Chemical Structure Inputs for PUG-REST

S. Kim, J. Cuadros

August 21st, 2019

Objectives

- Use SMILES and InChI strings to specify the input compound for a PUG-REST request.
- Use a structure-data (SD) file to specify the input compound for a PUG-REST request.
- Learn to submit a PUG-REST request using the HTTP-POST method.

Background

You can use a chemical structure as an input for a PUG-REST request. PUG-REST accepts some popular chemical structure line notations such as SMILES and InChI strings. It is also possible to use an Structure-Data File (SDF) as a structure input.

To learn how to specify the structure input in a PUG-REST request, one needs to know that there are two methods by which data are transferred from clients (users) and servers (PubChem) through PUG-REST. Discussing what these methods are in detail is beyond the scope of this material, and it is enough to know three things:

- When you make a PUG-REST request by typing the request URL in the address bar of your web browser (such as Google Chrome, MS Internet Explorer), the HTTP GET method is used
- The HTTP GET method transfers information encoded in a single-line URL.
- Some chemical structure inputs are not appropriate to encode in a single-line URL (because they may contain special characters not compatible with the URL syntax, span over multiple lines, or too long), and the HTTP POST needs to be used for such cases.

For more information on HTTP GET and POST, read the following documents.

- HTTP request methods (https://www.w3schools.com/tags/ref_httpmethods.asp (https://www.w3schools.com/tags/ref_httpmethods.asp))
- GET vs. POST (https://www.diffen.com/difference/GET-vs-POST-HTTP-Requests (https://www.diffen.com/difference/GET-vs-POST-HTTP-Requests))

Let's start checking if the `httr` package is available, installing it if needed. Then, we load it.

```
if(!require("httr", quietly=TRUE)) {
  install.packages("httr", repos="https://cloud.r-project.org/",
         quiet=TRUE, type="binary")
  library("httr", quietly=TRUE)
}
```

```
## Warning: package 'httr' was built under R version 3.6.1
```

Packages are the way that libraries (additional functions, data types, constants, data sets…) are distributed in the R environment. In order to use a package, it has to be installed (only once per running environment) with the `install.packages` function. Then, if we load it (in a specific R session) using `library` , its functions can be called as they were in the base environment.

The `require` function checks whether a package has already been installed and loads it if so. It returns a logical value than can be used to install the package if it was not available.

The `httr` package allows a finer grade manipulation of the HTTP communications. That's why we will use it in this activity. A quick introduction to the package is available at https://cran.r-project.org/web/packages/httr/vignettes/quickstart.html (https://cran.r-project.org/web/packages/httr/vignettes/quickstart.html).

1. Using the HTTP GET method.

1.1. Structure encoded in the URL path.

In some cases, you can encode a chemical structure in the PUG-REST request URL path as in the following example.

```
prolog <- 'https://pubchem.ncbi.nlm.nih.gov/rest/pug'
smiles1 <- "CC(C)CC1=CC=C(C=C1)C(C)C(=O)O"
url <- paste(prolog, "/compound/smiles/", smiles1, "/cids/txt", sep="")
url
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/CC(C)CC1=CC=C(C=C1)C(C)C(=
O)O/cids/txt"
```

```
res = GET(url)
content(res,"text",encoding="UTF-8")
```

```
## [1] "3672\n"
```

It should be noteworthy that some SMILES strings contain special characters, such as the forward slash ("/"), which is also used in the URL path. These special characters conflict with the PUG-REST request URL syntax, causing an error when used in the PUG-REST request URL.

Also note that the backslash character has to be escaped when used in a string in R. To include a blackslash, we have to write `\\` .

```
smiles2 <- "CC1=C([C@@](SC1=O)(C)/C=C(\\C)/C=C)O"
url <- paste(prolog, "/compound/smiles/", smiles2, "/cids/txt", sep="")
url
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/CC1=C([C@@](SC1=O)(C)/C=C
(\\C)/C=C)O/cids/txt"
```

```
res <- GET(url)
content(res,"text",encoding="UTF-8")
```

```
## [1] "Status: 400\nCode: PUGREST.BadRequest\nMessage: Unable to standardize the given struc
ture - perhaps some special characters need to be escaped or data packed in a MIME form?\nDet
ail: error: \nDetail: status: 400\nDetail: output: Caught ncbi::CException: Standardization f
ailed\nDetail: Output Log:\nDetail: Record 1: Warning: Cactvs Ensemble cannot be created from
input string\nDetail: Record 1: Error: Unable to convert input into a compound object\nDetai
l: \nDetail: \n"
```

