class 10 Structural Bioinformatics (pt1)

Bryn Baxter (PIDA69038039)

#The PDB database

The main repository for biomolecular data is called the PDB (protein data bank) can be found at: https://www.rcsb.org/

Lets see what it contains in terms of type of molecule and method of structure determination (Analyze > PDB stats > By mol type and method)

```
pdbstats <- read.csv("Data Export Summary.csv")
pdbstats
```

```
           Molecular.Type   X.ray     EM    NMR Multiple.methods Neutron Other
1          Protein (only) 169,563 16,774 12,578              208      81    32
2 Protein/Oligosaccharide   9,939  2,839     34                8       2     0
3              Protein/NA   8,801  5,062    286                7       0     0
4     Nucleic acid (only)   2,890    151  1,521               14       3     1
5                   Other     170     10     33                0       0     0
6  Oligosaccharide (only)      11      0      6                1       0     4
    Total
1 199,236
2  12,822
3  14,156
4   4,580
5     213
6      22
```

> Q1: What percentage of structures in the PDB are solved by X-Ray and Electron Microscopy

Side Note: Because the data is inputted as characters, we cannot do math with it. Need to convert charcaters to integers by removing the comma in our numbers.

```
nocomma <- sub(",", "", pdbstats$X.ray)
sum(as.numeric(nocomma))
```

```
[1] 191374
```

Lets try **readr** package and its newer `read_csv()` function.

```
library(readr)
pdbstats <- read_csv("Data Export Summary.csv")
```

```
Rows: 6 Columns: 8
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): Molecular Type
dbl (3): Multiple methods, Neutron, Other
num (4): X-ray, EM, NMR, Total

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
pdbstats
```

```
# A tibble: 6 x 8
  `Molecular Type`   `X-ray`    EM   NMR `Multiple methods` Neutron Other  Total
  <chr>                <dbl> <dbl> <dbl>              <dbl>   <dbl> <dbl>  <dbl>
1 Protein (only)      169563 16774 12578                208      81    32 199236
2 Protein/Oligosacc~    9939  2839    34                  8       2     0  12822
3 Protein/NA            8801  5062   286                  7       0     0  14156
4 Nucleic acid (onl~    2890   151  1521                 14       3     1   4580
5 Other                  170    10    33                  0       0     0    213
6 Oligosaccharide (~      11     0     6                  1       0     4     22
```

The resulting column names are "untidy" with spaces and a mix of upper and lower case letters that will make workign with the columns a pain. We can use the **janitor** package with its `clean_names()` function to fix this for us.

```
colnames(pdbstats
         )
```

```
[1] "Molecular Type"   "X-ray"            "EM"               "NMR"
[5] "Multiple methods" "Neutron"          "Other"            "Total"
```

```
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```
df <- clean_names(pdbstats)
df
```

```
# A tibble: 6 x 8
  molecular_type          x_ray    em   nmr multiple_methods neutron other  total
  <chr>                   <dbl> <dbl> <dbl>            <dbl>   <dbl> <dbl>  <dbl>
1 Protein (only)         169563 16774 12578              208      81    32 199236
2 Protein/Oligosacchar~    9939  2839    34                8       2     0  12822
3 Protein/NA               8801  5062   286                7       0     0  14156
4 Nucleic acid (only)      2890   151  1521               14       3     1   4580
5 Other                     170    10    33                0       0     0    213
6 Oligosaccharide (onl~        11     0     6                1       0     4     22
```

What percent of structurs in pdb are determined by x-ray and electron microscopy?

```
n.xray <- sum(df$x_ray)
n.total <- sum(df$total)
n.xray
```

```
[1] 191374
```

```
n.total
```

```
[1] 231029
```

In Uniprot there are 253,206,171 protein sequences and there are only 231,029 known structures in the PDB. This is a tiny fraction!

```
231029/253206171*100
```

```
[1] 0.09124146
```

