Fitting a TOPALS mortality model by Newton-Raphson

Carl Schmertmann

28 Feb 2018

TOPALS Mortality Schedule

The TOPALS model for a mortality schedule over A single-year ages $0 \dots (A-1)$ is

$$\boldsymbol{\lambda} = \boldsymbol{\lambda}^* + \boldsymbol{B}\boldsymbol{\alpha}$$

where $\boldsymbol{\lambda}$ is an $A \times 1$ vector of age-specific log mortality rates, $\boldsymbol{\lambda}^*$ s a vector with fixed constants representing a standard schedule, $\boldsymbol{B}$ is a $A \times 7$ matrix of linear B-spline constants, and $\boldsymbol{\alpha}$ is a 7-dimensional vector representing deviations from the standard log mortality schedule at specified ages.

In this model the log mortality rate at age $x \in \{0 \dots (A-1)\}$ is

$$\lambda_x = \ln \mu_x = \lambda^*_x + \boldsymbol{b}'_x \boldsymbol{\alpha}$$

the mortality rate is

$$\mu_x = exp\left(\lambda^*_x + \boldsymbol{b}'_x \boldsymbol{\alpha}\right)$$

where $\boldsymbol{b}'_x$ is the 1×7 row of B that corresponds to age x. The derivatives with respect to TOPALS parameters $\boldsymbol{\alpha}$ are

$$\frac{\partial \ln \mu_x}{\partial \boldsymbol{\alpha}} = \boldsymbol{b}_x \quad , \quad \frac{\partial \mu_x}{\partial \boldsymbol{\alpha}} = \mu_x \, \boldsymbol{b}_x$$

Sample Data

The observed data consists of age-specific exposure $N_0 \dots N_{A-1}$ and age-specific death counts $D_0 \dots D_{A-1}$.

Poisson Log Likelihood

We assume that deaths at each age have a Poisson distribution with expected value equal to the observed exposure N_x times the mortality rate:

$$D_x \sim Poisson\left(N_x \, \mu_x\right)$$

Thus the sample log likelihood for $\boldsymbol{\alpha}$ is

$$\begin{aligned} \ln P(\boldsymbol{\alpha}) &= c + \sum_{x=0}^{A-1} \left(D_x \ln \mu_x - N_x \mu_x \right) \\ &= c + \sum_{x=0}^{A-1} \left(D_x \ln \mu_x - \hat{D}_x \right) \end{aligned}$$

where we use $\hat{D}_x = N_x \mu_x$ to represent the expected number of deaths in the fitted model, remembering that it is a function of $\boldsymbol{\alpha}$.

Penalized Log Likelihood

To stabilize estimates in small populations with few deaths, we add a small penalty term to the log likehood:

$$\begin{aligned} Q(\boldsymbol{\alpha}) &= \ln P(\boldsymbol{\alpha}) - \text{penalty}(\boldsymbol{\alpha}) \\ &= c + \sum_{x=0}^{A-1} \left(D_x \ln \mu_x - \hat{D}_x \right) - \boldsymbol{\alpha}' \boldsymbol{S}' \boldsymbol{S} \boldsymbol{\alpha} \end{aligned}$$

where $\boldsymbol{S}$ is a 6×7 differencing matrix with 1s on the main diagonal, -1s on the first superdiagonal, and zeroes elsewhere This penalizes squared differences between consecutive α parameters, saying that *a priori* we think it's more likely that they are similar (i.e., that the log mortality schedule is most likely to look like an up-and-down vertical shift of the standard schedule).

Maximizing the Penalized Log Likelihood via Newton-Raphson iteration

We select $\boldsymbol{\alpha}$ to maximize Q. This requires setting a vector of derivatives equal to zero: $\frac{\partial Q}{\partial \boldsymbol{\alpha}} = 0 \in \mathbb{R}^7$. Using $\frac{\partial \ln \mu_x}{\partial \boldsymbol{\alpha}} = \boldsymbol{b}_x$ and $\frac{\partial \hat{D}_x}{\partial \boldsymbol{\alpha}} = N_x \frac{\partial \mu_x}{\partial \boldsymbol{\alpha}} = N_x \mu_x \boldsymbol{b}_x = \hat{D}_x \boldsymbol{b}_x$, this is

