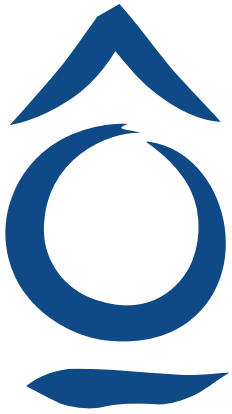


OpenEye
SCIENTIFIC

QUACPAC

Release 2.0.0.3

OpenEye Scientific Software, Inc.

December 12, 2018

CHAPTER

ONE

INTRODUCTION

1.1 Overview

The chemistry of molecular interactions is a matter of shape and electrostatics, but doing electrostatics poorly is worse than doing none at all; accurate charges are required. Even the best charge models are useless if protonation states are wrong. **QUACPAC** attempts to offer everything necessary to do charges correctly. It includes pKa and tautomer enumeration in order to get correct protonation states, partial charges using multiple models that cover a range of speed and accuracy, and electrostatic potential map construction and storage.

1.2 Applications

The **QUACPAC** distribution comprises 4 applications:

FixpKa

- Set the ionization state of input molecules using a rule based system

MolCharge

- Assigns appropriate atomic partial charges, both to small molecule ligands and to biopolymers

pKaTyper

- Enumerates the protonation states and assess ligand pKas

Tautomers

- Canonicalizes and enumerates the tautomeric forms of a small molecules

CHAPTER

TWO

FIXPKA

2.1 FixpKa

2.1.1 Overview

FixpKa has a rule-based system to set the ionization state of input molecules. If pKa normalization is turned on, the molecule is set to its most energetically favorable ionization state for `pH=7.4`. The rule-based nature of this calculation allows it to be very fast. Further, despite being rule-based, this approach takes into account many secondary charge interactions.

While more advanced levels of theory can be found for predicting ionization states, this method is very well suited to virtual-screening database preparation. However, **FixpKa** may not be appropriate for hit-to-lead or lead optimization.

2.1.2 Example Commands

The following section shows a common command-line execution of *fixpka*.

```
prompt> fixpka drugs.sdf fixpka_drugs.smi
```

Reads the file *drugs.sdf* in SD format, sets the pKa state of each molecule and writes them into the file *fixpka_drugs.smi*.

2.1.3 Command Line Help

A description of the commandline interface can be obtained by executing **FIXPKA** with the *--help* option.

```
prompt> fixpka --help
```

will generate the following output:

```
Help functions:
  fixpka --help simple      : Get a list of simple parameters (as seen above)
  fixpka --help all         : Get a complete list of parameters
  fixpka --help defaults    : List the defaults for all parameters
  fixpka --help <parameter> : Get detailed help on a parameter
  fixpka --help html        : Create an html help file for this program
  fixpka --help versions    : List the toolkits and versions used in the application
```

2.1.4 Required Parameters

-in
File containing one or more molecules from which pKa normalization will be done. An input file is required, but the *-in* flag is optional. The first parameter listed with no flag will be automatically mapped to the input file. Unflagged parameters must occur last in the parameter list. Any OpenEye supported molecule format can be used for input.

-out
File for writing output. An output file is required, but the *-out* flag is optional. The second parameter listed with no flag will be automatically mapped to the input file. Unflagged parameters must occur last in the parameter list. Any OpenEye supported molecule format can be used for output but *.smi* or *.ism* is recommended.

2.1.5 Optional Parameters

-ionize
Specifies ionization environment, either neutral pH or un-ionize, to remove all ions possible. Legal values are neutral, 7.4, un-ionize, and unionize.

CHAPTER

THREE

MOLCHARGE

3.1 MolCharge

3.1.1 Overview

molcharge uses the *-method* flag to determine what method to use to generate partial charges. Some methods can be applied to 2D input structures, others require 3D input structures; for the latter the run will fail if a 3D structure is not provided. Hydrogens will be sprouted on the molecules if they are not present already.

With default parameters, *molcharge* will sprout explicit hydrogens and assign MMFF94 partial charges.

3.1.2 Example Commands

An example run of the *molcharge* program is given below.

```
prompt> molcharge -method mmff drugs.sdf drugs.mol2
```

This executes *molcharge* with the default parameters, to sprout explicit hydrogens and assign MMFF94 partial charges. The file *drugs.sdf* is opened in SD format for input, and the output is written to the file *drugs.mol2* in Sybyl .mol2 format.

```
prompt> molcharge -method amberff99sb 2iko_prot.pdb 2iko_prot.oeb
```

In this sample execution, a protein is charged using the Amber charge set and written to OEB format.

3.1.3 Command Line Help

A description of the commandline interface can be obtained by executing **molcharge** with the `--help` option.

```
prompt> molcharge --help
```

will generate the following output:

```
Help functions:
  molcharge --help simple      : Get a list of simple parameters (as seen above)
  molcharge --help all         : Get a complete list of parameters
  molcharge --help defaults    : List the defaults for all parameters
  molcharge --help <parameter> : Get detailed help on a parameter
  molcharge --help html        : Create an html help file for this program
  molcharge --help versions    : List the toolkits and versions used in the application
```

3.1.4 Required Parameters

-in
This is the input file. This file can contain molecules in a wide-variety of molecular formats. Some of the partial charging models, such as AM1 and AM1BCC require coordinates in order to calculate charges. While the input file is required, the *-in* flag is optional. If no *-in* flag is specified, the first unflagged parameter is used as the input file.

-out
This is the output file and it is a required parameter. Since the object of *molcharge* is to generate partial charges, it will only write to formats that can specify a partial charge. These formats currently include only *.mol2*, *.mol2H* and *.oeb*. While the output file is required, the *-out* flag is optional. If no *-out* flag is specified, the second unflagged parameter is used as the output file.

3.1.5 Optional Parameters

-method `{option}`
Selects the charging method to be applied to the input molecule(s). Choose one of the following recommended options:

am1bccsym applies AM1BCC charges as published, including partial semiempirical AM1 geometry optimization and averaging bond-topologically symmetric charges. It requires a 3D geometry associated with an input structure. These are good-quality charges for pharmaceutical organic molecules for intermolecular interactions.

mmff applies MMFF94 charges. It will work with only a 2D geometry. This method is the best choice for use with the MMFF forcefield and these charges are of passable quality for intermolecular interactions. This is the default method.

amberff99sb applies to proteins the RESP charge set used for Amber forcefields ff94, ff96, ff98, ff99, ff99sb, and ff99sbc0 (the same charge set is used in all these cases). This method only works for standard amino acids. These are very good-quality charges for proteins for intermolecular interactions.

gasteiger applies gasteiger charges (a charge-equilibration method). **NOTE:** This method *should not be used for intermolecular interactions.* This method was intended for comparing relative reactivity of related organic chemical functional groups within different molecular contexts. It will work with only a 2D geometry.

There are a number of other options for charge methods, but none are recommended. They are listed in the *molcharge* appendix.

