Bayesian Linear Regression

Edps 590BAY

Carolyn J. Anderson

Department of Educational Psychology
ILLINOIS

Fall 2019

I Overview

- Generalized linear models
- Bayesian Simple linear regression
  - Numeric predictor (nels data, 1 school)
  - Robust regression
  - Categorical predictors (anorexia data)
- Model evaluation

Depending on the book that you select for this course, read either Gelman et al. on specific topics (I skipped around a bit) or Kruschke Chapters chapters 13, 15 & 16 . Also I used the coda and jags, rjags, runjags and jagsUI manuals.

Regression models

- All regression models focus on model the expected value of a response or outcome variable(s) as a function of other variable(s).
- The type of regression model depends on the nature of the response variable, e.g.,
  - A numerical response ("continuous") → linear regression
  - A dichotomous response → logistic regression
  - A count response → Poisson or negative binomial regression
  - A numerical response bounded from below by 0 → Gamma, Inverse Gaussian, or log-normal regression
  - Responses nested within groups or clustered observations → "hierarchical" or random effects regression
- The above are all common examples (except the last one) of generalized linear models.

Components of a GLM

There are 3 components of a generalized linear model (or GLM):

1. **Random Component** — identify the response variable (Y) and specify/assume a probability distribution for it.
2. **Systematic Component** — specify what the explanatory or predictor variables are (e.g., X_1, X_2, etc). These variable enter in a linear manner

$$b_0 + b_1X_1 + b_2X_2 + \ldots + b_kX_k$$

3. **Link** — Specify the relationship between the mean or expected value of the random component (i.e., $E(Y)$) and the systematic component.

Random Component

The distribution of the response variable is a member of the exponential (dispersion) family of distributions.

A short list of members of exponential family(or closely related):

Normal	Student-t distribution
Binomial	Multinomial
Poisson	Negative binomial
Beta	exponential

Systematic Component: Linear Predictor

This restriction to a linear predictor is not all that restrictive.

For example,

- $x_3 = x_1 x_2$ — an "interaction".
- $x_1 \Rightarrow x_1^2$ — a "curvilinear" relationship.
- $x_2 \Rightarrow \log(x_2)$ — a "curvilinear" relationship.

$$b_0 + b_1 x_1^2 + b_2 \log(x_2) + b_3 x_1^2 \log(x_2)$$

This part of the model is very much like what you know with respect to ordinary linear regression.
The xs may be numeric, ordinal, nominal or some combination.
For nominal, we can use dummy or effect coding.

The Link Function

"Left hand" side of an equation/model — the random component,	"Right hand" side of the equation— the systematic component; that is,
$E(Y) = \mu$	$b_0 + b_1x_1 + b_2x_2 + \ldots + b_kx_k$

We now need to "link" the two sides.

How is $\mu = E(Y)$ related to $b_0 + b_1x_1 + b_2x_2 + \ldots + b_kx_k$?
We do this using a "Link Function" $\Longrightarrow g(\mu)$

$$g(\mu) = b_0 + b_1x_1 + b_2x_2 + \ldots + b_kx_k$$

Simple Linear Regression as a GLM

$$Y_i = b_0 + b_1 x_i + \epsilon_i$$

- Random Component: y is the response normally distributed and typically assume $\epsilon_i \sim N(0, \sigma^2)$, os $y_i \sim N(\mu, \sigma^2)$.
- Systematic Component: Let x_i be some predictor or explanatory variable, then the systematic is linear and

$$b_0 + b_1 x_i$$

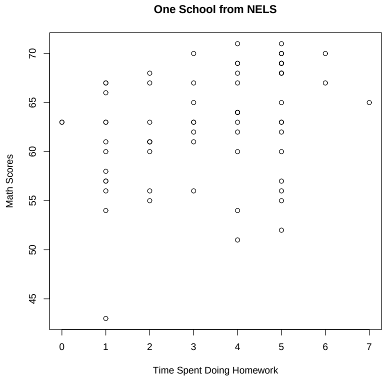
- Link Function: Is the identity link

$$g(E(Y_i)) = g(\mu|x_i) = E(Y_i) = b_0 + b_1 x_i$$

Note: The link is on the mean or expected value of Y_i.

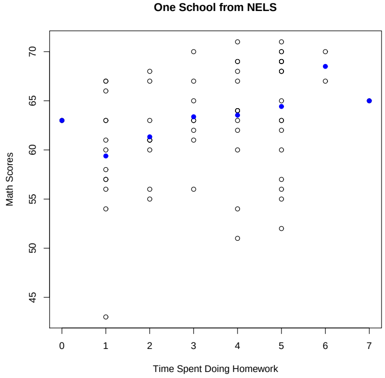
Simple Linear Regression as a GLM

Example: One school from the NELS, school id 62821 math scores by time spent doing homework:

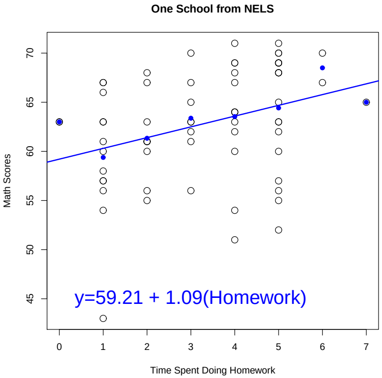


With the Means

Note that what we are really doing is fitting a linear function the means.



