Développement de protéines *de novo*

Elyna BOUCHEREAU

2023-02-17

I. Introduction

Identifiant PDB	Score Forsa global	Score Norm Forsa global	Z-Score Forsa local	Longueur	Remarque
2XIW	93.550	1.396	5.532	66	Sans tag Histidine
1AZP	122.628	1.858	7.023	66	
1AZQ	122.606	1.858	7.022	66	
1BF4	88.762	1.409	5.461	63	
1BNZ	97.222	1.519	5.847	64	
1C8C	78.492	1.226	4.896	64	
1CA5	72.121	1.076	4.505	67	
1CA6	118.235	1.791	6.804	66	
1SAP	116.701	1.768	6.727	66	
1WD0	117.895	1.786	6.787	66	
1WD1	102.041	1.546	5.996	66	
1WT0	11.647	1.692	6.475	66	
1WTP	98.599	1.494	6.137	66	
1WTR	117.136	1.775	6.749	66	
1WTV	108.200	1.639	6.304	66	
1WTX	112.8979	1.710	6.537	66	
1WVL	121.016	1.513	6.333	80	
1XX8	64.767	0.967	4.138	67	
1XYI	124.251	1.883	7.104	66	
3LWH	134.211	2.237	8.004	60	
3LWI	125.677	2.095	7.553	60	
4CJ0	45.527	0.641	3.085	71	Sans tag Histidine
4CJ1	80.191	1.129	4.742	71	Sans tag Histidine
4CJ2	80.979	1.191	4.876	68	Sans tag Histidine

Score maximal est à 8.004 et score minimal à 3.085.

II. Forsa

1.1. Chargement des librairies et initialisation des variables

```
## # A tibble: 26 x 26
```

```
##         A     B     C      D      E      F      G     H      I     J     K
##     <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl> <dbl>  <dbl> <dbl> <dbl>
##  1 0.0625     0     0 0.0938 0.109  0.0312 0.0781     0 0.0156     0 0.156
##  2 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
##  3 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
##  4 0.0333     0     0 0.05   0.117  0.0333 0.117      0 0.05       0 0.217
##  5 0.0333     0     0 0.05   0.117  0.0333 0.117      0 0.05       0 0.217
##  6 0.05       0     0 0.05   0.0833 0.0333 0.117      0 0.05       0 0.217
##  7 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
##  8 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
##  9 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
## 10 0.0625     0     0 0.0781 0.109  0.0312 0.0781     0 0.0156     0 0.203
## # ... with 16 more rows, and 15 more variables: L <dbl>, M <dbl>, N <dbl>,
## #   O <dbl>, P <dbl>, Q <dbl>, R <dbl>, S <dbl>, T <dbl>, U <dbl>, V <dbl>,
## #   W <dbl>, X <dbl>, Y <dbl>, Z <dbl>
```

```
#### II. Initial round ----
nb_of_bests <- 10
nb_of_mutants <- 1000
nb_of_init_seq <- 10000
```

