DATA COMPRESSION OF INCHIKEYS AND 2D COORDINATES

John Mayfield
NextMove Software Ltd

“Everything should be made as small as possible, but no smaller”

PART 1

INCHIKEY COMPRESSION

MOLECULE FOOTPRINT

How much space does a molecule take?

Representation	ChEML 19 (1.4M)	ChEMBL 28 (2M)	Enamine (1.2B)	Zinc20 (1.4B)
256-bit FP	32	32	32	32
1024-bit FP	128	128	128	128
SMILES	~56	~53	~46	~49
InChI	~156	~160		
Arthor (NextMove)	~174	~181	~104	~110
SDFile	~2,172	~2,702		
OEChem (OpenEye)	~8,500			
JChem (ChemAxon)	~11,400			
CDK	~11,700			
OpenBabel	~16,000			
RDKit	~20,600			

Toolkit numbers measure end of 2014

RECORD IDENTIFIERS

Often we want to associate an identifier with indexed chemical data

How much space does an identifier take?

Database	Example	Size (bytes)
CHEMBL	`CHEMBL409812`	**~12.6 (7-13)**
ZINC	`ZINC000256067317`	**16**
Enamine REAL	`Z3513844028` `s_2714____13153100____862376`	**~13.6 (9-44)**
InChIKey	`InChIKey=RYYVLZVUVIJVGH-UHFFFAOYSA-N`	**37**

INDEXING MODES

Arthor currently supports **four** indexing modes

```
-p pointer to source record (preferred)
-t title string
-i integer identifier
-l line num
```

A customer wanted to index their warehouse and join to external sources with **InChIKey**

- supported via **-t** taking 37 bytes (ouch)

PREFIX REMOVAL

An obvious saving is to remove any **fixed** prefixes (and add it back on later):

- “ChEMBL” **6 bytes**, “ZINC” **4 bytes**, “InChIKey=” **9 bytes**
- **4 bytes** on each record in Zinc20 adds **~5.7GB**
- for 10 billion records the **InChIKey=** prefix would take **90GB**

PREFIX REMOVAL

An obvious saving is to remove any **fixed** prefixes (and add it back on later):

- "ChEMBL" **6 bytes**, "ZINC" **4 bytes**, "InChIKey=" **9 bytes**
- **4 bytes** on each record in Zinc20 adds **~5.7GB**
- for 10 billion records the **InChIKey=** prefix would take **90GB**

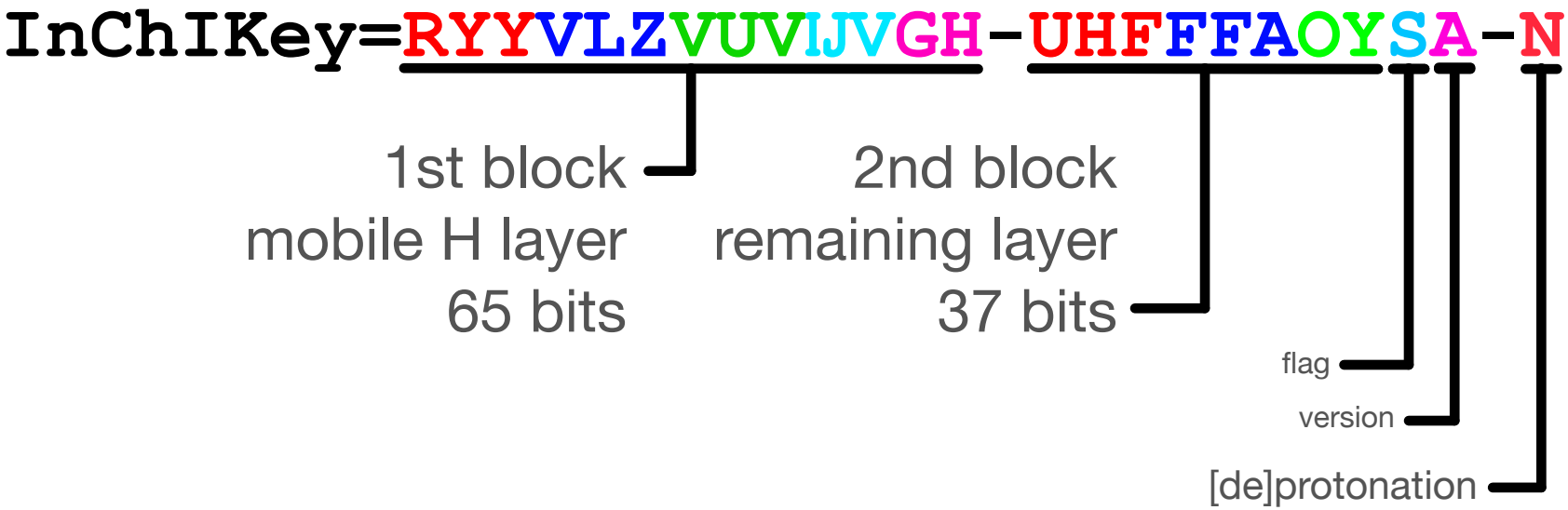
Can do even better by storing the numeric part as binary (**-i** option):

- ZINC000263614691 (**16 bytes**) -> 263,614,691 (**4 bytes**)
- **17 GB** saved on Zinc20

WHAT CAN WE DO WITH INCHIKEYS?

"This encoding is just a representation issue. In fact, ***the same hash may be represented by letters, digits, letters and digits and even with bare 0s and 1s*** *(as actually represented internally, in computer memory). However, representation issues may appear critical for applications like publishing or Web search."* ***- InChI Technical Manual***

STRUCTURE OF AN INCHIKEY



DECODING AN INCHIKEY

InChIKey=RYYVLZVUVIJVGH-UHFFFAOYSA-N

```
unsigned int decode3(const char *ptr)
{
  unsigned int x = 676 * (ptr[0]-'A') +
                   26  * (ptr[1]-'A') +
                         (ptr[2]-'A');
  if (x >= 12844) // 19*26*26
    x -= 516; // (19*26)+22 skipping TAA ... TTV
  if (x >= 2704) // 4*26*26
    x -= 676; // 26*26 skipping EAA ... EZZ
  return x;
}

unsigned int decode2(const char *ptr) {
  return 26 * (ptr[0]-'A') + (ptr[1]-'A');
}
```

Note: the actual InChI code is table driven but not needed

First Block

```
RYY = 11464 = 00101100 11001000
VLZ = 13315 = 00110100 00000011
VUV = 13545 = 00110100 11101001
IJV = 4987  = 00010011 01111011
GH  = 163   = 00000000 10100011
```

Second Block

```
UHF = 12515 = 00110000 11100011
FFA = 2834  = 00001011 00010010
OY  = 388   = 00000001 10000100
```

DECODING AN INCHIKEY

InChIKey=RYYVLZVUVIJVGH-UHFFFAOYSA-N

65-bits
(9 bytes)

11001000 11101100 00000000 10011101
01001110 11101111 01001101 10100011
00000000

37-bits
(5 bytes)

11100011 10110000 11000100 01000010
00011000

5-bits
(1 byte)

00001101

“uf-a-oy-sa”

InChIKey=RYYVLZVUVIJVGH-UHFFFAOYSA-N

Second block not a **uniform distribution**!