With the Means and Regression



Results of OLS for NELS

```
ols.lm <- lm(math~homework,data=nels)
summary(ols.lm)
```

Residuals:	Min	1Q	Median	3Q	Max
	-17.3049	-2.4941	0.4112	4.3166	7.5059

Coefficients:

	Estimate	Std. Error	t value	$Pr(> \|t\|)$
(Intercept)	59.2102	1.4314	41.366	$<$ 2e-16 ***
homework	1.0946	0.3852	2.842	0.00599 **

—

Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1

Residual standard error: 5.394 on 65 degrees of freedom
Multiple R-squared: 0.1105, Adjusted R-squared: 0.09681
F-statistic: 8.074 on 1 and 65 DF, p-value: 0.005991

Bayesian Linear Regression

Goal: Find the posterior distribution of the parameters of the regression model; namely, b_0, b_1 and σ^2.

$$p(b_0, b_1, \sigma^2|y_1, \ldots, y_n) \propto p(y_1, \ldots, y_n|b_0, b_1, \sigma^2)p(b_0, b_1, \sigma^2)$$

We'll use Gibbs sampling and set

- Likelihood or the data model:

$$p(y_1, \ldots, y_n|b_0, b_1, \sigma^2) \sim \prod_{i=1}^{n} N(\mu_i|b_0, b_1, \sigma^2)$$

  where $\mu_i = \mu|x_i = b_0 + b_1 x_i$.
- Diverse Priors
  - $b_0 \sim N(0, 1/(100 \times sd(y)^2))$ where $1/(100 \times sd(y)^2)$ is precision
  - $b_1 \sim N(0, 1/(100 \times sd(y)^2))$ where $1/(100 \times sd(y)^2)$ is precision
  - $\sigma^2 \sim \text{Uniform}(1E-3, 1E+30)$

rjags: dataList

```
set.seed(75)

dataList ← list(y=nels$math,
        x=nels$homework,
        N=length(nels$math),
        sdY = sd(nels$math)
        )
```

rjags: Model

```
nelsLR1 = ``model {
     for (i in 1:N){
     y[i] ~ dnorm(mu[i] , precision)
     mu[i] <- b0 + b1*x[i]
    }
    b0 ~ dnorm(0 , 1/(100*sdY^2) )
    b1 ~ dnorm(0 , 1/(100*sdY^2) )
    precision <- 1/sigma^2
    sigma ~ dunif( 1E-3, 1E+30 )
    }''

writeLines(nelsLR1, con=``nelsLR1.txt'')
```

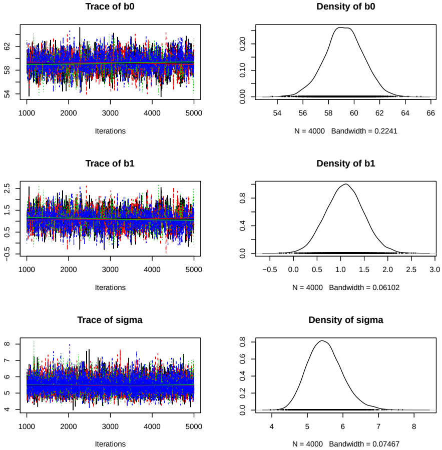
rjags: Compile & Initalize

```
b0Init → mean(nels$math)
b1Init → 0
sigmaInit → sd(nels$math)

initsList = list(b0=b0Init, b1=b1Init,
sigma=sigmaInit )

jagsNelsLR1 ← jags.model(file=‘‘nelsLR1.txt’’,
                data=dataList,
                inits=initsList,
                n.chains=4,
                n.adapt=500)
```

rjags: Sample and Summarize

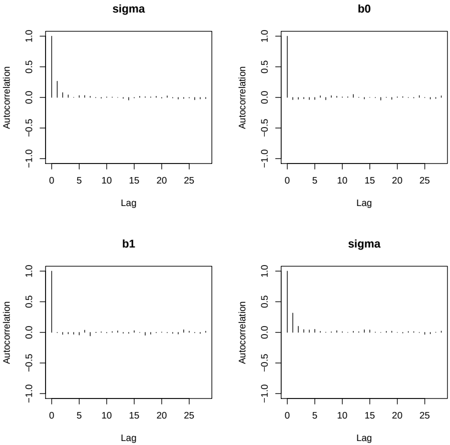
```
# run the Markov chain
update (jagsNelsLR1, n.iter=500)

# ‘‘wrapper’’: sets trace montior, up-dates model &
# puts output into single mcmc.list object
Samples ← coda.samples(jagsNelsLR1,
variable.names=c(‘‘b0’’,‘‘b1’’,‘‘sigma’’),
n.iter=4000)

# output summary information
summary(Samples)
plot(Samples)
```

