Wildcard
Pharmaceutical Consulting

Peptide and protein pI calculations using Rdkit

RDkit UGM 2016

Outline

- Part 1: Predicting pI of peptides and proteins
  - Isoelectric point and pI calculations
  - Assigning pKa values to peptides
  - Proteins: Pseudoatoms
  - Extended dictionary for modified AA and term caps
- Part 2: Preliminary results on a machine learning pKa predictor
  - pKa and linearity of Hammet Constants
  - Creating a dataset from Reaxys pKa data
  - Modelling Monoprotics and Multi site compounds
  - Future plans

Esben Jannik Bjerrum

- Ph.D (Computational Chemistry)
- Industry experience:
  - Drug Discovery IT Support and Databases, LEO Pharma A/S (DK)
- Postdoc #1
  - Protein NMR, Department of Biology, Copenhagen University (DK)
- Postdoc #2
  - Chemometrics and automatic PLS model tuning, Department of Food, Copenhagen University (DK)
- Founder Wildcard Pharmaceutical Consulting

Wildcard
Pharmaceutical Consulting

We connect research data with humans

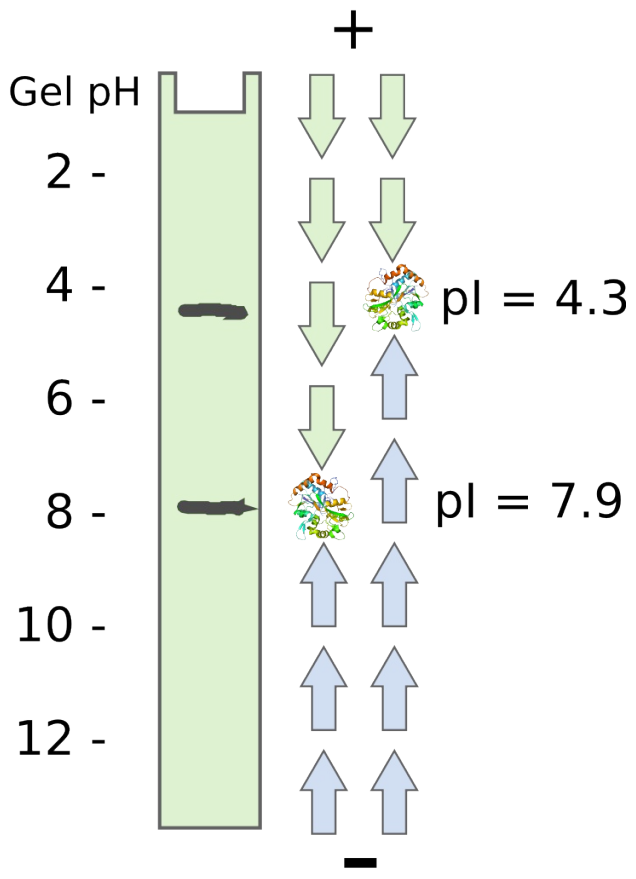
Computational Chemistry

Molecular Machine Learning

Research and Laboratory IT services

Part1: pI of peptides/proteins

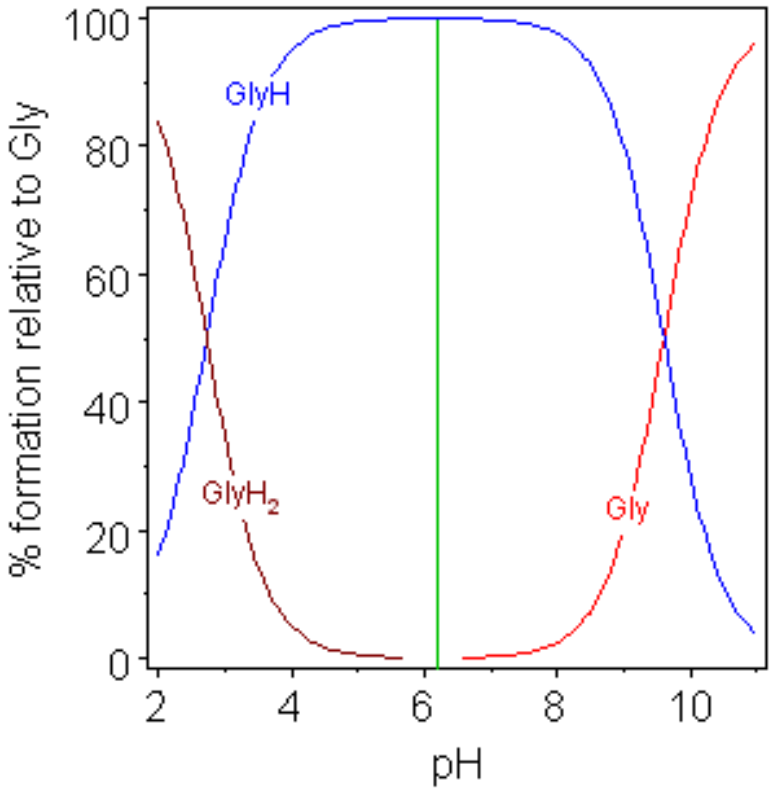
- Isoelectric point (pI, pH(I), IEP)
- The pH where the average charge on molecular ensemble is 0
- Must have both basic and acidic groups
- Can be measured experimentally by iso electric focusing in a pH gradient gel



pI calculation from pKa values

$$\mathrm{pI} = \frac{\mathrm{p}K_{\mathrm{a1}} + \mathrm{p}K_{\mathrm{a2}}}{2}$$

- Trivia: Bjerrum plot named after Niels Janniksen Bjerrum (1879-1958)



