Integrating R and the CDK

Enhanced Chemical Data Mining

Rajarshi Guha

School of Informatics
Indiana University

21st May, 2007
Covington, KY

Outline

Outline

Cheminformatics & Modeling

- Chemical information is available from multiple sources
  - Databases
  - Algorithms
- The information must be used in some way
  - Summarized in the form of a report
  - Modeled (numerically / statistically)
- A variety of software is available for both cheminformatics and modeling
  - Can we get something that allows us the best of both worlds?

Cheminformatics & Modeling

- Chemical information is available from multiple sources
  - Databases
  - Algorithms
- The information must be used in some way
  - Summarized in the form of a report
  - Modeled (numerically / statistically)
- A variety of software is available for both cheminformatics and modeling
  - Can we get something that allows us the best of both worlds?

Modeling Environments

- Many cheminformatics applications include statistical functionality
- Useful, but has a number of drawbacks
  - Their focus is on cheminformatics, not necessarily statistics
  - Usually limited in the statistics offerings
- Makes sense to use software designed for statistics
  - SAS, SPSS
  - Minitab, R
  - ...
- However, these focus on statistics and not cheminformatics

Combine a statistics package with a cheminformatics package

Modeling Environments

- Many cheminformatics applications include statistical functionality
- Useful, but has a number of drawbacks
  - Their focus is on cheminformatics, not necessarily statistics
  - Usually limited in the statistics offerings
- Makes sense to use software designed for statistics
  - SAS, SPSS
  - Minitab, R
  - ...
- However, these focus on statistics and not cheminformatics

Combine a statistics package with a cheminformatics package

Why Use R?

- Open source statistical programming environment
- Full fledged programming language
- A wide variety of statistical methods - traditional, well tested methods as well as state of the art methods
- Extensive and active community
- Easy to integrate into other environments
  - Workflow tools (Pipeline Pilot, Knime)
  - Java programs (CDK)

Why Use the CDK?

- Open source Java library for cheminformatics
- Covers basic cheminformatics functionality
- Also provides descriptors, 3D coordinate generation, rigid alignment etc.
- Active development community

