# ISyE 6421 Midterm

Stephen Thomas

## 1. Master of Light

Albert Michelson devised a clever way to increase precision of measuring the speed of light (Michelson, 1927[1]) by rotating mirrors. In his experiment the light was sent from Mount Wilson, reflected back from Mount San Antonio, and the time of flight was recorded. Back-engineered (when the speed of light is assumed known) crude calculations for an octagonal rotating mirror look as follows.

The round trip distance between Mount Wilson and Mount San Antonio is $d$ = 70,800 m. Dividing $d$ by speed of light $c$ = $2.997 \times 10^{10}$ m/s results in time of flight $t = 2.36 \times 10^{-4}$ s. When an octagonal mirror rotates at 529 rev/s the time between successive surfaces is (1/529)/8 =$2.36\times10^{-4}$, the same as $t$. Thus, the return image of light should fall in the same place. Michelson was able to control the mirror rotation rate and align perfectly the locations of sent and received lights. A diagram of his equipment and data can be found in Michelson (1927, page 4). The data in the table below represent the two last digits in his measurements and can be taught of as observations between 0 and 99.

[table suppressed]

(a) Using WinBUGS and a Bayesian one-way ANOVA test whether the precision measurements for different types of mirrors are statistically the same.

(b) Which treatment level(s) is(are) different. Find 95% credible sets for differences $\mu_1 - \mu_2$ and $\mu_3 - \mu_4$.

(c) Consider now data for only four treatment levels Glass/8, Glass/12, Steel/8, and Steel/12.Recode the data to conduct a two-way ANOVA with factors Material (levels Glass and Steel) and Sides (levels 8 and 12). Pretend you are data-analyst in Michelson's lab. If Michelson was interested whether the factors Material and Sides interact, how would you advise him. Use WinBUGS for your analysis.

