GLM with clustered data

A fixed effects approach

Göran Broström

Department of Statistics

Umeå University

SE–901 87 Umeå, Sweden

Background

Poisson or Binomial data with the following properties

- A large data set,
- partitioned into many relatively small groups,
- and where members within groups have something in common,

The problem

- the number of parameters tend to increase with sample size.
- This fact causes the standard assumptions underlying asymptotic results to be violated.

Solutions

There are (at least) two possible solutions to the problem,

1. a random intercepts model, and
2. a fixed effects model, with
   - asymptotics replaced by simulation.

Packages in R

- The package Matrix has lmer,
- the MASS package has glmmPQL,
- Jim Lindsey's glmm in his repeated package,
- Myles' and Clayton's GLMMGibbs for fitting mixed models by Gibbs sampling.
- Adding to that glmmML and **glmmboot** in the package glmmML.

Data structure

- n clusters of sizes $n_i, i = 1, \ldots, n$.
- For each cluster $i, i = 1, \ldots, n$, observe responses $(y_{i1}, \ldots, y_{in_i})$ and
- vectors of explanatory variables $(\mathbf{x}_{i1}, \ldots, \mathbf{x}_{in_i})$, where $\mathbf{x}_{ij}$ are p-dimensional vectors with
  - the first element identically equal to unity,
  - corresponding to the mean value of the random intercepts.
- The random part, u_i of the intercepts are normal with mean zero and variance σ^2, and
- it is assumed that $u_1, \ldots, u_n$ are independent.

The conditional distribution

given the random intercepts $\beta_1 + u_i, i = 1, \ldots, n$:

$$\Pr(Y_{ij} = y_{ij} \mid u_i; \mathbf{x}) = P(\boldsymbol{\beta}\mathbf{x}_{ij} + u_i, y_{ij}),$$
$$y_{ij} = 0, 1, \ldots; \; j = 1, \ldots, n_i, \; i = 1, \ldots, n.$$

- Bernoulli distribution
  - *logit* link,

$$P(x, y) = \frac{e^{xy}}{1 + e^x}, \quad y = 0, 1; \; -\infty < x < \infty,$$

  - *cloglog* link

$$P(x, y) = \bigl(1 - \exp(-e^x)\bigr)^y \exp\bigl(-(1 - y)e^x\bigr), \quad y = 0, 1; \; -\infty < x < \infty,$$

- Poisson distribution with log link

$$P(x, y) = \frac{e^{xy}}{y!} e^{-e^x}, \quad y = 0, 1, 2, \ldots; \; -\infty < x < \infty$$

Likelihood function

In the fixed effects model (and in the conditional random effects model), the likelihood function is

$$L\bigl((\boldsymbol{\beta}, \boldsymbol{\gamma}); \mathbf{y}, \mathbf{x}\bigr) = \prod_{i=1}^{n} \prod_{j=1}^{n_i} P(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i, y_{ij}).$$

The log likelihood function is

$$\ell\bigl((\boldsymbol{\beta}, \boldsymbol{\gamma}); \mathbf{y}, \mathbf{x}\bigr) = \sum_{i=1}^{n} \sum_{j=1}^{n_i} \log P(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i, y_{ij}),$$

Tests of cluster effect

Testing is performed via a simple bootstrap (glmmboot).
Under the null hypothesis of no grouping effect,

- the grouping factor can be randomly permuted without changing the probability distribution (the conditional approach), or
- a parametric bootstrap approach: simulate observations from the fitted model under the null hypothesis (the unconditional approach).

Computational aspects

- A profiling approach reduces an optimizing problem in high dimensions
- to a problem consisting of
  - solving several one-variable equations followed by
  - optimization in low dimensions.