UniChem shows **~68%** of InChIKeys have the **“empty”** hash

We can exploit this and indicate this default with **a single bit**

Second Block	Count	% Frequency
UHFFFAOYSA-N	119876059	67.97
UHFFFAOYSA-O	2587031	1.47
UHFFFAOYSA-M	1276672	0.72
AWEZNQCLSA-N	726157	0.41
ZDUSSCGKSA-N	717550	0.41
HNNXBMFYSA-N	696602	0.39
CQSZACIVSA-N	689985	0.39
CYBMUJFWSA-N	680713	0.39
LBPRGKRZSA-N	671004	0.38
OAHLLOKOSA-N	661297	0.37

PACKING AN INCHIKEY

InChIKey=RYYVLZVUVIJVGH-UHFFFAOYSA-N

9-14 bytes

65-bits
first block

1-bit
"UHFFFAOYSA"

5-bits
[de]protonation

```
11001000 11101100 00000000 10011101
01001110 11101111 01001101 10100011
00000000
000000 1
 01101 00
```

37-bits
second block

1-bit
non-std flag

```
                 11100011 10110000 11000100
01000010 000 11000
         00 1 00000
```

HOW DOES IT DO?

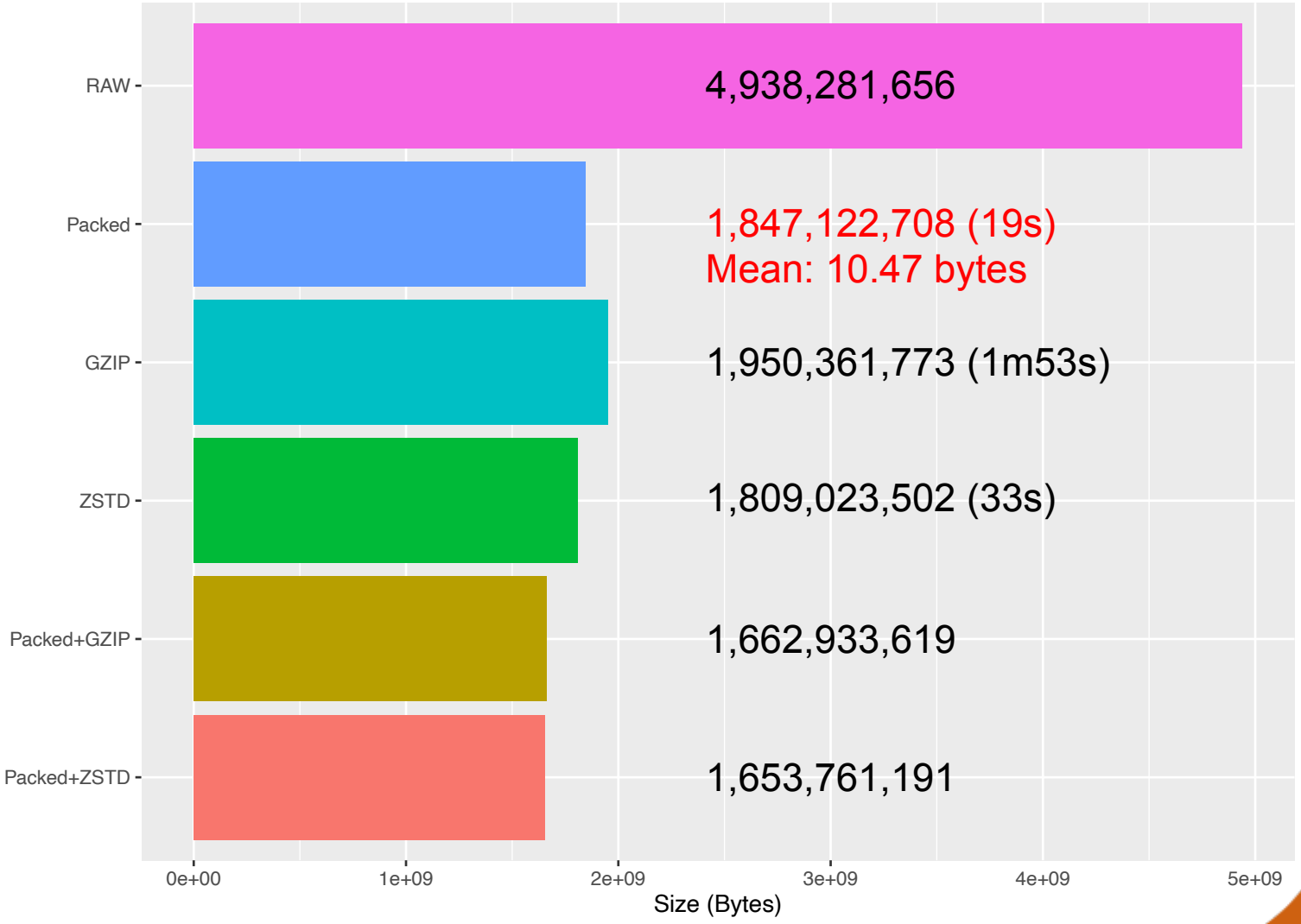
176 million keys from UniChem
- 27 bytes+newline
- about ~**4.8 GB**

Packed representation is on average **10.47 bytes** a total **~1.8 GB**

- Can be partition in to two sets of **9** and **14** bytes
- On disk binary search possible on these sets!

HOW DOES IT DO?