rjags: Trace and Densities



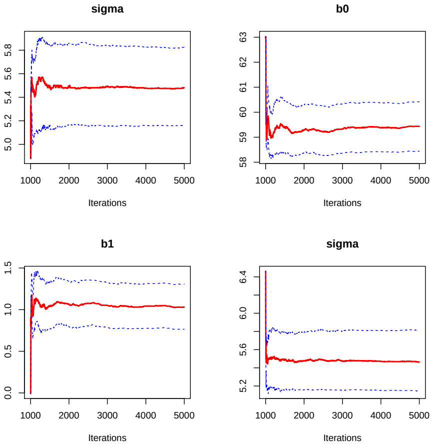
rjags: Auto-Correlations

One for each chain

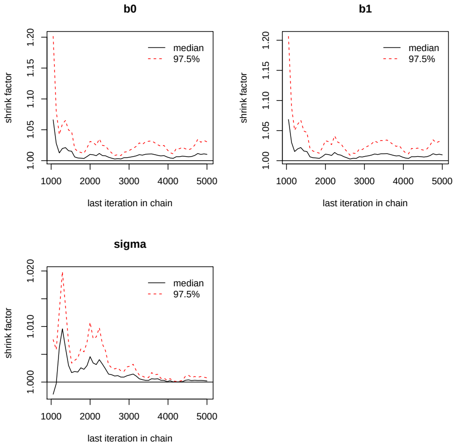


rjags: Estimates over Iterations

One for each chain



rjags: Gelman Plot



rjags — Summary Statistics

Iterations = 1001:5000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 4000

1. Empirical mean and standard deviation for each variable, plus standard error of the mean:

	Mean	SD	Naive SE	Time-series SE
b0	59.218	1.4925	0.011799	0.035264
b1	1.092	0.4010	0.003170	0.009364
sigma	5.500	0.4948	0.003912	0.005346

rjags — Summary Statistics (continued)

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b0	56.2152	58.2527	59.219	60.216	62.086
b1	0.3213	0.8216	1.093	1.356	1.886
sigma	4.6322	5.1528	5.468	5.807	6.580

rjags — Summary Statistics (continued)

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b0	56.239	58.191	59.180	60.141	62.127
b1	0.325	0.839	1.101	1.361	1.887
sigma	4.628	5.156	5.477	5.822	6.559

What can get from runjags

From the Mannual, the print(x) gives:

- "Lower95" and "Upper95" are the limits of the high density intervals.
- "SD" is the standard deviation of posterior samples.
- "MCerr" the Monte Carlo standard error $= SD/\sqrt{SSeff}$.
- "MC%ofSD" is The Monte Carlo standard error expressed as a percent of SD. The rule of thumb is that this should be less than 5% of sample sample SD.
- "SSeff" is effective sample size.
- "AC.XX" is the auto-correlation at lag XX where default if 10. This can be changed by giving the autocorr.diag argument.
- "psrf" is the potential scale factor or the Gelman-Rubin statistic (or "Rhat"). If there is a $ this indicates a problem with the model or sampler.

Note: plot(x) give 4 pannel plots.

runjags output

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000)

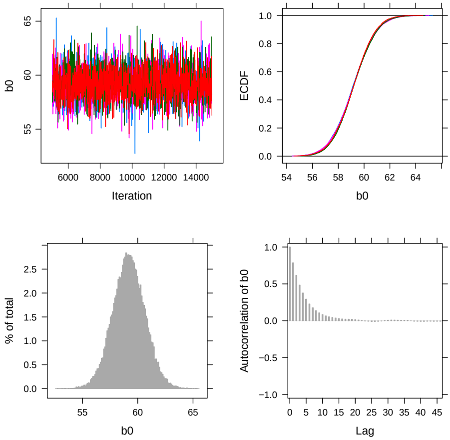
	Lower95	Median	Upper95	Mean	SD	Mode
b0	56.286	59.184	62.052	59.188	1.4641	–
b1	0.3429	1.1007	1.8876	1.0987	0.39336	–
sigma	4.5983	5.4634	6.5057	5.5001	0.49422	–

runjags output (continued)

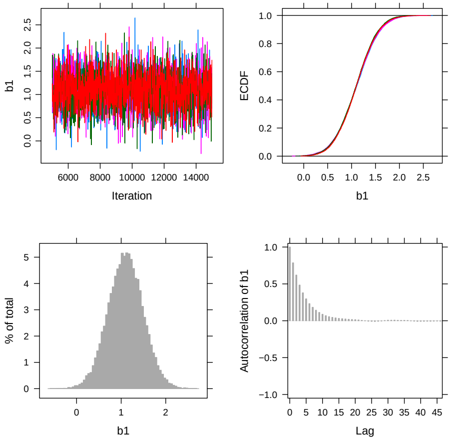
	MCerr	MC%ofSD	SSeff	AC.10	psrf
b0	0.021134	1.4	4799	0.084577	1.0007
b1	0.0056527	1.4	4842	0.087942	1.0006
sigma	0.0035014	0.7	19923	0.0084444	1.0001

Model fit assessment:
DIC = 420.153
PED not available from the stored object
Estimated effective number of parameters: pD =3.11249
Total time taken 4.8 seconds

