The European Nanotechnology Community Informatics Platform: Bridging data and disciplinary gaps for industry and regulators

Grant Agreement No 731032

Deliverable Report 5.2

Deliverable	D5.2 First big data (omics) analysis and mining tools integrated into KnowledgeBase
Work Package	WP5: Analysis and Modelling tools
Delivery date	M12 - 31 December 2018
Lead Beneficiary	University of Birmingham (UoB)
Nature of Deliverable	Demonstrator
Dissemination Level	Public (PU)

Submitted by	Philip Doganis, Harry Sarimveis and Pantelis Karatzas (NTUA), Dieter Maier (Biomax), and Iseult Lynch (UoB)
Revised by	Anastasios Papadiamantis (UoB) and Andrea Haase (BfR)
Approved by	Iseult Lynch (UoB)

This project has received funding from European Union Horizon 2020 Programme (H2020) under grant agreement nº 731032

Table of contents

Abbreviations

AOP: Adverse Outcome Pathways

API: Application Programming Interface

BC: Biological Processes

CC: Cellular Components

DEG: Differential Expressed Genes

DN: Dependency Network

DW: Data warehouse

ENM: Engineered Nanomaterials

FP7: 7th Framework Program

GO: Gene Ontology (Database, Function)

GSEA: Gene Set Enrichment Analysis

JRA: Joint Research Activity

KB: KnowledgeBase

NN: Neural Network

MF: Molecular Functions

MI: Mutual Information

PCF: Protein Corona Fingerprint

URI: Uniform Resource Identifier

VSN: Variance Stabilisation by Normalisation

WP9: Work Package 9

Summary

The current deliverable is part of Work Package 5 (WP5) Joint Research Activity 3 (JRA3) - Analysis and Modelling Tools, which aims to integrate the current state of the art tools for data mining and data analysis, utilising a linked data approach that will exploit, extract, and integrate knowledge from all available information (raw experimental and modelling data, and metadata) contained in the NanoCommons data warehouse (DW) and KnowledgeBase (KB). These tools and the expertise to apply them and interpret the outputs in a meaningful manner, once linked and interoperable via the NanoCommons platform, will be made available to the NanoCommons User community via the open calls for Transnational Access. This Deliverable report describes the initial set of tools directed towards analysis of "omics" datasets, including transcriptomics (analysis of changes in gene expression), proteomics (analysis of changes in protein expression), metabolomics and lipidomics (changes in expression of small molecules and lipids, respectively).

The integrated tools will be available to the nanosafety community for analysing 'omics data to identify biological responses to engineered nanomaterials (ENM) exposure using gene set enrichment analysis (GSEA) and pathway analysis workflows to draw conclusions on the general and specific biological pathways responding under different ENM exposure scenarios. Deliverable D5.2 describes the integration of the first R-based analysis tools into the NanoCommons KB which provide functions for omics data quality evaluation, normalisation, differential expression analysis, functional enrichment analysis and network reconstruction. In addition, the analysis tools available in the Jaqpot application (developed by NanoCommons partner NTUA) are described. The Application Programme Interface (API)-based integration of Jaqpot and the NanoCommons KB is described in deliverable D4.2 - Initial APIs, and is thus not described here. Detailed User guidance notes, and examples of the sorts of questions that can be addressed will also be available for potential users, along with details of the dataset requirements for each of these applications in order to achieve optimal outcomes.

Introduction

The integration of tools for big data (omics) analysis and mining into the NanoCommons KnowledgeBase (KB) serves two purposes. Firstly, to enable the identification of biological mechanisms and pathways associated with toxicity / adverse effects arising from exposure to ENMs, and to produce aggregated biologically enriched descriptors. Secondly, many laboratories currently process and analyse their data in a badly protocolled and irreproducible way. Standardising this processing, by providing integrated analysis tools with full capturing of metadata such as tool version, parameter settings, data version used etc. will help greatly with reproducibility of analysis and thus will trust in the results. Coupled with best-practice procedures and workflows realised by the integrated tools this will help to generate results that can be reused, validated and integrated into regulatory procedures, which is a key objective of NanoCommons.

Omics data analysis, or Systems Biology, has recently emerged as a powerful tool for understanding biological mechanisms at the molecular level and using such information to generate predictive and mechanistic approaches to toxicity. Standard workflows for integrated Systems Biology analysis, including genomic, transcriptomic and metabolomic data, have been developed [1, 2]. These use both static analysis (biostatistics) and dynamic computational modelling to identify subsets of the multi-dimensional, information rich, ‘omics datasets that represent adverse outcome pathways (AOPs), i.e. mechanistically based molecular biomarker signatures that can be implemented into diagnostic screening assays to identify and characterise the impacts of chemicals and ENMss [3, 4]. Static methods such as differential expression analysis, functional enrichment analysis and Network Reverse Engineering approaches reconstruct the underlying structure of biological pathways from observational ‘omics data. The dynamical models (from ordinary differential equations to probabilistic or Bayesian models) enable *in silico* simulations of the toxicity responses to ENM, which can be tested experimentally.

Within Deliverable D5.2 we report the integration of big data (omics) analysis and mining tools for statistical analysis based on standard workflows established previously in the field of Systems Biology for differential gene expression, functional enrichment analysis and network reconstruction. Subsequent work in WP5 will build on these and develop additional tools related to processing and visualisation of nanomaterials omics datasets [5-7].

NanoCommons KnowledgeBase R-tools

The tools for omics analysis integrated directly into the user interface of the NanoCommons Knowledge Base target non-expert users who will rely on pre-defined analysis methods and parameters without in-depth knowledge of data analysis approaches. TA Users can also access expert support for more advanced analyses and support with interpretation of the outputs from the models, as well as with optimisation of their datasets for use with these tools.

The analyses are based on the integration of R-tools[1] which have established themselves as the standard for the corresponding analysis approach in the omics data mining expert community. The definition of workflows and default parameters are based on published standards and dedicated research projects such as the 7th Framework Program (FP7) STATegra (unpublished deliverables available via Biomax).

The basic workflow enables users to:

1. Select experimental data (transcription, protein, metabolome) in the portal;
2. Submit the selected data into a selected analysis tool such as "differential expression". Only analysis methods suitable to the data type and dataset size are available for selection by users;
3. Run the analysis in background;
4. Access the results and store them locally and/or in the NanoCommons KB;
5. Visualise them in a graph;
6. Re-use the results in further analysis, queries and overviews;
7. Make the results available to other users.

The following methods are currently available and are described below in detail:

1. Evaluation of data quality;
2. Normalisation;
3. Differential Expression Analysis;
4. Functional Enrichment Analysis;
5. Network reconstruction.

Additional tools and approaches are at various levels of development and will be integrated over the coming months and years of the NanoCommons project.

Evaluation of data quality

This function checks the selected data and gives recommendations concerning the possible analyses. It computes the number of groups and replicates in the input data to provide a recommendation for suitable analyses methods (e.g. differential expression by linear models for multiple replicates versus Audic-Claverie for single replicates).

In a second step it evaluates whether the data is already normalised by testing whether all data samples belong to the same distribution. Since omics data are usually log-transformed the stricter method to test for normal distribution cannot be applied as even transformed data will still show a slight exponential-like distribution. In addition, standard tests for normal-distribution (like Kolmogorov-Smirnov or Shapiro-Wilk) will fail with large data.

[1] A collection of tools necessary for building R packages in Windows; https://cran.r-project.org/bin/windows/Rtools/

The distribution is tested with the R-package equivalence (function tost), a test for equivalence of two groups, which performs two one-sided t-tests [8, 9]. The Null hypothesis is that the two compared microarrays do not follow the same distribution. For equivalence testing, the definition of the magnitude of region of similarity (epsilon) is critical. It defines how much deviation in the sample-underlying distributions is accepted. Often epsilon is assessed by experience i.e. the mean of means of known differences in e.g. control and pharmacological agent are calculated and based on this an allowed equivalence range defines how different the compared groups can be, while still equality is found [10]. Here, however, we need a predefined epsilon. Since a Z-score of "1" corresponds to "1 sigma" (standard deviation), an epsilon of 1 or 0.5 was selected as acceptable. In addition, to the test result the method produces a graphical output with Boxplots of all samples as per the example shown in Figure 1.

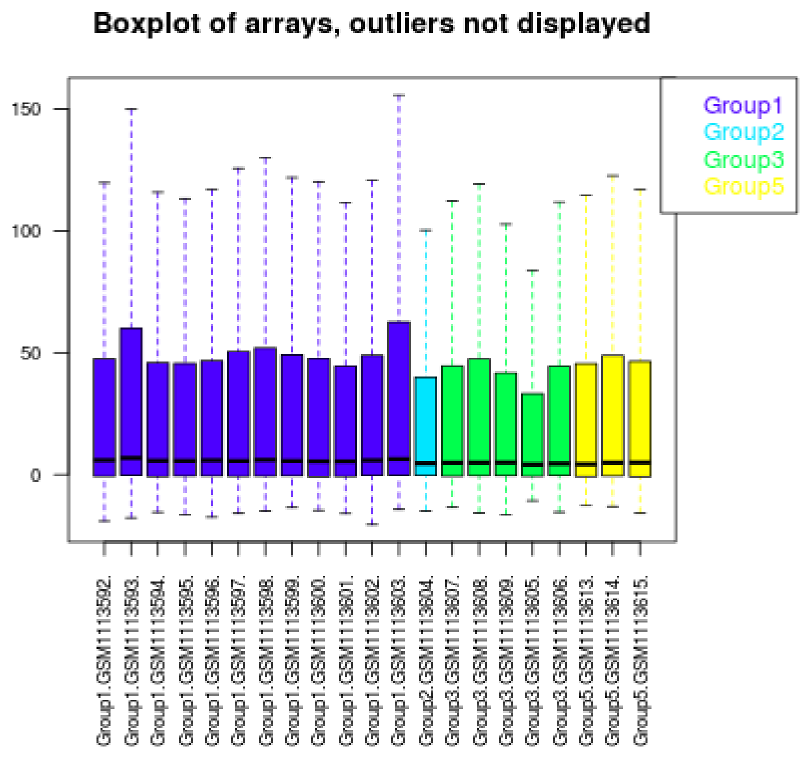


Figure 1. Testing the normality of data distribution using the R-package equivalence (function tost).

