Hyper-Fit: Fitting Linear Models to Multidimensional Data with Multivariate Gaussian Uncertainties

A. S. G. Robotham and D. Obreschkow
ICRAR, M468, University of Western Australia, Crawley, WA 6009, Australia
Email: Aaron.robotham@uwa.edu.au

(Received July 01, 2015; Accepted August 10, 2015)

Abstract
Astronomical data is often uncertain with errors that are heteroscedastic (different for each data point) and covariant between different dimensions. Assuming that a set of $D$-dimensional data points can be described by a $(D - 1)$-dimensional plane with intrinsic scatter, we derive the general likelihood function to be maximised to recover the best fitting model. Alongside the mathematical description, we also release the HYPER-FIT package for the R statistical language (github.com/asgr/hyper.fit) and a user-friendly web interface for online fitting (hyperfit.icrar.org). The HYPER-FIT package offers access to a large number of fitting routines, includes visualisation tools, and is fully documented in an extensive user manual. Most of the HYPER-FIT functionality is accessible via the web interface. In this paper, we include applications to toy examples and to real astronomical data from the literature: the mass-size, Tully–Fisher, Fundamental Plane, and mass-spin-morphology relations. In most cases, the HYPER-FIT solutions are in good agreement with published values, but uncover more information regarding the fitted model.

Keywords: fitting – statistics

1 INTRODUCTION

There is a common conception that the method used for fitting multidimensional data is a matter of subjective choice. However, if the problem is mathematically well-defined, the optimal fitting solution is, in many cases, unique. This paper presents the optimal maximum likelihood fitting solution (with the caveats that a maximum likelihood fitting approach entails) for the frequently encountered case schematised in Figure 1, where $D$-dimensional data (i) have multivariate Gaussian uncertainties (these may be different for each data point), and (ii) randomly sample a $(D - 1)$-dimensional plane (a line if $D = 2$, a plane if $D = 3$, a hyperplane if $D \geq 4$) with intrinsic Gaussian scatter. In particular, the data have no defined predictor/response variable. Hence, standard regression analysis does not apply, and in general, it would yield different geometric solutions depending on axis-ordering (though see von Toussaint 2015, which presents the necessary prior modifications to make traditional intrinsic-scatter-free regression analysis rotationally invariant).

Within the assumptions above, the $(D - 1)$-dimensional plane that best explains the observed data is unique and can be fit using a traditional likelihood method. In this paper, we present the general $D$-dimensional form of the likelihood function and release a package for the R statistical programming language (HYPER-FIT) that optimally fits data using this likelihood approach. We also release a user-friendly web interface to run HYPER-FIT online and apply our algorithm to common relations in galaxy population studies.

Other authors have investigated the problem of regression and correlation analysis (Kelly 2007; Hogg, Bovy, & Lang 2010; Kelly 2011; von Toussaint 2015) with the aim of improving on the ad-hoc use of standard linear or multi-variate regression, reduced $\chi^2$ analysis, computationally expensive (and usually unnecessary) bootstrap resampling, and the ‘bisector method’ that is sometimes wrongly assumed to be invariant with respect to an arbitrary rotation of the data. Of recent work that is familiar in the astronomical community, Kelly (2007) presents a comprehensive approach for linear regression analysis with support for non-detections, and made available IDL code that can analyse multi-dimensional data with a single predictor variable. More recently, Hogg et al. (2010) describes the approach to attack such joint regression problems in two dimensions. Taylor et al. (2015) describes an approach to extend the work of Hogg et al. (2010) to multiple Gaussian mixture models to model the red and blue populations in the Galaxy And Mass Assembly survey (Liske et al. 2015). Here, we extend Hogg et al.
Figure 1. Schematic representation of the linear model (blue) fitted to the data points (red). Both the model and the data are assumed to have Gaussian distributions, representing intrinsic model scatter and statistical measurement uncertainties, respectively. In both cases, 1σ contours are shown as dashed lines.

(2010) to arbitrary dimensions and include a suite of tools to aid scientists with such analysis.

In Section 2, we describe the likelihood method for an arbitrary number of dimensions and for different projection systems (to allow for maximum flexibility in its application). In Section 3, we describe and publicly release the HYPER-FIT package for R, a user friendly and fully-documented package that can be applied to such problems. We also introduce a simple web interface for the package. In Section 4, we show simple applications of HYPER-FIT to toy examples, including the dataset presented in Hogg et al. (2010). In Section 5, we show more advanced examples taken directly from published astronomy papers, where we can compare our fitting procedure against the relationships presented in the original work. In the appendix, we add a discussion regarding the subtle biases contained in an extraction of model intrinsic scatter, and provide a route to properly account for a non-uniformly distributed data. We also illustrate the implementation of the sampler (JAGS) and provide a route to properly account for a non-uniformly distributed data.

2 MATHEMATICAL DERIVATION

This section develops the general theory for fitting data points in D dimensions by a \((D-1)\)-dimensional plane, such as fitting points in two dimensions by a line (Figure 1). As described in Section 1, we assume that the data represent a random sample of a population exactly described by a \((D-1)\)-dimensional plane with intrinsic Gaussian scatter, referred to as the ‘generative model’. The objective is to determine the most likely generative model, by fitting simultaneously for the \((D-1)\)-dimensional plane and the intrinsic Gaussian scatter, given data points with multivariate Gaussian uncertainties: the uncertainties are (1) different for each data point (heteroscedasticity), (2) independent between different data points, and (3) covariant between orthogonal directions, e.g. x-errors and y-errors can be correlated. We first present the general method in D dimensions in coordinate-free form (Section 2.1) and in Cartesian coordinates (Section 2.2). Explicit equations for the two-dimensional case are given in Section 2.3. Subtleties and generalisations are presented in the appendix.

2.1 Coordinate-free solution in D dimensions

Consider \(N\) data points \(i = 1, \ldots, N\) in D dimensions. These points are specified by the measured positions \(x_i \in \mathbb{R}^D\) relative to a fixed origin \(O \in \mathbb{R}^D\), and Gaussian measurement uncertainties expressed by the symmetric covariance matrices \(C_i\). We can then fully express our knowledge about a point \(i\) as the probability density function (PDF) \(\rho(x_i|\lambda)\), describing the probability that point \(i\) is truly at the position \(x_i\) given the measured position \(\hat{x}_i\). In the case where the intrinsic distribution of points in any direction is uniform (the case of non-uniform point distributions is discussed in Appendix B), this PDF is symmetric, i.e. \(\rho(x_i|\lambda) = \rho(x_i|x)\), and reads

\[
\rho(x_i|x) = \frac{1}{\sqrt{(2\pi)^D |C_i|}} e^{-\frac{1}{2}(x-x_i)^\top C_i^{-1}(x-x_i)}, \tag{1}
\]

In Figure 1, these PDFs are shown as red shading.

