

Hierarchical linear models

Dr. Jarad Niemi

STAT 544 - Iowa State University

April 25, 2017

Outline

- Mixed effect models
- Seedling weight example
- Non-Bayesian analysis (missing pvalues/CI method)
- Bayesian analysis in Stan
- Compute posterior probabilities and CIs

Notation

Standard notation for mixed-effect models:

$$y = X\beta + Zu + e$$

where

- y is an $n \times 1$ response vector
- X is an $n \times p$ design matrix for fixed effects
- β is a $p \times 1$ unknown fixed effect parameter vector
- Z is an $n \times q$ design matrix for random effects
- u is a $q \times 1$ unknown random effect parameter vector
- e is an $n \times 1$ unknown error vector

Assumptions

$$y = X\beta + Zu + e$$

Typically assume

- $E[u] = E[e] = 0$
- $V[u] = \Omega$ and $V[e] = \Lambda$
- $Cov[u, e] = 0$

These assumptions imply

- $E[y|\beta, \Omega, \Lambda] = X\beta$
- $V[y|\beta, \Omega, \Lambda] = Z\Omega Z' + \Lambda = \Sigma_y$

Common addition assumptions

- $V[e] = \Lambda = \sigma_e^2 I$,
- $V[u] = \Omega = \text{diag}\{\sigma_{u,\cdot}^2\}$, (or $V[u] = \Omega = \sigma_u^2 I$ for single source), and
- u and e are normally distributed.

Rewrite as a standard linear regression model

We can rewrite

$$y = X\beta + Zu + e$$

as

$$y = \tilde{X}\tilde{\beta} + e$$

where \tilde{X} is $n \times (p + q)$ with

$$\tilde{X} = [X \ Z]$$

and $\tilde{\beta}$ is a $(p + q) \times 1$ vector with

$$\tilde{\beta} = \begin{bmatrix} \beta \\ u \end{bmatrix}.$$

The fixed and random effects have been concatenated into the same vector.

Hierarchical linear model

Assume $y \sim N(\tilde{X}\tilde{\beta}, \Lambda)$. A Bayesian analysis proceeds by assigning prior distributions to $\tilde{\beta}$ and Λ . In constructing the prior for $\tilde{\beta}$, consider the components β and u separately. Assume

$$\beta \sim N(b, B) \quad \text{and} \quad u \sim N(0, \Omega)$$

independently.

For the

- **fixed** effects β , we select b and B while for the
- **random** effects u , we assign a prior for Ω .

Therefore we have created a hierarchical model for the random effects and thus refer to this as a *hierarchical linear model*.

Summary

These models are referred to as

- mixed-effect models,
- hierarchical linear models, or
- multi-level models.

The parameters for the prior distribution for the

- fixed effects are not learned and
- random effects are learned.

This corresponds to a non-Bayesian analysis learning a variance parameter for random effects.

Seedling weight example

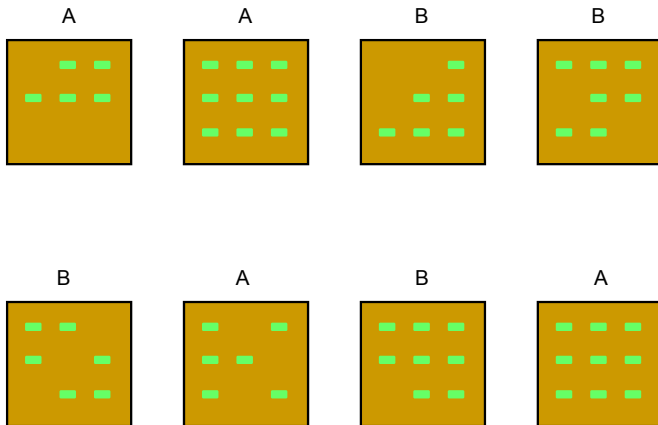
Example taken from Dan Nettleton:

Researchers were interested in comparing the dry weight of maize seedlings from two different genotypes (A and B). For each genotype, nine seeds were planted in each of four trays. The eight trays in total were randomly positioned in a growth chamber. Three weeks after the emergence of the first seedling, emerged seedlings were harvested from each tray and, after drying, weighed.