Normalisation

If the tested data are found not to follow a normal distribution, they can be normalised using two methods, scale and Variance Stabilisation by Normalisation (VSN):

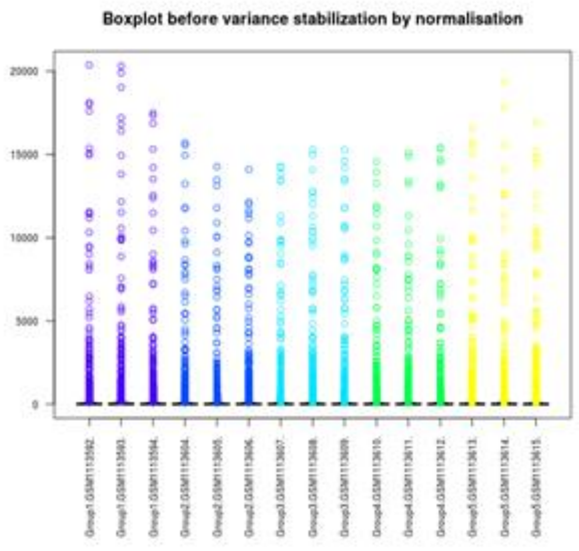
1. The Scale method scales and centers the columns of a numeric matrix. Centering is done by

subtracting the column means (omitting ‘NA’s) of ‘x’ from their corresponding columns. Scaling divides the (centered) columns of ‘x’ by their standard deviations. The function 'scale' is from the R base package. Scale can be used for any data type.

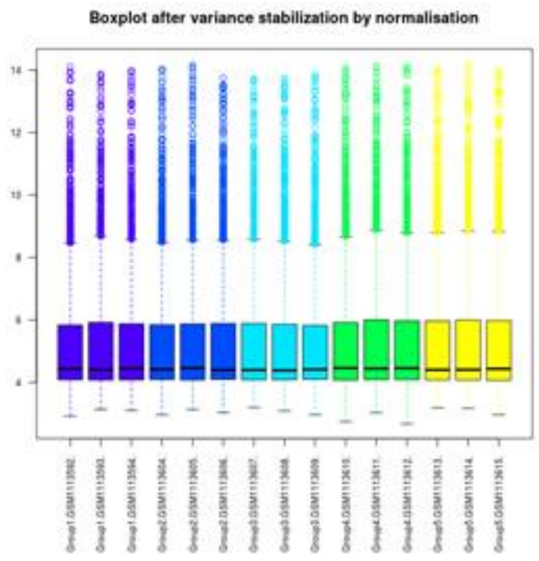
2. Variance Stabilisation by Normalisation (VSN, Figure 2) is a function derived from the Bioconductor package and can be applied for variance stabilisation and calibration of (gene, protein, metabolite) expression data. The function calibrates and log2 transforms expression data. This is a between-sample normalisation. Within-sample normalisation is normally not required for modern omics technologies and is therefore not implemented).

First an affine transformation takes place, shifting and scaling the data. Thereafter the generalised log (glog2) transformation is done (log(y + sqrt(y^2 + lambda)). This function results in a transformation equivalent to log2 for large values (large compared to the amplitude of the background noise), but is less steep for smaller values.

With increasing means (for individual probes) the standard deviation normally increases too. VSN is used to avoid this; it aims to remove the variance-on-mean relationship, so that the variance becomes constant relative to the mean.



(a)



(b)

Figure 2. Data normalisation using the VSN method. a. Data prior to normalisation, b. Normalised data.

Differential expression analysis

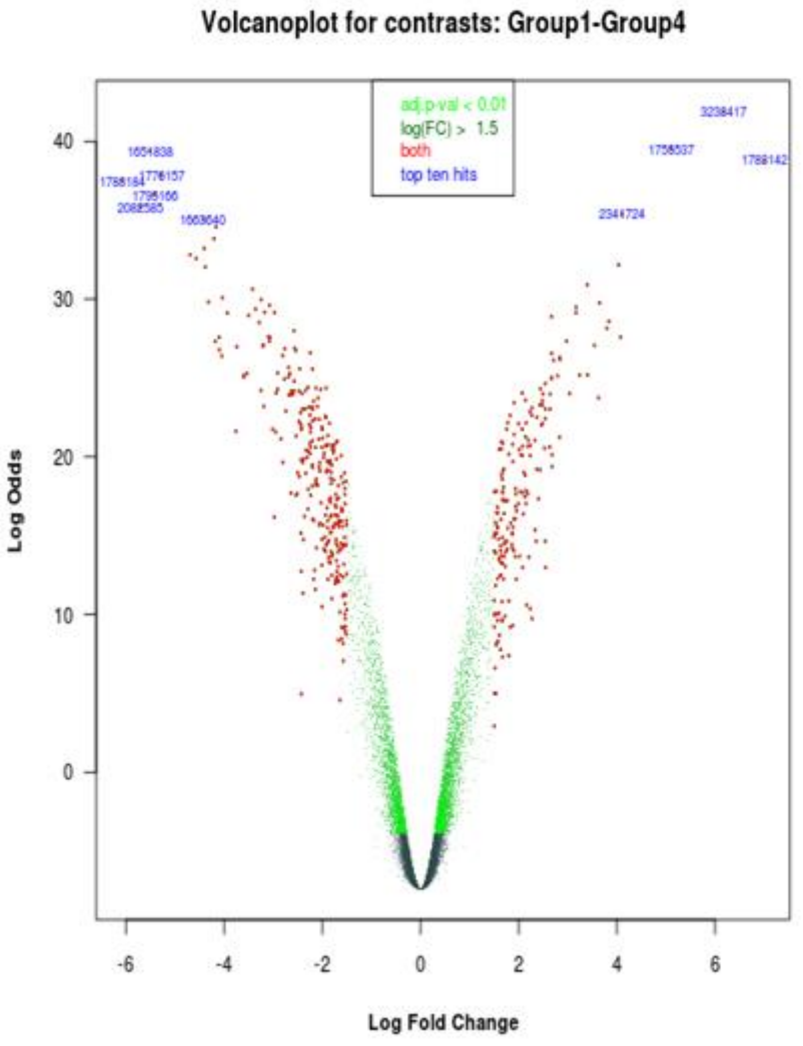
The R BioConductor packages limma [11], DEseq [12] and edgeR [13] are used to assess differential expression by linear models. A linear model is fitted to the expression data of each probe/gene of a series of samples (with identical probes/genes).

Input data for limma are log intensities of expression data for microarrays. For RNASeq data, limma uses read counts converted to log2-counts-per-million (logCPM) and the mean-variance relationship is modelled either with precision weights ("voom") or with an empirical Bayes prior trend ("limma-trend").

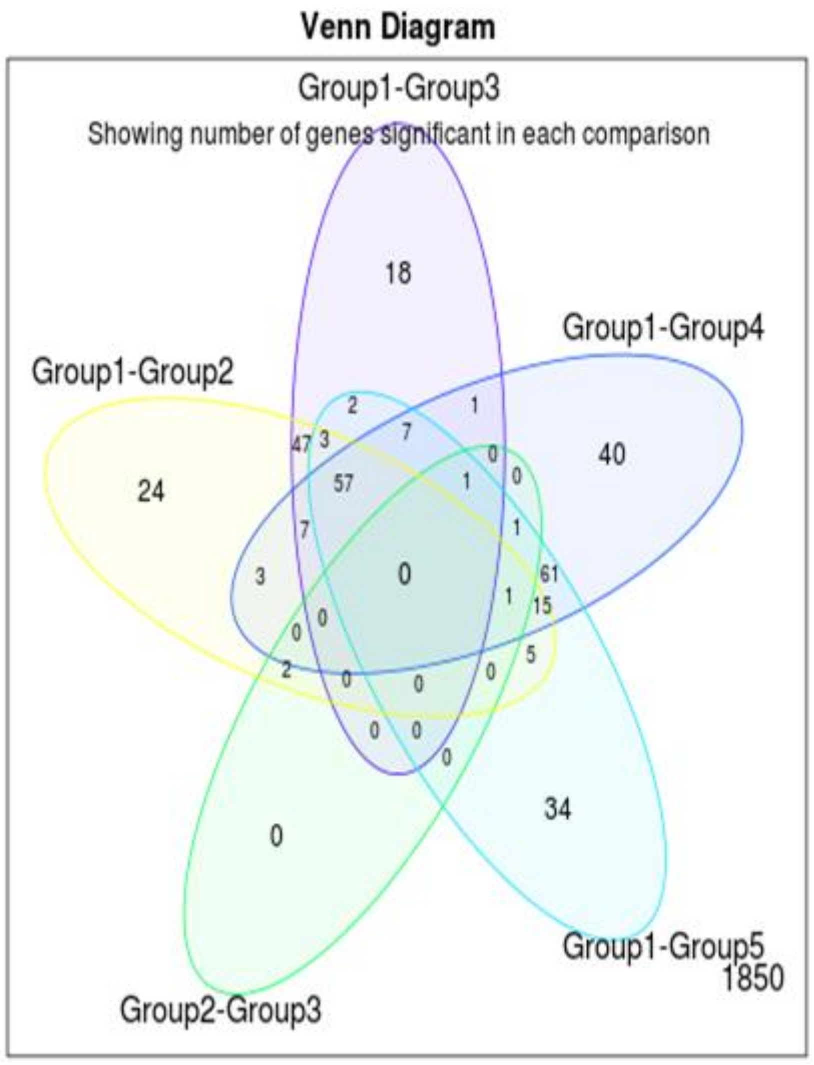
At least three samples are required as input, with at least one group comprising two samples (i.e. replicates) to compute standard deviation. Apart from this, all different numbers of samples and groups are possible. Groups are different experimental conditions (e.g. wildtype and mutant or control and treatment).

