This example is a novel application of LMOD for a different kind of large scale, production quality loop search on the influenza hemagglutinin HA1 in complex with a neutralizing antibody CH65 (3SM5, http://www.pnas.org/content/110/1/264). The key factor in this protein-protein complex system is docking via the 22-residue long J:96-117 (CDR H3) loop in the CH65 Fab fragment. The first step in this protein-protein docking problem is to explore all feasible loop conformations of J:96-117 and this task goes way beyond the difficulty of the `lmod_beta_secretase` loop optimization example.

The key to achieve this goal is the built-in ZIG-ZAG algorithm that is described in the AmberTools documentation under NAB: Molecular Dynamics and Mechanics/Low-MODe (LMOD) optimization methods. Basically, a single LMOD move inherently involves excessive bond stretching and bond angle bending in Cartesian space. For illustration see the movie file `CH65_Fab_low-modes_movie.mpeg` showing the first 20 low-frequency modes of the loop vibrations. Benzene rings in the side chains especially manifest the excessive deformation due to the linear nature of first approximation of low-mode perturbation. The primarily torsional trajectory drawn by the low-modes of vibration on the PES is severely contaminated by this naive, linear approximation and, therefore, the actual Cartesian LMOD trajectory often misses its target by climbing walls rather than crossing over into neighboring valleys at not too high altitudes. The current implementation of LMOD employs a so-called ZIG-ZAG algorithm, which consists of a series of alternating short LMOD moves along the low-mode eigenvector (ZIG) followed by a few steps of minimization (ZAG), which has been found to relax excessive stretches and bends more than reversing the torsional move. Therefore, it is expected that such a ZIG- ZAG trajectory will eventually be dominated by concerted torsional movements and will carry the molecule over the energy barrier in a way that is not too different from finding a saddle point and crossing over into the next valley like passing through a mountain pass. *Using the ZIG-ZAG algorithm, the low-mode move can be extended significantly, **passing over multiple energy valleys**, and still keeping the energy relatively low.* Thus, applying e.g. a constant 50 ZIG-ZAG steps along a single low mode regardless whether or not a barrier crossing event has been detected, significant loop backbone movement can be achieved and this procedure can be used to scan the entire conformational space available to the loop.

The results are displayed in a jpeg image (`CH65_Fab_LMOD_long-range-scan.jpg`) and an mpeg movie (`CH65_Fab_LMOD_long-range-scan_trajectory.mpeg`) which shows very vividly the extent of loop movement generated by LMOD. Note that although it very much looks like it, this movie IS NOT a molecular dynamics trajectory, it is simply a succession of loop conformations visited by 50 steps of LMOD simulation. The ensemble of 127 different loop conformations are stored in `CH65_Fab_lmod_X-ray_conf_1458846784.pdb` and the 51 frames of the LMOD trajectory are recorded in `CH65_Fab_lmod_trajectory_X-ray_1458846784.pdb`. (The individual conformations/frames in separate files were deleted to save disk space.)

To run this job, one simply needs to compile `CH65_Fab_lmod_X-ray.nab` (which will read `CH65_Fab_X-ray_4lmod.pdb` and `CH65_Fab_X-ray_4lmod.prmtop`). It is highly recommended to compile it in parallel using mpinab, and run it it on 10+ cores. 50 LMOD iterations with 500 total LMOD search steps (each including 50 ZIG-ZAG moves) took about 44 hours running on 8 cores of a 3.6 GHZ i7 processor. In general, 50-100 LMOD iterations are recommended for seriously exploring a protein system like this, which definitely requires parallel execution with as many cores as there are available. LMOD scales quite well thanks to the efficient force parallelization engine in AmberTools. The job log can be found in `CH65_Fab_run_lmod_X-ray.log`.

The structure of a typical LMOD nab script and associated job logfile is fully explained in the main AmberTools documentation under NAB: Molecular Dynamics and Mechanics/Low-MODe (LMOD) optimization methods, here some specifics are pointed out.

1. The general LMOD workflow for proteins includes i) fully flexible minimization of the entire system, ii) applying LMOD search on the partially frozen system meaning that the low frequency modes are those associated with the flexible (specified as "moving") loops in the presence of the external field (electrostatic, van der Waals, and solvation terms) exerted by the frozen atoms, and iii) re-minimizing the low-energy loop conformations after unfreezing the whole system. (In this example iii) was not actually performed, only single point unfrozen energies were calculated.)
2. It is good practice to use a hardware generated seed for the random number generator and save this seed value in the file names associated with an LMOD job. In the example script the number of seconds passed since zero hour, January first, 1970 is used for this purpose.
3. Noteworthy LMOD parameters: `lo.nmod=20` is the number of lowest frequency modes used, `lo.kmod=5` means that out of the 20, every new LMOD iteration explores 5 randomly selected modes (and follows them in both directions), and `lo.nrotran_dof=0` refers to the presence of frozen (or tethered) atoms. `lo.energy_window=50` is set intentionally very high to generate a variety of different loop conformations and not only focus on the lowest energy ones.

   NOTE: This type of LMOD loop scan requires that subsequent, focused LMOD searches will be run starting from multiple loop conformations carefully selected from the entire scan ensemble, in order to find the lowest-energy loop conformations that can be viable candidates for protein-protein docking.

   `lo.conf_separation_rms=2.0` means that in the final set of LMOD generated loops every single pair of them will be at least 2.0 Å superposition RMSD apart. This RMSD calculation includes all moving atoms (LMOD has no knowledge of atom types, etc.). This parameter is quite useful in controlling the diversity of low-energy loops, or conformations in general.
   `lo.nof_lmod_steps=50` instructs LMOD to skip trying to determine barrier crossing events automatically (see documentation) and to apply a constant 50 ZIG-ZAG steps (see above). This is the key to allow for fairly long moves on the PES resulting in significant flexing of the loop backbone. For example, something like this in the logfile

```
18/50 E = -900.950 ( 0.061)  Rg = 3.300  rmsd= 1.041  p= 1.000
18/50 E = -900.423 ( 0.081)  Rg = 3.604  rmsd= 4.047  p= 1.000
```

   means that starting from a particular conformation (E = -895.834), low-frequency mode #18 is explored taking 50 ZIG-ZAG curvilinear perturbation steps in both directions. After minimization, the resulting new loop conformations are, respectively, displaced by 1.041 and 4.047 Å RMSD from the same starting conformation. Rg is the radius of gyration (of the moving atoms) and p is the Boltzmann probability of a conformation with respect to the starting conformation in that particular LMOD iteration. Note that both extended LMOD moves found lower energy loop conformations than the starting energy (E = -895.834), but these two loop conformations are significantly different measured by the RMSD.
   `lo.mc_option=1` sets the Monte Carlo search to "Metropolis" which means that the traditional Metropolis criterion is applied to the minimized energies. Note that the temperature is set to an ultra high value `lo.rtemp=25.0` which means that a very high energy gap $E - E_0$

= 25 [kcal/mol] still has a significant Boltzmann probability of $e^{-1} \approx 37\%$. This is ideal for this type of LMOD scan but, of course, the secondary searches required to quench the selected loop conformations (see **NOTE** in bold above) should use either a low temperature such as `lo.rtemp=1.0` or one of the quenching options `lo.mc_option=2/3` .

With any questions or comments please contact me at istvan@kolossvary.hu or ikolossv@bu.edu.