RDAM Documentation

Release V1.0a

Michel Bellis

December 08, 2010

CONTENTS

CHAPTER

ONE

INTRODUCTION

* RDAM is the acronym of **Rank Difference Analysis of Microarrays**.

* Microarrays are devices which allow to measure simultaneously the expression level of thousands of genes in a particular biological condition. If several samples of the same biological conditions are at disposal, we speak of *replicates*.

* RDAM is intended to find genes that have a *statistically significant difference of expression level* when we compare replicates of two different biological conditions, e.g. four different *cancerous* liver biopsies versus three different *healthy* liver biopsies.

* RDAM can be applied in other situations where the same type of measure is applied to huge replicated series of different items observed in two different conditions (**multiple paired tests with replicates**).

* Bioinformatic project: http://code.google.com/p/arraymatic

* Code repository: http://github.com/mbellis

CHAPTER

TWO

TUTORIAL

RDAM uses a non-parametric approach

* Measured values are first **ranked** onto a 0-100 range scale.

    * this allows to normalise all the replicates to a common scale,

    * in graphical representation each interval, or bin, contains the same number of items.

* Variation is expressed as a **rank difference**, and a geometrical transformation of the plot of rank difference versus rank allows to derive a *normalised variation* called *zVar*.

    * this geometrical transformation applied to *two replicates of the same condition* gives the *null distribution of zVar* **common to all the items**,

    * from the null distribution, we are able to estimate, for each item, a p-value for each observed variation in all (or a subset of all) possible pairs of replicates between the two conditions,

    * for each item, the different Zvar and their associated p-values are combined in order to calculate a single Zvar and p-value that sum up the variation of that item between the two conditions.

* Multiple testing is taken in account, and **false discovery rate (Fdr)** and **sensitivity** are estimated in order to make adapted selection of items.

    * Fdr is the percentage of false positives in the final selection,

    * Sensitivity is the percentage of really varying items that are present in the final selection.
    Knowledge of these two statistics allows the user to select the most convenient compromise | between *specificity* and *sensitivity*

2.1 Installation

Create a directory (e.g. rdam) and unzip rdam_1.0.zip in this directory. Update the Matlab path. Run the main function demo.m from the command line in Matlab. Eventually, edit demo.m to modify some of the following parameters (see *RDAM, INPUT PARAMATERS*).

```
| CompScheme={[1,2;1,2];[1,2;2,1]};
| TGRankList= [2,3];
| CGRankList=[4,5];
| LoadDataFlag=1;
| RankThreshold=[0,0];
| CalibType='idem';
| ClearIndex=[];
| NormType='quantile';
| AnalyseType='transcriptome';
| SizerFittingDegrees=7;
| SingleCalibPointFlag=0;
| SingleCalibCurveFlag=0;
| CalibUpdateFlag=0;
| CalibSaveFlag=1;
| DisplayFlag=1;
| ComparisonFlag=1;
| ResRank=1;
| CalibSchemeFlag=0;
```

RDAM uses Sizer to calculate smoothed approximation of curves. Sizer is developped by J.S. Marron (http://www.unc.edu/~marron/marron.html). The needed scripts are in the sizer folder.

The following figures are the output of the demo function when DisplayFlag parameter is set to 1.

2.2 Data format

Signals are ranked on a [0-100] scale that means that we consider as important not the signal by itself but only its position in the whole distribution. As stated in the introduction, this simple procedure allows to normalize all the points on a common scale, and to populate each rank interval with the same number of points in graphical representations (as shown by the black lines which shows the cumulative frequency of points in figure 1).

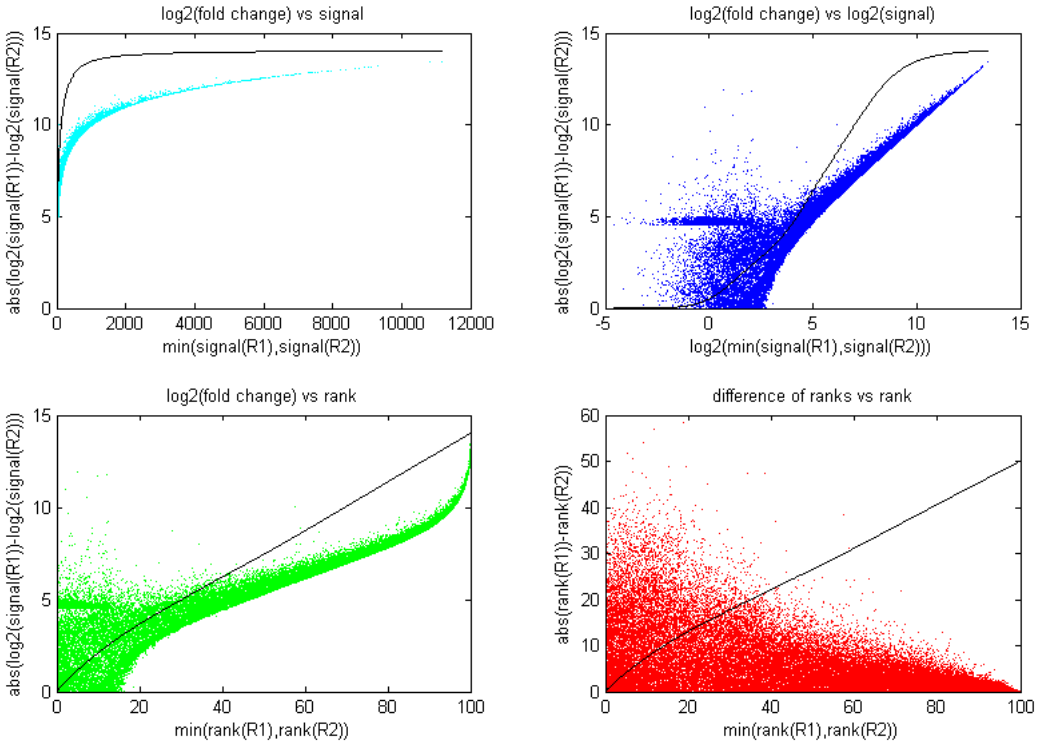

fig.1 Different type of plots displaying the variation between two replicates R1 and R2

2.3 Used data

Data are drawn from an experiment using the Oryza sativa Affymetrix chip.

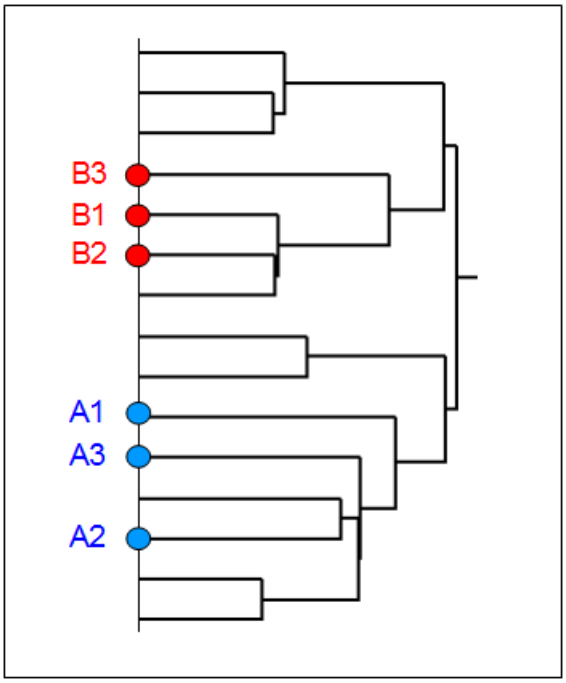

fig.2 Dendrogram of used samples

Two biological conditions, called A and B, are used, each with three replicates respectively called (A1,A2 and A3 in blue and B1,B2 and B3 in red).

