Modeling psychometric functions in R

ROSA YSSAAD-FESSELIER and KENNETH KNOBLAUCH
INSERM, U371, Bron, France
and Université Claude Bernard Lyon 1, IFR19, Bron, France

We demonstrate some procedures in the statistical computing environment R for obtaining maximum likelihood estimates of the parameters of a psychometric function by fitting a generalized nonlinear regression model to the data. A feature for fitting a linear model to the threshold (or other) parameters of several psychometric functions simultaneously provides a powerful tool for testing hypotheses about the data and, potentially, for reducing the number of parameters necessary to describe them. Finally, we illustrate procedures for treating one parameter as a random effect that would permit a simplified approach to modeling stimulus-independent variability due to factors such as lapses or interobserver differences. These tools will facilitate a more comprehensive and explicit approach to the modeling of psychometric data.

A psychometric function is used to summarize classification performance (such as detection or discrimination) from a psychophysical experiment. An observer classifies events within a limited set of response categories over a series of trials. The psychometric function is a sigmoidal curve that describes the probability of a correct classification as a function of stimulus strength (Falmagne, 1982; Klein, 2001). If there are n choices, the lower asymptote should approach $1/n$. Typically, the upper asymptote is expected to approach 1. Given a sufficiently difficult task, however, the observer might not achieve perfect performance over the realizable range of stimulus values (see, e.g., Higgins, Arditi, & Knoblauch, 1996).

The raw data consist of the numbers of trials on which the observer correctly and incorrectly classified the stimulus for a discrete number of levels of the stimulus, which typically is summarized as a proportion. Thus, it is natural to consider the data as arising from a binomial distribution. The experimenter would like to adjust the parameters of an analytic psychometric function to fit the data, in order to characterize the threshold or the precision of the observer's performance.

Logistic regression and probit analysis are two methods frequently used to model binomial data. These can be implemented as generalized linear models (GLMs; McCullagh & Nelder, 1989). These models are generalized because they extend the Gaussian linear model to the more general exponential family of distributions, of which the Gaussian and the binomial are special cases. They are linear because the logit and probit transformations (or link functions, in the terminology of GLM) are held to transform the response variable so that it is linear in its covari-

ates, thus permitting the modeling of the dependence of the psychometric function on the experimental factors as a linear model. A drawback of the logistic and probit GLMs for fitting psychometric functions is that it is necessary to transform the data, using the false alarm rate, so that the estimated probabilities span the interval (0,1), although Klein (2001) has demonstrated that when this is done appropriately, it establishes a link between the psychometric function and the statistics of signal detection theory (Green & Swets, 1966; MacMillan & Creelman, 1991).

Watson (1979) has proposed a maximum likelihood method based on a generalized nonlinear model to fit psychometric functions. The parameters of a sigmoidal function are adjusted so as to maximize the binomial likelihood of the responses of the subject. Watson used a Weibull function to describe the relation between probability correct (P) and stimulus strength (c):

$$P(c) = 1 - (1-\gamma)e^{-\left(\frac{c}{\alpha}\right)^{\beta}}, \qquad (1)$$

where α is a location parameter of the psychometric function on the stimulus axis and corresponds to a stimulus strength that yields a criterion level of performance, β is a parameter that determines how steeply the psychometric function rises, and γ is the lower asymptote. It is convenient to define the parameter α as the threshold stimulus strength.

Occasionally, observer or experimenter errors result in misclassifications within a stimulus range over which performance would be expected to be perfect. Wichmann and Hill (2001) have demonstrated that letting the upper asymptote vary as a nuisance parameter allows these lapses to be modeled and results in more stable estimates of the steepness parameter. The modified Weibull function can be written as

$$P(c) = \gamma + (1-\gamma-\lambda)\left(1 - e^{-\left(\frac{c}{\alpha}\right)^{\beta}}\right), \qquad (2)$$

where λ is the distance of the upper asymptote from 1. Wichmann and Hill have also made available programs for

fitting psychometric functions incorporating these pos-sibilities, as well as others, for a wide variety of platforms (www.bootstrap-software.com/psignifit/).

Often, however, one would like to model the dependence of the location or steepness parameters as a function of an experimental manipulation, rather than the psychometric function itself. It is common to fit the psychometric functions so as to obtain threshold estimates and then to model the threshold as a function of the experimental manipula-tion. This procedure, however, discards potentially valu-able information about observer performance. For example, suppose that the dependence of the psychometric function on radiance is assessed for each of several test wavelengths. The variation of threshold with changes in wavelength is typically used to define a spectral sensitivity. If the spectral sensitivity resembles a standard photopigment template, one might suppose that detection is mediated by a single mechanism. However, if the steepness of the psychometric functions changed as a function of test wavelength, it would indicate a failure of univariance and raise the possibility that more than one mechanism is active.

It would be more comprehensive to model the entire psychometric function and how its parameters vary as a function of the experimental variables. In some cases, modeling the data in this fashion can considerably reduce the number of parameters necessary for a good fit between model and data. Few off-the-shelf tools permit this type of extended modeling, and it has been traditional to program such models by using special purpose minimization rou-tines (Chandler, 1965; Gegenfurtner, 1992) in high-level languages, such as Fortran or C, or else by using the opti-mization tools within computational environments, such as Matlab or Mathematica.

The freeware program R is an implementation of the S programming language (Becker, Chambers, & Wilks, 1988) that provides a powerful environment for statistical computation and graphics (R Development Core Team, 2003). The purpose of this report is to demonstrate how extended modeling of the psychometric function can be performed easily in R with the use of a few, remarkably powerful functions from some R modules (or packages) developed by J. K. Lindsey and available from his Web page (popgen.unimaas.nl/~jlindsey/rcode.html). The rest of this report is divided into five sections that demonstrate (1) how to perform a maximum likelihood fit of a psycho-metric function by using a generalized nonlinear model, (2) how to compare the location parameters of two psy-chometric functions by using a linear model, (3) how to introduce differences in the steepness parameters while testing differences in the location, (4) how to fit a group of psychometric functions for which the location func-tion is constrained by a linear model, and (5) how to fit a psychometric function as a generalized nonlinear mixed effects model with one random parameter.

The analyses presented require three of Lindsey's pack-ages: gnlm, repeated, and rmutil. All calculations reported here were performed on a Powerbook Mac G4 under OS 10.3.5 using Version R-1.8.1, except where otherwise noted. It is not the purpose here to present a tutorial on
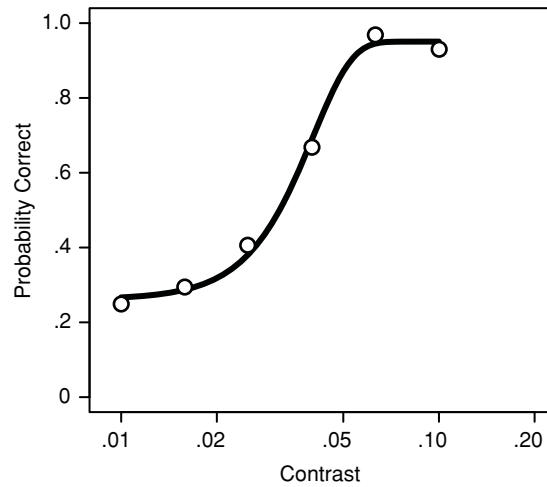

Figure 1. Psychometric function based on simulated data. The points show the proportions of correct responses as a function of contrast based on random binomial covariates, with *n* = 160 for each point and *P* determined by a Weibull function with the param-eters indicated in the text. The curve is based on the estimates from a maximum likelihood fit to the points, obtained using gnlr.

R. Familiarity with its syntax and basic commands is as-sumed. R comes with extensive documentation, and ad-ditional documents can be found on the Comprehensive R Archive Network (CRAN) Web site.

Fitting a Psychometric Function With gnlr

The gnlr() function from the gnlm package will fit a user-specified nonlinear regression to a number of one- and two-parameter probability distributions. In the case considered here, the distribution will always be binomial. For illustrative purposes, we will use the Weibull func-tion to model the psychometric function; but without a specific theoretical justification, other reasonable alterna-tives would perform as well.