Briefly, limma firstly applies a linear model fit to the expression data based on averages of the Groups and the fold changes. From these, “contrasts” are calculated by, in essence, computing a t-statistic, where the numerator is the difference in means between the two groups and the denominator is a measure of variability. The eBayes computes a moderated t-statistics for each contrast by modifying the standard errors of estimated log fold change. Finally, Benjamini-Hochberg is used for multiple-testing correction. Thus, raw p-value and an adjusted-p-value are computed for each pair-wise comparison. The method also computes F-statistics, yielding an F-p-value, regarding differential expression of genes over all pair-wise comparisons.

An F-distribution is used for the calculation of ratios of variances of normally distributed variables. F is an element of 0 to infinite. A t-distribution is used, for example, for the calculation of means of asymptomatic normally distributed variables. t is element of 1 to n2 (i.e. the number of items in groups). Finally, results can be viewed graphically as a Volcanoplot or Venn Diagram for example as shown in Figure 3.



(a)



(b)

Figure 3. a. Volcanoplot and b. Venn-diagram visualisations of data normalisation using the VSN function.

Functional analysis: Overrepresentation and Gene-Set-Enrichment analysis

In general, functional analysis tries to establish whether a certain biological function is affected in the differential conditions analysed. To this end, the overrepresentation function uses the results of the differential expression analysis, applies a cut-off threshold for significance of differential expression and checks whether any functional parameter associated with the differentially expressed genes (DEGs), e.g. a Gene Ontology (GO) attribute or an Adverse Outcome Pathway (AOP), is significantly more often associated with a DEG than expected by chance. Fisher's exact test is used to assess overrepresentation as described in [14].

While overrepresentation analysis is a straightforward method, it requires the use of an arbitrary threshold to select the "significant" DEGs (even so a corrected p-value < 0.05 is a generally accepted threshold). To avoid this restriction the Gene-Set-Enrichment Analysis (GSEA) searches for functional gene sets (e.g. a specific GO term, an AOP), which are enriched in the genes affected by a certain condition. Similar to DGE analysis, linear models are built to reflect the impact of the condition on the gene expression. Then all genes are ranked according to their affection by the different conditions (thus in the beginning are the genes upregulated in the sample comparison, in the middle are the genes which are not affected and at the end are the downregulated genes). Thereafter, all gene sets are tested to check whether the corresponding genes are enriched at one or the other end of the distribution (the corresponding Null hypothesis is that the genes in a given set are not differentially expressed). Only two conditions can be tested at a time.

The GSEA uses the R BioConductor package limma and the rank function is romer (gene set enrichment analysis for linear models using rotation tests) [15]. Romer tests a hypothesis similar to that of GSEA [16] but is designed for use with linear models. Romer does a competitive test in that the different gene sets are pitted against one another. Instead of permutation, it uses rotation, a parametric resampling method suitable for linear models [17]. Romer can be used with any linear model with some level of replication. In addition to the p-value calculated for a given set to be either up, down or differentially regulated, a graphical overview is provided, as shown in Figure 4.

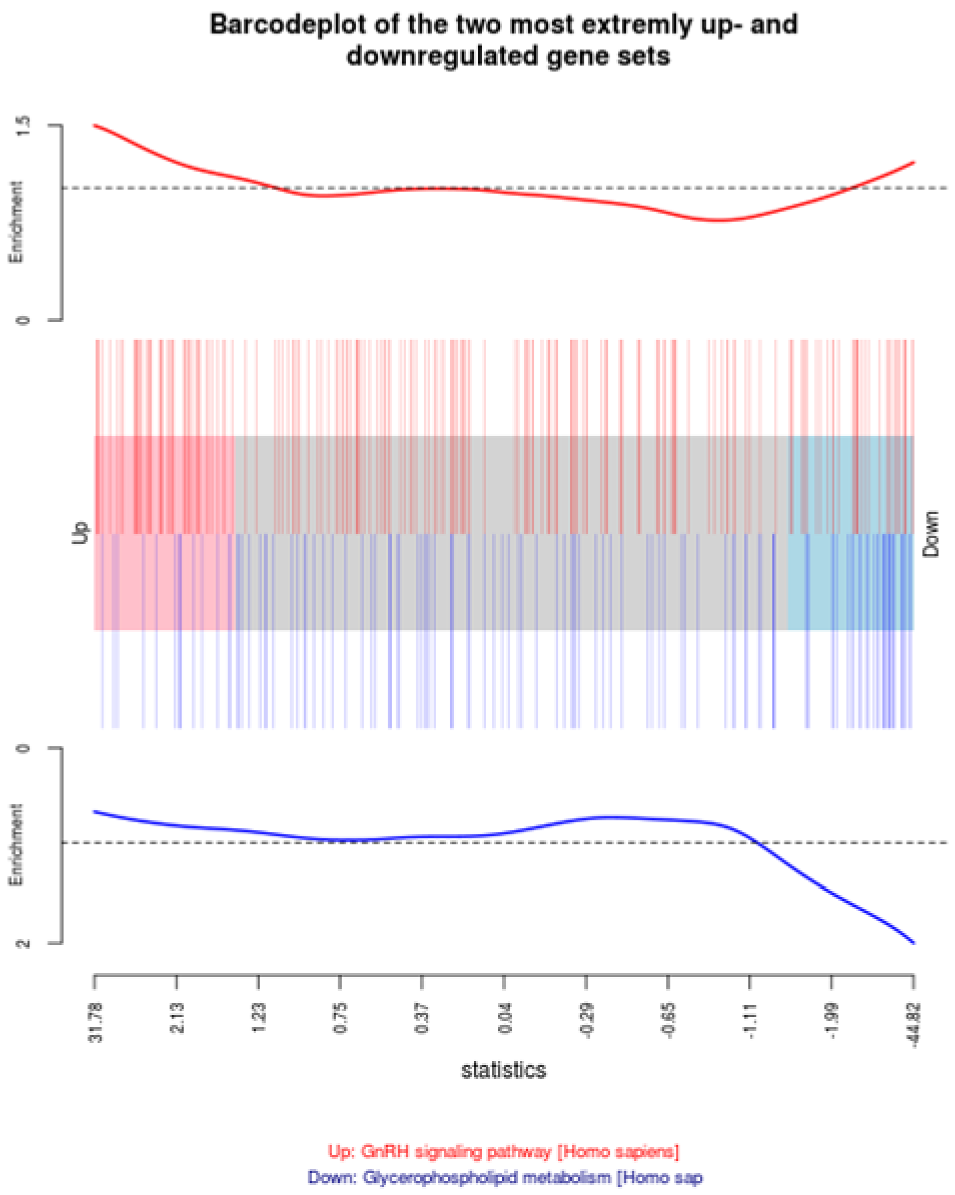


Figure 4. Gene-Set-Enrichment Analysis (GSEA) demonstrating the impact of a condition on the gene expression. The genes are ranked according to their affection by the different conditions. Firstly, the the upregulated genes are presented, followed by the non-affected genes and finally the downregulated genes.

Network reconstruction using Mutual Information

Construction of Dependency Networks (DNs) is achieved by using Mutual Information (MI), which is a measure of how well one variable (e.g. a gene) can be predicted by another (e.g. a second gene, a phenotype). In addition to standard correlation analysis, the concept of MI allows the identification of irreducible statistical dependencies that cannot be explained as an artifact of other statistical dependencies in the network. For example, if B correlates with C and A correlates with B and C, then MI will indicate that both B and C might be regulated by A rather than B regulating C.

Mathematically MI is a measure of how similar the joint distribution p(x,y) is to the product of a factorised marginal distribution p(x)*p(y).

The R-package parmigene [18] is used to infer the network based on the Kraskov Algorithm [19] for estimation of MI and the ARACNE Algorithm [20] for network reconstruction. Different data types with identical samples (experiments) can be combined after standardising each variable across all samples to have a mean of 0 and a standard deviation of 1 (implemented using the R-function scale). Networks can be queried and visualised directly in the NanoCommons Knowledge Base (Figure 5) or they can be exported for further analysis for example in Cytoscape [21].

