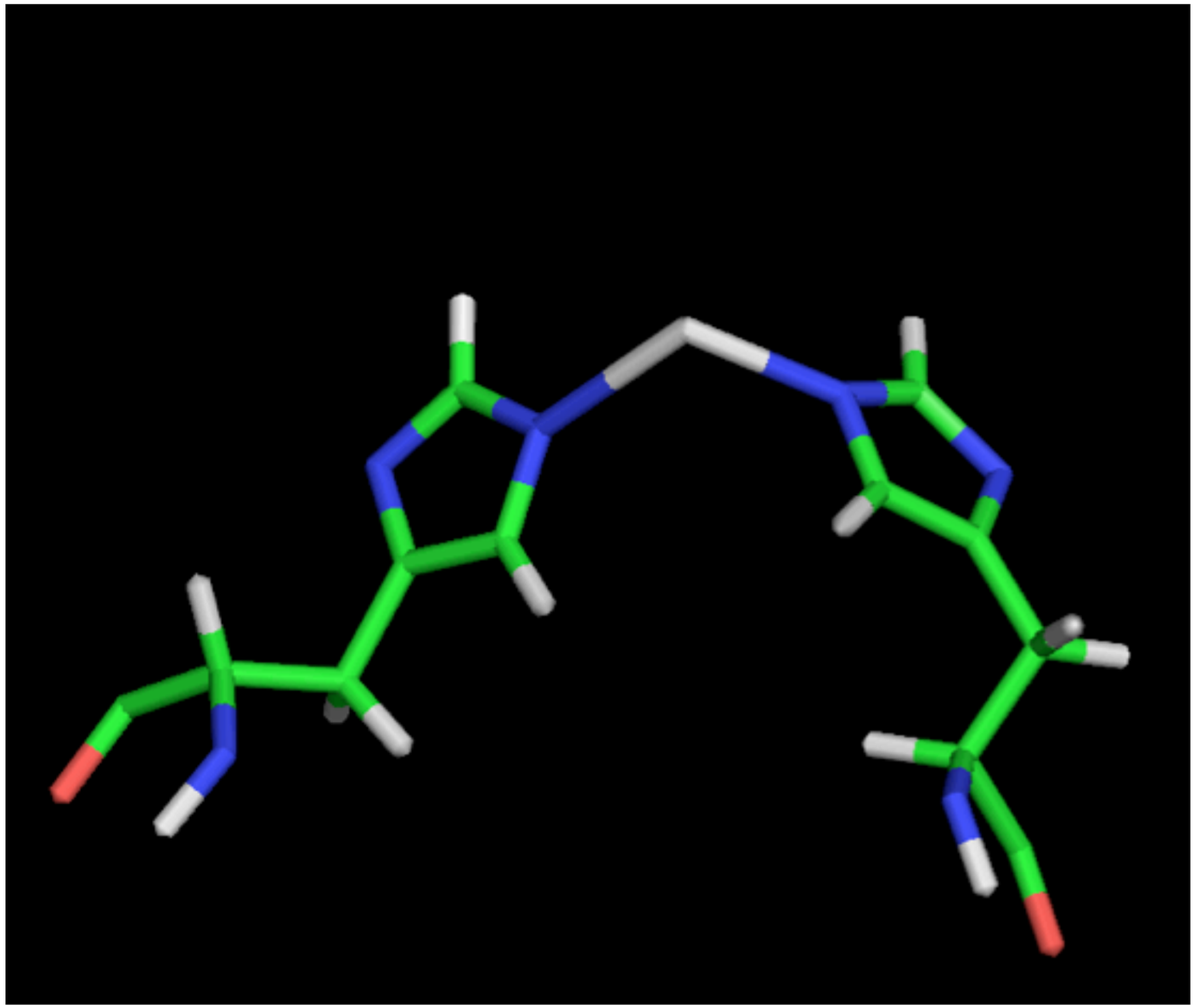
Not My Job: Matching
(and baby pictures)

Will Sheffler
Baker Lab

at the start: metal binding site issues

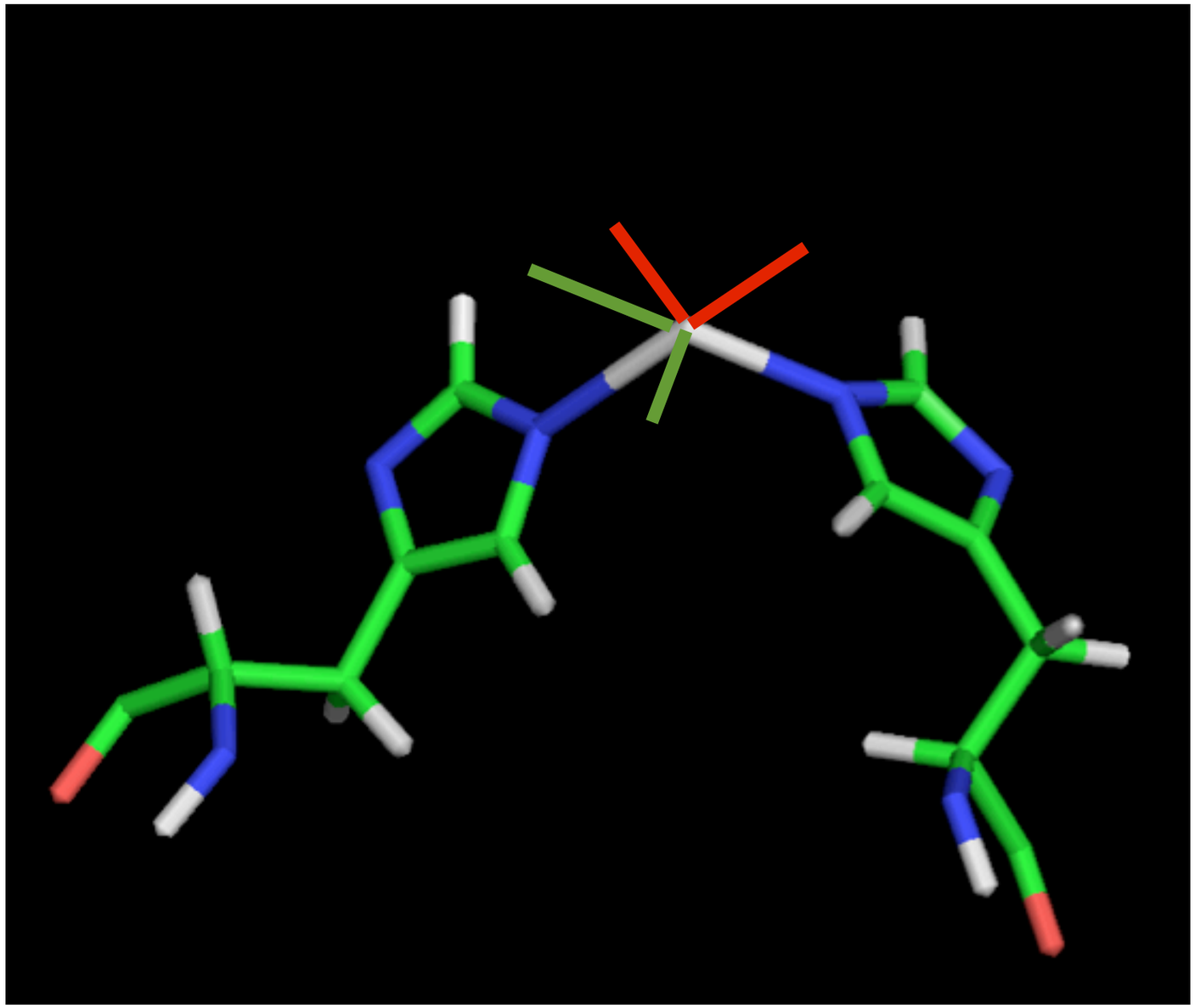


requires secondary matching or many placements

- because HIS - metal bonds can rotate

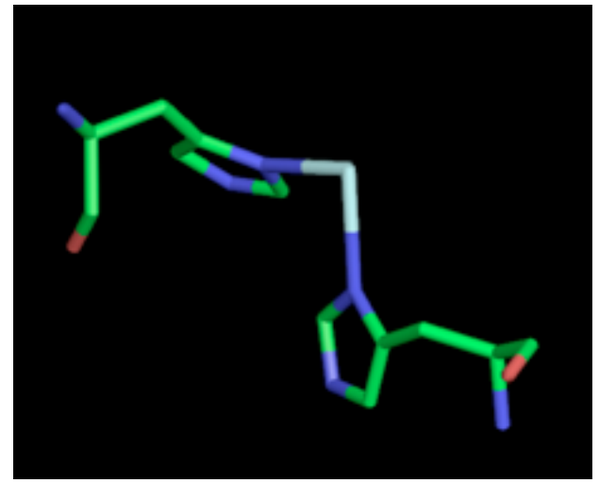
can’t ask for multi-residue constraints (like bond angle above)

at the start: metal binding site issues



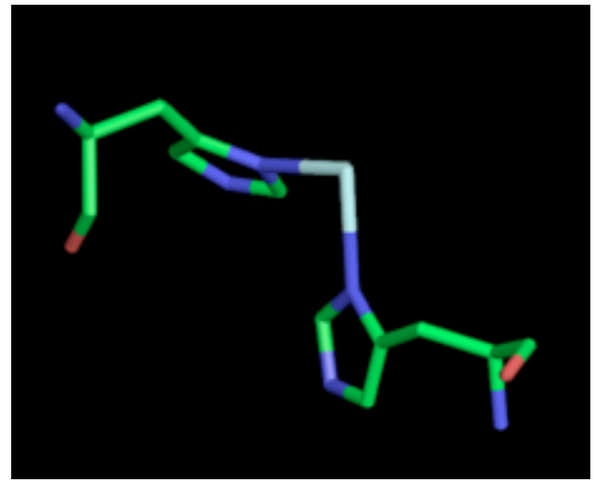
also, in this case, you miss most matches
because of coordinate frame rotation mismatch

enumerative matching



```
foreach bb position:
    foreach his1_chi1:
        foreach his1_chi2:
            foreach bb position:
                foreach his2_chi1:
                    foreach his2_chi2:
                        if his1 & his2 form a metal clamp:
                            HIT!
```

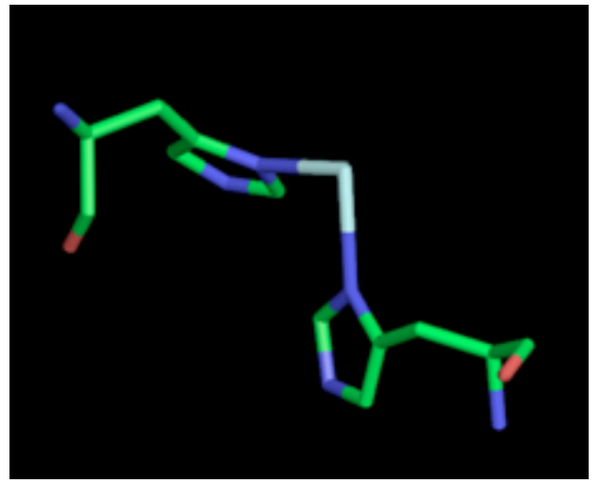
enumerative matching



```
foreach bb position:
    foreach his1_chi1:
        foreach his1_chi2:
            foreach bb position:
                foreach his2_chi1:
                    foreach his2_chi2:
                        if his1 & his2 form a metal clamp:
                            HIT!
```

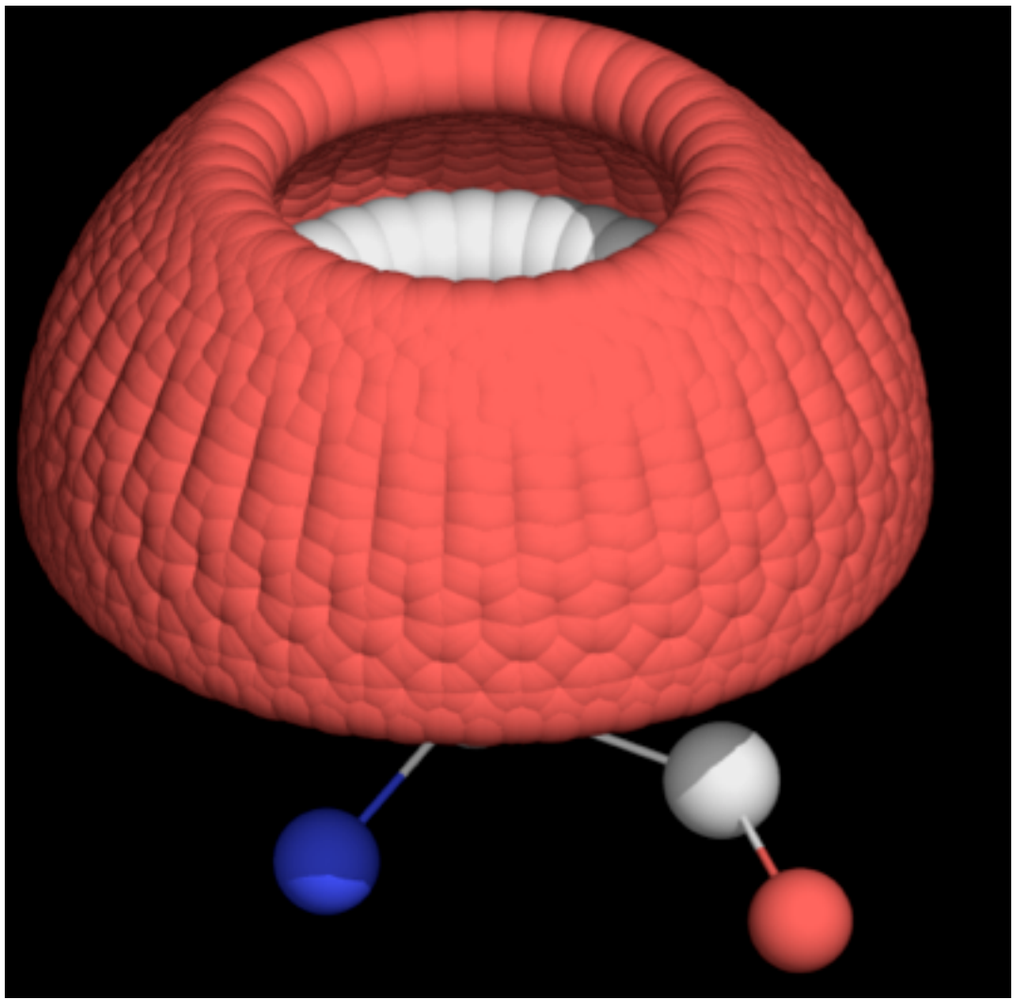
but isn't this horribly slow?

