Estimating Age-Specific Mortality Under Five Years

Katherine Paulson

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Public Health

University of Washington
2019

Committee:
Haidong Wang
Laura Dwyer-Lindgren

Program Authorized to Offer Degree:
Global Health

University of Washington

Abstract

Estimating Age-Specific Mortality Under Five Years

Katherine Paulson

Chair of the Supervisory Committee:
Haidong Wang
Department of Global Health

Abstract

Objective: To propose a novel method for estimating under-five mortality by day of age.

Data: All available complete birth histories from the Demographic and Health Survey (DHS) and the Multiple Indicator Cluster Survey (MICS), along with state-level vital registration from the United States (1959-2004) and Brazil (1979-2013), were included in this analysis. A total of 85 DHS surveys from 46 locations and 60 MICS surveys from 42 locations were compiled.

Methods: First, modified Oppermann, Steffensen, Bourgeois-Pichat, Heligman-Pollard, exponential, gamma, generalized-gamma, beta, and power-law models were fit to each location-source-year-sex separately by non-linear least squares optimization. The outcome of interest was a conditional probability density: probability of dying on day x conditional on death before 5 years of age. Each model's performance across the dataset was synthesized by comparing overall and age-specific residuals. Next, these parametric model fits were used to compile a model database, and a relational model system was proposed. With this relational model, summary measures of early, late, post-neonatal, and 1-4-year mortality are decomposed into by-day probabilities of death.

Results: Median sum of square log-errors is lowest for the power-law, beta, gamma, and generalized gamma models, closely followed by the Weibull, Steffensen, and Heligman-Pollard models. The exponential and Bourgeois-Pichat are ill-performing by comparison, and the Oppermann model performs well for a subset of cohorts. Each of the top models tends to under-estimate mortality in the first few days. The relational model application compiled from these model fits resulted in mean coverage of 46.8 percent for MICS, and 44.0, 56.1, and 52.6 percent for DHS, Brazil VR, and USA VR.

Conclusions: Childhood mortality has received a lot of attention by global health experts, and understanding age-specific mortality within the under-5 age group is essential to understand the progress made to-date and the challenges remaining. However, most research has focused on a few discrete age groups: neonatal, infant, and child (1-4 years). This paper examines the potential of various parametric models to describe the age pattern of mortality. We find that while no one existing model well describes the age pattern by day, a collective can be used to form an ensemble that when applied through a relational model, gives us the potential to take mortality from any two aggregate age groups and extrapolate to any age group of interest.

Introduction

Background

Over five million deaths occurred to children under five years in 2017, according to the Global Burden of Disease Study [**1**]. Reducing under-5 mortality has been a major global heath focus and achievement: there was a reduction from 216 deaths per 1000 livebirths in 1950 to 39 deaths per 1000 livebirths in 2017. The fourth Millennium Development Goal was to reduce under-5 mortality by two-thirds between 1990 and 2015, and the international commitment to this goal was renewed in the Sustainable Development Goals, where indicator 3.2.1 pertains to under-5 mortality, and 3.2.2 focuses on neonatal mortality. Measurement of these indicators is crucial for both the planning needed to meet these goals, and the evaluation used to assess progress on them.

To better understand the drivers of under-5 mortality, the successes to date, and the opportunities for further reduction, it is important to examine the distribution of under-5 deaths by age and sex. The main causes of under-5 mortality are not equally relevant across the entire under-5 age span. For example, deaths due to birth complications are most likely to occur during the first week [**10**]. Therefore, groups interested in particular causes of under-five mortality may find age-specific information useful. This also means that changes in the age pattern of under-five mortality over time can tell us about the relative change of different causes of death. For example, most of the reduction in child mortality observed in response to the MDGs was between the ages of 1 and 4 years, with the least improvement observed in the first week of life [**2**]. This discrepancy may be attributed to programs that focus on pneumonia, diarrhea, malaria, and vaccine-preventable diseases – each of which is a major cause of death after the first month of life [**2**].

Previous work

Evaluations of the distribution of under-5 mortality have mainly used categorical age groups: early-neonatal (ENN; first week), late-neonatal (LNN; 7-27 days), post-neonatal (PNN; 1 month - 1 year), and 1-4 years. These groups were chosen with length of the interval increasing with age to reflect the rate of change of mortality. Some large-scale studies, like the Global Burden of Disease study and the United Nations World Population Prospects, report modeled mortality by these discrete age groups. For the GBD, early, late, post-neonatal, and 1-4 mortality are modeled individually, and then scaled to total under-five mortality. The UN WPP reports neonatal (0-1) and 1-4 mortality separately.

The 2005 Lancet series on neonatal survival reports observed daily risk of death during the first month of life, based on 47 Demographic and Health Survey datasets from 1995 to 2003 [**2**]. From this analysis, it is apparent that daily risk of death is not constant within each of the standard categorical age groups. The phenomenon called age heaping, where deaths are more likely to be reported at even time intervals, such as one week or one month, is also clear.

An analysis of daily risk of death such as the one reported by Lawn et al., or an analysis of under-5 mortality with categorical age (ENN, LNN, PNN, and 1-4), is useful for those interested in a single location, or an age-specific comparison of locations, but it does not provide a more generalizable description of mortality over age. The field of demography is rich with mortality laws: mathematical expressions to describe trends in mortality over age. Several outcomes of interest are used in this type of analysis. A continuous set: the survival, hazard, and density functions; and an equivalent discrete set coming from life-table analysis: l_x, q_x, and d_x. For definitions of each of these outcomes, see the appendix.

One classic example of a mortality law is the Gompertz law, which presents the mortality hazard at age x, $h(x)$, as log-linear over age: $h(x) = \alpha e^{\beta x}$ [**3**]. To describe the rapidly decreasing mortality with age in young children, Bourgeois-Pichat parameterized the cumulative proportion dead in a cohort by day x as $F(x) = \alpha + \beta[\log(x+1)]^3$, with a focus on the age span of one month to one year [**4**]. The Bougeois-Pichat model rests on an assumption that child mortality can be decomposed into exogenous factors (due to the environment, or injury), and endogenous factors. Therefore, α represents an exogenous constant and the other term in the model represents decreasing endogenous mortality. Additional research has discussed variations on the Bourgeois-Pichat model, and specific instances where the assumptions do not hold.

Early actuarial work by Oppermann, followed up by later work by Steffensen, produced models for mortality by year in children. The Oppermann model is:

$$h(x) = \frac{\alpha}{\sqrt{x}} + \beta + \gamma\sqrt{x}$$

and the Steffensen model is:

$$\log(l_x) = 10^{\alpha\sqrt{x}+\beta} + \gamma$$

where $h(x)$ is the mortality hazard at age x and l_x is the proportion that lives to age x [**5**] [**6**]. The theory behind the Oppermann model, from a 1870 talk, is not particularly well documented, however, Steffensen explains the basic principle that the mortality hazard approaches infinity as x approaches zero, due to the $\sqrt{x}$ in the model. Steffensen attempts to interpret and improve upon the Oppermann model.

The Heligman-Pollard model also attempts to parameterize under-5 mortality by year. The model, which produces estimates for a complete age span, is the sum of a series of terms each representing a different age group with its own mortality feature. The full model is:

$$\frac{{}_1q_x}{{}_1p_x} = \alpha^{(x+\beta)^\gamma} + \delta e^{-\epsilon(\ln(x)-\ln(\zeta))^2} + \eta\theta^x$$

where ${}_1q_x$ is the probability of dying between the age x years and $x+1$ years conditional on survival to age x, and ${}_1p_x$ is the probability of survival, or $1 - {}_1q_x$ [**7**]. The first term is intended to capture the mortality pattern in young children. The second term accounts for the phenomenon of increased rate of injury deaths in young adults, and the third term is similar to a Gompertz model for increasing mortality in adulthood.

In addition to parametric models, the age pattern of mortality may be estimated using model life tables. A database of known age patterns serves as a baseline, and these standard life tables are adjusted with a relational model indexed to a summary measure, such as life expectancy, ${}_5q_0$, and/or ${}_{45}q_{15}$. The classic example of a relational model for mortality is the Brass Model, which describes the logit of q_x from the standard life tables as linearly related to the logit of q_x from the observed data [**8**]. This linear model can be fit to ${}_5q_0$ and ${}_{45}q_{15}$, and then used to predict q_x for any age x of interest. Model life tables with relational models are useful when we have a

subset of cohorts with only summary information, and another subset of cohorts with complete and high-quality age schedules. To my knowledge, no one has published work to use a model life table system to estimate under-five mortality by finer age groups than 0-1 and 1-4.

Available data for under-five mortality

There are two main sources of data that can be used to investigate under-five mortality: complete birth histories and vital registration. Complete birth histories (CBH) are household surveys, where respondents list all births, and age of deaths for children who have died. Complete birth histories, in particular, may be subject to age heaping. Vital registration (VR) systems are more robust and systematic, by recording date of birth and date of death routinely, however, can be subject to their own biases, such as differential completeness. Additionally, detailed age information may not be publicly available for many sources of vital registration. Both CBH and VR are utilized for this thesis.

Objectives

The primary aim of this thesis is to explore methods for estimating and describing the by-day age pattern of mortality under the age of five years. The paper explores existing methodology and findings, as well as data availability and limitations. We then propose a two-phase method, involving both parametric and relational modeling techniques.

First, I fit and evaluate existing parametric functions to describe the age pattern of under-five mortality. Although several models, as described, have been identified that account for rapidly decreasing child mortality, none was designed to produce estimates by day. Additionally, each of the models is rather old, and may not have been validated on modern data. Finally, there are several additional parametric models we test that may not have been evaluated for their performance in this particular application. This thesis is then an extension of previous work to parameterize the age pattern of mortality by contributing an assessment of models for their ability to predict under-five mortality by day.

Next, we leverage the parametric fits to compose a model database as input into a more flexible relational model approach, that uses observed early-neonatal, late-neonatal, post-neonatal, and 1-4 mortality to predict a full age series. While classic relational models use empirical tables to compile a model database, this may not be a feasible approach for under-five mortality by day. Detailed-age data is noisy, particularly for CBH, even when aggregated across many locations. Vital registration, too, can suffer from noise related to small sample sizes. Therefore, empirical data needs to be smoothed or aggregated to resemble a plausible age distribution that can be utilized in a relational model system. This thesis proposes an ensemble of parametric models as an alternative composition for a model database.

Given the limitations of age-specific data for mortality under the age of five, and the demand for estimates at custom age groups, the method proposed herein can be expanded upon to decompose more-reliable summary measures of under-five mortality into any under-five age group of interest.

Methods

Input data

Two sources of complete birth histories were extracted: the Demographic and Health Survey (DHS) and the Multiple Indicator Cluster Survey (MICS). In the birth history components of these surveys, respondents list all children living or deceased, including age at death for children who have died. Age at death is recorded by the day for the first month, by the month until the first year, and by the year thereafter. A total of 85 DHS surveys from 46 locations and 60 MICS surveys from 42 locations are available with birth history. All available DHS and MICS birth histories were included.

Two sources of vital registration were extracted to complement the CBH – from the United States and Brazil. United States National Vital Statistics System is publicly available at the state level from the National Center for Health Statistics, for the years 1959-2004. Vital registration from the Mortality Information System of Brazil is available for 1979-2013. These two countries were selected because many years were available, with detailed age data, and information about sub-national units. An expansion of this research beyond a proof-of-concept would ideally include all available VR data with detailed-age information.

The outcome of interest for this thesis is a density function. This outcome is convenient, because it may be re-framed as a conditional density function (probability of dying on day x, conditional on death before age 5 years) to facilitate scaling of mortality to established ${}_5q_0$ values, for which we have more reliable data. A density function also does not require accurate data for births or population. However, some work needs to be done to convert between parametric functions from the literature, and their corresponding density functions.

Deaths were recoded to singe-day units from 0-30 days, single-month units from 1-11 months, or single-year units from 1-4 years. Deaths recorded with a more specific age of death, such as 50 days, were recoded to the broader category – in this case 1 month. Deaths assigned to a more general category, such as 2 weeks, were distributed across the category proportional to the known distribution in that category. In the example of deaths assigned to 2 weeks, these deaths would be split across the period of 14-20 days proportional to distribution of deaths known to have occurred on one of these days.

For the remainder of this paper, I will refer to a subset of the data from a single location-source-year-sex as a *cohort*, while acknowledging that this is a synthetic cohort from a period.

After recoding, the proportion of deaths in a particular cohort occurring in each age category was calculated. Proportion of deaths was divided by the length of the interval in days (30.4 days per month and 365 days per year) to get the mean daily proportion in the interval. Then, the age at death was assigned to the midpoint of the interval in days for age categories in months and years, assuming the mean density occurs at the midpoint. From this process, we obtained age at death in days and corresponding proportion of under-5 deaths, as input into the series of models to estimate

the density function over age in days. By including one data point each for months 1-11 and one data point each for years 1-4, we place more weight on the earlier ages in model fitting, which is appropriate given the rate of change. Cohorts with fewer than 200 deaths were dropped from the analysis. The cutoff of 200 was chosen based on the distribution of sample sizes in the dataset, the number of age groups we are trying to estimate for, and the goal of excluding data that is too noisy.

Parametric models

The data from each cohort was fit separately to the Heligman-Pollard, Bourgeois-Pichat, Oppermann, and Steffensen models. The following general parametric models, used commonly in survival analysis, were fit as well: exponential, Weibull, gamma, and generalized gamma (Table 1). The beta model was added because it describes a family of probability distributions with a bounded domain, like the conditional density of interest. Initial data exploration indicated that the proportion of under-5 deaths occurring at age x ($f(x)$) appears to vary approximately linearly with x when both x and $f(x)$ are in log-space. From this observation, we added the power-law relationship $f(x) = \alpha x^{\beta}$, derived from $\log(f(x)) = \alpha + \beta \log(x)$, to the list of models as well.

Name	Original function	Adaptation
Bourgeois-Pichat [**4**]	$F(x) = \alpha + \beta[\log(x+1)]^3$	$f(x) = \beta[log(x+1)]^2/(x+1)$
Oppermann [**5**]	$h(x) = \frac{\alpha}{\sqrt{x}} + \beta + \gamma\sqrt{x}$	$f(x) = \left[\frac{\alpha}{\sqrt{\frac{x}{365}}} + \beta + \gamma\sqrt{\frac{x}{365}}\right] \cdot e^{2\alpha\sqrt{\frac{x}{365}} + \beta\frac{x}{365} + \frac{2}{3}\gamma(\frac{x}{365})^{\frac{3}{2}}}$
Steffensen [**6**]	$\log(l_x) = 10^{\alpha\sqrt{x}+\beta} + \gamma$	$f(x) = -\alpha\gamma(2^{\alpha\sqrt{\frac{x}{365}}+\beta-1})(5^{\alpha\sqrt{\frac{x}{365}}} + \beta) \cdot e^{(10^{\alpha\sqrt{\frac{x}{365}}+\beta}+\gamma)\frac{\log(10)}{\sqrt{\frac{x}{365}}}}$
Heligman-Pollard [**7**]	$\frac{{}_1q_x}{{}_1p_x} = \alpha^{(x+\beta)^\gamma} + \delta e^{-\epsilon(\ln(x)-\ln(\zeta))^2} + \eta\theta^x$	$f(x) = \epsilon \cdot h(\frac{x}{365})e^{-\int_0^{x/365} h(u)du}$ where $\mathrm{h(x)} = \alpha^{(x+\beta)^\gamma} + \delta$
Exponential	$f(x) = \alpha e^{(-\alpha x)}$	$f(x) = \beta\alpha e^{(-\alpha\frac{x}{365})}$
Weibull	$f(x) = \alpha\beta^\alpha x^{\alpha-1}e^{(-(\beta x)^\alpha)}$	$f(x) = \gamma\alpha\beta^\alpha\frac{x}{365}^{\alpha-1}e^{(-(\beta\frac{x}{365})^\alpha)}$
Beta	$f(x) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}x^{\alpha-1}(1-x)^{\beta-1}$	$f(x) = \gamma\frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}(\frac{x}{5\cdot365})^{\alpha-1}(1-(\frac{x}{5\cdot365}))^{\beta-1}$
Gamma	$f(x) = \beta(\beta x)^{\alpha-1}e^{\beta x}\frac{1}{\Gamma(\alpha))}$	$f(x) = \gamma\beta(\beta\frac{x}{365})^{\alpha-1}e^{\beta\frac{x}{365}}\frac{1}{\Gamma(\alpha))}$
Generalized Gamma	$f(x) = \gamma\beta^{\gamma\alpha}x^{\gamma\alpha-1}e^{(-(\beta x)^\gamma)}\frac{1}{\Gamma(\alpha)}$	$f(x) = \delta\gamma\beta^{\gamma\alpha}\frac{x}{365}^{\gamma\alpha-1}e^{(-(\beta\frac{x}{365})^\gamma)}\frac{1}{\Gamma(\alpha)}$
Power law	$f(x) = \alpha x^\beta$	$f(x) = \alpha x^\beta$

TABLE 1. Mortality models fit and evaluated: both the original functional form and the modified form used in this paper. Parameter symbols have been modified from the original sources for consistency. See the appendix for derivations of the adapted functional forms.

Adjustments were made to each of the models to solve for the density function, and to adapt for this application. The functional forms utilized for this thesis can be found in Table 1. For a full explanation of these adjustments, see the appendix.

Non-linear least squares estimation using the `optim` package in `R` was used to fit the models by minimizing the sum of square errors (SSE), with errors computed in log-space such that:

$$\text{SSE} = \sum_x \log\Big(\frac{f(x) \text{ observed}}{f(x) \text{ predicted}}\Big)^2.$$

Error was computed in log-space because the rapid change in $f(x)$ means a given value of relative error will result in very different absolute error depending on age. Mean residuals over age by model were investigated to examine directional bias by age. The distribution of SSE by source type and model were compared to synthesize model performance across the dataset.

Relational model

Finally, a database of plausible age distributions was compiled, and a relational model based on the Brass model was used to compute full age series from summary values of the observed proportion of under-5 deaths occurring in early, late, post-neonatal, and 1-4 age groups.

