

Combining X-ray crystallography, comparative modeling, and small angle X-ray scattering

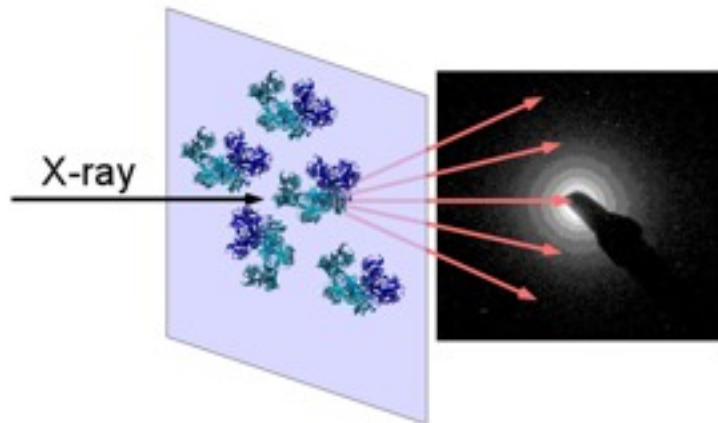
<http://salilab.org/>

Dr. Benjamin Webb

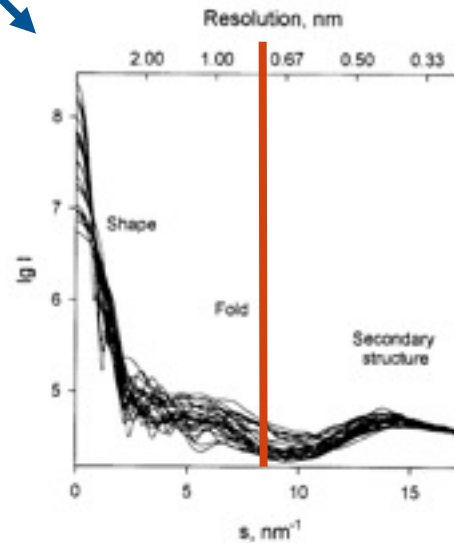
Sali Lab

University of California San Francisco

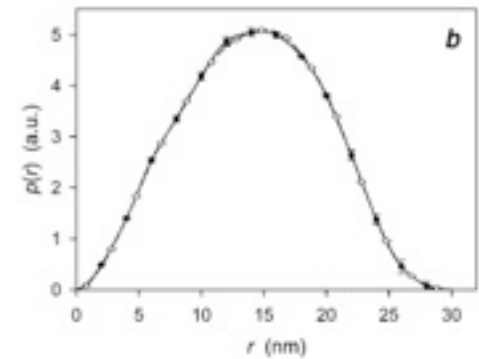
Small angle X-ray scattering (SAXS)



Rotational
average

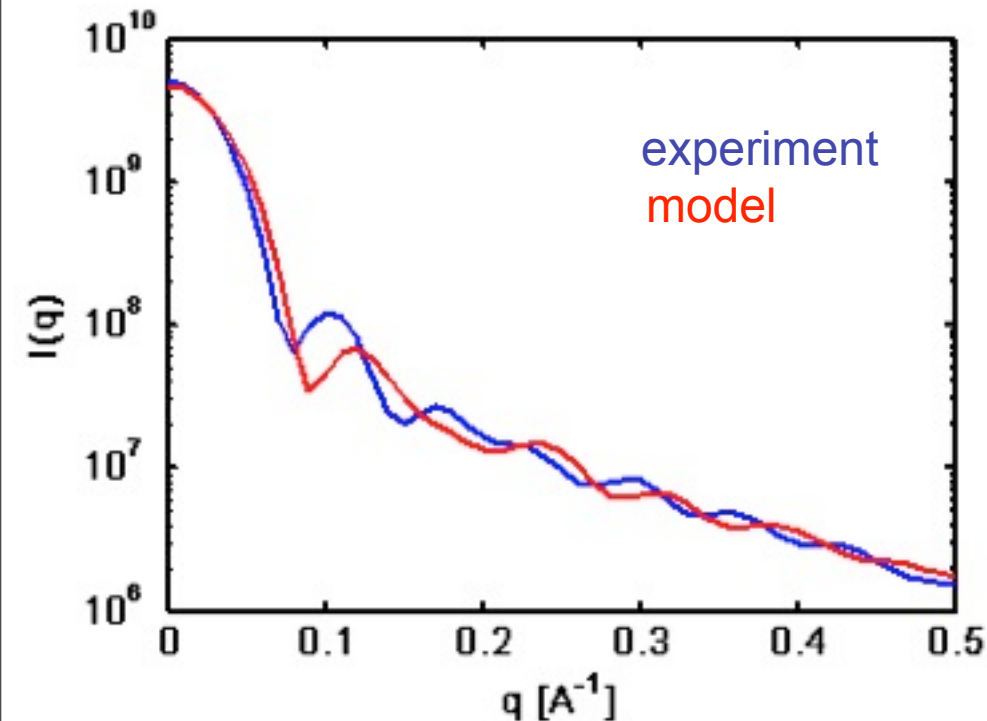


Svergun, *Biophys. J.* 2001



Fourier transform:
radial distribution
function $P(r)$

Minimize the difference between the measured and calculated model SAXS spectra



Model SAXS spectrum (Debye):

$$I_m(q) = \sum_{j=1}^{N_A} \sum_{j=1}^{N_A} f_i(q) f_j(q) \frac{\sin(qd_{ij})}{qd_{ij}}$$