Consider the following simulated data, which are bino-mial random deviates generated in R on the basis of 160 trials at each contrast level and on probabilities generated by Equation 2, with parameters set to $\alpha = 0.04$, $\beta = 3.5$, $\gamma = 0.25$, $\lambda = 0.05$:

	Contrast	NumYes	NumNo
1	0.010	40	120
2	0.016	47	113
3	0.025	65	95
4	0.040	107	53
5	0.063	155	5
6	0.100	149	11

The proportions correct based on the simulated data are plotted as circles in Figure 1. To fit a psychometric func-tion to these data requires two steps. (The code used to generate this analysis can be found in Appendix A.2.) First, define a function that describes the psychometric function with a vector, **p**, as argument to specify the input parameters. An example of how this is done in R for Equa-tion 2 is as follows:

```
wb <- function(p) {
p[3] + (1 − p[3] − p[4]) ∗ (1 − exp(−((cnt/
p[1])^exp(p[2])))) 
}
```

We have used here the trick of setting $\beta = \exp(\mathbf{p}[2])$, so that the parameter can vary along the whole real line but the estimate, $\hat{\beta}$, will always be nonnegative. A similar trick can be used to confine a parameter between two bounds, using the atan2 function. An example of this will be shown in the next section. Second, use the gnlr function to find the maximum likelihood estimates of the parameters. To fit the data in Figure 1, gnlr requires a minimum of four arguments as input: **y**, the 6×2 matrix of the responses with columns NumYes and NumNo (here, called resp .mat); distribution, the probability distribution on which the likelihood will be based; mu, the user-specified regression function (also referred to as the *location model* of the probability distribution, to be distinguished from α, one of its parameters, which was described above as the location parameter of the psychometric function); and pmu, a vector of initial estimates for the parameters. Here, we know the exact values of the psychometric function that generated these data. Normally, one would choose these by visual inspection of the data. The call and the output of the print method are shown below:

```
sim.fit <- gnlr(y = resp.mat, distribution = "binomial",
      mu = wb,
      pmu = c(0.04, log(3.4), 0.25, 0.017))
Call:
gnlr(y = resp.mat, distribution = "binomial", mu = wb,
      pmu = c(0.04, log(3.4), 0.25, 0.017))
binomial distribution
Log likelihood function:
{
    m <- mu1(p)
    −sum(wt ∗ (y[, 1] ∗ log(m) + y[, 2] ∗ log(1 − m)))
}

Location function:
{
    p[3] + (1 − p[3] − p[4]) ∗ (1 − exp(−((cnt/
    p[1])^exp(p[2])))) 
}
```

−Log likelihood	16.6
Degrees of freedom	2
AIC	20.6
Iterations	17

Location parameters:

	estimate	se
p[1]	0.04020	0.001711
p[2]	1.25675	0.197688
p[3]	0.26155	0.032378
p[4]	0.04942	0.012667

Correlations:

	1	2	3	4
1	1.0000	0.30439	0.50743	−0.22093
2	0.3044	1.00000	0.66110	0.09098
3	0.5074	0.66110	1.00000	0.03329
4	−0.2209	0.09098	0.03329	1.00000

In this case, the output is stored in a variable that we have named sim.fit. The output of gnlr is an object of class gnlm, which contains a wealth of information about the fit beyond what is printed out above and whose structure can be examined with the command str(sim.fit). Method functions exist for extracting the final estimates of the parameters, the residuals, and the values fitted to the raw data: coefficients(), residuals(), and fitted.values(), respectively. Other values can be extracted from the object, using the list extraction operator "$." For example, sim.fit$maxlike gives the final maximum likelihood estimate. When attempting to fit more complex models to the data, it may be necessary to adjust additional arguments—for example, specifying the number of iterations and the tolerance of the convergence criterion in order to obtain convergence. An example in which this was necessary is presented in the A More Complex Model of α section.

The estimate of β is obtained by taking the antilog of coefficient $\mathbf{p}[2]$ which gives $\hat{\beta} = 3.51$. To obtain a standard error for $\hat{\beta}$, the wb function can be redefined without the exp applied to the second parameter and the fit repeated with the current estimate, $\hat{\beta}$, used in pmu. The results of the fit are shown in Figure 1 as the smooth curve.

The Akaike information criterion (AIC; Akaike, 1973; Lindsey, 1999; Myung, 2000; Venables & Ripley, 2002) is defined here as the negative of the log of the likelihood plus the number of parameters and can serve as an aid in model selection. More parameters will always increase the likelihood. The AIC penalizes the likelihood by the number of parameters and so, in some sense, represents a balance between optimizing the likelihood and the parsimony of the model. Lower AIC values correspond to a better model. Unlike the likelihood, the AIC can be used to compare nonnested models. Some texts define the AIC as twice this value, but this has no effect on the model selection results.[1]

Comparing Two Psychometric Functions

Often, psychophysical data will be obtained under two different experimental conditions, and one would like to test the hypothesis that the conditions influenced the performance of the observer. For example, such influences could be changes in the position or the steepness of the functions, corresponding to changes in sensitivity or precision, respectively, of the observer. The gnlr function includes an optional argument, linear, which can be used to fit a linear model to one of the parameters of the location function, mu, assuming the other parameters to be constant, to evaluate such hypotheses in a handy and rigorous manner. To demonstrate its use here and subsequently, we will introduce a set of data obtained in a dual-judgment psychophysical task.

Yssaad-Fesselier (2001) collected data from a four-alternative, forced choice, double-judgment experiment, using the method of constant stimuli. On a given trial, the observer first reported in which of four locations (detection) a low-contrast letter of 50-msec duration was presented. Subsequently, the observer judged which of four possible letters (identification) was presented. Within a given ses-

sion, the letter height and the eccentricity of presentation in the visual field were fixed. Six contrasts were employed, and the session consisted of 192 trials. Four letter heights at each of three eccentricities were tested. Each of the 12 letter height/eccentricity conditions was repeated five times, so that each psychometric function was based on 960 judgments. The raw data from 1 observer for four letter heights presented at an eccentricity of 2º can be found in Appendix A.1. The data for the smallest letter height are plotted in Figure 2 as proportions of correct responses. The circles correspond to the detection judgments, and the triangles to identification. The identification proportion correct was calculated conditionally on a correct detection.

The solid lines in Figure 2 were obtained by fitting a Weibull function to the data from each task separately, just as in the preceding example (the code is given in Appendix A.3). The parameter estimates are presented as the first two lines of the ecc2.res.df table in Appendix A.1. The thresholds differ by about a factor of two, but $\hat{\beta}$s and $\hat{\gamma}$s are similar and might be taken as equal across the two curves. The upper asymptotes appear to be different, but this almost certainly reflects the fact that no data were collected at higher contrasts. In fact, in this example, the value of $\hat{\lambda}$ was constrained to be in the interval (0,0.05), using an arctangent transformation of this parameter in the definition of the Weibull function. Thus, it would be interesting to fit both data sets simultaneously with all of the parameters except $\hat{\alpha}$ constrained to be equal for both curves.