Assume the missing data (emergence) mechanism is ignorable.

Data: <http://www.public.iastate.edu/~dnett/S511/SeedlingDryWeight2.txt>

A picture



A mixed effect model for seedling weight

Let y_{gts} be the seedling weight of the

- g^{th} genotype with $g = 1, 2$,
- t^{th} tray $t = 1, 2, 3, 4$ of the g^{th} genotype, and
- s^{th} seedling with $s = 1, \dots, n_{gt}$.

Then, we assume

$$y_{gts} = \gamma_g + \tau_{gt} + e_{gts}$$

where

- $\tau_{gt} \stackrel{ind}{\sim} N(0, \sigma_\tau^2)$ and, independently,
- $e_{gts} \stackrel{ind}{\sim} N(0, \sigma_e^2)$.

The main quantity of interest is the difference in mean seedling weight: $\gamma_2 - \gamma_1$.

As a general mixed effects model

Let X have the following 2 columns

- col1: all ones (intercept) $[\gamma_1]$
- col2: ones if genotype B and zeros otherwise $[\gamma_2 - \gamma_1]$

Let Z have the following 8 columns

- col1: ones if genotype 1, tray 1 and zeros otherwise $[\tau_{11}]$
- col2: ones if genotype 1, tray 2 and zeros otherwise $[\tau_{12}]$
- \vdots
- col8: ones if genotype 2, tray 4 and zeros otherwise $[\tau_{24}]$

Then

$$y = X\beta + Zu + e$$

with $u \sim N(0, \sigma_u^2 I)$ and, independently, $e \sim N(0, \sigma_e^2 I)$.

Seedling weight data

```
head(d)
```

	Genotype	Tray	SeedlingWeight
1	A	1	8
2	A	1	9
3	A	1	11
4	A	1	12
5	A	1	10
6	A	2	17

```
summary(d)
```

Genotype	Tray	SeedlingWeight
A:29	Min.	:1.000 Min. : 6.00
B:27	1st Qu.	:2.750 1st Qu.:10.00
	Median	:4.000 Median :14.00
	Mean	:4.554 Mean :13.88
	3rd Qu.	:6.250 3rd Qu.:17.00
	Max.	:8.000 Max. :24.00

```
with(d, table(Genotype, Tray))
```

	Tray							
Genotype	1	2	3	4	5	6	7	8
A	5	9	6	9	0	0	0	0
B	0	0	0	0	6	7	6	8

Non-Bayesian analysis

```
m1 = lmer(SeedlingWeight ~ Genotype + (1|Tray), d); summary(m1)
```

```
Linear mixed model fit by REML ['lmerMod']
Formula: SeedlingWeight ~ Genotype + (1 | Tray)
Data: d
```

```
REML criterion at convergence: 247.1
```

```
Scaled residuals:
```

Min	1Q	Median	3Q	Max
-2.0928	-0.5697	0.0470	0.5146	3.2347

```
Random effects:
```

Groups	Name	Variance	Std.Dev.
Tray	(Intercept)	11.661	3.415
Residual		3.543	1.882

Number of obs: 56, groups: Tray, 8

```
Fixed effects:
```

	Estimate	Std. Error	t value
(Intercept)	15.289	1.745	8.761
GenotypeB	-3.550	2.469	-1.438

```
Correlation of Fixed Effects:
```

	(Intr)
GenotypeB	-0.707

Why no pvalues?

From <https://stat.ethz.ch/pipermail/r-help/2006-May/094765.html> (19 May 2006):

Users are often surprised and alarmed that the summary of a linear mixed model fit by lmer provides estimates of the fixed-effects parameters, standard errors for these parameters and a t-ratio but no p-values.

...

Most of the research on tests for the fixed-effects specification in a mixed model begin with the assumption that these statistics will have an F distribution with a known numerator degrees of freedom and the only purpose of the research is to decide how to obtain an approximate denominator degrees of freedom. I don't agree.

