Psychometric and Statistical Models in the **R** packages **sensR** and **ordinal**

Rune H B Christensen

DTU Informatics, IMM
Section for Statistics
Technical University of Denmark
rhbc@imm.dtu.dk

February 9th 2012

Background on Rune H B Christensen

PhD: “Sensometrics: Thurstonian and Statistical Models”
November 2008 — April 2012

Education:
Engineer from DTU in 2008 — Statistics and Data Analysis

Research interests:

- Sensometrics
- Likelihood methods
- Mixed effects models
- Computational statistics
- Applied statistics (food science, biology, . . .)

R-packages:

- **sensR** with Per Bruun Brockhoff
- **ordinal**
- **binomTools** with Merete Kjær Hansen

Outline

Outline

The sensR package

"Estimation and inference in Thurstonian models for sensory discrimination"

On CRAN since July 2008:
`www.cran.r-project.org/packages=sensR`

Development on R-Forge:
`https://r-forge.r-project.org/projects/sensr/`

Vignettes:

- Statistical methodology for sensory discrimination tests and its implementation in **sensR**
- Examples for papers

Psychometric protocols supported in **sensR**

Discrimination	d' estimation	Difference test	Similarity test	Power	Sample size	Simulation	Likelihood CI	Replicated	Regression analysis
Duo-Trio, Triangle	✓	✓	✓	✓	✓	✓	✓	✓	✓
2-AFC, 3-AFC	✓	✓	✓	✓	✓	✓	✓	✓	✓
A-not A	✓	✓	✓				✓	✓	✓
Same-Different	✓	✓	(✓)	✓		✓	✓		
2-AC	✓	✓	✓	✓			✓	✓	✓
A-not A w. Sureness	✓	✓	(✓)				✓	✓	✓

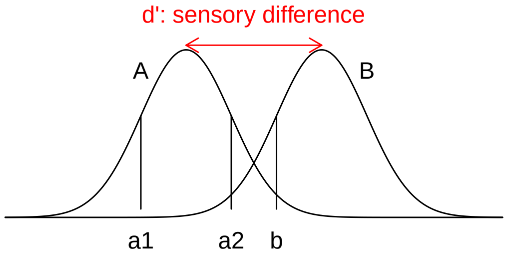
Basic functions in **sensR**

d', CI, tests	Power & Sample size	Transformation	Illustration	Miscellaneous
`discrim`	`discrimPwr`	`rescale`	`plot`	`findcr`
`AnotA`	`d.primePwr`	`psyfun`	`ROC`	`clm2twoAC`
`samediff`	`discrimSS`	`psyinv`	`AUC`	`SDT`
`twoAC`	`d.primeSS`	`pc2pd`		`samdiffSim`
`betabin`	`twoACpwr`	`pd2pc`		`discrimSim`

Beyond the basics:

- `glm` family objects for Thurstonian models:
  `twoAFC()`, `threeAFC()`, `duotrio()`, `triangle()`

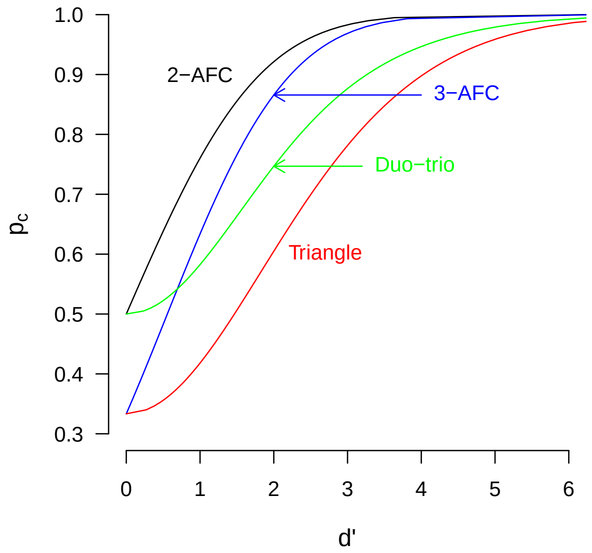
The Thurstonian model, 3-alternatives



What is the probability of a correct answer?

It depends on the question — 3-AFC or Triangle.

Psychometric functions



A GLM:

$$y \sim \text{binom}(p_c, n)$$

$$p_c = f_{\text{psy}}(d')$$

$$d' = \boldsymbol{X\beta}$$

Problem: $d' \geq 0$

Psychometric functions: Inverse link functions

$$f_{\text{3-AFC}}(d') = \int_{-\infty}^{\infty} \phi(z - d')\Phi(z)^2 \,\mathrm{d}z$$

$$f_{\text{2-AFC}}(d') = \int_{-\infty}^{\infty} \phi(z - d')\Phi(z) \,\mathrm{d}z = \Phi(d'/\sqrt{2})$$

$$f_{\text{triangle}}(d') = 2\int_{0}^{\infty} \left\{\Phi\left[-z\sqrt{3} + d'\sqrt{2/3}\right] + \Phi\left[-z\sqrt{3} - d'\sqrt{2/3}\right]\right\}\phi(z) \,\mathrm{d}z$$

$$f_{\text{duo-trio}}(d') = 1 - \Phi(d'/\sqrt{2}) - \Phi(d'/\sqrt{6}) + 2\Phi(d'/\sqrt{2})\Phi(d'/\sqrt{6}).$$

Family objects:
`twoAFC()`, `threeAFC()`, `duotrio()`, `triangle()`

psyphy: `mafc(m=3)`: $F(\eta) = \frac{1}{m}(1 - \lambda - \frac{1}{m})\Phi(\eta)$
→ only depends on no. alternatives.

Outline

1. The sensR package
2. The ordinal package — overview
3. Implementation in the ordinal package
4. Assessment of estimation accuracy
5. Cumulative link mixed models (CLMMs)

The **ordinal** package

Regression models for ordinal data via cumulative link models

On CRAN since March 2010:
`www.cran.r-project.org/packages=ordinal`

Development on R-Forge:
`https://r-forge.r-project.org/projects/ordinal/`

Vignettes:

- Analysis if ordinal data with cumulative link models (32 pages)
- clm tutorial (18 pages)
- clmm tutorial (9 pages)

What is a cumulative link model (CLM)?