-param
The argument for this flag is the name of a file containing control parameters. The control parameter file acts to either replace or augment the command line interface. All parameters necessary for program execution may be provided in the control parameter file, although any command given explicitly on the command line will supersede options found in the parameter file. *molcharge* generates a new parameter file containing the full set of execution parameters upon every execution. The name of the parameter file written by *molcharge* is created by combining the prefix base name with the *'.parm'* extension.

-paramfile
The filename for the output parameter file can be set with this flag and it overrides -prefix flag for the name of parameter file.

-prefix
The argument for this flag defines the prefix to be used for parameter and log files. The parameter and log files will be written to what follows this flag plus the extension *.param* or *.log*. [default = molcharge]

CHAPTER

FOUR

PKATYPER

4.1 pKaTyper

4.1.1 Overview

Assessment of ligand pKas can be broken into two phases. The first phase is enumeration of the protonation states of interest, and the second phase is assigning a pKa value to each of these states. An intermediate phase of assigning microscopic pKas to each of the atomic-deprotonations may also be considered.

It is common in the course of modeling small-molecules to explore the conformational ensemble of the small molecule. Often structures as high as 5-8 kcal/mol above the aqueous ground-state can be important to biological processes. It is also appropriate to enumerate a protonation-state ensemble of the small molecule.

Similar to *tautomers*, OpenEye has a solution for enumerating reasonable protonation states, but not for assessing the energetics of the state (e.g. assigning a pKa value). OpenEye's solution for pKa enumeration seeks to enumerate all of the pKa states that fall roughly in the pH range of 2-14 in aqueous solvent. This range of pKa values generates an ensemble that includes the ground-state plus all charge states within 8 kcals/mol ΔG. This value was chosen to correspond to the similar range that is often used for generating conformational ensembles of small molecules.

pKaTyper enumerates charge states based on primary, secondary and tertiary atom types of each atom in a molecule. The primary atom type is based on the atom's group and its valence. The primary atom-type defines the atom's basic propensity to support a formal charge. The secondary atom-type is defined by the atom-type of the neighbors for each atom. These secondary atom-types, such as aromaticity, alpha-beta unsaturation, or electronegative-groups, modulate each atom's basic propensity to support formal charges. The tertiary atom-types assess the effects of nearby formal charges on a given atom's formal charge. The combination of the primary, secondary and tertiary atom-types determine which formal charge states are allowed for each atom in a molecule. The primary and secondary atom-types are determined once, while the tertiary atom-types are determined as part of the enumeration process.

pKaTyper is a rudimentary approach to pKa prediction. While pkatyper is not suited for prediction of absolute pKas, it is quite amenable to enumeration of all reasonable charge states of a very wide variety of small-molecule chemistries.

pKaTyper is not a conformer generation program and will not create coordinates for molecules that are read in without coordinates. When used on molecules with three-dimensional coordinates, **pKaTyper** attempts to place new hydrogens in a reasonable manner. However, **pKaTyper** does not modify the heavy-atom coordinates of the molecule. In cases where the change in protonation-state dictates a change in conformation, one will need to use a conformer-generation tool (such as OMEGA) to generate reasonable conformations for the output from **pKaTyper**. We recommend that in preparation of small-molecules for study, charge-state and tautomer enumeration be performed before conformer generation.

4.1.2 Example Commands

The following section shows several common command-line executions of **pkatyper**. Each example is followed by an explanation of what the program will do.

```
prompt> pkatyper drugs.sdf enumerated_drugs.smi
```

Reads the file *drugs.sdf* in SD format, enumerates the pKa states of each molecule and writes them into the file *enumerated_drugs.smi*. Molecules with only one identified pKa state are passed through to the output file.

```
prompt> pkatyper -in drugs.sdf -out enumerated_drugs.smi
```

This command generates the exact same behavior as described above.

```
prompt> pkatyper -count drugs.sdf drugCounts
```

This command reads each of the molecules in *drugs.sdf* and counts the number of pKa states. For each molecule, a single line is added to the *drugCounts* output file that contains the molecule title and the number of states.

4.1.3 Command Line Help

A description of the commandline interface can be obtained by executing **pkatyper** with the *--help* option.

```
prompt> pkatyper --help
```

will generate the following output:

```
Help functions:
  pkatyper --help simple      : Get a list of simple parameters (as seen above)
  pkatyper --help all         : Get a complete list of parameters
  pkatyper --help defaults    : List the defaults for all parameters
  pkatyper --help <parameter> : Get detailed help on a parameter
  pkatyper --help html        : Create an html help file for this program
  pkatyper --help versions    : List the toolkits and versions used in the application
```

4.1.4 Required Parameters

-in
: Input file of molecules. An input file is required, but the *-in* flag is optional. The first parameter listed with no flag will be automatically mapped to the input file. Unflagged parameters must occur last in the parameter list.

4.1.5 Optional Parameters

-count
: Only count the number of pKa states rather than enumerating each of the states. If the *-count* flag is specified true, then the output file will contain the name of each compound followed by the number states of that molecule. This options writes to a text file or stdout. [default = false]

-max
: This integer parameter is the maximum number of pKa states which will be enumerated for any single molecule. If a value of zero is passed to this parameter, no limit will be set. [default = 100]

-out
: Output file of molecules. Both the output file and the *-out* flag are optional. The second parameter listed with no flag will be automatically mapped to the output file. Unflagged parameters must occur last in the parameter list.

If no output is specified at all, output will be written to *std::out* in SMILES format. Any OpenEye supported molecule format can be used for output. If the *-count* flag is specified true, then molecular format is no longer relevant to the output file.

-param

The argument for this flag is the name of a file containing control parameters. The control parameter file acts to either replace or augment the command line interface. All parameters necessary for program execution may be provided in the control parameter file, although any command given explicitly on the command line will supersede options found in the parameter file. **pkatyper** generates a new parameter file containing the full set of execution parameters upon every execution. The name of the parameter file written by *pkatyper* is created by combining the prefix base name with the *.parm* extension.

-paramfile

The filename for the output parameter file can be set with this flag and it overrides -prefix flag for the name of parameter file.

-prefix

The argument for this flag defines the prefix to be used for parameter and log files. The parameter and log files will be written to what follows this flag plus the extension *.param* or *.log*. [default = pkatyper]

CHAPTER

FIVE

TAUTOMERS

5.1 Tautomers

5.1.1 Overview

OpenEye's **Tautomers** program is used for canonicalizing and/or enumerating the tautomeric forms of a small molecule. Canonicalization converts any of the tautomeric forms of a given molecule into a single unique representation. This is useful for database registration where alternate representations of tautomeric compounds often leads to duplicate entries in a database.

Some effort is made by the **Tautomers** program to direct the `canonical` representation to be a physiologically preferred form. However, there are no guarantees the tautomer selected is indeed the lowest energy and, indeed, solvent effects, etc., preclude there being a single `best` form of a tautomer. Fortunately, this is not necessary for database work.

Tautomers is not a conformer generation program and will not create coordinates for molecules that are read in with no coordinates. When used on molecules with three-dimensional coordinates, **Tautomers** attempts to place hydrogens in a reasonable manner. However, **Tautomers** does not modify the heavy-atom coordinates of the molecule. In cases where the change in tautomer-state dictates a change in conformation, one will need to use a conformer-generation tool (such as **OMEGA**) to generate reasonable conformations for the output from **Tautomers**. We recommend that in the preparation of small-molecules for study, charge-state and tautomer enumeration be performed before conformer generation.

