Math 459 Lecture 20

Todd Kuffner

Last Time

Estimating the marginal likelihood

Laplace approximation

MCMC estimation

Today: generalized linear models (borrowing extensively from mages' blog)

Ice cream sales and temperature

Amount of ice cream sold at different temperatures:

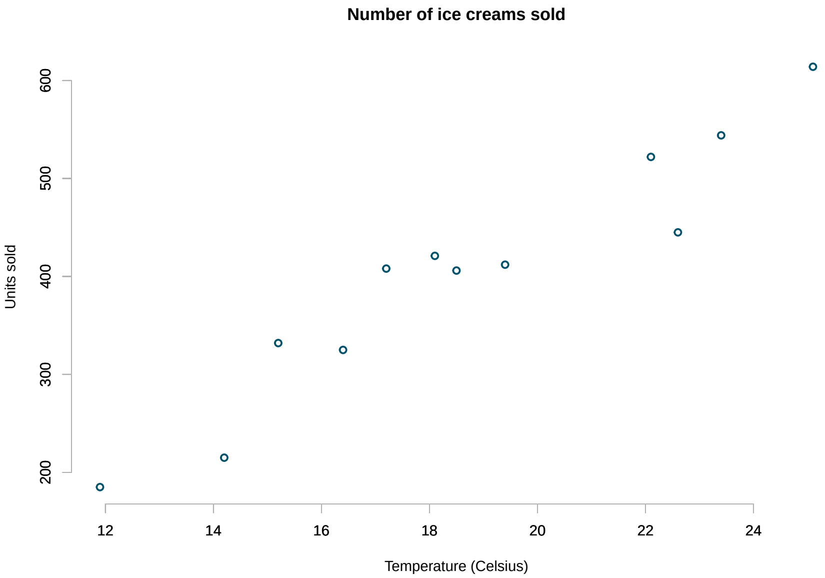
```
icecream <- data.frame(
  temp=c(11.9, 14.2, 15.2, 16.4, 17.2, 18.1,
         18.5, 19.4, 22.1, 22.6, 23.4, 25.1),
  units=c(185L, 215L, 332L, 325L, 408L, 421L,
          406L, 412L, 522L, 445L, 544L, 614L)
  )
```

Helper function for plots

```r
basicPlot <- function(...){
  plot(units ~ temp, data=icecream, bty="n", lwd=2,
       main="Number of ice creams sold", col="#00526D",
       xlab="Temperature (Celsius)",
       ylab="Units sold", ...)
  axis(side = 1, col="grey")
  axis(side = 2, col="grey")
}
```

Will allow for consistent graphical outputs across plots.

```
basicPlot()
```



Number of ice creams sold
Units sold
600
500
400
300
200
12
14
16
18
20
22
24
Temperature (Celsius)

Goal: build a good predictive model, even *outside the range of available data*

Particularly interested in how model performs in extreme cases (say temperature is 1 or 32 Celsius)

Reminder: Marginal (Prior Predictive) Distribution

Before data are observed, the distribution of the *unknown but observable* y is

$$p(y) = \int p(y, \theta) d\theta = \int p(\theta) p(y|\theta) d\theta$$

the marginal distribution of y

- some call it the prior predictive distribution
- *prior* since it is not conditional on a previous observation of the process
- *predictive* since it is the distribution of an observable quantity

Posterior Predictive Distribution

After data y observed, can predict *unknown but observable* $\tilde{y}$ from the same process.

- distribution of $\tilde{y}$ called posterior preditive distribution

$$
\begin{aligned}
p(\tilde{y}|y) &= \int p(\tilde{y}, \theta|y) d\theta \\
&= \int p(\tilde{y}|\theta, y) p(\theta|y) d\theta \\
&= \int p(\tilde{y}|\theta) p(\theta|y) d\theta.
\end{aligned}
$$

- *posterior* because it is conditional on observed y
- *predictive* because $\tilde{y}$ is observable

Second line: posterior predictive as average of conditional predictions over posterior of θ; last line follows by (assumed) conditional independence of y, $\tilde{y}$ given θ.

Prediction and Regression

Recall the usual linear regression model:

$$y|\beta, \sigma, X \sim \mathcal{N}(X\beta, \sigma^2 I_n)$$

Suppose we have observed $\tilde{X}$ for a new observation, and want to predict $\tilde{y}$.

- posterior predictive has two sources of uncertainty:
  1. unexplained variability, not accounted for by the observable part of the model $X\beta$; measured by σ^2
  2. posterior uncertainty about β, σ due to finite sample size of y
- as $n \to \infty$, posterior uncertainty of (β, σ^2) goes to zero, but predictive uncertainty remains

To sample $\tilde{y}$ from posterior predictive $p(\tilde{y}|y)$, first sample (β, σ) from joint posterior $p(\beta, \sigma|y)$, then draw $\tilde{y} \sim \mathcal{N}(\tilde{X}\beta, \sigma^2 I_n)$.

(Conditional) Mean of posterior predictive

Consider the conditional posterior predictive $p(\tilde{y}|\sigma, y)$ and then average over the posterior uncertainty in $\sigma|y$.

- given σ,

$$\begin{aligned} E(\tilde{y}|\sigma, y) &= E(E(\tilde{y}|\beta, \sigma, y)|\sigma, y) \\ &= E(\tilde{X}\beta|\sigma, y) \\ &= \tilde{X}\hat{\beta} \end{aligned}$$

Inner expectation averages over $\tilde{y}|\beta$; outer expectation averages over β. Conditioning on X and $\tilde{X}$ implicit.

(Conditional) Variance of posterior predictive

$$
\begin{aligned}
\mathrm{var}(\tilde{y}|\sigma, y) &= E[\mathrm{var}(\tilde{y}|\beta, \sigma, y)|\sigma, y] + \mathrm{var}[E(\tilde{y}|\beta, \sigma, y)|\sigma, y] \\
&= E[\sigma^2 I|\sigma, y] + \mathrm{var}[\tilde{X}\beta|\sigma, y] \\
&= \sigma^2 I + \tilde{X} V_\beta \tilde{X}^T \sigma^2
\end{aligned}
$$

- first term represents sampling variation
- second term represents uncertainty about β

Motivating Linear Model (version 1)

Assume $\tilde{y}_i \sim \mathcal{N}(\mu_i, \sigma^2)$.

- on different days with same temperature, ice cream sales can differ, but on average:

$$E(y_i) = \mu_i = \alpha + \beta x_i, \;\; \forall i$$

Makes clear that the observation is just *one realization* from a distribution, and that the parameter is modeled *linearly*.

Motivating Linear Model (version 2)

Alternatively, assume the residuals $\varepsilon_i = y_i - \mu_i \sim \mathcal{N}(0, \sigma^2)$

- then the statement

$$E(y_i) = \mu_i = \alpha + \beta x_i$$

  represents the belief that $E(y_i)$ is identical to the parameter (μ_i) of the underlying distribution and the variance is constant.

Equivalently,

$$y_i = \alpha + \beta x_i + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

Motivation of GLM

The classical linear model above predicts $E(y_i)$ is a linear combination of predictors.

