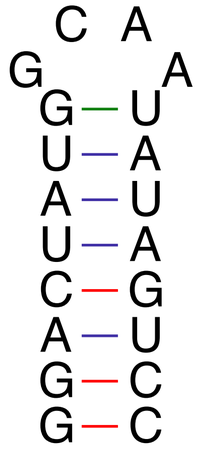
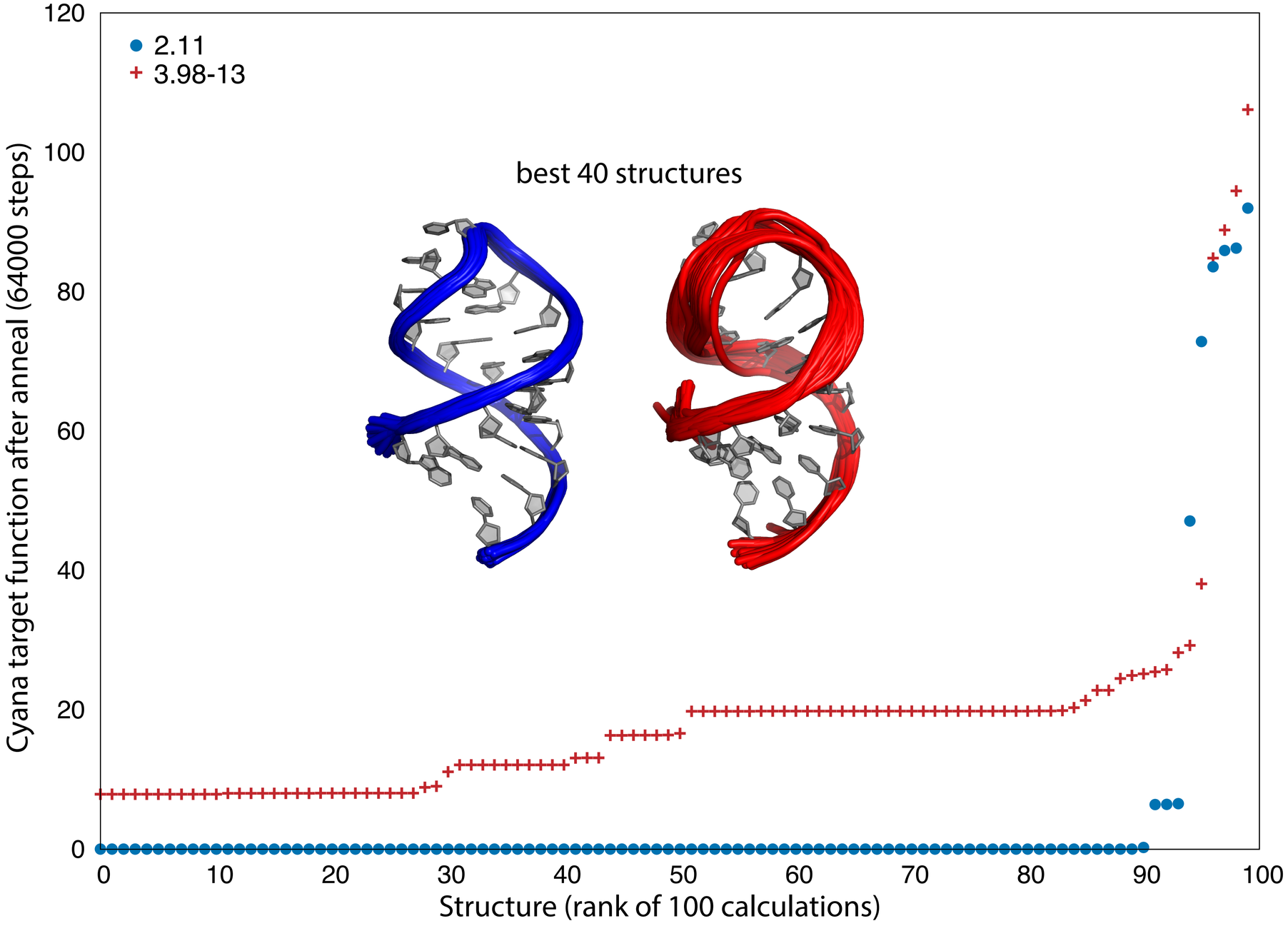
We see better convergence and lower target functions using Cyana 2 compared to Cyana 3