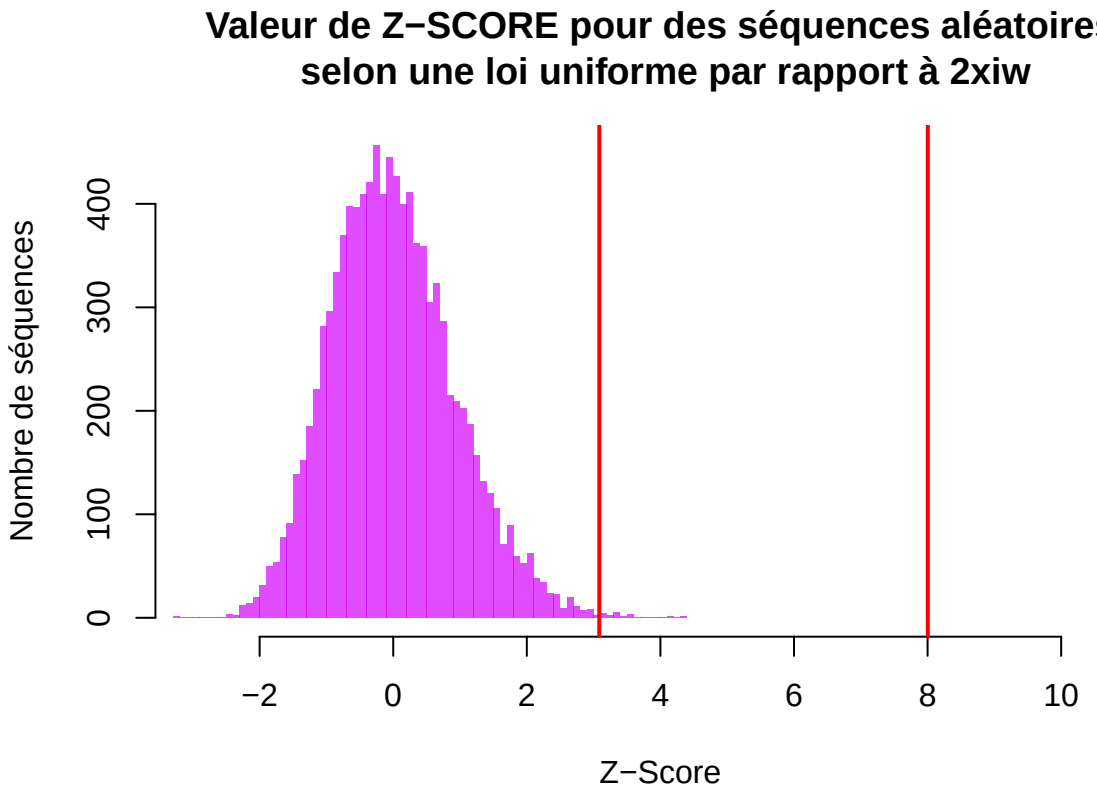
1.2. Création de la génération 0

```
### - 2.2. Generating the first batch of random sequences ----
sequences = {}
header = ""
sequences <- foreach(i=1:nb_of_init_seq,.combine = 'c') %dopar% {
  sequences[i]=paste(sample(amino_acid_pb_df$letter, seq_length, replace=TRUE, prob = amino_acid_pb_df$p
  header = paste("Random_Sequence_",i,sep = "")
  sequence = paste(">",header,"\n",sequences[i],sep = "")
  write(sequence,file = paste("RD_SEQS/",header,".fasta",sep=""))
}


align_raw={}
align_seq={}
align_raw <- foreach(i=1:nb_of_init_seq,.combine = 'rbind',.inorder = FALSE) %dopar% {
  name_seq = paste("Random_Sequence_",i,".fasta",sep = "")
  align_seq = system(paste("./SCRIPTS/FORSA/forsa_global RD_SEQS/",name_seq ," OUT_DSSP_backup/2xiw_A.ds
  ## We need to delete the file to avoid overfilling
  system(paste("rm ","RD_SEQS/",name_seq,sep=""),intern=T)
  align_seq[1] = paste("Random_Sequence_",i,sep="")
  align_raw[i] = t(align_seq)
}
align_raw[,4]=str_split_i(align_raw[,4],":",5)
align_df<-data.frame(query  = align_raw[,2],
                     target = align_raw[,3],
                     z_score = as.double(align_raw[,4]))
row.names(align_df)<-align_raw[,1]

par(mfrow = c(1,1)) ## Réinitilisation de l'affichage
hist(as.numeric(unlist(align_df$z_score)),breaks = 100,
     xlim=c(-3, 11),
     main = "Valeur de Z-SCORE pour des séquences aléatoires\n selon une loi uniforme par rapport à 2xi
```

```
     ylab = "Nombre de séquences", xlab = "Z-Score",
     col = rgb(0.83,0,1,0.7), border = F)
abline(v = c(3.085,8.004 ), col = "red",lwd=2)
```




```
top_hit<-head(align_df[order(-align_df$z_score),,drop = F],nb_of_bests)
print(head(top_hit))
```

La génération a été faite et évaluer à l'aide de **Forsa** pour 10000 séquences aléatoire pondérée en fonction de l'apparition des résidues dans des séquences de protéines Sac7d. La répartition des scores suit une parabole asymétrique, similaire à une loie de poisson. Les scores restent en majorité en dessous du seuil du plus faible Z-score pour un séquence de référence.