The data points are assumed to be randomly sampled from a PDF, called the generative model, defined by a \((D-1)\)-dimensional plane \(H \subset \mathbb{R}^D\) (a line if \(D = 2\), a plane if \(D = 3\), a hyperplane if \(D \geq 4\)) with Gaussian scatter \(\sigma_x\) orthogonal to \(H\). Note that the scatter remains Gaussian along any other direction, not parallel to \(H\). The blue shading in Figure 1 represents this generative model, with the solid line indicating the central line \(H\) and the dashed lines being the parallel lines at a distance \(\sigma_x\) from \(H\). The plane \(H\) is fully specified by the normal vector \(n \perp H\) going from \(O\) to \(H\), and the distance of a point \(x \in \mathbb{R}^D\) from \(H\) equals \(|\hat{n}^\top x - n|\), where \(n = |n|\) and \(\hat{n} = n/n\). Thus, \(H\) is the ensemble of points \(x\) that satisfy

\[
\hat{n}^\top x - n = 0. \tag{2}
\]

Given this parameterisation, the generative model is fully described by the PDF

\[
\rho_n(x) = \frac{1}{\sqrt{2\pi \sigma_x^2}} e^{-\frac{\hat{n}^\top(x-x_0)^2}{\sigma_x^2}}, \tag{3}
\]

expressing the probability of generating a point at position \(x\). Note that we here assume that the generative model is infinitely uniform along any dimension in the plane. If the selection function was, e.g. a power law distribution, then \(\rho(x|x_i) \neq \rho(x|x)\), and Equation (3) cannot be a simply stated. Such scenarios are discussed in Appendix B.
The likelihood of measuring data point \(i\) at position \(x_i\) then equals

\[
\mathcal{L}_i = \int_{\mathbb{R}^D} dx \rho(x|x)\rho_n(x) = \frac{1}{\sqrt{2\pi} \sigma_{x,i}^{2}} e^{-\frac{(\mathbf{n}^\top x - n)^2}{2\sigma_{x,i}^2}},
\]

where \(x_{x,i}^2 \equiv \sigma_{x,i}^2 + \mathbf{n}^\top C \mathbf{n}.\) Following the Bayesian theorem and assuming uniform priors on \(\mathbf{n}\) and \(\sigma_{x,i}\), the most likely model maximises the total likelihood \(\mathcal{L} = \prod_{i=1}^{N} \mathcal{L}_i\) and hence its logarithm \(\ln \mathcal{L}\). Up to an additive constant \(\ln 2\pi\),

\[
\ln \mathcal{L} = -\frac{1}{2} \sum_{i=1}^{N} \left( \ln \sigma_{x,i}^2 + \frac{(\mathbf{n}^\top x_i - n)^2}{\sigma_{x,i}^2} \right).
\]

In summary, assuming that the data samples a linear generative model described by Equation (3), the model-parameters that best describe the data are found by maximising the likelihood function given in Equation (5) or (6).

### 2.2 Notations in Cartesian coordinates

For plotting and further manipulation, it is often useful to work in a Cartesian coordinate system with \(D\) orthogonal coordinates \(j = 1, \ldots, D\). The typical expression of a linear model in such coordinates is

\[
x_j = \Sigma_{x,j} x_j + \beta, \tag{7}
\]

where \(x_j\) is the \(j\)-th coordinate of \(x = (x_1, \ldots, x_D)^\top\), \(\alpha_j = \partial x_j/\partial x_j\) is the slope along the coordinate \(j\), and \(\beta\) is the zero-point along the coordinate \(D\). Equation (7) can be rewritten as \(\alpha^\top x + \beta = 0\), where \(\alpha^\top \equiv (\alpha_1, \ldots, \alpha_{N-1}, -1)\) is a normal vector of the \((D-1)\)-dimensional plane. The Gaussian scatter about this plane is now parameterised with the scatter \(\sigma_D\) along the coordinate \(D\). A vector calculation shows that the transformation between the coordinate-free model parameters \([\mathbf{n}, \sigma_\perp]\) and the coordinate-dependent parameters \([\alpha, \beta, \sigma_D]\) takes the form

\[
\mathbf{n} = -\beta \alpha (\alpha^\top \alpha)^{-1}, \quad \sigma_\perp = \sigma_D (\alpha^\top \alpha)^{-1/2}
\]

or, inversely,

\[
\alpha = -\mathbf{n}/\sigma_D, \quad \beta = (\mathbf{n}^\top \mathbf{n})/\sigma_D, \quad \sigma_D = \sigma_{\perp} (\mathbf{n}^\top \mathbf{n})^{1/2}/|\mathbf{n}|.
\]

In practice, one can always fit for \([\mathbf{n}, \sigma_\perp]\) by maximising Equation (6) and then convert \([\mathbf{n}, \sigma_\perp]\) to \([\alpha, \beta, \sigma_D]\), if desired. Alternatively, we can rewrite Equation (6) directly in terms of the parameters \([\alpha, \beta, \sigma_D]\) as

\[
\ln \mathcal{L} = \frac{1}{2} \sum_{i=1}^{N} \left[ \ln \sigma_D^2 + \frac{(\alpha^\top x_i + \beta)^2}{\sigma_D^2} \right].
\]

which can also be parameterised by the normal vector \(\mathbf{n} = n(\cos \phi, \sin \phi)^\top\) going from the origin to the line. The parameters \(\alpha\), \(\beta\), and \(\sigma_\perp (\sigma_\perp = \sigma_D\) being the intrinsic scatter along \(y\)) are then given by [following Equation (9)]

\[
\alpha = -\frac{1}{\tan \phi}, \quad \beta = \frac{n}{\sin \phi}, \quad \sigma_\perp = \frac{\sigma_D}{|\sin \phi|},
\]

which manifestly diverge as the line becomes vertical (\(\phi \in \pi \mathbb{Z}\)). In these notations, Equations (5) and (10) simplify to

