

---

**NAME**

TorsionLibraryAlerts

**SYNOPSIS**

```
import TorsionLibraryAlerts
```

**DESCRIPTION**

TorsionLibraryAlerts module provides the following functions:

TorsionLibraryAlerts.GetTorsionLibraryFilePath, TorsionLibraryAlerts.IdentifyTorsionLibraryAlertsForRotatableBonds, TorsionLibraryAlerts.ListTorsionLibraryInfo, TorsionLibraryAlerts.TorsionLibraryAlerts

**FUNCTIONS**

TorsionLibraryAlerts.GetTorsionLibraryFilePath

```
GetTorsionLibraryFilePath(self)
```

Get torsion strain library file path.

*Arguments:*

Nothing.

*Returns:*

FilePath (str): Torsion strain library path.

TorsionLibraryAlerts.IdentifyTorsionLibraryAlertsForRotatableBonds

```
IdentifyTorsionLibraryAlertsForRotatableBonds(self, Mol)
```

Identify torsion library alerts for a molecule by matching rotatable bonds against SMARTS patterns specified for torsion rules in torsion energy library file.

*Arguments:*

Mol (object): RDKit molecule object.

*Returns:*

bool: True - Molecule contains strained torsions; False - Molecule contains no strained torsions or rotatable bonds.

dict or None: Torsion alerts information regarding matching of rotatable bonds to torsion strain library.

*Example(s):*

```
from TorsionAlerts.TorsionLibraryAlerts import TorsionLibraryAlerts

LibraryAlerts = TorsionLibraryAlerts()
AlertsStatus, AlertsInfo = LibraryAlerts.
    IdentifyTorsionLibraryAlertsForRotatableBonds(RDKitMol)

# List of rotatable bond IDs...
RotatableBondIDs = AlertsInfo["IDs"]

# Dictionaries containing information for rotatable bonds by using
# bond ID as key...
for ID in AlertsInfo["IDs"]:
    MatchStatus = AlertsInfo["MatchStatus"][ID]
    AlertTypes = AlertsInfo["AlertTypes"][ID]
    AtomIndices = AlertsInfo["AtomIndices"][ID]
    TorsionAtomIndices = AlertsInfo["TorsionAtomIndices"][ID]
    TorsionAngles = AlertsInfo["TorsionAngles"][ID]
    TorsionAngleViolations = AlertsInfo["TorsionAngleViolations"][ID]
    HierarchyClassNames = AlertsInfo["HierarchyClassNames"][ID]
    HierarchySubClassNames = AlertsInfo["HierarchySubClassNames"][ID]
```

```

TorsionRuleNodeID = AlertsInfo["TorsionRuleNodeID"][ID]
TorsionRulePeaks = AlertsInfo["TorsionRulePeaks"][ID]
TorsionRuleTolerances1 = AlertsInfo["TorsionRuleTolerances1"][ID]
TorsionRuleTolerances2 = AlertsInfo["TorsionRuleTolerances2"][ID]
TorsionRuleSMARTS = AlertsInfo["TorsionRuleSMARTS"][ID]

```

### TorsionLibraryAlerts.ListTorsionLibraryInfo

```
ListTorsionLibraryInfo(self)
```

List torsion strain library information.

#### Arguments:

Nothing.

#### Returns:

Nothing. The torsion library information is printed.

### TorsionLibraryAlerts.TorsionLibraryAlerts

```

TorsionLibraryAlerts(self, AlertsMode = "Red", MinAlertsCount = 1,
NitrogenLonePairAllowHydrogenNbrs = True, NitrogenLonePairPlanarityTolerance = 1.0,
RotBondsSMARTSMODE = "SemiStrict", RotBondsSMARTSPattern = None, TorsionLibraryFilePath =
"auto")

```

Identify strained molecules from an input file for torsion library [ Ref 146, 152, 159 ] alerts by matching rotatable bonds against SMARTS patterns specified for torsion rules in a torsion library file. The molecules must have 3D coordinates. The default torsion library file, TorsionLibrary.xml, is available in the directory containing this file.

