

```
#!/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 $sq=sqrt($r*0.1); if ( $sq ne $ch){ $chnum++; $ch=$sq; }
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 } {"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;
$r=$res;
```

```
}
```

# Структурная Биоинформатика

## Лекция 3, Предсказание вторичной структуры белка

Головин А.В. <sup>1</sup>

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

Москва, 2012

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

# Содержание

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

## Введение

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

## Определение ВС

```
my $filename=$ARGV[0];
$filename=~ s/~/\//;
$filename=~ s/\./_/;
my $filename="$schnum"."$qnum"."$filename.dat";
print "filename: ";
open OUT,">$filename";
print OUT "#INFO chain $schnum qnum $qnum\n";
```

## Предсказание ВС

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

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

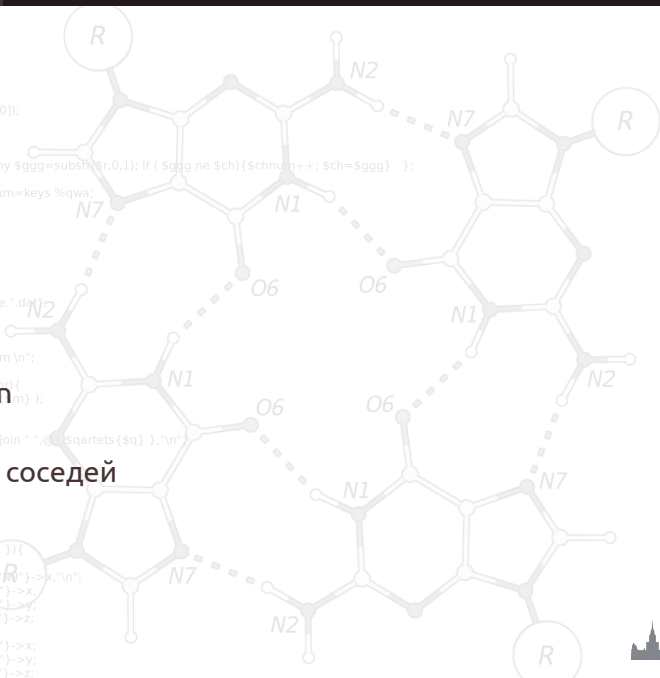
## Метод ближайших соседей

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

## Нейронные сети

```
# 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 );
```

# Первичная структура

```

#(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/ / /;
$filename="dir". $filename. ".dat";
print "$filename\n";
open OUT,">$filename";

```

Первичная структура – это аминокислотная последовательность:

Met-Ala-Gly-Trp-Ala-Val-Asp ...

```

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;
```

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

## Вторичная структура

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

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

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

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

**Вторичная структура белка** – это упорядоченные расположения атомов основной цепи полипептида, безотносительно к типам боковых цепей (групп) и их конформациям.

Если упорядоченность такова, что двугранные углы одинаковы у всех остатков, то говорят о регулярной вторичной структуре. Регулярными вторичными структурами являются спирали и  $\beta$ -структуры.

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

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

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

```
foreach my $res (@L_qartets/$q) {
```

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

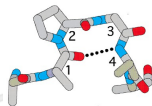
```
$ny=$ny+ $coor{$m} {$res} {"N9"}->y;
```

```
$nz=$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;
```



Пример нерегулярной вторичной структуры  $\beta$ -поворот ( $\beta$ -изгиб, реверсивный поворот).