Because a model database must consist of distributions that are plausible on their own, it is not possible to use the noisy empirical data for this purpose. Alternatively, by cohort, the top 3 performing parametric models by SSE were selected for inclusion in the model database. The cut-off of 3 was arbitrarily chosen, with the purpose of only including the best-fitting distributions in the database, but other cut-offs, or a weighting scheme, could be explored in future work.

Although we cannot use cohort-specific distributions in the model database, the age distributions look more reasonable from the empirical data at the location-aggregate level. The mean of the input data by source type, sex, and age was computed to generate a set of eight empirical distributions to accompany the parametric fits. A smoothing process was used to account for age heaping in DHS and MICS mean models. A 5-day moving average was used for ages greater than 3 days. For age 3, a 4 day weighted average with weights 0.4, 0.2, 0.2, and 0.2 for ages 2, 3, 4, and 5 days. This smoothing is needed at this phase because in order for distributions to be used as standards, they need to be plausible themselves, and the unsmoothed CBH data is too noisy (we did not smooth before the parametric model fitting, because the parametric models themselves are a type of smoothing process).

Each distribution in the model database was scaled such that the proportions across the complete under-5 age-span sum to 1, in order for them to represent true plausible age distributions.

The original processed input data was expanded to a set of 100 samples, based on mean proportion of under-5 deaths and the assumption of a binomial distribution such that:

$$\mathbf{D}_{i,j} \sim \text{Binomial}(n_i, p_{i,j})$$

where $\mathbf{D}_{i,j}$ is the sample of death counts in cohort i and age group j, n_i is the total number of under-five deaths observed in cohort i, and $p_{i,j}$ is the observed proportion of under-five deaths in cohort i that occur in age group j.

Each distribution in the model database, as well each sampled age series from the input data, was collapsed to the proportion of under-five deaths occurring in early, late, post-neonatal, and 1-4 years. Infant age groups were also included in an aggregate first-year age group. Then, the following was performed separately by cohort in the input data. The sum of square log-difference was computed comparing the data to each distribution in the model database separately, and the 100 distributions with the smallest SSE when compared to the observed data were selected to be

the set of standard distributions. The result was 100 standard distributions for each of 100 samples for each cohort.

Separately by distribution in this set of 10,000, the following relational model was fit:

$$\text{logit}(f_{o,x}) = \alpha + \beta \text{logit}(f_{s,x})$$

where $f_{o,x}$ is the observed proportion of under-five deaths in age group x and $f_{s,x}$ is the proportion of under-five deaths in age-group x in the standard distribution.

Finally, this relational model was used to predict the proportion of under-five deaths by day. This process produced 10,000 estimates for the by-day age distribution of under-five mortality, for every cohort in the dataset. These 10,000 estimates were collapsed into mean, 97.5th and 2.5th quantiles to get a point estimate and uncertainty interval.

Results

After removing 2071 cohorts with fewer than 200 deaths, 5166 cohorts remained. The majority, 1889, of those removed were from US states. The input data from vital registration was smooth, on average, while the input data from CBH demonstrates age heaping at even age intervals and irregular age reporting (Figures 0.1, 0.2, 0.3, 0.4).

FIGURE 0.1. Distribution of raw input data from DHS complete birth histories, for proportion of under-five deaths by age group.

Parametric models

Median SSE is lowest for the beta and power-law models, closely followed by the gamma, generalized gamma, Heligman-Pollard, Steffensen, and Weibull models. The order among these is variable between different source types (Figure 0.5). Although there is significant overlap in the interquartile ranges across models, the exponential, Bourgeois-Pichat, and Oppermann models stand out as being particularly ill-performing. The Oppermann model is variable, however, such that the model fits with the largest SSE are from the Oppermann model, but the median SSE for the Oppermann fits is not far above the median SSE for the well-performing models. The difference in data variance comparing the vital registration to the complete birth history is also stark, as there is a clear distance between SSE from VR and CBH model fits.

All models display some degree of age-specific bias (Figures 0.6, 0.7, 0.8, 0.9). In particular, the median residual for every model demonstrates a tendency to under-predict the proportion of deaths in the first few days of life. Many of these models over-estimate in the first month of life, after the

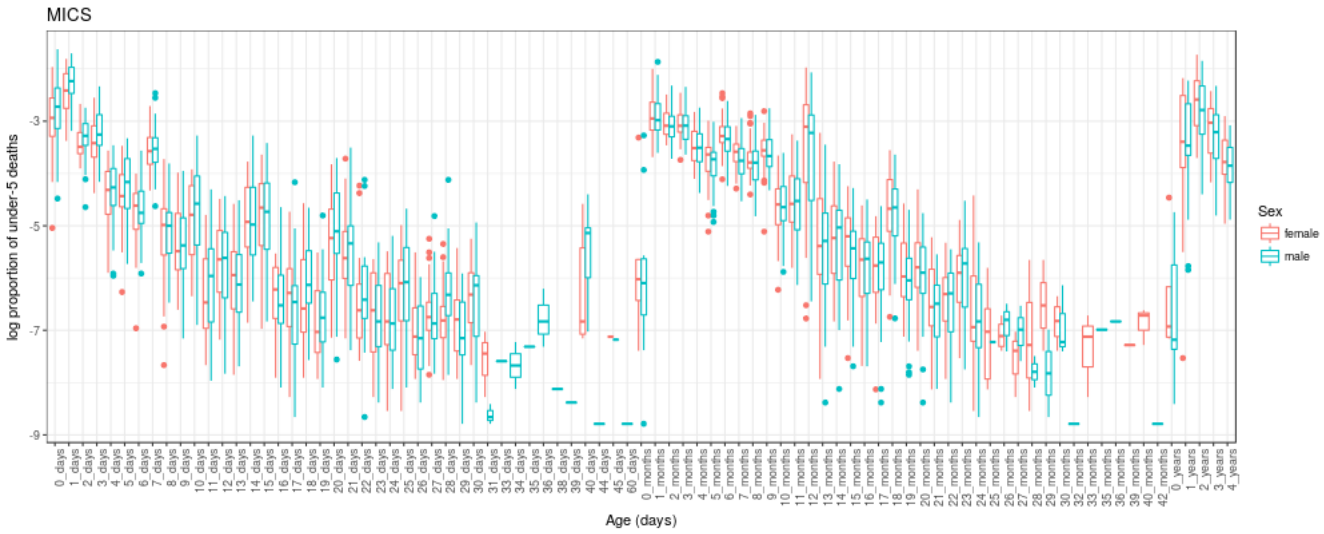

FIGURE 0.2. Distribution of raw input data from MICS complete birth histories, for proportion of under-five deaths by age group.

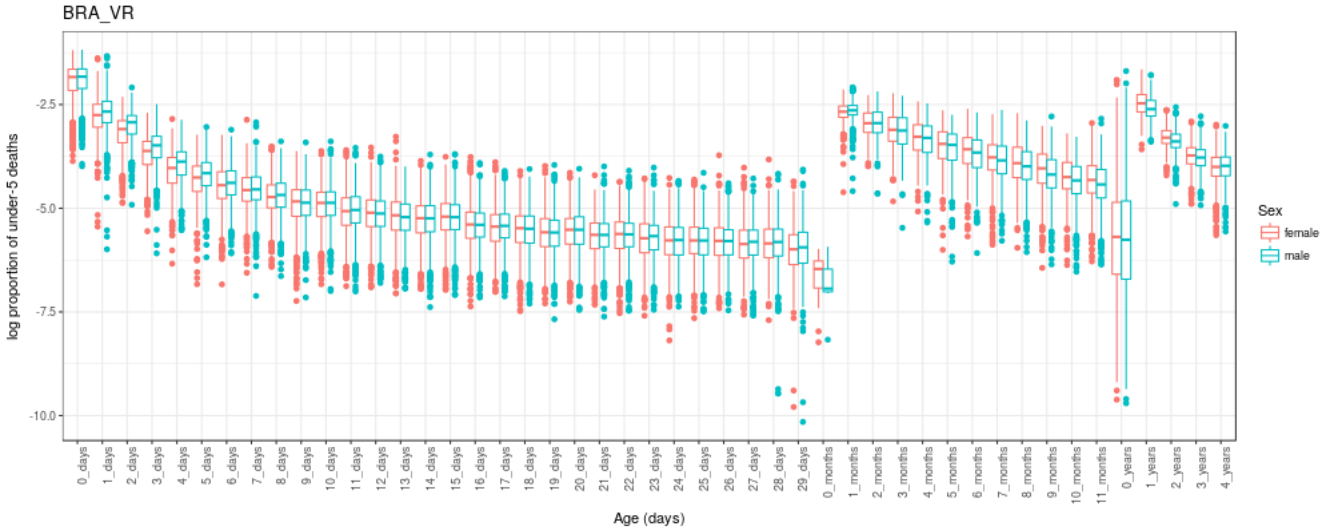

FIGURE 0.3. Distribution of raw input data from Brazil vital registration, for proportion of under-five deaths by age group.

first few days, and under-estimate in the rest of the first year and into the second through fifth year.

While there is a trend among the group of models where some perform better than others on average, there is variability of performance rankings within fits to a single dataset (Figure 0.10). In a subset of top-3 ranking models across the cohorts, all models except the exponential and Bourgeois-Pichat are captured.

Relational model

The distributions included in the model database are shown in Figure 0.11. Distributions are shown on a log-log scale to make the large drop that occurs in the first week most visible. The

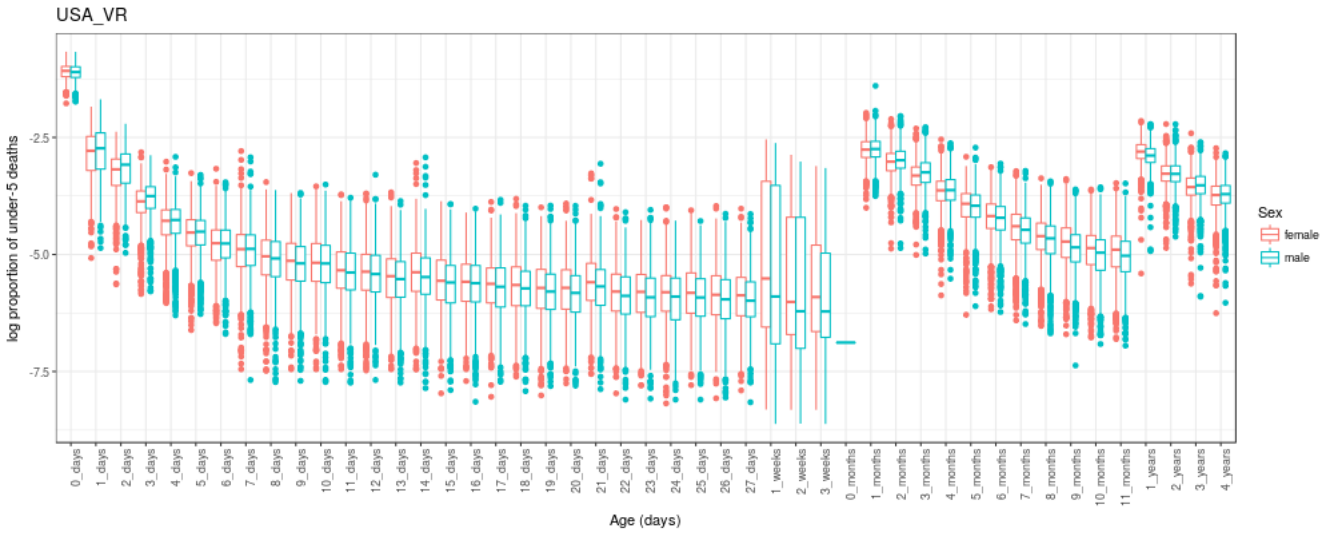

FIGURE 0.4. Distribution of raw input data from United States vital registration, for proportion of under-five deaths by age group.

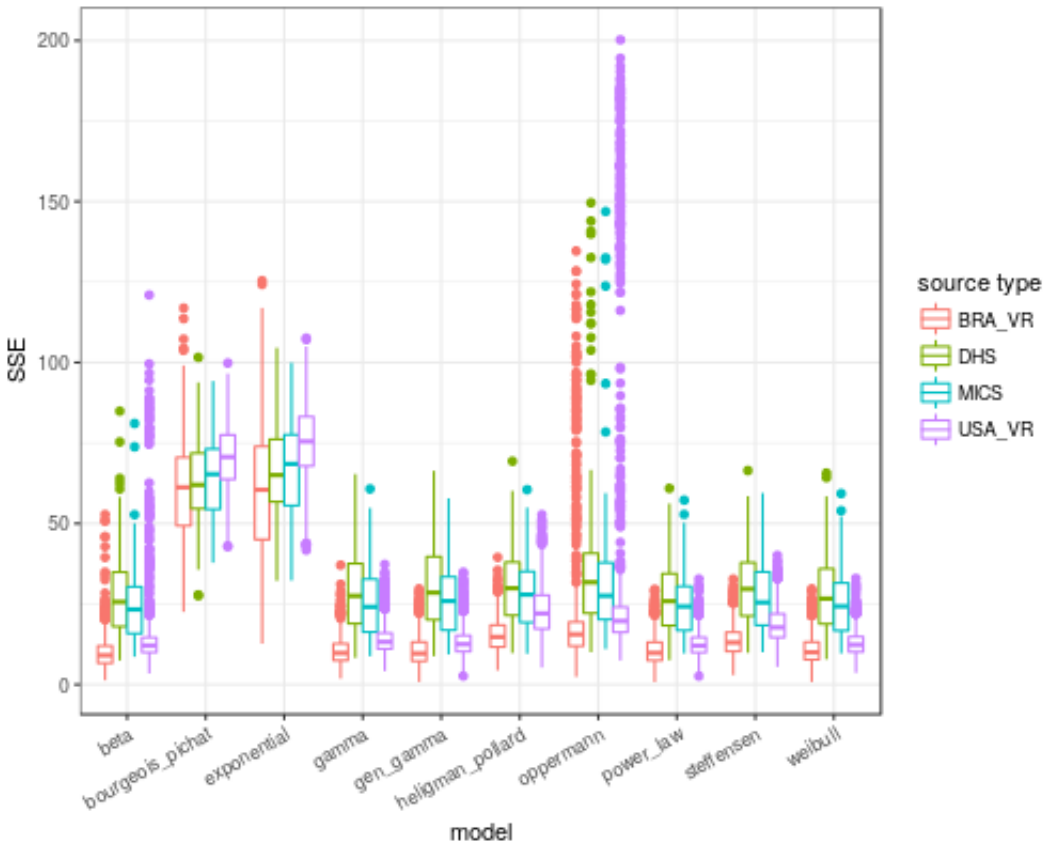

FIGURE 0.5. Boxplots for distribution of SSE across all cohorts included, by model and source type.

Oppermann and Heligman-Pollard models have the ability to take on very different shapes and

Model	Model database	Selected standards
Power-law	30.1	39.5
Beta	26.9	29.6
Weibull	18.7	14.3
Generalized-gamma	13.4	6.3
Gamma	7.9	4.5
Oppermann	1.8	0.8
Steffensen	1.0	0.7
Heligman-Pollard	0.1	0.4
Empirical mean	0.1	3.9
Bourgeois-Pichat	0	0
Exponential	0	0

TABLE 2. Percent composition of model database and selected standard distributions by model type.

still rank highly enough to be included in this set. The empirical mean models for the vital registration sources are smooth, but the empirical means for the CBH sources fluctuate even with the smoothing performed. The models selected for inclusion in model database are described in Table 2.

The set of models selected for inclusion in standard sets, the top 100 closest distributions to each draw of raw data at the aggregate age group level, were similar in composition to the best fitting models when SSE was calculated across the full age span, not just in the five summary age groups. This set of standard distributions is detailed in Table 2. The empirical-mean model was under-represented in the database as compared to the other models, because there were only eight distributions from this model – therefore 3.9 percent is large as a proportion of potential inclusions. The r-squared values from the relational models had mean 0.960 and standard deviation 0.035. The mean and standard deviation slope and intercept for the relational models are 0.974 (SD 0.079) and -0.019 (SD 0.39) respectively.

Mean coverage for the relational model, with final uncertainty interval, by cohort is 46.8 percent for MICS, and 44.0, 56.1, and 52.6 percent for DHS, Brazil VR, and USA VR.