branch & bound



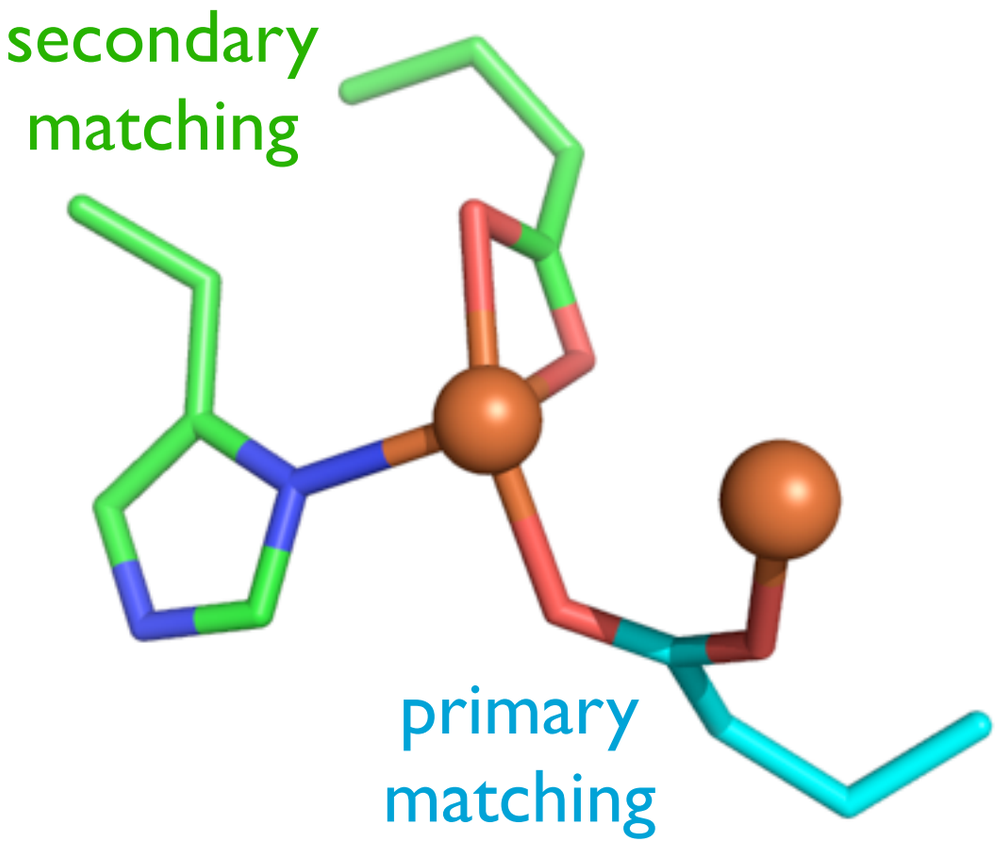
```
foreach bb position:
    foreach his1_chi1:
        if impossible: break
        foreach his1_chi2:
            if impossible: break
            foreach bb position:
                if impossible: break
                foreach his2_chi1:
                    if impossible: break
                    foreach his2_chi2:
                        if his1 & his2 form a metal clamp:
                            HIT!
```

branch & bound: if impossible: break



locus of oxygen positions: Asp

Comparison with RosettaMatch



RosettaMatch Setup:
primary matching on 1 residue
secondary matching on 2 residues

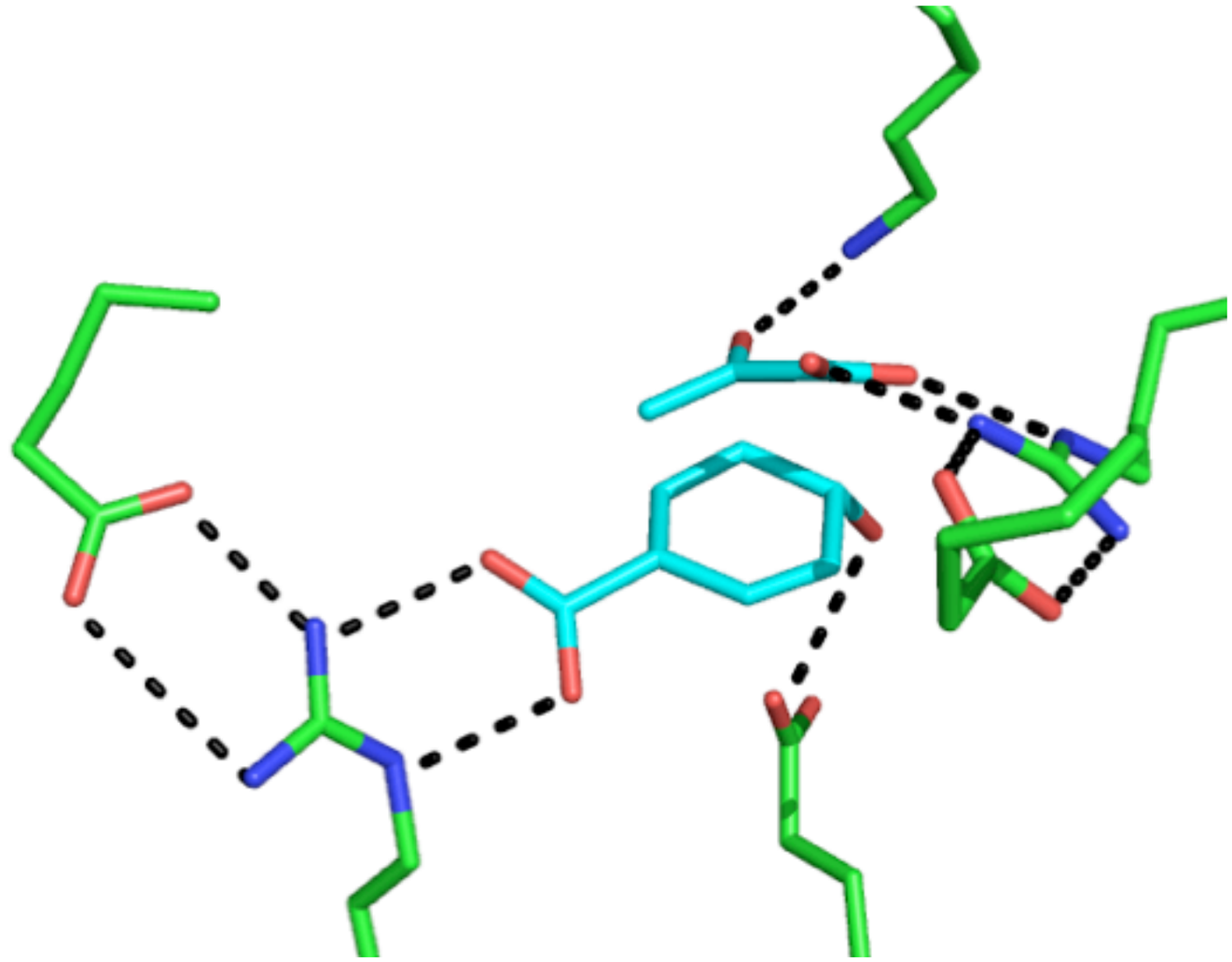
(unfixed DOFs require secondary matching or many many placements)

RESULTS

RosettaMatch:
50 matches
600sec runtime
2gb memory

B&B Enumerative Matching:
3063 matches
109s runtime
300mb memory

What about a bigger theozyme?



2x Arg: bidentate interaction w/ ligand
2x Asp/Glu: backing up Arg
1x Asp/Glu: rotatable hbond to ligand
1x Lys: rotatable hbond to ligand

6 residues
up to 24 rotatable angles!

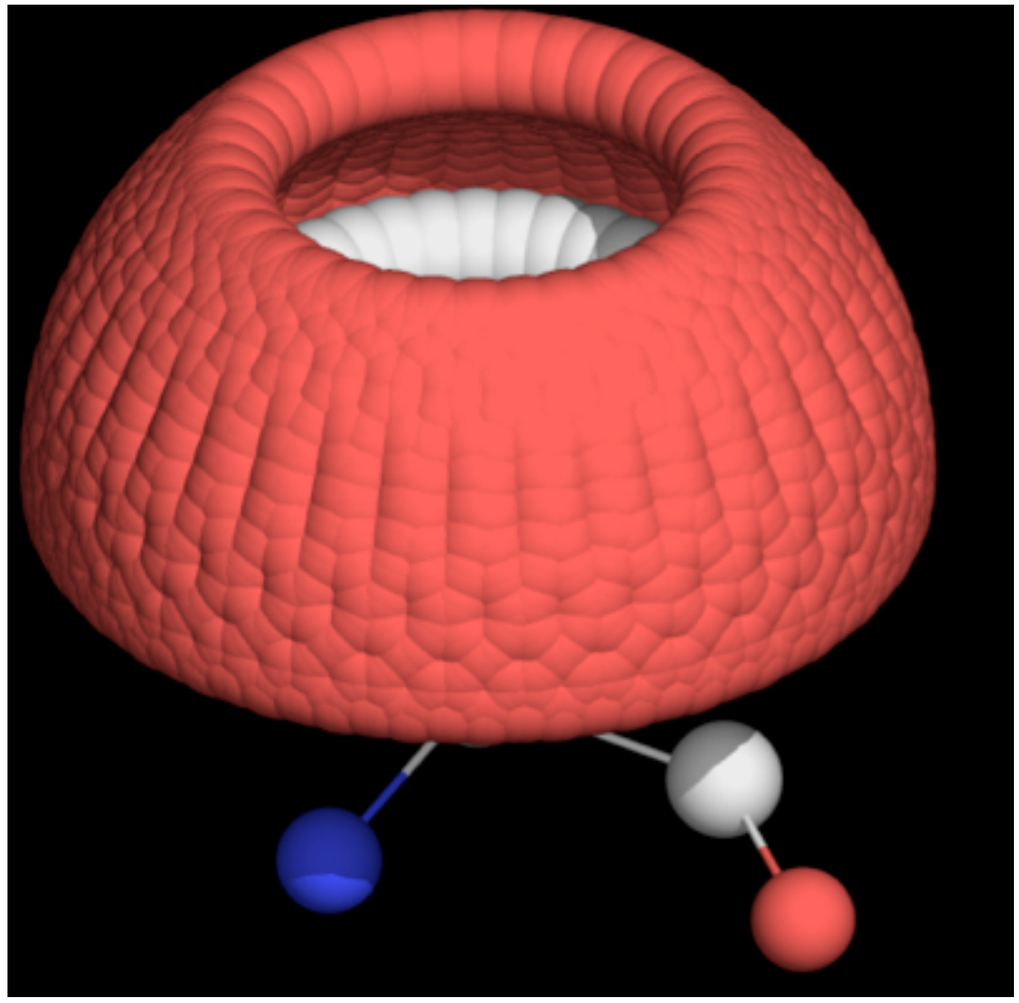
thanks to Florian Richter!

bigger theozyme?