The linear argument of gnlr allows one to perform such a fit in a simple fashion. To exploit this feature, one could initially redefine the Weibull function to take an argument, named *linear*, that takes the place of the parameter α:

```
wb2 <- function(p,linear) {
    p[2] + (1−p[2]−atn(p[3]))∗
    (1−exp(−((cnt/exp(linear))^exp(p[1]))))
}
```
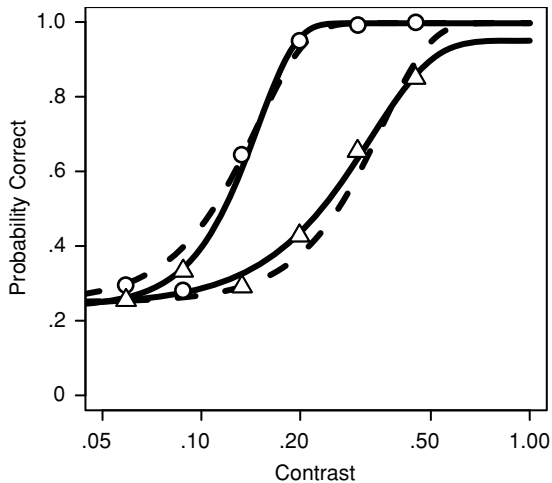

Figure 2. Psychometric functions based on detection (circles) and identification (triangles) of letters as a function of contrast from 1 observer. The solid curves are based on the maximum likelihood fits individually to each set of points. The dashed curves are based on a fit to both data sets simultaneously, with $\hat{\beta}$, $\hat{\gamma}$, and $\hat{\lambda}$ constrained to be identical for both tasks.

However, this entails renumbering the other three parameters in the input vector, **p**. We have used here the exp function to keep $\hat{\alpha}$ nonnegative, as for $\hat{\beta}$. Also, we define an atn function and its inverse, tn, which are used, respectively, to bound $\hat{\lambda}$ in the interval (0, 0.05), as has been suggested by Wichmann and Hill (2001), and as a convenience in specifying the initial values in pmu:

```
atn <- function(x) {
    (atan2(x, 1)/pi + 0.5)/20
}
tn <- function(x) {
    tan(pi ∗ (20 ∗ x − 0.5))
}
```

An alternative approach is to define the location function in the argument of gnlr as an inline function with named parameters. We will illustrate this approach here and will follow it for the rest of this article. The advantage of named parameters is that the estimates are labeled with meaningful names, rather than as array elements in the printout. The disadvantage is an increasingly complex argument in the calling function.

Next, we create a factor variable coding the levels of the two tasks, detection and identification, as they appear in the response matrix:

```
(Task <- factor(rep(c(“DET”, “ID”), 6)))
[1] DET ID DET ID DET ID DET ID DET ID DET ID
Levels: DET ID
```

This variable will appear in the argument of gnlr in the form linear = ~Task, which is interpreted as the linear model $\alpha_0 + \alpha_1$ *Task*. Thus, in the results, the value of $e^{\hat{\alpha}0}$ will correspond to the threshold for detection, and $e^{\hat{\alpha}0 + \hat{\alpha}1}$ will be the threshold for identification. The estimated standard error of $\hat{\alpha}_1$ can be used to evaluate whether this coefficient differs significantly from zero—that is, whether the thresholds differ significantly between the two curves. Finally, we use gnlr to perform the fit (see the code in Appendix A.3):

```
gnlr(y = resp.mat1, dist = “binomial”, mu =
    ~gamma + (1 − gamma − atn(tnlambda)) ∗ (1 −
    exp(−((cnt/exp(linear))^exp(logbeta)))), linear =
    ~Task, pmu = c(0.25, tn(0.01), log(0.15), log(2),
    log(3)))
```

−Log likelihood	35.7
Degrees of freedom	7
AIC	40.7
Iterations	29

Location parameters:

	estimate	se
gamma	0.2496	0.03344
tnlambda	−5.7210	6.84493
logbeta	1.0671	0.12392
(Intercept)	−1.9082	0.03638
TaskID	0.8746	0.04576

The common steepness parameter is 2.91, and the upper asymptote is 0.003. The thresholds are obtained from the

coefficients (Intercept) and TaskID, as exp(Intercept) = 0.15 and exp(Intercept + TaskID) = 0.36. These are very close to the values from the individual fits. The small standard error for the Task coefficient with respect to its magnitude attests to the significance of the difference between the location on the contrast axis of the two curves. The AIC for the constrained fit (40.7) is larger than the sum of the AICs for the individual fits (37.2) indicating that the constrained model with fewer parameters (five instead of eight) does not describe the data better. The predicted curves shown as dashes in Figure 2 suggest that a model that lets the slopes vary between tasks would perform better. We will examine how to implement this in the next section.

Comparing βs

The linear parameter could have been used in place of β to test the equality of slopes, but assuming equal values of α for both curves. The present data do not warrant such a treatment. Note that gnlr directly supports fitting a linear model only to one parameter of the location function, mu. (It also supports fitting a separate linear model to a dispersion parameter of the probability distribution, such as the variance, through an optional argument, shape.)

It is possible to fit multiple values of other variables by adding parameters in the definition of the location function, mu. For example, to fit different values of β in the example above, substitute a vector of parameters c(logb1,logb2) for the single parameter. This vector must be multiplied by an $n \times 2$ matrix, dm, where n is the total number of data points from both curves and each row is of the form either 1 0 or 0 1, as a function of the arrangement of the Task covariate in the response matrix, y. In the present case, **dm** and the argument mu are defined as follows:

```
dm <- matrix (c(2−as.vector(unclass(Task)),as.
     vector(unclass (Task))−1),
     ncol = 2)
mu = ~ gamma + (1−gamma−atn (tnlambda]))∗
     (1−exp(−((cnt/exp(linear))ˆexp
     (dm%∗%c(logb1,logb2)))))
```

The call to gnlr is similar to that in the previous example, keeping in mind that now there are initial estimates for each of six parameters: (γ, tnλ, α_0, α_1, $\log \beta_0$, $\log \beta_1$). The example call and partial printout from fitting the data in Figure 2 are shown below (see the code in Appendix A.4). Note that it is generally a good idea to base the initial estimates on those from the fits to the individual curves (first two lines of the ecc2.res.df table in Appendix A.2).

Call:
```
gnlr(y = resp.mat1, dist = "binomial", mu =
     ~gamma + (1 − gamma − atn(tnlambda)) ∗ (1−
     exp(−((cnt/exp(linear))ˆexp(dm %∗% c(logb1,
     logb2))))), linear = ~Task, pmu = c(0.24, tn(0.01),
     log(0.15), log(2.2), log(2.28), log(3.67)))
```

−Log likelihood	29.3
Degrees of freedom	6
AIC	35.3
Iterations	40

Location parameters:

	estimate	se
gamma	0.2351	0.03189
tnlambda	−4.7288	4.87143
logb1	1.2958	0.12459
logb2	0.7421	0.14415
(Intercept)	−1.9091	0.03393
TaskID	0.8541	0.04848

The values of $\hat{\beta}$ are extracted from the antilogs of the estimates logb1 and logb2 (3.65 and 2.10, respectively) and $\hat{\lambda}$ by applying the atn function, defined above, to the second parameter estimate (0.003). The values of $\hat{\alpha}$ are extracted as explained in the previous section (0.148 and 0.348). The AIC is lower for this model than for the individual fits, indicating that the best model for the data requires different values of β for detection and identification. The technique shown in this section can easily be extended to estimating parameters of several psychometric functions by adding additional columns to the dm matrix and parameters to the vector that it multiplies. This technique might be used to fit an individual λ to each curve, but an alternate method will also be presented in the Treating λ as a Random Effect section.

A More Complex Model of α

When several psychophysical functions are estimated as a function of a continuous parameter, the threshold and/or precision may vary systematically with this parameter. Accounting for such systematic variation can simplify the model by reducing the number of parameters, while revealing an underlying relation between the parameters of the psychometric function and the continuous parameter. The linear argument of gnlr is easily adapted to perform this more sophisticated analysis.

Figure 3 shows a more extensive set of conditions analyzed by Yssaad-Fesselier (2001). The data in the top panel are based on the detection task, and those in the bottom on identification. Each set of symbols corresponds to a different letter height. The circles on the far right are the data from Figure 2. Moving left, each successive data set corresponds to a larger letter height. Increasing letter height reduces the contrast necessary for detection and identification. The solid curves drawn through the data are based on fitting the Weibull function independently to each combination of size and task. The results of these fits are summarized in Appendix A.2.

