Package ‘AGDEX’

November 21, 2016

Type Package

Title Agreement of Differential Expression Analysis

Version 1.22.0

Date 2011-10-13

Author Stan Pounds <stanley.pounds@stjude.org>; Cuilan Lani Gao <cuilan.gao@stjude.org>

Maintainer Cuilan Lani Gao <cuilan.gao@stjude.org>

Depends R (>= 2.10), Biobase, GSEABase

Description A tool to evaluate agreement of differential expression for cross-species genomics

License GPL Version 2 or later

Imports stats

LazyLoad yes

biocViews Microarray, Genetics, GeneExpression

NeedsCompilation no

R topics documented:

AGDEX-package ................................................................. 2
agdex ................................................................. 3
agdex.res ................................................................. 6
agdex.scatterplot ................................................................. 6
get.gset.result.details ...................................................... 7
gset.data ................................................................. 8
human.data ................................................................. 9
make.dex.set.object ..................................................... 10
map.data ................................................................. 11
mouse.data ............................................................... 12
read.agdex.gset.details .................................................. 13
read.agdex.result ......................................................... 14
write.agdex.gset.details .................................................. 15
write.agdex.result ......................................................... 16

Index 18
Description

The objective of AGDEX is to evaluate whether the results of a pair of two-group differential expression analysis comparisons show a level of agreement that is greater than expected if the group labels for each two-group comparison are randomly assigned. The agreement is evaluated for the entire transcriptome and (optionally) for a collection of pre-defined gene-sets. Additionally, the procedure performs permutation-based differential expression and meta analysis at both gene and gene-set levels of the data from each experiment.

list of main functions:

make.dex.set.object: a function to generate a list object for agdex() function
agdex: a function to perform AGDEX analysis
agdex.scatterplot: a function to plot the results of AGDEX analysis for visualization
get.gset.result.details: a function to obtain gene level details for specified one gene-set or multiple gene-sets
write.agdex.result: a function to write the AGDEX result to a tab-delimited text file
read.agdex.result: a function to read the AGDEX result from an output file written by write.agdex.result
write.agdex.gset.details: a function to write a tab-delimited text file with the gene-level details for the results of a gene-set analysis
read.agdex.gset.details: a function to read the output file generated by write.agdex.gset.details

Details

AGDEX combines the differential expression analysis results from a pair of two-group comparisons. The two comparisons may utilize different species or platforms. AGDEX has been used to confirm that new mouse models accurately represent the transcriptomes of the pediatric brain tumors ependymoma (Johnson et al. 2010) and medulloblastoma (Gibson et al. 2010). Pounds et al. (2011) provide a detailed description of the AGDEX procedure. In summary, AGDEX analysis can perform the following analysis:

1. identify individual genes that are differentially expressed for each two-group comparison;
2. identify gene-sets that are differentially expressed for each two-group comparison;
3. integrate results across the pair of two-group comparisons to identify differentially expressed genes;
4. integrate results across the pair of two-group comparisons to identify differentially expressed gene-sets;
5. characterize and determine the statistical significance of similarities of differential expression profiles across the pair of two-group comparisons for the entire transcriptome and for specific gene-sets.

Package: AGDEX
Type: Package
Version: 1.0.4
Date: 2011-8-29
License: GPL (>=2)
LazyLoad: yes
agdex

Author(s)

Stan Pounds <<stanley.pounds@stjude.org>>; Cuilan Lani Gao <<cuilan.gao@stjude.org>>

References


Description

This function performs agreement of differential expression (AGDEX) analysis across a pair of two-group experiments. AGDEX measures and determines the statistical significance of the similarity of the results from two experiments that measure differential expression across two groups. A metric of agreement is defined to measure the similarity and the significance is determined by permutation of group labels. Please see our methodology paper for details [1] (Pounds et al. 2011).