The score vector

The partial derivatives wrt $\beta_m,\ m = 1; \ldots, p$, of the log likelihood function are:

$$\begin{aligned} U_m(\boldsymbol{\beta}, \boldsymbol{\gamma}) &= \frac{\partial}{\partial \beta_m} \ell\big((\boldsymbol{\beta}, \boldsymbol{\gamma}); \mathbf{y}, \mathbf{x}\big) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{n_i} x_{ijm} G(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i,\ y_{ij}), \quad m = 1, \ldots, p. \end{aligned}$$

where

$$G(x, y) = \frac{\partial}{\partial x} \log P(x, y) = \frac{\frac{\partial}{\partial x} P(x, y)}{P(x, y)}$$

Cluster components of the score

The partial derivatives wrt $\gamma_i,\ i = 1, \ldots, n$, are

$$\begin{aligned} U_{p+i}(\boldsymbol{\beta}, \boldsymbol{\gamma}) &= \frac{\partial}{\partial \gamma_i} \ell\big((\boldsymbol{\beta}, \boldsymbol{\gamma}); \mathbf{y}, \mathbf{x}\big) \\ &= \sum_{j=1}^{n_i} G(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i,\ y_{ij}), \quad i = 1, \ldots, n. \end{aligned}$$

With profiling

Setting $U_{p+i}(\boldsymbol{\beta},\boldsymbol{\gamma}) = 0$ defines $\boldsymbol{\gamma}$ implicitly as functions of $\boldsymbol{\beta}$, $\gamma_i = \gamma_i(\boldsymbol{\beta}),\ i = 1,\dots,n$:

$$F\big(\boldsymbol{\beta},\gamma_i(\boldsymbol{\beta})\big) = \sum_{j=1}^{n_i} G\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i(\boldsymbol{\beta}),\ y_{ij}\big) = 0, \quad i = 1,\dots,n.$$

From

$$\frac{\partial}{\partial \beta_m} F\big(\boldsymbol{\beta},\gamma_i(\boldsymbol{\beta})\big) = \frac{\partial \gamma_i}{\partial \beta_m}\frac{\partial F}{\partial \gamma_i} + \frac{\partial F}{\partial \beta_m} = 0$$

we get

Profile score

$$\begin{aligned}\frac{\partial \gamma_i(\boldsymbol{\beta})}{\partial \beta_m} &= -\frac{\frac{\partial F}{\partial \beta_m}}{\frac{\partial F}{\partial \gamma_i}} \\ &= -\frac{\sum_{j=i}^{n_i} x_{ijm} H\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i,\ y_{ij}\big)}{\sum_{j=1}^{n_i} H\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i,\ y_{ij}\big)}, \quad i = 1,\dots,n;\ m = 1,\dots,\end{aligned}$$

which is needed when calculating the score corresponding to the profile likelihood.

Profile loglihood

Replacing $\boldsymbol{\gamma}$ by $\boldsymbol{\gamma}(\boldsymbol{\beta})$ gives the profile log likelihood $\ell^{(P)}$:

$$\ell^{(P)}\big(\boldsymbol{\beta};\mathbf{y},\mathbf{x}\big) = \sum_{i=1}^{n}\sum_{j=1}^{n_i} \log P\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i(\boldsymbol{\beta}),\ y_{ij}\big),$$

as a function of $\boldsymbol{\beta}$ alone.

Profile partial derivatives

The partial derivatives wrt $\beta_m,\ m = 1;\dots,p$, of the log profile likelihood function becomes:

$$\begin{aligned}U_m^{(P)}(\boldsymbol{\beta}) &= \frac{\partial}{\partial \beta_m}\ell^{(P)}(\boldsymbol{\beta};\mathbf{y},\mathbf{x}) \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n_i}\left(x_{ijm} + \frac{\partial \gamma_i(\boldsymbol{\beta})}{\partial \beta_m}\right) G\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i(\boldsymbol{\beta}),\ y_{ij}\big) \\ &= U_m\big(\boldsymbol{\beta},\boldsymbol{\gamma}(\boldsymbol{\beta})\big) + \sum_{i=1}^{n}\frac{\partial \gamma_i(\boldsymbol{\beta})}{\partial \beta_m}\sum_{j=1}^{n_i} G\big(\boldsymbol{\beta}\mathbf{x}_{ij} + \gamma_i(\boldsymbol{\beta}),\ y_{ij}\big) \\ &= U_m(\boldsymbol{\beta},\boldsymbol{\gamma}(\boldsymbol{\beta})),\end{aligned}$$

Thus we get back the unprofiled partial derivatives.

