Recent changes to optking

Dr. Rollin King

Nov. 2014

Major changes

1. Support for optimization in cartesian coordinates or linear combinations of simple coordinates.

OPT_COORDINATES may set to:
REDUNDANT and **INTERNAL** are synonyms and the default.
DELOCALIZED are the coordinates of Baker.
NATURAL are the coordinates of Pulay. (coming soon; half done)
CARTESIAN uses only cartesian coordinates.
BOTH uses both redundant internals and cartesian coordinates.

2. A systematic exception handling for failing optimizations has been implemented.

I call this **DYNAMIC_LEVEL** and represent it by a whole number (presently 0-7). The current default is 0, which implies the non-dynamic algorithm (once 0, always 0). For higher levels, an increasingly more conservative algorithm is applied as necessary. Level 1 may soon become the default setting, if it can be confirmed that there is no loss of performance for easier cases.

a. If internal coordinates become poorly defined (e.g., 180 degree angles within a dihedral or newly 180 degree bends), the optimization is restarted at that point with new coordinates at the same level.

b. If a 'bad_step' occurs, e.g., if the energy increases in a minimization, then optking will go backwards along the previous step to try to find a lower energy point. It will try **CONSECUTIVE_BACKSTEPS** times, before giving up.

c. If optking is unable to converge the backtransformation to Cartesian coordinate displacements, and the first approximate step (generated with a static B matrix) is very large, it will give up immediately.

d. When the current level "gives up", the **DYNAMIC_LEVEL** is raised by one, and the optimization is restarted (new coordinates and guess Hessian) at the current point.

Minor changes

The **PRINT_TRAJECTORY_XYZ_FILE** option will produce a text output file (xyz format) containing each step for minima and transition-state optimizations, and each converged point along the reaction coordinate for IRC computations.

In the RFO method, root-following during the RS-RFO (iterative step restriction) algorithm improves cartesian optimizations particularly, and is now the default.

PRINT_OPT_PARAMS prints only the optimization parameters.

In libmints symmetrize(), I replaced an abort() with a throw() when symmetry is broken (geometry cannot be symmetrized). This allows optking to handle broken symmetry exceptions as it wishes. It is unclear in general, if an optimizing user wants to follow lower symmetry minima; we could make it a user option.

Changed Hessian update algorithm to, when selecting past structures with which to update, pass over ones that are too close in structure to the current one. Looks back as far as necessary to find HESS_UPDATE_USE_LAST number of distinct structures to use.

Some sample input

```
# redundant internals with RFO – converges in 7 steps
molecule hooh {
 0 1
  H
  O 1 1.0
  O 2 1.5 1 110.0
  H 3 1.0 2 110.0 1 120.0
}
set { basis cc-pVDZ }
optimize('scf')
```

```
# converges in 6 steps with delocalized coordinates specified
set optking { opt_coordinates = delocalized }
```

```
# converges in 43 iterations with xyz + internals using steepest-descent
set optking {
 step_type = sd
 opt_coordinates = both
}
```

```
# converges in 6 steps with initial, exact Hessian
set optking{ full_hess_every 0 }
```

```
# Freeze the OH bond distances at their starting value (similarly for frozen_bend
and frozen_dihedral; white space is irrelevant except as separating values.
set optking {
  frozen_distance = ("
    1  2
    3  4
  ")
}
```

```
# Impose a constraint not present in the starting structure.  Note the difference
# between 'frozen' and 'fixed' (similarly, for fixed_bend and fixed_dihedral).
set optking {
  fixed_distance = ("
    1  2 0.950
    3  4 0.950
  ")
}
```

frozen and fixed do not currently work properly for combination coordinates. For example, the behavior of

```
    opt_coordinates = delocalized
    frozen_distance = ("2 3")
```

is unpredictable. Fixing this does not seem urgent to me.