- implies constant change in predictor leads to constant change in response
- appropriate when y_i continuous and (at least approximately) normal
- inappropriate for many types of response variables

binary	credit worthiness of loan applicant (yes/no)
categorical	full-time, part-time or unemployed
count variable	number of cases of illness within a certain time period

Moreover, some continuous responses have features which cause problems for the linear model: e.g. highly-skewed variables (income, life span, amount of damages).

- sometimes can be transformed to be approximately symmetric
- often better to take a different approach

Motivation of generalized linear models (GLMs): unify different approaches to regression modeling for responses which are not necessarily normal

GLM Framework

A GLM has 3 components:

Distribution for response (often exponential family).

Linear predictor $\eta = X\beta$.

Link function $g(\cdot)$ such that $E(y) = \mu = g^{-1}(\eta)$.

Exponential families are easier to work with (can make variance calculation easier).

- GLMs fit by iteratively reweighted least squares
- posterior usually not available in closed form; typically fit by Laplace approximation or MCMC

Example: Linear Model

The classical linear model is also a GLM.

$$y_i = \beta_0 + \beta x_i + \varepsilon_i,$$

$$E(y_i) = \beta_0 + \beta x_i$$

Distribution $y_i \sim \mathcal{N}(\mu_i, \sigma^2)$

Linear predictor (continuous or discrete) and linear in parameters; predictors can also be transformed (still linear in transformed predictors)

Link function identity: $\eta = g(E(y_i)) = g(\mu_i) = E(y_i)$: modeling the mean directly, the simplest link function

Logit and Logistic Functions

The logit function of a number $q \in (0, 1)$ is

$$\text{logit}(q) = \log\left(\frac{q}{1-q}\right) = \log(q) - \log(1-q).$$

Hence when q is a probability, with $q/(1-q)$ the **odds**, $\text{logit}(q)$ is the log odds.

The logistic function of any number $d \in \mathbb{R}$ is the inverse logit function

$$\text{logit}^{-1}(d) = \text{logistic}(d) = \frac{1}{1+e^{-d}} = \frac{e^d}{1+e^d}.$$

The logit maps from $(0, 1) \mapsto \mathbb{R}$; the inverse logit (logistic) maps from $\mathbb{R} \mapsto (0, 1)$.

Example: Binary Logistic Regression

Models binary response y as function of k explanatory variables $X = (X_1, \ldots, X_k)$ (note we need values between 0 and 1)

$$\mathrm{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta x_i,$$

where $\pi_i = E(y_i) = \Pr(y_i = 1)$.

Distribution $y_i \sim \mathrm{Binomial}(n, \pi)$ with π the probability of success

Linear predictor same as above; predictors can be transformed as usual

Link function logit link

$$\eta = \mathrm{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right)$$

More generally, the logit link is a model for the log odds of the mean, and the mean here is π

Comment on Logit Model

Results from choosing the *logistic response function*

$$\pi = h(\eta) = \frac{\exp(\eta)}{1 + \exp(\eta)},$$

or equivalently the logit link function

$$g(\pi) = \log\left(\frac{\pi}{1-\pi}\right) = \eta = \beta_0 + \beta x$$

- yields a linear model for the log-odds $\log(\pi/(1-\pi))$
- transformation with the exponential function gives

  $$\frac{\pi}{1-\pi} = \exp(\beta_0)\exp(\beta_1 x_1)\cdots(\exp\beta_k x_k)$$

  implying the predictors affect the odds $\pi/(1-\pi)$ in an exponential-multiplicative manner.

Example: Log-Linear Poisson Model for Count Data Response

Distribution $y_i \sim \text{Poisson}(\lambda)$

Linear predictor same as above

Link function $\eta = \log \lambda = E(y)$

Connects the rate $\lambda_i = E(y_i)$ of Poisson distribution with linear predictor $\eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}$ using

$$\lambda_i = \exp(\eta_i) = \exp(\beta_0)\exp(\beta_1 x_{i1}) \cdots \exp(\beta_k x_{ik})$$

or in log-linear form through

$$\log(\lambda_i) = \eta_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}$$

Effect of covariates on the rate λ is exponentially multiplicative.

Overdispersion

In practice, the empirical variance (i.e. the estimate of the variance of the response) is larger than that *predicted by the assumed distribution*. In such cases, the model is said to be overdispersed.

Binomial model predicts

$$\mathrm{var}(y_i) = \tau \pi_i (1 - \pi_i)$$

when $\tau = 1$ it is the standard binomial model; when $\tau > 1$ the model exhibits overdispersion

Poisson model predicts

$$\mathrm{var}(y_i) = \tau \lambda_i$$

and $\tau > 1$ implies overdispersion

Reasons for overdispersion:

1. unobserved heterogeneity which is not explained by observed covariates
2. positive correlations between response variable observations, e.g. in binary response when several observations belong to the same cluster (such as a household)

Linear Fit with glm

```r
lin.mod <- glm(units ~ temp, data=icecream,
               family=gaussian(link="identity"))
library(arm) # for 'display' function only
display(lin.mod)
```

```
glm(formula = units ~ temp, family = gaussian(link = "identity"),
    data = icecream)
            coef.est coef.se
(Intercept) -159.47    54.64
temp          30.09     2.87
---
  n = 12, k = 2
 residual deviance = 14536.3, null deviance = 174754.9 (difference
  overdispersion parameter = 1453.6
  residual sd is sqrt(overdispersion) = 38.13
```

Linear model appears fine for range of temps observed, but doesn't make sense for 0 Celsius.

Log-transformed linear model

Perhaps transform data to ensure only positive values. Change model to

$$\log(y_i) \sim \mathcal{N}(\mu_i, \sigma^2), \quad E[\log(y_i)] = \mu_i = \alpha + \beta x_i$$

Implies sales follow log-normal distribution: $y_i \sim \log\mathcal{N}(\mu_i, \sigma^2)$, and $E(y_i) = \exp(\mu_i + \sigma^2/2) = \exp(\alpha + \beta x_i + \sigma^2/2)$.

- log-normal distribution skewed to right
- hence, higher sales values more likely than lower sales

Although model is linear on log-scale, have to remember to transform predictions back to original scale since $E[\log(y_i)] \neq \log(E[y_i])$.