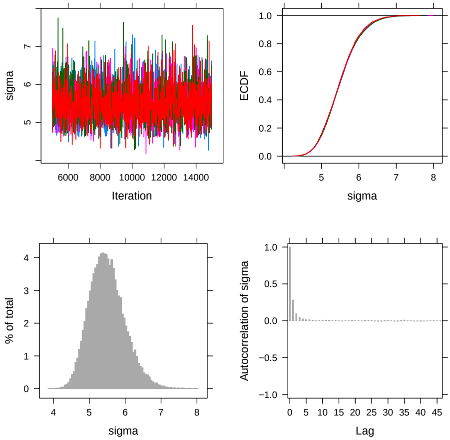
rjags: Gelman Plot



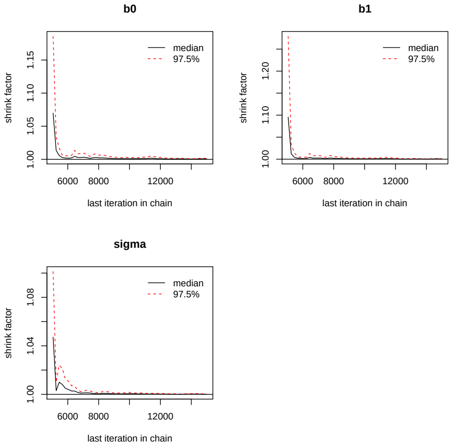
rjags: Gelman Plot



rjags: Gelman Plot



rjags: Gelman Plot



Robust Simple linear regression

We basically swap out the normal distribution and use Student's t-distribution. Everything stays the same, except the following (what changes is in red).

```
tMod1 = ‘‘model {
    for (i in 1:N){
     y[i] ∼ dt(mu[i] , precision,nu)
     mu[i] ← b0 + b1*x[i]
    }
    b0 ∼dnorm(0 , 1/(100*sdY^2) )
    b1 ∼ dnorm(0 , 1/(100*sdY^2) )
    sigma ∼dunif(0, 1E+10 )
    precision ← 1/sigma^2
    nuMinusOne ∼ dexp(1/29)
    nu ← nuMinusOne+1
    } ’’
```

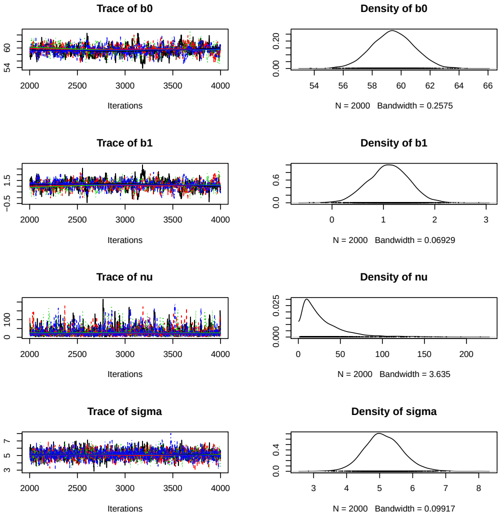
Change in code for t-distribution

```
b0Init ← mean(nels$math)
b1Init ← 0
sigmaInit ← sd(nels$math)
nuMinusOneInit = 20

initsList ← list(b0=b0Init, b1=b1Init,
sigma=sigmaInit,nuMinusOne=nuMinusOneInit )
```

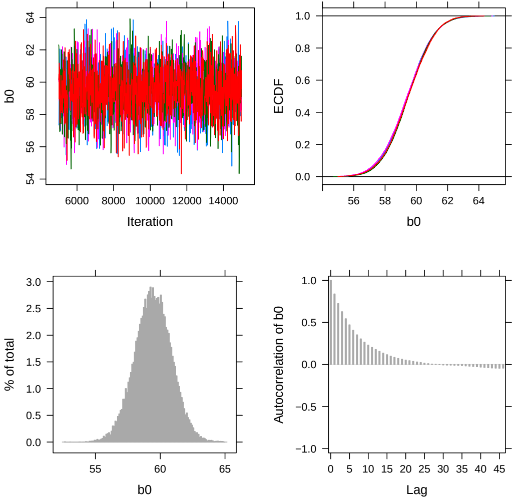
and

```
tSamples2 ← coda.samples(jagsModelLm2,
variable.names=c(‘‘b0’’,‘‘b1’’,‘‘sigma’’,‘‘nu’’),
n.iter=2000)
```

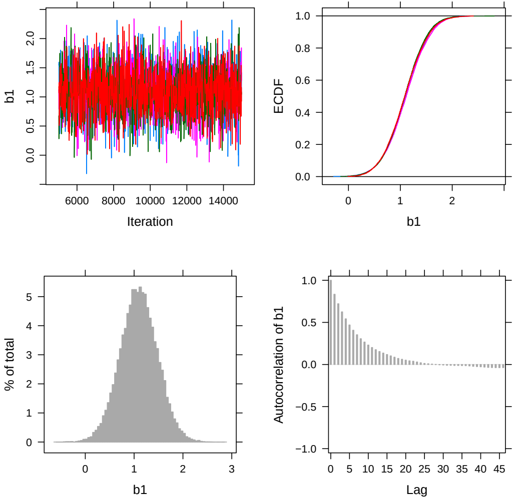
Diagnostics for estimates (from rjags)



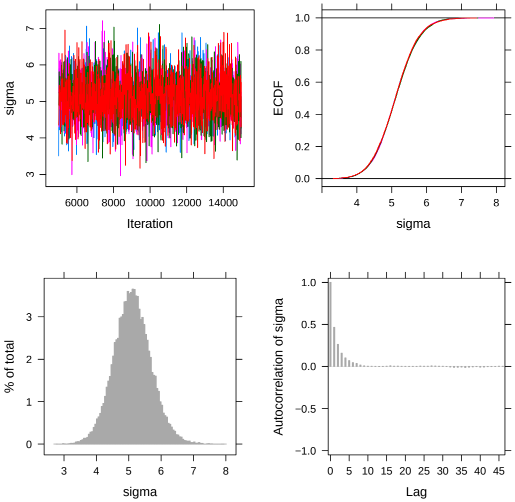
OR Diagnostics for b_0 (from runjags)