---

### OpenBUGS Code for parts (a) and (b)

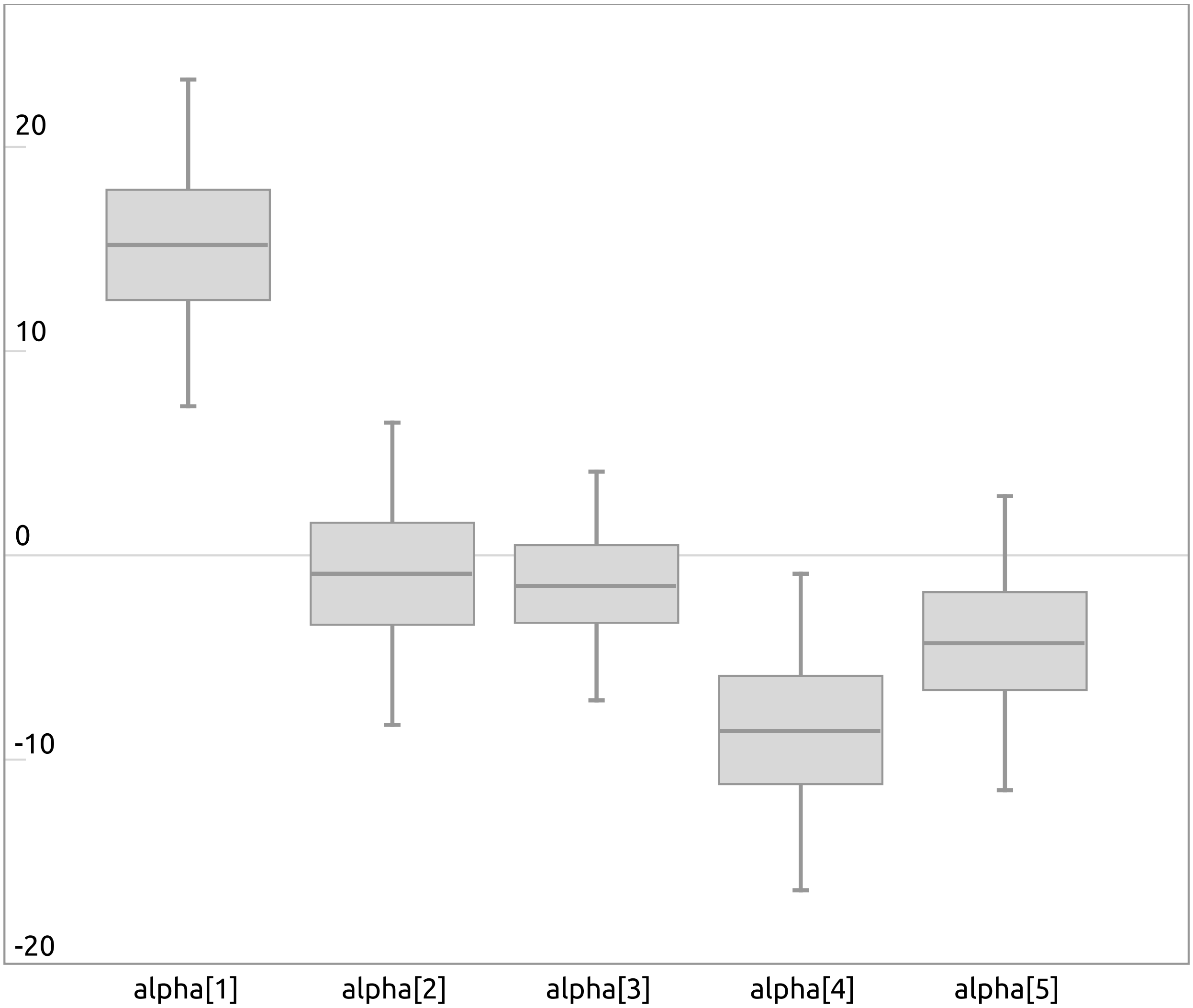
```
model {

    # For consistency, the following indices are used throughout the model:
    #
    #   i: 1:numObservations - the individual observations
    #   j: 1:numMirrors - the types of mirrors (Glass/8, Glass/12, ...)

    # Likelihoods

    for (i in 1:numObservations) {

        # Assume the measured value is normal with mean mu
        # and precision tau. Since the observations represent
        # different mirrors, the mean may differ for each.
        # The precision, however, is assumed to be the same
```

```
        # for all observations as it models measurement
        # error.

        measurement[i] ~ dnorm(mu[i], tau)

        # The mean for each measurement is calculated as the
        # sum of a base value (mu0) and the effect of the
        # mirror type. This later quantity is alpha[j], where
        # j is the mirror type. The value for j is read from
        # the data as mirror[i], where i is the observation.
        # Note that this assumes that mirror types are coded
        # as 1, 2, ….

        mu[i] <- mu0 + alpha[mirror[i]]
    }

    # Constraints on parameters. We use "sum-to-zero" so
    # that Sum(alpha) = 0. To enforce the constraints, let
    # the simulation pick values for indices 2, 3, … and
    # then calculate a value alpha[1] that satisfies the
    # constraint.

    alpha[1] <- - sum(alpha[2:numMirrors])

    # Priors (reference priors = non-informative)

    mu0 ~ dnorm(0, 0.0001)
    tau ~ dgamma(0.001, 0.001)
    for (j in 2:numMirrors) {
        alpha[j] ~ dnorm(0, 0.0001)
    }

    # Additional intermediate values to track
    #
    #   deltaAlpha - differences between alpha values

    for (j1 in 1:(numMirrors - 1)) {
        for (j2 in (j1 + 1):numMirrors) {
            deltaAlpha[j1,j2] <- alpha[j1] - alpha[j2]
        }
    }
}

# Observed data characteristics

list(
    numObservations = 89,
    numMirrors = 5
)

# Initial values

list(
    mu0 = 0,
    alpha = c(NA, 0, 0, 0, 0),
    tau = 1
)

# Observations

measurement[]  mirror[]
     47           1
     47           1
     38           1
# …
END
```

After a burn-in of 1000 iterations, 100000 updates were executed. The following table shows relevant statistics from the simulation.

