

NAME
ExtendedConnectivityFingerprints.pl - Generate extended connectivity fingerprints for SD files

SYNOPSIS

ExtendedConnectivityFingerprints.pl SDFfile(s)...

```
ExtendedConnectivityFingerprints.pl [-a, --AtomIdentifierType AtomicInvariantsAtomTypes] [--AtomicInvariantsToUse
"AtomicInvariant,AtomicInvariant..."] [--FunctionalClassesToUse "FunctionalClass1,FunctionalClass2..."] [--BitsOrder Ascending |
Descending] [-b, --BitStringFormat BinaryString | HexadecimalString] [--CompoundID DataFieldName or LabelPrefixString] [
--CompoundIDLabel text] [--CompoundIDMode] [--DataFields "FieldLabel1,FieldLabel2,..."] [-d, --DataFieldsMode All | Common |
Specify | CompoundID] [-f, --Filter Yes | No] [--FingerprintsLabel text] [-h, --help] [-k, --KeepLargestComponent Yes | No] [-m,
--mode ExtendedConnectivity | ExtendedConnectivityCount | ExtendedConnectivityBits] [-n, --NeighborhoodRadius number] [--OutDelim
comma | tab | semicolon] [--output SD | FP | text | all] [-o, --overwrite] [-q, --quote Yes | No] [-r, --root RootName] [-s, --size
number] [--UsePerCoreRandom Yes | No] [-v, --VectorStringFormat IDsAndValuesString | IDsAndValuesPairsString |
ValuesAndIDsString | ValuesAndIDsPairsString] [-w, --WorkingDir dirname] SDFfile(s)...
```

DESCRIPTION

Generate extended connectivity fingerprints [Ref 48, Ref 52] for *SDFfile(s)* and create appropriate SD, FP or CSV/TSV text file(s) containing fingerprints vector strings corresponding to molecular fingerprints.

Multiple SDFfile 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 current release of MayaChemTools supports generation of extended connectivity fingerprints corresponding to following *-a, --AtomIdentifierTypes*:

```
AtomicInvariantsAtomTypes, DREIDINGAtomTypes, EStateAtomTypes,
FunctionalClassAtomTypes, MMFF94AtomTypes, SLogPAtomTypes,
SYBYLAtomTypes, TPSAAtomTypes, UFFAtomTypes
```

Based on values specified for *-a, --AtomIdentifierType, --AtomicInvariantsToUse* and *--FunctionalClassesToUse*, initial atom types are assigned to all non-hydrogen atoms in a molecule and these atom types strings are converted into initial atom identifier integers using `TextUtil::HashCode` function. The duplicate atom identifiers are removed.

For *-n, --NeighborhoodRadius* value of 0, the initial set of unique atom identifiers comprises the molecule fingerprints. Otherwise, atom neighborhoods are generated for each non-hydrogen atom up to specified *-n, --NeighborhoodRadius* value. For each non-hydrogen central atom at a specific radius, its neighbors at next radius level along with their bond orders and previously calculated atom identifiers are collected which in turn are used to generate a new integer atom identifier; the bond orders and atom identifier pairs list is first sorted by bond order followed by atom identifiers to make these values graph invariant.

After integer atom identifiers have been generated for all non-hydrogen atoms at all specified neighborhood radii, the duplicate integer atom identifiers corresponding to same hash code value generated using `TextUtil::HashCode` are tracked by keeping the atom identifiers at lower radius. Additionally, all structurally duplicate integer atom identifiers at each specified radius are also tracked by identifying equivalent atoms and bonds corresponding to substructures used for generating atom identifier and keeping integer atom identifier with lowest value.

For *ExtendedConnectivity* value of fingerprints *-m, --mode*, the duplicate identifiers are removed from the list and the unique atom identifiers constitute the extended connectivity fingerprints of a molecule.

For *ExtendedConnectivityCount* value of fingerprints *-m, --mode*, the occurrence of each unique atom identifiers appears is counted and the unique atom identifiers along with their count constitute the extended connectivity fingerprints of a molecule.

