ΚΟΥΣΑ ΑΝΑΣΤΑΣΙΑ

Διπλωματική Εργασία

Μελέτη αλληλουχικά όμοιων, δομικά ανόμοιων πρωτεϊνών ή "Πώς να παρερμηνεύσετε το πείραμα του Anfinsen"

Επιβλέπων καθηγητής: Νικόλαος Μ. Γλυκός

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΤΜΗΜΑ ΜΟΡΙΑΚΗΣ ΒΙΟΛΟΓΙΑΣ & ΓΕΝΕΤΙΚΗΣ
2011

Περίληψη

Όταν ο Anfinsen πραγματοποιούσε τα πειράματά του με τη ριβονουκλεάση, δε μπορούσε να γνωρίζει τότε ότι τα αποτελέσματά του θα επηρέαζαν το επιστημονικό κοινό σε τόσο σημαντικό βαθμό, ώστε ο κόσμος να αρχίσει να αναρωτιέται για το αν η πρωτοταγής δομή μιας πρωτεΐνης καθορίζει σημαντικά την τριτοταγή της δομή. Η αλήθεια είναι ότι η φυσική επιλογή φαίνεται να αποτελεί το κύριο αίτιο για τη διαιώνιση αυτής της υπόθεσης, καθώς είναι αρκετά πιθανό παρόμοιες πρωτοταγείς δομές να υιοθετούν και παρόμοιες στερεοδιατάξεις, χάρη λειτουργικότητας και κατ' επέκταση επιβίωσης στον πλανήτη. Και αυτή είναι ακριβώς η εξήγηση στο γιατί οι μοριακοί βιολόγοι δεν καταλήγουν να αποτυγχάνουν παταγωδώς όταν από την αλληλουχία και μόνο μιας πρωτεΐνης επιλέγουν να θεωρούν δεδομένη τη λειτουργία της και στη συνέχεια να αποφαίνονται σχετικά με τη δομή της. Στα πλαίσια λοιπόν αυτής της διπλωματικής εργασίας αναλύουμε τη δική μας προοπτική όσον αφορά την αναδίπλωση των πρωτεϊνών και συζητάμε για το αν η πρωτοταγής δομή επηρεάζει την διαμόρφωση μιας πρωτεΐνης στο χώρο. Επιπλέον, προσπαθούμε να βγάλουμε τα δικά μας συμπεράσματα συγκρίνοντας τις δομές πρωτεϊνών που εμφανίζουν μεγάλη ομοιότητα στην αλληλουχία τους και διαφέρουν δομικά. Για τη σύγκριση αυτή επιλέγουμε να αντιπαραβάλουμε τους CATH κωδικούς που αντιστοιχούν στις πρωτεΐνες αυτές. Κατά τη διάρκεια της έρευνας όμως διαπιστώνουμε ότι η ιδέα αυτή δεν απαντάει το ερώτημα που έχουμε θέσει και γι αυτό στρέφουμε τις βλέψεις μας στον υπολογισμό του RMSD για να κρίνουμε δομική ανομοιότητα. Τέλος, κάνουμε κάποιες υποθέσεις σχετικά με το αν συγκεκριμένες αμινοξικές αντικαταστάσεις επηρεάζουν το μοτίβο αναδίπλωσης μιας πρωτεΐνης.

Abstract

When Anfinsen conducted his experiments with the ribonuclease, he could not have known then that he would influence the scientific thought in such a dramatic way, in which people would assume that a protein's structure may be substantially determined from the protein's sequence. In fact, natural selection constitutes the major reason for this belief's perpetuation, because it seems possible that proteins with high sequence identity tend to adapt same tertiary structures for the shake of functionality and survival. And that is exactly the explanation why molecular biologists have not ended up failing and failing again during their experiments, when from a protein's sequence they suppose that they are aware of its function. And they may also assume that if function is preserved then structures will be similar. So, in the terms of this final year thesis we try to analyze our own perspective in the matter that is called protein folding and we discuss if and how primary structure can influence the tertiary one and under what circumstances may this hypothesis be valid. Furthermore, we export our own conclusions by comparing the structures of high identical protein sequences. For this purpose, we initially compare these proteins CATH codes. During that procedure we realize that our thinking of using CATH codes for structure comparison does not answer our question, so we decide to calculate RMSD in order to evaluate structure dissimilarities. Finally, we discuss whether specific amino acid changes have a significant effect in the fold of a protein.

ΕΥΧΑΡΙΣΤΙΕΣ

Μου δίνεται η ευκαιρία με την περάτωση αυτής της διπλωματικής εργασίας να σημειώσω ότι τελικά υπάρχουν άπειροι τρόποι μία πτυχιακή εργασία να πάει στραβά χωρίς την κατάλληλη καθοδήγηση. Γι’ αυτόν ακριβώς το λόγο, θα πρέπει να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου κ. Νικόλαο Γλυκό που στήριξε την προσπάθεια μου σε δύσκολες στιγμές, με καθοδήγησε και μου αφιέρωσε πολύ από τον πολύτιμο χρόνο του για να φτάσουμε ως εδώ.

Θα πρέπει παράλληλα να ευχαριστήσω τους γονείς μου καθώς μου έδωσαν τα απαραίτητα εφόδια για να ολοκληρώσω τις σπουδές μου και με στήριξαν με το δικό τους τρόπο.