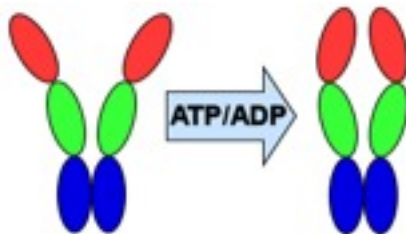
Deviation between the two spectra:

$$\chi^2 = \frac{1}{Q} \sum_{k=1}^Q \frac{1}{\sigma_{\text{exp}}^2(q_k)} \cdot \left(I_{\text{exp}}(q_k) - c \cdot I_m(q_k) \right)^2$$

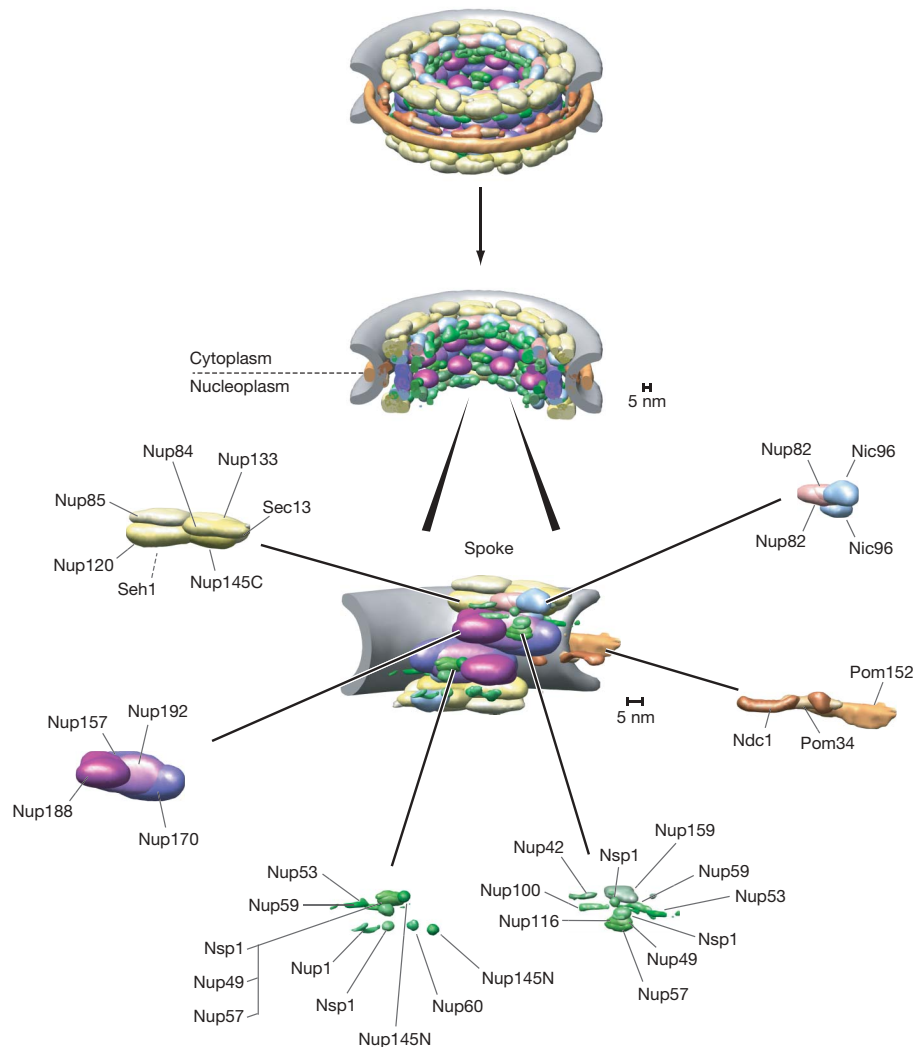
c : parameter due to insufficiently accurate concentration measurement

SAXS vs. X-ray crystallography

- X-ray crystallography
 - Requires crystals
 - Can yield very high resolution static structures
- SAXS
 - Works on a sample in solution; applicable to many more systems
 - Rotationally averaged, so less information content in a SAXS spectrum
 - Can still be used to detect shape changes, e.g. changes in quaternary structure



The Nuclear Pore Complex



- “Blob model”
- Individual subcomplexes have been characterized
- One example: the Nup84 subcomplex
- To generate an atomic model of the NPC, we first need atomic models of the subcomplexes
- In turn, we need atomic models of individual proteins
- So, let’s look at the Nup133 atomic structure

Nup133 X-ray structure

- Nup133 sequence (in yeast) was built, and the structure solved by X-ray crystallography as part of the Protein Structure Initiative, PDB code 3KFO



So, problem solved?

SAXS results for Nup133

- The Nup133 structure was also investigated with SAXS
- We obtained an experimental profile for the structure
- So, let's calculate the SAXS profile of the 3KFO crystal structure and compare it with the experimental profile
- To do this, we'll use an IMP application called FoXS
 - FoXS uses the IMP.atom module to read and handle PDB files, and the IMP.saxs module to calculate and fit the SAXS profile
 - You could also do the same thing by writing your own IMP Python script that used these two modules

Run FoXS

- Give FoXS the structure and the experimental profile
- It will report the fit (χ^2), and generate raw plot data and a gnuplot script to plot it

```
[ben@baton2]$ foxs 3KFO.pdb 23922_merge.dat
```

```
foxs 3KFO.pdb 23922_merge.dat
```

```
1669 atoms were read from PDB file 3KFO.pdb
```

```
read_SAXS_file: 23922_merge.dat size= 456 delta= 0.000608062 min_q=  
0.022805 max_q= 0.299473
```

```
Profile read from file 23922_merge.dat size = 456
```

```
Computing profile for 3KFO.pdb
```

```
start real partial profile calculation for 1669 particles
```

