NAME

CalculatePhysicochemicalProperties.pl - Calculate physicochemical properties for SD files

SYNOPSIS

CalculatePhysicochemicalProperties.pl SDFile(s)...

PhysicochemicalProperties.pl [--AromaticityModel *AromaticityModelType*] [--CompoundID DataFieldName or LabelPrefixString] [--CompoundIDLabel text] [--CompoundIDMode] [--DataFields "FieldLabel1, FieldLabel2,..."] [-d, --DataFieldsMode All | Common | Specify | CompoundID] [-f, --Filter Yes | No] [-h, --help] [--HydrogenBonds HBondsType1 | HBondsType2] [-k, --KeepLargestComponent Yes | No] [-m, --mode All | RuleOf5 | RuleOf3 | "name1, [name2,...]"] [--MolecularComplexity *Name,Value, [Name,Value,...]*] [--OutDelim comma | tab | semicolon] [--output SD | text | both] [-o, --overwrite] [--Precision Name,Number,[Name,Number,..]] [--RotatableBonds Name,Value, [Name,Value,...]] [--RuleOf3Violations Yes | No] [--RuleOf5Violations Yes | No] [-q, --quote Yes | No] [-r, --root RootName] [-w, --WorkingDir dirname] SDFile(s)...

DESCRIPTION

Calculate physicochemical properties for *SDFile(s)* and create appropriate SD or CSV/TSV text file(s) containing calculated properties.

The current release of MayaChemTools supports the calculation of these physicochemical properties:

```
MolecularWeight, ExactMass, HeavyAtoms, Rings, AromaticRings,
van der Waals MolecularVolume [ Ref 93 ], RotatableBonds,
HydrogenBondDonors, HydrogenBondAcceptors, LogP and
Molar Refractivity (SLogP and SMR) [ Ref 89 ], Topological Polar
Surface Area (TPSA) [ Ref 90 ], Fraction of SP3 carbons (Fsp3Carbons)
and SP3 carbons (Sp3Carbons) [ Ref 115-116, Ref 119 ],
MolecularComplexity [ Ref 117-119 ]
```

Multiple SDFile names are separated by spaces. The valid file extensions are *.sdf* and *.sd*. All other file names are ignored. All the SD files in a current directory can be specified either by **.sdf* or the current directory name.

The calculation of molecular complexity using *MolecularComplexityType* parameter corresponds to the number of bits-set or unique keys [Ref 117-119] in molecular fingerprints. Default value for *MolecularComplexityType*: *MACCSKeys* of size 166. The calculation of MACCSKeys is relatively expensive and can take rather substantial amount of time.

OPTIONS

--AromaticityModel *MDLAromaticityModel | TriposAromaticityModel | MMFFAromaticityModel | ChemAxonBasicAromaticityModel | ChemAxonGeneralAromaticityModel | DaylightAromaticityModel | MayaChemToolsAromaticityModel*

Specify aromaticity model to use during detection of aromaticity. Possible values in the current release are: *MDLAromaticityModel, TriposAromaticityModel, MMFFAromaticityModel, ChemAxonBasicAromaticityModel, ChemAxonGeneralAromaticityModel, DaylightAromaticityModel or MayaChemToolsAromaticityModel*. Default value: *MayaChemToolsAromaticityModel*.

The supported aromaticity model names along with model specific control parameters are defined in AromaticityModelsData.csv, which is distributed with the current release and is available under lib/data directory. Molecule.pm module retrieves data from this file during class instantiation and makes it available to method DetectAromaticity for detecting aromaticity corresponding to a specific model.

--CompoundID *DataFieldName or LabelPrefixString*

This value is --CompoundIDMode specific and indicates how compound ID is generated.

For *DataField* value of --CompoundIDMode option, it corresponds to datafield label name whose value is used as compound ID; otherwise, it's a prefix string used for generating compound IDs like LabelPrefixString<Number>. Default value, *Cmpd*, generates compound IDs which look like Cmpd<Number>.

Examples for *DataField* value of --CompoundIDMode:

```
MolID
ExtReg
```

Examples for *LabelPrefix* or *MolNameOrLabelPrefix* value of --CompoundIDMode:

```
Compound
```

The value specified above generates compound IDs which correspond to Compound<Number> instead of default value of Cmpd<Number>.