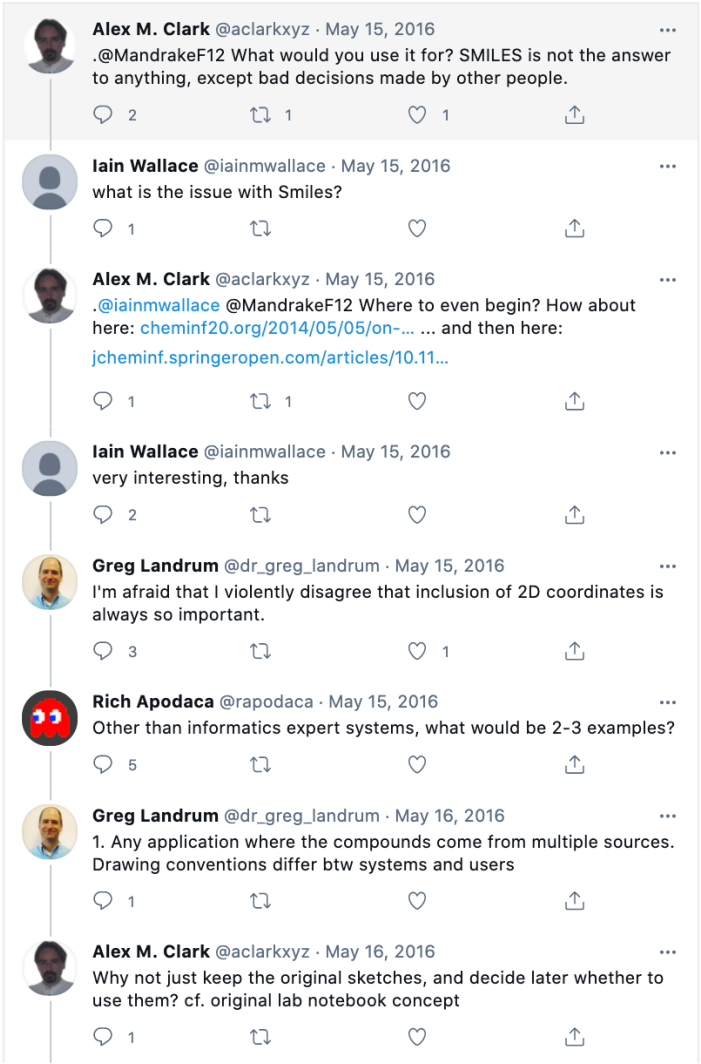
176 million keys from UniChem



PART 2

2D COORDINATE COMPRESSION

WHO CARES ABOUT COORDINATES?



Alex M. Clark @aclarkxyz · May 15, 2016
.@MandrakeF12 What would you use it for? SMILES is not the answer to anything, except bad decisions made by other people.
2 1 1

Iain Wallace @iainmwallace · May 15, 2016
what is the issue with Smiles?
1

Alex M. Clark @aclarkxyz · May 15, 2016
.@iainmwallace @MandrakeF12 Where to even begin? How about here: cheminf20.org/2014/05/05/on-... ... and then here: jcheminf.springeropen.com/articles/10.11...
1 1

Iain Wallace @iainmwallace · May 15, 2016
very interesting, thanks
2

Greg Landrum @dr_greg_landrum · May 15, 2016
I'm afraid that I violently disagree that inclusion of 2D coordinates is always so important.
3 1

Rich Apodaca @rapodaca · May 15, 2016
Other than informatics expert systems, what would be 2-3 examples?
5

Greg Landrum @dr_greg_landrum · May 16, 2016
1. Any application where the compounds come from multiple sources. Drawing conventions differ btw systems and users
1

Alex M. Clark @aclarkxyz · May 16, 2016
Why not just keep the original sketches, and decide later whether to use them? cf. original lab notebook concept
1

A chemical structure representation usually starts by being drawn in a sketcher by a chemist…

Well except for:
- **text-mining**
- **virtual library enumeration**
- **deep learning hit generation**
- **computer assisted structure elucidation**

- Useful to capture stylistic conventions
- Faster but it depends

Many databases will re-generate coords:
- PubChem Compound vs Substance

COORDINATES IN THOR/MERLIN

Figure 1: Previosly, the 2D datatype was incorporated into the SMI2GIF and allows the user to depict structures in a familair layout. The user can control whether the layout is determined by the Daylight depiction algorithm (left) or 2D data (right).

Two Wrongs Make it Look Right! Use of 2D data (without bond type information) is dangerous! By reversing 2D coordinates of geminal substituients, the original depiction (left) will show an inverted stereocenter. Additionally, if the stereocenter is inverted in the SMILES string, the depiction will appear to be the original structure (right).

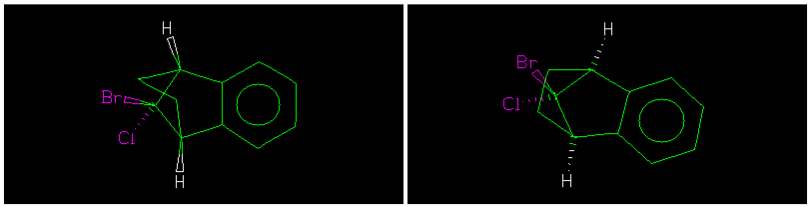
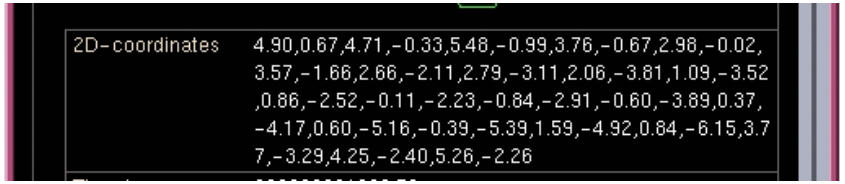


Figure 2: By reversing 2D coordinates of geminal substituients, the original depiction will show an inverted stereocenter. Additionally, if the stereocenter is inverted in the SMILES string (right), the depiction will appear to be the original structure.

