CS688: Graphical Models - Spring 2016

Assignment 4

Shamya Karumbaiah

Collaborated with Rafael Lizarralde

Bayesian linear regression. I have used the same naming convention as in the starter code in *linreg.py* in the answers too. Assuming its self explanatory.

(a) *function w = linreg_mle(X,Y)*
Implemented the linear regression MLE as,

$$w^{(MLE)} = (X'X)^{-1}X'Y$$

I verified the correctness with a bunch of datasets of different shapes and sizes. To illustrate in this and next question, I am using a simple intuitive example. Consider the following setting -

$$X = \begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \end{bmatrix}$$

and

$$Y = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

$w = linreg\_mle(X, Y)$ gives

$$w = \begin{bmatrix} 0.2000 \\ 0.4000 \end{bmatrix}$$

(b) *function [PosteriorMean, PosteriorCovar] = linreg_post(X,Y, priormean, priorvar_scalar, emissionvar_scalar)*

To verify the correctness, I did the following -
1. Using a prior of $w \sim N(0, big)$, yielded a posterior mean that is same as the MLE.
$[PosteriorMean, PosteriorCovar] = linreg\_post(X, Y, priormean, 1e100, 1)$, where $priormean$ is a vector of zeros - gave the result

$$PosteriorMean = \begin{bmatrix} 0.2000 \\ 0.4000 \end{bmatrix}$$

$$PosteriorCovar = \begin{bmatrix} 0.0029 & 0.0057 \\ 0.0057 & 0.0114 \end{bmatrix}$$

2. Varying the prior, I observed the posterior change in the following ways -
i) With a prior mean close to MLE, the value of prior variance didn't affect the posterior mean much as this gives a good estimate to start with. And the posterior mean was obviously close to MLE.
ii) With a prior mean proportional to MLE, the value of posterior mean was nearly same as the MLE even with a small prior variance. For example, consider

$$priormean = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

With $[PosteriorMean, PosteriorCovar] = linreg\_post(X, Y, priormean, 1, 1)$, we get,

$$PosteriorMean = \begin{bmatrix} 0.2113 \\ 0.4225 \end{bmatrix}$$

iii) When the prior covariance matrix was changed from being strictly diagonal to a less spread matrix, the posterior mean didn't match the MLE (unless if the prior variance was too big). Thus, only when the prior is diffuse, likelihood dominates the posterior.
For example, with a prior covariance matrix with all ones and prior mean as zeros-

$$PosteriorMean = \begin{bmatrix} -1.0000 \\ 1.0000 \end{bmatrix}$$

Whereas the same setting with a diagonal prior covariance matrix still gives a value close to MLE-

$$PosteriorMean = \begin{bmatrix} 0.1972 \\ 0.3944 \end{bmatrix}$$

(c) *function function [w_cdf] = cook_linreg(priormean, priorvar_scalar, emissionvar_scalar, Nsim)*

The below setting is used to call the $cook\_linreg$ function -
$priormean = [0, 0]$
$priorvar\_scalar = 1$
$emissionvar\_scalar = 1$
$Nsim = 10000$

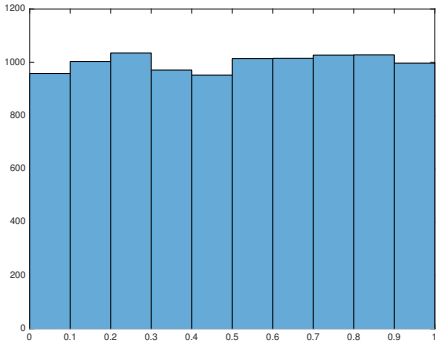
Using the Cook test for analyzing the correctness of the implementation, I followed design below-

Fixed prior (w0, v0) using the function parameters
For each simulation
    sample w from the prior N (w0 , sqrt(V0))
    sample each Y from N (w'X, emissionvar_scalar)
    PosteriorMean, PosteriorCovar = linreg_post(X,Y, priormean, priorvar_scalar, emissionvar_scalar)
    derive PosteriorVar from PosteriorCovar
    get the CDF of w
end for

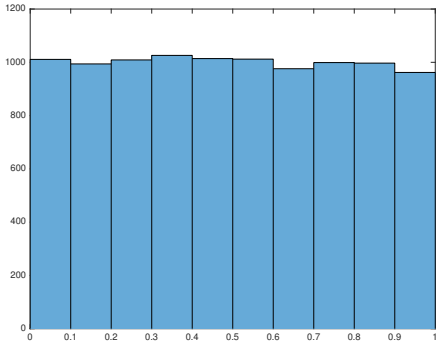
Note - V here refers to variance. Hence while sampling from normals, I have taken a square root.