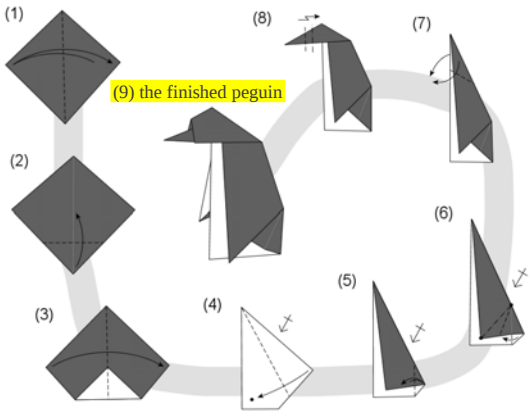
Τέλος, θα ήταν άδικο να μην ευχαριστήσω κάποιους φίλους και συμφοιτητές μου οι οποίοι με βοήθησαν και με στήριξαν σε δύσκολες στιγμές και συνέβαλαν στο να περάσουν πιο ευχάριστα τα τέσσερα αυτά χρόνια της φοίτησής μου.

ΠΕΡΙΕΧΟΜΕΝΑ

Κεφάλαιο 1

Εισαγωγή

-Υπάρχουν άραγε πρωτεΐνες που παρουσιάζουν μεγάλη ομοιότητα στην αλληλουχία τους αλλά διαφέρουν δομικά;

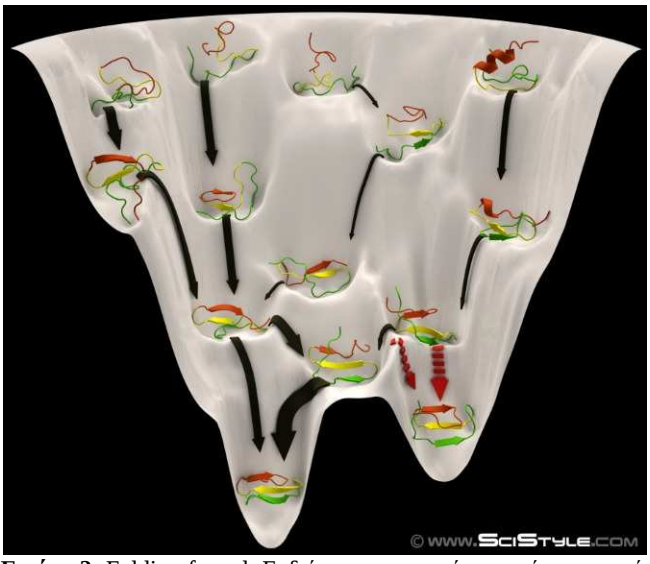


Εικόνα 1: Αναπαράγεται άνευ αδείας, *http://origami.island-three.net/penguin.html*

Θα μπορούσε να παραλληλίσει κανείς την αναδίπλωση των πρωτεϊνών με το ιαπωνέζικο παιχνίδι του Origami*. Όλα ξεκινούν από ένα κομμάτι χαρτί, όπως αντίστοιχα και από μια αποδιατεταγμένη πολυπεπτιδική αλυσίδα. Ακολουθώντας τις κατάλληλες οδηγίες το χαρτί θα περάσει από τις απαραίτητες αναδιπλώσεις και θα σχηματίσει μια συγκεκριμένη φιγούρα. Ένα τσάκισμα του χαρτιού στη μέση, λίγες αναδιπλώσεις στο ένα άκρο, μετά η ίδια διαδικασία στην αντίθετη πλευρά, λίγη υπομονή και … *the finished penguin*!

Μια σημαντική διαφορά μεταξύ της αναδίπλωσης των πρωτεϊνών και του Origami έγκειται στο ότι μια πολυπεπτιδική αλυσίδα μπορεί να ακολουθήσει πολλά πιθανά μονοπάτια (μέσω ενεργειακών ενδιαμέσων) για να καταλήξει σε μία σταθερή κατάσταση. Αυτή είναι η ιδέα του folding funnel. Μια πολυπεπτιδική αλυσίδα λοιπόν ίσως περνάει από πολλά ενδιάμεσα, σταδιακά χαμηλότερης ενέργειας, ακολουθώντας ποικίλες πορείες για να φτάσει στο ίδιο ενεργειακό ελάχιστο. Αντίθετα στο Origami ακολουθείται μία συγκεκριμένη πορεία για τη δημιουργία κάποιας φιγούρας. Αυτό όμως δε μας εμποδίζει να συγκρίνουμε αυτές τις δύο διαδικασίες δημιουργώντας στον αναγνώστη ένα αίσθημα ασφάλειας καθώς μπορεί να αντιληφθεί μακροσκοπικά τη διαδικασία της αναδίπλωσης.

* Το Origami αποτελεί μία τέχνη αναδίπλωσης του χαρτιού. Η λέξη στα ιαπωνέζικα κυριολεκτικά σημαίνει αναδιπλώνω (oru) και χαρτί (kami).



Εικόνα 2: Folding funnel: Ενδιάμεσες καταστάσεις ενέργειας από τις οποίες μπορεί να διέλθει μια πολυπεπτιδική αλυσίδα μέχρι να φτάσει σε κάποια με την ελάχιστη δυνατή ενέργεια (αναπαράγεται άνευ αδείας από http://www.SciStyle.com).