Array of tuples in Thor-Data



https://www.daylight.com/meetings/mug02/Kappler/smi2d/smi2d.html

COORDINATES IN CXSMILES AND INCHI

CXSMILES:

```
C1=CC(C=C2[C@]1([C@@]3([C@](CC2)([C@]4([C@](C[C@@]3(C)[H])([C@@]([C@@](C4)(C)[H])(O)C(CO)=O)C)[H])
[H])F)C)=O |
(-6.31,3.5,;-7.02,3.09,;-7.02,2.26,;-6.31,1.85,;-5.59,2.26,;-5.59,3.09,;-4.88,3.5,;-4.16,3.09,;-4.16
,2.26,;-4.88,1.85,;-3.45,3.5,;-3.45,4.33,;-4.16,4.74,;-4.88,4.33,;-5.59,4.74,;-4.88,5.15,;-2.66,4.58
,;-2.18,3.91,;-2.66,3.25,;-1.4,4.17,;-1.49,3.46,;-1.91,4.92,;-2.79,5.4,;-3.35,6.07,;-4.24,5.83,;-2.0
7,5.81,;-3.45,5.15,;-3.24,2.7,;-4.12,3.91,;-4.88,2.68,;-5.59,3.91,;-7.74,1.85,)|
```

InChI AuxInfo:

```
InChI=1S/C23H31FO4/
c1-13-9-18-17-6-5-15-10-16(26)7-8-20(15,3)22(17,24)14(2)11-21(18,4)23(13,28)19(27)12-25/
h7-8,10,13-14,17-18,25,28H,5-6,9,11-12H2,1-4H3/t13-,14+,17-,18+,20+,21+,22-,23+/m1/s1
AuxInfo=1/0/N:27,18,13,31,9,10,2,1,19,4,14,24,20,15,5,3,11,16,23,6,17,8,21,12,26,7,25,22/it:im/
rA:32CCCCCC.eOC.eCCC.oFCCC.oC.eC.eCCC.oC.eOCCOOCHHHCH/
rB:d1;s2;s3;d4;s1s5;d3;s6;s5;s9;s8s10;N8;P6;;s8s14;s11;s14s16;P15;s16;s19;s17s20;N21;P21;s23;d23;s24
;N20;N16;N11;N15;P17;P20;/
rC:-6.3073,3.5009,0;-7.0218,3.0884,0;-7.0218,2.2633,0;-6.3073,1.8508,0;-5.5929,2.2633,0;-5.5929,3.08
84,0;-7.7363,1.8508,0;-4.8784,3.5009,0;-4.8784,1.8508,0;-4.1639,2.2633,0;-4.1639,3.0884,0;-4.8784,2.
6759,0;-5.5929,3.9134,0;-4.1639,4.7384,0;-4.8784,4.3259,0;-3.4495,3.5009,0;-3.4495,4.3259,0;-5.5929,
4.7384,0;-2.6649,3.2459,0;-2.18,3.9133,0;-2.6648,4.5808,0;-1.9112,4.9163,0;-2.7939,5.3956,0;-3.3523,
6.0683,0;-2.0735,5.813,0;-4.2367,5.8313,0;-1.3953,4.1683,0;-3.236,2.704,0;-4.1208,3.9122,0;-4.8784,5
.1509,0;-3.4495,5.1509,0;-1.4881,3.464,0;
```

COORDINATES IN CXSMILES AND INCHI

CXSMILES:

```
C1=CC(C=C2[C@]1([C@@]3([C@](CC2)([C@]4([C@](C[C@@]3(C)[H])([C@@]([C@@](C4)(C)[H])(O)C(CO)=O)C)[H])
[H])F)C)=O |
(-6.31,3.5,;-7.02,3.09,;-7.02,2.26,;-6.31,1.85,;-5.59,2.26,;-5.59,3.09,;-4.88,3.5,;-4.16,3.09,;-4.16
,2.26,;-4.88,1.85,;-3.45,3.5,;-3.45,4.33,;-4.16,4.74,;-4.88,4.33,;-5.59,4.74,;-4.88,5.15,;-2.66,4.58
,;-2.18,3.91,;-2.66,3.25,;-1.4,4.17,;-1.49,3.46,;-1.91,4.92,;-2.79,5.4,;-3.35,6.07,;-4.24,5.83,;-2.0
7,5.81,;-3.45,5.15,;-3.24,2.7,;-4.12,3.91,;-4.88,2.68,;-5.59,3.91,;-7.74,1.85,)|
```

InChI AuxInfo:

```
InChI=1S/C23H31FO4/
c1-13-9-18-17-6-5-15-10-16(26)7-8-20(15,3)22(17,24)14(2)11-21(18,4)23(13,28)19(27)12-25/
h7-8,10,13-14,17-18,25,28H,5-6,9,11-12H2,1-4H3/t13-,14+,17-,18+,20+,21+,22-,23+/m1/s1
AuxInfo=1/0/N:27,18,13,31,9,10,2,1,19,4,14,24,20,15,5,3,11,16,23,6,17,8,21,12,26,7,25,22/it:im/
rA:32CCCCCC.eOC.eCCC.oFCCC.oC.eC.eCCC.oC.eOCCOOCHHHCH/
rB:d1;s2;s3;d4;s1s5;d3;s6;s5;s9;s8s10;N8;P6;;s8s14;s11;s14s16;P15;s16;s19;s17s20;N21;P21;s23;d23;s24
;N20;N16;N11;N15;P17;P20;/
rC:-6.3073,3.5009,0;-7.0218,3.0884,0;-7.0218,2.2633,0;-6.3073,1.8508,0;-5.5929,2.2633,0;-5.5929,3.08
84,0;-7.7363,1.8508,0;-4.8784,3.5009,0;-4.8784,1.8508,0;-4.1639,2.2633,0;-4.1639,3.0884,0;-4.8784,2.
6759,0;-5.5929,3.9134,0;-4.1639,4.7384,0;-4.8784,4.3259,0;-3.4495,3.5009,0;-3.4495,4.3259,0;-5.5929,
4.7384,0;-2.6649,3.2459,0;-2.18,3.9133,0;-2.6648,4.5808,0;-1.9112,4.9163,0;-2.7939,5.3956,0;-3.3523,
6.0683,0;-2.0735,5.813,0;-4.2367,5.8313,0;-1.3953,4.1683,0;-3.236,2.704,0;-4.1208,3.9122,0;-4.8784,5
.1509,0;-3.4495,5.1509,0;-1.4881,3.464,0;
```

INSPIRATION

Confirm that we can access the metadata when we read the file back in:

In [7]:
```
nimg = Image.open('/tmp/blah.png')
nimg.text
```

Out[7]:
```
{'RDKit_SMILES': 'COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC |
(6.46024,1.03002,;5.30621,1.98825,;3.89934,1.46795,;2.74531,2.42618,;1.33844,1.90588,;1.0856,0.427343,;
-0.228013,-0.296833,;0.1857,-1.73865,;-0.683614,-2.96106,;-2.18134,-3.04357,;-2.75685,-4.42878,;-4.2442
2,-4.62298,;-3.17967,-1.92404,;-4.62149,-2.33775,;-2.92683,-0.445502,;-1.61322,0.278673,;-2.02693,1.720
49,;-3.50547,1.97333,;-4.02577,3.3802,;-5.50431,3.63304,;-3.06754,4.53423,;-1.15762,2.9429,;0.340111,3.
02541,;2.23963,-0.530891,;1.98679,-2.00943,;3.14082,-2.96766,;3.6465,-0.0105878,;4.80053,-0.968822,;4.5
4769,-2.44736,)|'}
```