The same toy data set as in the starter code is used -

$$X = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$$



(a) Weight 1

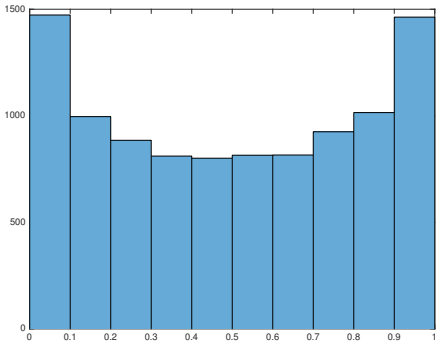


(b) Weight 2

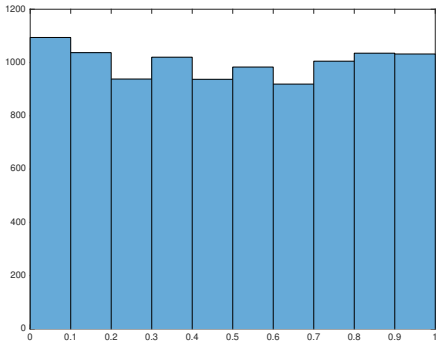
As we see in the figure, the posterior quantiles are very close to uniform. With a 10,000 simulations we could achieve a near to uniform distribution. Thus, according to Cook et all this suggests the correctness of the implementation. This follows their theorem - If the software is written correctly, for continuous θ, the distribution of posterior quantiles approaches Uniform(0, 1) as L tends to ∞.

I didn't have a bug in my code with my first implementation. So, it was really hard to think of a natural bug a programmer could have induced; especially with matlab when the matrix operations are crisp and clear. So, I added a pretty lame bug in the $linreg\_post$ function when calculating the $PosteriorCovar$ (posterior covariance) by adding a scalar one to the sum term of this equation.

This resulted in the below decile histogram visualization for the two weights. Weight 1 clearly demonstrate a shift from being uniform whereas weight 2 shows some wiggle. The bugginess was more clearer to visualize with a plot with 20 intervals.



(c) Weight 1



(d) Weight 2

Supreme Court latent space model.

(a) I have populated these to be relevant to this assignment, hence assuming constant positions. This table contains all the notations from the paper. Few of them arent use explicitly in the model.

Name	Description
K	Set of cases heard
J_k	Set of justices who heard case k (typically nine)
$v_{k,j}$	Justice j's vote on case k (0 for affirm, 1 for reverse)
$u_{k,j}^{(r)}$	Justice j's utility for voting to reverse (r) on case k
$u_{k,j}^{(a)}$	Justice j's utility for voting to affirm (a) on case k
$z_{k,j}$	utility difference between reversal and affirmation
θ_j	Justice j's ideal point (policy preference)
$x_k^{(a)}$	Location of the policy under an affirmance vote
$x_k^{(r)}$	Location of the policy under a reversal
$\xi_{k,j}^{(a)}$	Noise in the utility for justice j voting to affirm (a) on case k
$\xi_{k,j}^{(r)}$	Noise in the utility for justice j voting to reverse (r) on case k
α_k	Difference between squared position of the reverse and affirm for case k
β_k	Twice the difference between reversed and affirmed policies for case k
$\varepsilon_{k,j}$	Noise in the net utility difference for justice j on case k

(b) The model's assumed generative process that created the data is as below-

The priors,

$$\forall_k \alpha_k \sim \mathcal{N}(0,1), \beta_k \sim \mathcal{N}(0,1)$$
$$\forall_j \theta_j \sim \mathcal{N}(0,1)$$

The posteriors,

$$P(\alpha_k, \beta_k | \theta_j, z_{k,j})$$
$$P(\theta_j | \alpha_k, \beta_k, z_{k,j})$$
$$P(z_{k,j} | \alpha_k, \beta_k, \theta_j, v_{k,j})$$

The inference,

For each case $k \in K$,

For each justice $j \in J_k$,

$P(z_{k,j} | \alpha_k, \beta_k, \theta_j) = \mathcal{N}(\alpha_k + \beta_k \theta_j + \varepsilon_{k,j}, 1) \qquad \varepsilon_{k,j} \sim_{iid} \mathcal{N}(0,1)$

$P(v_{k,j} | z_{k,j}) = 1\{z_{k,j} > 0\}$

(c) Below are the gibb's sampling steps -

For each gibb's sampling iteration,

update $z_{k,j}$	resample_Z.m
update θ_j	resample_theta.m
update α_k, β_k	resample_alphabeta.m

function Z = resample_Z(Y, alpha, beta, theta)

For each $z_{k,j}$,

$P(z_{k,j}|\alpha_k, \beta_k, \theta_j) = \mathcal{N}(\alpha_k + \beta_k\theta_j + \varepsilon_{k,j}, 1)$

