

HPbayes Version 1 for R: A Package for Estimating the Heligman-Pollard Mortality Model

David J Sharrow

November 23, 2009

HPbayes is an R package for estimating the Heligman-Pollard mortality model (Heligman and Pollard, 1980). It provides parameter estimates via maximum likelihood or within a Bayesian framework, namely Bayesian Melding with IMIS (Poole and Raftery, 2000; Raftery and Bao, 2009). The package also provides functions for plotting and inspecting the resulting parameter/output distribution(s) along with uncertainty estimates.¹

Contents

1	Overview	2
1.1	A summary of selected package functions	2
2	Estimating the HP model using maximum likelihood	3
2.1	Basic example	3
2.2	Specifying an alternative algorithm for <code>optim()</code>	4
3	Estimating the HP model with Bayesian Melding	5
3.1	Forming a prior	5
3.2	An example using <code>hp.bm.imis()</code>	7
4	Plotting functions	8
4.1	Plotting the prior and posterior distributions	8
4.2	Plotting the posterior output	8
5	Other useful functions	10
5.1	Generating ${}_nq_x$ values from the posterior parameter distribution	11
5.2	Generating an unabridged life table	12

¹Many people have offered suggestions and comments to improve this software. Thanks to the members of the BayesPop working group (Hana Ševčíková, Samuel Clark, Adrian Raftery, Le Bao, Jennifer Chunn, Greg Matthews, Jason Thomas and Mark Wheldon) for their insightful comments.

1 Overview

The HPbayes software allows the user to estimate the eight parameters of the Heligman-Pollard model using either maximum likelihood or a Bayesian approach, Bayesian Melding. This manuscript describes the major functions contained within the package as well as some basic demonstrations of how to use the package functions with data contained in the package itself. First, I begin with a description of the functions and structure of the package followed by examples. For all examples, it is assumed that the package is loaded already (`> library(HPbayes)`). The list below describes the functions used throughout this manuscript so as to allow the reader an easy reference when working through the examples. The package does contain other secondary functions that allow the user to complete the Bayesian Melding process with smaller steps but these functions are bundled into a single function, `hp.bm.imis`, which runs all the steps sequentially and thus that is the function used in the example here.

1.1 A summary of selected package functions

Forming a prior

- `prior.form` - Draws from a uniform distribution with bounds set by the user to create a uniform prior distribution for each of the Heligman-Pollard parameters

Estimating the model

- `pri.mle` - Estimates the HP model via maximum likelihood after receiving user-specified start values (mle estimates are returned as `prior.mle$mle`).²
- `hp.bm.imis` - Runs all the necessary functions to estimate the eight Heligman-Pollard parameters in one step via Bayesian Melding with IMIS

Post-estimation functions

- `mod8p` - Calculates age-specific probabilities of death using the Heligman-Pollard model from a user-supplied set of parameters
- `hp.nqx` - Converts a set of Heligman-Pollard mortality model parameters into age-specific probabilities of death
- `hp.lifetab` - Generates a life table from the age-specific probabilities of death resulting from the estimation of the eight parameters of the Heligman-Pollard mortality model

²A uniform prior distribution can also be generated with this function when the user specifies the bounds of the uniform from which to draw as $i * s.e.$ where i is the number of standard errors on either side of the mle estimate. Some users may choose this option to generate a prior for the Bayesian estimation functions.

Plotting functions

- `hpbayes.plot` - Converts the posterior Heligman-Pollard parameter distribution resulting from a Bayesian Melding procedure to probabilities of death over a specified age range and plots the resulting curves. In addition, this function also calculates and plots the 95% CI over the specified age range.
- `postpri.plot` - Produces an eight panel plot of either the kernel density or box plots of both the prior and posterior distribution for the eight parameters of the Heligman-Pollard model

2 Estimating the HP model using maximum likelihood

The following provides a basic example of how to use the package to estimate the model via maximum likelihood methods. For this illustration we use the data set provided with the package (`data(HPprior)`).

2.1 Basic example

First we can examine the data. The following commands load the package and data. The data set contains a set of age specific death counts, `dx`, a set of age specific persons at risk in each age interval, `lx`, a vector containing the beginning ages for the following intervals, 0-1, 1-4, 5-9, 10-14...100+, and an 8000 x 8 matrix in which each row is a set of Heligman-Pollard parameters forming a prior distribution for each parameter. For the Bayesian Melding example, I generate a prior instead of using the provided one for illustrative purposes.

```
> library(HPbayes)
> data(HPprior)
```

The data looks as follows:³

```
> lx
[1] 1974 1906 1860 1844 1834 1823 1793 1700 1549 1361 1181 1025 870 721
[15] 571 450 344 256 142 79 41 8
> dx
[1] 68 47 16 10 13 29 92 151 188 179 156 155 147 150 122 106 88
[18] 113 63 38 32 8
```

