Supplementary Online Material Part 1

A Bayesian Approach for Summarizing and Modeling Time-Series Exposure Data with Left Censoring

E. Andres Houseman[1] and M. Abbas Virji[2]

[1]Oregon State University, College of Public Health and Human Sciences, Corvallis, OR.
[2]National Institute for Occupational Safety and Health, Respiratory Health Division, Morgantown, WV.

Supplementary Methods

Bayesian Spline Model

We assume that n time-series $\mathbf{Y}_i, i \in \{1,...,n\}$, have been collected, each series $\mathbf{Y}_i = (Y_{i1}, Y_{i2}, \cdots, Y_{iT_i})$ of varying length T_i and observed at a sequence of T_i times $(t_{i1}, t_{i2}, \cdots, t_{iT_i})$. Each measurement corresponds to one of m designated occupational tasks $w_{ir} \in \{1,2,\cdots,m\}$, which are thought to be associated with measurement Y_{ir}, where r indexes measurement sequence. In addition, each series may also correspond to a sequence of d –dimensional covariates $\mathbf{X}_i = (\mathbf{x}_{i1}, \mathbf{x}_{i2}, \cdots, \mathbf{x}_{iT_i})$, $\mathbf{x}_{ir} \in \mathrm{R}^d$, assumed also to influence Y_{ir} through a fixed effects regression model. We assume that $\mathbf{Y}_i$ follows a Gaussian process wherein each Y_{ir} is a normally distributed random variable (potentially the log-transform of a raw concentration) having a time-specific mean μ_{ir} that absorbs all autocorrelation inherent in the stochastic process. Specifically, we assume that the mean series $\mathrm{E}[\mathbf{Y}_i] = \boldsymbol{\mu}_i = (\mu_{i1}, \mu_{i2}, \cdots, \mu_{iT_i})$ is a time-sample from a smooth curve that can be represented with a *B-spline bas*is (Prautzsch et al 2002), a basis representation that is commonly used in non-parametric problems. Each mean parameter μ_{ir} is centered at a value $\tilde{\mu}_{ir}$, either a time-specific effect of task $\tilde{\mu}_{ir} = \alpha(w_{ir})$ [$\alpha(w)$ representing task-specific mean parameters for tasks $w \in \{1,2,\cdots,m\}$], or more generally, $\tilde{\mu}_{ir} = \alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta}$, which incorporates the additional effect of fixed covariates (with $\boldsymbol{\beta}$ a regression parameter fixed-effects). Note that $\tilde{\mu}_{ir}$ models the effect of task or other fixed characteristics, but does not involve terms that reflect within-series covariation; consequently, μ_{ir} incorporates an additional spline term representing a zero-mean stochastic process, $\mu_{ir} = \tilde{\mu}_{ir} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir})$, where $\mathbf{b}(t)$ is a k –dimensional vector of B-spline values dependent on time t and κ previously chosen *knots* (to be discussed below), and $\boldsymbol{\zeta}_i$ is a series-specific k – dimensional vector of *spline coefficients* that together with the spline basis $\mathbf{B}_i = [\mathbf{b}(t_{i1}), \cdots, \mathbf{b}(t_{iT_i})]^{\mathrm{T}}$ fully determine the sequence $\boldsymbol{\mu}_i$. In simpler terms, $\boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir})$ represents within-series covariation, $\mathbf{b}(t)$ represents the mathematical basis used to characterize the family of potential stochastic processes that govern possible within-series covariation structures, and $\boldsymbol{\zeta}_i$ represents the coefficients that ultimately determine the specific stochastic process that governs the within-series covariation. Furthermore, we model $\boldsymbol{\zeta}_i$ as a k –dimensional multivariate-normal random effect $\boldsymbol{\zeta}_i \sim MVN_k(\mathbf{0}_k, \sigma_\zeta^2 \mathbf{I}_k)$, where $\mathbf{0}_k$ is the k –dimensional zero vector, $\mathbf{I}_k$ is the $k \times k$ identity matrix and σ_ζ^2 is a variance component parameter. Finally, we assume an additional sequence $\mathbf{e}_i = (e_{i1}, e_{i2}, \cdots, e_{iT_i})$ of independent errors, each element e_{ir} of which has a task-specific variance $\mathrm{var}[e_{ir}] = \tilde{\sigma}_{ir}^2 = \sigma_e^2(w_{ir})$, so that $\mathrm{cov}[\mathbf{e}_i] = \boldsymbol{\Lambda}_i = \mathrm{diag}[\sigma_e^2(w_{i1}), \cdots, \sigma_e^2(w_{iT_i})]$, ultimately dependent upon m separate task-specific variance component parameters $\sigma_e^2(w)$. Thus $Y_{ir} = \tilde{\mu}_{ir} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}) + e_{ir}$ and $\mathrm{var}[Y_{ir} \mid \tilde{\mu}_{ir}] = \sigma_\zeta^2 \mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{ir}) + \sigma_e^2(w_{ir})$, which demonstrates that the variance is task-dependent and potentially non-constant across a series. As a consequence, $\mathrm{cov}[\mathbf{Y}_i \mid \tilde{\mu}_{i1}, \cdots, \tilde{\mu}_{iT_i}] = \sigma_\zeta^2 \mathbf{B}_i \mathbf{B}_i^{\mathrm{T}} + \boldsymbol{\Lambda}_i$ and, for two measurements corresponding to a single task *w* (say at times t_r and t_s), it follows that $\mathrm{cov}[Y_{ir}, Y_{is} \mid \tilde{\mu}_{ir}, \tilde{\mu}_{is}] = \sigma_\zeta^2 \mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{is})$ and $corr[Y_{ir}, Y_{is} \mid \tilde{\mu}_{ir}, \tilde{\mu}_{is}] = \sigma_\zeta^2 \mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{is}) / \{\sigma_\zeta^2 \mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{is}) + \sigma_e^2(w)\}$. Note that this autocorrelation formula is similar to that for the intra-class correlation coefficient (ICC) in a hierarchical model setting, except for the presence of the $\mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{is})$ coefficient that results from the spline basis. If $\sigma_\zeta^2 << \sigma_e^2(w_{ir})$ for all tasks w_{ir} in the series, then $\mathrm{cov}[\mathbf{Y}_i \mid \tilde{\mu}_{i1}, \cdots, \tilde{\mu}_{iT_i}] \approx \boldsymbol{\Lambda}_i$ and all measurements are essentially uncorrelated, but if $\sigma_\zeta^2 >> \sigma_e^2(w)$ then two measurements corresponding to the same task *w* will be correlated to the maximum degree allowed by the spline basis, $\mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{is}) / \sqrt{\mathbf{b}(t_{ir})^{\mathrm{T}}\mathbf{b}(t_{ir})\mathbf{b}(t_{is})^{\mathrm{T}}\mathbf{b}(t_{is})}$. Thus, specific choices of the variance component parameters σ_{ζ}^{2} and $\sigma_{e}^{2}(w_{ir})$ allow for distinct levels of autocorrelation across tasks within a series, ranging from 0 to a maximum level allowed by the choice of basis functions, ultimately determined by the choice of knots. Figure S2 (in supplementary online material part 2) demonstrates the autocorrelation profiles for three different knot configurations and different relative magnitudes of σ_{ζ}^{2} and $\sigma_{e}^{2}(w)$.