The data in torsion library file is organized in a hierarchical manner. It consists of one generic class and six specific classes at the highest level. Each class contains multiple subclasses corresponding to named functional groups or substructure patterns. The subclasses consist of torsion rules sorted from specific to generic torsion patterns. The torsion rule, in turn, contains a list of peak values for torsion angles and two tolerance values. A pair of tolerance values define torsion bins around a torsion peak value. For example:

```

<library>
  <hierarchyClass name="GG" id1="G" id2="G">
    ...
  </hierarchyClass>
  <hierarchyClass name="CO" id1="C" id2="O">
    <hierarchySubClass name="Ester bond I" smarts="O=[C:2][O:3]">
      <torsionRule smarts="[O:1]=[C:2]!@[O:3]~[CH0:4]">
        <angleList>
          <angle value="0.0" tolerance1="20.00"
            tolerance2="25.00" score="56.52"/>
        </angleList>
      </torsionRule>
    ...
  ...
</hierarchyClass>
<hierarchyClass name="NC" id1="N" id2="C">
  ...
</hierarchyClass>
<hierarchyClass name="SN" id1="S" id2="N">
  ...
</hierarchyClass>
<hierarchyClass name="CS" id1="C" id2="S">
  ...
</hierarchyClass>
<hierarchyClass name="CC" id1="C" id2="C">
  ...
</hierarchyClass>
<hierarchyClass name="SS" id1="S" id2="S">

```

```
...
</hierarchyClass>
</library>
```

The rotatable bonds in a 3D molecule are identified using a default SMARTS pattern. A custom SMARTS pattern may be optionally specified to detect rotatable bonds. Each rotatable bond is matched to a torsion rule in the torsion library and assigned one of the following three alert categories: Green, Orange or Red. The rotatable bond is marked Green or Orange for the measured angle of the torsion pattern within the first or second tolerance bins around a torsion peak. Otherwise, it's marked Red implying that the measured angle is not observed in the structure databases employed to generate the torsion library.

#### Arguments:

```
AlertsMode(str): Torsion library alert types to use for issuing
    alerts about molecules.
MinAlertsCount(int): Minimum number of alerts allowed in a molecule.
NitrogenLonePairAllowHydrogenNbrs (bool): Use hydrogen neighbors
    attached to nitrogen during the determination of its planarity.
NitrogenLonePairPlanarityTolerance (float): Angle tolerance in
    degrees allowed for nitrogen to be considered coplanar with its
    three neighbors.
RotBondsSMARTSMode (str): SMARTS pattern to use for identifying
    rotatable bonds in a molecule. Possible values: NonStrict,
    SemiStrict, Strict or Specify.
RotBondsSMARTSPattern (str): SMARTS pattern for identifying rotatable
    bonds. This parameter is only valid for 'Specify' value
    'RotBondsSMARTSMode'.
TorsionLibraryFilePath (str): A XML file name containing data for
    torsion library.
```

#### Returns:

```
object: An instantiated class object.
```

The following sections provide additional details for the parameters.

**AlertsMode:** Torsion library alert types to use for issuing alerts about molecules containing rotatable bonds marked with Green, Orange, or Red alerts. Possible values: Red or RedAndOrange.

**MinAlertsCount:** Minimum number of rotatable bond alerts allowed in a molecule.

**NitrogenLonePairAllowHydrogenNbrs, NitrogenLonePairPlanarityTolerance:** These parameters are used during the matching of torsion rules containing 'N\_lp' in their SMARTS patterns. The 'allowHydrogensNbrs' allows the use hydrogen neighbors attached to nitrogen during the determination of its planarity. The 'planarityTolerance' in degrees represents the tolerance allowed for nitrogen to be considered coplanar with its three neighbors.