Log-linear Fit

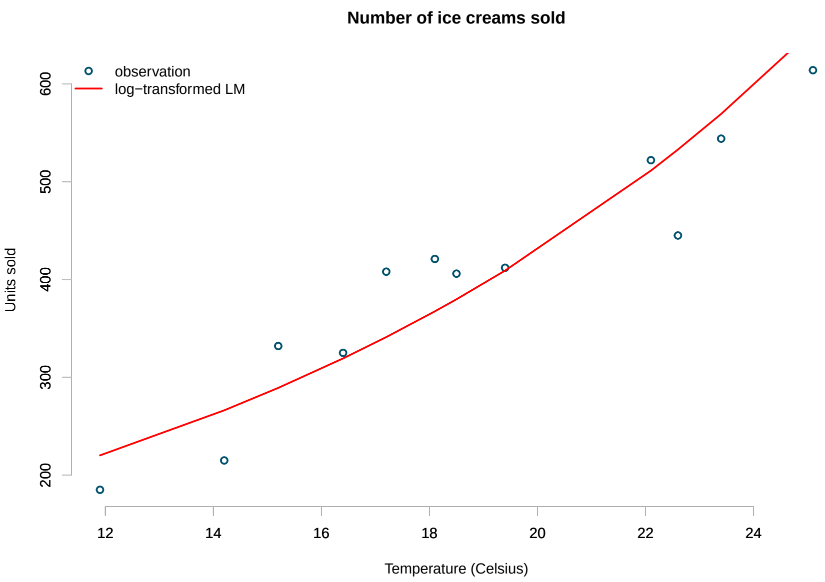
```r
log.lin.mod <- glm(log(units) ~ temp, data=icecream,
              family=gaussian(link="identity"))
display(log.lin.mod)
```

```
glm(formula = log(units) ~ temp, family = gaussian(link = "identity
    data = icecream)
            coef.est coef.se
(Intercept) 4.40     0.20
temp        0.08     0.01
---
  n = 12, k = 2
  residual deviance = 0.2, null deviance = 1.4 (difference = 1.2)
  overdispersion parameter = 0.0
  residual sd is sqrt(overdispersion) = 0.14
```

Plot

```
log.lin.sig <- summary(log.lin.mod)$dispersion
log.lin.pred <- exp(predict(log.lin.mod) + 0.5*log.lin.sig)
basicPlot()
lines(icecream$temp, log.lin.pred, col="red", lwd=2)
legend(x="topleft", bty="n", lwd=c(2,2), lty=c(NA,1),
       legend=c("observation", "log-transformed LM"),
       col=c("#00526D","red"), pch=c(1,NA))
```

Predicts sales at 82 for temp=0. However, still predicts too many sales at low and high-end of temp range. Moreover, sales are discrete but model predicts continuous values.



Number of ice creams sold
observation
log−transformed LM
Units sold
200
300
400
500
600
12
14
16
18
20
22
24
Temperature (Celsius)

Poisson regression

Poisson distribution has one parameter λ_i, also its expected value. Regard logarithm of λ_i as linear function of predictors.

- different from log-transformed linear model above: there the original data was transformed; here the expected value of the data is transformed
- also this will generate discrete values for y_i

$$\begin{aligned} y_i &\sim \text{Poisson}(\lambda_i) \\ E(y_i) &= \mu_i = \lambda_i = \exp(\alpha + \beta x_i) = \exp(\alpha)\exp(\beta x_i) \\ \log(\mu_i) &= \alpha + \beta x_i \end{aligned}$$

Poisson Fit

```
pois.mod <- glm(units ~ temp, data=icecream,
                family=poisson(link="log"))
display(pois.mod)
```

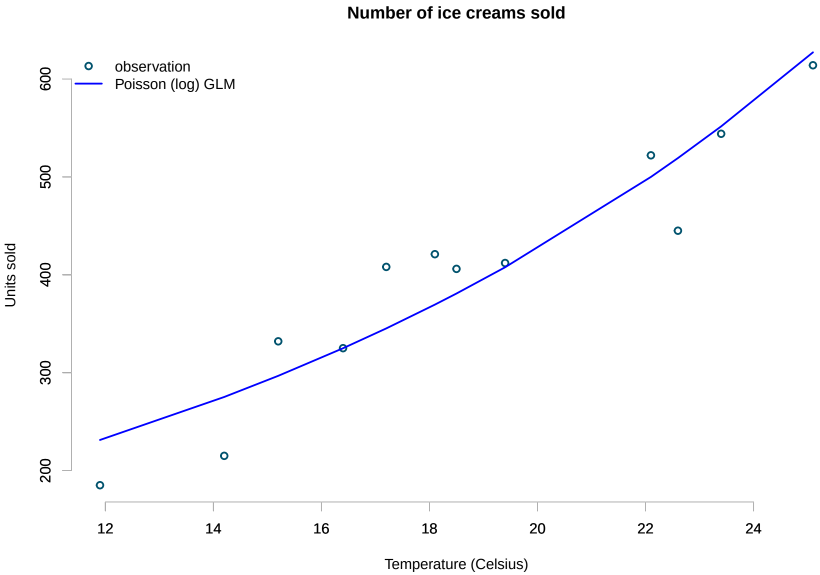
```
glm(formula = units ~ temp, family = poisson(link = "log"), data = i
            coef.est coef.se
(Intercept) 4.54     0.08
temp        0.08     0.00
---
  n = 12, k = 2
 residual deviance = 60.0, null deviance = 460.1 (difference = 400.
```

Need to use exp to predict sales for given temp.; `R` does this by setting `predict` statement to `type="response"`.

- at 0, expect to sell $\exp(4.45) = 94$ units
- increasing temp. by 1 yields predicted sales increase of $\exp(0.076) - 1 = 7.9\%$
- exponential function turns additive scale into multiplicative one

Plot

```r
pois.pred <- predict(pois.mod, type="response")
basicPlot()
lines(icecream$temp, pois.pred, col="blue", lwd=2)
legend(x="topleft", bty="n", lwd=c(2,2), lty=c(NA,1),
       legend=c("observation", "Poisson (log) GLM"),
       col=c("#00526D","blue"), pch=c(1,NA))
```


Number of ice creams sold
observation
Poisson (log) GLM
Units sold
200
300
400
500
600
12
14
16
18
20
22
24
Temperature (Celsius)

Criticism

Note the following problem:

```r
predict(pois.mod,newdata=data.frame(temp=32),
        type="response")
```

```
       1
1056.651
```

Perhaps this exponential growth is too optimistic; maybe the market demand will not exceed, say 800, for any temperature.

Suggestion

Let's try binomial regression.

- assume market is saturated at 800 sales; model the proportion sold (out of 800) at a given temperature
- suggests binomial distribution for # of 'successful' sales out of 800
- parameter is the probability that someone will buy ice cream as a function of temperature

Divide sales data by 800 to get proxy for probability of selling all ice cream at a given temperature.

- need an S-shaped curve which maps sales data into probabilities between 0 and 1

Binomial Regression

Use the logistic function; for $u \in \mathbb{R}$, the logistic function of u is the inverse logit

$$\text{logit}^{-1}(u) = \frac{e^u}{e^u + 1} = \frac{1}{1 + e^{-u}}.$$

The logit maps from $(0, 1) \mapsto \mathbb{R}$; the inverse logit (logistic) maps from $\mathbb{R} \mapsto (0, 1)$.