Thus the fundamental model we propose for a given choice of splines and knots can be written succinctly as

$$Y_{ir} \sim N\{\alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_{i}^{\mathrm{T}}\mathbf{b}(t_{ir}), \sigma_{e}^{2}(w_{ir})\} \text{ and } \boldsymbol{\zeta}_{i} \sim MVN_{k}(\mathbf{0}_{k}, \sigma_{\zeta}^{2}\mathbf{I}_{k}), \qquad (1)$$

with free parameters $\boldsymbol{\beta}$, σ_{ζ}^{2}, $\alpha(w)$, and $\sigma_{e}^{2}(w)$, ($w \in \{1,2,\cdots,m\}$). Note that the variance parameter $\sigma_{e}^{2}(w)$ governs the task-specific variation of the independent errors (i.e., innovations in classical time-series parlance), while the parameter σ_{ζ}^{2} governs the between-series variation. While our proposed model is complicated to fit within a frequentist framework, it is relatively straightforward to fit this model in a Bayesian paradigm using standard software such as JAGS. JAGS compiles a model described using a standardized language that supports flexible Bayesian model specification, together with data informing the model and initial guesses at values representative of the posterior distribution, and returns a Markov chain representing values sampled from the posterior distribution. The fundamental principle used by JAGS is Gibbs Sampling, a well-known Markov-Chain-Monte-Carlo (MCMC) technique.