Be different: conform! Ορισμένες φορές διαθέτουμε δύο πανομοιότυπα κομμάτια χαρτιού τα οποία μπορούν να σχηματίσουν την ίδια φιγούρα. Αν επιλέξουμε να τα αναδιπλώσουμε με διαφορετικό τρόπο τότε προκύπτουν ποικίλα σχήματα. Μία 'λάθος' κίνηση αρκεί πολλές φορές για να μετατρέψει το καραβάκι που πασχίζαμε να κατασκευάσουμε σε αεροπλάνο, αφήνοντας πολλά ερωτηματικά στο πρόσωπό μας όπως και ένα αίσθημα απογοήτευσης. Πραγματοποιώντας τώρα μια μετάβαση στον κόσμο των πρωτεϊνών, μπορεί να αναρωτηθεί κανείς αν και εδώ συμβαίνει κάτι ανάλογο. *-Είναι οι ενδιάμεσες αναδιπλώσεις υπεύθυνες για την τριτοταγή δομή που θα υιοθετήσει μια πρωτεΐνη; -Η μήπως η πρωτοταγής δομή καθορίζει απόλυτα την στερεοδιαμόρφωσή της;* Και έτσι επιστρέφουμε στο αρχικό μας ερώτημα:

-Υπάρχουν άραγε πρωτεΐνες που παρουσιάζουν μεγάλη ομοιότητα στην αλληλουχία τους αλλά διαφέρουν δομικά;

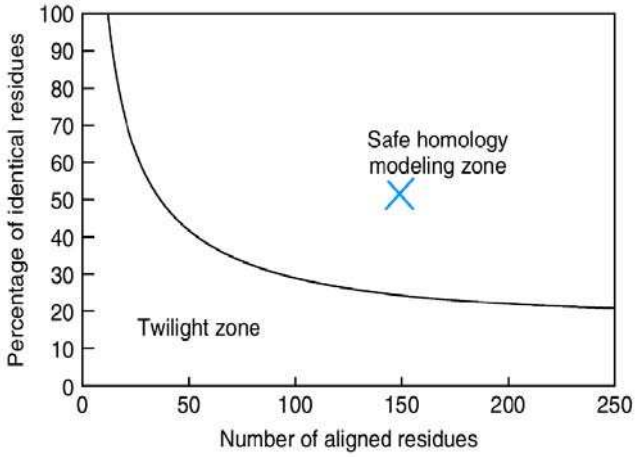
'If you would understand
anything, observe its beginning
and its development.'

Aristotle

Στην προσπάθεια μας λοιπόν να απαντήσουμε το ερώτημα αυτό, αρχικά θα παρακολουθήσουμε μια αναδρομή στον κόσμο των πρωτεϊνών: Από το 1972, όταν ο Anfinsen[1] πραγματοποίησε τα πειράματά του και άρχισε να διερωτάται για το αν η δομή μιας πρωτεΐνης καθορίζεται μοναδικά από την αμινοξική της ακολουθία, το επιστημονικό κοινό μπορεί ασυνείδητα να “παρερμηνεύσει” την υπόθεση αυτή και να θεωρήσει πως δύο παρόμοιες πρωτεϊνικές αλληλουχίες πιθανόν να υιοθετήσουν και σχεδόν ίδιες δομές. Ίσως έτσι “επισκιάστηκε” κατά κάτι μία άλλη πλευρά της έρευνάς του Anfinsen. Εκτός των άλλων, τα πειράματά του απέδειξαν ότι η διαδικασία αναδίπλωσης μιας πρωτεΐνης είναι αυθόρμητη, δηλαδή ότι η θερμοδυναμικά σταθερότερη κατάσταση είναι αυτή της αναδιπλωμένης πρωτεΐνης. Με άλλα λόγια, η φυσική δομή της πρωτεΐνης αντιστοιχεί στο ολικό ή (σε πλησίον του ολικού) ελάχιστο της ελεύθερης ενέργειας του συστήματος.

Κατά τη διάρκεια της εξέλιξης οι δομές φαίνεται να είναι σχετικά σταθερές και αλλαγές πραγματοποιούνται πολύ αργά σε σχετιζόμενες αλληλουχίες, με αποτέλεσμα παρόμοιες αλληλουχίες να υιοθετούν και ίδιες δομές. Αυτή η σχέση είχε αρχικά αναγνωριστεί από τους Chothia και Lesk (1986) [2] και ποσοτικοποιήθηκε αργότερα από τους Sander και Schneider (1991) [3]. Εξαιτίας μάλιστα της τεράστιας ανάπτυξης της Protein Data Bank (PDB), ο Rost (1999) [4] κατάφερε να εξαγάγει ένα ακριβές όριο για αυτόν τον κανόνα, που φαίνεται στην Εικόνα 3. Όσο το μήκος και το ποσοστό όμοιων καταλοίπων "πέφτουν" μέσα στην περιοχή που έχει σημανθεί ως "ασφαλής" (safe), οι δυο αλληλουχίες είναι πιθανό ότι θα υιοθετήσουν μια παρόμοια δομή.

Έτσι, λοιπόν, θεωρητικά αν είχαμε δύο παρόμοιες αλληλουχίες και διαθέταμε τη δομή της μίας, θα ήμασταν ικανοί να προβλέψουμε τη δομή της δεύτερης με μεγάλη ακρίβεια ανάλογη των πειραματικών αποτελεσμάτων. Αυτό αποτελεί και τη γενική ιδέα του homology modeling, σύμφωνα με το οποίο γίνεται in silico σχεδιασμός πρωτεϊνικών δομών όταν υπάρχει κατατεθειμένη η δομή για κάποια πρωτεΐνη που παρουσιάζει μεγάλη ομοιότητα στην αλληλουχία με την πρωτεΐνη της οποίας η αλληλουχία μας είναι γνωστή. Η λογική της διαδικασίας αυτής προφανώς και δεν είναι λάθος σε ένα γενικότερο πλαίσιο, καθώς στην αντίθετη περίπτωση (α) οι κρυσταλλογράφοι θα κρυστάλλωναν συνεχώς διαφορετικές δομές από ότι προέβλεπαν τα θεωρητικά δεδομένα, (β) οι μοριακοί βιολόγοι θα έβγαζαν λανθασμένα συμπεράσματα καθώς θα θεωρούσαν ότι η πρωτεΐνη τους έχει την τάδε λειτουργία ή (γ) σε τελική ανάλυση κάποιος θα είχε παρατηρήσει κάτι!