Model is summarized by

$$\begin{aligned} y_i &\sim \text{Binom}(n, \mu_i = \pi_i) \\ E(y_i) &= \mu_i = \text{logit}^{-1}(\alpha + \beta x_i) \\ \text{logit}(\mu_i) &= \alpha + \beta x_i \end{aligned}$$

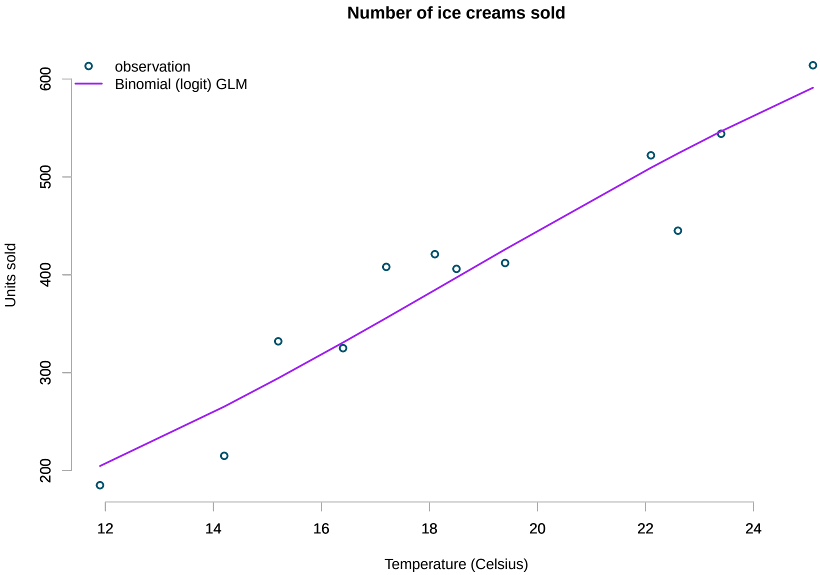
Binomial regression fit

```
market.size <- 800
icecream$opportunity <- market.size - icecream$units
bin.glm <- glm(cbind(units, opportunity) ~ temp, data=icecream,
    family=binomial(link = "logit"))
display(bin.glm)
```

```
glm(formula = cbind(units, opportunity) ~ temp, family = binomial(l
    data = icecream)
            coef.est coef.se
(Intercept) -2.97     0.11
temp         0.16     0.01
---
  n = 12, k = 2
  residual deviance = 84.4, null deviance = 909.4 (difference = 825.
```

Plot

```
bin.pred <- predict(bin.glm, type="response")*market.size
basicPlot()
lines(icecream$temp, bin.pred, col="purple", lwd=2)
legend(x="topleft", bty="n", lwd=c(2,2), lty=c(NA,1),
       legend=c("observation", "Binomial (logit) GLM"),
       col=c("#00526D","purple"),  pch=c(1,NA))
```


Number of ice creams sold
observation
Binomial (logit) GLM
Units sold
200
300
400
500
600
12
14
16
18
20
22
24
Temperature (Celsius)

Predicting sales at 0 and 35 Celsius

R function plogis: inverse of logistic function

```
# Sales at 0 Celsius
plogis(coef(bin.glm)[1])*market.size
```

```
(Intercept)
   39.09618
```

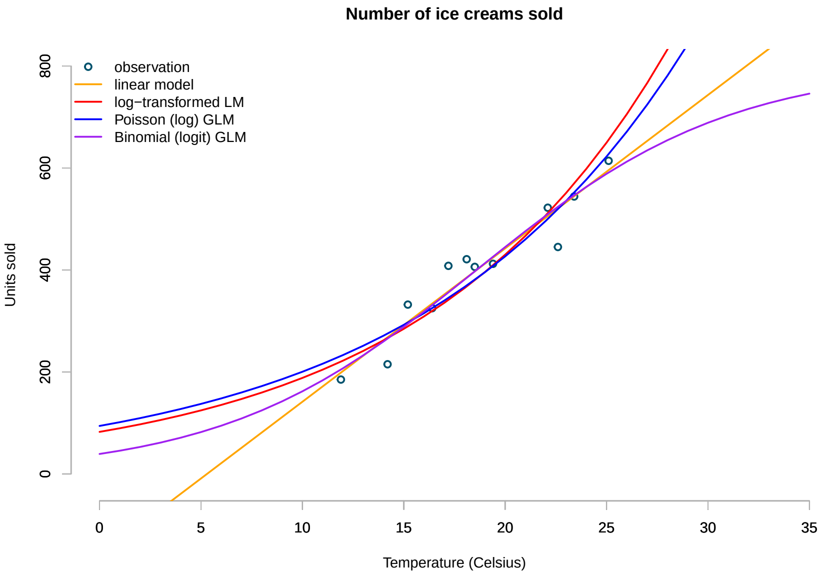
```
# Sales at 35 Celsius
plogis(coef(bin.glm)[1]+coef(bin.glm)[2]*35)*market.size
```

```
(Intercept)
   745.7449
```

These results will change if we change the assumption about the market saturation point, e.g. if we use a maximum value of 1000 sales, these predictions change to 55 and 846.

Plotting together

```
temp <- 0:35
p.lm <- predict(lin.mod, data.frame(temp=temp), type="response")
p.log.lm <- exp(predict(log.lin.mod, data.frame(temp=0:35), type="response") +
                  0.5 * summary(log.lin.mod)$dispersion)
p.pois <- predict(pois.mod, data.frame(temp=temp), type="response")
p.bin <- predict(bin.glm, data.frame(temp=temp), type="response")*market.size
basicPlot(xlim=range(temp), ylim=c(-20,market.size))
lines(temp, p.lm, type="l", col="orange", lwd=2)
lines(temp, p.log.lm, type="l", col="red", lwd=2)
lines(temp, p.pois, type="l", col="blue", lwd=2)
lines(temp, p.bin, type="l", col="purple", lwd=2)
legend(x="topleft",
       legend=c("observation",
                "linear model",
                "log-transformed LM",
                "Poisson (log) GLM",
                "Binomial (logit) GLM"),
       col=c("#00526D","orange", "red",
             "blue", "purple"),
       bty="n", lwd=rep(2,5),
       lty=c(NA,rep(1,4)),
       pch=c(1,rep(NA,4)))
```

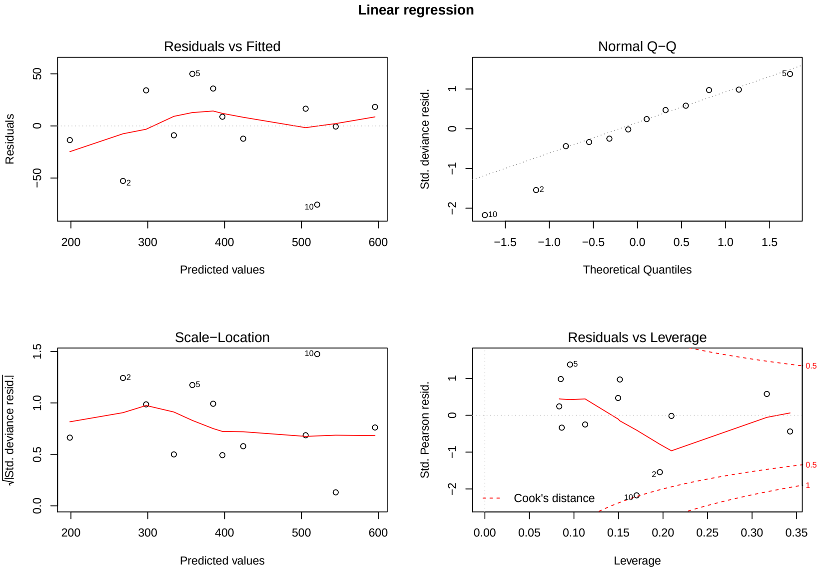

Number of ice creams sold
observation
linear model
log−transformed LM
Poisson (log) GLM
Binomial (logit) GLM
Units sold
0
200
400
600
800
0
5
10
15
20
25
30
35
Temperature (Celsius)