Easy wins: less decimal places, remove leading zeros
Maybe we can do better?

http://rdkit.blogspot.com/2020/07/adding-molecular-metadata-to-pngs.html

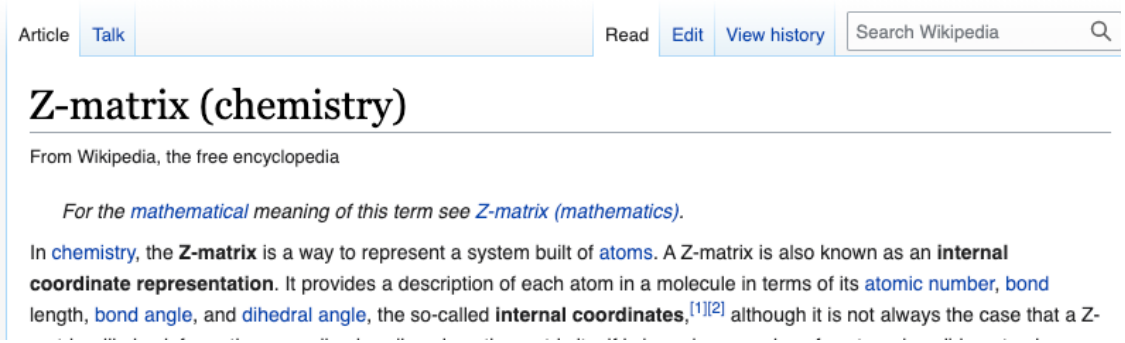
INSPIRATION

Confirm that we can access the metadata when we read the file back in:

In [7]:
```
nimg = Image.open('/tmp/blah.png')
nimg.text
```

Out[7]:
```
{'RDKit_SMILES': 'COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC |
(6.46024,1.03002,;5.30621,1.98825,;3.89934,1.46795,;2.74531,2.42618,;1.33844,1.90588,;1.0856,0.427343,;
-0.228013,-0.296833,;0.1857,-1.73865,;-0.683614,-2.96106,;-2.18134,-3.04357,;-2.75685,-4.42878,;-4.2442
2,-4.62298,;-3.17967,-1.92404,;-4.62149,-2.33775,;-2.92683,-0.445502,;-1.61322,0.278673,;-2.02693,1.720
49,;-3.50547,1.97333,;-4.02577,3.3802,;-5.50431,3.63304,;-3.06754,4.53423,;-1.15762,2.9429,;0.340111,3.
02541,;2.23963,-0.530891,;1.98679,-2.00943,;3.14082,-2.96766,;3.6465,-0.0105878,;4.80053,-0.968822,;4.5
4769,-2.44736,)|'}
```

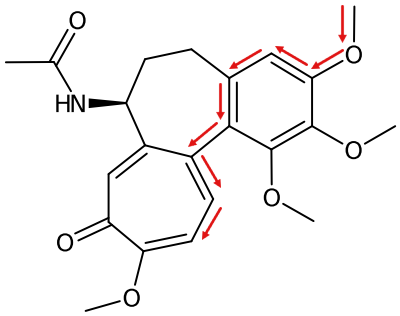
Easy wins: less decimal places, remove leading zeros
Maybe we can do better?



http://rdkit.blogspot.com/2020/07/adding-molecular-metadata-to-pngs.html
https://en.wikipedia.org/wiki/Z-matrix_(chemistry)

INTERNAL TRAVERSAL COORDINATES

Both SMILES and InChI are a graph traversal



Unit Bond Lengths

ChemDraw: 0.825

Marvin: 0.825

BIOVIA: 1

OEChem: 1

MOE: 1.2

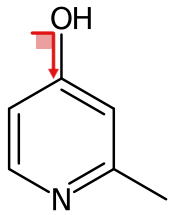
CDK: 1.5

RDKit: 1.5

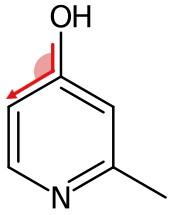
For each bond in the traversal we can encode a **rotation** and optionally a **length** scale from the unit bond length

```
r<num> : rotate
s<num> : scale (relative)
```

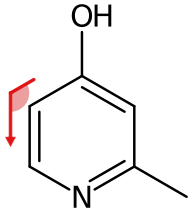
EXAMPLE

1. 

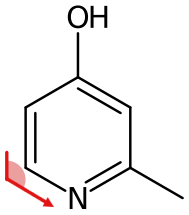
OC=1C=CN=C(C)C1
r−90

2. 

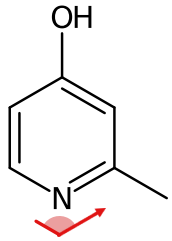
OC=1C=CN=C(C)C1
r−120

3. 

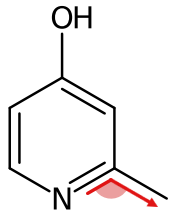
OC=1C=CN=C(C)C1
r120

4. 

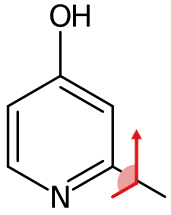
OC=1C=CN=C(C)C1
r120

5. 

OC=1C=CN=C(C)C1
r120

6. 

OC=1C=CN=C(C)C1
r−120

7. 