Steinbeck, C. et al., *Curr. Pharm. Des.*, **2007**, *12*, 2110 – 2120

Outline

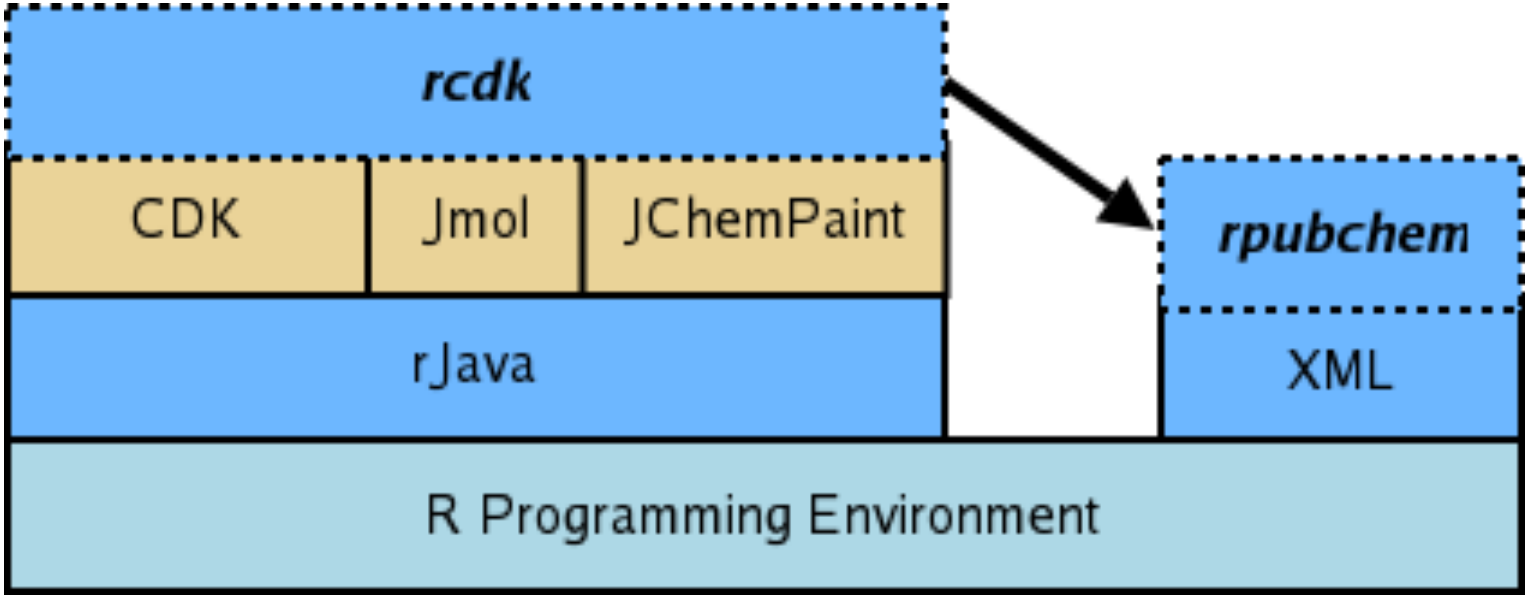
Goals of the Integration

- Allow one to use cheminformatics tools within R
- Ensure that cheminformatics functionality follows R idioms
- Provide access to PubChem data - compound and bioassay

Using the CDK in R

Overview

- Based on the `rJava` package
- Access to basic cheminformatics
- View 3D structures via Jmol
- View and edit 2D structures via JChemPaint
- Single R package to install - no extra downloads



rcdk Functionality

	Function	Description
Editing and viewing	`draw.molecule`	Draw 2D structures
	`view.molecule`	View a molecule in 3D
	`view.molecule.2d`	View a molecule in 2D
	`view.molecule.table`	View 3D molecules along with data
IO	`load.molecules`	Load arbitrary molecular file formats
	`write.molecules`	Write molecules to MDL SD formatted files
Descriptors	`eval.desc`	Evaluate a single descriptor
	`get.desc.engine`	Get the descriptor *engine* which can be used to automatically generate all descriptors
	`get.desc.classnames`	Get a list of available descriptors
Miscellaneous	`get.fingerprint`	Generate binary fingerprints
	`get.smiles`	Obtain SMILES representations
	`parse.smiles`	Parse a SMILES representation into a CDK molecule object
	`get.property`	Retrieve arbitrary properties on molecules
	`set.property`	Set arbitrary properties on molecules
	`remove.property`	Remove a property from a molecule
	`get.totalcharge`	Get the total charge of a molecule
	`get.total.hydrogen.count`	Get the count of (implicit and explicit) hydrogens
	`remove.hydrogens`	Remove hydrogens from a molecule

Clustering Molecules

Generating fingerprints

```
> library(rcdk)
> mols <- load.molecules('dhfr_3d.sd')
> fp.list <- lapply(mols, get.fingerprint)
```

- The fingerprints are hashed, 1024-bit
- Gives us a `list` of numeric vectors, indicating the bit positions that are on
- The next step is to evaluate a distance matrix

Clustering Molecules

Fingerprint manipulation

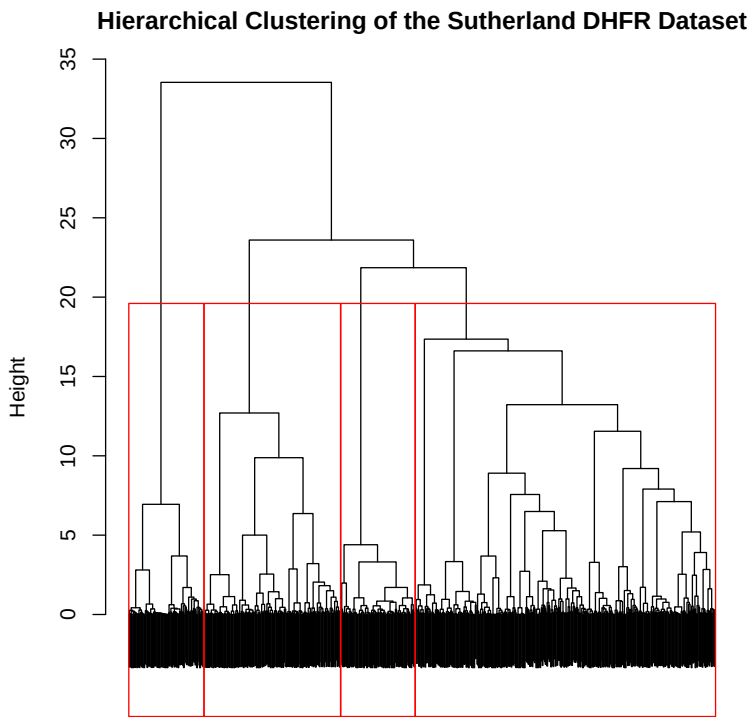
```
> library(fingerprint)
> fp.dist <- fp.sim.matrix(fp.list)
> fp.dist <- as.dist(1-fp.dist)
```