$$\frac{\partial Q}{\partial \boldsymbol{\alpha}} = \sum_{x=0}^{A-1} \left(D_x - \hat{D}_x \right) \boldsymbol{b}_x - 2\, \boldsymbol{S}' \boldsymbol{S}\, \boldsymbol{\alpha} = 0$$

or more compactly

$$\frac{\partial Q}{\partial \boldsymbol{\alpha}} = \boldsymbol{B}' \left(\boldsymbol{D} - \hat{\boldsymbol{D}} \right) - 2\, \boldsymbol{S}' \boldsymbol{S}\, \boldsymbol{\alpha} = 0$$

where $\boldsymbol{D}$ and $\hat{\boldsymbol{D}}$ are $A \times 1$ vectors of observed and predicted deaths by age.

For Newton-Raphson iteration to solve this equation, we also need the second derivatives of Q:

$$\begin{aligned} \frac{\partial^2 Q}{\partial \boldsymbol{\alpha} \boldsymbol{\alpha}'} &= \sum_{x=0}^{A-1} \left(-\frac{\partial \hat{D}_x}{\partial \boldsymbol{\alpha}'}\ \boldsymbol{b}_x \right) - 2\, \boldsymbol{S}' \boldsymbol{S} \\ &= \sum_{x=0}^{A-1} \left(-\boldsymbol{b}_x'\, \hat{D}_x\, \boldsymbol{b}_x \right) - 2\, \boldsymbol{S}' \boldsymbol{S} \\ &= -\boldsymbol{B}' \left(diag(\hat{\boldsymbol{D}}) \right) \boldsymbol{B} - 2\, \boldsymbol{S}' \boldsymbol{S} \end{aligned}$$

The basic Newton-Raphson method for solving $\frac{\partial Q}{\partial \boldsymbol{\alpha}} = 0$ is

$$\boldsymbol{\alpha}_{i+1} = \boldsymbol{\alpha}_i - \left[\frac{\partial^2 Q}{\partial \boldsymbol{\alpha} \boldsymbol{\alpha}'} \right]^{-1} \left[\frac{\partial Q}{\partial \boldsymbol{\alpha}} \right]$$

which in the TOPALS case is

$$\boldsymbol{\alpha}_{i+1} = \boldsymbol{\alpha}_i + \left[\boldsymbol{B}' \left(diag(\hat{\boldsymbol{D}}) \right) \boldsymbol{B} + 2\, \boldsymbol{S}' \boldsymbol{S} \right]^{-1} \left[\boldsymbol{B}' \left(\boldsymbol{D} - \hat{\boldsymbol{D}} \right) - 2\, \boldsymbol{S}' \boldsymbol{S}\, \boldsymbol{\alpha} \right]$$

where we begin with $\boldsymbol{\alpha}_0 = \boldsymbol{0}$ and iterate to convergence. This procedure is faster and more precise than using a general optimizer like *optim( )*.

Example: Fitting a schedule to a fictional sample from a small, Estonian-like population

Suppose that the population of interest has 5000 total females, with the age distribution in the plot below. This is similar to the age structure of Estonian females in 2010.

```r
N = c(62, 62, 50, 65, 56, 56, 40, 50, 43, 50,
42, 39, 34, 43, 45, 42, 53, 42, 45, 72, 66, 65, 63,
67, 64, 78, 65, 69, 65, 60, 70, 57, 46, 64, 58, 62,
59, 69, 69, 76, 69, 56, 58, 61, 50, 52, 79, 65, 75,
78, 73, 62, 76, 63, 83, 63, 61, 77, 84, 67, 72, 62,
60, 60, 50, 55, 37, 48, 75, 51, 59, 66, 71, 45, 46,
45, 44, 58, 50, 40, 34, 42, 27, 35, 25, 25, 25, 18,
16, 5, 2, 7, 2, 0, 5, 1, 1, 0, 1, 0)

age = 0:99

plot(age, N, type='h', main = "Exposure by Age")
```