Figure 4 shows how $\hat{\alpha}$ (left panel) and $\hat{\beta}$ (right panel) vary as a function of letter height. The circles indicate detection, and the triangles indicate identification. On these double logarithmic coordinates, the contrast threshold for both tasks decreases with increases in letter size. Detection requires less contrast at all letter heights, although there appears to be a tendency for the values to converge at large letter heights. There is a slight curvature in the data that has been confirmed under other conditions, as well as with other observers (see also Strasburger, Harvey, & Rentschler, 1991). These observations suggest modeling the data with a quadratic function. Using a quadratic func-
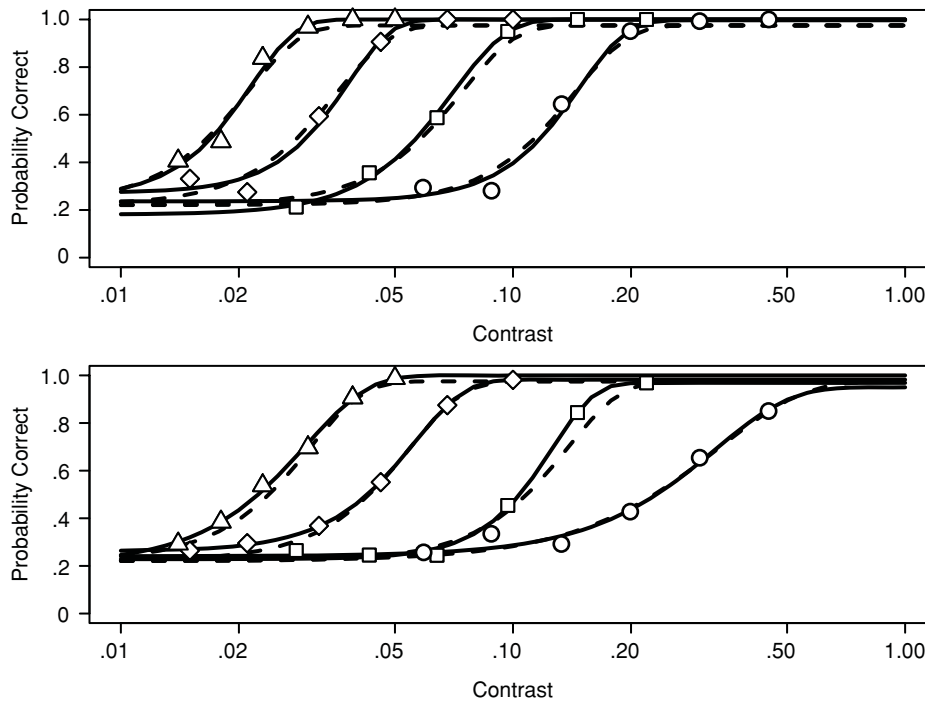
Figure 3. Psychometric functions from 1 observer for detection (upper graph) and identification (lower graph) of letters. Each symbol type refers to a different letter height: 12.4 min (circles), 20.6 min (squares), 41.3 min (diamonds), and 83 min (triangles). Data were collected at 2º eccentricity in the visual field. The solid curves are based on maximum likelihood fits separately to each data set. The dashed curves were fit constraining log($\hat{\alpha}$) to be a quadratic function of log(letter height).

tion of size to describe threshold would permit the four estimates of α to be described with three parameters. However, if data from more sizes were obtained, the number of parameters in the individual fits would increase with each new size tested, whereas the number of parameters describing α under the quadratic model would still just be three, thus gaining parsimony in the description of the data. The solid curves in the left panel in Figure 4 correspond to quadratic functions fit directly to the values of $\hat{\alpha}$ obtained from the individual fits (code in Appendix A.5).

Above, we found that identification required a smaller value of β than did detection for the smallest letter height tested. The data from other sizes do not lend support to a generalization of this observation. For this more extended set of conditions, a constant value of β may suffice, or perhaps a model that is constant except for the identification judgments at the smallest letter size.

The linear argument of gnlr is easily adapted to fitting a quadratic model, since such a model is linear in its coefficients. The formula becomes

$$\text{linear} = \text{~log10(size)} + \text{I(log10(size)^2)},$$

which corresponds to a polynomial fit with an intercept and linear and quadratic terms. It could easily be extended to higher powers, although such higher order polynomials are rarely necessary to describe data.

Powers in R model formulae are normally used to specify the order of the highest interaction term when several factors are crossed. The AsIs function, I(), must be applied

to the quadratic term so that the interpretation as a factor is inhibited and the covariate is squared. Both the detection and the identification data can be fit simultaneously with different coefficients by including the Task factor variable as before, but now defined over the full data set. A model with only a difference in intercepts would be fit by linear = ~log10(size) + I(log10(size)^2) + Task. If the coefficients differ between the two tasks, as will be the case here, the interaction terms must be included. This is accomplished by the formula linear = ~(log10(size) + I(log10(size)^2)) ∗ Task.

Initially, we fit the latter model to the data shown in Figure 3, with $\hat{\beta}$ constrained to be fixed across all sizes and tasks. The AIC was 140, higher than the summed AICs over all the independent fits (137). We will show below the call and partial output from the fit in which we constrained $\hat{\beta}$ to be equal across all conditions, except for identification, at the smallest letter height for which it was allowed to differ (see the code in Appendix A.5). The AIC is reduced to 132, indicating that the addition of this one extra parameter resulted in a better model. When each curve is fit separately, there are 32 parameters. With the quadratic constraint on $\hat{\alpha}$ and the two values of $\hat{\beta}$, only 10 parameters were needed to account for all the data.

Call:
gnlr(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
     mu = ~gamma + (1 − gamma − atn(tnlambda))
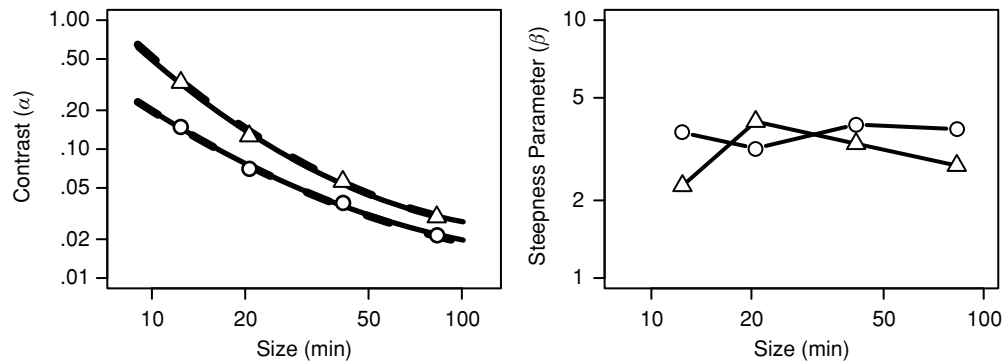   ∗ (1 − exp(−((cnt/exp(linear))^exp(dm %∗%

Figure 4. The left panel shows $\hat{\alpha}$ as a function of letter height for detection (circles) and identification (triangles) tasks. The solid curves are quadratic functions fit directly to these values. The dashed curves result from constraining $\log(\hat{\alpha})$ in the psychometric functions to be a quadratic function of log(letter height). The right panel shows estimates of $\hat{\beta}$ as a function of letter height.

c(logb1, logb2)))))), linear = ~(log10(size) + I(log10(size)^2)) * Task, pmu = c(0.23, tn(0.01), as.vector(pmu.D[1:3]), as.vector(pmu.I[1:3] − pmu. D[1:3]), log(3.2), log(2)), iterlim = 1000, steptol = 1e-05)

−Log likelihood	122	
Degrees of freedom	38	
AIC	132	
Iterations	112	

Location parameters:

	estimate	se
gamma	0.2205	0.01947
tnlambda	−5.7676	4.03531
logb1	1.1612	0.06435
logb2	0.7040	0.12334
(Intercept)	2.9763	0.40535
log10(size)	−5.7251	0.55418
I(log10(size)^2)	1.1274	0.18209
TaskID	3.0113	0.65604
log10(size):TaskID	−2.7778	0.88535
I(log10(size)^2):TaskID	0.7291	0.28787

For this more complex model, we found the results to be sensitive to the initial estimates, pmu. The six initial estimates of the coefficients of the linear term were obtained from fitting a quadratic polynomial to the values of $\hat{\alpha}$ displayed in the left graph in Figure 4. The other values were adjusted as well to ensure convergence to the smallest AIC. Note the inclusion of arguments iterlim and steptol to adjust the maximum number of iterations and the final tolerance of the stepsize in searching for a minimum, respectively.