Visualizing GLMs (function to produce 3d plots)

```
glmModelPlot <- function(x, y, xlim,ylim, meanPred,  LwPred, UpPred,
                         plotData, main=NULL){
  ## Based on code by Arthur Charpentier:
  ## http://freakonometrics.hypotheses.org/9593
  par(mfrow=c(1,1))
  n <- 2
  N <- length(meanPred)
  zMax <- max(unlist(sapply(plotData, "[[", "z")))*1.5
  mat <- persp(xlim, ylim, matrix(0, n, n), main=main,
               zlim=c(0, zMax), theta=-30,
               ticktype="detailed",box=FALSE)
  C <- trans3d(x, UpPred, rep(0, N),mat)
  lines(C, lty=2)
  C <- trans3d(x, LwPred, rep(0, N), mat)
  lines(C, lty=2)
  C <- trans3d(c(x, rev(x)), c(UpPred, rev(LwPred)),
               rep(0, 2*N), mat)
  polygon(C, border=NA, col=adjustcolor("yellow", alpha.f = 0.5))
  C <- trans3d(x, meanPred, rep(0, N), mat)
  lines(C, lwd=2, col="grey")
  C <- trans3d(x, y, rep(0,N), mat)
  points(C, lwd=2, col="#00526D")
  for(j in N:1){
    xp <- plotData[[j]]$x
    yp <- plotData[[j]]$y
    z0 <- plotData[[j]]$z0
    zp <- plotData[[j]]$z
    C <- trans3d(c(xp, xp), c(yp, rev(yp)), c(zp, z0), mat)
    polygon(C, border=NA, col="light blue", density=40)
    C <- trans3d(xp, yp, z0, mat)
    lines(C, lty=2)
    C <- trans3d(xp, yp, zp, mat)
    lines(C, col=adjustcolor("blue", alpha.f = 0.5))
```

```
xlim <- c(min(icecream$temp)*0.95, max(icecream$temp)*1.05)
ylim <- c(floor(min(icecream$units)*0.95),
          ceiling(max(icecream$units)*1.05))

lin.mod <- glm(units ~ temp, data=icecream,
                  family=gaussian(link="identity"))
```

```
par(mfrow=c(2,2))
plot(lin.mod)
title(outer=TRUE, line = -1,
      main = list("Linear regression",
                  cex=1.25,col="black", font=2))
```



Observations 2, 5 and 10 are extreme points; will show up on the outer edge or outside of the yellow area on plot below.

Creating 3D plot

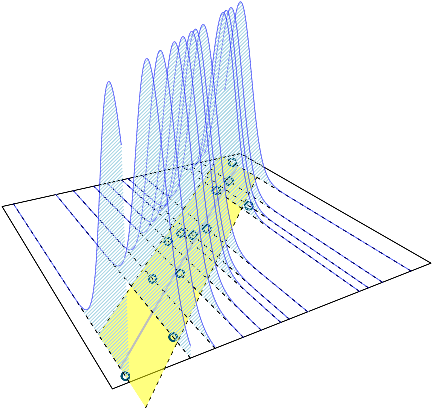
```
meanPred <- predict(lin.mod, type="response")
sdgig <- sqrt(summary(lin.mod)$dispersion)

UpPred <- qnorm(.95, meanPred, sdgig)
LwPred <- qnorm(.05, meanPred, sdgig)

plotData <- lapply(
seq(along=icecream$temp),
  function(i){
    stp <- 251
    x = rep(icecream$temp[i], stp)
    y = seq(ylim[1], ylim[2], length=stp)
    z0 = rep(0, stp)
    z = dnorm(y, meanPred[i], sdgig)
    return(list(x=x, y=y, z0=z0, z=z))
  }
)
```

Linear Model 3D Plot

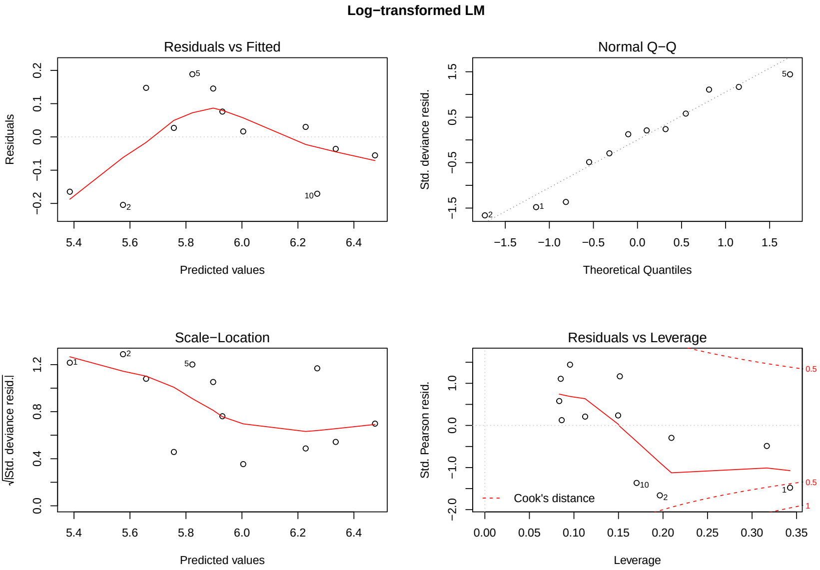
```r
glmModelPlot(x = icecream$temp, y=icecream$units,
             xlim=xlim, ylim=ylim,
             meanPred = meanPred, LwPred = LwPred,
             UpPred = UpPred, plotData = plotData,
             main = "Linear regression")
```

Observations are circles on xy-plane; mean predictions are the solid line. Theoretical residual interval between 5th and 95th quantile of normal distributions parameterized by model output shown in yellow. Density of the response at each observation in blue.

