

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

```
##(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 $sqn=substr($r,0,1); if ( $sqn ne $ch) { $chnum++; $ch=$sqn; } );
my %qwa=find_quart( $coor{"O"});
```

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

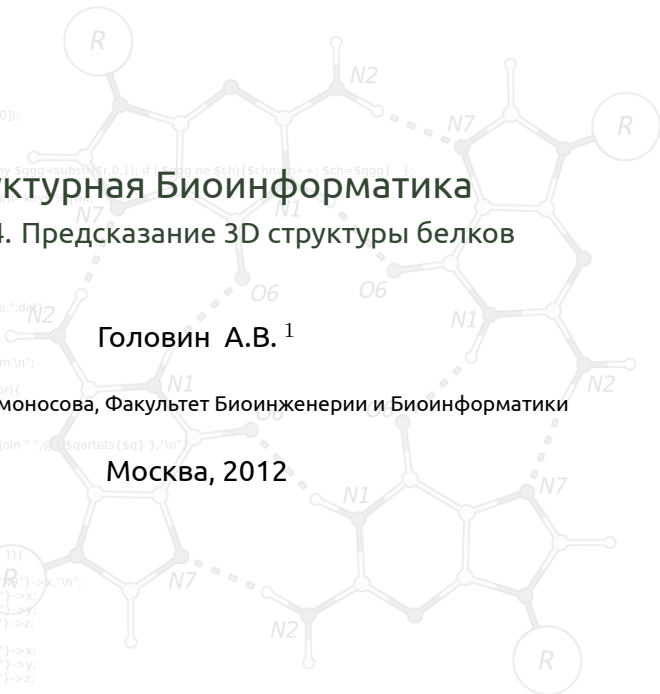
Структурная Биоинформатика

Лекция 4. Предсказание 3D структуры белков

Головин А.В.¹

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

Москва, 2012



Содержание

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

Введение

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

Сравнительное моделирование

```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
if ( $qnum > 0 ){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename=~ s/\/\//;
$filename=schrnd( " " . $qnum / $filename . " " );
$filename="dir".$filename.".dat";
print "$filename\n";
open OUT.">$filename";
```

Моделирование *Ab initio*

Threading — протягивание нити

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

Распознавание укладки

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

Мета серверы

```
my $r;
```

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

Заключение

```
my $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 );
```

Основные проблемы:

```

#(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 } ;

```

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

- **Парадокс Левинталя:** "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 %q;
my %q;
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;

```

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

Последовательность-структура

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

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

Причина парадокса Левинталя:

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

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

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

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

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

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

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

```
  # $filename="$chain"."$qnum"."$filename"."$date";
```

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

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

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

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

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

```
    my %qartets = %qwa; #find quartet $coor{$sm};
```

```
    my %q = %qartets{$q};
```

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

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

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

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

```
        $nx=$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;
```

```
  }
}
```

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

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

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

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

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

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

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

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

- 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

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

Сравнительное моделирование

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

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

```
if ($qnum > 0){
#system("cat $dir/$sch/$chnum/$qnum.pdb > $dir/$sch/$chnum/$qnum.pdb");
my $filename="$dir/$sch/$chnum/$qnum.pdb";
$filename=~ s/^\./\./;
$filename=~ s/\./_/;
#system("cat $dir/$sch/$chnum/$qnum.pdb > $dir/$sch/$chnum/$qnum.pdb");
my $filename="$dir/$sch/$chnum/$qnum.pdb";
print "$filename\n";
open OUT ">OUT";
print OUT "NAME Chain SchNum QNum\n";
```

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

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

- Надо отфильтровать ошибки полученные при определении структуры.

```
#
foreach my $s ( keys %quartets ){
my $sx, my $sy, my $sz;
my $ox, my $oy, my $oz;
my $r;

foreach my $res ( @{$quartets{$s}} ){
print "$s $coor{$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 );
```

Известные структуры и последовательности

```
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 } ;
```

- 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.

