Estimation of 1-PL Model

```r
library(lme4, include.only = "GHrule")
```

```
Warning in check_dep_version(): ABI version mismatch:
lme4 was built with Matrix ABI version 2
Current Matrix ABI version is 1
Please re-install lme4 from source or restore original 'Matrix' package
```

```r
library(mirt)
```

For simplicity, I assume $Da = 1$ here; in practice a is usually estimated with 1-PL.

Conditional likelihood (assumed θ known)

Let $P_i(\theta_j) = P(Y_i = 1 \mid \theta = \theta_j)$

If $Y_{ij} = 1$,

$$\mathcal{L}(b_i; y_{ij}, \theta_j) = P_i(\theta_j)$$

If $y_{ij} = 0$,

$$\mathcal{L}(b_i; y_{ij}, \theta_j) = 1 - P_i(\theta_j)$$

Marginal Likelihood (Integrating out θ)

Let $\theta_j \sim N(0, 1)$, meaning its probability density is

$$f(\theta) = \Phi(\theta) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\theta^2/2}$$

$$\mathcal{L}(b_i; Y_{ij} = 1) = \int_{-\infty}^{\infty} P_i(t)\Phi(t)\,\mathrm{d}t$$

and

$$\ell(b_i; Y_{ij} = 1) = \log\left[\int_{-\infty}^{\infty} P_i(t)\Phi(t)\,\mathrm{d}t\right]$$

Combining N independent observations, and let $\bar{y}_i$ be the mean of item i (i.e., proportion scoring 1),

$$\begin{aligned}\ell(b_i; y_{i1}, y_{i2}, \dots, y_{iN}) = N \Bigg\{&\bar{y}_i \log\left[\int_{-\infty}^{\infty} P_i(t)\Phi(t)\,\mathrm{d}t\right] \\ &+ (1 - \bar{y}_i)\log\left[1 - \int_{-\infty}^{\infty} P_i(t)\Phi(t)\,\mathrm{d}t\right]\Bigg\}\end{aligned}$$

Maximum likelihood estimation finds values of b_i that maximizes the above quantity. However, the above quantity involves an integral, which does not generally have an analytic solution. More typically, we use Gaussian-Hermite quadrature to approximate the integral.

Quadrature

See https://en.wikipedia.org/wiki/Gauss%E2%80%93Hermite_quadrature

```r
# Approximate z^2 for z ~ N(0, 1)
ghq5 <- lme4::GHrule(5)
sum(ghq5[, "w"] * ghq5[, "z"]^2)
```

```
[1] 1
```

```r
pj <- function(theta, bj) plogis(theta - bj)
sum(ghq5[, "w"] * pj(ghq5[, "z"], bj = 1))
```

```
[1] 0.303218
```

```r
sum(ghq5[, "w"] * pj(ghq5[, "z"], bj = 2))
```

```
[1] 0.1555028
```

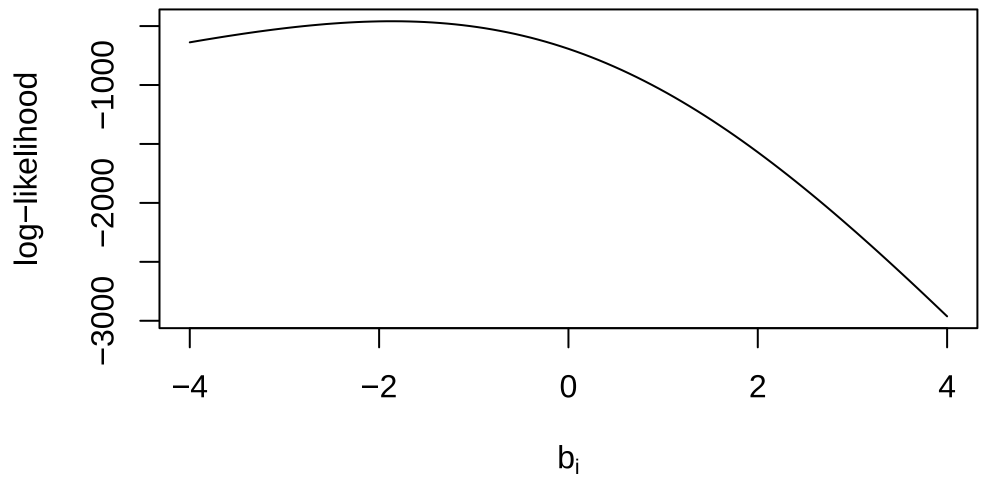
```r
loglik_1pl <- function(b, n, ybar, gh_ord = 5) {
    ghq <- lme4::GHrule(gh_ord)
    int_pj <- sum(ghq[, "w"] * pj(ghq[, "z"], bj = b))
    n * (ybar * log(int_pj) + (1 - ybar) * log1p(-int_pj))
}
loglik_1pl(1, n = 1000, ybar = 828 / 1000)
```

```
[1] -1050.196
```

```r
# Vectorized version
pj <- function(theta, bj) plogis(outer(theta, Y = bj, FUN = "-"))
loglik_1plj <- function(b, n, ybar, ghq = lme4::GHrule(5)) {
    int_pj <- crossprod(ghq[, "w"], pj(ghq[, "z"], bj = b))
    as.vector(
        n * (ybar * log(int_pj) + (1 - ybar) * log1p(-int_pj))
    )
}
loglik_1plj(c(-1, 0, 1), n = 1000, ybar = 828 / 1000)
```

```
[1]  -504.3902  -693.1472 -1050.1957
```

```r
curve(loglik_1plj(x, n = 1000, ybar = 828 /1000),
      from = -4, to = 4,
      xlab = expression(b[i]),
      ylab = "log-likelihood")
```




```r
# Optimization
optim(0,
    fn = loglik_1plj,
    n = 1000, ybar = 828 / 1000,
    method = "L-BFGS-B",
    # Minimize -2 x ll
    control = list(fnscale = -2)
)
```

```
$par
[1] -1.862783

$value
[1] -459.0433

$counts
function gradient
       7        7

$convergence
[1] 0

$message
```

```
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

While here we assume a is known, in reality, a needs to be estimated as well and require we obtain the joint likelihood by adding up the likelihood across all items (assuming local independence).