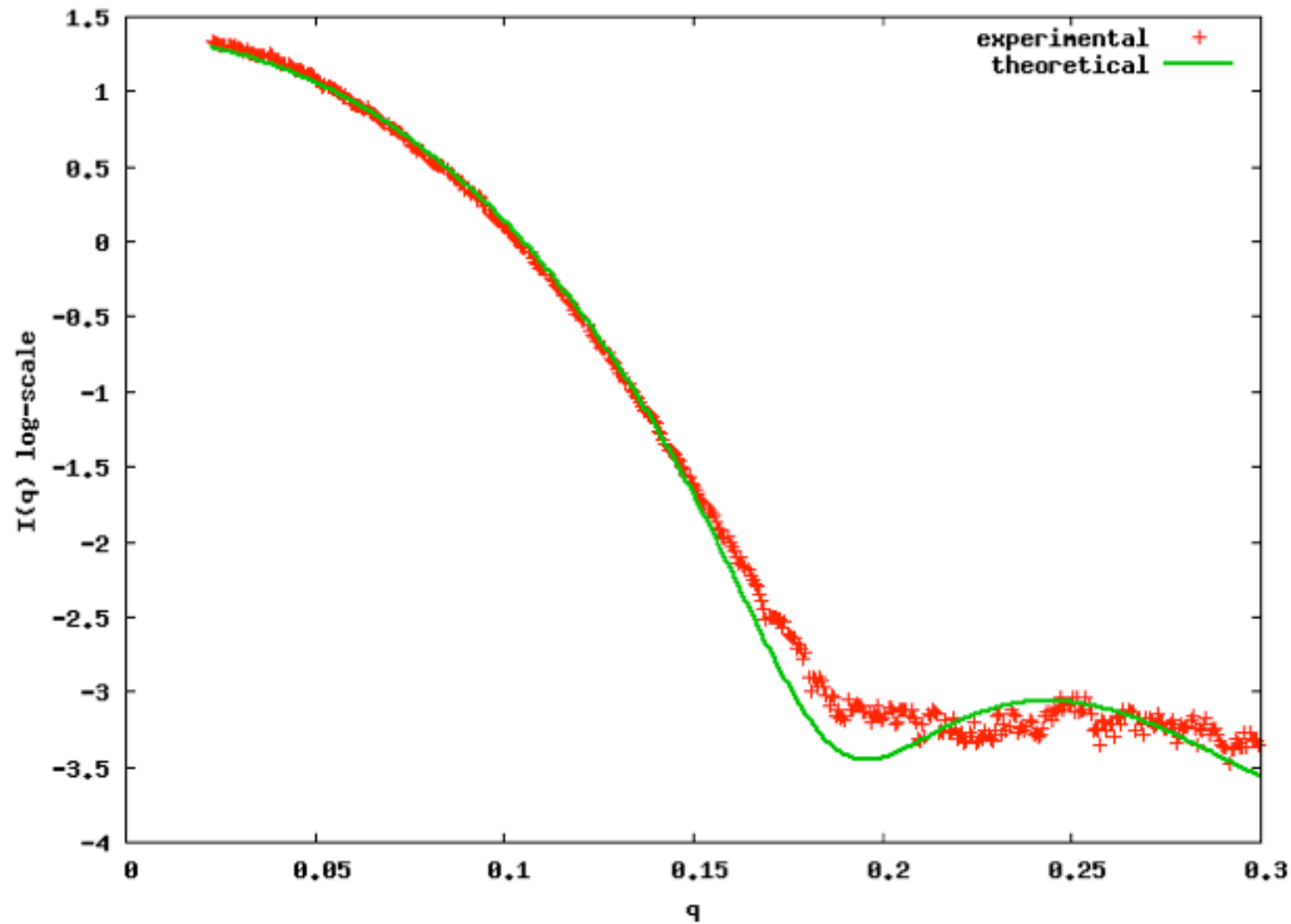
```
3KFO.pdb 23922_merge.dat Chi = 2.96261 c1 = 1.08 c2 = 3.3 default chi =  
9.87925
```


Plot the results with gnuplot

- FoXS generates a .plt file; pass it to gnuplot
- View the resulting 3KFO_23922_merge.png image