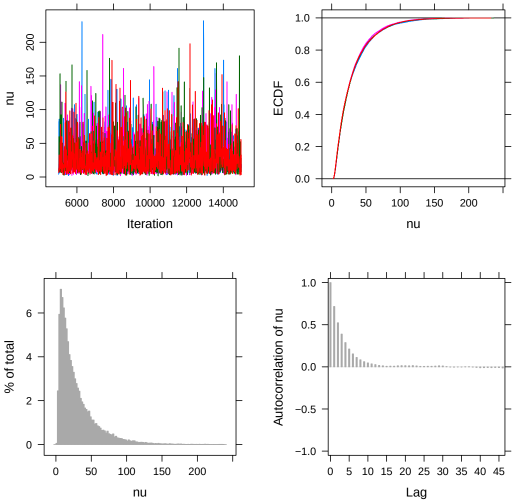
Diagnostics for b_1 (from runjags)



Diagnostics for σ (from runjags)



Diagnostics for ν (from runjags)



Robust Diagnostics (rjags)

```
effectiveSize(tSamples2)
```

```
       b0         b1          nu       sigma
 567.5006   604.8280   1287.6339   2445.2968

Potential scale reduction factors:
        Point est.  Upper C.I.
 b0          1.01        1.02
 b1          1.01        1.02
 nu          1.01        1.02
 sigma       1.00        1.01

 Multivariate psrf
         1.01
```

Robust Results: rjags

Iterations = 2001:4000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 2000

1. Empirical mean and standard deviation for each variable, plus standard error of the mean:

	Mean	SD	Naive SE	Time-series SE
b0	59.441	1.4660	0.016391	0.06438
b1	1.101	0.3956	0.004423	0.01683
nu	29.156	26.9500	0.301311	0.75915
sigma	5.118	0.5799	0.006484	0.01177

Robust Results: rjags (continued)

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b0	56.5776	58.4525	59.431	60.421	62.324
b1	0.3216	0.8424	1.106	1.371	1.861
nu	3.8901	10.6212	20.443	38.346	110.092
sigma	4.0183	4.7353	5.092	5.492	6.316

Note:

- Students's t with $\nu > 30$ is nearly a normal distribution
- Small values of ν can only be accurately estimated if the data have heavy tails.

Robust Results: runjags

Note: I first ran without giving it starting values and it gave very bad results; however, when I gave it reasonable starting values, the alogrithm gave the following:

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

	Lower95	Median	Upper95	Mean	SD	Mode
b0	56.642	59.451	62.265	59.453	1.4289	–
b1	0.34661	1.0957	1.8518	1.0955	0.38324	–
sigma	3.9875	5.1056	6.2838	5.1137	0.57833	–
nu	2.0294	20.828	84.275	29.593	26.939	–

Robust Results: runjags

	MCerr	MC%ofSD	SSeff	AC.10	psrf
b0	0.026115	1.8	2994	0.22984	1.0007
b1	0.0070362	1.8	2967	0.22999	1.0006
sigma	0.0053038	0.9	11890	0.0045078	1
nu	0.33956	1.3	6294	0.046536	1.0002

Model fit assessment:
DIC = 419.9194
PED not available from the stored object
Estimated effective number of parameters: pD = 3.57627

Total time taken: 2.1 minutes

Normal or t?

For the nels data, Student's t seems to have a *very slight* edge on the Normal, but not enough for me to switch to t for these data.

- $\nu \approx 20$
- DIC: Deviance information criteria (more on this later)

$$DIC_{normal} = 420.153 \qquad \approx \qquad DIC_{t-dist} = 420.034$$

- pD: estimated number of effective parameters (more on this later)

$$pD_{normal} = 3.11249 \qquad \leq \qquad pD_{t-dist} = 3.57627$$

Note: I first ran robust for 2,000 iterations, but diagnostics suggested that it didn't converge so I increased to 4,000. It might be worth while increasing this further.

Categorical Predictors

If there are only 2 level of a categorical predictor, we could do something akin to a 2 sample t-test:

- We approximate the posterior distributions of each level, with different means and possibly different variances.
- Approximate the posterior distribution of the difference between the means and if desired effect size.

If there are 2 or more levels,

- Do a Bayesian ANOVA model.
- We could dummy (or effect) code them and use them in a multiple regression model.

We will deal with 2 categorical, but will do multiple regression for 3 level predictor later.