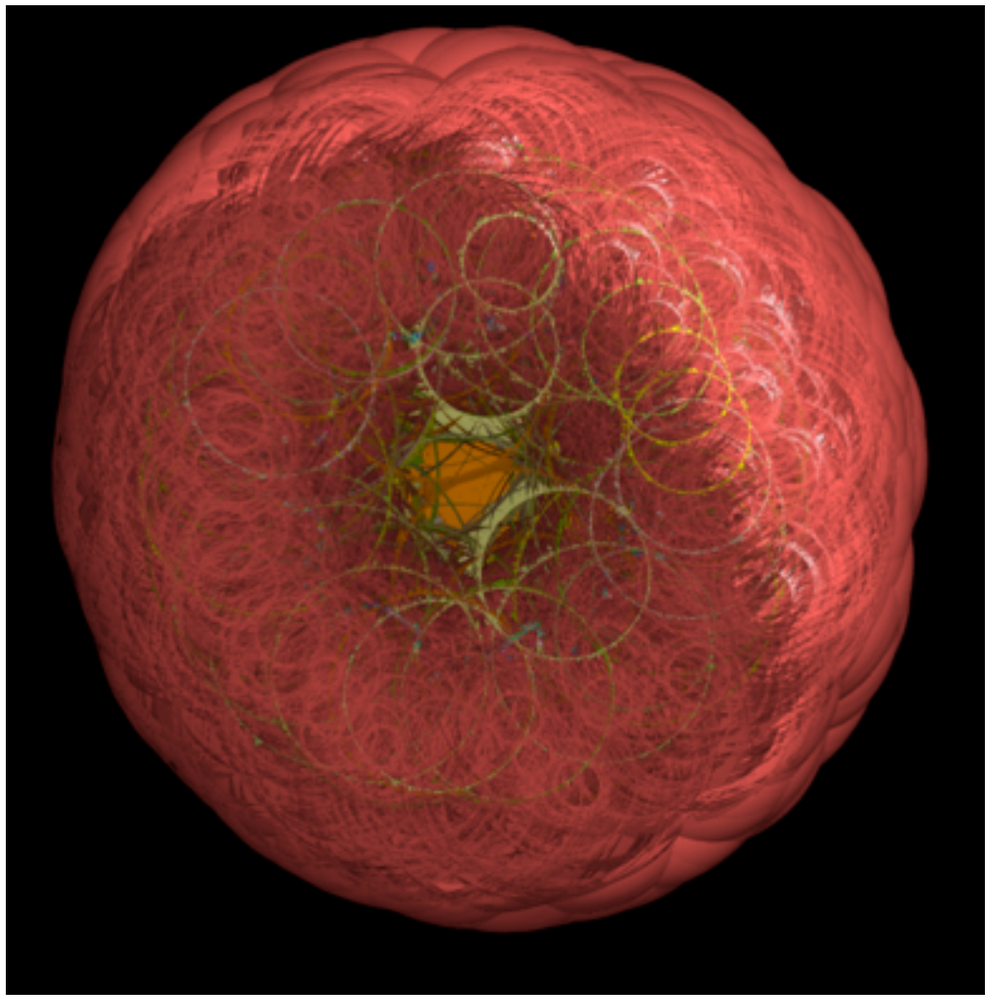
you're doing it wrong

What about a bigger theozyme?

for > 2 rotatable bonds (chi or other), enumerative method doesn't work



Asp

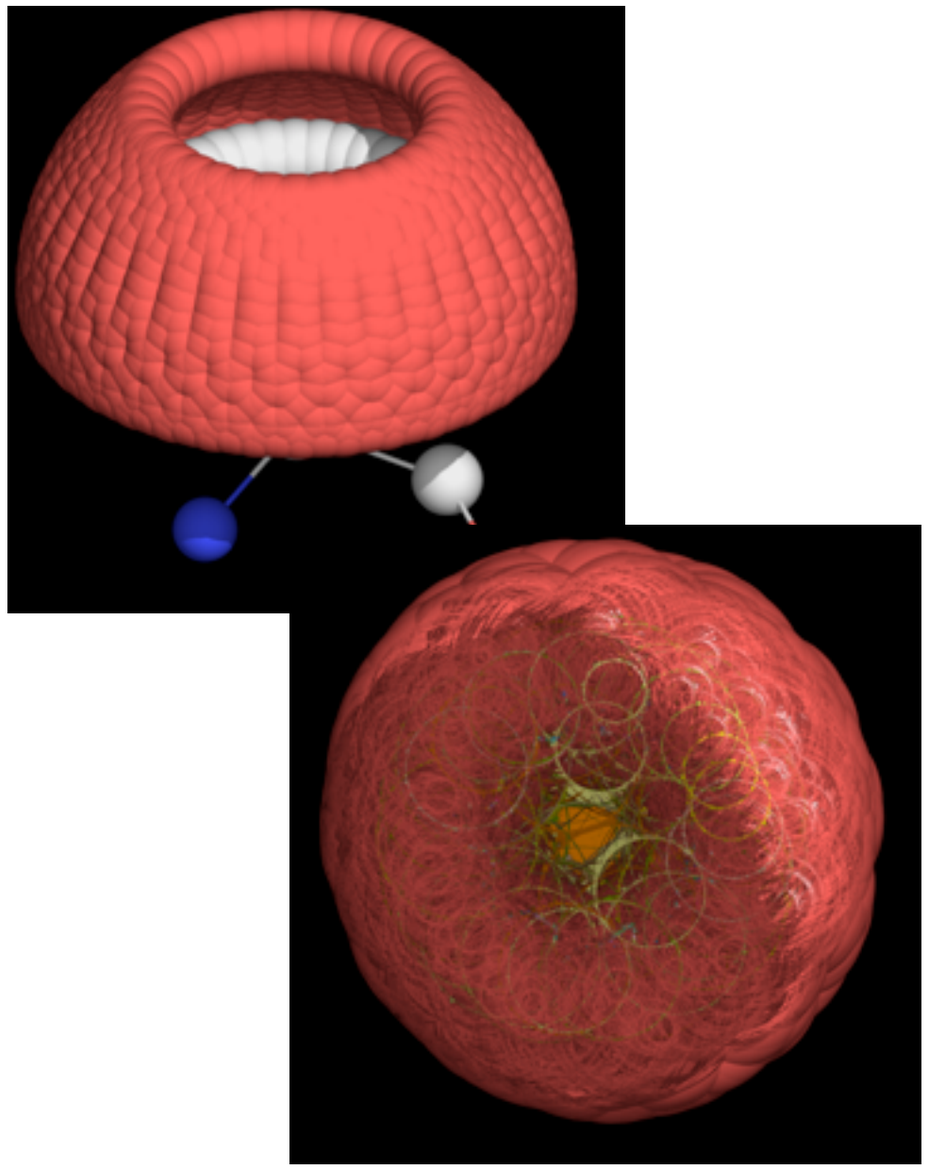


Glu

What about a bigger theozyme?

for > 2 rotatable bonds (chi or other), enumerative method doesn't work

```
foreach bb position:
    foreach his1_chi1:
        if impossible: break
        foreach his1_chi2:
            if impossible: break
            foreach bb position:
                if impossible: break
                foreach his2_chi1:
                    if impossible: break
                    foreach his2_chi2:
                        if his1 & his2 form a metal clamp:
                            HIT!
```



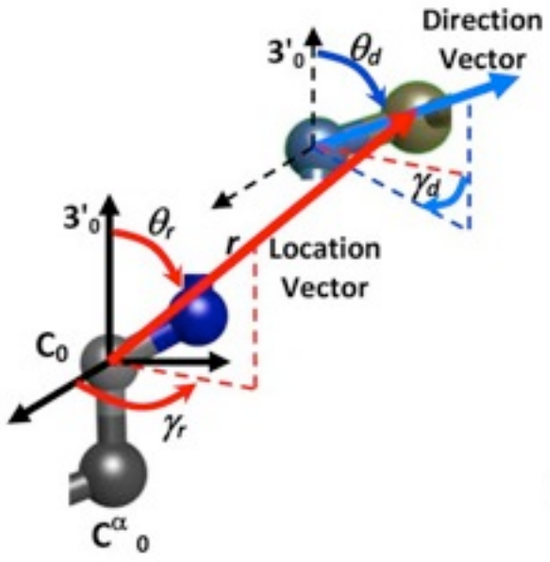
The solution: Inverse Kinematics

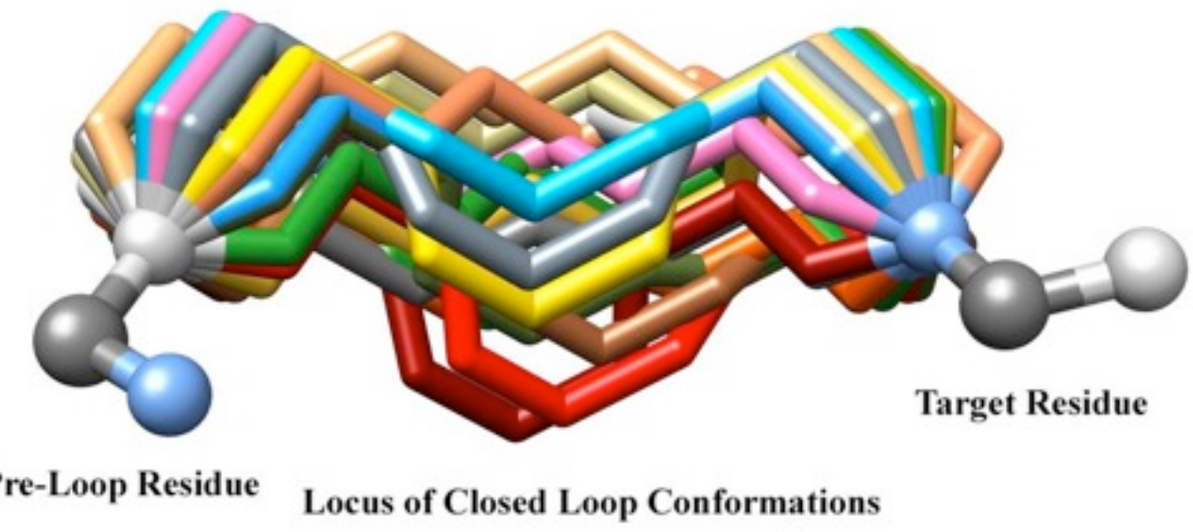
Thanks Evangelos A. Coutsias & Dan Mandell!

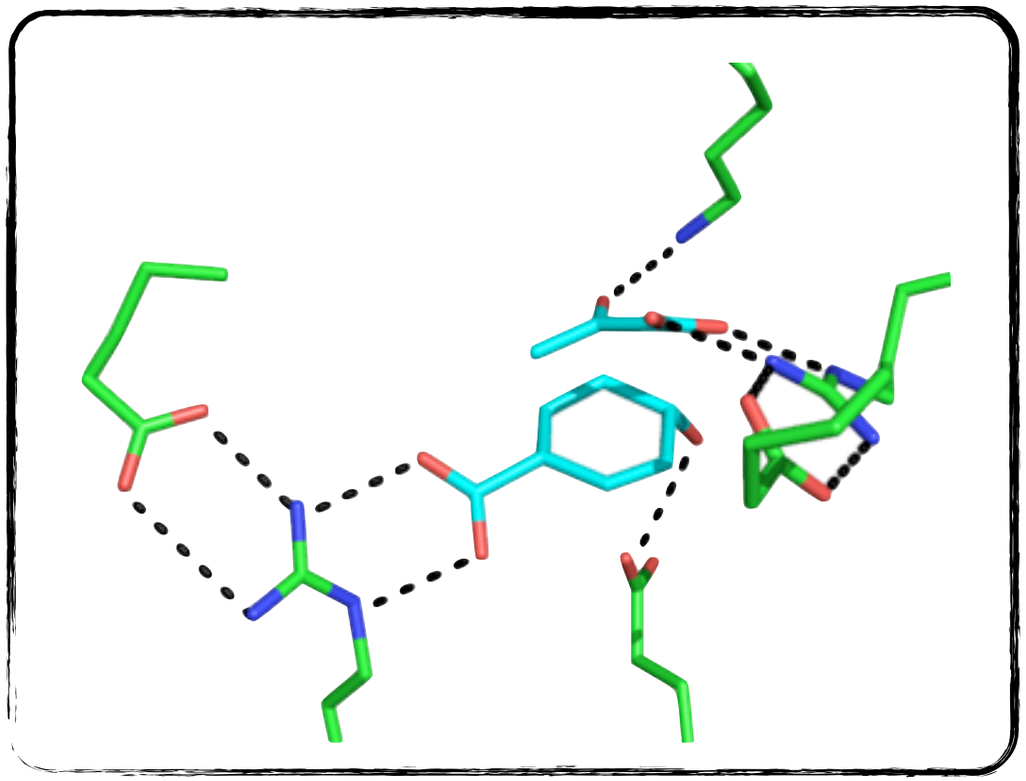
Inverse Kinematics:

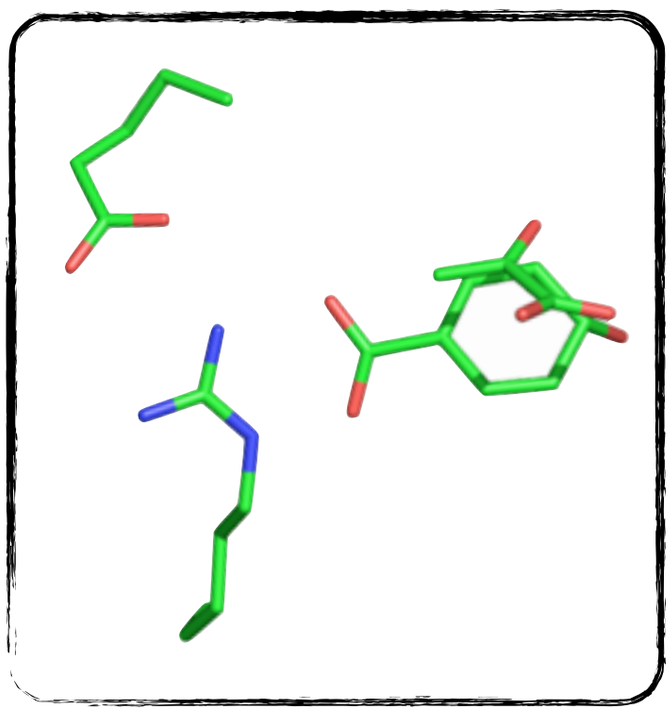
You've seen this before...