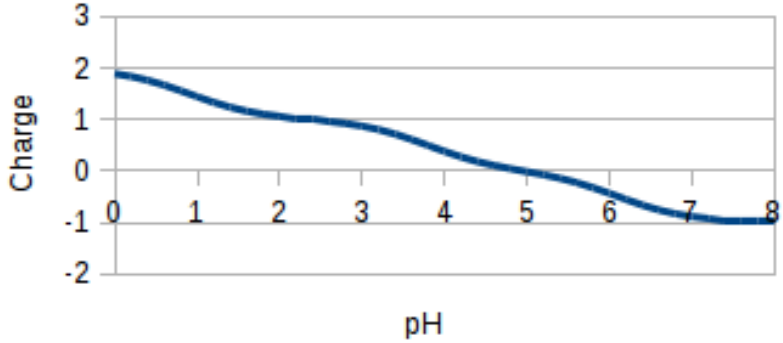
More complex model

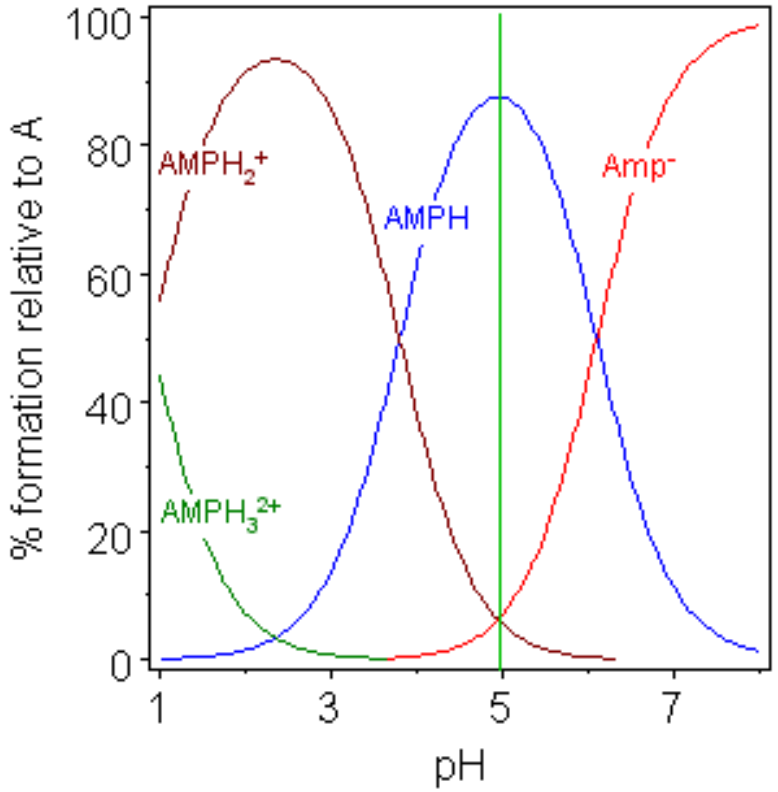
- Sum partial charge from individual contributions
- Identify pH where charge = 0
- Note negative signs for acidic groups

$$Charge = \frac{1}{1+10^{(pH-pKa_1)}} + \frac{-1}{1+10^{-(pH-pKa_2)}} + \ldots$$