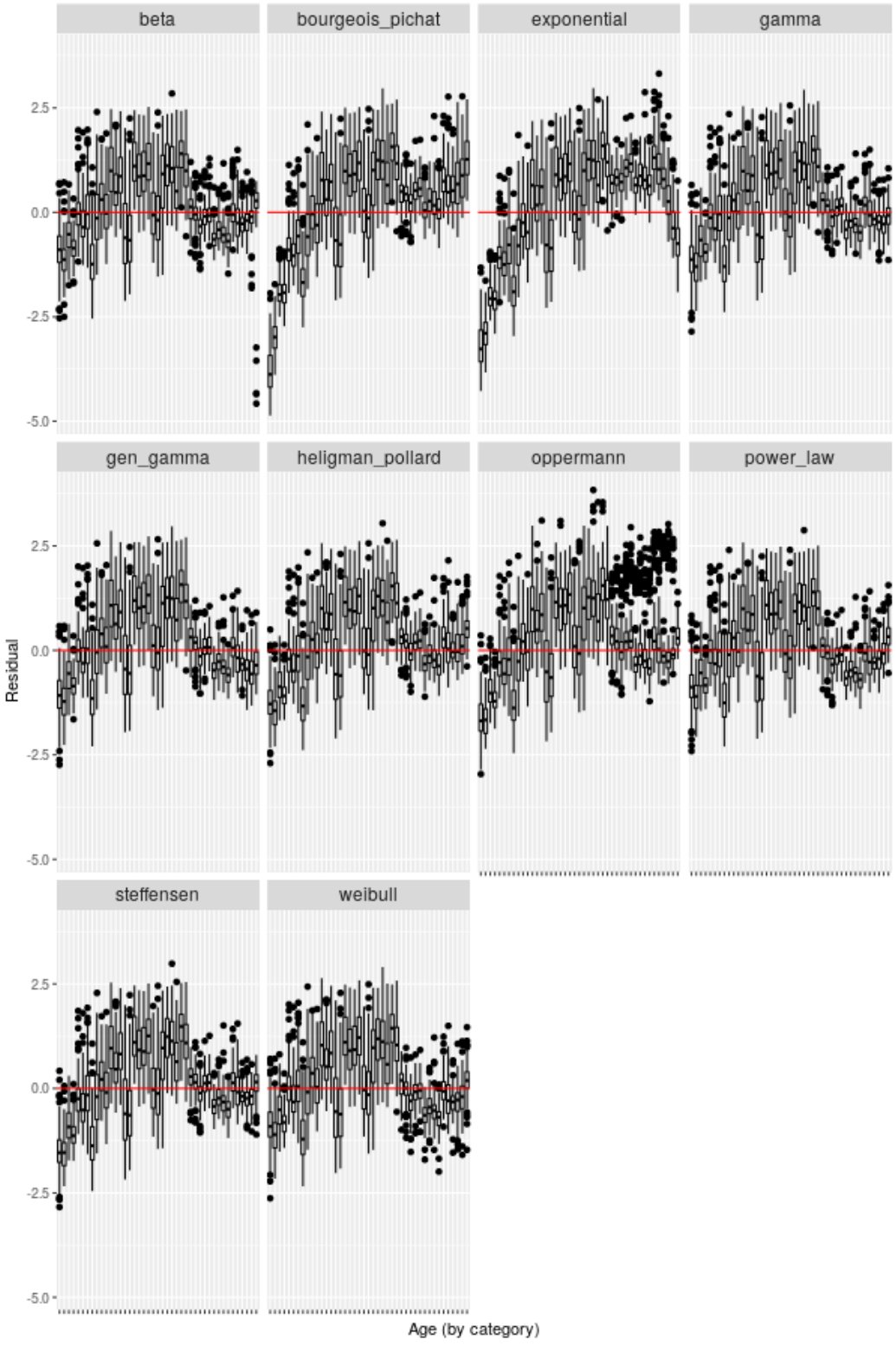

FIGURE 0.6. Boxplots for distribution of age-specific log-errors (log(ratio of predicted to observed proportion of under-5 deaths)), across all cohorts included from DHS, stratified by model. Age (x-axis) is categorical: 0-27 days by day, $1-11$ months by month, $1-4$ years by year.

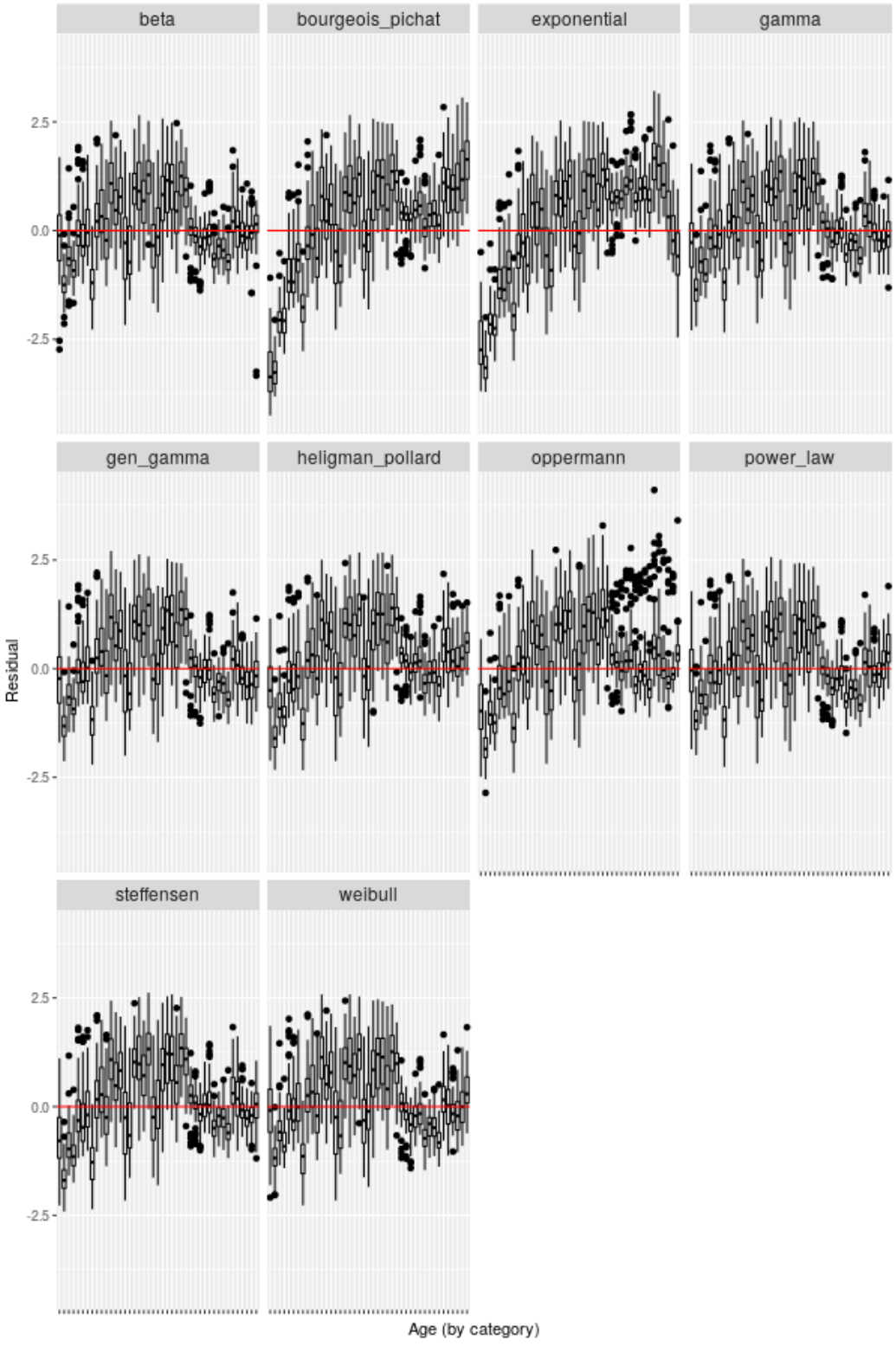

FIGURE 0.7. Boxplots for distribution of age-specific log-errors (log(ratio of predicted to observed proportion of under-5 deaths)), across all cohorts included from MICS, stratified by model. Age (x-axis) is categorical: 0-27 days by day, $1-11$ months by month, $1-4$ years by year.

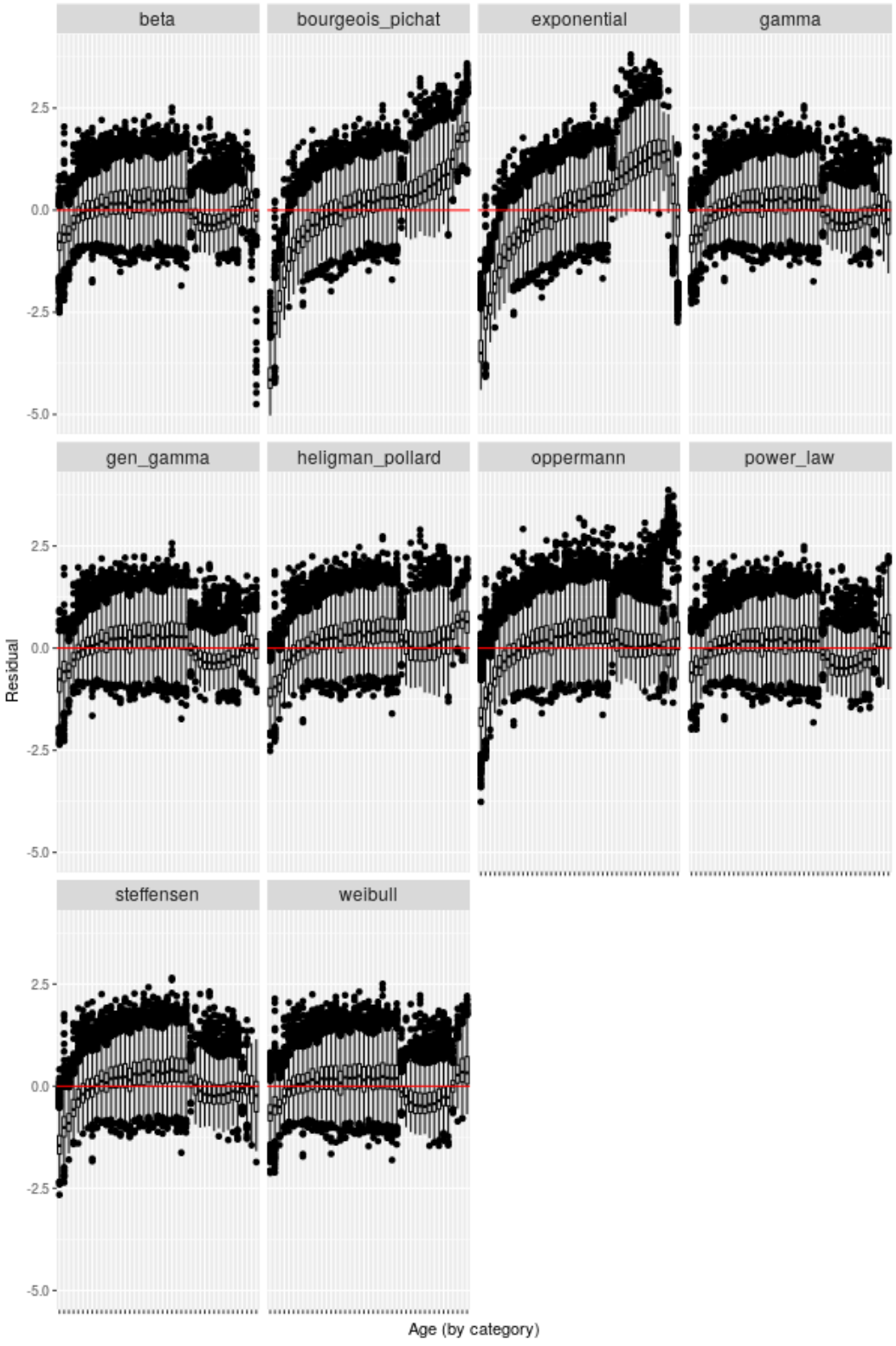

FIGURE 0.8. Boxplots for distribution of age-specific log-errors (log(ratio of predicted to observed proportion of under-5 deaths)), across all cohorts included from Brazil vital registration, stratified by model. Age (x-axis) is categorical: 0-27 days by day, $1-11$ months by month, $1-4$ years by year.

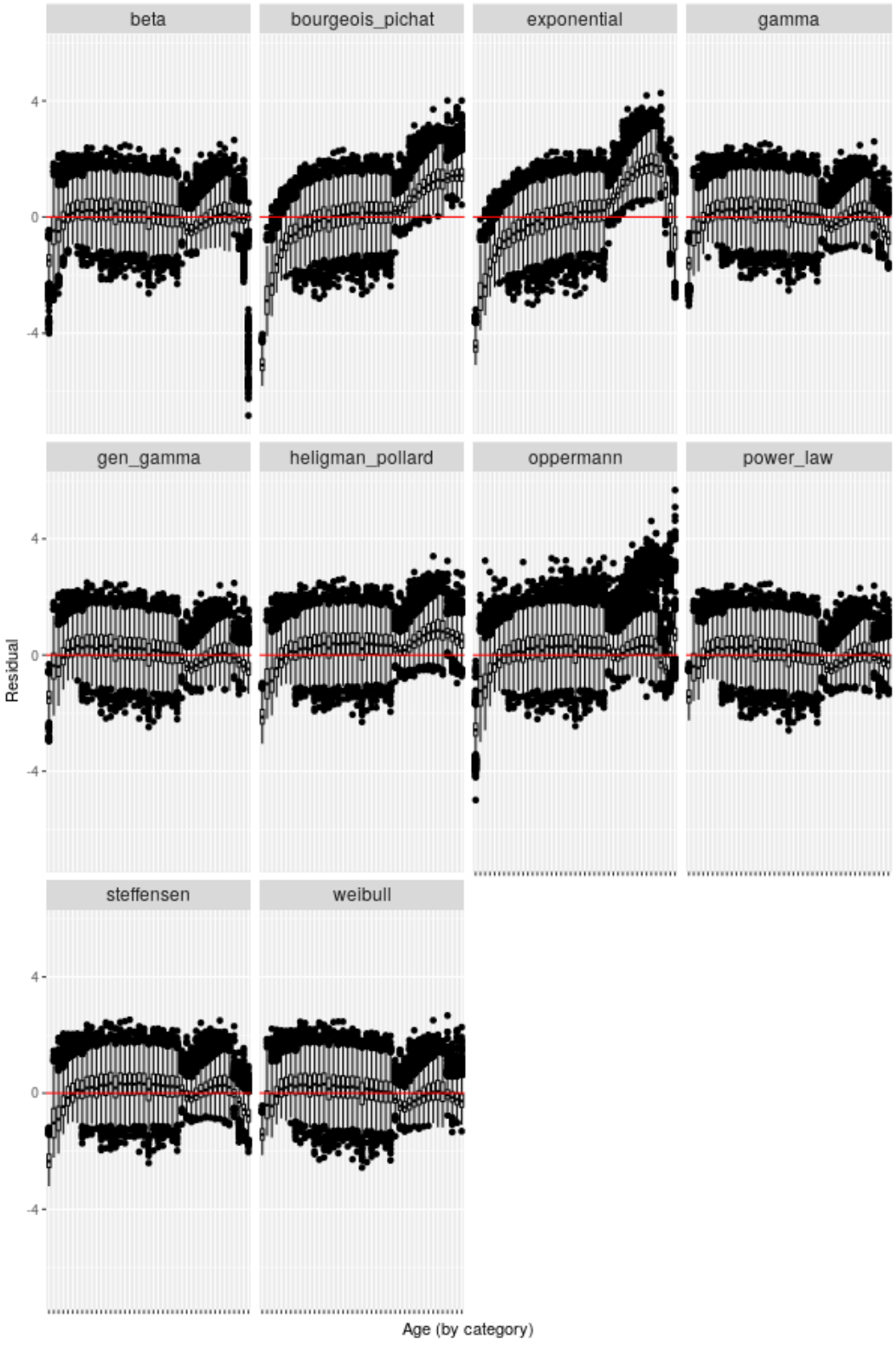

FIGURE 0.9. Boxplots for distribution of age-specific log-errors (log(ratio of predicted to observed proportion of under-5 deaths)), across all cohorts included from USA vital registration, stratified by model. Age (x-axis) is categorical: 0-27 days by day, $1-11$ months by month, $1-4$ years by year.

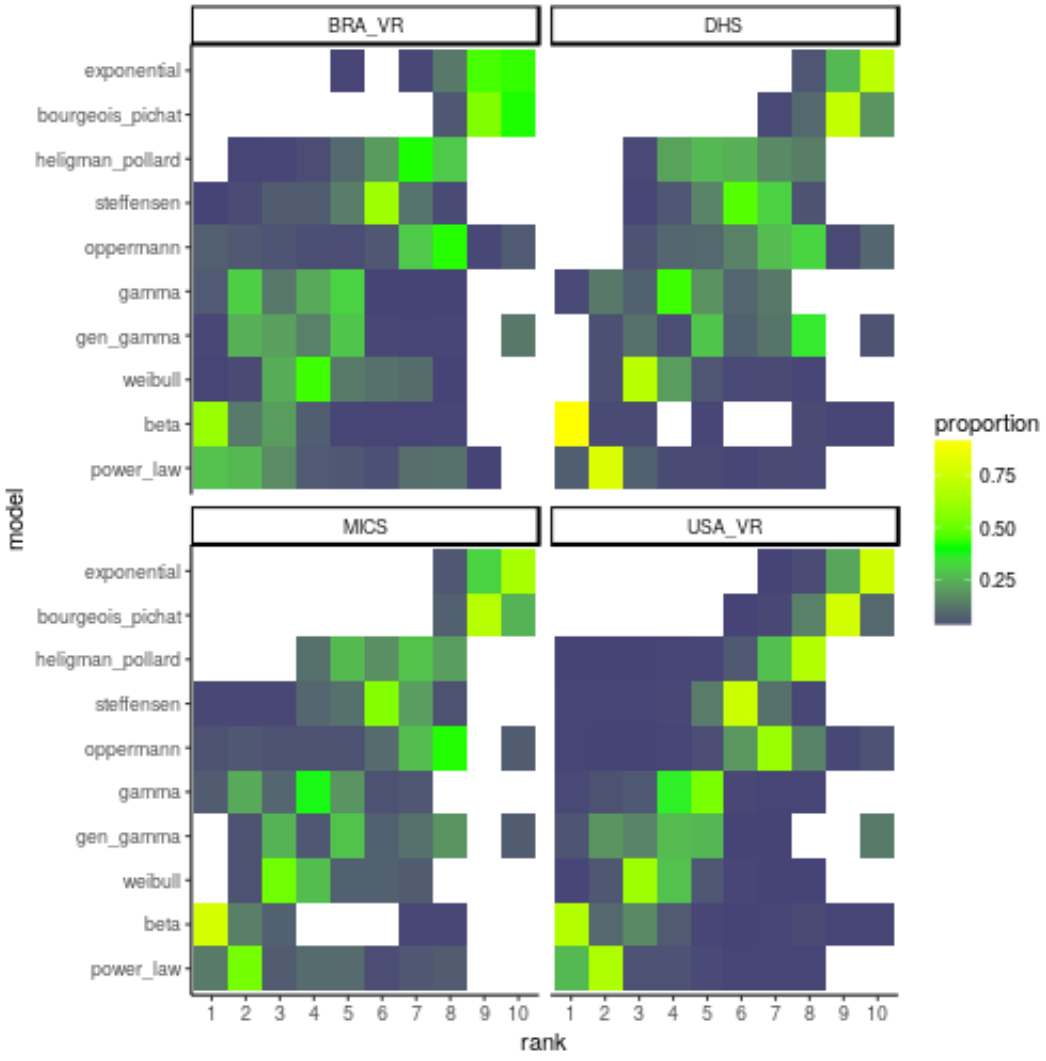

FIGURE 0.10. Heatmap of rankings of models by SSE, stratified by source type.

