STAT-221: Pset 2

Kevin Kuate Fodouop
Harvard University

Abstract

Large scale computation with clusters like Odyssey allows computation of Bayesian methods' frequency characteristics, like frequency coverage of Bayesian interval. In this problem set, we use MCMC samplling to evaluate such frequency properties for a non-conjugate hierarchical Bayesian model. A section is dedicated to each task (section number corresponds to task number).

1 Model and Posterior Distribution

We use the log-Normal/Poisson model, a non-conjugate hierarchical model. The data generating process for this model is:

$$\begin{aligned} p\left(\mu, \sigma^2\right) &\propto \frac{1}{\sigma^2} \\ log\left(\theta_j\right) &\sim N\left(\mu, \sigma^2\right), \quad j = 1, ..., J \\ Y_{jn} &\sim Pois\left(w_j / theta_j\right), \quad j = 1, ..., J; n = 1, ..., N \end{aligned}$$

- μ and σ^2 unknown constants, hyperparameters of the model.
- θ_j latent variables, basic intensity of unit j, drawn from a distribution common to each unit.
- w_j known constant, exposure weight for unit j.
- $Y_{j,n}$ random variables, set of N Poisson observations drawn for each unit.

We derive the joint posterior for μ, σ^2, log $\left(\vec{\theta}\right)$ for our model:

$$\begin{aligned} P\left[\mu, \sigma^2, log\left(\vec{\theta}\right)|Y\right] &\propto P\left[Y|\mu, \sigma^2, log\left(\vec{\theta}\right)\right] \times P\left[\mu, \sigma^2, log\left(\vec{\theta}\right)\right] \\ &= P\left[Y|\mu, \sigma^2, log\left(\vec{\theta}\right)\right] \times P\left[log\left(\vec{\theta}\right)|\mu, \sigma^2\right] \times P\left[\mu, \sigma^2\right] \\ &\propto \prod_{j=1}^{J}\prod_{n=1}^{N}\frac{(w_j\theta_j)^{y_{jn}}}{y_{jn}!}e^{-w_j\theta_j} \times \prod_{j=1}^{J}\frac{1}{\sigma}e^{-\frac{\left(log\left(\theta_j\right)-\mu\right)^2}{2\sigma^2}} \times \frac{1}{\sigma^2} \\ &= \frac{1}{\sigma^{J+2}}e^{-\sum_{j=1}^{J}\frac{\left(log\left(\theta_j\right)-\mu\right)^2}{2\sigma^2}}\prod_{j=1}^{J}\prod_{n=1}^{N}\frac{(w_j\theta_j)^{y_{jn}}}{y_{jn}!}e^{-w_j\theta_j} \\ &\propto \frac{1}{\sigma^{J+2}}e^{-\sum_{j=1}^{J}\frac{\left(log\left(\theta_j\right)-\mu\right)^2}{2\sigma^2}}\prod_{j=1}^{J}\prod_{n=1}^{N}\theta_j^{y_{jn}}e^{-w_j\theta_j} \qquad (1) \end{aligned}$$

And the the conditional posterior of $log\left(\theta_j\right)$ is given by ($\vec{w}$ is not included in the conditioning, as supposed known) normalizing (1)'s component corresponding to unit j by (μ, σ^2)'s posterior probability:

$$\begin{aligned} P\left[log\left(\theta_j\right)|\mu, \sigma^2, Y\right] &= \frac{P\left[\mu, \sigma^2, log\left(\theta_j\right)|Y\right]}{P\left[\mu, \sigma^2|Y\right]} \\ &\propto P\left[\mu, \sigma^2, log\left(\theta_j\right)|Y\right] \\ &\propto e^{-\frac{\left(log\left(\theta_j\right)-\mu\right)^2}{2\sigma^2}} \theta_j^{S_j} e^{-nw_j\theta_j} \end{aligned}$$

with $S_j = \sum_{n=1}^{N} y_{jn}$.
Denoting $z_j = log(\theta_j)$, the conditional posterior hence has a density

$$f_{post}(z_j) = e^{-\frac{\left(z_j-\mu\right)^2}{2\sigma^2}} e^{S_j z_j} e^{-nw_j e^{z_j}}$$

And a log-density

$$\begin{aligned} l_{post}(z_j) &= log \circ f_{post}(z_j) \\ &= -\frac{\left(z_j - \mu\right)^2}{2\sigma^2} + S_j z_j - nw_j e^{z_j} \end{aligned}$$

Which is concave as a sum of 3 concave (including 1 linear) functions. Hence **the conditional posterior is log-concave**.