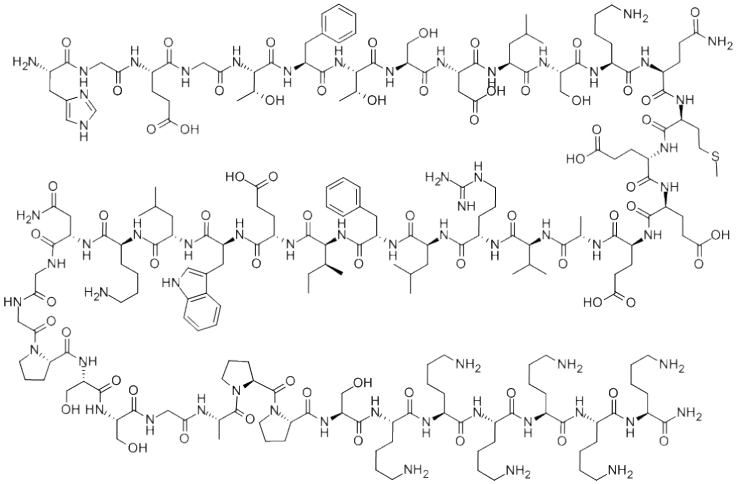
Multiprotic Example

- adenosine monophosphate pK = 0.9, 3.8, 6.1





From Molfile to pI



Lixenatide, GLP-1 receptor agonist

RDKit
Open-Source Cheminformatics and Machine Learning

List of pKa values

pI

Assignment of pKa values using dictionary

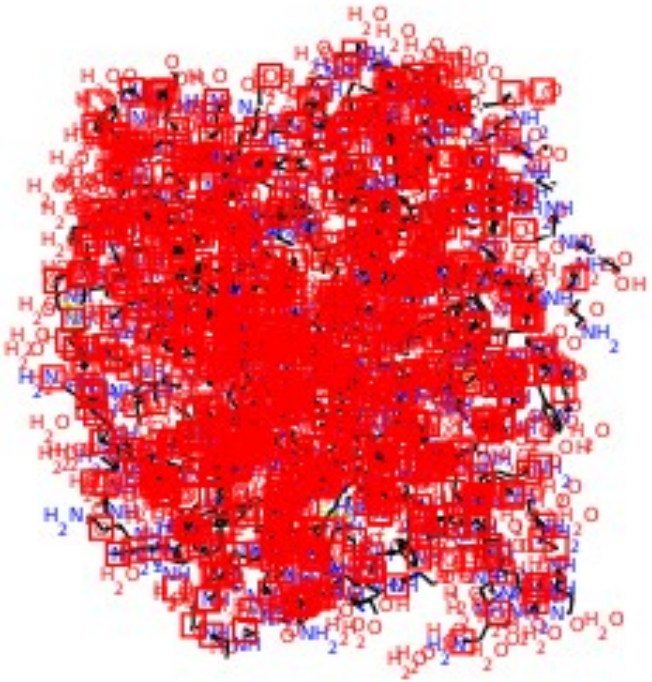
```
naturalAAdict = [('Tyrosine',
Chem.MolFromSmarts('[$
([O-,OH]c1ccccc1)]'),10.46 ,
0),

('Histidine1',
Chem.MolFromSmarts('[$
(cC)]1cnc[nH]1'), 6.04 ,1),

('Histidine2',
Chem.MolFromSmarts('[$
(cC)]1c[nH]cn1'), 6.04, 1),

...
```

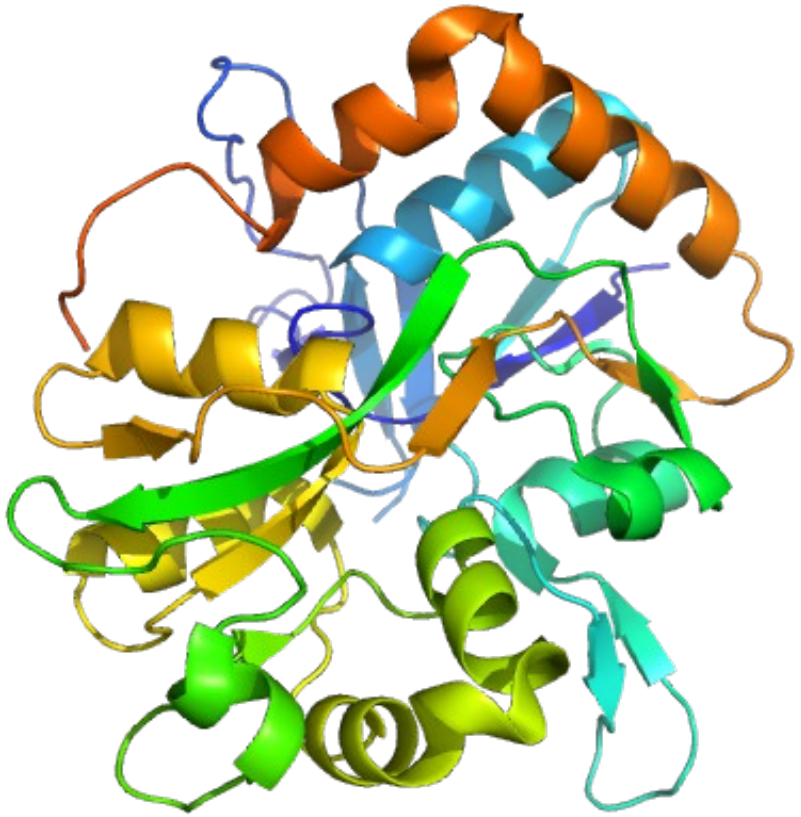
- import Molfile
- Iterate through smarts list:
  - Match and Collect pKa and charge
  - Delete subtructure match
- gives pKa list + charges
- Simulate charge at ph 0 – 14 with 0.01 interval, identify value closest to zero => pI
- Wrapped in PlpythonU in Postgres database

Calculating charge at set pH.

- Can also be used to predict number of counter ions
- Example: To predict number of TFA counter ions, calculate charge at pH -2

Proteins

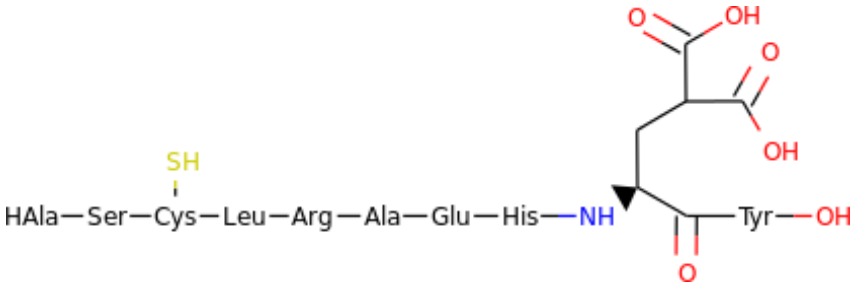




Pseudo Atoms

- Each amino acid treated as a single atom
- Extended table in Code/GraphMol/atomic_data.cpp
- All AA has valence of 2
  - except cysteine, which has 3 and excludes Sulphur. Allows modeling of disulfide bridges)
- Assigned atom numbers from 171 forward to be compatible with BCF Proteax and ISIS/Draw
- Proteax (Biochemfusion) has functionality to go from sequence or full structure to condensed (and mixed) and back.

- 171 Ala 1.9 2 5.0 71.079 6 300 71.03711378 2 \n"
- "172 Arg 1.9 2 6.6 156.189 6 300 156.101111004 2 \n \
- 173 Asn 1.9 2 5.7 114.104 6 300 114.042927432 2 \n \
- 174 Asp 1.9 2 5.6 115.088 6 300 115.02694302 2 \n \ ...