We'll also suppose that the true mortality rates by age are those in the mx column of the Estonia 2010 female life table from the Human Mortality Database.

```r
mu = c(0.00246, 0.00064, 0.00026, 0.00014, 0.00014, 0.00059, 0.00031,
0.00016, 0, 0, 0.00034, 0, 0.00018, 0, 0.00016, 0.00047, 0.00015,
0.00013, 0, 0.00023, 0.00021, 2e-04, 1e-04, 0.00021, 0.00053,
0.00073, 0.00021, 0.00085, 0.00033, 0.00033, 0.00045, 0.00079,
0.00034, 0.00045, 0.00077, 0.00066, 0.00077, 0.00099, 0.00074,
0.00096, 0.00127, 0.00099, 0.00103, 7e-04, 0.0022, 0.00177, 0.00263,
0.00247, 0.00171, 0.00189, 0.00408, 0.00297, 0.00326, 0.00285,
0.00402, 0.00441, 0.00584, 0.00485, 0.00475, 0.00484, 0.0074,
0.00782, 0.00777, 0.01002, 0.0108, 0.01277, 0.01026, 0.0135,
0.01316, 0.01467, 0.01353, 0.01839, 0.02011, 0.02176, 0.02507,
```

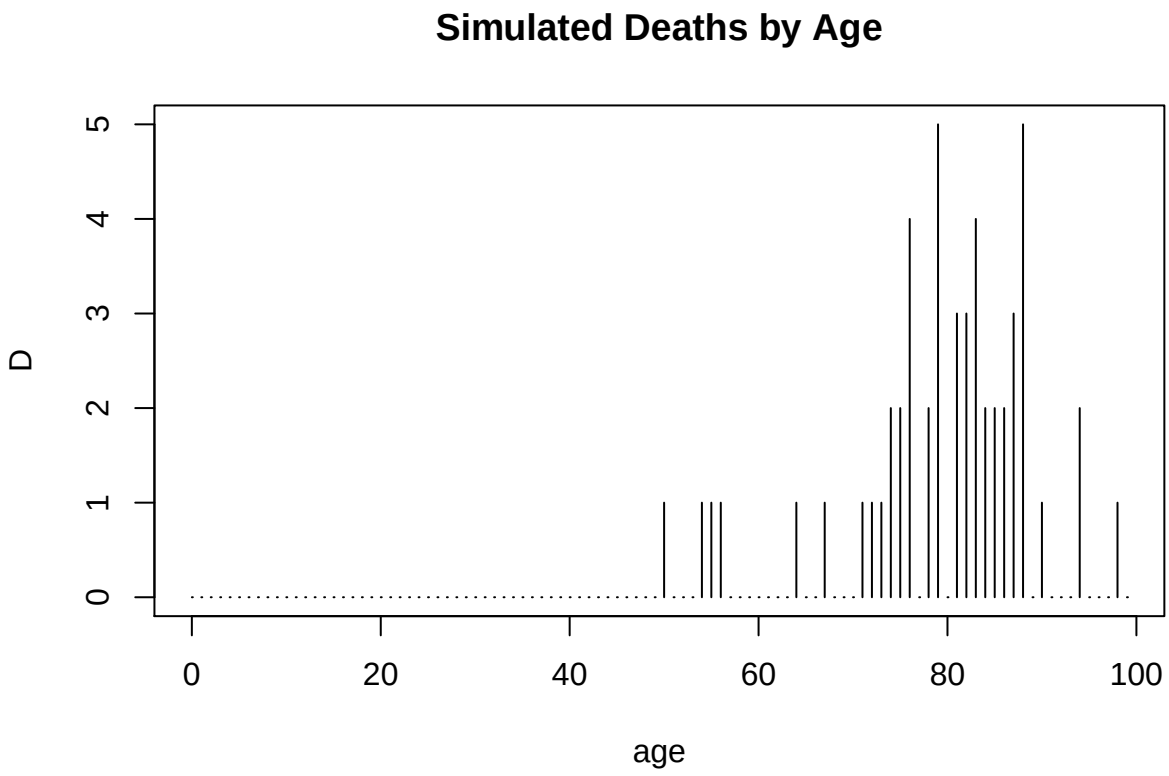
```
0.02801, 0.03008, 0.03849, 0.04071, 0.0516, 0.05487, 0.06088,
0.06675, 0.07599, 0.08657, 0.09597, 0.12556, 0.11733, 0.14262,
0.1601, 0.17928, 0.2002, 0.2229, 0.24739, 0.27361, 0.3015, 0.33094,
0.36176, 0.39377, 0.42671)

plot(age, log(mu), pch=16, main='True Log Mortality Rates by Age\nEstonia 2010')
```


```
# trapez approx of life expectancy from a logmx schedule over ages 0..99
e0 = function(logmx) {
  mx = exp(logmx)
  px = exp(-mx)
  lx = c(1,cumprod(px))
  return( sum(head(lx,-1) + tail(lx,-1)) / 2)
}
```