Anorexia: 2 Groups

We will do this by example, using the Anorexia data.
The girls in the study received one of 3 possible treatments:

- Standard (coded as 2)
- Family (coded as 3)
- Cognitive (coded as 1)

But, dichotomoizing for now:

Treatment	mean	sd	var	n
standard	-0.450	7.989	63.819	26
alternative	4.580	7.468	55.767	46

Anorexia: standard analysis

t-test: $H_0 : \mu_s = \mu_a$ versus $H_1 : \mu_s \neq \mu_a$

$$t = -2.627, \qquad df = 70, \qquad p - value < .05$$

mean in group 0	mean in group 1
-0.450000	4.580435

jags: dataList

Looks mostly like any standard regression:

```
dataList <- list(y=ano$change,
              x=ano$altRx,
              N=length(ano$change),
              sdY = sd(ano$change)
              )
```

jags: Model with Discrete

```
TwoLevels1 = ‘‘model {
## Likelihood
   for (i in 1:N){
            y[i] ~ dnorm(mu[i] , 1/sigma^2)
            mu[i] <- b0 + b1*x[i]
   }
## Priors
   b0 ~ dnorm(0 , 1/(10*sdY^2) )
   b1 ~ dnorm(0 , 1/(10*sdY^2) )
   sigma ~ dunif( 1E-3, 1E+30 )
} ’’

writeLines(TwoLevels1, con=‘‘TwoLevels1.txt’’)
```

jags: Initialize and Compile

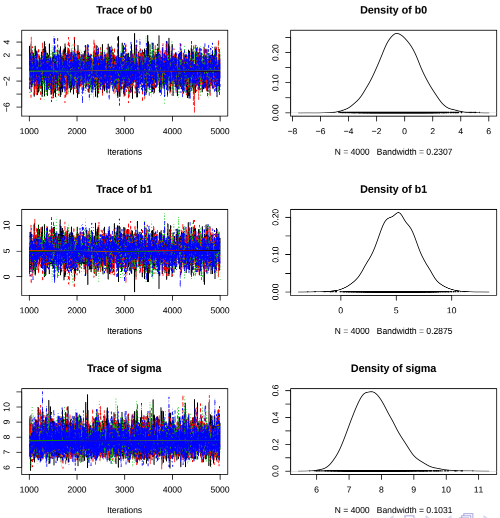
```
b0Init ← mean(ano$change)
b1Init ← 0
sigmaInit ← sd(ano$change)
initsList ← list(b0=b0Init, b1=b1Init,
sigma=sigmaInit )

jagsTwoLevels1 ← jags.model(file=''TwoLevels1.txt'',
                   data=dataList,
                   inits=initsList,
                   n.chains=4,
                   n.adapt=500
)
```

jags: Run and Summary

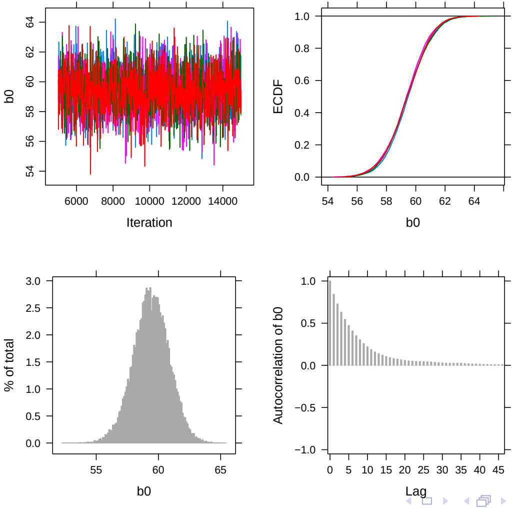
```
update (jagsTwoLevels1, n.iter=500)

twoSamp1 ← coda.samples(jagsTwoLevels1,
variable.names=c("b0","b1","sigma"), n.iter=4000)

summary(twoSamp1)
par(ask=TRUE)
plot(twoSamp1)
```

And run all diagnostics on results to check for convergence. . .

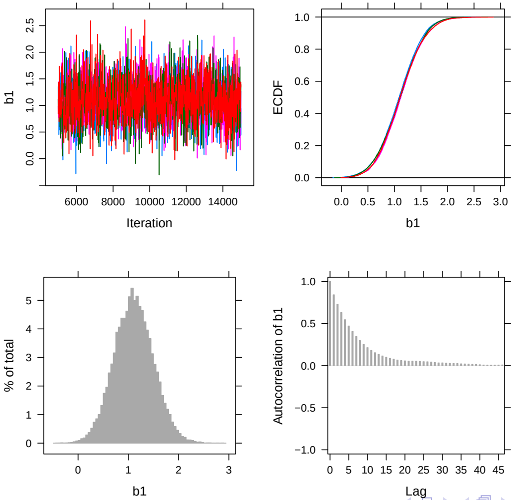
jags: Some diagnostics



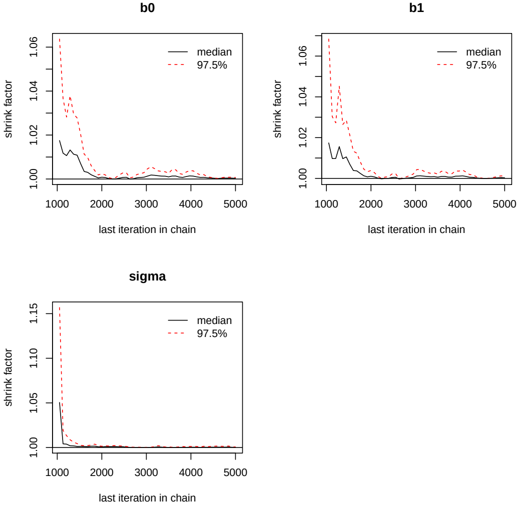
jags: Some diagnostics



jags: Some diagnostics



jags: Some diagnostics



Results

Iterations = 1001:5000
Thinning interval = 1
Number of chains = 4
Sample size per chain = 4000

