

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
use Math::Trig ;
use strict;
```

```
my ($coor,$schnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $sch){ $schnum++; $sch=$ggg } ;
```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum >0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/^\.//;
$filename=~ s/\.pdb//;
#$filename=$schnum."_"$qnum."_"$filename.".dat";
$filename="$dir"/"$filename.".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $schnum qnum $qnum\n";
```

```
foreach my $m (sort { $a<=>$b } keys %coor){
my %qartets;
my %q= find_q( $coor{$m} );
```

```
# foreach my $q ( keys %qartets){ print join " ",@{ $qartets{$q} },"\n";
```

```
foreach my $q ( keys %qartets){
```

```
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
```

```
foreach my $res ( @{ $qartets{$q} }){
```

```
# print "$q $coor{$m}{ $res }{"N7"}->x,"\n";
```

```
$nx=$nx+ $coor{$m}{ $res }{"N9"}->x;
```

```
$ny=$ny+ $coor{$m}{ $res }{"N9"}->y;
```

```
$nz=$nz+ $coor{$m}{ $res }{"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{ $res }{"O6"}->x;
```

```
$oy=$oy+ $coor{$m}{ $res }{"O6"}->y;
```

```
$oz=$oz+ $coor{$m}{ $res }{"O6"}->z;
```

```
$r=$res;
```

```
}
```

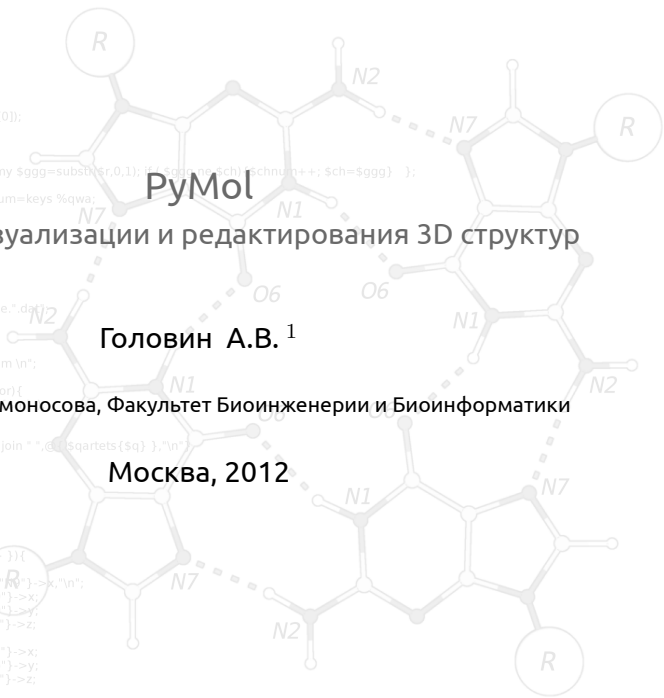
PyMol

как среда визуализации и редактирования 3D структур

Головин А.В. <sup>1</sup>

<sup>1</sup>МГУ им М.В. Ломоносова, Факультет Биоинженерии и Биоинформатики

Москва, 2012



# Содержание:

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
my ($R,$N1,$N2,$N7,$O6) = ();
```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

## Введение

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
```

## Визуализация с PyMol

```
$filename =~ s/\/.*//;
my $dir=$filename;
$filename="$dir".$filename.".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "$qnum $qnum $qnum\n";
```

## Selections

```
foreach my $m (sort {$a<=>$b} keys %coor){
my %qartets = %qwa; #find_quart( $coor{$m} );
my %q = find_q( $coor{$m} );
```

## Анимация

```
foreach my $q ( keys %qartets){ print join " ",@{$qartets{$q}},"\n";
}
foreach my $q ( keys %qartets){
```

## Моделирование и редактирование в PyMol

```
my $r;
foreach my $res ( @{$qartets{$q}}){
my $r;
```

## Скриптование в PyMol

```
$nx=$nx+ $coor{$m} {$res} {"N9"}->x;
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
$nz=$nz+ $coor{$m} {$res} {"N9"}->z;
```

```
$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Визуализация с PyMol

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
```

```
my $my $coor=read_pdb($
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $f ( sort key
```

```
my %qwa=find_quart( :
```

```
if ($sqnum >0){
#system("mkdir $ARGV
my $filename=$ARGV[
$filename="-- s/^~/V//;
$filename="-- s/\.pdb//;
# $filename=$schnum."
$filename="-- $mdir". $filei
print "$filename\n";
open OUT,">$filename
print OUT "#INFO chait
```

```
foreach my $m (sort {
my %qartets= %qwa
my %q= find_qf $co
```

```
# foreach my $q ( k
```

```
foreach my $q ( k
```

```
my $nx; my $ny
my $ox; my $oy
my $r;
```

```
foreach my $res
```

```
# print "$
```

```
$nx=$nx+ $
```

```
$ny=$ny+ $
```

```
$nz=$nz+ $
```

```
$ox=$ox+ $
```

```
$oy=$oy+ $
```

```
$oz=$oz+ $
```



# Для чего нужен PyMol

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
use File::Glob qw( :all );

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){ $chnum++; $ch=$ggg } };

my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=$ch.$qnum;
# $filename=$ch.$qnum."/".$ch.$qnum;
$filename=$ch.$qnum;
print "file: $filename\n";
open OUT, ">$filename";
print OUT "#INFO: chain $chnum group $qnum\n";
foreach my $m (sort {$a->$b} keys %{$coor{"0"}}){
my %qartets= %qwa; #find_quart( %coor{$m} );
my %q= find_q( %coor{$m} );
```

```
# foreach my $q ( keys %qartets){ print join " ",@{$qartets{$q}},"\n";
foreach my $q ( keys %qartets){
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @{$qartets{$q}}){
# print "$q %coor{$m}{ $res }{"N9"}->x,\n";
$nx=$nx+ %coor{$m}{ $res }{"N9"}->x;
$ny=$ny+ %coor{$m}{ $res }{"N9"}->y;
$nz=$nz+ %coor{$m}{ $res }{"N9"}->z;

$ox=$ox+ %coor{$m}{ $res }{"O6"}->x;
$oy=$oy+ %coor{$m}{ $res }{"O6"}->y;
$oz=$oz+ %coor{$m}{ $res }{"O6"}->z;
```

- Визуализация pdb и прочих файлов с координатами атомов
- Изготовление высококачественных изображений
- Начальное редактирование структур



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Системные требования

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } };
my $qwa=find_quart( $coor{"0"} ); my $sqnum=keys %qwa;
```

```
if ($sqnum > 0){
```

**Компьютер:** чем мощнее процессор и чем больше памяти, тем лучше

**3D монитор** не обязателен, но поддерживается

**Операционная система:** любая, под Linux проще установить, и он лучше работает с памятью.

```
# foreach my $q { keys %qartets } { print join " ", @{$qartets{$q}}, "\n";
```

```
foreach my $q { keys %qartets } {
```

```
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
# print "$q $coor{$m} {$res} {"N"}->x, "\n";
```

```
$nx=$nx+ $coor{$m} {$res} {"N9"}->x;
```

```
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
```

```
$nz=$nz+ $coor{$m} {$res} {"N9"}->z;
```

```
$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
```

```
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
```

```
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

# Как установить?

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );

#(my %$coor,my $chnum)=read_pdb($ARGV[0]);
my %$coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;
```

```
my %$qwa=find_quart( %$coor{"0"} ); my $qnum=keys %$qwa;
```

- Компиляция из исходников: <http://pymol.svn.sourceforge.net/>

```
if ($qnum > 0){
```

- Установка бинарных пакетов в Ubuntu Linux: `sudo apt-get install pymol`

- Установка бинарных пакетов в Windows:

- Ресурс для установки с python:

<http://www.lfd.uci.edu/~gohlke/pythonlibs/#pymol>

- Компиляция под Windows:

<http://arcib.dowling.edu/~darakevn/installerpaper.pdf>

```
foreach my $m (sort keys %$coor{"0"}){
my %$qartets= %$qwa{find_quart($coor{"0"}{$m})};
my %$q= find_quart($coor{"0"}{$m});
# foreach my $s ( @{$qartets{$q}} ){
foreach my $s ( @{$qartets{$q}} ){
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
foreach my $res ( @{$qartets{$q}} ){
# print "$q $coor{$m}{$res} {"$R"}->x,"n";
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# PyMol - это GPL программа?

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){ $chnum++; $ch=$ggg } };

my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

Да, PyMol это GPL-программа;

- исходный код доступен на [sourceforge.net](http://sourceforge.net)
- Бинарные пакеты для windows стоят денег и продаются: <http://pymol.org/academic.html>
- Бинарные пакеты для Linux собираются майтенерами

```

# foreach my $q ( keys %qartets){ print join " ",@{$qartets{$q}} ,"\n";
foreach my $q ( keys %qartets){

my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @{$qartets{$q}}){

print "$q $coor{$m}{$res}{"$R"}->x,\n";
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;

