Xopt - an Xternal OPTimizer v.0.8b

Holger Kruse(mail2holger@gmail.com)

July 21, 2014

1 Introduction

The purpose of this program is to read the energy and gradient of a program (quantum-chemical, semi-empirical or even force field programs) and to optimize the structure with a hopefully very efficient algorithm. The interface is kept reasonable simple while still allowing many modifications. Since the code is open-source, adjustment of any feature to one needs is possible and encouraged.

As the program is still under heavy development the manual might be incomplete/incorrect at times.

2 Overview of the Features

Table 1: List of supported programs

- Turbomole 6.X
  - using `ridft`, `dscf` and `ricc2` modules.
  - _huge binaries are also supported.
  - combination with the external programs `dftd3` and `gcp`.
- ORCA
- mopac12
  - combination with the external program `dftd3` .
- Amber12 (`sander` module)
- Gaussian09
- (more support added as needed)

Table 2: Available optimization algorithms

- conjugate gradient
- QN-relaxation
  - rational function optimization (RFO) and size-independent RFO (SI-RFO).
- Hessian updates:
  - BFGS
  - SR1
  - SR1-BFGS
  - SS-BFSG (spectral scaling)
- coordinate systems
  - cartesian coordinates
  - approx. normal coordinates
- Initial Hessians
  - unit Hessian
- Constraints/Restraints
  - Cartesian constraints (fixing atom positions in 3D space)
  - primitive restraints (restraining potential on internal coordinates)
- work in progress
  - ONIOM optimizations
  - transition state search
  - initial model Hessian

3 Commandline Options

The program itself is called as: `xopt <structure> [-options]`
and prints to standard output. `<structure>` refers to either a xyz-file (angstrom) or a Turbomole file (tmol format, bohrs).

`-tm / -tmhuge`
Run Turbomole, run the _huge binaries variant.
prepare a standard input using define.

`-orca`
Run ORCA. Prepare an input file orca.in according to section 4.1.

`-g09`
Run Gaussian09. See section 4.5 for details.

`-mopac`
Run mopac. See section 4.4 for details.

`-gcp <level>`
Run Turbomole in combination with the `gcp` program for gCP-correction. Prepare input for Turbomole as usual (section 4.2 for details) and give the level for gcp (see gcp manual).

`-d3 "<func + damping>"`
Run Turbomole in combination with Grimme's `dftd3` program. See `dftd3` manual for input specifics. The version of the damping has to be given! Eg. -d3 "tpss -bj". The "" are necessary.

`-amber`
Run sander module of Amber12 suite. Prepare *amber.in* and *amber.top* files as you would for sander. Further details in section 4.3.

`-apbs`
Not working correctly, sry.

`-rhess / -restart`
Read initial Hessian from *xopt.chess* (turbomole format)

`-restart`
Read Hessian and restraints from *xopt.restart* to restart a calculation. Geometry is not read!

`-cart`, `-cart-si`, `-anc`,`-anc-si`,`-anc-cg`,`-cgrfo`,`-int`
Coordinate system and step control.

`-sg1-bfgs`,`-sg1`,`-bfgs`,`-ss-bfgs`,`-ss-sg1-bfgs`
Hessian update method

`-c <integer>`
Max. number of optimization cycles.

`-econv <real>`
energy change convergence threshold

`-gconv <real>`
gradient norm convergence threshold

`-dpl <real>`
Maximal displacement per step in bohrs.

`-amber-pb`, `-amber-gb`
Predefined input for amber. Stills needs a provided topology file *amber.top*

4 Preparing Specific Inputs

xopt cannot handle the input preparation for every program. Thus, the following sections describe how xopt expects the input for the various programs. This is not necessarily user friendly.

4.1 ORCA

Requirement: ORCA binaries
input file: orca.in
It is necessary that the following input lines are included:

```
! ENGRAD
*xyzfile <chrg> <mult> xopt.xyz
```

where `<chrg>` is the molecular charge and `<mult>` the multiplicity.

4.2 Turbomole

Prepare a standard input using define. Supported are HF/DFT calculations using `ridft` and `dscf`, and calculations using the `ricc2` module (eg. for MP2.). A 'coord' file is automatically written and updated.

4.3 Amber

Requirement: `sander` Input file: *amber.in* (`sander` input) and *amber.top* (amber topology file). sander needs to write the forcedump.dat file (see AMBER manual).

4.4 Mopac

Requirement: `gcx` conversion tool (will be integrated soon). mopac binary called `mopac2012`
input file: A mopac SETUP file (named *SETUP*) with the following keywords

```
1SCF GRAD XYZ NOMM PRECISE aux(42,PRECISION=9,MOS=-99999,COMP)
```

Dont forget to add the method, charge, etc.!

4.5 Gaussian09

input file: g.in
Requirement: special *run-g09* script to execute Gaussian.(not included)

5 Using *xopt.control*

Instead of using the command line it is possible to write a control file for `xopt` named *xopt.control*. Its main purpose lies in providing parameters for the constrained or restrained optimization.

5.1 Restraints and Constraints in Optimizations

The implementation in `Xopt` is limited: Constraints are only possible for Cartesian coordinates, while restraints can be defined for internal (primitive) coordinates, e.g. bond lengths and angles. Of course, a very restrictive restraint on an internal coordinates leads to a practical constraint. Thus, using high values for the restraining potential, also internal coordinates can be 'constrained'.

The restraining potential is in the form of a simple, quadratic function with the restraining energy R being defined as

$$R(x) = k_r \cdot (p - p_{ref}) \ ,$$

with k_r as the force constant of the restraint, p the type of internal (primitive) coordinate (bond, angle or torsion) and its reference value p_{ref}.

The actual value of k_r needs to be chooses carefully and is structure and level of theory specific. The reference value x_{ref} is taken from the initial input geometry (or from the xopt.restart file if requested).

xopt checks for the file "xopt.control" at the startup. Its content will overwrite any commandline options!

```
legend:
i,j,k,l = <integer>, atomic numbers starting from 1
F = <floating point> eg. "1.0" NOT "1"

Restrained optimizations:
 $restr
 bond i j F
 angle i j k F
 dihed i j k l F
```

An additional gradient from the restraining potential (G_r) will be added to the total gradient: G_{total}: G + G_r. gnorm and all other values are calculated using G_{total} ! Thus is it better to rely on energy convergence for restrained optimizations.

```
Cartesian constrains:
 $freeze
 Hopt             #just optimize hydrogen positions, freeze the rest
 atom i           #freeze atom number i
 elem <string>    #freeze all elements <string> (lower case !!)
```

For constrains the gradient components (and respective entries in the B-matrix) are set to zero.

6 General Notes

The current default is: RFO with SR1-BFGS hessian update, which seems so far the most robust algorithm. Currently, the initial Hessian consists of a unit hessian, which is of course not ideal for QN-methods. A model Hessian implementation is under development.

If you find bugs or have suggestions, please let me hear about them!

Good luck & Don't panic!