For *ExtendedConnectivityBits* value of fingerprints *-m, --mode*, the unique atom identifiers are used as a random number seed to generate a random integer value between 0 and *--Size* which in turn is used to set corresponding bits in the fingerprint bit-vector string.

Example of *SD* file containing extended connectivity fingerprints string data:

```
... ..
... ..
$$$$
... ..
... ..
... ..
41 44 0 0 0 0 0 0 0 0 0999 v2000
-3.3652 1.4499 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
... ..
2 3 1 0 0 0 0
... ..
M END
> <CmpdID>
Cmpd1
```


OPTIONS

-a, --AtomIdentifierType *AtomicInvariantsAtomTypes* | *FunctionalClassAtomTypes* | *DREIDINGAtomTypes* | *EStateAtomTypes* | *MMFF94AtomTypes* | *SLogPAtomTypes* | *SYBYLAtomTypes* | *TPSAAAtomTypes* | *UFFAtomTypes*

Specify atom identifier type to use for assignment of initial atom identifier to non-hydrogen atoms during calculation of extended connectivity fingerprints [Ref 48, Ref 52]. Possible values in the current release are: *AtomicInvariantsAtomTypes*, *FunctionalClassAtomTypes*, *DREIDINGAtomTypes*, *EStateAtomTypes*, *MMFF94AtomTypes*, *SLogPAtomTypes*, *SYBYLAtomTypes*, *TPSAAAtomTypes*, *UFFAtomTypes*. Default value: *AtomicInvariantsAtomTypes*.

--AtomicInvariantsToUse "*AtomicInvariant,AtomicInvariant...*"

This value is used during *AtomicInvariantsAtomTypes* value of a, --AtomIdentifierType option. It's a list of comma 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 [Ref 24]: *AS,X,BO,H,FC,MN*.

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.

--BitsOrder *Ascending* | *Descending*

Bits order to use during generation of fingerprints bit-vector string for *ExtendedConnectivityBits* value of -m, --mode option. Possible values: *Ascending*, *Descending*. Default: *Ascending*.

Ascending bit order which corresponds to first bit in each byte as the lowest bit as opposed to the highest bit.

Internally, bits are stored in *Ascending* order using Perl vec function. Regardless of machine order, big-endian or little-endian, vec function always considers first string byte as the lowest byte and first bit within each byte as the lowest bit.

-b, --BitStringFormat *BinaryString* | *HexadecimalString*

Format of fingerprints bit-vector string data in output SD, FP or CSV/TSV text file(s) specified by --output used during *ExtendedConnectivityBits* value of -m, --mode option. Possible values: *BinaryString*, *HexadecimalString*. Default value: *BinaryString*.

BinaryString corresponds to an ASCII string containing 1s and 0s. *HexadecimalString* contains bit values in ASCII hexadecimal format.

Examples:

```
FingerprintsBitVector;ExtendedConnectivityBits:AtomicInvariantsAtomTyp
```


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

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

Comma delimited list of *SDFFile(s)* data fields to extract and write to CSV/TSV text file(s) along with generated fingerprints for *text | all* 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 *SDFFile(s)* are transferred to output CSV/TSV text file(s) along with generated fingerprints for *text | all* 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 *SDFFile(s)*. Possible values: *Yes or No*. Default value: *Yes*.

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

--FingerprintsLabel *text*

SD data label or text file column label to use for fingerprints string in output SD or CSV/TSV text file(s) specified by --output. Default value: *ExtendedConnectivityFingerprints*.

-h, --help

Print this help message.

-k, --KeepLargestComponent *Yes | No*

Generate fingerprints for only the largest component in molecule. Possible values: *Yes or No*. Default value: *Yes*.

For molecules containing multiple connected components, fingerprints can be generated 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 generation of fingerprints.

-m, --mode *ExtendedConnectivity | ExtendedConnectivityCount | ExtendedConnectivityBits*

Specify type of extended connectivity fingerprints to generate for molecules in *SDFFile(s)*. Possible values: *ExtendedConnectivity, ExtendedConnectivityCount or ExtendedConnectivityBits*. Default value: *ExtendedConnectivity*.