For the maximum likelihood example a prior is unnecessary so we proceed with the `pri.mle` command. This command performs a maximum likelihood estimation of the parameters after a user specified set of deaths and the number of persons entering each interval

³Although the HP law models the age specific probability of death (${}_nq_x = d_x/l_x$) the user need only supply the persons at risk and age-specific deaths counts. The various commands will calculate the ${}_nq_x$ values within the function whenever necessary.

are identified. There is no need to provide start values for the maximum likelihood estimation as there is already a set of default values (`theta.test= c(0.06008, 0.31087, 0.34431, 0.00698, 1.98569, 26.71071, 0.00022, 1.088)`). This command uses `optim()` (R Development Core Team, 2009) to select a set of parameters to maximize the log likelihood. The default `method` command for `optim()` is Nelder-Mead.

```
> result.mle <- pri.mle(nrisk=lx, ndeath=dx, age=age)
> names(result.mle)
[1] "q0"      "mle"      "se.out"   "d.vc"
```

The output from `pri.mle` contains four objects. The first, `result.mle$q0` is a prior based on the mle estimates (more on this in the next section), `se.out` is the standard error for each parameter estimate obtained by solving the negative hessian matrix and taking the square root of the diagonal elements (`d.vc`) of the resulting matrix. Using `result.mle$mle`, we can view the mle results.

```
> result.mle$mle
[1] 0.053189942 1.637759782 0.293710043 0.106077892
[5] 4.226995789 41.155559982 0.001763765 1.075216815
```

2.2 Specifying an alternative algorithm for `optim()`

In some instances it may be desirable to use an algorithm other than Nelder-Mead or one that allows the user to set upper and lower bounds on the parameter estimates. Changing the `method` argument for `optim()` is accomplished by changing the `opt.meth` argument passed to `pri.mle` to one of the options for `method` in `optim()`. The `lo` and `hi` arguments are vectors of length eight containing the upper and lower bounds for each parameter. Both arguments default to the following `lo = c(1e-08, 1e-07, 1e-07, 1e-07, 1e-07, 15, 1e-07, 1)`, `hi = c(1, 1, 1, 0.5, 15, 55, 0.1, 1.5)`.

```
> result2.mle <- pri.mle(nrisk=lx, ndeath=dx, age=age, opt.meth="L-BFGS-B")
> result2.mle$mle
[1] 0.032588597 1.000000000 0.328714375 0.087708060
[5] 10.344012577 37.095971829 0.005593035 1.061588810
```

Notice that the second parameter estimate (1.000) is now exactly one because we have set an upper bound on that parameter of 1. We can now turn to estimating the parameters via Bayesian Melding using the function `hp.bm.imis`.

3 Estimating the HP model with Bayesian Melding

Bayesian Melding (Poole and Raftery, 2000) was developed specifically for use with deterministic models like the Heligman-Pollard and provides uncertainty estimates around the model parameters and the outputs (i.e. the age-specific probabilities of death). The result of the Bayesian Melding procedure is a posterior distribution for each of the eight parameters which can then be used in the model and, as a function of age, will produce a set of ${}_nq_x$ values along the range of ages provided.⁴

3.1 Forming a prior

Working with the same data as above, we must first form a prior distribution for the eight parameters. This is easily accomplished in one of two ways. The first, `prior.form`, simply takes x number (user-specified) of draws from a uniform distribution with bounds set by the user for each of the eight parameters and column binds each vector (the x draws for parameter i) together to get an $x \times 8$ matrix consisting of the x sets of parameters. No arguments need be passed to `prior.form` because defaults are provided for each argument (i.e. a set of lower and upper bounds along with a specified number of draws, $x = 8000$, are the defaults).

```
> q0 <- prior.form()
> dim(q0)
[1] 8000      8
> summary(q0)
```

V1	V2	V3
Min. :3.075e-05	Min. :8.877e-05	Min. :5.732e-05
1st Qu.:3.737e-02	1st Qu.:2.484e-01	1st Qu.:2.529e-01
Median :7.614e-02	Median :5.047e-01	Median :5.012e-01
Mean :7.561e-02	Mean :5.001e-01	Mean :5.009e-01
3rd Qu.:1.139e-01	3rd Qu.:7.523e-01	3rd Qu.:7.465e-01
Max. :1.500e-01	Max. :9.999e-01	Max. :9.997e-01

V4	V5	V6
Min. :1.004e-05	Min. :5.195e-05	Min. :15.01
1st Qu.:6.378e-02	1st Qu.:3.658e+00	1st Qu.:25.05
Median :1.243e-01	Median :7.417e+00	Median :35.11
Mean :1.254e-01	Mean :7.450e+00	Mean :35.11
3rd Qu.:1.891e-01	3rd Qu.:1.126e+01	3rd Qu.:45.09
Max. :2.500e-01	Max. :1.500e+01	Max. :54.98

V7	V8
Min. :2.346e-06	Min. :0.0002067
1st Qu.:2.502e-02	1st Qu.:0.3080434

⁴See Poole and Raftery (2000) and Raftery and Bao (2009) for a description of Bayesian Melding using the IMIS algorithm and Sharroo and Clark (ND) for an application to the Heligman-Pollard.

