Datarail 2.0

# Datacube structure

The datacube is an instance of a class DATACUBE. The class contains the data, hash tables for each dimension (extracted from dictionaries) and has functions/routines.

Each datacube will contain the results of an experiment. The definition of an experiment is not completely settled, but in general it should have the following properties:

* Consistency of the data measured (same type, same assay, same transformation)
* An ‘almost complete’ cube
* Replicate should be consistent (although they may not be complete) but the number may differ for the different positions.

The main dimensions of the datacube are the following:

1. Cell lines
2. Measured property such as:
   1. Kinase level (through fluorescence, ELISA, mass spec, …)
   2. mRNA level
   3. GI50, or other phenotypes
3. Perturbation (drug, ligand, shRNA) that can have multiple dimensions (how to implement it: independent dimensions? Or linked?):
   1. Dose (different doses may be chosen depending on the drug/ligand)
   2. Treatment time
4. Time points (for trajectories) 🡪 this may be redundant with the treatment time.
5. Replicate (may not be the same number for each position 🡪 fill with NaNs?)

In addition, we can consider each position of the datacube as a vector (e.g. growth-response curve, single cell data). This vector will NOT have a size consistent across the different position of the cube. 🡪 how to integrate trajectories? Use of a matrix or have a correspondence between the ‘time points’ (4th dimension)

The main dimensions should be linked to a hash table that constitutes the ‘dictionary’ for each dimension.

# Dictionaries

The goal of the dictionary is to enable the comparison, concatenation and handling of datacubes derived from the same pool of experiments. The dictionary is used when constructing the datacube, it serves as a control when importing external data (from excel, or other experimental output) into a DATACUBE class.

The list for each dimension should be settled very early in the project. The dictionary may contain definition, and open fields, but the keys should be kept constant. For example, the dictionary for:

* cell lines can contain the fields:
  + Short name (key)
  + Long name
  + Subtype
  + Culture conditions
* Drugs contain the fields:
  + Name (key)
  + Alternative name
  + Drug family
  + Primary targets (derived from another dictionary)
  + Secondary targets (derived from another dictionary)
* Ligands:
  + Name (key)
  + Alternative name
  + Ligand family
  + Bound receptors (derived from another dictionary)
* Measured kinases (by antibody):
  + Name (key); distinguishable for the different antibodies
  + Target protein (derived from another dictionary)
  + Total or phosphorylation site
  + Antibody name/label
* shRNA:
  + Name (key); distinguishable for the different constructs
  + Target protein (derived from another dictionary)
  + Other property?
* mRNA:
  + probe name/ number (key)
  + gene name
  + corresponding protein (if direct relation; derived from another dictionary)
* proteins/kinases:
  + name (key)
  + other properties

# Functions

Functions to be embedded in the datacube (DATACUBE class methods):

* extraction of a sub-datacube by selecting some instances in a dimension
  + addressing by names/keys/ values (for numeric fields like doses, time points, replicates)
  + addressing by numbers/indexes
* replicate average 🡪 generate a new datacube with one less dimension
* normalization (along certain dimensions)
* embedment of the keys/hash table in the datacube for portability and internal calls; this represents a partition of the dictionary
* apply a function on a position of the datacube (vector/matrix) and extract a triplet of properties (new datacubes)

Additional functions for proper handling should contain:

* string matching to make correspondences/intersections between datacubes
* addition/subtraction of a vector/matrix across many dimensions (automatic ‘repmat’)