To complete the specification of the model in a Bayesian setting, it is necessary to specify prior distributions for the free parameters. We propose using commonly accepted prior distributions as follows: $\boldsymbol{\beta} \sim MVN(\mathbf{0}_d, \sigma^2_{0,\beta}\mathbf{I}_d)$, $\alpha(w) \sim N(0, \sigma^2_{0,\alpha})$, $\sigma^{-2}_{\zeta} \sim N_+(0, \lambda^2_{\zeta})$, and $\sigma^{-2}_e(w) \sim N_+(0, \lambda^2_e)$, where N_+ denotes the “half-normal” distribution, a normal distribution restricted to non-negative values, and $\sigma^2_{0,\beta}$, $\sigma^2_{0,\alpha}$, λ^2_{ζ} , and λ^2_e are hyper-parameters chosen in advance to produce relatively non-informative (“vague” or “flat”) distributions. Note that the priors of variance components are specified in terms of the inverse (i.e., the *precision*), and that the prior of each precision is specified as having a “half-normal” distribution instead of the common historic alternative, a gamma random variable; we propose this specification in order to circumvent a problem with the gamma prior wherein larger precision values (smaller variances) are given relatively high prior probability and thus corresponds to a somewhat informative prior distribution (Gelman 2006). The half-normal prior for variance components is becoming increasingly more common to avoid this problem.

When there are a large number n of profiles, it may be desirable to add an addition component of variation to account for “series” effects:

$$Y_{ir} \sim N\{a_i + \alpha(w_{ir}) + \mathbf{x}^{\mathrm{T}}_{ir}\boldsymbol{\beta} + \boldsymbol{\zeta}^{\mathrm{T}}_i \mathbf{b}(t_{ir}), \sigma^2_e(w_{ir})\}, \qquad (2)$$

where $a_i \sim N(0, \sigma^2_a)$ is a series-specific random intercept and the variance component σ^2_a has prior distribution specified in terms of its precision, $\sigma^{-2}_a \sim N_+(0, \lambda^2_a)$ with hyper-parameter λ^2_a. If multiple instruments were used within a study and each series used a single instrument, then a series-specific fixed instrument effect might also be incorporated:

$$Y_{ir} \sim N\{a_i + \alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}) + \mathbf{z}_i^{\mathrm{T}}\boldsymbol{\gamma}, \sigma_e^2(w_{ir})\}, \tag{3}$$

where $\mathbf{z}_i$ is a l-dimensional indicator vector (corresponding to $l+1$ instruments) encoding the instrument used for the specific series and $\boldsymbol{\gamma}$ is a corresponding l-dimensional fixed-effects regression parameter whose prior distribution is specified as a non-informative normal distribution in a manner similar to that described for $\boldsymbol{\beta}$. Additional series-specific fixed effects, such as production level effects that do not vary within a series, could be addressed by incorporating additional terms within $\mathbf{z}_i$. Note that $\mathbf{x}_{ir}$ and $\mathbf{z}_i$ are both fixed effects and could be combined into a single matrix, but we separate them here to emphasize the distinction between task-specific effects and potential exposure modifying factors or confounders. Finally, if many individuals have been sampled on more than one occasion, so that several series may correspond to a single subject, then an additional subject-specific random effect could be added in a manner similar to the manner in which the series-specific intercept a_i was added. In principle, complicated hierarchical relationships (e.g. subjects nested within positions nested within distinct employers) can flexibly be incorporated at this level.

In many exposure assessment settings it is common to observe values that are left-censored by LOD. Bayesian model fitting simplifies a faithful rendering of the left-censoring by seamlessly integrating over the left tail of the distributions given for Y_{ir} in models (1), (2), and (3). We introduce notation for a *truncated normal distribution* $N(\mu,\sigma^2)[a,b]$, a normal distribution truncated to the interval $[a,b]$ so that, for example, the half-normal distribution $N_+(\mu,\sigma^2)$ may be written alternatively as $N(\mu,\sigma^2)[0,+\infty]$. Thus, in our simplest model corresponding to (1), a left censored value observed for subject i at time t_{ir} would be specified as $Y_{ir} \sim N\{\alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}), \sigma_e^2(w_{ir})\}[-\infty, \delta]$, where δ represents the LOD. JAGS easily accommodates the specification of a truncated normal distribution.

We note that the models proposed above require a specification of knots to construct the spline basis matrices $\mathbf{B}_i = [\mathbf{b}(t_{i1}), \cdots, \mathbf{b}(t_{iT_i})]$, and that the choice of knots impacts the maximum autocorrelation that can be modeled between two given time points. B-splines are piecewise polynomial functions (usually cubic and twice-differentiable) that are determined recursively from a set of pre-specified knots, including two boundary knots defining the functional domain over which the spline basis will be applied. The set of knots can be thought of as a grid that represents an anticipated level of curvature internal to the functional domain; in general, a denser set with larger number of knots supports a larger magnitude of curvature, so that stochastic processes with extremely “wiggly” realizations require a denser grid of knots. Note, however, that a function with low-curvature can still be faithfully represented by a spline basis with a dense set of knots, so there is no effective penalty for a dense grid other than increased computation time. On the other hand, a dense grid of knots may lower the maximum autocorrelation possible to model for more widely separated times. For the applications we describe below, each of which entailed 3 or 5-minute averages sampled during shifts lasting multiple hours and occurring around the clock, and for which every series occurred on a single calendar day, we chose a basis with boundary knots at the minimum and maximum sampling hour (where hour ranges from 0 to 24) and internal knots placed at 7.5 minute intervals.