From the dendrogram, it can be seen that B3 could be considered as an outlier, and that reproducibility of Bs is globally better than reproducibility of As.

Points are arranged in the following order, A1,A2,A3,B1,B2,B3 and stored in DataRanks, a 57381 x 6 matrix.

2.4 Indication of comparisons

The individual comparisons to be made, and the way their results are to be combined is indicated by the CompScheme parameter(see *RDAM, INPUT PARAMATERS*).

```
| CompScheme={[1,2;1,2];[1,2;2,1]};
```

This value of the CompScheme parameter indicates that two composit comparisons will be made successively, and that each composit comparison is made of two independant comparisons. A composit comparison of K independant comparisons is represented by a 2xK matrix, the upper and lower lines of which listing respectively the test (T) and the control C) points, in the K individual Tk vs Ck comparisons. With this convention, [1,2;1,2] means that the two independant comparisons to be made in the first composit comparison are respectively T1 vs C1 and T2 vs C2 ([1,2;2,1] points therefore to T1 vs C2 and T2 vs C1 comparisons).

As the numbering of the involved points is arbitrary at this step, we need to indicate what is the correspondance between the used numerotation and the real rank of the points in the DataRanks variable.

```
| TGRankList= [2,3];
| CGRankList=[4,5];
```

TGRanklist = [2,3] and CGRankList = [4,5], indicates that test points numbered T1 and T2 correspond respectively to columns 2 and 3 (A2 and A3) and that control points numbered C1 and C2 correspond respectively to columns 4 and 5 (B1 and B2).

2.5 ZVar construction with calibration curves

Last panel of figure 1 shows that the distribution of variation (rank difference) is greatly conditionned by the strength of the signal (variation distribution is considerably noisier for low ranks than for large ranks). In order to obtain a **single distribution for all the items**, we apply a transformation to rank difference, and construct ZVar, a variation which distribution is more equal and almost independant of the signal (see fig. 4 and 7). To make effective this kind of transformation we use calibration curves that are traced onto rank difference versus rank plots. Two types of calibration curves can be used, either **percentile curves** or **mean and standard deviation curves**.

```
| CalibType='idem';
|
```

'idem' value of the CalibType parameter indicates that we assume that the distributions of positive and negative variation are identical, and that we can mix them (this hypothesis stands well when we use replicates of the same biological conditions).

2.5.1 Mean and std curves

This is the method originally developped in RDAM (see *Martin DE et al. 2004*)

Tracing curves

```
| NormType='standardization';
```

'standardization' value of the CalibType parameter indicates that we use mean and standard deviation of rank difference as calibration curves.

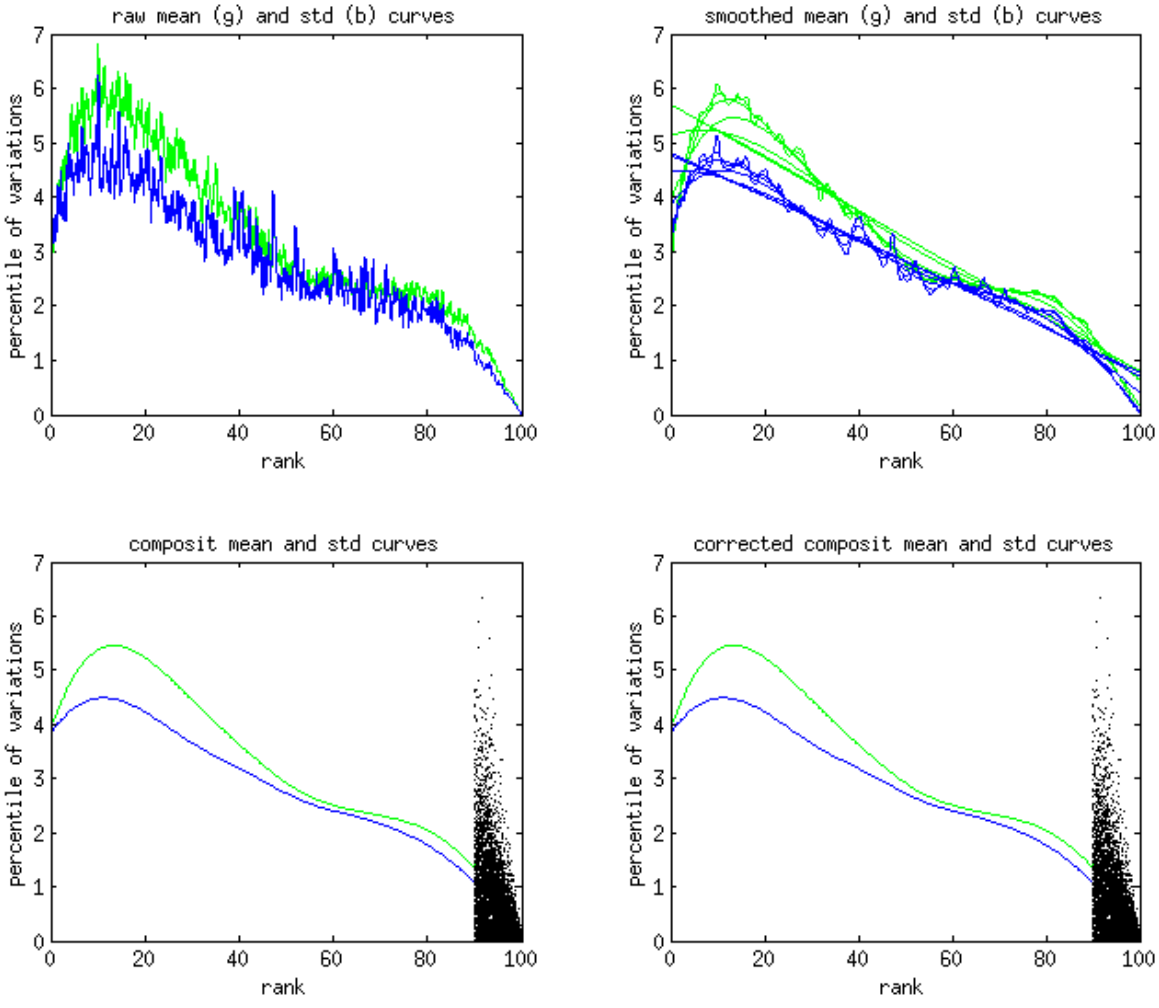

fig.3 Mean and std calibration curves. The different plots show how the two curves calculated locally with a sliding windows (first panel) are smoothed by using the third (see panel two) of the seven curves interpolated with Sizer (if SizerFittingDegrees=7).

Calculating ZVar

The transformation is a classical standardization (ZVar= Var - mean (Var) /std(Var)).

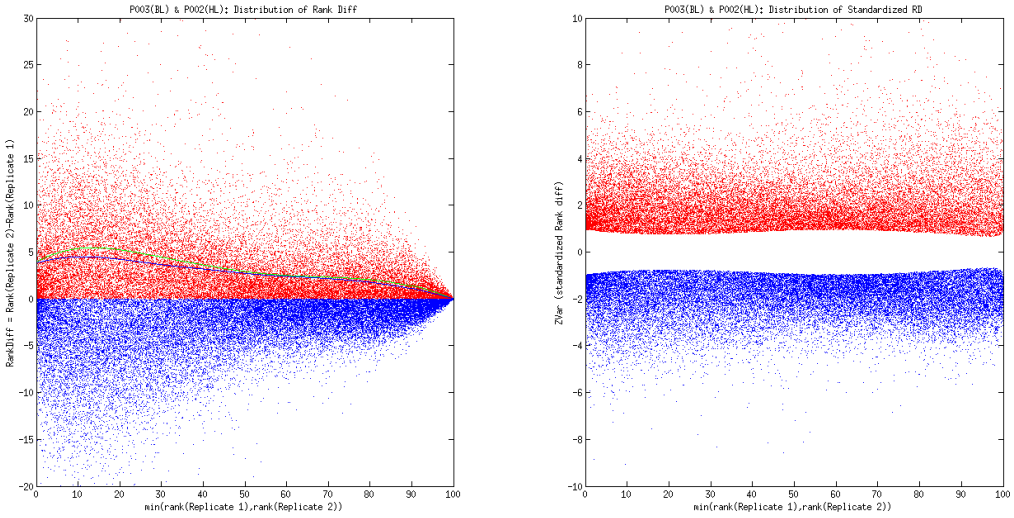

fig.4 ZVar construction with mean and std curves

2.5.2 Percentile curves

This second method is the one we recommend because it is of more general application than the first method.