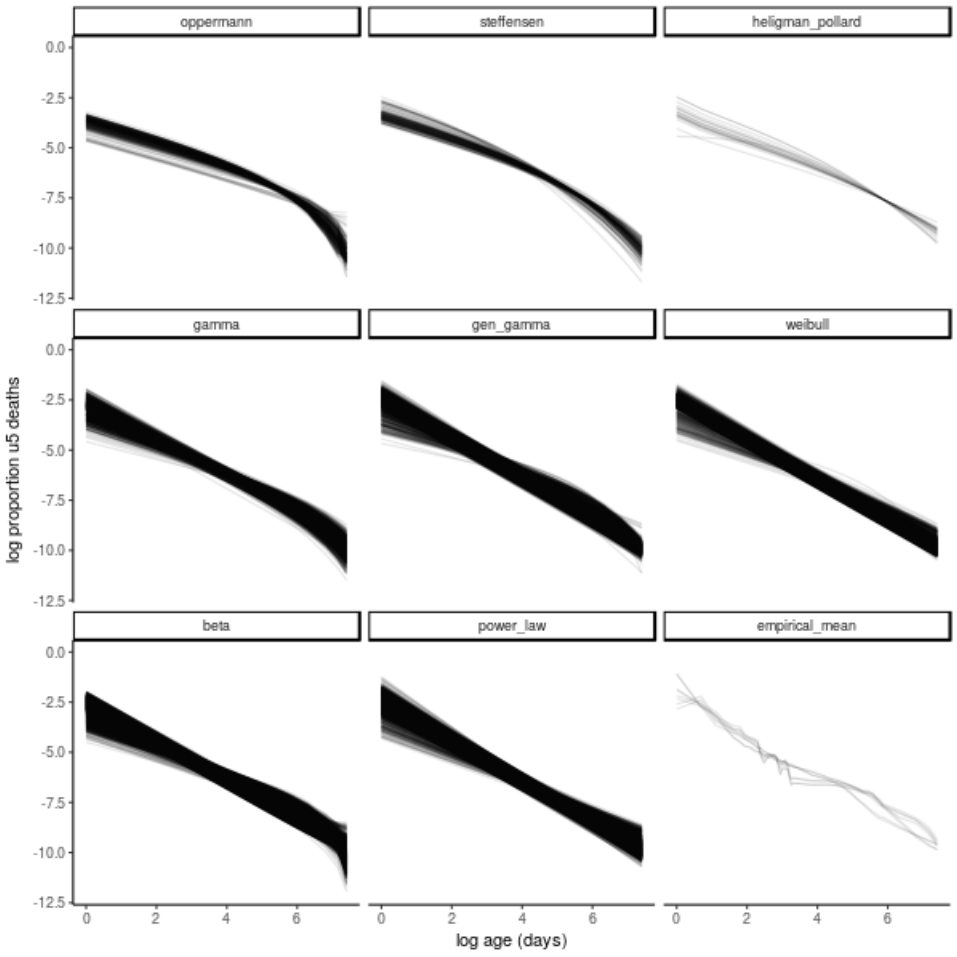

FIGURE 0.11. Models included in model database for relational model. These models ranked in top three by SSE for a model fit in the first part of this analysis. Empirical mean models by sex from DHS, MICS, USA VR, and BRA VR are also included.

Case study

To illustrate the steps of this analysis, I will walk through the example of males in Washington State in 1961 in this chapter.

Input data

A total of 1007 male deaths under the age of 5 years were recorded in Washington State for 1961. The largest share of these, with a total of 377, were recorded to have occurred before the age of one day (Figure 0.12). Between day 10 and day 27, a range of 0-5 deaths per day were observed. Deaths binned by the month after the first month, and by the year after the first year are greater in number-per-bin, because the interval length increases. There is no obvious age heaping in this sample, like we see for CBH. The log-proportion of total deaths, by day, as prepared by the process described in this paper, can be found in Figure 0.13. Note that standard error inflates as per day number of deaths approaches zero.

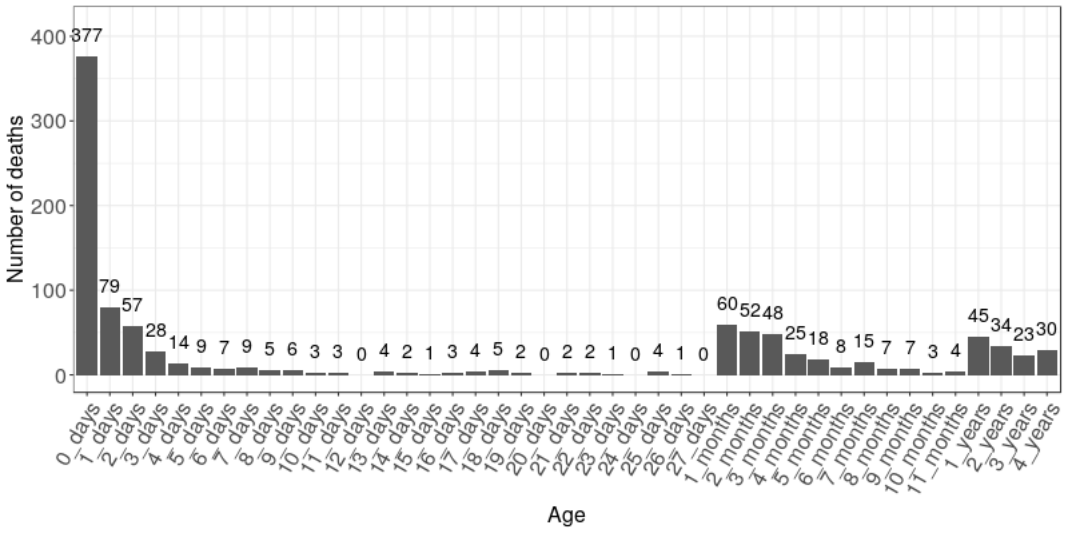

FIGURE 0.12. Number of observed deaths by age category, in Washington State, for males in 1961.

Parametric models

By SSE, beta is the best performing model, followed by power-law, generalized-gamma, Weibull, gamma, Steffensen, Oppermann, and Heligman-Pollard in that order (Figure 0.14). After a large jump, the exponential and Bourgeois-Pichat models are the worst performing for this fit. Most of the models, apart from the Oppermann, exponential, Bourgeois-Pichat, and Heligman-Pollard, take similar shapes. This group also clearly under-estimates mortality in the first few days, at least by the specification and model fitting method used here. From this set, the beta, power-law, and

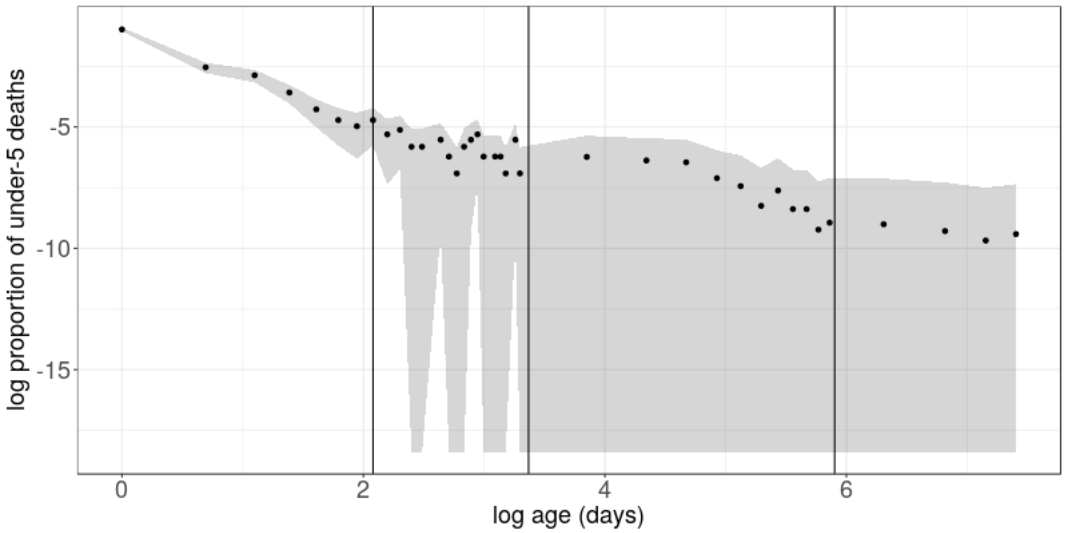

FIGURE 0.13. Log proportion of under-5 deaths by age in log-day, including standard error, for males in Washington State in 1961. Vertical bars mark one week, one month, and one year.

generalized-gamma fits are entered into the model database for the relational model.

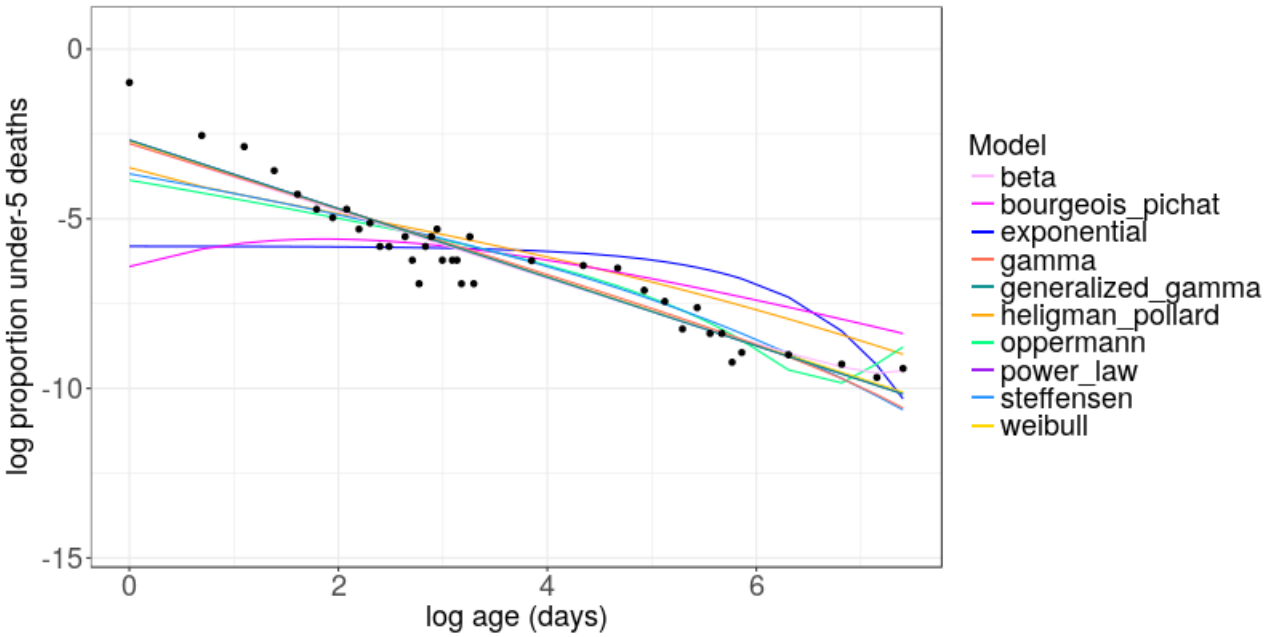

FIGURE 0.14. Fits of all parametric models to data from Washington State males in 1961, relating daily proportion of under-five deaths to age in days.

Relational model

For each of 100 draws of the observed data, the values were collapsed to early, late, post-neonatal, infant, and 1-4 probabilities, and the closest 100 distributions from the model database

were selected as standards. Of these ten-thousand distributions, they were: 5054 power-law, 2066 beta, 1017 Weibull, 750 generalized-gamma, 727 gamma, 327 empirical mean (evenly divided between the source-types), and 59 gamma. Then, the relational model was fit ten-thousand times – once per draw-standard pair – and predictions were generated from these model fits for the complete age series (Figure 0.15). The median and 95 percent confidence interval from these draws resulted in coverage of 56.4 percent (Figure 0.16).

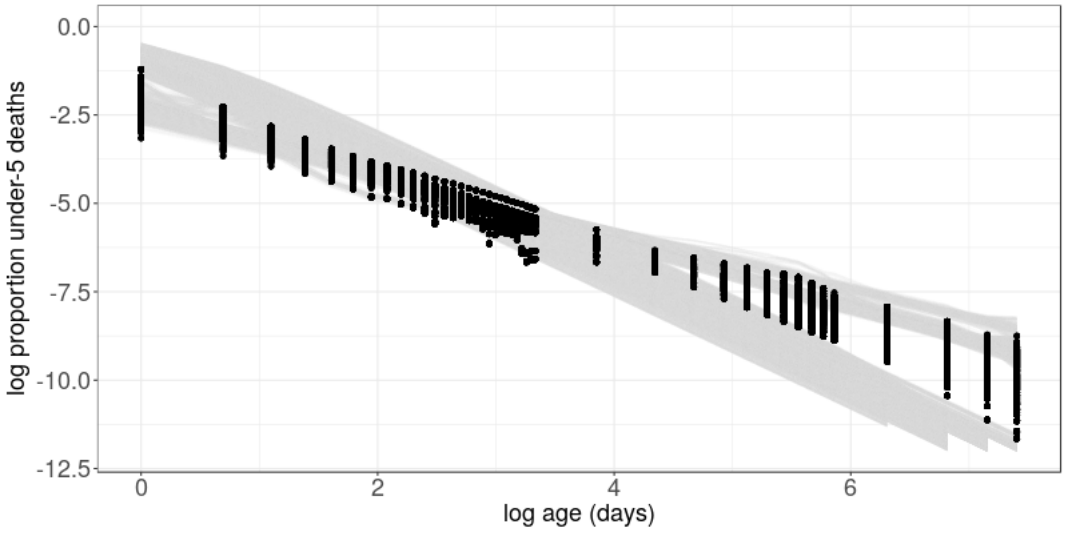

FIGURE 0.15. log-probabilities from model distributions (black), with predicted relational models (grey) for Washington State males, 1961.

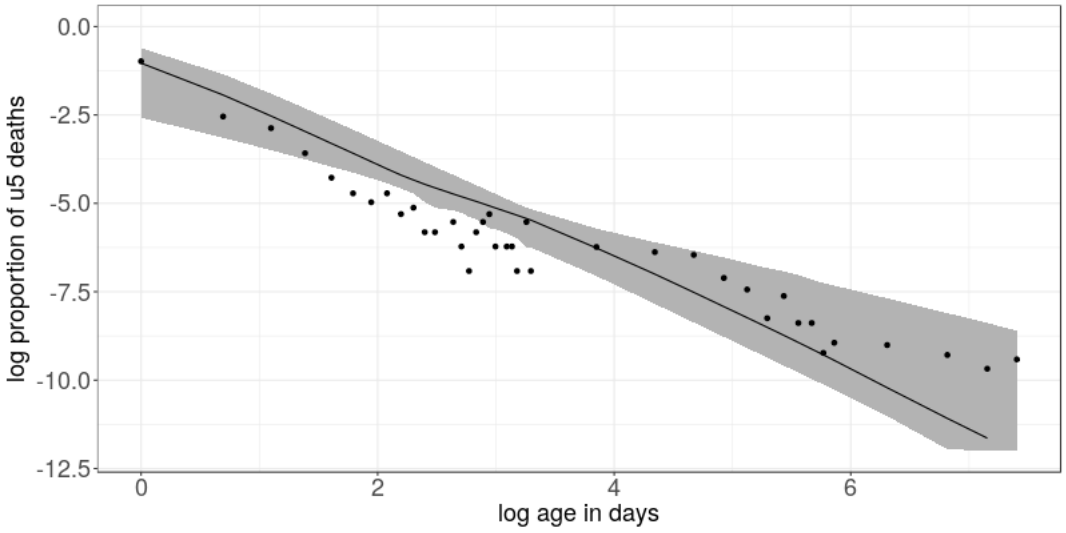

FIGURE 0.16. Observed data, with final mean and uncertainty interval for Washington State males, 1961.

Using the predicted age distribution of mortality, and a value for ${}_5q_0$ in this population, we could calculate estimated day-specific q_x values. First, we could calculate the marginal probability of dying at age 0 days (q_0) as the probability of dying before age 5 years (${}_5q_0$), times the conditional

probability of dying at age 0 days given death before age 5 years (d_0, the outcome estimated by our model). Then, for $x > 0$, we recursively calculate:

$$q_x = \frac{{}_5q_0 \cdot d_x}{\prod_{i=0}^{x-1}(1 - q_i)}$$

where the numerator represents the proportion that dies on day x and the denominator represents the proportion surviving to day x.

Discussion

Together, the parametric and relational models explored in this paper help us to describe the age pattern of mortality where we have detailed data by age, and provide a blueprint for expanding this work to location-years with lower data availability.

The first part of this paper is an extension of the research of previous demographers to find a mortality law that describes the pattern of child mortality. It compares existing models, many of them very old, in the context of modern complete birth history and vital registration data. The results are supportive evidence that the beta, gamma, generalized gamma, power-law, Steffensen, Heligman-Pollard and Weibull distributions are acceptable distributions with which to estimate the age pattern of under-five mortality. While none of these is perfect, and there does appear to be some age-specific bias, they are flexible enough to fit well with many different cohorts. The exponential and Bourgeois-Pichat models failed by comparison, and the Oppermann model did not perform uniformly well. The Bourgeois-Pichat model was designed to capture the change in mortality between the end of the first month and the end of the first year. From the results of this analysis, it is clear that the model cannot be extended to the first month without modification. Future work might still examine alternate parametric functions to suit the functional form of these age patterns.