Next day we will see how bioinformatics methods can help predict structure from sequence with accuracy approaching X-ray methods.

```
n.xray/n.total*100
```

```
[1] 82.83549
```

Percent of Em structures?

```
n.em <- sum(df$em)

n.em/n.total*100
```

```
[1] 10.75017
```

> Q2: What proportion of structures in the PDB are protein?

```
round(df$total[1]/n.total *100, digits=2)
```

```
[1] 86.24
```

2. Molecular visualization with Mol*

Mol* is a new online structure viewer that is taking over the world of biomolecular visualization. Lets see how to use it from https://molstar.org/viewer/.

My first image from Mol* of HIV-Pr

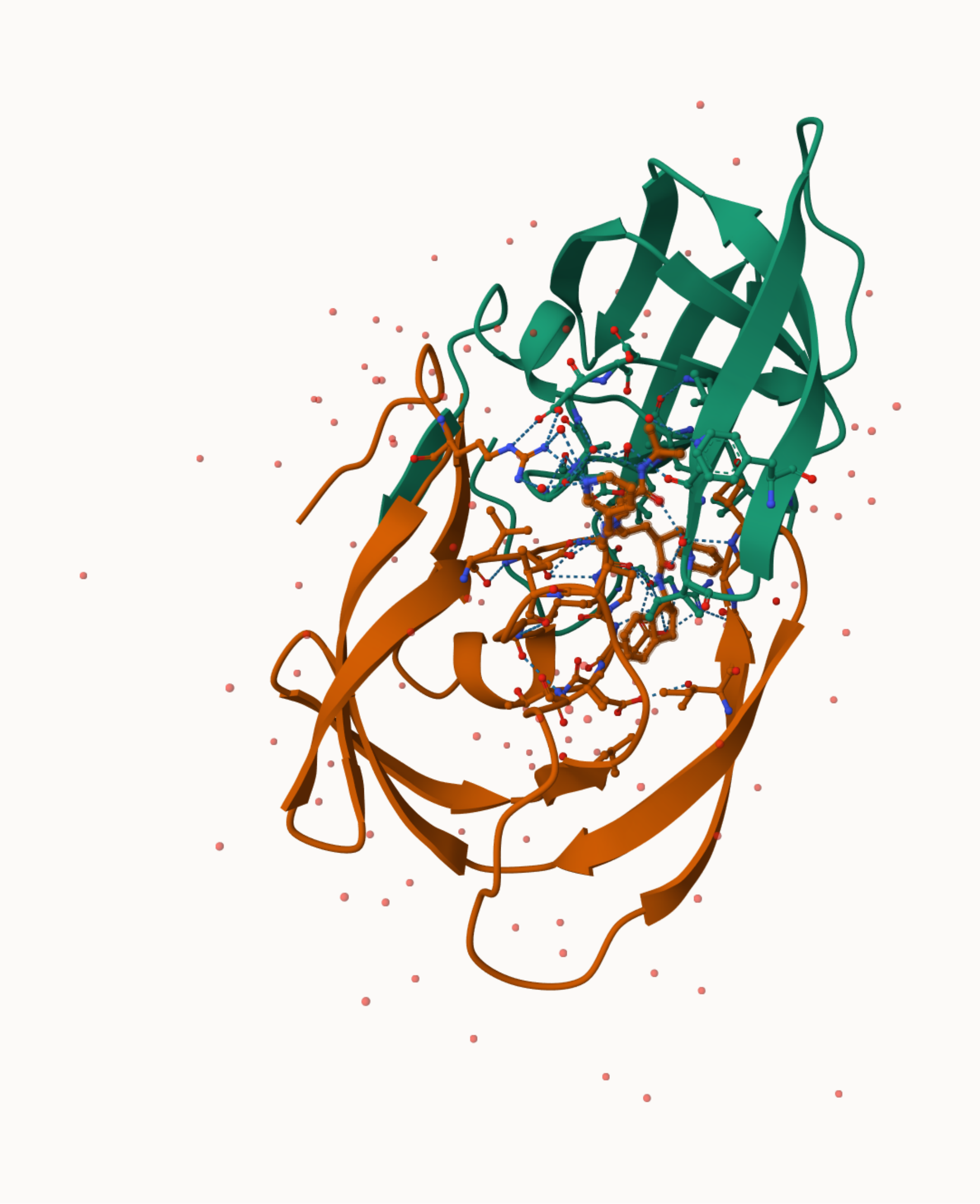


Figure 1: Fig.1 A first view of the HIV-Pr dimer

I want an image that shows the binding cleft for the MK1 inhibitor, an image of the most valuable water in human history, and an image showing the catalytic ASP amino-acids.

