1 Zeta Analysis - Comparison of Callahan's MathCad script to Lampe's analysis in R

1.1 Load

```
source("DoAll.R")

## Warning:  package 'pracma' was built under R version 3.1.1
## Loading required package:  wavethresh
## Loading required package:  MASS
## WaveThresh:  R wavelet software, release 4.6.6, installed
##
## Copyright Guy Nason and others 1993-2013
##
## Note:  nlevels has been renamed to nlevelsWT
##
## Loading required package:  adlift
## Loading required package:  EbayesThresh
##
##  **********************************************
##  adlift:  a package to perform wavelet lifting schemes
##
##  --- Written by Matt Nunes and Marina Knight ---
##    Current package version:  1.3-2  ( 01/11/2012 )
##
##              -+ packaged by MAN +-
##  **********************************************
##
##  adlift 1.3-2 loaded
##
##
## Attaching package:  'adlift'
##
## The following object is masked from 'package:EbayesThresh':
##
##    postmean.cauchy
##
##
##  **********************************************
##  binhf:  Haar-Fisz functions for binomial data
##
##  --- Written by Matt Nunes ---
##    Current package version:  1.0-1  ( 24/04/2014 )
##
##
##  **********************************************
##
##  binhf 1.0-1 loaded
##
##
## Attaching package:  'binhf'
##
## The following objects are masked from 'package:EbayesThresh':
```

```
##
##     ebayesthresh.wavelet.wd, negloglik.laplace, wandafromx
##
## The following object is masked from 'package:wavethresh':
##
##     madmad
##
## The following object is masked from 'package:base':
##
##     norm
##
##
## Attaching package:  'deSolve'
##
## The following object is masked from 'package:pracma':
##
##     rk4
##
## Loading required package:  rootSolve
##
## Attaching package:  'rootSolve'
##
## The following objects are masked from 'package:pracma':
##
##     gradient, hessian
##
## Loading required package:  coda
## Loading required package:  lattice
##
## Attaching package:  'FME'
##
## The following object is masked from 'package:pracma':
##
##     Norm
##
## Loading required package:  timeDate
## Loading required package:  timeSeries
##
## Attaching package:  'fBasics'
##
## The following object is masked from 'package:deSolve':
##
##     rk
##
## The following object is masked from 'package:binhf':
##
##     norm
##
## The following objects are masked from 'package:pracma':
##
##     akimaInterp, inv, kron, pascal
##
## The following object is masked from 'package:base':
```

```
##
##     norm
                                                                    se/CSV_M_SC/1
```

```
## 10  2.34 -1.487e-05 0.000e+00 -7.015e-06 0.000e+00
## 11  2.25 -1.119e-05 0.000e+00 -6.524e-06 0.000e+00
## 12  2.21 -1.406e-05 0.000e+00 -9.305e-06 0.000e+00
## 13  2.27 -2.178e-05 2.778e-09 -1.050e-05 1.389e-09
## 14  2.52 -1.354e-05 0.000e+00 -7.716e-06 0.000e+00
## 15  2.19 -1.220e-05 0.000e+00 -1.060e-05 0.000e+00
## 16  2.33 -1.509e-05 2.556e-08 -8.127e-06 0.000e+00
## 17  2.29 -1.355e-05 6.805e-09 -9.277e-06 0.000e+00
## 18  2.33 -1.649e-05 5.556e-09 -1.308e-05 0.000e+00
```

1.2 function to evaluate zeta

```
# ==== DIFFERENTIAL EQUATION ====
ZETA.01 <- function(Time, State, Parm){