A complete by-day age series for under-five mortality can be estimated using the relational model proposed here, provided overall under-five mortality and any combination of two of the following: early, late, post-neonatal, infant, or 1-4 mortality. From, there, q_x and m_x can be calculated by age. While the parametric models demonstrated some biases, particularly for the first few days of life, the collection of them, in conjunction with the relational system, produced a promising ensemble-type model. It is also noteworthy that some parametric models performed much better at the age-aggregate level than at the by-day level. This is apparent from the discrepancy between the compilation of top-3 fits per cohort and the set of distributions that were ultimately selected to be standard-distributions based on their SSE assessed at the age-aggregated level.

The use of parametric models as a model database for a relational model is atypical – the standard approach is to use empirical data for a model database. Due to small numbers, misreporting, and poor record keeping, detailed-age data for deaths under five years is noisy and sparse. Therefore, empirical data needs to be aggregated or smoothed to resemble an acceptable age pattern. This paper acts as a proof-of-concept to demonstrate that an ensemble of parametric models may collectively form a model database that adequately captures a range of plausible age distributions. Future work might seek more empirical distributions, particularly from vital registration systems with large sample sizes, to add to the database.

While we have methods to adjust for age heaping, it is still a limitation as any adjustment is an approximation. Estimating by time and space is outside of the scope of this project, and may be the subject of future research. Fitting the models by optimization of SSE in log-space may not be the best method – maximum likelihood can be a more robust option, but requires the derivation

of likelihood functions. Additionally, the `optim` function in `R` is highly dependent on the starting parameter values selected. I iterated through several options for starting values, and used median parameter values from models that fit in a second iteration of model fitting. Where possible, I chose starting values based on parameter estimates reported in the original papers. Further work to assess these parametric functions might implement a multiple-start optimization to avoid this challenge. The aggregated age data available for this paper required us to make assumptions in order to include month-specific or year-specific proportions in models estimating day-specific proportions. Our assumption that the mean proportion occurs at the midpoint of an interval is likely wrong, and sensitivity analyses could be used to assess the extent to which this affects the results. Lastly, this thesis uses period-based data to compose synthetic cohorts: this is a limitation in cases where under-five mortality changes rapidly.

The coverage produced by the relational method leaves room for improvement before implementation would be viable. Addressing some of the limitations, such as weighting the first week more in optimization and giving better performing fits more weight in the ensemble, might increase coverage. The addition of new parametric models, or more empirical distributions, would also likely improve coverage. Additionally, more work could be done to improve data processing, and to examine cases where low coverage is due to data noise as opposed to model bias.

There are many established methods for estimating the age pattern of mortality, but relatively few for estimating the particular age pattern of mortality in children under the age of five. This research is important, because it allows us to understand the gains to date in reducing under-5 mortality, as well as the existing challenges. A full age schedule may be integrated to any age group of interest, and analyzed for context based on which causes may be more or less relevant at a given age. Using a relational model, we might also leverage any full age patterns established, in conjunction with existing summary measures of mortality by discrete under-5 age groups, to get full age patterns where detailed mortality data is not available. In this way, describing the age pattern of child mortality opens up the potential for future research in all-cause and cause-specific mortality for the under-five age group.

Bibliography

[1] Dicker D, Nguyen G, Abate D, Abate KH, Abay SM, Abbafati C, Abbasi N, Abbastabar H, Abd-Allah F, Abdela J, Abdelalim A. Global, regional, and national age-sex-specific mortality and life expectancy, 19502017: a systematic analysis for the Global Burden of Disease Study 2017. The lancet. 2018 Nov 10;392(10159):1684-735.

[2] Lawn JE, Cousens S, Zupan J, Lancet Neonatal Survival Steering Team. 4 million neonatal deaths: when? Where? Why?. The lancet. 2005 Mar 5;365(9462):891-900.

[3] Gompertz B. On the nature of the function expressive of the law of mortality. Philosophical Transactions. 1825. 27: 513-85.

[4] Bourgeois-Pichat, J. 1951. "La Mesure de la Mortalite Infantile. II, Les Causes de Deces," Population, 6(3): 459-80.

[5] Oppermann LHF. On the graduation of life tables, with special application to the rate of mortality in infancy and childhood. The Insurance Record Minutes from a meeting in the Institute of Actuaries. 1870, 42.

[6] Steffensen JF. Infantile mortality from an actuarial point of view. Scandinavian Actuarial Journal. 1930:2, 272-286.

[7] Heligman L, Pollard JH. The age pattern of mortality. Journal of the Institute of Actuaries. 1980 Jan;107(1):49-80.

[8] Brass W. On the scale of mortality. Biological aspects of demography. 1971: 69-110.

[9] Preston S, Heuveline P, Guillot M. Demography: measuring and modeling population processes. 2001. Malden, MA: Blackwell Publishers. 2000.

[10] Sankar MJ, Natarajan CK, Das RR, Agarwal R, Chandrasekaran A, Paul VK. When do newborns die? A systematic review of timing of overall and cause-specific neonatal deaths in developing countries. Journal of Perinatology. 2016 Apr 25;36(S1):S1.

[11] ICF. 2004-2017. Demographic and Health Surveys (various). Funded by USAID. Rockville, Maryland: ICF.

[12] UNICEF. Multiple indicator cluster survey (MICS).

[13] National Center for Health Statistics, Centers for Disease Control and Prevention. United States NVSS Mortality Data. Atlanta, United States: Centers for Disease Control and Prevention (CDC).

[14] Ministry of Health (Brazil). Brazil Mortality Information System - Deaths. Rio de Janeiro, Brazil: Ministry of Health (Brazil).

Appendix: Parametric models

Each parametric model used in this analysis was adapted for the purposes of this application. This appendix describes in detail the transformations that were made.

Heligman-Pollard

The Heligman-Pollard model, as it is reported, is:

$$\frac{{}_1q_x}{{}_1p_x} = \alpha^{(x+\beta)^\gamma} + \delta e^{-\epsilon(\ln(x)-\ln(\zeta))^2} + \eta\theta^x$$

where ${}_1q_x$ is the probability of dying between the age x years and $x+1$ years conditional on survival to age x, and ${}_1p_x$ is the probability of survival, or $1 - {}_1q_x$.

The first term of this model captures child mortality. The parameter α measures the level of mortality, γ measures rate of decline, and β represents age shift to account for infant mortality. Higher values of β mean q_0 is closer to q_1. The original paper claims that β is close to zero, and typically between 0.01 and 0.03 [**7**].

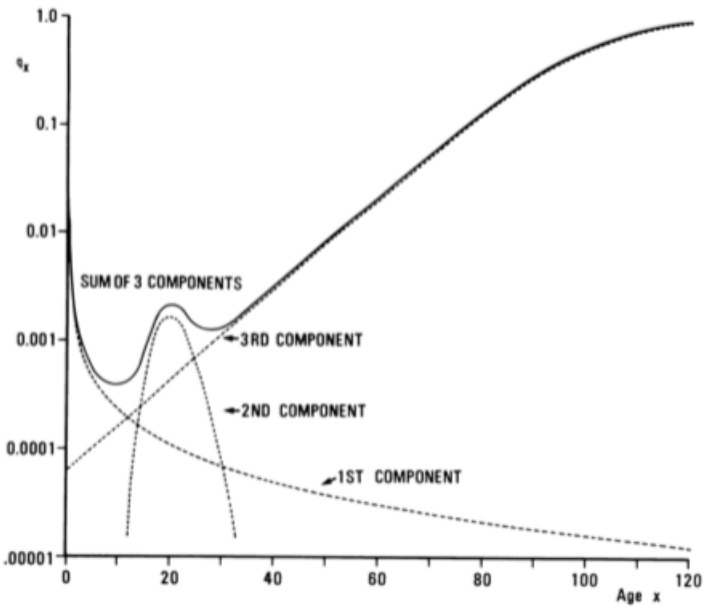

Figure 1. The graduated q_x curve and its three components: Australian national mortality, 1970–72 (males).

FIGURE 0.17. A breakdown of the three components of the Heligman-Pollard model, taken directly from the original paper [**7**].

The second term creates a hump, between approximately the ages 10 and 40 years, to account for the phenomenon of increased rate of injury deaths in young adults, in addition to maternal mortality. The term has parameters ζ for location, ϵ for spread, and δ for severity. It will evaluate to zero before age 5 years when the parameters fall within the intended range (Figure 0.17).

The final term is similar to a Gompertz model for increasing mortality in adults. It does not reduce to zero before age 5, however, it is not a major driver of the age pattern before age five because it is on a different order of magnitude for this age period. Therefore, for simplification, I remove the second term and reduce the third term to a constant intercept, such that

$$\frac{{}_1q_x}{{}_1p_x} \approx \alpha^{(x+\beta)^\gamma} + \delta.$$

When q_x is small, as a daily q_x is, $\frac{{}_1q_x}{{}_1p_x} \approx_1 q_x$. Heligman and Pollard discuss this, and explain that while the intention of using $\frac{{}_1q_x}{{}_1p_x}$ is that it bounds the results between 0 and 1, a q_x-based model is also viable. Given this, I simplify the model by using q_x alone:

$${}_1q_x = \alpha^{(x+\beta)^\gamma} + \delta.$$

Now, this model must be transformed such that it defines the outcome of interest: the probability of a death occuring on day x given that it occurs before 5 years of age.

First, equate the discrete hazard measure of q_x with the continuous measure, $h(x)$. So,

$$h(x) = \alpha^{(x+\beta)^\gamma} + \delta.$$

Next, use the relationship between the hazard function and the density function:

$$f(x) = h(x)e^{-\int_0^x h(u)du}$$

where $h(x)$ is defined above, to get density. Numerical integration in R is used instead of the cumbersome closed-form solution to this integral.

Note that this is the density function for the marginal probability distribution, but we are defining a conditional density assuming death before age five. To account for this, we add a scaling parameter, ϵ:

$$f(x) = \epsilon \cdot h(x)e^{-\int_0^x h(u)du}.$$

Additionally, given that the Heligman-Pollard model is intended for a x-scale on the order of years, while the x-scale in this model is in the units of days, we divided x by 365.

Then, the final parameterization used in this analysis is:

$$f(x) = \epsilon \cdot h(\frac{x}{365})e^{-\int_0^{x/365} h(u)du}$$
$$\text{where } h(x) = \alpha^{(x+\beta)^\gamma} + \delta.$$

Bourgeois-Pichat

The Bourgeois-Pichat model is written as the cumulative density function:

$$F(x) = \alpha + \beta[\log(x+1)]^3.$$

Given the relationship between density and cumulative density, $f(x) = \frac{d}{dx}F(x)$, we get:

$$f(x) = \frac{d}{dx}\Big(\alpha + \beta[\log(x+1)]^3\Big).$$

By differentiating, and allowing the parameter β to absorb multiplicative constants, we get $f(x)$ from the Bourgeois-Pichat model:

$$f(x) = \frac{\beta[\log(x+1)]^2}{x+1}.$$

Oppermann

The Oppermann model is given with a hazard function:

$$h(x) = \frac{\alpha}{\sqrt{x}} + \beta + \gamma\sqrt{x}$$

Given the relationship between density and hazard, we get the density function:

$$f(x) = h(x)e^{-\int_0^x h(u)du}$$

where $h(x)$ is as defined by Oppermann.

As with the Heligman-Pollard model, I divided x by 365 to match the intended x-scale of years.

After integration, I get:

$$f(x) = \Big[\frac{\alpha}{\sqrt{\frac{x}{365}}} + \beta + \gamma\sqrt{\frac{x}{365}}\Big]e^{2\alpha\sqrt{\frac{x}{365}}+\beta\frac{x}{365}+\frac{2}{3}\gamma(\frac{x}{365})^{\frac{3}{2}}}.$$

Steffensen

The Steffensen model is defined with $log(l_x)$ as the dependent variable:

$$\log(l_x) = 10^{\alpha\sqrt{x}+\beta} + \gamma.$$

We can equate the discrete survival measure, l_x, to the continuous survival measure, $S(x)$:

$$\log(S(x)) = 10^{\alpha\sqrt{x}+\beta} + \gamma.$$

Next, use the relationship $f(x) = -\frac{d}{dx}S(x)$ to solve for $f(x)$. So,

$$S(x) = e^{(10^{\alpha\sqrt{x}+\beta}+\gamma)}$$

$$\implies f(x) = -\frac{d}{dx}e^{10^{\alpha\sqrt{x}+\beta}+\gamma}$$

$$\implies f(x) = -\alpha\gamma(2^{\alpha\sqrt{x}+\beta-1})(5^{\alpha}\sqrt{x}+\beta)e^{(10^{\alpha\sqrt{x}+\beta}+\gamma)}(\log(10)/\sqrt{x}).$$

Again, I divided x by 365 to match the intended x-scale of years.

Then, the functional form fit in my analysis was:

$$\implies f(x) = -\alpha\gamma(2^{\alpha\sqrt{\frac{x}{365}}+\beta-1})(5^{\alpha}\sqrt{\frac{x}{365}}+\beta)e^{(10^{\alpha\sqrt{\frac{x}{365}}+\beta}+\gamma)}(\log(10)/\sqrt{\frac{x}{365}}).$$

Other

The exponential, Weibull, beta, gamma, and generalized gamma models were all given additional y-scaling parameters to improve fit. The beta model is also given an x scalar of $\frac{1}{5*365}$, because the beta model is designed to produce non-zero density only for x values from zero to one. The others were given an x scalar of $\frac{1}{365}$. These adjustments were made because I wanted to use the distribution shapes that are created by these models, but allow enough flexibility for those shapes to be scaled up or down to accommodate the domain and range of the data.

Appendix: Outcomes of interest

Several different outcomes of interest are used to evaluate mortality. This appendix is intended as a reference for these outcomes: both their definitions and their notation as used in this paper.

There are two analogous sets of outcomes described here. The first is a continuous set, which comes from survival analysis, and the second is an analogous discrete set that is utilized in life tables. Cumulative versions of the continuous functions are also listed where applicable. Note that these outcomes are all related: mathematical relationships allow us to convert between them.

Survival: The proportion surviving to day x.

- Continuous: $S(x)$
- Discrete: l_x

Hazard: The probability of dying on day x, conditional on survival to day x.

- Continuous: $h(x)$
- Cumulative: $H(x)$
- Discrete: q_x

Density: The proportion of the population that dies on day x.