Pseudo Atoms 2

- A lot of things seem to work straight away :-)
- Known bug: Smiles can be written but not read
- Seems due to only reading two letters for atom typing (=Specification)

- Mol = Chem.MolFromSmarts(“[#171][#172]”)
- Chem.MolToSmiles(mol)
- '[AlaH][ArgH]'
- Chem.MolToMolFile(mol, 'Dipept_condensed.mol')
- Chem.MolFromSmiles('[AlaH][ArgH]')

  => Smiles Parse Error:

- RDKit
-
- 2 1 0 0 0 0 0 0 0 0999 V2000
- 0.0000 0.0000 0.0000 Ala 0 0 0 0 0 0 0 0 0 0 0 0
- 0.0000 0.0000 0.0000 Arg 0 0 0 0 0 0 0 0 0 0 0 0
- 1 2 6 0
- M END

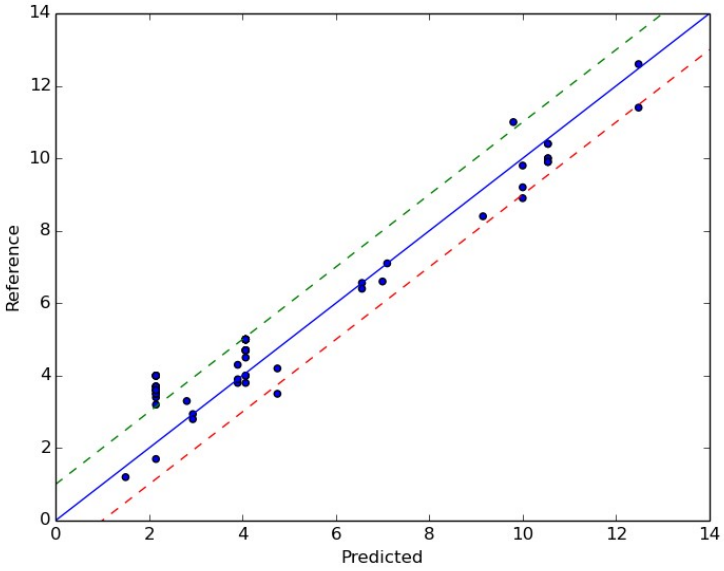
Extension with Pseudo Atoms

- Separate dictionary added to handle pseudo atoms
- Matched Pseudo atoms substituted with Glycine.
- Special treatment of uncapped terminal AA's

```
condensedtable = [
('Condensed Tyrosine',
Chem.MolFromSmarts('[#189]'),
10.46, 0),
('Condensed Histidine',
Chem.MolFromSmarts('[#179]'),
6.04, 1),
('Condensed Glutamic Acid',
Chem.MolFromSmarts('[#177]'),
4.07, 0),
('Condensed Cysteine',
Chem.MolFromSmarts('[#175]
[SH]'), 8.37, 0),
```

Extended dictionary for modified AA and Term caps

- Peptide medicinal chemists uses un-natural amino acids
- > 300 unnatural side chains were modeled by extending the dictionary
- pKa compared to values from commercial pKa predictor



Wrapping it up in PLpythonU

```
CREATE OR REPLACE FUNCTION
standardization.sanitizeparent_log(
    IN mol text,
    IN ruleset text DEFAULT 'FDA'::text,
    IN debug integer DEFAULT 0,
    OUT mol text,
    OUT log text)
  RETURNS record AS
$BODY$
from rdkit import Chem
...
$BODY$
  LANGUAGE plpythonu VOLATILE
  COST 100;
ALTER FUNCTION
standardization.sanitizeparent_log(text, text, integer)
  OWNER TO postgres;
```

- Risk of crashing database if process hangs
- Problem with Cartridge functions shadowing rdkit + differences in cartridge + rdkit versions