$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```

# PyMol

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

```
my $coor = read_pdb($ARGV[0]);
```

```
my $dir = `pwd`;
```

```
my $sch = `cat $dir`;
```

```
foreach
```

```
my %qw
```

```
if ($qu
```

```
#system
```

```
my $file
```

```
$filename
```

```
$filename
```

```
$filename
```

```
$filename
```

```
print "s
```

```
open OU
```

```
print OU
```

```
foreach
```

```
my %
```

```
my %
```

```
# fo
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

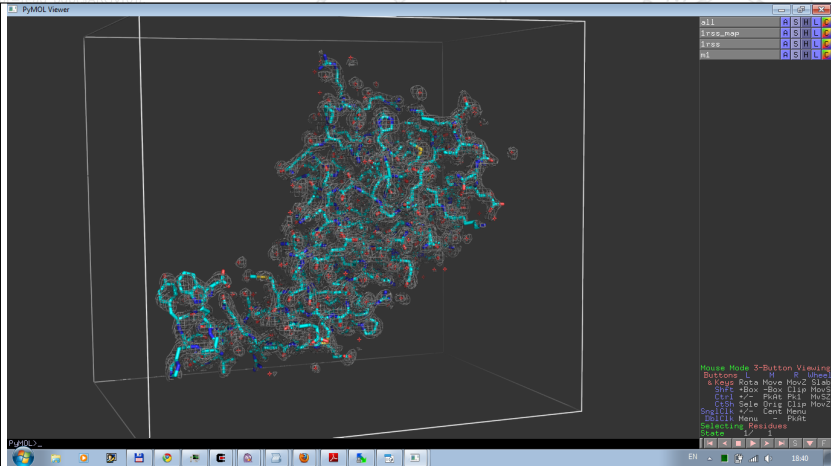
```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```



```
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```

```
#!usr/bin/perl
use Math::VectorReal qw( :all );
```

# ОСНОВНОЙ ВИД

```
my ($my $coor, my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir;
my $sch;
foreach
```

```
my %qv;
```

```
if ($qu
```

```
my $system
```

```
my $file
```

```
$filename
```

```
$filename
```

```
$filename
```

```
$filename
```

```
print "s
```

```
open OU
```

```
print OU
```

```
foreach
```

```
my %
```

```
my %
```

```
# fo
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

```
#
```

```
fo
```

The screenshot displays the PyMOL Molecular Graphics System interface. The main window shows a 3D molecular model of a protein structure, rendered with a cyan mesh overlay. The left sidebar contains a command list with options like 'Reset', 'Zoom', 'Orient', 'Draw', 'Ray', 'Unpick', 'Deselect', 'Rock', 'Get View', etc. The bottom right panel shows a list of objects: 'all', 'irss\_map', 'irss', 'n1', each with 'A S N L' buttons. The status bar at the bottom indicates 'Image size = 1146 x 698' and 'PyMOL\_...'.

```
$ny=$ny+ $coor{$m}{ $res}{ "N9" }->y;
$nz=$nz+ $coor{$m}{ $res}{ "N9" }->z;
```

```
$ox=$ox+ $coor{$m}{ $res}{ "O6" }->x;
$oy=$oy+ $coor{$m}{ $res}{ "O6" }->y;
$oz=$oz+ $coor{$m}{ $res}{ "O6" }->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Как загрузить структуру?

```

#(my %$coor,my $schnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } };

```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

- Из интернет:

- в меню выбрать соответствующий plugin

- или в командной строке: fetch 1xxx

```

if ($qnum) {
#system("mkdir $ARGV[1]");
my $filename=$ARGV[1];
$filename-- s/ / /;
$filename-- s/\ / /;
#$filename=$schnum;
$filename="$dir". $filename ".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $schnum qnum $qnum\n";

```

```

foreach my $m (sort { $a<=>$b } keys %coor){
my %qartets = %qwa; #find_quart( $coor{$m} );
my %q;

```

- Локальный файл:

- File->Open

```

my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

```

```

foreach my $res ( @ { $qartets{$q} } ){
#
print "$q $coor{$m}{ $res } {" $r "}->x,\n";

```

```

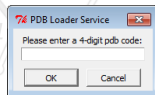
$nx=$nx+ $coor{$m}{ $res } {"N9"}->x;
$ny=$ny+ $coor{$m}{ $res } {"N9"}->y;
$nz=$nz+ $coor{$m}{ $res } {"N9"}->z;

```

```

$ox=$ox+ $coor{$m}{ $res } {"O6"}->x;
$oy=$oy+ $coor{$m}{ $res } {"O6"}->y;
$oz=$oz+ $coor{$m}{ $res } {"O6"}->z;

```



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Использование мыши

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;

```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

- Левый клик + движение = вращение молекулы

```

if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $fdir=$ARGV[1];
my $filename=$fdir."/";
my $filename=$fdir."/";
my $filename=$fdir."/";
my $filename=$fdir."/";
print "$filename\n";
open OUT, ">$filename";
print OUT "$ch chain $chnum $qnum $qwa{$r}\n";
}

```

- Средний клик + движение = перемещение молекулы
- Правый клик + движение вверх/вниз = приближение/удаление молекулы

```

foreach my $r ( sort keys %{$coor{"0"}} ){
my %q=find_quart( $coor{"0"} );
my %q=find( $coor{"0"} );
}

```

- Колесо = изменение уровня обрезания молекулы

- Все манипуляции относятся к камере, а не координатам структуры

```

foreach my $res ( @{$ $partets{$q} } ){
print "$q $coor{$m} {$res} {"$R"}->x,\n";
$nx=$nx+ $coor{$m} {$res} {"$R"}->x;
$ny=$ny+ $coor{$m} {$res} {"$R"}->y;
$nz=$nz+ $coor{$m} {$res} {"$R"}->z;

$ox=$ox+ $coor{$m} {$res} {"$R"}->x;
$oy=$oy+ $coor{$m} {$res} {"$R"}->y;
$oz=$oz+ $coor{$m} {$res} {"$R"}->z;
}

```

```
#!usr/bin/perl
use Math::VectorReal qw( :all );
```

# Меню объекта/выборки

```
my %m
my %co
my %dir
my %sch
foreach
```

```
if ($qu
#system
my $file
$filenam
$filenam
$filenam
$filenam
print "$
open OU
print OU
```

```
foreach
my %
my %
```

```
# fo
```

```
fo
```

```
#
```

The screenshot shows the PyMOL Viewer interface. A 3D molecular model is displayed in the center. A context menu is open over the model, listing the following objects and their corresponding action buttons:

Object	A	S	H	L	C
all	Button	Button	Button	Button	Button
1rss_map	Button	Button	Button	Button	Button
1rss	Button	Button	Button	Button	Button
m1	Button	Button	Button	Button	Button

Below the menu, the text **A,S,H,L,C** is displayed in red.

```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# A=Action

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"O"}} ){ my $sggg=substr($r,0,1); if ( $sggg ne $ch ){ $chnum++; $ch=$sggg
```

## Манипуляции с ориентацией

## Предустановки изображения и т.д.

## Манипуляция с объектом

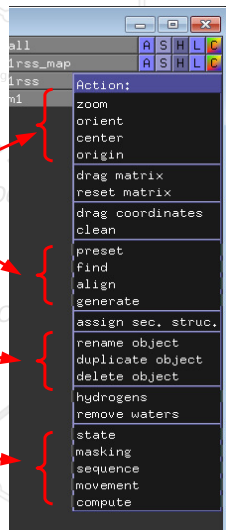
## Прочее

```
my %qwa=find_quart( %coor{"O"} ); my $qnum=keys %qwa;
if ($qnum > 0){
  #system("mkdir $ARGV[1]");
  my $filename=$ARGV[0];
  $filename="-- s/^.*\//";
  $filename="-- s/\.pdb//";
  print "$filename\n";
  open OUT,">$filename";
  print OUT "#INFO chain $chnum qnum $qnum\n";

  foreach my $m (sort { $a<=>$b } keys %coor){
    my %qartets = %qwa; #find quart( %coor{$m} );
    # foreach my $q { keys %qartets } { print join " ",@{$qartets{$q}},"n"; }
    foreach my $q { keys %qartets } {
      foreach my $ny; my $nz;
      foreach my $oy; my $oz;
      my $r;

      foreach my $res ( @{$qartets{$q}} ){
        print "$q $coor{$m}{$res} {"$R"}->x,"n";
        $nx=$nx+ $coor{$m}{$res} {"$R"}->x;
        $ny=$ny+ $coor{$m}{$res} {"$R"}->y;
        $nz=$nz+ $coor{$m}{$res} {"$R"}->z;

        $ox=$ox+ $coor{$m}{$res} {"O6"}->x;
        $oy=$oy+ $coor{$m}{$res} {"O6"}->y;
        $oz=$oz+ $coor{$m}{$res} {"O6"}->z;
```



```
#!usr/bin/perl
use Math::VectorReal qw( :all );
```