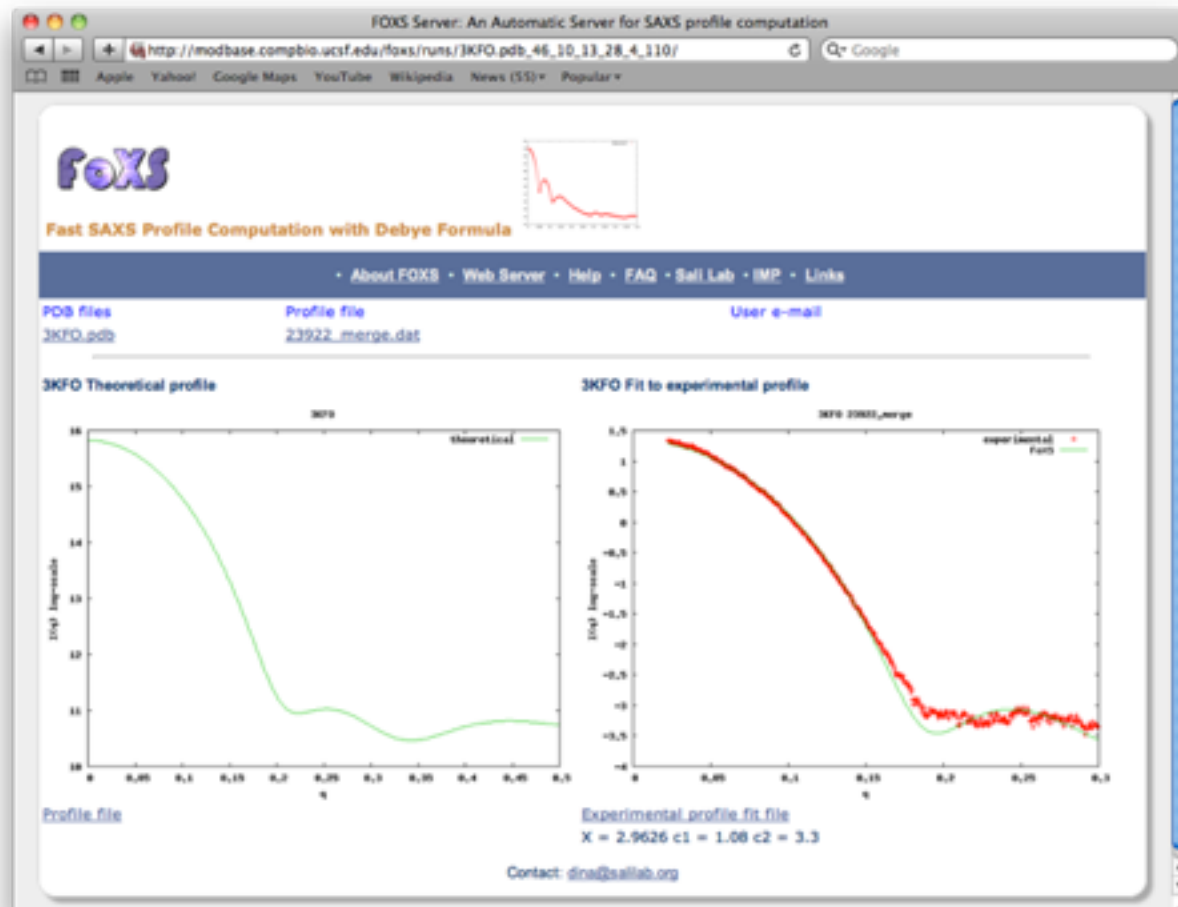
```
[ben@baton2]$ gnuplot 3KFO_23922_merge.plt
```

3KF0



FoXS web server

- You can get similar results using our FoXS web server, at <http://salilab.org/foxs>



Why isn't the fit perfect?

- Fit is definitely **not** perfect ($\chi^2 = 2.96$)
- Could the crystal structure be wrong?
 - Let's check out the header (particularly the REMARK records) in the PDB file

Problem 1. Incomplete sequence

- The Nup133 construct is supposed to be residues 881 through 1157:

```
COMPND    4  FRAGMENT: C-TERMINAL DOMAIN, RESIDUES 881-1157;
```

- but...

```
REMARK 999 SEQUENCE
```

```
REMARK 999 N-TERMINAL RESIDUES ARE NOT DEFINED THE ELECTRON DENSITY DUE
```

```
REMARK 999 TO IN-CELL PROTEOLYSIS OF THE ORIGINALLY DESIGNED CONSTRUCT.
```

- Mass spectroscopy and SDS-PAGE confirm that the protein is only residues 944-1157, not 881-1157 as originally intended
- Note though that the SAXS protein would be similarly affected

Problem 2. Missing residues

```
REMARK 465 MISSING RESIDUES
REMARK 465 THE FOLLOWING RESIDUES WERE NOT LOCATED IN THE
REMARK 465 EXPERIMENT. (M=MODEL NUMBER; RES=RESIDUE NAME; C=CHAIN
REMARK 465 IDENTIFIER; SSSEQ=SEQUENCE NUMBER; I=INSERTION CODE.)
REMARK 465
REMARK 465      M RES C SSSEQI
...
REMARK 465      LEU A    944
REMARK 465      ARG A    945
REMARK 465      GLY A   1159
REMARK 465      HIS A   1160
REMARK 465      HIS A   1161
REMARK 465      HIS A   1162
REMARK 465      HIS A   1163
REMARK 465      HIS A   1164
REMARK 465      HIS A   1165
```

- SAXS is seeing Leu-Arg and a His tag that X-ray misses

Problem 3. Missing atoms

```
REMARK 470 MISSING ATOM
REMARK 470 THE FOLLOWING RESIDUES HAVE MISSING ATOMS (M=MODEL NUMBER;
REMARK 470 RES=RESIDUE NAME; C=CHAIN IDENTIFIER; SSEQ=SEQUENCE NUMBER;
REMARK 470 I=INSERTION CODE) :
REMARK 470      M RES CSSEQI  ATOMS
REMARK 470      LYS A 958      CG   CD   CE   NZ
REMARK 470      LYS A 966      CG   CD   CE   NZ
REMARK 470      LYS A 973      CG   CD   CE   NZ
REMARK 470      LYS A 976      CG   CD   CE   NZ
REMARK 470      GLU A1032      CG   CD   OE1  OE2
REMARK 470      GLU A1053      CG   CD   OE1  OE2
REMARK 470      LYS A1063      CG   CD   CE   NZ
REMARK 470      LYS A1072      CG   CD   CE   NZ
REMARK 470      ASN A1073      CG   OD1  ND2
REMARK 470      LEU A1083      CG   CD1  CD2
REMARK 470      GLU A1093      CG   CD   OE1  OE2
REMARK 470      GLU A1097      CG   CD   OE1  OE2
REMARK 470      ASP A1126      CG   OD1
REMARK 470      LYS A1144      CG   CD   CE   NZ
REMARK 470      TYR A1150      CG   CD1  CD2  CE1  CE2  CZ   OH
REMARK 470      GLU A1151      CG   CD   OE1  OE2
```