Profile hessian

$$
\begin{aligned}
-I^{(P)}_{ms}(\boldsymbol\beta) &= \frac{\partial}{\partial \beta_s} U_m(\boldsymbol\beta, \boldsymbol\gamma(\boldsymbol\beta)) \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n_i} x_{ijm} \left(x_{ijs} + \frac{\partial \gamma_i(\boldsymbol\beta)}{\partial \beta_s} \right) H\big(\boldsymbol\beta \mathbf{x}_{ij} + \gamma_i(\boldsymbol\beta),\ y_{ij}\big) \\
&= -I_{ms}(\boldsymbol\beta, \boldsymbol\gamma(\boldsymbol\beta)) \\
&\quad - \sum_{i=1}^{n} \frac{\sum_{j=1}^{n_i} x_{ijm} H_{ij} \sum_{j=1}^{n_i} x_{ijs} H_{ij}}{\sum_{j=1}^{n_i} H_{ij}}, \\
&\quad m, s = 1, \dots, p.
\end{aligned}
$$

where

$$H_{ij} = H\big(\boldsymbol\beta \mathbf{x}_{ij} + \gamma_i(\boldsymbol\beta),\ y_{ij}\big), \quad j = 1, \dots n_i;\ i = 1, \dots, n.$$

At the maximum

Justifying the use of the profile likelihood:

Theorem 1 (Patefield) *The inverse hessians from the full likelihood and from the profile likelihood for $\boldsymbol\beta$ are equal when*

$$(\boldsymbol\gamma, \boldsymbol\beta) = (\hat{\boldsymbol\gamma}, \hat{\boldsymbol\beta}).$$

Preparation for R

- $\ell^{(P)}(\boldsymbol\beta) = \sum_{i=1}^{n} \sum_{j=1}^{n_i} \log P\big(\boldsymbol\beta \mathbf{x}_{ij} + \gamma_i(\boldsymbol\beta),\ y_{ij}\big)$,
- $U^{(P)}_m(\boldsymbol\beta) = \sum_{i=1}^{n} \sum_{j=1}^{n_i} x_{ijm} G\big(\boldsymbol\beta \mathbf{x}_{ij} + \gamma_i(\boldsymbol\beta),\ y_{ij}\big)$, $m = 1, \dots, p$.
- For fixed $\boldsymbol\beta$, $\gamma_i(\boldsymbol\beta)$ is found by solving

  $$\sum_{j=1}^{n_i} G(\boldsymbol\beta \mathbf{x}_{ij} + \gamma_i,\ y_{ij}) = 0,$$

  with respect to γ_i, $i = 1, \dots, n$.
- The maximization is performed by optim, via the *C* function vmmin, available as an entry point in the *C* code of **R**.

Implementation in R

- Implemented in the package `glmmML` in **R**.
- Covers three cases,
  1. Binomial with `logit` link,
  2. Binomial with `cloglog` link,
  3. Poisson with `log` link.
- The function is glmmboot,
- Testing of cluster effect is done by simulation (a simple form of bootstrapping).
  - conditionally, or
  - unconditionally.

Binomial with logit link

- $P(x, y) = \exp(xy)/(1 + \exp(x))$,
- $G(x, y) = y - P(x, 1)$.
- We get $(\gamma_1, \ldots, \gamma_n)$ by solving the equations

$$\sum_{j=1}^{n_i} y_{ij} = \sum_{j=1}^{n_i} \frac{\exp(\boldsymbol{\beta} x_{ij} + \gamma_i)}{1 + \exp(\boldsymbol{\beta} x_{ij} + \gamma_i)}$$

  for $i = 1, \ldots, n$ (using the *C* version of `uniroot`).
- Special cases: $\sum y_{ij} = 0$ or n_i; giving $\gamma_i = -\infty$ or $+\infty$, respectively.
  - Corresponding cluster can be thrown out.
  - (Should be used in `glm`?)