# S=Show, H=Hide

```

#(my %cco
my %ccor=
my $mdir=$A
my $sch, my
foreach my

my %qwa=

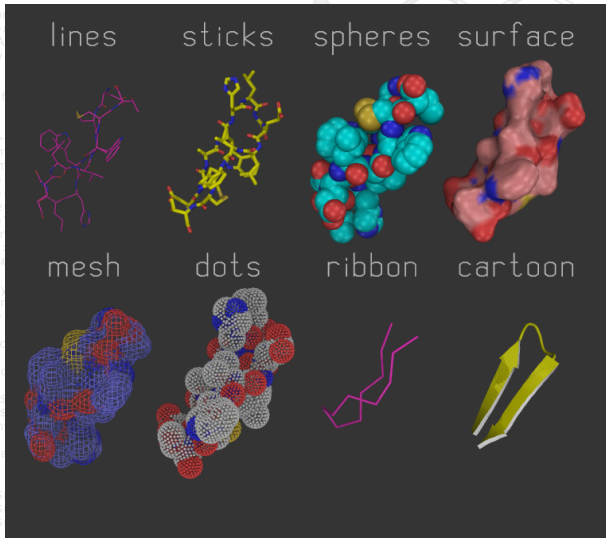
if ($qnum >
#system("m
my $filename
$filename=
$filename=
# $filename
$filename=
print "$filer
open OUT, ">
print OUT ">

foreach my
my %qar
my %q=

# foreach
foreach
my $
my $
m
fore:

#
$
$
$

```



all	A S H L P
irss_map	A S H L P
irss	Show:
m1	as
irss_e_chg	lines
irss_e_map	sticks
irss_e_pot	ribbon
(bk)	cartoon
(sеле)	label
(бет)	cell
nonbonded	
dots	
spheres	
nb_spheres	
mesh	
surface	
organic	
main chain	
side chain	
disulfides	

```

Mouse Mode 3-Button Viewing
Buttons L M R Wheel
a Keys Rota Move MovZ Slab
Shft +Box -Box Clip MovS
Ctrl +/- PkAt Pk1 MvSZ
CtrlSh Sele Orig Clip MovZ
SingleClick +/- Cent Menu
DoubleClick Menu - PkAt
Selecting Residues
State 1/ 1

```

```

$ox=$ox+ $ccor{$m} {$res} {"O6"}->x;
$oy=$oy+ $ccor{$m} {$res} {"O6"}->y;
$oz=$oz+ $ccor{$m} {$res} {"O6"}->z;

```



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
my $dir = "C:\Program Files\PyMOL\";
my $q = "1.00";
```

# C=Color

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
my $my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $fr ( sort keys %{$coor{"0"}} ){ my $
my $my $qwa=find_quart( $coor{"0"} ); my $schnum
```

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename-- s/\^.*\///;
$filename-- s/\.pdb//;
#$filename=$schnum." ".$qnum." ".$filename."
$filename="$mdir"."$filename.".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $schnum qnum $qnum\n";
```

```
foreach my $m (sort { $a-<=>$b } keys %coor){
my %qartets = %qwa ; #find_quart( $coor{$m} );
my %q = find_q( $coor{$m} );
```

```
# foreach my $q { keys %qartets } { print join
foreach my $q { keys %qartets } {
```

```
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
```

```
foreach my $res (@{ $qartets{$q} })
```

```
# print "$q $coor{$m} {$res} {"N
$nx=$nx+ $coor{$m} {$res} {"N9"};
$ny=$ny+ $coor{$m} {$res} {"N9"};
$nz=$nz+ $coor{$m} {$res} {"N9"};
```

```
$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

The screenshot shows the PyMOL Viewer interface. On the left, a 3D molecular model is displayed with a color scale from blue (low) to red (high). The right side of the window contains a menu for 'Color' and 'Mouse Mode'.

Color	Mouse Mode
ell	Buttons 1 - M, R Wheel
lrsr_map	& Keys Rotate Move MovZ Slab
m1	ShFt +Box -Box Clip MovS
lrsr_e	Dct1 +/- PKAt Pk1 MvS2
lrsr_e	Dctn Sele Driz Clip MovZ
lrsr_e	ShfClk +/- Cent Menu - PKAt
(bk)	DbfClk Menu - PKAt
(sle)	Selecting Residues
(bet)	State 1/ 1

Below the menu, the text 'Mouse Mode 3-Button Viewings' is visible, followed by a list of mouse actions and their corresponding keyboard shortcuts.



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Операторы множеств

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"O"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg };
my %qwa

```

- Логические операторы AND, OR, NOT
- Операция OR может быть записана как " , " .

```

if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename="";
$filename="-- $ch $dir";
$filename="-- $ch $dir";
# $filename="-- $ch $dir";
$filename="-- $dir $ch $dir";
print "$filename";
open OUT,">$filename";
print OUT "$ch $ch $chnum";
foreach my $m ( sort { $a-<=>$b } keys %coor ){
my %qwa
my %qwa

```

Упражнение: Документ PDB содержит описание структуры, состоящей из белка, фрагмента ДНК и молекул воды. Что получится, если задать следующие команды ?

```

select protein or dna
select protein and dna
select not water

```

- Оператор WITHIN(...)
- ```

select all within 3.5 of resi 20
select s1, (byres n. ca) within 3.5 of resn LIG

```

```

foreach my $res ( @{ $qartets{$q} } ){
#
print "$q $coor{$m} {$res} {" $ch " }->x,"n";
$nx=$nx+ $coor{$m} {$res} {" $ch " }->x;
$ny=$ny+ $coor{$m} {$res} {" $ch " }->y;
$nz=$nz+ $coor{$m} {$res} {" $ch " }->z;

$ox=$ox+ $coor{$m} {$res} {" $ch " }->x;
$oy=$oy+ $coor{$m} {$res} {" $ch " }->y;
$oz=$oz+ $coor{$m} {$res} {" $ch " }->z;

```

# Help selections

```

#!/usr/bin/perl
use Math::VectorReal qw( :all );

#(my %$coor,my $schnum)=read_pdb($ARGV[0]);
my %$coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch,my $schnum;
foreach my $f ( sort keys %{$coor{"O"}} ){ ( my $qgg=substr($f,0,1); if ( $qgg ne $sch ){ $schnum++; $sch=$qgg } );

my %$qwa=find_quart( $coor{"O"} ); my $qnum=keys %$qwa;
name <atom names>          n. <atom names>
resn <residue names>      r. <residue names>
resi <residue identifiers> i. <residue identifiers>
chain <chain-ID>         c. <chain identifiers>
id <original-index>
hydrogen
all                         *
visible                     v.
hetatm
byres <selection>         br. <selection>
byobj <selection>        bo. <selection>
around <distance>        a. <distance>
expand <distance>       e. <distance>
in <selection>          keys %$qartets{ } { print join " ", @$qartets{$q} }, "\n"
like <selection>        l. <selection>

<selection> within <distance> of <selection>
<selection> w. <distance> of <selection>

foreach my $res ( @ { $qartets{$q} } ){
#
print "$q $coor{$m}{ $res } { "R" }->x, "\n";
$nx=$nx+ $coor{$m}{ $res } {"N9"}->x;
$ny=$ny+ $coor{$m}{ $res } {"N9"}->y;
$nz=$nz+ $coor{$m}{ $res } {"N9"}->z;

$ox=$ox+ $coor{$m}{ $res } {"O6"}->x;
$oy=$oy+ $coor{$m}{ $res } {"O6"}->y;
$oz=$oz+ $coor{$m}{ $res } {"O6"}->z;

```

Длинное

Короткое



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Примеры выборки

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;

```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

## sel=select

```

#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
my $filename=$ARGV[1];
my $filename="-- s/\.pdb//";
my $filename="$dir/$chnum??.?.$filename";
my $filename="";
print "$filename\n";
open OUT,">$filename";
print OUT;

```

- **sel s1, n. ca and c. A** : все атомы CA в цепи A
- **sel s2, n. ca and (c. A or c. B)** : атомы CA цепей A и B
- **sel s3, resn GLU and resi 100** : остаток 100 если он GLU
- **sel s4, resi 100-120+130** : атомы остатков 100-120 и 130
- **sel s5, byres( name CG)** : атомы остатков где есть CG

```

foreach my $m (sort {$a<=>$b} keys %coor){
my %q = find_quart($coor{$m});
my %q = find_quart($coor{$m});

#
foreach my $q ( keys %qartets){

my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @{$qartets{$q}} ){

print "$q $coor{$m}{$res} {"$r"}->x,"n";
$nx=$nx+ $coor{$m}{$res} {"$r"}->x;
$ny=$ny+ $coor{$m}{$res} {"$r"}->y;
$nz=$nz+ $coor{$m}{$res} {"$r"}->z;

$ox=$ox+ $coor{$m}{$res} {"O6"}->x;
$oy=$oy+ $coor{$m}{$res} {"O6"}->y;
$oz=$oz+ $coor{$m}{$res} {"O6"}->z;