The torsion rules containing 'N\_lp' in their SMARTS patterns are categorized into the following two types of rules:

```
TypeOne:

[ CX4:1 ][ CX4H2:2 ] !@[ NX3 ; "N_lp" :3 ][ CX4:4 ]
[ C:1 ][ CX4H2:2 ] !@[ NX3 ; "N_lp" :3 ][ C:4 ]
... ..

TypeTwo:

[ !#1:1 ][ CX4:2 ] !@[ NX3 ; "N_lp" :3 ]
[ C:1 ][ $( S(=O)=O ) :2 ] !@[ "N_lp" :3 ]
... ..
```

The torsions are matched to torsion rules containing 'N\_lp' using specified SMARTS patterns without the 'N\_lp' along with additional constraints using the following methodology:

```
TypeOne:

. SMARTS pattern must contain four mapped atoms and the third
  mapped atom must be a nitrogen matched with 'NX3:3'
```

- . Nitrogen atom must have 3 neighbors. The 'allowHydrogens' parameter controls inclusion of hydrogens as its neighbors.
- . Nitrogen atom and its 3 neighbors must be coplanar. 'planarityTolerance' parameter provides tolerance in degrees for nitrogen to be considered coplanar with its 3 neighbors.

TypeTwo:

- . SMARTS pattern must contain three mapped atoms and the third mapped atom must be a nitrogen matched with 'NX3:3'. The third mapped atom may contain only 'N\_lp:3' The missing 'NX3' is automatically detected.
- . Nitrogen atom must have 3 neighbors. 'allowHydrogens' parameter controls inclusion of hydrogens as neighbors.
- . Nitrogen atom and its 3 neighbors must not be coplanar. 'planarityTolerance' parameter provides tolerance in degrees for nitrogen to be considered coplanar with its 3 neighbors.
- . Nitrogen lone pair position equivalent to VSEPR theory is determined based on the position of nitrogen and its neighbors. A vector normal to 3 nitrogen neighbors is calculated and added to the coordinates of nitrogen atom to determine the approximate position of the lone pair. It is used as the fourth position to calculate the torsion angle.

RotBondsSMARTSMode: SMARTS pattern to use for identifying rotatable bonds in a molecule for matching against torsion rules in the torsion library. Possible values: NonStrict, SemiStrict, Strict or Specify. The rotatable bond SMARTS matches are filtered to ensure that each atom in the rotatable bond is attached to at least two heavy atoms.

The following SMARTS patterns are used to identify rotatable bonds for different modes:

NonStrict: [!\$(\*\*\*)&!D1]-&!@[!\$(\*\*\*)&!D1]

SemiStrict:

[!\$(\*\*\*)&!D1&!\$(C(F)(F)F)&!\$(C(Cl)(Cl)Cl)&!\$(C(Br)(Br)Br)&!\$(C([CH3])([CH3])[CH3])]-&!@[!\$(\*\*\*)&!D1&!\$(C(F)(F)F)&!\$(C(Cl)(Cl)Cl)&!\$(C(Br)(Br)Br)&!\$(C([CH3])([CH3])[CH3])]

Strict:

[!\$(\*\*\*)&!D1&!\$(C(F)(F)F)&!\$(C(Cl)(Cl)Cl)&!\$(C(Br)(Br)Br)&!\$(C([CH3])([CH3])[CH3])&!\$([CD3](=[N,O,S])-[#7,O,S!D1])&!\$([#7,O,S!D1]-@[CD3]=[N,O,S])&!\$([CD3](=[N+])-[#7!D1])&!\$([#7!D1]-@[CD3]=[N+])]-&!@[!\$(\*\*\*)&!D1&!\$(C(F)(F)F)&!\$(C(Cl)(Cl)Cl)&!\$(C(Br)(Br)Br)&!\$(C([CH3])([CH3])[CH3])]

The 'NonStrict' and 'Strict' SMARTS patterns are available in RDKit. The 'NonStrict' SMARTS pattern corresponds to original Daylight SMARTS specification for rotatable bonds. The 'SemiStrict' SMARTS pattern is derived from 'Strict' SMARTS pattern.

TorsionLibraryFilePath: Specify a XML file name containing data for torsion library hierarchy or use default file, TorsionLibrary.xml, available in the directory containing this file.

## AUTHOR

Manish Sud <msud@san.rr.com>

## Collaborator

Pat Walters

## Acknowledgments

Wolfgang Guba, Patrick Penner, and Levi Pierce

## COPYRIGHT

Copyright (C) 2025 Manish Sud. All rights reserved.

This module uses the Torsion Library jointly developed by the University of Hamburg, Center for Bioinformatics, Hamburg, Germany and F. Hoffmann-La-Roche Ltd., Basel, Switzerland.

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.