\[
\ln \mathcal{L} = -\frac{1}{2} \sum_{i=1}^{N} \left( \ln \sigma_\perp^2 + \frac{(\alpha x_i + \beta - n)^2}{\sigma_\perp^2} \right) = \frac{1}{2} \sum_{i=1}^{N} \left( \ln \alpha_{\perp,i}^2 + \frac{(\alpha x_i - \beta)^2}{\sigma_{\perp,i}^2} \right), \tag{13}
\]

where \(\alpha_{\perp,i}^2 \equiv \sigma_\perp^2 + \sigma_\perp^2 \cos^2 \phi + \sigma_\perp^2 \sin^2 \phi + 2\sigma_\perp \sigma_\perp \cos \phi \sin \phi\) and \(\sigma_{\perp,i}^2 \equiv \sigma_\perp^2 + \sigma_\perp^2 \cos^2 \phi - 2\sigma_\perp \sigma_\perp \cos \phi \sin \phi\) (reminder: \(\sigma_\perp\) is the intrinsic vertical scatter of the model, whereas \(\sigma_{\perp,i}\) is the vertical uncertainty of point \(i\)). The most likely fit to the data is obtained by maximising Equation (13), either through adjusting the parameters \((\phi, n, \sigma_\perp)\) or \((\alpha, \beta, \sigma_D)\). Following Appendix A, unbiased estimators for the population-model (as opposed to the sample-model) of the scatter \(\sigma_\perp\) or \(\sigma_\perp\) and its variance \(\sigma_{\perp,i}^2\) or \(\sigma_{\perp,i}^2\) can be obtained via Equations (A2) and (A1), respectively.

It is worth stressing that the exact Equation (13) never reduces to the frequently used, simple minimisation of \(\chi^2 \equiv \sum_i (y_i - \alpha x_i - \beta)^2\), not even if the data has zero uncertainties and if the intrinsic scatter is fixed. This reflects the fact that the \(\chi^2\)-minimisation breaks the rotational invariance by assuming a uniform prior for the slope \(\alpha\) (rather than the
angle $\phi$). The term $\ln(\alpha^2 + 1)$ in Equation (13) is needed to recover the rotational invariance.

3 NUMERICAL IMPLEMENTATION

Having established the above mathematical framework for calculating the likelihood for a multi-dimensional fit, we developed a fully documented and tested R package called HYPER-FIT. This is available through a github repository and can be installed from within R with the following commands:

```r
> install.packages('magicaxis', dependencies=TRUE)
> install.packages('MASS', dependencies=TRUE)
> install.packages('rgl', dependencies=TRUE)
> install.packages('devtools')
> library(devtools)
> install_github('asgr/LaplacesDemon')
> install_github('asgr/hyper.fit')
```

To load the package into your R session, run

```r
> library(hyper.fit)
```

The core function that utilises the maths presented in the previous Section is `hyper.like`, which calculates the likelihood (Equation 6) for a given set of hyperplane parameters and a given dataset. This dataset can be error-free, or include covariant and heteroscedastic errors. To fit a hyperplane, the `hyper.like` function is accessed by `hyper.fit`, a utility fitting function that attempts to optimally fit a generative model to the data. The package includes a number of user-friendly summary and plotting outputs in order to visualise the fitted model and investigate its quality. The full 35 pages manual detailing the HYPER-FIT package can be viewed at hyperfit.icrar.org, and is included within the help files of the package itself (e.g. see `?hyper.fit` within R once the package is loaded.).

3.1 Hyper.like

The basic likelihood function `hyper.like` requires the user to specify a vector of model parameters `parm`, an $N \times D$ dimensional position matrix `X` (where $N$ is the number of data points and $D$ the dimensionality of the dataset, i.e. 10 rows by 2 columns in the case of 10 data points with $x$ and $y$ positions) and a $D \times D \times N$ array `covarray` containing the covariance error matrix for each point stacked along the last index. A simple example looks like

```r
> hyper.like(parm, X, covarray)
```

`parm` must be specified as the normal vector $n$ that points from the origin to the hyperplane concatenated with the intrinsic scatter $\sigma_i$ orthogonal to the hyperplane. E.g. if the user wants to calculate the likelihood of a plane in 3D with equation $z = 2x + 3y + 1$ and intrinsic scatter along the $z$-axis of 4, this becomes a normal vector with elements $[-0.143, -0.214, 0.071]$ with intrinsic scatter orthogonal to the hyperplane equal to 1.07. In this case, `parm=c(-0.143, -0.214, 0.071, 1.07)`, i.e. as expected, a 3D plane with intrinsic scatter is fully described by 4 ($= D + 1$) parameters. Because of the large number of coordinate systems that unambiguously define a hyperplane (and many of them might be useful in different circumstances), the HYPER-FIT package also includes the function `hyper.convert` that allows the user to convert between coordinate systems simply. In astronomy, the Euclidean $z = \alpha_1x + \alpha_2y + \beta$ system is probably the most common (e.g. Fundamental Plane definition, Faber et al. 1987; Binney & Merrifield 1998), but this has limitations when it comes to efficient high dimensional hyperplane fitting, as we discuss later.

3.2 Hyper.fit

In practice, we expect most user interaction with the HYPER-FIT package will be via the higher level `hyper.fit` function (hence the name of the package). For example, to find an optimal fit for the data in the $N \times D$ dimensional matrix `X` (see Section 3.1) with no specified error, it suffices to run

```r
> fit=hyper.fit(X)
```

This function interacts with the `hyper.like` function and attempts to find the best generative hyperplane model via a number of schemes. The user has access to three main high-level fitting routines: the base R `optim` function (available with all installations), the `LaplaceApproximation` function, and the `LaplacesDemon` function (both available in the LAPLACESDEMON package).

3.2.1 Optim

`optim` is the base R parameter optimisation routine. It allows the user to generically find the maximum or minimum of a target function via a number of built-in schemes that are popular in the statistical optimisation community. The default option is to use a Nelder–Mead (Nelder & Mead 1965) optimiser that uses only function values. This is robust but relatively slow, with the advantage that it will work reasonably well for non-differentiable functions, i.e. it does not compute the local hessian at every step. Conjugate gradient (Hestenes & Stiefel 1952); quasi-Newtonian BFGS
(Broyden, Fletcher, Goldfarb, and Shanno; Fletcher 1970); and simulated annealing (Belisle 1992) are some of the other popular methods available to the user. All of these schemes are fully documented within the optim manual pages that is included with any standard R implementation.