Linear regression



Log-transformed linear model plots



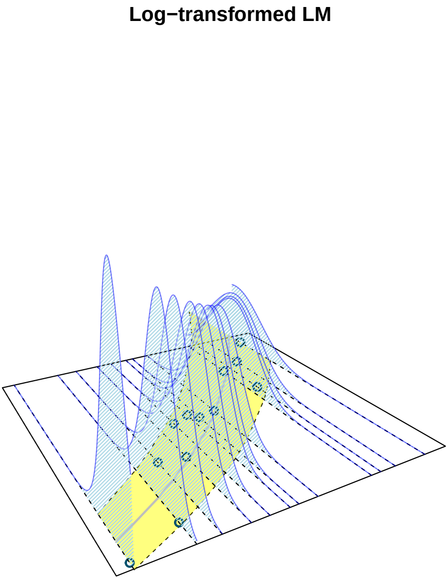
Comment

Shows increasing variance around mean as temperatures increase; this is a property of log-normal distribution. Variance is given by

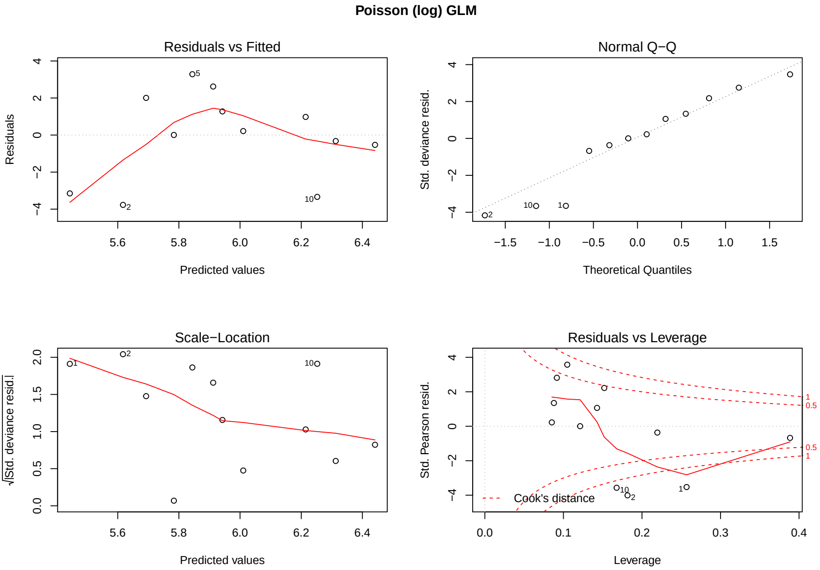
$$\mathrm{var}(X) = \exp(2\mu + \sigma^2)(\exp(\sigma^2) - 1) = E[X]^2(\exp(\sigma^2) - 1).$$

- also finds points 2,5 and 10 as extreme values
- first residual plot shows the log-transformed model over-predicts for colder temperatures

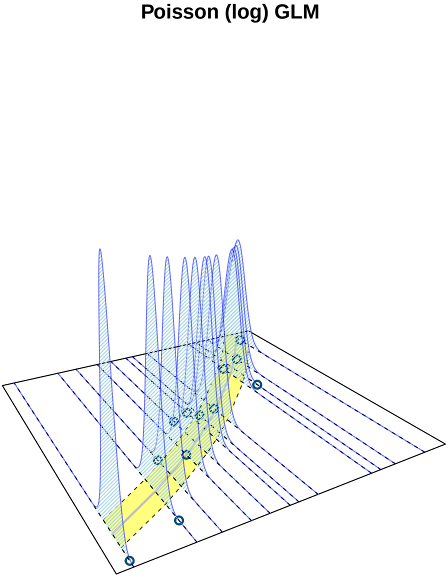
Log-transformed linear model 3D plot



Poisson regression plots



Poisson regression 3D plot

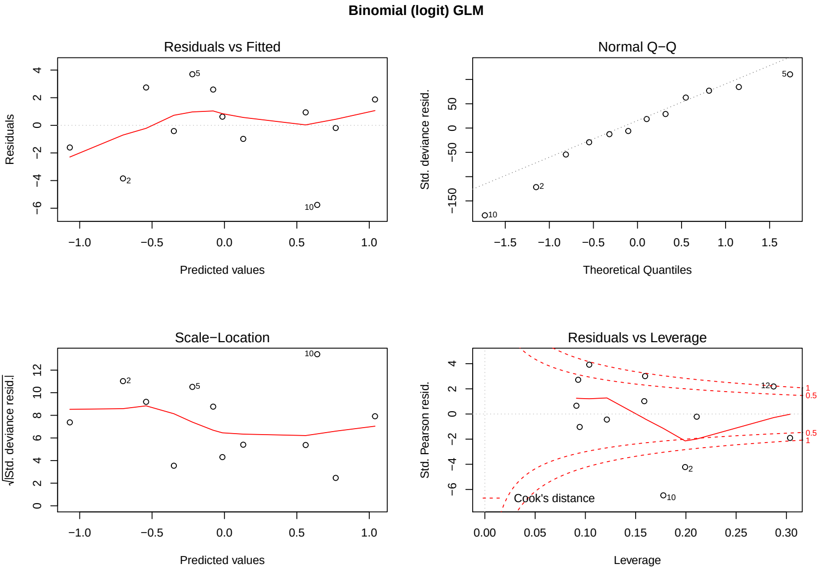


Comment

Shows narrower range between 5th and 95th quantile compared to previous models.

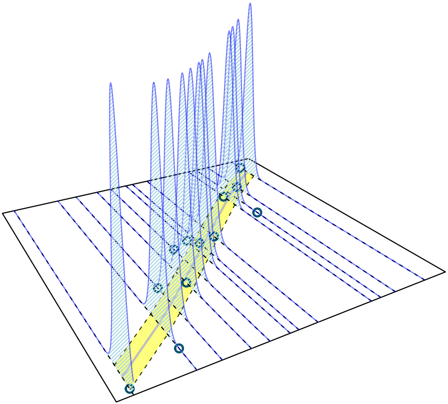
- didn't assume overdispersion, so the yellow range increases with the mean
- same points 2,5 and 10 seen as extreme, but now are outside the yellow area

Binomial logit model plots



Binomial logit model 3D plot

Binomial (logit) GLM



Visualizing Posterior Predictive Distribution of Log-Transformed Model

We use `rstan` to model the sales statistics and generate samples from the posterior predictive.

- we are interested in a 95% prediction interval, which will be wider than the theoretical 95% interval since it takes into account parameter uncertainty as well (due to finite sample computation of posterior)