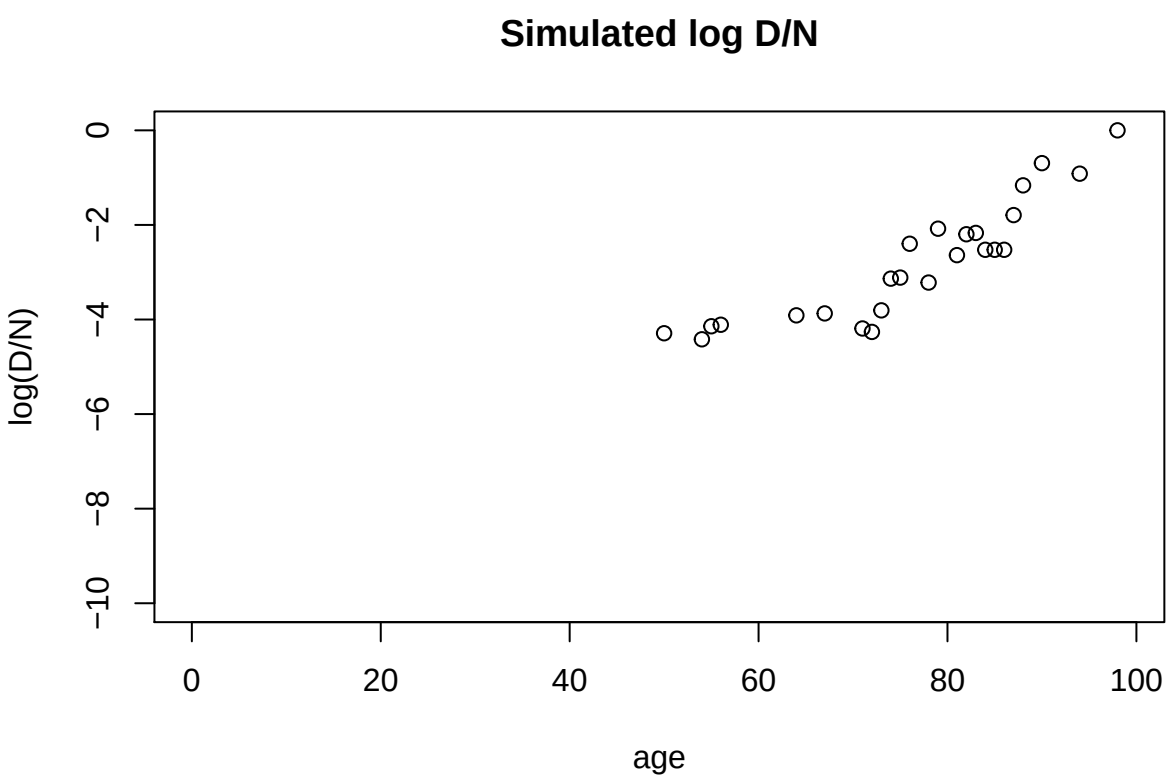
The true life expectancy in this case is **80.54**.

Using this N_x and μ_x data, one realization of deaths $\{D_x\}$ from the Poisson distribution $D_x \sim Pois(N_x \mu_x)$ is

```
set.seed(6447100)    # change this if you want a different random dataset

D = rpois(100, N*mu)

plot( age, D, type='h', main='Simulated Deaths by Age')
```

