

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

```
#!/(my %coor,my $snum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $ch,$s,$snum;
foreach my $atom (keys %coor){
```

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

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

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

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

```
}
```

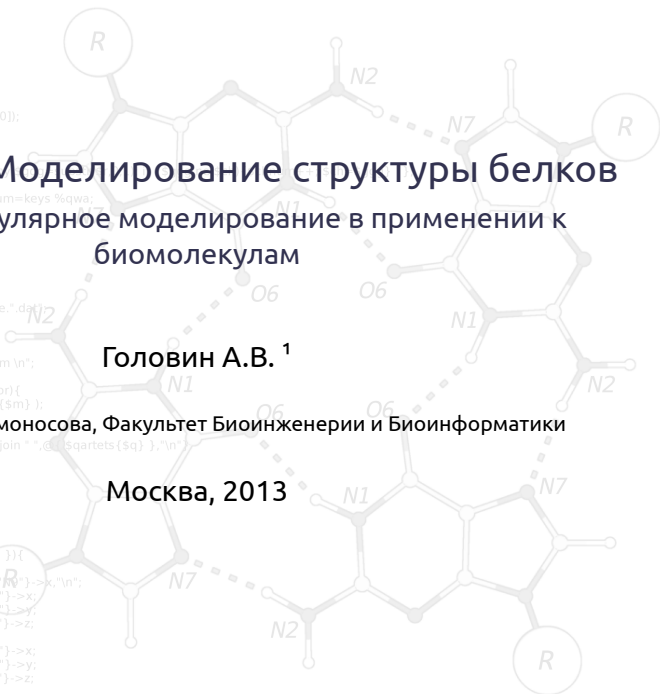
Лекция 11. Моделирование структуры белков

Курс: Молекулярное моделирование в применении к биомолекулам

Головин А.В. ¹

¹МГУ им М.В. Ломоносова, Факультет Биотехнологии и Биоинформатики

Москва, 2013



Содержание

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
#(my %$coor,$my $schnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch,$my $schnum;
```

Введение

```
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;
```

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

Сравнительное моделирование

Моделирование *Ab initio*

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

Мета серверы

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

Заключение

```
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;
```

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

Основные проблемы:

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

- Монте-Карло: 100 а.к. 3N степеней свободы, получаем 10^{48} конформаций.

- **Парадокс Левинтала:** "Промежуток времени, за который полипептид приходит к своему скрученному состоянию, на много порядков меньше, чем если бы полипептид просто перебирал все возможные конфигурации".

- Для решения разумно использовать накопленные знания для моделирования.

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

Последовательность-структура

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

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

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

- Теоретические модели, не соответствуют тому, что природа старается оптимизировать;

- В ходе эволюции были отобраны только те белки, которые легко сворачиваются;

- белки могут сворачиваться разными путями, не обязательно следуя глобально оптимальному пути.

- Считается, что структура определяется последовательностью, но иногда нужны другие факторы.

- Структура более консервативна чем последовательность

```
$nx=$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);

Сравнительное моделирование

```
#!/(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;
```

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

```
  #system("rm -f $qnum");
```

```
  my $filename="";
```

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

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

```
  # $filename="-- s/~/~/";
```

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

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

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

```
  print OUT "Chain $schnum $qnum\n";
```

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

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

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

```
  }
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

- Зачем искать конформации если можно представить, что при подобии последовательностей подобны и структуры.

- Надо оценить насколько вероятно, что отличие в последовательности может привести изменению способа укладки цепи.

- Надо отфильтровать ошибки полученные при определении структуры.

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

Известные структуры и последовательности

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

- Сейчас известно порядка 10^5 структур. Примерно 10% это уникальные белки.
- Только 30% из первого пункта имеют разрешение лучше 3.0 ангстрем.
- Примерно 25% известных последовательностей можно использовать для сравнительного моделирования.
- Для 50% последовательностей можно предсказать способ укладки.

#!/usr/bin/perl

use Math::VectorReal qw(:all);

Степень идентичности и сравнительное

моделирование