First we take the log-transformed linear model and turn it into a `Stan` model and include a chunk to generate output from the posterior predictive distribution.

```
stanLogTransformed <-"
data {
  int N;
  vector[N] units;
  vector[N] temp;
}
transformed data {
  vector[N] log_units;
  log_units <- log(units);
}
parameters {
  real alpha;
  real beta;
  real tau;
}
transformed parameters {
  real sigma;
  sigma <- 1.0 / sqrt(tau);
}
model{
  // Model
  log_units ~ normal(alpha + beta * temp, sigma);
  // Priors
  alpha ~ normal(0.0, 1000.0);
  beta ~ normal(0.0, 1000.0);
  tau ~ gamma(0.001, 0.001);
}
generated quantities{
  vector[N] units_pred;
  for(i in 1:N)
    units_pred[i] <- exp(normal_rng(alpha + beta * temp[i], sigma));
}
"
```

```r
temp <- c(11.9,14.2,15.2,16.4,17.2,18.1,18.5,19.4,22.1,22.6,23.4
units <- c(185L,215L,332L,325L,408L,421L,406L,412L,522L,445L,544
library(rstan)
stanmodel <- stan_model(model_code = stanLogTransformed)
fit <- sampling(stanmodel,
                data = list(N=length(units),
                            units=units,
                            temp=temp),
                iter = 1000, warmup=200)
```

```
SAMPLING FOR MODEL '8da2b955fe5507dee64b749cacbbeeb9' NOW (CHAIN 1

Chain 1, Iteration:   1 / 1000 [  0%]  (Warmup)
Chain 1, Iteration: 100 / 1000 [ 10%]  (Warmup)
Chain 1, Iteration: 200 / 1000 [ 20%]  (Warmup)
Chain 1, Iteration: 201 / 1000 [ 20%]  (Sampling)
Chain 1, Iteration: 300 / 1000 [ 30%]  (Sampling)
Chain 1, Iteration: 400 / 1000 [ 40%]  (Sampling)
Chain 1, Iteration: 500 / 1000 [ 50%]  (Sampling)
Chain 1, Iteration: 600 / 1000 [ 60%]  (Sampling)
Chain 1, Iteration: 700 / 1000 [ 70%]  (Sampling)
```

```r
## Extract generated posterior predictive quantities
Sims <- data.frame(stanoutput[["units_pred"]])

## Calculate summary statistics
SummarySims <- apply(Sims, 2, summary)
colnames(SummarySims) <- paste(icecream$temp,"ºC")

## Extract estimated parameters
(parms <- sapply(stanoutput[c("alpha", "beta", "sigma")], mean))
```

```
     alpha       beta      sigma
4.39353561 0.08301128 0.14961131
```

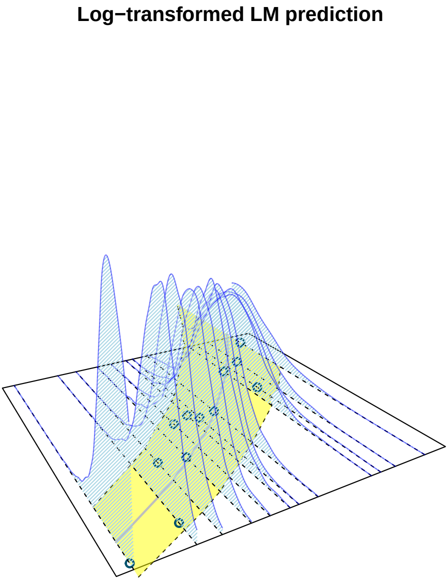
```r
## Use parameters to predict median and mean
PredMedian <- exp(parms['alpha'] + parms['beta']*temp)
PredMean <- exp(parms['alpha'] + parms['beta']*temp + 0.5*parms[
```

Compare predictions based on parameters with simulation statistics

```r
round(rbind(SummarySims, PredMedian, PredMean),1)
```

```
           11.9 ºC 14.2 ºC 15.2 ºC 16.4 ºC 17.2 ºC 18.1 ºC 18.5 ºC 19.4 ºC
Min.          94.5   108.9   122.6   129.9   113.2   189.0   176.4   164.0
1st Qu.      194.2   236.9   258.7   285.2   304.9   328.6   337.6   366.9
Median       218.1   263.9   287.1   315.9   337.9   361.3   372.7   404.7
Mean         220.9   267.5   289.7   319.9   342.1   366.9   379.0   410.7
3rd Qu.      243.5   294.3   317.3   349.7   374.0   400.6   413.9   447.8
Max.         482.1   544.7   526.5   754.4   618.5   734.0   785.6   878.3
PredMedian   217.3   263.0   285.8   315.7   337.4   363.6   375.9   405.0
PredMean     219.8   266.0   289.0   319.3   341.2   367.7   380.1   409.6
           22.1 ºC 22.6 ºC 23.4 ºC 25.1 ºC
Min.         248.4   225.2   288.8   307.3
1st Qu.      454.7   477.6   506.0   584.4
Median       505.1   529.9   565.2   648.0
Mean         513.2   539.5   572.0   661.2
3rd Qu.      563.4   591.0   629.4   727.1
Max.        1307.0  1046.0  1961.0  2107.0
PredMedian   506.8   528.3   564.5   650.1
PredMean     512.5   534.2   570.9   657.4
```

Posterior Predictive 3D plot



Bayesian GLMs

We use **brms** package to create prediction intervals for the four GLM models discussed above. Want to predict how much ice cream should be kept in stock when temperature is 35, such that you only run out of ice cream with posterior probability 2.5%.

```
library(brms)
# Linear Gaussian model
lin.mod <- brm(units ~ temp, family="gaussian")
```

```
SAMPLING FOR MODEL 'gaussian(identity) brms-model' NOW (CHAIN 1).

Chain 1, Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1, Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1, Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1, Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1, Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1, Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1, Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1, Iteration: 1200 / 2000 [ 60%]  (Sampling)
```

Log-transformed linear model using brms (similar to Stan coding above)

```
log.lin.mod
```

```
 Family: gaussian (identity)
Formula: log_units ~ temp
   Data: NULL (Number of observations: 12)
Samples: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup samples = 4000
   WAIC: Not computed

Fixed Effects:
          Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
Intercept     4.40      0.24     3.93     4.88       1627    1
temp          0.08      0.01     0.06     0.11       1589    1

Family Specific Parameters:
              Estimate Est.Error l-95% CI u-95% CI Eff.Sample Rhat
sigma(log_units)    0.16      0.04      0.1     0.26       1310    1

Samples were drawn using sampling(NUTS). For each parameter, Eff.Sample
is a crude measure of effective sample size, and Rhat is the potential
```

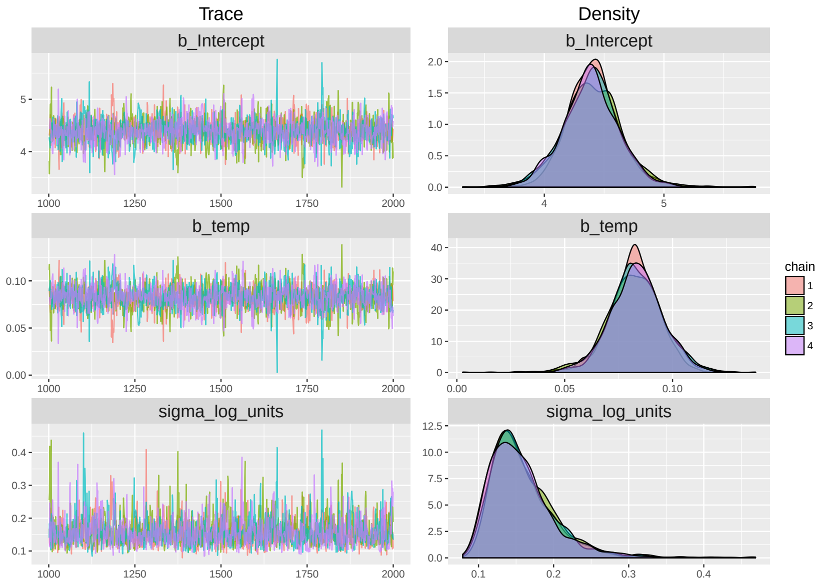
Comment

Here the prior for σ is Cauchy, whereas in the coding above it was inverse gamma.