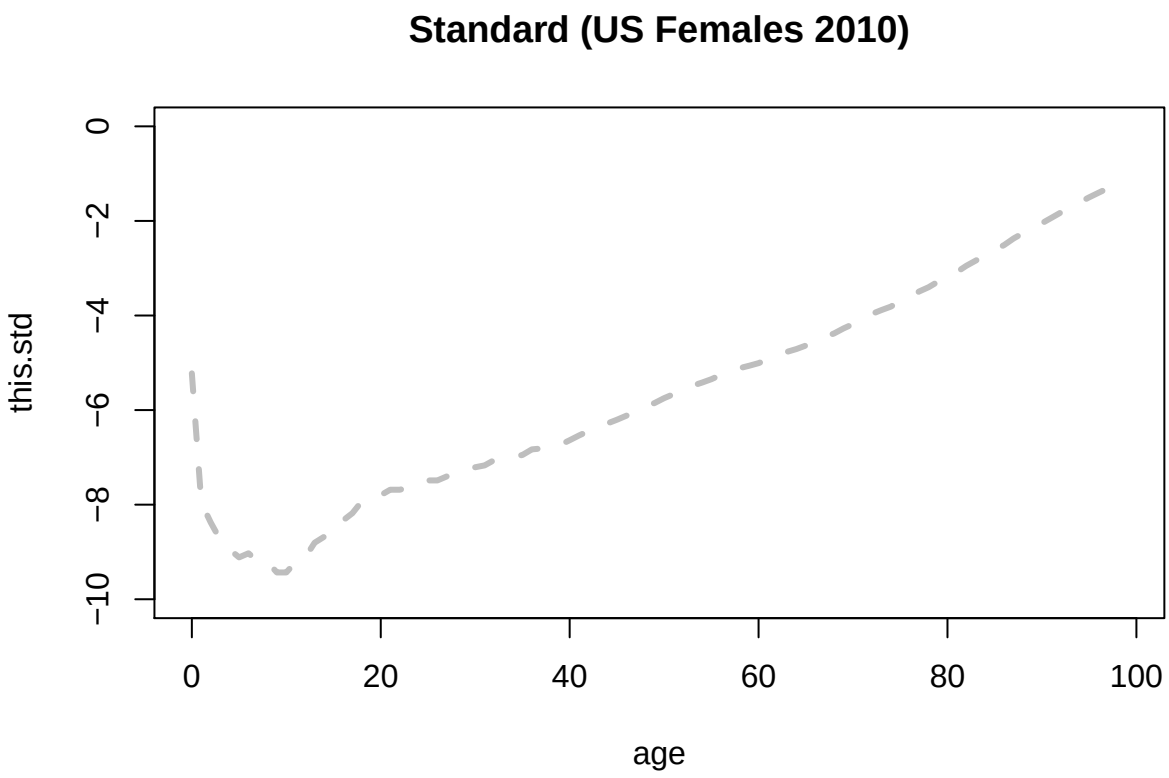



```
plot(age, log(D/N), pch=1, ylim=c(-10,0), main='Simulated log D/N')
```



For this data we can estimate TOPALS parameters for offsets α from a specified standard. We'll arbitrarily use the US 2015 female schedule as a standard.

```
this.std = c(-5.2232, -7.9576, -8.3774, -8.7403, -8.948, -9.115, -9.028,
-9.2103, -9.2103, -9.4335, -9.4335, -9.2103, -9.115, -8.8049,
-8.6797, -8.6226, -8.3349, -8.1807, -7.9294, -7.824, -7.7994,
-7.6843, -7.6843, -7.6417, -7.5811, -7.4876, -7.4876, -7.4021,
-7.354, -7.2934, -7.2089, -7.1691, -7.0586, -7.0243, -6.9911,
-6.9486, -6.8308, -6.8124, -6.7338, -6.7338, -6.6377, -6.5362,
-6.444, -6.383, -6.2818, -6.2047, -6.1193, -6.0407, -5.9257,
-5.85, -5.7477, -5.6636, -5.5649, -5.4846, -5.4194, -5.3475,
-5.2572, -5.1832, -5.1127, -5.0625, -5.0071, -4.9281, -4.8422,
-4.7689, -4.7094, -4.6356, -4.5497, -4.4542, -4.3788, -4.2723,
-4.1819, -4.0757, -3.966, -3.8859, -3.8126, -3.6977, -3.6071,
-3.4917, -3.4016, -3.2834, -3.1696, -3.0791, -2.9481, -2.8382,
-2.7308, -2.614, -2.5092, -2.371, -2.2583, -2.167, -2.0485, -1.935,
-1.8211, -1.6996, -1.6052, -1.5011, -1.4032, -1.3082, -1.2165,
-1.1282)

plot(age, this.std, type='l', ylim=c(-10,0), lty=2, col='grey',
     lwd=3, main='Standard (US Females 2010)')
```