```
my %qartets = %qwa; # find quart { $coor{$m} };
my %q;
foreach my $q { keys %qartets } { print join " ", @qartets{$q} , "\n";
    my $nx, my $ny, my $nz;
    my $ox, my $oy, my $oz;
    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 );
```

Степень идентичности и сравнительное моделирование

```
my %scoor=read_pdb($ARGV[0]);
my $sdir=$ARGV[1];
my $sch, my $schnum;
foreach my $f ( sort keys %{$scoor{"0"}} ){ my $s
my %qwa=find_quart( $scoor{"0"} ); my $sqnum=
```

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

```
my $q1=$schnum.".".$sqnum.".".$filename."."
my %q1=find_quart($scoor{"1"});
# foreach my $q { keys %qartets } { print join
#
foreach my $q { keys %qartets } {
```

Sali, A. & Kuriyan, J.

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

```
my $nx=$q1{"N1"};
my $ny=$q1{"N2"};
my $nz=$q1{"N3"};
my $sx=$q1{"S1"};
my $sy=$q1{"S2"};
my $sz=$q1{"S3"};
```

```
foreach my $res ( @{$qartets{$sq}} ){
#
print "$q $scoor{$sm} {$res} {"$N"}
$nx=$nx+ $scoor{$sm} {$res} {"N1"}->
$ny=$ny+ $scoor{$sm} {$res} {"N2"}->
$nz=$nz+ $scoor{$sm} {$res} {"N3"}->
```

```
$sx=$sx+ $scoor{$sm} {$res} {"O6"}->
$sy=$sy+ $scoor{$sm} {$res} {"O6"}->
$sz=$sz+ $scoor{$sm} {$res} {"O6"}->
```

```
my $q2=$schnum.".".$sqnum.".".$filename."."
my %q2=find_quart($scoor{"2"});
# foreach my $q { keys %qartets } { print join
#
foreach my $q { keys %qartets } {
```

```
my $nx=$q2{"N1"};
my $ny=$q2{"N2"};
my $nz=$q2{"N3"};
my $sx=$q2{"S1"};
my $sy=$q2{"S2"};
my $sz=$q2{"S3"};
```

99 %

- Поиск ингибитора



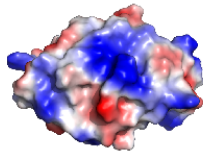
60 %

- 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 $schnum)=read_pdb($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;
```

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

```
  #system("ls -l $ARGV[0].*");
```

```
  my $filename=$ARGV[0].*";
```

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

```
  $filename="-- $d.pdb//";
```

```
  #file=$dir.$filename; dat";
```

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

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

```
  print OUT "MolPro chain $schnum $qnum $sch\n";
```

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

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

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

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

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

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

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

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

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

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

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

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

```
        #
```

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

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

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

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

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

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

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

- Надо найти белок заготовку с известной структурой.
- Построить первичное выравнивание.
- Улучшить выравнивание.
- Построить ход основной цепи.
- Моделирование петель
- Достроить/моделировать положение боковых радикалов
- Проверка модели


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

Поиск белка заготовки

```

#(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=sort { $coor{"0"}{$chnum}=keys %$qwa;
```

- Поиск по PDB с помощью:

```

if ($qnum > 0){
  #system("mkdir -p $dir/$qnum");
  my $filename=$ARGV[0];
  $filename="-- s/";
  $filename="-- s/\,pdb/";
  # $filename=$ch;
  $filename="$dir/$qnum";
  print "$filename\n";
  open OUT,">$filename";
  print OUT "#INFO chain $chnum qnum $qnum\n";

```

- Blast
- Psi-Blast
- Методов распознавания упаковки

```

foreach my $q ( keys %$qartets){
  my %$q= find_q( $coor{$q} );
  my %$q= find_q( $coor{$q} );

```

- Используя биологическую информацию.

```
#
```

- Функциональное аннотирование в базах данных.

```

foreach my $q ( keys %$qartets){
  my $ix, my $iy, my $iz;
  my $ox, my $oy, my $oz;
  my $r;

```

- Используя информацию об активных сайтах, или мотивы.

```

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 );
```

Улучшение выравнивания

```

1 2 3 4 5 6 7 8 9 10 11 12 13 14
my %scoor=read_pdb($ARGV[0]);
my $scoor=read_pdb($ARGV[0]);
my $scoor=...;
foreach my $r ( sort keys %{$scoor{"0"}} ) { my $scoor=substr($r,0,1); if ( $scoor ne $ch ) { $scoor=$ch; } }
my %qwa=find_quart( $scoor{"0"} ); my $snum=keys %qwa;
PHE ASP ILE CYS ARG LEU PRO GLY SER ALA GLU ALA VAL CYS
PHE ASN VAL CYS ARG THR PRO --- --- --- --- GLU ALA ILE CYS
PHE ASN VAL CYS ARG --- --- --- THR PRO GLU ALA ILE CYS