Illustrated here with a small hairpin:



Target function calculation is the same in both versions - if we load a structure generated using v2 into v3, it reports the same target function



A large contributer to these differences seems to be in how the van der Waals interaction pair list is populated

In version 2, vdW lower distance bounds are always active for sequential atoms, even if they are outside the cutoff distance when the list is updated

In version 3, this is only true for sequential backbone atoms

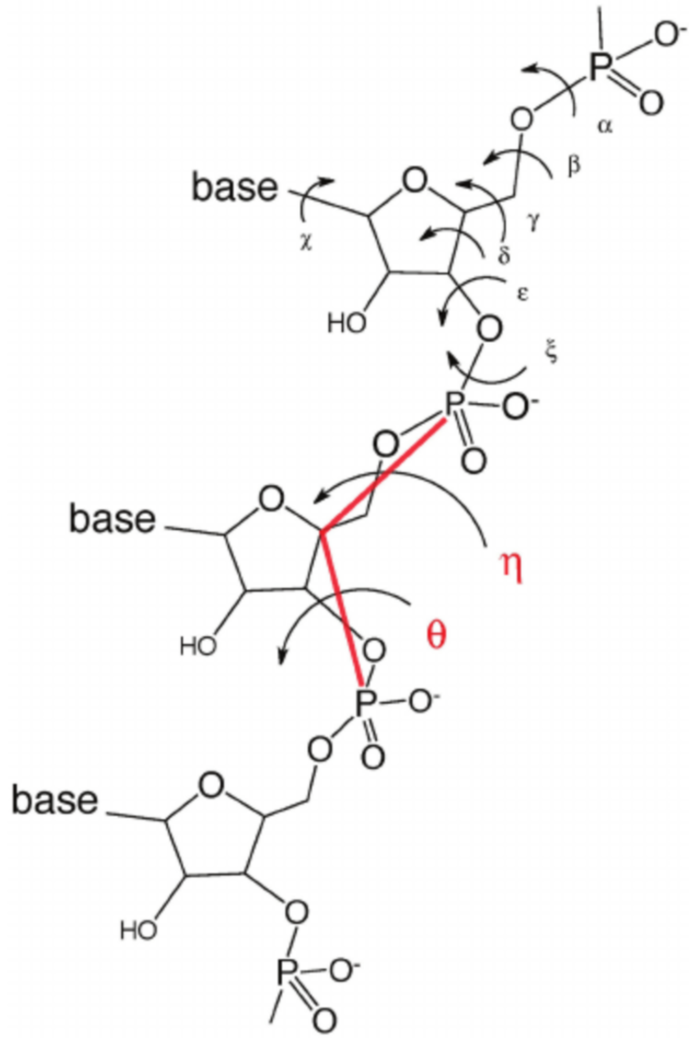
Seems to be unintended behavior in version 2:

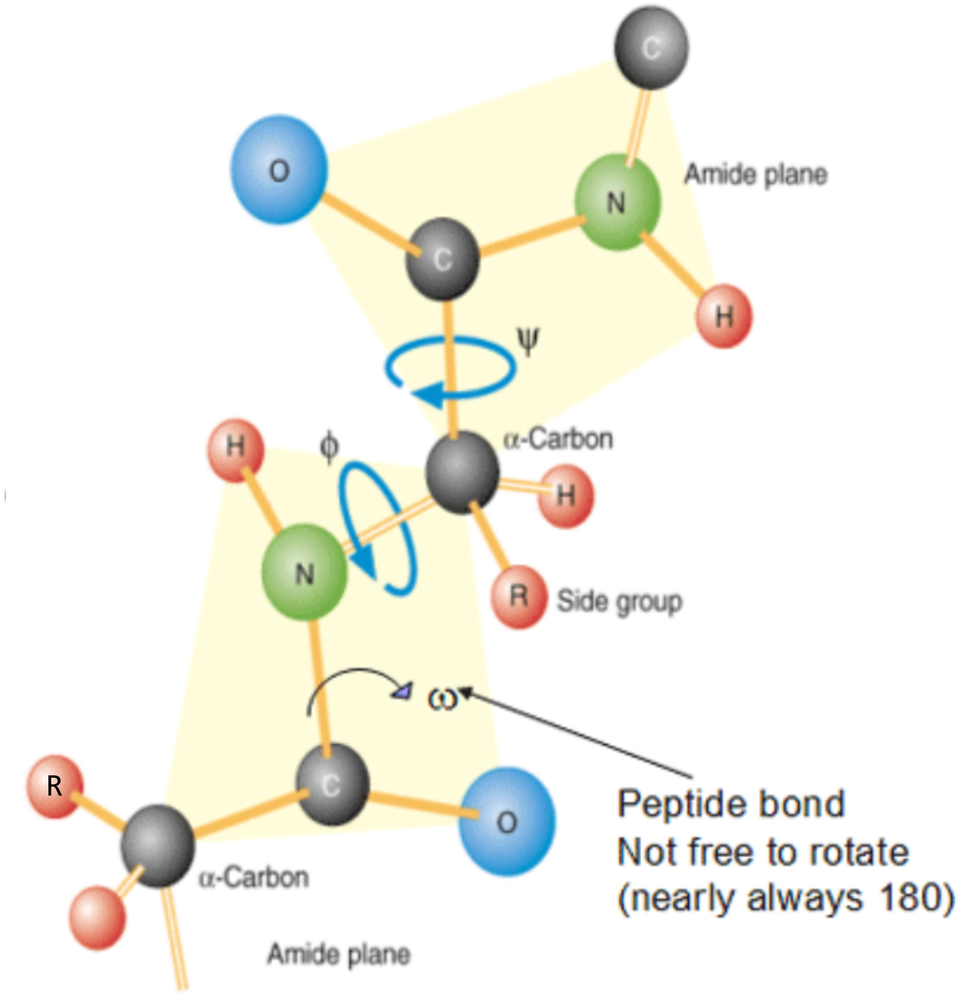
```
      iarb(1:na)=iar(1:na)
cgfortran    where (lda(iaunit(1:na)).le.na) iarb(1:na)=iar(1:na)*nr2
      do i=1,na
        if (lda(iaunit(i)).le.na) iarb(l)=iar(l)*nr2
      end do
```

Should read `i` - `l` is the total number of torsion angles in the loaded molecule

`iarb(i)` is the residue index of backbone atoms, or a big number, and is checked when setting sequential van der Walls lower distance bounds

Perhaps it's reasonable that sequential nucleic acid sidechains would be more likely to clash between updates than protein sidechains?