The two estimates of $\log(\beta)$ yield $\hat{\beta}$ estimates equal to 3.21 and 2.02, respectively, the second applying only to the identification data at the smallest letter height. The Intercept, log10(size), and I(log10(size)^2) parameters are the estimates of the constant and the linear and quadratic coefficients, respectively, of the quadratic function describing the variation of $\log(\hat{\alpha})$ as a function of log(size) for the detection data. The coefficients for the identification

data are obtained by adding the detection coefficients to the last three estimates. In other words, the constant term of the identification quadratic is obtained by the sum Intercept + TaskID, the linear coefficient by log10(size) + log10(size):TaskID, and so forth.

The predicted psychometric functions under this model are shown as dashed curves in Figure 3 and appear to describe the data well for all but possibly one curve, the second to smallest letter height for the identification task. Examination of the predicted curves (dashed lines) with respect to the values of $\hat{\alpha}$ obtained from the independent fits on the left graph in Figure 4 reveals this to be an unimportant difference, however. The model fits the data very well with 22 fewer parameters than if each condition had been fit independently.

Closer examination of the coefficients with respect to their standard errors raises a question as to the significance of the interaction term I(log10(size)^2):TaskID. A model without this interaction term is easily fit using the formula ~log10(size) * Task + I(log10(size)^2) but leads to a higher AIC (134). This model generates identical parabolic curves for both tasks, which, however, differ in their vertical and horizontal positions. The larger AIC of this model is interesting because a similar relation has been proposed between contrast thresholds for sinewave gratings and for discrimination of the presence of added higher harmonics as a function of spatial frequency (Campbell & Robson, 1968) and, also, between contrast detection and reading thresholds as a function of letter size (Legge, Rubin, & Luebker, 1987).[2]

Finally, for completeness, we note that eliminating the quadratic term completely from the model raises the AIC to 184.

Treating λ as a Random Effect

In the analyses in the preceding section, the λ parameter was constrained to be equal across all eight curves. As was mentioned earlier, Wichmann and Hill (2001) have advocated treating the upper asymptote as a nuisance parameter that is free to vary for each psychometric function, in order to stabilize estimates of the steepness parameter. This can

be implemented in the same fashion as was done to fit simultaneously multiple values of the steepness parameter in the Comparing βs section. First, an $n \times q$ indicator matrix is defined, where n is the number of responses and q the number of values of λ to estimate. Then the function to be assigned to the argument mu is redefined so that the value of the coefficient for λ is replaced by a product of the indicator matrix and a vector containing the q parameters of λ to estimate. The number of additional parameters will equal the number of curves fit (here, $q = 8$). Fitting this model with two values of $\hat{\beta}$, as previously, yields an AIC = 130.1, lower than that obtained with just one value of $\hat{\lambda}$ (see the code in Appendix A.6).

This leads us to consider what would happen if we also let $\hat{\beta}$ vary for each curve. Again, we add additional parameters in the same way, with an indicator matrix times a vector of parameters. This model yields an AIC = 132.4, a less good fit than the model with only two values of $\hat{\beta}$ (see the code in Appendix A.7). Here, we see an example of overfitting, in which the increase in the number of parameters has offset the increased flexibility in the fit to raise the AIC.

In data with a greater number of conditions, the effect on the AIC of adding a parameter to each condition could be even more dramatic, leading to higher AICs just by letting λ vary as we have above. Since one of the reasons that λ is being treated as a nuisance factor is that it can vary in a fashion unrelated to the experimental conditions, it might be better to treat it as a random, rather than as a fixed, effect parameter. This approach entails estimating the parameters of the random distribution of λ, rather than the individual values. Then, as the number of conditions increases, the number of parameters associated with λ remains fixed.

Although providing a conceptually elegant solution to the estimation of λ above, the fitting of random parameters in nonnormal, nonlinear regression problems presents a daunting challenge in calculation. Estimating the maximum likelihood values for such models is nontrivial, because it involves integrating the product of the conditional probability of the responses and the random effects (or mixing) distribution to determine the marginal distribution of the responses at each step of the iterative fitting process. Analytic solutions exist only in the case of normally distributed responses with random effects and in a few other special cases (Lindsey, 1999). Lindsey's repeated library contains two functions, gnlmix and hnlmix, that permit maximum likelihood fitting of a nonlinear regression including one random parameter with a specified mixing distribution. We will demonstrate each in turn below.

gnlmix. The first of these, gnlmix, performs the integration numerically for the random parameter, which results in its being rather slow, even for small models. The function requires several arguments in addition to those used with gnlr. The mixture argument is used to select one of 11 mixture distributions. Here, the parameters of the location function, mu, are g, lambda, linear, logb1, and logb2. We have defined an atng function to constrain the value of g to the interval (0.2, 0.3). The random argument

is used to specify which parameter is to be treated as a random effect. The initial estimates of the fixed effects are specified in pmu in the order of their appearance in the definition of mu. In the present case, this is g, the six values of linear followed by logb1 and logb2. The pmix argument specifies an initial estimate for the logarithm of the dispersion parameter of the mixing distribution. Finally, the nest argument indexes the observations by the units to which the different values of the random parameter are associated. Here, these correspond to the eight combinations of size and task. An example call for a normal mixing distribution is shown below:

```
gnlmix(matrix(c(nyes, nno), ncol = 2), distribution =
    "binomial", mixture = "normal", mu = ~atng(g,
    0.1) + (1 − atng(g, 0.1) − atn(lambda)) ∗ (1 −
    exp(−((cnt/exp(linear))^exp(dm %∗% c(logb1,
    logb2)))))), random = "lambda", pmu = pmu1,
    pmix = 6, linear = ~(log10(size) + I(log10(size)^2))
    ∗ Task, nest = c(rep(1:2,6), rep(3:4, 6), rep(5:6, 6),
    rep(7:8, 6)), iterlim = 1000,steptol = 1e−05)
```

The partial output is shown below (see the code in Appendix A.8). Initial attempts to perform this fit on a Powerbook Mac G4 failed to converge after 6 h. The results shown below were obtained under Red Hat linux, Version 9.0, with a dual processor running at 1.8 GHz in just over 2 h. The fit requires 10 parameters, the 9 fixed effects parameters and 1 parameter for the dispersion of the mixing distribution. The coefficients are quite similar to those from the fixed effect model, which required 17 parameters, although the AIC is higher (132.2). Three other mixing distributions were examined, each giving nearly identical values of AIC: Cauchy 132.3, logistic 132.3, and Laplace 132.4. Choice of mixing distribution and an initial estimate for pmix can be guided by kernel density estimation applied to the values of $\hat{\lambda}$ obtained when they are fit as fixed effects.

−Log likelihood	122.1819
Degrees of freedom	38
AIC	132.1819
Iterations	133

Location parameters:

	estimate	se
g	−0.6237	0.77521
logb1	1.2385	0.05867
logb2	0.7878	0.13487
(Intercept)	2.9848	0.18795
log10(size)	−5.7217	0.25307
I(log10(size)^2)	1.1265	0.08355
Task	3.4246	0.75531
log10(size):Task	−3.4916	1.00400
I(log10(size)^2):Task	0.9890	0.32158

Mixing dispersion parameter:

estimate	se
6.067	1.448

hnlmix. The hnlmix function employs a novel and ingenious approach to modeling a random parameter that

avoids having to perform an integration (and thus achieves convergence much more rapidly than does gnlmix) by interpreting the integral as a penalized likelihood in which the random effects are estimated as fixed effects, subject to two constraints: (1) that their sum (product) equal zero (one) and (2) that their distribution follow as closely as possible the chosen mixing distribution. The procedure generalizes the h-likelihood approach of Lee and Nelder (1996) to nearly arbitrary distributions (distributions with infinite variance, such as the Cauchy, are excluded) and yields results quite similar to those obtained by fitting directly a random effects model (Lindsey, 2006).