--CompoundIDLabel *text*

Specify compound ID column label for CSV/TSV text file(s) used during *CompoundID* value of --DataFieldsMode option. Default value: *CompoundID*.

--CompoundIDMode *DataField | MolName | LabelPrefix | MolNameOrLabelPrefix*

Specify how to generate compound IDs and write to CSV/TSV text file(s) along with calculated physicochemical properties for *text | both* values of --output option: use a *SDFile(s)* datafield value; use molname line from *SDFile(s)*; generate a

sequential ID with specific prefix; use combination of both MolName and LabelPrefix with usage of LabelPrefix values for empty molname lines.

Possible values: *DataField | MolName | LabelPrefix | MolNameOrLabelPrefix*. Default value: *LabelPrefix*.

For *MolNameAndLabelPrefix* value of --CompoundIDMode, molname line in *SDFile(s)* takes precedence over sequential compound IDs generated using *LabelPrefix* and only empty molname values are replaced with sequential compound IDs.

This is only used for *CompoundID* value of --DataFieldsMode option.

--DataFields *"FieldLabel1,FieldLabel2,..."*

Comma delimited list of *SDFiles(s)* data fields to extract and write to CSV/TSV text file(s) along with calculated physicochemical properties for *text | both* values of --output option.

This is only used for *Specify* value of --DataFieldsMode option.

Examples:

```
Extreg
MolID,CompoundName
```

-d, --DataFieldsMode *All | Common | Specify | CompoundID*

Specify how data fields in *SDFile(s)* are transferred to output CSV/TSV text file(s) along with calculated physicochemical properties for *text | both* values of --output option: transfer all SD data field; transfer SD data files common to all compounds; extract specified data fields; generate a compound ID using molname line, a compound prefix, or a combination of both. Possible values: *All | Common | specify | CompoundID*. Default value: *CompoundID*.

-f, --Filter *Yes | No*

Specify whether to check and filter compound data in SDFile(s). Possible values: *Yes or No*. Default value: *Yes*.

By default, compound data is checked before calculating physiochemical properties and compounds containing atom data corresponding to non-element symbols or no atom data are ignored.

-h, --help

Print this help message.

--HydrogenBonds *HBondsType1 | HBondsType2*

Parameters to control calculation of hydrogen bond donors and acceptors. Possible values: *HBondsType1, HydrogenBondsType1, HBondsType2, HydrogenBondsType2*. Default value: *HBondsType2* which corresponds to RuleOf5 definition for number of hydrogen bond donors and acceptors.

The current release of MayaChemTools supports identification of two types of hydrogen bond donor and acceptor atoms with these names:

```
HBondsType1 or HydrogenBondsType1
HBondsType2 or HydrogenBondsType2
```

The names of these hydrogen bond types are rather arbitrary. However, their definitions have specific meaning and are as follows:

```
HydrogenBondsType1 [ Ref 60-61, Ref 65-66 ]:

    Donor: NH, NH2, OH - Any N and O with available H
    Acceptor: N[!H], O - Any N without available H and any O

HydrogenBondsType2 [ Ref 91 ]:

    Donor: NH, NH2, OH - N and O with available H
    Acceptor: N, O - And N and O
```

-k, --KeepLargestComponent *Yes | No*

Calculate physicochemical properties for only the largest component in molecule. Possible values: *Yes or No*. Default value: *Yes*.

For molecules containing multiple connected components, physicochemical properties can be calculated in two different ways: use all connected components or just the largest connected component. By default, all atoms except for the largest connected component are deleted before calculation of physicochemical properties.

-m, --mode *All | RuleOf5 | RuleOf3 | "name1, [name2,...]"*

Specify physicochemical properties to calculate for SDFile(s): calculate all available physical chemical properties; calculate properties corresponding to Rule of 5; or use a comma delimited list of supported physicochemical properties. Possible values: *All | RuleOf5 | RuleOf3 | "name1, [name2,...]"*.

Default value: *MolecularWeight, HeavyAtoms, MolecularVolume, RotatableBonds, HydrogenBondDonors, HydrogenBondAcceptors, SLogP, TPSA*. These properties are calculated by default.