Tracing curves

```
| NormType='quantile';
```

'quantile' value of the NormType parameter indicates that we use the following quantile curves [0.10:0.10:0.90,0.95,0.99,0.999,0.9999].

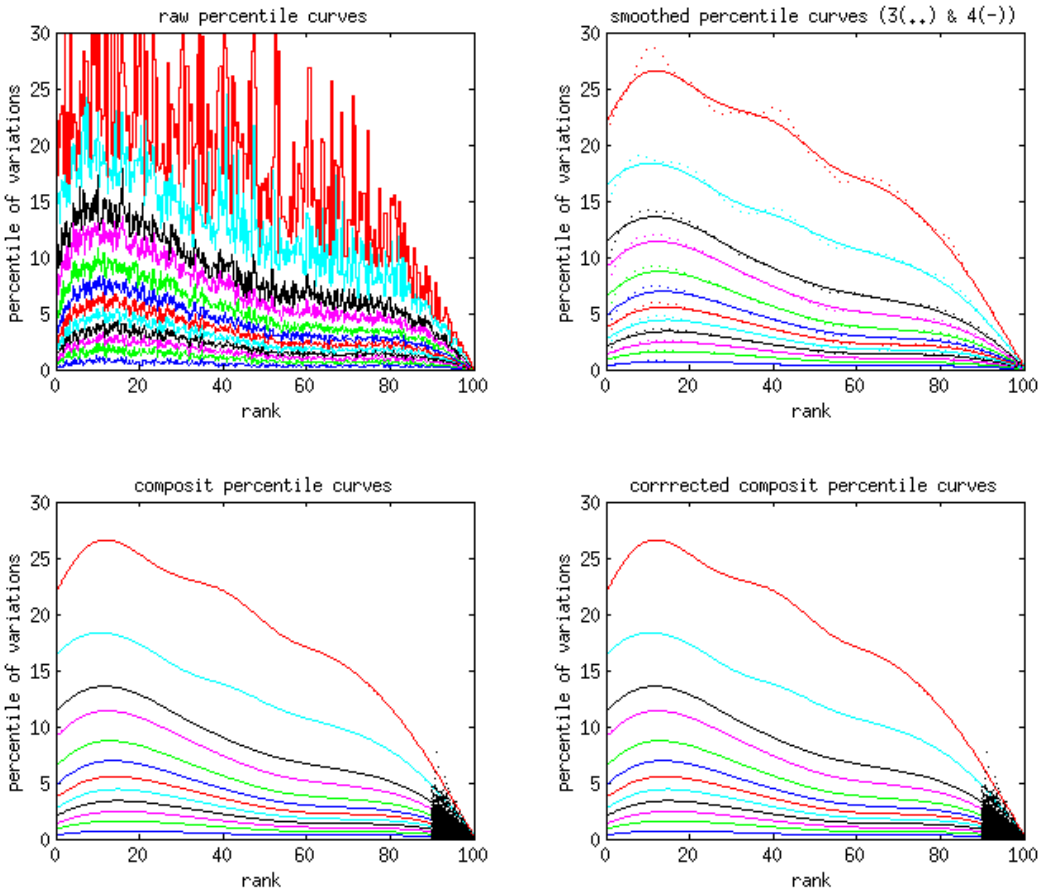

fig.5 Percentile curves. The different plots show how the quantile curves calculated locally with a sliding windows (first panel) are smoothed by using the third (see panel two) of the seven curves interpolated with Sizer (if SizerFittingDegrees=7).

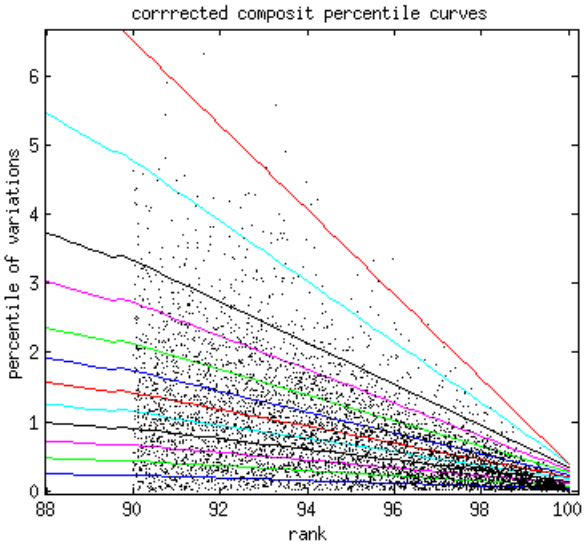

fig.6 Zoom at the right extremity of the fourth panel. RDAM make a correction at the end of the curves to make sure that interpolated quantile curves do not cross each other.

Calculating ZVar

The transformation is in this case a geometrical one,and is a simple linear transformation which bring each quantile curve onto the horizontal line that is tangent at its maximal value.

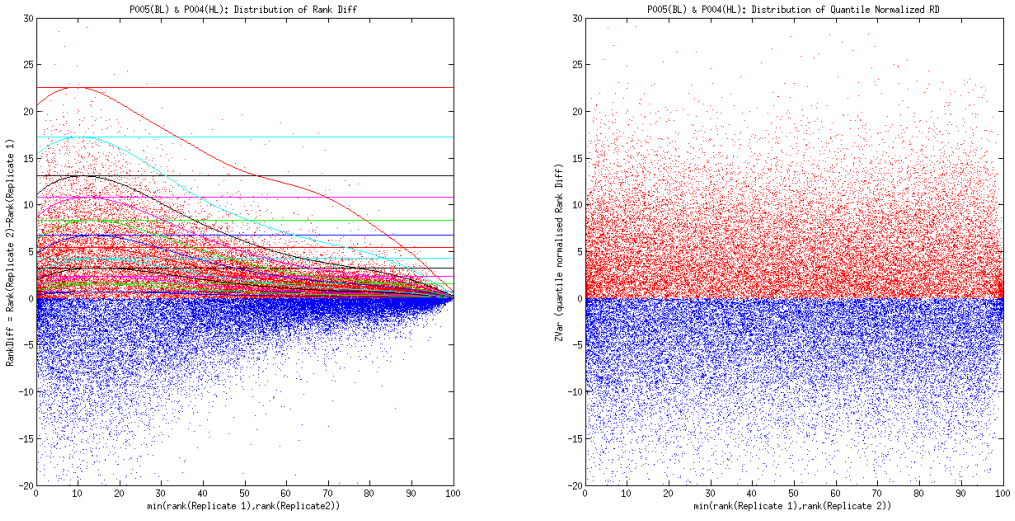

fig.7 ZVar construction with quantile calibration curves

2.6 Statistically significative variations

To calculate the p-value of variations in an independant comparison, RDAM uses two replicates of the same condition to recover the distribution of ZVar in case of null hypothesis. But for a given comparison, for example the comparison T1 vs C1, there exist two possibilitities , and we can take as null hypothesis either T2 vs T1 or C2 vs C1. We use the distribution which is the most conservative (i.e. which minimize the number of significative variations).

