

```
#!/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} ) {
```

```
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=$chnum.".".$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= %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;
```

```
}
```

Структурная Биоинформатика

Лекция 6. Поиск новых биоактивных молекул и химиоинформатика.

Головин А.В.¹

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

Москва, 2012

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

Содержание

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

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

Активные молекулы

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

Химоинформатика

SMILES

SMARTS

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

QSAR

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

Докинг




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

Свойства лекарства

```

#(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;
```

- Лекарством обычно являются не только те молекулы, которые хорошо связываются с биополимером.

- Лекарство должно иметь приемлемую растворимость

- Часто бывает, что лекарству надо проникнуть сквозь мембрану,

- Хорошо когда лекарство в итоге метаболизируется, а не накапливается в тканях.

```

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

```

```

# foreach my $q ( keys %$qwa ){
  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;

```


Компьютерное представление молекул

- Хранение в компьютере молекулы как изображения имеет малую ценность
- Большинство современных баз данных представляет молекулу как граф, с узлами и рёбрами
- Графы представляются как таблицы связей.

Marvin 04200617372D

```

4 3 0 0 0 0 0 999 V2000
0.0000 0.0000 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.7145 -0.4125 0.0000 D 0 0 0 0 0 0 0 0 0 0 0 0 0 0
-0.7145 -0.4125 0.0000 C 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0.0000 0.8250 0.0000 D 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 4 2 0 0 0 0
2 1 1 0 0 0 0
3 1 1 0 0 0 0
M END
sox=$ox+ $coor{$m}{sres}{"O6"}->x;
soy=$oy+ $coor{$m}{sres}{"O6"}->y;
soz=$oz+ $coor{$m}{sres}{"O6"}->z;

```

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

Содержание

```
#!/(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;
```

Активные молекулы

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

Химоинформатика

SMILES

SMARTS

```
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";
```

QSAR

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

Докинг

```
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;
```



Линейное представление молекул, SMILES

Примеры:

CC ethane

[OH3+] hydronium ion

O=C=O carbon dioxide

[2H]]O[[2H] deuterium oxide

C#N hydrogen cyanide

[235U] uranium-235

CCN(CC)CC triethylamine

F/C=C/F E-difluoroethene

CC(=O)O acetic acid

F/C=C/F Z-difluoroethene

C1CCCCC1 cyclohexane

N[C@@H](C)C(=O)O L-alanine

c1ccccc1 benzene

N[C@H](C)C(=O)O D-alanine

Реакции в виде SMILES

[I-] . [Na+] . C=CCBr >> [Na+] . [Br-] . C=CCI
(C(=O)O) . (OCC) >> (C(=O)OCC) . (O)

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

Стандартизация SMILES

- Очевидно, что одну молекулу можно описать разными способами.
- Морган в 1965 году предложил рассматривать каждый атом по свойству его окружения.
- Стандартные SMILES называют Unique.

```
my %qartets = %qwa; #find_q
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 $res ( @($ $q
print "$q $coor($n
$nx=$nx+ $coor($m) {
$ny=$ny+ $coor($m) {
$nz=$nz+ $coor($m) {
$ox=$ox+ $coor($m) {$res}{"O6"}->x;
$oy=$oy+ $coor($m) {$res}{"O6"}->y;
$oz=$oz+ $coor($m) {$res}{"O6"}->z;
```

Input SMILES

Unique SMILES

OCC

CCO

[CH3][CH2][OH]

CCO

C-C-O

CCO

C(O)C

CCO

OC(=O)C(Br)(Cl)N

NC(Cl)(Br)C(=O)O

ClC(Br)(N)C(=O)O

NC(Cl)(Br)C(=O)O

O=C(O)C(N)(Br)Cl

NC(Cl)(Br)C(=O)O

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

Описание SMILES: атомы

```

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

```

- Однобуквенные атомы, а именно : B, C, N, O, P, S, F, Cl, Br, I записываются как есть, как один символ.

- Все остальные атомы записываются в квадратных скобках [Pt]

- Так как атомы водорода обычно не указываются, то “валентность” атомов определяется как наименьшая из ближайших T.e. B (3), C (4), N (3,5), O (2), P (3,5), S (2,4,6).