```

#(my %scoor=my $schnum = read_pdb($ARGV[0]);
my %scoor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys %{$scoor{"0"}} ){ my $i
my %qwa=find_quart( $scoor{"0"} ); my $qnum=

```

```

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

```

```

Sali,A. & Kuriyan,J.

```

Trends Biochem. Sci. 22,
M20-M24 (1999)

```

# 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} {"N9"
$nx=$nx+ $coor{$m} {$res} {"N9"}->
$ny=$ny+ $coor{$m} {$res} {"N9"}->
$nz=$nz+ $coor{$m} {$res} {"N9"}->

```

```

$ox=$ox+ $coor{$m} {$res} {"O6"}->
$oy=$oy+ $coor{$m} {$res} {"O6"}->
$oz=$oz+ $coor{$m} {$res} {"O6"}->

```

99 %

- Поиск ингибитора



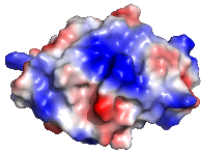
60 %

- Молекулярное замещение в PCA
- Планирование мутагениза



30 %

- Поиск сайтов связывания
- Определение упаковки



0 %

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

Как это реализовать?

```
#!/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;
```

- Надо найти белок заготовку с известной структурой.

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

```
  #system("cp $qwa/$qnum.pdb $dir");
```

```
  my $filename=$qwa{"0"};
```

```
  $filename="-- s/" . $qnum . "/";
```

```
  $filename=$qwa{"0"};
```

```
  # $filename="$dir/$qnum.pdb";
```

```
  $filename="$dir/$qnum.dat";
```

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

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

```
  print OUT "ATOM $chnum $qnum $qnum $qnum $qnum $qnum\n";
```

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

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

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

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

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

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

```
          foreach my $q ( keys %$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"} {"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;
```

```
            }
```

```
          }
```

```
        }
```

```
      }
```

```
    }
```

```
  }
```

- Построить первичное выравнивание.

- Улучшить выравнивание.

- Построить ход основной цепи.

- Моделирование петель

- Построить/моделировать положение боковых радикалов

- Проверка модели


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

Поиск белка заготовки

```
my ($my $coor, my $chnum)=read_pdb($ARGV[0]);  
my $my $coor=read_pdb($ARGV[0]);  
my $dir=$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 %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```

- Поиск по PDB с помощью:

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

- Blast

- Psi-Blast

- Методов распознавания упаковки

- Используя биологическую информацию.

- Функциональное аннотирование в базах данных.

- Используя информацию об активных сайтах, или мотивах.

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

Улучшение выравнивания

#(my \$coor = read_schnum(\$nb(\$SARGV[0]));

my \$coor = read_pdb(\$SARGV[0]);

my \$seq = \$SARGV[1];

PHE ASP ILE CYS ARG LEU PRO GLY SER ALA GLU ALA VAL CYS

foreach my \$r (sort keys %{\$coor{"0"}}){ my \$sggg = substr(\$r,0,1); if (\$sggg ne sch){ \$sch = \$sggg; }

PHE ASN VAL CYS ARG THR PRO --- --- --- GLU ALA ILE CYS

PHE ASN VAL CYS ARG --- --- --- THR PRO GLU ALA ILE CYS

#system("mkdir \$SARGV[1]");

my \$filename = \$SARGV[0];

\$filename =~ s/\^.*//;

\$filename =~ s/\.pdb//;

#\$filename = \$schnum.".".\$qnum.".".\$fi

\$filename = "\$dir".\$filename.".dat";

print "\$filename\n";

open OUT,">\$filename";