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

# Регулярные вторичные структуры

```
my ($my %coor, my $chain) = read_pdb($ARGV[0]);
my %coor = read_pdb($ARGV[0]);
my $idir = $ARGV[1];
my $chain, my $atom;
foreach my $res ($chain->residues) {
```

```
my %qwa = find
```

```
if ($sqnum > 0)
#system("mkdir -p $idir/$chain/$sqnum");
my $filename = "alpha-helix-$sqnum.pdb";
my $filename = "parallel-beta-sheet-$sqnum.pdb";
my $filename = "anti-parallel-beta-sheet-$sqnum.pdb";
print "$filename\n";
open OUT, ">$idir/$chain/$sqnum/$filename";
print OUT "#PDB\n";
```

```
foreach my $res ($chain->residues) {
my %qwa = find
my %q = find
```

```
# foreach
```

```
foreach
```

```
my $nx = $res->name;
my $ny = $res->name;
my $nz = $res->name;
```

```
foreach
```

```
#
```

```
print "sq $sqnum ($chain) ($idir) \n";
```

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

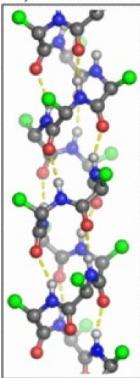
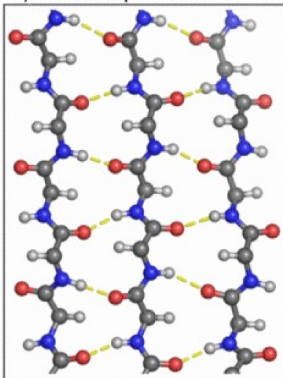
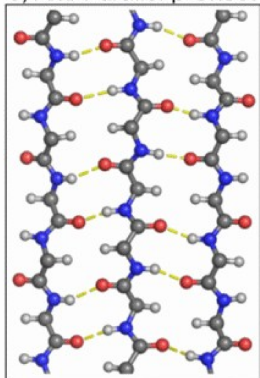
```
$ny = $ny + $coor{$res}{N9}->y;
```

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

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

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

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

a)  $\alpha$ -Helixb) Parallel  $\beta$ -Sheetc) Anti-Parallel  $\beta$ -Sheet

# Постулаты

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

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

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

- Современные методы не позволяют в целом предсказывать структуру белка

```
if ($snum > 0){
#system("cat $snum $qwa");
my $filename=$snum.$qwa;
$filename=~ s/~/V//;
$filename=~ s/\./pdb//;
#$filename=$snum.".".$qnum.".".$filename.".dat";
$filename=$mdir.$filename.".dat";
print "INFO: $filename\n";
open OUT, ">$filename";
print OUT "INFO: chain $snum $qwa\n";
foreach my $m (sort { $a-<=>$b } keys %$coor){
my %qartets = %qwa; #find_quart( $coor{$m} );
my %q = find_q( $coor{$m} );
```

- Неточность методов предсказания структуры связано с недостаточной мощностью компьютеров.

```
#
foreach my $q ( keys %qartets ){
my $ix=$q[0]; my $iy=$q[1];
my $ox=$q[2]; my $oy=$q[3];
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 );
```

# Классификация

```

#(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 } ;

```

## Особенности:

- Паттерны водородных связей
- Двугранные углы

## Программы для определения

- DSSP
- STRIDE

## • Continuum

```

foreach my $res ( @ { $qartets{$sq} } ){
#
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;

```

# Обозначения

## Восемь состояний аминокислоты в DSSP:

- H:  $\alpha$ -helix
- G:  $3_{10}$ -helix
- I:  $\pi$ -helix
- E:  $\beta$ -strand
- B: bridge
- T:  $\beta$ -turn
- S: bend
- C: coil

## CASP:

- H = (H, G, I), E = (E, B), C = (C, T, S)

24	26	E	H	<	S+	0	0	132
25	27	R	H	<	S+	0	0	125
26	28	N		<		0	0	41
27	29	K				0	0	197
28		!				0	0	0
29	34	C				0	0	73
30	35	I	E	-cd		58	89B	9
31	36	L	E	-cd		59	90B	2
32	37	V	E	-cd		60	91B	0
33	38	G	E	-cd		61	92B	0