- “Валентности”, отличные от “нормальных”, указывают в скобках [S], [H+], [Fe+2], [OH-], [Fe++], [OH3+], [NH4+]

```

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

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

```

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

Описание SMILES: СВЯЗИ

```

#(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 $filename=$ARGV[0];
$filename="-- s/^.*\//";
$filename="-- s/\.pdb//";
#$filename=$schnum."_"$qnum."_"$filename.".dat";
$filename="$dir"$.filename.".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $schnum qnum $qnum\n";
```

```

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

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

Ветвление цепи отображается в скобках ()

Пример: CCC(CC)COO

```

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;
```

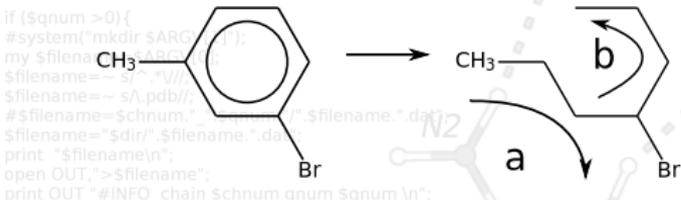
CC	этан
C=C	этилен
O=C=O	CO2
C#N	HCN
CCO	этанол
[H][H]	водород

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

Описание SMILES: циклы

- C1CCCCC1 циклогексан
- Или более сложный пример:

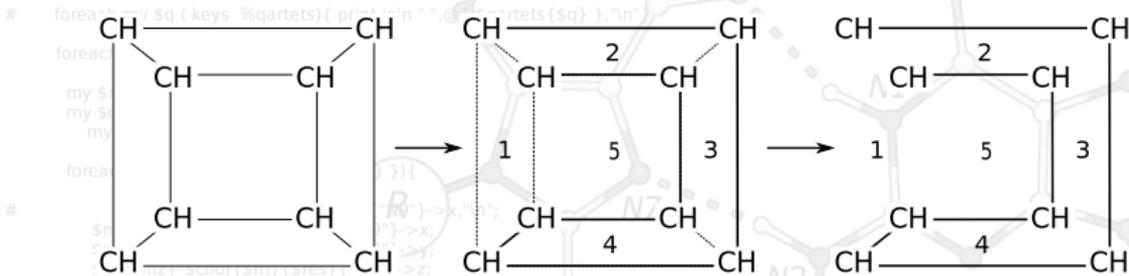
```
my %qwa=find_quart( $coor{"0"} ); my $qnum=keys %qwa;
```



- a) CC1=CC(=CC=C1)Br
- b) CC1=CC(Br)=CC=C1

- Самый сложный пример: C12C3C4C1C5C4C3C25

```
for( my %qartels= %qwa , #find_quart( $coor{ $m } );
      my %q= find_q( $coor{ $m } );
```



```
$ox=$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 );
```

Описание SMILES: ароматика

```

#(my %coor,my $chnum)=read_pdb($ARGV[0]);
my %coor=read_pdb($ARGV[0]);
my $dir=$ARGV[1];
my $cni=$ARGV[2];
foreach my $f ( sort { $a-<=>$b } keys %coor ) {
    my $q = $coor{$f};
    my %q = ( %coor{$f}, ( $q{N1}, $q{N2}, $q{N7}, $q{O6} ) );
}
my %qwa = ( %coor{$f}, ( $q{N1}, $q{N2}, $q{N7}, $q{O6} ) );

if ($qni) {
    #system("mkdir $ARGV[1]");
    my $filename = "chem_$cni_$qni";
    $filename = "$dir/$filename";
    # $filename = "$dir.pdb/";
    # $filename = "$dir/$cni_$qni";
    $filename = "$dir/$cni_$qni";
    print "$filename\n";
    open OUT, ">$filename";
    print OUT "C1=CC=CC=C1\n";
}

foreach my $m ( sort { $a-<=>$b } keys %coor ) {
    my $q = $coor{$m};
    my %q = ( %coor{$m}, ( $q{N1}, $q{N2}, $q{N7}, $q{O6} ) );
}

# 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} {"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;
    }
}

```

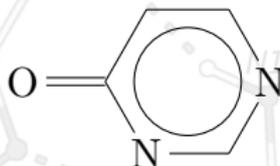
- SMILES для определения ароматичности использует расширенный алгоритм Хюккеля.

- c1ccccc1 eq C1=CC=CC=C1 тут все атомы находятся в sp^2 -гибридизации

- c1ccccc1 eq C1=CC=CC1, последний атом в гибридации sp^3 .

- Ароматичными могут быть атомы: C, N, O, P, S, As, Se, и *.

- Пример: c1cnc[nH]c(=O)1



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

Содержание

```

#(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;

```

Активные молекулы

```

if ($qnum > 0){
#system("mkdir $ARGV[1]");
my $filename=$ARGV[1];
my $filename=$dir."/".$qnum.".dat";
my $filename=$dir."/".$qnum.".dat";
my $filename=$dir."/".$qnum.".dat";
print "SMILES";
print "SMARTS";
open OUT, ">".$filename;
print OUT "INFO Chain $chnum qnum $qnum \n";

```

Химоинформатика

```

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

```

QSAR

```

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

```

Докинг

```

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 );
```

SMARTS: паттерны для SMILES

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

В принципе, SMARTS это SMILES + операторы логики и варианты в позициях.

Примеры для атомов:

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