...

For the time being, I would recommend using a Markov Chain Monte Carlo sample (function mcmcscamp) to evaluate the properties of individual coefficients (use HPDinterval or just summary from the "coda" package).

Dr. Douglas Bates

```
confint(m1, method="profile")
```

	2.5 %	97.5 %
.sig01	1.837050	5.379221
.sigma	1.560415	2.332764
(Intercept)	11.926526	18.637543
GenotypeB	-8.287734	1.204894

```
confint(m1, method="Wald")
```

	2.5 %	97.5 %
.sig01	NA	NA
.sigma	NA	NA
(Intercept)	11.868527	18.709148
GenotypeB	-8.388448	1.288046

```
confint(m1, method="boot")
```

	2.5 %	97.5 %
.sig01	1.529722	5.404522
.sigma	1.542917	2.195049
(Intercept)	11.907640	19.013466
GenotypeB	-8.758632	1.066519

Bayesian model

An alternative notation convenient for programming in Stan is

- y_s is the weight for seedling s with $s = 1, \dots, n$
- $g[s] \in \{1, 2\}$ is the genotype for seedling s
- $t[s] \in \{1, 2, \dots, 8\}$ is the **unique** tray id for seedling s

Then the model is

$$y_s = \gamma_{g[s]} + \tau_{t[s]} + e_s$$

with $e_s \stackrel{ind}{\sim} N(0, \sigma_e^2)$ and, independently, $\tau_t \stackrel{ind}{\sim} N(0, \sigma_\tau^2)$ with $t = 1, \dots, 8$.

Prior:

$$p(\gamma_1, \gamma_2, \sigma_e, \sigma_\tau) \propto Ca^+(\sigma_e; 0, 1) Ca^+(\sigma_\tau; 0, 1).$$


```
stan_model = "  
data {  
  int<lower=1> n;  
  int<lower=1> n_genotypes;  
  int<lower=1> n_trays;  
  
  real y[n];  
  int genotype[n];  
  int tray[n];  
}  
parameters {  
  real gamma[n_genotypes]; // Implicit improper prior over whole real line  
  real tau[n_trays];  
  real<lower=0> sigma_e; // Implicit prior over positive reals  
  real<lower=0> sigma_tau; // Implicit prior over positive reals  
}  
  
model {  
  sigma_e ~ cauchy(0,1);  
  sigma_tau ~ cauchy(0,1);  
  
  tau ~ normal(0,sigma_tau);  
  
  for (i in 1:n) y[i] ~ normal(gamma[genotype[i]]+tau[tray[i]], sigma_e);  
}  
  
generated quantities {  
  real delta;  
  delta = gamma[2] - gamma[1];  
}  
"
```

```
m = stan_model(model_code=stan_model)
```

```
In file included from C:/Users/jarad/Documents/R/win-library/3.3/BH/include/boost/config.hpp:39:0,
                 from C:/Users/jarad/Documents/R/win-library/3.3/BH/include/boost/math/tools/config.hpp:13,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core/var.hpp:13,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core/geom_vv.hpp:13,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/core.hpp:12,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/stan/math/rev/mat.hpp:4,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/stan/math.hpp:4,
                 from C:/Users/jarad/Documents/R/win-library/3.3/StanHeaders/include/src/stan/model/model_header.hpp:4,
                 from file155c12667ab.cpp:8:
```

```
C:/Users/jarad/Documents/R/win-library/3.3/BH/include/boost/config/compiler/gcc.hpp:186:0: warning: "BOOST_NO_CXX11_RVALUE_REFERENCES"
# define BOOST_NO_CXX11_RVALUE_REFERENCES
~
```

```
<command-line>:0:0: note: this is the location of the previous definition
cc1plus.exe: warning: unrecognized command line option "-Wno-ignored-attributes"
```

```
r = sampling(m,
  list(n = nrow(d),
       n_genotypes = nlevels(d$Genotype),
       n_trays      = max(d$Tray),
       genotype     = as.numeric(d$Genotype),
       tray         = d$Tray,
       y            = d$SeedlingWeight),
  c("gamma", "tau", "sigma_e", "sigma_tau", "delta"))
```

```
SAMPLING FOR MODEL '3fc96290350c6978ac1e05d04811ca34' 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)
```

r

Inference for Stan model: 3fc96290350c6978ac1e05d04811ca34.