We can now write a function to estimate the TOPALS parameters $\boldsymbol{\alpha}$that maximize the penalized likelihood.

```
library(splines)
B  = bs( age, knots=c(0,1,10,20,40,70), degree=1)  # linear B-spline basis

## penalized log lik function
Q = function(alpha) {
  logmx.hat = as.numeric( this.std + B %*% alpha)
  penalty = sum(diff(alpha)^2)
  return( sum(D * logmx.hat - N * exp(logmx.hat)) - penalty)
}

## expected deaths function
Dhat = function(alpha) {
  lambda.hat = this.std + B %*% alpha
  return(  as.numeric( N * exp(lambda.hat) ))
}

## S matrix for penalty
S = matrix(0,6,7)
diag(S[,1:6]) = -1
diag(S[,2:7]) = +1
SS = crossprod(S)

print(S)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
```

```
## [1,]   -1    1    0    0    0    0    0
## [2,]    0   -1    1    0    0    0    0
## [3,]    0    0   -1    1    0    0    0
## [4,]    0    0    0   -1    1    0    0
## [5,]    0    0    0    0   -1    1    0
## [6,]    0    0    0    0    0   -1    1
```

```
print(SS)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,]    1   -1    0    0    0    0    0
## [2,]   -1    2   -1    0    0    0    0
## [3,]    0   -1    2   -1    0    0    0
## [4,]    0    0   -1    2   -1    0    0
## [5,]    0    0    0   -1    2   -1    0
## [6,]    0    0    0    0   -1    2   -1
## [7,]    0    0    0    0    0   -1    1
```

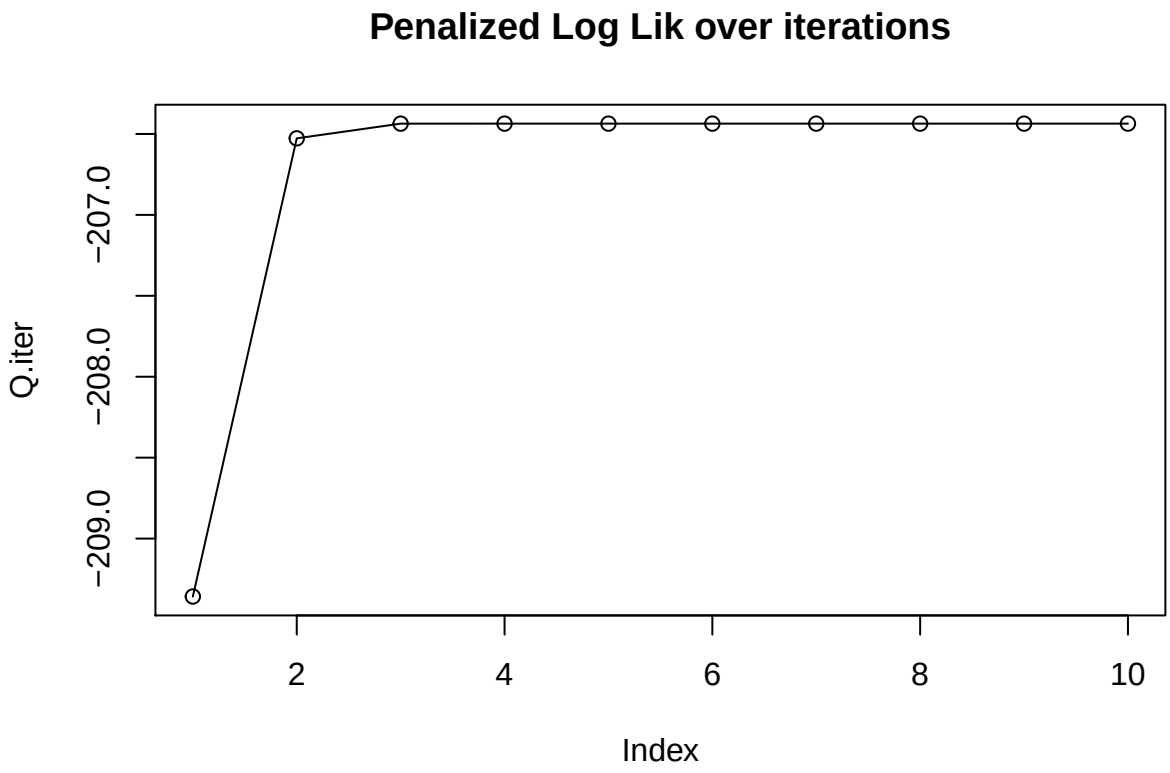
```
#-----------------------------------------------
# iteration function:
# next alpha vector as a function of current alpha
#-----------------------------------------------

next_alpha = function(alpha) {
  dhat = Dhat(alpha)
  M = solve ( t(B) %*% diag(dhat) %*% B + 2*SS)
  v = t(B) %*% (D - dhat) - 2* (SS %*% alpha)
  return( alpha + M %*% v)
}
```

The alpha value converges very quickly from an initial zero vector, usually within 5 iterations.

