Computational platforms for integrative analysis in Open source

Srinivasan Ramachandran, Chaitali Paul, Shreya Chakraborty, Srikant Verma, Ab Rauf Shah, Bhanwar Lal Puniya, Rupanjali Chaudhuri , Rahul ShubhraMandal, [OSDD Consortium]*

G.N. Ramachandran Knowledge Centre for Genome Informatics, Institute of Genomics and Integrative Biology, (CSIR), Mall Road, Delhi 110 007, India;* Council of Scientific and Industrial Research, Anusandhan Bhavan, 2, Rafi Marg, New Delhi 110 001.

This is the age of Systems and Integrative Biology. This is the age of big size data. We are faced with new challenges including the speed of transactions and analyzing through algorithms. We may be moving to data driven research as opposed to the standard practice of hypothesis driven research. Computational platforms with capability for carrying out integrative analysis are required for rapid analysis to capture the essential trends hidden in datasets. The R is a High-level interpreted language suitable for developing new computational methods (R Development Core Team. 2010). Many packages for computational biology are being developed in R language. Availability of computational packages in R offers the dual benefit of carrying out the analysis locally and also building further tools and scripts. This facilitates development of both new applications and extension of existing applications. The striking feature of R is the power to accomplish complex tasks using simple scripts. R also offers a large set of statistical and mathematical tools. These can be applied on the datasets for analysis. R is open source, controlled by GNU General Public License, and allows future developments and customizations more widely. The responsibility for the maintenance of R is taken upon by a core group thereby ensuring its availability for long life.

We have been developing data packages in R for integrative analysis for several years now. Our first release was the SysBorg in R, in which we packaged the analysis data of *M. tuberculosis* using more than 50 algorithms into the R package. Datasets were collected in different arms of activity: Annotation, Drug activity, GeneExpression, Host-Pathogen relationships, strain polymorphisms and Pathways. This set constituted more than million data points. Scripts were developed to process the data and the power of the R platform allowed very rapid analysis. Integrative analysis was possible through the linking of data through scripts with other packages such as Bioconductor, and many other packages available from CRAN repository. Although each package needs data to be prepared in the required format for operation through the corresponding functions, this is easily achievable through basic scripts for conversions using base functions from R base package. After this success, we have now developed a upgraded package through the OSDD consortium. The OSDD consortium has many contributors, who contribute different facets of data. These data accrue through further analysis. We have now been able to package these new sets as upgrades of the initial SysBorg in R. As the platform is same, up-gradation and modification of scripts is straightforward. Further, we have been developing packages in R platform for immunoinformatics analysis as well. Various tools of immunoinformatics are now available. Thus it is possible to package analytical data from several from more than 20 algorithms. We have focused on *Plasmodium falciparum, Plasmodium vivax and Plasmodium yoelii* (malaria causing species), *Aspergillus fumigatus, Candida albicans, Candida glabrata, Blastomyces dermatitidis, Histoplasma capsulatum, Coccidioides immitis, Coccidioides posadasii and Paracoccidioides brasiliensis* (fungal pathogens), *Mycobacterium tuberculosis* (H37Rv and H37Ra strains) and *Chikungunya Virus*.

The integrative analysis for short listing top genes (or proteins) meeting certain criteria starts with a clearly laid out process plan followed by script writing.

Example 1:

```
Q: Which Rvids are highly expressed in clinical strains and essential?

To answer this question we set out a process plan as follows:

1. Get all Rvids with z score > 1 in all strains
2. Get all Rvids with essential including combining data
3. Get the common Rvids between 1 & 2
```

The corresponding scripts would be:

a) Use function zscoregrx() and place 1 in the parentheses and assign to a new R object as:

```
zscoregr1<- zscoregrx(1)

Notes:

The zscoregrx function can be written as follows and entered prior to its use.

zscoregrx<-   function(x)   {z<-   NULL;for   (   i   in   1:4686)   {   if
(         (MtbStrainWiseExpressionZScores[i,2]         >         x)         &&
(MtbStrainWiseExpressionZScores[i,3]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,4]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,5]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,6]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,7]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,8]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,9]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,10]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,11]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,12]            >            x)            &&
(MtbStrainWiseExpressionZScores[i,13] > x) ) z<- c(z,i)};z}

length(zscoregr1)

246

temp<-NULL;   for   (i   in   1:246)   {   tmp<-   zscoregr1[i];temp   <-
c(temp,as.character(MtbStrainWiseExpressionZScores[tmp,1]))}

   a<- as.vector(as.character(HighProbabilityOfEssentialGenes[1:22,1]))

   b<- as.vector(as.character(ExperimentallyValidatedEssentialGenes[1:7,1]))

   b1<- as.vector(as.character(GenesRequiredForOptimalGrowth[1:614,1]))

   temp1<- union(a,b)

   temp2<- union(temp1,b1)

   result1<- intersect(temp,temp2)
```

The advantage of R scripts is that they are generic in the sense that by using minor modifications we can get analyze the data in different ways by modifying the criteria.

Example 2:

```
draw heatmap of all genes with z scores greater than 1 in all strains

** Need Bioconductor **

1. Get all Rvids with zscores greater than 1 in all 12 strains with actual
values
2. Convert the data frame to a numeric matrix
3. Set color palette
4. draw heat map
```

The corresponding scripts would be:

```
draw heatmap of all genes with z scores greater than 1 in all strains

** Need Bioconductor **

rowidsgr1<-zscoregrx(1)

length(rowidsgr1)

tmp<- rowidsgr1[1]

allscoresgr1<- MtbStrainWiseExpressionZScores[tmp,]

for (i in 2:246) {tmp<- rowidsgr1[i]; allscoresgr1<- rbind(allscoresgr1, MtbStrainWiseExpressionZScores[tmp,])}

temp<- allscoresgr1[1:246,2:13]

dim(temp)

tempmat<- data.matrix(temp)

library(limma)

library(marray)

rwg<- maPalette(low="green",high="red", mid="white")

heatmap(tempmat, col=rwg)
```

References:

Srinivasan Ramachandran et al. *Mycobacterium tuberculosis* systems biology data in R. *Biobytes*, Vol.5, ISSN 0971 3271, 2009, 40-48.

Srinivasan Ramachandran, Rupanjali Chaudhuri, Rajni Verma, Ab Rauf Shah, Rituparna Sen, Chaitali Paul. Immunological Data Modeling and Scripting in R. (submitted).