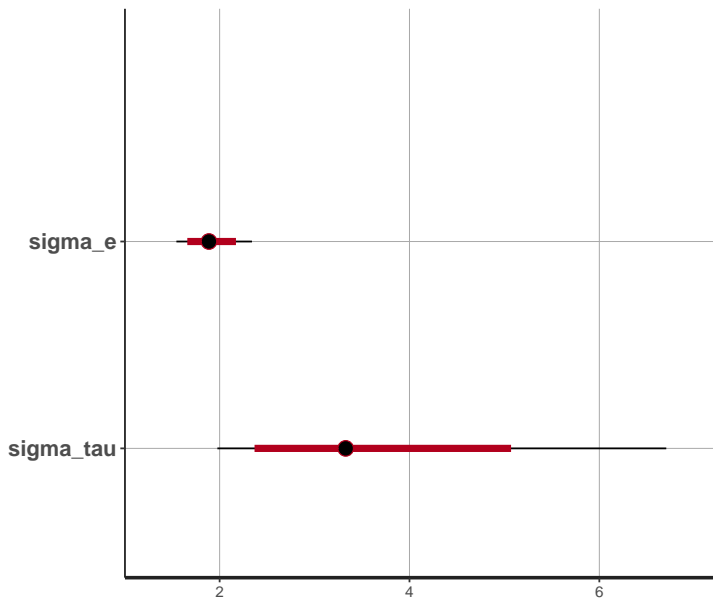
4 chains, each with iter=2000; warmup=1000; thin=1;

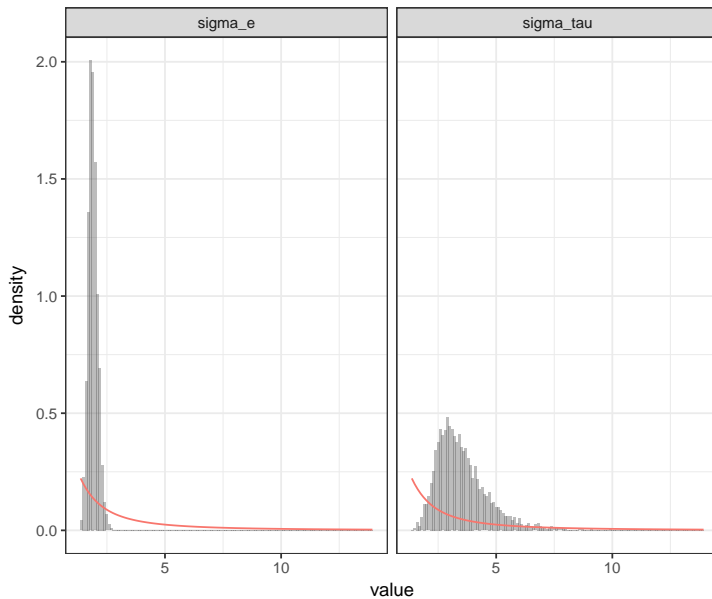
post-warmup draws per chain=1000, total post-warmup draws=4000.