The example call, shown below, is very similar to that of gnlmix, with two exceptions (see the code in Appendix A.9). First, the pmix argument represents the dispersion and not its logarithm, as in gnlmix. If this argument is not specified, its value is estimated during the fitting process. Second, an initial estimate of the random effect must be furnished by means of the prandom argument as either a single value or a vector with one estimate for each condition. In this case, the last estimate is ignored. Recall that the sum of the random effects will be constrained to equal zero.

```
hnlmix(matrix(c(nyes,nno),ncol = 2), distribution =
    "binomial", mixture = "normal", mu = ~atng(g) +
    (1−atng(g)−atn(lambda)) * (1−exp(−((cnt/exp
    (linear))^exp(dm%*%c(logb1,logb2))))), linear =
    ~(log10(size) + I(log10(size)^2))*Task, pmu =
    pmu1, pmix = 408, prandom = tn(0.01), random =
    "lambda", nest = c(rep(1:2,6),rep(3:4,6),rep(5:6,6),r
    ep(7:8,6)), iterlim = 1000,steptol = 1e−6)
```

The output of hnlmix, obtained on a Powerbook Mac G4 in seconds, rather than in minutes or hours, is shown below. The AIC was 129.0, lower than that for any of the other models. The model required 15 parameters, which is still less than the fixed effects model that treated λ as a fixed effect, nuisance parameter. Nine of these parameters were due to the fixed effects. The eight random effects contributed only 6 parameters, one being used for the sum constraint and the other in the estimation of the mixing distribution.

−Log likelihood	114
Penalty	32.8
Degrees of freedom	33
AIC	129
Iterations	270

Location parameters:

	estimate	se
g	−1.6924	4.65970
logb1	1.2495	0.06143
logb2	0.8032	0.13129
(Intercept)	2.9636	0.39339
log10(size)	−5.6953	0.53713
I(log10(size)^2)	1.1185	0.17639
Task	2.9349	0.68404
log10(size):Task	−2.7994	0.92318
I(log10(size)^2):Task	0.7567	0.30032

Fixed mixing shape parameter: 408

Variances: conditional = 18.7, mixing = 129

Random effect parameters:

	effect	se
1	−3.69862	3.1144
2	28.73601	10.7282
3	−7.74941	5.8836
4	−0.05468	0.7343
5	−8.02126	6.8539
6	0.11232	0.9269
7	−8.31563	6.6738
8	−1.00873	NA

Discussion

One objective of this article has been to demonstrate how an *explicit* modeling strategy can lead to a comprehensive description of the data with a minimum number of parameters. We have only scratched the surface of what is possible with the tools demonstrated above from R. For example, random effects could be used to model individual differences between subjects or variation across days within a subject. Currently, Lindsey's tools permit only a single random effect, but it would not be difficult to modify them to include multiple random effects (J. K. Lindsey, personal communication, 2004). The major limitation will be the computational time required for convergence.

Random sources of variability can be modeled in other ways. For example, if the observer cannot maintain a stable criterion (because of the difficulty of the task or perhaps because of learning effects), the probabilities estimated over sessions may vary. This situation can result in overdispersion; that is, the variability of the estimated probabilities is greater than that predicted by a binomial distribution. In such a case, one alternative is to model the likelihood as a beta binomial distribution. The three functions described above permit this, as well as several other mixture distributions (including user-defined likelihood functions), to be used in place of the binomial distribution. The AIC provides a convenient index for comparing different distributions applied to the same data (Lindsey, 1999).

More elaborate models than those shown above may be specified. Of course, modeling the data for their own sake is not the ultimate goal. The approaches demonstrated here are most powerful when they permit the differentiation of experimental hypotheses. Yssaad-Fesselier (2001) conducted similar experiments at several eccentricities in the visual field. The regression equations in the models above can be extended to include a parameter-coding eccentricity, in this fashion permitting an evaluation of whether the same model is applicable across the visual field, which corresponds to a test of a certain model of the organization of the visual system.

A second objective of this article has been to demonstrate the ease with which the type of modeling discussed here is performed in R. There are particularities of Lindsey's functions, however, that some might view as drawbacks. As the complexity of the model increases, so does the difficulty in choosing initial estimates that avoid converging to a local minimum of the negative log

likelihood. This, in fact, is a problem that is common to all nonlinear minimization routines. It is always wise to run such minimizations from multiple starting points in order to *maximize the likelihood* of finding the global minimum. A second particularity is that the way Lindsey's functions are implemented, the only variables that can be passed to the regression function are those that the functions themselves will manipulate. Other quantities that one might want to have vary across calls, such as the cnt variable in our examples, must be defined in the global environment. This means that these functions will not work correctly when called from within a function, unless the ancillary variables are defined as global variables (e.g., using the $<<-$ operator). Such a situation would arise, for example, in the implementation of a bootstrap function using gnlr in which it was necessary to call it over and over again. This is less a limitation, however, than a question of programming esthetics.

In summary, we have demonstrated how several functions from a suite of tools available in R can be exploited to model psychometric functions as a generalized nonlinear regression. A parameter can be specified as a linear model, which permits comparisons of psychometric functions across experimental conditions. In addition, the introduction of a random effect may provide an effective procedure with which to treat nuisance parameters, such as the lapse rate.

<div align="center">

REFERENCES

</div>

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov & F. Csáki (Eds.), *Second International Symposium on Inference Theory* (pp. 267-281). Budapest: Akadémia Kiadó.

Becker, R. A., Chambers, J., & Wilks, A. R. (1988). *The new S language*. London: Chapman & Hall.

Campbell, F. W., & Robson, J. G. (1968). Application of Fourier analysis to the visibility of gratings. *Journal of Physiology*, **197**, 551-566.

Chandler, J. P. (1965). *STEPIT—direct search optimization: Solutions of least squares problems* (Quantum Chemistry Program Exchange, QCEP Program No. 307). Bloomington: Indiana University, Chemistry Department.

Falmagne, J.-C. (1982). Psychometric functions theory. *Journal of Mathematical Psychology*, **25**, 1-50.

Gegenfurtner, K. R. (1992). PRAXIS: Brent's algorithm for function minimization. *Behavior Research Methods, Instruments, & Computers*, **24**, 560-564.

Green, D. M., & Swets, J. A. (1966). *Signal detection theory and psychophysics*. Huntington, NY: Krieger.

Higgins, K. E., Arditi, A., & Knoblauch, K. (1996). Detection and discrimination of mirror-image letter pairs in central and peripheral vision. *Vision Research*, **36**, 331-337.

Klein, S. A. (2001). Measuring, estimating, and understanding the psychometric function: A commentary. *Perception & Psychophysics*, **63**, 1421-1455.

Lee, Y., & Nelder, J. (1996). Hierarchical generalised linear models. *Journal of the Royal Statistical Society B*, **58**, 619-678.

Legge, G. E., Rubin, G. S., & Luebker, A. (1987). Psychophysics of reading: V. The role of contrast in normal vision. *Vision Research*, **27**, 1165-1177.

Lindsey, J. K. (1999). *Models for repeated measurements* (2nd ed.). Oxford: Oxford University Press.

Lindsey, J. K. (2006). *On h-likelihood, random effects, and penalised likelihood*. Available at popgen.unimaas.nl/~jlindsey/manuscripts.html.

Macmillan, N. A., & Creelman, C. D. (1991). *Detection theory: A user's guide*. Cambridge: Cambridge University Press.

McCullagh, P., & Nelder, J. A. (1989). *Generalized linear models*. London: Chapman & Hall.

Myung, I. J. (2000). The importance of complexity in model selection. *Journal of Mathematical Psychology*, **44**, 190-204.