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

# Двугранные углы

```

#(my %coor,my $schnum)
my %coor=read_pdb($AF
my $dir=$ARGV[1];
my $sch, my $schnum;
foreach my $r ( sort keys
my %qwa=find_quart( $C

```

```

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

```

```

foreach my $m (sort {$s
my %qartets = %qwa;
my %q= find_q( $coor

```

```
# foreach my $q { key
```

```
foreach my $q ( key
```

```

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

```

```
foreach my $res
```

```

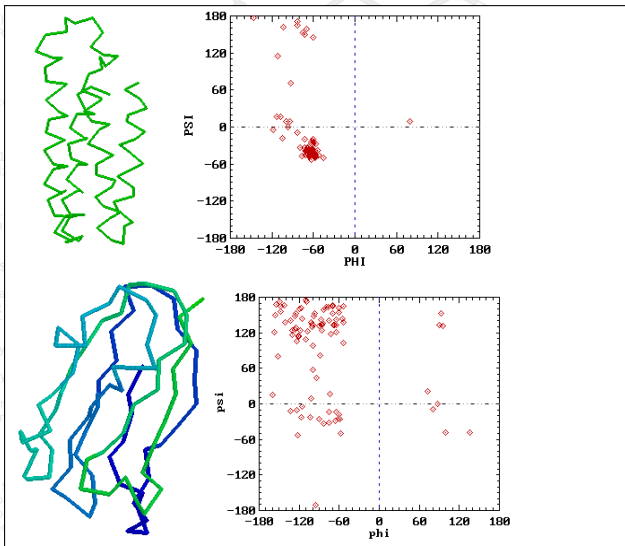
# print "$q
$nx=$nx+ $co
$ny=$ny+ $co
$nz=$nz+ $co

```

```

$ox=$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 );
```

# Предсказание вторичной структуры

```

#(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 $sqnum=keys %$qwa;
```

- Для последовательности:  
**GHWIATRGQLIREAYEDYRHFSSSECPFI**

- Предсказать состояние каждой аминокислоты

```

GHWIATRGQLIREAYEDYRHFSSSECPFI
CEEEEEESNNNNNNNNNNNNHCCSNHCCCCC
```

```
# 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;
```

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

# Цели предсказания

```

#(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 } ;

```

- Предсказание вторичной структуры проще чем предсказание третичной

- Аккуратное предсказание может упростить предсказание третичной структуры

- На основе вторичной структуры можно предположить функцию белка

- Классификация белков

- Предсказание изменения структуры при функционировании белка

```

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 );
```

# Основные методы

```

#(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;
```

- Статистические:  
Chou-Fasman method, GOR I-IV
- Методы ближайших соседей:  
NNSSP, SSPAL

- Нейронные сети  
PHD, Psi-Pred, J-Pred

- HMM

```

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

# Аккуратность

```
#!/(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("cat $ARGV[1]");
  my $filename=$ch.$chnum;
  $filename="-- s/\.pdb//";
  $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 $q;
  my $q;
  • Для всех петель  $Q_3 \sim 40\%$ 
```

```
#   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};
```

• Three-state prediction accuracy:  $Q_3$

$$Q_3 = \frac{\text{correctly predicted residues}}{\text{number of residues}}$$

R

R

R

R



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

# Развитие аккуратности

```

#(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"});
```

```

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

```

```

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

```

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

```
foreach my $q { keys %qart
```

```

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;
```

1974	Chou, Fasman	50-53%
1978	Garnier	63%
1987	Zvelebil	66%
1988	Qian, Sejnowski	64.3%
1993	Rost, Sander	70.8-72.0%
1997	Frishman, Argos	<75%
1999	Cuff, Barton	72.9%
1999	Jones	76.5%
2000	Petersen et al.	77.9%

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

# Основные предположения

```

#(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;

```

- Последовательность содержит достаточно информации для предсказания
- Боковые группы определяют структуру
- Окно в 13-17 остатков достаточно для предсказания
- Основы для выбора размера окна:
  - $\alpha$ -спирали 5-40 остатков
  - $\beta$ -тяжи 5-10 остатков