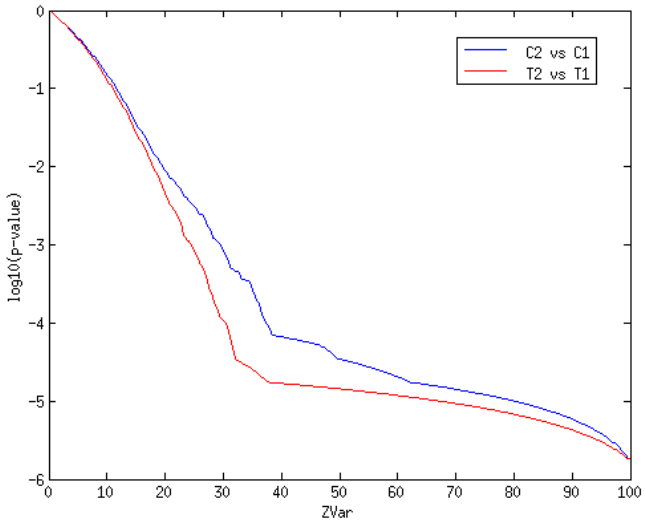

fig.8 Distribution of ZVar for the two null hypothesis (T2 vs T1 and C2 vs C1). The p-values calculated with the blue curve are higher (less significative) than the ones calculated with the red curve, which is expected since condition A used as control (C1=A2, C2=A3) is less reproducible than condition B used as test (T1=B1, T2=B2). This curve will be therefore used to assign p-values to the ZVar calculated in e.g. T1 vs C1, in order to minimize false positives.

2.6.1 Ppv vs ZVar

Using the distribution of ZVar in null hypothesis (fig.8), RDAM assigns a p-value for each ZVar in all independant comparisons that constitute a composit comparison. Product of these p-values (Ppv) is used as a new statistics which null distribution is known (product of independant uniform distributions). Several algorithms are applied to first resolve conflicts between comparisons (some genes can be set as decreased in a a subset of comparisons and increased in another subset), and secondly to assure that there exist a monotonous relation between the different statistics (e.g. Ppv vs Zvar). These two kind of correction are applied when several individual independant comparisons are combined, and when several composit comparisons are combined.

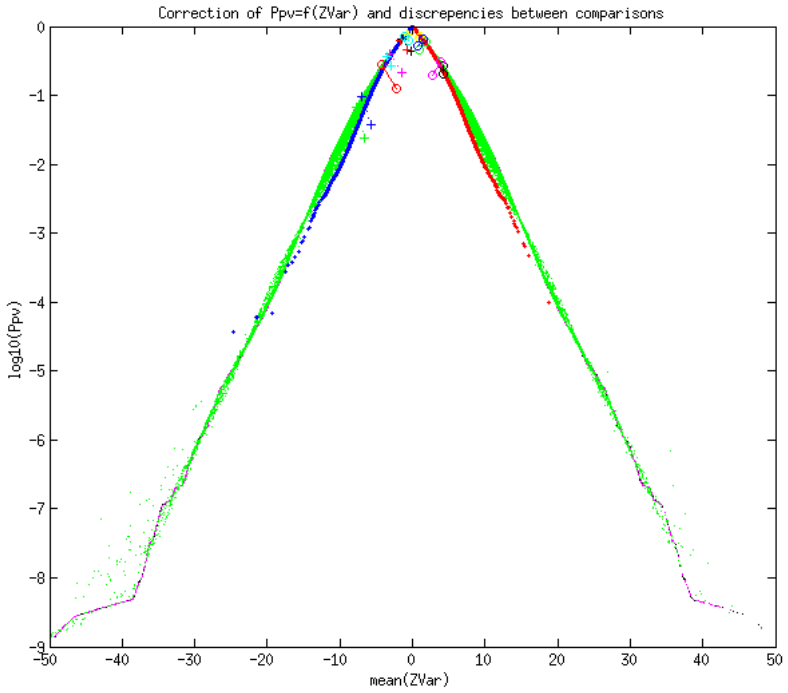

fig.9 Ppv vs ZVar. The figure shows the effect of corrections in a composit comparison with two independant comparisons. Seven points which had different increase/decrease assignation between the two independant comparisons are displayed before correction (mean of ZVar, and Ppv) and after the first correction. The red and blue curves shows the monotonous relation that exist between Ppv and ZVar, once the second correction has been applied to the true values, plotted in green.

2.6.2 Reproducibility

The following figure shows how good is the reproducibility of the whole process. Red and blue colors indicates the direction of variation, respectively increased and decreased, as it is delivered at the end of the whole RDAM process. In particular is shown the very good reproducibility of ZVar between two independant comparisons (second panel), and that only a small subset of rank differences that have low values in a given independant comparison, have their direction of variation changed (fourth panel).

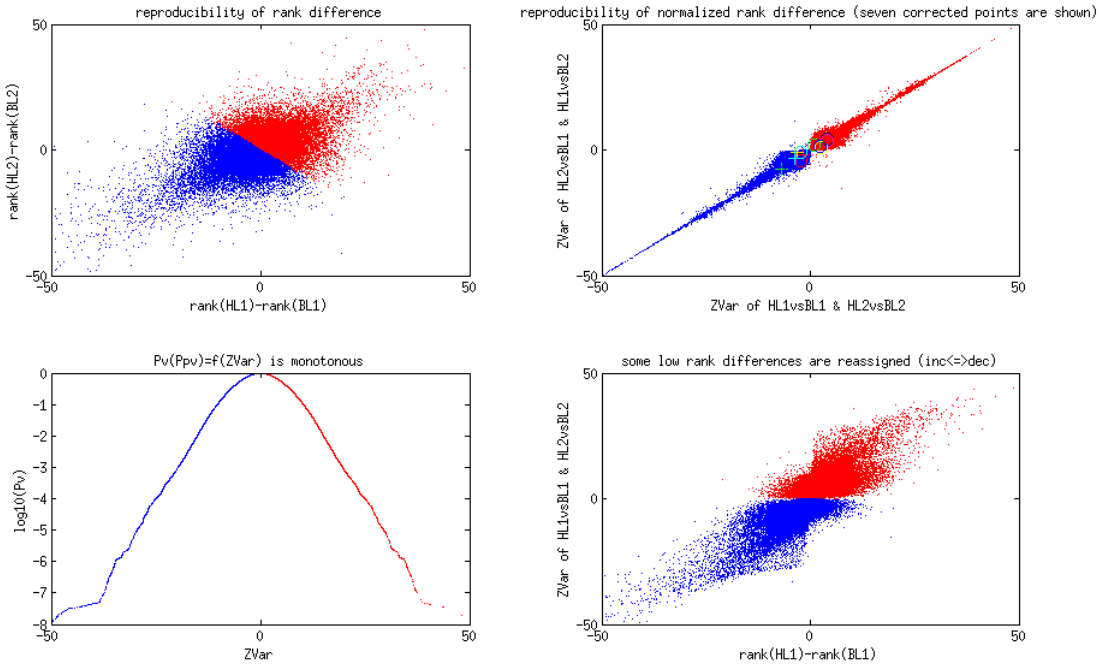

fig.10 Reproducibility.

2.6.3 Statistics versus ZVar

The following figure shows the different statistics calculated onto a composit comparison, and how the corrections discussed before allows to obtain monotonous relationships between these statistics and ZVar.