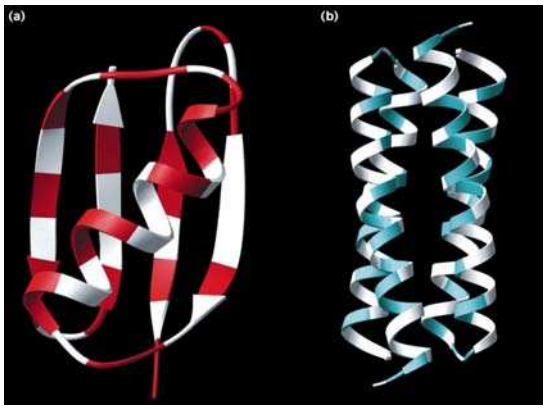


Εικόνα 3: Οι δύο ζώνες των αλληλικών στοιχίσεων. Δύο αλληλουχίες είναι αρκετά πιθανό ότι θα αποκτήσουν την ίδια στερεοδιαμόρφωση αν το μήκος και το ποσοστό ομοιότητας βρίσκονται στην περιοχή που είναι σημασμένη ως "safe". Ένα παράδειγμα δύο αλληλουχιών 150 καταλοίπων που διαθέτουν και 50% όμοια αμινοξέα έχει σημανθεί με X (αναπαράγεται άνευ αδείας από τους Krieger, Nabuurs, και Vriend 2003 [5]).

Αν όμως η θεωρία του Anfinsen αποτελούσε μια πιο γενικευμένη αλήθεια, τότε κανείς δε θα ασχολιόταν με το περιβόητο "folding problem" και η ζωή των κρυσταλλογράφων θα ήταν πολύ πιο εύκολη. Για τις περισσότερες πρωτεΐνες η αλληλουχία των αμινοξέων όντως καθορίζει την τριτοταγή δομή τους. Η σχετική σημασία όμως των "μεμονωμένων" αμινοξέων στον προσδιορισμό της αναδίπλωσης παραμένει θολή. Για να το επισημάνουν αυτό, οι Creamer και Rose(1994) [6] δημιούργησαν το "Paracelsus challenge": ένα βραβείο για τη μετατροπή μιας πρωτεϊνικής δομής σε μια άλλη, διατηρώντας το 50% της αρχικής αλληλουχίας.

Money definetely makes the world go round! Το 1997 οι Dalal, Balasubramanian και Regan [7] απαντούν σε αυτή την πρόκληση και κερδίζουν το χρηματικό έπαθλο, δημοσιεύοντας το “Protein Alchemy: Changing β-sheet into α-helix” όπου και δείχνουν ότι 2 αλληλουχίες με μεγάλη αλληλουχική ομοιότητα γίνεται να υιοθετούν διαφορετικές στερεοδιατάξεις. Οι δύο πρωτεΐνες που επέλεξαν να μετατρέψουν ήταν η B1 επικράτεια της συνδεόμενης με το IgG G πρωτεΐνης του Streptococcus, και η Rop, που είναι ένα ομοδιμερές σύμπλεγμα τεσσάρων ελίκων (Εικόνα 4). Ο μετασχηματισμός έγινε με κατεύθυνση από την B1 επικράτεια προς τη Rop, καθώς ο σχηματισμός των ελίκων και οι αλληλεπιδράσεις μεταξύ τους παραμένουν καλύτερα κατανοητές από το σχηματισμό β-φύλλων.

Janus was born... Τα αποτελέσματα τους δείχνουν ότι ήταν εφικτό να “πειράξουν” το συνδυασμό των τοπικών και ολικών αλληλεπιδράσεων, ώστε να επηρεάσουν δραματικά την ολική στερεοδιαμόρφωση μιας πρωτεΐνης. Επίσης τόνισαν ότι όλα τα κατάλοιπα δε παίζουν εξίσου σημαντικό ρόλο στη δημιουργία ενός συγκεκριμένου μοτίβου αναδίπλωσης.



Εικόνα 4: Αναπαράσταση της αναδίπλωσης των a. B1 επικράτειας b. Rop, c. στοίχισης των B1 (μπλε), Rop (κόκκινο) και Janus. Τα αμινοξέα στη Janus: τα μπλε κατάλοιπα προέρχονται από τη B1, τα κόκκινα από τη Rop και τα πράσινα είναι διατηρημένα και στις δύο. (αναπαράγεται άνευ αδείας από τους Dalal, Balasubramanian και Regan (1997) [7])

*-Άρα η πρωτοταγής δομή **δεν** προσδιορίζει την διαμόρφωση μιας πρωτεΐνης στο χώρο;*

Αν επιστρέψουμε πάλι στο Origami βλέπουμε ότι από ένα φύλλο χαρτιού μπορούν να προκύψουν ποικίλα σχήματα. Τα περισσότερα από αυτά μοιάζουν απλά σαν να τσαλακώσαμε το χαρτί, αλλά μέσα από δοκιμές θα προκύψει κάποιο το οποίο θα αναπαριστά μια πραγματική φιγούρα. Οπότε την επόμενη φορά που θα μας δοθεί ένα κομμάτι χαρτί, ίσως επιλέξουμε να σχεδιάσουμε αυτή τη φιγούρα.