- Continuous: $f(x)$
- Cumulative: $F(x)$
- Discrete: d_x

Appendix: R Code

Data preparation

```
## Data prep : process input data files into the raw input data for this analysis

## SET-UP ================================================

rm(list=ls())
memory.limit(30000)

library(data.table)
library(rdhs)
library(ggplot2)
library(haven)

root <- FILEPATH

## DHS ====================================================

download <- F # set to true to download data from website
if(download == T){

  # characteristic 31 = "Birth registration"
  surveys <- dhs_surveys(surveyCharacteristicIds = 31)
  datasets <- dhs_datasets(surveyIds = surveys$SurveyId, fileFormat = "flat")
  datasets <- datasets[datasets$FileType == "Births Recode",]

  # set credentials
  set_rdhs_config(email = "krpaul@uw.edu",
                  project = "Estimating age and sex-specific mortality under 5-years",
                  config_path = "~/.rdhs.json")

  # download
  downloads_1 <- get_datasets(datasets$FileName,
                              download_option = "rds",
                              output_dir_root = paste0(root, "/DHS"))
}

# read in
files <- list.files(paste0(root, "/DHS"), full.names = T, pattern = ".rds")
dt <- data.table()
for(f in files){
  print(f)
  temp <- readRDS(f)
  labels <- get_variable_labels(temp)
  tempDT <- data.table(country_code_and_phase = temp$v000,
                       child_alive = temp$b5,
                       child_aod_raw = temp$b6,
                       child_sex = temp$b4)
  tempDT <- tempDT[child_alive == 0]
  dt <- rbind(dt, tempDT, fill=T)
}
```

```
# recode child_aod_raw according to description above
dt[, units := substr(as.character(child_aod_raw), 1, 1)]
dt[units == 1, units := "days"]
dt[units == 2, units := "months"]
dt[units == 3, units := "years"]
dt[units == 9, units := "special"]
dt[, aod := substr(as.character(child_aod_raw), 2, 3)]
dt[, aod := as.numeric(aod)]
write.csv(dt, paste0(root, "/DHS/0_all_DHS.csv"), row.names = F)


## MICS ============================================================

# get list of variables needed
all_vars <- c("hh1", "hh2", "hh6", "hh7", "hh7a", "hh7b", "hn3", "hn10_u",
              "hn10_i", "hhstrat", "wm1", "wm2", "bh3", "bh5", "cm19", "hn6",
              "memid", "cm11f", "bh9u", "bh9a", "cm20u", "bh9n", "bh9b",
              "cm20n", "stratum", "strata", "hhstrat", "ln", "xx1")

files <- list.files(paste0(root, "/MICS/"), recursive = T, pattern = "bh.")
dt <- data.table()
for(ff in files){
  temp <- read_sav(paste0(root, "/MICS/", ff))
  temp <- as.data.table(temp)
  loc_year <- strsplit(ff, 'MICS')[[1]][1]
  loc_year <- gsub("_", "", loc_year)
  loc_year <- gsub(" ", "", loc_year)
  loc_year <- gsub("-", "", loc_year)
  loc_year <- gsub("\\(", "", loc_year)
  loc_year <- gsub("\\)", "", loc_year)
  print(loc_year)
  temp$loc_year <- loc_year
  names(temp) <- tolower(names(temp))
  write.csv(temp, paste0(root, "/MICS/0_all_cbh/", loc_year, ".csv"), row.names = F)
  dt <- rbind(dt, temp, fill = T)
}

# subset to births where child is not still alive
temp <- temp[(bh5 != 1 | is.na(bh5)) &
               (cm11f != 1 | is.na(cm11f)) &
               (hn6 != 1 | is.na(hn6))]

# subset to needed columns
keep_cols <- intersect(names(dt), all_vars)
dt <- dt[, c(keep_cols), with = F]

# save
write.csv(dt, paste0(root, "/MICS/0_all_cbh/0_all_mics_deaths.csv"), row.names = F)


## USA VR ==========================================================================

files <- list.files(paste0(root, "/USA_VR/"), pattern = "mort2")
```

```r
dt <- data.table()
for(ff in files){
  print(ff)
  temp <- read_dta(paste0(root, "/USA_VR/", ff))
  temp <- as.data.table(temp)
  temp$filename <- ff
  year <- as.numeric(substr(ff, 5, 8))
  temp$year <- year
  if("ager52" %in% names(temp)){
    temp <- temp[ager52 <= 27]
  }
  if("ager27" %in% names(temp)){
    temp <- temp[ager27 <= 6]
  }
  write.csv(temp, paste0(root, "/USA_VR/usa_", year, ".csv"), row.names = F)
  dt <- rbind(dt, temp, fill = T)
}
write.csv(dt, paste0(root, "/USA_VR/0_usa_compiled.csv"), row.names = F)


## BRA VR ================================================================================

files <- list.files(paste0(root, "/BRA_VR"),
                    pattern = ".DTA",
                    recursive = T)
dt <- data.table()
for(ff in files){
  print(ff)
  temp <- read_dta(paste0(root, "/BRA_VR/", ff))
  temp <- as.data.table(temp)
  setnames(temp, c("idade","sexo"), c("age","sex"))
  temp$filename <- ff
  temp$source <- "BRA_VR"
  temp <- temp[, age <= 405]  # subset to under 5 years (unit = 4, value = 05)
  dt <- rbind(dt, temp, fill = T)
}
write.csv(dt, paste0(root, "/BRA_VR/0_bra_compiled.csv"), row.names = F)

# end
```

```r
## Data processing :
##     map extracted data to common varnames
##     redistribute age of death to form uniform categories
##     collapse microdata into summaries

## SET-UP ================================================

library(data.table)
library(ggplot2)

main_dir <- FILEPATH

# DHS ================================================
```

```r
dhs <- fread(paste0(main_dir, "/DHS/0_all_DHS.csv"))
dhs <- dhs[, .(country_code_and_phase, units, aod, child_sex)]
dhs[, aod_cat := paste0(aod, "_", units)]
dhs <- dhs[!(units == "years" & aod > 4)]
setnames(dhs, "child_sex", "sex")
dhs[, location := substr(country_code_and_phase, 1, 2)]
dhs[, source := paste0("DHS_", substr(country_code_and_phase, 3, 3), "_", location)]
dhs[, broadsource := "DHS"]
dhs[, country_code_and_phase := NULL]

# MICS ================================================

mics <- fread(paste0(main_dir, "/MICS/0_all_cbh/0_all_mics_deaths.csv"))

# recode sex
setnames(mics, "bh3", "sex")
mics[is.na(sex), sex := hn3]
mics <- mics[!is.na(sex)]
mics[, sex := as.numeric(sex)]

# recode is_alive, subset to dead
mics[, is_alive := bh5]
mics[is.na(is_alive), is_alive := cm11f]
mics[is.na(is_alive), is_alive := hn6]
mics <- mics[is_alive %in% c(0,2)]
mics[, c("bh5", "cm11f", "hn6", "is_alive") := NULL]

# recode units
mics[, units_id := bh9a]
mics[is.na(units_id), units_id := bh9u]
mics[is.na(units_id), units_id := hn10_u]
mics[, c("bh9a", "bh9u", "hn10_u") := NULL]
mics_units <- data.table(units_id = c(1, 2, 3),
                         units = c("days","months","years"))
mics <- merge(mics, mics_units, by = "units_id", all.x = F)
mics[, units_id := NULL]

# recode aod
mics[, aod := bh9n]
mics[is.na(aod), aod := bh9b]
mics[is.na(aod), aod := hn10_i]
mics[, c("bh9n", "bh9b", "hn10_i") := NULL]
mics <- mics[!(is.na(aod) & !is.na(units))]

# recode source
mics[ ,source := tstrsplit(filename, ".csv", keep = 1)]
mics[ ,location := gsub('[[:digit:]]+', '', source)]
mics[ ,broadsource := "MICS"]
mics[ ,filename := NULL]

# recode aod_cat
mics[, aod_cat := paste0(aod, "_", units)]
mics <- mics[!(units == "years" & aod > 4)]
```

```
# special characters
mics[source %like% "Ivoire", source := "CotedIvoire"]

# USA_VR ============================================

usa <- fread(paste0(main_dir, "/USA_VR/0_usa_compiled.csv"))
usa <- usa[,.(staters, sex, age, ager52, ager12,
              ager27, ager22, filename, year)]

# recode age
usa[year < 2003 & floor(age/100) == 0, units := "years"]
usa[year < 2003 & floor(age/100) == 1, units := "years100"]
usa[year < 2003 & floor(age/100) == 2, units := "months"]
usa[year < 2003 & floor(age/100) == 3, units := "weeks"]
usa[year < 2003 & floor(age/100) == 4, units := "days"]
usa[year < 2003 & floor(age/100) == 5, units := "hours"]
usa[year < 2003 & floor(age/100) == 6, units := "minutes"]
usa[year < 2003 & floor(age/100) == 9, units := "unknown"]
usa[year < 2003, aod := age - floor(age/100)*100]

# recode age, year 2003-2004
usa[year >= 2003 & floor(age/1000) == 1, units := "years"]
usa[year >= 2003 & floor(age/1000) == 2, units := "months"]
usa[year >= 2003 & floor(age/1000) == 3, units := "weeks"]
usa[year >= 2003 & floor(age/1000) == 4, units := "days"]
usa[year >= 2003 & floor(age/1000) == 5, units := "hours"]
usa[year >= 2003 & floor(age/1000) == 6, units := "minutes"]
usa[year >= 2003 & floor(age/1000) == 9, units := "unknown"]
usa[year >= 2003, aod := age - floor(age/1000)*1000]

# deal with unknowns based on recode codes
recodes <- fread(paste0(main_dir, "/USA_VR/age_recode.csv"))
for(set in c(12, 52, 22, 27)){
  setnames(recodes, "id", paste0("ager", set))
  usa <- merge(usa,
               recodes[,c(paste0("ager", set),
                          paste0("aod_recode", set),
                          paste0("unit_recode", set)), with = F],
               by = paste0("ager", set),
               all.x = T)
  setnames(recodes, paste0("ager", set), "id")
}
usa[units != "years", `:=` (aodr = aod_recode22, unitsr = unit_recode22)]
usa[units == "years", `:=` (aodr = aod_recode27, unitsr = unit_recode27)]
usa[is.na(aodr), `:=` (aodr = aod_recode52, unitsr = unit_recode52)]
usa[is.na(aodr), `:=` (aodr = aod_recode12, unitsr = unit_recode12)]
usa[, aod := as.character(aod)]
usa[aod %in% c(99, 999), `:=` (aod = aodr, units = unitsr)]
usa[,c("ager27","ager22","ager52","ager12","aod_recode12","aod_recode52",
       "aod_recode22","aod_recode27","unit_recode27","unit_recode52",
       "unit_recode12","unit_recode22","aodr","unitsr","age") := NULL]

# consolidate minutes/hours to days
```

```
usa[units %in% c("minutes", "hours"), `:=` (aod = "0", units = "days")]

# recode aod_cat
usa[, aod_cat := paste0(aod, "_", units)]
usa <- usa[!(units == "years" & aod > 4)]

# recode source
usa[, broadsource := "USA_VR"]
usa[, location := paste0("usa_", staters)]
usa[, source:= paste0(location,"_",year,"_VR")]
usa[, c("filename","staters") := NULL]

# recode sex
usa[sex == "M", sex := "2"]
usa[sex == "F", sex := "1"]
usa[, sex := as.numeric(sex)]

# BRA_VR ===============================================

bra <- fread(paste0(main_dir, "/BRA_VR/0_bra_compiled.csv"))

# recode age
setnames(bra, "aod", "aod_raw")
bra[, age_code := as.numeric(age_code)]
bra[floor(age_code/100) == 0, units := "minutes"]
bra[floor(age_code/100) == 1, units := "hours"]
bra[floor(age_code/100) == 2, units := "days"]
bra[floor(age_code/100) == 3, units := "months"]
bra[floor(age_code/100) == 4, units := "years"]
bra[, aod := as.numeric(substr(as.character(age_code), 2, 3))]
bra <- bra[!is.na(aod)]
bra[units == "hours" & aod <24, `:=` (units = "days", aod = 0)]
bra[units == "minutes" & aod < 60*24, `:=` (units = "days", aod = 0)]
bra <- bra[!(units=="years" & aod > 4)]
bra[, aod_cat := paste0(aod, "_", units)]
bra[,c("age_code","dob","dod","aod_raw") := NULL]

# recode source
bra[, year := tstrsplit(filename, "/", keep = 1)]
bra <- bra[!year %like% "BRA_SIM"]
bra[, year := as.numeric(year)]
bra[, location := tstrsplit(filename, "_", keep = 6)]
bra[, location := paste0("BRA_", location)]
bra[, filename := NULL]
bra[, source := paste0(source, "_", year, "_", location)]
bra[, broadsource := "BRA_VR"]

## combine and collapse by cohort-sex ======================

dt <- rbindlist(list(dhs, mics, usa, bra), fill = T)
write.csv(dt, "master_combined.csv", row.names = F)

# fix sex variable
```

```
dt[is.na(sex), sex := "3"]
dt[, sex := as.numeric(sex)]
dt <- dt[sex %in% c(1,2)]

# switch 7to13_days to 1 week
dt[aod_cat == "7to13_days", `:=` (aod = "1",
                                  units = "weeks",
                                  aod_cat = "1_weeks")]

# remove unknown age categories
dt[, aod := as.numeric(aod)]
dt <- dt[!(aod_cat %like% "special")]
dt <- dt[!(aod > 90 & units %in% c("weeks","months", "years"))]

# calculate summary measures
dt[, num_deaths := .N, by = c("aod_cat", "sex", "source", "location")]
dt[, total_u5_deaths := .N, by = c("sex", "source", "location")]
dt[, prop_deaths := num_deaths/total_u5_deaths]

# keep only cohorts with sample size of u5 deaths of 200+
under200 <- dt[total_u5_deaths < 200]
under200 <- under200[, c("source", "sex")]
under200 <- unique(under200)
print(nrow(under200))
nrow(under200[source %like% "BRA"])
nrow(under200[source %like% "usa"])
nrow(under200[source %like% "DHS"])
dt <- dt[total_u5_deaths >= 200]
test <- dt[, c("source", "sex")]
test <- unique(test)
print(nrow(test))

# collapse
dt_collapsed <- unique(dt, by = c("aod_cat", "sex", "source", "location"))

# modify age factor
aod_table <- expand.grid(aod = 0:99, units = c("days","weeks","months","years"))
aod_table <- as.data.table(aod_table)
aod_table[, aod_cat := paste0(aod, "_", units)]
aod_categories <- aod_table$aod_cat
dt_collapsed$aod_cat <- factor(dt_collapsed$aod_cat, levels = aod_categories)

# drop unknowns
dt_collapsed <- dt_collapsed[aod < 80]

# calculate prop in log-space
dt_collapsed[, log_prop_deaths := log(prop_deaths)]

## plot data  ===============================================

# recode sex for plot
dt_collapsed[sex == 1, sex_name := "male"]
dt_collapsed[sex == 2, sex_name := "female"]
```

```r
# save dt_collapsed
write.csv(dt_collapsed, paste0(main_dir, "/raw_data_before_redistribution.csv"),
          row.names = F)

# boxplots to summarize
for(bb in unique(dt_collapsed$broadsource)){
  png(paste0(main_dir,"/raw", bb, ".png"),
      width = 1000, height = 400)
  gg <- ggplot(data = dt_collapsed[broadsource==bb],
               aes(x = aod_cat,
                   y = log_prop_deaths,
                   color = sex_name)) +
    geom_boxplot() +
    xlab("Age (days)") +
    ylab("log proportion of under-5 deaths") +
    labs(color = "Sex") +
    theme_bw() +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    ggtitle(bb)
  print(gg)
  dev.off()
}

# recode age to days
dt_collapsed[units == "weeks", prop_deaths := prop_deaths/7]
dt_collapsed[units == "months", prop_deaths := prop_deaths/30.4]
dt_collapsed[units == "years", prop_deaths := prop_deaths/365]

## redistribution ============================================================

dt2 <- copy(dt)

# recode all deaths to:
# 0-27 days
# 1-11 months
# 1-4 years

dt2[units == "days" & aod %in% c(28, 29, 30),
    `:=` (units = "months", aod = 1)]
dt2[units == "days" & aod > 27, `:=` (units = "months",
                                      aod = floor(aod/30.4))]
dt2[units == "months" & aod > 11, `:=` (units = "years",
                                        aod = floor(aod/12))]

# re-create aod_cat
dt2[, aod_cat := paste0(aod, "_", units)]

# calculate summary measures
dt2[, num_deaths := .N, by = c("aod_cat", "sex", "source", "location")]
dt2[, total_u5_deaths := .N, by = c("sex", "source", "location")]
dt2[, prop_deaths := num_deaths/total_u5_deaths]

# keep only cohorts with sample size of u5 deaths of 200+
```

```
dt2 <- dt2[total_u5_deaths >= 200]

# collapse
dt_collapsed <- unique(dt2, by = c("aod_cat", "sex", "source", "location"))

# redistribute weeks
weeks <- dt_collapsed[units == "weeks"]
weeks <- weeks[,.(aod, sex, source, year, location, num_deaths)]
weeks <- unique(weeks)
setnames(weeks, c("aod", "num_deaths"), c("week_num", "week_redist"))
dt_collapsed[units == "days", week_num := floor(aod/7)]
dt_collapsed <- merge(dt_collapsed, weeks,
            by=c("sex", "source", "year", "location", "week_num"), all.x = T)
dt_collapsed[, total_in_week := sum(num_deaths),
             by = c("sex","source","year","location","week_num")]
dt_collapsed[, num_deaths := as.double(num_deaths)]
dt_collapsed[units == "days" & !is.na(week_redist),
             num_deaths := num_deaths + week_redist * (num_deaths/total_in_week),
             by = c("sex", "source", "year", "location", "week_num")]
dt_collapsed <- dt_collapsed[units != "weeks"]
dt_collapsed[, total_u5_deaths := as.double(total_u5_deaths)]
dt_collapsed[, total_u5_deaths := sum(num_deaths),
          by = c("sex", "source", "location")]
dt_collapsed[, prop_deaths := num_deaths/total_u5_deaths]

# scale zeros
zeros <- dt_collapsed[units != "days" & aod == 0]
dt_collapsed <- dt_collapsed[!(units != "days" & aod == 0)]
zeros[, prop_zeros := prop_deaths]
zeros <- zeros[, c("units", "sex", "source", "prop_zeros")]
zero_months <- zeros[units == "months"]
zero_years <- zeros[units == "years"]
setnames(zero_months, "prop_zeros", "prop_zero_months")
setnames(zero_years, "prop_zeros", "prop_zero_years")
zero_months[, units := NULL]
zero_years[, units := NULL]
dt_collapsed <- merge(dt_collapsed, zero_months, by = c("sex", "source"), all.x = T)
dt_collapsed <- merge(dt_collapsed, zero_years, by = c("sex", "source"), all.x = T)
dt_collapsed[, sum_nonzero := sum(prop_deaths), by = c("units", "sex", "source")]
dt_collapsed[!is.na(prop_zero_months) & units == "days",
                prop_deaths := prop_deaths *
               (sum_nonzero + prop_zero_months) / sum_nonzero]
dt_collapsed[!is.na(prop_zero_years) & units == "months",
                prop_deaths := prop_deaths *
               (sum_nonzero + prop_zero_years) / sum_nonzero]

# divide by interval length and set to mid-point
dt_collapsed[units == "days", aod_days := aod]
dt_collapsed[units == "months", `:=` (aod_days = round(aod * 30.4 + 15.2, 0),
                                      prop_deaths = prop_deaths/30.4)]
dt_collapsed[units == "years", `:=` (aod_days = round(aod * 365 + (365/2), 0),
                                     prop_deaths = prop_deaths/365)]
dt_collapsed[, log_prop_deaths := log(prop_deaths)]
```

```
dt_collapsed$aod_cat <- factor(dt_collapsed$aod_cat, levels = aod_categories)

# calculate SE assuming binomial distribution
dt_collapsed[, standard_error := sqrt(prop_deaths *
                                (1 - prop_deaths) / total_u5_deaths)]
dt_collapsed[, prop_deaths_lower := prop_deaths - 1.96 * standard_error]
dt_collapsed[, prop_deaths_upper := prop_deaths + 1.96 * standard_error]
dt_collapsed[prop_deaths_lower <  10^(-7), prop_deaths_lower := 10^(-7)]
dt_collapsed[prop_deaths_upper > 1, prop_deaths_upper := 1]

# shift age to age-at-next birthday, because we can't have age 0 when
#   we model in log-space
dt_collapsed[, aod_days := aod_days + 1]

# save for later
write.csv(dt_collapsed, paste0(main_dir, "/processed_data_age_days.csv"),
          row.names = F)

## replot ======================================

# boxplots for summary
pdf(paste0(main_dir,"/redist_data_by_source_boxplot.pdf"),
    width = 15, height = 8)
for(bb in unique(dt_collapsed$broadsource)){
  gg <- ggplot(data = dt_collapsed[broadsource==bb],
               aes(x = aod_cat,
                   y = log_prop_deaths,
                   color = as.factor(sex))) +
    geom_boxplot() +
    xlab("age (days)") +
    ylab("log proportion of under-5 deaths") +
    labs(color = "sex") +
    theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
    ggtitle(bb)
  print(gg)
}
dev.off()

# END
```