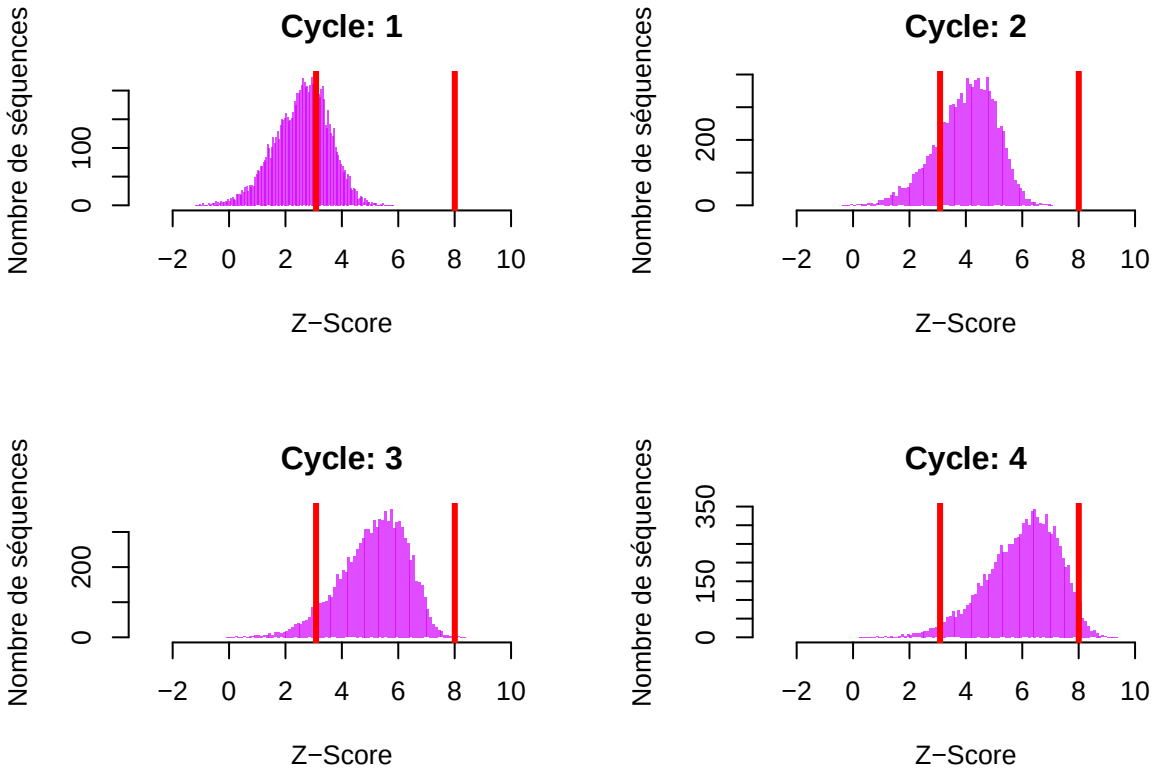
1.3. Création des générations suivantes par mutation des meilleures

```
### III. Mutation cycles ----
par(mfrow = c(2,2))
for(cycles in 1:4){
  print(cycles)
  top_hit[,1]<-gsub('-', '', top_hit[,1])
  new_sequences = {}
  new_sequences_tmp = {}
  file_name_l <<- {}
  cpt = 1
  foreach(i=1:nb_of_bests,.combine = 'c') %dopar% {
    new_sequences_tmp=system(paste("./mutate_seq",nb_of_mutants,top_hit[i,1]),intern = T)
    #new_sequences_tmp = mutate_seq(top_hit[i,1],nb_of_mutants = nb_of_mutants)
```

```
    for (j in 1:(nb_of_mutants*2)){
      if(j %% 2 != 0){
        header <- new_sequences_tmp[j]
      } else {
        file_name = paste("Sequence_mutated_0.2_from",i,"_child_",j/2,sep="")
        sequence <- paste(header,"\n",new_sequences_tmp[j],sep="")
        #new_sequences[i] = sequence
        write(sequence,file = paste("MUT_SEQ/",file_name,".fasta",sep=""))
      }
    }
  }

  align_raw_mut={}
  align_seq_mut={}
  file_names = list.files("MUT_SEQ/")
  align_raw_mut <- foreach(i=1:(nb_of_mutants*nb_of_bests),.combine = 'rbind',.inorder = FALSE) %dopar%
    align_seq_mut = system(paste("./SCRIPTS/FORSA/forsa_global MUT_SEQ/",file_names[i] ," OUT_DSSP_backu
    system(paste("rm ","MUT_SEQ/",file_names[i],sep=""),intern=T)
    align_seq_mut[1] = paste("Random_Sequence_",i,sep="")
    align_raw_mut[i] = t(align_seq_mut)
  }
  align_raw_mut[,4]=str_split_i(align_raw_mut[,4],":",5)
  align_df_mut<-data.frame(query  = align_raw_mut[,2],
                           target = align_raw_mut[,3],
                           z_score = as.double(align_raw_mut[,4]))
  row.names(align_df_mut)<-align_raw_mut[,1]

  hist(as.numeric(unlist(align_df_mut$z_score)),breaks = 100,
       xlim=c(-3, 11),
       main = paste("\nCycle:",cycles),
       ylab = "Nombre de séquences", xlab = "Z-Score",
       col = rgb(0.83,0,1,0.7), border = F)
  abline(v = c(3.085,8.004 ), col = "red", lwd = 3)
  top_hit <- head(align_df_mut[order(-align_df_mut$z_score),,drop = F],nb_of_bests)
  print(head(top_hit))
  write.csv(head(align_df_mut[order(-align_df_mut$z_score),,drop = F],100),file = paste("Top_HITS_",cycl
}
```




```
#mtext("Valeur de Z-SCORE pour des séquences aléatoires\n selon une loi uniforme par rapport à 2xiw",
#      side = 3, line = -21, outer = T)
```