```
  #system("mkdir $ARGV[0]");
```

```
  my $filename=$AR
```

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

```
  $filename=~ s/\.pd
```

```
  # $filename=$schnu
```

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

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

```
  open OUT,">$filen
```

```
  print OUT "#INFO
```

```
  foreach my $m (s
```

```
    my %qartets= %
```

```
    my %q= find_q
```

```
  # foreach my $
```

```
    foreach my $
```

```
    my $nx; my
```

```
    my $ox; my
```

```
    my $r;
```

```
    foreach my
```

```
  # print "atom name: $m\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;
```

C алифатический углерод

c ароматический углерод

a любой ароматический атом

[#6] любой атом углерода

[++] атом с зарядом +2

[R] атом в кольце

[D3] атом с тремя связями (не с водородами)

[X3] атом с тремя связями, включая водороды

[v3] атом с валентностью 3.

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

SMARTS: логические операторы и примеры

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

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

```
# foreach m
```

```
foreach n
```

```
my $nx;
```

```
my $ox;
```

```
my $r;
```

```
foreach
```

```
#
```

```
$nx=
$ny=$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;
```

Логика:

!e1 not e1

e1&e2 a1 and e2

e1,e2 e1 or e2

e1;e2 a1 and e2

Примеры:

[!C;R]

не алифатический C в кольце

[n;H1], [n&H1], [nH1]

H в пирроле

[c,n&H1]

C или H в пирроле

[X3&H0]

Атом с тремя связями не с H

[c,n;H1]

N или C в связи с одним H1


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

Проблемы с фармакофорами

```

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

```

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

- Если молекулы более или менее подвижны, то это накладывает дополнительные требования на учёт конформационных превращений.

- Для определения фармакофора надо определить, какой набор групп располагается в биополимере идентично.

- Надо быть уверенным, что выбранный набор молекул связывается с белком в одном и том же месте. Однозначное указание на это можно получить только экспериментально.

```

foreach my $m (sort { $a<=>$b } keys %coor){
  my $q;
  my $q;
  #
  foreach my $q ( keys %qartets){
    foreach my $s ( $coor{"$q"} ){
      my $nx, my $ny, my $nz;
      my $ox, my $oy, my $oz;
      my $r;
      foreach my $res ( @{$ $qartets{"$q"} }){
        #
        print "$q $coor{$m}{$res} {"$R"}->x,"n";
        $nx=$nx+ $coor{$m}{$res} {"$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 );
```

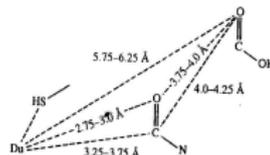
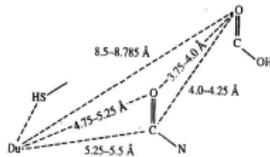
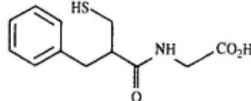
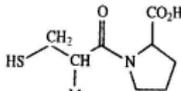
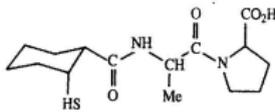
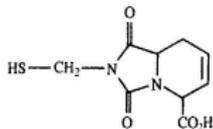
Систематический поиск

```
my ($my $coor, my $schnum)=read_pdb($ARGV[0]);
my $my $coor=read_pdb($ARGV[0]);
my $mdir=$ARGV[1];
my $m;
foreach my $m ($mdir) {
    my %qwa;
```

• Есть проблема:

```
my %qwa;
```

```
if ($qnum
#system("
my $filena
$filename=
$filename=
$filename=
$filename=
print " $file
open OUT
print OUT;
```



- Выбирают точки, которые по мнению исследователей определяют активность. Делают конформационный поиск для всех молекул. Если находят пересечения по геометрии, то на основе этих точек и геометрии пересечения формулируют фармакофор.

```
foreach my $m ($mdir) {
    my %qartets = %qwa; #find quart ($coor ($m) );
    my $q;
    # foreach my $q1 ($qartets) { print " $qartets ($q1) ";
    foreach my $q2 ($qartets) {
        my $sox; my $soy; my $soz;
        foreach my $sres ($qartets ($q2)) {
            print " $scoor ($m) ($sres) {" "N9" }->x;
            $nx=$nx+ $coor ($m) ($sres) {" "N9" }->x;
            $ny=$ny+ $coor ($m) ($sres) {" "N9" }->y;
            $nz=$nz+ $coor ($m) ($sres) {" "N9" }->z;
            $sox=$sox+ $coor ($m) ($sres) {" "O6" }->x;
            $soy=$soy+ $coor ($m) ($sres) {" "O6" }->y;
            $soz=$soz+ $coor ($m) ($sres) {" "O6" }->z;
```



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

Положение в сайте связывания