Fit parametric models

```
## Fit parametric models :
##     Run parallel by sex and source
##     Fit all parametric models and save parameters and predictions

## SET-UP ======================================================

library(data.table)
library(ggplot2)
library(stats)
library(rmutil, lib.loc = "FILEPATH/packages")
```

```
library(optimx, lib.loc = "FILEPATH/packages")
library(startupmsg, lib.loc = "FILEPATH/packages")
library(sfsmisc, lib.loc = "FILEPATH/packages")
library(SweaveListingUtils, lib.loc = "FILEPATH/packages")
library(distr, lib.loc = "FILEPATH/packages")
library(distrEx, lib.loc = "FILEPATH/packages")
library(argparse)

# get arguments
parser <- ArgumentParser()
parser$add_argument('--source', type="character", required=TRUE,
                    help='The source of interest')
parser$add_argument('--sex', type="integer", required=TRUE,
                    help='The sex of interest')
args <- parser$parse_args()
sex_id <- args$sex
source_name <- args$source

# get prepped data
dt <- fread("FILEPATH/processed_data_age_days.csv")

# subset
temp <- dt[sex==sex_id & source == source_name & !is.na(prop_deaths)]
temp[, aod_days := as.numeric(aod_days)]

# create parameter data.table
param_table <- data.table(model = NA, parameter = NA, value = NA)


# exponential function ================================
exponential <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  pred <- b*a*exp(-1*a*(1/365)*aod_days)
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days,
                             exponential,
                             params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=1, b=0.003),
             fn = sse,
             data = temp,
             method = "L-BFGS-B",
             lower = c(0.1, 0.0000001),
             upper = c(2, 1),
             control = list(maxit=10000))
# save parameters
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]))
```

```
temp$pred_exponential <- unlist(lapply(temp$aod_days,
                                       exponential,
                                       params = params))
# save parameters
param_table_temp <- data.table(model = rep("exponential", 2),
                               parameter = c("a", "b"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)

# power law function ===================================

fit <- lm(data = temp, log(prop_deaths) ~ log(aod_days))
params <- c(coef(fit)[[1]], coef(fit)[[2]])
temp[, pred_power_law := params[1] + params[2]*log(aod_days)]
temp[, pred_power_law := exp(pred_power_law)]
# save parameters
param_table_temp <- data.table(model = rep("power_law", 2),
                               parameter = c("a", "b"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# heligman pollard =====================================
heligman_pollard <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  d <- params[4]
  haz <- function(x) return(a^(x+b)^c + d)
  pred <- haz(aod_days/365)*exp(-1*distrExIntegrate(haz, lower = 0,
                                        upper = aod_days/365))
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days,
                             heligman_pollard,
                             params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
if(!(0 %in% temp$aod_days)) temp$aod_days <- temp$aod_days - 1
fit <- optim(par = c(a=0.002, b=0.02, c=0.12, d=0.0001),
             fn = sse,
             data = temp,
             method = "L-BFGS-B",
             lower = c(0.0005, 0.001, 0, 0.000001),
             upper = c(0.01, 0.04, 0.2, 0.01),
             control = list(maxit=100000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]), as.numeric(fit$par[[4]]))
temp$pred_heligman_pollard <- unlist(lapply(temp$aod_days,
```

```
                                          heligman_pollard,
                                          params = params))
if(0 %in% temp$aod_days) temp$aod_days <- temp$aod_days + 1
# save parameters
param_table_temp <- data.table(model = rep("heligman_pollard", 4),
                               parameter = c("a", "b", "c", "d"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# Bourgeois-Pichat ====================================
bourgeois_pichat <- function(params, aod_days){
  a <- params[1]
  pred <- a*log(aod_days+1)^2 / (aod_days+1)
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days,
                             bourgeois_pichat,
                             params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optimize(sse, interval = c(0, 10), data = temp)

# predict
params <- fit$minimum
temp$pred_bourgeois_pichat <- unlist(lapply(temp$aod_days,
                                          bourgeois_pichat,
                                          params = params))
# save parameters
param_table_temp <- data.table(model = rep("bourgeois_pichat", 1),
                               parameter = c("a"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# oppermann ===========================================
oppermann <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  d <- 1/365
  pred <- (a/((d*aod_days)^0.5) + b + c*((d*aod_days)^0.5)) *
          exp(-1*(2*a*(d*aod_days)^0.5 + b*d*aod_days +
                    (2/3)*c*(d*aod_days)^(3/2)))
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days,
                             oppermann,
                             params = params))
```

```
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=0.035, b=-0.01, c=0.0016),
             fn = sse,
             data = temp,
             control = list(maxit=100000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]))
temp$pred_oppermann <- unlist(lapply(temp$aod_days,
                                     oppermann,
                                     params = params))
# save parameters
param_table_temp <- data.table(model = rep("oppermann"),
                               parameter = c("a", "b", "c"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# steffensen ========================================
steffensen <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  pred <- -(2^(a*(aod_days/365)^0.5 + b - 1))*
            (5^(a*(aod_days/365)^0.5 + b)) * a *
            exp(10^(a*(aod_days/365)^0.5+b)+c) *
            log(10) / (aod_days/365)^0.5
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days,
                             steffensen,
                             params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=-0.0005, b=1, c=-7),
             fn = sse,
             data = temp,
             control = list(maxit=10000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]))
temp$pred_steffensen <- unlist(lapply(temp$aod_days,
                                      steffensen,
                                      params = params))
# save parameters
param_table_temp <- data.table(model = rep("steffensen", 3),
                               parameter = c("a", "b", "c"),
```

```
                                value = params)
param_table <- rbind(param_table, param_table_temp)

# weibull ======================================
weibull <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  pred <- c*dweibull(x = aod_days/365, shape = a, scale = b)
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days, weibull, params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=0.03, b=10, c=0.02),
             fn = sse,
             data = temp,
             method = "L-BFGS-B",
             upper = c(1, 50, 10),
             lower = c(0.0001, 0.0001, 0.0001),
             control = list(maxit=10000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]))
temp$pred_weibull <- unlist(lapply(temp$aod_days, weibull, params = params))
# save parameters
param_table_temp <- data.table(model = rep("weibull", 3),
                               parameter = c("a", "b", "c"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)

# beta =========================================
beta <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  pred <- c*dbeta(x = aod_days/(365*5), shape1 = a, shape2 = b)
  return(pred)
}
sse <- function(params, data){
  data$pred <- unlist(lapply(data$aod_days, beta, params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=1, b=3, c=1),
             fn = sse,
             data = temp,
             control = list(maxit=10000))
# predict
```

```
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]))
temp$pred_beta <- unlist(lapply(temp$aod_days, beta, params = params))
# save parameters
param_table_temp <- data.table(model = rep("beta", 3),
                               parameter = c("a", "b", "c"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# gamma ======================================
gamma <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  pred <- c*dgamma(x = aod_days/365, shape = a, scale = b)
  return(pred)
}
sse <- function(params, data){
  data[, pred := gamma(params, aod_days)]
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=0.4, b=4, c=0.004),
             fn = sse,
             data = temp,
             method = "L-BFGS-B",
             upper = c(1, 1000, 6),
             lower = c(0.000000001, 0.0001, 0.0001),
             control = list(maxit=10000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]))
temp$pred_gamma <- unlist(lapply(temp$aod_days, gamma, params = params))
# save parameters
param_table_temp <- data.table(model = rep("gamma", 3),
                               parameter = c("a", "b", "c"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)


# gen gamma ==================================

gen_gamma <- function(params, aod_days){
  a <- params[1]
  b <- params[2]
  c <- params[3]
  d <- params[4]
  pred <- d*dggamma(aod_days/365, a, b, c)
  return(pred)
}
sse <- function(params, data){
```

```
  data$pred <- unlist(lapply(data$aod_days, gen_gamma, params = params))
  data[, sq_error := (log(prop_deaths/pred))^2]
  return(sum(data$sq_error))
}
# numerical optimization of sse function
fit <- optim(par = c(a=0.9, b=1, c= 0.9, d=0.5),
             fn = sse,
             data = temp,
             method = "L-BFGS-B",
             lower = c(0.000001, 0.000001, 0.000001, 0.000001),
             control = list(maxit=10000))
# predict
params <- c(as.numeric(fit$par[[1]]), as.numeric(fit$par[[2]]),
            as.numeric(fit$par[[3]]), as.numeric(fit$par[[4]]))
temp$pred_gen_gamma <- unlist(lapply(temp$aod_days, gen_gamma, params = params))
# save parameters
param_table_temp <- data.table(model = rep("gen_gamma", 4),
                               parameter = c("a", "b", "c", "d"),
                               value = params)
param_table <- rbind(param_table, param_table_temp)

# save ============================================

write.csv(temp, paste0("FILEPATH/", sex_id, "_", source_name,
                       "_predictions.csv"), row.names = F)

# save parameters ==================================

param_table <- param_table[!is.na(model)]
param_table$sex <- sex_id
param_table$source <- source_name
write.csv(param_table, paste0("FILEPATH/", sex_id, "_", source_name,
                              "_parameters.csv"), row.names = F)

# END
```

Compile parametric fits

```
## Compile parametric models
##    Calculate SSE
##    Rank model performance
##    Generate plots for paper
##    Save model database for relational models

library(data.table)

## get inputs ======================================================

input <- fread("FILEPATH/processed_data_age_days.csv")

pred_files <- list.files("FILEPATH", pattern = "predictions", full.names = T)
param_files <- list.files("FILEPATH", pattern = "parameters", full.names = T)
```

```
preds <- data.table()
for(ff in pred_files){
  temp <- fread(ff)
  preds <- rbind(preds, temp, fill = T)
}

params <- data.table()
for(ff in param_files){
  temp <- fread(ff)
  params <- rbind(params, temp, fill = T)
}

preds[, source_sex := paste0(source, "_", sex)]
params[, source_sex := paste0(source, "_", sex)]
input[, source_sex := paste0(source, "_", sex)]

## plot residuals over age ================================================

# reshape long on model
preds <- melt(preds,
              measure.vars = c("pred_exponential", "pred_power_law",
                  "pred_heligman_pollard", "pred_bourgeois_pichat",
                  "pred_oppermann", "pred_steffensen", "pred_beta",
                  "pred_gamma", "pred_gen_gamma", "pred_weibull"))
preds[, c("pred", "sq_error") := NULL]
setnames(preds, "value", "pred")
setnames(preds, "variable", "model")
preds[, model := gsub("pred_", "", model)]
preds[, pred := as.numeric(pred)]

# calculate residual
preds[, resid := log(pred/prop_deaths)]
preds[, mean_resid := mean(resid, na.ram=T),
      by=c("sex","aod_days","model","broadsource")]
dt_resids <- unique(preds, by=c("sex","aod_days","model","broadsource"))

# plot
png("/FILEPATH/resid_by_age_all.png", height=900, width=600)
  gg_resid_age <- ggplot(data=preds,
                         aes(x=as.factor(aod_days))) +
    geom_boxplot(aes(y=resid), color="black") +
    geom_abline(slope=0, intercept=0, color="red") +
    facet_wrap("model") +
    scale_x_discrete(name="Age (by category)") +
    theme(strip.text.x = element_text(size = 12),
          axis.text.x = element_blank()) +
    ylab("Residual")
  print(gg_resid_age)
dev.off()

## rank models ============================================================

# calculate SSE
```

```
preds[, sse_pred := sum(resid^2), by = c("sex", "model", "source")]
ranks <- preds[,c("sex","model","broadsource","source","sse_pred")]
ranks <- unique(ranks)

# boxplots for SSE by model
png(paste0("FILEPATH/sse.png"), height=400, width=500)
gg_sse <- ggplot(data=ranks) +
  geom_boxplot(aes(x = model, y = sse_pred, color = broadsource)) +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 30, hjust = 1)) +
  ylab("SSE") + labs(color = "source type")
print(gg_sse)
dev.off()

# find median SSE by model & broadsource
ranks[, median_sse := median(sse_pred, na.rm = T),
      by = c("model", "broadsource")]
median_sse_tab <- unique(ranks, by = c("model", "broadsource"))
median_sse_tab <- median_sse_tab[order(broadsource, median_sse)]

# calculate rank
ranks <- ranks[order(broadsource, source, sex, sse_pred)]
ranks[, rank := sequence(.N), by = c("source", "sex")]
ranks[, source_sex := paste0(source, "_", sex)]
ranks_tab <- copy(ranks)
ranks_tab[, Ntot := .N, by = c("model", "broadsource")]
ranks_tab[, Nrank := .N, by = c("model", "rank", "broadsource")]
ranks_tab <- unique(ranks_tab, by = c("model", "rank", "broadsource"))
ranks_tab[, prop := Nrank/Ntot]

ranks_tab$model <- factor(ranks_tab$model,
                          levels = c("power_law", "beta", "weibull", "gen_gamma",
                                     "gamma", "oppermann", "steffensen",
                                     "heligman_pollard", "bourgeois_pichat",
                                     "exponential"))

# # plot ranks
png("FILEPATH/ranks.png", height = 500, width = 500)
gg_ranks <- ggplot(ranks_tab)+
            geom_tile(aes(x = as.factor(rank), y = model, fill = prop)) +
            scale_fill_gradient2(low="navy", mid="green", high="yellow",
                       midpoint=0.4, limits=range(ranks_tab$prop)) +
            theme_classic() + labs(fill = "proportion") +
            facet_wrap("broadsource") + xlab("rank")
gg_ranks
dev.off()

## subset to high-ranking models ========================================

# grab models ranking 1-5
keep <- ranks[rank <= 3]

# merge parameters back on
```

```
keep <- merge(keep, params, by = c("model", "sex", "source"), all.x = T)

# recalc param quantiles
params <- keep[,c("model", "sex", "source", "broadsource", "parameter", "value")]
params[, value := as.numeric(value)]
params[, quantile90 := round(quantile(value, 0.90), 2),
       by = c("model", "parameter")]
params[, quantile10 := round(quantile(value, 0.10), 2),
       by = c("model", "parameter")]
params[, quantile50 := round(quantile(value, 0.50), 2),
       by = c("model", "parameter")]
param_quantiles <- unique(params, by = c("model", "parameter"))
param_quantiles <- param_quantiles[, c("model", "parameter",
                                       "quantile10", "quantile50", "quantile90")]

# reshape wide on parameter
keep <- dcast(keep, model + sex + source + rank ~ parameter, value.var = "value")

# merge predictions back on
keep <- merge(keep, preds, by = c("model", "sex", "source"), all.x = T)
keep[, source_sex := paste0(source, "_", sex)]
keep <- keep[order(broadsource, source, sex, model, aod_days)]

# remove any instances where not continuously decreasing
keep[, pred_prev_age := shift(pred), by = c("model", "source", "sex")]
keep[, flag := ifelse((pred - pred_prev_age) > 0, 1, 0)]
keep[, flag := sum(flag, na.rm=T), by = c("model", "sex", "source")]
test <- keep[flag > 0]
keep <- keep[flag == 0]

# remove any instances where prediction goes outside 0-1
keep[, flag := ifelse((pred < 0 | pred > 1), 1, 0)]
keep[, flag := sum(flag, na.rm=T), by = c("model", "sex", "source")]
keep <- keep[flag == 0]

# add mean model
mean_mod <- preds[, c("sex", "source", "broadsource", "aod_days", "prop_deaths")]
mean_mod <- unique(mean_mod)
mean_mod[, pred := mean(prop_deaths), by = c("sex", "broadsource", "aod_days")]
mean_mod <- unique(mean_mod, by = c("sex", "broadsource", "aod_days"))
mean_mod[, source := broadsource]
mean_mod[, model := "empirical_mean"]
mean_mod[, source_sex := paste0(source, "_", sex)]

# smooth over DHS and MICS age heaping
mean_mod <- mean_mod[order(source, sex, aod_days)]
mean_mod[, pred_prev_age := shift(pred, type = "lag"), by = c("source", "sex")]
mean_mod[, pred_next_age := shift(pred, type = "lead"), by = c("source", "sex")]
mean_mod[, pred_prev2_age := shift(pred, type = "lag", 2), by = c("source", "sex")]
mean_mod[, pred_next2_age := shift(pred, type = "lead", 2), by = c("source", "sex")]
mean_mod[source %in% c("MICS", "DHS") & aod_days > 4 & aod_days < 549,
         pred := 0.2*pred + 0.2*pred_prev_age + 0.2*pred_next_age +
                 0.2*pred_prev2_age + 0.2*pred_next2_age]
```

```
mean_mod[source %in% c("MICS", "DHS") & aod_days == 4 & aod_days < 549,
         pred := 0.2*pred + 0.4*pred_prev_age + 0.2*pred_next_age +
           0*pred_prev2_age + 0.2*pred_next2_age]
keep <- rbind(keep, mean_mod, fill = T)

# plot all fits
keep$model <- factor(keep$model, levels = c("bourgeois_pichat", "oppermann",
                                    "steffensen", "heligman_pollard", "gamma",
                                    "gen_gamma", "weibull", "beta", "power_law",
                                    "empirical_mean"))
png("FILEPATH/keep_fits_log.png", height = 600, width = 600)
gg_all_fits <- ggplot(data = keep) +
               geom_line(aes(x = log(aod_days),
                             y = log(pred),
                             group = source_sex), alpha = 0.1) +
              facet_wrap("model") +
              theme_classic() +
              ylab("log proportion u5 deaths") + xlab("log age (days)") +
              ylim(-12,0)
gg_all_fits
dev.off()

# format for model database
keep <- keep[, c("model", "sex", "source", "broadsource", "aod_days", "pred")]

# table of model database
tab_keep <- unique(keep, by = c("model", "sex", "source", "broadsource"))
tab_keep <- as.data.table(table(tab_keep$model))
tot <- sum(tab_keep$N)
tab_keep[, prop := round(N/tot,3)]
tab_keep <- tab_keep[order(-prop)]
print(tab_keep)

# save as model database
write.csv(keep, "FILEPATH/model_database.csv", row.names = F)

# END
```