| | MEAN | SD | 2.5% | MEDIAN | 97.5% |
|---|---|---|---|---|---|
| alpha[1] | 15.27 | 4.075 | 7.322 | 15.25 | 23.29 |
| alpha[2] | −0.9187 | 3.748 | −8.252 | −0.9088 | 6.453 |
| alpha[3] | −1.453 | 2.857 | −7.08 | −1.465 | 4.18 |
| alpha[4] | −8.605 | 3.945 | −16.38 | −8.602 | −0.8503 |
| alpha[5] | −4.289 | 3.655 | −11.46 | −4.289 | 2.891 |
| deltaAlpha[1,2] | 16.18 | 6.31 | 3.776 | 16.18 | 28.61 |
| deltaAlpha[1,3] | 16.72 | 5.47 | 5.971 | 16.7 | 27.5 |
| deltaAlpha[1,4] | 23.87 | 6.502 | 11.15 | 23.85 | 36.79 |
| deltaAlpha[1,5] | 19.56 | 6.225 | 7.388 | 19.55 | 31.84 |
| deltaAlpha[2,3] | 0.534 | 5.075 | −9.412 | 0.5333 | 10.54 |
| deltaAlpha[2,4] | 7.687 | 6.17 | −4.395 | 7.685 | 19.87 |
| deltaAlpha[2,5] | 3.371 | 5.858 | −8.161 | 3.371 | 14.85 |
| deltaAlpha[3,4] | 7.153 | 5.322 | −3.306 | 7.144 | 17.63 |
| deltaAlpha[3,5] | 2.837 | 4.956 | −6.882 | 2.819 | 12.59 |
| deltaAlpha[4,5] | −4.316 | 6.074 | −16.3 | −4.324 | 7.609 |

Comparing box plots of the alpha values shows clear non-overlapping credible sets, indicating that the precision measurements for different types of mirrors are *not* statistically the same.




The treatment levels that differ significantly are indicated by the credible sets for `deltaAlpha` that do not include the value 0. The following table lists the significant treatment differences.

| | | 2.5% | MEDIAN | 97.5% |
|---|---|---|---|---|
| Glass/8 vs. Glass/12 | deltaAlpha[1,2] | 3.776 | 16.18 | 28.61 |
| Glass/8 vs. Glass/16 | deltaAlpha[1,3] | 5.971 | 16.7 | 27.5 |
| Glass/8 vs. Steel/12 | deltaAlpha[1,4] | 11.15 | 23.85 | 36.79 |
| Glass/8 vs. Steel/8 | deltaAlpha[1,5] | 7.388 | 19.55 | 31.84 |

The 95% credible set for $\mu_1 - \mu_2$ ranges from 3.776 to 28.61. The 95% credible set for $\mu_3 - \mu_4$ ranges from −3.306 to 17.63.

## OpenBUGS Code for Part (c)

```
# For consistency, the following indices are used throughout the code:
#
#   i: 1:numObservations - the individual observations
#   j: 1:numMaterials - the types of materials (Glass/8, Glass/12)
#   k: 1:numSides - the types of sides (8, 12)

model {

    for (i in 1:numObservations) {

        # Assume the measured value is normal with mean mu
        # and precision tau. Since the observations represent
        # different mirrors, the mean may differ for each.
        # The precision, however, is assumed to be the same
        # for all observations as it models measurement
        # error.

        measurement[i] ~ dnorm(mu[i], tau)

        # The mean for each measurement is calculated as the
        # sum of several values:
        #   mu0 - a base value (mu0)
        #   alpha[j] - the effect of the material j
        #   beta[k] - the effect of the number of sides k
        #   alpha.beta[j,k] - the effect of the interaction between
        #                     material j and the number of sides k
        #
        # Note that this assumes that both material types and
        # the number of sides are coded as 1, 2, ….

        mu[i] <- mu0 + alpha[material[i]] + beta[sides[i]] +
                       alpha.beta[material[i],sides[i]]
    }

    # Constraints on parameters. We use "sum-to-zero" so
    #   Sum(alpha) = 0
    #   Sum(beta) = 0
    #   Sum(alpha.beta) = 0

    # To enforce these constraints, let the simulation pick
    # values for indices 2, 3, … and then calculate a for
    # index 1 that satisfies the constraint.

    alpha[1] <- - sum(alpha[2:numMaterials])
    beta[1] <- - sum(beta[2:numSides])
    for (j in 1:numMaterials) {
        alpha.beta[j,1] <- - sum(alpha.beta[j, 2:numSides])
    }
    for (k in 2:numSides) {
        alpha.beta[1,k] <- - sum(alpha.beta[2:numMaterials, k])
    }

    mu0 ~ dnorm(0, 0.0001)
    tau ~ dgamma(0.001, 0.001)
    for (j in 2:numMaterials) {
        alpha[j] ~ dnorm(0, 0.0001)
    }
    for (k in 2:numSides) {
        beta[k] ~ dnorm(0, 0.0001)
    }
    for (j in 2:numMaterials) {
        for (k in 2:numSides) {
            alpha.beta[j,k] ~ dnorm(0, 0.0001)
        }
    }
}

# Observed data characteristics

list(
    numObservations = 56,
    numMaterials = 2,
```

```
    numSides = 2
)

# Initial values

list(
    mu0 = 0,
    alpha = c(NA, 0),
    beta = c(NA, 0),
    alpha.beta = structure(
        .Data = c(
            NA, NA,
            NA,  0
        ),
        .Dim = c(2, 2)
    ),
    tau = 1
)

# Observations

measurement[]  material[]  sides[]
     47            1           1
     47            1           1
     38            1           1
# …
END
```