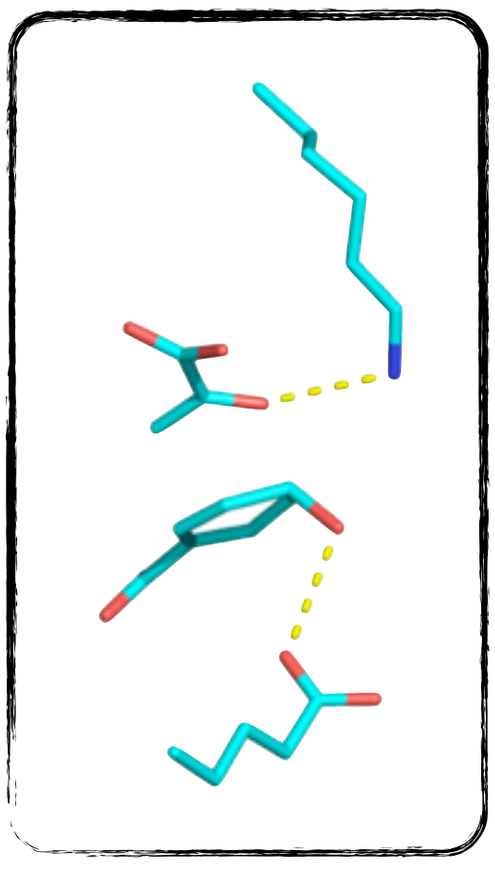
loop closure

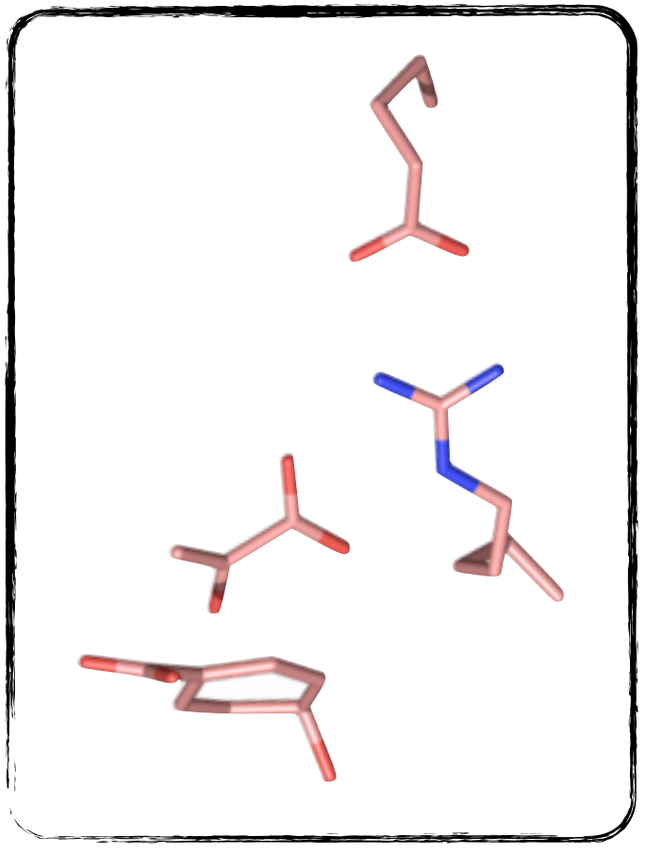




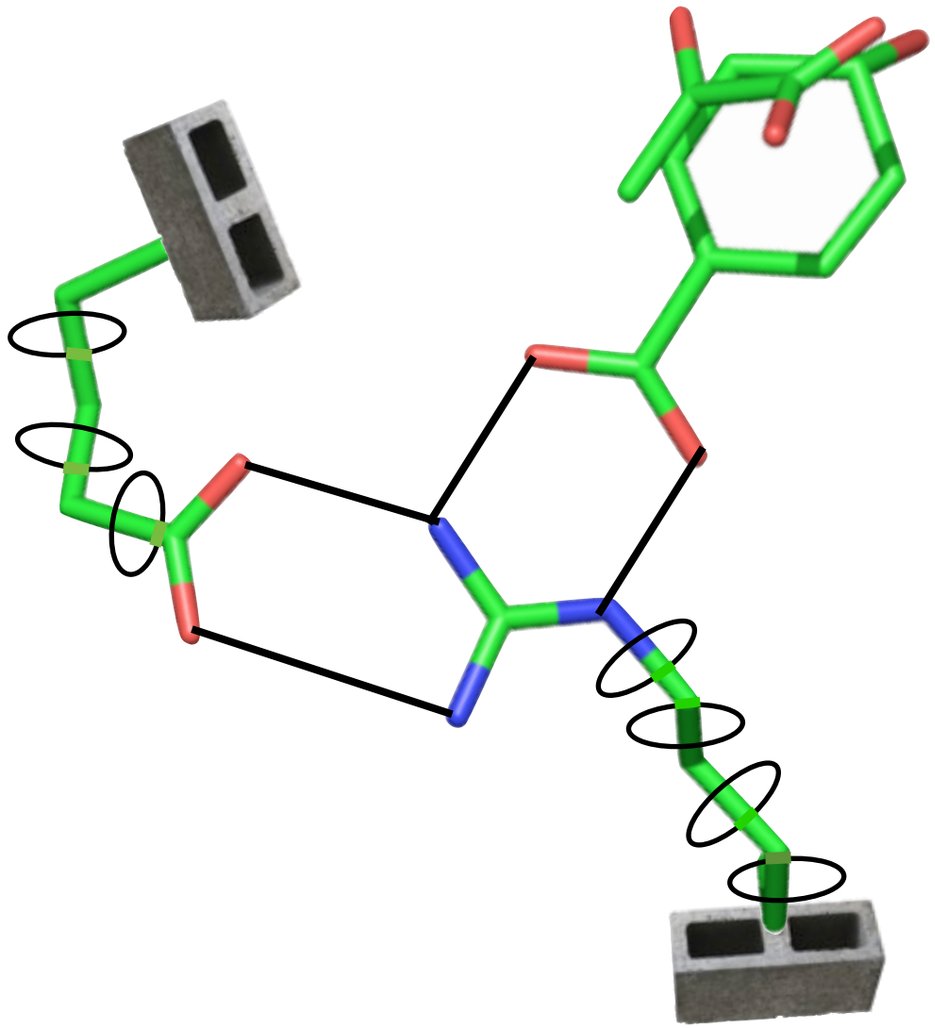


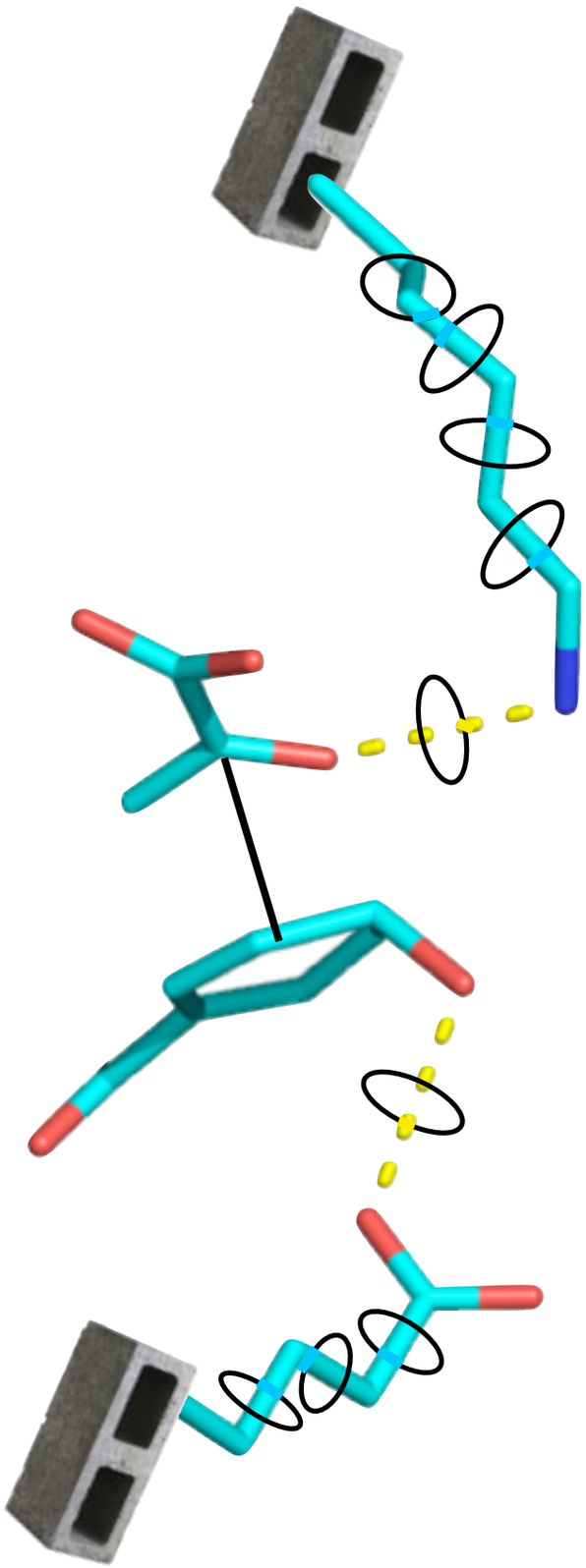




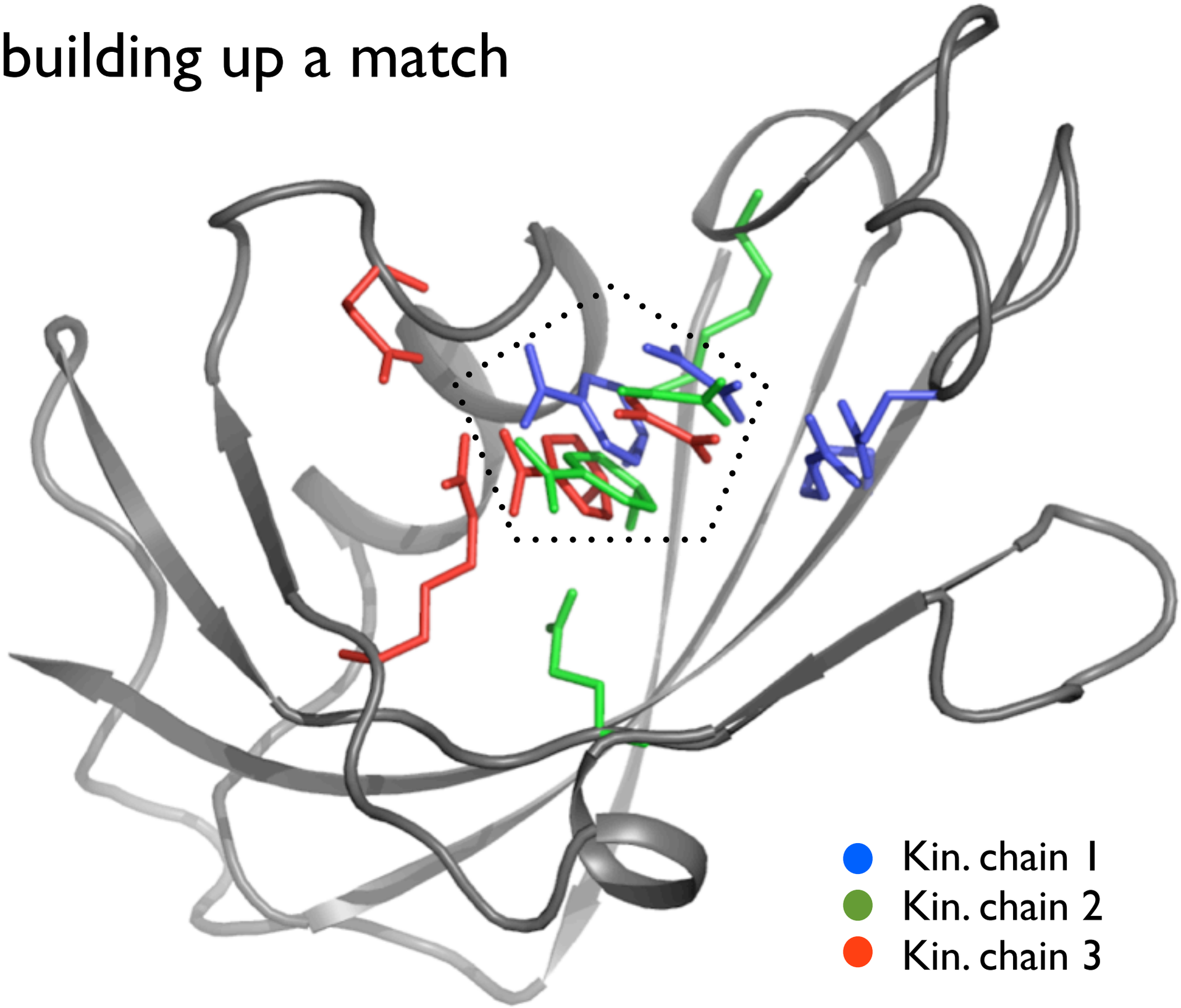