Finally, we remark on a detail of implementation that impacts the properties of the Markov chain returned by MCMC software such as JAGS. In order to obtain chains that have limited autocorrelation, it is necessary to use *hierarchical centering* in the model specification (Gelfand et al 1996). For example, model (3) is most naturally specified as

$$Y_{ir} \sim N\{a_i + \alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}) + \mathbf{z}_i^{\mathrm{T}}\boldsymbol{\gamma}, \sigma_e^2(w_{ir})\} \text{ and } a_i \sim N(0, \sigma_a^2),$$

together with the specification of the spline portion of the model $\boldsymbol{\zeta}_i \sim MVN_k(\mathbf{0}_k, \sigma_\zeta^2 \mathbf{I}_k)$.

However, better results are obtained with the following equivalent formulation:

$$Y_{ir} \sim N\{a_i + \alpha(w_{ir}) + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}), \sigma_e^2(w_{ir})\} \text{ and } a_i \sim N(\mathbf{z}_i^{\mathrm{T}}\boldsymbol{\gamma}, \sigma_a^2).$$

The principle of hierarchical centering may also apply to model (2), since the task-specific portion of the fixed-effects mean parameter $\tilde{\mu}_{ir}$ could be absorbed into the regression model $\mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta}$ by selecting a single task w_0 as reference and incorporating into $\mathbf{x}_{ir}$ the indicators for the non-reference tasks; in this formulation, model (2) is rewritten as

$$Y_{ir} \sim N\{a_i + \mathbf{x}_{ir}^{\mathrm{T}}\boldsymbol{\beta} + \boldsymbol{\zeta}_i^{\mathrm{T}}\mathbf{b}(t_{ir}), \sigma_e^2(w_{ir})\} \text{ and } a_i \sim N(\alpha(w_0), \sigma_a^2),$$

i.e., the random intercepts are centered at the mean of the reference task.

Data Source

We demonstrate our proposed analytical approach within the context of two exposure assessment scenarios. We have described the details of our primary data example in the main text. Here we provide the details for our smaller, secondary dataset: real-time exposure to nanoparticles collected during a walkthrough visit at an ultrafine titanium dioxide (TiO_2) and lithium titanate (Li_2TiO_3 or LTO) manufacturing facility. Process samples were collected for 35 hours over a two-day period using four area samplers: a Nanoparticle Surface Area Monitor

(NSAM) measuring alveolar-deposited surface area (size range: 0.01-1.0 μm) and three instruments measuring particle number concentration, a Scanning Mobility Particle Sizer (SMPS, size range: 0.01-0.5 μm), an Optical Particle Counter (OPC size range: 0.3 to >20 μm), and a Condensation Particle Counter (CPC, size range: 0.01-1.0 μm). Processes carried out on the sampling days were monitored and included the spray drying of a titanium tetrachloride solution, production of ultrafine lithium titanate, and spray drying a mixture of lithium hydroxide and uTiO2. Detailed information on tasks performed during sampling, which included receiving powder, preparing and scooping material, and shaking out baghouse, as well as information on ventilation use and the types of process enclosures was recorded in real-time over the sampling period. For data analysis, the averaging times of all the instruments was standardized to 3-minutes intervals and the total number concentration was used for OPC and SMPS, which is the sum of all size bins. In this example, there were no measurements below the LOD, and the approach illustrates the summarization of real-time exposure data accounting for autocorrelation and non-stationary data, as well as summary exposures for tasks accounting for fixed effects of covariates such as process enclosure.

Detailed Results for Secondary Data Example

The smaller, secondary example involves the analysis of four nanoparticle datasets described above. Summary characteristics of these data sets appear in Table 1 of the main text. For each of the four nanoparticle data sets, we fit two models of type (1) described above (i.e., with no series or instrument effects), one for each of two covariates: *source enclosure* [$d = 1$: *no* or *some of the time* (referent) | *yes*] and *enclosure type* [$d = 1$: *3-sided booth or sealing* (referent) | *none*]. Each model was fit using JAGS implemented in the R package *rjags* (version 3.4.0) run in R (version 3.2.2). The JAGS model used is described at the end of this document under *Code for Example 2*.