After a burn-in of 1000 iterations, 100000 updates were executed. The following table shows relevant statistics from the simulation. (In this case, there is only one interaction effect, but the code above supports an arbitrary number.)

| | MEAN | SD | 2.5% | MEDIAN | 97.5% |
|---|---|---|---|---|---|
| alpha.beta[1,2] | −2.965 | 1.86 | −6.628 | −2.966 | 0.6907 |

As the 95% credible set includes the value 0, no claim may be made about interaction between material type and the number of sides.

*(As an aside, a classical 2-way ANOVA performed using the Stata application shows a p-value for interaction effects of 10.9%. This result is consistent with a Bayesian analysis of the 90% credible set which shows a range from −6.024 to 0.1002, a range which only barely includes the value 0.)*

## 2. Body Fat Affecting Accuracy of Heart Rate Monitors

A team of students investigated whether the readings of heart rate from a chest strap monitor (Polar T31) were influenced by the subject's percent body fat. Hand counts facilitated by a stethoscope served as the gold standard. The absolute differences between device and hand counts (AD) were regressed on body fat (BF) measurements. The measurements for 28 subjects are provided below.

| SUBJ | BF | AD | SUBJ | BF | AD | SUBJ | BF | AD | SUBJ | BF | AD |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 17.8 | 4 | 8 | 18.8 | 1 | 15 | 25.1 | 3 | 22 | 24.1 | 6 |
| 2 | 13.2 | 3 | 9 | 13.4 | 0 | 16 | 18.3 | 2 | 23 | 12.9 | 2 |
| 3 | 7.7 | 3 | 10 | 39.4 | 7 | 17 | 16.9 | 3 | 24 | 30.1 | 6 |
| 4 | 11.8 | 1 | 11 | 6.8 | 1 | 18 | 27.8 | 6 | 25 | 17.1 | 4 |
| 5 | 23.9 | 0 | 12 | 25.0 | 6 | 19 | 36.0 | 5 | 26 | 18.4 | 4 |
| 6 | 27.2 | 0 | 13 | 19.9 | 0 | 20 | 31.9 | 1 | 27 | 14.6 | 4 |
| 7 | 27.6 | 0 | 14 | 23.0 | 9 | 21 | 17.4 | 2 | 28 | 26.8 | 3 |

A significant non-constant representation of AD as a function of BF can be translated as a significant influence of percent of body fat on the accuracy of the device.

(a) Why the linear regression is not adequate here?

(b) Fit the Poisson regression AD ~ *Poi* (exp{ $b_0 + b_1$ BF }). Is the slope $b_1$ from the linear part significantly positive?

(c) Using WinBUGS find 95% credible set for the slope $b_1$ in the linear part of a Bayesian Poisson regression model. Use the non-informative priors.

---

The response variable reported as an absolute value and no information is provided for its algebraic sign, making a linear regression inappropriate. As all response values are positive, however, data may be fit to a Poisson regression. Because the sample variance of the response (5.92) is greater than its sample mean (3.07), the code below uses a Poisson-lognormal (Millar 2009) model to account for overdispersion.

```
model {

    # We enhance the basic Poisson model by including a random effect
    # size to account for excess variance. This addition is modeled
    # with a Normal distribution.

    for (i in 1:numObservations) {
        ad[i] ~ dpois(lambda[i])
        log(lambda[i]) <- b0 + b1 * bf[i] + eps[i]
        eps[i] ~ dnorm(0, tau.eps)
    }

    b0 ~ dnorm(0, 0.0001)
    b1 ~ dnorm(0, 0.0001)
    tau.eps ~ dgamma(0.001, 0.001)
}
```