For *ExtendedConnectivity* value of fingerprints -m, --mode, a fingerprint vector containing unique atom identifiers constitute the extended connectivity fingerprints of a molecule.

For *ExtendedConnectivityCount* value of fingerprints -m, --mode, a fingerprint vector containing unique atom identifiers along with their count constitute the extended connectivity fingerprints of a molecule.

For *ExtendedConnectivityBits* value of fingerprints -m, --mode, a fingerprint bit vector indicating presence/absence of structurally unique atom identifiers constitute the extended connectivity fingerprints of a molecule.

-n, --NeighborhoodRadius *number*

Atomic neighborhood radius for generating extended connectivity neighborhoods. Default value: *2*. Valid values: *>= 0*. Neighborhood radius of zero correspond to just the list of non-hydrogen atoms.

Default value of *2* for atomic neighborhood radius generates extended connectivity fingerprints corresponding to path length or diameter value of *4* [Ref 52b].

--OutDelim *comma | tab | semicolon*

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

--output *SD | FP | text | all*

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

-o, --overwrite

Overwrite existing files.

-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: *<SDFFileName><ExtendedConnectivityFP>.<Ext>*. The file type determines *<Ext>* value. The *sdf, fpf, csv, and tsv <Ext>* values are used for SD, FP, comma/semicolon, and tab delimited text files, respectively. This option is ignored for multiple

input files.

-s, --size *number*

Size of bit-vector to use during generation of fingerprints bit-vector string for *ExtendedConnectivityBits* value of *-m, --mode*. Default value: **1024**. Valid values correspond to any positive integer which satisfies the following criteria: power of 2, ≥ 32 and $\leq 2^{**} 32$.

Examples:

```
512
1024
2048
```

--UsePerlCoreRandom *Yes / No*

Specify whether to use Perl CORE::rand or MayaChemTools MathUtil::random function during random number generation for setting bits in fingerprints bit-vector strings. Possible values: *Yes or No*. Default value: **Yes**.

No value option for *--UsePerlCoreRandom* allows the generation of fingerprints bit-vector strings which are same across different platforms.

The random number generator implemented in MayaChemTools is a variant of linear congruential generator (LCG) as described by Miller et al. [Ref 120]. It is also referred to as Lehmer random number generator or Park-Miller random number generator.

Unlike Perl's core random number generator function rand, the random number generator implemented in MayaChemTools, MathUtil::random, generates consistent random values across different platforms for a specific random seed and leads to generation of portable fingerprints bit-vector strings.

-v, --VectorStringFormat *ValuesString | IDsAndValuesString | IDsAndValuesPairsString | ValuesAndIDsString | ValuesAndIDsPairsString*

Format of fingerprints vector string data in output SD, FP or CSV/TSV text file(s) specified by *--output* used during *<ExtendedConnectivityCount>* value of *-m, --mode* option. Possible values: *ValuesString, IDsAndValuesString | IDsAndValuesPairsString | ValuesAndIDsString | ValuesAndIDsPairsString*.

Default value during *<ExtendedConnectivityCount>* value of *-m, --mode* option: *IDsAndValuesString*.

Default value during *<ExtendedConnectivity>* value of *-m, --mode* option: *ValuesString*.

Examples:

```
FingerprintsVector;ExtendedConnectivity:AtomicInvariantsAtomTypes:Radius2;60;AlphaNumericalValues;ValuesString;73555770 333564680 352413391
666191900 1001270906 1371674323 1481469939 1977749791 2006158649 21414
08799 49532520 64643108 79385615 96062769 273726379 564565671 85514103
5 906706094 988546669 1018231313 1032696425 1197507444 1331250018 1338
532734 1455473691 1607485225 1609687129 1631614296 1670251330 17303...
```

```
FingerprintsVector;ExtendedConnectivityCount:AtomicInvariantsAtomTypes
:Radius2;60;NumericalValues;IDsAndValuesString;73555770 333564680 3524
13391 666191900 1001270906 1371674323 1481469939 1977749791 2006158649
2141408799 49532520 64643108 79385615 96062769 273726379 564565671...;
3 2 1 1 14 1 2 10 4 3 1 1 1 1 2 1 2 1 1 1 2 3 1 1 2 1 3 3 8 2 2 2 6 2
1 2 1 1 2 1 1 1 2 1 1 2 1 1 1 1 1 1 1 1 1 1 1 2 1 1
```

-w, --WorkingDir *DirName*

Location of working directory. Default: current directory.