- X-ray misses the sidechains of 16 residues

Add missing atoms/residues

- We need to add back the missing atoms and residues to get a correct Nup133 structure
- We can use Modeller for this, as a “mini homology modeling” problem
 - Original 3KFO structure is the template
 - 3KFO plus missing atoms and residues is the model

Get the current 3KFO sequence

- Modeller script, get-sequence.py:

```
from modeller import *  
  
e = environ()  
e.libs.topology.read('${LIB}/top_heav.lib')  
  
m = model(e, file='3KFO')  
  
a = alignment(e)  
a.append_model(m, align_codes='3KFO', atom_files='3KFO')  
  
a.write(file='3KFO.seq')
```

Get the current 3KFO sequence

- Modeller script, get-sequence.py:

```
from modeller import *
```

```
e = environ()
```

```
e.libs.topology.read('${LIB}/top_heavy.lib')
```

Read the 3KFO structure (ATOM records)
from the 3KFO PDB file

```
m = model(e, file='3KFO')
```

```
a = alignment(e)
```

```
a.append_model(m, align_codes='3KFO', atom_files='3KFO')
```

```
a.write(file='3KFO.seq')
```

Get the current 3KFO sequence

- Modeller script, get-sequence.py:

```
from modeller import *
```

```
e = environ()
```

```
e.libs.topology.read('${LIB}/top_heav.lib')
```

```
m = model(e, file='3KFO')
```

```
a = alignment(e)
```

Add the sequence corresponding to the
ATOM records to the alignment

```
a.append_model(m, align_codes='3KFO', atom_files='3KFO')
```

```
a.write(file='3KFO.seq')
```

Get the current 3KFO sequence

- Modeller script, get-sequence.py:

```
from modeller import *  
  
e = environ()  
e.libs.topology.read('${LIB}/top_heav.lib')  
  
m = model(e, file='3KFO')  
  
a = alignment(e)  
a.append_model(m, align_codes='3KFO', atom_files='3KFO')  
  
a.write(file='3KFO.seq')
```

Output: 3KFO.seq

```
>P1;3KFO
```

```
structureX:3KFO: 946 :A:+213 :A:NUP133:SACCHAROMYCES CEREVISIAE: 1.90: 0.19  
KIQYNLDTIDAEKNISNKLKKGEVQICKRFKNGSIREVFNILVEELKSTTVVNLSDLVELYSMLDDEESLFIPLR  
LLSVDGNLLNFEVKKFLNALVWRRIVLLNASNEGDKLLQHIVKRVFDEELPKNNDFFPLPSVDLLCDKSLLTPEYI  
SETYGRFPIDQNAIREEIYEEISQVETLNSDNSLEIKLHSTIGSVAKEKNYTINYETNTVEYE*
```

- We need to make a new alignment file containing two modified versions of this sequence:
 - One is this original sequence, with gaps added where missing residues are present
 - The second is the full sequence, with missing residues added
- Note that Modeller adds the missing sidechains automatically

3KFO.seq

```
>P1;3KFO
```

```
structureX:3KFO: 946 :A:+213 :A:NUP133:SACCHAROMYCES CEREVISIAE: 1.90: 0.19  
KIQYNLDTIDA EK NISNKLKKGEVQICKR FKNGSIREVFNILVEELKSTTVVNLSDLVELYSMLDDEESLFIPLR  
LLSVDGNLLNF EVKKFLNALVWRRIVLLNASNEGDKLLQHIVKRVFDEELPKN NDFPLPSVDLLCDKSLLTPEYI  
SETYGRFPIDQNAIREEEIYEEISQVETLNSDNSLEIKLHSTIGSVAKEKNYTINYETNTVEYE*
```



3KFO-fill.ali

```
>P1;3KFO
```

```
structureX:3KFO: 946 :A:+213 :A:NUP133:SACCHAROMYCES CEREVISIAE: 1.90: 0.19  
--KIQYNLDTIDA EK NISNKLKKGEVQICKR FKNGSIREVFNILVEELKSTTVVNLSDLVELYSMLDDEESLFIPL  
LRLLSVDGNLLNF EVKKFLNALVWRRIVLLNASNEGDKLLQHIVKRVFDEELPKN NDFPLPSVDLLCDKSLLTPE  
YISETYGRFPIDQNAIREEEIYEEISQVETLNSDNSLEIKLHSTIGSVAKEKNYTINYETNTVEYE-----*
```

```
>P1;3KFO-fill
```

```
sequence:::::::::
```

```
LRKIQYNLDTIDA EK NISNKLKKGEVQICKR FKNGSIREVFNILVEELKSTTVVNLSDLVELYSMLDDEESLFIPL  
LRLLSVDGNLLNF EVKKFLNALVWRRIVLLNASNEGDKLLQHIVKRVFDEELPKN NDFPLPSVDLLCDKSLLTPE  
YISETYGRFPIDQNAIREEEIYEEISQVETLNSDNSLEIKLHSTIGSVAKEKNYTINYETNTVEYEGHHHHHH*
```