5.1.2 Example Commands

The following section shows several common command-line executions of **tautomers**. Each example is followed by an explanation of what the program will do.

Consider the following input file, *guanine.smi*, that contains just the following line: *c1[nH]c2c(=O)[nH]c(nc2n1)N*

```
prompt> tautomers guanine.smi output.smi
```

Which should write the following 15 structures to the file *output.smi*.

```
c1[nH]c2c(=O)[nH]c(nc2n1)N untitled1_1
c1[nH]c2c(n1)c(=O)[nH]c(n2)N untitled1_2
```

Please note that if **tautomers** fails to find tautomers for a molecule, it will output the molecule to a fail file. This occurs if the algorithm times out. e.g. benzene would not be in the fail file but a very large peptide might be. Name of the fail file is either 'tautomers.fail' or 'prefix.fail' if prefix is specified by *-prefix* flag.

5.1.3 Command Line Help

A description of the commandline interface can be obtained by executing **tautomers** with the *--help* option.

```
prompt> tautomers --help
```

will generate the following output:

```
Help functions:
  --help simple      : Get a list of simple parameters (as seen above)
  --help all         : Get a complete list of parameters
  --help defaults    : List the defaults for all parameters
  --help <parameter> : Get detailed help on a parameter
  --help html        : Create an html help file for this program
  --help versions    : List the toolkits and versions used in the application
```

5.1.4 Required Parameters

-in
: File containing one or more molecules for which tautomers will be generated. An input file is required, but the *-in* flag is optional. The first parameter listed with no flag will be automatically mapped to the input file. Unflagged parameters must occur last in the parameter list. Any OpenEye supported molecule format can be used for input.

5.1.5 Optional Parameters

-ch3
: Allows carbon atoms that are close by appropriate heteroatoms to change hybridization state. This permits structures such as cyclohexa-2,4-dien-1-one to be considered a tautomer of phenol, and other examples of keto-enol tautomerism. Unfortunately, the *-ch3* flag may cause the calculation time to increase by many orders of magnitude for some molecules. [default = true]

-maxtoreturn
: Specify a maximum number of tautomers to enumerate for a single input structure. Over 99% of compounds require less than 100 tautomers, indeed most organic compounds gave only a single tautomer, however some pathological dyes and chromophores may individually have nearly a million possible tautomeric forms. The current default is a limit of 256 tautomers per input structure. [default = 256]

-maxgenerated
: Specify a maximum number of tautomers that may be generated per input molecule.. [default = 4096]

-maxtautomericatoms
: Specify a maximum number of tautomeric atoms allowed in a molecule. [default = 70]

-maxtime
: This flag limits the amount of time (in seconds) spent generating tautomer for each molecule. [default = 120.0]

-maxzonesize
: Specify a maximum number of atoms allowed in a continuous tautomerization zone. [default = 35]

-out
: Output filename, where the file extension indicates the output format. All OpenEye supported molecule formats are allowed but .smi and .ism are recommended. The default setting will print .ism format to standard out.

-param
: The argument for this flag is the name of a file containing control parameters. The control parameter file acts to either replace or augment the command line interface. All parameters necessary for program execution may

be provided in the control parameter file, although any command given explicitly on the command line will supersede options found in the parameter file. *tautomers* generates a new parameter file containing the full set of execution parameters upon every execution. The name of the parameter file written by **tautomers** is created by combining the prefix base name with the *.parm* extension.

-paramfile
The filename for the output parameter file can be set with this flag and it overrides -prefix flag for the name of parameter file.

-pkanorm
This flag determines whether to apply pKa normalization. If it is true the ionization state of each tautomer will be assigned to a predominate state at pH~7.4. [default = true]

-prefix
The argument for this flag defines the prefix to be used for parameter and log files. The parameter and log files will be written to what follows this flag plus the extension *.param* or *.log*. [default = tautomers]

-rank
Specify whether or not tautomers are ranked and ordered before being returned. If this is false, i.e., the canonical tautomer is returned as the first molecule in the iterator. As opposed to the most "reasonable" tautomer being returned first. [default = true]

-stereo
Specify a stereo loss behavior for the tautomer algorithm. There are three valid values: *None* means that atoms and bonds with labeled stereochemistry are fixed to participate in tautomerization; *LocalSampled* means that individual tautomers that have double bonds to tetrahedral stereocenters will have the stereo removed; *EverSampled* means that if an atom which are tetrahedral stereocenters has a double bonds in any of the tautomers returned by the function, then those tetrahedral stereocenters will have the stereo removed in ALL of the returned tautomers. [default = *Localsampled*]

-warts
Add wart number with _ to each tautomer. [default = true]

CHAPTER

SIX

THEORY

6.1 Theory

6.1.1 MolCharge Theory

Introduction

The assignment of appropriate atomic partial charges, both to small molecule ligands and to biopolymers (such as proteins and nucleic acids) is essential to getting meaningful results from any electrostatics calculation.

A molecule may be considered a collection of atomic nuclei and the electrons that surround them. The number of protons in each nucleus defines its atomic number/element. If the number of electrons exactly matches the number of protons in these nuclei, the molecule is neutral and has no net charge. If there are more electrons than protons, the molecule has a net negative charge, and if there are less, the molecule has a net positive charge.

It is both the atomic nuclei and the net charge that define the identity of a molecule. Indeed, this is a representation common to quantum chemistry. Adding or removing electrons (or atoms) from a molecule produces a different molecule.

In the discrete world of cheminformatics, valence bond theory allows the electrons present in a system to be represented in terms of bonds with formal bond orders, and formal charges assigned to particular atoms. The sum of the formal charges is equal to the net charge on the molecule, but which atoms are assigned which formal charges can be to some extent arbitrary due to resonance delocalization. In such cases the same molecule may be represented by similar connection tables, but with formal charges assigned to different sets of atoms.

For example, guanidinium may be expressed as either N[C+](N)N with the formal charge assigned to the carbon, or as [NH2+]=C(N)N with the formal charge assigned arbitrarily to one of the otherwise equivalent nitrogens. A similar example is a thiocarboxylate group, where either C(=O)[S-] or C(=S)[O-] are both equally appropriate representations of the same chemical functionality.

A zwitterion is an electrically neutral molecule that is represented as containing atoms with positive formal charge as well as atoms with negative formal charge.

Perhaps the most important fact to appreciate when considering formal charges on atoms is that they are all artificial constructs by chemists to accommodate a particular chemical model. A figment of a chemist's fevered imagination. Like valence bond theory, they are an exceptionally useful and powerful discretized model of the universe. But as with any model of reality, it has its limitations. Formal charges, for all their numerous benefits to mankind, unfortunately, are not localized on an atom.