Usage

agdex(dex.setA, dex.setB, map.data, min.nperms = 100, max.nperms = 10000)

Arguments

dex.setA A list object with 4 components that defines a two-group comparison "A", for example "human tumor-human control". These components are express.set, comp.def, comp.variable and gset.collection (optional). The express.set component is a Bioconductor ExpressionSet object with a matrix of expression data in exprs and the phenotype data in pData. The comp.variable component gives the name or numeric index of the column of group label in pData of express.set object. comp.def is a string with the format "tumor-control" to define a comparison of expression between samples labeled as "tumor" and samples labeled as "control". The gset.collection (optional) belongs to GeneSetCollection class. See details.

dex.setB A list object that defines the other two-group comparison. It has the same structure as dex.setA.

map.data a list object with 3 components that defines how probe-sets from dex.setA are matched with probe-sets from dex.setB. The probe.map component is a data frame with each row defining how probe-sets are matched across the pair of two-group comparisons. The components map.A.probe.col and map.B.probe.col give the names or numeric index of the column containing probe-set identifiers in dex.setA and dex.setB respectively.
minimum number of permutations for adaptive permutation testing of gene-set level results, default is set to 100. Adaptive permutation testing permutes data until observing min.nperms statistics that exceed the observed statistic in absolute value or until max.nperms permutations are performed. Adaptive permutation testing greatly reduces computational effort for permutation analysis in many genomics applications. See [2] (Pounds et al. 2011) for more details.

maximum number of permutations for adaptive permutation testing of gene-set level results and fixed total number of permutations for classical permutation testing of probe-set level results and genome-wide agreement of differential expression, default is set to 10000.

Details

Object express.set belongs to ExpressionSet class. express.set includes two components: exprs: a matrix of gene expression data with row of probe-sets and columns of subjects. pData: a data frame with each row representing a sample and two columns are sample ID and sample group label.

gset.collection component contains a GeneSetCollection object defined in the Bioconductor package GSEABase. The gset.collection object must be the same identifiers for probe-sets as those used in expression matrix in express.set.

Value

A list object with the following components

dex.compA this string echoes the comp.def component of dex.setA that defines the definition for two-group comparison "A", for example "human tumor-human control"

dex.compB this string echoes the comp.def component of dex.setB that defines the definition for two-group comparison "B", for example "mouse tumor-mouse control"

gwide.agdex.res a data.frame with the agreement statistics, p-values, and number of permutations for genome-wide agreement of differential expression analysis.

gset.res a data.frame with results of gene-set differential expression analysis for each comparison and gene-set agreement of differential expression analysis results.

meta.dex.res a data.frame with results for probe-sets matched across comparisons "A" and "B". The data.frame includes the differential expression statistic and p-value from each comparison and the meta-analysis z-statistic and p-value for differential expression.

dex.resA a data.frame with differential expression analysis results for individual probe-sets for two-group comparison "A". The data.frame includes the probe-set identifier, difference of mean log-expression statistic, and the p-value.

dex.resB a data.frame with the same structure as dex.resA that gives the results for two-group comparison "B".

dex.asgnA a data.frame that echoes the group label assignments for comparison "A"

dex.asgnB a data.frame that echoes the group label assignments for comparison "B"

gset.listA a data.frame with gene-set lists for comparison "A". Each row indicates an assignment of a probe-set identifier to a gene-set.

gset.listB a data.frame with gene-set lists for comparison "B".

gset.list.agdex a data.frame that assigns probe-set pairs (probe-sets from comparisons A and B that query the same gene) to gene-sets for gene-set agreement of differential expression analysis.
agdex

Author(s)

Stan Pounds <<stanley.pounds@stjude.org>; Cuilan Lani Gao <<cuilan.gao@stjude.org>>

References


See Also

ExpressionSet class: ExpressionSet.
GeneSetCollection class: GeneSetCollection.

Examples

# load data
data(human.data)
data(mouse.data)
data(map.data)
data(gset.data)

# make dex.set object for human data
dex.set.human <- make.dex.set.object(human.data,
  comp.var=2,
  comp.def="human.tumor.typeD-other.human.tumors",
  gset.collection=gset.data)

# make dex.set object for mouse data
dex.set.mouse <- make.dex.set.object(mouse.data,
  comp.var=2,
  comp.def="mouse.tumor-mouse.control",
  gset.collection=NULL)

# call agdex routine
res <- agdex(dex.set.human,dex.set.mouse,map.data,min.nperms=5,max.nperms=10)

# see visualization result of the whole genome
agdex.scatterplot(res, gset.id=NULL)

# see visualization result of a specific gene-set
agdex.scatterplot(res, gset.id="DNA_CATABOLIC_PROCESS")

# get the gene-set result of a specific gene-set
gset.detail <- get.gset.result.details(res, gset.id="DNA_CATABOLIC_PROCESS", alpha=0.01)
agdex.res  Saved result returned by agdex()

Description

*agdex.res* is result object returned by *agdex* function. We saved the result so that we do not have to repeat calling *agdex* to show the examples in the documentation files (for visualization, writing output files etc.). Users may not need to load this result data as long as the returned result of *agdex* is not deleted from user’s current R workspace.