Ordinal data: “large”, “medium”, “small”

- Human assessments — subjective judgements (preference, grades)
- Grouped continuous, e.g., age (15-24, 25-34, 35-50)

$$\text{CLM:} \qquad \gamma_{ij} = P(Y_i \leq j) = F(\theta_j - \boldsymbol{x}_i^T \boldsymbol{\beta})$$

A regression model for an ordered variable
(Agresti, 2002; Greene and Hensher, 2010)

Intuitively:

- A logistic regression model for $J \geq 2$ (ordered) categories
- A linear model that respects the ordered categorical nature of the response

---

Ordinal data — the wine data

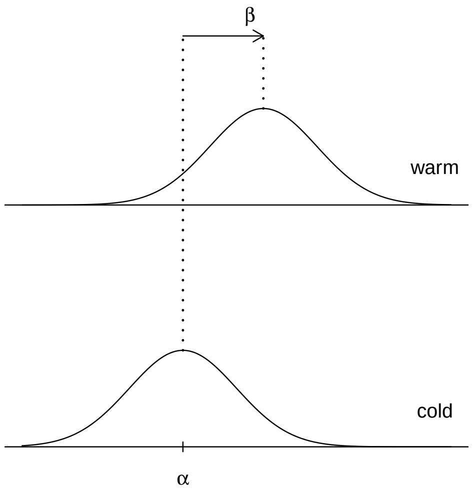
Objective:
How does perceived bitternes depend on temperature and contact?

Table: The wine data (Randall, 1989), N=72

Variables	Type	Values
bitterness	response	1, 2, 3 ,4, 5
		less — more
temperature	predictor	cold, warm
contact	predictor	no, yes
judges	random	1, . . . , 9

Temperature and contact between juice and skins can be controlled when cruching grapes during wine production.

Interpretation of the cumulative link model

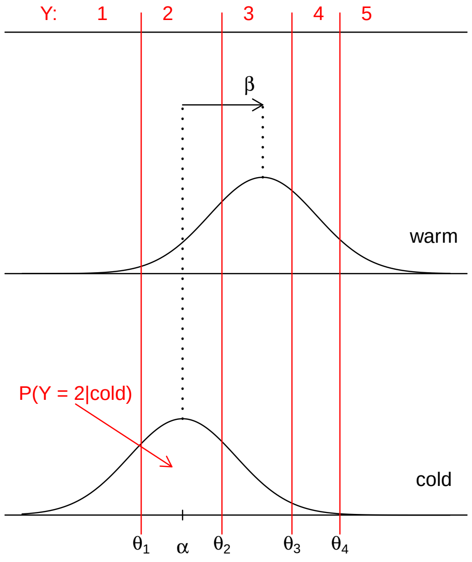


- *Latent* bitterness follows a linear model:

$$\begin{aligned} S_i &= \alpha + \boldsymbol{x}_i^T \boldsymbol{\beta} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2) \\ &= \alpha + \beta(\texttt{temp}_i) + \varepsilon_i \end{aligned}$$

- We only observe a grouped version of S_i:

Interpretation of the cumulative link model



- *Latent* bitterness follows a linear model:

$$\begin{aligned} S_i &= \alpha + \boldsymbol{x}_i^T \boldsymbol{\beta} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2) \\ &= \alpha + \beta(\texttt{temp}_i) + \varepsilon_i \end{aligned}$$

- We only observe a grouped version of S_i:
- $\theta_{j-1} \leq S_i < \theta_j \rightarrow Y = j$

$$P(Y_i \leq j) = F(\theta_j - \boldsymbol{x}_i^T \boldsymbol{\beta})$$

A teaser — Fitting cumulative link models with `clm`

```
> data(wine)
> fm1 <- clm(rating ~ contact + temp, data=wine, link="probit")
> summary(fm1)
```

```
formula: rating ~ contact + temp
data:    wine

 link   threshold nobs logLik AIC    niter max.grad cond.H
 probit flexible  72   -85.76 183.52 5(0)  1.53e-13 2.2e+01

Coefficients:
           Estimate Std. Error z value Pr(>|z|)
contactyes   0.8677     0.2669   3.251  0.00115 **
tempwarm     1.4994     0.2918   5.139 2.77e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Threshold coefficients:
    Estimate Std. Error z value
1|2  -0.7733     0.2829  -2.734
2|3   0.7360     0.2499   2.945
3|4   2.0447     0.3218   6.353
4|5   2.9413     0.3873   7.595
```