Let's compute the derivatives of l_{post}:

$$\begin{aligned} \frac{\partial l_{post}}{\partial z_j} &= -\frac{z_j - \mu}{\sigma^2} + S_j - nw_j e^{z_j} \\ \frac{\partial^2 l_{post}}{\partial z_j^2} &= -\frac{1}{\sigma^2} - nw_j e^{z_j} \end{aligned}$$

$\frac{\partial^2 l_{post}}{\partial z_j^2} < 0$, so $\frac{\partial l_{post}}{\partial z_j}$ is decreasing. Having $\frac{\partial l_{post}}{\partial z_j} \overset{-\infty}{\to} +\infty$ and $\frac{\partial l_{post}}{\partial z_j} \overset{+\infty}{\to} -\infty$, $\frac{\partial l_{post}}{\partial z_j}$ has one and only zero. So that the **conditional posterior of $log\left(\theta_j\right)$ is unimodal**.

– To be removed – We compute the denominator's probability by integrating $log\left(\vec{\theta}\right)$ out of (1):

$$P\left[\mu, \sigma^2|Y\right] \propto P\left[Y|\mu, \sigma^2\right] \times P\left[\mu, \sigma^2\right]$$

We obtain the first term by integrating $log\left(\vec{\theta}\right)$ out of the likelihood:

$$\begin{aligned} P\left[Y|\mu, \sigma^2\right] &= \int_{-\infty}^{+\infty} P\left[Y|\mu, \sigma^2, log\left(\vec{\theta}\right)\right] P\left[log\left(\vec{\theta}\right)|\mu, \sigma^2,\right] \, dlog(\vec{\theta}) \\ &\propto \end{aligned}$$

– –

2 Linear Basis Function Regression

We tried replication Bishop 1.4 plots of MLE weight vectors for 4 values of M, using both closed form expressions and gradient descent. The former gets very close agreement (figure 1).

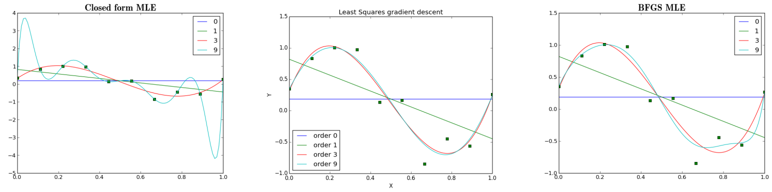

Figure 1: *MLE weight vector obtained using closed form expression (reproduces Bishop 1.4), using gradient descent and BFGS algorithm (optimize package), in order.*

The gradient descent and BFGS method uses the SSE corresponding to current weight vector as objective function to minimize. Both optimization on the SSE fails for too high orders. We think this is due to the optimized function having a plateau, from which the descent fails to progress towards the optimum.

3 Ridge Regression

3.1 Ridge regression for Bishop's data

Ridge regression regularizes coefficients for high polynomial orders, which were of in ranges up to 10^{15} for OLS. The cost function is $\text{cost}_{SSE} + \lambda \sum_{i>0} w_i^2$
Plot obtained are more smooth and less likely to overfit. $\lambda = 10^{-4}$ gives a pretty good fit for order superior to 3. λ of 10^{-2} or more regularizes the weights too much, so that the fit for all orders is too flat to accurately fit the data.

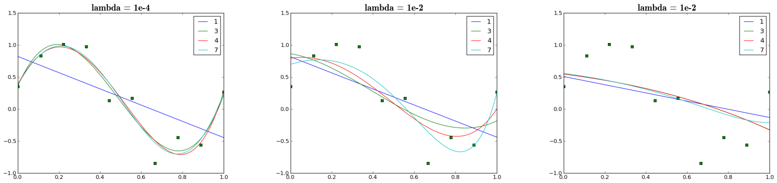

Figure 2: *Ridge regression fit for several values of M, for $\lambda \in [10^{-4}, 10^{-1}, 1]$ (in order).*

3.2 Model selection with ridge regression

We train ridge regressions for different M and λ on the two training sets A and B, $train_A$ and $train_B$. The *validation data* is split in a validation set and a test set. SSE of models on the validation set is used to choose the best model originating from $train_A$ or $train_B$. Validation errors are compiled in table 2.