3.2.2 LaplaceApproximation
To offer the user more flexibility, we also allow access to the non-base open-source MIT-licensed LAPLACESDEMON package that users can install from github.com/asgr/LaplacesDemon (the direct R installation code is provided above). It is developed by Statisticat and currently contains 18 different optimisation schemes. Many of these are shared with optim (e.g. the default option of Nelder–Mead; conjugate gradient descent), but many are unique to the package (e.g. Levenberg–Marquardt, Marquardt 1963; particle swarm optimisation, Kennedy & Eberhart 1995).

The LaplaceApproximation function offers a host of high level user tools, and with these object oriented extensions, it tends to be slower to converge compared to the same method in optim. The large range of methods means the user should be able to find a robustly converged solution in most situations, without having to resort to much more computationally expensive Markov-Chain Monte-Carlo (MCMC) methods accessed through the LaplacesDemon function which is also available in the LAPLACESDEMON package.

The LaplaceApproximation function and the wider LaplacesDemon package is extensively documented, both in the bundled manual and at the www.bayesian-inference.com website maintained by the package developers.

3.2.3 LaplacesDemon
The final hyperplane fitting function that users of HYPER-FIT have access to is LaplacesDemon which is also included in the LAPLACESDEMON package. The function includes a suite of 41 MCMC routines, and the number is still growing. For HYPER-FIT, the default routine is Griddy–Gibbs (Ritter & Tanner 1992), a component-wise algorithm that approximates the conditional distribution by evaluating the likelihood model at a set of points on a grid. This proved to have attractive qualities and well-behaved systematics for relatively few effective samples during testing on toy models. The user also has access to a large variety of routine families, including Gibbs; Metropolis–Hasting (Hastings 1970); slice sampling (Neal 2003); and Hamiltonian Monte Carlo (Duane et al. 1987).

In practice, the user should investigate the range of options available, and the optimal routine for a given problem might not simply be the default provided. To aid the user in choosing an appropriate scheme, the LAPLACESDEMON package includes a comprehensive suite of analysis tools to analyse the post convergence Markov-Chains. The potential advantages to use an MCMC approach to analyse the generative likelihood model are manifold, allowing the user of HYPER-FIT to investigate complex high-dimensional multi-modal problems that might not be well suited to the traditional downhill gradient approaches included in optim and LaplaceApproximation. Again, the function is extensively described both in the LAPLACESDEMON package and on the www.bayesian-inference.com website.

3.3 Plot
The HYPER-FIT package comes with a class sensitive plotting function where the user is able to execute a command such as

```r
> plot(hyper.fit(X))
```

where X is a 2 × N or 3 × N matrix. The plot displays the data with the best fit. If (uncorrelated or correlated) Gaussian errors are given, e.g. via plot(hyper.fit(X, covarray)), they are displayed as 2D ellipses or 3D ellipsoids, respectively. The built-in plotting only works in 2 or 3 dimensions (i.e. x,y or x,y,z data), which will likely be the two most common hyperplane fitting situations encountered in astronomical applications.

3.4 Summary
The HYPER-FIT package comes with a class sensitive summary function where the user is able to execute a command such as

```r
> summary(hyper.fit(X))
```

where X can be any (D × N)-array with D ≥ 2 dimensions. Using summary for 2D data will produce an output similar to

```r
> summary(fitnoerror)
Call used was:
hyper.fit(X = cbind(xval, yval))
```

Data supplied was 5rows × 2cols.
Requested parameters:
alpha1 beta.vert scat.vert
0.4680861 0.6272718 0.2656171

Errors for parameters:
err_alpha1 err_beta.vert err_scat.vert
0.12634307 0.11998805 0.08497509

---

The full parameter covariance matrix:

\[
\begin{pmatrix}
\alpha_1 & \beta \cdot \text{vert} & \text{scat} \cdot \text{vert} \\
\alpha_1 & 0.015962572 & -0.002139042 \cdot 0.0016270601 \\
\beta \cdot \text{vert} & -0.002139042 \cdot 0.0143971319 & -0.0002182796 \\
\text{scat} \cdot \text{vert} & 0.001627060 & 0.0072207660
\end{pmatrix}
\]

The sum of the log-likelihood for the best fit parameters:

LL = 4.623924

Unbiased population estimator for the intrinsic scatter:

\[
\text{scat} \cdot \text{vert} = 0.3721954
\]

Standardised parameters vertically projected along dimension 2 (yval):

\[
\begin{pmatrix}
\alpha_1 & \beta \cdot \text{vert} & \text{scat} \cdot \text{vert} \\
0.4680861 & 0.6272718 & 0.2656171
\end{pmatrix}
\]

Standardised generative hyperplane equation with unbiased population estimator for the intrinsic scatter, vertically projected along dimension 2 (yval):

\[
y \cdot \text{val} \sim \mathcal{N}(\mu = 0.4681 \cdot x \cdot \text{val} + 0.6273, \sigma = 0.3722)
\]

3.5 Web interface

To aid the accessibility of the tools, we have developed in this paper, a high-level web interface to the HYPER-FIT package has been developed that runs on a permanent virtual server using R Shiny. The HYPER-FIT computations are executed remotely, and the web interface itself is in standard JavaScript and accessible using any modern popular web browser. This website interface to HYPER-FIT can be accessed at hyperfit.icrar.org. The HYPER-FIT website also hosts the latest version of the manual, and the most recent version of the associated HYPER-FIT paper. The design philosophy for the page is heavily influenced by modern web design, where the emphasis is on clean aesthetics, simplicity, and intuitiveness. The front page view of the web interface is shown in Figure 2.

The HYPER-FIT web tool allows access to nearly all of the functionality of HYPER-FIT, with a few key restrictions:

- Only 2D or 3D analysis available (R HYPER-FIT allows for arbitrary ND analysis)
- Uploaded files are limited to 2,000 or fewer row entries (R HYPER-FIT allows for hard-disk limited size datasets)
- When using the LaplaceApproximation or LaplacesDemon algorithms only 20,000 iterations are allowed (R HYPER-FIT has no specific limit on the number of iterations)
- A number of the more complex LaplacesDemon methods are unavailable to the user because the specification options are too complex to describe via a web interface (R HYPER-FIT allows access to all LaplacesDemon methods)

In practice, for the vast majority of typical use cases, these restrictions will not limit the utility of the analysis offered by the web tool. For the extreme combination of 2,000 rows of data and 20,000 iterations of one of the LaplacesDemon MCMC samplers, users might need to wait a couple of minutes for results.

4 MOCK EXAMPLES

In this Section, we give some examples using two sets of idealised mock data.