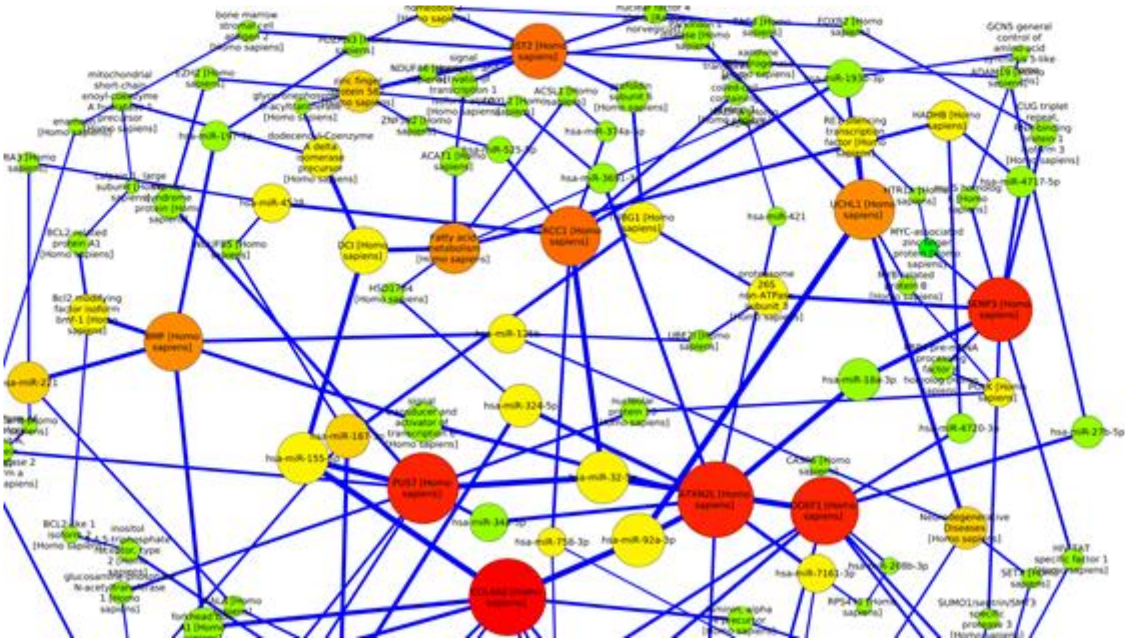


Figure 5. Network reconstruction using MI, available via the NanoCommons KB.

Bio-descriptor calculations using the Jaqpot computational platform

The interest towards omics data is growing in the field of toxicology owing to the diverse knowledge they generate, which can improve prediction and dosage profiling for more accurate safety assessment both to humans and the environment. In [22] the authors presented a methodology to compute BIO-descriptors, i.e. descriptors which are derived by enriching high-throughput omics data with biological-pathway information based on the decomposition of the original experimental omics data into meaningful clusters in terms of both their mechanistic interpretation and correlation affinity. The methodology employs a generalised simulated annealing algorithm to estimate the optimal partition of the enriched data, using clustering theory, and accordingly produces descriptors based on gene content similarity. An important result of this methodology is the derivation of a reduced set of BIO-descriptors which can be easier handled by predictive modelling algorithms, compared to using all individual genes. In [22], the authors demonstrate the increasing predictability power for toxicity endpoints by employing the BIO-descriptors as independent variables in a singlehidden-layer feed-forward neural network (NN) modelling scheme. There is a good agreement between the important set of genes reported by [22] and both the original publication [23] and its follow-up study [24].

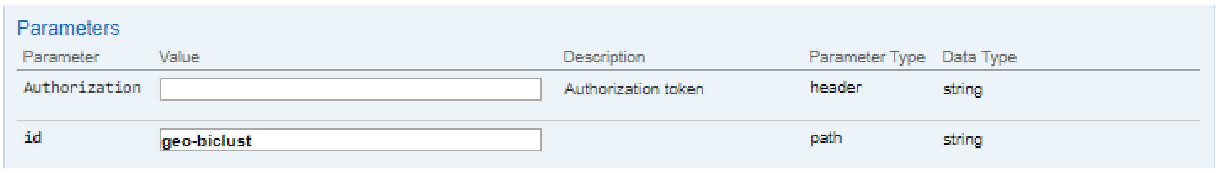
In the NanoCommons project, the methodology has been integrated via the Jaqpot module and is provided as a web service. For the particular service employed, the biological information used is derived from the GO database (http://www.geneontology.org), a well-known source of biology ontological information that defines concepts/classes used to describe gene function, and relationships between these concepts. GO is divided into three aspects, namely Molecular Functions (MF) which are molecular activities of gene products, Cellular Components (CC) where gene products are active, and Biological processes (BC) which are pathways and larger processes made up of the activities of multiple gene products. Apart from the biological information, other parameters need to be defined by the user in order to customise the service to fit their specific scientific problem or question. In particular, there are two functions available depending on which clustering algorithm is used, with the two options being hierarchical clustering and biclustering. Hierarchical clustering is the default approach only clustering groups of genes, however if the user believes that there are similarities between samples as well, then the biclustering method will be a better match. For hierarchical clustering the extra parameters needed are the distance metric (a function defining the distance between two data points), the agglomeration method (a method defining how data will be clustered), and the expected number of clusters in the data. For the biclustering option the extra parameters needed are the numbers of clusters in both axes (x,y). Overall, the input and parameters needed from Users for both approaches are given in the following list:

1. Describe the data and specifically the annotation used for proteins. Here a character string needs to be specified for the database used to annotate the proteins in the data. The default value is ‘UNIPROT’, but other identifiers can be used such as RefSeq, Ensemble, Entrez.
2. Biological information currently implemented via the GO database. The user needs to specify with a character vector of two values: a. the pathway database, and b. its particular component, from which all the biological information will be derived (character vector). Default values are c(‘GO’, ‘MF’).
3. The threshold for statistically significant p-values from GSEA need to be specified. Based on this threshold only the significant GO ids will be retained. In this case, the default value is 0.05.
4. Clustering parameters. For the hierarchical clustering the distance method - (the default value for ‘distMethod’ parameter is ’euclidean’), the agglomeration method used (the default value for the ‘hclustMethod’ parameter is the ’ward.D2’), and the number of clusters in the data (the default value for ‘nORh’ is 10) need to be provided. For biclustering ‘nclust’ corresponds to the number of clusters in the x and y axis respectively. The default value is c(4,2).
5. ‘FUN’ is the summary function applied to the clustered values. The default value for ‘FUN’ is ’Mean’.

The exhaustive list of options for 'distMethod' parameter dissimilarity index is 'manhattan', 'euclidean', 'canberra', 'clark', 'bray', 'kulczynski', 'jaccard', 'gower', 'altGower', 'morisita&#39;, 'horn', 'mountford', 'raup', 'binomial', 'chao', 'cao' or 'mahalanobis'. For the agglomeration method parameter, the full list of options is the following: 'ward.D', 'ward.D2', 'single', 'complete', 'average', 'mcquitty', 'median' or 'centroid'. Finally, for the 'FUN' parameter, as mentioned above a function is needed to specify the method of summarizing data in the estimated clusters; except for the default 'mean', other options can be the 'mode', 'var' or any other tailor-made function metric provided by the user. Overall, the default values would produce statistically robust results, however since the above is a wide collection of options for different metrics and clustering methods, the users would benefit from experienced partners guidance, e,g, via the TA helpdesk, mostly in cases where certain aspects of the data need to be taken into consideration.

The BIO-descriptor calculation services have been registered in the Jaqpot platform and can be accessed through the Jaqpot Descriptor API. More specifically, they are registered under the ids *'geo-biclust'* and *'geo-hclust'* for the biclustering and hierarchical clustering service respectively.

Under the operation https://api.jaqpot.org/jaqpot/swagger/#!/descriptor/getDescriptor one can request the service description with the proper *id* (Figure 6):



Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization		Authorization token	header	string
id	geo-biclust		path	string

Figure 6. Parameter to GET the biclustering BIO-descriptor calculation service

The JSON received by providing the *'geo-biclust' ID is:*

```
{
 "meta": {
  "descriptions": [
   "geo",
   "Geo Descriptors service with biclust"
  ],
  "titles": [
   "geo",
   "Geo Descriptors biclust"
  ],
  "subjects": [
   "Geo Descriptors creation"
  ],
  "creators": [
   "6d9b8261-e255-4fca-9289-0fde48644cf8"
  ],
  "date": "2018-12-20T14:04:19.967+0000",
  "locked": false
 },
 "ontologicalClasses": [
  "ot:Geo"
 ],
 "parameters": [
  {
   "name": "key",
    "value":                                                                    [
     "UNIPROT"
    ],
    "scope":                                                          "MANDATORY",
    "allowedValues":                                                            [
     [
      "UNIPROT"
     ]
    ],
    "_id":                                                                  "key"
   },
   {
    "name":                                                                "FUN",
    "value":                                                                    [
     "mean"
    ],
    "scope":                                                          "MANDATORY",
    "_id":                                                                  "FUN"
   },
   {
    "name":                                                         "pvalCutoff",
    "value":                                                                    [
     0.05
    ],
    "scope":                                                          "MANDATORY",
    "_id":                                                           "pvalCutoff"
   },
   {
    "name":                                                             "nclust",
    "value":                                                                    [
     4,
     2
    ],
    "scope":                                                          "MANDATORY",
    "_id":                                                               "nclust"
   },
   {
    "name":                                                               "onto",
    "value":                                                                    [
     "GO",
     "MF"
    ],
    "scope":                                                          "MANDATORY",
    "_id":                                                                 "onto"
   }
  ],
  "ranking":                                                                   0,
  "descriptorService": "http://jaqpot-geo-descriptors-service.jaqpot:8004/ocpu/library/geoDescriptors/R/generate.biclust.model/json",
  "_id":                                                             "geo-biclust"
}
```