Binomial with cloglog link

- $P(x, y) = (1 - \exp(-\exp(x))^y \exp(-(1 - y)\exp(x))$,
- $G(x, y) = \frac{\exp(x)}{P(x,1)}\{y - P(x, 1)\}$
- We get $(\gamma_1, \ldots, \gamma_n)$ by solving the equations

$$\sum_{j=1}^{n_i} y_{ij} = n_i - \sum_{j=1}^{n_i} \exp(-\exp(\boldsymbol{\beta} x_{ij} + \gamma_i))$$

  for $i = 1, \ldots, n$ (using the *C* version of `uniroot`).
- Special cases: $\sum y_{ij} = 0$ or n_i; $\gamma_i = -\infty$ or $+\infty$, respectively.
  - Corresponding cluster can be thrown out.

Poisson with log link

- $P(x, y) = \frac{e^{xy}}{y!} \exp(-\exp(x))$
- $G(x, y) = y - e^x$
- We get $(\gamma_1, \ldots, \gamma_n)$ from

$$\sum_{j=1}^{n_i} y_{ij} = e^{\gamma_i} \sum_{j=1}^{n_i} \exp(\boldsymbol{\beta}\mathbf{x}_{ij}), \quad i = 1, \ldots, n,$$

  giving

$$\gamma_i = \log\left\{\frac{\sum_j y_{ij}}{\sum_j \exp(\boldsymbol{\beta}\mathbf{x}_{ij})}\right\}, \quad i = 1, \ldots, n.$$

- Special case: $\sum y_{ij} = 0$, giving $\gamma_i = -\infty$.

Simulation

Model:

$$P(Y_{ij} = 1 \mid \gamma_i) = 1 - P(Y_{ij} = 0 \mid \gamma_i)$$
$$= \frac{e^{\gamma_i}}{1 + e^{\gamma_i}}, \quad j = 1, \ldots, 5; \quad i = 1, \ldots, n,$$

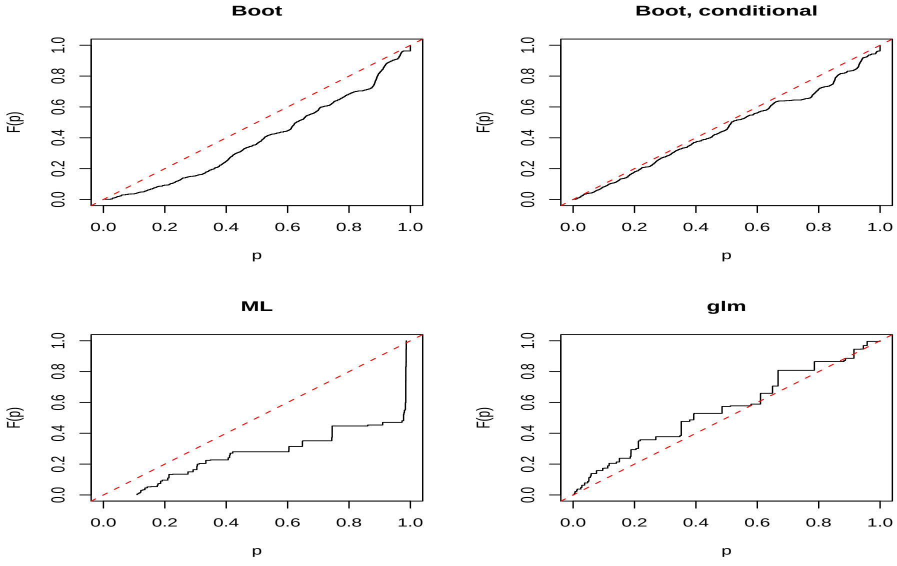
where $\gamma_1, \ldots, \gamma_n$ are *iid* $N(0, \sigma)$.