```

if ($qnum)
#system("mkdir $ARGV[1]");
my $filename=$ARGV[1];
$filename=$filename.".pdb";
# $filename=$filename.".n";
print "$filename\n";
open OUT,">$filename";
print OUT "PDB: $ARGV[0]\n";

foreach my $m ( keys %$coor ){
my %$q=keys %$qwa;
my %$q=find_q( $coor{$m} );
# foreach my $s ( keys %$q ){ print join( " ", $q{$s}, "\n" ); }

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

foreach my $res ( @{$q{$s}} ){
# print "$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( // );
```

# Метод Chou-Fasman

```

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

- Из банка данных PDB установим вероятность нахождения остатка в той или иной структуре:

```

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

$$P_{\alpha}^i = \frac{P(\alpha|aa_i)}{p(\alpha)} = \frac{p(\alpha, aa_i)}{p(\alpha)p(aa_i)}$$

- Пример:

#Ala=2000, #residues=20000, #helix=4000, #Ala in helix=500

$P(\alpha, aa_i) = 500/20000$

$p(\alpha) = 4000/20000$

$p(aa_i) = 2000/20000$

$P = 500 / (4000/10) = 1.25$

```

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 );
```

Результат:

## Chou-Fasman Parameters

$P_{\alpha}$		$P_{\beta}$		$P_t$	
Glu	1.51	Val	1.70	Asn	1.56
Met	1.45	Ile	1.60	Gly	1.56
Ala	1.42	Tyr	1.47	Pro	1.52
Leu	1.21	Phe	1.38	Asp	1.46
Lys	1.16	Trp	1.37	Ser	1.43
Phe	1.13	Leu	1.30	Cys	1.19
Gln	1.11	Cys	1.19	Tyr	1.14
Trp	1.08	Thr	1.19	Lys	1.01
Ile	1.08	Gln	1.10	Gln	0.98
Val	1.06	Met	1.05	Thr	0.96
Asp	1.01	Arg	0.93	Trp	0.96
His	1.00	Asn	0.89	Arg	0.95
Arg	0.98	His	0.87	His	0.95
Thr	0.83	Ala	0.83	Glu	0.74
Ser	0.77	Ser	0.75	Ala	0.66
Cys	0.70	Gly	0.75	Met	0.60
Tyr	0.69	Lys	0.74	Phe	0.60
Asn	0.67	Pro	0.55	Leu	0.59
Pro	0.57	Asp	0.54	Val	0.50
Gly	0.57	Glu	0.37	Ile	0.47

#!/usr/bin/perl

use Math::VectorReal qw( :all );

## Chou-Fasman, Алгоритм

#(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;

• Для элементов Спираль, Тяж

• Ищем окно с 6 остатками где суммарный score &gt; 1

• Расширяем окно до score &lt; 1

• Двигаемся вперёд и повторяем

• Конфликт: Если получается, что в одном месте и спираль и тяж, то сравниваем P(h) и P(b)

• Точность: до ~ 60%

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"}-&gt;x,"n";

\$nx=\$nx+ \$coor{\$m}{\$res} {"N9"}-&gt;x;

\$ny=\$ny+ \$coor{\$m}{\$res} {"N9"}-&gt;y;

\$nz=\$nz+ \$coor{\$m}{\$res} {"N9"}-&gt;z;

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

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

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

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

# Метод ближайших соседей

```

#(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;

```

- Предсказываем ВС центрального остатка для выбранного сегмента, для которого известен ближайший гомолог
  - Из базы данных находим близкую последовательность
  - Или строим N лучших локальных выравнивания входной последовательности со всеми последовательностями базы
- Используем  $\max(p_\alpha, p_\beta, p_c)$  для ближайшего соседа или  $\max(s_\alpha, s_\beta, s_c)$  для консенсуса из выравнивания