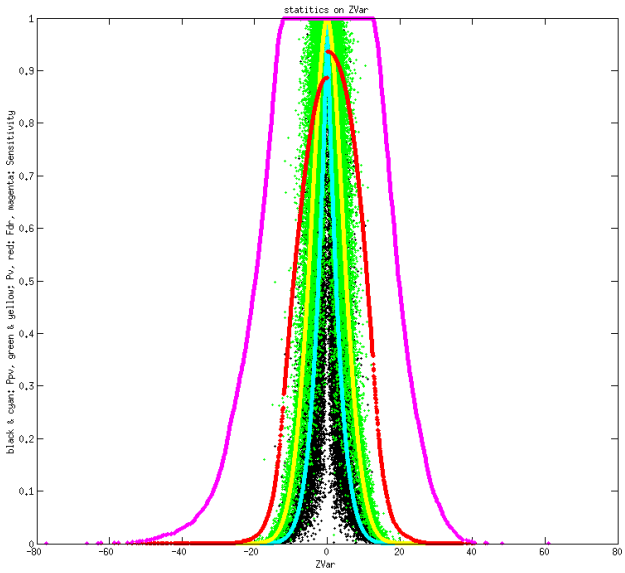

fig.11 Statistics. Before correction: Ppv (black points) and p-values (green points). After correction: Ppv (cyan), p-values (yellow curve), false discovery rate (red curve) and sensitivity (magenta curve).

2.6.4 Selection abaccus

An algorithm makes an estimate of the total variation (crossed circle, and numerical values in fig.12). False discovery rate (Fdr) and sensitivity are estimated by using total variation estimate. By combining Fdr and sensitivity, user can optimize his selection, by doing the right adjustement between sensitivity and specificity.

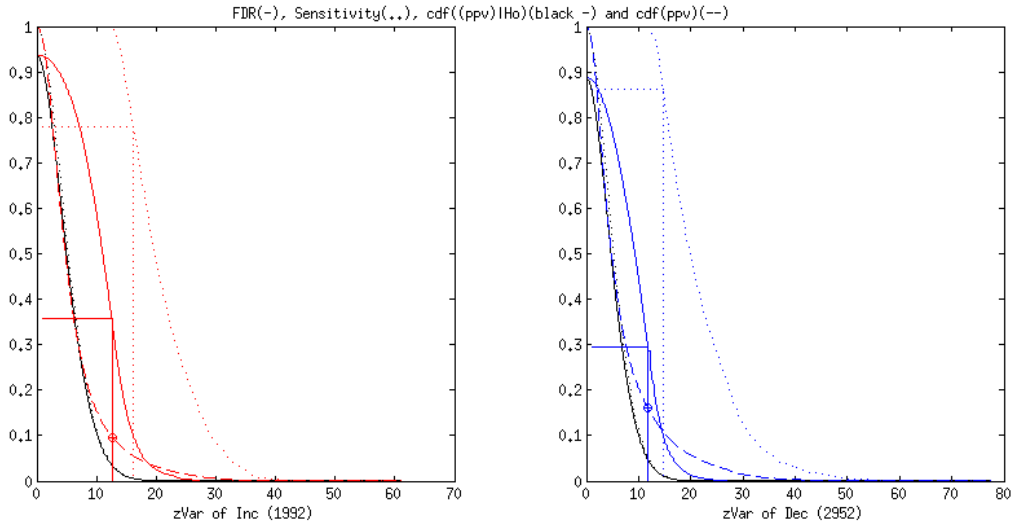

fig.12 Selection abaccus (increased items on left panel, decreased items on right panel). The main curves to be considered are the Fdr (red continuous curve) and the sensitivity (red dotted line). On the left panel, by using these curves, we can see that a a selection at Fdr=10| will recover around 80| of the estimated variation, i.e. 0.8*1992~=1600 true positives. As the Fdr is set to 10| , the total number of selected item should be 1600/0.9~1780 (see the red dotted reference line which vertical part crosses Fdr at 0.1 and Sensitivity at ~0.8).

CHAPTER

THREE

REFERENCE MANUAL

3.1 DEMO

```
================
 FUNCTION DEMO
================

  DEMO runs a demonstration
  demo must be run from inside the main directory:
  'cd .../RDAMm/'
  'demo'

  GLOBAL VARIABLES
   DataRanks contains data ranks (if LoadDataFlag==1,  DataRanks is empty)
   P contains metadata (P.point : description of points, P.biol: description of
     biological conditions ...)
   S contains calibration sets

  Comment/uncomment lines to test different combination of parameters as indicated
```