Hypothesis: $\sigma = 0$.

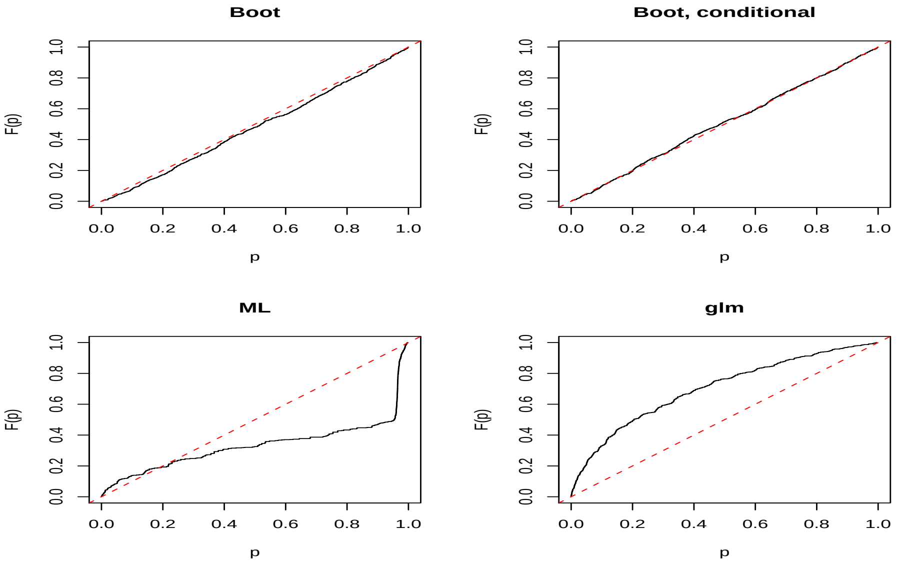
Simulation specifications

- $\sigma = 0, 0.5$.
- $n = 5, 50, 500$.
- Four methods:
  - `glmmboot`, unconditional and conditional,
  - `glmmML`,
  - `glm` (naively?).