This is a truncated normal to be specific on a constraint, given by -

$$P(z_{k,j}|\alpha_k, \beta_k, \theta_j) = \begin{cases} \mathcal{N}_{[0,\infty]}(\alpha_k + \beta_k\theta_j, 1) & \text{if } v_{k,j} = 1 \\ \mathcal{N}_{[-\infty,0]}(\alpha_k + \beta_k\theta_j, 1) & \text{if } v_{k,j} = 0 \\ \mathcal{N}(\alpha_k + \beta_k\theta_j, 1) & \text{if } v_{k,j} \text{is missing} \end{cases}$$

function theta = resample_theta(Z, alpha, beta)

For each θ_j,

$[mean, covar] = linreg\_post(beta, z - alpha, 0, 1, 1)$

$\theta_k = \mathcal{N}(mean, covar)$

function [alpha, beta] = resample_alphabeta(Y, Z, theta)

For each α_k and β_k,

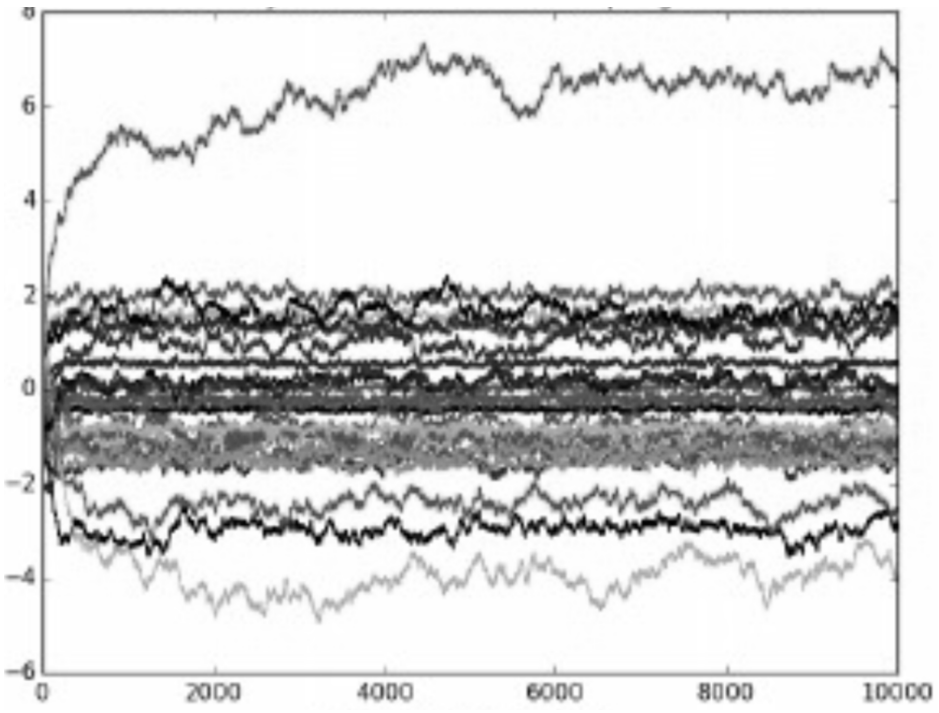
$[mean, covar] = linreg\_post(\theta_j, z_{k,j}^{\top}, b0, B0, 1)$

$\alpha_k, \beta_k = \mathcal{N}(mean, covar)$

where, a bias column is added to θ_j. The weights for θ_j is β_k, and the weights on the bias column is α_k.

(d) *Question2.m* Nothing to report.

(e) Running just 1000 iterations wasn't enough to see the convergence especially for justice Douglas (top line in plot). So, I ran for 10000 iterations with first 2000 steps as burn-in. Most justices seem to converge in less than $1,000$ iterations except the ones in extremes. After convergence, the wiggling is lot reduced, though there is still some.



(f) A summary of the θ_j parameters: for each justice, their posterior mean and standard deviations from samples after burn-in -

Justice	Post. mean	Post. SD	Justice	Post. mean	Post. SD
Harlan	1.529	0.123	Minton	0.824	0.252
Black	-1.267	0.094	Jackson	0.845	0.274
Douglas	- 6.506	0.326	Burger	1.549	0.096
Stewart	0.372	0.053	Blackmun	-0.032	0.052
Marshall	-2.018	0.118	Powell	0.823	0.066
Brennan	-1.609	0.077	Rehnquist	2.946	0.172
White	0.423	0.046	Stevens	-0.552	0.059
Warren	-1.389	0.089	O'Connor	1.317	0.083
Clark	0.198	0.074	Scalia	2.395	0.201
Frankfurter	1.047	0.121	Kennedy	1.329	0.113
Whittaker	1.042	0.139	Souter	0.253	0.087
Burton	1.205	0.188	Thomas	3.972	0.352
Reed	1.088	0.187	Ginsburg	-0.178	0.114
Fortas	-1.664	0.211	Breyer	-0.193	0.136
Goldberg	-1.008	0.178			

(g) Douglas is the most liberal justice and Thomas is the most conservative according to the model. Wikipedia agrees with both of this. Other than these two extremes also, the model and wikipedia seems to agree with most justices which Blackmun being a moderate liberal.