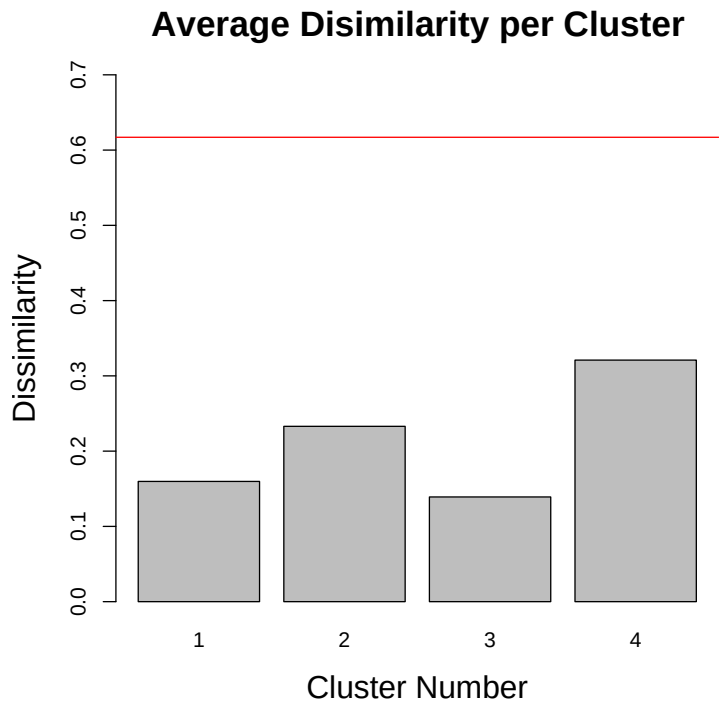
- The `fingerprint` package for R allows one to easily handle fingerprint data
- Recognizes CDK, BCI and MOE fingerprint formats
- We can now perform the clustering

Clustering Molecules

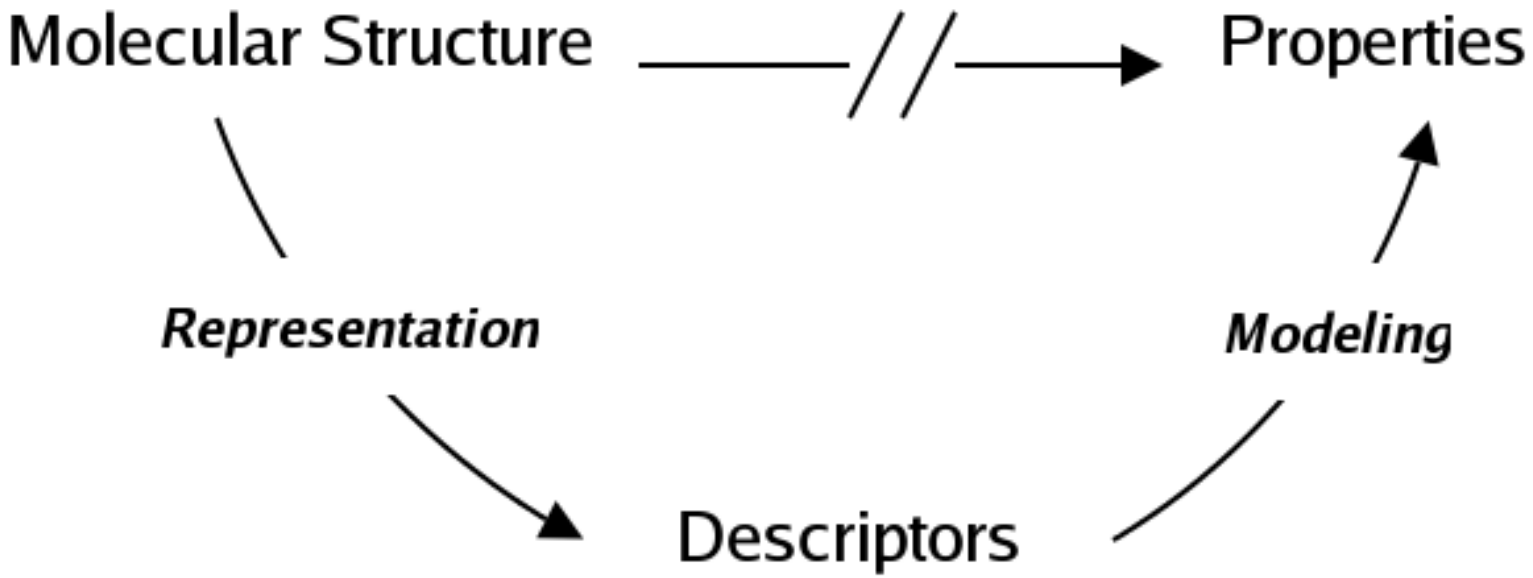
Hierarchical clustering

```
> clus.hier <- hclust(fp.dist, method='ward')
```