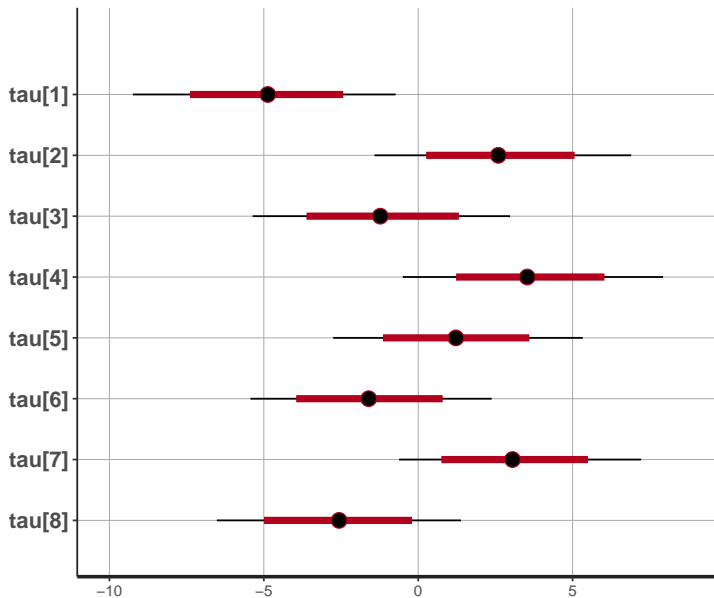
	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
gamma[1]	15.26	0.06	1.96	11.15	14.14	15.30	16.41	19.30	1111	1.00
gamma[2]	11.70	0.05	1.98	7.82	10.56	11.70	12.86	15.55	1454	1.00
tau[1]	-4.90	0.06	2.06	-9.26	-6.14	-4.87	-3.66	-0.73	1241	1.00
tau[2]	2.63	0.06	2.00	-1.44	1.40	2.59	3.81	6.90	1175	1.00
tau[3]	-1.20	0.06	2.05	-5.37	-2.44	-1.23	0.05	2.99	1203	1.00
tau[4]	3.59	0.06	2.01	-0.51	2.40	3.53	4.74	7.94	1191	1.00
tau[5]	1.23	0.05	2.05	-2.76	0.01	1.22	2.42	5.33	1534	1.00
tau[6]	-1.61	0.05	2.02	-5.43	-2.80	-1.60	-0.38	2.38	1585	1.00
tau[7]	3.12	0.05	2.04	-0.62	1.92	3.05	4.34	7.22	1505	1.00
tau[8]	-2.58	0.05	2.04	-6.52	-3.80	-2.56	-1.36	1.38	1504	1.00
sigma_e	1.90	0.00	0.20	1.54	1.76	1.89	2.03	2.34	2603	1.00
sigma_tau	3.58	0.03	1.22	1.98	2.75	3.33	4.12	6.71	1287	1.00
delta	-3.56	0.07	2.75	-9.03	-5.23	-3.58	-1.87	1.77	1480	1.00
lp__	-80.54	0.08	2.73	-86.85	-82.06	-80.24	-78.54	-76.21	1264	1.01

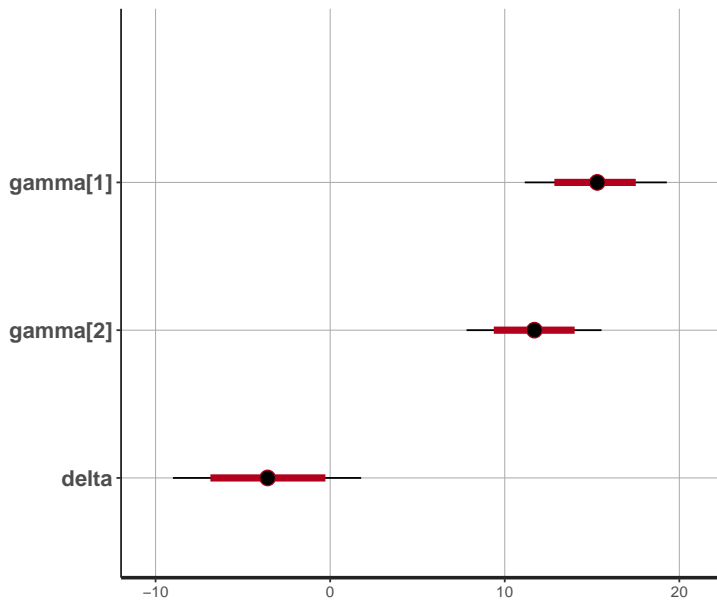
Samples were drawn using NUTS(diag_e) at Tue Apr 25 06:57:46 2017.