#   # ---- EFFECTIVE STRESS PARAMETERS FOR CRUSHED SALT FORMULATION OF MODEL ----
#   KAP0 <- 10.119
#   KAP1 <- 1.005
#   DDT  <- 0.896
#   NK   <- 1.331
#   KAP2 <- 1
#   ETA0  <- 0.102854
#   ETA1  <- 3.9387
#   ETA2   <- 1
#   NF    <- 3.5122

  # ---- EFFECTIVE STRESS PARAMETERS FOR CRUSHED SALT FORMULATION OF MODEL ----
#   # ---- VALUES FOR MATHCAD CHECK
  KAP0 <- 10.119
  KAP1 <- 1.005
  NK   <- 1.331
  #--------------
  DDT  <- 0.8854
  KAP2 <- 1
  ETA0  <- 0.015
  ETA1  <- 1
  ETA2   <- 0.7
  NF    <- 8.40075

#   # ---- Munson-Dawson Creep Parameters (17) ---- FOR CLEAN SALT
#   A1    <- 8.386e22
#   A2  <- 9.672e12
#   Q1R  <- 12581
#   Q2R  <- 5033
#   N1  <- 5.5
#   N2  <- 5.0
#   B1  <- 6.0856e6
#   B2  <- 3.034e-2
#   Q  <- 5335
#   S0  <- 20.57
#   M  <- 3
#   K0  <- 6.275e5
#   C  <- 9.198e-3
#   ALPHA <- -17.37
#   BETA  <- -7.738
#   DELTA <- 0.58
#   MU  <- 12400

  # ---- Munson-Dawson Creep Parameters (17) ---- FOR ARGILLACEOUS SALT
  A1    <- 1.407e23
  A2            <- 1.314e13
  Q1R   <- 12581
  Q2R   <- 5033
```

```
  N1            <- 5.5
  N2            <- 5.0
  B1            <- 8.988e6
  B2            <- 4.289e-2
  Q             <- 5335
  S0            <- 20.57
  M             <- 3
  K0            <- 2.47e6
  C             <- 9.198e-3
  ALPHA <- -14.96
  BETA  <- -7.738
  DELTA <- 0.58
  MU            <- 12400

  #---- DATA INTERPRETED FROM TEST DATA
#   TEMP <- temp.interp(Time)
#   AS   <- as.interp(Time)
#   LS   <- ls.interp(Time)
#   D    <- d.interp(Time)

  #---- constant data values for comparison to Callahan's analysis
  TEMP <- 300
  AS   <- (-6/300.8)*Time
  LS   <- (-2/300.8)*Time
  D    <- 0.9

  # ---- calculate variables ----
# browser()
  MS  <- (2.0 * LS + AS) / 3   # MEAN STRESS
  DS    <- LS - AS                                     # STRESS DIFFERENCE
  DEN <- D                     # CURRENT FRACTIONAL DENSITY

  Z3  <- State[1] # internal variable "zeta" for the transient function (FU)

  VAR <- ifelse(DEN <= DDT, DDT, DEN)

  # ---- Equivalent Stress ----
  OMEGAA   <- ((1 - DEN) * NF / (1 - (1 - DEN)^(1/NF))^NF)^(2/(NF + 1))
  ETA             <- ETA0 * OMEGAA^ETA1
  TERMA    <- ((2 - DEN)/DEN)^((2 * NF)/(NF + 1))

  # ---- Eqn. 2-3 (SAND97-2601) ----
  # Equivalent stress measure for Disl. Creep
  SEQF  <- sqrt(ETA * MS^2 + ETA2 * TERMA * DS^2)

  # ==== START: equivalent inelastic strain rate form for dislocation creep ====
  # ---- Steady State Strain Rate Calc ----
  ES1 <- A1 * (SEQF / MU)^N1 * exp(-Q1R/TEMP)   # Dislocation climb - Eqn. 2-30
  ES2 <- A2 * (SEQF / MU)^N2 * exp(-Q2R/TEMP)   # Undefined Mechanism - Eqn. 2-31

  # Slip - Eqn. 2-32 (SAND98-2601)
#   browser()
  ARG <- Q * ((SEQF - S0) / MU)
```

```
  ES3 <- (B1 * exp(-Q1R / TEMP) + B2 * exp(-Q2R / TEMP)) *
          sinh(ARG) * Heaviside(SEQF - S0)

  ESS = ES1 + ES2 + ES3 # Steady-state strain rate, Eqn. 2-29 (SAND97-2601)

  # ---- EVALUATE TRANSIENT FUNCTION, 3 branches: work hardening, equilibrium, recovery
#   browser()
  EFT  <- K0 * exp(C * TEMP) * (SEQF / MU) ^ M  # Transient Strain Limit, Eqn. 2-28
  BIGD <- ALPHA + BETA * log10(SEQF / MU)       # Work-Hardening parameter, Eqn 2-28