1.2. Structure encoded as a URL argument

To circumvent the issue mentioned above, the SMILES string may be encoded as the URL argments (as an optional parameter followed by the “?” character).

```
url2 <- paste(prolog, "/compound/smiles/cids/txt?smiles=", smiles2, sep="")
url2
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/cids/txt?smiles=CC1=C([C@@]
(SC1=O)(C)/C=C(\\C)/C=C)O"
```

```
res2 <- GET(url2)
content(res2,"text",encoding="UTF-8")
```

```
## [1] "135403829\n"
```

1.3. Structure encoded as a URL-encoded URL argument

It is also possible to pass the structure query as a URL-encoded argument. The following example does the same task as the previous example does.

This is safer in case the argument includes `&`, `?` or other reserved characters.

```
url3 <- paste(prolog, "/compound/smiles/cids/txt?smiles=", URLencode(smiles2,reserved=TRUE),
 sep="")
url3
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/cids/txt?smiles=CC1%3DC%28%
5BC%40%40%5D%28SC1%3DO%29%28C%29%2FC%3DC%28%5CC%29%2FC%3DC%29O"
```

```
res3 <- GET(url3)
content(res3,"text",encoding="UTF-8")
```

```
## [1] "135403829\n"
```

The object returned from a web service request (res, res2, and res3 in our examples) contains information on the request URL through which the data have been retrieved. This information can be accessed using the `$url` attribute of the object, as shown in this example:

```
res2$url   # from (2) structure encoded as a URL argument
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/cids/txt?smiles=CC1=C([C@@]
(SC1=O)(C)/C=C(\\C)/C=C)O"
```

```
res3$url   # from (3) structure encoded as a URL-encoded URL argument
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/cids/txt?smiles=CC1%3DC%28%
5BC%40%40%5D%28SC1%3DO%29%28C%29%2FC%3DC%28%5CC%29%2FC%3DC%29O"
```

Note that URL-encoding does not work for PubChem when including structure information in the URL path.

Exercise 1a: Retrieve the hydrogen bond donor and acceptor counts, TPSA, and XLogP of the chemical represented by the SMILES string: “C1=CC(=C(C=C1Cl)O)OC2=C(C=C(C=C2)Cl)Cl”. When construct a PUG-REST url for this request, encode the structure in the URL path.

```
# Write your code here
```

Exercise 1b: Get the CID corresponding to the following InChI string, using the HTTP GET method.

```
inchi <- "InChI=1S/C17H14O4S/c1-22(19,20)14-9-7-12(8-10-14)15-11-21-17(18)16(15)13-5-3-2-4-6-13/h2-10H,11H2,1H3"

# Write your code here
```

2. Using the HTTP POST method

2.1. Comparison of HTTP POST and GET

All the three examples above use the HTTP GET method, as implied in the use of the `GET` function. Alternatively, one can use the HTTP POST method. For example, the following example returns the identical result as the last two HTTP GET examples.

```
url <- paste(prolog, "/compound/smiles/cids/txt", sep="")
struct <- list(smiles=smiles2)

res <- POST(url, body = struct)
res$url
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/smiles/cids/txt"
```

```
content(res,"text",encoding="UTF-8")
```

```
## [1] "135403829\n"
```

When using the HTTP POST method, information is passed as data through the body argument. Because of this, the URL stored in the `res$url` does not contain structure informaion.