Run Modeller

- fill.py:

```
from modeller import *
from modeller.automodel import *

e = environ()

a = automodel(e, alnfile='3KFO-fill.ali',
              knowns='3KFO', sequence='3KFO-fill')
a.starting_model = 1
a.ending_model = 5
a.make()
```

Run Modeller

- fill.py:

```
from modeller import *
from modeller.automodel import *

e = environ()

a = automodel(e, alnfile='3KFO-fill.ali',
              knowns='3KFO', sequence='3KFO-fill')
a.starting_model = 1
a.ending_model = 5
a.make()
```


Run Modeller

- fill.py:

```
from modeller import *
from modeller.automodel import *

class MyModel(automodel):
    def special_patches(self, aln):
        self.rename_segments(segment_ids=['A'],
                               renumber_residues=[944])

e = environ()

a = automodel(e, alnfile='3KFO-fill.ali',
               knowns='3KFO', sequence='3KFO-fill')
a.starting_model = 1
a.ending_model = 5
a.make()
```

Run Modeller

- fill.py:

```
from modeller import *
from modeller.automodel import *

class MyModel(automodel):
    def special_patches(self, aln):
        self.rename_segments(segment_ids=['A'],
                               renumber_residues=[944])

e = environ()

a = MyModel(e, alnfile='3KFO-fill.ali',
             knowns='3KFO', sequence='3KFO-fill')
a.starting_model = 1
a.ending_model = 5
a.make()
```

Run FoXS on each output model

```
[ben@baton2 foxs]$ foxs 3KFO-fill1.B99990003.pdb 23922_merge.dat
```

```
foxs 3KFO-fill1.B99990003.pdb 23922_merge.dat
```

```
1817 atoms were read from PDB file 3KFO-fill1.B99990003.pdb
```

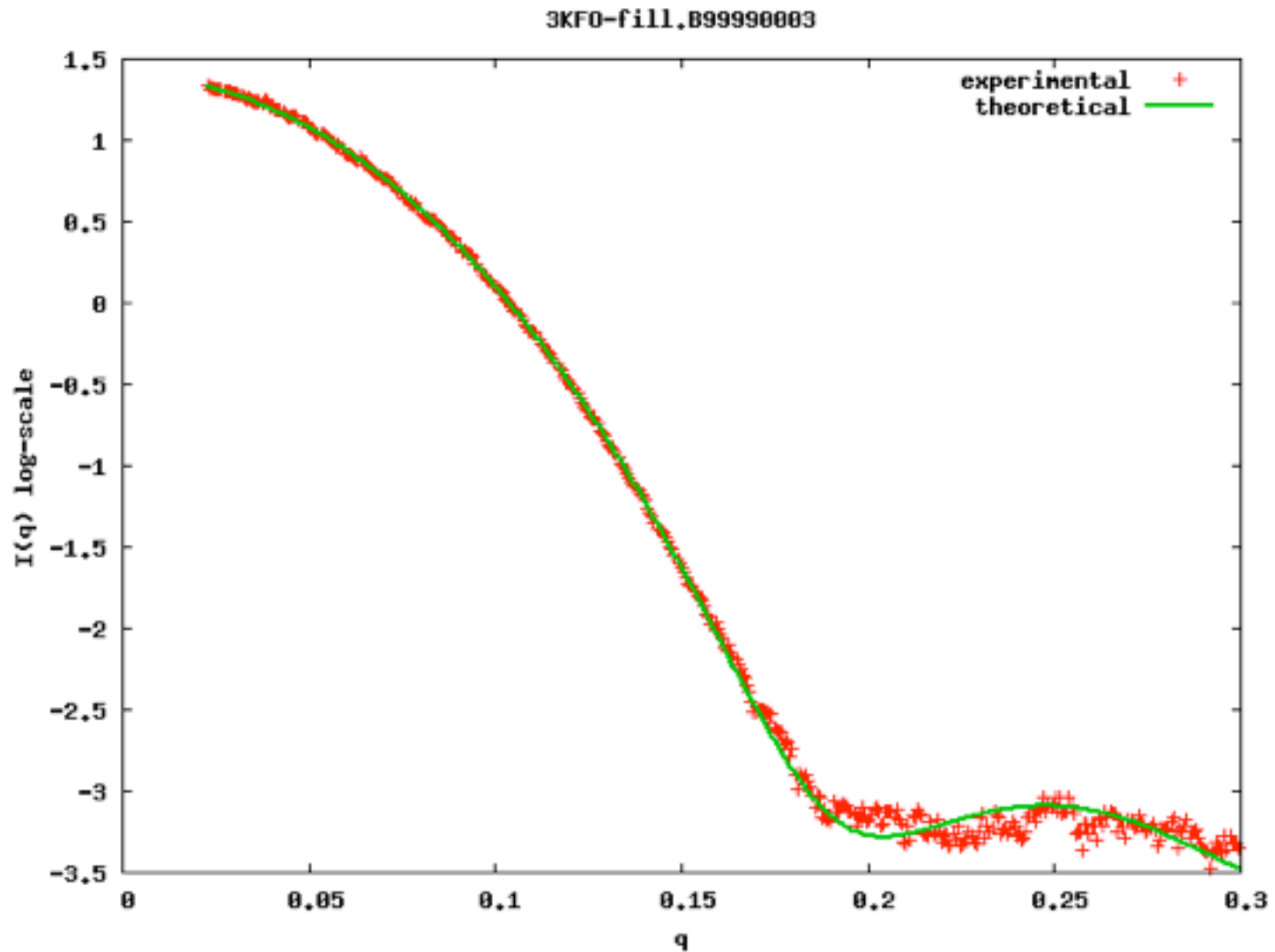
```
read_SAXS_file: 23922_merge.dat size= 456 delta= 0.000608062 min_q=  
0.022805 max_q= 0.299473
```

```
Profile read from file 23922_merge.dat size = 456
```

