STAT 770 Nov. 23 Lecture 24

Random-Intercept GLMM Likelihoods and Software

Reading and Topics for this and next lecture: Sections 13.1–13.3, Handout 1(ii) from course web-page, `GLMMscript.RLog` under Handout 9, and introductory material on EM algorithm in `https://www.math.umd.edu/ slud/s705/`.

(1) Random Intercept Logistic Likelihood for clustered data

(2) Overview: Approximate Likelihood Computation Techniques

(3) Adaptive Gaussian Quadrature

(4) R Scripts and Software

A Loose End – Parameters from `loglin`

I earlier suggested that you needed to fit loglinear models with `loglm` to get fitted coefficients (satisfying sum side-conditions). These can be obtained also in the **$param** list-component of the **loglin** R-function if you include **param=T** in the function call.

A Commercial for my Spring 2021 Course

I teach **STAT 818D Bootstrap Methods** at **MWF 9** in Spring 2021. Please register if you are or might be interested.

(Topics courses with small enrollment may get cancelled in 7 - 10 days.) The overall topic is Resampling methods for bias reduction, variance and CI estimation, and calculation of null reference distributions for hypothesis testing.

The course is MA level, more or less at the level of STAT 770, mixing theory, software-oriented applications and data analyses.

I will send around a rough syllabus in the next few days. The source material for the course will be lecture notes (adapted from a colleague's) along with several freely downloadable book chapters and e-book texts.

GLMM Likelihood

Consider logit-normal model, restricted to case where all Y_{ij} are binary, cluster i has n_i unit observations, and $\underline{X}_{ij} \equiv \underline{X}_i$.

Then

$$P(Y_{i,j} = y_j,\ 1 \le j \le n_i) \;=\; E\Big(P(Y_{i,j} = y_j,\ 1 \le j \le n_i \,\Big|\, \epsilon_i)\Big)$$

$$= \int \exp\Big(\sum_{j=1}^{n_i} y_j\,(\underline{X}_i^{tr}\,\beta + \sigma z)\Big)\Big[\prod_{j=1}^{n_i}(1 + e^{\underline{X}_i^{tr}\,\beta + \sigma z})\Big]^{-1}\phi(z)\,dz$$

We will consider strategies for maximizing in β, σ the sum of log's over i, mostly in the simpler case where all covariates $\underline{X}_{i,j} = \underline{X}_i$ are at cluster level.

Computations of Approximate MLEs

(i) Approximate MLEs by EM Algorithm

(ii) Approximate logLik using Laplace Asymptotic Method

(iii) Approximate logLik using Adaptive Gaussian Quadrature

(iv) Estimation via Bayes Computation, i.e., MCMC

EM Algorithm Approach

Idea: consider unobserved random intercepts as "missing data", write joint logLik for that data together with observed data

$$\text{logLik}^{aug}(\beta,\sigma,\underline{\epsilon}) = \sum_{i=1}^{m} \Big\{ \sum_{j=1}^{n_i} Y_{i,j}\,(\underline{X}_i^{tr}\beta + \epsilon_i) - n_i \log(1+e^{\underline{X}_i^{tr}\beta+\epsilon_i}) - \log(\sigma) - \epsilon_i^2/(2\sigma^2) \Big\}$$

aug = 'augmented'

EM defines $Q(\beta,\sigma,\beta^{(k)},\sigma^{(k)}) = E_{\beta^{(k)},\sigma^{(k)}}\Big[\text{logLik}^{aug}(\beta,\sigma,\underline{\epsilon}) \,\Big|\, \{Y_{i,j}\}\Big]$

$$(\beta^{(k+1)},\sigma^{(k+1)}) = \text{argmax}_{(\beta,\sigma)}\; Q(\beta,\sigma,\beta^{(k)},\sigma^{(k)})$$

and iterates to MLE, as in `R` package `mcemGLM`. **(Conditional $\underline{\epsilon}$ distributions given $\underline{Y}$ are computable via numerical integration in special cases, otherwise via Monte Carlo.)**

Laplace's Asymptotic Method

General Laplace method approximates

$$\int e^{h_n(x)}\,dx \approx e^{h_n(x_n^*)} \int \exp\Big(h_n''(x_n^*)(x - x_n^*)^2/2\Big)\,dx$$

where $x_n^* = \text{argmax}_x\, h_n(x)$. This leads to

$$\log \int e^{h_n(x)}\,dx \approx h_n(x^*) + \log\Big[-2\pi\Big/ h_n''(x_n^*)\Big]^{1/2}$$

and can be taken to higher-order Taylor series terms. My hand-out Tech. report documents that these approximations are not generally very accurate in logistic regression.

Gaussian Quadrature

To integrate smooth functions against Gaussian density:

$$\int f(x)e^{-x^2}dx \approx \sum_{k=-q}^{q} w_k\, f(p_k)$$

$$\int g(z)\,\phi(z)\,dz \approx \frac{1}{\sqrt{\pi}} \sum_{k=-q}^{q} w_k\, g(p_k\sqrt{2})$$

where p_k are symmetrically placed points, and $w_k = w_{-k}$ are symmetric weights, both carefully chosen.

Adaptive Gaussian Quadrature

Now for logLik as in Logit-Normal GLMM, want

$$\int \text{exp}\left(l_n(\underline{Y}_i, \underline{X}_i, z)\right) \phi(z)\, dz \; =$$

$$\boxed{e^{h_{ni}(z^*_{ni})}\, D_{ni} \int \text{exp}\left(h_{ni}(z^*_{ni} + D_{ni}x) - h_{ni}(z^*_{ni}) \, + \, \frac{x^2}{2}\right) \phi(x)\, dx}$$

where

$$h_{ni}(z) \; = \; l_n(\underline{Y}_i, \underline{X}_i, z) - \frac{z^2}{2} \, , \qquad z^*_{ni} \; = \; \text{argmax}_z \, h_{ni}(z)$$

$$D_{ni} = \left[1 - \frac{\partial^2}{\partial z^2} \, l_n(\underline{Y}_i, \underline{X}_i, \, z^*_{ni})\right]^{-1/2} , \qquad z = z^*_{ni} + D_{ni}\, x$$

and integrate the blue boxed expression numerically by Gaussian Quadrature.

Software Comments

glmer in R package lme4 – uses AGQ

PROC NLINMIX or GLIMMIX in SAS – use AGQ

EM, MCMC and **Adaptive Gaussian Quadrature** ideas for logLik maximization

respectively in mcemGLM, bayesglm (or blme or brms), and glmer or glmmML or GLMMadaptive.

R Script Examples in GLMMscript.RLog