Median	:4.982e-02	Median	:0.6198962
Mean	:5.006e-02	Mean	:0.6214362
3rd Qu.	:7.518e-02	3rd Qu.	:0.9332462
Max.	:9.998e-02	Max.	:1.2499607

The second option is to use `prior.mle` as above which will create a prior based on the mle estimates. Similarly to `prior.form`, the prior is drawn from a uniform distribution but with this function, the center of the uniform distribution from which to draw is the mle estimate for parameter i and the upper and lower bounds extend to a user specified number of standard errors on either side of the estimate. The user can control the number of standard errors (i.e. the width of the uniform distribution) with the argument `se.num`. This argument defaults to 15, creating a uniform distribution 30 standard errors wide. When forming the prior, the `hi` and `lo` arguments place bounds on the draws as well. For instance, if `lo` is set such that the lower bound for each parameter is 0, then any draws resulting in a negative number will be thrown out and redrawing will take place until the conditions are satisfied. The following code demonstrates how to draw a prior using the mle estimates.

```
> prior.mle <- prior.mle(ndeath=dx, nrisk=lx, age=age)
> names(prior.mle)
[1] "q0"      "mle"      "se.out"   "d.vc"
> summary(prior.mle$q0)
```

V1		V2		V3	
Min.	:4.237e-06	Min.	:3.204e-05	Min.	:9.196e-05
1st Qu.	:1.243e-01	1st Qu.	:2.443e-01	1st Qu.	:2.512e-01
Median	:2.450e-01	Median	:4.944e-01	Median	:5.017e-01
Mean	:2.453e-01	Mean	:4.955e-01	Mean	:4.990e-01
3rd Qu.	:3.686e-01	3rd Qu.	:7.436e-01	3rd Qu.	:7.479e-01
Max.	:4.904e-01	Max.	:1.000e+00	Max.	:9.998e-01

V4		V5		V6	
Min.	:0.02839	Min.	: 0.002050	Min.	:22.60
1st Qu.	:0.06670	1st Qu.	: 3.138085	1st Qu.	:30.95
Median	:0.10589	Median	: 6.285214	Median	:39.11
Mean	:0.10557	Mean	: 6.300748	Mean	:39.04
3rd Qu.	:0.14455	3rd Qu.	: 9.423824	3rd Qu.	:47.28
Max.	:0.18377	Max.	:12.581748	Max.	:54.99

V7		V8	
Min.	:0.001353	Min.	:1.059
1st Qu.	:0.001558	1st Qu.	:1.067
Median	:0.001766	Median	:1.076
Mean	:0.001765	Mean	:1.075
3rd Qu.	:0.001969	3rd Qu.	:1.083
Max.	:0.002174	Max.	:1.092

In this case, the default upper bound for the sixth parameter is 55, so that prior distribution is bounded on the upper end at 55.

3.2 An example using `hp.bm.imis()`

Once we have a prior distribution for each parameter, age-specific death counts and the age-specific persons at risk, we can proceed with the estimation process. `K` is the user specified number of IMIS iterations (defaults to 100). `hp.bm.imis`⁵ will print a summary of the median of each parameter posterior distribution along with a user-specified upper and lower credible interval bound.

```
> result <- hp.bm.imis(prior=q0, K=100, nrisk=lx, ndeath=dx)
  Low CI Median High CI
1  0.023  0.026   0.029
2  0.086  0.113   0.136
3  0.122  0.142   0.163
4  0.117  0.125   0.131
5  2.749  3.069   3.391
6 45.590 46.743  47.743
7  0.001  0.001   0.001
8  1.066  1.078   1.090
> names(result)
[1] "H.final"      "h.mu"         "h.sig"        "log.like"     "log.like.0"
[6] "wts.0"       "d.keep"
```

The result of `hp.bm.imis` contains several objects. The posterior distribution of parameters is contained in `H.final` and a more detailed summary than the print out can be obtained as shown below. `h.mu` and `h.sig` are the estimates and standard errors from the optimizer step respectively. The rest of the objects are various other output from the estimation process including the likelihoods at various steps. See `?hp.bm.imis` for a full description. The next section demonstrates some of the functions included in the package to plot this posterior distribution.

```
> summary(result$H.final)
      V1          V2          V3          V4
Min.   :0.02173  Min.   :0.06463  Min.   :0.1120  Min.   :0.1152
1st Qu.:0.02472  1st Qu.:0.10382  1st Qu.:0.1353  1st Qu.:0.1222
Median :0.02574  Median :0.11251  Median :0.1418  Median :0.1247
Mean   :0.02575  Mean   :0.11234  Mean   :0.1422  Mean   :0.1246
3rd Qu.:0.02682  3rd Qu.:0.12106  3rd Qu.:0.1489  3rd Qu.:0.1271
Max.   :0.03021  Max.   :0.14524  Max.   :0.1735  Max.   :0.1342
```

⁵Because there are multiple sampling iterations, this function can take several minutes to complete depending on the size of `K`.