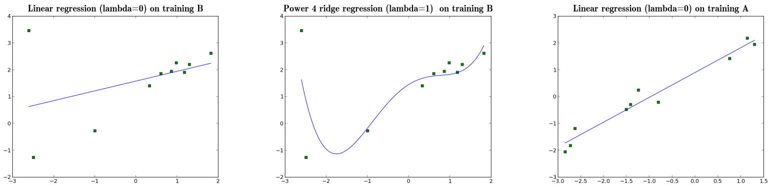

Figure 3: *$train_A$'s data has a very linear shape, whereas an outlier in $train_B$ prevents the linear model from capturing the right correlation in data.*

Table 1: *Some (M, λ) combinations validation errors. Minimum SSE for each training set in blue.*

Training set	M	λ	SSE_{val}
A	1	0	0.73
A	3	10^{-1}	1.6
A	7	10	88.7
B	1	0	16
B	3	10^{-4}	3.6
B	4	1	0.41

We fitted ridge regression with $\lambda = 10^{\pm i}, i = 0...4$ and $\lambda = 0$. The data in $train_B$ has a very linear form, so that no regularization is needed and regular linear regression is chosen. $train_B$ however has an outlier, which tampers the simple linear regression model (shifted up and lower slope). So that a well regularized ($\lambda = 1$) model of order 4 captures better the shape of the data, and obtains the best validation error.

When tried on the test set, the model issued from $train_A$ unsurprisingly performs better, with $SSE_{test}^{(A)} = 1.2$ compared to $SSE_{test}^{(B)} = 28$. Fit on test set plotted on figure 4.

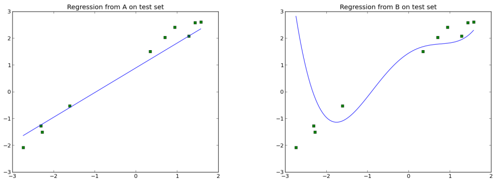

Figure 4: *Final models from $train_A$ and $train_B$'s fit on the test set.*

3.3 BlogFeedback data

We perform a ridge regression on the BlogFeedback data set. We note that the design matrix X is not full rank ($rank = 246$ for 280 features), so that $X^T X$ is not invertible (some features are redundant). In consequence we use a pseudo-inverse function to compute the ridge regression.

Table 2: *Evolution of training error and validation error with λ (optimal validation error in blue).*

λ	MSE_{train}	$MSE_{v}al$
0	881	4×10^6
10^{-3}	881	4×10^6
10^{-1}	881	4×10^6
10^2	882	7×10^5
10^6	895	5×10^5
10^{10}	1209	1.5×10^4
10^{14}	1372	3×10^5

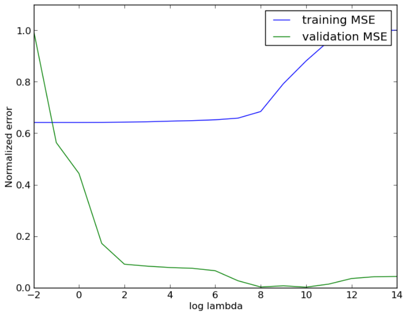

Figure 5: *Training and validation MSE plotted against $log_{10}(\lambda)$.*

The OLS model suffers from overfitting due to the high number of variables. A high regularization coefficient is needed to obtain good validation error, and we find an optimal $\lambda = 10^{10}$. To find optimal λ, we started from small values (10^{-4}), assessed that increasing it led to better validation error, and increased it by ten-fold until an increase in validation error. Corresponding training and validation error are plotted on figure 5.

The selected model has an error on the test set $MSE_{selected}^{(test)} = 15 \times 10^3$, compared to $MSE_{OLS}^{(test)} = 4.5 \times 10^6$. Ridge weights are typically of an order of magnitude around 10^{-6} the ones of non-regularized weights.

4 Generalizations

We limit this part's study to order 5. Lower orders fit better the data (less overfitting) and benefit less from the regularization methods.

4.1 Least absolute deviation

Still using $train_A$ and $train_B$, we fit a *least absolute deviations* (LAD) regression. We repeat our experiment of trying different regularizer weight λ, assess best fits on the validation error, and computes selected models fit on the test set. Sum of absolute errors (SAE) on the validation set are compiled for some λ in table 4.