3.2 RDAM

```
================
 FUNCTION RDAM
================

  RDAM implements Rank Difference Analysis of Microarray algorithm:
  Find which variations are statistically significative when comparing two groups
     of points, each point being a series of rank values. One group is called the test group (TG)
     and the other is called the control group (CG). The comparison is TG vs CG.  Increased
    (devreased) probesets are those which are higher (lower) in the test group.
  A group which contains several points is not 'reduced' to a one-point
  group by using e.g. the mean of all the points.
      Therefore, there exist multiple way for doing the comparison between two groups which have
      variable number of replicates. For exemple if two points are available in each condition
      (TG1,TG2 in the TG group and CG1,CG2 in the CG group, we can do either a 'parallel'
      comparison (that is a unique one-to-one comparison as with paired sample that uses only
      TG1vsCG1 and TG2vsCG2). In this case one set of two comparisons is made. In each comparison
      a p-value is calculated for each probeset, and combined (e.g. by doing their product)
      to give the finale statistical values (p-values, FDR, sensitivity). If three, four ...
      replicates were available, we would have done one set of three, four ... comparisons.
```

```
    If the samples are not paired, we can do crossed comparisons that uses all possible
    comparisons(TG1vsCG1, TG2vsCG2, TG1vsCG2, TG1vsCG2. In this case two sets of two comparisons
    are made.
    If the number of replicates is higher, there exist more possibilities and a comparison
    sheme is necessary in order to indicate how many sets of comparisons are to be made,
    and which comparisons are made in each set.
The comparison scheme is coded in a cell array : { 1st set of comparisons, 2nd set of
    comparisons, ...}. A set of comparisons indicates the points that must be used in the
    TG and CG group to do each comparison
    {[1,2;1,2]} indicates that the first set of comparisons is TG1vsCG1 and TG2vsCG2
    {[1,2;1,2],[1,2;2,1]} indicates that the first set of comparisons is TG1 v sCG1 and
    TG2vsCG2 and the second set of comparisons is TG1vsCG2 and TG2vsCG1
For each comparison
    a calibration set must be used (calculated by the noise_distribution script).
    (which allows two things:
    first it is a way of normalizing the variation (Var => zVar, e.g. Rank Difference (RD) =>
        normalized RD (zRD), either by the standardization procedure (orignal method of RDAM),
    or by a more sophisticated algorithm (surface mapping),
    and secondly it gives the p-value for any zVar value (PvCurve))


INPUT PARAMETERS

 1           CompScheme : The comparisons to be made
 2            TGRankList: list of data ranks used in the comparisons
                          if TGRankList=[12,15,23], the Data used for TG point in position 1 is
                          DataRanks(:,12)
 3            CGRankList: list of data ranks used in the comparisons
 4          LoadDataFlag: = 1 indicates that each data point must be loaded when to be used
                         (no permanent storage of DataRanks in the memory)
 5         RankThreshold: a vector of two values allowing to start the process of
                          constructing quantile curves using
                          a defined range of rank values (>=RankThreshold(1)&<=RankThreshold(2))
 6             CalibType: indicates the type of couple of two replicates used to calculate the
                          calibration set for the current comparison. The couple noted {G1,G2},
                          which is passed as parameter, indicates the G1 and G2 will be
                          identified to the HL and BL lines, respectively, in the
                          noise_distribution function.
                          If (TG1,TG2) and (CG1,CG2) are the two couples of replicates for the
                          TG and CG condition, respectively, we have three possibilities:
                            CalibType='idem' =>  {TG1,TG2} or {CG1,CG2} is used,
                            CalibType='up' =>  {TG1,CG1} or {TG2,CG2} is used,
                            CalibType='down' => {CG1,TG1} or {CG2,TG2} is used.
                          A rule allows to choose the right couple from the two existing ones
                          (FirstCouple, SndCouple).
 7            ClearIndex: allows to clear some points from the points used in the calibration
                          process (e.g. probe sets that are linked to gender can'b eliminated
                          from calibration process in case where points used as replicates in
                          the calibration process belong to different gender)
 8              NormType: either 'standardization' (original method in RDAM = var - mean /std)
                          or 'quantile' (more general method based on percentile curves)
 9           AnalyseType: either 'transcriptome' or 'chipchip' (some parameters have to be adapted
                          to the type of analysis)
 10  SizerFittingDegrees: used by sizer (the procedure of curve smoothing).Indicates the number
                          of increasing fitting degrees used
 11 SingleCalibPointFlag: = 1 indicates that the same TG (CG) point is used to construct
                          the pairs of TG (CG) points that are considered as replicates in
                          the calibration process (noise_distribution script)
```

```
 12 SingleCalibCurveFlag: = 1 indicates that a single calibration set is used for all the
                          comparisons
 13      CalibUpdateFlag: indicates if noise_distribution must be run even if calibration set
                          already exist (should be 0 if clearIndex=[] unless calculus must be
                          done again for a special reason; important if ClearIndex is not empty,
                          because an existing calibration set can be different)
 14        CalibSaveFlag: indicates if the output of noise_distribution must be saved or not
 15          DisplayFlag: indicates if figures must be drawn or not
 16       ComparisonFlag: indicates if the comparison must be done (if not the function is
                          simply used to construct calibration sets which are stored in
                          S{CalibRank})
 17              ResRank: Calibration set is stored in S and saved as sprintf
                          (CalibSet_02u,ResRank) (allows to disconnect construction of points
                          dendrogram from calculus on biological conditions which allows to
                          add easily new points).
 18      CalibSchemeFlag: if equal to 1 indicates that a particular combination of
                          calibration set is used (e.g. in double chanel technics)
                          if equal to 2, indicates the correspondance between TGRankList and
                          CGRankList and the real position in S (used when P.flag.loadData==1
                          to allow a correct selection of calibration set)

[ZVar,Pv,Ppv,Fdr,Sensitivity,TotalVar]=rdam({[1,2;1,2];[1,2;2,1]},[1,2],[3,4],0,[0,0],'idem',[],
                                    'quantile','transcriptome',7,0,0,0,1,0,0,1,0);

VARARGIN PARAMETERS
 if SingleCalibPointFlag==1: HLCalibRank = varargin{1} & CGCalibRank = varargin{2} indicates
                             the ranks of points that are systematically used to construct pairs
                             of  calibration points (HLCalibRank (CGCalibRank) must not be equal
                             to any of the points contained in TGRankList (CGRankList))
 if SingleCalibCurveFlag==1: only one calibration set is used zval=varargin{1} pv=varargin{2}
      if CalibSchemeFlag==1: CalibScheme (varargin{}) indicates which calibration set to use for
                             each comparison

OUTPUT PARAMETERS
 1        ZVar: normalized variation (from 0 to 100 for INCREASED and -100 to 0 for DECREASED)
 2          Pv: p-value of Ppv  (from 0 to 1 for INCREASED and -1 to 0 for DECREASED)
 3         Ppv: product of p-values (from 0 to 1 for INCREASED and -1 to 0 for DECREASED)
 4         Fdr: false discovery rate
 5 Sensitivity: from 0 to 1 for INCREASED and -1 to 0 for DECREASED
 6    TotalVar: estimate of the total number of increased and decreased probe sets

GLOBAL VARIABLES
 DataRanks contains data ranks (if LoadDataFlag==1,  DataRanks is empty)
 P contains metadata (P.point : description of points, P.biol: description of
   biological conditions ...)
 S contains calibration sets
```

3.3 QUANTILE CURVE

```
============================
 FUNCTION QUANTILE_CURVES
============================

  QUANTILE_CURVES calculates quantile curves:
  Variations (rank differences) are sorted according to the rank to which they correspond.
  Successive, range of ranks are selected to calculate either the mean, the std or several
```

```
  percentiles of all the positive variations (indexed by the SelIndex parameter) contained
  in the current range of ranks. The percentile that are calculated correspond to the
  fraction stored in PercRange. The ranges are determined by a sliding window of size
  WinSize, shifted by steps equal to WinStep.
  The current X position corresponds to a rank. The first occurence of this rank
  is the current position.
  The positive variations values used for calculating the fractiles are put in the range
  of positions going from the position used in the previous step (the value of which is
  stored in MemAllPosition) to the actual current position. The window size is constant
  but the process act on a selection of point (corresponding to positive variations),
  so in the final result a window is represented by rank intervals of different length

  INPUT PARAMETERS
   1               Rank: ranks
   2                Var: variations (rank diff)
   3        AnalyseType: either 'transcriptome' or 'chipchip'
   4           SelIndex: position of positive variations used
   5      RankThreshold: range of rank (in general at the beginning of the rank range)
                         to be processed as the first window position (can be empty)
   6            WinStep: window step
   7            WinSize: window size
   8          PercRange: the fraction to which percentile must be calculated
   9 SizerFittingDegree: the number of different fitting degree used by SIZER
   10       DisplayFlag: indicates if figures must be drawn or not

  OUTPUT PARAMETERS
   1 OutputRes: structure with normalized variations
   2      Rank: rank of positive variations (indexed by SelIndex)
   3       Var: positive variations
   4  RankGrid: sampling range of ranks [0.25:0.25:100]
   5      Grid: smoothed variation curves corresponding to RankGrid

  OUTPUTRES STRUCTURE
       OutputRes.rank: rank of the X values
       OutputRes.perc: series of raw percentile curves in the sliding window
    OutputRes.fitperc: series of smoothed percentiles curves (SIZER)
       OutputRes.mean:  mean of the variation in the sliding window
    OutputRes.fitmean: smoothed mean (SIZER)
        OutputRes.std: std of the variation in the sliding window
     OutputRes.fitstd: smoothed std (SIZER)
    OutputRes.stepvar: standardised variation (original RDAM method) in the sliding window
     OutputRes.fitvar: smoothed standardised variation  (SIZER)
   OutputRes.fraction: percentage of positive variation in the sliding window
   OutputRes.gridperc: smoothed percentiles curves corresponding to a sampling range of
                       ranks [0.25:0.25:100]

  VARIABLE NAMES
   Pos: position (index)
     F: filtered (selected with SelIndex)
     S: sorted
```