The limitations of describing formal charges with valence bond theory is apparent even within cheminformatics. Sydnones, for example, are a class of heterocyclic compound that cannot be written using normal covalent bonds without introducing and arbitrarily assigning both positive and negative charges. Similarly, in inorganic chemistry, the ditechnetium cation, Te_2^{+5}, causes similar problems where the +5 formal charge cannot be assigned to both technetium atoms without breaking symmetry.

A better model, or approximation, of the wave function describing the distribution of electron density around a molecule is the use of atomic partial charges. A partial charge is a floating-point value assigned to each atomic center intended to model the distribution of electrons over a molecule.

Atomic partial charges are yet another approximation, much like the formal charges described above. However, partial charges provide a much better model to describe the electric field, dipole moment and other observable properties of a molecule.

A common limitation of the use of partial charges is the assumption that they are conformationally invariant. Unfortunately, the distribution of electrons around a molecule depends upon the spatial configuration of its nuclei. Some partial charge assignment algorithms, such as the method of Goddard and Rappe, consider these conformational effects, whilst others that are based on quantum mechanics, such as the RESP and AM1BCC methods of Bayly et al., go to great lengths to eliminate conformational effects, for example, by restraining and symmetrizing symmetric atom positions. This is necessary in order to be able to properly handle multiple conformations and changes in geometry (e.g. geometry optimization) with a single set of atomic charges.

Marsili-Gasteiger Partial Charges

Marsili-Gasteiger partial charges are assigned using a two stage algorithm. In the first stage, seed charges are assigned to each atom in the molecule. For example, carboxylate oxygens are each assigned the value -0.5. During the second stage, these initial charges are then shared across bonds, moving a certain amount of charge from one atom to another. The partial charge moved and its direction is determined by difference in electronegativities of the atoms on each end of the bond. The relaxation algorithm is then iterated several times (by default eight passes), attenuating the charge moved with each iteration. OpenEye does not recommend use of this charge model for intermolecular interactions; it was never intended for this purpose. The author of the method (Johann Gasteiger) developed it to compare relative reactivity of related organic chemical functional groups within different molecular contexts. Here it is included for comparison purposes.

MMFF94 Partial Charges

The partial charges used by the MMFF94 and MMFF94s force fields are assigned using a four stage algorithm. In the first stage, each atom of the molecule is assigned an MMFF94 atom type. In the second stage, an initial seed partial charge is assigned to each atom based upon its atom type. For a few atom types, the initial partial charge also depends upon the local environment. In the third stage, the initial charges assigned to aromatic rings are shared between all atoms of the aromatic ring. Finally, in the fourth stage, a table of bond charge increments (BCI) is used to move charges across bonds based upon the bond type of the bond (single, double, triple) and the atom types of the atoms at each end. Developed for the electrostatic interactions within the above-mentioned force fields, they are the appropriate charges to use with these force fields most notably for intramolecular interactions of pharmaceutical and bio-organic small molecules. They are less well-suited (but still passable) for intermolecular interactions using the common two-body additive Coulomb interactions as used in Amber, Charmm, Gromacs. For these better choices would be amber99sb charges on proteins and peptides, and am1bccsym charges on the ligand.

AM1 Charges

AM1 charges are a set of Mulliken-type charges derived from a semi-empirical quantum-mechanical calculation. For further discussion of this method, please see Dewar et. al. These should not be used for intermolecular interactions of force fields.

AM1BCC Charges

AM1BCC charges start with Mulliken-type partial charges derived from the AM1 semi-empirical quantum mechanical (QM) wave-function. In a second stage, bond-charge corrections (BCCs) are applied to the partial charges on each

atom to generate new partial charges. Many different variants of AM1BCC charges are offered within our API because of the significant influences of several different factors on these QM-derived charges. Specifically, these factors are

- Optimization: whether or not the input geometry is optimized. QM wavefunctions in general are quite sensitive to geometry, especially bond lengths and bond angles, so this can markedly affect the partial charges. In general, optimizing the geometry is recommended. To avoid a collapse of the conformation due to strong intramolecular electrostatic interactions, light restraints to the starting geometry are applied.
- Symmetrization: whether or not topologically similar atoms (for example the two oxygens on a carboxylate) are constrained to have identical values. The true QM wavefunction is usually asymmetric around topologically similar atoms, leading to asymmetric partial charges. However, if the same partial charges are to be used on different conformers (as with general fixed-charge force fields) it is important that these charges be symmetrized or else interconverting between formally degenerate conformers (e.g. 180 degree rotation of the carboxylate) will have non-degenerate electrostatic energies. In general, if the partial charges are to be applied to more than the single conformer used to generate them, symmetrization is strongly recommended.

Another important issue with AM1BCC charges is if highly conformer-specific charges are generated which are unsuitable for other conformers, leading to undesirable perturbed electrostatic energies for those other conformers. To address this problem, we strongly recommend the ELF conformer selection protocol described below.

"Standard" AM1BCC includes both optimization and symmetrization.

OpenEye considers AM1BCC charges to be the best partial charge model currently available. For further discussion, please see the work of Christopher I. Bayly.

ELF Conformer Selection

ELF conformer selection is a method to select one or more conformers having Electrostatically Least-interacting Functional groups (ELF) from a large conformer database. The purpose of this method is to resolve important issues with QM-derived charges in general, including AM1BCC charges. The issue is that strong short-range intramolecular polarizations specific to a certain conformation, as with an intramolecular hydrogen bond or salt bridge, usually leads to strongly perturbed partial charges for the atoms involved. These charges can be very different from those found for other conformers which do not have that intramolecular polar interaction. If such partial charges are applied to all conformers, some of those other conformers are very likely to have wrongly over-stabilized solvation energies.

A second problem arising from this issue is the precision or sensitivity of the electrostatic energies resulting from QM-derived charges. By this we mean the variance in electrostatic relative energies between different conformers depending on what conformer is used for the QM-derived charges. Imagine two different sets of QM-derived partial charges for a molecule, each coming from a conformer having different strong intramolecular hydrogen bonds. Each set of partial charges will have strongly perturbed partial charges for the internally hydrogen bonded atoms, but they will be different. The relative energies between conformers will be different depending which partial charge set is used.

For these reasons it is important to avoid generating QM-derived partial charges from a conformer having electrostatically strongly interacting functional groups. This is what ELF conformer selection does.

ELF needs to start with enough conformers so that it can find a population of conformers that do not have strongly-interacting functional groups. In the first stage, the Coulomb electrostatic energy is calculated for every conformer using the absolute value of the MMFF94 partial charges (original negative charges are replaced with their absolute values). The electrostatic energies with such charges destabilize all strong polar interactions, and thus the lowest electrostatic energies correspond to the Electrostatically Least-interacting Functional groups. The lowest-energy 2% of conformers is selected as the 2% ELF population. We find that averaging 10 diverse conformers from the 2% ELF population is sufficient to provide a well-behaved set of QM-derived partial charges even for highly polar and charged molecules.