Table 3: *Some (M, λ) combinations validation errors for LAD (order $= 5$). Minimum SAE for each training set in blue.*

Training set	λ	SAE_{val}
A	10^{-4}	5.8
A	10^{-1}	5.7
A	10^{3}	11
B	10^{-3}	24
B	10	14
B	10^{2}	16

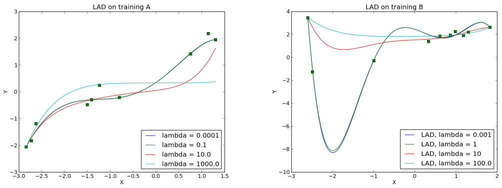

Figure 6: *Training fit for LAD (order $= 5$) with different λ for $train_A$ (left) and $train_B$ (right).*

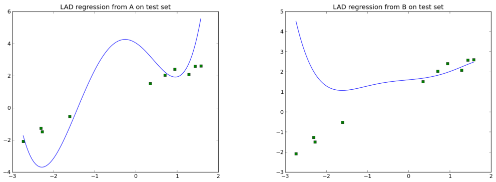

Figure 7: *Final LAD models (order $= 5$) from $train_A$ and $train_B$'s fit on the test set.*

Final test errors for both selected models are $SAE_{test}^{(A)} = 13$, $SAE_{test}^{(B)} = 16$. We can see on figure 7 that the model from $train_A$ has less extreme test errors, whereas the model from $train_B$ predicts very well a cluster of data while having catastrophic fit on another cluster (small x). However they have comparable test error, as LAD does not penalize for extreme errors (as OLS does), as is discussed in 3.

4.2 LASSO

We implement the LASSO regularization of OLS, using gradient descent. Same process as in 1. is replicated.

Table 4: *Some (M, λ) combinations validation errors for LASSO (order $= 5$). Minimum SSE for each training set in blue.*

Training set	λ	SSE_{val}
A	10^{-4}	5.8
A	1	3.4
A	10^{3}	24
B	10^{-2}	139
B	1	94
B	10^{4}	25

Due to its preference to models with few non-zero parameters, LASSO on $train_B$ drives all parameters close to 0 with $\lambda = 10^3$ or more. The selected model (minimum validation error), with $\lambda = 10^4$, has all its coefficients of order 10^{-4} or less, and is almost a flat line.

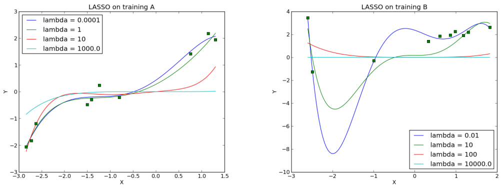

Figure 8: *Training fit for LASSO (order $= 5$) with different λ for $train_A$ (left) and $train_B$ (right).*

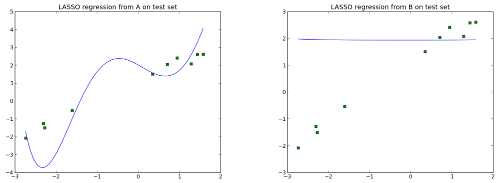

Figure 9: *Final LASSO models (order $= 5$) from $train_A$ and $train_B$'s fit on the test set.*

Final test errors for both selected models are $SSE_{test}^{(A)} = 6.02$, $SSE_{test}^{(B)} = 15.6$.

4.3 Difference of approach

LAD changes from ridge regression in the objective function we try to minimize, i.e. the error function we put on our fit and predictions. LAD is notably derived from using the absolute value loss function $L(p, a) = |p - a|$ in a prediction problem. Unlike least squares, this loss function does not penalize more heavily incremental regression error when the predicted value is far from the observed value (linear error). We might want to use LAD when we want every observation to be treated equally in the model fit (whereas LS gives greater weight to observations predicted poorly), to avoid strong effect from outliers. The training set $train_B$ gives a good example of a case where this is problematic for the linear fit (robust regression also provides a solution to this issue). LAD has notably been used in economics and biomedical studies.

Lasso has the advantage of bringing less significant parameters to 0, which is a useful for feature selection. This characteristic of Lasso to prefer fewer nonzero parameters make it a privileged solution for application like compressed sensing (signal processing).

A Figures