```

```

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

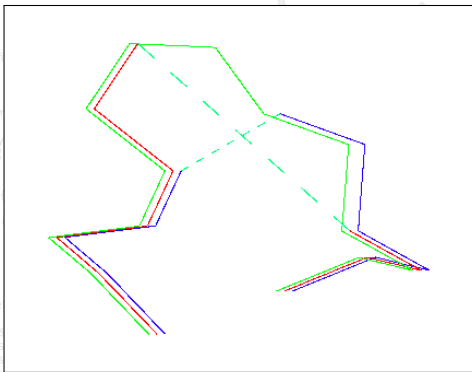
    foreach my $m ( sort { $a<=>$b } keys %qartets ) {
        my %qartets = %qwa; #find_quart(
        my %q = find_q( $scoor{$m} );

        # 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 $res ( @{$qartets{$q}} ) {
                #
                print "$q $scoor{$m} {$s";
                $nx=$nx+ $scoor{$m} {$res};
                $ny=$ny+ $scoor{$m} {$res};
                $nz=$nz+ $scoor{$m} {$res};
                $ox=$ox+ $scoor{$m} {$res}{"O6"}->x;
                $oy=$oy+ $scoor{$m} {$res}{"O6"}->y;
                $oz=$oz+ $scoor{$m} {$res}{"O6"}->z;
            }

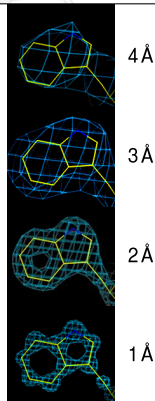
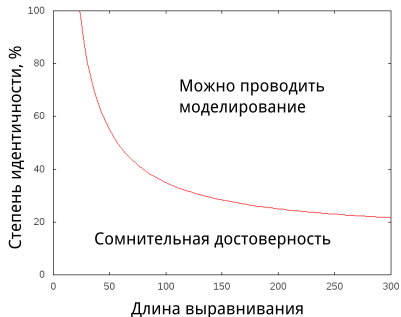
```



Из книги "Professional Gambling" от Bert Vriend

Качество белка заготовки

- Выбор качественного белка заготовки очень важен.
- Лучший вариант не обязательно обладает лучшей степенью идентичности.
 - Белок 1: ID 93%, 3.5 ангстрема разрешение. Хуже.
 - Белок 2: ID 90%, 1.5 ангстрема разрешение. Лучше!



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 %coor;
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sdir=$dir~/^([a-z]+)/;
foreach my $r ( sort keys %{$coor{0}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $sch ){ $schnum++; $sch=$ggg } };

my %a;

if ($qnum)
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename="-- s/^.*\//";
$filename="-- s/\.pdb//";
#$filename=$schnum.".".$qnum.".".$filename.".dat";
$filename="$sdir".".$filename.".dat";
print "$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}} ;

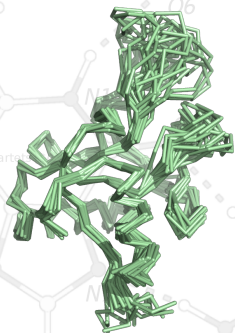
  foreach my $q { keys %qartets){

    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;

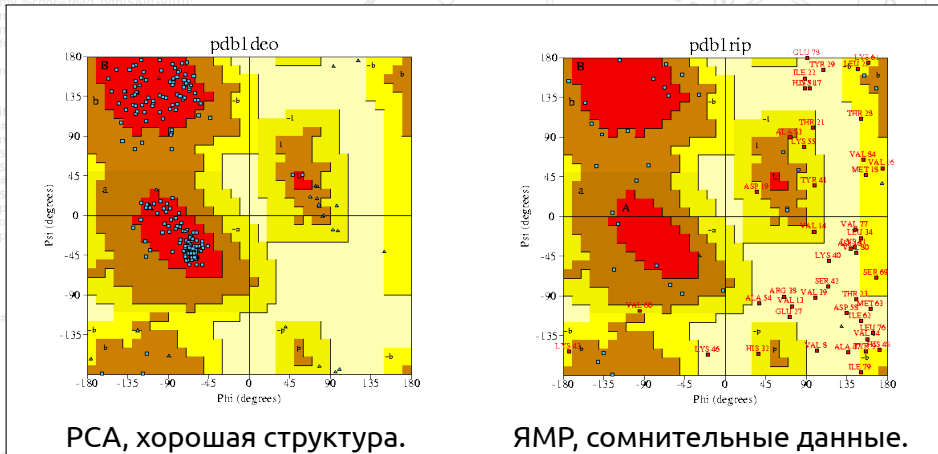
      $ox=$ox+ $coor{$m} {$res} {"O6"}->x;
      $oy=$oy+ $coor{$m} {$res} {"O6"}->y;
      $oz=$oz+ $coor{$m} {$res} {"O6"}->z;
```



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

Качество заготовки, Рамачандран