QSAR Modeling



QSAR Modeling

Representation

- Molecules must be represented in ome machine readable format
- SD files are commonly used
- The `draw.molecule` function can be used to get a 2D structure editor
- The resultant molecule can be accessed in the R workspace
- Currently can't generate 3D structures

QSAR Modeling

Calculating descriptors

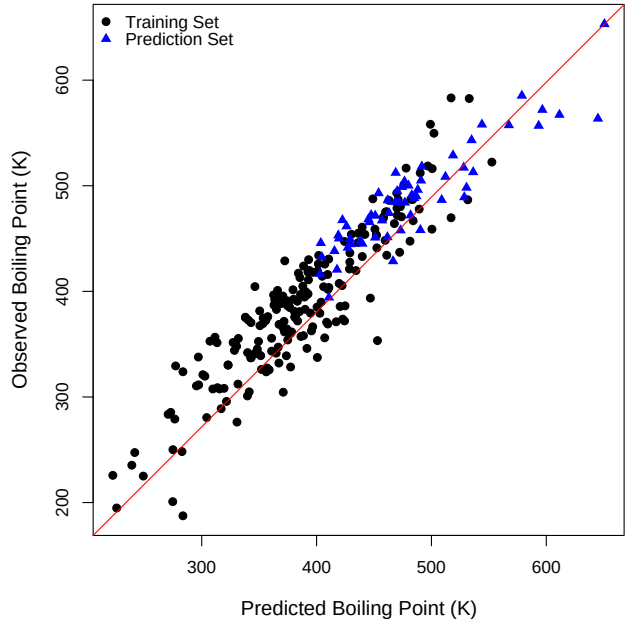
```
> mols <- load.molecules('bp.sdf')
> desc.names <- c(
+  'org.openscience.cdk.qsar.descriptors.molecular.BCUTDescriptor',
+  'org.openscience.cdk.qsar.descriptors.molecular.CPSADescriptor',
+  'org.openscience.cdk.qsar.descriptors.molecular.XLogPDescriptor',
+  'org.openscience.cdk.qsar.descriptors.molecular.KappaShapeIndicesDescriptor')

> desc.list <- list()
> for (i in 1:length(mols)) {
+  tmp <- c()
+  for (j in 1:length(desc.names)) {
+    values <-  eval.desc(mols.qsar[[i]], desc.names[j])
+    tmp <- c(tmp, values)
+    if (i == 1) data.names <- c(data.names, paste(prefix[j], 1:length(values), sep='.'))
+  }
+  desc.list[[i]] <- tmp
+ }
> desc.data <- as.data.frame(do.call('rbind', desc.list))
```

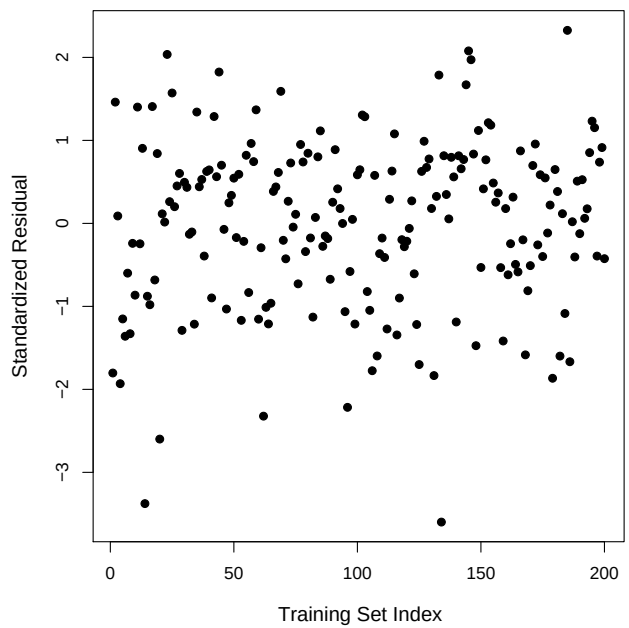
- Currently have to specify descriptors and generate names
- This will be automated in the next release
- The result is a data matrix which can then be modeled using the various methods available in R