3.4 NOISE DISTRIBUTION

```
==============================
 FUNCTION NOISE_DISTRIBUTION
==============================
```

```
NOISE_DISTRIBUTION calculates noise distribution:
The distribution of the variation between two series of rank values (one called a baseline (BL)
and the other called a highline (HL)) is computed
The units used for representing the variation  is the difference between the ranks of values
(rank diff)
The script allows to compute the values of the variation unit for which there is a predetermined
percentage (threshold of 10, 3 and 1) of points with a greater variability : these are the
calibration curves of the percentiles (or fractiles)
These curves which are irregular, because they are obtained by computing the points in a sliding
window, are smoothed by interpolation (spline)

INPUT PARAMETERS
 1             HLValues: the highline values
 2             BLValues: the baseline values
 3        RankThreshold: a vector of two values allowing to start the process of constructing
                         quantile curves using a defined range of rank values
                         (>=RankThreshold(1)&<=RankThreshold(2))
 4               HLRank: the rank of point used as highline
 5               BLRank: the rank of point used as baseline
 6           ClearIndex: allow to clear some points from the points used in the calibration
                         process
 7             NormType: either 'standardization' (original method in RDAM = var - mean /std)
                         or 'quantile' (more general method based on percentile curves)
 8         AnalyseType : either 'transcriptome' or 'chipchip' (some parameters have to be
                         adapted to the type of analysis)
 9            CalibType: the type of data, either
                         up or down: distinct biological conditions
                         up: only the values which are increased in the HL vs BL comparison
                             are used
                         down: only the values which are decreased in the HL vs BL comparison
                               are used
                               used for chiphip analysis type and for biological conditions
                               without replicates
                         idem: replicates of the same biological condition
                               [HL,BL] is compared to [BL,HL] (no distinction between increased
                               and decreased distribution which should be equal)
 10 SizerFittingDegrees: used by sizer (the procedure of curve smoothing).Indicates the number of
                         increasing fitting degrees used
 11         DisplayFlag: indicates if figures must be drawn or not

OUTPUT PARAMETERS
 1 RankGrid: sampling range of ranks [0.25:0.25:100]
 2     Grid: smoothed variation curves corresponding to RankGrid
 3 ZVarGrid: Normalized variations used for 2-D interpolation
 4     ZVar: normalized variations (ZVar=interp2(RankGrid,VarGrid,ZVarGrid,Rank,Var))

VARIABLE NAMES
  Val: values (ranks)
  Var: variations (rank differences)
 Perc: percentile
  Pos: positive variations
  Neg: negative variations
   S~: sorted values
   P~: values that keep the correspondance with sorted value

SORT USING
 Y=f(X) => [SX,SortIndex]=sort(X) => PY=Y(SortIndex)
 the original order can be find with : [temp ReverseIndex]=sort(SortIndex) => X=SX(ReverseIndex)
```

```
   DIFFERENCE BETWEEN INDEX AND BINDEX:
   a=[0,2,45,0]
   Index=find(a==0) => Index=[1,4]
   Bindex=a==0 => Bindex=[1,0,0,1]
```

3.5 FILL_S

```
========
FILL_S
========

 FILL_S fills the S strcture

 INPUT PARAMETERS

 1  NormType: either 'standardization' (original method in RDAM = var - mean /std)
              or 'quantile' (more general method based on percentile curves)
 2 CalibType: indicates the type of couple of two replicates used to calculate
              the calibration set for the current comparison.
              The couple noted {G1,G2}, which is passed as parameter, indicates the
              G1 and G2 will be identified to the HL and BL lines, respectively, in the
              noise_distribution function.
              If (TG1,TG2) and (CG1,CG2) are the two couples of replicates for the the
              TG and CG condition, respectively, we have three possibilities:
                CalibType='idem' =>  {TG1,TG2} or {CG1,CG2} is used,
                CalibType='up' =>  {TG1,CG1} or {TG2,CG2} is used,
                CalibType='down' => {CG1,TG1} or {CG2,TG2} is used.
              A rule allows to choose the right couple from the two existing ones
              (FirstCouple, SndCouple).
 3      Grid: A regular grid ([0:0.25:100]) on which Zvar and cdf(ZVar) are interpolated
 4      ZVar: normalized variation (from 0 to 100 for INCREASED and -100 to 0 for DECREASED)
 5   ResRank: Calibration set is stored in S and saved as sprintf(CalibSet_02u,ResRank)
              (allows to disconnect construction of points dendrogram from calculus
              on biological conditions which allows to add easily new points).
 6  SaveFlag: indicates if S must be saved
```

3.6 FDR

```
===============
 FUNCTION FDR
===============

  FDR calculates fdr and sensitivity.

  INPUT PARAMETERS
   1        ZVar: normalized variation
   2         Ppv: the product of p-values
   3       PpvPv: the p-value of Ppv (calculated in case of null hypothesis)
   4      PpvCdf: the observed cumulative distribution frequency of Ppv
   5 DisplayFlag: indicates if figures must be drawn or not
   6        FigH: the figure handle
   7      SubPos: subplot position
```

```
  OUTPUT PARAMETERS
   1         Fdr: False Discovery Rate of Ppv
   2 Sensitivity: sensitivity of Ppv
   3   TruePosNb: estimated true variation
```

3.7 PRODUNIFORM

```
=======================
 FUNCTION PRODUNIFORM
=======================

  PRODUNIFORM calculates the product of several independant uniform random variables.

  INPUT PARAMETERS
   1 ProdNb: the number of variables in the product

  OUTPUT PARAMETERS
   1 PpvList: the list of Ppv
   2  PPvCdf: the probabilities for Ppv

   COPYRIGHT
   Translated from C program with this Copyright
         Copyrights Orestis Georgiou   25-8-09

         This program is related to the paper:
         Product of n independent Uniform Random Variables.

          When Theorem 1. of the above paper is integrated, the incomplete gamma
          function, Gamma[n,x] is obtained which for integer n can be expressed
          analytically. Thus, this program calculates the probability that a random
          variable X, consisting of the product of n independent and identically
          distributed uniform [a,b] random variables X_{i}, i=1,2..n, takes on a value
          less than or equal to tau.
```

3.8 CUMUL DISTRIBUTION

```
====================
 CUMUL_DISTRIBUTION
====================

  CUMUL_DISTRIBUTION calculates a cumulative frequency distribution.

  INPUT PARAMETERS
   1      Values: the values to be cumulated
   2       Limit: values <=Limit are cleared
   3    FlipFlag: if equal to 1, values are flipped
   4 DisplayFlag: display cumulative curve

  OUTPUT PARAMETERS
   1 SortedFValues: sorted values that are filtered (values <= Limit have been cleared)
   2    FCumulDist: cumulative frequency distribution of filtered values
   3 SortedUValues: sorted unique values (doublons in filtered values have been cleared)
   4    UCumulDist: cumulative frequency distribution of unique values
```

```
   5      MinValue: minimal value of filtered values (and of unique values by the way)
   6      MaxValue: maximal value of filtered values (and of unique values by the way)
```

3.9 MAKE MONOTONOUS

```
============================
 FUNCTION MAKE_MONOTONOUS
============================

  MAKE_MONOTONOUS makes a vector of values monotonous either by keeping
  the last greatest or last smallest value.
  Outliers are eliminated before by a simple procedure (deviation from mean local value).

  INPUT PARAMETERS
   1      Val : values that must be rendered monotonous
   2     Type : either inc (monotonous increasing) or dec (monotonous decreasing)
   3 FlipFlag : if ==1 flip the data before making them monotonous

  OUTPUT PARAMETERS
   1 Val : values made monotonous
```

3.10 SORT2SERIES

```
=======================
 FUNCTION SORT2SERIES
=======================

  SORT2SERIES sorts one series of values according to the order of another series of values

  INPUT PARAMETERS
   1 Val1: The first series of values which is sorted
   2 Val2:  The second series of values sorted as the first one (sayed as 'in phase')

  OUTPUT PARAMETERS
   1         SVal1: sorted first series of values
   2         PVal2: 'in phase' second series of values
   3 DirSortIndex1: direct sort index of first series of values (Val1 => SVal1)
   4 InvSortInde1x: inverse sort index of the first series of values (SVal1 => Val1)
   5     S2PIndex2: index allowing to find in phase values from sorted values
                    of the second series of values (SVal2 => PVal2)
```

3.11 SIZER

RDAM uses Sizer to calculate smoothed approximation of curves. Sizer is developped by J.S. Marron.

