Supplemental Material

ReadMe file: A user's guide for the vitality model with initial Gaussian distribution parameter fitting routine and the R functions contained in file vitality.gaussian.R.

Ting Li and James J. Anderson
University of Washington
June 11, 2009

The supplemental R file vitality.gaussian.R (also available at www.cbr.washington.edu/vitality/) contains all functions necessary to run the MLE parameter fitting routine for the vitality-based survival model with an initial Gaussian distribution. The functions are written in the R programming language. Most of the functions are very similar to those written by Salinger et al. (2003) in an S-plus file "VitalityModelFitting.ssc" for fitting the vitality model with a Dirac delta function initial distribution. The fitting routines differ in how they derive initial parameter estimates for the Newton-Ralphson method.

Because the target function has many local minimums, looking for starting values is always an important issue when using the Newton-Ralphson optimal algorithm. Unlike the method of Salinger et al. (2003), by establishing relationships between initial values, this new routine relies upon a search approach: Simulated annealing. The algorithm allows a probability of jumping out local traps and finally stabilizes around the global optimal solution. However, since the method doesn't give exactly the same answers every time, the program only uses its results as the initial values of Newton Ralphson algorithm. The simulated annealing is time consuming; it usually takes 2-30 minutes to run one search depending on the data size for an Intel® 1.7G processor. For more information on simulated annealing, the fitting routine and the model details, see Kirkpatrick et al. (1983), Salinger et al. (2003) and Anderson et al. (2000, 2008).

This ReadMe file contains:

1. The header for the main function vitality.sa. The header explains all user options and provides some example function calls.
2. A list of subordinate functions with brief descriptions.
3. Tips of using the user options and data.

1. Header for the function vitality.sa:

```
vitality.sa <- function(
time,sdata,init.params=F,rc.data=F,se=F,gfit=F,AIC=F,datatype="CUM",ttol=.000001,
pplot=T,tlab="years",lplot=T,cplot=F,silent=F,L=2)
{
#
#  Vitality based survival model: parameter fitting routine:    VERSION: 25/08/2008
#
# REQUIRED PARAMETERS:
#    time - time component of data: time from experiment start.  Time should start after the
#         imposition of a stressor is completed.
#    sdata - survival or mortality data.  The default expects cumulative survival fraction.  If
#         providing incremental mortality fraction instead, use option: datatype="INC".
#         The default also expects the data to represent full mortality. Otherwise, use option:
#         rc.data=T to indicate right censored data.
#
# OPTIONAL PARAMETERS:
#     init.params =F  has default values (default: =F) for initial parameter r,s,k,u. r is usually
#         taken as 1/(the time of 50% survival) and s, k and u are set to be 0.1, 0.01 and 0.1
#         separately. These initial parameters are the starting values for simulated annealing
#         not for the final Newton-Ralphson algorithm. Generally speaking, it does not matter
#         what values they are, however a wise choice of initial values will save the
#         computing time. If you wish to specify initial parameter values rather than have the
#         routine choose them, specify init.params=c(r,s,k,u) in that order (eg.
#         init.params=c(.1,.02,.003,0.1)).
#     rc.data =T  - specifies Right Censored data.   If the data does not represent full mortality,
#         it is probably right censored.  The default is rc.data=F.  A third option is
#         rc.data="TF".  Use this case to add a near-term zero survival point to data which
#         displays nearly full mortality ( <.01 survival at end).  If rc.data=F but the data does
#               not show full mortality, rc.data="TF" will be invoked automatically.
#     se =<population size>  calculates the standard errors for the MLE parameters.
#         Default is se=F.  The initial study population is necessary for computing these
```

```
#         standard errors.
#    gfit =<population size>  provides a Pearson C type test for goodness of fit.
#              Default is gfit=F. The initial study population is necessary for computing goodness
#         of fit.
#    AIC=<population size>  provides AIC value for the model fitting. It is useful when
#         conducting model comparison.
#         Default is AIC=F. The initial study population is necessary for computing AIC.
#    datatype ="CUM" -cumulative survival fraction data- is the default.
#         Other option: datatype="INC" - for incremental mortality fraction data.
#    ttol (stopping criteria tolerance.)  Default is .000001. specify as ttol=.0001.
#         If one of the liklihood plots (esp. for "k") does not look optimal, try decreasing ttol.
#         If the program crashes, try increasing ttol.
#    pplot =T provides plots of cumulative survival and incremental mortality - for both data
#         and fitted curves (default: =T).  pplot=F provides no plotting.  A third option:
#         pplot=n  (n>=1) extends the time axis of the fitting plots (beyond the max time in
#         data).  For example: pplot=1.2 extends the time axis by 20%.  (Note:  the
#         incremental mortality plot is a continuous representation of the appropriately-binned
#         histogram of incremental mortalities.)
#    tlab ="<time units>" specifies units for x-axis of plots.  Default is tlab="years".
#    lplot =T provides likelihood function plotting (default =T).
#         Note:  these plots are not "likelihood profiles" in that while one parameter is varied,
#         the others are held fixed, rather than re-optimized. (must also have pplot=T.)
#    cplot =T provides a likelihood contour plot for a range of r and s values (can be slow so
#         default is F). Must also have lplot=T (and pplot=T) to get contour plots.
#    silent =T stops all print and plot options (still get most warning and all error messages)
#         Default is F.
#    L =1  times of running simulated annealing. Default is 1. There is a chance that the
#         simulated annealing doesn't converge to the true value by one run. Choosing L>1
#         makes increasing chance of getting the right answer, but accordingly consumes more
#         time. If choose L=0, the function will run Newton-Ralphson algorithm directly using
#         the default initial values.

# RETURN:
#    vector of final MLE r, s , k, u parameter estimates.
#    standard errors of MLE parameter estimates (if se=<population> is
#    specified).

# PRINT:
#   print both vectors of initial values and final MLE r, s, k, u parameter estimates.
#   print the minimal value of –log likelihood
#   if silence=F,
```

print s.e. for each parameter (if se=<population> is specified).
print goodness of fit (if gft=<population> is specified)..
print AIC value for the model fitting (if AIC=<population> is specified).