QSAR Modeling

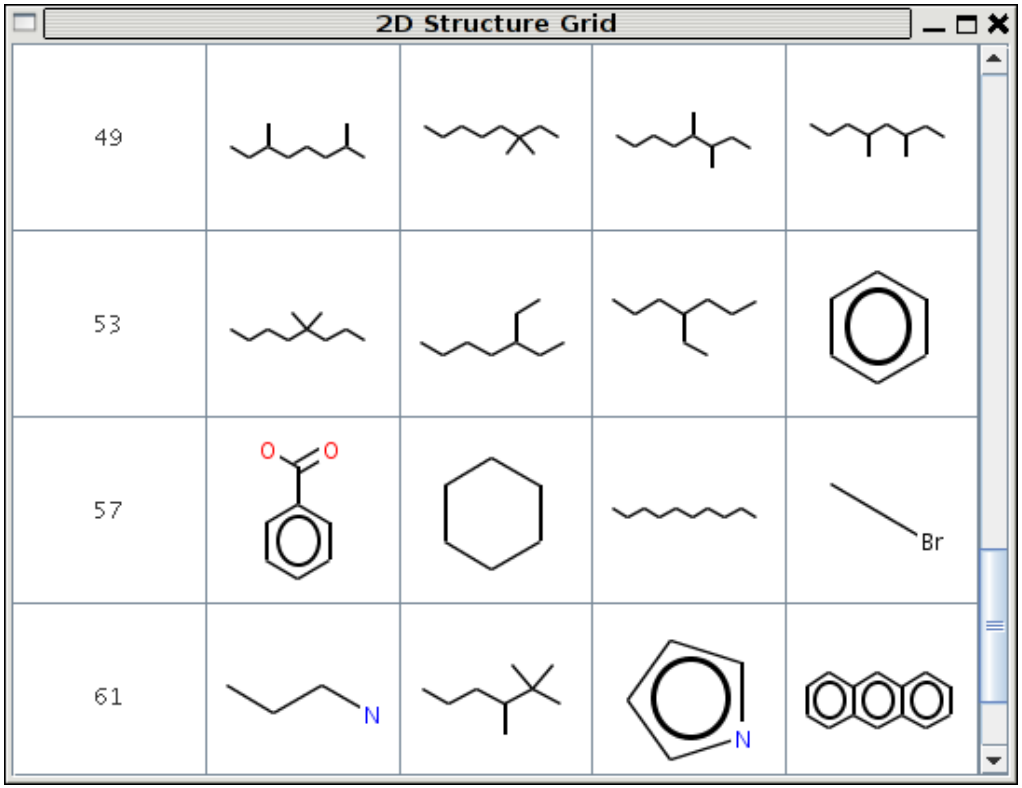
The results for an OLS model

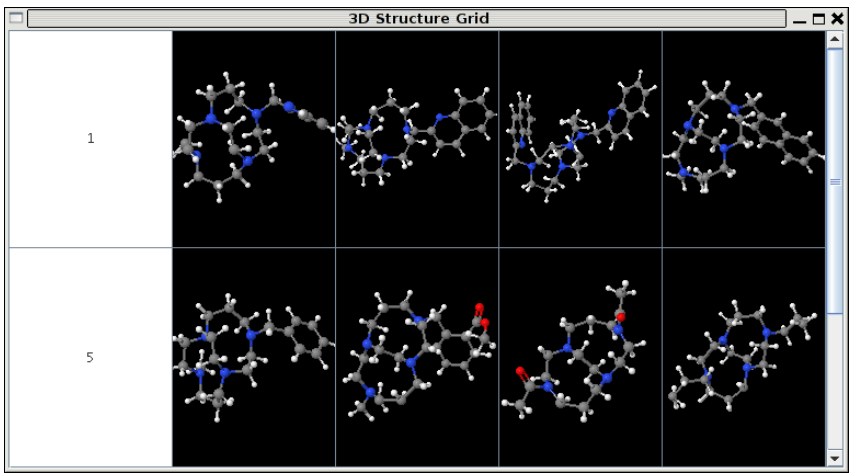
```
> summary(model)
Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)      25.611     13.743   1.864   0.0639 .
BCUT.6           26.155      1.503  17.406  < 2e-16 ***
CPSA.12        2565.552    181.668  14.122  < 2e-16 ***
WeightedPath.1    7.645      0.608  12.573  < 2e-16 ***
MDE.11           91.355     17.916   5.099 8.05e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 28.88 on 195 degrees of freedom
Multiple R-Squared: 0.8311,     Adjusted R-squared: 0.8276
F-statistic: 239.9 on 4 and 195 DF,  p-value: < 2.2e-16
```