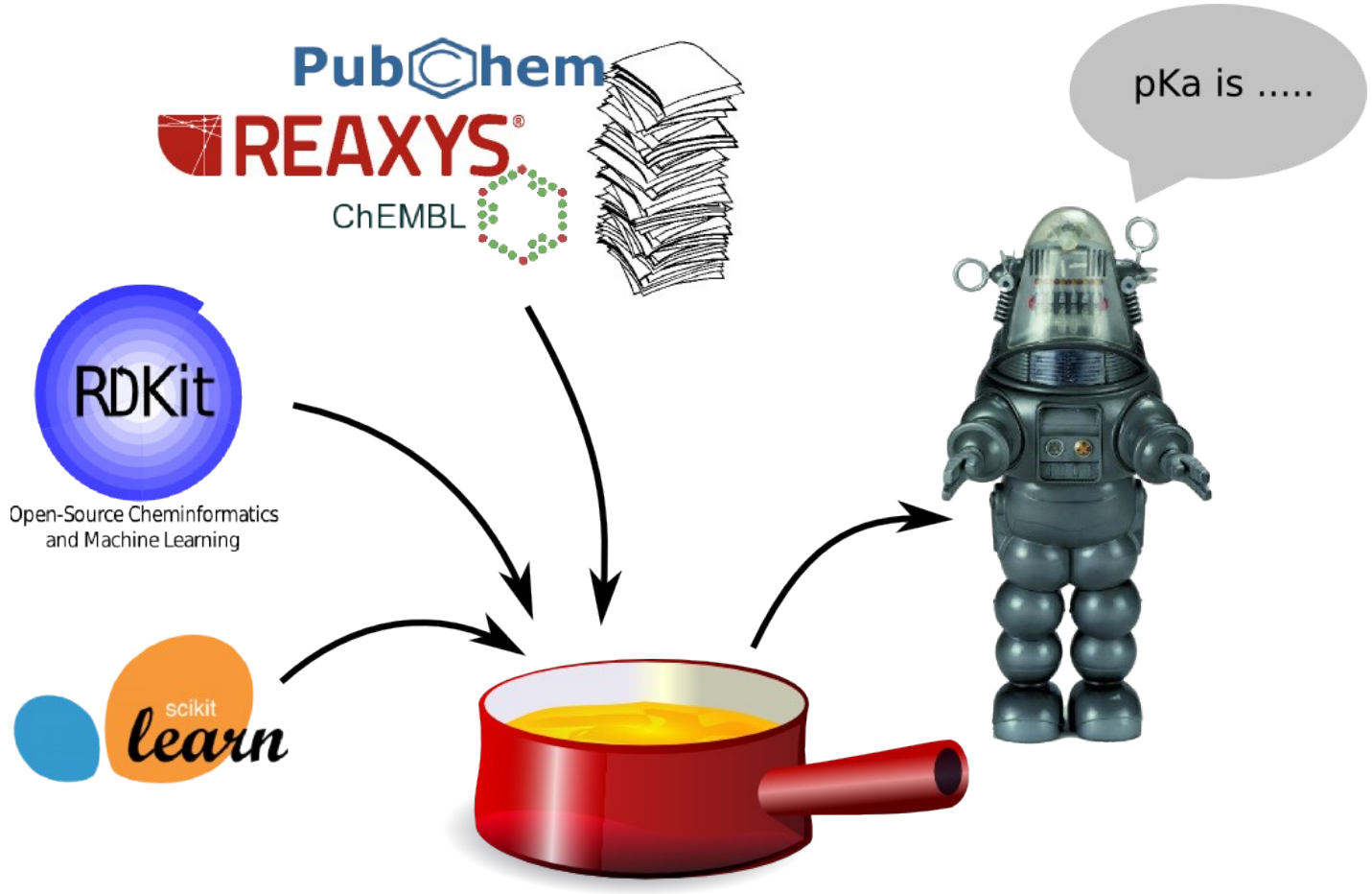
Wrapping it up in PlpythonU 2

```
CREATE OR REPLACE FUNCTION peputils.estimate_charge(
    IN mol text,
    IN ph double precision,
    OUT chargeout double precision,
    OUT log text)
  RETURNS record AS
$BODY$
import sys, subprocess
ext_process = subprocess.Popen(['python','/var/lib/pgsql/peputils/estimate_charge.py',mol,str(ph)], stdout=subprocess.PIPE,
stderr=subprocess.PIPE)
result = ext_process.communicate()
chargeout = result[0]
error = result[1]
... Processing of results and exception handling ...
$BODY$
  LANGUAGE plpythonu VOLATILE
  COST 100;
ALTER FUNCTION peputils.estimate_charge(text, double precision)
  OWNER TO postgres;
```

Conclusion Part 1

- Pros
  - Simple approach
  - Works well for natural Amino acids
- Cons
  - Dictionary gets complicated to work with as size increases with unnatural Amino acids

Part2: Machine Learning model of pKa



pKa revisited

$$HA \rightleftharpoons A^- + H^+$$

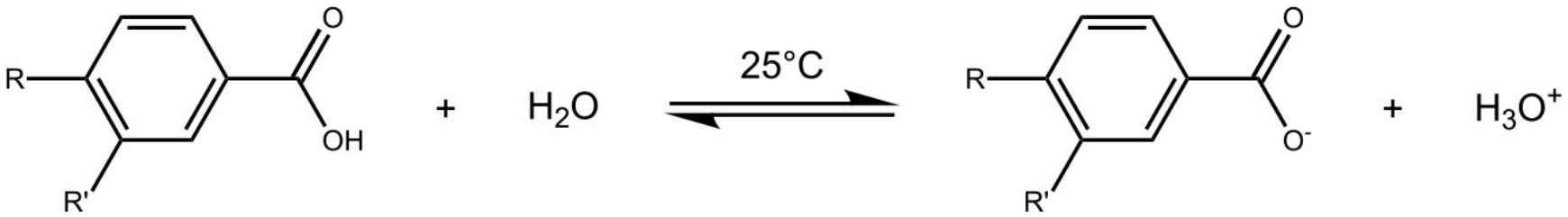
$$K_a = \frac{[A^-][H^+]}{[HA]}$$

$$pKa = -\log(K_a)$$

- pKa influenced by
  - Electronegativity of atom
  - Electron withdrawing substituents/groups
  - Resonance stabilisation of anion
  - Nearby charges/partial charges
  - Formation of intra molecular hydrogen bonds
  - Steric effects
  - Solvent
  - ...

Hammett constants are linearly additive

- Hammett finds linear effects of para and meta substituents of benzoic acid derivatives for reaction rates and equilibrium constants
    - Louis P. Hammett J. Am. Chem. Soc., 1937, 59 (1), pp 96–103
- The constants are linearly additive
- pKa = pKa0 - rho*sum(sigma)



Getting a dataset (Reaxys)

- Good and large datasets in readily usable formats are hard to get by
- Most extensive was found in Reaxys database
- Found 22588 compound hits with associated pKa values determined in aqueous solution (DE.SOL = ‘H2O’)