Dès le premier cycle, il est possible d'observer une nette amélioration des Z-scores, avec le pic de la répartition des scores sur le permier seuil. Au bout de 4 cycles, une grande partie des séquences permettent d'obtenir un Z-score supérieur à 8.

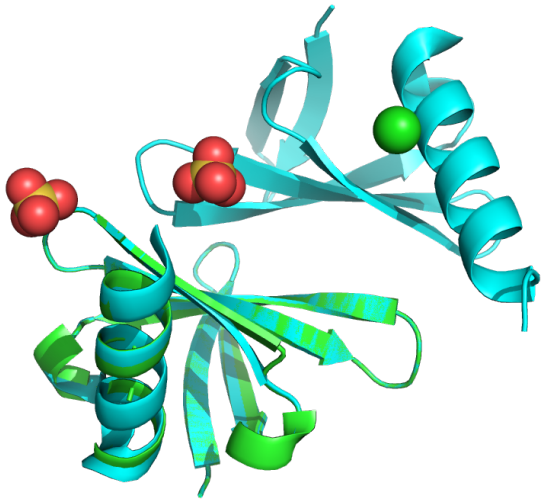
III. Conclusion et discussion

Lors de différents tests, il s'est avéré que de garder seulement le top 10 des meilleures z-score permettait d'avoir une convergence des résultats plus rapide vers des scores élevés. Le pool de séquences mutées passant à 10000 séquences au lieu des 100000 proposées. Cela permet également de diminuer grandement le temps de calcul malgré la parallélisation.

Avec un nombre de générations faible, il y a rapidement des séquences obtenant des z-scores élevé qui apparaissent. Des tests sur un nombre plus faible de séquences (1000) ont révélés des résultats similaires, mais avec une variabilité plus importante entre les essais. Les calculs ont été pu être réalisés dans un temps raisonnable (<5 min) sur un ordinateur portable grâce à la parrallelisation avec l'utilisation de 8 *threads*.

Ainsi avec 4 cycles de génération seulement, on obtient des z-score supérieurs à 9, pour un maximum à 9.735 avec la séquence "AQLPYEFYPGWPMLVDPEGSEKIIPDGDESIPIFSMDGEKITHIVSEKGYVPEWWMT-LADPFSE". Il serait possible ensuite de réaliser de la prédiction de structure par homologie, ayant une structure de référence. Cependant la protéine est très petite et regarder la différence entre les deux modèles sans tenir compte de la flexibilité de la protéine pourrais biaiser la comparaison. En utilisant le l'outil ROBETTA (https://robetta.bakerlab.org/), une modélisation par homologie à 2xiw a été réalisée et obtenant un modèle avec un RMSD de 0.000 avec la référence.