```

#(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("cp $dir/$ch/$qnum/$chnum/$coor{"0"}.$qnum.dat");
my $filename=$dir.$ch.$qnum.$chnum.$coor{"0"}.$qnum.dat;
$filename=~ s/\./\//;
$filename=~ s/\./\./;
# $filename=$chnum.".".$qnum."/".$filename.".dat";
$filename="$dir".$filename.".dat";
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $chnum qnum $qnum\n";

```

• Сайт связывания — место связывания лиганда

```

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

```

• Геометрия связывания — место связывания, ориентация и конформация лиганда

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

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

```

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

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

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

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

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

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

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

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

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

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

Использование докинга

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

Основные цели докинга:

- Виртуальный поиск лигандов
- Определение геометрии связывания лиганда

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

```
my %qwa
if ($qnum)
#system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
$filename="--s/^.*\\.v//";
$filename="--s/\\.pdb//";
#$filename=$chnum.".$qnum.".$filename.".dat";
$filename="$dir".$filename.".dat";
print "$filename\n";
```

Если мы знаем, как связывается лиганд, то:

- Мы можем узнать, какие части важны для связывания
- Можно предложить изменения для улучшения константы связывания
- Можем избежать ошибок

```
foreach my $m (sort {$a<=>$b} keys %coor){
my $qwa=$coor{$m}["O"];
my %qwa
```

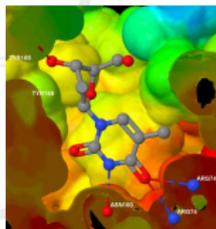
```
# foreach my $q ( keys %qartets){
```

```
foreach my $q ( keys %qartets){
my $nx=$coor{$m}{"N9"}->x;
my $ny=$coor{$m}{"N9"}->y;
my $nz=$coor{$m}{"N9"}->z;
```

```
foreach my $res ( @{$qartets{$q}} ){
print "$q $coor{$m}{"$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 );
```

Два основных компонента программ для докинга

```

#(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("cp $ch $dir");
my $filename=$ARGV[0];
$filename="-- s/ /_/";
$filename="-- s/\.pd/ /";
#$filename=$ch.$chnum;
$filename="-- $dir"/$filename;
print "$filename\n";
open OUT,">$filename";
print OUT "#INFO chain $chnum qnum $qnum\n";

```

```

foreach my $r ( sort keys %{$coor{"$ch"}}){
my %q=find_q( %coor{"$ch"} );
my %q=find_q( %coor{"$ch"} );
# foreach my $tets ( keys %qartets){ print join " ",@{$qartets{$tets}},"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;

```

- Алгоритм поиска

- Установление места связывания

- Установление геометрии связывания

- Алгоритм расчёта константы связывания областей с низкой энергией.

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

Реализация

```

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

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

Сегодня существует много программ для докинга

- AutoDock, DOCK, e-Hits, FlexX, FRED, Glide, GOLD, LigandFit, QXP, Surflex-Dock...и т.д.

- разные алгоритмы оценки аффинности и разные алгоритмы поиска

- Важно не путать лиганд-белок докинг и белок-белок докинг

```

# system("mkdir $ARGV[1]");
my $filename=$ARGV[0];
my $filename=$ARGV[0];
my $filename=$ARGV[0];
my $filename=$ARGV[0];
print "$filename\n";
open OUT,">$filename";
print OUT;

foreach my $m (=>$b) keys %$coor{
  my %$q=find_quart( $coor{$m} );
  my %$q=find_q( $coor{$m} );
#
  foreach my $q ( keys %$qartets){
    my $nx; my $ny; my $nz;
    my $ox; my $oy; my $oz;
    my $r;

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

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

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

Практические аспекты

```

#(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;

```

• Часто PDB-структура содержит молекулы воды, почти всегда их надо убрать.

• Надо добавлять протоны к структуре; His?

• Часто в PDB неточно определена ориентация некоторых групп, что сказывается на паттерне водородных связей.

• Протонирование лиганда и его таутомерные формы.

```

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

  #
  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;
    }
  }
}

```

Rigid|Flexible докинг

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

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

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

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

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

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

```
  $filename=$sch.$schnum;
```

```
  $filename="-- $schpdb/";
```

```
  # $filename=$sch.$schnum;
```

```
  $filename=$sch.$schnum;
```

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

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

```
  print OUT
```

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

```
    my $q=find_q( $coor{$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}{$res} {"$res"}->x;
```

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

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

• **Rigid:** лиганд не имеет внутренних степеней свободы, т.е. вращение вокруг связей запрещено.

• **Flexible:** предполагает учёт вращения вокруг связей лиганда.

• Часто белок рассматривается как жёсткое тело