# Isometry test and size correction

```
isometry.test=function(response, reference, specimen.id, expected.slope,
alpha=0.05){ #1 Creates function "isometry.test", with arguments "response",
"reference", "specimen.id", "expected.slope" and "alpha".
  ## VERIFYING THE ARGUMENTS
  if ((missing(response)) ||
(is.na(suppressWarnings(as.numeric((response)[[1]])))) ||
is.factor(data.frame(response)[[1]])) #2 Verifies if "response" was inserted
and if its coercible to numeric.
  {
    stop("response should be specified as a numeric vector containing the
response measurement(s) for isometry test") #3 If not, interrupts the
function and gives a message to the user.
  }
  response=as.data.frame(response) #4 Transforms "response" in a data frame,
so its length can be read as number of variables.
  if ((missing(reference)) || (length(reference) !=
max(c(length(response),length(as.data.frame(t(response)))))) ||
(is.na(suppressWarnings(as.numeric(reference[[1]])))) ||
is.factor(data.frame(reference)[[1]])) #5 Verifies if "reference" was
inserted, if it has the same length as response, and is coercible to
numeric.
  {
    stop("reference should be specified as a numeric vector containing the
reference measurements for isometry test.") #6 If not, interrupts the
function and gives a message to the user.
  }
  if ((missing(specimen.id)) || (class(specimen.id) != "character") ||
(length(specimen.id) != length(reference))) #7 Verifies if "specimen.id" was
inserted, if it is a character and if it has the same length as "reference".
  {
    stop("specimen.id should be specified as a character vector containing
the names of specimens in data table.") #8 If not, interrupts the function
and gives a message to the user.
  }
  if ((missing(alpha)) || (class(alpha) != "numeric") || (alpha<=0) ||
(alpha>=1)) #9 Verifies if "alpha" was inserted, if it is numeric and >0 and
<1.
  {
    message("\talpha should be numeric and >0 or <1; herein default set to
0.05.\n") #10 If not, gives a message to the user and uses the default value
set above with the argument.
  }
  if
((colnames(response)==as.character(rep(1:length(colnames(response)),1))) ||
(match(colnames(response), specimen.id, nomatch = 0) !=
rep(0,length(response)))) #11 Verifies if the input data frame "response"
has the specimens in rows.
```

```
  {
    response=as.data.frame(t(response)) #12 If not, transposes so the
specimens are in rows. Will have further implications in calculating number
of response variables.
  }
  if ((missing(expected.slope)) || (class(expected.slope) != "numeric") ||
(length(expected.slope) != length(colnames(response)))) #13 Verifies if
"expected.slope" was inserted, if it is numeric and has the same length as
"response".
  {
    message("\texpected.slope should be a concatenation of the ratios: power
of response variable/power of the reference variable; herein default set to
1 for all.\n") #14 #10 If not, gives a message to the user.
    expected.slope=rep(1,length(colnames(response))) #15 Uses the default
value (set here).
  }
  ## FUNCTION FOR ISOMETRY TEST AND SIZE CORRECTION
  # Setting and configuring specific parameters
  ref.name=readline("\tPlease tell me the name of your reference variable:
") #16 Asks for name of reference variable, that will be used in the output.
  result=matrix(data=NA, nrow=1, ncol=length(colnames(response))) #17
Creates the empty "result" matrix, with 1 row and "response" columns.
  result=as.data.frame(result) #18 Transforms "result" into a data frame, so
it will later accept characters.
  result2=matrix(data=NA, nrow=length(rownames(response)),
ncol=length(colnames(response))+1) #19 Creates the empty "result2" matrix
with as muchs rows as specimens, and as much columns as reference+response
variables.
  reference=as.numeric(reference) #20 Coerces "reference" to numeric, in
case it was read before as characters.
  result2[,1]=reference #21 Sets reference variable to the first column of
"result2".
  if (length(response)==1) #22 Verifies if there is only one response
variable.