2.2. HTTP POST for multi-line structure input

The HTTP POST method should be used if the input molecular structure for PUG-REST request span over multiple lines (e.g., stored in a structure-data file (SDF) format). The SDF file contains structure information of a molecule in a multi-line format, along with other data.

```
mysdf <- "1983
  -OEChem-07241917072D

 20 20  0     0  0  0  0  0  0999 V2000
    2.8660   -2.5950    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
    4.5981    1.4050    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
    2.8660    1.4050    0.0000 N   0  0  0  0  0  0  0  0  0  0  0  0
    2.8660    0.4050    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.7320   -0.0950    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.0000   -0.0950    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.7320   -1.0950    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.0000   -1.0950    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.8660   -1.5950    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.7320    1.9050    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.7320    2.9050    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.2690    0.2150    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    1.4631    0.2150    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.3291    1.7150    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.2690   -1.4050    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    1.4631   -1.4050    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.3520    2.9050    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    3.7320    3.5250    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    3.1120    2.9050    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.3291   -2.9050    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  9  1  0  0  0  0
  1 20  1  0  0  0  0
  2 10  2  0  0  0  0
  3  4  1  0  0  0  0
  3 10  1  0  0  0  0
  3 14  1  0  0  0  0
  4  5  2  0  0  0  0
  4  6  1  0  0  0  0
  5  7  1  0  0  0  0
  5 12  1  0  0  0  0
  6  8  2  0  0  0  0
  6 13  1  0  0  0  0
  7  9  2  0  0  0  0
  7 15  1  0  0  0  0
  8  9  1  0  0  0  0
  8 16  1  0  0  0  0
 10 11  1  0  0  0  0
 11 17  1  0  0  0  0
 11 18  1  0  0  0  0
 11 19  1  0  0  0  0
M  END
> <PUBCHEM_COMPOUND_CID>
1983

> <PUBCHEM_COMPOUND_CANONICALIZED>
1

> <PUBCHEM_CACTVS_COMPLEXITY>
139

> <PUBCHEM_CACTVS_HBOND_ACCEPTOR>
2
```

```
> <PUBCHEM_CACTVS_HBOND_DONOR>
2

> <PUBCHEM_CACTVS_ROTATABLE_BOND>
1
$$$$
"
```

Multi-line string in R can be written without any special consideration. This multi-line sdf data is used as an input for a PUG-REST request through the HTTP POST.

```
url <- paste(prolog, "/compound/sdf/cids/txt", sep="")
mydata <- list(sdf=mysdf)

res <- POST(url, body = mydata)
res$url
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/sdf/cids/txt"
```

```
content(res,"text",encoding="UTF-8")
```

```
## [1] "1983\n"
```