Amber ff94, ff96, ff99, ff99sb, and ff99sbc0 Partial Charges

The partial charges used by the AmberFF94 force field are based on fitting quantum mechanical electrostatic potentials (esp). They were developed to address two key issues with earlier esp-fit charge sets: unrealistically high charges on charge centers and the variation of atomic charges with conformation. While the latter should have some basis in electronic structure, numerical instability in the charge fitting process was the source of both these pathologies. AmberFF94 charge sets use restrained esp-fitting (RESP) to control the numerical instabilities and simultaneous multi-conformer fitting to lead to conformation-independent charges that are restricted to individual residues. Particular attention was given to ensure that backbone amides have consistent charges. The Amber force fields ff94, ff96, ff99, ff99sb, and ff99sbc0 all use the same set of RESP charges, they differ in other terms (mostly torsional).

CHAPTER

SEVEN

RELEASE NOTES

7.1 Release Notes

7.1.1 QUACPAC 2.0.0

November 2018

- This version of **QUACPAC** has been built using **OEToolkits 2018.Oct**. The previous version was built using **OEToolkits 2016.Feb**.

New features

- The tautomers functionality has gone through major improvements, and the **Tautomer** application options have been redesigned to take advantage of these improvements.
- Keto-enol tautomerization has been added to improve the canonical grouping of unique molecules by their representative tautomers.

Major bug fixes

The following issues has been fixed in **Tautomers**:

- The handling of keto-enol tautomerization by default has been improved. The improvement extends to include imine-enamine tautomerization and related analogs. The keto and imine forms are favored except in the case of 1,3 diketones, where the internal hydrogen bond can be formed.
- The process for generating reasonable tautomers for many simple and complex pyridone analogs, pyrrole analogs, and pyridine analogs have been improved.
- An issue that caused double bonds and hydrogens to move too far through a series of sp3 centers has been fixed.
- The default behavior of peptide tautomers and peptide stereochemistry have been improved.

Minor bug fixes

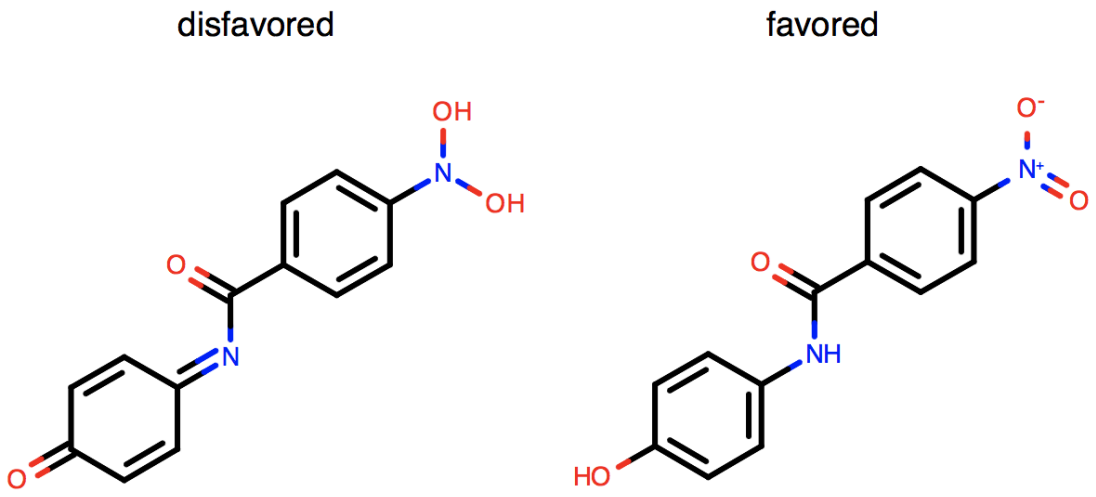
- **pKaTyper**'s output now includes neutral species.

7.1.2 QUACPAC 1.7.0

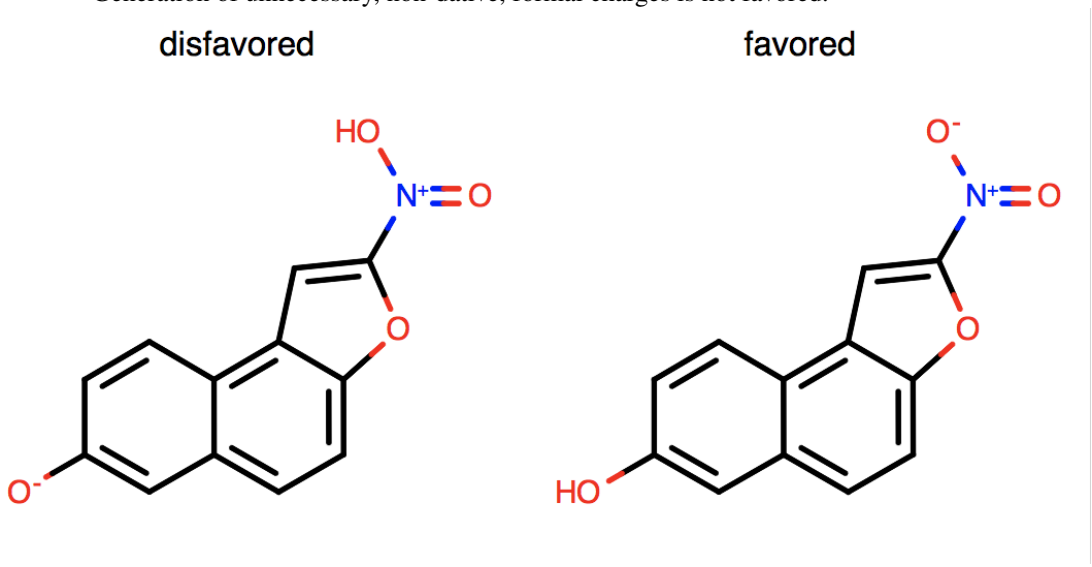
Released October 2016

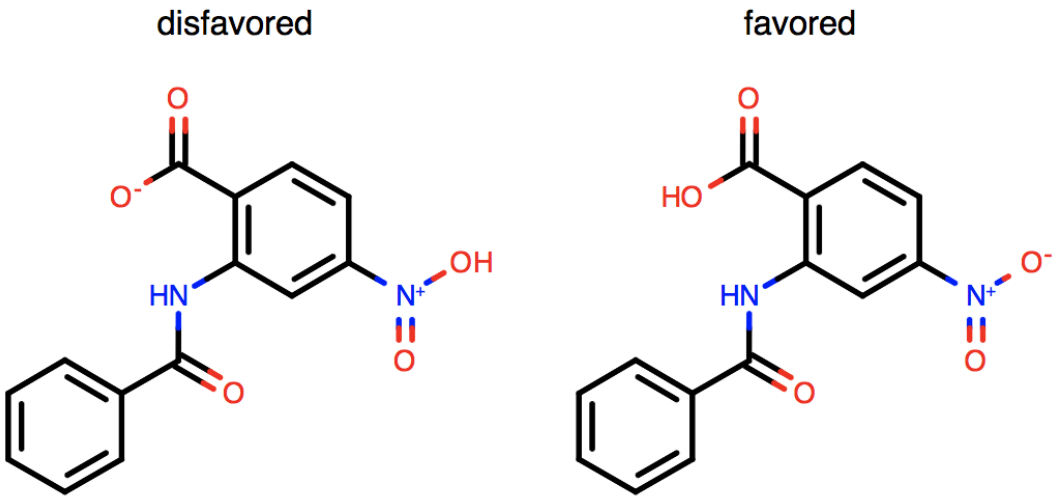
New features

- *-maxtime* flag has been added to tautomer enumeration for setting time limits on searching. Its default value is 60 seconds.
- *tautomers* has been dramatically improved to provide a low-energy, medicinally relevant, "reasonable" tautomeric form that is suitable for depiction for chemists. Significant improvements have been made to the reasonable tautomer algorithm that affect its aliphatic and non-aromatic resonance portions. The depictions below are a useful guide for recognizing how tautomers are favored as reasonable:
  - Conversion of carboxylates to diols and nitros to di-hydroxy amines is not favored.