OC=1C=CN=C(C)C1
r120

r−90r−120r120r120r120r−120r120

REALITY

OC=1C=CN=C(C)C1 <r-90r-120r120r120r120r-120r120>

…encode the acute angles

OC=1C=CN=C(C)C1 <r-90r-60r60r60r60r-60r60>

…scale to avoid -ve

OC=1C=CN=C(C)C1 <r90r120r240r240r240r120r240>

…quantised radians with accuracy config

OC=1C=CN=C(C)C1 <r78r104r209r209r209r104r209>

HOW ARE WE DOING?

```
AuxInfo=1/…/rC:0,3.0783,0;0,1.5392,0;-1.3339,.7696,0;
                -1.3339,-.7696,0;0,-1.5392,0;1.3339,-.7696,0;
                 2.668,-1.5392,0;1.3339,.7696,0;

OC=1C=CN=C(C)C1 |(0,3,;,1.5,;-1.3,0.75,;-1.3,-0.75,;
                  ,-1.5,;1.3,-0.75,;2.6,-1.5,;1.3,0.75,)|

OC=1C=CN=C(C)C1 <r78r104r209r209r209r104r209>
```

HOW ARE WE DOING?

```
AuxInfo=1/…/rC:0,3.0783,0;0,1.5392,0;-1.3339,.7696,0;
                -1.3339,-.7696,0;0,-1.5392,0;1.3339,-.7696,0;
                 2.668,-1.5392,0;1.3339,.7696,0;

OC=1C=CN=C(C)C1 |(0,3,;,1.5,;-1.3,0.75,;-1.3,-0.75,;
                  ,-1.5,;1.3,-0.75,;2.6,-1.5,;1.3,0.75,)|

OC=1C=CN=C(C)C1 <r78r104r209r209r209r104r209>
```

Common rotations can be replaced with a single char

```
static final double[] ANGLE_DB = new double[]{
    0,                                  // a 0
    +Math.PI * (1d / 6d),               // b 30
    -Math.PI * (1d / 6d),               // c 30
    +Math.PI - (5 * Math.PI / 7),       // d ~51.42 (reg heptagon)
    -(Math.PI - (5 * Math.PI / 7)),     // e ~51.42 (reg heptagon)
    +Math.PI * (2d / 6d),               // f 60
    -Math.PI * (2d / 6d),               // g 60
    +Math.PI - (3 * Math.PI / 5),       // h 72 (reg pentagon)
    -(Math.PI - (3 * Math.PI / 5)),     // i 72 (reg pentagon)
    +Math.PI * (3d / 6d),               // j 90
    -Math.PI * (3d / 6d),               // k 90
    +Math.PI * (4d / 6d),               // l 120
    -Math.PI * (4d / 6d),               // m 120
    +Math.PI * (5d / 6d),               // n 150
    -Math.PI * (5d / 6d),               // o 150
    +Math.PI                            // p 180
};
```

HOW ARE WE DOING?

```
AuxInfo=1/…/rC:0,3.0783,0;0,1.5392,0;-1.3339,.7696,0;
              -1.3339,-.7696,0;0,-1.5392,0;1.3339,-.7696,0;
               2.668,-1.5392,0;1.3339,.7696,0;

OC=1C=CN=C(C)C1 |(0,3,;,1.5,;-1.3,0.75,;-1.3,-0.75,;
                  ,-1.5,;1.3,-0.75,;2.6,-1.5,;1.3,0.75,)|

OC=1C=CN=C(C)C1 <r78r104r209r209r209r104r209>

   r78  (-90) => k
   r104 (-60) => g
   r209 (+60) => f

OC=1C=CN=C(C)C1 <kgfffgf>
```

“xysmi”

HOW ARE WE DOING?

```
AuxInfo=1/…/rC:0,3.0783,0;0,1.5392,0;-1.3339,.7696,0;
             -1.3339,-.7696,0;0,-1.5392,0;1.3339,-.7696,0;
              2.668,-1.5392,0;1.3339,.7696,0;

OC=1C=CN=C(C)C1 |(0,3,;,1.5,;-1.3,0.75,;-1.3,-0.75,;
                  ,-1.5,;1.3,-0.75,;2.6,-1.5,;1.3,0.75,)|

OC=1C=CN=C(C)C1 <r78r104r209r209r209r104r209>

   r78  (-90) => k
   r104 (-60) => g
   r209 (+60) => f

OC=1C=CN=C(C)C1 <kgfffgf>                          "xysmi"

   Multigram compression [1]
     ff  => :
     fgf => &

OC=1C=CN=C(C)C1 <kg:&>                             "xysmiz"
```

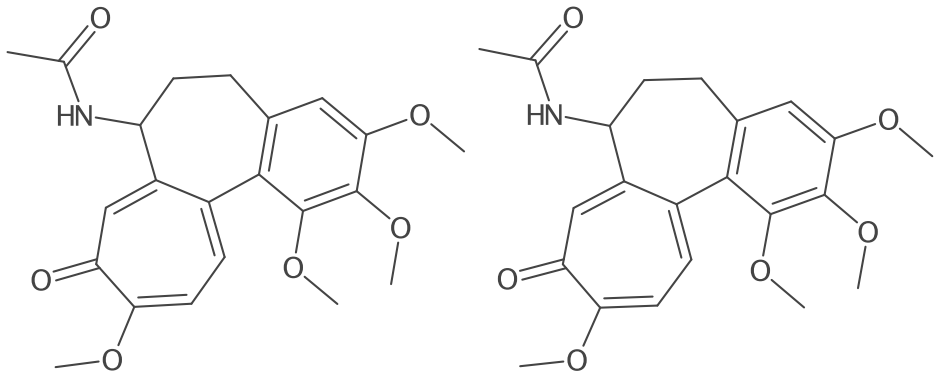
[1] https://www.daylight.com/meetings/mug01/Sayle/SmiZip/sld001.htm

GREG'S BLOG EXAMPLE

```
COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC |(6.46024,1.03002,
;5.30621,1.98825,;3.89934,1.46795,;2.74531,2.42618,;1.33844,1.90588,;1.085
6,0.427343,;-0.228013,-0.296833,;0.1857,-1.73865,;-0.683614,-2.96106,;-2.1
8134,-3.04357,;-2.75685,-4.42878,;-4.24422,-4.62298,;-3.17967,-1.92404,;-4
.62149,-2.33775,;-2.92683,-0.445502,;-1.61322,0.278673,;-2.02693,1.72049,;
-3.50547,1.97333,;-4.02577,3.3802,;-5.50431,3.63304,;-3.06754,4.53423,;-1.
15762,2.9429,;0.340111,3.02541,;2.23963,-0.530891,;1.98679,-2.00943,;3.140
82,-2.96766,;3.6465,-0.0105878,;4.80053,-0.968822,;4.54769,-2.44736,)|
```

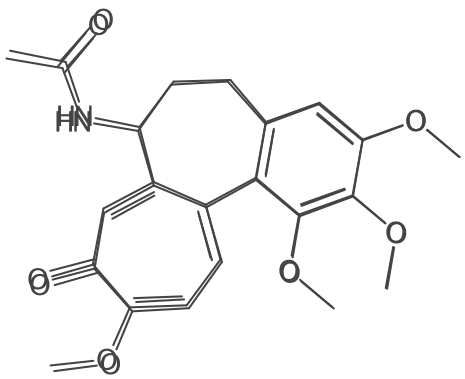
```
COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC
<r279fgffer224eer213ger213eer224r213gfgeefgffgg>
```

```
COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC
<+79F:E2+24E2E2+13aE2+13E2E2+24+13/E2E2&C>
```