Pour conclure, cette approche utilisant les protéines blocs est efficace pour développer des protéines similaires à une référence mais avec propriété possiblement différentes.



3.2. Liste des séquences après n-cycles

query	z_score
AQLPYEFYPGWPMLVDPEGSEKIIPDGDESIPIFSMDGEKITHIVSEKGYVPEWWMDDPFSE	11.35
CTLSYEFYYGWPMLVDPEGSCKIIPDGDESIPISSYDGEKIPHIVSEKGYSPDYWQTDPFSE	9.72
AQLMYYFPPGWPTVVDPEHSCKIIPDGDESIPISSHDGEYIPHIVSEKGTSPEWFMTDPFSE	9.70
ASLMTETYPGQDMLYDPEGDYKILPPGDEPICISSMDGMKIQFIQSEKGYDPEKWHDDADHRVA	9.68
AHVMYSLYPGYDMLVDPEGLCKYVPDGDEAIPISSMDGEKIPYTQSEDGYPPEKWVDTDDPFSE	9.67
AQLPEEFYPGWPYLVDPEGSCKIIPDGDESIPISSLDGERIPHITCEKGYPPEWHMTDADPESK	9.63
AHAAYCFCPGWQYLVDPEGRCKIFYDGDEEIFISCMEGEFIFTTQSEEGFSPEIWMDDPFSP	9.60
AKLMYEFAPGWDMLLDPSGTCQIIPDGDESIPIISMDGEKIPHTQSPKGYDPEKWMDDADPSSE	9.59
AKLMVETYPGQDMLNDPEGWYKIIVPGDEFIPIMSMDGEKIQFTVSEKGYDPEKDVDTLADHFSE	9.57
ADLPYEFYPGWPMLVDPEASCIIIEDGDEYIPIFSRDGEKIPHIVPEKGYNPEWWETDDPFVE	9.53
ASLMDETYPGQDMLMDPEGLYKIIPPGDELIPISIMDGRKIIFPQSEKGYYPEKWKDDADHFSE	9.51
AHWIYFFYPGWDCVVDPQGLCKYVPDGDEPIPISSMDGEKCPHWQSEKGFPPEKWVDTDLADKFSE	9.51
AKAMYEFYPGWQMLVDPEGWYKIFYDGDEEIPISCMDGEFIPTTQSEEGYNPEWWADDLADPFSG	9.48
AKLQYEFYPGKWQLVDPEGHEKIIPDGDERIIISMMDGEKIFHTFSEKGYGPSKVLDDTVPFSE	9.46
CHEMYNFYPGRVMLHDPEGLCKAVPDGDEQIPISSMDGEKIPWTQSEDGYPPEKEVDTLADPFSE	9.45
AHLMYEFYQGWDMLLDPNGRCTIICDGDEVTPISLSDGTKIPHYLSEKGYPPEKWVDTLVDPFSE	9.44
AAIMYMFYEGWDMLVDPERLCKIIPDGDNPIMIMSMDGEMIFRYVSEKGYVPEWWVDDLRDVFSE	9.43
ASLMLHTYNGQDMLNDPEGCYKIIPPGDFCIPISWMDGFPIQFTQSEKGEDPEKWKDDLADFRSE	9.41
AMEMYNFYPGWDMLVDPEGLCKYVPDGDEPIAISSMDGEFIPHTQSPKGYPPRKWVDDLADPFSE	9.39
AKIMYELYPGWDTLVDPERLCKIIPDGRNCIMIMSMDGEMIVGYVSEKGYPPEWWVDDLRDPFSE	9.38
ASEMYNFYPGWMFLVSPEGLCWAVYDGDEEIPILSLDGEKITHTHSEQGTPPEKWVDDLKDPFST	9.37
AHEMYNFYPGWDPLVDPEGLCKYVPDGDEEIPISERDGSKVPHTMSEKGYPPEKWVDTLADWFFC	9.37
AELMYEFAPGWDMLVHPSGKVKIIPDGDEYIPISSMDGEKIPHTWSPKGICPEKWMDDADPWSE	9.37
AKGDDEFYPGKWQLVDPEGREKIIPDGDERIIISMMDGEKIPHTMSTKGYGPDKVADDLMVMFSE	9.36
ANEKYFFYPGWDMLVDPEGRCWMIYLGDEEIHIMRPDGEKITNYYNENGYDPEDWVDDCAQPFTE	9.35
ANRKYFFYPGWDMLVDPEGRCWMICNGDEEIHISFMDGYKITHLYNEKGIDPEKWVDDLHVPFYE	9.34
AQLPYEFYPGWPHLVDPEGSCKIIPDGDVSIPVSVVDGEKIPHIVSEKGYVPEWCMDDADRFSE	9.33