For each parameter, n_{eff} is a crude measure of effective sample size, and R_{hat} is the potential scale reduction factor on split chains (at convergence, $R_{\text{hat}}=1$).

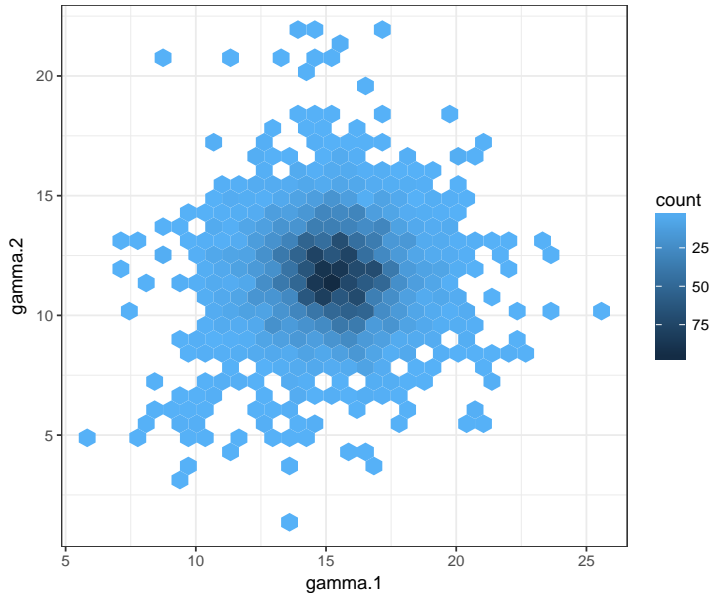


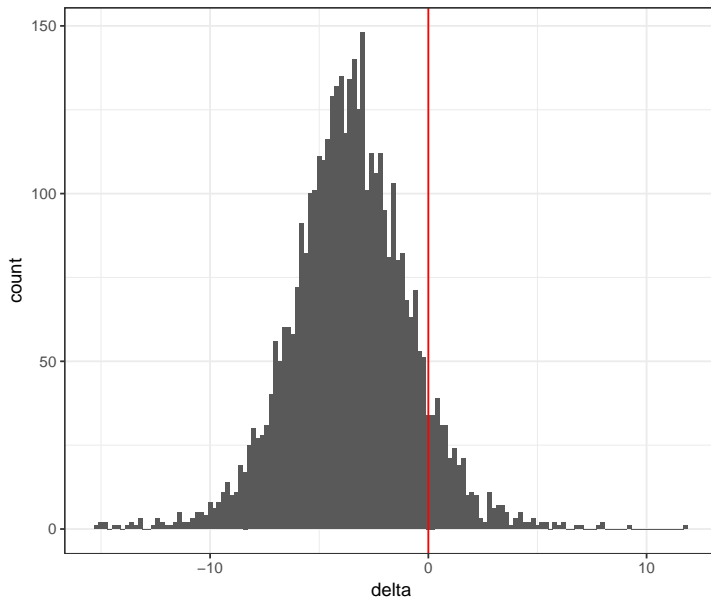






Warning: package 'hexbin' was built under R version 3.3.3





Probability that genotype B has greater mean seedling weight than genotype A.

Given our prior, i.e.

$$p(\gamma_1, \gamma_2, \sigma_e, \sigma_\tau) \propto Ca^+(\sigma_e; 0, 1)Ca^+(\sigma_\tau; 0, 1),$$

Our posterior probability that genotype B has greater mean seedling weight than genotype A is

$$P(\gamma_2 > \gamma_1 | y) = P(\delta > 0 | y) = E[\mathbf{I}(\delta > 0) | y] = E[\mathbf{I}(\gamma_2 > \gamma_1) | y].$$

If $\delta^{(m)}$ are MCMC samples from $p(\delta | y)$, then

$$\frac{1}{M} \sum_{m=1}^M \mathbf{I}(\delta^{(m)} > 0) \xrightarrow{a.s.} P(\gamma_2 > \gamma_1 | y)$$

and (if the regularity conditions hold)

$$\frac{1}{M} \sum_{m=1}^M \mathbf{I}(\delta^{(m)} > 0) \xrightarrow{d} N(P(\gamma_2 > \gamma_1 | y), \sigma^2/M).$$

Probability that genotype B has greater mean seedling weight than genotype A.