Note that HTTP POST should be used for the input specification using a SDF format. Although HTTP GET may work if data is URL-encoded, it will be more dependent of URL length limitations.

```
url3 <- paste(prolog, "/compound/sdf/cids/txt?sdf=", URLencode(mysdf,reserved=TRUE), sep="")
url3
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/sdf/cids/txt?sdf=1983%0A%20%20-OEC
hem-07241917072D%0A%0A%2020%2020%20%200%20%20%20%20%200%20%200%20%200%20%200%20%20099
9%20V2000%0A%20%20%20%202.8660%20%20%20-2.5950%20%20%20%200.0000%20O%20%20%200%20%200%20%200%
20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%204.5981%20%20%20%
201.4050%20%20%20%200.0000%20O%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%
200%20%200%20%200%20%200%0A%20%20%20%202.8660%20%20%20%201.4050%20%20%20%200.0000%20N%20%20%2
00%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%2
02.8660%20%20%20%200.4050%20%20%20%200.0000%20C%20%20%200%20%200%20%200%20%200%20%200%20%200%
20%200%20%200%20%200%20%200%20%200%0A%20%20%20%203.7320%20%20%20-0.0950%20%20%20%200.0
000%20C%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%20
0%0A%20%20%20%202.0000%20%20%20-0.0950%20%20%20%200.0000%20C%20%20%200%20%200%20%200%20%200%2
0%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%203.7320%20%20%20-1.0950%2
0%20%20%200.0000%20C%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%20
0%20%200%20%200%0A%20%20%20%202.0000%20%20%20-1.0950%20%20%20%200.0000%20C%20%20%200%20%200%2
0%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%202.8660%20%
20%20-1.5950%20%20%20%200.0000%20C%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%20
0%20%200%20%200%20%200%0A%20%20%20%203.7320%20%20%20%201.9050%20%20%20%200.0000%20C%2
0%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%2
0%20%203.7320%20%20%20%202.9050%20%20%20%200.0000%20C%20%20%200%20%200%20%200%20%200%20%200%2
0%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%204.2690%20%20%20%200.2150%20%20%
20%200.0000%20H%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%2
00%20%200%0A%20%20%20%201.4631%20%20%20%200.2150%20%20%20%200.0000%20H%20%20%200%20%200%20%20
0%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%202.3291%20%20%2
0%201.7150%20%20%20%200.0000%20H%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%2
0%200%20%200%20%200%20%200%0A%20%20%20%204.2690%20%20%20-1.4050%20%20%20%200.0000%20H%20%20%2
00%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%2
01.4631%20%20%20-1.4050%20%20%20%200.0000%20H%20%20%200%20%200%20%200%20%200%20%200%20%200%2
0%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%204.3520%20%20%20%202.9050%20%20%20%200.
0000%20H%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%2
00%0A%20%20%20%203.7320%20%20%20%203.5250%20%20%20%200.0000%20H%20%20%200%20%200%20%200%20%20
0%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%20%20%203.1120%20%20%20%202.9
050%20%20%20%200.0000%20H%20%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%2
0%200%20%200%20%200%0A%20%20%20%202.3291%20%20%20-2.9050%20%20%20%200.0000%20H%20%20%200%20%2
00%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%20%200%0A%20%201%20%209%20%
201%20%200%20%200%20%200%20%200%0A%20%201%2020%20%201%20%200%20%200%20%200%20%200%0A%20%202%2
010%20%202%20%200%20%200%20%200%20%200%0A%20%203%20%204%20%201%20%200%20%200%20%200%20%200%0
A%20%203%2010%20%201%20%200%20%200%20%200%20%200%0A%20%203%2014%20%201%20%200%20%200%20%200%2
0%200%0A%20%204%20%205%20%202%20%200%20%200%20%200%20%200%0A%20%204%20%206%20%201%20%200%20%2
00%20%200%20%200%0A%20%205%20%207%20%201%20%200%20%200%20%200%20%200%0A%20%205%2012%20%201%2
0%200%20%200%20%200%20%200%0A%20%206%20%208%20%202%20%200%20%200%20%200%20%200%0A%20%206%201
3%20%201%20%200%20%200%20%200%20%200%0A%20%207%20%209%20%202%20%200%20%200%20%200%20%200%0A%2
0%207%2015%20%201%20%200%20%200%20%200%20%200%0A%20%208%20%209%20%201%20%200%20%200%20%200%2
0%200%0A%20%208%2016%20%201%20%200%20%200%20%200%20%200%0A%2010%2011%20%201%20%200%20%200%20%
200%20%200%0A%2011%2017%20%201%20%200%20%200%20%200%20%200%0A%2011%2018%20%201%20%200%20%200%
20%200%20%200%0A%2011%2019%20%201%20%200%20%200%20%200%20%200%0AM%20%20END%0A%3E%20%3CPUBCHEM
_COMPOUND_CID%3E%0A1983%0A%0A%3E%20%3CPUBCHEM_COMPOUND_CANONICALIZED%3E%0A1%0A%0A%3E%20%3CPUB
CHEM_CACTVS_COMPLEXITY%3E%0A139%0A%0A%3E%20%3CPUBCHEM_CACTVS_HBOND_ACCEPTOR%3E%0A2%0A%0A%3E%2
0%3CPUBCHEM_CACTVS_HBOND_DONOR%3E%0A2%0A%0A%3E%20%3CPUBCHEM_CACTVS_ROTATABLE_BOND%3E%0A1%0A%2
4%24%24%24%0A"
```

```
res3 <- GET(url3)
content(res3,"text",encoding="UTF-8")
```

```
## [1] "1983\n"
```