```



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Иерархическое определение выборки

```
#!/(my %coor,my $chain)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $idir=$ARGV[1];
my $ch, my $chain;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chain++; $ch=$ggg } };
```

```
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
$filename="$idir.pdb/";
$filename="$idir/$chainname.dat";
$filename="$idir/$chainname.dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $chain qnum $qnum\n";
```

Легко увидеть иерархию правым  
кликом по атому

```
foreach my $m ( sort { $a-<=>$b } keys %coor ){
#system("mkdir $ARGV[1]");
$filename="$idir.pdb/";
$filename="$idir/$chainname.dat";
$filename="$idir/$chainname.dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $chain qnum $qnum\n";
```

**sel s1, a/102/cz : атом cz в остатке 102**  
**sel s2, 100-120/N and c. A : атомы N в остатках 100-120 цепи a**  
**sel s3, a/100+120/ : все атомы остатков 100 и 120 в цепи A**

```
my $ix, my $ny, my $nz;
my $ox, my $oy, my $oz;
my $r;

foreach my $res ( @{$ $qartets{$q} } ){
#
print "$q $coor{$m} {$res} {" "N" }->x,\n";
$nx=$nx+ $coor{$m} {$res} {"N9"}->x;
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
$nz=$nz+ $coor{$m} {$res} {"N9"}->z;

$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

```
/1rss//A/ARG*102/CZ
drag object matrix
drag object coords
atom
residue
chain
segment
object
molecule
Fragment
fragment+joint(s)
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
use ...
```

# Трассировка лучей, команда ray

Подробнее: <http://www.pymolwiki.org/index.php/Ray>



No ray



ray\_trace\_mode,0



ray\_trace\_mode,1



ray\_trace\_mode,2



ray\_trace\_mode,3

```
if (
#s
my
sfi
sfi
#$
sfi
pri
open OUT,">$filename";
my $OUT="#INFO chain $chain qnum $qnum ln";
foreach my $m (so
my %qartets= %
my %q= find_q(
#
foreach my $q
foreach my $c
my $nx; my !
my $ox; my !
my $r;
foreach my $
#
print
$nx=$nx+ $coor{$m} {$res} {"N9"}->x;
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
$nz=$nz+ $coor{$m} {$res} {"N9"}->z;
$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Настройки изображения

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;

```

```

my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
http://www.pymolwiki.org/index.php/Category:Settings

```

- PyMol содержит порядка 600 настроек
- Не все документированы
- Большинство интуитивно понятны

- Настройки доступны через меню или в командной строке набрать:  
*set первые буквы имени опции и клавиша tab для построения*

```

# foreach my $s ( sort keys %$quartets ){ print join " ", @$quartets{$s} }, "\n";
#
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @ { $quartets{$s} } ){
#
print "$s $coor{$s}{$res} {"$r"}->x,\n";
$nx=$nx+ $coor{$s}{$res} {"$r"}->x;
$ny=$ny+ $coor{$s}{$res} {"$r"}->y;
$nz=$nz+ $coor{$s}{$res} {"$r"}->z;

$ox=$ox+ $coor{$s}{$res} {"O6"}->x;
$oy=$oy+ $coor{$s}{$res} {"O6"}->y;
$oz=$oz+ $coor{$s}{$res} {"O6"}->z;

```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Примеры

```
#[my %coor,my $snum]=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $snum;
```

## #initial setup

```
iewport 600, 600 --- размер графического окна
```

```
set auto_zoom, off --- не приближать новые объекты
```

```
set auto_show_lines, off --- не показывать линии автоматически
```

```
set auto_show_selections, off --- не показывать выборку автоматически
```

## #cartoon parameters

```
set cartoon_fancy_helices,1 --- изменение вида спиралей
```

```
set cartoon_highlight_color, grey60 ---цвет внутренней стороны спиралей
```

```
set cartoon_dumbbell_length,1.0 ---ширина ленты в спирали
```

```
set cartoon_rect_length,1.40000 --- ширина ленты в бета
```

```
set cartoon_loop_radius,0.3 --- толщина неструкт. участка
```

```
set cartoon_smooth_loops=0 --- без сглаживания
```

```
foreach my $res (@{ $qartets{$sq} }) {
# print "$q $coor{$sm} {$res} {"R"}->x,"n";
$nx=$nx+ $coor{$sm} {$res} {"N9"}->x;
$ny=$ny+ $coor{$sm} {$res} {"N9"}->y;
$nz=$nz+ $coor{$sm} {$res} {"N9"}->z;
```

```
$ox=$ox+ $coor{$sm} {$res} {"O6"}->x;
$oy=$oy+ $coor{$sm} {$res} {"O6"}->y;
$oz=$oz+ $coor{$sm} {$res} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Анимация в PyMol

Если структура содержит более чем одну модель, то в PyMol можно анимировать движение молекулы переходом от одной модели к другой

```
my $q=($ARGV[0]);
my $sch=$ARGV[1];
my $sch2=$ARGV[2];
my $sch my $schnum;
my $q($q) ( my $sggg=substr($q,0,1); if ( $sggg ne $sch){ $schnum++; $sch=$sggg } );
my %qwa=find_quart( $coor("0") ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/\^.*\///;
$filename=~ s/\./_./;
#$filename=$schnum."_".$sch;
$filename="$dir"/$filename;
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $sch\n";
```

```
foreach my $m (sort {$a<
my %qartets= %qwa; #
my %q= find_q( $coor($
```

```
# foreach my $q { keys
```

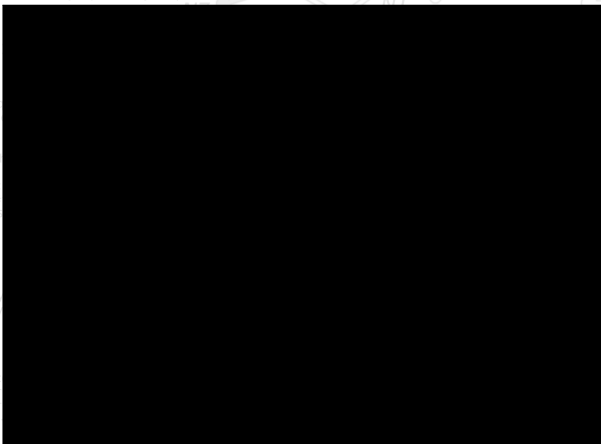
```
foreach my $q { keys
```

```
my $nx; my $ny; my
my $ox; my $oy; my
my $r;
```

```
foreach my $res {
```

```
# print "$q $sch
$nx=$nx+ $coor
$ny=$ny+ $coor
$nz=$nz+ $coor
```

```
$ox=$ox+ $coor($m){ $res }{"O6"}->x;
$oy=$oy+ $coor($m){ $res }{"O6"}->y;
$oz=$oz+ $coor($m){ $res }{"O6"}->z;
```



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# АНИМАЦИЯ, ОСНОВЫ

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
my $my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } ;
```

```
my %qwa=find_quart( $coor{"0"} ); my $sqnum=keys %qwa;
```

## GUI:

Вращение вокруг объекта на N секунд:

- Movie->Program->Camera->X-Roll->N Seconds
- Movie->Program->Camera->Y-Roll->N Seconds

## Покачивание:

- Movie->Program->Camera->X-Rock->X-Degrees over N-Seconds

```
foreach my $q ( keys %qartets ){
    my $nx; my $ny; my $nz;
    my $ox; my $oy; my $oz;
    my $r;

    foreach my $res ( @{$ $qartets{$q} } ){
        print "$q $coor{$m} {$res} {"$R"}->x,"n";
        $nx=$nx+ $coor{$m} {$res} {"$R"}->x;
        $ny=$ny+ $coor{$m} {$res} {"$R"}->y;
        $nz=$nz+ $coor{$m} {$res} {"$R"}->z;

        $ox=$ox+ $coor{$m} {$res} {"$R"}->x;
        $oy=$oy+ $coor{$m} {$res} {"$R"}->y;
        $oz=$oz+ $coor{$m} {$res} {"$R"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Пример

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){ $chnum++; $ch=$ggg } ;
```

```
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum) {
```

- Action->Preset->Technical (viewer gui)

- Scene->Store->F1

- zoom i. 90 # увеличение остатка 90

- Scene->Store->F2

- Movie->Program->Scene Loop->Y-Rock->4 Seconds Each

- File-> Save movie

```

#
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @{$ $qartets{$ $q } } ){
#
print "$q $coor{$ $m }{$ $res }{"R"}->x,"n";
$nx=$nx+ $coor{$ $m }{$ $res }{"N9"}->x;
$ny=$ny+ $coor{$ $m }{$ $res }{"N9"}->y;
$nz=$nz+ $coor{$ $m }{$ $res }{"N9"}->z;