#PLOT:
if silence=F
plots of cumulative survival and incremental mortality - for both data and fitted curves will
be provided (if pplot!=F).
a likelihood function plotting will be provided (if lplot!=F).
a likelihood contour plot for a range of r and s values will be provided (if pplot!=F, lplot!=F
and cplot!=F).

Examples:

a) Basic call, with times and cumulative survival fractions in (time, sdata): vitality.sa (time,sdata).

b) If data is right censored (<100% mortality at completion of study) and if standard errors and AIC are desired (given that study population is 200): vitality.sa (time, sdata, rc.data=T, se=200, AIC=200)

c) If time units are days, and the data covers 10 days but you desire plots showing 12 days: vitality.sa (time, sdata, tlab="days", pplot=1.2)

d) If sdata is incremental mortality (rather than cumulative survival) and no plots or output is desired (the final parameters will still be returned): vitality.sa (time, sdata, datatype="INC", silent=T)

2. A list of functions (in the order they appear in vitality.gaussian.R):

- vitality.sa ( ) – the main program described above.
- dataPrep ( ) – removes NAs from data; creates incremental mortality data from cumulative survival data (or vise – versa); deals with right censored data.
- sa.lt ( ) – simulated annealing for searching the starting values
- SurvFn ( ) – the vitality based survival model with initial Gaussian distribution.
- SurvProbInc ( ) – computes incremental survival probabilities.
- LogLikelihhod ( ) – returns a vector of terms which sum to the negative log likelihood.
- ndexFinder ( ) – an auxiliary function which returns the index of the first value of vector x that is less than or equal to a given value.
- stdErr ( ) – computes the standard error of the MLE estimates.
- Cl.calc ( ) – provides the Pearson's C1 goodness of fit test.
- AIC.calc ( ) – provides the AIC values for the model fitting
- plotting ( ) – provides the cumulative survival and incremental mortality plots
- profilePlot ( ) – provides the likelihood plots (one each when varying r, s, k and u about their MLE) and a contour likelihood plot (created by varying r and s).

3. Tips for using data and user options:

a) Choosing time scale for the original data should be careful. If the time scale is too small (eg. days for the population which can survival over 60 years), the estimated r, s and k would be too tiny to be precise. And the simulated annealing algorithm may fail to converge in this case. Changing scale from “days” to “years” is a wise choice.

b) The initial parameters for simulated annealing are not critical here, and the default values are provided. But a smart choice of the initial values will save the searching time. Usually, r has the same scale of 1/(the time of 50% survival), k will not exceed r and u is in a range of (0, 0.35).

c) To save the computing time, one can initially run the model with L specified as 0. This requires the model run Newton-Ralphson algorithm directly. If the estimated parameters fit the original data well, we can believe the global optimization has achieved. There is no need to run simulated annealing.

Reference:

Anderson, J. J. 2000. A vitality-based model relating stressors and environmental properties to organism survival. Ecological Monographs 70,445-470.

Anderson, J. J., Gildea, M. C., Williams, D. W. and Li. T., 2008. Linking growth, survival and heterogeneity through stochastic vitality. The American Naturalist Vol.171, No.1 E-article.

Kirkpatrick, S., Gelatt, C.D., Jr., and Vecchi, M.P. 1983. Optimization by simulated annealing. Science 220, 4598, 671-680.

Salinger, D. H., Anderson, J. J. and Hamel,O. S., 2003. A Parameter estimation routine for the vitality-based survival model. Ecological Modeling 166, 287-294.

Example of running code in R (http://www.r-project.org/)

Example data from

Letcher, B.H., Rice, J.A., Crowder, L.B, Binkowski, F.P., 1996. Size-dependent effects of continuous and intermittent feeding on starvation time and mass loss in starving yellow perch larvae and juveniles. Trans. Am. Fish. Soci. 125, 14_26.

Data vectors: X = days, Y = survival

```
X <- c(0.000,1.003,2.006,3.036,4.012,4.988,6.018,6.988,7.946,9.006,9.956,10.956,11.936)
Y <- c( 1.000,0.994,0.988,0.969,0.951,0.945,0.935,0.801,0.464,0.093,0.007,0.007,0.007)
```

R command

```
vitality.sa(X,Y)
```

Model output

```
[1] "Initial r s k u:" "0.1"            "0.1"            "0.01"
[5] "0.1"
[1] "estimated r s k u and minimum value:"
[2] "0.126544379442617"
[3] "0.0368899183347269"
[4] "0.00982100105611862"
[5] "0.0340466633438851"
[6] "1.59261253585637"
[1] 0.126540 0.036890 0.009821 0.034047
```