formulating side chain / ligand placement as inverse kinematics



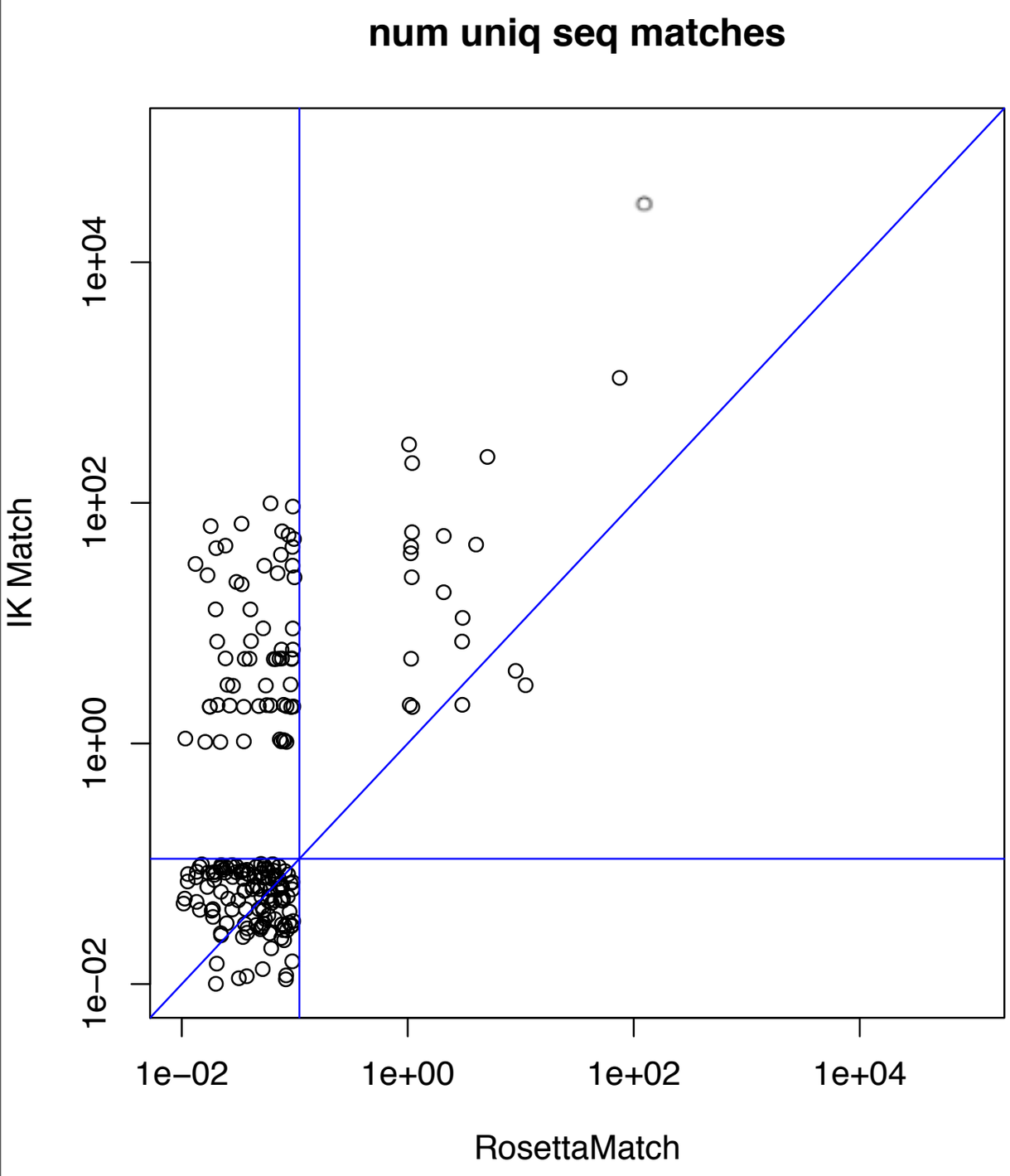


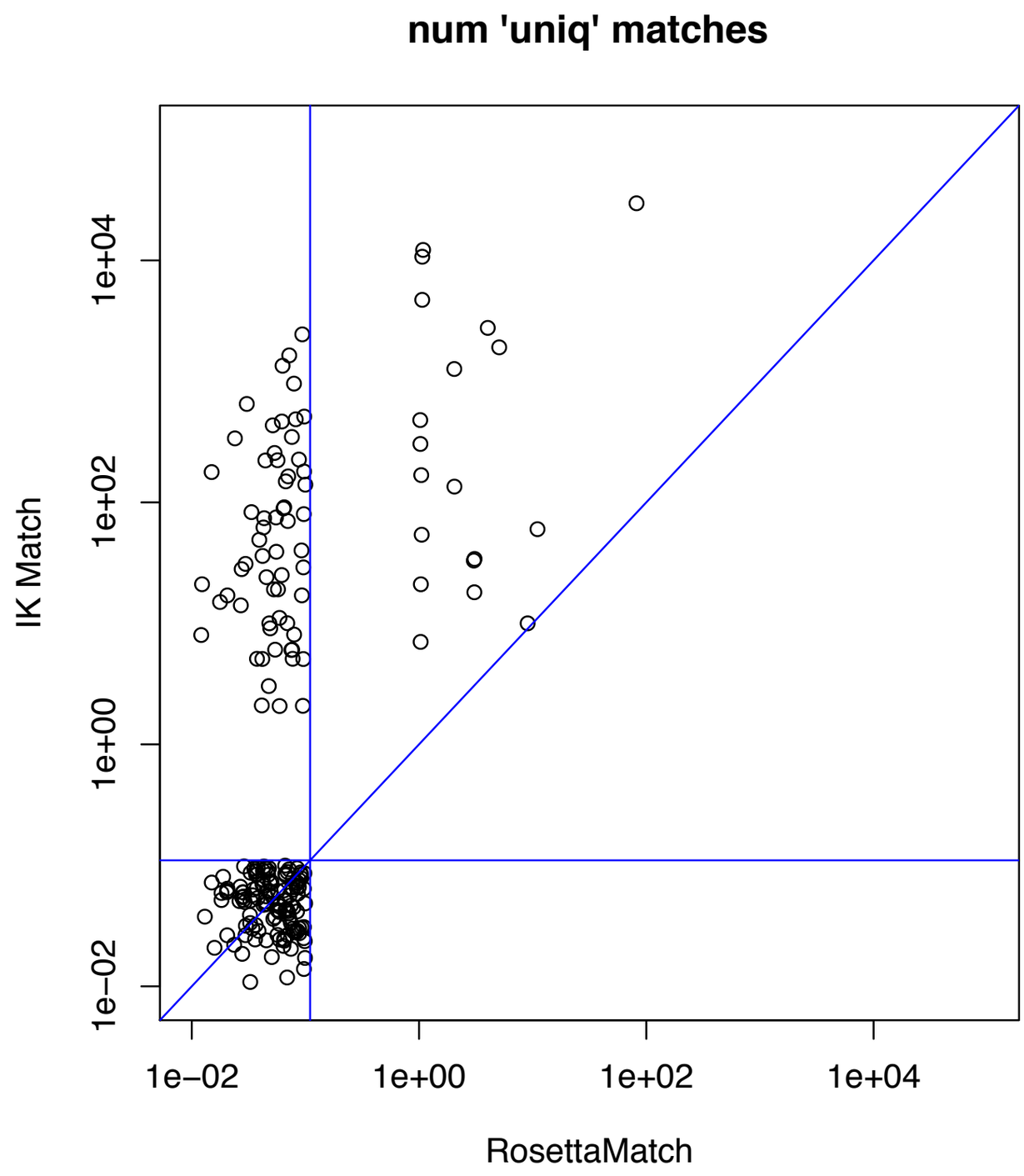
building up a match



KinMatch Prototype Results!