Usage

```r
data(agdex.res)
```

Value

Components of *agdex.res* have the same meaning as the result object returned by function *agdex()*. 

See Also

*agdex*

agdex.scatterplot  scatter plot of AGDEX result

Description

A function to visualize the results of AGDEX analysis for the entire genome or specific gene-sets in a scatterplot.

Usage

```r
agdex.scatterplot(agdex.res, gset.id = NULL)
```

Arguments

- `agdex.res`: result of an AGDEX analysis, the returned result of the function *agdex*.
- `gset.id`: a specified gene-set identifier. Default is set to NULL, which produces a visualization result of the entire genome.

Value

Returns either a scatter plot of pairs of difference in average log-expression values for genome-wide analysis or a specified gene-set.

Author(s)

Stan Pounds <<stanley.pounds@stjude.org>>; Cuilan Lani Gao <<cuilan.gao@stjude.org>>
get.gset.result.details

See Also

agdex; get.gset.result.details

Examples

data(agdex.res)
# see visualization result of the whole genome
agdex.scatterplot(agdex.res, gset.id=NULL)
# scatterplot for a specified gene-set
agdex.scatterplot(agdex.res, gset.id="DNA_CATABOLIC_PROCESS")

get.gset.result.details

Extract gene-level details from gene-set results

Description

A function to extract the probe-sets level details for gene-set results, i.e. obtain the differential expression statistics for probe-sets assigned to the gene-sets. This allows users to explore which probe-sets results contribute the most to gene-set differential expression analysis statistics and gene-set agreement of differential expression statistics.

Usage

get.gset.result.details(agdex.result, gset.ids = NULL, alpha=0.01)

Arguments

agdex.result  agdex result returned by function agdex

getset.ids    a vector of gene-set IDs. If NULL, the result will return gene level details for all significant gene-sets at a chosen significant level alpha.

alpha         significance level of gene-set, default set to 0.01

Value

This function returns a list of three components.

gsetA.details Gene-set details result for experiment A, including differential expression statistic and p-value for each probe-set in each gene-set. Each row represents a probe-set from A. The columns give gene-set name, enrichment statistic and its corresponding p-values, differential expression statistics and p-values.

gsetB.details similar result of gene-set details for experiment B. Rows and columns give the similar information to gsetA.details.

agdex.details A data frame. Each row gives results for one probe-set pair. The columns give the gene-set names, cosine statistic and difference of proportions statistic and p-values, meta statistic and its p-value.

Author(s)

Stan Pounds <<stanley.pounds@stjude.org>>; Cuilan Lani Gao <<cuilan.gao@stjude.org>>
gset.data

See Also
write.agdex.gset.details; read.agdex.gset.details

Examples

# Load saved result run by agdex routine
data(agdex.res)

data(agdex.res)

data(gset.res)

data(gset.res)

# obtain gene-set result
gset.res.all <- get.gset.result.details(agdex.res, gset.ids = NULL, alpha=0.01)

# obtain the detailed gene set for specified gene-sets
gset.res0 <- get.gset.result.details(agdex.res, gset.ids=c("DNA_CATABOLIC_PROCESS","GOLGI_STACK"), alpha=0.01)

gset.data a sample gene-set data

description
gset.data A gene-set data belongs to the GeneSetCollection class of GSEABase package.

Usage
data(gset.data)

Details
This sample gene-sets data contain 10 small gene-sets which are randomly selected from the full pathway gene-sets downloaded from http://www.broadinstitute.org/gsea/msigdb/download_file.jsp?filePath=/resources/msigdb/3.0/c5.all.v3.0.orig.gmt. The full gene-set data contain 1454 gene-sets. For the sample gene-set data of AGDEX package, 10 gene-sets are randomly selected, each has 20 to 30 probe-sets. Each gene-set has 20 to 30 probe-sets. We used getGmt function from GSEABase to read GMT format into a GeneSetCollection class object, then map the genes to probe-set IDs using hgu133 plus2 annotation data which contains the mapping from genes to probe-sets indentifiers.