(http://www.unc.edu/~marron/marron.html).

http://www.unc.edu/~marron/marron_software.html

3.11.1 GPANAL_CALIB

```
==========================
 FUNCTION GPANAL_CALIB
==========================

 GPANAL, General Purpose data ANALysis
 Inputs:
     data    - either n x 1 column vector of density estimation data
                  or n x 2 matrix of regression data:
                              X's in first column,  Y's in second
     vrange  - 3 vector:  use minx as vrange(1) and maxx as vrange(2)
                               and number of grid points as vrange(3)
     nhp     - value is number in family, and rows in SiZer

 Outputs:

 Assumes path can find personal functions:
    gpnpr.m

    Copyright (c) J. S. Marron 1997, 1998
    Modified M Bellis 2001
```

3.11.2 GPNR

```
==================
 FUNCTION GPNR
==================

 GPNPR, General Purpose NonParametric Regression (1-d, Local poly))
     Does 1-d kernel local polynomial (usually linear) regression,
     using binned (default), direct (either matrix, or loops for
     bigger data sets), or moving window (for higher degree)
     implementations, with the bandwidth either user specified
     (can be vector), or data driven (Ruppert Sheather and Wand ROT
      or DPI).  Kernel shape is Gaussian.
   Can use first 1, 2, 3, 4, 5 or 6 arguments.
 Inputs:
     data   - either n x 2 matrix of Xs (1st col) and Ys (2nd col)
                 or g x 2 matrix of bincts, when imptyp = -1
     vh     - vector of bandwidths, or specifies data driven:
                   0 (or not specified)  -  Ruppert, Sheather, Wand DPI
                   -1  -  Ruppert, Sheather, Wand ROT
                       Note: <= 0 only works for imptype = 0
                   >0  -  Use input number (numbers if vector)
                       Note: this (these) MUST be >0 for imptyp >= 1
                                  (the direct implementations)
     vxgrid - vector of parameters for, or values of, grid to evaluate at:
                   0 (or not specified)  -  use endpts of data and 401 bins
                   [le; lr]  -  le is left end, re is right, 401 bins
                          (get error message and no return if le > lr)
                   [le; lr; nb] - le left, re right, and nb bins
                   xgrid  -  Use input values
                             Note:  need to have more than 3 entries,
                                  and only works when imptyp = 1, 2 or 3
    imptyp  - flag indicating implementation type:
                  -1  -  binned version, and "data" is assumed to be
```

```
                                bincounts of prebinned data
                0 (or not specified)  -  linear binned version
                                and bin data here
                1  -  Direct matrix implementation
                2  -  Slow looped implementation (only useful when
                          1 creates matrices that are too large)
                3  -  Moving window implementation (slow, but allows
                          higher polynomial degrees than 1 (linear)
   polydeg - scalar degree of local polynomial.  For imptyp < 3, can
                only use:    0 - local constant (ie. Nadaraya-Watson)
                             1 - local linear (the default)
                For imptyp = 3, can be 0,1,2,...
   eptflag - endpoint truncation flag (only has effect when imptyp = 0):
                0 (or not specified)  -  move data outside range to
                                nearest endpoint
                1  -  truncate data outside range
 For equally spaced X's, and a return at those X's only,
     use all 1's in 1st column of bincts, and Y's in the 2nd
Output:
    (none)  -  Draws a graph of the result (in the current axes)
    npr     -  col vector of heights of kernel kernel density estimate,
                   unless vh is a vector (then have matrix, with
                   corresponding cols as density estimates)
                   Note: when a grid is used where the data are
                   too sparse for a given bandwidth, direct
                   implementations return an interpolation of
                   the data, and binned implementations return
                   an interpolation of the estimate
    xgrid   -  col vector grid of points at which estimate(s) are
                   evaluted (useful for plotting), unless grid is input,
                   can also get this from linspace(le,re,nb)'
    mker    -  matrix (vector) of kernel functions, evaluated at xgrid,
                   which can be plotted to show "effective window
                   sizes" (currently scaled to have mass 0.05, may
                   need some vertical rescaling)

Used by:   gpfam.m

Assumes path can find personal functions:
   vec2mat.m
   gplbinr.m
   bwrswb.m

   Copyright (c) J. S. Marron 1997
```

3.11.3 GPLBINR

```
======================
 FUNCTION GPLBINR
======================

 GPLBINR, General Purpose Linear BINneR (density and regression est.)
     Does linear binning of either density or regression 1-d data,
     to an equally spaced grid.
   Can use first 1, 2, 3 or 4 arguments.
 Inputs:
     data   - either n x 1 column vector of density estimation data
```

```
                or n x 2 matrix of regression data:
                          X's in first column,  Y's in second
    vgridp - vector of grid parameters:
                0 (or not specified)  -  use endpts of data and 401 bins
                [le; lr]  -  le is left end, re is right, 401 bins
                       (get error message and no return if le > lr)
                [le; lr; nb] - le left, re right, and nb bins
    eptflag - endpoint truncation flag:
                0 (or not specified)  -  move data outside range to
                                  nearest endpoint
                1  -  truncate data outside range
   ibtype - flag indicating binning type:
                0 - Simple (histogram) binning
                          Note:  for larger data sets, this is MUCH
                                faster than matlab's HIST.
                1 - (or unspecified) - Linear binning
                          (default, when ibtype not specified)
 Output:
    bindat  - binned data:
                nb x 1 column vector of bin counts for density estimation
                nb x 2 matrix, with (1) bin counts, (2) bin sum of Y's,
                           for regression  {(2) / (1) gives bin avgs}
    bincent - nbin x 1 vector of
                bin centers, can also get this from linspace(le,re,nb)'

 Used by:   gpkde.m
            gpnpr.m
```

3.11.4 INTERP1S

```
======================
 FUNCTION INTERP1S
======================

 INTERP1S, linear interpolation, with constant extrapolation outside
    slight modification of linear version of INTERP1, which
    allows values xi values outside the range of x, and
    returns the closest values at such points.
 Inputs:
    x  -  col. vector of x values of function (assumed increasing)
    y  -  col. vector of y values of function
    xi -  new col. vector of values to plug into function
 Output:
    yi -  linearly interpolated approximate values of f(xi)

    For more details, try "help interp1"

   Copyright (c) J. S. Marron 1997


   Copyright (c) J. S. Marron 1996-1997
```

CHAPTER

FOUR

PUBLICATIONS

Hennetin J, Pehkonen P, Bellis M

Construction and use of gene expression covariation matrix

BMC Bioinformatics (2009) 10: 240

http://www.biomedcentral.com/1471-2105/10/214

Hennetin J, Bellis M

Application of Gene DIVER to the study of geometrical representations of gene expression covariation

Tcbbsupp3 in **Gupta G, Liu A, Ghosh J**

Automated Hierarchical Density Shaving: A robust, automated clustering and visualization framework forlarge biological datasets

IEEE/ACM Transactions on Computational Biology and Bioinformatics, 11 Mar 2008, IEEE Computer Society Digital Library

IEEE Computer Society

http://doi.ieeecomputersociety.org/10.1109/TCBB.2008.32

Hennetin J and Bellis M

Clustering methods for analyzing large datasets: gonad development, a study case

Methods In Enzymology (2006) 411: 387-408

Martin DE, Demougin P, Hall MN, Bellis M

Rank Difference Analysis of Microarrays (RDAM), a novel approach to statistical analysis of microarray expression profiling data

BMC Bioinformatics (2004) 5: 148

http://www.biomedcentral.com/1471-2105/5/148