V5		V6		V7		V8	
Min.	:2.652	Min.	:45.15	Min.	:0.0004572	Min.	:1.057
1st Qu.	:2.974	1st Qu.	:46.35	1st Qu.	:0.0009049	1st Qu.	:1.074
Median	:3.069	Median	:46.74	Median	:0.0010728	Median	:1.078
Mean	:3.079	Mean	:46.72	Mean	:0.0010619	Mean	:1.078
3rd Qu.	:3.195	3rd Qu.	:47.11	3rd Qu.	:0.0011987	3rd Qu.	:1.082
Max.	:3.682	Max.	:48.05	Max.	:0.0016506	Max.	:1.096

4 Plotting functions

The following outlines two plotting functions included in the package. The first plots the prior and posterior distribution of the parameters while the second plots the posterior distribution output (i.e. each set of parameters in the posterior distribution are plugged into the model and are plotted along a user specified age range).

4.1 Plotting the prior and posterior distributions

The user may wish to compare the prior and posterior distributions in order to assess the ability of the prior to accurately capture the posterior distribution. One can compare the prior and posterior distributions (the posterior should be proportional to the density of the prior) using the function `postpri.plot`. This function takes as arguments the prior distribution, the posterior distribution and several aesthetic arguments. The following code produces figure one where the dashed vertical lines are the bounds of the prior distribution while the smoothed curve is the Kernel density of the posterior. Several other arguments along with any arguments passed to `par()` can be inserted as the first arguments to change aesthetic features of the plot.

```
> postpri.plot(bty="o", prior=q0, hpp=result$H.final, line.bound=TRUE,
line.col=c("red", "black"))
```

4.2 Plotting the posterior output

Once the posterior parameter estimates have been obtained, one can easily plot the posterior output (${}_nq_x$ values are the output from this deterministic model) using the function `hpbayes.plot`. Similarly to the function above, this argument takes the posterior parameter distribution as an argument along with the ages at which to calculate the ${}_nq_x$ values. The function takes several other arguments such as `plotdata`, which controls whether or not to plot the observed ${}_nq_x$ values (the user needs to provide the age-specific deaths and persons at risk to use this argument), `log`, a logical, which if `TRUE` plots on the log scale, `plotgrey` controls whether to plot each set of posterior parameters.

The four plots in figure two were produced with the following variations of `hpbayes.plot`. The user can also pass arguments to `par()` in the (...) argument of `hpbayes.plot`.