Filtering of Dataset

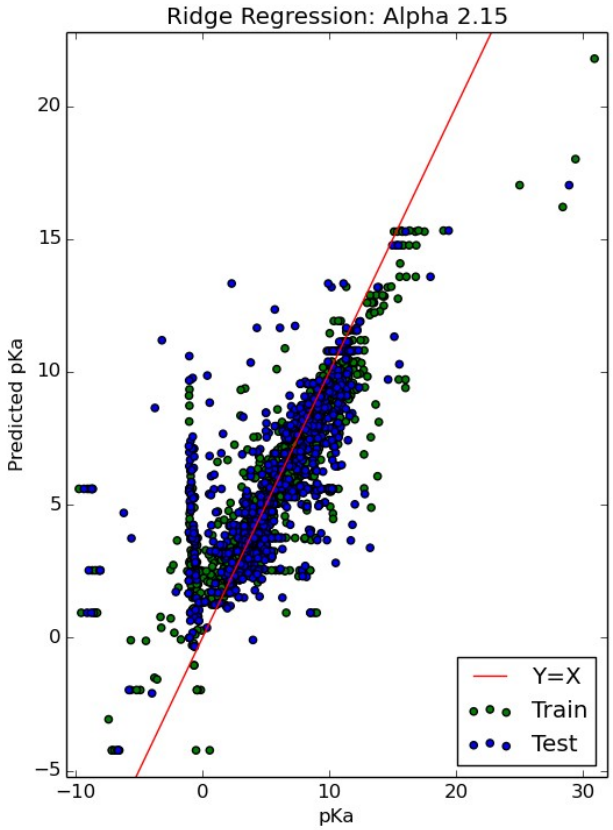
- Solvent is not very consistently tagged
  - What's the difference between 'H2O', 'Water' and 'Aqueous Solution'? Also buffers etc...
- Inconsistent tagging of substructure associated with pKa
- a1/apparent sometimes seem mixed up with b1/apparent?
- Spectrophotometric determination often found as outliers
- Each compound may be associated with many pKa values for multiple sites

Initial filtering and selection of monoprotic compounds

- matches = (data['Solvent (Dissociation Exponent)'] == 'H2O') & (data['Type (Dissociation Exponent)'] == 'a1/apparent')
- seldata = data[matches]
- Compound must only one time match OH, N with H or protonable, S. (smarts matching in RDkit)
- => 2250 compounds and pKa values

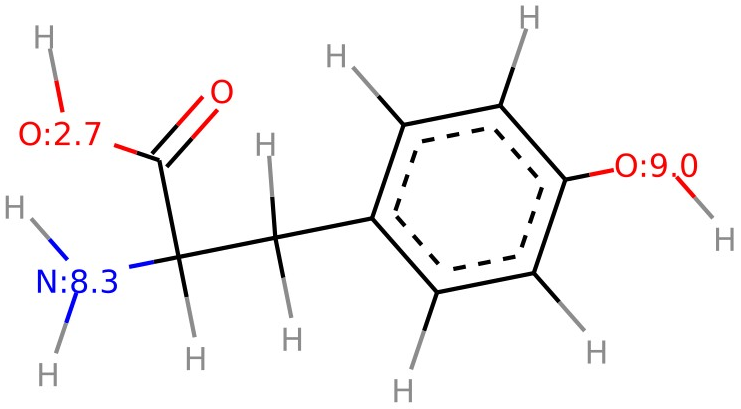
First model on uncurated data

- Compounds fingerprinted with path based Rdkit fingerprints
- Ridge regression from scikit-learn
- Later removed pKa < 0 and > 14



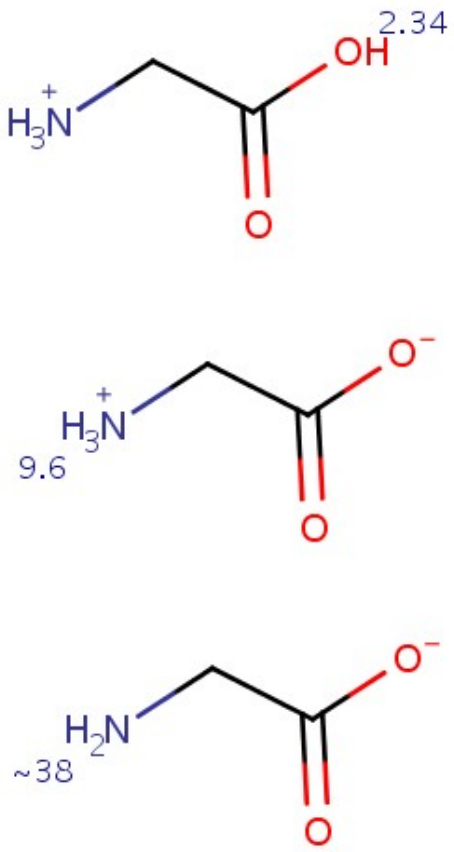
Using rooted paths allows for multi site prediction

- Using the matched atom from filtering allow “rooting” of the FP
- Symmetry in path perception may pose a problem
- Switched to Morgan type