Fit relational model

```
## Fit relational model:
##     Pull in processed data and model database
##     Expand data to sims based on binomial SE
##     Collapse to ENN, LNN, PNN, 0-1, and 1-4
##     Fit relational model on collapsed age
##     Predict out for all ages

## SET-UP ========================================================

library(data.table)
library(ggplot2)
library(stats)
```

```
library(rmutil, lib.loc = "FILEPATH/packages")
library(argparse)
library(plyr)

# get arguments
parser <- ArgumentParser()
parser$add_argument('--source', type="character", required=TRUE,
                    help='The source of interest')
parser$add_argument('--sex', type="integer", required=TRUE,
                    help='The sex of interest')
args <- parser$parse_args()
sex_id <- args$sex
source_name <- args$source

# get prepped data
dt_raw <- fread("FILEPATH/processed_data_age_days.csv")

# subset
dt_raw <- dt_raw[sex==sex_id & source == source_name & !is.na(prop_deaths)]
dt_raw <- dt_raw[order(aod_days)]
dt_raw <- dt_raw[, c("sex", "source", "broadsource", "prop_deaths",
                     "aod_days", "total_u5_deaths")]

# get model database
modeldb <- fread("FILEPATH/model_database.csv")
if("prop_deaths" %in% names(modeldb)) modeldb[, prop_deaths := NULL]
setnames(modeldb, "pred", "prop_deaths")

# expand to draws ======================================================

sims <- data.table(draw = 0:99, mergeid = 1)
dt_raw$mergeid <- 1
dt_raw <- merge(dt_raw, sims, by = "mergeid", allow.cartesian = T)
setnames(dt_raw, "prop_deaths", "prop_deaths_meanval")
dt_raw[, prop_deaths := rbinom(.N, total_u5_deaths,
                               prop_deaths_meanval)/total_u5_deaths]
dt_raw[, minobserved := min(prop_deaths_meanval)]
dt_raw[prop_deaths <= 0.1*minobserved, prop_deaths := 0.1*minobserved]
dt_raw[, minobserved := NULL]

# collapse on age ======================================================

dt_raw[, group := draw]
modeldb[, group := .GRP, by= c("model", "sex", "source")]

interp_age <- function(dt){

  # some models are missing an age. If so, log-log-linearly interpolate
  #   to create complete age series -- could also fix this at model fit
  full_age_set <- expand.grid(group = unique(dt$group),
                              aod_days = c(seq(1,28,1), 47, 77, 107, 138,
                                           168, 199, 229, 259, 290, 320, 351,
                                           549, 913, 1279, 1643))
```

```
  dt <- merge(dt, full_age_set, by = c("group", "aod_days"), all.y = T)
  dt[, sex := unique(sex)[1], by = "group"]
  dt[, source := unique(source)[1], by = "group"]
  dt[, broadsource := unique(broadsource)[1], by = "group"]
  if("model" %in% names(dt)) dt[, model := unique(model)[1], by = "group"]
  interp_prop_deaths <- function(d){
    d$prop_deaths <- exp(approx(log(d$aod_days),
                                log(d$prop_deaths),
                                xout = log(d$aod_days))$y)
    return(d)
  }
  dt <- ddply(dt, "group", interp_prop_deaths)
  dt <- as.data.table(dt)
  return(dt)
}
dt_raw <- interp_age(dt_raw)
modeldb <- interp_age(modeldb)

# add age group variable for enn, lnn, pnn, 1-4
add_age_group <- function(dt){
  dt <- copy(dt)
  # create age_group variable
  dt[aod_days < 7, age_group := "enn"]
  dt[aod_days >=7 & aod_days < 28, age_group := "lnn"]
  dt[aod_days >= 28 & aod_days < 365, age_group := "pnn"]
  dt[aod_days >= 365, age_group := "ch"]
  # convert proportions back from per/day units
  dt[age_group %in% c("enn", "lnn"), prop_deaths_full := prop_deaths]
  dt[age_group == "pnn", prop_deaths_full := prop_deaths*30.4]
  dt[age_group == "ch", prop_deaths_full := prop_deaths*365]
}
dt_raw <- add_age_group(dt_raw)
modeldb <- add_age_group(modeldb)

# scale to 100%
scale100 <- function(dt){
  dt <- copy(dt)
  # sum over group and scale
  dt[, sum_by_group := sum(prop_deaths_full), by = "group"]
  dt[, prop_deaths := prop_deaths / sum_by_group]
  return(dt)
}
#dt_raw <- scale100(dt_raw)
modeldb <- scale100(modeldb)

collapse_age <- function(dt){

  dt <- copy(dt)

  # convert proportions back from per/day units
  dt[, prop_deaths := prop_deaths_full]

  # make copy for 0-1 group (sum of enn, lnn, pnn)
```

```
  inf <- dt[age_group != "ch"]
  inf[, age_group := "inf"]
  dt <- rbind(dt, inf)

  # collapse by group
  dt[, prop_deaths := sum(prop_deaths), by = c("group", "age_group")]
  dt <- unique(dt, by = c("group", "age_group"))

  # convert back to by-day prop
  dt[age_group == "enn", prop_deaths := prop_deaths/7]
  dt[age_group == "lnn", prop_deaths := prop_deaths/21]
  dt[age_group == "pnn", prop_deaths := prop_deaths/337]
  dt[age_group == "inf", prop_deaths := prop_deaths/(7+21+337)]
  dt[age_group == "ch", prop_deaths := prop_deaths/(365*4)]

  return(dt)

}
collapsed_dt <- collapse_age(dt_raw)
collapsed_modeldb <- collapse_age(modeldb)

# choose standards ====================================================

# merge together data and model db
setnames(collapsed_modeldb, "prop_deaths", "prop_deaths_model")
all <- merge(collapsed_modeldb, collapsed_dt, by = "age_group",
             all.x = T, allow.cartesian = T)

# calculate distance between models and data
all[, diff := sum(log(prop_deaths_model/prop_deaths)^2), by = c("group.x", "draw")]

# select 100 standards that are closest
all <- all[order(diff, group.x, draw)]
for(draw_num in 0:99){
  all[draw == draw_num, rank := .GRP, by = "group.x"]
}
all <- all[rank <= 100]
used <- as.data.table(table(all[age_group == "enn"]$model))
setnames(used, c("model", "count"))
write.csv(used, paste0("FILEPATH/models_used_std/", source_name,
                  "_", sex_id, ".csv"), row.names = F)

# fit relational models ===============================================

# logit transform
logit <- function(x) return(log(x /(1 - x)))
inv_logit <- function(x) return(exp(x)/(1 + exp(x)))
all[, logit_stan := logit(prop_deaths_model)]
all[, logit_obs := logit(prop_deaths)]

# model: logit(prop_deaths) = a + b * logit(prop_deaths_model)
relation <- data.table()
for(gg in unique(all$group.x)){
```

```
  temp1 <- all[group.x == gg]
  for(draw_num in temp1$draw){
    temp <- temp1[draw == draw_num]
    fit <- lm(data = temp, logit_obs ~ logit_stan)
    temp <- unique(temp[, c("model", "sex.x", "source.x",
                            "group.x", "draw", "rank")])
    temp$intercept <- as.numeric(coef(fit)[1])
    temp$slope <- as.numeric(coef(fit)[2])
    temp$rsquared <- summary(fit)$r.squared
    relation <- rbind(relation, temp, fill = T)
  }
}
setnames(relation, c("sex.x", "source.x", "group.x"),
         c("sex", "source", "group"))

# remove any with slope < 0
relation <- relation[slope >= 0]
relation <- unique(relation)

# save relationships
write.csv(relation, paste0("FILEPATH/relation_summaries/relation_summary_",
                           source_name, "_", sex_id, ".csv"), row.names = F)

# predict relational models ============================================

# go back to complete age series "standards" and apply model transformation
preds <- merge(relation, modeldb, by = c("model", "sex", "source", "group"),
               all.x = T, all.y = F, allow.cartesian = T)
preds[, logit_prop_deaths := logit(prop_deaths)]
preds[, logit_relational := intercept + slope * logit_prop_deaths]
preds[, pred_relational := inv_logit(logit_relational)]

write.csv(preds, paste0("FILEPATH/relational_fits_precollapse/",
                        source_name, "_", sex_id, ".csv"))

# find mean lower upper
preds[, median_val := median(pred_relational), by = "aod_days"]
preds[, lower_val := quantile(pred_relational, 0.025), by = "aod_days"]
preds[, upper_val := quantile(pred_relational, 0.975), by = "aod_days"]
preds <- preds[, c("aod_days", "median_val", "lower_val", "upper_val")]
preds <- unique(preds, by = c("aod_days"))

# merge data back on
dt_raw <- dt_raw[draw == 1]
preds <- merge(preds, dt_raw, by = "aod_days")

# # # # plot
gg_pred <- ggplot(preds, aes(x = log(aod_days))) +
           geom_ribbon(aes(ymin = log(lower_val),
                           ymax = log(upper_val)),
                           fill = "grey70") +
           geom_line(aes(y = log(median_val))) +
           geom_point(aes(y = log(prop_deaths_meanval))) + ylim(-16, 0)
```

```r
# save for compilation
write.csv(preds, paste0("FILEPATH/relational_fits/relational_pred_",
                        source_name, "_", sex_id, ".csv"), row.names = F)

# END
```

Compile relational models

```r
## Compile relational models
##    Calculate coverage

param_files <- list.files("FILEPATH/relation_summaries", full.names = T)

params <- data.table()
for(ff in param_files){
  temp <- fread(ff)
  params <- rbind(params, temp, fill = T)
}

mean(params$rsquared)
mean(params$slope)
mean(params$intercept)

sd(params$rsquared)
sd(params$slope)
sd(params$intercept)

tab_params <- as.data.table(table(params$model))
tot <- sum(tab_params$N)
tab_params[, prop := round(N/tot,3)]
tab_params <- tab_params[order(-prop)]
tab_params

# END
```

Case-study

```r
## Case-study
## WA males 1961

library(data.table)
library(ggplot2)

main_dir <- "FILEPATH"

sex_id <- 1
sex <- "males"
source_name <- "usa_48_1961_VR"
source_label <- "Washington State 1961"
```

```r
# input data prep ==============================

dt <- fread("FILEPATH/raw_data_before_redistribution.csv")
dt <- dt[sex == sex_id & source == source_name]
dt <- dt[, c("units", "aod", "sex", "aod_cat", "num_deaths",
             "total_u5_deaths", "prop_deaths")]

aod_table <- expand.grid(aod = 0:99,
                         units = c("days","weeks","months","years"))
aod_table <- as.data.table(aod_table)
aod_table[, aod_cat := paste0(aod, "_", units)]
aod_categories <- aod_table$aod_cat
dt$aod_cat <- factor(dt$aod_cat, levels = aod_categories)
dt <- dt[order(aod_cat)]

# expand to all expected age groups
aod_table <- aod_table[!(units == "days" & aod > 27) &
                       !(units == "months" & (aod > 11 | aod == 0)) &
                       !(units == "years" & (aod > 4 | aod == 0)) &
                       units != "weeks"]
aod_table <- aod_table[, c("aod_cat")]
dt <- merge(dt, aod_table, by = c("aod_cat"), all.x = T, all.y = T)
dt$aod_cat <- factor(dt$aod_cat, levels = aod_categories)
dt <- dt[order(aod_cat)]
dt[is.na(num_deaths), num_deaths := 0]

# bar plot
png(paste0(main_dir,"/", sex_id, "_", source_name, "_raw.png"),
    width = 800, height = 400)
  gg <- ggplot(data = dt,
               aes(x = aod_cat,
                   y = num_deaths)) +
    geom_bar(stat="identity") +
    geom_text(aes(label = num_deaths), vjust= -1, size = 5) +
    xlab("Age") +
    ylab("Number of deaths") +
    theme_bw() +
    theme(axis.text.x = element_text(angle = 60, hjust = 1, size = 16),
          axis.text.y = element_text(size = 16),
          axis.title.x = element_text(size = 16),
          axis.title.y = element_text(size = 16)) +
    ylim(0, 1.1 * max(dt$num_deaths))
  print(gg)
dev.off()

# processed data with standard error
dt <- fread("FILEPATH/processed_data_age_days.csv")
dt <- dt[sex == sex_id & source == source_name]
dt <- dt[, c("units", "aod", "aod_days", "sex", "aod_cat", "num_deaths",
             "prop_deaths", "log_prop_deaths", "standard_error",
             "prop_deaths_lower", "prop_deaths_upper", "total_u5_deaths")]

# plot with standard error
```

```
png(paste0(main_dir,"/", sex_id, "_", source_name, "_se.png"),
    width = 800, height = 400)
  gg <- ggplot(data = dt,
               aes(x = log(aod_days),
                   y = log(prop_deaths))) +
    geom_ribbon(aes(ymin = log(prop_deaths_lower),
                    ymax = log(prop_deaths_upper)),
                    alpha = 0.2) +
    geom_point() +
    geom_vline(xintercept = log(8)) +
    geom_vline(xintercept = log(29)) +
    geom_vline(xintercept = log(366)) +
    xlab("log age (days)") +
    ylab("log proportion of under-5 deaths") +
    theme_bw() +
    theme(axis.text.x = element_text(size = 18),
        axis.text.y = element_text(size = 18),
        axis.title.x = element_text(size = 18),
        axis.title.y = element_text(size = 18))
  print(gg)
dev.off()


# parametric ==================================

temp <- fread("FILEPATH/1_usa_48_1961_VR_predictions.csv")

cols <- c(exponential = "blue",
          power_law = "purple",
          heligman_pollard = "orange",
          bourgeois_pichat = "magenta",
          oppermann = "springgreen",
          steffensen = "dodgerblue",
          weibull = "gold",
          beta = "plum1",
          gamma = "coral1",
          generalized_gamma = "darkcyan")

png(paste0("FILEPATH/", sex_id, "_", source_name, "_parametric.png"),
    width=800, height=400)
gg <- ggplot(data = temp, aes(x=log(aod_days))) +
  geom_line(aes(y = log(pred_exponential), color = "exponential")) +
  geom_line(aes(y = log(pred_power_law), color = "power_law")) +
  geom_line(aes(y = log(pred_heligman_pollard), color = "heligman_pollard")) +
  geom_line(aes(y = log(pred_bourgeois_pichat), color = "bourgeois_pichat")) +
  geom_line(aes(y = log(pred_oppermann), color = "oppermann")) +
  geom_line(aes(y = log(pred_steffensen), color = "steffensen")) +
  geom_line(aes(y = log(pred_weibull), color = "weibull")) +
  geom_line(aes(y = log(pred_beta), color = "beta")) +
  geom_line(aes(y = log(pred_gamma), color = "gamma")) +
  geom_line(aes(y = log(pred_gen_gamma), color = "generalized_gamma")) +
  geom_point(aes(y = log(prop_deaths)), color = "black") +
  theme_bw() +
```

```
  ylim(1.5*min(temp$log_prop_deaths), 0.5) +
  scale_color_manual(name = "Model", values = cols) +
  ylab("log proportion under-5 deaths") +
  xlab("log age (days)") +
  theme(axis.text.x = element_text(size = 18),
        axis.text.y = element_text(size = 18),
        axis.title.x = element_text(size = 18),
        axis.title.y = element_text(size = 18),
        legend.text = element_text(size = 18),
        legend.title = element_text(size = 18))
print(gg)
dev.off()

# get rankings and sse
# reshape long on model
preds <- copy(temp)
preds <- melt(preds,
              measure.vars = c("pred_exponential", "pred_power_law",
                  "pred_heligman_pollard", "pred_bourgeois_pichat",
                  "pred_oppermann", "pred_steffensen", "pred_beta",
                  "pred_gamma", "pred_gen_gamma", "pred_weibull"))
preds[, c("pred", "sq_error") := NULL]
setnames(preds, "value", "pred")
setnames(preds, "variable", "model")
preds[, model := gsub("pred_", "", model)]
preds[, pred := as.numeric(pred)]

# calculate residual
preds[, resid := log(pred/prop_deaths)]
preds[, mean_resid := mean(resid, na.ram=T), by=c("aod_days","model")]
dt_resids <- unique(preds, by=c("aod_days","model"))

# calculate SSE
preds[, sse_pred := sum(resid^2), by = c("sex", "model")]
ranks <- preds[,c("model", "sse_pred")]
ranks <- unique(ranks)
ranks <- ranks[order(sse_pred)]


# relational ===================================

dt <- fread(paste0("FILEPATH/relational_fits/",
                        source_name, "_", sex_id, "_precollapse.csv"))
dt[, group := paste0(group, "_", draw)]

png(paste0("FILEPATH/", sex_id, "_", source_name, "_relational.png"),
    width=800, height=400)
gg <- ggplot(data = dt) +
            geom_line(aes(x=log(aod_days),
                          y=log(pred_relational),
                          group = group),
                          alpha = 0.1,
                          color = "grey88") +
```

```
            geom_point(aes(x=log(aod_days),
                 y=log(prop_deaths)),
                 color = "black") +
            theme_bw() +
            theme(axis.text.x = element_text(size = 18),
              axis.text.y = element_text(size = 18),
              axis.title.x = element_text(size = 18),
              axis.title.y = element_text(size = 18),
              legend.text = element_text(size = 18),
              legend.title = element_text(size = 18)) +
            xlab("log age (days)") +
            ylab("log proportion under-5 deaths") +
            ylim(-12, 0)
 print(gg)
dev.off()

# table models used
dt <- unique(dt, by = c("group"))
dt <- as.data.table(table(dt$model))
dt <- dt[order(-N)]

# collapsed predictions & raw data
dt <-  fread(paste0("FILEPATH/relational_fits/relational_pred_",
                        source_name, "_", sex_id, ".csv"))

png(paste0("FILEPATH/", sex_id, "_", source_name, "_final.png"),
    width=800, height=400)
    gg_pred <- ggplot(dt, aes(x = log(aod_days))) +
                  geom_ribbon(aes(ymin = log(lower_val),
                                  ymax = log(upper_val)),
                              fill = "grey70") +
                  geom_line(aes(y = log(median_val))) +
                  geom_point(aes(y = log(prop_deaths_meanval))) +
                  xlab("log age in days") + ylab("log proportion of u5 deaths") +
                  theme_bw() +
                  theme(axis.text.x = element_text(size = 18),
                        axis.text.y = element_text(size = 18),
                        axis.title.x = element_text(size = 18),
                        axis.title.y = element_text(size = 18),
                        legend.text = element_text(size = 18),
                        legend.title = element_text(size = 18)) +
                  ylim(-12, 0)
    print(gg_pred)
dev.off()

# calculate coverage
dt[, covered := ifelse(prop_deaths_meanval >= lower_val &
                        prop_deaths_meanval <= upper_val, 1, 0)]
dt[, coverage := sum(covered)/.N]

## END
```