```

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

# Environment preference score

```
#!/(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;
```

- Предположим, что каждая а.к. имеет предпочтение для специфического структурного окружения
- Структурные переменные: вторичная структура, доступность растворителю
- Для базы данных уникальных белков FSSP

```
if ($qnum > 0){
#system("cat $coor{$r} | perl -e 'use Math::VectorReal qw( :all );
my $filename=$coor{$r};
$filename=~s/~/~/;
$filename=~s/~/~/;
#system("cat $coor{$r} | perl -e 'use Math::VectorReal qw( :all );
$filename=$dir.$filename.dat";
print "$filename\n";
open OUT, ">$filename";
print OUT "#INFO chain $chnum qnum $qnum\n";
```

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

```
# 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 $coor{$r}{$res} {"R"}->x,\"n";
$nx=$nx+ $coor{$r}{$res} {"N9"}->x;
$ny=$ny+ $coor{$r}{$res} {"N9"}->y;
$nz=$nz+ $coor{$r}{$res} {"N9"}->z;
$ox=$ox+ $coor{$r}{$res} {"O6"}->x;
$oy=$oy+ $coor{$r}{$res} {"O6"}->y;
$oz=$oz+ $coor{$r}{$res} {"O6"}->z;
```

$$S(i, j) = \log \frac{p(aa_i | E_j)}{p(aa_i)} = \log \frac{p(aa_i, E_j)}{p(aa_i)p(E_j)}$$

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

# Environment preference score

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

```
my %coor=();
my $dir=$$;
my $sch,$num;
foreach my $res ($res){
```

```
my %qwa=();
```

```
if ($qnum > 0){
#system("rmdir $dir/$num");
my $filename=$dir.$num;
$filename=$dir.$num;
# $filename=$dir.$num;
$filename=$dir.$num;
print "$filename\n";
open OUT,">$filename.out";
print OUT "$num\n";
```

```
foreach my $res ($res){
my %qa=();
my %q=();
```

```
# foreach my $res ($res){
```

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

```
my %qa=();
```

```
my %q=();
```

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

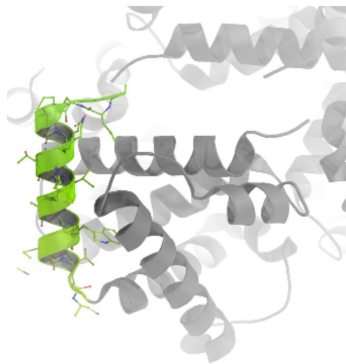
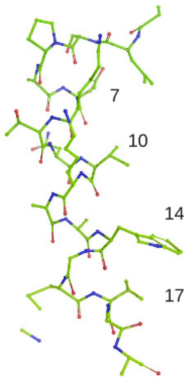
```
#
```