RuleOf5 [Ref 91] includes these properties: *MolecularWeight, HydrogenBondDonors, HydrogenBondAcceptors, SLogP*. *RuleOf5* states: MolecularWeight <= 500, HydrogenBondDonors <= 5, HydrogenBondAcceptors <= 10, and logP <= 5.

RuleOf3 [Ref 92] includes these properties: *MolecularWeight, RotatableBonds, HydrogenBondDonors, HydrogenBondAcceptors, SLogP, TPSA*. *RuleOf3* states: MolecularWeight <= 300, RotatableBonds <= 3, HydrogenBondDonors <= 3, HydrogenBondAcceptors <= 3, logP <= 3, and TPSA <= 60.

All calculates all supported physicochemical properties: *MolecularWeight, ExactMass, HeavyAtoms, Rings, AromaticRings, MolecularVolume, RotatableBonds, HydrogenBondDonors, HydrogenBondAcceptors, SLogP, SMR, TPSA, Fsp3Carbons, Sp3Carbons, MolecularComplexity*.

--MolecularComplexity *Name,Value, [Name,Value,...]*

Parameters to control calculation of molecular complexity: it's a comma delimited list of parameter name and value pairs.

Possible parameter names: *MolecularComplexityType, AtomIdentifierType, AtomicInvariantsToUse, FunctionalClassesToUse, MACCSKeysSize, NeighborhoodRadius, MinPathLength, MaxPathLength, UseBondSymbols, MinDistance, MaxDistance, UseTriangleInequality, DistanceBinSize, NormalizationMethodology*.

The valid paramater valuse for each parameter name are described in the following sections.

The current release of MayaChemTools supports calculation of molecular complexity using *MolecularComplexityType* parameter corresponding to the number of bits-set or unique keys [Ref 117-119] in molecular fingerprints. The valid values for *MolecularComplexityType* are:

```
AtomTypesFingerprints
ExtendedConnectivityFingerprints
MACCSKeys
PathLengthFingerprints
TopologicalAtomPairsFingerprints
TopologicalAtomTripletsFingerprints
TopologicalAtomTorsionsFingerprints
TopologicalPharmacophoreAtomPairsFingerprints
TopologicalPharmacophoreAtomTripletsFingerprints
```

Default value for *MolecularComplexityType*: *MACCSKeys*.

AtomIdentifierType parameter name correspods to atom types used during generation of fingerprints. The valid values for *AtomIdentifierType* are: *AtomicInvariantsAtomTypes, DREIDINGAtomTypes, EStateAtomTypes, FunctionalClassAtomTypes, MMFF94AtomTypes, SLogPAtomTypes, SYBYLAtomTypes, TPSAAtomTypes, UFFAtomTypes*. *AtomicInvariantsAtomTypes* is not supported for during the following values of *MolecularComplexityType*: *MACCSKeys, TopologicalPharmacophoreAtomPairsFingerprints, TopologicalPharmacophoreAtomTripletsFingerprints*. *FunctionalClassAtomTypes* is the only valid value for *AtomIdentifierType* for topological pharmacophore fingerprints.

Default value for *AtomIdentifierType*: *AtomicInvariantsAtomTypes* for all except topological pharmacophore fingerprints where it is *FunctionalClassAtomTypes*.

AtomicInvariantsToUse parameter name and values are used during *AtomicInvariantsAtomTypes* value of parameter *AtomIdentifierType*. It's a list of space separated valid atomic invariant atom types.

Possible values for atomic invariants are: *AS, X, BO, LBO, SB, DB, TB, H, Ar, RA, FC, MN, SM*. Default value for *AtomicInvariantsToUse* parameter are set differently for different fingerprints using *MolecularComplexityType* parameter as shown below:

```
MolecularComplexityType              AtomicInvariantsToUse

AtomTypesFingerprints                AS X BO H FC
TopologicalAtomPairsFingerprints     AS X BO H FC
TopologicalAtomTripletsFingerprints  AS X BO H FC
TopologicalAtomTorsionsFingerprints  AS X BO H FC

ExtendedConnectivityFingerprints     AS X  BO H FC MN
PathLengthFingerprints               AS
```