4.1 Simple examples

To help novice users get familiar with R and the HYPER-FIT package, we start this Section by building the basic example shown in Figure 1. We first create the central x and y positions of the five data points, encoded in the five-element vectors x_val and y_val,

\[
x \cdot \text{val} = c(-1.22, -0.78, 0.44, 1.01, 1.22)
\]
Figure 3. 2D fit with no errors. Figure shows the default plot output of the hyper.plot2d function (accessed via the class specific plot method) included as part of the R HYPER-FIT package, where the best generative model for the data is shown as a solid line with the intrinsic scatter indicated by dashed lines. The colouring of the points shows the ‘tension’ with respect to the best-fit linear model and the measurement errors (zero in this case), where redder colours indicate data that is less likely to be explainable.

\[
> \text{yval = c(-0.15, 0.49, 1.17, 0.72, 1.22)}
\]

We can obtain the HYPER-FIT solution and plot the result (seen in Figure 3) with

\[
> \text{fitnoerror=hyper.fit(cbind(xval, yval))}
> \text{plot(fitnoerror) (Figure 3)}
\]

Since this example has no errors associated with the data positions, the generative model created has a large intrinsic scatter that broadly encompasses the data points (dashed lines in Figure 3). Extending the analysis, we can add x-errors x_err (standard deviations of the errors along the x-dimension) and y-errors y_err and recalculate the fit and plot (Figure 4) with

\[
> \text{xerr = c(0.12, 0.14, 0.20, 0.07, 0.06)}
> \text{yerr = c(0.13, 0.03, 0.07, 0.11, 0.08)}
> \text{fitwitherror=hyper.fit(cbind(xval, yval), vars=cbind(xerr, yerr)^2)}
> \text{plot(fitwitherror) (Figure 4)}
\]

Since we have added errors to the data, the intrinsic scatter required in the generative model is reduced. To extend this 2D example to the full complexity of Figure 1, we assume that we know xy_cor (the correlation between the errors for each individual data point) and recalculate the fit and plot (Figure 5) with

\[
> \text{xcor = c(0.90, -0.40, -0.25, 0.00, -0.20)}
> \text{fitwitherrorandcor=hyper.fit(cbind(xval, yval), covarray=makecovarray2d(xerr, yerr, xcor))}
> \text{plot(fitwitherrorandcor) (Figure 5)}
\]

The effect of including the correlations can be quite subtle, but inspecting the intersections on the y-axis, it is clear that both the best-fitting line and the intrinsic scatter have slightly changed. To see how dramatic the effect of large errors can be on the estimated intrinsic scatter, we can simply inflate our errors by a large factor. We recalculate the fit and plot (Figure 6) with
Per-data-point error covariance is often overlooked during regression analysis because of the lack of readily available tools to correctly make use of the information, but the impact of correctly using them can be non-negligible. As a final simple example, we take the input for Figure 6 and simply rotate the covariance matrix by 90°. We recalculate the fit and plot (Figure 7) with

```r
> fitwithbigerrorandrotcor=hyper.fit
  (cbind(xval, yval),
   covarray=makecovarray2d(yerr*1.9, xerr*1.9, -xycor))
> plot(fitwithbigerrorandrotcor)  (Figure 7)
```

The impact of simply rotating the covariance matrix is to flatten the preferred slope and to slightly reduce the intrinsic scatter.

### 4.2 Hogg example

In the arXiv paper by Hogg et al. (2010), the general approach to the problem of generative fitting is explored, along with exercises for the user. The majority of the paper is concerned with 2D examples, and an idealised data set (Table 1 in the paper) is included to allow readers to recreate their fits and plots. This dataset comes bundled with the HYPER-FIT package, making it easy to recreate many of the examples of Hogg et al. (2010). Towards the end of their paper, they include an exercise on fitting 2D data with covariant errors. We can generate the HYPER-FIT solution to this exercise and create the relevant plots (Figure 8) with

```r
> fitwithbigerrorandcor=hyper.fit
  (cbind(xval, yval),
   covarray=makecovarray2d(xerr*1.9, yerr*1.9, xycor))
> plot(fitwithbigerrorandcor)  (Figure 6)
```
The fit we generate agrees very closely with the example in Hogg et al. (2010), which is not surprising since the likelihood function we maximise is identical, bar a normalising factor (i.e. an additive constant in \( \ln L \)). The differences seen between Hogg’s Figure 14 and the bottom panel of our Figure 8, showing the projected posterior chain density distribution of the intrinsic scatter parameter, are likely to be due to our specific choice of MCMC solver. In our hyper-fit example, we use Componentwise Hit-And-Run Metropolis (CHARM, Turchin 1971), being a particularly simple MCMC for a novice user to specify, and one that performs well as long as the number of data points is greater than \( \sim 10 \).

5 ASTROPHYSICAL EXAMPLES

In this Section, we have endeavoured to include a mixture of real examples from published astronomy papers. In some cases, aspects of the data are strictly proprietary (in the sense that the tables used to generate paper Figures have not been explicitly published), but the respective authors have allowed us to include the required data in our package.

5.1 2D Mass-size relation

The galaxy mass-size relation is a topic of great popularity in the current astronomical literature (Shen et al. 2003; Trujillo et al. 2004; Lange et al. 2015). The hyper-fit package includes data taken from the bottom-right panel of Figure 3 from Lange et al. (2015). To do our fit on these data and create Figure 9, we make use of the published data errors and the \( 1/\sigma^2 \) weights calculated for each galaxy by running

```r
> data(GAMAsmVsize)
> plot(hyper.fit(GAMAsmVsize[,c('logmstar', 'logrekpc')],
vars=GAMAsmVsize[,c('logmstar_err', 'logrekpc_err')]^2,
weights=GAMAsmVsize[, 'weights'])
```

This analysis produces a mass-size relation of

\[
\log_{10} R_e \sim \mathcal{N}[\mu = (0.382 \pm 0.006) \log_{10} M_* - (3.53 \pm 0.06), \sigma = 0.145 \pm 0.003],
\]

where \( R_e \) is the effective radius of the galaxy in kpc, \( \mathcal{N} \) is the normal distribution with parameters \( \mu \) (mean) and \( \sigma \) (standard deviation), and \( M_* \) is the stellar mass in solar mass units. The relation published in Lange et al. (2015) is \( \log_{10} R_e = 0.46 \pm 0.02(\log_{10} M_*) - 4.38 \pm 0.03 \). The function in Lange et al. (2015) is noticeably steeper due to being fit to the running median with weighting determined by the spread in the data and the number of data points in the stellar mass bin, rather than directly to the data including the error.