In order to use the descriptor services one can POST on the Descriptor resource, through https://api.jaqpot.org/jaqpot/swagger/#!/descriptor/applydescriptor, a request on a specific descriptor by providing:

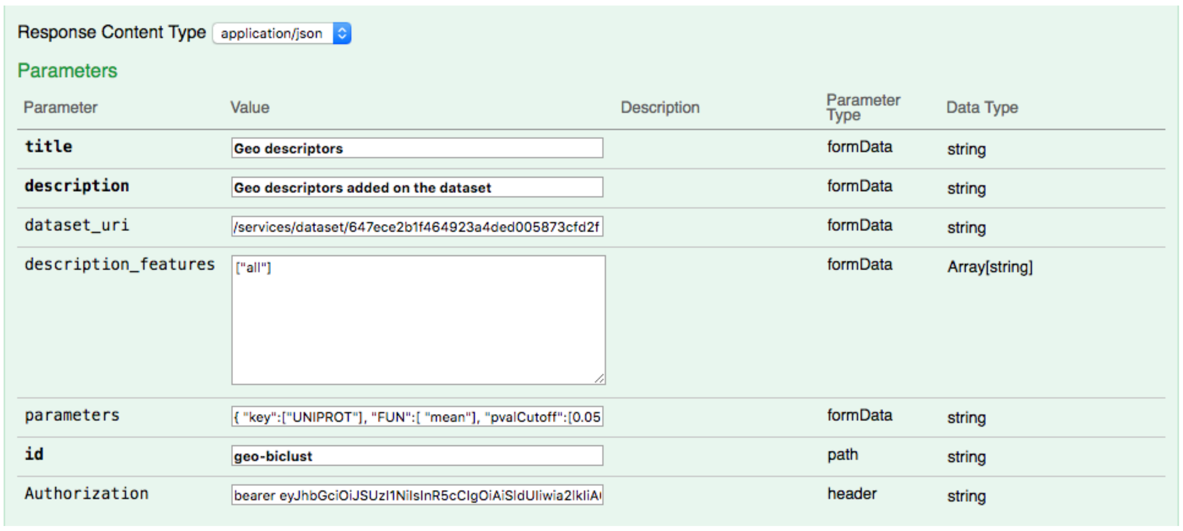
1. “title”: a title for the produced final descriptors Dataset,
2. “description”: a description of the dataset that will be returned as the output of the service,
3. “dataset_uri”: the specific dataset Uniform Resource Identifier (URI) from Jaqpot that will be used to produce the new dataset of descriptors,

4. “description_features”: an array of strings specifying the features of the dataset that the service takes as an input (in our case [“all”]),
5. “parameters”: the parameters of the service supplied as an array of values and/or strings.
6. “Id”: “geo-biclust” and “geo-hclust”.

We present next an example of applying the BIO-descriptor calculation service to an extensive protein corona data dataset [23], which contains 129 Protein Corona Fingerprint (PCF) data from 84 gold nanoparticles (NPs). In [22], it was illustrated that BIO- descriptors are beneficial for modelling and prediction of biological effects, outperforming the prediction accuracy of the original omics data, and offering a readily available biological interpretation of the findings. The method aggregates relative abundances of spectral counts from PCF data with GO molecular-function information specific to each NP protein corona to produce a reduced set of informative BIO-descriptors. The csv file containing the example dataset used as input can be found at the following URI: https://github.com/KinkyDesign/descriptor-csv-examples/blob/master/pcorona.csv. Before using the service, we need to upload the csv file to the Jaqpot platform, using the procedure described in Annex 1. One can see the dataset by GETting on the Dataset API, https://api.jaqpot.org/jaqpot/swagger/#!/dataset/getDataset, with the id of the dataset, in our case, 647ece2b1f464923a4ded005873cfd2f. The GET operation can return either application/json or text/csv depending on the ‘Response Content Type’ field selected. In addition, the user can select if he/she wants the data entries or just the meta information of the dataset by checking true or false on the *dataEntries* field.

The application of the POST request on the Jaqpot Descriptor service is shown in Figure 7 with the following parameters:

1. “title”: “Geo descriptors”
2. “description”: “Geo descriptors added to the dataset”
3. “dataset_uri”: https://api.jaqpot.org/jaqpot/services/dataset/647ece2b1f464923a4ded005873cfd2f
4. “description_features”: [“all”]
5. parameters: {"key":["UNIPROT"], "FUN":["MEAN"], "pvalCutoff":[0.05], "nclust":[4, 2], "onto":[GO,MF]}
6. “id”: “geo-biclust”



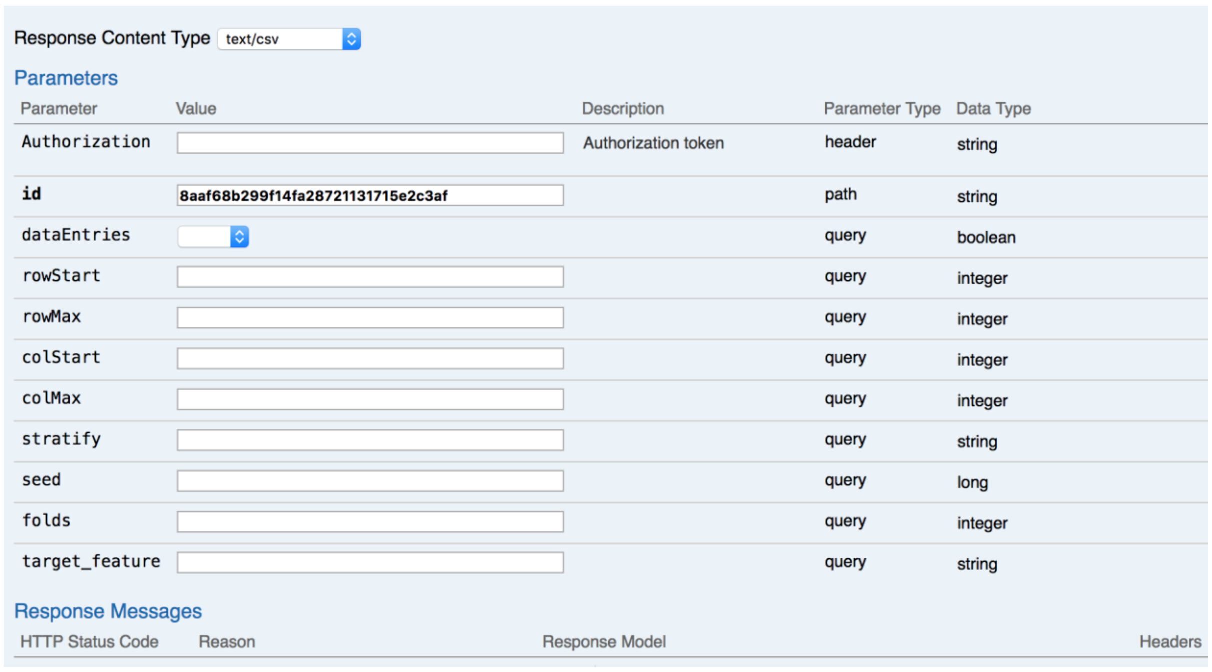
Response Content Type: application/json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
title	Geo descriptors		formData	string
description	Geo descriptors added on the dataset		formData	string
dataset_uri	/services/dataset/647ece2b1f464923a4ded005873cfd2f		formData	string
description_features	["all"]		formData	Array[string]
parameters	{ "key":["UNIPROT"], "FUN":["mean"], "pvalCutoff":[0.05		formData	string
id	geo-biclust		path	string
Authorization	bearer eyJhbGciOiJSUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA		header	string

Figure 7. Parameters to POST using the BIO-descriptor calculation service.

The POST operation initiates a descriptor calculation procedure. The user can access information about this task through the Task API, using the id of the task that is returned from the POST operation. When the asynchronous task is finished successfully, the task reports the id of the resulting dataset. In our case, this dataset got the id 8aaf68b299f14fa28721131715e2c3af and we can GET it from the Dataset API, (Figure 8). If a user wishes to view the actual data, the “dataEntries” field should be changed to “True”. By adding the rows of the actual dataset that he/she wishes to see “rowStart, rowMax” the corresponding rows will be returned. The dataset can be viewed either as a csv or as a Json. This corresponds to the Response Content Type that the user has selected. It can be either “text/csv” or “application/json”.



Response Content Type: text/csv

Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization		Authorization token	header	string
id	8aaf68b299f14fa28721131715e2c3af		path	string
dataEntries			query	boolean
rowStart			query	integer
rowMax			query	integer
colStart			query	integer
colMax			query	integer
stratify			query	string
seed			query	long
folds			query	integer
target_feature			query	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers

Figure 8. Parameters to GET the resulting Dataset following application of the BIO-descriptor calculation for PCF.

The first few entries of the produced dataset in csv format is presented in Figure 9. The two BIO-descriptors produced are illustrated in larger bold font.