Likelihood ratio tests of CLMs

```
> fm2 <- update(fm1, ~.-temp)
> anova(fm1, fm2)
```

```
Likelihood ratio tests of cumulative link models:

    formula:                link:  threshold:
fm2 rating ~ contact        probit flexible
fm1 rating ~ contact + temp probit flexible

    no.par    AIC   logLik LR.stat df Pr(>Chisq)
fm2      5 210.05 -100.026
fm1      6 183.52  -85.761  28.529  1  9.231e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What is unique about the implementation in ordinal?

Extensive model framework

Efficient computational methods

Convergence assessment

Carefully designed printing

Functions (exported) in ordinal

Fitting	Miscellaneous	Former impl.	Distributions
`clm`	`convergence`	`clm2`	`[pdqrg]gumbel`^{C}
`clmm`^{C}	`slice`	`clmm2`^{C}	`[pdg]lgamma`^{C}
`clm.fit`	`drop.coef`	`clm2.control`	`gnorm`^{C}
`clm.control`		`clmm2.control`	`glogis`^{C}
`clmm.control`			`gcauchy`^{C}

^{C}: Implementations in C

Methods for `clm` objects

Extractor and Print		Inference	Checking
coef	print	anova	slice
fitted	summary	drop1	convergence
logLik	model.frame	add1	
nobs	model.matrix	confint	
vcov	update	profile	
AIC, BIC		predict	
extractAIC		step, stepAIC	

An extended CLM Framework

Standard CLM:

$$F(\theta_j - \boldsymbol{x}_i^T \boldsymbol{\beta})$$

Extended CLM:

$$F\left(\frac{g(\theta_j) - \boldsymbol{w}_i^T \boldsymbol{\beta}_j^* - \boldsymbol{x}_i^T \boldsymbol{\beta}}{\exp(\boldsymbol{z}_i^T \boldsymbol{\zeta})}\right)$$

threshold effects

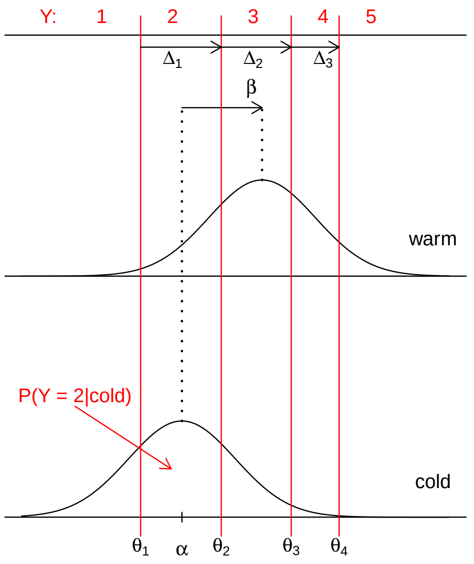
nominal effects

scale effects

CLMM (Mixed effects):

$$F(\theta_j - \overset{\text{fixed}}{\boldsymbol{X}\boldsymbol{\beta}} - \overset{\text{random}}{\boldsymbol{Z}\boldsymbol{b}})$$

Thresholds: impose restrictions

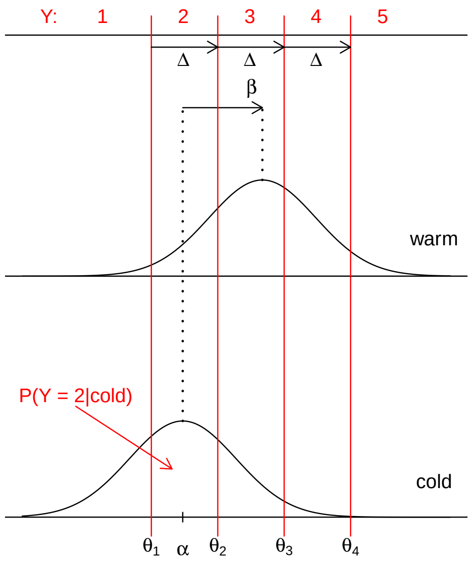


- The cumulative link model:

  $$P(Y_i \leq j) = F(\theta_j - \beta(\texttt{temp}_i))$$

- θ_j — ordered, but otherwise not restricted
- require symmetry?
- require equidistance?

Thresholds: impose restrictions



- The cumulative link model:

  $$P(Y_i \leq j) = F(\theta_j - \beta(\texttt{temp}_i))$$

- θ_j — ordered, but otherwise not restricted
- require symmetry?
- require equidistance?

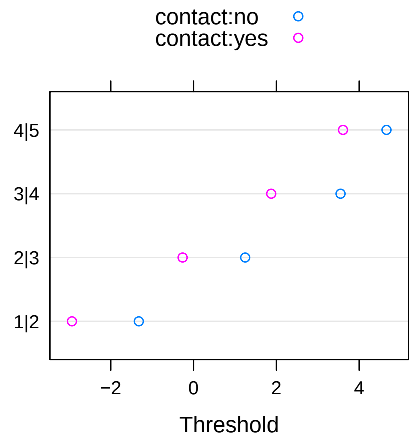
Thresholds: impose restrictions

```
> fm.equi <- clm(rating ~ contact + temp, data=wine,
                link="probit", threshold="equidistant")
> summary(fm.equi)
formula: rating ~ contact + temp
data:    wine

 link   threshold   nobs logLik AIC    niter max.grad cond.H
 probit equidistant 72   -87.24 182.47 4(0)  1.40e-08 3.2e+01

Coefficients:
           Estimate Std. Error z value Pr(>|z|)
contactyes   0.8571     0.2645   3.241  0.00119 **
tempwarm     1.4891     0.2882   5.166 2.39e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Threshold coefficients:
            Estimate Std. Error z value
threshold.1  -0.5865     0.2326  -2.522
spacing       1.2415     0.1284   9.668
```

Thresholds: impose restrictions

```
> fm.equi <- clm(rating ~ contact + temp, data=wine,
                link="probit", threshold="equidistant")
> fm.flex <- clm(rating ~ contact + temp, data=wine,
                link="probit")
> anova(fm.flex, fm.equi)
Likelihood ratio tests of cumulative link models:

        formula:                link:  threshold:
fm.equi rating ~ contact + temp probit equidistant
fm.flex rating ~ contact + temp probit flexible

        no.par    AIC  logLik LR.stat df Pr(>Chisq)
fm.equi      4 182.47 -87.237
fm.flex      6 183.52 -85.761  2.9515  2     0.2286
```

Nominal effects: relax restrictions



- The cumulative link model:

  $\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$

- Nominal effects:

  $\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_{2j}(\texttt{contact}_i))$

- This is:

  "partial proportional odds"

  (Peterson and Harrell Jr., 1990)

- Treatment coding:

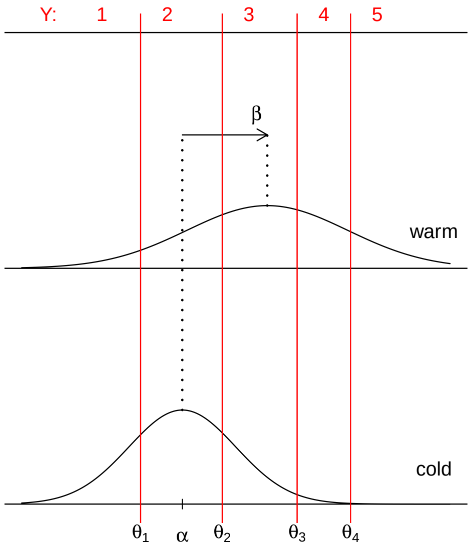
  θ_j: for contact: no

  β_{2j}: contact: yes – no

Fitting nominal effects with `clm`

```
> fm3 <- clm(rating ~ temp, nominal=~contact, data=wine, link="probit")
> summary(fm3)
formula: rating ~ temp
nominal: ~contact
data:    wine

 link   threshold nobs logLik AIC    niter max.grad cond.H
 probit flexible  72   -85.33 188.65 5(0)  3.73e-13 3.9e+01
.....