The atomic invariants abbreviations correspond to:

```
AS = Atom symbol corresponding to element symbol

X<n>   = Number of non-hydrogen atom neighbors or heavy atoms
BO<n> = Sum of bond orders to non-hydrogen atom neighbors or heavy atoms
LBO<n> = Largest bond order of non-hydrogen atom neighbors or heavy atoms
SB<n> = Number of single bonds to non-hydrogen atom neighbors or heavy atoms
DB<n> = Number of double bonds to non-hydrogen atom neighbors or heavy atoms
TB<n> = Number of triple bonds to non-hydrogen atom neighbors or heavy atoms
H<n>   = Number of implicit and explicit hydrogens for atom
Ar     = Aromatic annotation indicating whether atom is aromatic
RA     = Ring atom annotation indicating whether atom is a ring
FC<+n/-n> = Formal charge assigned to atom
MN<n> = Mass number indicating isotope other than most abundant isotope
SM<n> = Spin multiplicity of atom. Possible values: 1 (singlet), 2 (doublet) or
         3 (triplet)
```

Atom type generated by AtomTypes::AtomicInvariantsAtomTypes class corresponds to:

```
AS.X<n>.BO<n>.LBO<n>.<SB><n>.<DB><n>.<TB><n>.H<n>.Ar.RA.FC<+n/-n>.MN<n>.SM<n>
```

Except for AS which is a required atomic invariant in atom types, all other atomic invariants are optional. Atom type specification doesn't include atomic invariants with zero or undefined values.

In addition to usage of abbreviations for specifying atomic invariants, the following descriptive words are also allowed:

```
X : NumOfNonHydrogenAtomNeighbors or NumOfHeavyAtomNeighbors
BO : SumOfBondOrdersToNonHydrogenAtoms or SumOfBondOrdersToHeavyAtoms
LBO : LargestBondOrderToNonHydrogenAtoms or LargestBondOrderToHeavyAtoms
SB :  NumOfSingleBondsToNonHydrogenAtoms or NumOfSingleBondsToHeavyAtoms
DB : NumOfDoubleBondsToNonHydrogenAtoms or NumOfDoubleBondsToHeavyAtoms
TB : NumOfTripleBondsToNonHydrogenAtoms or NumOfTripleBondsToHeavyAtoms
H :  NumOfImplicitAndExplicitHydrogens
Ar : Aromatic
RA : RingAtom
FC : FormalCharge
MN : MassNumber
SM : SpinMultiplicity
```

AtomTypes::AtomicInvariantsAtomTypes module is used to assign atomic invariant atom types.

FunctionalClassesToUse parameter name and values are used during *FunctionalClassAtomTypes* value of parameter *AtomIdentifierType*. It's a list of space separated valid atomic invariant atom types.

Possible values for atom functional classes are: *Ar, CA, H, HBA, HBD, Hal, NI, PI, RA*.

Default value for *FunctionalClassesToUse* parameter is set to:

```
HBD HBA PI NI Ar Hal
```

for all fingerprints except for the following two *MolecularComplexityType* fingerints:

```
MolecularComplexityType                       FunctionalClassesToUse

TopologicalPharmacophoreAtomPairsFingerprints    HBD HBA P, NI H
TopologicalPharmacophoreAtomTripletsFingerprints  HBD HBA PI NI H Ar
```

The functional class abbreviations correspond to:

```
HBD: HydrogenBondDonor
HBA: HydrogenBondAcceptor
PI :  PositivelyIonizable
NI : NegativelyIonizable
Ar : Aromatic
Hal : Halogen
H : Hydrophobic
RA : RingAtom
CA : ChainAtom
```

```
 Functional class atom type specification for an atom corresponds to:

 Ar.CA.H.HBA.HBD.Hal.NI.PI.RA
```

AtomTypes::FunctionalClassAtomTypes module is used to assign functional class atom types. It uses following definitions [Ref 60-61, Ref 65-66]:

```
HydrogenBondDonor: NH, NH2, OH
HydrogenBondAcceptor: N[!H], O
PositivelyIonizable: +, NH2
NegativelyIonizable: -, C(=O)OH, S(=O)OH, P(=O)OH
```

MACCSKeysSize parameter name is only used during *MACCSKeys* value of *MolecularComplexityType* and corresponds to the size of MACCS key set. Possible values: *166 or 322*. Default value: *166*.