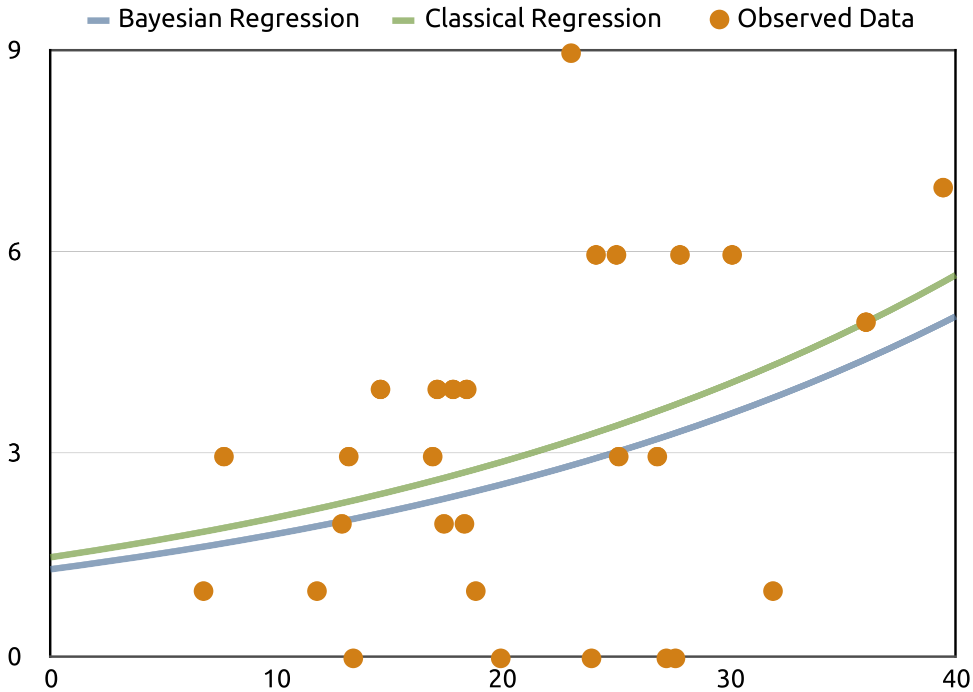
```
# Observed data characteristics

list(
    numObservations = 28
)

# Initialization

list(
    b0 = 0,
    b1 = 0,
    tau.eps = 1
)

# Observations

bf[]  ad[]
17.8    4
13.2    3
 7.7    3
# …
END
```

After a burn-in of 1000 iterations, 100000 updates were executed. The following table shows relevant statistics from the simulation. (In this case, there is only one interaction effect, but the code above supports an arbitrary number.)

| | MEAN | SD | 2.5% | MEDIAN | 97.5% |
|---|---|---|---|---|---|
| b0 | 0.2516 | 0.4419 | −0.6752 | 0.2673 | 1.08 |
| b1 | 0.03418 | 0.01833 | −0.002432 | 0.03415 | 0.0704 |

As the 95% credible set for `b1` includes the value 0, the slope $b_1$ from the linear part is not significantly positive. The95% credible set for $b_1$ ranges from −0.002432 to 0.0704.

*(The following graph compares the Bayesian Poisson regression model above—using mean values for the parameters—and a classical Poisson regression model from the Stata application against the observed data.)*




## 4. Squids

Data analyzed by Freund and Wilson 1998[2] were obtained on 22 squids.The dependent variable *y* is the weight of the squid in pounds. The predictor variables represent measurements on the beak or mouth of the squid. The data are provided in the file `squids.dat`.

| | | |
|---|---|---|
| Column 1 | Observation | - |
| Column 2 | Rostral length in inches | $x_1$ |
| Column 3 | Wing length in inches | $x_2$ |
| Column 4 | Rostral to notch length | $x_3$ |
| Column 5 | Notch to wing length | $x_4$ |
| Column 6 | Width in inches | $x_5$ |
| Column 7 | Weight of the squid in pounds | $y$ |

Note that value for $x_1$ is missing for observation #11, and that the response is missing for observation #21. Both represent ignorable missingness with distribution on $x_1 \sim N(3/2, 0.5^2)$.