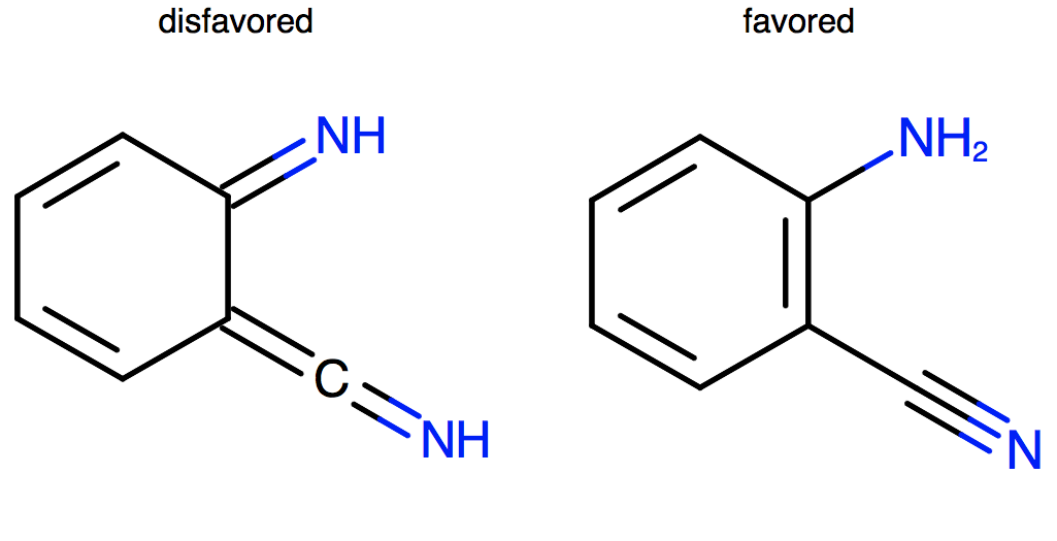


  - Generation of unnecessary, non-dative, formal charges is not favored.

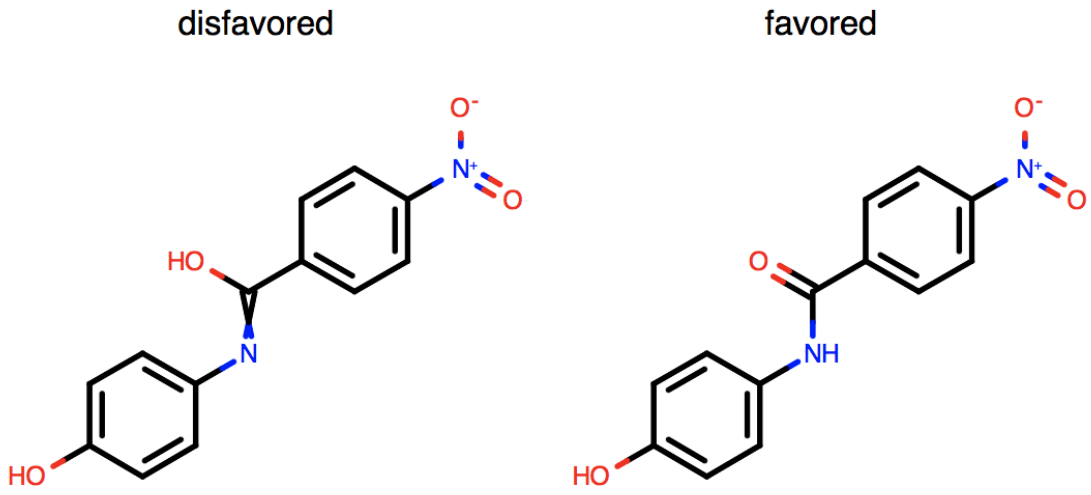




– Exocyclic bonds adjacent to aromatic rings are accounted for.



– Priority is given to aliphatic double-bond positions.



- AM1BCC ELF10, a new method for applying partial charges to a molecule, has been added to *molcharge*. AM1BCC ELF10 allows up to 10 diverse conformers to be selected from those having the Electrostatically Least-interacting Functional groups (ELF). These conformers are then charged with the AM1BCCSym method and the charge sets are averaged to come up with a single charge set that is applied to all conformers. This yields good quality charge sets even for charged molecules.
- TIP3P water charges are now assigned when using Amber charge sets on molecules containing waters. The application of the other Amber charge sets to waters with explicit hydrogens will produce oxygen charges of -0.834 and hydrogen charges of +0.417.
- The default value of *-can* flag for *tautomers* has been changed to true.
- *tautomers* will output failed molecules to a fail file. The file will be named either *tautomers.fail* or, if the prefix is specified by *-prefix* flag, *prefix.fail*.
- A log file has been added for *molcharge*, *tautomers*, and *pkatyper*.
- *tautomers*, *molcharge*, *pkatyper*, and *fixpka* will now label unnamed input molecules as output_1, output_2, etc.

Bug fixes

- *molcharge* now explicitly supports methods AM1BCCSym (a synonym for am1bcc) and AM1BCCSymspt (a synonym for am1bccspt).
- Descriptions for *molcharge* methods have been updated in the usage documentation. In addition, a description for *-formal* has been added.
- *pkatyper* no longer uses the deprecated OETyperMolFunction API.
- *tautomers* no longer uses the deprecated OETautomerMolFunction API.
- The wart symbol has been changed from "@" to "_" to help with parsing SMILES formatting.
- Unbounded stack allocations have been removed from *tautomers*.
- amberff94 will no longer identify a CYS residue as anionic when it is bonded by a sulfur to something other than another CYS residue.
- *pkatyper* no longer outputs an empty file if the output format is not supported.

Other changes

- opls method has been removed from *molcharge*.