print OUT "#INFO chain \$schnum qnum

foreach my \$m (sort {\$a-<=>\$b} key

my %qartets = %qwa; #find_quart

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 (@{\$qarte

print "\$q \$coor{\$m} {\$

\$nx = \$nx + \$coor{\$m} {\$res

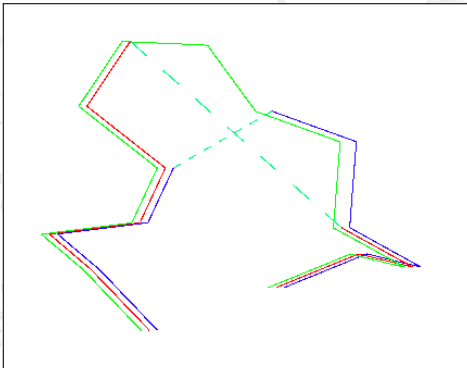
\$ny = \$ny + \$coor{\$m} {\$res

\$nz = \$nz + \$coor{\$m} {\$res

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

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

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



Из книги "Professional Gambling" от Gert Vriend

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

Качество белка заготовки

```
#{my %codon; my $schnum}=read_pdb($ARGV[0]);
```

- Выбор качественного белка заготовки очень важен.

- Лучший вариант не обязательно обладает лучшей степенью идентичности.

```
my %qwa=...um=keys %qwa;
```

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

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

```
  my $filename=$snum;


```
 $filename=~ s/ /_/;


```
  $filename=~ s/\./_/;


```
 $filename=~ s/\./_/;

```



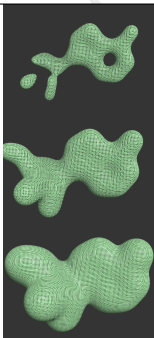
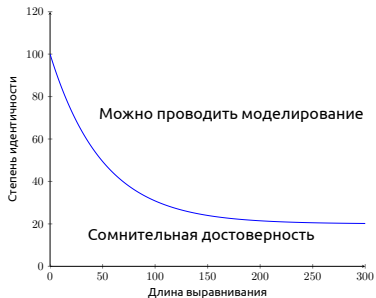
- Белок 1: ID 93%, 3.5 ангстрема разрешение. Хуже.

```


```


```

- Белок 2: ID 90%, 1.5 ангстрема разрешение. Лучше!



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

Если структура белка заготовки получена ЯМР

- Определимся какие области определены лучше.
- Соотнесём с выравниванием.
- Если низкая гомология выпадает на “подвижные” области, то структура подходит.

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

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

my %q;

if ($qnum >
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/^\.*/\./;
$filename=~ s/\./_/;
#$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= %qwa ; #find quartl { $coor { $m } };
my %q= find_q { $coor { $m } };

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

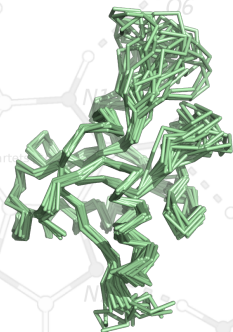
foreach my $q ( keys %qartets ){

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;

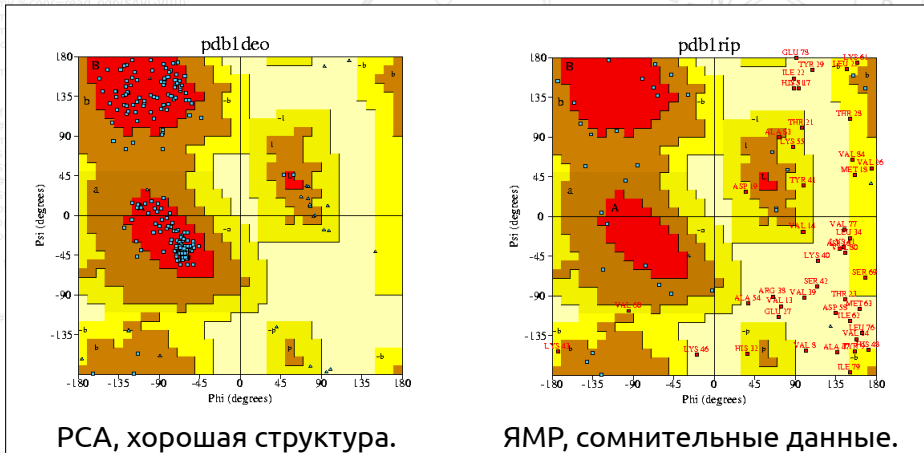
$ox=$ox+ $coor { $m } { $res } { "O6" }->x;
$oy=$oy+ $coor { $m } { $res } { "O6" }->y;
$oz=$oz+ $coor { $m } { $res } { "O6" }->z;
}
```



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

Качество заготовки, Рамачандран

```
#!/my %$coor,$my $snum)=read_pdb($ARGV[0]);
my ($R,$N2,$N7,$R);
```



РСА, хорошая структура.

ЯМР, сомнительные данные.