Table S4 (in supplementary online material part 2) provides posterior statistics for the NSAM model with the *source enclosure* covariate. Note that there was some variation in the posterior distributions of the task-specific standard deviation [$\sigma_e(w)$] parameters, and that these standard deviations were considerably smaller than the spline standard deviation σ_ζ, indicating substantial autocorrelation within a series. For each task, Figure S3 (in supplementary online material part 2) illustrates the autocorrelation profile across 30 minutes, based on the values $\sigma_e(w)$ and σ_ζ, demonstrating in more direct terms the autocorrelation implied by the posterior parameter distributions.

A complete set of results for all four nanoparticle measurements and both covariate models appears in Table S5 (in supplementary online material part 2). For each model, we have also included model fitting results from several frequentist methods (Table S6 in supplementary online material part 2), as well as plots that compare methods with respect to estimates, confidence intervals, and posterior statistics (Figure S4 in supplementary online material part 2). Naïve linear regression estimates (via OLS) were obtained using the R linear regression function `lm`; this method completely ignores autocorrelation within a series. Several LME models were also fit using the R LME function `lme` in the *nlme* library (version 3.1-117), including a model with a single random intercept for series ("RI") as well as a model that assumes a CAR structure within each time series ("CAR"). Finally, we fit ARMA models using the R function `arima`, with autoregressive and moving average parameters obtained by optimizing the Akaike Information Criterion (AIC). Note that in the Supplement, in order to more tractably compare with frequentist results, we have transformed our `alpha` parameters into parameters that more conventionally reflect regression-based estimates, with the "Prep Scooping" task chosen as referent (so that the intercept term represents the mean for the "Prep Scooping" task) and task-specific regression parameters corresponding to differences in mean between task and reference. This was very simple to achieve by applying the required linear transformation to each posterior sample.

In general, frequentist estimates were similar to, but not identical with, estimates from our proposed model. In particular, some regression parameters that were significant by frequentist methods corresponded in our proposed model to credible intervals that contained zero. For example, *source enclosure* coefficients for NSAM were significant in the ARMA model but corresponded to credible intervals containing zero in our proposed model; the OPC "Scooping LiOH" coefficient was significant in the OLS model but the corresponding credible interval from our model did contain zero. On the other hand, many coefficients for which our credible intervals did not contain zero corresponded to frequentist coefficients that did contain zero. This was true for many CAR coefficients, whose confidence intervals were extremely wide compared with other methods. Note also that the frequentist standard deviation estimates did not in general match the posterior distributions of the $\sigma_e(w)$ parameters of our proposed model, although in the case of OLS and CAR, they are not even directly comparable, since they represent the scale of errors that are assumed to be autocorrelated and hence do not represent the scale of independent errors. For NSAM, Figure S3 (in supplementary online material part 2) illustrates that the *total* posterior standard deviation estimates ($\sqrt{\sigma_\zeta^2 \mathbf{b}(t)^{\mathrm{T}} \mathbf{b}(t) + \sigma_e^2(w)}$, incorporating both levels of error) were slightly less than the OLS, RI and CAR standard deviation estimates. Finally, note that the frequentist models do not allow for task-dependent variance components. It's worth remarking that all frequentist models other than ARMA demonstrated significant autocorrelation in errors, as evidenced by Durbin-Watson test for serial autocorrelation.

For all four nanoparticle measurements and for the *source enclosure* covariate model, Figure S5 (in supplementary online material part 2) shows plots of observed measurements Y_{ir} by expected value μ_{ir} along with a 95% credible intervals for μ_{ir}. Corresponding plots for *enclosure type* covariate model (not shown) were almost identical. These plots demonstrate good fit of the proposed model to the data. As an example of how our proposed model fits the autocorrelated data within a time series, Figure S6 (in supplementary online material part 2) shows the observed data for a single series $\mathbf{Y}_i$ (March 16, 2006 NSAM sampling of Baghouse location, shown in black), overlaid with the mean profile $\boldsymbol{\mu}_i$; the posterior mean of $\boldsymbol{\mu}_i$ is shown in red, while 100 samples from the posterior distribution of $\boldsymbol{\mu}_i$ are shown in yellow. Essentially, the red curve represents the realization of a smooth stochastic process modeled by the spline term, while differences between the black and red curves represent the independent error process.

As a final note, we assessed convergence of the MCMC chains by applying the Gelman and Rubin convergence diagnostic (available in the R package *coda*) to two independent chains for each data analysis performed. The convergence statistic, $\hat{R}$, should be close to one, preferably less than 1.2 or (a more stringent criterion) less than 1.1. As it was not possible to use all variables in the multivariate chain to calculate $\hat{R}$, we assessed convergence only for the α variables, i.e. those governing the task-specific means. For the primary example, $\hat{R} = 1.01$. In the secondary example, $\hat{R}$ ranged between 1.01 and 1.07 for all but one analysis, SMPS with the covariate Enclosure Type: None, which resulted in $\hat{R} = 1.19$. The slightly worse convergence for this data set was likely the result of collinearity between the covariate and some of the tasks.