7.1.3 QUACPAC 1.6.3

Release August 2013

New Features

- *molcharge* now uses the *-method* parameter for selecting a charge model. All individual flags for charge models have been removed.
- *tautomers* will now allow interconversion of [NH2+]= and [NH3+]-.
- *tautomers* now has a *-warts* option. Enabling this options will number the output molecules with an @ symbol.
- The default AM1BCC charge model in *molcharge* now lightly restrains the AM1 geometry optimization to the starting coordinates. This allows the important relaxation of bond and angle degrees of freedom while greatly reducing the potential to alter the molecule's conformation away from its starting coordinates.
- The AM1BCC charge models *-method am1bccsym* and *-method am1bccspt* in *molcharge* now symmetrize the partial charges over bond-topologically equivalent atoms, e.g. methyl hydrogens, in keeping with the original model. This is especially important with conformationally flexible molecules.
- *molcharge* now has a .fail file to output molecules that do not charge correctly.
- *fixpka* has been refined to reflect feedback from collaborators. In particular isoxazoles and oxadiazoles were added while pyrazoles and aryl sulfonamides were refined. Aryl sulfonamide refinement also incorporated changes based on newly obtained experimental data.

Bug Fixes

- *tautomers* will now properly set stereochemistry on planar carbon to *OEAtomStereo::Undefined*. Previously, planar carbon could inherit stereochemistry from an input molecule with stereochemistry set on a tetrahedral carbon.
- A bug has been fixed in *tautomers* where 3D hydrogens where being suppressed.
- *fixpka* now correctly addresses beta di-amino groups so that if all other considerations are equivalent, secondary amines are treated as most basic, followed by primary amines, with tertiary amine being treated as least basic.
- A few bugs have been fixed in *fixpka* involving quaternary nitrogen and protonated aromatic nitrogens.
- *tautomers* has been fixed so that 3D molecules have 3D hydrogens instead of implicit hydrogens.
- *molcharge* no longer inadvertently converts reduced cysteine residues (CYS) to the oxidized form (CYX) when assigning Amber partial charges to protein residues.

Other changes

- The *-all* flag has been removed from *tautomers*. Equivalent results can been achieved with the following options: *-level 7 -ch3 true*. Level 7 will likely yield a plethora of very unreasonable tautomers and is not recommended.
- This will be the last release to support SuSe 10.

7.1.4 QUACPAC 1.5.0

New Features

- *fixpka* has been added. This application may be used to set a molecule to an energetically favorable ionization state for pH=7.4. This is the same pH model that was available in the Filter application. Additionally, the perception of acceptable valence states has been improved to include phosphorus as well as aromatic oxygen and sulfur with +1 formal charge.
- In *tautomers*: Major improvement for carbon hybridization. Now carbon atoms with bonds to one, two, or three heavy atoms are able to change hybridization state. However, only carbons close to an appropriate heteroatom are allowed to change hybridization state. Additionally, unreasonable charge states of carbon have been removed. Now the enabling the *-ch3* option will only generate tautomers appropriate for the given level.
- In *tautomers*: Improvement of the *-reasonable* option, particularly involving exocyclic heteroatoms.
- In *tautomers*: Added stereo preservation flag *-savestereo* with a default value of *false*. Stereo chemistry can be lost during creation and removal of double bonds, but if the user desires a certain stereo setting to be preserved this flag will prevent the associated atoms and bonds from taking part in tautomerization.
- *pkatyper* now implicitly uses the *-reasonable true* setting for its internal call to *tautomers*. This provides a better reference tautomer when enumerating pKa states.
- In *pkatyper*: Improvements to pka states.
- In *molcharge*: Aromaticity settings on molecules are now unchanged on returned molecules. Aromaticity may be temporarily changed inside the function while charges are being calculated, but the molecule will have the same aromaticity after the function as before.
- In *molcharge*: AmberFF94 charges can now be applied to standard protein residues.
- In *molcharge*: A new *-altlocs* flag will retain all alternative locations in a disordered .pdb structure when charging. This option allows dictionary-based charges such as AmberFF94 to be applied to all atoms in a disordered molecule (but should not be used with conformation-dependent charges such as AM1BCC).

Bug Fixes

- In *tautomers*, atom-ordering could in rare instances be inconsistent for output molecules. This led to cases where even though all associated tautomers led to the same unique tautomer, the unique tautomer could have different SMILES representations depending on the input structure. This has been fixed.

7.1.5 QUACPAC 1.3.1

New features

- The distribution and installation of **QUACPAC** has been modified. The Windows distribution is now a standard EXE installer, and the OS X distribution is a dmg containing a standard pkg installer. The executables are now scripts that chose the correct version of the program at runtime. Please see the Application Installation section for details.

Bug fixes

- The previous version had a license failure when AM1 charges were calculated. This bug has been fixed.

7.1.6 QUACPAC 1.3.0

This release represents the first major reworking of the **QUACPAC** toolkits and applications in several years. Any users of prior versions of **QUACPAC** will quickly notice several significant changes. First, this version of **QUACPAC** does not contain a new release of *protein_pka* program. We are in the midst of a major rewrite of the *protein_pka* program. We hope that this work will bring major improvements in the usability, science and analysis of the *protein_pka* program, yet we do not want to delay the release of the entire **QUACPAC** package waiting for the *protein_pka* program. The current **QUACPAC** release does not contain *protein_pka*, but it will be included in a future release when the rewrite is complete. In the interim, we hope you find the bug fixes and new features included in this release useful.

The preps and qpd programs will no longer be supported future versions of **QUACPAC**.

New features

- A reasonable looking tautomer may be calculated by using the *-reasonable* flag.
- A multiconformer method has been added for calculating AM1BCC charges

Bug fixes

- SMILES map indexes and names are no longer lost when outputting molecules.

7.1.7 QUACPAC 1.1.0

- Special Note: Version 1.4 of **Molcharge** and version 1.2b of the library have been released and inserted into version 1.1 of **QUACPAC**. Both of these have removed support for the VC2003 partial charging method.
- Version 1.1 is the first full stable release of **QUACPAC**. **QUACPAC** remains a heterogeneous release including five primary applications and a programming library with a C++ api.
- **Tautomers** is in version 2.0. It provides enumeration of energetically reasonable tautomers of input molecules.
- **Pkatyper** is in version 1.1. **Pkatyper** provides enumeration of protonation states (pKa ~2-~14).
- **Molcharge** is in version 1.3. **Molcharge** provides MMFF, AM1, AM1BCC, VC2003 and other partial charges on small molecules.
- **Protein_pka** is in version 1.3. **Protein_pka** carries out PB calculations to assess the shifts in protein residue pKa's.
- This release includes the oeproton library. The oeproton library exposes the features of **Pkatyper**, **Tautomers** and **Molcharge** in three high level C++ api points. The version of the library in this release is 1.1b. However, the beta moniker signifies only that at this time, OpenEye reserves the right to modify this api. We believe the quality of the code in this library is very solid and the beta designation does not reflect its quality.

CHAPTER

EIGHT

CITATION

8.1 Citation

Note: To cite **QUACPAC** please use the following:

QUACPAC 2.0.0.3: OpenEye Scientific Software, Santa Fe, NM. http://www.eyesopen.com.