5.2 2D Tully–Fisher relation

One of the most common relationships in extragalactic astronomy is the so-called Tully–Fisher relation (TFR, Tully & Fisher 1977). This tight coupling between inclination corrected disc rotation velocity and the absolute magnitude (or, more fundamentally, stellar mass) of spiral galaxies is an important redshift-independent distance indicator, as well as a probe for galaxy evolution studies. Here, we use TFR data included in the hyper-fit package that has been taken from Figure 4 of Obreschkow & Meyer (2013), where we fit the data and create Figure 10 with

```r
> data(TFR)
> plot(hyper.fit(TFR[,c('velocity', 'absolute_k_mag')],
vars=TFR[,c('velocity_err', 'absolute_k_mag_err')]^2,
weights=TFR[, 'weights'])
```

The function in Obreschkow & Meyer (2013) is \( \log_{10} M_* = 1.15 \pm 0.01\log_{10} V_c + 11.9 \pm 0.1 \). The function in Obreschkow & Meyer (2013) is noticeably steeper due to the spread in the data and the number of data points in the stellar mass bin, rather than directly to the data including the error.

---

References:

This yields a TFR of

\[ M_K \sim \mathcal{N}[\mu = (-9.3 \pm 0.4) \log_{10} v_{\text{circ}} - (2.5 \pm 0.9), \sigma = 0.22 \pm 0.04], \]

where \( M_K \) is the absolute \( K \)-band magnitude and \( v_{\text{circ}} \) is the maximum rotational velocity of the disc in \( \text{km s}^{-1} \).

Obreschkow & Meyer (2013) find

\[ M_K \sim \mathcal{N}[\mu = (-9.3 \pm 0.3) \log_{10} v_{\text{circ}} - (2.5 \pm 0.7), \sigma = 0.22 \pm 0.06], \]

where all the fit parameters comfortably agree within quoted errors. The parameter errors in Obreschkow & Meyer (2013) were recovered via bootstrap resampling of the data points, whereas HYPER-FIT used the covariant matrix via the inverse of the parameter Hessian matrix. Using HYPER-FIT, we find marginally less constraint on the slope and intercept, but slightly more on the intrinsic scatter.

### 5.3 3D fundamental plane

Moving to a 3D example, perhaps the most common application in the literature is the Fundamental Plane for elliptical galaxies (Faber 1987; Binney & Merrifield 1998). This also offers a route to a redshift-independent distance estimator, but from a 3D relation rather than a 2D relation (although an approximate 2D version also exists for elliptical galaxies via the Faber–Jackson relation; Faber & Jackson 1976). In HYPER-FIT, we include 6dFGS data released in Campbell et al. (2014), where we fit the data and generate Figure 11 with

\[ \log_{10} R_{e_j} \sim \mathcal{N}[\mu = -(0.853 \pm 0.005) \log_{10} I_{e_j} + (1.508 \pm 0.012) \log_{10} \sigma_{\text{vel}} - (0.42 \pm 0.03), \sigma = 0.060 \pm 0.001], \]

where \( R_{e_j} \) (in kpc) is the effect radius in the \( J \)-band, \( I_{e_j} \) (in mag arcsec\(^{-2}\)) is the average surface brightness intensity within \( R_{e_j} \) and \( \sigma_{\text{vel}} \) (in \( \text{km s}^{-1} \)) is the central velocity dispersion. Magoulas et al. (2012) find

\[ \log_{10} R_{e_j} \sim \mathcal{N}[\mu = -(0.89 \pm 0.01) \log_{10} I_{e_j} + (1.52 \pm 0.03) \log_{10} \sigma_{\text{vel}} - (0.19 \pm 0.01), \sigma = 0.12 \pm 0.01]. \]

It is notable that our fit requires substantially less intrinsic scatter than Magoulas et al. (2012). It is difficult to assess the origin of this difference since the methods used to fit the plane differ between this work and Magoulas et al. (2012), where the latter fit a 3D generative Gaussian with 8° of parameter dispersion.
freedom, which is substantially more general than the 3D plane with only 4° of parameter freedom that HYPER-FIT used.

5.4 3D mass-spin-morphology relation

The final example of HYPER-FIT uses astronomical data from Obreschkow & Glazebrook (2014). They measured the fundamental relationship between mass (here the baryon mass (stars+cold gas) of the disc), spin (here the specific baryon angular momentum of the disc), and morphology (as measured by the bulge-to-total mass ratio). Hereafter, this relation is referred to as the MJB relation. This dataset is interesting to analyse with HYPER-FIT because it includes error correlation between mass and spin, caused by their common dependence on distance. In HYPER-FIT, we included the MJB data presented in Obreschkow & Glazebrook (2014). We can fit these data and generate Figure 12 with

```r
> data(MJB)
> plot(hyper.fit(X=MJB[,c('logM', 'logj', 'B.T')],
      covarray=makecovarray3d(MJB$logM_err, MJB$logj_err, MJB$B.T_err, MJB$corMJ, 0, 0)))
```

(Figure 12)

This leads to a MJB relation of

\[
\frac{B}{T} = (0.34 \pm 0.03) \log_{10} M - (0.35 \pm 0.04) \log_{10} j - (0.04 \pm 0.02),
\]

where \(\frac{B}{T}\) is the galaxy bulge-to-total ratio, \(M\) is the disc baryon mass in \(10^{10}\) solar masses, and \(j\) is the specific baryon angular momentum in \(10^{-3}\) kpc km s\(^{-1}\) (see Obreschkow & Glazebrook 2014, for details on measuring this quantity). Obreschkow & Glazebrook (2014) find

\[
\frac{B}{T} \sim \mathcal{N}[\mu = (0.34 \pm 0.03) \log_{10} M - (0.35 \pm 0.04) \log_{10} j - (0.04 \pm 0.02),
\sigma = 0].
\]

The agreement between HYPER-FIT and Obreschkow & Glazebrook (2014) is excellent, with all parameters consistent within quoted 1σ errors. Our results are consistent with zero intrinsic scatter within parameter errors. In fact, running the analysis using MCMC and the CHARM algorithm, we used HYPER-FIT to directly sample the likelihood of the intrinsic scatter being exactly 0. This is possible because we use the wall condition that any intrinsic scatter sample below 0 (which is non-physical) is assigned to be exactly 0. We find that 90.4% of the posterior chain samples have intrinsic scatter of exactly 0 post application of the wall condition (which imposes a spike + slab posterior), which can be interpreted to mean that the data is consistent with \(P(\sigma = 0) = 0.904\).