NeighborhoodRadius parameter name is only used during *ExtendedConnectivityFingerprints* value of *MolecularComplexityType* and corresponds to atomic neighborhoods radius for generating extended connectivity fingerprints. Possible values: positive integer. Default value: *2*.

MinPathLength and *MaxPathLength* parameters are only used during *PathLengthFingerprints* value of *MolecularComplexityType* and correspond to minimum and maximum path lengths to use for generating path length fingerprints. Possible values: positive integers. Default value: *MinPathLength - 1*; *MaxPathLength - 8*.

UseBondSymbols parameter is only used during *PathLengthFingerprints* value of *MolecularComplexityType* and indicates whether bond symbols are included in atom path strings used to generate path length fingerprints. Possible value: *Yes or No*. Default value: *Yes*.

MinDistance and *MaxDistance* parameters are only used during *TopologicalAtomPairsFingerprints* and *TopologicalAtomTripletsFingerprints* values of *MolecularComplexityType* and correspond to minimum and maximum bond distance between atom pairs during topological pharmacophore fingerprints. Possible values: positive integers. Default value:

MinDistance - 1; *MaxDistance - 10*.

UseTriangleInequality parameter is used during these values for *MolecularComplexityType*: *TopologicalAtomTripletsFingerprints* and *TopologicalPharmacophoreAtomTripletsFingerprints*. Possible values: *Yes or No*. It determines wheter to apply triangle inequality to distance triplets. Default value: *TopologicalAtomTripletsFingerprints - No*; *TopologicalPharmacophoreAtomTripletsFingerprints - Yes*.

DistanceBinSize parameter is used during *TopologicalPharmacophoreAtomTripletsFingerprints* value of *MolecularComplexityType* and correspons to distance bin size used for binning distances during generation of topological pharmacophore atom triplets fingerprints. Possible value: positive integer. Default value: *2*.

NormalizationMethodology is only used for these values for *MolecularComplexityType*: *ExtendedConnectivityFingerprints*, *TopologicalPharmacophoreAtomPairsFingerprints* and *TopologicalPharmacophoreAtomTripletsFingerprints*. It corresponds to normalization methodology to use for scaling the number of bits-set or unique keys during generation of fingerprints. Possible values during *ExtendedConnectivityFingerprints*: *None or ByHeavyAtomsCount*; Default value: *None*. Possible values during topological pharmacophore atom pairs and triplets fingerprints: *None or ByPossibleKeysCount*; Default value: *None*. *ByPossibleKeysCount* corresponds to total number of possible topological pharmacophore atom pairs or triplets in a molecule.

Examples of *MolecularComplexity* name and value parameters:

```
MolecularComplexityType,AtomTypesFingerprints,AtomIdentifierType,
AtomicInvariantsAtomTypes,AtomicInvariantsToUse,AS X BO H FC

MolecularComplexityType,ExtendedConnectivityFingerprints,
AtomIdentifierType,AtomicInvariantsAtomTypes,
AtomicInvariantsToUse,AS X BO H FC MN,NeighborhoodRadius,2,
NormalizationMethodology,None

MolecularComplexityType,MACCSKeys,MACCSKeysSize,166

MolecularComplexityType,PathLengthFingerprints,AtomIdentifierType,
AtomicInvariantsAtomTypes,AtomicInvariantsToUse,AS,MinPathLength,
1,MaxPathLength,8,UseBondSymbols,Yes

MolecularComplexityType,TopologicalAtomPairsFingerprints,
AtomIdentifierType,AtomicInvariantsAtomTypes,AtomicInvariantsToUse,
AS X BO H FC,MinDistance,1,MaxDistance,10

MolecularComplexityType,TopologicalAtomTripletsFingerprints,
AtomIdentifierType,AtomicInvariantsAtomTypes,AtomicInvariantsToUse,
AS X BO H FC,MinDistance,1,MaxDistance,10,UseTriangleInequality,No

MolecularComplexityType,TopologicalAtomTorsionsFingerprints,
AtomIdentifierType,AtomicInvariantsAtomTypes,AtomicInvariantsToUse,
AS X BO H FC

MolecularComplexityType,TopologicalPharmacophoreAtomPairsFingerprints,
AtomIdentifierType,FunctionalClassAtomTypes,FunctionalClassesToUse,
HBD HBA PI NI H,MinDistance,1,MaxDistance,10,NormalizationMethodology,
None

MolecularComplexityType,TopologicalPharmacophoreAtomTripletsFingerprints,
AtomIdentifierType,FunctionalClassAtomTypes,FunctionalClassesToUse,
HBD HBA PI NI H Ar,MinDistance,1,MaxDistance,10,NormalizationMethodology,
None,UseTriangleInequality,Yes,NormalizationMethodology,None,
DistanceBinSize,2
```