1. Empirical mean and standard deviation for each variable plus standard error of the mean:

	Mean	SD	Naive SE	Time-series SE
b0	-0.4191	1.5087	0.011927	0.025225
b1	4.9912	1.8874	0.014921	0.031464
sigma	7.7980	0.6764	0.005348	0.007256

Note OLS:

	Mean	SD
b0	-0.450	1.502
b1	5.030	1.879
sigma	7.658	—

Results

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b0	-3.381	-1.427	-0.4378	0.595	2.507
b1	1.240	3.746	5.0115	6.265	8.672
sigma	6.613	7.320	7.7471	8.224	9.263

Results (runjags)

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

	Lower95	Median	Upper95	Mean	SD	Mode
b0	-3.413	-0.44216	2.5999	-0.43925	1.5236	–
b1	1.1796	5.0162	8.724	5.0122	1.9097	–
sigma	6.526	7.7556	9.1795	7.8018	0.68151	–

Results (runjags)

	MCerr	MC%ofSD	SSeff	AC.10	psrf
b0	0.0076523	0.5	39642	0.0013856	1
b1	0.0095995	0.5	39577	-0.0056292	1.0001
sigma	0.0045814	0.7	22129	-0.0025553	1.0001

Model fit assessment:
DIC = 501.678
PED not available from the stored object
Estimated effective number of parameters: pD = 3.12037

Total time taken: 5.8 seconds

Model Evaluation

- Model assumptions
- Comparing posterior inferences to substantive knowledge
- Posterior predictive checking
- Sensitivity analysis (prior and likelihood)

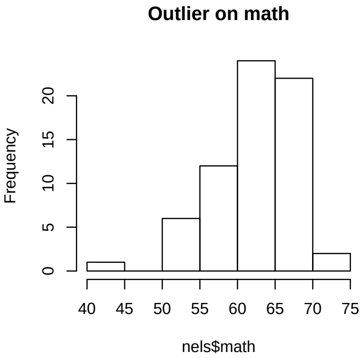
We talk about each and do them on the NELS dataset.

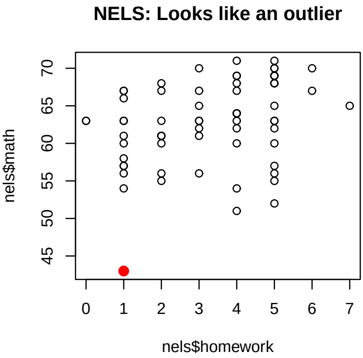
Model Assumptions

Things to consider:

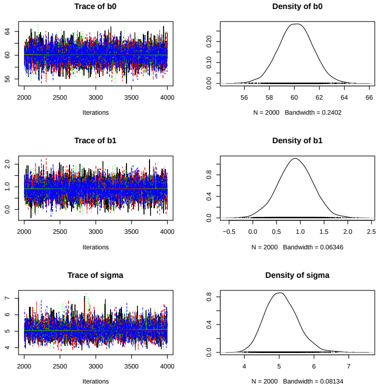
- Design of the study?
  - Random (or least independent observations)
  - Large enough sample size.
  - Was data checked for outliers or iregularities? $\rightarrow$ there is an outlier
  - Is normality reasonable?
- Exchangeability (before seeing results – a priori): The data tells us about relative ordering of effects and similarity between effects. If we don't include features.
  - NELS: If it is known that students in group A get a better test scores that students in groups B (not due to how much homework they do), this would suggest that prior is not a single normal distribution.
  - The data should be an independent sample from the same population.
  - The left handiness of presidents didn't meet this assumption (nor independence)
- Normality Reason? (a priori)
- Should the scale (and location) be uniformly distributed?

The Outlier


Outlier on math
Frequency
0
5
10
15
20
40
45
50
55
60
65
70
75
nels$math


NELS: Looks like an outlier
nels$math
45
50
55
60
65
70
0
1
2
3
4
5
6
7
nels$homework

Model without Outlier



Re-run Model without Outlier

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

	Lower95	Median	Upper95	Mean	SD	Mode
b0	57.477	60.137	62.862	60.131	1.3714	–
b1	0.1779	0.89154	1.6213	0.89476	0.36696	–
sigma	4.23	5.0332	6.0037	5.0667	0.46093	–

With the outlier, $b_0 = 59.180(MCerr = 0.008)$, $b_1 = 1.100(MCerr = 0.010)$ and $\sigma = 5.508(MCerr = 0.005)$.

Keep it in data or leave it out? (I left it out in the following, but this is debatable)

Re-run Model without Outlier (continued)

	MCerr	MC%ofSD	SSeff	AC.10	psrf
b0	0.006857	0.	5 40000	-0.011618	1
b1	0.0018278	0.	5 40307	-0.0094452	1
sigma	0.0031472	0.	7 21451	0.0059596	1

Model fit assessment:
DIC = 403.0281
PED not available from the stored object
Estimated effective number of parameters: pD = 3.10001

Total time taken: 5.3 seconds

Posterior Inferences & Substantive Knowledge

Inference about the parameters and what we know about the data.