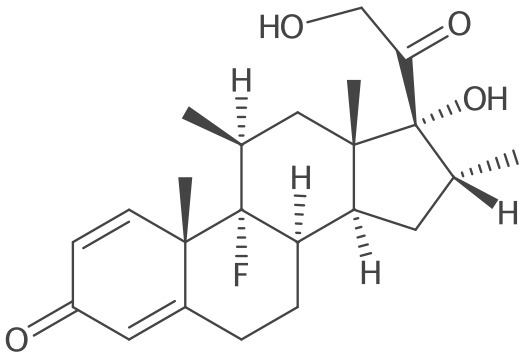
GREG'S BLOG EXAMPLE

```
COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC |(6.46024,1.03002,
;5.30621,1.98825,;3.89934,1.46795,;2.74531,2.42618,;1.33844,1.90588,;1.085
6,0.427343,;-0.228013,-0.296833,;0.1857,-1.73865,;-0.683614,-2.96106,;-2.1
8134,-3.04357,;-2.75685,-4.42878,;-4.24422,-4.62298,;-3.17967,-1.92404,;-4
.62149,-2.33775,;-2.92683,-0.445502,;-1.61322,0.278673,;-2.02693,1.72049,;
-3.50547,1.97333,;-4.02577,3.3802,;-5.50431,3.63304,;-3.06754,4.53423,;-1.
15762,2.9429,;0.340111,3.02541,;2.23963,-0.530891,;1.98679,-2.00943,;3.140
82,-2.96766,;3.6465,-0.0105878,;4.80053,-0.968822,;4.54769,-2.44736,)|

COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC
<r279fgffer224eer213ger213eer224r213gfgeefgffgg>

COc1cc2c(-c3ccc(OC)c(=O)cc3[C@@H](NC(C)=O)CC2)c(OC)c1OC
<+79F:E2+24E2E2+13aE2+13E2E2+24+13/E2E2&C>
```



RMSD: 0.10912

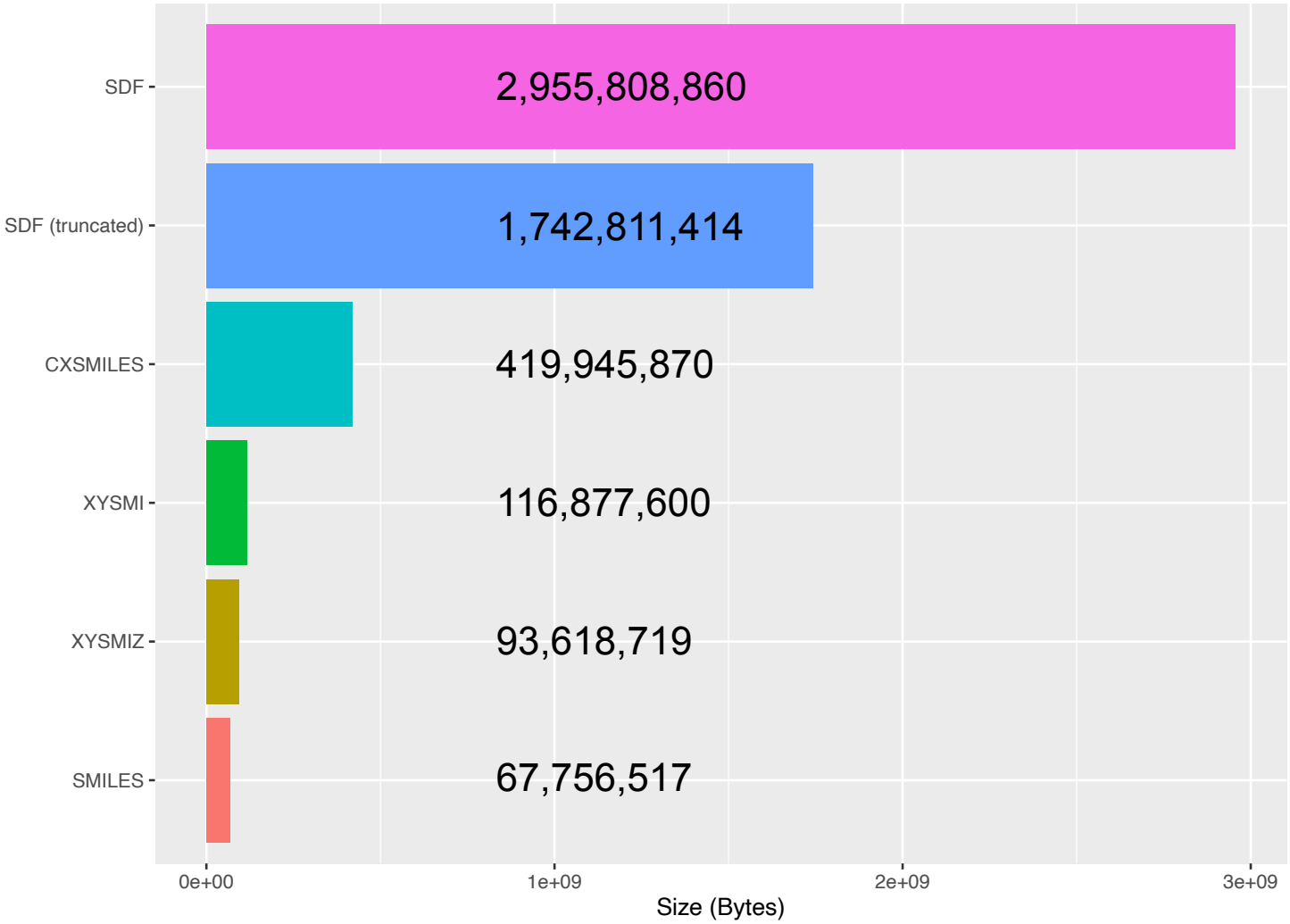
MICK'S MUG EXAMPLE




```
C1=CC(C=C2C1(C3(C(CC2)(C4(C(CC3(C)[H])(C(C(C4)(C)[H])(O)C(CO)=O)C)[H])
[H])F)C)=O |(-6.31,3.5,;-7.02,3.09,;-7.02,2.26,;-6.31,1.85,;-5.59,2.26,;
-5.59,3.09,;-4.88,3.5,;-4.16,3.09,;-4.16,2.26,;-4.88,1.85,;-3.45,3.5,;-3.4
5,4.33,;-4.16,4.74,;-4.88,4.33,;-5.59,4.74,;-4.88,5.15,;-2.66,4.58,;-2.18,
3.91,;-2.66,3.25,;-1.4,4.17,;-1.49,3.46,;-1.91,4.92,;-2.79,5.4,;-3.35,6.07
,;-4.24,5.83,;-2.07,5.81,;-3.45,5.15,;-3.24,2.7,;-4.12,3.91,;-4.88,2.68,;-
5.59,3.91,;-7.74,1.85,)|

C1=CC(C=C2[C@]1([C@@]3([C@](CC2)([C@]4([C@](C[C@@]3(C)[H])([C@@]([C@@](C4)
(C)[H])(O)C(CO)=O)C)[H])[H])F)C)=O
<offffggggr259fr115hr81r125hhr62r115fmgr68r133s112bs123r214s102r96r65mag>

C1=CC(C=C2[C@]1([C@@]3([C@](CC2)([C@]4([C@](C[C@@]3(C)[H])([C@@]([C@@](C4)
(C)[H])(O)C(CO)=O)C)[H])[H])F)C)=O
<z:I+59bldr81(25ddr62lbJar68(33O12mO23+14%2r96r65Jha>
```