Scientists wanted to know how useful beak measurements are in predicting the weight of the squid. Answering this question was important in the study of sizes of squid eaten by sharks and tuna, since the beak is indigestible.

(a) Using multiple linear regression, estimate a linear model that expresses the squid weight $y$ using the predictors $x_1$, …, $x_5$. Find Bayesian $R^2$. What are estimates for missing $x_1$ and $y$?

(b) What $y$ is predicted for $x_1 = 1.52$, $x_2 = 1.12$, $x_3 = 0.622$, $x_4 = 0.917$, and $x_5 = 0.324$? What is the 95% credible set for the prediction?

(c) Suppose that one variable needs to be eliminated. Which model consisting of 4 variables would you recommend according to Ibrahim-Laud criteria?

---

## OpenBUGS Code for Parts (a) and (b)

```
model {

    for (i in 1:numObservations) {
        y[i] ~ dnorm(mu[i], tau)
        mu[i] <- beta0 + inprod(beta[], x[i,])
    }

    beta0 ~ dorm(0, 0.0001)
    for (j in 1:numCovariates) {
        beta[j] ~ dnorm(0, 0.0001)
    }
    tau ~ dgamma(0.01, 0.01)

    # Prior for missing covariate value
    x[11,1] ~ dnorm(1.5, 4)

    # Calculate R^2
    sse <- (numObservations - numCovariates - 1) * (1/tau)
    for (i in 1:numObservations) {
        diff.y[i] <- y[i] - mean(y[])
    }
    sst <- inprod(diff.y[], diff.y[])
    r2 <- (1 - (sse/sst))

    # Predicted response for new x
    y.predicted ~ dnorm(mu.predicted, tau)
    mu.predicted <- beta0 + inprod(beta[], x.new[])
}

# Characteristics of Observations
list(
    numObservations = 22,
    numCovariates = 5
)

# Desired Prediction
list(
    x.new = c(1.52, 1.12, 0.622, 0.917, 0.324)
)

# Initial Values
list(
    beta0 = 0,
    beta = c(0, 0, 0, 0, 0),
```

```
    tau = 1
)

# Observations
x[,1]  x[,2]  x[,3]  x[,4]  x[,5]  y[]
1.31   1.07   0.44   0.75   0.35   1.95
1.55   1.49   0.53   0.90   0.47   2.90
0.99   0.84   0.34   0.57   0.32   0.72
0.99   0.83   0.34   0.54   0.27   0.81
1.05   0.90   0.36   0.64   0.30   1.09
1.09   0.93   0.42   0.61   0.31   1.22
1.08   0.90   0.40   0.51   0.31   1.02
1.27   1.08   0.44   0.77   0.34   1.93
0.99   0.85   0.36   0.56   0.29   0.64
1.34   1.13   0.45   0.77   0.37   2.08
 NA    1.10   0.45   0.76   0.38   1.98
1.33   1.10   0.48   0.77   0.38   1.90
1.86   1.47   0.60   1.01   0.65   8.56
1.58   1.34   0.52   0.95   0.50   4.49
1.97   1.59   0.67   1.20   0.59   8.49
1.80   1.56   0.66   1.02   0.59   6.17
1.75   1.58   0.63   1.09   0.59   7.54
1.72   1.43   0.64   1.02   0.63   6.36
1.68   1.57   0.72   0.96   0.68   7.63
1.75   1.59   0.68   1.08   0.62   7.78
2.19   1.86   0.75   1.24   0.72    NA
1.73   1.67   0.64   1.14   0.55   6.88
END
```

After generating initial values and a burn-in of 1000 iterations, 100000 updates were executed. The following table shows relevant statistics from the simulation.

| | MEAN | SD | 2.5% | MEDIAN | 97.5% |
|---|---|---|---|---|---|
| beta0 | −6.088 | 1.032 | −8.113 | −6.094 | −3.993 |
| beta[1] | 0.7909 | 2.693 | −3.902 | 0.4869 | 6.881 |
| beta[2] | −4.255 | 3.0 | −10.49 | −4.277 | 1.998 |
| beta[3] | 3.58 | 6.659 | −10.27 | 3.794 | 16.02 |
| beta[4] | 6.771 | 3.973 | −1.702 | 6.899 | 14.44 |
| beta[5] | 14.68 | 4.89 | 4.988 | 14.5 | 24.51 |
| x[11,1] | 1.376 | 0.4002 | 0.6381 | 1.338 | 2.25 |
| r2 | 0.9559 | 0.01909 | 0.9083 | 0.9601 | 0.9793 |
| y.pred | 3.537 | 1.686 | 0.07384 | 3.576 | 6.78 |

Linear prediction for squid weight (using mean values) is

$y = -6.088 + 0.7909 \cdot x_1 - 4.255 \cdot x_2 + 3.58 \cdot x_3 + 6.771 \cdot x_4 + 14.68 \cdot x_5$

Bayesian $R^2$ is estimated to be 0.9559 (using mean value).