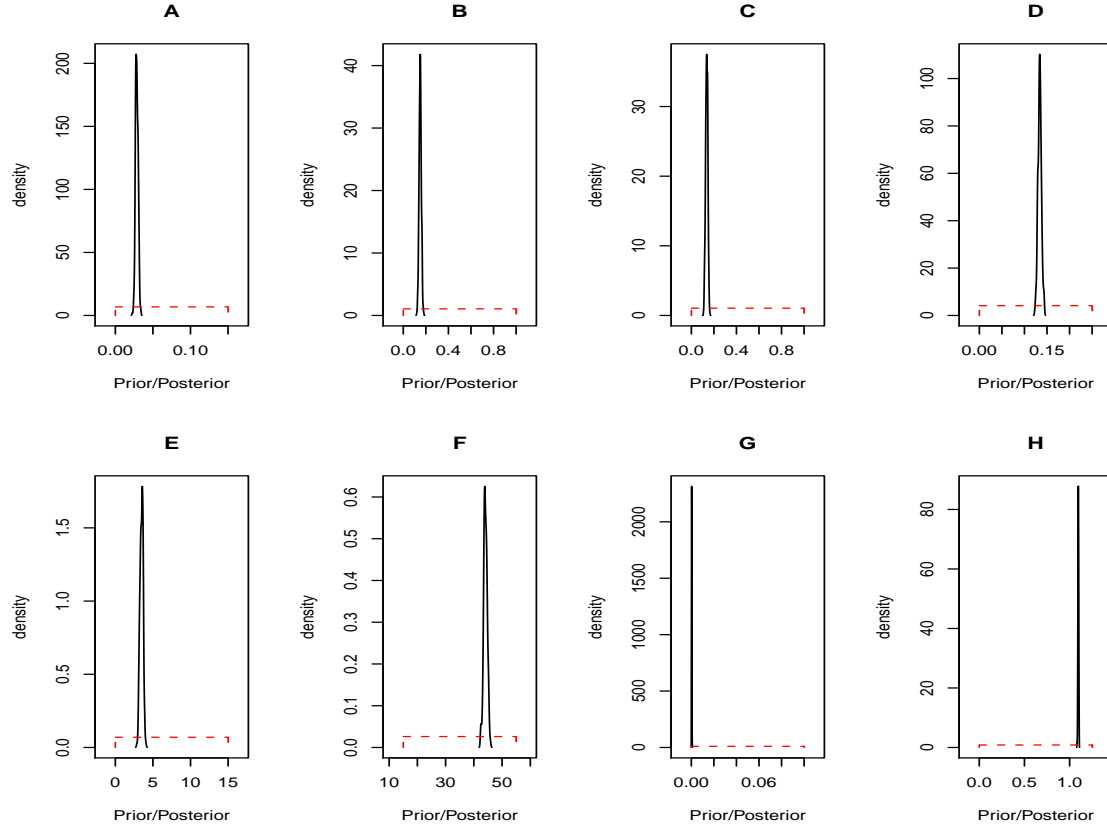


Figure 1: Resulting plot from `pripost.plot`. The dashed red boxes mark the minimum and maximum of the prior distribution along with the height of its density while the smoothed curve is the kernel density of the posterior distribution.

- **upper left** `hpbayes.plot(bty="n", nrisk=lx, ndeath=dx, age=age, hpp=result$H.final, plotdata=TRUE, plotpost=TRUE)`
- **upper right** `hpbayes.plot(bty="n", nrisk=lx, ndeath=dx, age=age, hpp=result$H.final, plotdata=FALSE, plotpost=FALSE)`
- **lower left** `hpbayes.plot(bty="n", nrisk=lx, ndeath=dx, age=age, hpp=result$H.final, plotdata=TRUE, plotpost=TRUE, line.col=c("grey", 1, 1, 1), data.type="p")`
- **lower right** `hpbayes.plot(bty="n", nrisk=lx, ndeath=dx, age=age, hpp=result$H.final, log=TRUE, plotdata=TRUE, plotpost=TRUE)`