$ox=$ox+ $coor{$ $m }{$ $res }{"O6"}->x;
$oy=$oy+ $coor{$ $m }{$ $res }{"O6"}->y;
$oz=$oz+ $coor{$ $m }{$ $res }{"O6"}->z;

```

# Результат

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %

my %qwa=find_quart( $coo

if ($qnum >0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/^\s*//;
$filename=~ s/\.pdb//;
#$filename=$chnum."_".$
$filename="$dir/".$filename;
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $c

foreach my $m (sort {$a<
my %qartets = %qwa ; #
my %q= find_q( $coor($

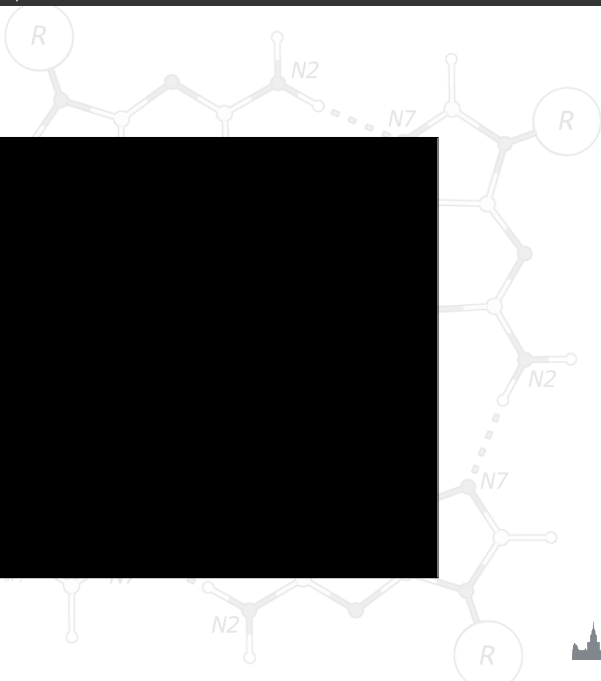
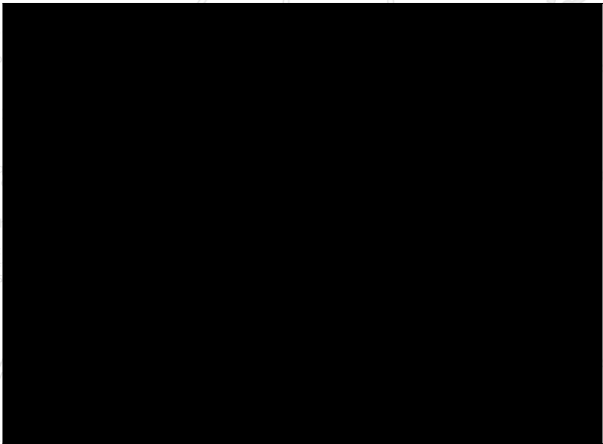
# foreach my $q { keys
foreach my $q { keys

my $nx; my $ny; my
my $ox; my $oy; my
my $r;

foreach my $res {

# print "$q So
$nx=$nx+ $coor{$m}{$res}{ "N9"}->x;
$ny=$ny+ $coor{$m}{$res}{ "N9"}->y;
$nz=$nz+ $coor{$m}{$res}{ "N9"}->z;

$ox=$ox+ $coor{$m}{$res}{ "O6"}->x;
$oy=$oy+ $coor{$m}{$res}{ "O6"}->y;
$oz=$oz+ $coor{$m}{$res}{ "O6"}->z;
```





```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Анимация, терминология

```
my ($my $coor, my $chnum)=read_pdb($ARGV[0]);
my $my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $chnum++; $sch=$ggg } ;
```

my %o

- **Объект и выборка : смотри выше**

- **states: конформация или набор координат**

- **scene: позиция камеры и отображение**

- **frames: это кадры в анимации, содержит state и scene**

```
if ($qnum)
#system("cd $mdir; mv $filename $qnum");
my $filename=$ARGV[0];
$filename=$ARGV[0];
$filename=$ARGV[0];
# $filename=$chnum . "_" . $qnum . "/" . $filename . ".dat";
$filename=$chnum . "_" . $qnum . "/" . $filename . ".dat";
print "file: " . $filename . "\n";
open OUT, ">$filename";
print OUT "#INFO chain $chnum qnum $qnum \n";
```

```
foreach my $m (sort { $a-<=>$b } keys %coor){
my %qartets = %qwa ; #find quart( $coor{$m} );
my %q = find_q( $coor{$m} );
```

```
# foreach my $q ( keys %qartets ){ print join " ", @{$qartets{$q}} , "\n";
```

```
foreach my $q ( keys %qartets ){
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
# print "$q $coor{$m} {$res} {"N"}->x, "\n";
$nx=$nx+ $coor{$m} {$res} {"N9"}->x;
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
$nz=$nz+ $coor{$m} {$res} {"N9"}->z;
```

```
$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

Movie panel:



```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Анимация, команды

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
```

```
my $my $coor=read_pdb($ARGV[0]);
```

```
my $mdir=$ARGV[1];
```

```
my $sch, my $schnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } ;
```

```
my $my $qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

```
my $my $qwa=find_quart( $coor{"0"} );
```

**mset 1 -55** : задать анимацию от 1 до 55 state на 55 кадров (frames)

**mset 1 x90** : задать анимацию первого state от 1 до 90 кадров

**mset 1 x30 1 -15 15 x30 15 -1** : первые 30 кадров state 1, следующие 15 кадров это состояния 1-15, следующие 30 кадров состояние 15, следующие 15 кадров состояния от 15 до 1

```
# foreach my $q { keys %qartets } { print join " ", @qartets{$q} , "\n";
```

```
foreach my $q { keys %qartets } {
```

```
my $nx; my $ny; my $nz;
```

```
my $ox; my $oy; my $oz;
```

```
my $r;
```

```
foreach my $res ( @qartets{$q} ) {
```

```
print "$q $coor{$m}{$res} {"N"}->x, "\n";
```

```
$nx=$nx+ $coor{$m}{$res} {"N"}->x;
```

```
$ny=$ny+ $coor{$m}{$res} {"N"}->y;
```

```
$nz=$nz+ $coor{$m}{$res} {"N"}->z;
```

```
$ox=$ox+ $coor{$m}{$res} {"O"}->x;
```

```
$oy=$oy+ $coor{$m}{$res} {"O"}->y;
```

```
$oz=$oz+ $coor{$m}{$res} {"O"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Анимация, команды

```
my ($my $coor, my $snum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
```

**mview** : команда для создания ключевых точек

**Пример** : \$coor{"0"} ); my \$snum=keys %qwa;

- mset 1 x100**

- frag leu** # создаём LEU

- orient** # ориентируем его

- mview store** # запоминаем ключевую точку

- frame 100** # переходим в кадр 100

- zoom ID 10** # увеличиваем атом №10

- mview store** # запоминаем ключевую точку

- mview interpolate** # делаем интерполяцию

```
#
print "$q $coor{$snum} {$sres} {"$snum"}->x,"$n";
$nx=$nx+ $coor{$snum} {$sres} {"$snum"}->x;
$ny=$ny+ $coor{$snum} {$sres} {"$snum"}->y;
$nz=$nz+ $coor{$snum} {$sres} {"$snum"}->z;

$ox=$ox+ $coor{$snum} {$sres} {"O6"}->x;
$oy=$oy+ $coor{$snum} {$sres} {"O6"}->y;
$oz=$oz+ $coor{$snum} {$sres} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Результат mview

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %
my %qwa=find_quart( $coor
```

```
if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/~/./;
$filename=~ s/\.pdb//;
#$filename=$chnum."_".$
$filename="$dir"/.$filename;
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $c
```

```
foreach my $m (sort {$a<
my %qartets = %qwa ; #
my %q= find_q( $coor($
```

```
# foreach my $q { keys
```

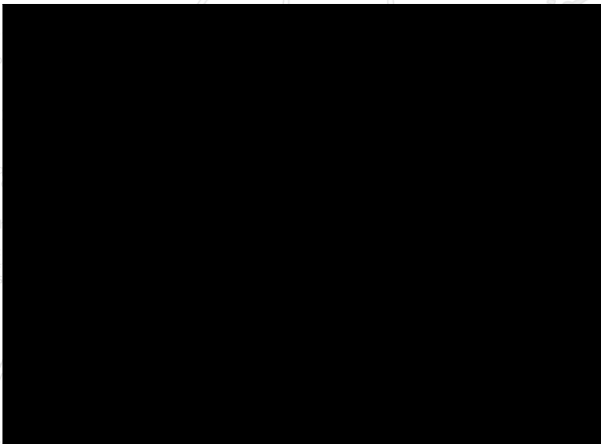
```
foreach my $q { keys
```

```
my $nx; my $ny; my
my $ox; my $oy; my
my $r;
```

```
foreach my $res (
```

```
# print "$q $o";
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```



# Дополнительные команды

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
my $q = $ARGV[0];
my $R = $ARGV[1];
my $sch = $ARGV[2];
my $schnum = $ARGV[3];
foreach my $f ( sort keys %{$coor{"0"}} ) { my $ggg = substr($f,0,1); if ( $ggg ne $sch ) { $schnum++; $sch=$ggg } ;
```

```
my %qwa = find_quart( $coor{"0"} ); my $qnum = keys %qwa;
```