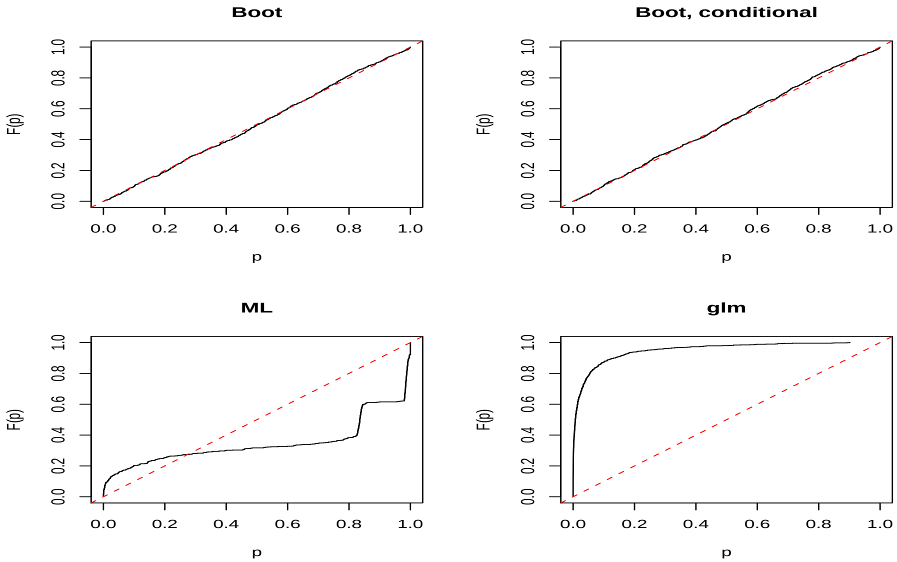
Null model ($\sigma = 0$); 5 clusters



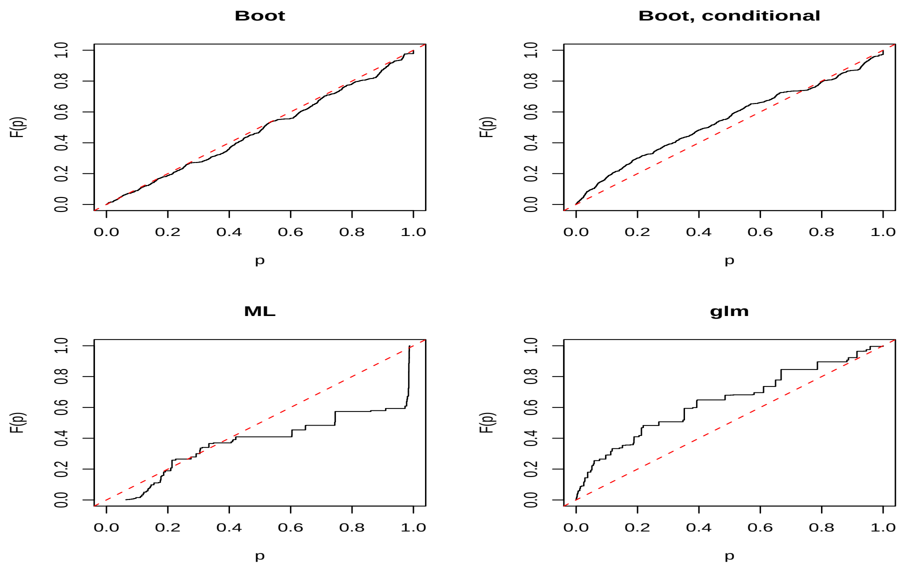
Null model ($\sigma = 0$); 50 clusters



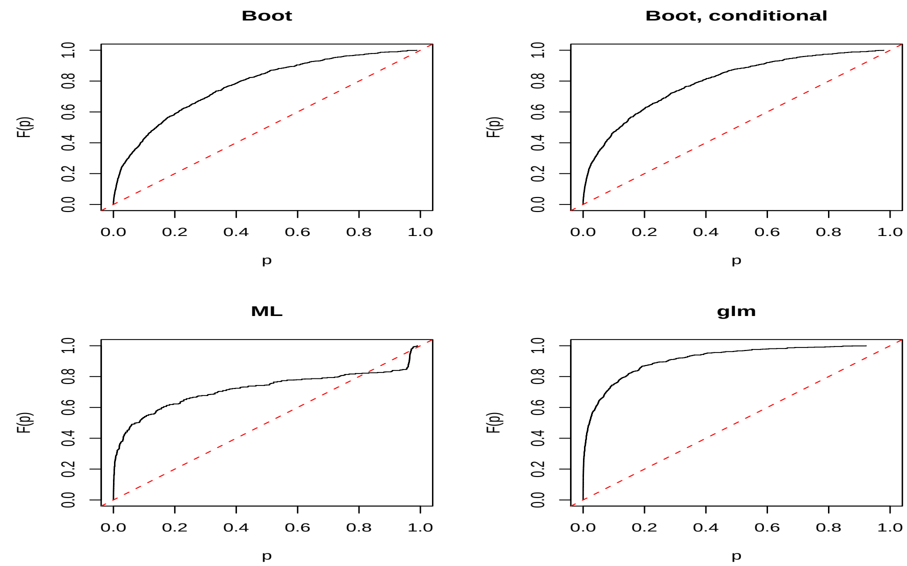
Null model ($\sigma = 0$); 500 clusters



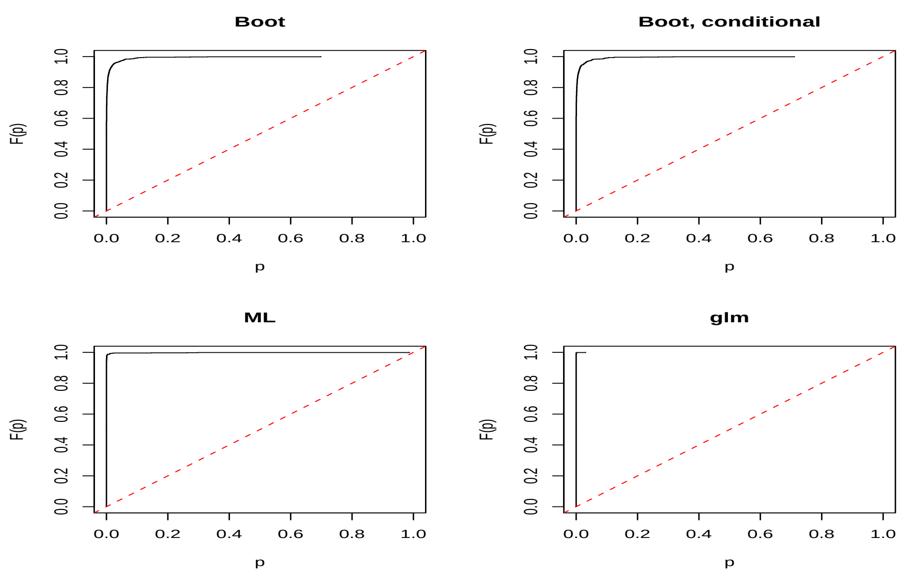
Clustering ($\sigma = 0.5$); 5 clusters



Clustering ($\sigma = 0.5$); 50 clusters



Clustering ($\sigma = 0.5$); 500 clusters



Timings, 5 clusters

```
> system.time(glmmboot(y ~ 1, cluster = cluster,
+ data = timing, conditional = FALSE, boot = 2000))
[1] 0.06 0.00 0.06 0.00 0.00

> system.time(glmmboot(y ~ 1, cluster = cluster,
data = timing, conditional = TRUE, boot = 2000))
[1] 0.044 0.000 0.044 0.000 0.000

> system.time(glmmML(y ~ 1, cluster = cluster,
data = timing))
[1] 0.013 0.000 0.012 0.000 0.000

> system.time(glm(y ~ factor(cluster), data = timing,
family = binomial))
[1] 0.008 0.000 0.008 0.000 0.000
```

Timings, 50 clusters

```
> system.time(glmmboot(y ~ 1, cluster = cluster,
data = timing, conditional = FALSE, boot = 2000))
[1] 0.529 0.000 0.529 0.000 0.000