```
#[my %$coor,$my $chnum]=read_pdb($ARGV[0]);
my $Ramu=Ramachandran($my $ARGV[0]);
```



```
$nx=$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 );
```

Построение остова

```

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

- Генерируем координаты остова моделируемого белка для остатков из выравненных областей.

- Не обязательно использовать координаты, могут подойти дистанционные ограничения.

- Большинство исследователей предпочитают Modeller. Modeller использует дистанционные ограничения.

```

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

Моделирование петель

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

• **Эмпирическое моделирование:**

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

```
my %qwa=find_qwa($my $coor, $my $coor);
```

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

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

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

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

```
  $filename=$filename."
```

• Молекулярная механика.

• Монте-Карло.

• Rosseta:

- Поиск фрагментов близких по последовательности.
- Комбинирование результатов поиска с помощью Монте-Карло.

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

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

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

```
  my $r;
```

for **Комбинации выше перечисленных.**

```
  # 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 );
```

Моделирование боковых радикалов

```

#(my %$coor,my $snum)=read_pdb($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;
```

- 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 %$quartets = %qwa ; #find quart( $coor{$m} );
    #
    foreach my $q ( keys %$quartets ){
        print "sq $coor{$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 );
```

Моделирование 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{"O"}}){ my $ggg=substr($r,0,1); if ( $ggg ne $ch){ $chnum++; $ch=$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.

Это эмпирический метод на основе теории графов.

<http://dunbrack.fccc.edu/SCWRL3.php>

```

# foreach my $q ( keys %qwa ){
  foreach my $q ( keys %qwa ){
    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 );
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{"O"}} ){ my $ggg=substr($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } };

my %qwa=find_quart( $coor{"O"} ); 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 $q ( keys %qartets ){
    foreach my $q ( keys %qartets ){
        my $nx; my $ny; my $nz;
        my $ox; my $oy; my $oz;
        my $r;

        foreach my $res ( @{$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 );
```

Улучшение модели

```

#(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("cat $mdir/$filename.$qnum.out");
my $filename=$mdir.$sch.$qnum;
$filename=~s/"/"/g;
#system("cat $mdir/$filename.out");
print "INFO chain $schnum qnum $qnum\n";
open OUT, ">$filename.out";
print OUT "INFO chain $schnum qnum $qnum\n";

```

```

foreach my $m ( keys %$coor ){
my %$qartets= %$qwa ; #find quart( %$coor{$m} );
my %$q= %$qartets{$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;

```

- Методы минимизации энергии.
- Моделирование молекулярной динамики (оптимизация гидрофобики)
- Моделирование Монте-Карло.
- Любой известный подход для оптимизации структуры.

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

Ошибки

```

#(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 $ggg=subst($r,0,1); if ( $ggg ne $ch ){ $chnum++; $ch=$ggg } ;

```

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

• Обычно ошибки не исправляются на последующих этапах моделирования.

• Хорошее выравнивание не исправит плохой выбор белка заготовки.

• Хорошее моделирование петель не исправит плохое выравнивание.

• При обнаружении ошибки необходимо повторять некоторые этапы.

```

foreach my $m (sort keys %qwa){
  my %qartets= %qwa{$m}; my $qnum=keys %qartets;
  my %q= find_q( %coor{$m} );
  #
  foreach my $s ( @ { $qartets{$q} } ){
    print "$q $coor{$m} {$s} {" $R "}->x,\n";
    $nx=$nx+ $coor{$m} {$s} {"N9"}->x;
    $ny=$ny+ $coor{$m} {$s} {"N9"}->y;
    $nz=$nz+ $coor{$m} {$s} {"N9"}->z;

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

```

Проверка

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

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

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

- 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.