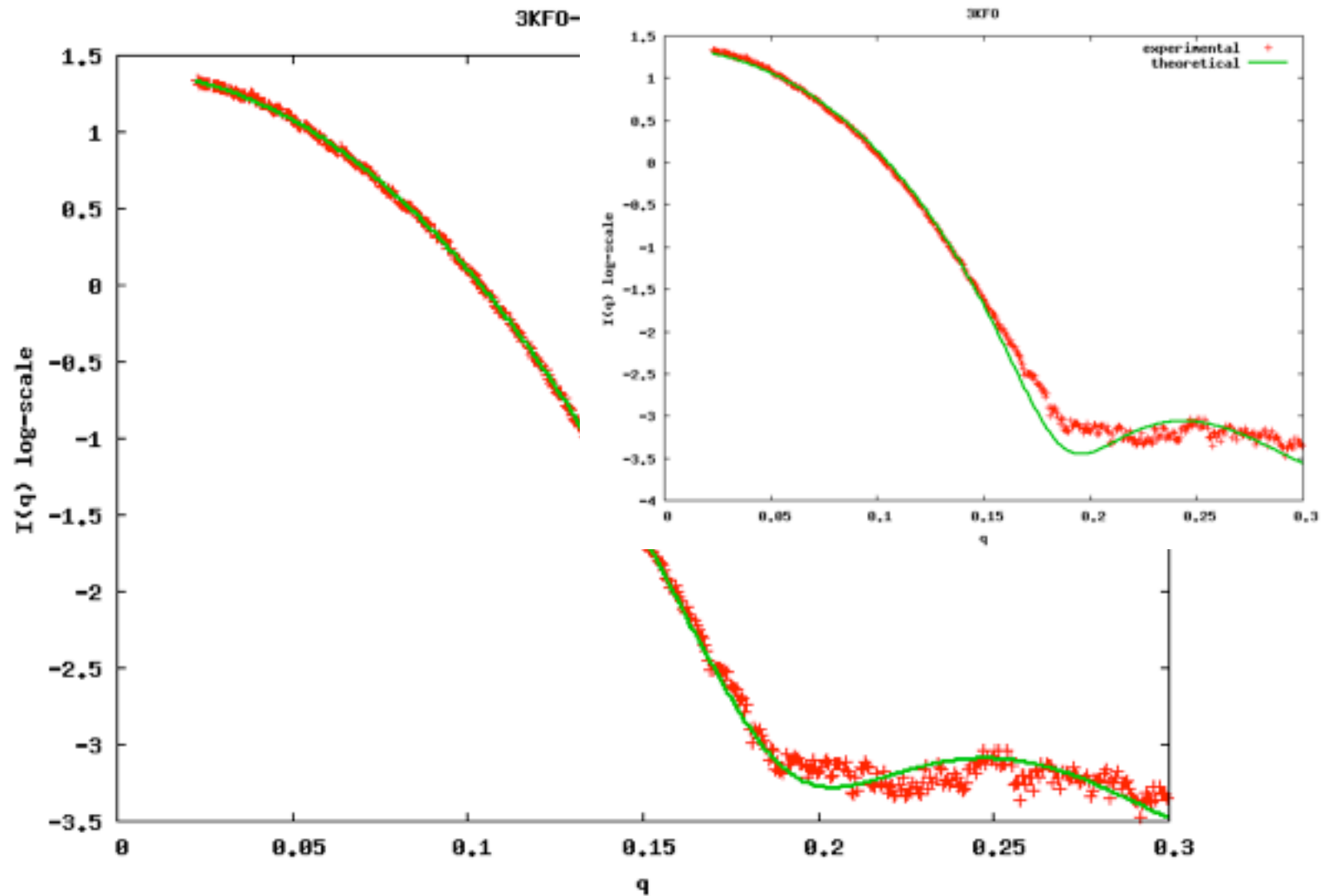
```
Computing profile for 3KFO-fill1.B99990003.pdb
```

```
start real partial profile calculation for 1817 particles
```

```
3KFO-fill1.B99990003.pdb 23922_merge.dat Chi = 1.21422 c1 = 1.08 c2 = 1  
default chi = 6.2183
```



- Best model ($\chi^2 = 1.21$, improved from 2.96) clearly fits the SAXS data better than the original structure