Κάπως έτσι μάλλον λειτουργεί και ο βιολογικός κόσμος. Από τις διάφορες στερεοδιατάξεις που μπορεί να υιοθετήσει μια πολυπεπτιδική αλυσίδα είναι πολύ πιθανό να επιλεγεί εκείνη που είναι θερμοδυναμικά η πιο σταθερή, δηλαδή που θα οδηγεί στη χαμηλότερη δυνατή ελεύθερη ενέργεια του συστήματος. Επειδή μια πρωτεΐνη μπορεί να διαθέτει περισσότερα από ένα ενεργειακά ελάχιστα η φυσική επιλογή ίσως προτιμήσει εκείνο στο οποίο θα μπορεί να αναδιπλωθεί πιο γρήγορα η πρωτεΐνη μας. Αν μία πρωτεΐνη, η οποία είναι απαραίτητο να συντίθεται και να αναδιπλώνεται καθημερινά για να καλύψει τις ανάγκες ενός οργανισμού χρειαζόταν κάποιους μήνες για να αναδιπλωθεί, τότε ο οργανισμός αυτός δε θα μπορούσε να επιβιώσει. Η ύπαρξη ενεργειακών ενδιαμέσων από τα οποία η πρωτεΐνη δε μπορεί να ξεφύγει (kinetic trap) ίσως παγιδεύσουν την πρωτεΐνη σε μία κατάσταση όπου δεν είναι λειτουργική απαγορεύοντας της να αναδιπλωθεί στη φυσική της κατάσταση. Επιπλέον, πολλές φορές μεταλλαγές σε ένα μόνο κατάλοιπο

της πρωτεΐνης μπορούν να οδηγήσουν σε αποδιατάξη της πρωτεΐνης καταστρέφοντας έτσι τη λειτουργία της. Σε αυτές τις περιπτώσεις υπάρχουν πιθανότητες η εξέλιξη να επιλέξει κάποια ομόλογη πρωτεϊνική αλληλουχία που να είναι πιο ανθεκτική σε μεταλλάξεις. Έτσι όταν συμβούν μεταλλαγές σε αυτή την αλληλουχία, αυτές δε θα είναι τόσο ισχυρές ώστε να χάσει η πρωτεΐνη τη λειτουργία της ή να υιοθετήσει κάποια άλλη διάταξη. Με άλλα λόγια, πολυπεπτιδικές αλυσίδες που παρουσιάζουν μεγάλη αλληλουχική ομοιότητα θα αναδιπλωθούν και με πανομοιότυπο τρόπο. Στον φυσικό κόσμο, λοιπόν, προς χάρη της επιβίωσης και της εξοικονόμησης ενέργειας φαίνεται να ισχύει ότι παρόμοιες αλληλουχίες θα επιδείξουν και δομές που να μοιάζουν.

Αυτό που πρέπει να επισημάνουμε και για αυτό που θα πρέπει να είμαστε περισσότερο επιφυλακτικοί είναι για όλα τα μη φυσικά προϊόντα. Καθώς στη συγκεκριμένη περίπτωση η φυσική επιλογή δεν υφίσταται και δεν υπάρχει η πίεση για διατήρηση της δομής που θα εμφανίζει λειτουργικό πλεονέκτημα για την επιβίωση στον πλανήτη, βρισκόμαστε σε πλήρη άγνοια στο έλεος της αναδίπλωσης των πρωτεϊνών!

Οπότε επιστρέφουμε πάλι στην αρχική μας ερώτηση:

-Υπάρχουν πρωτεΐνες που παρουσιάζουν μεγάλη ομοιότητα στην αλληλουχία τους αλλά διαφέρουν δομικά;

ή αν την διατυπώσουμε πιο σωστά:

*-Υπάρχουν **φυσικές** πρωτεΐνες που παρουσιάζουν μεγάλη ομοιότητα στην αλληλουχία τους αλλά παρόλα αυτά διαφέρουν δομικά;*

και αν ναι:

-Διαφέρουν τόσο ώστε να αλλάζει το μοτίβο αναδίπλωσης της δομής τους;

και αν όχι:

-Ισχύουν τα ίδια για τις πρωτεΐνες που δεν υπόκεινται σε φυσική επιλογή και εξέλιξη;

'Look at all the sentences which seem true and question them.'

David Reismann

Η έρευνα μας λοιπόν προσπαθεί να δώσει απαντήσεις στα παραπάνω ερωτήματα. Εφόσον θέλουμε να συγκρίνουμε ομοιότητα σε επίπεδο αλληλουχίας με ανομοιότητα σε επίπεδο δομής, πρωταρχικό βήμα είναι εύρεση όλων των ζευγών πρωτεϊνών με ομοιότητα στην πρωτοταγή τους δομή. Ο αριθμός των πρωτεϊνών που έχουν κατατεθεί στην PDB μέχρι σήμερα ανέρχεται στις 72244 (data obtained from PDB's official site on 9 April 2011). Αυτός ο αριθμός θα αντιστοιχούσε σε n(n-1)/2 στοιχίσεις (για n=72244) δηλαδή 10^9 στοιχίσεις. Με λίγα λόγια, αν τρέχαμε το πρόγραμμα του BLAST για αυτές τις 72244 πρωτεΐνες ως προς την PDB, ακόμη και αν κάθε στοίχιση διαρκούσε 1 δευτερόλεπτο, τότε ο συνολικός χρόνος για το σύνολο των στοιχίσεων θα άγγιζε το 10^9 δευτερόλεπτα και θα απαιτούσε χρόνια υπολογισμών...