Figure 9: Example of the dataset produced by GETting the dataset as shown in Figure 8.

```
"EntryId","Q92954","P00738","P01042","P02655","P05452","P01024","P02760","P00450","P22891",
"P35443","P02775","clust.go.sep2","P35858","P00740","P04220","P02749","P0CG05","O43866",
"O94985","P02735","P01766","Q9UGM5","Q15485","P23528","P55056","P02748","P00736","P0C0L5","
P01859","P02763","P10720","P02787","P48740","P07996","P14618","P00742","O95445","P27797","P
09871","P02746","P02747","P00734","P04003","P01019","P02652","P35542","P08567","P01008","Q5
XUX0","P27918","Q6Q788","Q9Y490","P04217","P20851","P06727","Q99467","P03950","P12259","P03
952","P0C0L4","P02765","P01011","P01857","P07225","P00751","P06396","P02741","Q9UK55","P016
20","P04004","P27169","P02788","P02766","Q03591","P04070","P04196","P69905","P00451","P0274
3","P05155","P01031","P10909","P35579","P01860","P01861","P04114","P02745","P15169","P51884
","P02654","P02656","P01871","P01023","P02751","P05546","P02649","B9A064","Q14520","P08697"
,"P00747","Q96KN2","P19823","Q13790","Q14624","P14625","P68871","P19827","P08603","P00739",
"P05154","P18065","P02790","P01834","P08709","P02774","P18428","P00748","P04180","P03951","
P49908","Q13103","Q06033","O14791","P02671","P01876","P01009","Q9BXR6","P49747","P02647","c
lust.go.sep1","Q08380"
"1","0.023314749","0","0.020273695","0","0","0.073492144","0","0","0","0","0","0.0072","0",
"0","5.06842E-4","0","0.007095793","0","0","0","5.06842E-
4","0","0","0","0","0.005575266","0.011657375","0.001520527","0.003547897","0","0.001520527
","0","0","0.03852002","0.003041054","0","0","0","0.010136847","0.008109478","0.008109478",
"0.061834769","0.031931069","0","0.003041054","0","0","0.112012164","0","0.002027369","0","
0.007602636","0","0.001520527","0","0","0.002534212","0.004054739","0","0.027369488","0","5
.06842E-
4","0.014698429","0.003041054","0","0.006588951","0","0","0.001520527","0.033451597","0","0
.01064369","0.002027369","0.002027369","0","0.044095286","0","0","0.031424227","0.004054739
","0","0.012671059","0.005068424","0.005068424","5.06842E-
4","0.079574252","0.003041054","5.06842E-
4","0","0","0","0.016725798","0.011657375","0.022301064","0.007602636","0.085656361","0.003
547897","0","0.001013685","5.06842E-
4","0","0.009123163","0","0.002534212","0","0","0.003547897","0.021794222","0.002027369","0
.01064369","0","0","0.011657375","0","0","0","0.004561581","0","0.013684744","0.003041054",
"0.002534212","0","0.001520527","0","0.008109478","0.006588951","0.013177902","5.06842E-
4","0.004561581","0.0109","0"
"2","0","0.001899696","0","0","0","0.021656535","0.00493921","0","0.013677812","0","0","0.0
068","0","0.050911854","0.005699088","0","0.005319149","0.003039514","0","0","7.59878E-
4","0","0","0","0","0","3.79939E-4","0.007598784","0.001139818","0","3.79939E-
4","0.004559271","0","0.001519757","0","0.082446809","0","0","0","0","0","0.263677812","0.1
27279635","7.59878E-
4","0.003039514","0","0.002659574","0.047112462","0","0","0","0","0","0.007978723","0.00113
9818","0","0","0.002659574","0","0.055091185","0","0","0.015577508","0.068389058","0","3.79
939E-
4","0","0.003419453","0.001139818","0.021656535","0","0","0.004179331","0","0.024316109","0
.001899696","0","0","0.002659574","3.79939E-
4","0","0.005319149","0","0.001139818","3.79939E-
4","0.021276596","0","0","0","0","3.79939E-
4","0.018237082","0.004559271","0","0","0.001899696","0.002659574","0","0.003419453","0","0
","0.005319149","0","0.00493921","0.001519757","0","0","0","0.001899696","0","0","0","0.015
957447","0.009878419","7.59878E-
4","0","0","0","0","0","0.003039514","0","0","0","0.011398176","0.016717325","0","0","0.007
978723","0.0134","0"
"3","0.00882825","0","0.016853933","0","0","0.08105939","0","0","0","0","0","0.0082","0","0
","8.02568E-
4","0","0","0","0","0","0","0","0","0","0","0.003210273","0","0.001605136","0","0","0.00642
0546","0","0","0.016853933","0.001605136","0","0","0","0","0","0","0.123595506","0","0","0.
010433387","0","0","0.313001605","0","0","0","0","0","0","0","0","0.008025682","0","0","0.0
17656501","0","0","0.003210273","0","0","0.002407705","0","0","0","0.085072231","0","0.0104
33387","0.004815409","0.005617978","0","0.031300161","0","0","0.016051364","0","0","0.00882
825","0","0.001605136","0","0.016853933","0","0","0","0","0","0.008025682","0.00882825","0"
,"0.016051364","0.107544141","0.001605136","0","0.007223114","0","0","0.003210273","0","0.0
02407705","0","0","0","0","0","0.010433387","0","0","0.005617978","0","0","8.02568E-
4","0.004012841","0","0","0.003210273","0.001605136","0","0","0","0.005617978","0.006420546
```

```
","0.005617978","0","0.005617978","0.0053","0"
"4","0.008709422","0","0.017418844","0","0","0.084718923","0","0","0","0","0","0.0077","0",
"0","7.91766E-4","0","0.004750594","0","0","0","7.91766E-4","0","0","0","0","7.91766E-
4","0","0.002375297","7.91766E-4","0","0.003958828","0","7.91766E-
4","0.009501188","0","0","0","0","0","0","0","0.106096595","0.001583531","0","0.006334125",
"0","0","0.288202692","0","0","0","0","0","0","0","0","0.007125891","0","0","0.015835313","
0","0","0.004750594","0","0","0.004750594","0","0","0","0.071258907","0","0.01266825","0.00
3958828","0.010292953","0","0.040380048","0","0","0.053840063","0","0","0.005542359","0","7
.91766E-
4","0","0.026128266","0","0.001583531","0","0","0","0.015043547","0.004750594","0","0.00950
1188","0.102137767","0.005542359","0","0.005542359","0","0","0","0","0.003958828","0","0","
0","0.007125891","0","0.011876485","0","0","0.011876485","0","0","7.91766E-
4","0","0","0","0.001583531","0.002375297","0","0","0","0.006334125","0.003167063","0.00475
0594","0","0.007125891","0.0078","0"
"5","0","0","0","0.003805497","0","0.024524313","8.45666E-4","4.22833E-
4","0.019450317","0","0","0.007","0","0.026638478","0.00422833","0","0.002536998","0.002114
165","0","0","4.22833E-4","0","0","0","0","0","0","0.005496829","8.45666E-
4","0","0","0.002114165","0","0.007188161","0","0.046511628","0","0","0","0","0","0.2266384
78","0.101057082","4.22833E-
4","0.003382664","0","0.003382664","0.043128964","0","0","0","0","0","0.005919662","0.00549
6829","0.003382664","0","0.005919662","0","0.054968288","0","0","0.006342495","0.051162791"
,"0","0","0","4.22833E-
4","0","0.095560254","0","0","0.024524313","0","0.022410148","0.001691332","0","4.22833E-
4","0.002114165","0","0","0.008033827","0","0.002114165","0","0.023678647","0","0","0","0",
"4.22833E-
4","0.016490486","0.005496829","0","0.002114165","0.012262156","0.002536998","0.020295983",
"0.007610994","0","0","0.010993658","0","0.002536998","0","8.45666E-
4","0.005919662","0","0.002114165","0.001268499","0","0","0.014799154","0.007610994","0.001
268499","0","0","0","0","0","0.003805497","0","0","0","0.009302326","0.01987315","0","4.228
33E-4","0.012684989","0.0119","0"
```

The same dataset can also be viewed in Json form, where additional information is depicted. This way Ontological classes or other information can easily be added to the desired property. The first few lines of the JSON representation of the produced dataset are shown in Figure 10.