This agrees with the assessment in Obreschkow & Glazebrook (2014), where they conclude that the 3D plane generated does not require any additional scatter to explain the observed data, probably indicating that the errors have been overestimated6.

---

6If the real generative model is a hyperplane with no scatter then we expect to observe \(P(\sigma = 0) = U(0, 1)\), where \(U\) is the uniform distribution. \(P(\sigma = 0)\) can be cast as the frequentist p-value, where the null hypothesis is a generative model with no intrinsic scatter. The posterior we form via the addition of the wall condition is formally not a Bayesian posterior. A fully rigorous approach would be to compare Bayes factors for a model with \(\sigma = 0\) and another model with \(\sigma > 0\). Using the wall condition, posterior as described should alert the user to situations where such a model comparison is potentially necessary.
6 CONCLUSION
In this paper, we have extended previous work (Kelly 2007; Hogg et al. 2010; Kelly 2011) to describe the general form of the $D$ dimensional generative hyperplane model with intrinsic scatter. We have also provided the fully-documented hyper-fit package for the R statistical programming language, available immediately for installation via github. A user friendly Shiny web interface has also been developed and can be accessed at hyperfit.icrar.org. The hyper-fit version released in conduction with this paper is v1.0.2, and regular updates and support are planned.

To make the package as user-friendly as possible, we have provided examples for simple 2D datasets, as well as complex 3D datasets utilising heteroscedastic covariant errors taken from Obreschkow & Glazebrook (2014). We compared the published fits to the new hyper-fit estimates, and in most cases find good agreement for the hyperplane parameters (not all original papers attempt to make an intrinsic scatter estimate). hyper-fit offers users access to higher level statistics (e.g. the parameter covariance matrix) and includes simple class sensitive summary and plot functions to assist in analysing the quality of the generative model.

The ambition is for hyper-fit to be regularly updated based on feedback and suggestions from the astronomy and statistics community. As such, in the long term, the documentation included as part of the package might supersede some details provided in this paper.

7 github.com/asgr/hyper.fit

ACKNOWLEDGEMENTS
Thank you to Joseph Dunne who developed the web tool interface for hyper-fit which we now host at hyperfit.icrar.org. This was done as part of an extended summer student programme and required remarkably little direct supervision. Thank you to Ewan Cameron who assisted in describing the correct JAGS 2D version of the hyper-fit code presented in Appendix C. Thank you finally to the members of the Astrostatistics FaceBook group who contributed helpful comments at early stages of this work, and gave useful feedback on the R hyper-fit package prior to publication.

REFERENCES
Duane, S., Kennedy, A. D., Pendleton, B. J., & Roweth, D. 1987, PhilB, 195, 216
Faber, S. M., Dressler, A. D., Roger, L., Burstein, D., & Lynden-Bell, D. 1987, in Nearby Normal Galaxies: From the Planck Time to the Present, ed. S. M. Faber (New York: Springer), 175
Fletcher, R. 1970, CompJ, 13, 317
Hastings, W. K. 1970, Biometrika, 57, 97
Hestenes, M. R., & Stiefel, E. 1952, JRNBS, 49, 409
Kelly, B. C. 2011, in Statistical Challenges in Modern Astronomy V, ed. E. Feigelson, & J. Babu (New York: Springer), 147
Marquardt, D. 1963, Siam, 11, 431
Neal, R. 2003, AnSta, 31, 705
Nelder, J. A., & Mead, R. 1965, CompJ, 7, 308
Ritter, C., & Tanner, M. 1992, JASA, 87, 861
von Toussaint, U. 2015, Entropy, 17, 3898

7 github.com/asgr/hyper.fit

8 www.facebook.com/groups/astro.r
A INTRINSIC SCATTER ESTIMATOR BIAS

A subtle point is that the expectation of the most likely model is not generally identical to the true model. Let us assume a fixed generative model described by Equation (3) with known parameters \( \mathbf{n}^{\text{true}} \) and \( \sigma_\perp^{\text{true}} \). From this model, we randomly draw a sample of \( N \) points and estimate the most likely values, i.e. the modes, \( \mathbf{n}^{\text{ML}} \) and \( \sigma_\perp^{\text{ML}} \) by maximising Equation (5). This procedure is repeated ad infinitum, always with the same model and the same number of points, resulting in infinitely many different estimates \( \mathbf{n}^{\text{ML}} \) and \( \sigma_\perp^{\text{ML}} \). By definition, the expectations of these estimators are the ensemble-averages \( \langle \mathbf{n}^{\text{ML}} \rangle \) and \( \langle \sigma_\perp^{\text{ML}} \rangle \). The question is whether these expectations are identical to the input parameters \( \mathbf{n}^{\text{true}} \) and \( \sigma_\perp^{\text{true}} \). If so, the estimators are called unbiased, if not, they are biased. The mirror-symmetry of our generative model implies that \( \mathbf{n}^{\text{ML}} \) is an unbiased estimator\(^9\), i.e. \( \langle \mathbf{n}^{\text{ML}} \rangle = \mathbf{n}^{\text{true}} \).

By contrast, the intrinsic scatter is generally biased, i.e. \( \langle \sigma_\perp^{\text{ML}} \rangle \neq \sigma_\perp^{\text{true}} \). There are two reasons for this bias. Firstly, maximising the likelihood function \( L \) finds the most likely generative model of a sample of \( N \) data points, which is not generally the most likely model of the population, i.e. the imaginary infinite set, from which the \( N \) points were drawn at random. For example, consider a two-dimensional population described by a straight line with some intrinsic scatter. From this population, we draw a sample of two points. No matter which two points we choose, they can always be fitted by a straight line without scatter. Hence, the most likely sample model, maximising Equation (5), will have zero intrinsic scatter, even if the population model has non-zero scatter. The transition from the most likely model of the sample to the most likely model of the population is achieved by the so-called Bessel correction,

\[
\tilde{\sigma}_\perp^2 = \frac{N}{N - D} \sigma_\perp^2,
\]

where \( D \) is the number of parameters of the hyperplane (also referred to as the ‘degrees of parameter freedom’ or simply ‘degrees of freedom’), which equals the number of dimensions. In the special case of Gaussian data with no errors \( (C_i \equiv 0) \), the variance estimator \( \tilde{\sigma}_\perp^2 \) is unbiased, i.e. \( \langle (\tilde{\sigma}_\perp^{\text{ML}})^2 \rangle = \langle \sigma_\perp^{\text{true}} \rangle^2 \). In the general case, one can estimate the population variance by computing the mean of its marginal posterior PDF. This is often achieved using a Markov Chain Monte Carlo (MCMC) approach, but note that for the presence of heteroscedastic errors, but they should offer the minimum appropriate correction to apply. Bootstrap resampling offers a window to properly correct for such data biases, but this can be extremely expensive to compute.