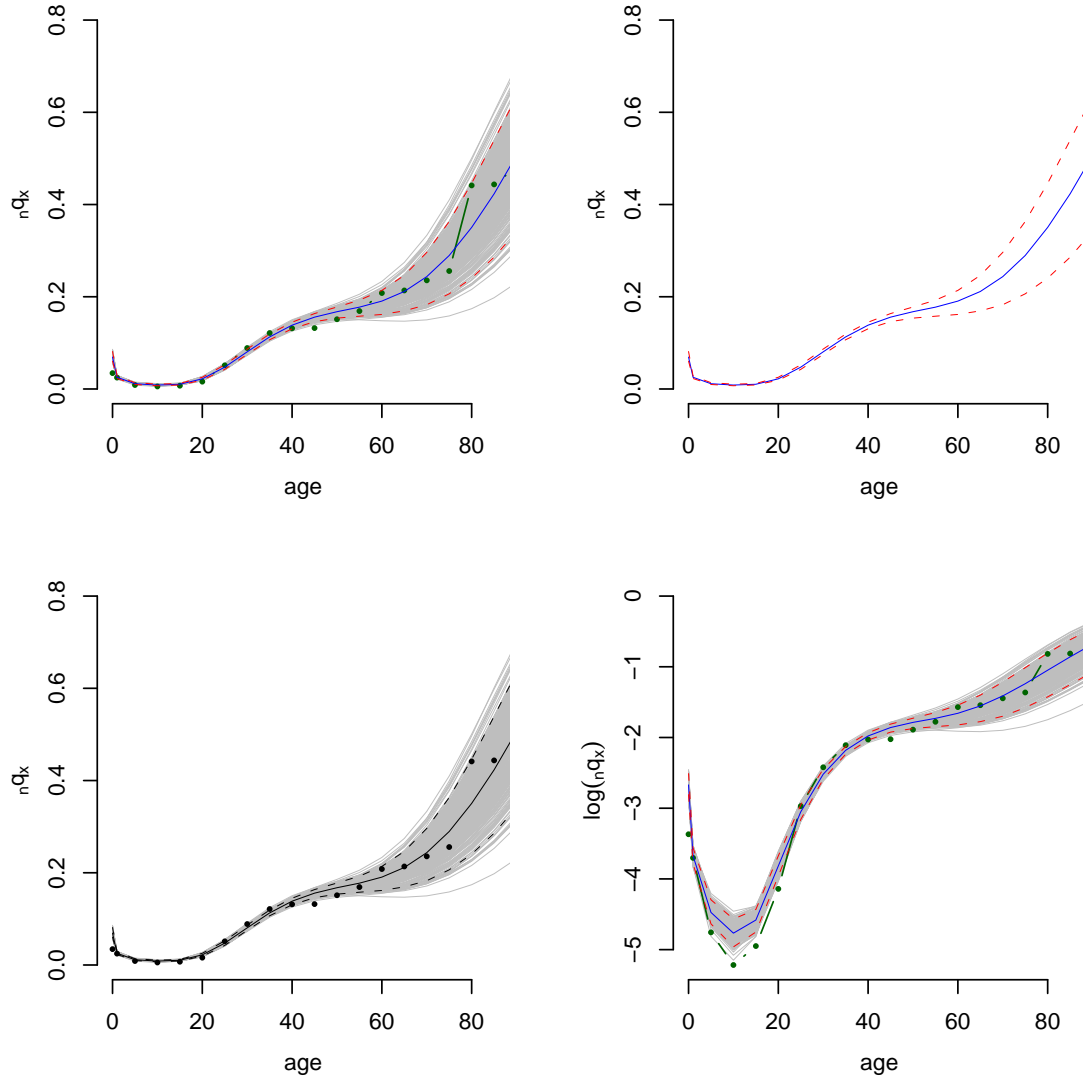


Figure 2: Resulting plots from `hpbayes.plot`.

5 Other useful functions

This section describes two useful functions once parameter estimates have been obtained. The first simply converts the posterior distribution of HP parameters into a set of output nq_x values while the second takes a set of parameter estimates and generates a life table.

5.1 Generating nq_x values from the posterior parameter distribution

The function `hp.nqx` takes as arguments the posterior parameter distribution (`result$H.final`) and generates the age-specific nq_x resulting from that set of parameters. In the example below, age has 22 categories and thus the 22 columns in `hpq`. The length of each column depends on the number of rows in `result$H.final`.

```
> hpq <- hp.nqx(H.out=result$H.final, age=age)
> summary(hpq)
```

V1		V2		V3	
Min.	:0.05658	Min.	:0.02153	Min.	:0.008133
1st Qu.	:0.06660	1st Qu.	:0.02449	1st Qu.	:0.010748
Median	:0.06924	Median	:0.02541	Median	:0.011405
Mean	:0.06965	Mean	:0.02549	Mean	:0.011463
3rd Qu.	:0.07287	3rd Qu.	:0.02646	3rd Qu.	:0.012157
Max.	:0.08607	Max.	:0.02954	Max.	:0.014914

V4		V5		V6	
Min.	:0.005801	Min.	:0.008099	Min.	:0.01773
1st Qu.	:0.007920	1st Qu.	:0.009674	1st Qu.	:0.02101
Median	:0.008507	Median	:0.010239	Median	:0.02200
Mean	:0.008539	Mean	:0.010249	Mean	:0.02207
3rd Qu.	:0.009130	3rd Qu.	:0.010793	3rd Qu.	:0.02309
Max.	:0.011599	Max.	:0.012530	Max.	:0.02711

V7		V8		V9		V10	
Min.	:0.04109	Min.	:0.07274	Min.	:0.1037	Min.	:0.1256
1st Qu.	:0.04562	1st Qu.	:0.07832	1st Qu.	:0.1106	1st Qu.	:0.1357
Median	:0.04729	Median	:0.08051	Median	:0.1129	Median	:0.1384
Mean	:0.04735	Mean	:0.08061	Mean	:0.1129	Mean	:0.1382
3rd Qu.	:0.04904	3rd Qu.	:0.08265	3rd Qu.	:0.1151	3rd Qu.	:0.1406
Max.	:0.05508	Max.	:0.09051	Max.	:0.1243	Max.	:0.1501

V11		V12		V13		V14	
Min.	:0.1392	Min.	:0.1466	Min.	:0.1500	Min.	:0.1479
1st Qu.	:0.1526	1st Qu.	:0.1630	1st Qu.	:0.1706	1st Qu.	:0.1805
Median	:0.1557	Median	:0.1676	Median	:0.1774	Median	:0.1906
Mean	:0.1554	Mean	:0.1669	Mean	:0.1769	Mean	:0.1902
3rd Qu.	:0.1587	3rd Qu.	:0.1713	3rd Qu.	:0.1833	3rd Qu.	:0.1996
Max.	:0.1678	Max.	:0.1857	Max.	:0.2054	Max.	:0.2332

V15		V16		V17		V18	
Min.	:0.1470	Min.	:0.1499	Min.	:0.1584	Min.	:0.1741
1st Qu.	:0.1966	1st Qu.	:0.2225	1st Qu.	:0.2601	1st Qu.	:0.3105
Median	:0.2114	Median	:0.2436	Median	:0.2894	Median	:0.3504
Mean	:0.2110	Mean	:0.2431	Mean	:0.2890	Mean	:0.3493