See Also
GeneSetCollection-class
agdex; human.data; mouse.data; map.data

Examples

# download the pathway gene-sets data #
# Not run:
gset.url <- "http://www.broadinstitute.org/gsea/msigdb/download_file.jsp?filePath=/resources/msigdb/3.0/c5.all.v3.0.orig.gmt"
gset.file.name <- unlist(strsplit(gset.url, split="/"))
gset.file.name <- gset.file.name[length(gset.file.name)]
gset.destination <- paste(local.data.dir, gset.file.name, sep="")
download.file(gset.url, gset.destination)
gset.file <- gset.destination
gset.data <- getGmt(gset.file)
# read in human U133+2 array annotation file#
human.ann.data <- read.table("local human U133+2 array annotation data", head=T, sep="\t")
genes.in.ann <- human.ann.data[,3] # get the gene symbols from annotation file

# map the genes to probe-set IDs using human annotation data.##########
for (i in 1:length(gset.data))
{
  genes.this.gset <- geneIds(gset.data[[i]])
  match.rows <- is.element(genes.in.ann, genes.this.gset)
  probe.this.gset <- human.ann.data$ID[match.rows]
  geneIds(gset.data[[i]]) <- as.character(probe.this.gset)
}

## End(Not run)

----------

**human.data**  
*Sample ExpressionSet object of human data*

**Description**

An *ExpressionSet* object of human data.

**Usage**

data(human.data)

**Details**

`human.data` is an *ExpressionSet* object where *exprs* slot carries the human gene expression data and the *pData* contains the phenotype data. This sample data `human.data` is a subset taken from our published study of human brain tumor ependymoma (Johnson et al. 2010). The original full human expression data contains 54,613 probe-sets for 83 human ependymoma tumors. The gene expression is profiled with Affymetrix U133+2(mRNA) array and the expression data were normalized with MAS 5.0 algorithm.

The *expr* of `human.data` is a gene expression matrix with rows of probe-sets and columns representing ependymoma tumors which are classified as belonging to the novel subgroup D or others. Probe-sets in the gene expression matrix are randomly selected from the full human gene profile such that the selected probe-sets belong to the gene-sets in *gset.data*. `pData` slot of `human.data` is a data frame with two columns indicating sample ID and sample group label for each sample(either "human.tumor.typeD" or "other.human.tumors").

**Value**

*expr*(human.data)

A matrix with 246 rows and 83 columns with rows representing probe-sets and columns of human sample IDs.

*pData*(human.data)

A data frame with 83 rows and 2 columns. Each row represents one human sample. Column *id* is the human sample ID and *grp* is the assigned sample group label.
make.dex.set.object

References

See Also
ExpressionSet-class

Examples

```r
data(human.data)
human.express.set <- exprs(human.data)
human.pheno.data <- pData(human.data)
```

Description

This function generates a list object containing four components for function agdex()

Usage

```r
make.dex.set.object(Eset.data, comp.var, comp.def, gset.collection)
```

Arguments

- `Eset.data`: an `ExpressionSet` object carries the gene expression data (`Exprs`) and Phenotype data (`pData`).
- `comp.var`: the column name or numeric index for group labels in `pData` of object `Eset.data`.
- `comp.def`: a string definition of comparison, group labels connected by "-".
- `gset.collection`: an object belongs to class `GeneSetCollection`.

Details

The ExpressionSet includes two components: `exprs`: a matrix of expression values `pData`: a data frame contains the sample IDs and their assigned group labels.

`gset.collection` contains a `GeneSetCollection` object defined in the Bioconductor package GSEABase. The `gset.collection` object must use the same identifiers for probe-sets as that used in the `exprs` component of `Eset.data`.

Value

A list object containing the four components described in argument section.

Author(s)

Stan Pounds <<stanley.pounds@stjude.org>>; Cuilan Lani Gao <<cuilan.gao@stjude.org>>
map.data

See Also
ExpressionSet class: ExpressionSet.
GeneSetCollection class: GeneSetCollection.