> system.time(glmmboot(y ~ 1, cluster = cluster,
data = timing, conditional = TRUE, boot = 2000))
[1] 0.376 0.000 0.376 0.000 0.000

> system.time(glmmML(y ~ 1, cluster = cluster,
data = timing))
[1] 0.079 0.000 0.080 0.000 0.000

> system.time(glm(y ~ factor(cluster),
data = timing, family = binomial))
[1] 0.047 0.002 0.061 0.000 0.000
```


Timings, 500 clusters

```
> system.time(glmmboot(y ~ 1, cluster = cluster,
data = timing, conditional = FALSE, boot = 2000))
[1] 5.208 0.000 5.214 0.000 0.000

> system.time(glmmboot(y ~ 1, cluster = cluster,
data = timing, conditional = TRUE, boot = 2000))
[1] 3.713 0.003 3.719 0.000 0.000

> system.time(glmmML(y ~ 1, cluster = cluster,
data = timing))
[1] 0.611 0.000 0.611 0.000 0.000

> system.time(glm(y ~ factor(cluster),
data = timing, family = binomial))
[1] 27.840  0.593 28.434  0.000  0.000
```


glm vs. glmmboot(boot = 0)

Execution times

No. of clusters	glm	glmmboot
5	0.008	0.007
25	0.019	0.008
100	0.182	0.011
500	28.434	0.031
1000	223.288	0.056

Conclusion: Profiling is numerically very efficient.