + \alpha r_{ui},$$

where α is a parameter, which is set to 40 in the original paper.

Given n movies, our first objective will be to learn a matrix of *factors* $\mathbf{Y} \in \mathbb{R}^{f \times n}$ for these movies. That is, each movie will be described by a column vector $\mathbf{y}_i \in \mathbb{R}^{f \times 1}$ of f factors (features). Our second objective will be to learn a matrix of factors for the m users of the social network $\mathbf{X} \in \mathbb{R}^{m \times f}$. Each user will be described by a row vector of factors $\mathbf{x}_u \in \mathbb{R}^{1 \times f}$.

Since we have two unknowns, the method of solution will be alternating ridge regression. That is, we first fix $\mathbf{Y}$ and solve for $\mathbf{X}$, use this estimate of $\mathbf{X}$ and solve for $\mathbf{Y}$, use the latest estimate of $\mathbf{Y}$ and solve for $\mathbf{X}$ again, etc.

To compute the factors for each user, we assume the $\mathbf{y}_i$ are given, and proceed to minimize the following quadratic cost function:

$$J(\mathbf{x}_u) = \sum_{i=1}^{n} c_{ui}(p_{ui} - \mathbf{x}_u \mathbf{y}_i)^2 + \lambda \|\mathbf{x}_u\|_2^2 \qquad \text{for each user } u.$$

This is known as a weighted ridge regression problem.

1. For each user, assume we construct the diagonal matrix $\mathbf{C}_u \in \mathbb{R}^{n \times n}$ with diagonal entries c_{ui}. Let $\mathbf{p}_u \in \mathbb{R}^{1 \times n}$ denote the vector of preferences for user u. Show that the above weighted ridge regression objective can be re-written in matrix format as follows:

$$J(\mathbf{x}_u) \quad = \quad (\mathbf{p}_u - \mathbf{x}_u \mathbf{Y})\mathbf{C}_u(\mathbf{p}_u - \mathbf{x}_u \mathbf{Y})^T + \lambda \mathbf{x}_u \mathbf{x}_u^T$$

Since $x_u \in R^{1 \times f}$, $Y \in R^{f \times n}$, $y \in R^{f \times 1}$, we have

$$P_u - x_u Y = [P_{u_1} \ P_{u_2} \ \cdots \ P_{un}] - x_u [y_1 \ \cdots \ y_n] = [P_{u_1} - x_u y_1 \ \ P_{u_2} - x_u y_2 \ \cdots \ P_{un} - x_u y_n]$$
$$\in R^{1 \times n}$$

therefore

$$[P_{u_1} - x_u y_1 \ \cdots \ P_{un} - x_u y_n] \begin{bmatrix} c_{u_1} & & 0 \\ & c_{u_2} & \\ 0 & & \ddots \\ & & & c_{un} \end{bmatrix} \begin{bmatrix} P_{u_1} - x_u y_1 \\ \vdots \\ P_{un} - x_u y_n \end{bmatrix} = \sum_{i=1}^{n} c_{ui} (P_{ui} - x_u y_i)^2 \in R.$$

$$J(x_u) = (P_u - x_u Y) C_u (P_u - x_u Y)^T + \lambda x_u x_u^T = \sum_{i=1}^{n} c_{ui} (P_{ui} - x_u y_i)^2$$
$$+ \lambda \|x_u\|^2.$$

2. What is the solution $\widehat{\mathbf{x}}_u$ to the above objective?

From The Matrix Cookbook. we know,

$$\frac{\partial J(x_u)}{\partial x_u} = -2(P_u - x_u Y) C_u Y^T + 2\lambda x_u$$

set to zero:

$$x_u Y C_u Y^T + \lambda x_u = P_u C_u Y^T$$

we have

$$x_u = P_u C_u Y^T (Y C_u Y^T + \lambda I_f)^{-1} \quad (\in R^{1 \times f}).$$