Simulations

We conducted several simulation experiments to investigate the properties of the proposed method when applied with a frequentist interpretation. For each experiment, we used posterior statistics obtained from the NSAM model (with covariate *enclosure type*) as the basis of the simulation experiment, using posterior means for regression coefficients and posterior medians for variance components (see Table S5 in supplementary online material part 2). For simplicity, we omitted the *enclosure type* regression coefficient. For each simulated data set within each experiment, we kept the same number of measurements (168) and profiles (3), with identical times in each series. For two sets of experiments, we assumed six separate values of LOD: $\delta \in \{0,20,25,30,40,50\}$. Note that the same spline basis functions were used to generate a non-stationary sequence of errors. For each value of δ we simulated 500 data sets, fit our proposed method (with 2500 burn-in samples, subsequently obtaining a chain of length 20,000 and thinning by 10), and fit three frequentist methods: OLS, CAR, and ARMA, as described above in the Examples section, with the value of below-detection data set to half the LOD (on the original scale before applying the logarithmic transform).

Figure S7 (in supplementary online material part 2) displays boxplots of the percent of left-censored values by LOD over the 500 data sets simulated for each LOD in each set of simulations. Figure S8 (in supplementary online material part 2) shows the root-mean-square-error (RMSE) of frequentist estimates or frequentist interpretations of Bayesian posterior statistics for the intercept (i.e. “Prep Scooping” reference task) and four other coefficients representing differences between task-specific mean and reference task. Figure S9 (in supplementary online material part 2) shows the percent coverage of the 95% credible interval or the nominally 95% confidence interval for the same coefficients. Note that lower-than-nominal coverage indicates bias in estimation of coefficients or standard errors, while higher-than-nominal coverage indicates statistical inefficiency. Finally, Figure S10 (in supplementary online material part 2) shows boxplots of the ratio of standard error (or posterior standard deviation) to simulation standard deviation, representing the level of bias in estimating the sampling standard deviation of the estimate or posterior statistic interpreted as an estimate. It is evident from the RMSE plots in Figure S8 (in supplementary online material part 2) that our proposed method is more efficient than the competing methods, generating the lowest values of RMSE in every case. The percent coverage plots shown in Figure S8 (in supplementary online material part 2) demonstrate more accurate coverage from our proposed method than from OLS (whose coverage is too low at low values of δ and sometimes too high at high values of δ), from CAR (which tended to have higher-than-nominal coverage and thus potentially leading to statistical inefficiency), and from ARMA, which failed to achieve nominal coverage in a variety of ways. The boxplots of ratios shown in Figure S10 (in supplementary online material part 2) reinforce the bias and efficiency point, showing biased estimates of sampling standard deviations for all frequentist methods. In general, it was difficult for frequentist methods to accurately estimate variation using an erroneous error model. As expected, this was especially true for OLS, which almost always underestimated the sampling standard deviations. Note that our method resulted in lower-than-nominal coverage (about 90%) for two tasks that had small sample size, but the other methods also performed poorly for those tasks (with either unstable or higher-than-nominal coverage).

In the final set of simulations, we set $\delta = 0$ (no below-detection values) but generated the error sequence using an AR(1) or AR(2) model, in order to examine the robustness of our method when the true data-generating mechanism followed a more conventional error model. We fit four AR(1) models, with autocorrelation parameters 0.10, 0.25, 0.50, and 0.75 respectively, and fit three AR(2) models, with autocorrelation vectors (0.25,-0.50), (0.50, -0.25), and (0.50,-0.50). Figures S11, S12, and S13 (in supplementary online material part 2) display results. RMSE values were about the same for all methods (except OLS in some cases); in terms of interval coverage our proposed method performed about as well as CAR and ARIMA in the AR(2) models and in the AR(1) models with lower autocorrelation, but tended to break down at the highest level of autocorrelation (0.75). We surmise that the reason for this is that the dense placement of knots was inconsistent with the high levels of autocorrelation, leading to an insufficiently high value for the maximum possible autocorrelation that could be modeled; the remedy, in such a situation, would be to use a sparser set of knots.