2.3. HTTP POST for multi-line structure input

One may want to use the structure stored in a file as the input for a PUG-REST request. The following code shows how to read an SDF file into a variable.

```
mysdf <- paste(readLines('Structure2D_CID_5288826.sdf'),collapse="\n")
cat(mysdf)
```

```
## 5288826
##   -OEChem-08171913162D
##
##  40 44  0     1  0  0  0  0  0999 V2000
##     2.2314    0.0528    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
##     2.0000   -2.4021    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
##     2.0000    2.4021    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
##     6.1607   -0.9511    0.0000 N   0  0  3  0  0  0  0  0  0  0  0  0
##     3.6897   -0.4755    0.0000 C   0  0  1  0  0  0  0  0  0  0  0  0
##     4.5133   -0.9511    0.0000 C   0  0  2  0  0  0  0  0  0  0  0  0
##     5.3370   -0.4755    0.0000 C   0  0  1  0  0  0  0  0  0  0  0  0
##     2.8660   -0.9511    0.0000 C   0  0  2  0  0  0  0  0  0  0  0  0
##     4.2392    0.2219    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6897    0.4755    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     5.3370    0.4755    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     5.5918    0.2219    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     4.5133    0.9511    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     2.8660   -1.9022    0.0000 C   0  0  2  0  0  0  0  0  0  0  0  0
##     4.5133   -1.9022    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     2.8660    0.9511    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6897   -2.3777    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     6.8418   -1.6832    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     4.5133    1.9022    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     2.8660    1.9022    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6897    2.3777    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
##     5.0597   -1.6022    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.6284   -1.2740    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     2.0496   -1.1875    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     4.3760    0.8266    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6795    0.4887    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.9476    0.3679    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.5490    1.0581    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     6.1840    0.4057    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.4989    0.8349    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     2.8660   -2.5222    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.0503   -2.2122    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6897   -2.9977    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     6.3879   -2.1055    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     7.2641   -2.1371    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     7.2957   -1.2609    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     5.0503    2.2122    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     3.6897    2.9977    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     2.0000   -3.0222    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##     2.0000    3.0222    0.0000 H   0  0  0  0  0  0  0  0  0  0  0  0
##   1  8  1  0  0  0  0
##   1 16  1  0  0  0  0
##  14  2  1  6  0  0  0
##   2 39  1  0  0  0  0
##   3 20  1  0  0  0  0
##   3 40  1  0  0  0  0
##   4  7  1  0  0  0  0
##   4 12  1  0  0  0  0
##   4 18  1  0  0  0  0
##   5  6  1  0  0  0  0
##   5  8  1  0  0  0  0
##   5  9  1  1  0  0  0
##   5 10  1  0  0  0  0
```