Node values shows the estimated value for the missing `y[21]` to be 9.601. The simulation results provide a mean estimate for the missing `x[11,1]` of 1.376.

The predicted response value for the supplied covariates is 3.537 with a 95% credible set ranging from 0.07384 to 6.78.

## OpenBUGS Code for Part (c)

```
model {

    # The following indices are used throughout this section
    #
    #   m: 1:numCovariates+1  -  Index for the model. Model m=i
    #                            is the model with covariate i
    #                            omitted. The last model has all
    #                            covariates included.
    #   p: 1:numCovariates    -  Index for covariate.
    #   n: 1:numObservations  -  Index for observation.

    # Create a separate GLM for each model
    for (m in 1:(numCovariates+1)) {
        for (n in 1:numObservations) {

            #  mu[m,n]   -  response mean for model m, observation n
            #  beta0[m]  -  intercept for model m
            #  beta[m,]  -  coefficents for model m
            #  x[n,]     -  covariates for observation n
            mu[m,n] <- beta0[m] + inprod(beta[m,], x[n,])

            # To calculate Ibrahim-Laud, we use both the observed
            # response values and a separate set of simulated
            # responses from the simulated GLM parameters
            y.observed[m,n] ~ dnorm(mu[m,n], tau[m])
            y.synthetic[m,n] ~ dnorm(mu[m,n], tau[m])

            # Set y.observed to the response values
                y.observed[m,n] <- y[n]

        }

        tau[m] ~ dgamma(10, 50)
        beta0[m] ~ dnorm(0, 0.0001)
        for (p in 1:numCovariates) {
            a[m,p] ~ dnorm(0, 0.0001)
            beta[m,p] <- (1 - equals(m,p)) * a[m,p]
        }
    }

    # Calculate Ibrahim-Laud as L[m]
    for (m in 1:(numCovariates+1)) {
        for (n in 1:numObservations) {
            # Compute the square of the difference between
            # observed values and synthetic values
            diff2[m,n] <- pow((y.observed[m,n] - y.synthetic[m,n]), 2)
        }
        L.diff2[m] <- sum(diff2[m,])
        L.var[m] <- pow(sd(y.synthetic[m,]),2)
        L[m] <- sqrt(L.diff2[m] + L.var[m])
    }
}

# Characteristics of Observations
list(
```

```
    numObservations = 22,
    numCovariates = 5
)

# Initial Values
list(
    tau = c(1, 1, 1, 1, 1, 1)
)

# Observations
x[,1]   x[,2]   x[,3]   x[,4]   x[,5]    y[]
1.31    1.07    0.44    0.75    0.35    1.95
1.55    1.49    0.53    0.90    0.47    2.90
# …
END
```

After generating random initial values for those not supplied in the code and after a burn-in of 1000 iterations, 100000 updates were executed. The table below resulting mean values for `L[ ]` as well as its constituent components. Model 3, which omits $x_3$, has the lowest value and would be recommended.

| MODEL | OMITS | L-VALUE | $(EZ_l - Y_l)^2$ | $VAR(Z_l)$ |
|---|---|---|---|---|
| 1 | $x_1$ | 10.16 | 92.97 | 13.60 |
| 2 | $x_2$ | 10.21 | 93.99 | 13.58 |
| 3 | $x_3$ | 10.05 | 90.51 | 13.52 |
| 4 | $x_4$ | 10.11 | 91.77 | 13.42 |
| 5 | $x_5$ | 10.46 | 99.00 | 13.48 |
| 6 | - | 10.38 | 97.30 | 13.81 |

Of the models that omitted a covariant, only model 5 performed worse than the complete model with all covariates. Also, most of the L-value consists of the means of the predictive distribution rather than the variance.