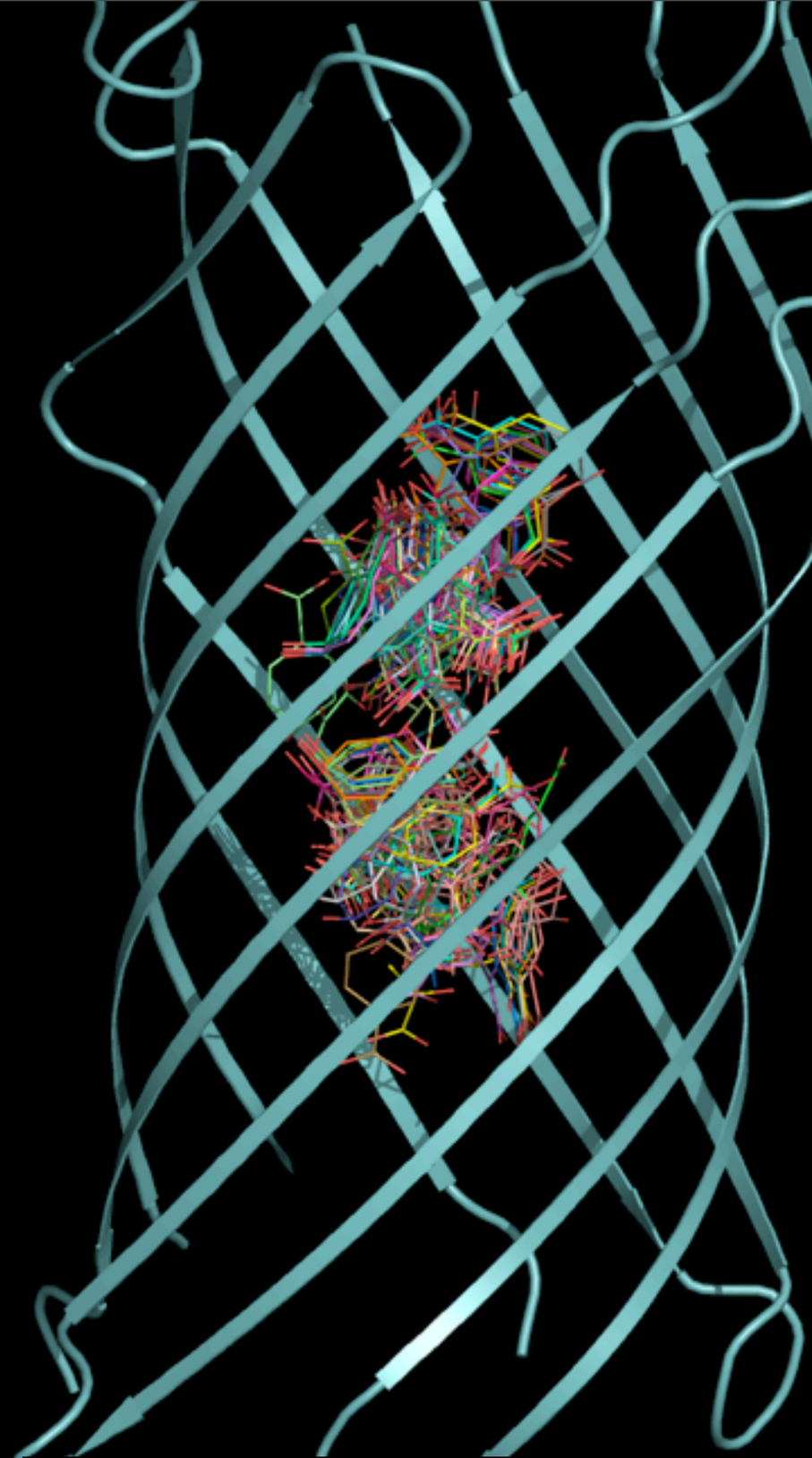
Going of rotamer: lots more matches



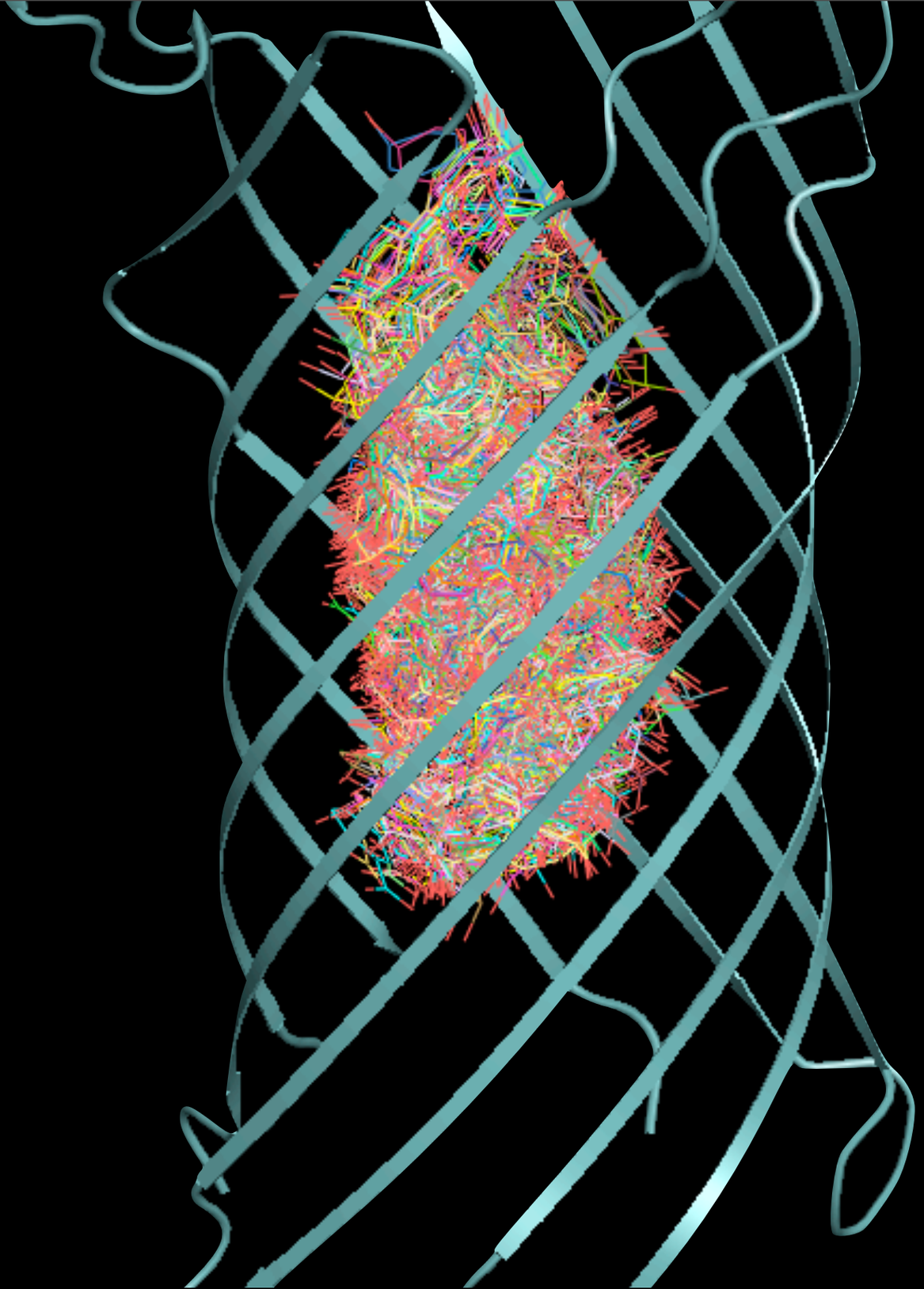


2vdf
RosettaMatch (no grid)

2vdf
KinMatch Prototype (no grid)

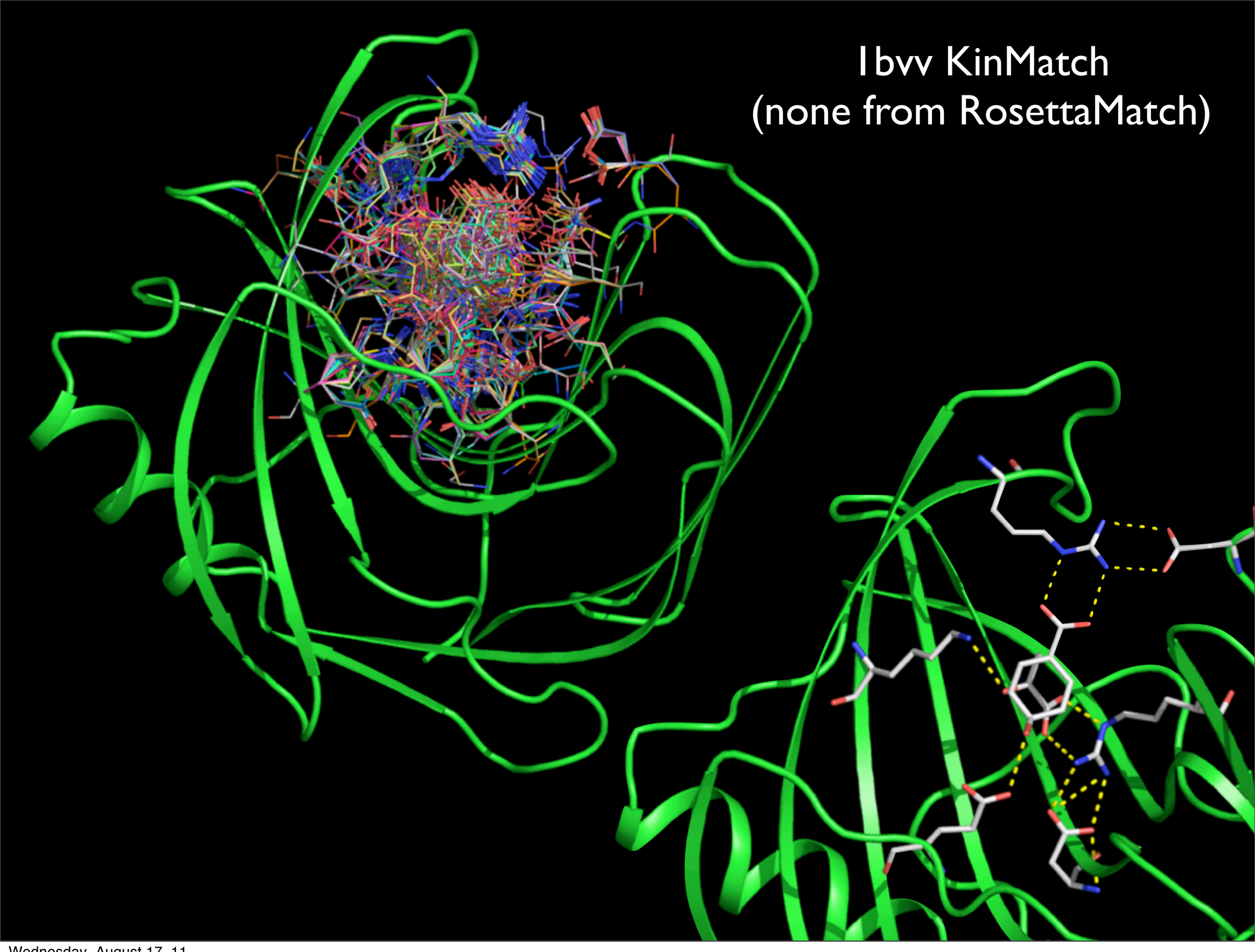


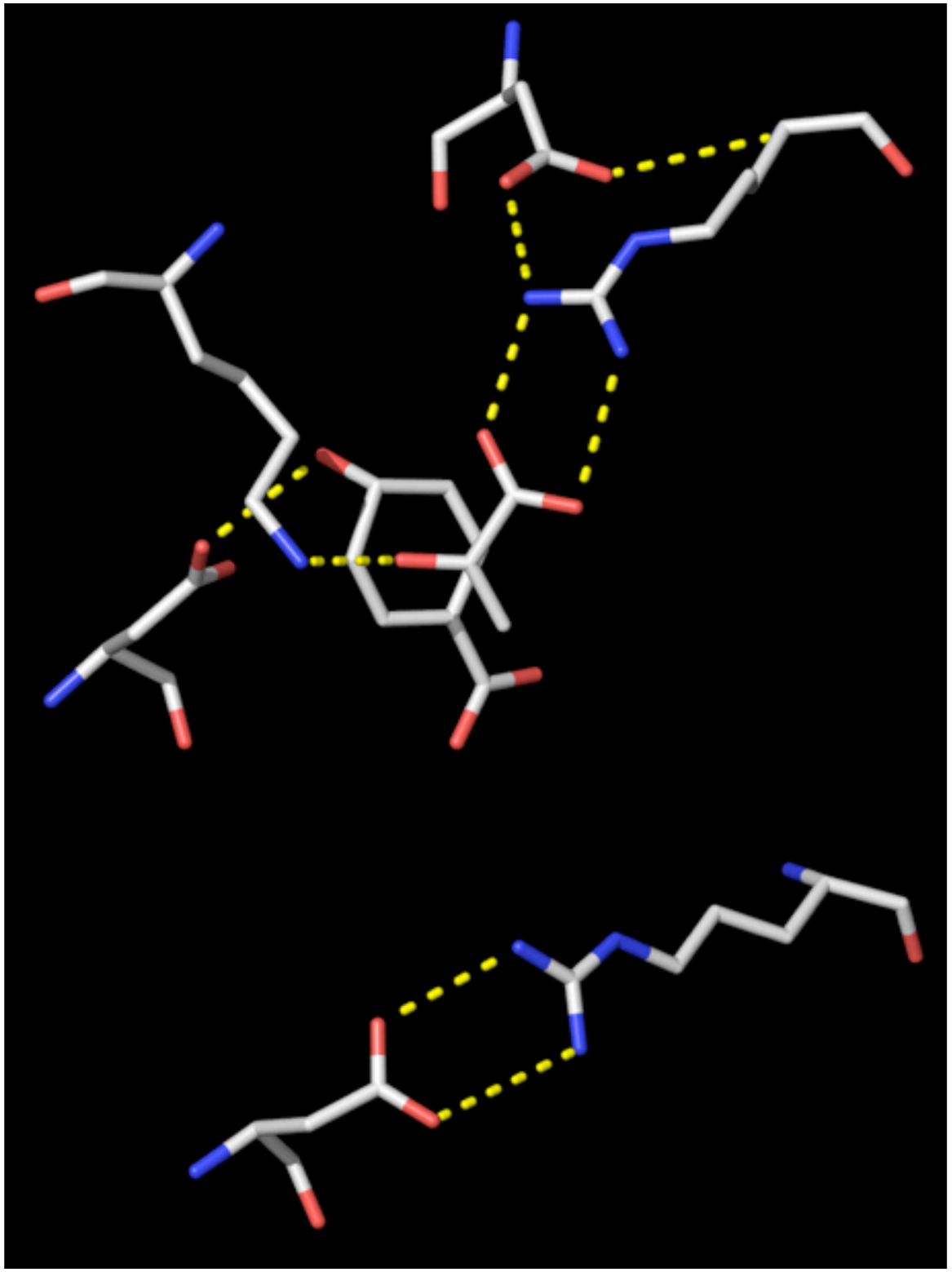
2vdf
RosettaMatch (no grid)



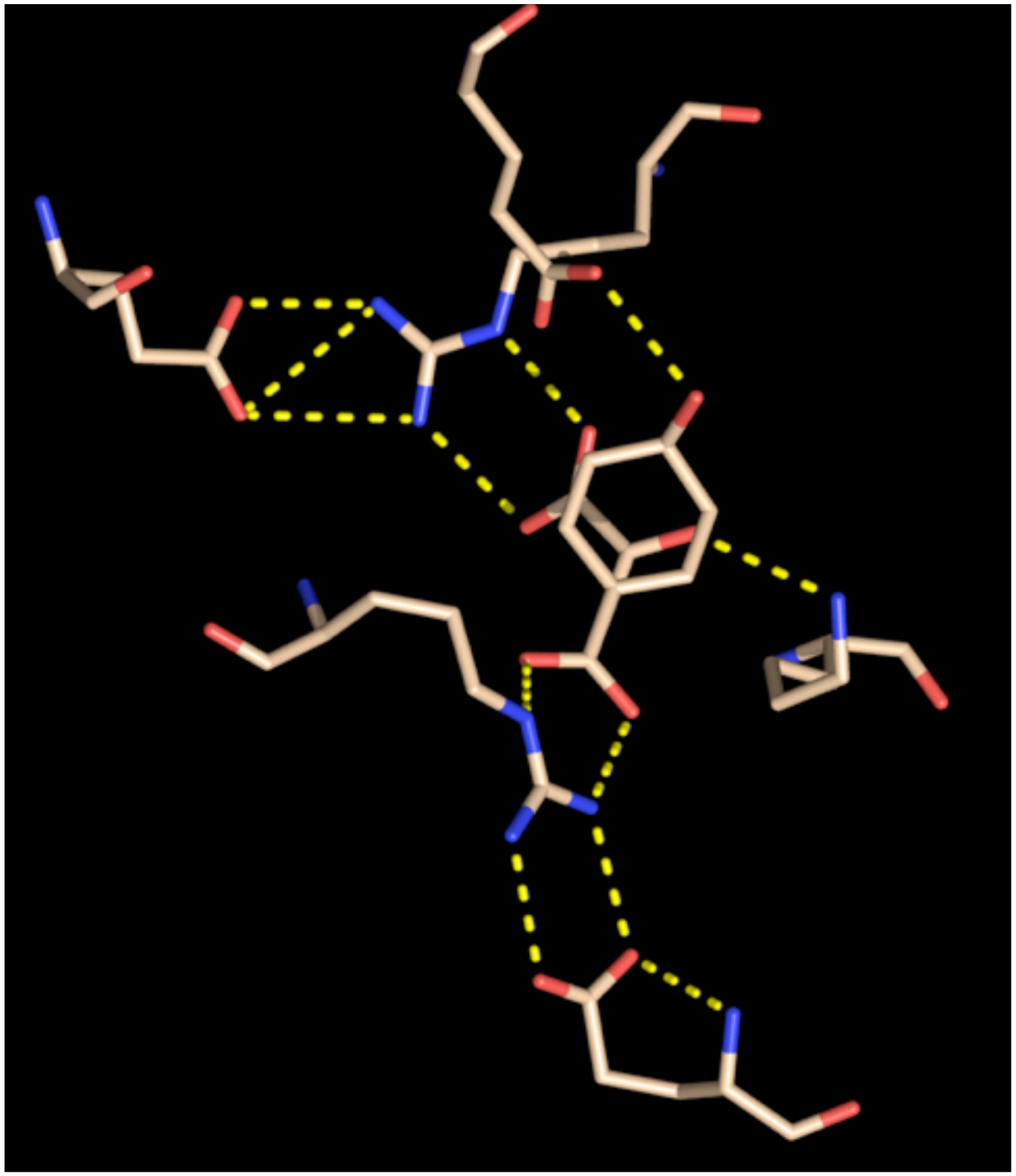
2vdf
KinMatch Prototype (no grid)