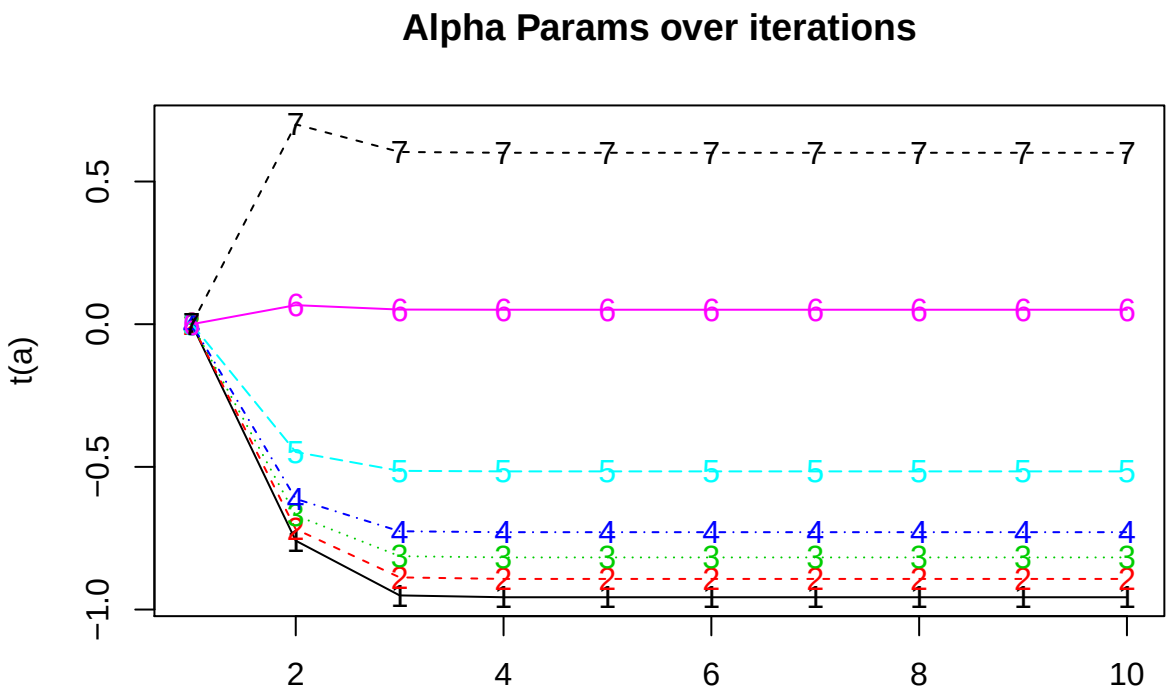
```
# MAIN ITERATION, starting at alpha=0

a = matrix(0, 7, 10)

for (i in 2:ncol(a)) { a[,i] = next_alpha(a[,i-1])}

round(a, 4)
```

```
##      [,1]    [,2]    [,3]    [,4]    [,5]    [,6]    [,7]    [,8]    [,9]
## [1,]    0 -0.7586 -0.9505 -0.9568 -0.9568 -0.9568 -0.9568 -0.9568 -0.9568
## [2,]    0 -0.7183 -0.8872 -0.8927 -0.8927 -0.8927 -0.8927 -0.8927 -0.8927
## [3,]    0 -0.6702 -0.8130 -0.8174 -0.8174 -0.8174 -0.8174 -0.8174 -0.8174
## [4,]    0 -0.6121 -0.7256 -0.7289 -0.7289 -0.7289 -0.7289 -0.7289 -0.7289
## [5,]    0 -0.4480 -0.5140 -0.5158 -0.5158 -0.5158 -0.5158 -0.5158 -0.5158
## [6,]    0  0.0666  0.0513  0.0507  0.0507  0.0507  0.0507  0.0507  0.0507
## [7,]    0  0.7003  0.6035  0.6008  0.6008  0.6008  0.6008  0.6008  0.6008
##        [,10]
## [1,] -0.9568
## [2,] -0.8927
## [3,] -0.8174
## [4,] -0.7289
## [5,] -0.5158
## [6,]  0.0507
## [7,]  0.6008
```

```r
Q.iter = apply(a,2, Q)

plot(Q.iter, type='o', main='Penalized Log Lik over iterations')
```