```
$nx=$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;
```

Построение остова

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

my $R = 10;

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

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

if ($snum > 0){
  #system("mkdir $ARGV[1]");
  my $filename=$dir."/chain/$snum";
  $filename=$filename.".pdb";
  # $filename=$filename.".txt";
  # $filename=$filename.".log";
  print "$filename\n";
  open OUT, ">$filename";
  print OUT "SMPO chain $snum $snum $snum\n";

  foreach my $m ( keys %$coor ){
    my %quarts = %qwa; #find_quart($coor{$m});
    my $q=find_of($coor{$m});
    # foreach my $k ( keys %quarts ){ print join " ", $quarts{$q}, "\n"; }
    foreach my $res ( @{$quarts{$q}} ){
      print "$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;
    }
  }
}
```

- Генерируем координаты остова моделируемого белка для остатков из выравненных областей.
- Не обязательно использовать координаты, могут подойти дистанционные ограничения и любые другие подходящие ограничения.
- Большинство исследователей предпочитают Modeller. Modeller использует дистанционные ограничения.

#!/usr/bin/perl

use Math::VectorReal qw(:all);

Моделирование петель

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %scoor=;
my $d=;
my $sch, my $schnum;
foreach my $r ( (keys %scoor{"O"})) {
my %qwa=find_quart( %scoor{"O"} );
my $qnum=key %qwa;
if ($qnum > 0) {
#system("cp $filename.$d.$schnum.$qnum.ne.*.dat";
print "file: $filename.$d.$schnum.$qnum.ne.*.dat";
open OUT ">";
print OUT "#INFO: chain $chnum group $qnum ln";
foreach my $m (sort { $a->{sb} k($a->{coor}) } %qartets) {
my %qartets= %qwa;
my %q= find_q($scoor{"O"}, %qartets);
# foreach my $q {
foreach my $q {
my $nx; my $ny; my $nz;
my $ox; my $oy; my $oz;
my $r;
foreach my $res ( @{$qartets{$q}} ) {
$ny=$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;
}
}
}
}

```

- Эмпирическое моделирование:

- Поиск подходящего фрагмента по PDB
- Использовать базы данных (LIP, etc.)

- Молекулярная механика.

- Монте-Карло.

- Rosseta:

- Поиск фрагментов близких по последовательности и предполагаемой вторичной структуре.
- Комбинирование результатов поиска с помощью Монте-Карло.

- Комбинации выше перечисленных.

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

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

Моделирование боковых радикалов

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

```
my $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;
```

- Если идентичность последовательностей высока то можно ожидать высокую консервативность третичных контактов.

- Если анализ показывает, что важные контакты консервативны то:

Лучше оставить конформацию боковых радикалов из заготовки чем моделировать.

```
foreach my $m (sort { $a<=>$b } keys %coor){
  my %qartets = %qwa; #find quart( $coor{$m} );
  #foreach my $q (keys %qartets){ print join " ", $m, $q, "\n";
```

```
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);

Моделирование боковых радикалов

```
#!/(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{"O"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg };
```

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

- Конформация боковых радикалов зависит от конформации основной цепи.

- Существуют базы данных ротамеров.

- Некоторые исследователи считают, что SCWRL метод самый удачный.

foreach my \$q { keys %qwa } Это эмпирический метод на основе теории графов.

foreach my \$q { keys %qwa } <http://dunbrack.fccc.edu/SCWRL3.php>

```
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 );
use IO::File;
use Data::Dumper;
```

Точность моделирования боковых радикалов

```

#(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;
```

- **Высокая точность моделирования достигается для боковых радикалов внутри глобулы.**