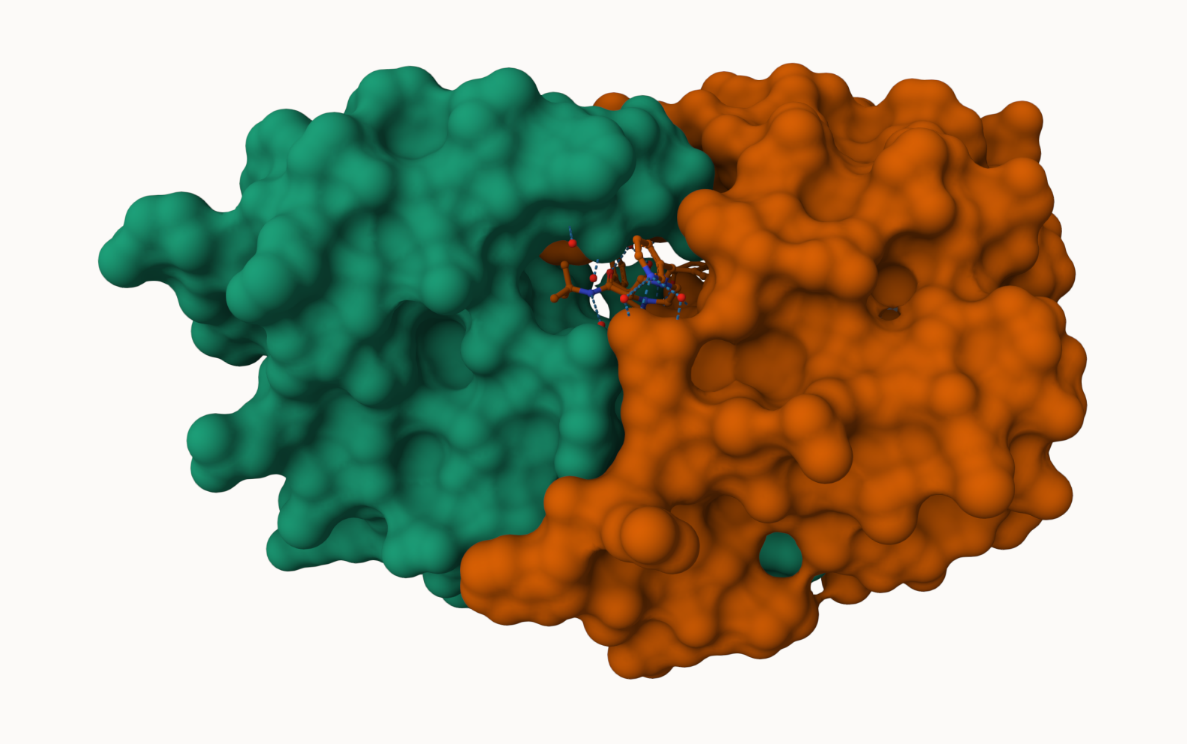


Figure 2: Fig 2. Binding cleft

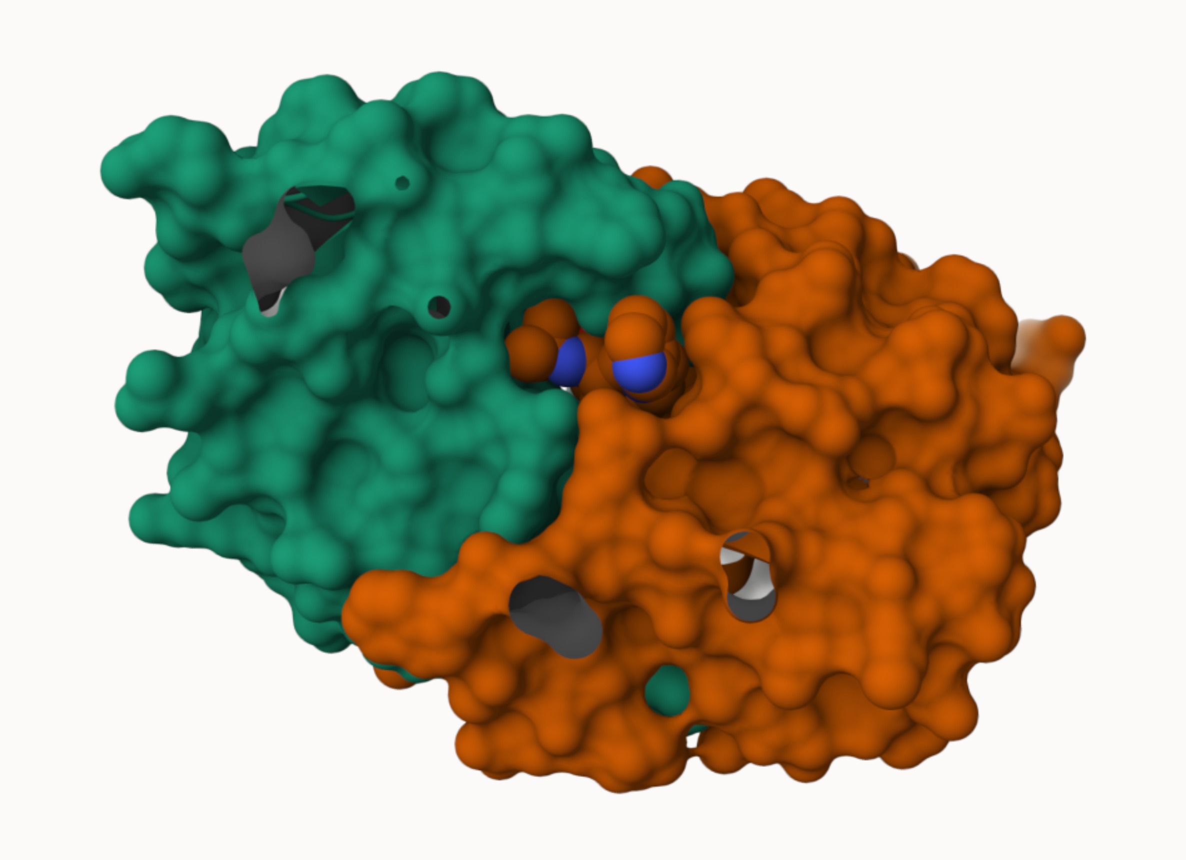


Figure 3: Fig. 3 Binding Cleft option 2

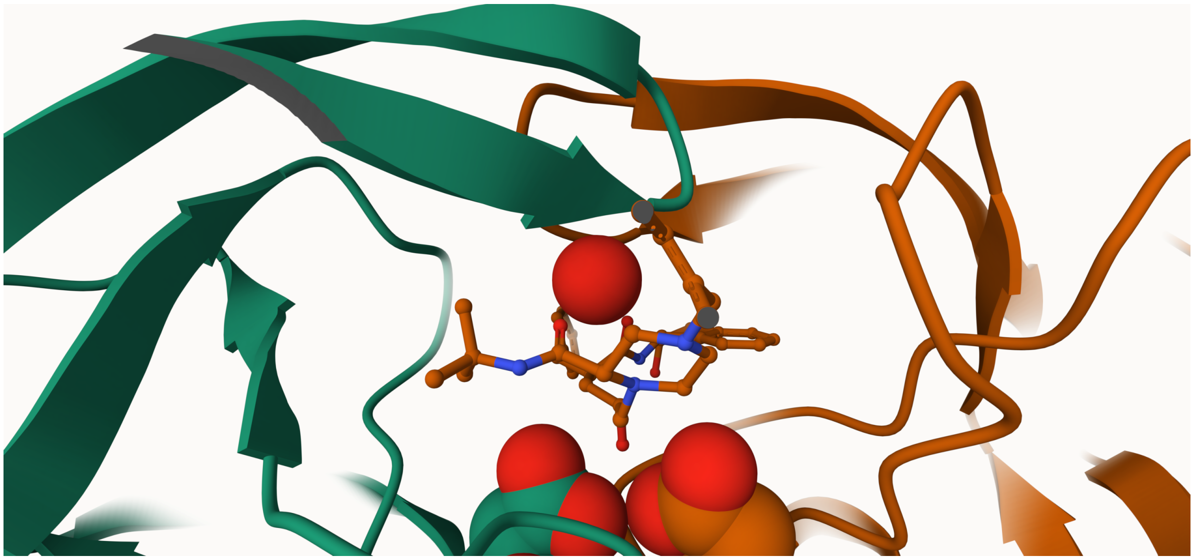


Figure 4: Fig. 4 Most expensive water and catalytic aspartic acids

##3. Using Bio3D package

This package has tons of tools and utilities for structural bioinformatics.

```
library(bio3d)

hiv <- read.pdb("1hsg")
```

```
  Note: Accessing on-line PDB file
```

```
hiv
```

```

 Call:  read.pdb(file = "1hsg")

   Total Models#: 1
     Total Atoms#: 1686,  XYZs#: 5058  Chains#: 2  (values: A B)