Examples

# load data
data(human.data)
data(mouse.data)
data(gset.data)
# make dex.set object for human data
dex.set.human <- make.dex.set.object(human.data,
comp.var=2,
comp.def="human.tumor.typeD-other.human.tumors",
gset.collection=gset.data)

# make dex.set object for mouse data
dex.set.mouse <- make.dex.set.object(mouse.data,
comp.var=2,
comp.def="mouse.tumor-mouse.control",
gset.collection=NULL)

map.data

Probe-set Mapping Data

Description
A mapping between Human probe-sets and Mouse probe-sets

Usage
data(map.data)

Details

map.data is a list object containing 3 components, probe.map, map.Aprobe.col and map.Bprobe.col. The component probe.map is a data frame with 2 columns of probe-sets identifiers from human and mouse array respectively. This sample mapping data probe.map is a subset of the full mapping data set across human genome U133 Plus 2.0 Array and mouse Expression 430 Array available at www.affymetrix.com. They provide multiple match mode data sets, such as Good Match, Complex Match Best match etc. We downloaded the Best Match data set as our mapping data. The probe-sets in probe.map are selected such that they are contained in both expression matrix of human.data and mouse.data. Users can choose mapping data according to the species and platforms of their gene expression profiles either by downloading from www.affymetrix.com or from other sources. The array platforms of mapping data must match that of gene expression profile of each species.

probe.map component of the sample data map.data contains 490 rows of ortholog-matched probe-sets across human array and mouse array. map.Aprobe.col and map.Bprobe.col specify column number or name containing the probe-sets from study A and study B respectively.
mouse.data

Value

probe.map A data frame with 490 rows and 2 columns of probe-sets identifiers of human and mouse

map.Aprobe.col Column number or name in data frame probe.map containing probe-sets IDs from study A

map.Bprobe.col Column number or name in data frame probe.map containing probe-sets IDs from study B

See Also

agdex; human.data; mouse.data; gset.data

Examples

# download the "best match" mapping data across human array and mouse array #
## Not run:
map.url <- "http://www.affymetrix.com/Auth/analysis/downloads/na31/ivt/Mouse430_2.na31.ortholog.csv.zip"
map.file.name <- unlist(strsplit(map.url,split="/"))
map.filename <- map.file.name[length(map.file.name)]
map.destination <- paste(local.data.dir,map.file.name,sep="")
download.file(map.url,map.destination) # Affy website may need users to register first
unzip(map.destination)
affy.ortho.file <- substring(map.destination,1,nchar(map.destination)-4)

# read in the mapping data #
ortho.data <- read.csv(affy.ortho.file,quote="",as.is=T)
keep the probe-sets identifiers of human array "HG-U133_Plus_2" only
keep.rows <- is.element(ortho.data$Ortholog.Array,"HG-U133_Plus_2")
ortho.data <- ortho.data[keep.rows,]
ortho.data <- ortho.data[,c(1,3)] # keep the columns containing probe-sets only

# prepare the list object of map data for calling AGDEX routine #
map.data <- list(probe.map=ortho.data,
    map.Aprobe.col=2, # the column index containing human probe-sets IDs in data frame probe.map
    map.Bprobe.col=1) # the column index containing mouse probe-sets IDs in data frame probe.map

## End(Not run)

mouse.data Mouse Data

Description

mouse.data is an ExpressionSet object where exprs slot carries the mouse gene expression data and pData carries the phenotype data.
Usage

data(mouse.data)

Details

This `mouse.data` is a subset taken from our published mouse data (Johnson et al. 2010). The original full mouse data was profiled by affymetrix 430 v2(mRNA). It contains 45037 probe-sets for 13 mice brain tumors and 179 normal mice stem cells.

We used the best-match data (available from www.affymetrix.com) as the mapping data of ortholog-matched probe-sets across human gene expression data and mouse gene expression data. `expr` slot of `mouse.data` is a matrix of subset of the full mouse gene expression data. Those selected mouse probe-sets are ortholog-matched with human probe-sets in `human.data`. `pData` slot of the `mouse.data` is a data frame with row representing mouse samples and two columns indicating sample IDs and sample group labels for each sample (either "mouse.tumor" or "mouse.control").

Value

`expr(mouse.data)`

A matrix with 264 rows and 192 columns with rows representing probe-sets and columns of mouse sample IDs. Each row name of the matrix is a probe-set Identifier.

`pData(mouse.data)`

A data frame with 192 rows and 2 columns. Each row represents one mouse sample. Column `id` is the mouse sample ID and `grp` is the assigned sample group label.