- **mmatrix** : устанавливает вид для первого кадра
- **util.mrock** : скачивание сцены на определённый угол
- **util.mrock(start, finish, angle, phase, loop-flag)**
- **util.mroll** : вращение вокруг оси Y

```
foreach my $m ( 0..$qnum-1 ) {
my $q = $qwa{ $m };
my %q = find_q( $coor{$m} );
```

- **mdo** : (устарело) запуск какой-либо команды в заданном кадре

```
foreach my $sres ( @{$qartets{$q}} ) {
print "$q $coor{$m} {$sres} {" $R "}->x,\n";
$nx=$nx+ $coor{$m} {$sres} {"N9"}->x;
$ny=$ny+ $coor{$m} {$sres} {"N9"}->y;
$nz=$nz+ $coor{$m} {$sres} {"N9"}->z;

$ox=$ox+ $coor{$m} {$sres} {"O6"}->x;
$oy=$oy+ $coor{$m} {$sres} {"O6"}->y;
$oz=$oz+ $coor{$m} {$sres} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Сохранение анимации

```
#!/(my %coor,my $schnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $sch){ $schnum++; $sch=$ggg };
```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

## Старый путь:

```
set ray_trace_frames,1
```

```
mpng mymovie
```

Нужны программы avidemux, Virtual Dub, mencoder для того, чтобы собрать ролик с нужным сжатием (кодек)

**Новый путь: File->Save movie** ; есть недостаток, старый офис понимает только avi с определённым кодеком

```
foreach my $q ( keys %qartets){
```

```
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
```

```
foreach my $res ( @{$ $qartets{$q} }){
```

```
# print "$q $coor{$m}{$res}{"$R"}->x,"n";
```

```
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
```

```
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
```

```
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
```

```
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
```

```
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Моделирование и редактирование в PyMol

```
my ($coor,$sch,$snum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch,$snum;
```

- Можно перемещать объекты и сохранять их новые координаты

```
my %qwa=find_quart( $coor{"O1"} ); my $qnum=keys %qwa;
```

- Можно рассчитать вторичную структуру

```
if ($qnum > 0){
```

```
  #system( "cp $coor $mdir/$snum/$snum.dat" );
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

```
  $filename=$mdir/$snum/$snum.dat;
```

- Можно вносить мутации в белок (но не НК)

- Можно конвертировать L->D аминокислоты

- Можно добавлять протоны

- Можно выравнивать в пространстве молекулы

- Можно добавлять некоторые фрагменты из библиотеки и собственные

```
print "$q $coor{$snum} {$sres} {"$R"}->x,"$n";
```

```
$nx=$nx+ $coor{$snum} {$sres} {"N9"}->x;
```

```
$ny=$ny+ $coor{$snum} {$sres} {"N9"}->y;
```

```
$nz=$nz+ $coor{$snum} {$sres} {"N9"}->z;
```

```
$ox=$ox+ $coor{$snum} {$sres} {"O6"}->x;
```

```
$oy=$oy+ $coor{$snum} {$sres} {"O6"}->y;
```

```
$oz=$oz+ $coor{$snum} {$sres} {"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Перемещение объектов

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;
```

## Рекомендуемый порядок действий:

- **set retain\_order #** надо сохранить порядок атомов
- **create newobj, sele #** создаём новый объект, страховка
- **translate [0,10,0], newobj #** перемещаем
- **rotate x,90,newobj #** вращаем
- **save newfile.pdb, newobj**

## Операции по перемещению и вращению можно делать мышкой в режиме editing

```
my $x; my $y; my $z;
my $r;

foreach my $res ( @{$ $qartets{$q} }){
#
print "$q $coor{$m} {$res} {"N"}->x,"n";
$x=$x+ $coor{$m} {$res} {"N"}->x;
$y=$y+ $coor{$m} {$res} {"N"}->y;
$z=$z+ $coor{$m} {$res} {"N"}->z;

$ox=$ox+ $coor{$m} {$res} {"O6"}->x;
$oy=$oy+ $coor{$m} {$res} {"O6"}->y;
$oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```



```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Изменение координат отдельных атомов и

## объектов

```
my (%coor,%schnum)=read_pdb($ARGV[0]);
```

```
my %coor=read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
my $sch, my $schnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } ;
```

```
my %qwa=find_quart( %coor{"0"} ); my $sqnum=keys %qwa;
```

```
if ($sqnum > 0){
```

```
  #system("mkdir $ARGV[1]");
```

```
  my $filename=$ARGV[0];
```

```
  $filename="-- s/^.*\//";
```

```
  $filename="-- s/\.pdb//";
```

```
  # $filename=$schnum."-$sqnum.".$filename.".dat";
```

```
  $filename="-- s/dir/$filename/";
```

```
  open STDOUT, ">$filename";
```

```
  print OUT "INFO: chain $schnum, $sqnum, $r\n";
```

```
  foreach my $m ( sort { $a<=>$b } keys %coor ){
```

```
    my %qartets = %qwa; #find_quart( %coor{$m} );
```

```
    my %q = find_q( %coor{$m} );
```

```
  #   foreach my $q ( keys %qartets ){ print join " ", @{$qartets{$q}}, "\n";
```

```
    foreach my $q ( keys %qartets ){
```

```
      my $nx; my $ny; my $nz;
```

```
      my $ox; my $oy; my $oz;
```

```
      my $r;
```

```
      foreach my $res ( @{$qartets{$q}} ){
```

```
        #   print "$q %coor{$m} {$res} {"N7"}->x, "\n";
```

```
        $nx=$nx+ %coor{$m} {$res} {"N7"}->x;
```

```
        $ny=$ny+ %coor{$m} {$res} {"N7"}->y;
```

```
        $nz=$nz+ %coor{$m} {$res} {"N7"}->z;
```

```
        $ox=$ox+ %coor{$m} {$res} {"O6"}->x;
```

```
        $oy=$oy+ %coor{$m} {$res} {"O6"}->y;
```

```
        $oz=$oz+ %coor{$m} {$res} {"O6"}->z;
```

alter\_state 1,(pdb1cse),x=x-10.0

Или translate [0,10,0], A/100/NZ

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

## Удаление связей, но не атомов

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;
```

```
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
```

```

#system("rm -f $ch$chnum$dir");
my $filename=$ch$chnum$dir;
my $filename=$ch$chnum$dir;
my $filename=$ch$chnum$dir;
my $filename=$ch$chnum$dir;
my $filename=$ch$chnum$dir;

```

- Выберите первый атом, ctrl+middle click, выберите второй атом, ctrl+middle click
- И unbond или ctrl+D

**Внимание! Координаты атомов не меняются, только исчезает изображение связи**

```
foreach my $q ( keys %qartets ){ print join " ",@{$qartets{$q}},"n";
```

```
foreach my $q ( keys %qartets ){
```

```

my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
# print "$q $coor{$m}{$res}{"N"}->x,"n";
```

```
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
```

```
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
```

```
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
```

```
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
```

```
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Мутация аминокислот

```
#!/(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } };
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

- Запустите wizard->mutagenesis
- Выберите аминокислоту для мутации
- Справа выберите, на что мутировать
- Выберите ротамет с помощью управления movie
- Закончите процедуру с Apply

```
if ($qnum > 0){
#system("cat $filename | perl $filename.pl $ch $chnum $dir");
$filename=$ch.$chnum;
# $filename=$ch.$chnum.$dir;
$filename="$dir/$filename.dat";
print "File $filename\n";
open OUT, ">$filename";
print OUT "#INFO chain $chnum qnum $qnum\n";
foreach my $m ( keys %coor{"0"} ){
my %qartets = %qwa; #find_quart( %coor{$m} );
my %q = find_of( %coor{$m} );
# foreach my $q ( keys %qartets ){ print join( "\t", $qartets{$q}, $m, "\n" ); }
foreach my $q ( keys %qartets ){
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
foreach my $res ( @{$qartets{$q}} ){
# print "$q $coor{$m}{$res} {"$r"}->x,\"n\";
$nx=$nx+ $coor{$m}{$res} {"$r"}->x;
$ny=$ny+ $coor{$m}{$res} {"$r"}->y;
$nz=$nz+ $coor{$m}{$res} {"$r"}->z;
$ox=$ox+ $coor{$m}{$res} {"O6"}->x;
$oy=$oy+ $coor{$m}{$res} {"O6"}->y;
$oz=$oz+ $coor{$m}{$res} {"O6"}->z;
```

```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Добавление протонов

```
my (%my%coor,%my %schnum)=read_pdb($ARGV[0]);
```

```
my %coor=read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
my $ch, my $chnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;
```

```
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
my $max=0;
```

Работает с молекулами, т.е. объектами

[create gln, A/101/](#)

[h\\_add gln](#)

Или через меню action объекта.

Есть вероятность, что протоны будут добавлены неверно, если PyMol неправильно угадал валентность тяжёлых атомов.