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

Гидрофильная часть

Гидрофобная часть



## Матрица "Singleton"

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

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

```

	Helix			Sheet			Loop		
	Buried	Inter	Exposed	Buried	Inter	Exposed	Buried	Inter	Exposed
ALA	-0.578	-0.119	-0.160	0.010	0.583	0.921	0.023	0.218	0.368
ARG	0.997	-0.507	-0.488	1.267	-0.345	-0.580	0.930	-0.005	-0.032
ASN	0.819	0.090	-0.007	0.844	0.221	0.046	0.030	-0.322	-0.487
ASP	1.050	0.172	-0.426	1.145	0.322	0.061	0.308	-0.224	-0.541
CYS	-0.360	0.333	1.831	-0.671	0.003	1.216	-0.690	-0.225	1.216
GLN	1.047	-0.294	-0.939	1.452	0.139	-0.555	1.326	0.486	-0.244
GLU	0.670	-0.313	-0.721	0.999	0.031	-0.494	0.845	0.248	-0.144
GLY	0.414	0.932	0.969	0.177	0.565	0.989	-0.562	-0.299	-0.601
HIS	0.479	-0.223	0.136	0.306	-0.343	-0.014	0.019	-0.285	0.051
ILE	-0.551	0.087	1.248	-0.875	-0.182	0.500	-0.166	0.384	1.336
LEU	-0.744	-0.218	0.940	-0.411	0.179	0.900	-0.205	0.169	1.217
LYS	1.863	-0.045	-0.865	2.109	-0.017	-0.901	1.925	0.474	-0.498
MET	-0.641	-0.183	0.779	-0.269	0.197	0.658	-0.228	0.113	0.714
PHE	-0.491	0.057	1.364	-0.649	-0.200	0.776	-0.375	-0.001	1.251
PRO	1.090	0.705	0.236	1.249	0.695	0.145	-0.412	-0.491	-0.641
SER	0.350	0.260	-0.020	0.303	0.058	-0.075	-0.173	-0.210	-0.228
THR	0.291	0.215	0.304	0.156	-0.382	-0.584	-0.012	-0.103	-0.125
TRP	-0.379	-0.363	1.178	-0.270	-0.477	0.682	-0.220	-0.099	1.267
TYR	-0.111	-0.292	0.942	-0.267	-0.691	0.292	-0.015	-0.176	0.946
VAL	-0.374	0.236	1.144	-0.912	-0.334	0.089	-0.030	0.309	0.998

```
$nx=$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;
```

# Total score

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

- Общее значение это сумма для окна  $l$

$$Score(i, j) = \sum_{k=-l/2}^{l/2} [M(i+k, j+k) + cS(i+k, j+k)]$$

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

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

	i 4	i 3	i 2	i 1	i	i+1	i+2	i+3	i+4															
	T	R	G	Q	L	I	R	E	A	Y	E	D	Y	R	H	F	S	S	E	C	P	F	I	P
	.	.	.	E	C	Y	E	Y	B	R	H	R	.	.	.	.	.	.	.	.	.	.	.	.
	j 4	j 3	j 2	j 1	j	j+1	j+2	j+3	j+4															
	L	H	H	H	H	H	H	L	L															

```
# print "sq $coor{$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( || );
"Соседи"
```

```
#!/(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"} ); m
```

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

1	-	L	H	H	H	H	H	L	L	-	S <sub>1</sub>
2	-	L	L	H	H	H	H	L	L	-	S <sub>2</sub>
3	-	L	E	E	E	E	E	L	L	-	S <sub>3</sub>
4	-	L	E	E	E	E	E	L	L	-	S <sub>4</sub>
n	-	L	L	L	L	E	E	E	E	-	S <sub>n</sub>
n+1	-	H	H	H	L	L	L	E	E	-	S <sub>n+1</sub>

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

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

$$\max(n_\alpha, n_\beta, n_L) \quad OR \quad \max(\sum S_\alpha, \sum S_\beta, \sum S_L)$$

```
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 );
```

# Как использовать эту информацию:

```

#(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;
```

- Выравнивание последовательность-профиль

```

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

```

- Сравнение последовательности с белковыми семействами

```

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

```

- BLAST против PSI-BLAST.

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

- Использование PSSM вместо матриц PAM или BLOSUM.

```

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;

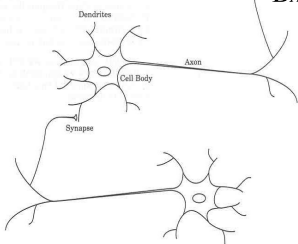
```

# Нейронные сети

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