Έτσι πολύτιμη βοήθεια στην έρευνά μας αποτέλεσε το άρθρο των Mickey Kosloff και Rachel Kolodny, “Sequence-similar, structure-dissimilar protein pairs in the PDB” [8] που δημοσιεύτηκε στο Proteins το Μάιο του 2008 και στους οποίους από εδώ και πέρα θα αναφερόμαστε με τη συντομογραφία K&K. Σκοπός της έρευνας των K&K λοιπόν ήταν να αναλύσουν περαιτέρω τις περιπτώσεις πρωτεϊνών με εμφανή ομοιότητα στην αλληλουχία και σημαντικές διαφορές στη δομή. Στα πλαίσια λοιπόν της έρευνας τους, είχαν επιλέξει να τρέξουν το πρόγραμμα του blast για τις καταχωρήσεις της PDB (Απρίλιος 2005) και είχαν χωρίσει τα ζεύγη αλυσίδων που προέκυψαν σε οχτώ (αλληλεπικαλυπτώμενες) ομάδες, βασισμένες στην αλληλουχική (≥ 50, 70, 99, και 100) και στη δομική τους ομοιότητα (RMSD μεγαλύτερο από 3, και 6 Å). Αυτή η βάση δεδομένων για αλληλουχικά όμοιες, και δομικά διαφορετικές πρωτεΐνες είναι διαθέσιμη στο διαδίκτυο (http://mt.cs.haifa.ac.il/seqsimstrdiff/seqsimstrdiff_local.htm).

Καθώς δεν ήταν υπολογιστικά δυνατό να συγκεντρώσουμε οι ίδιοι τα ζεύγη πρωτεϊνών (για όλες τις κατατεθειμένες αλληλουχίες της PDB) τα οποία εμφάνιζαν πάνω από 50% αλληλουχική ομοιότητα και ωστόσο διέφεραν στη δομή, αποφασίσαμε να χρησιμοποιήσουμε τα έτοιμα αρχεία των K&K. Το επόμενο βήμα ήταν η σύγκριση των δομικών στοιχίσεων για τα ζεύγη αυτά.

-Πως κοιτάμε λοιπόν δομική ομοιότητα?

Η αλήθεια είναι ότι δεν υπάρχει ένας μοναδικός τρόπος που να υπολογίζει βέλτιστα την ομοιότητα μεταξύ δύο πρωτεϊνικών δομών. Σε γενικότερο πλαίσιο όμως μπορούμε να χωρίσουμε τις στοιχίσεις των δομών σε δύο κατηγορίες: (α) αυτές που στηρίζονται στην ίδια τη δομή των πρωτεϊνών για να υπερθέσουν τις δύο πρωτεΐνες και (β) αυτές που βασίζονται στην στοίχιση των αλληλουχιών για να παράγουν τις ανάλογες υπερθέσεις.

Η στερεοδιαμόρφωση μιας πρωτεΐνης αντιστοιχεί βασικά σε μια ομάδα σημείων. Κάθε σημείο αποτελεί έναν πίνακα 1x3, δηλαδή μια τριάδα συντεταγμένων που καθορίζουν τη σχετική θέση των ατόμων του σημείου στο χώρο. Μία δεύτερη πρωτεϊνική αλυσίδα θα μπορούσε να αντιπροσωπευθεί από μια άλλη ομάδα σημείων με παρόμοιο τρόπο.

Ο αλγόριθμος του Kabsch (1983) [9], που πήρε το όνομα του από τον Wolfgang Kabsch, αποτελεί μια μέθοδο η οποία υπολογίζει το βέλτιστο πίνακα περιστροφής, έτσι ώστε να ελαχιστοποιείται το RMSD (root mean squared deviation) μεταξύ 2 ομάδων σημείων. Πιο συγκεκριμένα, ελαχιστοποιείται το άθροισμα των τετραγώνων των αποστάσεων μεταξύ των ισοδύναμων Cα των δύο πρωτεϊνών ως συνάρτηση της περιστροφής και της μετατόπισης (της μιας πρωτεΐνης πάνω στην άλλη).

Στη δεύτερη κατηγορία ανήκουν οι sequence-based δομικές στοιχίσεις οι οποίες αντιστοιχούν την ομοιότητα των καταλοίπων από επίπεδο αλληλουχίας σε επίπεδο δομής. Αν δηλαδή έχουμε μια στοίχιση, τότε ο αλγόριθμος θα πάρει το 1ο Cα άτομο της μιας πρωτεΐνης και θα το στοιχίσει με το 1ο, το 3ο με το 5ο κ.ο.κ., ανάλογα με την ομοιότητα των αλληλουχιών και αγνοώντας την πραγματική χωρική τους απόσταση.

Σε αντίθεση με προηγούμενες μελέτες στις οποίες είχαν χρησιμοποιηθεί geometry-based στοιχίσεις (Gan et al. 2002) [10], η ανάλυση των K&K βασίζεται σε sequence-based υπερθέσεις. Κατά την πραγματοποίηση δομικών στοιχίσεων βάση ομοιότητας αλληλουχίας αγνοούμε εντελώς τις στερεοδιατάξεις των πρωτεϊνών. Για το λόγο αυτό, ένα λάθος που ίσως έχει προκύψει κατά τη στοίχιση των αλληλουχιών θα μεταφερθεί στη δομική στοίχιση και θα προκαλέσει αύξηση του RMSD που δε συνάδει με τη βιολογική πραγματικότητα. Το λάθος αυτό μπορεί να αντιστοιχεί σε μια απλή ένθεση ή έλλειψη ενός βρόχου το οποίο δεν ανιχνεύεται κατά τις sequence-based στοιχίσεις. Έτσι όταν θα δούμε ένα RMSD ίσο για παράδειγμα με 6, θα θεωρήσουμε ότι οι δύο δομές διαφέρουν σημαντικά, παρόλο που δεν υπάρχει καμία αντιπροσώπευση στη βιολογική πραγματικότητα. Αν δηλαδή τοποθετούσαμε τις δύο πρωτεΐνες τη μία δίπλα στην άλλη, δε θα παρατηρήσουμε δομικές διαφορές. Αντιθέτως, αν κοιτούσαμε απλά το RMSD της υπέρθεσης τους, θα καταλήγαμε σε λάθος συμπεράσματα.