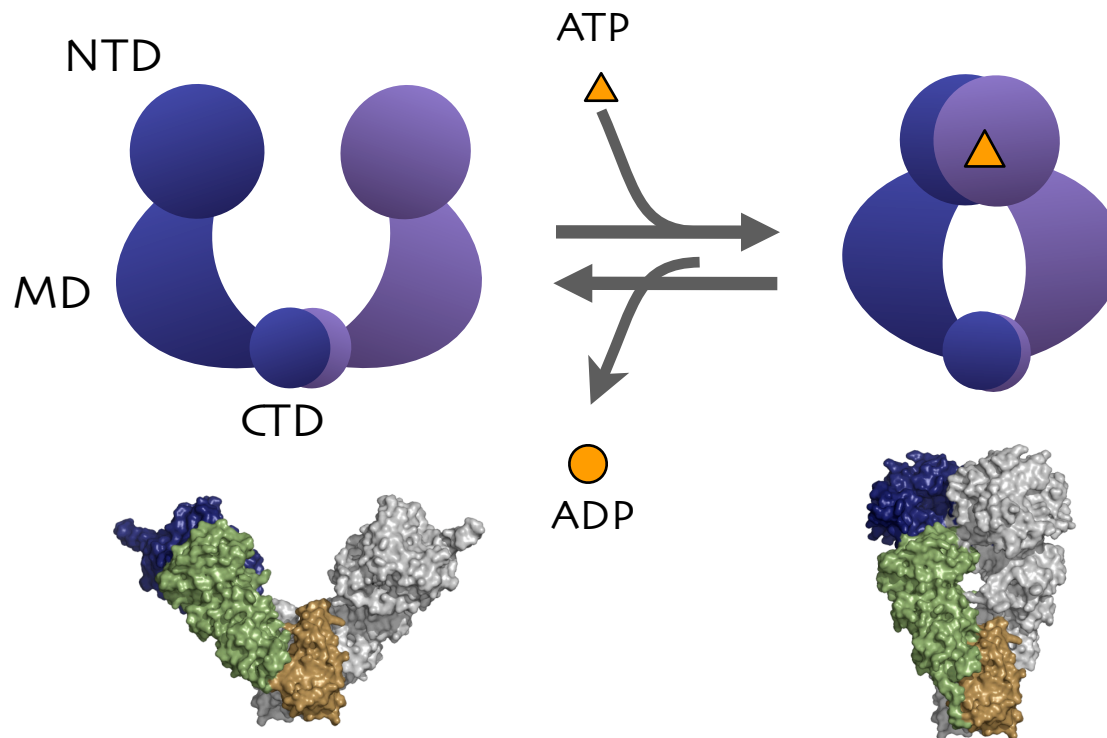


- Best model ($\chi^2 = 1.21$, improved from 2.96) clearly fits the SAXS data better than the original structure

ATP cycle of Hsp90?

K.A. Krukenberg, F. Förster, L. Rice, A. Sali, D.A. Agard, *Structure* **16**, 755-765, 2008.

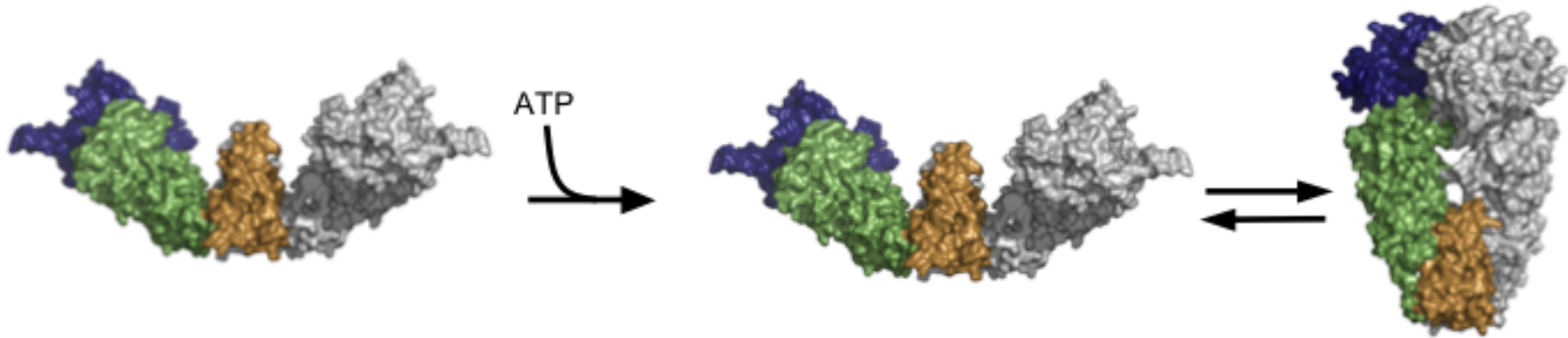
Previously proposed model:



based on Hsp90 crystallographic structures and GHKL superfamily structures (MutL, GyrB, Topo IV, VI).

SAXS maps Hsp90 states

K.A. Krukenberg, F. Förster, L. Rice, A. Sali, D.A. Agard, *Structure* **16**, 755-765, 2008.



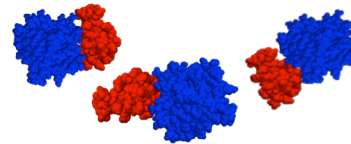
- Crystal structure of Hsp90+ATP is closed, but this does not match the SAXS spectrum
 - Crystallographic structures of opened and closed states are probably inaccurate representations of solution states.
- The apo structure of *E. coli* Hsp90 is wide open.
- *E. coli* ATP-Hsp90 is in equilibrium between the wide-open and closed states.

Example: SAXS and docking

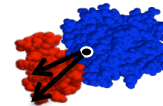
Input structures



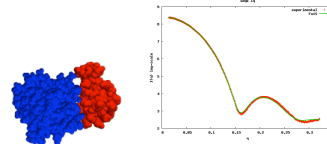
**Global Search by
Rigid Docking**



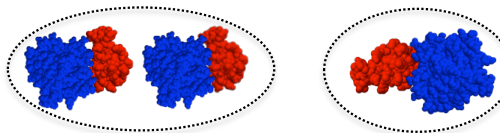
**Coarse SAXS Filtering by
Radius of Gyration**



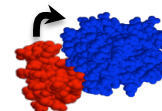
**SAXS Scoring by
Profile Fitting**



Clustering



**Local Search by
Conformational Refinement**



Dina Schneidman,
in press