     Protein Atoms#: 1514  (residues/Calpha atoms#: 198)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 172  (residues: 128)
     Non-protein/nucleic resid values: [ HOH (127), MK1 (1) ]
   Protein sequence:
      PQITLWQRPLVTIKIGGQLKEALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYD
      QILIEICGHKAIGTVLVGPTPVNIIGRNLLTQIGCTLNFPQITLWQRPLVTIKIGGQLKE
      ALLDTGADDTVLEEMSLPGRWKPKMIGGIGGFIKVRQYDQILIEICGHKAIGTVLVGPTP
      VNIIGRNLLTQIGCTLNF

+ attr: atom, xyz, seqres, helix, sheet,
        calpha, remark, call
```

```
head(hiv$atom)
```

```
  type eleno elety  alt resid chain resno insert      x      y     z o     b
1 ATOM     1     N <NA>   PRO     A     1   <NA> 29.361 39.686 5.862 1 38.10
2 ATOM     2    CA <NA>   PRO     A     1   <NA> 30.307 38.663 5.319 1 40.62
3 ATOM     3     C <NA>   PRO     A     1   <NA> 29.760 38.071 4.022 1 42.64
4 ATOM     4     O <NA>   PRO     A     1   <NA> 28.600 38.302 3.676 1 43.40
5 ATOM     5    CB <NA>   PRO     A     1   <NA> 30.508 37.541 6.342 1 37.87
6 ATOM     6    CG <NA>   PRO     A     1   <NA> 29.296 37.591 7.162 1 38.40
  segid elesy charge
1  <NA>     N   <NA>
2  <NA>     C   <NA>
3  <NA>     C   <NA>
4  <NA>     O   <NA>
5  <NA>     C   <NA>
6  <NA>     C   <NA>
```

```
s <- pdbseq(hiv)
head(s)
```

```
  1   2   3   4   5   6 
"P" "Q" "I" "T" "L" "W" 
```