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

```
$nx=$nx+ $coor($m)($res){ "O6" }->x;
$ny=$ny+ $coor($m)($res){ "O6" }->y;
$nz=$nz+ $coor($m)($res){ "O6" }->z;
```

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

Ресурсы для гомологического моделирования

```
#!/(my $coor,$schnum)=read_pdb($ARGV[0]);
my $coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $sch,$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;
```

- Modeller

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

- SwissModel

```
foreach my $q ( keys %qartets ){
my %qartets=@qartets;
my %q=find_quart( $coor{$m} );
my %q=find_q( $coor{$m} );
```

- Eva-CM

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

- Nest И т.д.

```
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 );
```

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];
```

- Теоретически можно использовать молекулярную динамику.

- Моделирование отжига, как в МД так и в Монте-Карло.

- На основе фрагментов, Rosseta

```

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} {"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;
```

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.

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

my $coor = $ARGV[0];
my $mdir = $ARGV[1];
my $sch = $ARGV[2];
foreach my $r ( 0..$sch ){
    my $sggg = substr($r,0,1); if ( $sggg ne $sch ){ $schnum++; $sch=$sggg };
}

my $qnum = $schnum;
if ($qnum < 3 || $qnum > 9) {
    #system("mkdir $mdir/$qnum");
    my $filename = $qnum;
    $filename = "q$num";
    # $filename = "q$num";
    $filename = "q$num";
    print "$filename\n";
    open OUT, ">$filename";
    print OUT "#INFO chain $schnum qnum $qnum\n";

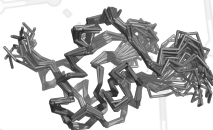
    foreach my $m ( sort { $a-<=>$b } keys %coor ){
        my %qartets = %qwa; #find quartet( $coor{ $m } );
        my %q = find_q( $coor{ $m } );

        # 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;
            }
        }
    }
}

```



Ab initio, Rosseta

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

my ($R,$N1,$N2,$N7,$O6);

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

• 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 $my $res ( @{$my $partets{$my $q}} ){
#
#   print "$my $q $my $coor{$my $res} {"$my $R"}->x,"$my $n";
#   $my $nx=$my $nx+ $my $coor{$my $res} {"$my $N9"}->x;
#   $my $ny=$my $ny+ $my $coor{$my $res} {"$my $N9"}->y;
#   $my $nz=$my $nz+ $my $coor{$my $res} {"$my $N9"}->z;
#
#   $my $ox=$my $ox+ $my $coor{$my $res} {"$my $O6"}->x;
#   $my $oy=$my $oy+ $my $coor{$my $res} {"$my $O6"}->y;
#   $my $oz=$my $oz+ $my $coor{$my $res} {"$my $O6"}->z;
# }
```

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

```

Threading — протягивание нити

```

#(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 %$a
```

- 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 упаковок

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

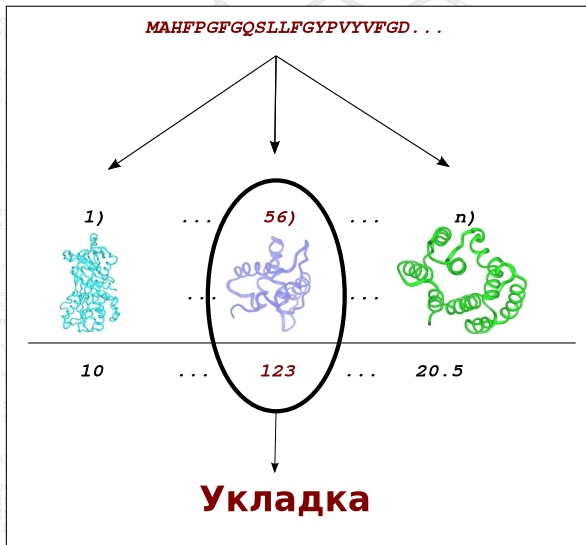
- На основе суммарного результата определяем белок заготовку.

```

foreach my $m (sort keys %$a){
  my %$qartets= %$a{$m};
  my %$q= find_q($m);
  #
  foreach my $s (@{ $qartets{$m} }){
    my $nx; my $ny; my $nz;
    my $ox; my $oy; my $oz;
    my $r;
    foreach my $res ( @{ $qartets{$s} } ){
      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;
    }
  }
}