```
my %qwa=
{
  "Dendrites" => "Dendrites",
  "Cell Body" => "Cell Body",
  "Axon" => "Axon",
  "Synapse" => "Synapse"
};

if ($schnum > 1) {
  #system("cp $filename $filename.$schnum");
  $filename=$filename.$schnum;
  #system("cp $filename $filename.$schnum");
  print "$filename\n";
  open OUT, ">$filename";
  print OUT "$schnum\n";
  foreach my $f (sort keys %{$coor{"0"}}) {
    my %qwa=
    {
      "Dendrites" => "Dendrites",
      "Cell Body" => "Cell Body",
      "Axon" => "Axon",
      "Synapse" => "Synapse"
    };
    my %qwa=
    {
      "Dendrites" => "Dendrites",
      "Cell Body" => "Cell Body",
      "Axon" => "Axon",
      "Synapse" => "Synapse"
    };
  }
}
```

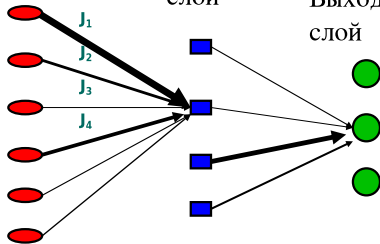


нейроны

Входной слой

Скрытый  
слой

Выходной  
слой



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

# Обучение сети

```
my %coor; my $chain; readpdb($ARGV[0]);
```

```
my %coor;
my $mdir=$
my $ch, m
foreach m
my %qwa-
```

```
if ($qnum
#system("
my $fina
$file:=$
$file:=$
$file:=$
print "$file
open OUT
print OUT;
```

```
foreach m
my %q;
my %q;
```

```
# fore
```

```
fore
```

```
my
```

```
my
```

```
for
```

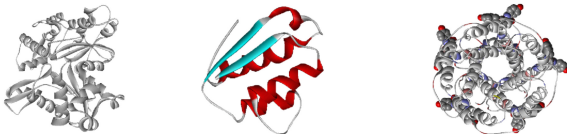
```
#
```

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

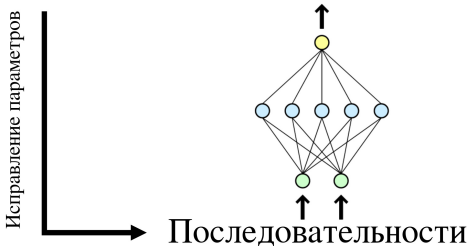
```
$ox=$ox+ $coor{$m}{$res}{"O6"}->x;
```

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

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



## Сравнение предсказания с реальностью



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

# Стратегия разумного предсказания

```
#!/(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 $dir/$ch/$qnum");
my $filename=$dir/$ch/$qnum;
$filename=$ch.$qnum;
$filename=$ch.$qnum;
#system("cp $dir/$ch/$qnum/$ch.pdb $filename");
$filename=$ch.$qnum;
print "filename=$filename\n";
open OUT, ">$filename.dat";
print OUT "#IN:O Chain $chnum $qnum $qnum\n";
```

```
foreach my $m ( keys %coor ){
my %qartets = %qwa; #find quart( $coor{$m} );
my %q = find_quart($coor{$m});
# foreach my $q ( keys %qartets ){ print join " ", @{$qartets{$q}}, "\n";
```

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

```
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;
```

```
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;
```

- Построить выравнивание с гомологами, чем больше тем лучше
- Применить как можно больше разных современных методов предсказания
- Внимательно следить за паттернами и консервативными остатками
- Построить консенсус

# Реализация

## Серверы:

- JPREД: <http://www.compbio.dundee.ac.uk/~www-jpred/>
- PHD: <http://cubic.bioc.columbia.edu/predictprotein/>
- PSIPRED: <http://bioinf.cs.ucl.ac.uk/psipred/>
- NNPREДICT:  
<http://www.cmpharm.ucsf.edu/~nomi/nnpredict.html>
- Chou and Fassman:  
[http://fasta.bioch.virginia.edu/fasta\\_www/chofas.htm](http://fasta.bioch.virginia.edu/fasta_www/chofas.htm)

## Интересная статья:

Rost and Eyrich. EVA: Large-scale analysis of secondary structure prediction. *Proteins* 5:192-199 (2001)