  FU <- ifelse(Z3 == EFT, 1, ifelse(Z3 < EFT, exp(BIGD * (1 - Z3 / EFT) ^ 2),
                                    exp(-DELTA * (1 - Z3 / EFT) ^ 2)))

  MD <- FU * ESS  # equivalent inelastic strain rate form for dislocation creep, Eqn 2-23

  DZ3 <- (FU - 1) * ESS  # derivative of internal variable "ZETA"
  DZ <- list(c(DZ3), MD, FU, ESS, ES1, ES2, ES3,
             DZ3, EFT, SEQF, BIGD, AS, LS, ETA, OMEGAA)

  return(DZ)
}
```

1.3 load settings for zeta function

```
# ================================================================================
PAR.TEST <- DATA.INP[which(DATA.INP$ITEST == "SC1B"),] # SUBSET OF DATA FOR ANALYSIS
# PAR.TEST <- PAR.TEST[1:28,]

# ---- linear interpolation functions to be called in "ZETA.01" ----
temp.interp <- approxfun(x = PAR.TEST$TIME, y = PAR.TEST$TEMP)
as.interp   <- approxfun(x = PAR.TEST$TIME, y = PAR.TEST$AS)
ls.interp   <- approxfun(x = PAR.TEST$TIME, y = PAR.TEST$LS)
d.interp    <- approxfun(x = PAR.TEST$TIME, y = PAR.TEST$D)

# ---- intial values for state variables ----
Z3      <- 1e-12 # internal variable "ZETA" for the transient function (FU)
# integral of Eqn 2-27, (initial values)

IC <- (c(Z3 = Z3)) # array of initial values

# TIME <- PAR.TEST£TIME
TIME <- c(0, 50, 52.5, 65, 91.61, 126.101, 174.21, 240.386, 300) #for comparison

# ---- function for Predicting the Creep Strain(E) Rates ----
 ZETA.ODE <- ode(func =ZETA.01, y = IC, times = TIME, verbose = TRUE,
                 maxsteps = 1000000, rtol = 1e-14, atol = 1e-14, hmax = 1 )

##
## --------------------
## Time settings
## --------------------
##
```

```
##   Normal computation of output values of y(t) at t = TOUT
##
## --------------------
## Integration settings
## --------------------
##
##   Model function an R-function:
##   Jacobian not specified
##
##
## --------------------
## lsoda return code
## --------------------
##
##   return code (idid) =  2
##   Integration was successful.
##
## --------------------
## INTEGER values
## --------------------
##
##   1 The return code : 2
##   2 The number of steps taken for the problem so far: 330
##   3 The number of function evaluations for the problem so far: 567
##   5 The method order last used (successfully): 5
##   6 The order of the method to be attempted on the next step: 5
##   7 If return flag =-4,-5: the largest component in error vector 0
##   8 The length of the real work array actually required: 36
##   9 The length of the integer work array actually required: 21
##  14 The number of Jacobian evaluations and LU decompositions so far: 0
##  15 The method indicator for the last succesful step,
##            1=adams (nonstiff), 2= bdf (stiff): 1
##  16 The current method indicator to be attempted on the next step,
##            1=adams (nonstiff), 2= bdf (stiff): 1
##
## --------------------
## RSTATE values
## --------------------
##
##   1 The step size in t last used (successfully): 1
##   2 The step size to be attempted on the next step: 1
##   3 The current value of the independent variable which the solver has reached: 300.6
##   4 Tolerance scale factor > 1.0 computed when requesting too much accuracy: 0
##   5 The value of t at the time of the last method switch, if any: 0
##
```