```

Threading — протягивание нити



```
#!/usr/bin/perl
use Math::VectorReal qw( :all );
use List::Util qw( min );
use List::MoreUtils qw( first );
```

Threading — недостатки

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

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

Есть гибридные методы Rosseta/Threading: I-Tasser

```
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 );
```

Распознавание укладки, Phyge2

```

#(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="-- s/^.*\//";
$filename="-- s/\.pdb//";
ARLDLVIPMIYCGHGY

```

PSI-Blast

HMM

```

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}} ){
```

```

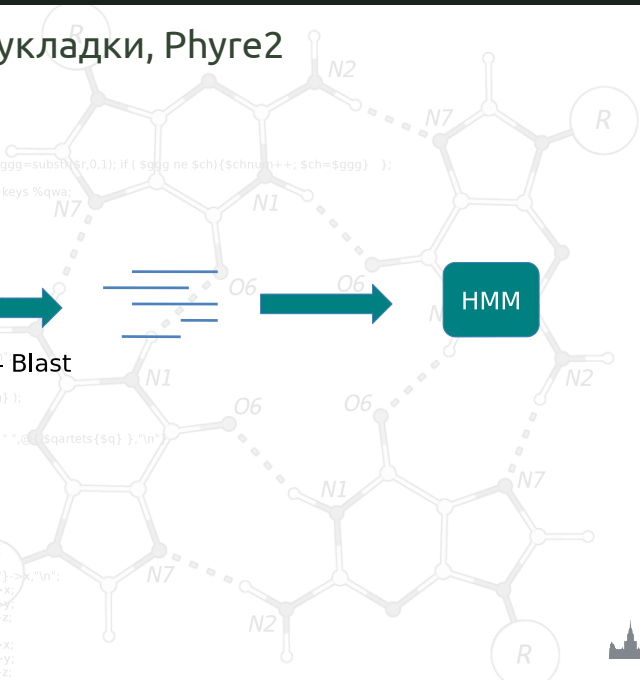
# print "$q %coor{$m}{ $res } {"N1"}->x, "\n";
$nx=$nx+ %coor{$m}{ $res } {"N1"}->x;
$ny=$ny+ %coor{$m}{ $res } {"N1"}->y;
$nz=$nz+ %coor{$m}{ $res } {"N1"}->z;

```

```

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

```



Phyre2

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

my $seq = "ARDLVIPMIYCGHGY";
my $target = "AFDLCDLIPV";

my ($coor, $schnum) = read_pdb($ARGV[0]);
my $coor = read_pdb($ARGV[0]);
my $dir = $ARGV[1];

substr($seq, 0, 1); if ( $seq =~ /C/ ) {
    $seq = "C" . substr($seq, 1);
}
```

PSI- Blast

HMM

HMM
профили
известных
структур

HMM-HMM
сравнение

```
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 . "_" . $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 = %$qwa; #find_quart($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} {$res} {"$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;
            }
        }
    }
}
```

ARDL - -VIPMIYCGHGY
AFDLCDLIPV - -CGMAY



Phyre2

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

ARDLVIPMIYCGHGY → PSI- Blast → HMM
```

HMM - Позволяет найти далёкие гомологи
Часто предсказываются хорошие модели при идентичности менее 15%

3D-Model



HMM профили известных структур

HMM-HMM сравнение

ARLD - - VIPMIYCGHGY
AFDLCDLIPV - - CGMAY

```
# print "$q :>X,";
# $nx=$nx+ $co
# $ny=$ny+ $co
# $nz=$nz+ $co;

# $ox=$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 );
```

Мета серверы

```

#(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 %da

```

- Сравнение разных методов.

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

```

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

```

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

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

- Суперпозиция результатов, взвешивание.

- Часто выдают только позиции атомов остова.

```

foreach my $res ( @ { $quarters { $q } } ){
#
print "$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;

```


Заключение

```
#!/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;
```

- 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.

```
if ($qnum)
#system("mv $mdir/$schnum.$qnum.$sch $mdir/$schnum.$qnum.$sch.$qnum.$qnum.$sch");
my $filename=$ARGV[0];
my $filename=$schnum.".".$qnum.".".$filename;
my $filename=$schnum.".".$qnum.".".$filename;
print "$filename\n";
open OUT,">$filename";
print OUT;

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

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