> Q. How long is this sequence/ how many amino acids are in the structrue?

```
length(s)
```

```
[1] 198
```

Predict fucntional motions

Lets read a new structure "6s36"

```
pdb <-read.pdb("6s36")
```

```
  Note: Accessing on-line PDB file
   PDB has ALT records, taking A only, rm.alt=TRUE
```

```
pdb
```

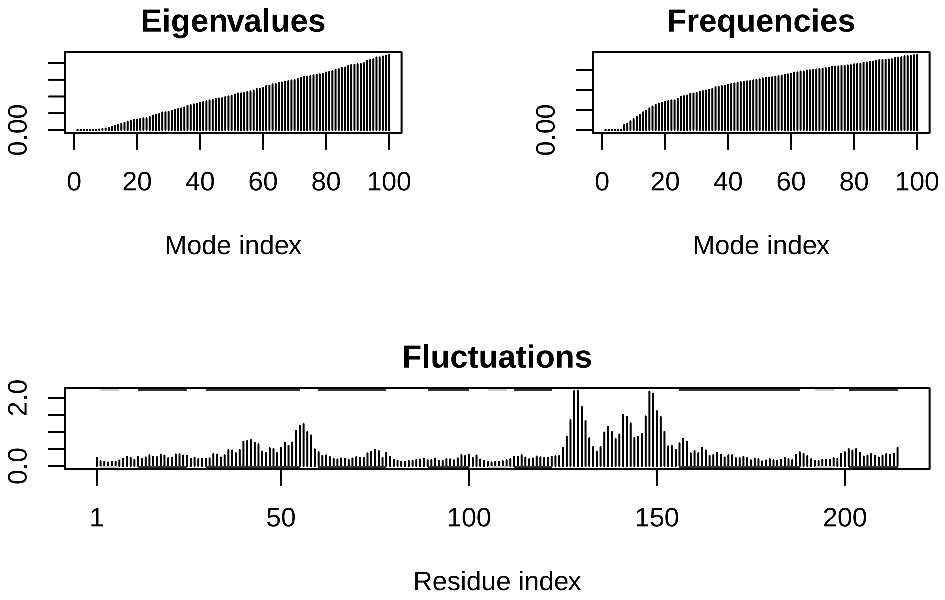
```


 Call:  read.pdb(file = "6s36")

   Total Models#: 1
     Total Atoms#: 1898,  XYZs#: 5694  Chains#: 1  (values: A)

     Protein Atoms#: 1654  (residues/Calpha atoms#: 214)
     Nucleic acid Atoms#: 0  (residues/phosphate atoms#: 0)

     Non-protein/nucleic Atoms#: 244  (residues: 244)
     Non-protein/nucleic resid values: [ CL (3), HOH (238), MG (2), NA (1) ]

   Protein sequence:
      MRIILLGAPGAGKGTQAQFIMEKYGIPQISTGDMLRAAVKSGSELGKQAKDIMDAGKLVT
      DELVIALVKERIAQEDCRNGFLLDGFPRTIPQADAMKEAGINVDYVLEFDVPDELIVDKI
      VGRRVHAPSGRVYHVKFNPPKVEGKDDVTGEELTTRKDDQEETVRKRLVEYHQMTAPLIG
      YYSKEAEAGNTKYAKVDGTKPVAEVRADLEKILG

+ attr: atom, xyz, seqres, helix, sheet,
        calpha, remark, call
```

We can run a NMA calculation on this structure:

```
m <- nma(pdb)
```

```
 Building Hessian...        Done in 0.06 seconds.
 Diagonalizing Hessian...   Done in 0.35 seconds.
```

```
plot(m, sse=pdb)
```



We can write out a wee trajectory of the predicted dynamics using the `mtkrj()` function:

```
mktrj(m, file="results.pdb")
```