Figure 10: Example of the JSON representation of the dataset shown in Figure 9.

```
{
  "meta":                                                                          {
    "comments":                                                                    [
      "Created          by           task           TSK92c62cb74bc64c3a803fa4f7d93024f0"
    ],
    "descriptions":                                                                [
      "Geo        descriptors          added          on          the          dataset"
    ],
    "titles":                                                                      [
      "Geo                                                                 descriptors"
    ],
    "creators":                                                                    [
      "6d9b8261-e255-4fca-9289-0fde48644cf8"
    ],
    "hasSources":                                                                  [
      "https://api.jaqpot.org/jaqpot/services/dataset/647ece2b1f464923a4ded005873cfd2f"
    ],
    "date":                                              "2018-12-20T14:34:25.347+0000",
    "locked":                                                                     false
  },
  "visible":                                                                      true,
  "dataEntry":                                                                     [
    {
      "datasetId":                                   "8aaf68b299f14fa28721131715e2c3af",
      "entryId":                                                                   {
        "name":                                                                    "1",
        "ownerUUID":                                  "6d9b8261-e255-4fca-9289-0fde48644cf8"
      },
      "values":                                                                        {
        "https://api.jaqpot.org/jaqpot/services/feature/016d56472bbe4038840f8b1d9edcccdc":
0.023314749,
        "https://api.jaqpot.org/jaqpot/services/feature/07c1e1948a0c4b869c9af5fdf214a710":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/0aeee667ceae4348a0c8b6d1cf10c59d":
0.020273695,
        "https://api.jaqpot.org/jaqpot/services/feature/0d0ef1ba0b704de8a946c81f52e9a9fa":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/0da6b4b39b464132955bdd23cf156f28":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/13225819ee0a4576babad7a2e904f00d":
0.073492144,
        "https://api.jaqpot.org/jaqpot/services/feature/144703beb4d041fa82ce9972fcf2abc2":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/17b1b639ec374e699dc15eac1a18174d":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/18934450f349415a9ade0aceb5fc5cbc":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/1cb734ed562c47efa17287a5c9bc88f2":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/1cdad4aa81584ae190cffbdfafb90bd0":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/1fc9ff97be394342bb3a082286ccc853":
0.0072,
        "https://api.jaqpot.org/jaqpot/services/feature/23e2d63ebfff4519a33fc4843fcb9061":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/244972868e6544759c43d572982c11d2":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/287082f03aac45548355d848b231a846":
0.000506842,
        "https://api.jaqpot.org/jaqpot/services/feature/28a64eea5186491cbc7c113e20dfb4c7":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2a3722071d0a438cb07e420045b85211":
0.007095793,
        "https://api.jaqpot.org/jaqpot/services/feature/2a63798e00d04f458c89f02a4d264cc9":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2a70dfbc806e47038643435445539ce7":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2b4aefed10124b5d86e4d6586c15e19b":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2c4fef040de3449f9dd5124487ee510c":
0.000506842,
        "https://api.jaqpot.org/jaqpot/services/feature/2e11c27ffe4047b2b3b59bab2fe125c9":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2f3a65cf517f41a98c3fecced3ececf0":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/2fea1173689f4f06b5e5be27c40cd933":
0,
        "https://api.jaqpot.org/jaqpot/services/feature/308f2c785a9e46e882bc2e3f67e5b813":
0,
```

Conclusions

The tools presented in this deliverable are intended to meet the increasing needs of the nanosafety scientific community for omics data analysis and visulaisation tools. NanoCommons aim is to, firstly, integrate the state of the art of existing tools and develop them further, or if not possible to develop new to fill any existing gaps. Omics tools are highly significant as they can provide information on nanosafety-relevant pathways, linked to AOPs and incorporated and visualised with the

WikiPathways database. Further breakthrough is anticipated by grouping omics data into biologically enriched clusters based on the application of hierarchical clustering or bi-clustering machine learning algorithms on public molecular pathway databases such as GO and the use these datasets for predictive modelling purposes.

A key feature of their integration in NanoCommons is that users are guided through the steps, presented with only the applicable / relevant tools, and given feedback on the suitability of their datasets for use with the various tools (as well as guidance for future experimental design to optimise the datasets available). All user generated data is captured and added to the original datasets with the relevant metadata, thereby enriching the overall pool of available data, and allowing mining and comparisons of gene or protein enrichment data, and other BIO-descriptors across different ENMs, different cell lines etc. Finally, the NanoCommons TA provision, and the TA helpdesk (see Deliverable D7.1 for further details) are available to users to support in the implementation of the tools, and/or to provide guidance on data interpretation and options for further analysis.

The tools presented here are just the first step in the implementation of the NanoCommons suite of omics analysis and visualization tools, which will be updated regularly throughout the project lifetime, and beyond.

References

1. Ghosh, S., et al., *Software for systems biology: from tools to integrated platforms.* Nature Reviews Genetics, 2011. **12**: p. 821.
2. Wedge, D.C., et al., *Automated workflows for accurate mass-based putative metabolite identification in LC/MS-derived metabolomic datasets.* Bioinformatics, 2011. **27**(8): p. 1108-1112.
3. Tollefsen, K.E., et al., *Applying Adverse Outcome Pathways (AOPs) to support Integrated Approaches to Testing and Assessment (IATA).* Regulatory Toxicology and Pharmacology, 2014. **70**(3): p. 629-640.
4. Walser, T. and C. Studer, *Sameness: The regulatory crux with nanomaterial identity and grouping schemes for hazard assessment.* Regulatory Toxicology and Pharmacology, 2015. **72**(3): p. 569-571.
5. Dekkers, S., et al., *Multi-omics approaches confirm metal ions mediate the main toxicological pathways of metal-bearing nanoparticles in lung epithelial A549 cells.* Environmental Science: Nano, 2018. **5**(6): p. 1506-1517.
6. Genta-Jouve, G., et al., *MUSCLE: automated multi-objective evolutionary optimization of targeted LC-MS/MS analysis.* Bioinformatics, 2014. **31**(6): p. 975-977.
7. Guo, R., et al., *RepLong: de novo repeat identification using long read sequencing data.* Bioinformatics, 2017. **34**(7): p. 1099-1107.
8. Schuirmann, D. *On hypothesis-testing to determine if the mean of a normal-distribution is contained in a known interval*. in *Biometrics*. 1981. INTERNATIONAL BIOMETRIC SOC 808 17TH ST NW SUITE 200, WASHINGTON, DC 20006-3910.
9. Thomas, B.L.K. and W.J. Westlake, *Bioequivalence Testing -- A Need to Rethink.* Biometrics, 1981. **37**(3): p. 589-594.
10. Castañeda-Hernández, G., et al., *Biosimilars in rheumatology: what the clinician should know.* RMD Open, 2015. **1**(1): p. e000010.
11. Ritchie, M.E., et al., *limma powers differential expression analyses for RNA-sequencing and microarray studies.* Nucleic Acids Research, 2015. **43**(7): p. e47-e47.
12. Anders, S. and W. Huber, *Differential expression analysis for sequence count data.* Genome Biology, 2010. **11**(10): p. R106.
13. Robinson, M.D., D.J. McCarthy, and G.K. Smyth, *edgeR: a Bioconductor package for differential expression analysis of digital gene expression data.* Bioinformatics, 2010. **26**(1): p. 139-140.
14. Khatri, P. and S. Drăghici, *Ontological analysis of gene expression data: current tools, limitations, and open problems.* Bioinformatics, 2005. **21**(18): p. 3587-3595.
15. Majewski, I.J., et al., *Opposing roles of polycomb repressive complexes in hematopoietic stem and progenitor cells.* Blood, 2010. **116**(5): p. 731-739.
16. Subramanian, A., et al., *Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles.* Proceedings of the National Academy of Sciences, 2005. **102**(43): p. 15545-15550.
17. Wu, D., et al., *ROAST: rotation gene set tests for complex microarray experiments.* Bioinformatics, 2010. **26**(17): p. 2176-2182.
18. Sales, G. and C. Romualdi, *parmigene—a parallel R package for mutual information estimation and gene network reconstruction.* Bioinformatics, 2011. **27**(13): p. 1876-1877.
19. Kraskov, A., H. Stögbauer, and P. Grassberger, *Estimating mutual information.* Physical review E, 2004. **69**(6): p. 066138.
20. Margolin, A.A., et al. *ARACNE: an algorithm for the reconstruction of gene regulatory*

networks in a mammalian cellular context. in *BMC bioinformatics*. 2006. BioMed Central.

21. Shannon, P., et al., *Cytoscape: a software environment for integrated models of biomolecular interaction networks.* Genome research, 2003. **13**(11): p. 2498-2504.
22. Tsiliki, G., et al., *Enriching nanomaterials omics data: an integration technique to generate biological descriptors.* Small Methods, 2017. **1**(11): p. 1700139.
23. Walkey, C.D., et al., *Protein corona fingerprinting predicts the cellular interaction of gold and silver nanoparticles.* ACS nano, 2014. **8**(3): p. 2439-2455.
24. Liu, R., et al., *Prediction of nanoparticles-cell association based on corona proteins and physicochemical properties.* Nanoscale, 2015. **7**(21): p. 9664-9675.

Annex 1

Importing csv files in the Jaqpot platform

Jaqpot supports different ways of importing data into the system. One straightforward way is via the upload of .csv files using Jaqpot's Dataset API.

More specifically, under the https://api.jaqpot.org/jaqpot/swagger/#/dataset API there is the *POST /dataset/csv* operation. The different parameters required by the method are as follows:

1. "title": the title of the resulting dataset
2. "description": a description of the content of the resulting dataset
3. "file": the .csv file to upload.

The *file* field accepts all kinds of .csv files, however, in order for the operation to complete successfully some general rules should be followed when creating the .csv file, as follows:

- The first row of the .csv file must contain the name of each column. This populates the *feature* names of the resulting dataset;
- The service understands the default csv value separator, comma. A full stop or point is accepted as a decimal separator in numerical values.
- Numerical values should not be enclosed in quotes.