References


See Also

ExpressionSet-class

`agdex; human.data; map.data; gset.data`

Examples

```r
data(mouse.data)
mouse.express.set <- exprs(mouse.data)
mouse.pheno.data <- pData(mouse.data)
```

Description

A function to read output file saved by write.agdex.gset.details.
read.agdex resultat

Usage
read.agdex.gset.details(gset.detail.file)

Arguments
gset.detail.file
the file name of the probe-set level gene-set result saved by function write.agdex.gset.details

Author(s)
Stan Pounds<<stanley.pounds@stjude.org>>; Cuilan Lani Gao<<cuilan.gao@stjude.org>>

See Also
agdex;write.agdex.gset.details; get.gset.result.details

Examples
#read agdex gene-set details from an output .txt file written by function "write.agdex.gset.details"
## Not run:
read.agdex.gset.details("gset.details.txt")
## End(Not run)

read.agdex.result Read the output file of agdex analysis

Description
A function to read the output file produced by the function write.agdex.result

Usage
read.agdex.result(res.file)

Arguments
res.file the name of output file of AGDEX analysis

Value
Returns a list of objects of result.
dex.compA comparison definition for "A"
dex.compB comparison definition for "B"
gwide.agdex.res genome-wide AGDEX result
gset.res gene-set Results
meta.dex.res Individual Matched-Gene results
dex.resA Individual Gene Results for comparison "A"
dex.resB Individual Gene Results from comparison "B"
write.agdex.gset.details

write the output of get.gset.result.details to a tab-delimited text file.

Description

A function to write an output file with detailed AGDEX gene-set result.

Usage

write.agdex.gset.details(gset.details, out.file)

Arguments

- gset.details: result produced by get.gset.result.details
- out.file: name of the output file

Value

the path and name of output file of the result of gene-level agdex analysis

Author(s)

Stan Pounds<<stanley.pounds@stjude.org>; Cuilan Lani Gao<<cuilan.gao@stjude.org>>

See Also

agdex; get.gset.result.details; read.agdex.gset.details
Examples

```r
# load the saved result run agdex routine
data(agdex.res)

# obtain all gene-set result
gset.res.all <- get.gset.result.details(agdex.res,gset.ids = NULL, alpha=0.01)

# obtain the gene set result of member genes

gset.res@ <- get.gset.result.details(agdex.res,gset.ids = "DNA_CATABOLIC_PROCESS", alpha=0.01)

# write the gene set details to an output file
## Not run:
write.agdex.gset.details(gset.res@, "gset.details.txt")
## End(Not run)
```

write.agdex.result  Write the AGDEX results to output file

Description

A function to write the results of an AGDEX analysis to a tab-delimited text output file that can be viewed in Excel or re-imported with the function `read.agdex.result`.

Usage

```r
write.agdex.result(agdex.res, out.file)
```

Arguments

- `agdex.res`: result object produced by the `agdex` function
- `out.file`: name of the output file

Author(s)

Stan Pounds<<stanley.pounds@stjude.org>; Cuilan Lani Gao<<cuilan.gao@stjude.org>>

See Also

- `read.agdex.result`
- `agdex`

Examples

```r
data(agdex.res)

## Not run:
#set the wording dictionary
setwd("localWorking dictionary")

#write the agdex result to an out file
\dontrun{
write.agdex.result(agdex.res, "agdex.result.txt")
}
```
#read the result file stored on dist back into R
agdex.res2 <- read.agdex.result("agdex.result.txt")

## End(Not run)
Index

Topic **misc**
  gset.data, 8

agdex, 2, 3, 6–8, 10–16
AGDEX-package, 2
agdex.res, 6
agdex.scatterplot, 2, 5, 6, 11

ExpressionSet, 5, 10, 11, 13

GeneSetCollection, 5, 8, 11
get.gset.result.details, 2, 5, 7, 7, 11, 14,
  15
gset.data, 5, 8, 10, 12, 13

human.data, 5, 8, 9, 12, 13

make.dex.set.object, 2, 10
map.data, 5, 8, 10, 11, 13
mouse.data, 5, 8, 10, 12, 12

read.agdex.gset.details, 2, 5, 8, 11, 13, 15
read.agdex.result, 2, 5, 14, 16

write.agdex.gset.details, 2, 5, 8, 14, 15
write.agdex.result, 2, 5, 11, 15, 16