query	z_score
AHEMYNFYPGWHMLVDPEGLCWPVPDGDEEILISSMDGEKIPHPQSEKGYPPEKWMDSLNDPFSE	9.323
AQRPYEFYPGWPLLVDPEGSAKIIPDGDESIPIMSMDGEKIPHIVSAKGKVPEYWMTLDIVPFSE	9.311
AQRPYHIYPGWPYLVDHEGSCKIIPDGDESIPIVIKDGEKIPHIVSEKGYVPEWIMTLADTPFSE	9.297
YSLMTFTYPGGMMLTDPEGMYLIVPPGDEEIPISSMDGEKIQFIQSEKGYDPEKDCDLAIDSFMG	9.294
AQLKYEFYPGYPMLVDPEGSQKIYPDGDETIPICSDDGLKINHIVSEHGYVPEWWMTLCCFPFSE	9.279
NKRQYEFMPGKWQLVDPEGREKIIPDGDSRIIISCMDGEKIPFTDSEKGYGPQKVADLAVPFSE	9.273
AHLMYYFCFGWDMYVDPEGRVKIIPDGDEQTPIVLWDGEKIPHPLSEKGYCPEKLMDLYDIFSS	9.269
ASLMTETKPGQDMLLDPEGDYKIIPPGDEEIPISSFDGEEIQFLQSEKGIDPEKWKDLHSCFCE	9.268
ANLKYFFYPGWQMLVDPEGCSWMIYLGDEEYHIPTMDGEIVTHTYNEKGEPPEAQLRLADRFYS	9.246
AQLPYEFYPGWPMLVDPEGSCKIIPDGDESIPISSMDGEKIPHIVSEKGYVPEWWMTLADPFSE	9.244
AQWPYEADPGCPYLVDPEGSNNIIMDGGESIPIIQMDGEKIPHIVSEKGYVPEDLMTNADIFSE	9.239
AKIMYEFVPGFDMLVDPERLQCIIPDGDNCIMISSMNGESIMRYQSNKGYNPEWWMDLLEPFSE	9.237
AQLPYESYPGWPLLVDPEGSCKIIPDGDESIPISSMDGEKIPIIVSEMGYVPEWWMTLADHECF	9.233
AKIMMEFVPGWDGLVCPEGLCLFIPDGDNCIMIMSMNGEMIYRYQSWKGYPPEKWMDLLRDPTSE	9.222
ADTPYEFYPGWPMLVDPEGSKKIIPDGDESIPWCSGDGEKIPHIVCELDYDPEWWMTLACHFSE	9.209
AQLWYEFYPGWPMLVDPEGSCKAEWDGDEKIPISSMDGEKIPCILSEKGSPPEWWMDLEADHFSE	9.208
AQFPYEFSPGHIMHVGPAGACKIIPDGDESIPISSMDGEKICHIVSEKGYPPEWWYTFADIPFSP	9.201
AKLMYNFAPGWDMLVDPSGQCKIIIDGDERILISSMDGEKIPHTASPKGYHPEKWMDLADPFSE	9.198
AKWQYESWDGKWQLVDPEGREKIIPDGDEVIPISMMDGEKIPWTASEKGYGPMKVADLAVPHSE	9.197
AKLQYEFYPGKWQLVDPEGREKIWPDGDKRIIISMMDGLKIPHTKSEKGYHPQKVDDLAVPFWF	9.195
AGEPYNFYPGWDMLIDPEGLCKYVPDGDEYIPPSSMDGEKIPRTQSESGYPPEKWMKLHAPFSE	9.191
AKLMTETYPGQFMLNDPEGDPKIIQPGDEEIPISSMDGEKIKFTWSEKGYDPEKWKDLADHFWE	9.179
AKMMREQYRGWDMLVDPEALCKVIPDGRWCIMIMSMDGECIPFYYSIKGYDPEWWMTLRDIFSR	9.178
AHLMYEFYFGWDMLVDPEGRCKIIPDGDEEKEISLPDGEKIPHYMSEKGYSPEDWMDLVDPFSA	9.177
VKLQYQTYPGKWQLVDPEGRTKIIPDGDERIIICMCDGEQIPFTDSETGYGPQKVAGLAVPFSE	9.174
AKAIWEFYPGWQMLVDPEGRCKIFYDGDEVIPITCQDGEFIPTEQSEYGYTPDKWMDLADPFSG	9.165