Για να κρίνουμε λοιπόν τη δομική ανομοιότητα των αλληλουχικά παρόμοιων πρωτεϊνών και προσπαθώντας να αποφύγουμε τον επανα-υπολογισμό του RMSD σκεφτήκαμε να συγκρίνουμε τις καταχωρήσεις των πρωτεϊνών μας (ανά ζεύγη) στην CATH: Protein Structure Classification Database (Knudsen et al. 2010) [11]. Η CATH περιλαμβάνει ένα σύνολο ταξινομημένων δομικών επικρατειών. Κάθε πρωτεΐνη έχει κοπεί στις δομικές τις επικράτειες και στη συνέχεια σε ομόλογες υπεροικογένειες (δηλαδή ομάδες επικρατειών που συσχετίζονται μεταξύ τους λόγω εξέλιξης). Με αυτό τον τρόπο μπορούμε να εξετάσουμε πόσο μπορεί να διαφέρουν δύο πρωτεΐνες που διαθέτουν παρόμοιες πρωτοταγείς δομές, κοιτώντας απλά τον αριθμό που τους αντιστοιχεί στην CATH. Η CATH και οι καταχωρήσεις της αναλύονται στο επόμενο κεφάλαιο.

Παρόλο που θα μαρτυρήσω μελλοντικές συζητήσεις, πρέπει να αναφερθεί ότι τα αποτελέσματα της ανάλυσης με την CATH δεν ήταν πολύ ξεκάθαρα όσον αφορά το ερώτημα που θέλαμε να απαντήσουμε και έτσι επιλέξαμε να κάνουμε περαιτέρω υπολογισμούς με τη βοήθεια του TM-align (ενός structure based αλγορίθμου) (Zang et al. 2005) [12] υπολογίζοντας RMSDs για ζεύγη πρωτεϊνών με υψηλό ποσοστό ομοιότητας στην αλληλουχία τους. Στο 3ο και τελευταίο κεφάλαιο αυτής της εργασίας γίνεται επίσης μια προσπάθεια να αποφανθούμε για το αν συγκεκριμένες αμινοξικές αλλαγές παίζουν σημαντικό ρόλο στην αλλαγή της στερεοδιαμόρφωσης των πρωτεϊνών.

Συνοψίζοντας, στα πλαίσια αυτής της διπλωματικής εργασίας: αναλύουμε αν πρωτεΐνες που εμφανίζουν μεγάλη ομοιότητα στην πρωτοταγή τους δομή (≥50%) παρουσιάζουν σημαντικές δομικές διαφορές, αν αυτό ισχύει για πρωτεΐνες που έχουν υποστεί την φυσική επιλογή και εξέλιξη ή απλά έχουν παραχθεί πειραματικά, αμφισβητούμε την ικανότητα των sequence-based υπερθέσεων (σε ορισμένες περιπτώσεις) να αντικατοπτρίζουν τη δομική πραγματικότητα και συζητάμε για το αν υπάρχουν αμινοξικές μεταλλαγές οι οποίες προτιμούνται όταν υπάρχει αλλαγή δομής.

'The important thing is not to stop questioning. Curiosity has its own reason for existing. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery every day. Never lose a holy curiosity.'

Albert Einstein

Κεφάλαιο 2

Αλληλουχικά όμοιες – Δομικά ανόμοιες ?

2.1

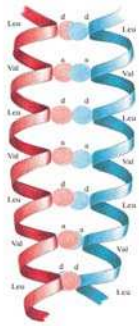
Μοτίβα αναδίπλωσης πρωτεϊνών

Πριν επεκταθούμε στην ανάλυση του θέματός μας, δηλαδή στη συγκριτική μελέτη των δομών αλληλουχικά όμοιων αλυσίδων, ας μιλήσουμε λίγο για μοτίβα αναδίπλωσης πρωτεϊνών. Με αυτό τον τρόπο, θα γίνει πιο εύκολα κατανοητή η σύγκριση των αλληλουχιών αυτών μέσω της CATH, εφόσον οι καταχωρήσεις στην CATH είναι ανάλογες των μοτίβων αναδίπλωσης και της τοπολογίας των διαφόρων πρωτεϊνών. Η CATH, οι καταχωρήσεις της όπως και η σύγκριση των δομών μας μελετούνται διεξοδικά στα υποκεφάλαια που έπονται. Επίσης, στις γραμμές που ακολουθούν, για την ανάλυση των μοτίβων αναδίπλωσης και για την παράθεση αντιπροσωπευτικών εικόνων των μοτίβων, πολύτιμη βοήθεια αποτέλεσε το βιβλίο των C. Branden και J.Tooze “ Εισαγωγή στη Δομή των Πρωτεϊνών ” (1991) [13].