EXAMPLES

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity count fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -m ExtendedConnectivityCount
-r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity bits fingerprints as hexadecimal bit-string corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -m ExtendedConnectivityBits
-r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity bits fingerprints as binary bit-string corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -m ExtendedConnectivityBits
--BitStringFormat BinaryString -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create SampleECAIFP.sdf, SampleECAIFP.fpf and SampleECAIFP.csv files containing sequential compound IDs in CSV file along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl --output all -r SampleECAIFP
-o Sample.sdf
```

To generate extended connectivity count fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create SampleECAIFP.sdf, SampleECAIFP.fpf and SampleECAIFP.csv files containing sequential compound IDs in CSV file along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -m ExtendedConnectivityCount
--output all -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using functional class atom types in vector string format and create a SampleECFCFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes
-r SampleECFCFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using DREIDING atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a DREIDINGAtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using E-state atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a EStateAtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using MMFF94 atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a MMFF94AtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using SLogP atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a SLogPAtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using SYBYL atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a SYBYLAtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using TPSA atom types in vector string format and create a SampleECFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a TPSAAtomTypes
-r SampleECFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using UFF atom types in vector string

format and create a SampleEFCFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a UFFAtomTypes
-r SampleEFCFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 3 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a AtomicInvariantsAtomTypes -n 3
-r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 3 using functional class atom types in vector string format and create a SampleECFCFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes -n 3
-r SampleECFCFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using only AS,X atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a AtomicInvariantsAtomTypes
--AtomicInvariantsToUse "AS,X" -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using only HBD,HBA functional class atom types in vector string format and create a SampleECFCFP.csv file containing sequential compound IDs along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes
--FunctionalClassesToUse "HBD,HBA" -r SampleECFCFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.csv file containing compound ID from molecule name line along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode CompoundID -CompoundIDMode MolName
-r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using functional class atom types in vector string format and create a SampleECFCFP.csv file containing compound IDs using specified data field along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes
--DataFieldsMode CompoundID -CompoundIDMode DataField --CompoundID Mol_ID
-r SampleECFCFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.tsv file containing compound ID using combination of molecule name line and an explicit compound prefix along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode CompoundID -CompoundIDMode MolnameOrLabelPrefix
--CompoundID Cmpd --CompoundIDLabel MolID -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using functional class atom types in vector string format and create a SampleECFCFP.csv file containing specific data fields columns along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes
--DataFieldsMode Specify --DataFields Mol_ID -r SampleECFCFP
-o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using atomic invariants atom types in vector string format and create a SampleECAIFP.tsv file containing common data fields columns along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a AtomicInvariantsAtomTypes
--DataFieldsMode Common -r SampleECAIFP -o Sample.sdf
```

To generate extended connectivity fingerprints corresponding to neighborhood radius up to 2 using functional class atom types in vector string format and create SampleECFCFP.sdf, SampleECFCFP.fpf and SampleECFCFP.csv files containing all data fields columns in CSV file along with fingerprints vector strings data, type:

```
% ExtendedConnectivityFingerprints.pl -a FunctionalClassAtomTypes
--DataFieldsMode All --output all -r SampleECFCFP
-o Sample.sdf
```

AUTHOR

Manish Sud <msud@san.rr.com>

SEE ALSO

InfoFingerprintsFiles.pl, SimilarityMatricesFingerprints.pl, AtomNeighborhoodsFingerprints.pl, MACCSKeysFingerprints.pl, PathLengthFingerprints.pl, TopologicalAtomPairsFingerprints.pl, TopologicalAtomTorsionsFingerprints.pl, TopologicalPharmacophoreAtomPairsFingerprints.pl, TopologicalPharmacophoreAtomTripletsFingerprints.pl

COPYRIGHT

Copyright (C) 2004-2012 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.