- **Причина: в экспериментах остатки на поверхности более подвижны.**

- **Вычислительное проще упаковать гидрофобные остатки, чем учесть полярные контакты и водородные связи с водой или с участием воды.**

```

foreach my $m ( sort keys %{$coor{"0"}} ){
  my %qartets= %qwa; #find quart( %coor{$m} );
  my %q= find_q( %coor{$m}, %qartets );
  # foreach my $q ( keys %qartets ){
  #   print "$q $coor{$m} {"$res"} {"$res"}->x,"n";
  #   $nx=$nx+ $coor{$m} {"$res"} {"$res"}->x;
  #   $ny=$ny+ $coor{$m} {"$res"} {"$res"}->y;
  #   $nz=$nz+ $coor{$m} {"$res"} {"$res"}->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 );
```

Улучшение модели

```
#!/(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){
```

- Методы минимизации энергии.

- Моделирование молекулярной динамики (оптимизация гидрофобики)

- Моделирование Монте-Карло.

- Любой известный подход для оптимизации структуры.

```
# 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 );
```

Ошибки

```
#!/(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;
```

- Обычно ошибки не исправляются на последующих этапах моделирования.

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

```
  #system("rm -f $filename");
  my $filename=$ARGV[0];
```

```
  $filename="-- s/($pub
```

```
  $filename="-- s/($pub
```

```
  # $filename=$schnum
```

```
  $filename="$dir/$. $
```

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

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

```
  print OUT "#INFO chain $schnum $qnum $schnum\n";
```

```
  foreach my $m ( sort
```

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

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

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

```
  #
```

- Хорошее выравнивание не исправит плохой выбор белка заготовки.

- Хорошее моделирование петель не исправит плохое выравнивание.

- При обнаружении ошибки необходимо повторять некоторые этапы.

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

```
    #
```

```
    #
```

```
    #
```

```
    #
```

```
    #
```

```
    #
```

```
    #
```

```
    #
```

```
    #
```

Проверка

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

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my $coor=load_pdb($ARGV[0]);
my $d=$d($coor);
my $ch, my $chnum;
foreach my $q ($coor{"O6"}){ my $qnum=$ch{$q}{"O6"};
my %qwa=find_quart($coor{"O6"}); my $qnum=keys %qwa;
```

- Большинство программ для моделирования по гомологии выдают правильные значения для связей и валентных углов.

```
if ($qnum > 0){
#system("cat $qnum.pdb > $qnum.pdb");
my $filename=$qnum.pdb;
$filename=~s/\.pdb//;
$filename=$qnum.pdb;
#system("cat $qnum.pdb > $qnum.pdb");
my $filename=$qnum.pdb;
print "$filename\n";
open OUT, ">$filename.dat";
print OUT "#INFO chain $chnum qnum $qnum\n";
```

- Карта Рамачандрана в большинстве случаев для модели выглядит также, как для белка заготовки
- Проверка на ориентацию или положение заряженных остатков может быть полезна.

- Использование любых экспериментальных данных:

- Остатки активного центра.
- Места модификаций.
- Места контактов.

РгоQ сервер оптимизирован на поиск правильной модели а не нативной структуры.

```
foreach my $res (@{ $qartets{$q} ){
my $nx=$x+$coor{$m}{$res}{"N9"}->x;
my $ny=$y+$coor{$m}{$res}{"N9"}->y;
my $nz=$z+$coor{$m}{$res}{"N9"}->z;
my $ox=$x+$coor{$m}{$res}{"O6"}->x;
my $oy=$y+$coor{$m}{$res}{"O6"}->y;
my $oz=$z+$coor{$m}{$res}{"O6"}->z;
```

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

Ресурсы для гомологичного моделирования

```
#!/(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;
```

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

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

- Modeller

- SwissModel

- Eva-CM

- Nest И т.д.



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

Предсказание структуры белка *Ab initio*

```
#!/(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";
```

```
  $filename="$ch.$qnum";
```

```
  $filename="$ch.$qnum";
```

```
  print "file: $filename";
```

```
  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;
```

- Теоретически можно использовать молекулярную динамику.
- Моделирование отжига, как в МД так и в Монте-Карло.
- На основе фрагментов, Rosseta

Ab initio, Rosseta

- Метод использует информацию о предсказании вторичной структуры
- Сравниваем фрагменты от 3 до 9 остатков с библиотекой известных структур. Строим эти фрагменты.
- Соединяем эти фрагменты и используем Монте-Карло для оптимизации третичной структуры.



Ab initio, Rosseta