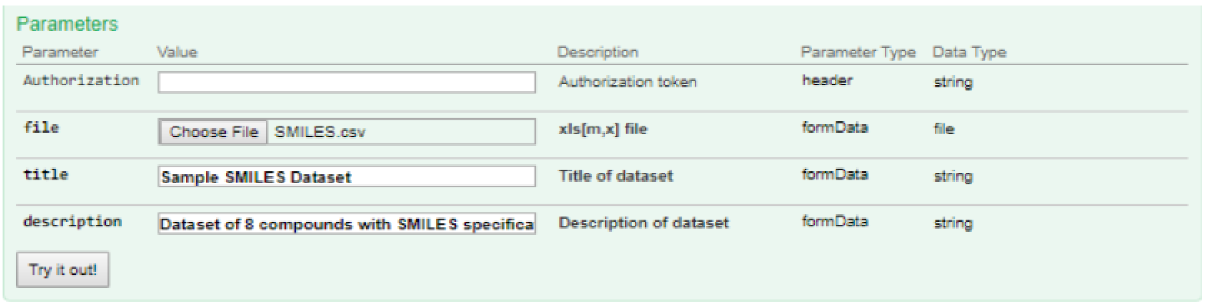
Below is an example of importing a csv file into the system:
We will use https://github.com/KinkyDesign/descriptor-csv-examples/blob/master/SMILES.csv as the example csv to import.

```
compound_name,                                   ChemblID,                                   SMILES
"GRISEOFULVIN","CHEMBL562","COC1=CC(=O)C[C@@H](C)[C@]12Oc3c(Cl)c(OC)cc(OC)c3C2=O"
"ACYCLOVIR","CHEMBL184","Nc1nc(O)c2ncn(COCCO)c2n1"
"TIMOLOL","CHEMBL499","CC(C)(C)NC[C@H](O)COc1nsnc1N2CCOCC2"
"OXPRENOLOL","CHEMBL546","CC(C)NCC(O)COc1ccccc1OCC=C"
"NEVIRAPINE","CHEMBL57","Cc1ccnc2N(C3CC3)c4ncccc4C(=O)Nc12"
"NICOTINE","CHEMBL3","CN1CCC[C@H]1c2cccnc2"
"FELODIPINE","CHEMBL1480","CCOC(=O)C1=C(C)NC(=C(C1c2cccc(Cl)c2Cl)C(=O)OC)C"
"DOPAMINE","CHEMBL59","NCCc1ccc(O)c(O)c1"
```

As indicated, all entries, except from the first row, are enclosed in quotes. That is because we do not have any numerical values in this example.

We fill the POST parameters with the following values as shown in Figure A1:

1. "title": "Sample SMILES Dataset"
2. "description": "Dataset of 8 compounds with SMILES specifications"
3. "file": SMILES.csv



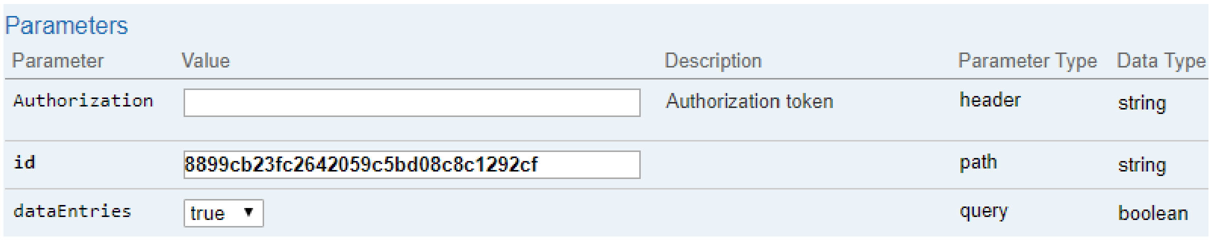
Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization		Authorization token	header	string
file	Choose File SMILES.csv	xls[m,x] file	formData	file
title	Sample SMILES Dataset	Title of dataset	formData	string
description	Dataset of 8 compounds with SMILES specifica	Description of dataset	formData	string

Try it out!

Figure A1 : Parameters to POST on dataset/csv service

If the service is successful it returns the dataset description (dataset without the dataEntries). We can receive the full dataset by GETting the dataset through the https://api.jaqpot.org/jaqpot/swagger/#!/dataset/getDataset method, using the id of the dataset, which is provided in the dataset description. In our example, the dataset id is *8899cb23fc2642059c5bd08c8c1292cf* (Figure A2).



Parameters

Parameter	Value	Description	Parameter Type	Data Type
Authorization		Authorization token	header	string
id	8899cb23fc2642059c5bd08c8c1292cf		path	string
dataEntries	true		query	boolean

Figure A2 : Parameters to GET the imported Dataset

The produced Jaqpot dataset in JSON format is:

```
{
  "meta": {
    "descriptions": [
      "Smile Dataset without smiles descriptors"
    ],
    "titles": [
      "Initial Smiles Dataset"
    ],
    "creators": [
      "6d9b8261-e255-4fca-9289-0fde48644cf8"
    ],
    "date": "2018-12-29T21:03:24.214+0000",
    "locked": false
  },
  "visible": true,
  "featured": false,
  "dataEntry": [
  {
   "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
   "entryId": {
    "name": "row1",
    "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
    "URI":
"https://api.jaqpot.org/jaqpot/services/substance/898b8ce77db641a5a85902439d5a93ab"
   },
   "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL562",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"COC1=CC(=O)C[C@@H](C)[C@]12Oc3c(Cl)c(OC)cc(OC)c3C2=O",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "GRISEOFULVIN"
   },
   "_id": "103c000ce306491595853edac194b876"
  },
  {
   "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
   "entryId": {
    "name": "row2",
    "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
    "URI":
"https://api.jaqpot.org/jaqpot/services/substance/4125500ad98144028ba1173410d13a01"
   },
   "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL184",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"Nc1nc(O)c2ncn(COCCO)c2n1",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "ACYCLOVIR"
   },
   "_id": "6adc390417574061ac13faae41533016"
  },
  {
   "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
   "entryId": {
    "name": "row3",
    "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
    "URI":
```

```
"https://api.jaqpot.org/jaqpot/services/substance/8a2bee65212e4fdfa583029d892a9a1c"
      },
      "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL499",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"CC(C)(C)NC[C@H](O)COc1nsnc1N2CCOCC2",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "TIMOLOL"
      },
      "_id": "5cad628b9c0a4f0c94ac7f9e30f10d26"
    },
    {
      "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
      "entryId": {
        "name": "row4",
        "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
        "URI":
"https://api.jaqpot.org/jaqpot/services/substance/9c012592054a4d39abe1eff41e5cbcd0"
      },
      "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL546",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"CC(C)NCC(O)COc1ccccc1OCC=C",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "OXPRENOLOL"
      },
      "_id": "77f4111fd08f4b6cbe4faf5faa386607"
    },
    {
      "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
      "entryId": {
        "name": "row5",
        "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
        "URI":
"https://api.jaqpot.org/jaqpot/services/substance/ff67c275e1934afaa39a1d7c1b5c8b15"
      },
      "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL57",
"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"Cc1ccnc2N(C3CC3)c4ncccc4C(=O)Nc12",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "NEVIRAPINE"
    },
    "_id": "f2aa060d75cc4bbfb6a54f9d290ce000"
   },
   {
    "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
    "entryId": {
     "name": "row6",
     "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
     "URI":
"https://api.jaqpot.org/jaqpot/services/substance/8b5559ca72f94ff29972100de4ff882f"
    },
    "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL3",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"CN1CCC[C@H]1c2cccnc2",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "NICOTINE"
    },
    "_id": "50e0101db274460f955d9be3f25b45ec"
   },
   {
    "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
    "entryId": {
     "name": "row7",
     "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
     "URI":
"https://api.jaqpot.org/jaqpot/services/substance/5bc8cdc9818e44d2bc7192547ecd075a"
    },
    "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL1480",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"CCOC(=O)C1=C(C)NC(=C(C1c2cccc(Cl)c2Cl)C(=O)OC)C",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "FELODIPINE"
```

```
      },
      "_id": "54a3dc6a42174726a5cfdc5e865237cd"
    },
    {
      "datasetId": "8899cb23fc2642059c5bd08c8c1292cf",
      "entryId": {
        "name": "row8",
        "ownerUUID": "7da545dd-2544-43b0-b834-9ec02553f7f2",
        "URI":
"https://api.jaqpot.org/jaqpot/services/substance/f963e0a1e968447e87b9a9b88d8e0451"
      },
      "values": {

"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d": "CHEMBL59",

"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a":
"NCCc1ccc(O)c(O)c1",

"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb": "DOPAMINE"
      },
      "_id": "cc8871515978476dadbda7d16d652cc8"
    }
  ],
  "features": [
    {
      "name": " ChemblID",
      "units": "NA",
      "conditions": {},
      "category": "EXPERIMENTAL",
      "uri":
"https://api.jaqpot.org/jaqpot/services/feature/_ChemblID_00867b85decb42f89b935f3434c423a
d"
    },
    {
      "name": "compound_name",
      "units": "NA",
      "conditions": {},
      "category": "EXPERIMENTAL",
      "uri":
"https://api.jaqpot.org/jaqpot/services/feature/compound_name_e44910d667ef4ee0805d850a0
96826cb"
    },
    {
      "name": " SMILES",
      "units": "NA",
      "conditions": {},
      "category": "EXPERIMENTAL",
      "uri":
"https://api.jaqpot.org/jaqpot/services/feature/_SMILES_650a25bc84df4adfa7cea0503cb4282a"
    }
  ],
  "totalRows": 8,
  "totalColumns": 3,
  "_id": "8899cb23fc2642059c5bd08c8c1292cf"
}
```