R Development Core Team (2003). *R: A language and environment for statistical computing*. Vienna: R Foundation for Statistical Computing. Available at www.R-project.org.

Strasburger, H., Harvey, L. O., Jr., & Rentschler, I. (1991). Contrast thresholds for identification of numeric characters in direct and eccentric view. *Perception & Psychophysics*, **49**, 495-508.

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with S* (4th ed.). New York: Springer.

Watson, A. B. (1979). Probability summation over time. *Vision Research*, **19**, 515-522.

Wichmann, F. A., & Hill, N. J. (2001). The psychometric function: I. Fitting, sampling, and goodness of fit. *Perception & Psychophysics*, **63**, 1293-1313.

Yssaad-Fesselier, R. (2001). *Analyse psychophysique du champ visuel: Détection, identification, effet de groupement et apprentissage perceptive*. Unpublished doctoral thesis, Université Lumière Lyon 2.

<div align="center">

NOTES

</div>

1. Alternative measures for model selection, such as the Bayesion information criterion and the minimal description length may be more appropriate under certain circumstances (Myung, 2000). Since the object of this article is not to compare such measures, we will consider only the AIC, for simplicity. It is usually rather simple to calculate other measures, and in a formal analysis, they should be given serious consideration. In the case of nested (or hierarchical) models, as here, the AIC (or other measure) can be used to identify a candidate best model, and nearby models can be evaluated using a likelihood ratio test (Venables & Ripley, 2002).

2. This statement requires further elaboration. Campbell and Robson (1968) compared contrast threshold for detecting a sine-wave grating with that for discriminating whether a third harmonic at one-third contrast had been added to the same spatial frequency. The multichannel model that they were considering predicted that discrimination would be possible when the contrast of the third harmonic reached its own threshold, independently of the contrast of the fundamental. Thus, the frequency dependence of the discrimination task would follow that of the contrast threshold for a single frequency, but shifted vertically and horizontally by a factor of three along both log contrast and log frequency axes. Legge et al. (1987) performed a similar analysis in which they compared the contrast sensitivity for gratings with the contrast threshold for reading. They believed that optimal reading depended on the sensitivity to spatial frequencies up to an octave above a measure that they defined as the fundamental frequency of the letter size of the text. Thus, they expected that the reading thresholds would be shifted by a factor of two along the log frequency axis with respect to the contrast sensitivity for sine-wave gratings. Suppose that the data treated here are replotted in terms of reciprocal contrast (sensitivity) as a function of reciprocal size (a measure comparable to spatial frequency). We note that the identification curve is similar but is shifted to higher inverse sizes than is the detection curve and is of lower sensitivity. If letter identification were based on the contrast thresholds of frequencies in a fixed band above the frequencies necessary for detection, we might find that the two curves had the same shape on these axes but were simply shifted vertically and horizontally.

APPENDIX

A.1. The Data

Detection and letter identification data at 2º eccentricity: ecc2

	Contr	task	size	nyes	nno
1	0.059	DET	12.4	47	113
2	0.059	ID	12.4	12	35
3	0.088	DET	12.4	45	115
4	0.088	ID	12.4	15	30
5	0.133	DET	12.4	103	57
6	0.133	ID	12.4	30	73
7	0.199	DET	12.4	152	8
8	0.199	ID	12.4	65	87
9	0.299	DET	12.4	159	1
10	0.299	ID	12.4	104	55
11	0.449	DET	12.4	160	0
12	0.449	ID	12.4	136	24
13	0.028	DET	20.6	34	126
14	0.028	ID	20.6	9	25
15	0.043	DET	20.6	57	103
16	0.043	ID	20.6	14	43
17	0.064	DET	20.6	94	66
18	0.064	ID	20.6	23	71
19	0.097	DET	20.6	152	8
20	0.097	ID	20.6	69	83
21	0.146	DET	20.6	160	0
22	0.146	ID	20.6	135	25
23	0.219	DET	20.6	160	0
24	0.219	ID	20.6	155	5
25	0.015	DET	41.3	53	107
26	0.015	ID	41.3	14	39
27	0.021	DET	41.3	44	116
28	0.021	ID	41.3	13	31
29	0.032	DET	41.3	95	65
30	0.032	ID	41.3	35	60
31	0.046	DET	41.3	145	15
32	0.046	ID	41.3	80	65
33	0.068	DET	41.3	160	0
34	0.068	ID	41.3	140	20
35	0.100	DET	41.3	160	0
36	0.100	ID	41.3	157	3
37	0.014	DET	83.0	65	95
38	0.014	ID	83.0	19	46
39	0.018	DET	83.0	78	82
40	0.018	ID	83.0	30	48
41	0.023	DET	83.0	134	26
42	0.023	ID	83.0	72	62
43	0.030	DET	83.0	155	5
44	0.030	ID	83.0	108	47
45	0.039	DET	83.0	160	0
46	0.039	ID	83.0	145	15
47	0.050	DET	83.0	160	0
48	0.050	ID	83.0	158	2

Results of fits to individual conditions: ecc2.res.df

	alpha	beta	gamma	lambda	task	size	AIC	ML
1	0.1483	3.67	0.236	3.29e-03	Det	12.4	17.9	13.9
2	0.3283	2.28	0.241	5.00e-02	ID	12.4	19.3	15.3
3	0.0707	3.18	0.181	7.57e-05	Det	20.6	14.4	10.4
4	0.1263	4.04	0.228	3.14e-02	ID	20.6	17.6	13.6
5	0.0380	3.94	0.272	7.39e-05	Det	41.3	16.9	12.9
6	0.0560	3.32	0.262	1.79e-02	ID	41.3	17.2	13.2
7	0.0215	3.78	0.249	7.24e-05	Det	83.0	16.4	12.4
8	0.0297	2.73	0.207	7.44e-05	ID	83.0	17.7	13.7

APPENDIX (Continued)

A.2. Fit With Simulated Data

```
> library(gnlm)
> beta <- log(3.5)
> gamma <- 0.25
> lambda <- 0.05
> alpha <- 0.04
> p <- c(alpha, beta, gamma, lambda)
> num.tr <- 160
> cnt <- 10^seq(−2, −1, length = 6)
> wb <- function(p) {
+        p[3] + (1 − p[3] − p[4]) * (1 − exp(−((cnt/p[1])^exp(p[2])))))
+ }
> NumYes <- rbinom(length(cnt), num.tr, wb(p))
> NumNo <- num.tr − NumYes
> phat <- NumYes/(NumYes + NumNo)
> resp.mat <- matrix(c(NumYes, NumNo), ncol = 2)
> sim.fit <- gnlr(y = resp.mat, distribution = "binomial",
+        mu = wb, pmu = c(0.04, log(3.4), 0.25, 0.017))
```

A.3. Simple Covariate

```
> atn <- function(x) {
+        (atan2(x, 1)/pi + 0.5)/20
+ }
> tn <- function(x) {
+        tan(pi * (20 * x − 0.5)
+ }
> wb2 <- function(p, linear) {
+        p[2] + (1 − p[2] − atn(p[3])) *
+        (1 − exp(−((cnt/exp(linear))^exp(p[1]))))
+ }
> ecc2 <- read.table("ecc2.dat", header = TRUE, sep = "\t")
> subdata <- subset(ecc2, size == 12.4, select = Contr:nno)
> names(subdata) <- c("Contrast", "Task", "NumYes",
+        "NumNo")
> resp <- subset(ecc2, size == 12.4 & task == "DET",
+        select = c(NumYes, NumNo))
> cnt <- subset(ecc2, size == 12.4 & task == "DET")$Contr
> fit10D <- gnlr(y = resp, distribution = "binomial",
+        mu = wb, pmu = c(0.15, log(3.5), 0.25, tn(0.01)))
> resp <- subset(ecc2, size == 12.4 & task == "ID",
+        select = c(NumYes, NumNo))
> cnt <- subset(ecc2, size == 12.4 & task == "ID")$Contr
> fit10I <- gnlr(y = resp, distribution = "binomial",
+        mu = wb, pmu = c(0.3, log(3.5), 0.25, tn(0.1)))
> Task <- subset(ecc2, size == 12.4)$task
> cnt <- subset(ecc2, size == 12.4)$Contr
> resp.mat1 <- as.matrix(subset(ecc2, size == 12.4,
+        select = c(NumYes, NumNo)))
> fit10DID <- gnlr(y = resp.mat1, dist = "binomial",
+        mu = ~gamma + (1 − gamma − atn(tnlambda)) * (1 −
+        exp(−((cnt/exp(linear))^exp(logbeta)))),
+        linear = ~Task, pmu = c(0.25, tn(0.01), log(0.15),
+        log(2), log(3)))
```