Threshold coefficients:
                Estimate Std. Error z value
1|2.(Intercept)  -0.7829     0.3178  -2.464
2|3.(Intercept)   0.7521     0.2707   2.779
3|4.(Intercept)   2.1323     0.3674   5.804
4|5.(Intercept)   2.7544     0.4512   6.105
1|2.contactyes   -0.8229     0.5650  -1.457
2|3.contactyes   -0.8892     0.3431  -2.592
3|4.contactyes   -1.0094     0.3797  -2.659
4|5.contactyes   -0.5818     0.4800  -1.212
```

Including scale effects



Model for latent bitterness:

$$S_i = \alpha + \beta_1(\texttt{temp}_i) + \beta_2(\texttt{contact}_i) + \varepsilon_i,$$

$$\varepsilon_i \sim N(0, \sigma^2(\texttt{temp}_i))$$

(Cox, 1995)

$$\gamma_{ij} = F\left(\frac{\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i)}{\zeta_1(\texttt{temp}_i)}\right)$$

Outline

1. The sensR package
2. The ordinal package — overview
3. Implementation in the ordinal package
4. Assessment of estimation accuracy
5. Cumulative link mixed models (CLMMs)

ML estimation of CLMs

Objective: Optimize the log-likelihood function

$$\ell(\boldsymbol{\theta}, \boldsymbol{\beta}; \boldsymbol{y}) = \sum_{i=1}^{n} w_i \log \pi_i$$

$$\pi_i = \gamma_{ij} - \gamma_{i,j-1} \qquad \gamma_{ij} = F(\theta_j - \boldsymbol{x}_i^T \boldsymbol{\beta})$$

- Accurately
- Reliably
- Fast

Approaches to ML estimation of CLMs

Conventional approaches:

- IRLS for multivariate GLMs (Fahrmeir and Tutz, 2001)
  → `vglm` in **VGAM** (Yee, 2010)
- General purpose optimization (quasi-Newton)
  → `polr` from **MASS** using `optim` (Venables and Ripley, 2002).

Approach in **ordinal**:

- CLM-specific Newton-Raphson algorithm

IRLS for multivariate GLMs:

Solve for $\boldsymbol{\psi} = [\boldsymbol{\theta}^T, \boldsymbol{\beta}^T]^T$:

$$\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X} \boldsymbol{\psi} = \boldsymbol{X} \boldsymbol{W} \boldsymbol{z}$$

Size is important here!

$\boldsymbol{\psi}$ is $q + p = r$
$\boldsymbol{X}$ is $nq \times r$
$\boldsymbol{W}$ is block diagonal with n blocks of $q \times q$ — in total $nq \times nq$

A large computational problem

Implementation — the approach

Key aspects of the implementation in `clm`:

- A novel matrix expression of CLMs (this is key!)
- ML Estimation via a Newton-Raphson algorithm
- Parameters updated in an **R**-environment

A novel matrix expression of CLMs

From:

$$\gamma_{ij} = F(\theta_j - \boldsymbol{x}_i^T \boldsymbol{\beta})$$

To:

$$\boldsymbol{\gamma}_k = F(\boldsymbol{B}_k \boldsymbol{\psi} + \boldsymbol{o}_k) \qquad k = 1, 2$$

$\boldsymbol{B}_k$ and $\boldsymbol{o}_k$ are fixed — generate them once!

Why?

- It leads to a fast and simple algorithm
- Gradient is simple and fast
- Hessian is simple and fast
- Covers extended model framework

A few details on matrix the expression

Step 1: Change index $j \to k$; $j = Y_i - k + 1$ where $k = 1, 2$

$$\gamma_{ik} = F(\eta_{ik}) \qquad \eta_{ik} = \theta_{ik} - \boldsymbol{x}_i^T \boldsymbol{\beta}$$

Step 2: Generate design matrices:

$$\boldsymbol{\gamma}_k = F(\boldsymbol{\eta}_k) \qquad \boldsymbol{\eta}_k = \boldsymbol{A}_k \boldsymbol{\theta} - \boldsymbol{X} \boldsymbol{\beta} + \boldsymbol{o}_k$$

Step 3: Concatenate design matrices:

$$\boldsymbol{\eta}_k = \boldsymbol{B}_k \boldsymbol{\psi} + \boldsymbol{o}_k$$

Generating matrices in R

Initialize environment:

```
> rho <- new.env(parent = parent.frame())
```

Generate $\boldsymbol{o}_k$ from y (a factor):

```
> A <- 1 * (col(matrix(0, n, nlevels(y))) == c(unclass(y)))
> rho$o1 <- c(1e5 * A[, nlevels(y)])
> rho$o2 <- c(-1e5 * A[,1])
```

Generate $\boldsymbol{A}_k$:

```
> A1 <- A[, -(ntheta + 1), drop = FALSE]
> A2 <- A[, -1, drop = FALSE]
```

Assign $\boldsymbol{B}_k = [\boldsymbol{A}_k, -\boldsymbol{X}]$ to rho:

```
> rho$B1 <- cbind(A1, -X)
> rho$B2 <- cbind(A2, -X)
```

A (modified) Newton-Raphson algorithm

The Newton step:

$$\boldsymbol{\psi}^{(i+1)} = \boldsymbol{\psi}^{(i)} - \alpha \boldsymbol{h} \qquad \boldsymbol{H}(\boldsymbol{\psi}^{(i)})\boldsymbol{h} = \boldsymbol{g}(\boldsymbol{\psi}^{(i)})$$

- Step halving: $\alpha \leftarrow \alpha/2$ in case of "overshoot"
- Stop when: $\max |\boldsymbol{g}(\boldsymbol{\psi})| < \varepsilon$ (default: $\varepsilon = 10^{-6}$)

Why is NR good for CLM estimation?

- NR step is in right direction (log-likelihood is concave)
- Quadratic convergence
- Gradient and Hessian are easy and fast to compute

The negative log-likelihood

The cumulative link model:

$$\ell(\boldsymbol{\psi}; \boldsymbol{y}) = \sum_{i=1}^{n} w_i \log \pi_i \qquad \boldsymbol{\pi} = F(\boldsymbol{\eta}_1) - F(\boldsymbol{\eta}_2) \qquad \boldsymbol{\eta}_k = \boldsymbol{B}_k \boldsymbol{\psi} + \boldsymbol{o}_k$$

```
> clm.nll <- function(rho) { ## negative log-likelihood
   with(rho, {
     eta1 <- drop(B1 %*% par) + o1
     eta2 <- drop(B2 %*% par) + o2
     fitted <- pfun(eta1) - pfun(eta2)
     if(all(fitted > 0))
       -sum(wts * log(fitted))
     else Inf
   })
 }
```

The gradient

$$\boldsymbol{g}(\boldsymbol{\psi}; \boldsymbol{y}) = \boldsymbol{C}^T \boldsymbol{\varpi}$$

$$\boldsymbol{C}^T = \boldsymbol{B}_1^T \boldsymbol{\Phi}_{1_1} - \boldsymbol{B}_2^T \boldsymbol{\Phi}_{1_2}$$

- $\boldsymbol{\Phi}$ are $n \times n$ diagonal
- A simple cross product

```
> clm.grad <- function(rho) { ## gradient of the negative log-likelihood
   with(rho, {
     p1 <- dfun(eta1)
     p2 <- dfun(eta2)
     wtpr <- wts/fitted
     dpi.psi <- B1 * p1 - B2 * p2
     -crossprod(dpi.psi, wtpr)
   })
 }
```

The Hessian

$$\boldsymbol{H}(\boldsymbol{\psi}; \boldsymbol{y}) = \boldsymbol{B}_1^T \boldsymbol{\Phi}_{2_1} \boldsymbol{B}_1 - \boldsymbol{B}_2^T \boldsymbol{\Phi}_{2_2} \boldsymbol{B}_2 - \boldsymbol{C}^T \boldsymbol{\Phi}_3 \boldsymbol{C}$$

- $\boldsymbol{\Phi}$ are $n \times n$ diagonal
- Simple cross products

```
> clm.hess <- function(rho) { ## hessian of the negative log-likelihood
   with(rho, {
     dg.psi <- crossprod(B1 * gfun(eta1) * wtpr, B1) -
       crossprod(B2 * gfun(eta2) * wtpr, B2)
     -dg.psi + crossprod(dpi.psi, (dpi.psi * wtpr / fitted))
   })
 }
```

How does this estimation routine handle the extended model framework?

Structured thresholds

$$\eta_{ij} = g(\theta_j) - \boldsymbol{x}_i^T \boldsymbol{\beta}$$

Step 1: Define Jacobian $\boldsymbol{J}$ for the transformation: $\boldsymbol{\theta} = \boldsymbol{J}\boldsymbol{\alpha}$
Step 2: Redefine $\boldsymbol{\psi} = [\boldsymbol{\alpha}^T, \boldsymbol{\beta}^T]^T$ and $\boldsymbol{B}_k = [\boldsymbol{A}_k \boldsymbol{J}^T, -\boldsymbol{X}]$

Result:

- The model can still be written as: $\boldsymbol{\eta}_k = \boldsymbol{B}_k \boldsymbol{\psi} + \boldsymbol{o}_k$
- The algorithm does not change!
- The log-likelihood, the gradient and the Hessian apply unchanged!

```
> B1 <- cbind(A1 %*% tJac, -X)
> B2 <- cbind(A2 %*% tJac, -X)
```

Nominal effects

$$\eta_{ij} = \theta_j - \boldsymbol{w}_i^T \boldsymbol{\beta}_j^* - \boldsymbol{x}_i^T \boldsymbol{\beta}$$

Step 1: $\boldsymbol{W}$ is design matrix for a single factor or covariate
Step 2: Define:

$$\underset{n \times qs}{\boldsymbol{D}_k} = \underset{n \times q}{\boldsymbol{A}_k} : \underset{n \times s}{\boldsymbol{W}}$$

Step 3: Redefine $\boldsymbol{B}_k = [\boldsymbol{D}_k, -\boldsymbol{X}]$

Result:

- The model can still be written as: $\boldsymbol{\eta}_k = \boldsymbol{B}_k \boldsymbol{\psi} + \boldsymbol{o}_k$
- The algorithm does not change!
- The log-likelihood, the gradient and the Hessian apply unchanged!

```
> tmp1 <- lapply(1:ncol(NOM), function(x) A1 * NOM[,x])
> B1 <- do.call(cbind, tmp1)
```

Scale effects

$$\eta_{ij} = \frac{\theta_j - \boldsymbol{x}_i^T\boldsymbol{\beta}}{\exp(\boldsymbol{z}_i^T\boldsymbol{\zeta})}$$

In matrices: $\boldsymbol{\eta}_k = \boldsymbol{\Upsilon}(\boldsymbol{B}_k\boldsymbol{\psi} + \boldsymbol{o}_k), \qquad \boldsymbol{\Upsilon}_{ii} = \exp(-\boldsymbol{Z}\boldsymbol{\zeta})_i$

Gradient: $\boldsymbol{g} = [\boldsymbol{C}_2, \boldsymbol{C}_3]^T\boldsymbol{\varpi}$

Hessian: $\boldsymbol{H} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{E}^T \\ \boldsymbol{E} & \boldsymbol{F} \end{bmatrix}$

Result:

- The log-likelihood, the gradient and the Hessian are slightly more complicated
- The algorithm changes slightly

Outline

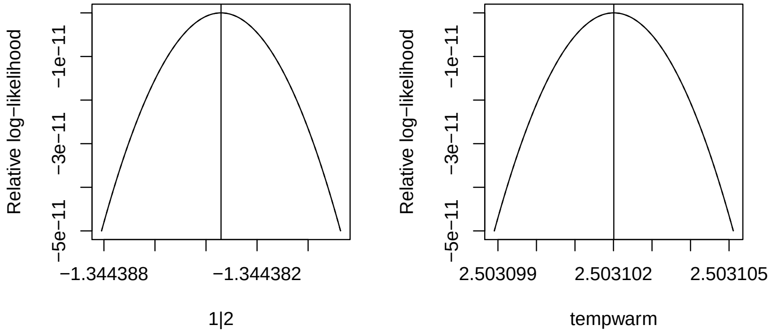
Accuracy of parameter estimates