--OutDelim *comma | tab | semicolon*

Delimiter for output CSV/TSV text file(s). Possible values: *comma, tab, or semicolon* Default value: *comma*.

--output *SD | text | both*

Type of output files to generate. Possible values: *SD, text, or both*. Default value: *text*.

-o, --overwrite

Overwrite existing files.

--Precision *Name,Number,[Name,Number,..]*

Precision of calculated property values in the output file: it's a comma delimited list of property name and precision value pairs. Possible property names: *MolecularWeight, ExactMass*. Possible values: positive intergers. Default value: *MolecularWeight,2, ExactMass,4*.

Examples:

```
ExactMass,3
MolecularWeight,1,ExactMass,2
```

-q, --quote *Yes | No*

> Put quote around column values in output CSV/TSV text file(s). Possible values: *Yes or No*. Default value: *Yes*.

-r, --root *RootName*

> New file name is generated using the root: <Root>.<Ext>. Default for new file names: <SDFileName><PhysicochemicalProperties>.<Ext>. The file type determines <Ext> value. The sdf, csv, and tsv <Ext> values are used for SD, comma/semicolon, and tab delimited text files, respectively.This option is ignored for multiple input files.

--RotatableBonds *Name,Value, [Name,Value,...]*

> Parameters to control calculation of rotatable bonds [Ref 92]: it's a comma delimited list of parameter name and value pairs. Possible parameter names: *IgnoreTerminalBonds, IgnoreBondsToTripleBonds, IgnoreAmideBonds, IgnoreThioamideBonds, IgnoreSulfonamideBonds*. Possible parameter values: *Yes or No*. By default, value of all parameters is set to *Yes*.

--RuleOf3Violations *Yes | No*

> Specify whether to calculate RuleOf3Violations for SDFile(s). Possible values: *Yes or No*. Default value: *No*.

> For *Yes* value of RuleOf3Violations, in addition to calculating total number of RuleOf3 violations, individual violations for compounds are also written to output files.

> RuleOf3 [Ref 92] states: MolecularWeight <= 300, RotatableBonds <= 3, HydrogenBondDonors <= 3, HydrogenBondAcceptors <= 3, logP <= 3, and TPSA <= 60.

--RuleOf5Violations *Yes | No*

> Specify whether to calculate RuleOf5Violations for SDFile(s). Possible values: *Yes or No*. Default value: *No*.

> For *Yes* value of RuleOf5Violations, in addition to calculating total number of RuleOf5 violations, individual violations for compounds are also written to output files.

> RuleOf5 [Ref 91] states: MolecularWeight <= 500, HydrogenBondDonors <= 5, HydrogenBondAcceptors <= 10, and logP <= 5.

--TPSA *Name,Value, [Name,Value,...]*

> Parameters to control calculation of TPSA: it's a comma delimited list of parameter name and value pairs. Possible parameter names: *IgnorePhosphorus, IgnoreSulfur*. Possible parameter values: *Yes or No*. By default, value of all parameters is set to *Yes*.

> By default, TPSA atom contributions from Phosphorus and Sulfur atoms are not included during TPSA calculations. [Ref 91]

-w, --WorkingDir *DirName*

> Location of working directory. Default value: current directory.