```
# foreach my $q ( keys %qartets ){ print join( "@", $qartets{$q} ), "\n";
```

```
foreach my $q ( keys %qartets ){
```

```
my $nx; my $ny; my $nz;
```

```
my $ox; my $oy; my $oz;
```

```
my $r;
```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
# print "$q %coor{$m}{ $res } {"N"}->x,"n";
```

```
$nx=$nx+ %coor{$m}{ $res } {"N9"}->x;
```

```
$ny=$ny+ %coor{$m}{ $res } {"N9"}->y;
```

```
$nz=$nz+ %coor{$m}{ $res } {"N9"}->z;
```

```
$ox=$ox+ %coor{$m}{ $res } {"O6"}->x;
```

```
$oy=$oy+ %coor{$m}{ $res } {"O6"}->y;
```

```
$oz=$oz+ %coor{$m}{ $res } {"O6"}->z;
```

```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Суперапозиция в пространстве

```
my (%my%coor,%my%schnum)=read_pdb($ARGV[0]);
```

```
my %coor=read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
my $ch, my $chnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;
```

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

**Задача достаточно нетривиальная, и есть разные пути:**

**Белки:**

**align, super, fit**

**Другое:**

**pair\_fit**

**Желательно указывать родственные атомы в молекулах**

**pair\_fit ( trna10 and resid 10:15 and name P ), ( ref4 and resid 10:15 and name P )**

```
my $nx; my $ny; my $nz;
```

```
my $ox; my $oy; my $oz;
```

```
my $r;
```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
# print "$q $coor{$m}{$res} {"N"}->x,"n";
```

```
$nx=$nx+ $coor{$m}{$res} {"N9"}->x;
```

```
$ny=$ny+ $coor{$m}{$res} {"N9"}->y;
```

```
$nz=$nz+ $coor{$m}{$res} {"N9"}->z;
```

```
$ox=$ox+ $coor{$m}{$res} {"O6"}->x;
```

```
$oy=$oy+ $coor{$m}{$res} {"O6"}->y;
```

```
$oz=$oz+ $coor{$m}{$res} {"O6"}->z;
```

# Добавление органических фрагментов или а.к.

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
use File::Find;

my ($coor,$sch,$nnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch,$nnum;
foreach my $f ( sort keys %{$coor{"0"}} ){ my $ggg=substr($f,0,1); if ( $ggg ne $sch ){ $nnum++; $sch=$ggg } }
```

- С помощью **ctrl+middle click** выделите шариком атом, к которому будет присоединяться фрагмент

```
if ($nnum > 0){
```

```
  #system("mkdir $ARGV[1]");
```

```
  my $filename;
```

```
  $filename=
```

```
  $filename="-- s/\.pdb//";
```

```
  # $filename="";
```

```
  print "$filename\n";
```

```
  open OUT,">$filename";
```

```
  print OUT "MOL0 chain $nnum qnum $nnum\n";
```

```
  foreach my $q (
```

```
    my %q=
```

```
    my %q= find_q( $coor{$m} );
```

```
  # foreach my $q ( keys %qartets ){
```

```
    my $nx,$ny,$nz;
```

```
    my $nx=$nx+ $coor{$m} {$sres} {"N9"}->x;
```

```
    my $ny=$ny+ $coor{$m} {$sres} {"N9"}->y;
```

```
    my $nz=$nz+ $coor{$m} {$sres} {"N9"}->z;
```

```
    my $ox=$ox+ $coor{$m} {$sres} {"O6"}->x;
```

```
    my $oy=$oy+ $coor{$m} {$sres} {"O6"}->y;
```

```
    my $oz=$oz+ $coor{$m} {$sres} {"O6"}->z;
```

```
  # print "$q $coor{$m} {$sres} {"N9"}->x,\n";
```

```
  $nx=$nx+ $coor{$m} {$sres} {"N9"}->x;
```

```
  $ny=$ny+ $coor{$m} {$sres} {"N9"}->y;
```

```
  $nz=$nz+ $coor{$m} {$sres} {"N9"}->z;
```

```
  $ox=$ox+ $coor{$m} {$sres} {"O6"}->x;
```

```
  $oy=$oy+ $coor{$m} {$sres} {"O6"}->y;
```

```
  $oz=$oz+ $coor{$m} {$sres} {"O6"}->z;
```

- В меню **Build** выберите нужный фрагмент

- С помощью **ctrl+left click** выберите торсионный угол

Или

- Создайте свою молекулу (ChemSketch)

- Сохраните как **pkl** в `<pyamol_path>/data/chempy/fragments`

- **editor.attach\_fragment('pk1','my\_fragment\_name',11,0)**

**11** - это номер атома в фрагменте для связи

# Sculpting, что ЭТО?

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );

#(my %$coor,my $schnum)=read_pdb($ARGV[0]);
my %$coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } };

my %$qwa=find_quart( %$coor{"0"} ); my %$qnum=keys %$qwa;

if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
my $name="--s/c $V/C";
my $filename=$schnum . $qnum . $filename . ".d";
open OUT,">$filename";
foreach my $m (sort { $a<=>$b } keys %$coor ){
my %$qartets= %$qwa; #find_quart( %$coor{$m} );
my %$q= find_q( %$coor{$m} );

# foreach my $q ( keys %$qartets ){ print join " ",@{$qartets{$q}},"\n";

foreach my $q ( keys %$qartets ){

my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;

foreach my $res ( @{$qartets{$q}} ){
print "$q $coor{$m}{$res} {"$res"}->x,\n";
$nx=$nx+ $coor{$m}{$res}{"N9"}->x;
$ny=$ny+ $coor{$m}{$res}{"N9"}->y;
$nz=$nz+ $coor{$m}{$res}{"N9"}->z;

$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
$oy=$oy+ $coor{$m}{$res}{"O6"}->y;
$oz=$oz+ $coor{$m}{$res}{"O6"}->z;
}
```

Это похоже на real-time оптимизатор геометрии, но это алгоритм, который старается сохранить значения длины связей, углов, торсионных углов при изменении координат.

```
#!usr/bin/perl
use Math::VectorReal qw( :all );
```

# Как запустить sculpting?

```
##(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %{$coor{"O"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){$chnum++; $ch=$ggg };
```

У вас достаточно мощный компьютер? Тогда:

- Переводим мышь в режим редактирования
- Выбираем "auto-sculpting" из меню Sculpting
- Выбираем Sculpting из меню Wizard
- Выбираем центральный атом для модификаций Ctrl-middle-click
- Тянем атом в любую сторону ctrl-left-click-and-drag

```
if ($chnum > 0){
#system("perl $ARGV[0].pl $ARGV[1]");
my $filename=$ARGV[0];
$filename=~s/"/"/;
$filename=~s/ /_/;
#system("perl $ARGV[0].pl $ARGV[1]");
$filename="$dir/$filename.dat";
print "$filename\n";
open OUT, ">$filename";
print OUT "NAME $chnum\n";
foreach my $r ( sort keys %{$coor{"O"}}){
my %q={};
my %q={};
my %q={};
my %q={};
# foreach my $q ( keys %{$qartets} ){ print join " ",@{$qartets{$q}},"n";
#
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
foreach my $res ( @{$qartets{$q}} ){
#
print "$q $coor{$m}{$res} {"$r"}->x,"n";
$nx=$nx+ $coor{$m}{$res} {"$r"}->x;
$ny=$ny+ $coor{$m}{$res} {"$r"}->y;
$nz=$nz+ $coor{$m}{$res} {"$r"}->z;
$ox=$ox+ $coor{$m}{$res} {"O6"}->x;
$oy=$oy+ $coor{$m}{$res} {"O6"}->y;
$oz=$oz+ $coor{$m}{$res} {"O6"}->z;
```



```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Скриптование в PyMol

```
#!/(my %coor,my $schnum)=read_pdb($ARGV[0]);
```

```
my %coor=read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
my $ch, my $chnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $schnum++; $ch=$ggg } };
```

```
my %qwa=find_quart( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum > 0){
```

```
my $file=$dir.$ARGV[0].".
```

```
$filename="-- s/~/V/C/
```

```
$filename=$dir.$ARGV[0].".
```

```
$filename=" $ch $chnum.dat";
```

```
open( OUT, ">$filename";
```

```
print OUT "#INFO chain $schnum group $qnum ln";
```

```
foreach my $m ( sort keys %{$coor{"0"}} ){
```

```
my $q=find_quart( %coor{"$m"} );
```

```
my $qnum=keys %qwa;
```

Возможны как скрипты из команд, так и скрипты на Python

Запуск скриптов из команд:

@ myfile.pml

Запуск скриптов на питоне:

run myfile.py

```
# foreach my $q ( keys %qartets ){ print join " ", @{$qartets{$q}} , "\n";
```