```
> (fm1 <- clm(rating ~ temp + contact, data=wine, link="probit"))
formula: rating ~ temp + contact
data:    wine

 link   threshold nobs logLik AIC    niter max.grad
 probit flexible  72   -85.76 183.52 5(0)  1.59e-13

Coefficients:
  tempwarm contactyes
    1.4994     0.8677

Threshold coefficients:
    1|2     2|3     3|4     4|5
-0.7733  0.7360  2.0447  2.9413
```

- Has the model converged?
- How accurate are these estimates?

Assessment of model convergence

```
> slice.fm1 <- slice(fm1, parm = c(1, 6))
> par(mfrow=c(1,2))
> plot(slice.fm1)
```



See vignette for more details.

Assessment of parameter accuracy

```
> convergence(fm1)
 nobs logLik niter max.grad cond.H  logLik.Error
 72   -85.76 5(0)  1.59e-13 2.2e+01 <1e-10

           Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
1|2         -0.7733  0.2829  1.86e-14  3.60e-16      15      15
2|3          0.7360  0.2499  1.38e-13 -4.53e-16      15      15
3|4          2.0447  0.3218 -1.59e-13 -8.48e-15      13      14
4|5          2.9413  0.3873  2.66e-15 -7.40e-15      13      14
tempwarm     1.4994  0.2918 -9.83e-15 -4.64e-15      14      15
contactyes   0.8677  0.2669 -5.50e-15 -2.61e-15      14      14

Eigen values of Hessian:
61.616 53.876 32.283 17.241 13.393  2.825
```

"The Method Independent Error Theorem" (Eldén et al., 2004)

Robustness of starting values

- Standard starting values can fail:

```
> data(iris)
> iris.polr <- polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
                  Petal.Width, data=iris)
Error in polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Widt
  attempt to find suitable starting values failed
In addition: Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Robustness of starting values

- Standard starting values can fail:

```
> data(iris)
> iris.polr <- polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
                  Petal.Width, data=iris)
Error in polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Widt
  attempt to find suitable starting values failed
In addition: Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

- This runs fine, though:

```
> set.seed(1)
> iris.polr <- polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
             Petal.Width, data=iris, start = runif(6), Hess=TRUE)
```

Robustness of starting values

- Standard starting values can fail:

```
> data(iris)
> iris.polr <- polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
                  Petal.Width, data=iris)
Error in polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Widt
  attempt to find suitable starting values failed
In addition: Warning messages:
1: glm.fit: algorithm did not converge
2: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

- This runs fine, though:

```
> set.seed(1)
> iris.polr <- polr(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
             Petal.Width, data=iris, start = runif(6), Hess=TRUE)
```

- and so does:

```
> iris.clm <- clm(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
             Petal.Width, data=iris)
```

Comparing parameter estimates

`polr`:

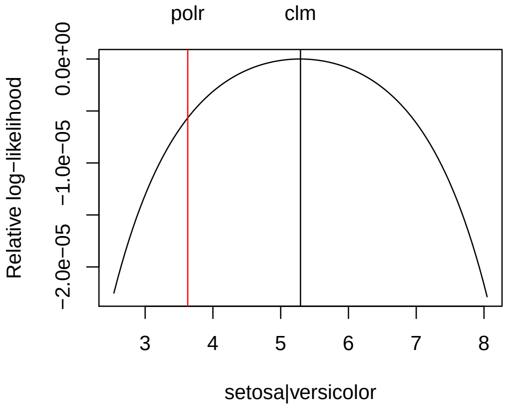
```
                       Value Std. Error
Sepal.Length          -2.464      2.393
Sepal.Width           -6.681      4.480
Petal.Length           9.427      4.734
Petal.Width           18.286      9.739
setosa|versicolor      3.629      0.013
versicolor|virginica  42.631     25.670
```

`clm`:

```
                     Estimate Std. Error
Sepal.Length           -2.465      2.394
Sepal.Width            -6.681      4.479
Petal.Length            9.429      4.737
Petal.Width            18.286      9.742
setosa|versicolor       5.292    550.912
versicolor|virginica   42.638     25.707
```

Comparing parameter estimates

```
> plot(slice(iris.clm, parm=1, lambda=5e-3, quad=FALSE))
> abline(v=iris.polr$zeta[1], col="red")
> mtext(c("polr", "clm"), at=c(iris.polr$zeta[1], iris.clm$alpha[1]),
        line=1)
```



Assessing the accuracy of parameter estimates

```
> convergence(iris.clm)
 nobs logLik niter max.grad cond.H  logLik.Error
 150  -5.95  18(0) 1.56e-07 4.0e+07 <1e-10

                     Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
setosa|versicolor       5.292 550.912  2.23e-08  6.75e-03       1       2
versicolor|virginica   42.638  25.707  5.79e-12 -1.79e-05       4       6
Sepal.Length           -2.465   2.394 -1.41e-07  2.34e-07       6       7
Sepal.Width            -6.681   4.479 -4.00e-08  2.61e-06       5       6
Petal.Length            9.429   4.737 -1.56e-07 -3.14e-06       5       6
Petal.Width            18.286   9.742 -5.59e-08 -6.69e-06       4       6

Eigen values of Hessian:
1.329e+02 1.686e-01 6.959e-02 1.933e-02 1.367e-03 3.295e-06
```