```
##   6  7  1  0  0  0  0
##   6 15  1  0  0  0  0
##   6 22  1  1  0  0  0
##   7 11  1  0  0  0  0
##   7 23  1  6  0  0  0
##   8 14  1  0  0  0  0
##   8 24  1  1  0  0  0
##   9 12  1  0  0  0  0
##   9 25  1  0  0  0  0
##   9 26  1  0  0  0  0
##  10 13  2  0  0  0  0
##  10 16  1  0  0  0  0
##  11 13  1  0  0  0  0
##  11 27  1  0  0  0  0
##  11 28  1  0  0  0  0
##  12 29  1  0  0  0  0
##  12 30  1  0  0  0  0
##  13 19  1  0  0  0  0
##  14 17  1  0  0  0  0
##  14 31  1  0  0  0  0
##  15 17  2  0  0  0  0
##  15 32  1  0  0  0  0
##  16 20  2  0  0  0  0
##  17 33  1  0  0  0  0
##  18 34  1  0  0  0  0
##  18 35  1  0  0  0  0
##  18 36  1  0  0  0  0
##  19 21  2  0  0  0  0
##  19 37  1  0  0  0  0
##  20 21  1  0  0  0  0
##  21 38  1  0  0  0  0
## M  END
## > <PUBCHEM_COMPOUND_CID>
## 5288826
##
## > <PUBCHEM_COMPOUND_CANONICALIZED>
## 1
##
## > <PUBCHEM_CACTVS_COMPLEXITY>
## 494
##
## > <PUBCHEM_CACTVS_HBOND_ACCEPTOR>
## 4
##
## > <PUBCHEM_CACTVS_HBOND_DONOR>
## 2
##
## > <PUBCHEM_CACTVS_ROTATABLE_BOND>
## 0
##
## > <PUBCHEM_CACTVS_SUBSKEYS>
## AAADceB6MAAAAAAAAAAAAAAAAAAASAAAAA8YIEAAAAWAEjBAAAAHgAACAAADzzhmAYyBoMABgCAAiBCAAACCAAgIA
AIiAAOiIgNNiKGsRuGeCOkwBGLuAew8PcPoAABAAAYQADQAAaAADSAAAAAAAAAAA==
##
## > <PUBCHEM_IUPAC_OPENEYE_NAME>
## (4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
##
```

```
## > <PUBCHEM_IUPAC_CAS_NAME>
## (4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
##
## > <PUBCHEM_IUPAC_NAME_MARKUP>
## (4<I>R</I>,4<I>a</I><I>R</I>,7<I>S</I>,7<I>a</I><I>R</I>,12<I>b</I><I>S</I>)-3-methyl-2,4,
4<I>a</I>,7,7<I>a</I>,13-hexahydro-1<I>H</I>-4,12-methanobenzofuro[3,2-e]isoquinoline-7,9-dio
l
##
## > <PUBCHEM_IUPAC_NAME>
## (4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
##
## > <PUBCHEM_IUPAC_SYSTEMATIC_NAME>
## (4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
##
## > <PUBCHEM_IUPAC_TRADITIONAL_NAME>
## (4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
##
## > <PUBCHEM_IUPAC_INCHI>
## InChI=1S/C17H19NO3/c1-18-7-6-17-10-3-5-13(20)16(17)21-15-12(19)4-2-9(14(15)17)8-11(10)18/h
2-5,10-11,13,16,19-20H,6-8H2,1H3/t10-,11+,13-,16-,17-/m0/s1
##
## > <PUBCHEM_IUPAC_INCHIKEY>
## BQJCRHHNABKAKU-KBQPJGBKSA-N
##
## > <PUBCHEM_XLOGP3>
## 0.8
##
## > <PUBCHEM_EXACT_MASS>
## 285.136493
##
## > <PUBCHEM_MOLECULAR_FORMULA>
## C17H19NO3
##
## > <PUBCHEM_MOLECULAR_WEIGHT>
## 285.34
##
## > <PUBCHEM_OPENEYE_CAN_SMILES>
## CN1CCC23C4C1CC5=C2C(=C(C=C5)O)OC3C(C=C4)O
##
## > <PUBCHEM_OPENEYE_ISO_SMILES>
## CN1CC[C@]23[C@@H]4[C@H]1CC5=C2C(=C(C=C5)O)O[C@H]3[C@H](C=C4)O
##
## > <PUBCHEM_CACTVS_TPSA>
## 52.9
##
## > <PUBCHEM_MONOISOTOPIC_WEIGHT>
## 285.136493
##
## > <PUBCHEM_TOTAL_CHARGE>
## 0
##
## > <PUBCHEM_HEAVY_ATOM_COUNT>
## 21
##
```

```
## > <PUBCHEM_ATOM_DEF_STEREO_COUNT>
## 5
##
## > <PUBCHEM_ATOM_UDEF_STEREO_COUNT>
## 0
##
## > <PUBCHEM_BOND_DEF_STEREO_COUNT>
## 0
##
## > <PUBCHEM_BOND_UDEF_STEREO_COUNT>
## 0
##
## > <PUBCHEM_ISOTOPIC_ATOM_COUNT>
## 0
##
## > <PUBCHEM_COMPONENT_COUNT>
## 1
##
## > <PUBCHEM_CACTVS_TAUTO_COUNT>
## -1
##
## > <PUBCHEM_COORDINATE_TYPE>
## 1
## 5
## 255
##
## > <PUBCHEM_BONDANNOTATIONS>
## 10  13  8
## 10  16  8
## 13  19  8
## 16  20  8
## 19  21  8
## 14  2  6
## 20  21  8
## 5  9  5
## 6  22  5
## 7  23  6
## 8  24  5
##
## $$$$
```