The probability is estimated to be

```
library(mcmcse)
delta = extract(r, "delta")$delta
as.data.frame(mcse(delta>0))
```

	est	se
1	0.086	0.004073628

A point estimate (posterior median) and a 95% credible interval are calculated below:

```
ddply(dd <- data.frame(q=c(.025,.5,.975)), .(q), function(x) as.data.frame(mcse.q(delta, x$q)))
```

	q	est	se
1	0.025	-9.047126	0.14093370
2	0.500	-3.584679	0.04633239
3	0.975	1.772863	0.12567965

Prediction for a new comparison

The real question is whether this idea generalizes, i.e. is true for other representatives of these genotypes. Let \tilde{y}_A and \tilde{y}_B be some future observation of seedling weight (on the same tray) for genotype A and B, respectively. We might be interested in

$$P(\tilde{y}_B > \tilde{y}_A | y) = P(\tilde{\delta} > 0 | y) = E[\mathbf{I}(\tilde{\delta} > 0) | y]$$

where $\tilde{\delta} = \tilde{y}_B - \tilde{y}_A$. If $\tilde{\delta}^{(m)} = \tilde{y}_B^{(m)} - \tilde{y}_A^{(m)}$ is a sample from the posterior predictive distribution, then we can estimate this probability via

$$\frac{1}{M} \sum_{m=1}^M \mathbf{I}(\tilde{\delta}^{(m)} > 0)$$

and have a similar LLN and CLT (if regularity conditions hold).

Prediction for a new comparison

Assuming \tilde{y}_A and \tilde{y}_B are independent conditional on γ_1, γ_2 , and σ_e , then

$$\tilde{\delta} = \tilde{y}_B - \tilde{y}_A \sim N(\gamma_2 - \gamma_1, 2\sigma_e^2)$$

and

$$p(\tilde{\delta}|y) = \int N(\tilde{\delta}; \gamma_2 - \gamma_1, 2\sigma_e^2) p(\gamma_1, \gamma_2, \sigma_e|y) d\gamma_1 d\gamma_2 d\sigma_e$$

```
samps = extract(r, c("gamma", "sigma_e"))
gamma1 = samps['gamma']$gamma[,1]
gamma2 = samps['gamma']$gamma[,2]
sigmae = samps['sigma_e']$sigma_e
tilde_delta = rnorm(length(gamma1), gamma2-gamma1, sqrt(2)*sigmae)
as.data.frame(mcse(tilde_delta>0))
```

	est	se
1	0.16225	0.005692745

```
ddply(data.frame(q=c(.025,.5,.975)), .(q), function(x) as.data.frame(mcse.q(tilde_delta, q=x$q)))
```

	q	est	se
1	0.025	-11.457558	0.19561096
2	0.500	-3.521776	0.09234766
3	0.975	3.728930	0.13801116

Extensions

Consider the model

$$y_s = \gamma_{g[s]} + \tau_{t[s]} + e_s$$

and the following modeling assumptions:

- $\gamma_g \stackrel{ind}{\sim} N(\mu, \sigma_\gamma^2)$ and learn μ, σ_γ
- $\tau_t \stackrel{ind}{\sim} La(0, \sigma_\tau^2)$
- $\gamma_g \stackrel{ind}{\sim} La(\mu, \sigma_\gamma^2)$
- $e_s \stackrel{ind}{\sim} La(0, \sigma_e^2)$
- $e_s \stackrel{ind}{\sim} t_\nu(0, \sigma_e^2)$

From a Bayesian perspective these changes do not affect the approach to inference.