Assessing the accuracy of parameter estimates

```
> iris.clm2 <- update(iris.clm, gradTol = 1e-07)
> convergence(iris.clm2)
 nobs logLik niter max.grad cond.H  logLik.Error
 150  -5.95  19(0) 6.59e-11 4.0e+07 <1e-10

                     Estimate Std.Err  Gradient     Error Cor.Dec Sig.Dig
setosa|versicolor       5.286 550.920  4.55e-13  1.36e-07       6       7
versicolor|virginica   42.638  25.707  7.13e-14 -1.11e-08       7       9
Sepal.Length           -2.465   2.394 -2.13e-11  1.45e-10       9      10
Sepal.Width            -6.681   4.479  1.07e-11  1.63e-09       8       9
Petal.Length            9.429   4.737 -6.59e-11 -1.96e-09       8       9
Petal.Width            18.286   9.742 -2.50e-11 -4.16e-09       8      10

Eigen values of Hessian:
1.329e+02 1.686e-01 6.959e-02 1.933e-02 1.367e-03 3.295e-06
```

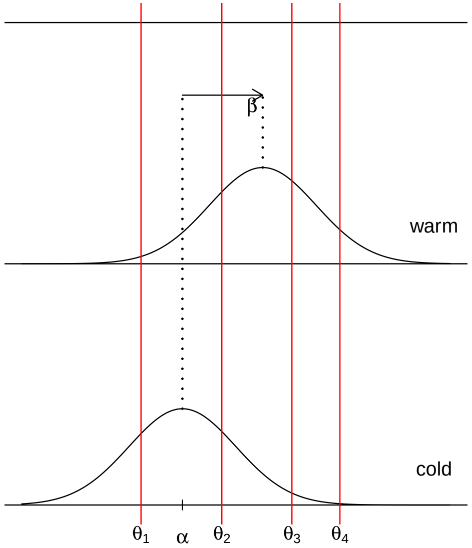
“Silent divergence” is an important issue!

See also (Marschner, 2011) for similar issues with `glm`.

Outline

1. The sensR package
2. The ordinal package — overview
3. Implementation in the ordinal package
4. Assessment of estimation accuracy
5. Cumulative link mixed models (CLMMs)

Including random effects



The cumulative link model:

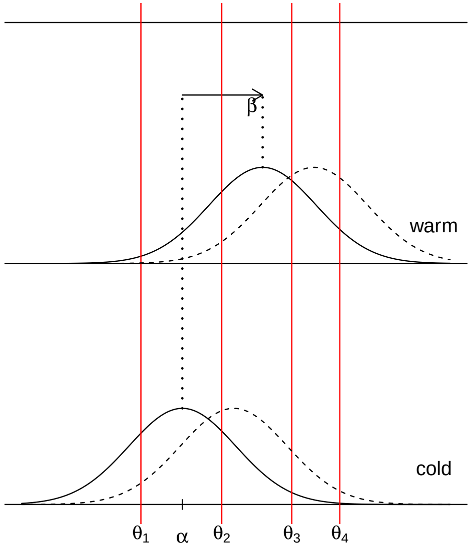
$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$$

- Judges perceive wine bitterness differently
- Judges use the response scale differently

Add random effects for judges:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i) - b(\texttt{judge}_i)), \quad b \sim N(0, \sigma_b^2)$$

Including random effects



The cumulative link model:

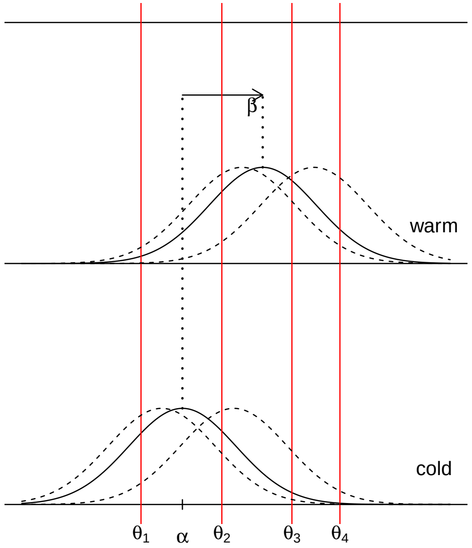
$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$$

- Judges perceive wine bitterness differently
- Judges use the response scale differently

Add random effects for judges:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i) - b(\texttt{judge}_i)), \quad b \sim N(0, \sigma_b^2)$$

Including random effects



The cumulative link model:

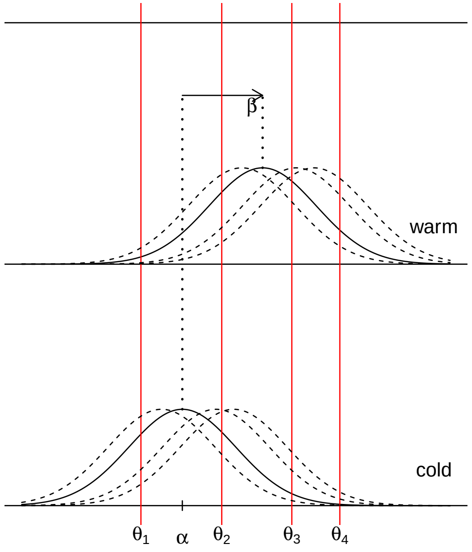
$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$$

- Judges perceive wine bitterness differently
- Judges use the response scale differently

Add random effects for judges:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i) - b(\texttt{judge}_i)), \quad b \sim N(0, \sigma_b^2)$$

Including random effects



The cumulative link model:

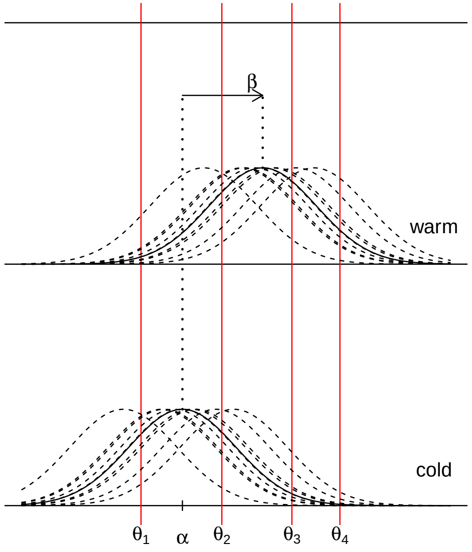
$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$$

- Judges perceive wine bitterness differently
- Judges use the response scale differently

Add random effects for judges:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i) \\ -b(\texttt{judge}_i)), \quad b \sim N(0, \sigma_b^2)$$

Including random effects



The cumulative link model:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i))$$

- Judges perceive wine bitterness differently
- Judges use the response scale differently

Add random effects for judges:

$$\gamma_{ij} = F(\theta_j - \beta_1(\texttt{temp}_i) - \beta_2(\texttt{contact}_i) \\ -b(\texttt{judge}_i)), \quad b \sim N(0, \sigma_b^2)$$

Fitting cumulative link mixed models with `clmm`

```
> fm.ran <- clmm(rating ~ temp + contact + (1|judge), nAGQ=10, data=wine)
> summary(fm.ran)
Cumulative Link Mixed Model fitted with the adaptive Gauss-Hermite
quadrature approximation with 10 quadrature points

formula: rating ~ temp + contact + (1 | judge)
data:    wine

 link  threshold nobs logLik AIC    niter   max.grad cond.H
 logit flexible  72   -81.53 177.06 16(723) 3.23e-06 2.8e+01

Random effects:
        Var Std.Dev
judge 1.288   1.135
Number of groups:  judge 9

Coefficients:
           Estimate Std. Error z value Pr(>|z|)
tempwarm     3.0619     0.5951   5.145 2.67e-07 ***
contactyes   1.8334     0.5124   3.578 0.000346 ***
.....
```

Cumulative link mixed models

$$\boldsymbol{\gamma}_k = F(\boldsymbol{B}_k\boldsymbol{\psi} - \boldsymbol{Z}\boldsymbol{v} - \boldsymbol{o}_k) \qquad \boldsymbol{V} \sim N(\boldsymbol{0}, \boldsymbol{\Sigma}_{\boldsymbol{\tau}})$$

The log-likelihood function:

$$\ell(\boldsymbol{\psi}, \boldsymbol{\tau}; \boldsymbol{y}) = \log \int_{\mathbb{R}^r} p_{\boldsymbol{\psi}}(\boldsymbol{y}|\boldsymbol{v}) p_{\boldsymbol{\tau}}(\boldsymbol{v})\, \mathrm{d}\boldsymbol{v}$$

Integration methods:

- Laplace approximation (Tierney and Kadane, 1986; Pinheiro and Bates, 1995; Joe, 2008)
- Gauss-Hermite quadrature (GHQ) (Hedeker and Gibbons, 1994)
- Adaptive Gauss-Hermite quadrature (AGQ) (Liu and Pierce, 1994)

A Newton-Raphson algorithm updates the conditional modes of the random effects (Laplace and AGQ)

Estimation of cumulative link mixed models

1 random term:

- implemented in C
- Laplace, GHQ, AGQ

$\geq$ 2 random terms:

- sparse matrix methods from **Matrix** (Bates and Maechler, 2012)
- exclusively in **R**
- Laplace

Speed is an issue here:
"The speed of clmm is really stunning. A tryout three-level model in GLLAMM took 3 hours, in clmm about 15 minutes."

References

Agresti, A. (2002). *Categorical Data Analysis* (Second ed.). Wiley.

Bates, D. and M. Maechler (2012). *Matrix: Sparse and Dense Matrix Classes and Methods.* R package version 1.0-3.

Cox, C. (1995). Location-scale cumulative odds models for ordinal data: A generalized non-linear model approach. *Statistics in medicine 14*, 1191–1203.

Eldén, L., L. Wittmeyer-Koch, and H. B. Nielsen (2004). *Introduction to Numerical Computation — analysis and MATLAB illustrations.* Studentlitteratur.

Fahrmeir, L. and G. Tutz (2001). *Multivariate Statistical Modelling Based on Generalized Linear Models* (Second ed.). Springer series in statistics. Springer-Verlag New York, Inc.

Greene, W. H. and D. A. Hensher (2010). *Modeling Ordered Choices: A Primer.* Cambridge University Press.

Hedeker, D. and R. D. Gibbons (1994). A random-effects ordinal regression model for multilevel analysis. *Biometrics 50*, 933–944.

Joe, H. (2008). Accuracy of laplace approximation for discrete response mixed models. *Comput. Stat. Data Anal. 52*(12), 5066–5074.

Liu, Q. and D. A. Pierce (1994). A note on gauss-hermite quadrature. *Biometrika 81*(3), 624–629.

Marschner, I. C. (2011, December). glm2: Fitting Generalized Linear Models with Convergence Problems. *The R Journal 3*(2), 12–15.

Peterson, B. and F. E. Harrell Jr. (1990). Partial proportional odds models for ordinal response variables. *Applied Statistics 39*, 205–217.

Pinheiro, J. C. and D. M. Bates (1995). Approximations to the nonlinear mixed-effects model. *Jounal of Computational and Graphical Statistics 4*(1), 12–35.

Randall, J. (1989). The analysis of sensory data by generalised linear model. *Biometrical journal 7*, 781–793.

Tierney, L. and J. B. Kadane (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association 81*(393), 82–86.

Venables, W. N. and B. D. Ripley (2002). *Modern Applied Statistics with S* (Fourth ed.). New York: Springer. ISBN 0-387-95457-0.

Yee, T. W. (2010, 1). The vgam package for categorical data analysis. *Journal of Statistical Software 32*(10), 1–34.

Thank you for listening!