B EXTENSION TO NON-UNIFORM SELECTION FUNCTIONS

We now consider the more general situation, where the population is not sampled uniformly, but with a probability proportional to \( \rho_i(x) \), known as the selection function in astronomy. In this case, the effective PDF \( \rho_\ast(x) \) of the generative model and the selection function is the product \( \rho_\ast(x) \rho_i(x) \), renormalised along the direction perpendicular to \( \mathbf{h} \), \( \int_{\mathbb{R}^D} d\mathbf{l} \rho_\ast(\mathbf{l}) = 1 \). The renormalisation is crucial to avoid a dependence of the normalisation of \( \rho_\ast(x) \) on the fitted-model parameters. The effective PDF reads

\[
\rho_\ast(x) = \frac{\rho_i(x) \rho(x)}{\int_{\mathbb{R}^D} d\mathbf{l} \rho_\ast(\mathbf{l}) \rho_i(\mathbf{n})}.
\]

The likelihood of point \( i \) then becomes

\[
\mathcal{L}_i = \int_{\mathbb{R}^D} dx \rho(x | \mathbf{n}) \rho_i(x).
\]
Analytical closed-form solutions of Equation (44) can be found for some functions $\rho(x)$. To illustrate this, let us consider the case of an exponential selection function

$$
\rho_s(x) = e^{k^\top x}, \tag{B3}
$$

where $k$ is the known (and fixed) sampling gradient, i.e. in the direction of $k$ the probability of drawing a point from the population increases exponentially. This selection function is important in many fields of science, since it represents a power-law sampling.

Incidentally, in analogy to Equation (4), the likelihood terms then become

$$
L_i = \int d\theta \rho(x|x) \rho_w(x) = \frac{1}{\sqrt{2\pi \sigma_i^2}} e^{-\frac{\theta^2}{2\sigma_i^2}}, \tag{B6}
$$

where $\sigma_i^2 = \sigma_x^2 + \hat{n}^\top C_i \hat{n}$ and $\xi_i = x_i - \sigma_x^2 \hat{n}$. Up to an additive constant, the total likelihood function $\ln L = \sum_{i=1}^N \ln L_i$ reads

$$
\ln L = -\frac{1}{2} \sum_{i=1}^N \left[ \ln(\sigma_{i,j}^2) + \frac{(\hat{n}^\top \xi_i - \hat{n})^2}{\sigma_{i,j}^2} \right]. \tag{B7}
$$

This likelihood is identical to that of the uniform case (Equation 5), as long as we substitute the positions $x_i$ for $\xi_i$. Upon performing this substitution, all results of Sections 2.1 to 2.3 remain valid in the case of a power-law sampling. Incidentally, it is easy to see that $\hat{n} = x_i$ if the data are sampled uniformly ($k = 0$).

The HYPER-FIT package allows for the addition of such a power-law selection function via the $k.vec$ vector input argument in the `hyper.like` and `hyper.fit` functions.

C HIERARCHICAL IMPLEMENTATION OF 2D HYPER-FIT USING JAGS

Over recent years, a number of hierarchical Bayesian inference software programs have become popular, e.g. Bayesian inference Using Gibbs Sampling (BUGS), Just Another Gibbs Sampler (JAGS), and STAN. They are all somewhat similar in terms of the modelling language, with an emphasis on constructing complex hierarchical models with relative ease (Gelman & Hill 2006). An example 2D implementation of the generative HYPER-FIT model can be constructed in JAGS using the following model specification:

10www.mrc-bsu.cam.ac.uk/software/bugs/
11mcmc-jags.sourceforge.net/
12mc-stan.org

PASA, 32, e033 (2015)
doi:10.1017/pasa.2015.33

model {
  for (i in 1:N) {
    xyobs[1:2,i] ~ dmnorm(xytrue[1:2,i], errors[1:2,1:2,i])
  }
  for (i in 1:N) {
    errors[1,1,i] <- 1/xsd[i]^2/(1-rho[i]^2)
    errors[2,2,i] <- 1/ysd[i]^2/(1-rho[i]^2)
    errors[1,2,i] <- rho[i]/xsd[i]/ysd[i]/(1-rho[i]^2)
    errors[2,1,i] <- rho[i]/xsd[i]/ysd[i]/(1-rho[i]^2)
  }
  for (i in 1:N) {
    xytrue[1,1] <- distfromorigin[i]*cos(angle*0.01745) - sin(angle*0.01745)*scat[i]
    xytrue[2,2] <- distfromorigin[i]*sin(angle*0.01745) + cos(angle*0.01745)*scat[i] + originy
  }
  angle ~ dunif(-90,90)
  originy ~ dunif(-1000,1000)
  intrinvar ~ dunif(0,10000)
  intrinvar ~ dunif(0,10000)
}

In the above 2D model, the observed $x$-$y$ data is contained in the $2 \times N$ matrix $xytrue$, the $x$ errors are in the $N$ vector $xsd$, the $y$ errors are in the $N$ vector $ysd$, and the error correlations between $x$ and $y$ are in the $N$ vector $rho$. The rotation angle of the generative model, the offset of the model from the origin, and the intrinsic variance of the model are all sampled from uniform prior distributions in this example. The model specification using STAN or other BUGS style software programs is very similar in construct to the above.

Compared to the HYPER-FIT software, the JAGS modelling solution converges slowly (factors of a few slower for equally well-sampled posteriors) and tends to suffer from poor mixing between highly correlated parameters. This is a common problem with Gibbs sampling, and in practice the typical user can gain confidence in their fit solution using HYPER-FIT by attempting a number of the available routines, where e.g. Slice Sampling is an empirical approximation to true Gibbs sampling. Fully hierarchical based solutions (e.g. BUGS, JAGS, and STAN) are also hard to extend into arbitrary dimensions, and in general need to be carefully hand-modified for bespoke datasets. However, these programs offer clear advantages in elegantly handling left or right censored data, with very little modification to the above specified model required.