<u>Code for Example 2:</u>

We used the JAGS model to analyze the four nanoparticle datasets defined as follows:

```
model{
  tauZ ~ dnorm(0,1.0E-6)T(0,1.0E10) # 1/sigma^2 for spline coefficients

  for(j in 1:m){# Number of tasks
    tauE[j] ~ dnorm(0,1.0E-6)T(0,1.0E10) # 1/sigma^2 for tasks
  }

  for(j in 1:mm1){# Number of tasks minus one
    alpha[j] ~ dnorm(0,0.01) # Task coefficients
  }

  for(l in 1:d){ # Number of additional covariates
    beta[l] ~ dnorm(0,0.0001)  # Coefficients for additional covs
  }

  for(i in 1:n){ # number of profiles (series)
    for(h in 1:k){ # number of basis vectors
      zeta[i,h] ~ dnorm(0, tauZ) # spline coefficients
    }
  }

  # Model specification for above-DL measurements
  for(r in 1:numSamplesDetect){ # number of measurements above DL
    # Series error process
    muTimedep[r] <- inprod(B[r,1:k],zeta[idSeries[r],1:k])

    # Task mean
    muTask[r] <-  alpha[idTask[r]]

    # Effect of other covariates
    muExtra[r] <- inprod(x[r,1:d], beta[1:d])

    # Expected value of measurement
    mu[r] <-  muTimedep[r] + muTask[r] + muExtra[r]
    Y[r] ~ dnorm(mu[r], tauE[idTask[r]])
  }
}
```

We remark that in JAGS, the normal distribution is specified in terms of precision (inverse of variance) rather than by variance, and that prior distributions for variance components are also specified in terms of precision parameters using a half-normal distribution symbolized by the string "`dnorm(0,1.0E-6)T(0,1.0E10)`". We also note that since series length T_i varied across series, it was necessary to use an indexing scheme where each individual measurement Y_{ir} (the natural logarithm of the measured value) was related back to its parent series i through the vector `idSeries`, i.e. the value of `idSeries` corresponding to (Y_{ir}, t_{ir}) is simply i, and that the measurement data were passed into JAGS in one long vector "`Y`". Thus, the model definition also required the total number of measurements `numSamples` $= \sum_{i=1}^{n} T_i$. Similarly, specific tasks were identified by the vector `idTask`. To use the model, we saved it in a text file (e.g. `nano.jags`) and supplied it to the *rjags* function *`jags.model`* along with the data and initial values:

```
theSampler <- jags.model("nano.jags", dataList, initList)
```

The data object `dataList` was a list of the following named elements, corresponding to data objects defined in the model: `numSamples`, `n`, `m`, `k`, `d`, `idSeries`, `idTask`, `Y`, `x`, and `B`. Note that the $\left(\sum_{i=1}^{n} T_i\right) \times k$ matrix `B`, which contained the spline basis functions evaluated at the sampling time points, was constructed using the b-spline function `bs` available in the R package *splines* (version 3.1.0). The initial values object `initList` was a list with exactly one element (corresponding to one intended Markov chain), which itself was a list having the following named elements, corresponding to parameters that complete the model definition: `alpha`, `beta`, `zeta`, `tauZ`, and `tauE`. Initial values for `alpha` and `beta` were obtained via ordinary least squares (OLS) linear regression; although the m–length vector `tauE` represents a distinct error precision parameter for each task, it was initialized as an m–length vector, each of whose entries was the inverse of the single variance parameter obtained from OLS. The $n \times k$ matrix `zeta`, containing the spline coefficients, was initialized to the zero matrix, and the spline precision parameter `tauZ` $= \sigma_{\zeta}^{-2}$ was initialized to a moderately large value, 10. We pre-sampled 10,000 iterations to "burn-in" the chain, i.e. initialize it so that the values we store for subsequent analysis are assured to arise from the stationary Markov-chain that represents the posterior distributions of the parameters of interest:

```
update(theSampler, 1000)
```

Finally, we collect the desired posterior samples. In this case, we collect 1000 samples from a chain of length 50,000, thinning by 50 (i.e. saving only once every 50 iterations) in order to reduce autocorrelation of values within the chain:

```
results <- coda.samples(theSampler, c("alpha","beta","zeta","tauZ","tauE"),
      50000, thin=50)
```

The object `results` is a list of length equal to the length of the initial values list `initList` (one in this case), with a potentially separate chain for each set of initial values. The chain stored in the first element of `results` was used to compute posterior statistics. Although the Bayesian paradigm is based on statistical principles that are fundamentally distinct from those upon which the more commonly understood frequentist paradigm is based, notably the concept that unknown parameters are treated as random rather than fixed quantities, Bayesian posterior statistics are often informally interpreted in a manner similar to their more common frequentist statistcs: the posterior mean and median are often interpreted as parameter estimates, the posterior standard deviation is often interpreted as a standard error, and the posterior 2.5Th and 97.5th percentiles are often interpreted as (respectively) upper and lower limits of a 95% confidence interval (although formally the confidence interval is based on frequentist concepts and instead the term *credible set* is used in Bayesian contexts). Note that posterior statistics for the variance parameters are obtained simply by first transforming the tau parameters (e.g. σ_{ξ} is the inverse square-root of `tauZ`).