3rd Qu.:0.2253	3rd Qu.:0.2642	3rd Qu.:0.3189	3rd Qu.:0.3893
Max. :0.2744	Max. :0.3331	Max. :0.4097	Max. :0.5007
V19	V20	V21	V22
Min. :0.1979	Min. :0.2304	Min. :0.2720	Min. :0.3222
1st Qu.:0.3741	1st Qu.:0.4486	1st Qu.:0.5303	1st Qu.:0.6121
Median :0.4232	Median :0.5067	Median :0.5935	Median :0.6764
Mean :0.4222	Mean :0.5035	Mean :0.5876	Mean :0.6684
3rd Qu.:0.4724	3rd Qu.:0.5615	3rd Qu.:0.6505	3rd Qu.:0.7318
Max. :0.6012	Max. :0.7012	Max. :0.7876	Max. :0.8556

5.2 Generating an unabridged life table

Likewise we can take the ${}_nq_x$ values from the posterior output (specifically we use the median of each age vector) and generate a life table from those values using the function `hp.lifetab`. This function takes as arguments `age`, a vector of the ages (specifically the beginning of each age interval in the table) at which to calculate the various life table columns, `l0`, a scalar defining the radix of the life table, `hpp`, the posterior parameter distribution, and `nax`, which is a necessary set of ${}_na_x$ values to compute the life table columns. One can also generate two additional life tables whose ${}_nq_x$ values are the upper and lower credible interval bounds. The arguments, `ciw` (a scalar specifying the percent confidence, e.g. `ciw=95` creates a 95 percent credible interval) and `with.CI` (a logical indicating whether to produce the upper and lower CI tables), once specified, will generate the two additional tables.

```
> agelt <- c(0, 1, seq(5, 100, 5))
> nax <- c(0.5, 2.0, rep(2.5, length(agelt)-2))
> demolt <- hp.lifetab(hpp=result$H.final, age=agelt, nax=nax, l0=100000,
with.CI=TRUE, CI=95)
> names(demolt)
[1] "lt"      "lt.lo"   "lt.hi"
> demolt
$lt
      Age nax  nqx  npx  ndx    lx   nLx    Tx   ex
[1,]   0 0.5 0.0692 0.9308 6924 100000  96538 4928234 49.28
[2,]   1 2.0 0.0254 0.9746 2365  93076 367574 4831696 51.91
[3,]   5 2.5 0.0114 0.9886 1035  90711 450968 4464122 49.21
[4,]  10 2.5 0.0085 0.9915  763  89676 446472 4013155 44.75
[5,]  15 2.5 0.0102 0.9898  910  88913 442290 3566682 40.11
[6,]  20 2.5 0.0220 0.9780 1936  88003 435175 3124392 35.50
[7,]  25 2.5 0.0473 0.9527 4070  86067 420160 2689218 31.25
[8,]  30 2.5 0.0805 0.9195 6602  81997 393480 2269058 27.67
[9,]  35 2.5 0.1129 0.8871 8511  75395 355698 1875578 24.88
[10,] 40 2.5 0.1384 0.8616 9254  66884 311285 1519880 22.72
[11,] 45 2.5 0.1557 0.8443 8975  57630 265712 1208595 20.97
```

[12,]	50	2.5	0.1676	0.8324	8153	48655	222892	942882	19.38
[13,]	55	2.5	0.1774	0.8226	7185	40502	184548	719990	17.78
[14,]	60	2.5	0.1906	0.8094	6350	33317	150710	535442	16.07
[15,]	65	2.5	0.2114	0.7886	5700	26967	120585	384732	14.27
[16,]	70	2.5	0.2436	0.7564	5181	21267	93382	264148	12.42
[17,]	75	2.5	0.2894	0.7106	4656	16086	68790	170765	10.62
[18,]	80	2.5	0.3504	0.6496	4005	11430	47138	101975	8.92
[19,]	85	2.5	0.4232	0.5768	3142	7425	29270	54838	7.39
[20,]	90	2.5	0.5067	0.4933	2170	4283	15990	25568	5.97
[21,]	95	2.5	0.5935	0.4065	1254	2113	7430	9578	4.53
[22,]	100	2.5	1.0000	0.0000	859	859	2148	2148	2.50