`cat` is used in this example instead of `print` (which can be omitted) as it produces a nicer print-out for multiple-line strings.

Now the structure stored in the “mysdf” can be used in a PUG-REST request through HTTP-POST. For example, the code cell below shows how to retrieve various names (also called “synonyms”) of the input structure.

```
url <- paste(prolog, "/compound/sdf/synonyms/txt", sep="")
mydata <- list(sdf=mysdf)

res <- POST(url, body = mydata)
res$url
```

```
## [1] "https://pubchem.ncbi.nlm.nih.gov/rest/pug/compound/sdf/synonyms/txt"
```

```
cat(content(res,"text",encoding="UTF-8"))
```

```
## morphine
## Morphia
## Morphinum
## Morphium
## Morphina
## Morphin
## (-)-Morphine
## Duromorph
## MS Contin
## DepoDur
## Meconium
## Morphinism
## Moscontin
## Ospalivina
## Morfina
## l-Morphine
## Dulcontin
## Nepenthe
## Roxanol
## Kadian
## 57-27-2
## MORPHINE SULFATE
## Infumorph
## Dreamer
## Morpho
## Avinza
## Hocus
## Unkie
## Cube juice
## Hard stuff
## Oramorph SR
## Statex SR
## M-Eslon
## Ms Emma
## Morphin [German]
## Morfina [Italian]
## Duramorph
## Morphina [Italian]
## Morphine [BAN]
## Astramorph PF
## Duramorph PF
## CCRIS 5762
## Dolcontin
## HSDB 2134
## (5R,6S,9R,13S,14R)-4,5-Epoxy-N-methyl-7-morphinen-3,6-diol
## UNII-76I7G6D29C
## D-(-)-Morphine
## CHEBI:17303
## CHEMBL70
## EINECS 200-320-2
## 4,5alpha-Epoxy-17-methyl-7-morphinen-3,6alpha-diol
## 7,8-Didehydro-4,5-epoxy-17-methyl-morphinan-3,6-diol
## (7R,7AS,12BS)-3-METHYL-2,3,4,4A,7,7A-HEXAHYDRO-1H-4,12-METHANO[1]BENZOFURO[3,2-E]ISOQUINOL
INE-7,9-DIOL
## DEA No. 9300
## Morphine Anhydrate
## 76I7G6D29C
```