  {
    res.name=readline("\tPlease tell me the name of your response variable:
") #23 If yes, asks the user for the name of the response variable.
    colnames(response)=res.name #24 Attributes the name of the response
variable to its specific column, which will be used to construct the output
matrices.
  }
  response.all=as.matrix(response) #25 Creates the matrix "response.all"
with "response", as "response" will enter the for loop that follows.
  # Loop to compare each response variable to the reference variable,
perform the isometry test and the specific size-correction.
  for (k in 1:length(colnames(response))) #26 Loop using counter k from 1 to
the length of "response"
  {
    response=response.all[,k] #27 Creates object "response" with k column of
"response.all"
```

```
    response=as.numeric(response) #28 Coerces "response" to numeric, in case
it was read before as character.
    response=log(response) #29 Log transformation of response variable
attributed to "response"
    reference2=log(reference) #30 Log transformation of reference variable
attributed to "reference2", so that it overwrites "reference", similar to
what was done with "response.all" and "response".
    response.m=lm(response~reference2) #31 Creates the object "response.m"
with the linear model of "response" as a function of "reference2".
    f=(summary(response.m))$fstatistic #32 Creates the object "f" with the
F-statistic values of the linear model.
    p=pf(f[1],f[2],f[3],lower.tail=F) #33 Creates the object "p" with the p-
value from the F-statistic values above.
    if (p > alpha) #34 Verifies if "p" is bigger than "alpha".
    {
      result[k]="lm not significant" #35 If yes, attribute "lm not
significant" to element k in "result".
    }
    else #36 If not, linear model is significant, so proceeds with
calculations.
    {
      slope=(summary(response.m))$coefficients[2] #37 Creates the object
"slope" and stores the slope coefficient from "response.m".
      slope.se=(summary(response.m))$coefficients[4] #38 Creates the object
"slope.se" and stores the slope's standard error coefficient from
"response.m".
      tvalue=abs((slope-expected.slope[k])/slope.se) #39 Creates the object
"tvalue", with the t test value from the comparison between slope and
expected slope.
      pvalue=(1-pt(tvalue,length(response)-2))*2 #40 Creates the object
"pvalue", with the p-value corresponding to the t test value above (degrees
of freedom = n-2).
      if (pvalue > alpha) #41 Verifies if "pvalue" is bigger than "alpha".
      {
        result[k]="Isometry" #42 If yes, the slope from response.m is not
significantly different from the expected slope, so it attributes "Isometry"
to element k in "result".
result2[,k+1]=as.numeric(response.all[,k])*((mean(result2[,1])/result2[,1])^
expected.slope[k]) #43 Makes isometric transformation based on expected
slope, following the formula in Lleonart et al. (2000).
      }
      else #44 Verifies if "pvalue" is not bigger than alpha.
      {
        result[k]="Allometry" #45 If it is not bigger, the slope from
response.m is significantly different from the expected.slope, so it
attributes "Allometry" to element k in "result".
result2[,k+1]=as.numeric(response.all[,k])*((mean(result2[,1])/result2[,1])^
slope) #46 Makes allometric transformation based on observed slope,
following the formula in Lleonart et al. (2000).
      }
    }
```

```
    colnames(result)[k]=colnames(response.all)[k] #47 Establishes column
names of "result" as the column names in response.all.
  }
  ## SETTING AND RETURNING THE OUTPUT
  result2[,1]=mean(result2[,1]) #48 Makes the size correction of the
reference variable, by changing it in the first column of "result2" for its
mean.
  colnames(result2)=c(ref.name,colnames(response.all)) #49 Establishes the
column names of "result2" as the reference name given by the user, and the
column names of "response.all".
  rownames(result2)=specimen.id #50 Establishes the row names of "result2"
as the specimen names given by the user.
  all=array(list(result, result2)) #51 Creates the output object "all", as
array with a list that includes "result" and "result2".
  return(all) #52 Returns to the user the output array "all" with the
results of the isometry test and size correction.
}
```