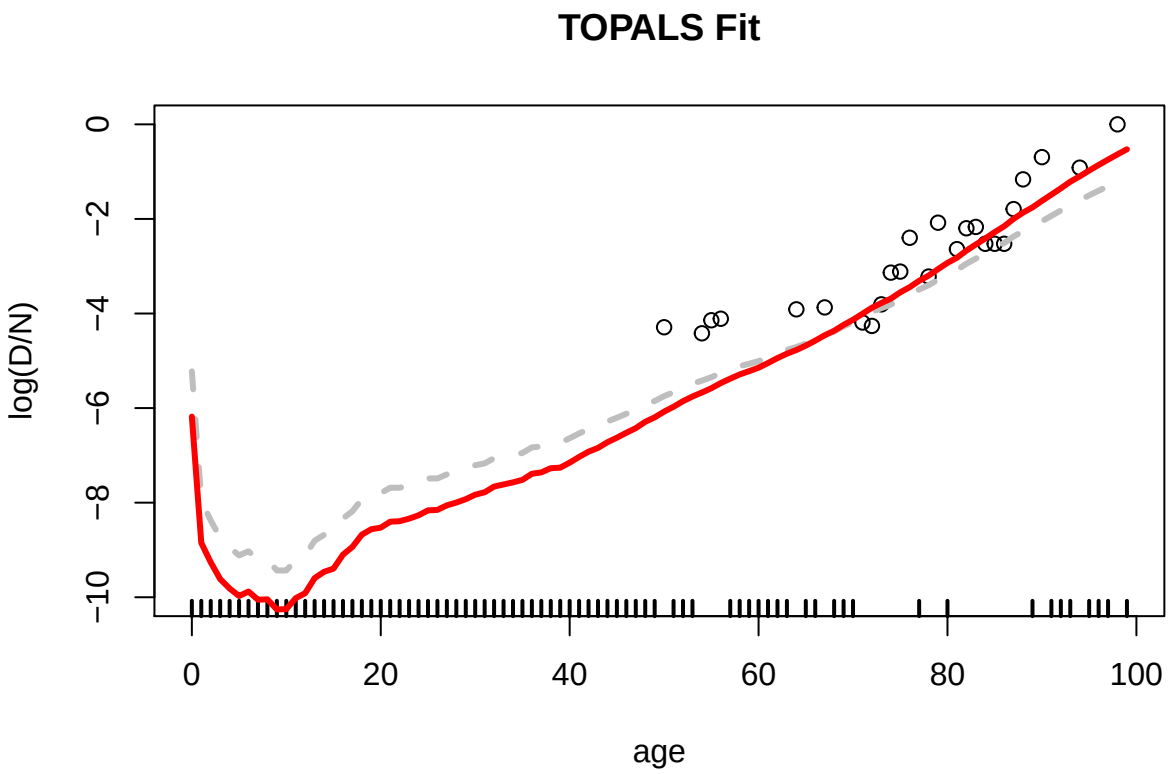

```r
matplot(  t(a), type='o',main= 'Alpha Params over iterations')
```




```
fitted.logmx = this.std + B %*% a[,ncol(a)]
```

The final fit in this sample is $\alpha = (-0.957, -0.893, -0.817, -0.729, -0.516, 0.051, 0.601)$, with an estimated life expectancy of **81.18**.

```
plot( age, log(D/N), ylim=c(-10,0), main='TOPALS Fit')
rug( age[D==0], lwd=2)
lines(age, this.std, type='l', lty=2, col='grey', lwd=3)
lines(age, fitted.logmx, lty=1, col='red', lwd=3)
```


TOPALS Fit
log(D/N)
0
−2
−4
−6
−8
−10
0
20
40
60
80
100
age