- should imply a small difference in σ as compared to above

Now we give traceplots and density plots for the MCMC samples.

Plot

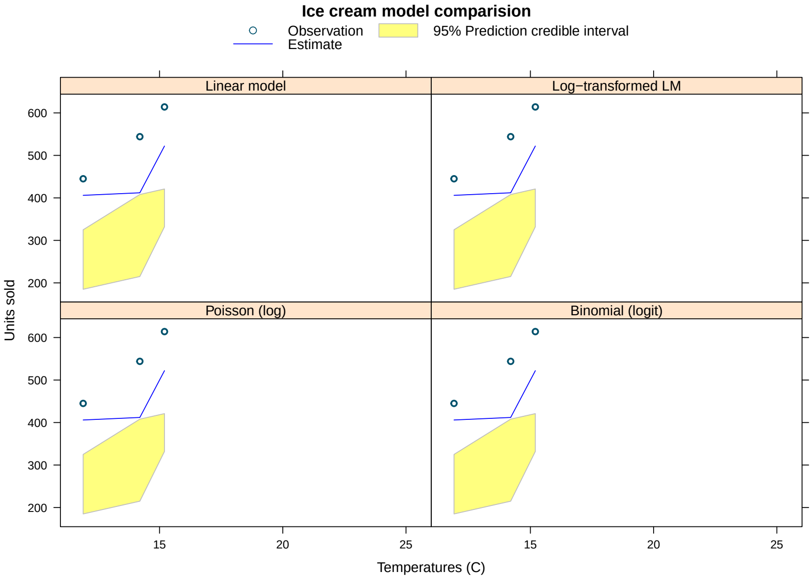


Combine prediction credible intervals in data frame (for single plot)

```
modelData <- data.frame(
  Model=factor(c(rep("Linear model", n),
                 rep("Log-transformed LM", n),
                 rep("Poisson (log)",n),
                 rep("Binomial (logit)",n)),
               levels=c("Linear model",
                        "Log-transformed LM",
                        "Poisson (log)",
                        "Binomial (logit)"),
               ordered = TRUE),
  Temperature=rep(temp, 4),
  Units_sold=rep(units, 4),
  rbind(predict(lin.mod),
        exp(predict(log.lin.mod) +
              0.5 * mean(extract(log.lin.mod$fit)[["sigma_log_units"]])),
        predict(pois.mod),
        predict(bin.mod)
  ))
```

Creating plot

```
library(lattice)
key <- list(
  rep=FALSE,
  lines=list(col=c("#00526D", "blue"), type=c("p","l"), pch=1),
  text=list(lab=c("Observation","Estimate")),
  rectangles = list(col=adjustcolor("yellow", alpha.f=0.5), border="gre
  text=list(lab="95% Prediction credible interval"))
xyplot(l.95..CI + u.95..CI + Estimate + Units_sold ~ Temperature | Mode
       data=modelData, as.table=TRUE, main="Ice cream model comparision
       xlab="Temperatures (C)", ylab="Units sold",
       scales=list(alternating=1), key=key,
       panel=function(x, y){
         n <- length(x)
         k <- n/2
         upper <- y[(k/2+1):k]
         lower <- y[1:(k/2)]
         x <- x[1:(k/2)]
         panel.polygon(c(x, rev(x)), c(upper, rev(lower)),
                       col = adjustcolor("yellow", alpha.f = 0.5),
                       border = "grey")
         panel.lines(x, y[(k+1):(k+n/4)], col="blue")
         panel.points(x, y[(n*3/4+1):n], lwd=2, col="#00526D")
```


Ice cream model comparision
Observation
Estimate
95% Prediction credible interval
Linear model
Log−transformed LM
Poisson (log)
Binomial (logit)
Units sold
200
300
400
500
600
15
20
25
Temperatures (C)

How much ice cream stock to hold on hot day

Set probability of selling out at 2.5%; want to have enough ice cream to meet demand with posterior probability 97.5%. Need 97.5% percentile of posterior predictive MCMC samples.

```
A <- function(samples){
  as.matrix(samples[,c("b_Intercept" ,"b_temp")])
}
x <- c(1, 35)
prob <- 0.975
lin.samples <- posterior_samples(lin.mod)
n <- nrow(lin.samples)
mu <- A(lin.samples) %*% x
sigma <- lin.samples[,"sigma_units"]
(lin.q <- quantile(rnorm(n, mu, sigma), prob))
```

```
   97.5%
1032.851
```

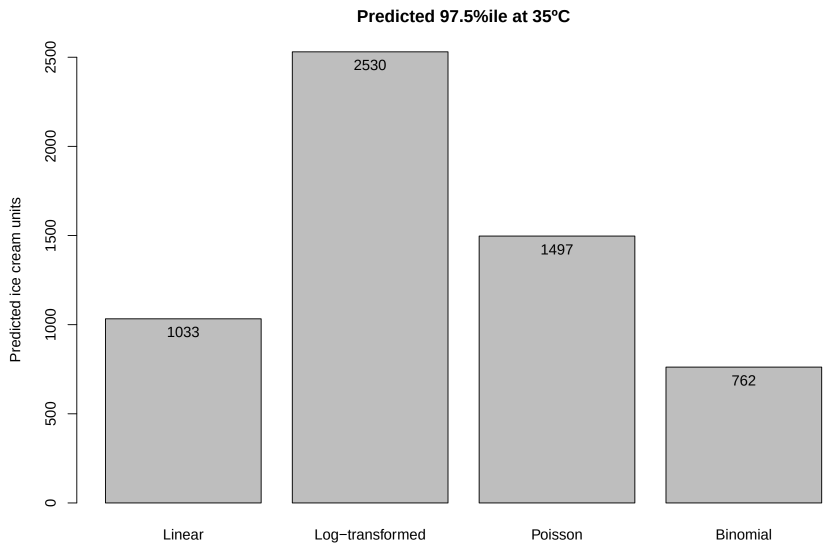
```
log.lin.samples <- posterior_samples(log.lin.mod)
mu <- A(log.lin.samples) %*% x
sigma <- log.lin.samples[,"sigma_log_units"]
(log.lin.q <- quantile(exp(rnorm(n, mu +  0.5*sigma^2, sigma)), prob))
```

```
   97.5%
2530.313
```

```
pois.samples <- posterior_samples(pois.mod)
mu <- exp(A(pois.samples) %*% x)
(pois.q <- quantile(rpois(n, mu) , prob))
```

```
97.5%
 1497
```

```
percentiles <- c(lin.q, log.lin.q, pois.q, bin.q)
b <- barplot(percentiles,
             names.arg = c("Linear", "Log-transformed",
                           "Poisson", "Binomial"),
             ylab="Predicted ice cream units",
             main="Predicted 97.5%ile at 35ºC")
text(b, percentiles-75, round(percentiles))
```



Predicted 97.5%ile at 35ºC
Predicted ice cream units
0
500
1000
1500
2000
2500
1033
2530
1497
762
Linear
Log−transformed
Poisson
Binomial