For example, NELS data:

- Is $b_0 = 60.131$ reasonable? Is it similar to overall mean of data, which is 63.121?
- Is $b_1 = 0.89476$ reasonable? Note that the $\min(y) = 51$ and $\max(y) = 71$, and interquartile range goes from 60.00 to 67.75. Why would $b_1 = 20$ would be unreasonable?
- Is $\sigma = 5.0667$ reasonable? Note that $sd(y) = 5.155$.

These seem OK for NELS data.

Posterior Predictive Check

Inference about predicted values

For this we can simulated the posterior predictive distribution using Monte Carlo method.

We can use Monte Carlo methods. Lets replicate predictions of y many times:

1. Draw b_0 from it's posterior: $b_0 \sim N(\bar{b}_0, sd(b_0)^2)$, where $\bar{b}_0 = 60.131$ and $sd(b_0) = 1.3714$
2. Draw b_1 from it's posterior: $b_1 \sim N(\bar{b}_1, sd(b_1)^2)$, where $\bar{b}_1 = 0.89476$ and $sd(b_1) = 0.46093$
3. For each student:
   - Draw ϵ_i from it's posterior: $\epsilon_i \sim N(0, \bar{\sigma}^2)$, where $\bar{\sigma} = 5.0667$
   - Compute:

$$y_i^{(rep)} = b_0 + b_1 x_i + \epsilon_i$$

4. Repeat steps 1-3 for desired number of replications.

Output from Monte-Carlo

Suppose that we have drawn S replications from the posterior predictive distributions so that we have

$$\begin{pmatrix} y_1^{(1)} & y_1^{(2)} & \dots & y_1^{(S)} \\ y_2^{(1)} & y_2^{(2)} & \dots & y_2^{(S)} \\ y_3^{(1)} & y_3^{(2)} & \dots & y_3^{(S)} \\ \vdots & \vdots & \ddots & \vdots \\ y_N^{(1)} & y_N^{(2)} & \dots & y_N^{(S)} \end{pmatrix} \rightarrow \begin{matrix} g(y_1) \\ g(y_2) \\ g(y_3) \\ \vdots \\ g(y_N) \end{matrix}$$

$$\downarrow$$

$$h(y^{(1)}) \quad h(y^{(2)}) \quad \dots \quad h(y^{(S)})$$

where $g(\ )$ and $h(\ )$ are statistics, e.g., mean, min, max, sd, etc.

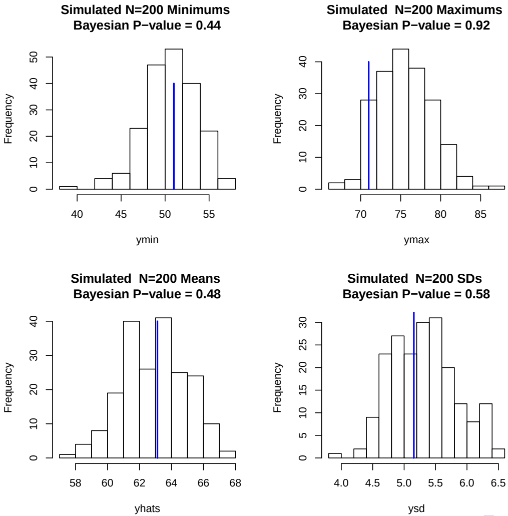
Bayesian p-value

A Bayesian *p*-value is the number of times the $h(y^{(rep)})$s are greater than the statistic computed on the data.

For example, I ran 200 replications and 48% of the time the means from the Monte Carlo simulation were greater than the mean for the data (i.e., 63.12).

We want *p*-values near .5.

NELS: Posterior Predictive Distribution



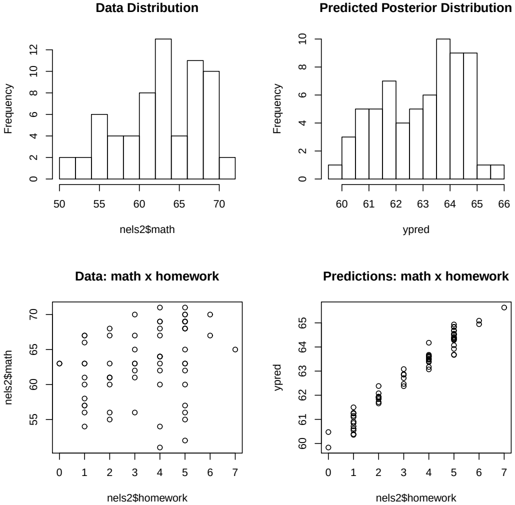
Means over Replications

If we take the mean over replications,

$$(y_i^{(1)} + y_i^{(2)} + \ldots + y_i^{(S)})/S = \bar{y}_i$$

This gives us a posterior prediction of y the ith students.

Distributions of Predictions



Sensitivity Analysis

- Try different priors:
  - Different parameters for them (if have prior beliefs)
  - Different distributions, e.g., Gamma for precision rather than uniform
- Try different likelihoods, e.g., use Student-t instead of Normal.

Summary

What we did in this set of notes:

- Brief overview of Generalize linear models
- Gained more experience with jags, including what is in the output summaries given by runjags..
- Put in a linear model for the mean.
- Used a normal likelihood for the data.
- Used a t-distribution as the likelihood for the data: “robust”.
- Covered some methods for model evaluation.
- Did some posterior predictive checks of statistics computed on data and of predictions of y, in effect we solved integration problems via Monte Carlo simulations.

Next topic: Multiple Linear Regression