COMPRESSION NUMBERS

(ChEMBL 28 100k random)



COMPRESSION NUMBERS

(ChEMBL 28 100k random)

```
C1=CN=CC2=C1C(=O)CCC2 <o-&:>    CHEMBL3585999
CCN1CCCC1CNC(=O)CC=2C(=NNC2C3=CC=C(C=C3)OC)C <r88bKe"&eGceL:e>  CHEMBL3439222
CC(C)SC1=NC2=C(C3=CC=CC=C3N2C)N=N1 <Q?:cy-ye:>  CHEMBL1736920
C1=CC(=CC=C1NC2=C(C=CC3=NC4(CCCCC4)N=C32)[N+](=O)[O-])[N+](=O)[O-] <%1(82ka!k"&:ld(62-dd"U>
CHEMBL401604
CC(C)NC(=O)C1CC2C(C1)OCCN2CC=3C=CSC3 <JbUKB;j>  CHEMBL3553703
CCOC(=O)C(C)(CC1=CC=CN=C1)C2=CC=NC=3N2N=CC3C4=CC=C(C=C4)Cl <gbTwqaC-#*e."Mk>    CHEMBL3260526
COC1=CC=C(C=C1S(=O)(=O)NC2=CC=CC(=C2)C3=CC=C(N=N3)N4CCCCCC4)F <gb,wqa:')'T,Q0%1x+02xx%1+02b>
CHEMBL1316736
C1=CC=C(C=C1)CN2CCC(CC2)CCC=3C4=CC=C(C=C4ON3)Br <+72#U)VfPda>   CHEMBL330004
CC1=C(C=NC=C1F)N2C(=O)C3=CC(=CC=C3N=C2C(C)NC4=C(C(=NC(=N4)N)N)C#N)F <mSD,a?,:hb>        CHEMBL3943088
C=1C=NN(C1)CCC(=O)N2CCN(CC2)CC(F)F <JGcebT$/>   CHEMBL3494549
C1=CC(=CC=C1CN2N=C(N=N2)C3=CC=C(C=C3)N)C(=O)NO <(20-:fjdH$?>    CHEMBL4574641
CC1=CC=C(C(=C1)Br)N2C(=CC(=CC3=CC=CC=C3)C2=O)C4=CC=CC=C4 <mSNcceI'ceeb->        CHEMBL4062449
C=CCON=CCOC1=CC=C(C=C1)CC2=CC=CC=C2 <(31&VR"->  CHEMBL2271778
CC1=CC=C(C=C1Cl)N2C(=O)C3=CC=CC=C3C(=N2)C(=O)O <z":,$?> CHEMBL1496634
CC(C)CC(C(=O)NC(CCCNC(=N)N)C(=O)NC(CCCNC(=N)N)C(=O)NC(CCC(=O)N)C(=O)NC(CCC(=O)O)C(=O)NC(CNC(=O)CCOP(=O)
(O)OP(=O)(O)OP(=O)
(O)OCC1C(C(C(O1)N2C=NC3=C2N=CN=C3N)O)O)C(=O)CNC(C(C)C)C(=O)NC(CC(=O)O)C(=O)NC(C)C(=O)NC(CC(C)C)C(=O)O)N
C(=O)C(C)NC(=O)C(CCCCN)NC(=O)C(CCCCN)NC(=O)CCCCC4C5C(CS4)NC(=O)N5 <r82C?TUD:?//T&&&CVC?
vp:wq)vp&Ke.fjd%4d.ks92a%3ks93aAeeaV?/V:U/?TU"TCSVU?E0j%3+17+21'r93r95Nr92>      CHEMBL392518
CN(C)C(=O)C1=C(C=CC(=N1)N2CCC(CC2)C=3C4=C(NN3)NC(=O)CC4(C(F)(F)F)O)C(F)(F)F <r303bT,,KD:p@vb@>
CHEMBL4462410
CC1=CC=C(O1)C(=O)N2CCC3=C(C=NC=C3CNC4=NC=CC=N4)C2.Cl <r5KeU)#T$>        CHEMBL3497479
CC1=C(C=CC(=C1C#N)F)C2CN3CCN(CC3CO2)C(=O)CC4=CC=C(N=C4)N5C=NN=N5 <z,h:?)/?$eGc> CHEMBL3943156
C1=CC(=CC=C1OCN2C=CC(=N2)C(=O)NCCN3CCOCC3)F <u-DE0ddHbV,>       CHEMBL3444995
CN(CC1=CN=C(S1)N2CCOCC2)CC3=NC(=NO3)C4=CC=CC=C4 <r73)KeS;Hb->   CHEMBL3472212
```

Conclusions

- Demonstrated specialised compression techniques for minimising data footprint of **InChIKeys** and **2D coordinates**
- **InChIKey** binary packing on GitHub https://github.com/johnmay/inchikey-compress
- More analysis (and possibly room for more optimisations) needed for the coordinate compression:
  - Imperfect layouts, CDK uses ChEBI ring templates
  - If you can generate an identical layout could avoid storing completely?
  - Angle quantisation improvements
- **Reaction Atom-Maps** is another example of other data that can benefit from specialised compression

Acknowledgements

NextMove Software Ltd:
Roger Sayle, Richard Gowers, Ingvar Lagerstedt