CHAPTER

NINE

APPENDIX

9.1 *molcharge* Appendix: Complete list of *-method* options

While there are many allowed options for the *-method* parameter in *molcharge*, only **am1bccsym**, **mmff** (Default), **amberff99sb**, and **gasteiger** are recommended (indicated in italics). The full list is given below:

am1 applies the AM1 semiempirical method to derive the atomic charges for the molecule. A full geometry optimization is carried out. A 3D geometry for the input molecule is required.

am1spt applies the AM1 semiempirical method to derive the atomic charges for the molecule. Only a single-point calculation is carried out, so it is much faster but also very sensitive to the input geometry (bad charges from a bad geometry). A 3D geometry for the input molecule is required.

am1bcc is a synonym for **am1bccnosym** (described below).

Note: Note that this differs from the "standard" *AM1BCC* charging scheme as published, where symmetrization is done (same as **am1bccsym**).

---

am1bccnosym applies AM1BCC charges, but *without* averaging bond-topologically symmetric charges. The partial semiempirical AM1 geometry optimization is done. It requires a 3D geometry associated with an input structure. Non-symmetric charges can lead to problems with geometry optimizations and with multi-conformer molecules.

am1bccsym (*recommended*) applies AM1BCC charges as published, including partial semiempirical AM1 geometry optimization and averaging bond-topologically symmetric charges. It requires a 3D geometry associated with an input structure. These are good- quality charges for intermolecular interactions involving organic molecules.

am1bccspt is a synonym for **am1bccnosymspt** (described below).

am1bccnosymspt applies AM1BCC charges as published, but *without* averaging bond-topologically symmetric charges. Only a single-point AM1 calculation is carried out, so it is much faster but also very sensitive to the input geometry (bad charges from a bad geometry). A 3D geometry for the input molecule is required.

am1bccsymspt applies AM1BCC charges as published, averaging bond-topologically symmetric charges. Only a single-point AM1 calculation is carried out, so it is much faster but also very sensitive to the input geometry (bad charges from a bad geometry). A 3D geometry for the input molecule is required.

amberff99sb (*recommended*) applies to proteins the RESP charge set used for Amber forcefields ff94, ff96, ff98, ff99, ff99sb, and ff99sbc0 (the same charge set is used in all these cases). This method only works for standard amino acids. These are very good-quality charges for intermolecular interactions involving proteins.

am1bccelf10 charging is designed to average the AM1BCC charges from 10 conformers chosen from the 2% of the conformer population having the Electrostatically Least-interacting Functional (ELF) groups. 10 conformers from 2% means there must be at least 500 conformers to start with; ligands which have fewer rotatable bonds may not have this many. In such cases, this method is designed to take all the conformers in the 2% ELF population.

amberff94, **amberff99**, and **amberff99bsc0** are synonyms for **amberff99sb** The Amber forcefields use the same RESP charge set all cases

gasteiger (*recommended*) applies Gasteiger charges (a charge-equilibration method).

Note: This method *should not be used for intermolecular interactions.* This method was intended for comparing relative reactivity of related organic chemical functional groups within different molecular contexts. It will work with only a 2D geometry.

---

mmff (*recommended*) applies MMFF94 charges. It only requires a 2D geometry. This method ia the best choice for use with the MMFF forcefield and these charges are of passable quality for intermolecular interactions. This is the default method.

initial sets the charges to the MMFF94 initial fractional charges used for charged functional groups, smearing unit charges in the partial charge field onto resonance shared atoms.

formal copies the formal charge field of atoms into the partial charge field.

none sets the partial charges to zero.

[Alexov-1999] E. Alexov and M.R. Gunner,
Incorporating Protein Conformational Flexibility into pH-titration Calculations: Results on T4 Lysozyme,
Biophysical Journal, Vol. 74, pp. 2075-2093, **1999**.

[Baker-1934] John W. Baker, *Tautomerism*, D. Van Nostrand Company Inc.
Publishers, **1934**.

[Dewar-1985] M.J.S Dewar, E.G. Zoebisch, E.F. Healy and J.J.P. Stewart,
AM1: A New General Purpose Quantum Mechanical Model,
Journal of the American Chemical Society, Vol. 107, pp. 3902-3909, **1985**.

[Elguero-1976] Jose Elguero, Claude Marzin, Alan R. Katritzky and Paolo Linda,
The Tautomerism of Heterocycles,
Supplement 1, *Advances in Heterocyclic Chemistry*, Academic Press, **1976**.

[Ertl-1997] Peter Ertl,
Simple Quantum Chemical Parameters as an Alternative to the Hammett Sigma Constants in QSAR Studies,
Quantitative Structure-Activity Relationships (QSAR), Vol. 16, pp. 377-382, **1997**.

[Gasteiger-1978] J. Gasteiger and M. Marsili,
A New Model for Calculating Atomic Charges in Molecules,
Tetrahedron Letters, pp. 3181-3184, **1978**.

[Gasteiger-1980] J. Gasteiger and M. Marsili,
Iterative Partial Equalization of Orbital Electronegativity - A Rapid Access to Atomic Charges,
Tetrahedron, Vol. 36, pp. 3219-3228, **1980**.

[Jakalian-2000] Araz Jakalian, Bruce L. Bush, David B. Jack and Christopher I. Bayly,
Fast, Efficient Generation of High-Quality Atomic Charges. AM1-BCC Model: I: Method,
Journal of Computational Chemistry,
Vol. 21, pp. 132-146, **2000**.

[Jakalian-2002] Araz Jakalian, David B. Jack and Christopher I. Bayly,
Fast, Efficient Generation of High-Quality Atomic Charges. AM1-BCC Model: II: Parameterization and Validation,
Journal of Computational Chemistry, Vol. 23, pp. 1623-1641, **2002**.

[Katritzky-2000] Alan R. Katritzky and A.F. Pozharskii,
Handbook of Heterocyclic Chemistry,
Academic Press, 2nd Edition, **2000**.

[McGuire-2000] A. Ting, R. McGuire, A.P. Johnson and S. Green,
Expert System Assisted Pharmacophore Identification,
Journal of Chemical Information and Computer Science (JCICS), Vol. 40, No. 2, pp. 347-353,
2000.

[Sayle-1999] Roger Sayle and Jack Delany,
Canonicalization and Enumeration of Tautomers, in
Innovative Computational Applications,
Institute for International Research, Sir Francis Drake Hotel,
San Francisco, 25-27 October **1999**.

[Talgren-1996] T.A. Halgren,
Merck Molecular Force Field: II. MMFF94 van der Waals and Electrostatic Parameters for Intermolecular Interactions,
Journal of Computational Chemistry, Vol. 17, No. 5, pp. 520–552, **1996**.

[Trepalin-2003] Sergey V. Trepalin, Andrey V. Skorenko, Konstantin V. Balakin,
Anatoly F. Nasonov, Stanley A. Lang, Andrey A. Ivashchenko and
Nikolay P. Savchuk,
Advanced Exact Structure Searching in Large Databases of Chemical Compounds,
Journal of Chemical Information and Computer Science (JCICS), Vol. 43, No. 3, pp. 852–860, **2003**.

[Yang-1993] Yang, A.-S., M. R. Gunner, R. Sampogna, K. Sharp and B. Honig,
On the Calculation of pKa's in Proteins, *Proteins*, Vol. 15,
pp. 252-265, **1993**.

Symbols

F

M

P

T