AHLMYEFYFGWDVLVDPRGRTKIIPDGDSWTPIILRDGEKIPHYGSEHGYCPEKWVDLCDPFSE	9.163
AHEMYNFYPGWDMLVDPEGLQKYVPDGDEEHPWFSMDGQKIPHQQSEKGYPPEKWIDLADFFYE	9.162
ANLKYCRYTGWDMLVDPEGRCWMIYNGDEQIHIIQMDGEKITHTYNKKGFDPEKWIWDLADPFPE	9.161
MKIMEWFYPGWIMLVDPERLCKIIPPGDNPIEIMSMDGEMIPRYQSEKGVVPEWWMDLFMPSQE	9.158
AQLPYEFLPGWPMNVDPEGNCKIIPDGDESELISSMKGSKIPHIVSPKGYDPKWIMTLADPFQE	9.152
WNLLYFFYPGWDMLVDPEGRCMMIYDGREEIHIMQMDGEKITHIYNEKGYDPEKWADLADPFYE	9.146
AQWPYEAYPGWPMLVDPEGSCKTIPDGDESIPWSSMDGEKIPHWVSEDGYVPEWWMYYLADPFSE	9.139
AKIMYKFYPGWDMLVDPEQLTVIFLDGDNCIMIMSMDGQMIPRYTSEKGYVPESWMDVTDPFSE	9.139
ASFMTETYPGQDMLADPEGDYKEIPPGDGLIPISSMNGEWIIFTQSEKGYDPEKWKDLADVMSE	9.138
AQLPYERYPGWPMLVDPEGSCKVVPDGDESIPISSMDGEKIPHIMSEKGYVPEWWMDLADPFST	9.133
AKLQYEFYPGKWQFVDPDGREKIVPDGFERIIFFMMDGEKIPHTPCEEGYHPQRTADLAVHGSE	9.133
AHEMLHKQPGWAMLPDPEGLCKYVKDGDEEIVISSYDGESIPHTQSEKGYPPEQWHDLADNFSG	9.132
AKLQYPFYPGKWTVVDPKDREKIIPDGDERIIISMMDGEKIPHSDSPKGYGPQKVADLAVPFSE	9.129
AQLPYEFYPGWPYLVDPEGSCAIIPDGDWSAVVSSMDGEKPPHIVSEKGYVPEWWMTNADPFSW	9.112
IHEMYNFYPGWDMLVCPEGLCKYVPDGDEEIMISIMDGEWIPATQSEAGYPPEKWMDLADPFSG	9.112
AHLYYEFYFGWDWLVDPEGRCKIVPDGDNETPIWLWDGQPIPHYLSEKGNCPEKWMDLVDPFSE	9.108
AKLNWEFYPGKWQLVDPEGREKSIPWGDERIIIVMMDGETIPITDDEKGYGPQKVADLAVPFSE	9.105
AHLMYEFYFGWVMLVDPWGRKKIIPDGDEEWPISEWDGEPIPVYLCEKGQCPEKWMDLVQPFSD	9.104
ARLPYEFYPGLPELVDPEGRCKIIPDGDISIPISSMDGEWIPHIVSEKGYAPEWWMTLADPFSE	9.095
AKLMYEMAPGWDMLVDPSGKCKIIPDGDEKICIVSMDGEKIPHTQSPKCYHPEKWYDLADCFSG	9.095
LKIMYEFTPGWDMVVDPERLGAIIPDGDNCIMIMSMDGEMIPRYQSEKGYCPEWWMDLSDPLSE	9.092
AKIFYIFYPGWRMLVDPERLCKIIPDGENIIMIMSMDGEMIPRYYNAKGYYPENWMDKRHPFSW	9.088
AHENDNFYPGEIGLVDPEGLCKYVPDGDECIPISSSDGEKIPHTQSEKGYPPEKWMFLADPFQE	9.087
AVLMYEFYPGWDMLKDPEGRQKIIPDGDEDIPICSPKGEKIPHTQSEKGYPPEHWCDLAGMNSA	9.081
AGLQYEHHPGKWQLVNPEGREKIIPDGDERIIISMMDGEKVPHTPSEKGYGPQKVTDNAVHFSE	9.076
AEIMFYFYPGWDMLVDPERLCYIIPDGDNCIMIMSMDGECIPFFQSEKGYVPEWWMDLRKPFNE	9.075