<u>Code for Example 1</u>

The following code displays several modifications:

```
model{
  tauZ ~ dnorm(0,1.0E-6)T(0,1.0E10) # 1/sigma^2 for spline coefficients
  tauA ~ dnorm(0,1.0E-6)T(0,1.0E10) # 1/sigma^2 for series random intercept
  alpha0 ~ dnorm(0,0.01) # Reference task mean (intercept)

  for(j in 1:m){# Number of tasks
    tauE[j] ~ dnorm(0,1.0E-6)T(0,1.0E10) # 1/sigma^2 for tasks
  }

  for(j in 1:mm1){# Number of tasks minus one
    alpha[j] ~ dnorm(0,0.01) # Task coefficients
  }

  for(i in 1:n){ # number of profiles (series)
    for(h in 1:k){ # number of basis vectors
      zeta[i,h] ~ dnorm(0, tauZ) # spline coefficients
    }
    a[i] ~ dnorm(alpha0, tauA)
  }

  # Model specification for above-DL measurements
  for(r in 1:numSamplesDetect){ # number of measurements above DL
    # Series error process
    muTimedep[r] <- inprod(B[r,1:k],zeta[idSeries[r],1:k])

    # Task mean
    muTask[r] <-  inprod(Wmod[r,1:mm1],alpha[1:mm1])

    # Expected value of measurement
    mu[r] <-  a[idSeries[r]] + muTimedep[r] + muTask[r]

    Y[r] ~ dnorm(mu[r], tauE[idTask[r]])
  }

  # Model specification for below-DL measurements
  for(r in 1:numSamplesND){# number of measurements below DL
```

```
    # As above
    muTimedepND[r] <- inprod(BND[r,1:k],zeta[idSeriesND[r],1:k])
    muTaskND[r] <-  inprod(WmodND[r,1:mm1],alpha[1:mm1])
    muND[r] <-  a[idSeriesND[r]] + muTimedepND[r] + muTaskND[r]

    # Note that YND data are passed in as NA (missing)
    #   but will be imputed by the model
    YND[r] ~ dnorm(muND[r], tauE[idTaskND[r]])

    # Detection index (enforces the constraint that YND < DETECTIONLIMT)
    isDetectND[r] ~ dinterval( YND[r] ,  DETECTLIMIT)
  }
}
```

First, note that no covariates have been introduced into this model, so that code referring to `x` and `beta` have been deleted. Next, note that a more typical regression parameterization has been used to represent the task mean portion of the model, so that the intercept `alpha0` corresponds to the mean of the reference task and now `alpha` corresponds to `mm1` = $m-1$ regression parameters representing mean differences from the reference task; the corresponding covariate matrix is denoted as `Wmod`. Additional code for the random intercept `a` has been introduced, with the principle of hierarchical centering applied in the specification of the random intercept distribution. Finally, and most importantly, the below detection data have been separated from the above-detection data, with data elements `numSamples`, `idSeries`, `idTask`, `Y`, `B`, `muTimedep`, `muTask`, and `mu` having been duplicated with "ND" counterparts. An additional data element `isDetectND` appears in the loop corresponding to the below-detection data. When passed as data to the `jags.model` function, it is simply a zero vector of length `numSamplesND`; and it corresponds to a vector `YND` of NA's the same length. The zeroes in `isDetectND` and the NAs in `YND`, together with the JAGS distribution `dinterval` specifications, indicate that the value `YND[r]`is known only to lie below the value `DETECTLIMIT`, i.e. is left-censored by `DETECTLIMIT`. The reason for the separation is that it is now very easy to initialize the variable `YND` at a vector of assumed fixed values (e.g. half of the LOD), so that the sampler can more easily find the stationary distribution. Without the separation, initialization becomes very difficult because JAGS does not easily facilitate the initialization of missing values mixed together with observed values in the same data object. Finally, we specify more informative priors for `alpha0` and `alpha` in order to prevent absurdly small values from being sampled for the missing data in `YND`.

We also fit a model that adjusts for instrument-specific effects along the lines of model (3) by replacing the distribution specification for the intercept `a` with the following:

```
    a[i] ~ dnorm(a0[i], tauA)
    a0[i] <- alpha0 + inprod(Z[i,1:l], gamma[1:l])
```

References

Gelfand A, Sahu SK, Carlin B. 1996. Efficient Parameterizations for Generalized Linear Mixed Models,". *Bayesian statistics* 5: 48-74

Gelman A. 2006. Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). *Bayesian analysis* 1: 515-34

Prautzsch H, Boehm W, Paluszny M. 2002. *Bézier and B-spline techniques*. Berlin: Springer.