A.4. Comparing βs

```
> dm <- matrix(c(2 − as.vector(unclass(Task)),
+        as.vector(unclass(Task)) − 1), ncol = 2)
> TwoBeta.fit <- gnlr(y = resp.mat1, dist = "binomial",
+        mu = ~gamma + (1 − gamma − atn(tnlambda)) * (1 −
+        exp(−((cnt/exp(linear))^exp(dm %*% c(logb1,
+        logb2))))), linear = ~Task, pmu = c(0.24,
+        tn(0.01), log(0.15), log(2.2), log(2.28),
+        log(3.67)))
```

APPENDIX (Continued)

A.5. Quadratic Model with Task Interaction

```
> attach(ecc2.res.df)
> sz <- unique(size)
> pmu.D <- lm(log(alpha[seq(1, 8, 2)]) ~ log10(sz) +
+         I(log10(sz)^2))$coefficients
> pmu.I <- lm(log(alpha[seq(2, 8, 2)]) ~ log10(sz) +
+         I(log10(sz)^2))$coefficients
> detach(ecc2.res.df)
> bI <- c(rep(c(1, 0), 6), rep(1, 36))
> dm <- matrix(c(bI, 1 − bI), ncol = 2)
> Task <- ecc2$task
> attach(ecc2)
> cnt <- Contr
> TxQ <- gnlr(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
+         mu = ~gamma + (1 − gamma − atn(tnlambda)) * (1 −
+         exp(−((cnt/exp(linear))^exp(dm %*% c(logb1,
+         logb2)))))), linear = ~(log10(size) +
+         I(log10(size)^2)) * Task, pmu = c(0.23, tn(0.01),
+         as.vector(pmu.D[1:3]), as.vector(pmu.I[1:3] −
+         pmu.D[1:3]), log(3.2), log(2)), iterlim = 1000,
+         steptol = 1e-05)
> detach(ecc2)
```

A.6. λ Unconstrained as a Fixed Effect

```
> bI <- c(rep(c(1, 0), 6), rep(1, 36))
> bm <- matrix(c(bI, 1 − bI), ncol = 2)
> cdm <- matrix(c(rep(c(1, 0), 6), rep(c(0, 1), 6)),
+         ncol = 2)
> dm <- cbind(rbind(cdm, matrix(0, 36, ncol = 2)),
+         rbind(matrix(0, 12, ncol = 2), cdm, matrix(0,
+         24, ncol = 2)), rbind(matrix(0, 24, ncol = 2),
+         cdm, matrix(0, 12, ncol = 2)), rbind(matrix(0,
+         36, ncol = 2), cdm))
> wb8d <- function(p, linear) {
+         p[3] + (1 − p[3] − atn(dm %*% c(p[4], p[5], p[6],
+         p[7], p[8], p[9], p[10], p[11]))) * (1 −
+         exp(−((cnt/exp(linear))^(exp(bm %*% c(p[1],
+         p[2]))))))
+ }
> pmu <- c(log(3.38), log(2.21), 0.23, tn(ecc2.res.df$lambda),
+         as.vector(pmu.D), as.vector(pmu.I − pmu.D))
> attach(ecc2)
> cnt <- Contr
> d8 <- gnlr(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
+         mu = wb8d, linear = ~(log10(size) + I(log10(size)^2)) *
+         Task, pmu = pmu, iterlim = 1000, steptol = 1e-05)
> detach(ecc2)
```

A.7. β and λ Unconstrained

```
> cdm <- matrix(c(rep(c(1, 0), 6), rep(c(0, 1), 6)),
+         ncol = 2)
> dm <- cbind(rbind(cdm, matrix(0, 36, ncol = 2)),
+         rbind(matrix(0, 12, ncol = 2), cdm, matrix(0,
+         24, ncol = 2)), rbind(matrix(0, 24, ncol = 2),
+         cdm, matrix(0, 12, ncol = 2)), rbind(matrix(0,
+         36, ncol = 2), cdm))
```

APPENDIX (Continued)

```
> wb8d8 <- function(p, linear) {
+        p[9] + (1 − p[9] − atn(dm %*% c(p[10], p[11],
+        p[12], p[13], p[14], p[15], p[16], p[17]))) *
+        (1 − exp(−((cnt/exp(linear))^(exp(dm %*%
+        c(p[1], p[2], p[3], p[4], p[5], p[6],
+        p[7], p[8])))))))
+ }
> pmu <- c(log(ecc2.res.df$beta), 0.23, tn(ecc2.res.df$lambda),
+        as.vector(pmu.D), as.vector(pmu.I − pmu.D))
> attach(ecc2)
> cnt <- Contr
> bd8 <- gnlr(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
+        mu = wb8d8, linear = ~(log10(size) + I(log10(size)^2)) *
+        Task, pmu = pmu, iterlim = 1000)
> detach(ecc2)
```

A.8. Mixed-Effect Model Fit With gnlmix

```
> library(repeated)
> Task <- ecc2$task
> atng <- function(g, rg) {
+        (0.25 − rg/2 + (atan2(g, 1)/pi + 0.5) * rg
+ }
> tng <- function(g, rg) {
+        tan(pi * ((g − (0.25) − rg/2))/rg − 0.5))
+ }
> pmu1 <- c(tng(TxQ$coef[3], 0.1), TxQ$coef[4:9], TxQ$coef[1:2])
> attach(ecc2)
> cnt <- Contr
> dI <- c(rep(c(1, 0), 6), rep(1, 36))
> dm <- matrix(c(dI, 1 − dI), ncol = 2)
> TxQg <- gnlmix(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
+        mixture = "normal", mu = ~atng(g, 0.1) + (1 − atng(g, 0.1) −
+        atn(lambda)) * (1 − exp(−((cnt/exp(linear))^exp(dm %*%
+        c(logb1, logb2))))), random = "lambda", linear = ~(log10(size) +
+        I(log10(size)^2)) * Task, pmu = pmu1, pmix = log(420),
+        nest = c(rep(1:2, 6), rep(3:4, 6), rep(5:6, 6),
+        rep(7:8, 6)), iterlim = 1000, steptol = 1e-06)
> detach(ecc2)
```

A.9. Mixed-Effect Model Fit With hnlmix

```
> Task <- ecc2$task
> attach(ecc2)
> cnt <- Contr
> dI <- c(rep(c(1, 0), 6), rep(1, 36))
> dm <- matrix(c(dI, 1 − dI), ncol = 2)
> atng <- function(g) 0.225 + atn(g)
> tng <- function(g) tn(g − 0.225)
> pmu1 <- c(tng(0.23), TxQ$coef[5:10], log(3.5), log(2.2))
> TxQh <- hnlmix(matrix(c(nyes, nno), ncol = 2), dist = "binomial",
+        mixture = "normal", mu = ~atng(g) + (1 − atng(g) −
+        atn(lambda)) * (1 − exp(−((cnt/exp(linear))^exp(dm %*%
+        c(logb1, logb2))))), random = "lambda", linear = ~(log10(size) +
+        I(log10(size)^2)) * Task, pmu = pmu, pmix = 408,
+        prandom = tn(0.01), nest = c(rep(1:2, 6), rep(3:4,
+        6), rep(5:6, 6), rep(7:8, 6)), iterlim = 1000,
+        steptol = 1e-05)
> detach(ecc2)
```