1bvv KinMatch
(none from RosettaMatch)



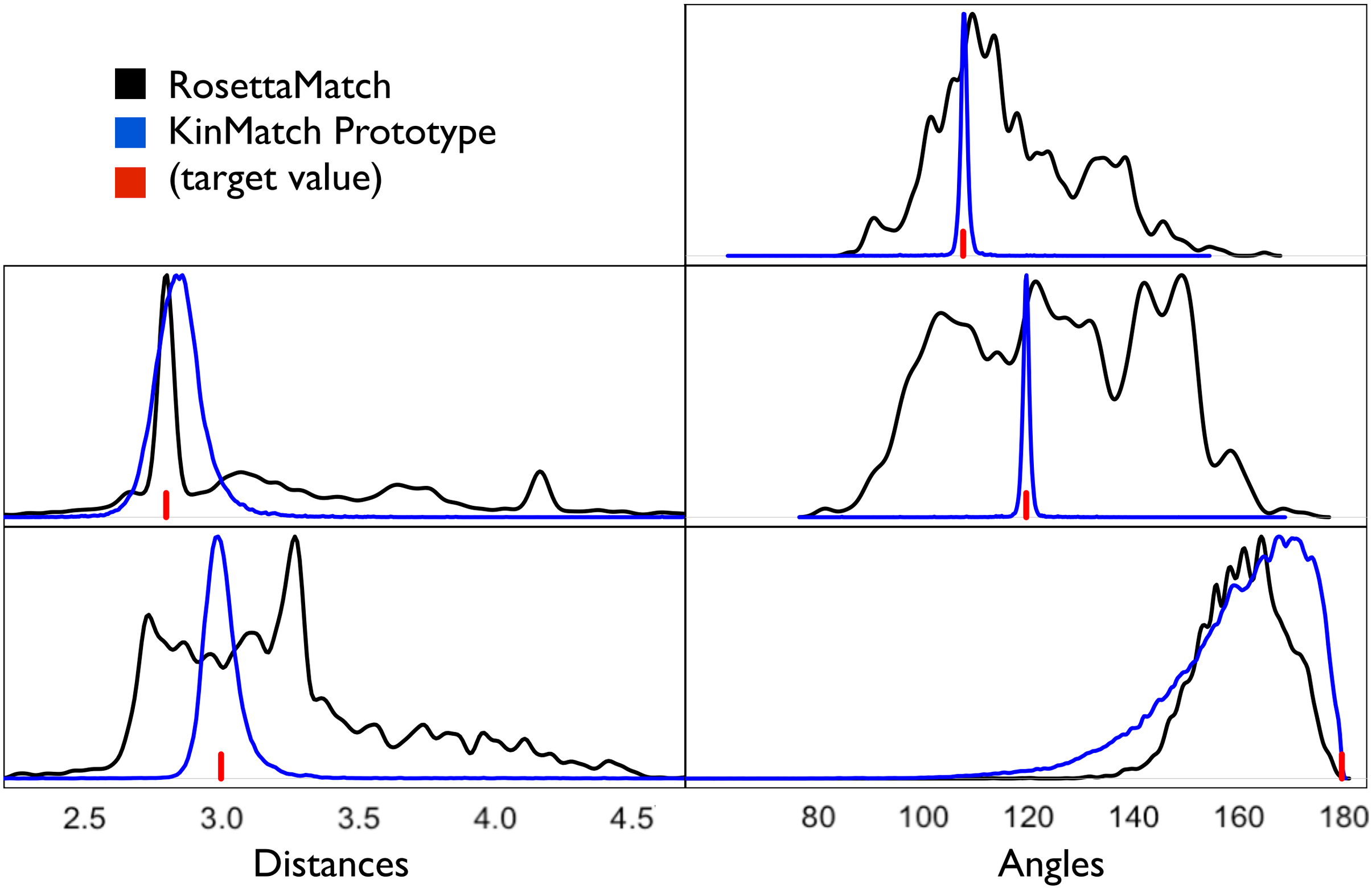


RosettaMatch

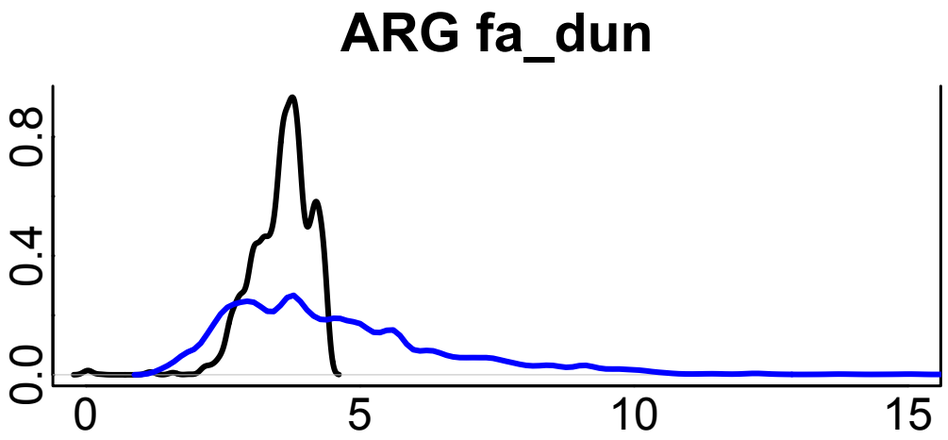


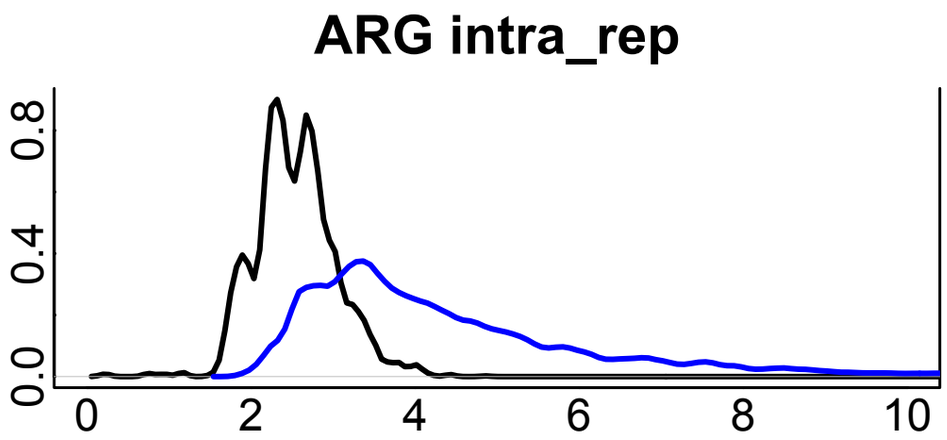
KinMatch Prototype

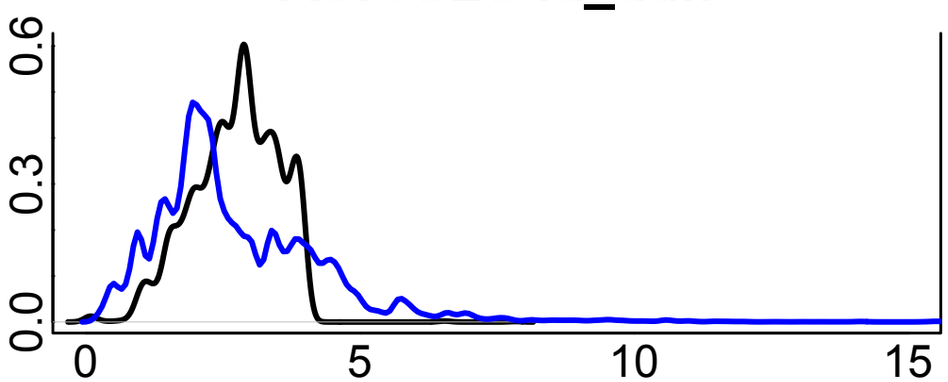
Quality of Geometry, Interactions

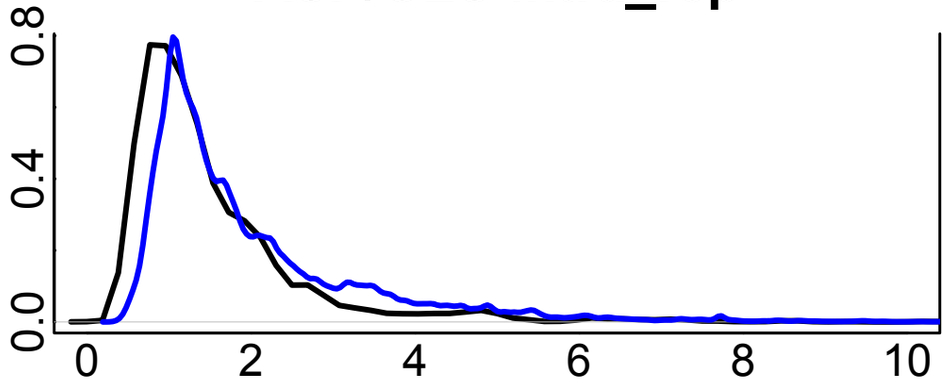


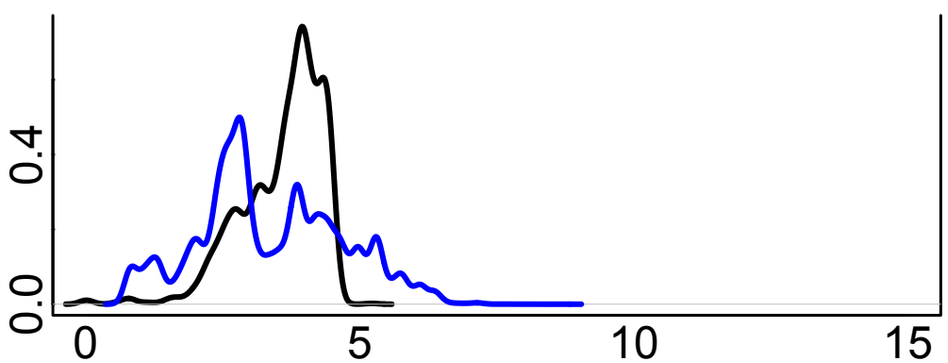
Quality of Geometry, Internal Energy

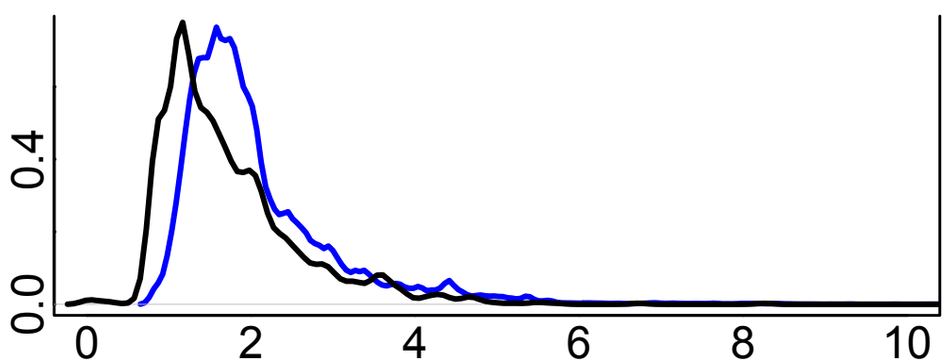






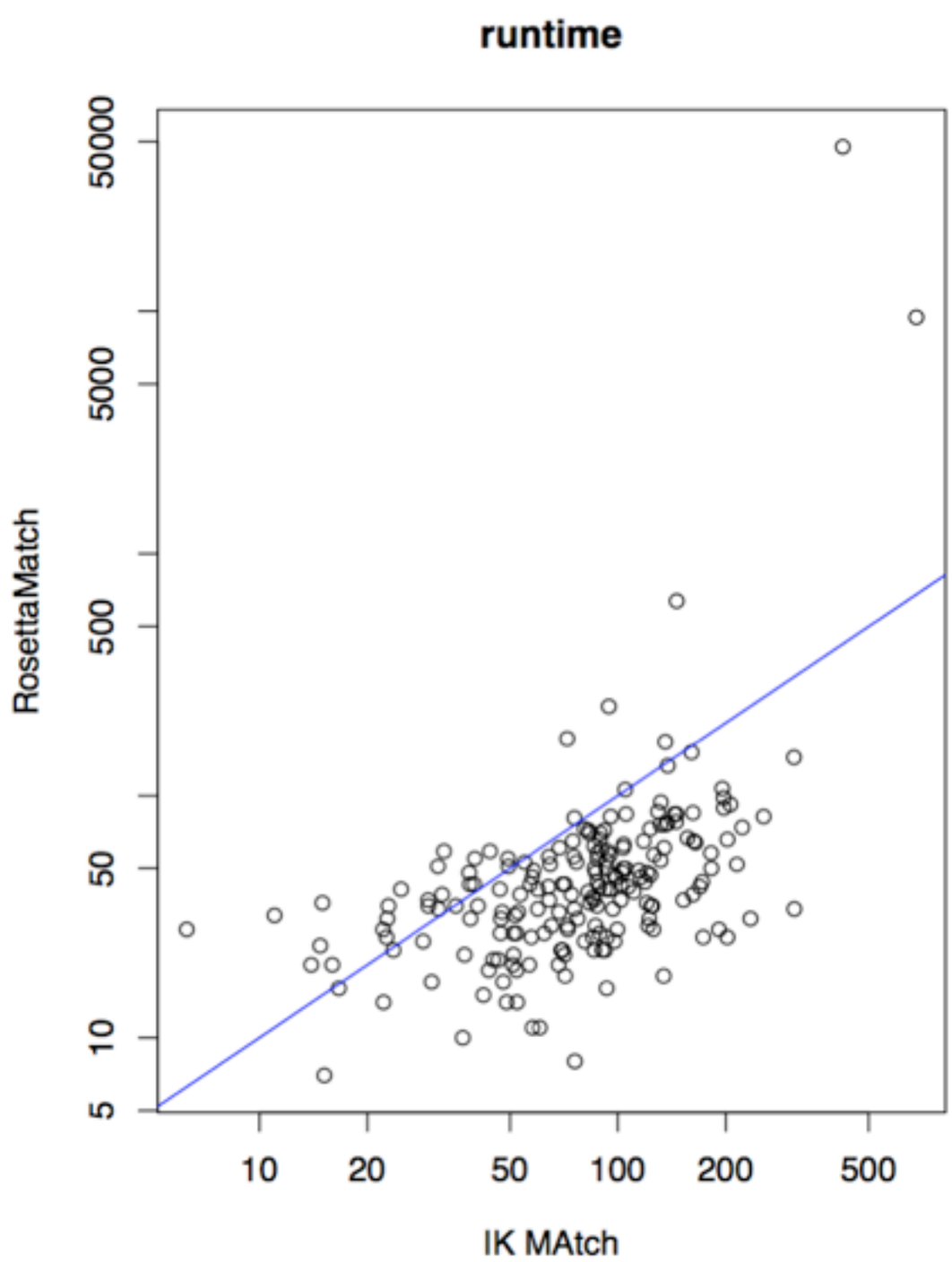


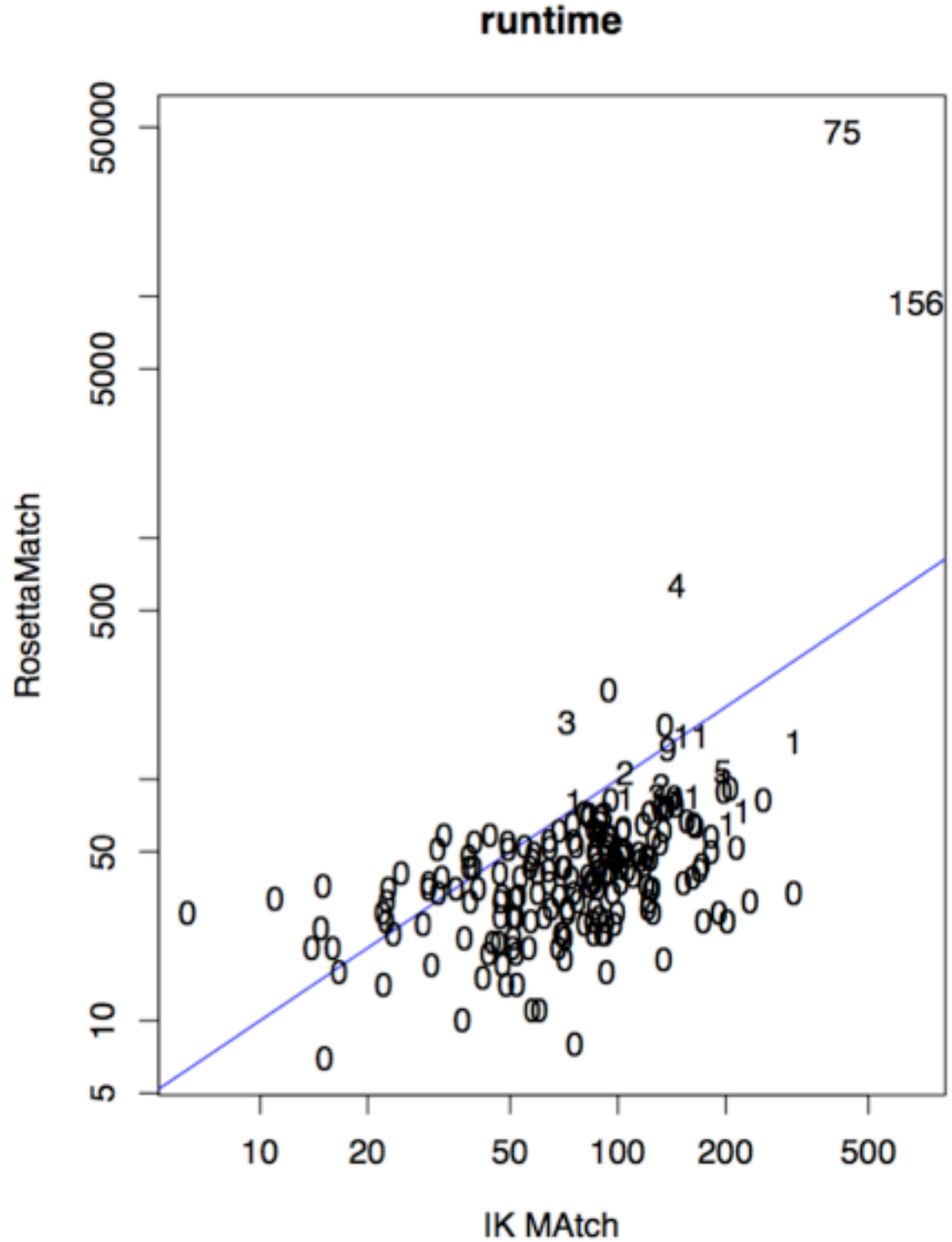




RosettaMatch KinMatch Prototype

computational issues: speed





computational issues: memory

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
32537	sheffler	20	0	277m	196m	26m	R	100	1.2	7:31.05	test_ikrs.linux
346	sheffler	20	0	340m	299m	9356	R	32	1.9	0:04.80	cc1plus
32710	sheffler	20	0	224m	144m	26m	R	32	0.9	0:48.12	test_ikrs.linux
354	sheffler	20	0	157m	117m	9304	R	32	0.7	0:02.32	cc1plus
32566	sheffler	20	0	223m	143m	26m	R	31	0.9	3:07.98	test_ikrs.linux
32554	sheffler	20	0	235m	155m	26m	R	31	1.0	2:54.56	test_ikrs.linux
32746	sheffler	20	0	400m	359m	9448	R	31	2.2	0:18.01	cc1plus
342	sheffler	20	0	321m	281m	9212	R	30	1.8	0:04.83	cc1plus
32570	sheffler	20	0	221m	141m	26m	R	29	0.9	2:49.78	test_ikrs.linux
32558	sheffler	20	0	220m	140m	25m	R	28	0.9	3:34.78	test_ikrs.linux
32754	sheffler	20	0	408m	367m	8572	R	27	2.3	0:09.71	cc1plus
32540	sheffler	20	0	213m	132m	25m	R	25	0.8	4:49.78	test_ikrs.linux
32708	sheffler	20	0	208m	127m	25m	R	25	0.8	1:21.22	test_ikrs.linux
32545	sheffler	20	0	231m	151m	26m	R	24	0.9	3:40.63	test_ikrs.linux
362	sheffler	20	0	135m	95m	8856	R	24	0.6	0:00.74	cc1plus
358	sheffler	20	0	129m	89m	9100	R	23	0.6	0:01.43	cc1plus
338	sheffler	20	0	285m	243m	9144	R	23	1.5	0:03.94	cc1plus
32686	sheffler	20	0	211m	130m	25m	R	11	0.8	0:14.78	test_ikrs.linux
32666	sheffler	20	0	211m	131m	25m	R	7	0.8	0:29.71	test_ikrs.linux
32669	sheffler	20	0	211m	130m	25m	R	7	0.8	0:15.79	test_ikrs.linux
32674	sheffler	20	0	211m	130m	25m	R	7	0.8	0:15.32	test_ikrs.linux
32663	sheffler	20	0	211m	131m	25m	R	7	0.8	0:30.82	test_ikrs.linux
32672	sheffler	20	0	211m	130m	25m	R	7	0.8	0:19.72	test_ikrs.linux
32678	sheffler	20	0	211m	130m	25m	R	7	0.8	0:19.29	test_ikrs.linux
32681	sheffler	20	0	211m	130m	25m	R	7	0.8	0:14.88	test_ikrs.linux
32689	sheffler	20	0	211m	130m	25m	R	7	0.8	0:14.56	test_ikrs.linux
32668	sheffler	20	0	211m	130m	25m	R	6	0.8	0:12.64	test_ikrs.linux
32670	sheffler	20	0	211m	131m	25m	R	6	0.8	0:26.62	test_ikrs.linux