```
#!/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 $i ( sort keys %{$coor{"O"}} ) { my $sggg=substr($i,0,1); if ( $sggg ne $sch ) { $schnum++; $sch=$sggg } ;
my %$qwa
```

- Для определения хорошей конформации использую специальные потенциалы, которые делают модель похожей на нативную

- Что можно использовать:

- Потенциалы для третичных контактов
- Гидрофобные потенциалы
- Потенциал для уменьшения радиуса вращения молекулы
- Водородные связи и т.д.

Можно добавить знание об дисульфидных мостиках, местах связывания катионов металлов и т.д.

```
# foreach my $q ( keys %$qartets ) {
my $id=$q; my $coor;
my $sox,$my $soy,$my $soz;
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;
```

Threading — протягивание нити

```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
my $PDB = $ARGV[0];
my $DIR = $ARGV[1];
my $SCH, my $SCHNUM;
foreach my $R ( sort keys %{$COOR{"0"}} ) { my $GGGG=substr($R,0,1); if ( $GGGG ne $SCH ) { $SCHNUM++; $SCH=$GGGG };
```

- Сравниваем последовательность со всеми известными способами укладки.

- Используем потенциалы для определения тенденций в известных способах укладки.

- Каждую аминокислоту из модели помещаем в позиции белков разных укладок

- Определяем как хорошо эта аминокислота подходит белку заготовке на основе парных взаимодействий

- Но основе суммарного результата определяем белок заготовку.

```
foreach my $RES ( @{$SQRTETS{$SQ}} ) {
#
print "$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 );
use IO::File;
my $f = IO::File->new($ARGV[0], "r");
my $q = $f->readline();
```

Threading — недостатки

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

- Взаимодействия в белке не всегда описываются парными контактами.
- Потенциалы часто основываются на профилях последовательностей.

```
foreach my $m (sort { $a-<=>$b } keys %coor){
  my $my $qartets = %qwa; #find_quart( $coor{$m} );
  my $my $q = find_q( $coor{$m} );
  #
  foreach my $q ( keys %qartets ){ print join " ", @{$qartets{$q}} , "\n";
```

Есть гибридные методы Rosseta/Threading: I-Tasser

```
my $my $nx; my $my $ny; my $my $nz;
my $my $ox; my $my $oy; my $my $oz;
my $my $r;
foreach my $res ( @{$qartets{$q}} ){
  #
  print "$q $coor{$m} {$res} {"N1"}->x, "\n";
  $my $nx=$nx+ $coor{$m} {$res} {"N9"}->x;
  $my $ny=$ny+ $coor{$m} {$res} {"N9"}->y;
  $my $nz=$nz+ $coor{$m} {$res} {"N9"}->z;
  $my $ox=$ox+ $coor{$m} {$res} {"O6"}->x;
  $my $oy=$oy+ $coor{$m} {$res} {"O6"}->y;
  $my $oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```

Мета серверы

```
#!/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{"O"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){ $chnum++; $ch=$ggg } };
my %($$$$);

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

foreach my $m ( keys %$coor ){
my $sq=$coor{$m};
my %sq= %of { $coor{$m} };
# foreach my $i ( keys %$sq ){ print out "#INFO $i $sq\n"; }

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

foreach my $res ( @{ $sqartets{$sq} } ){
# print "$sq $coor{$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;
}
}
```

- Сравнение разных методов.

- Большинство методов предсказывают правильную укладку в первых 10-20 результатах.

- Удаление структур с высоким значением параметров модели, но с единственной укладкой.

- Суперпозиция результатов, взвешивание.

- Часто выдают только позиции атомов остова.

Заключение

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

my ($R) = @ARGV;

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;
```

- Суть современного моделирования белков - эмпирическая
- Чем больше известной информации используется при моделировании тем точнее модель.

- Каждый метод имеет недостатки.
- Критический анализ модели позволяет выявить ошибки и улучшить модель.

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

    # foreach my $q ( keys %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;
```