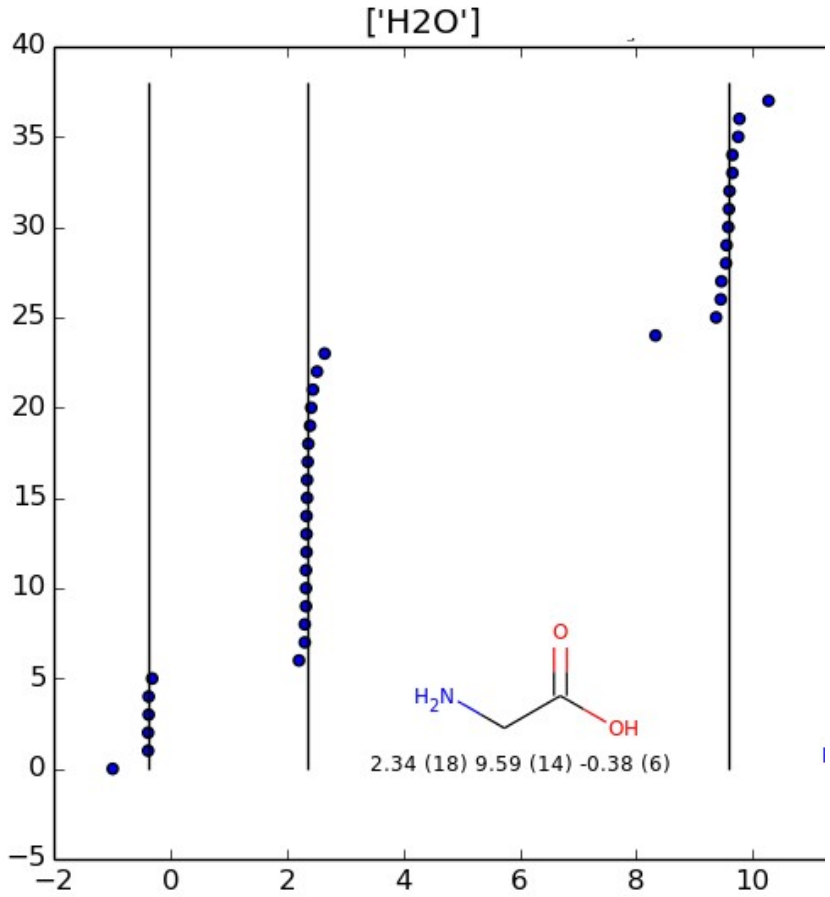
Model v0.2

- Clustering of multiple pKa values
- Stepwise de-protonation
- Assign pKa value to value tag of Atom
- Benchmarking with dataset from Liao et al. 2009

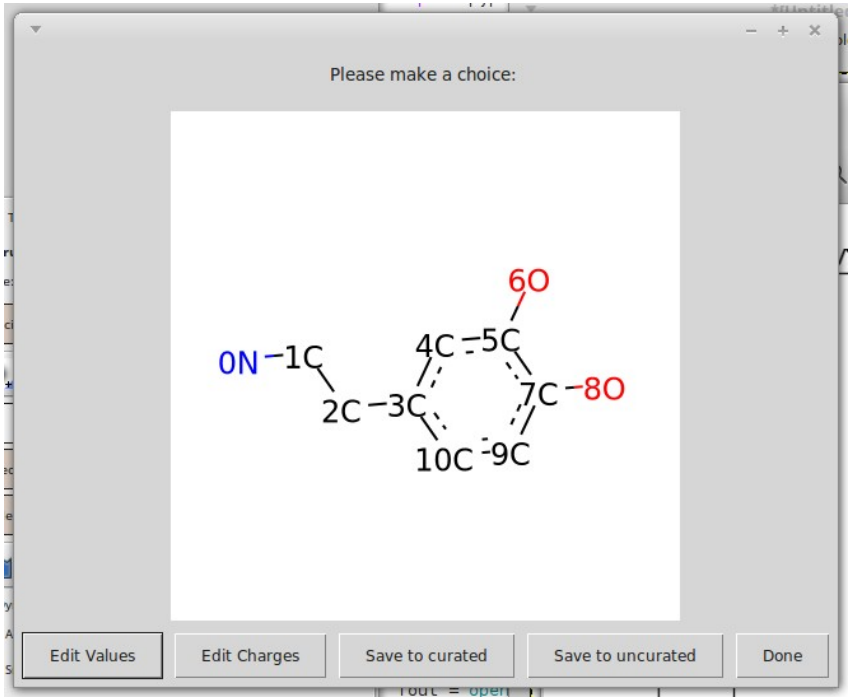


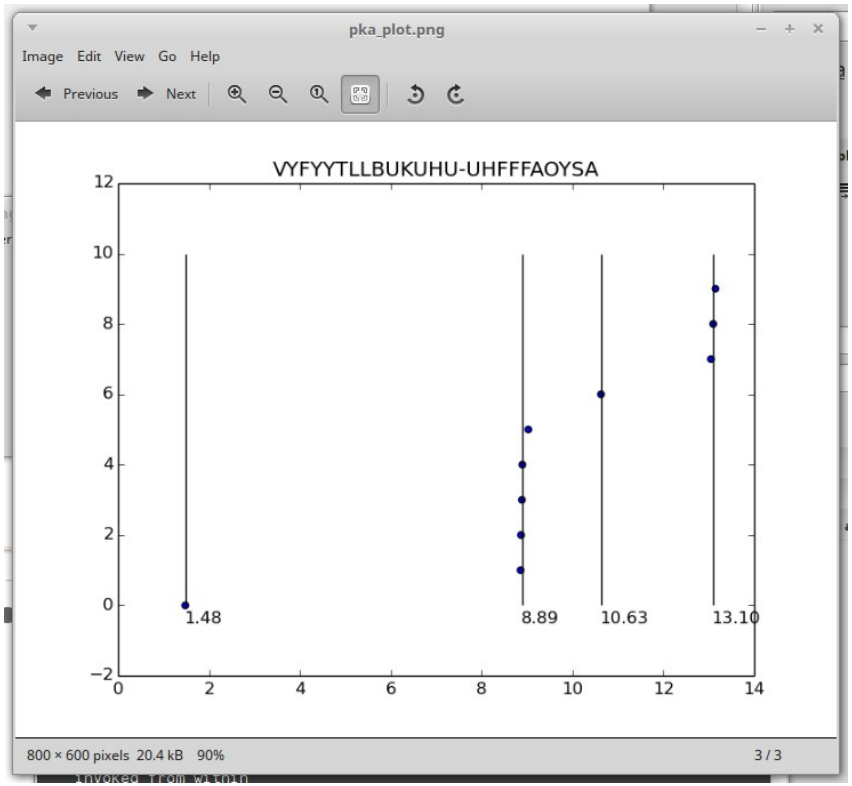
Clustering of pKa values

- Pka values for similar InChikey clustered
- from sklearn.cluster import MeanShift
- ms = MeanShift(cluster_all = False,bandwidth=1)
- ms.fit(X)
- ms.predict(X)
- ms.cluster_centers_ # or take medians of groups.