(5alpha,6alpha)-17-methyl-7,8-didehydro-4,5-epoxymorphinan-3,6-diol
Morphine (BAN)
Morphine Forte
RMS
Morphine H.P
(5alpha,6alpha)-Didehydro-4,5-epoxy-17-methylmorphinan-3,6-diol
Morphinan-3,6-alpha-diol, 7,8-didehydro-4,5-alpha-epoxy-17-methyl-
Morphine Extra-Forte
Morphinan-3,6-diol, 7,8-didehydro-4,5-epoxy-17-methyl-, (5alpha,6alpha)-
9H-9,9c-Iminoethanophenanthro(4,5-bcd)furan-3,5-diol, 4a,5,7a,8-tetrahydro-12-methyl-
methyl[?]diol
Aguettant
Dinamorf
Sevredol
Dimorf
MOI
Epimorph
Morphitec
Oramorph
Rescudose
Statex Drops
OMS Concentrate
RMS Uniserts
Roxanol UD
(Morphine)
Substitol (TN)
Mscontin, Oramorph
(4R,4aR,7S,7aR,12bS)-3-methyl-2,4,4a,7,7a,13-hexahydro-1H-4,12-methanobenzofuro[3,2-e]isoq
uinoline-7,9-diol
(-)-(etorphine)
MSIR
Roxanol 100
(-)Morphine sulfate
Morfina Dosa (TN)
SDZ202-250
NSC11441
SDZ 202-250
MS/L
MS/S
Epitope ID:116646
Morphinan-3,6-diol, 7,8-didehydro-4,5-epoxy-17-methyl- (5alpha,6alpha)-
SCHEMBL2997
M.O.S
BIDD:GT0147
GTPL1627
DTXSID9023336
Morphine 0.1 mg/ml in Methanol
Morphine 1.0 mg/ml in Methanol
BQJCRHHNABKAKU-KBQPJGBKSA-N
ZINC3812983
BDBM50000092
AKOS015966554
DB00295
AN-23579
AN-23737
LS-91748
C01516
D08233

```
## 7,8-Didehydro-4,5-epoxy-17-methylmorphinan-3,6-diol
## UNII-1M5VY6ITRT component BQJCRHHNABKAKU-KBQPJGBKSA-N
## 17-methyl-7,8-didehydro-4,5alpha-epoxymorphinan-3,6alpha-diol
## 7,8-Didehydro-4,5-epoxy-17-methylmorphinan-3,6-diol(morphine)
## (5A,6A)-7,8-DIDEHYDRO-4,5-EPOXY-17-METHYLMORPHINIAN-3,6-DIOL
## (5alpha,6alpha)-7,8-Didehydro-4,5-epoxy-17-methylmorphinan-3,6-diol
## (5alpha,6beta)-17-methyl-7,8-didehydro-4,5-epoxymorphinan-3,6-diol
## 3-(4-Hydroxy-phenyl)-1-propyl-piperidine-3-carboxylic acid ethyl ester
## 6-tert-Butyl-3-methyl-1,2,3,4,5,6-hexahydro-2,6-methano-benzo[d]azocine
## (-)(5.alpha.,6.alpha.)-7,8-Didehydro-4,5-epoxy-17-methylmorphinan-3,6-diol
## Morphinan-3,6-diol, 7,8-didehydro-4,5-epoxy-17-methyl- (5..alpha.,6.alpha.)-
## Morphine solution, 1.0 mg/mL in methanol, ampule of 1 mL, certified reference material
## (1S,5R,13R,14S)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(18),8,10,15-tetr
aene-10,14-diol
## (1S,5R,13R,14S,17R)-4-methyl-12-oxa-4-azapentacyclo[9.6.1.0;{1,13}.0;{5,17}.0;{7,18}]octad
eca-7(18),8,10,15-tetraene-10,14-diol
## (1S,5R,13R,14S,17R)-4-methyl-12-oxa-4-azapentacyclo[9.6.1.0^{1,13}.0^{5,17}.0^{7,18}]octad
eca-7,9,11(18),15-tetraene-10,14-diol
## (morphine) 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]oct
adeca-7(18),8,10,15-tetraene-10,14-diol
## 2-{4-[2,4-diamino-6-pteridinylmethyl(methyl)amino]phenylcarboxamido}pentanedioic acid(morp
hine)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol ; HydroChloride
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol ;sulphate salt(morphine)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol((Morphine))
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol(morphine sulfate)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol(morphine)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol(Morphine)(HCl)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diol,sulfate(Morphinesulfate)
## 4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(1
8),8,10,15-tetraene-10,14-diolMorphine
## 4-methyl-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(18),8,10,15-tetraene-1
0,14-diol
## 4-methyl-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(18),8,10,15-tetraene-1
0,14-diol (morphine)
## 4-methyl-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octadeca-7(18),8,10,15-tetraene-1
0,14-diol(Morphine)
## 6,11-Dimethyl-3-(3-methyl-but-2-enyl)-1,2,3,4,5,6-hexahydro-2,6-methano-benzo[d]azocin-8-o
l(Morphine)
## 9H-9,9c-Iminoethanophenanthro(4,5-bcd)furan-3,5-diol, 4alpha,5,7alpha,8-tetrahydro-12-meth
yl-
## MORPHINE, (5A,6A)-7,8-DIDEHYDRO-4,5-EPOXY-17-METHYLMORPHINIAN-3,6-DIOL, MORPHIUM, MORPHIA,
DOLCONTIN, DUROMORPH, MORPHINA, NEPENTHE
## Morphine;4-methyl-(1S,5R,13R,14S,17R)-12-oxa-4-azapentacyclo[9.6.1.01,13.05,17.07,18]octad
eca-7(18),8,10,15-tetraene-10,14-diol
```

Exercise 2a: Retrieve (in the **CSV** format) the XlogP, molecular weight, hydrogen bond donor count, hydrogen bond aceptor count, and TPSA of the compounds contained in the five sdf files (link to the data files).

- Use a for loop to retrieve the data for each compound.
- Remember to add some sleep time (e.g 0.5 seconds) after retrieving the data for each compound.
- Refer to the **lecture 1** notebook to see how to merge the multiple CSV outputs into a single data frame.

```
files <- c('l02_ex2b_compound1.sdf','l02_ex2b_compound2.sdf','l02_ex2b_compound3.sdf',
         'l02_ex2b_compound4.sdf','l02_ex2b_compound5.sdf')

# Write your code here
```