Molecular Visualization





Outline

Accesing PubChem

What can we get?

- 2D structures for ∼10M compounds
- Some precomputed properties
- ∼500 bioassay datasets ranging from 100 observations to 80,000 observations
- Searchable by keyword

What's the data like?

- Structure data can be retrieved in the form of SD or XML files
- Structures are represented as 2D coordinates as well as SMILES
- Bioassay data consists of XML or CSV files
  - Column descriptions etc. are located in a separate XML file

Accessing PubChem

What's the problem?

- Combining bioassay data & metadata by hand is painful
- The XML descriptions are readable but verbose
- Getting data can be a multi-step process, involving search, download and processing

How does `rpubchem` help?

- Automated download of data & metadata files for compounds and bioassays
- Extracts data into a `data.frame`
- Assigns appropriate column names and adds extra information (such as units) as attributes
- Allows keyword searches for bioassays

Example Usage

Getting structural data

```
> library(rpubchem)
> library(rcdk)
> structs <- get.cid( sample(1:10000, 10) )
> mols <- lapply(structs[,2], parse.smiles)
> view.molecule.2d(mols)
```

- We retrieve the SMILES along with some of the precomputed values
- Heavy atom count, formal charge, H count, H-bond donors and acceptors

Example Usage

Getting bioassay data

```
> aids <- find.assay.id("lung+cancer+carcinoma")
> sapply(aids, function(x) get.assay.desc(x)$assay.desc)
> adata <- get.assay(175)
```

- Search for assays by keywords
- Get short descriptions
- Get the actual assay data

Example Usage

What do we get?

Field Name	Meaning
`PUBCHEM.SID`	Substance ID
`PUBCHEM.CID`	Compound ID
`PUBCHEM.ACTIVITY.OUTCOME`	Activity outcome, which can be one of *active, inactive, inconclusive, unspecified*
`PUBCHEM.ACTIVITY.SCORE`	PubChem activity score, higher is more active
`PUBCHEM.ASSAYDATA.COMMENT`	Test result specific comment

- A set of fixed fields
- Arbitrary number of assay specific fields
- Have to process a description file

Example Usage

What do we get?

```
> names(adata)
[1] "PUBCHEM.SID"                "PUBCHEM.CID"
[3] "PUBCHEM.ACTIVITY.OUTCOME"   "PUBCHEM.ACTIVITY.SCORE"
[5] "PUBCHEM.ASSAYDATA.COMMENT" "conc"
[7] "giprct"                     "nbrtest"
[9] "range"

> attr(adata, "comments")
[1] "These data are a subset of the data from the NCI yeast anticancer
drug screen. Compounds are identified by the NCI NSC number. In the
Seattle Project numbering system, the mlh1 rad18 strain is SPY number
50858. Compounds with inhibition of growth(giprct) >= 70% were
considered active. Activity score was based on increasing values of giprct."

> attr(adata, 'types')
$conc
[1] "concentration tested, unit: micromolar"
[2] "uM"

$giprct
[1] "Inhibition of growth compared to untreated controls (%)"
[2] "NA"

$nbrtest
[1] "Number of tests averaged for this data point"
[2] "NA"

$range
[1] "Range of values for this data point" "NA"
```

Outline

Future Work

- Improve the coverage of the CDK functionality
- Improve descriptor generation and naming
- Include support for more chemical file formats
- 2D Structure-data tables
- Avoid in-memory processing of large PubChem assay datasets

Conclusions

- Integrating the CDK with R enhances the utility of the statistical environment for cheminformatics applications
- The integration is still at the *programmer* level
  - Applications can be built up from the available functionality
- `rpubchem` provides easy access to large amounts of biological and chemical data

Acknowledgements

- NIH
- CDK developers
- Simon Urbanek