```
foreach my $q ( keys %qartets ){
```

```
my $nx; my $ny; my $nz;
```

```
my $ox; my $oy; my $oz;
```

```
my $r;
```

```
foreach my $res ( @{$qartets{$q}} ){
```

```
print "$q $coor{$m}{$res} {"$r"}->x, "\n";
```

```
$nx=$nx+ $coor{$m}{$res} {"$r"}->x;
```

```
$ny=$ny+ $coor{$m}{$res} {"$r"}->y;
```

```
$nz=$nz+ $coor{$m}{$res} {"$r"}->z;
```

```
$ox=$ox+ $coor{$m}{$res} {"06"}->x;
```

```
$oy=$oy+ $coor{$m}{$res} {"06"}->y;
```

```
$oz=$oz+ $coor{$m}{$res} {"06"}->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Пример

```
my (%coor,%schnum)=read_pdb($ARGV[0]);
```

```
my %coor=read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
foreach my $r (sort keys %coor{"N1"}){ if ($rggg=substr($r,0,1); if ($rggg ne $sch){ $schnum++; $sch=$rggg } );
```

```
my %qwa=find_quart($coor{"0"}); my $snum=keys %qwa;
```

```
zoom i. 4+5
```

```
mset 1 x1440
```

```
mview store
```

```
python
```

```
for x in range(0,144):
```

```
cmd.frame((10*x)+1)
```

```
cmd.zoom("n. CA and i. " + str(x) + "+" + str(x+1))
```

```
cmd.mview("store")
```

```
python end
```

```
frame 288
```

```
mview store
```

```
mview reinterpolate
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```

```
my $r; my $n; my $nz;
```



# Результат

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch, my $chnum;
foreach my $r ( sort keys %coor ) {
    my %qwa=find_quart( $coor{$r} );

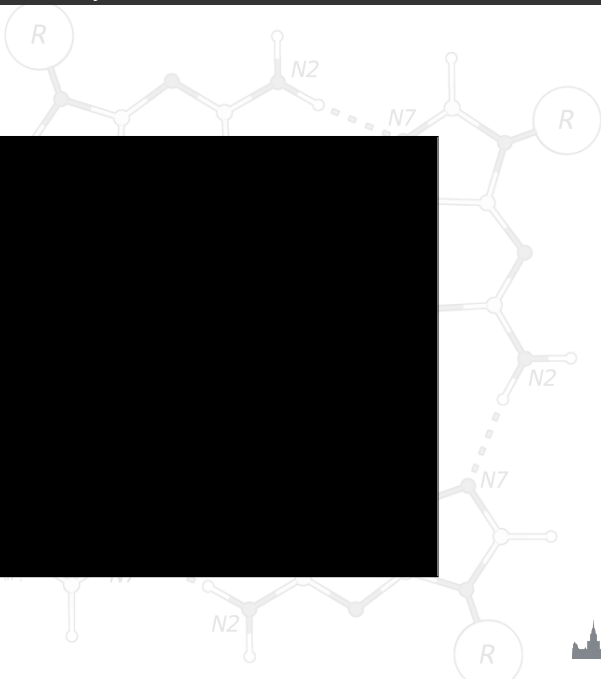
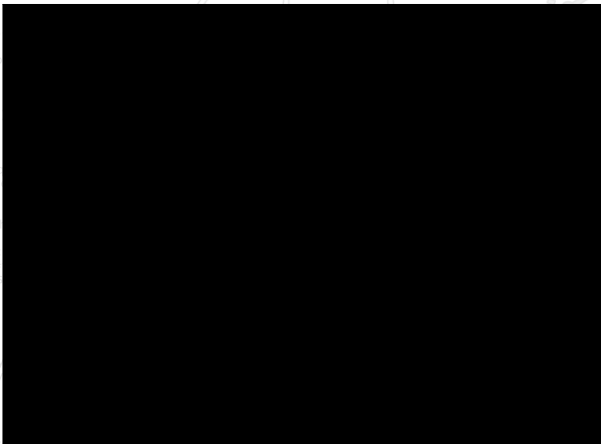
    if ($qnum >0){
        #system("mkdir $ARGV[1]");
        my $filename=$ARGV[0];
        $filename=~ s/~/~/;
        $filename=~ s/\././;
        # $filename=$chnum."_".$r;
        $filename="$dir/".$filename;
        print "$filename\n";
        open OUT,">$filename";
        print OUT "#INFO chain $r\n";

        foreach my $m (sort { $a<
            my %qartets = %qwa ; #
            my %q= find_q( $coor{$m} );

            #   foreach my $q { keys %qartets }
            foreach my $q { keys %qartets } {
                my $nx; my $ny; my $nz;
                my $ox; my $oy; my $oz;
                my $r;

                foreach my $res ( keys %coor ) {
                    #   print "$q $res\n";
                    $nx=$nx+ $coor{$m} {$res} {"N9"}->x;
                    $ny=$ny+ $coor{$m} {$res} {"N9"}->y;
                    $nz=$nz+ $coor{$m} {$res} {"N9"}->z;

                    $ox=$ox+ $coor{$m} {$res} {"O6"}->x;
                    $oy=$oy+ $coor{$m} {$res} {"O6"}->y;
                    $oz=$oz+ $coor{$m} {$res} {"O6"}->z;
                }
            }
        }
    }
}
```



```
#!/usr/bin/perl
```

```
use Math::VectorReal qw( :all );
```

# Объекты из PyMol можно использовать в разных 3D программах

```
my %coor=(); my $schnum= read_pdb($ARGV[0]);
```

```
my %coor= read_pdb($ARGV[0]);
```

```
my $dir=$ARGV[1];
```

```
my $sch, my $schnum;
```

```
foreach my $r ( sort keys %{$coor{"0"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } };
```

```
my %qwa=find_quat( %coor{"0"} ); my $qnum=keys %qwa;
```

```
if ($qnum >
```

```
#system("m
```

```
my $filenam
```

```
$filename=
```

```
$filename=
```

```
# $filename
```

```
$filename=
```

```
print "$filer
```

```
open OUT,">
```

```
print OUT,">
```

```
foreach my
```

```
my %qar
```

```
my %q=
```

```
# foreac
```

```
foreac
```

```
my $
```

```
my $
```

```
m
```

```
foreach my $res ( @{$schatoms{$sch}} ){
```

```
# print "$q $coor{$m}{$res}{ "N9" }->x,"n";
```

```
$nx=$nx+ $coor{$m}{$res}{ "N9" }->x;
```

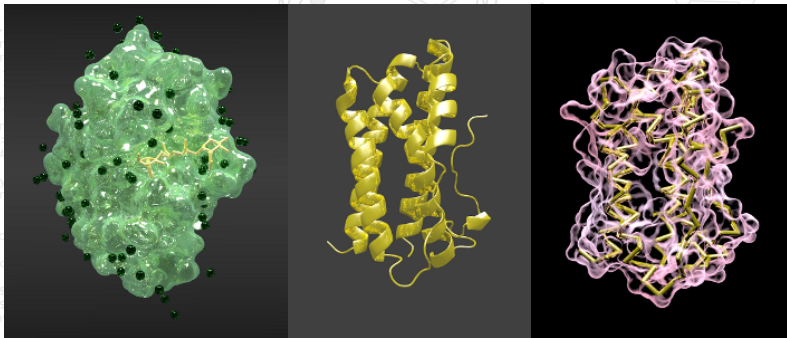
```
$ny=$ny+ $coor{$m}{$res}{ "N9" }->y;
```

```
$nz=$nz+ $coor{$m}{$res}{ "N9" }->z;
```

```
$ox=$ox+ $coor{$m}{$res}{ "O6" }->x;
```

```
$oy=$oy+ $coor{$m}{$res}{ "O6" }->y;
```

```
$oz=$oz+ $coor{$m}{$res}{ "O6" }->z;
```



```
# print "$q $coor{$m}{$res}{ "N9" }->x,"n";
```

```
$nx=$nx+ $coor{$m}{$res}{ "N9" }->x;
```

```
$ny=$ny+ $coor{$m}{$res}{ "N9" }->y;
```

```
$nz=$nz+ $coor{$m}{$res}{ "N9" }->z;
```

```
$ox=$ox+ $coor{$m}{$res}{ "O6" }->x;
```

```
$oy=$oy+ $coor{$m}{$res}{ "O6" }->y;
```

```
$oz=$oz+ $coor{$m}{$res}{ "O6" }->z;
```

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
```

# Анимации изображения наложенная на ролик из youtube

```
my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch, my $chnum;
foreach my $r ( sort keys %{$coor{"O"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch){ $chnum++; $sch=$ggg } ;
```

```
my %qwa=find
```

```
if ($qnum >0){
#system("mkdir
my $filename=
$filename="-- s
$filename="-- s
# $filename=$
$filename="$dir
print "$filename
open OUT,">$f
print OUT "#IN
```

```
foreach my $r
my %qartet
my %q= find
```

```
# foreach r
```

```
foreach r
```

```
my $nz
my $soy
my $
```

```
foreach
```

```
#
```

```
$nx=$nx+ $coor{$m}{ $res }{"N9"}->x;
$ny=$ny+ $coor{$m}{ $res }{"N9"}->y;
$nz=$nz+ $coor{$m}{ $res }{"N9"}->z;

$ox=$ox+ $coor{$m}{ $res }{"O6"}->x;
$oy=$oy+ $coor{$m}{ $res }{"O6"}->y;
$oz=$oz+ $coor{$m}{ $res }{"O6"}->z;
```