\$lt.lo

	Age	nax	nqx.lo	npx.lo	ndx.lo	lx.lo	nLx.lo	Tx.lo	ex.lo
[1,]	0	0.5	0.0603	0.9397	6034	100000	96983	5235380	52.35
[2,]	1	2.0	0.0227	0.9773	2135	93966	371594	5138396	54.68
[3,]	5	2.5	0.0097	0.9903	891	91831	456928	4766802	51.91
[4,]	10	2.5	0.0070	0.9930	637	90940	453108	4309875	47.39
[5,]	15	2.5	0.0086	0.9914	781	90303	449562	3856768	42.71
[6,]	20	2.5	0.0191	0.9809	1707	89522	443342	3407205	38.06
[7,]	25	2.5	0.0424	0.9576	3727	87815	429758	2963862	33.75
[8,]	30	2.5	0.0750	0.9250	6305	84088	404678	2534105	30.14
[9,]	35	2.5	0.1067	0.8933	8302	77783	368160	2129428	27.38
[10,]	40	2.5	0.1305	0.8695	9066	69481	324740	1761268	25.35
[11,]	45	2.5	0.1460	0.8540	8822	60415	280020	1436528	23.78
[12,]	50	2.5	0.1536	0.8464	7925	51593	238152	1156508	22.42
[13,]	55	2.5	0.1578	0.8422	6892	43668	201110	918355	21.03
[14,]	60	2.5	0.1616	0.8384	5944	36776	169020	717245	19.50
[15,]	65	2.5	0.1694	0.8306	5222	30832	141105	548225	17.78
[16,]	70	2.5	0.1827	0.8173	4680	25610	116350	407120	15.90
[17,]	75	2.5	0.2063	0.7937	4319	20930	93852	290770	13.89
[18,]	80	2.5	0.2404	0.7596	3992	16611	73075	196918	11.85
[19,]	85	2.5	0.2857	0.7143	3605	12619	54082	123842	9.81
[20,]	90	2.5	0.3419	0.6581	3082	9014	37365	69760	7.74
[21,]	95	2.5	0.4077	0.5923	2419	5932	23612	32395	5.46
[22,]	100	2.5	1.0000	0.0000	3513	3513	8782	8782	2.50

\$lt.hi

	Age	nax	nqx.hi	npx.hi	ndx.hi	lx.hi	nLx.hi	Tx.hi	ex.hi
[1,]	0	0.5	0.0812	0.9188	8123	100000	95938	4681496	46.81
[2,]	1	2.0	0.0285	0.9715	2615	91877	362278	4585558	49.91
[3,]	5	2.5	0.0136	0.9864	1217	89262	443268	4223280	47.31

[4,]	10	2.5	0.0104	0.9896	914	88045	437940	3780012	42.93
[5,]	15	2.5	0.0119	0.9881	1036	87131	433065	3342072	38.36
[6,]	20	2.5	0.0254	0.9746	2184	86095	425015	2909008	33.79
[7,]	25	2.5	0.0523	0.9477	4389	83911	408582	2483992	29.60
[8,]	30	2.5	0.0865	0.9135	6881	79522	380408	2075410	26.10
[9,]	35	2.5	0.1191	0.8809	8648	72641	341585	1695002	23.33
[10,]	40	2.5	0.1448	0.8552	9265	63993	296802	1353418	21.15
[11,]	45	2.5	0.1635	0.8365	8949	54728	251268	1056615	19.31
[12,]	50	2.5	0.1777	0.8223	8137	45779	208552	805348	17.59
[13,]	55	2.5	0.1926	0.8074	7250	37642	170085	596795	15.85
[14,]	60	2.5	0.2138	0.7862	6497	30392	135718	426710	14.04
[15,]	65	2.5	0.2464	0.7536	5887	23895	104758	290992	12.18
[16,]	70	2.5	0.2961	0.7039	5333	18008	76708	186235	10.34
[17,]	75	2.5	0.3631	0.6369	4602	12675	51870	109528	8.64
[18,]	80	2.5	0.4461	0.5539	3601	8073	31362	57658	7.14
[19,]	85	2.5	0.5393	0.4607	2412	4472	16330	26295	5.88
[20,]	90	2.5	0.6343	0.3657	1306	2060	7035	9965	4.84
[21,]	95	2.5	0.7228	0.2772	545	754	2408	2930	3.89
[22,]	100	2.5	1.0000	0.0000	209	209	522	522	2.50

References

- Heligman, Larry and John H. Pollard. 1980. “The Age Pattern of Mortality.” *Journal of the Institute of Actuaries* 107:49–80.
- Poole, David and Adrian Raftery. 2000. “Inference for Deterministic Simulation Models: The Bayesian Melding Approach.” *Journal of the American Statistical Association* 95:1244–1255.
- R Development Core Team. 2009. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. `optim()`.
- Raftery, Adrian and Le Bao. 2009. “Estimating and Projecting Trends in HIV/AIDS Generalized Epidemics Using Incremental Mixture Importance Sampling.” Technical Report 560, Department of Statistics, University of Washington.
- Sharrow, David J. and Samuel J. Clark. ND. “A Parametric Investigation of Mortality at All Ages in a Rural, South African Population.” working paper.