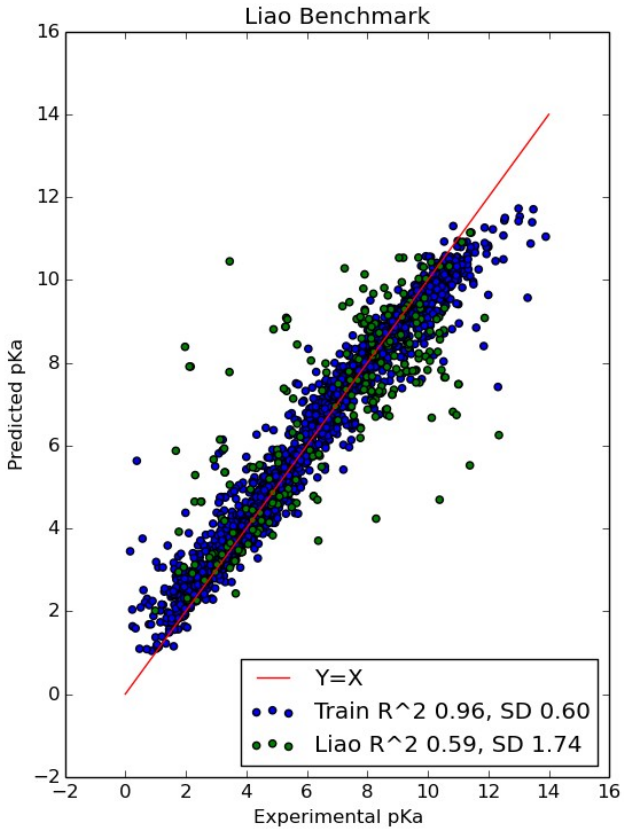
Tagging of Datasets using easygui GUI





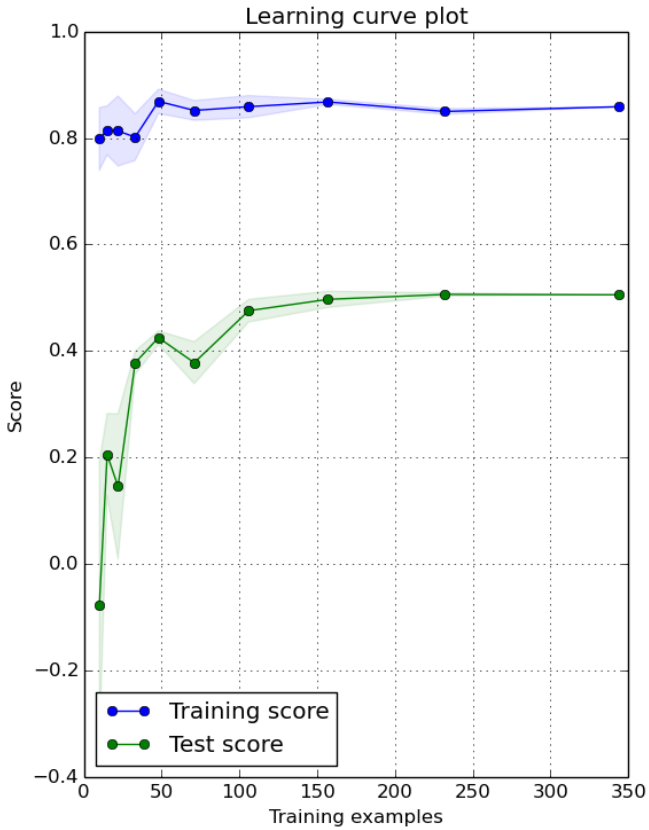
Benchmark Liao dataset

- **197** pharma molecules with **261** highly reliable pKa values
- "Comparison of Nine Programs Predicting pKa Values of Pharmaceutical Substances" J. Chem. Inf. Model. 2009, 49, 2801–2812
- R^2 0.59 and SD 1.74 is *slightly* better than the worst (Jaguar 0.58, 1.81)



Learning Curve

- Learning curve levels off
- => Much larger dataset (or clever selection)
- => less regularization (no effect)
- => more complex model (little effect)
- => **more suitable descriptors**



Future work and plans

- Develop better anchored descriptors
  - Drop symmetry in existing path based FP's
  - Handle charges
  - Weight based on distance
  - Build local models based on primary atom type (RF should catch this?)
- Further data curation: Automatic protonation and tagging of large dataset?
- Next steps:
  - Dissemination, How to package for others to use?
  - Student project (Bioinformatics, Cph. Uni)
  - Collaborate and extend dataset

Acknowledgments

- Jan Holst Jensen (BioChemFusion Aps)
- Jakob Tolborg (Zealand Pharma A/S)
- Rdkitters:
  - Greg Landrum
  - Rdkit Community