query	z_score
AALPPAEYPGQPMLVDAEGSDKIIPEGDESIPISSMDGEDIIHIVSEKGYVPEWWMTQLPDPFSE	9.47
AHLMTETYPGQDMLNDPEGDYYIIPPGDFEIPISSIDGEKIQFTQSEKGLDPEKWKPQLDHFSQ	9.06
WELQYEFYPGKWQLVDPEGREHHIWDGDERIIISMMDGEKIPHTMSEKGYKPQKQAMSYAVPFSN	9.05
AKLQYEFQPGKWQLLDPEGREKAIPDGDEWIIISYMDGEVIPHTKSEKGYGPQKVAPSLVPFSE	9.03
AHECYNFYPGWDMLVDPEGLCKPVPSGDEFIPESIMNGEKIPHTQSEKGYPPEKWVPSGADPFSE	9.03
AQLPYEFMPGWPMLVDPEGSCKIIPDGDGSIPISSMDGEKIPRIVSEKGYVPEWRMTLPSADPFSE	9.03
RKAMYEFYPGWQMLVDPEGRRKIFNDGDEPIPISCMDGEFIPTTQSEEGYVPEKWVQPSRADPFSG	9.02
APLPYEFQPGWCYLVDPSGKEKPIPDGEERCPINSMDGNKIPHTQSPKGYNPEKLMPLSADPFSE	9.01
AQLFYEFYPGWPMLVDPAGSQKIIPDGDHSIPISSMDGEKIPVIVSEKGYVSEWWMTLAAIPFSE	9.01
AKAIYEFYPGWQMCVDPEGRCKIFMQGDEEIKISCMDGEFIMTTFSEEGYVPEKWVPALAALASG	9.01
AHLMYEFAPGWDVGVDPSGRCWIIPDGDARVPISSMDGAHICWTQSPKGYNPEKWVPLFPDPFSE	9.01
AQCPYEFYPGTPMLVDPESSCRIIDDGDESIPISSYDGEPVPHIVSEKGYQPEWWMTLPLADPFSE	9.01
AKLQHHFYPGKWGLVAPEGRKKIIPDGDERQIISMMDGEKIFHATSEKGYDPQKVAPLAVERSE	9.00
FNLKYFFFKGWMMLVDPEGRCWMIYLGDEMHIRSQMDGAKITCTYNEKGYDPEKWVPLDLAGTFYE	8.99
AKLQYEFQPGKWHLVDPEGREKIIPDGDERIIISMMDGEKIPMTDSEKGYGPQKDAPLAVPFSE	8.98
ACLKTETYPGQDMLNDPEGQYKIVPPGDQEIPISSMDGEKIQFTQSEKGYDPEKWVPLSAWHFSE	8.98
AQLTYEFYPGWPKLVDPEGRMKIIPDGDESIPYSSMDGEKIPHIVSEKGMVPEWWVPLPDTFSE	8.97
QDQTHEFKPGKWQCFDPELREKIIPDGEERIIIPMMDGEKPIHVMSEKDYCPQKVAPLAVPFSE	8.97
ANGMYNFYPGWDMLVDPEGLTKYVPDGDEEIPYSSMDGEKIAHTVSEKGSTPEKWVPLHLEDPQSE	8.96
AHEMQNFYPGWDMLVDPEGLCMYVPDGDEIQPISSMNGEKIRHDMSEKGFPPEKWVPLDLWDPFSQ	8.96
AHEMYNFYPGWDMLVDPEGLCKRVPDGDEMIPISSMDGEKIPHTQTEKGYPPEKWVPLDLADPFSE	8.96