EXAMPLES

To calculate default set of physicochemical properties - MolecularWeight, HeavyAtoms, MolecularVolume, RotatableBonds, HydrogenBondDonor, HydrogenBondAcceptors, SLogP, TPSA - and generate a SamplePhysicochemicalProperties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -o Sample.sdf
```

To calculate all available physicochemical properties and generate both SampleAllProperties.csv and SampleAllProperties.sdf files containing sequential compound IDs in CSV file along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All --output both
  -r SampleAllProperties -o Sample.sdf
```

To calculate RuleOf5 physicochemical properties and generate a SampleRuleOf5Properties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m RuleOf5
  -r SampleRuleOf5Properties -o Sample.sdf
```

To calculate RuleOf5 physicochemical properties along with counting RuleOf5 violations and generate a SampleRuleOf5Properties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m RuleOf5 --RuleOf5Violations Yes
  -r SampleRuleOf5Properties -o Sample.sdf
```

To calculate RuleOf3 physicochemical properties and generate a SampleRuleOf3Properties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m RuleOf3
  -r SampleRuleOf3Properties -o Sample.sdf
```

To calculate RuleOf3 physicochemical properties along with counting RuleOf3 violations and generate a SampleRuleOf3Properties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m RuleOf3 --RuleOf3Violations Yes
  -r SampleRuleOf3Properties -o Sample.sdf
```

To calculate a specific set of physicochemical properties and generate a SampleProperties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m "Rings,AromaticRings"
  -r SampleProperties -o Sample.sdf
```

To calculate HydrogenBondDonors and HydrogenBondAcceptors using HydrogenBondsType1 definition and generate a SampleProperties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m "HydrogenBondDonors,HydrogenBondAcceptors"
  --HydrogenBonds HBondsType1 -r SampleProperties -o Sample.sdf
```

To calculate TPSA using sulfur and phosphorus atoms along with nitrogen and oxygen atoms and generate a SampleProperties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m "TPSA" --TPSA "IgnorePhosphorus,No,
  IgnoreSulfur,No" -r SampleProperties -o Sample.sdf
```

To calculate MolecularComplexity using extendend connectivity fingerprints corresponding to atom neighborhood radius of 2 with atomic invariant atom types without any scaling and generate a SampleProperties.csv file containing sequential compound IDs along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m MolecularComplexity --MolecularComplexity
  "MolecularComplexityType,ExtendedConnectivityFingerprints,NeighborhoodRadius,2,
  AtomIdentifierType, AtomicInvariantsAtomTypes,
  AtomicInvariantsToUse,AS X BO H FC MN,NormalizationMethodology,None"
  -r SampleProperties -o Sample.sdf
```

To calculate RuleOf5 physicochemical properties along with counting RuleOf5 violations and generate a SampleRuleOf5Properties.csv file containing compound IDs from molecule name line along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m RuleOf5 --RuleOf5Violations Yes
  --DataFieldsMode CompoundID --CompoundIDMode MolName
  -r SampleRuleOf5Properties -o Sample.sdf
```

To calculate all available physicochemical properties and generate a SampleAllProperties.csv file containing compound ID using specified data field along with along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All
  --DataFieldsMode CompoundID --CompoundIDMode DataField --CompoundID Mol_ID
  -r SampleAllProperties -o Sample.sdf
```

To calculate all available physicochemical properties and generate a SampleAllProperties.csv file containing compound ID using combination of molecule name line and an explicit compound prefix along with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All
  --DataFieldsMode CompoundID --CompoundIDMode MolnameOrLabelPrefix
  --CompoundID Cmpd --CompoundIDLabel MolID  -r SampleAllProperties
  -o Sample.sdf
```

To calculate all available physicochemical properties and generate a SampleAllProperties.csv file containing specific data fields columns along with with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All
  --DataFieldsMode Specify --DataFields Mol_ID -r SampleAllProperties
  -o Sample.sdf
```

To calculate all available physicochemical properties and generate a SampleAllProperties.csv file containing common data fields columns along with with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All
  --DataFieldsMode Common -r SampleAllProperties -o Sample.sdf
```

To calculate all available physicochemical properties and generate both SampleAllProperties.csv and CSV files containing all data fields columns in CSV files along with with properties data, type:

```
% CalculatePhysicochemicalProperties.pl -m All
  --DataFieldsMode All  --output both -r SampleAllProperties
  -o Sample.sdf
```

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

ExtractFromSDtFiles.pl, ExtractFromTextFiles.pl, InfoSDFiles.pl, InfoTextFiles.pl

COPYRIGHT

Copyright (C) 2017 Manish Sud. All rights reserved.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.