32671	sheffler	20	0	211m	130m	25m	R	6	0.8	0:12.30	test_ikrs.linux
32675	sheffler	20	0	211m	130m	25m	R	6	0.8	0:12.01	test_ikrs.linux
32680	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.73	test_ikrs.linux
32683	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.59	test_ikrs.linux
32685	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.50	test_ikrs.linux
32693	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.25	test_ikrs.linux
32694	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.22	test_ikrs.linux
32699	sheffler	20	0	211m	130m	25m	R	6	0.8	0:14.82	test_ikrs.linux
32665	sheffler	20	0	211m	130m	25m	R	6	0.8	0:13.31	test_ikrs.linux
32698	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.10	test_ikrs.linux
32700	sheffler	20	0	211m	130m	25m	R	6	0.8	0:11.05	test_ikrs.linux
32682	sheffler	20	0	211m	130m	25m	R	6	0.8	0:14.64	test_ikrs.linux
32687	sheffler	20	0	211m	130m	25m	R	6	0.8	0:14.42	test_ikrs.linux
32688	sheffler	20	0	211m	130m	25m	R	6	0.8	0:14.40	test_ikrs.linux
32691	sheffler	20	0	211m	130m	25m	R	6	0.8	0:19.95	test_ikrs.linux
32697	sheffler	20	0	211m	130m	25m	R	6	0.8	0:19.79	test_ikrs.linux
32701	sheffler	20	0	211m	130m	25m	R	6	0.8	0:19.70	test_ikrs.linux
32703	sheffler	20	0	211m	130m	25m	R	6	0.8	0:19.64	test_ikrs.linux
32676	sheffler	20	0	211m	130m	25m	R	5	0.8	0:14.94	test_ikrs.linux
32673	sheffler	20	0	211m	130m	25m	R	5	0.8	0:12.76	test_ikrs.linux
32684	sheffler	20	0	211m	130m	25m	R	5	0.8	0:12.10	test_ikrs.linux
32690	sheffler	20	0	211m	130m	25m	R	5	0.8	0:11.88	test_ikrs.linux
32702	sheffler	20	0	211m	130m	25m	R	5	0.8	0:11.55	test_ikrs.linux
32664	sheffler	20	0	211m	130m	25m	R	5	0.8	0:14.21	test_ikrs.linux
32677	sheffler	20	0	211m	130m	25m	R	5	0.8	0:12.46	test_ikrs.linux
32695	sheffler	20	0	211m	130m	25m	R	5	0.8	0:11.72	test_ikrs.linux
32696	sheffler	20	0	211m	130m	25m	R	5	0.8	0:11.70	test_ikrs.linux
32667	sheffler	20	0	211m	130m	25m	R	4	0.8	0:13.44	test_ikrs.linux
32679	sheffler	20	0	211m	130m	25m	R	4	0.8	0:12.33	test_ikrs.linux
32692	sheffler	20	0	211m	130m	25m	R	4	0.8	0:11.79	test_ikrs.linux

Conclusions

	short residues (1-2 CHI)	long residues (3-4+ CHI)	GPU?
Enumeration	great	terrible	probably
Inverse kinematics	???	great	maybe

Possible way forward:

use I.K. to do “primary” matching on pairs
use light-weight enumeration for 2ndary matching

more testing, obviously.... volunteers?

Acknowledgements

Florian Richter

Andrew LF

Evangelos A. Coutsias & Dan Mandell

Jeremy Mills

David Baker

Baker Lab