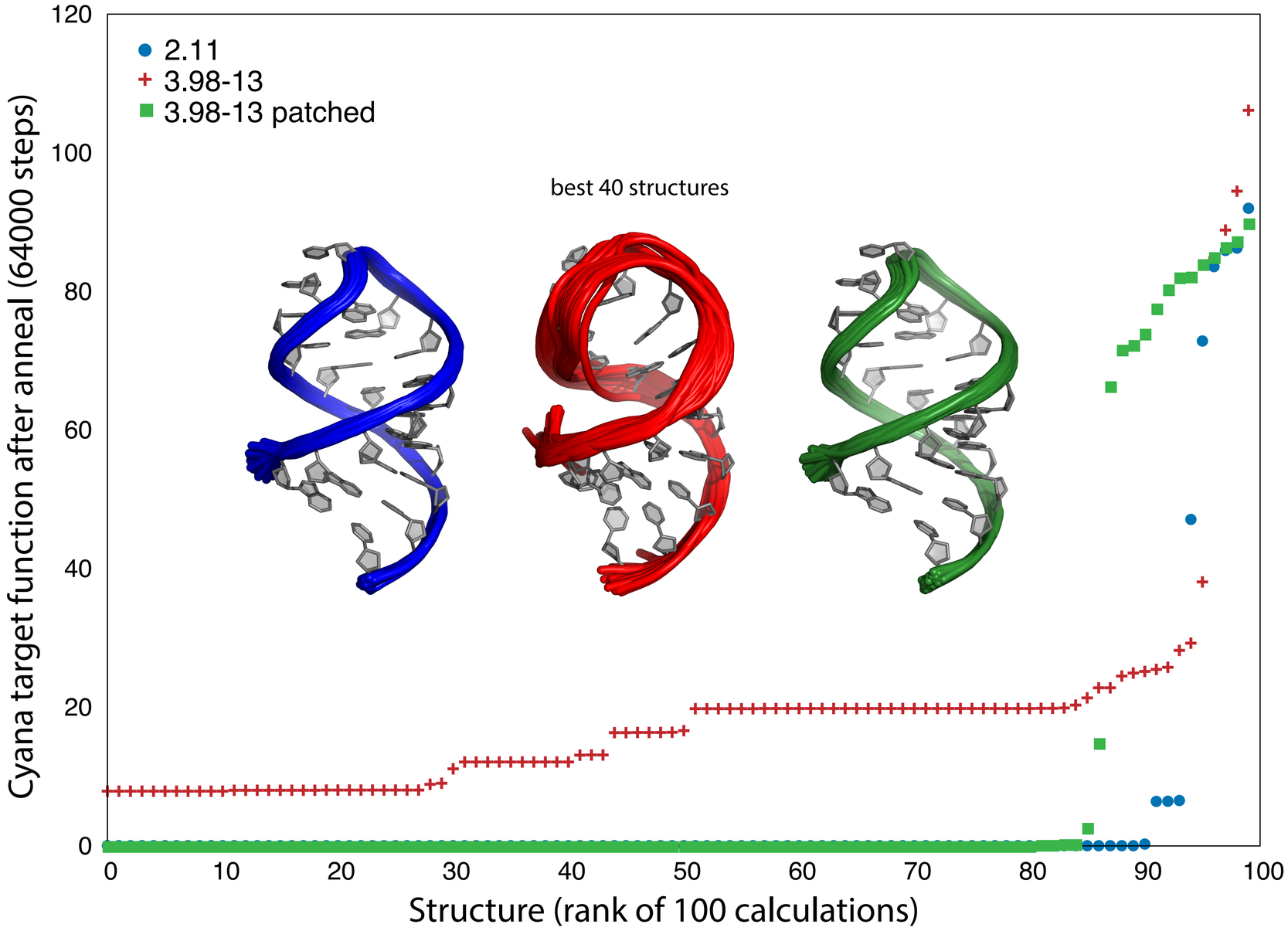




To test, we patch the `iarb` array in memory while running v3, so that it matches v2 (i.e. all atoms are treated as backbone atoms, aside from the atom whose index matches the number of torsion angles in the molecule)

The results are not identical, but much improved.

All of the remaining differences seem to be down to the changes in `md` - everything else I've looked at seems to be perform identically with the patch applied.



We wondered whether updating the van der Waals pair list more frequently would prevent this problem

We independently tested setting `vdwupdate=1` for the `minimize` and `md` steps of the annealing protocol

Most sensitive during the `md` phases

(Very similar results when setting `vdwupdate=1` for both)