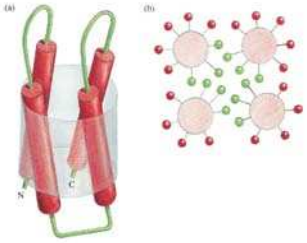
Ο καλύτερος τρόπος για να ξεκινήσουμε την περιγραφή των μοτίβων αναδίπλωσης που υπάρχουν, είναι η δομή της πρωτεΐνης που προσδιορίστηκε πρώτα και ένα Nobel χημείας. Η πρωτεΐνη για την οποία μιλάμε είναι προφανώς η μυοσφαιρίνη και οι κύριοι που μοιράστηκαν το βραβείο είναι οι John Kendrew και Max Perutz (1961) [14].

Η μυοσφαιρίνη αποτελεί μια σφαιρική πρωτεΐνη που ανήκει στην **τάξη δομών τύπου α**. Υιοθετεί μια δομή που ονομάζεται “αναδίπλωση σφαιρίνης” και αποτελεί χαρακτηριστικό παράδειγμα μιας οικογένειας δομών με επικράτειες τύπου α. Σε αυτή την οικογένεια, δημιουργείται ένας υδρόφοβος πυρήνας από μικρές α-έλικες οι οποίες συνδέονται με βρόχους και πακετάρονται μαζί. Εξαιτίας αυτών των υδρόφοβων αλληλεπιδράσεων η δομή είναι πολύ σταθερή, ενώ η υδρόφιλη επιφάνεια επιτρέπει στην πρωτεΐνη να διαλύεται στο νερό. Οι α-έλικες είναι αρκετά ευέλικτες, με αποτέλεσμα να παράγουν αρκετές κατηγορίες δομών:

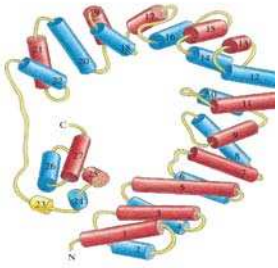
- **σπειρωμένα σπειράματα**: Οι α-έλικες σταθεροποιούνται εντός των πρωτεϊνών χάρη στο πακετάρισμά τους μέσω των υδρόφοβων πλευρικών ομάδων, διαφορετικά μια α-έλικα που βρίσκεται μόνη της σε ένα διάλυμα δεν είναι ιδιαίτερα σταθερή. Ο πιο απλός τρόπος προκειμένου να διατηρηθεί αυτή η σταθερή τους διαμόρφωση είναι να πακεταριστούν δύο α-έλικες μαζί σε μια διπλή υπερέλικα. Τα σπειρωμένα σπειράματα α-ελίκων περιέχουν στις αλληλουχίες τους ένα επαναλαμβανόμενο πρότυπο επτά αμινοξέων.



- **4-α-ελικοειδές δεμάτιο**: Πολλές φορές έχουμε τέσσερις α-έλικες (παράλληλες ή αντιπαράλληλες) οργανωμένες σε ένα “δεμάτιο” με τους ελικοειδείς τους άξονες σχεδόν παράλληλους μεταξύ τους. Το σύνολο των αμινοξέων που “θάβονται” μεταξύ των ελίκων είναι συνήθως υδρόφοβα δημιουργώντας έναν υδρόφοβο πυρήνα, αποκλείοντας το νερό από την περιοχή.

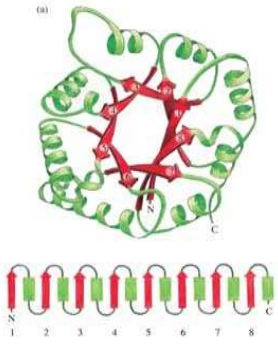


- **α-ελικοειδείς επικράτειες μεγάλες και πολύπλοκες**: Σε πολλά ένζυμα που διαθέτουν μία μεγάλη πολυπεπτιδική αλυσίδα, αυτή αναδιπλώνεται σε περισσότερες από μία α-έλικες, σχηματίζοντας συνολικά μια σφαιρική επικράτεια.

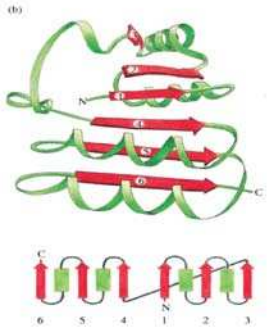


Οι **<u>α/β επικράτειες</u>** είναι αυτές που συναντώνται συχνότερα στις πρωτεϊνικές δομές. Στο συγκεκριμένο μοτίβο, α-έλικες περιβάλλουν μία κεντρική παράλληλη ή μεικτή β-πτυχωτή επιφάνεια και βρόχοι ενώνουν τις διάφορες περιοχές. Υπάρχουν τρεις κύριες κατηγορίες α/β-πρωτεϊνών.

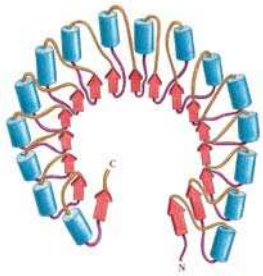
- **Βαρέλια ή πτυχωτές επιφάνειες (παράλληλοι β-κλώνοι)**: Στην κατηγορία αυτή, σχηματίζεται ένας πυρήνας συστραμμένων παράλληλων β-κλώνων που διευθετούνται κοντά ο ένας στον άλλο. Οι κλώνοι αυτοί συνδέονται μέσω α-ελίκων που βρίσκονται στο εξωτερικό αυτού του βαρελιού. Η δομή της συγκεκριμένης επικράτειας ονομάζεται συχνά και TIM barrel.



- **α/β-δομές ανοικτών πτυχωτών επιφανειών**: “Μια ανοιχτή συστραμμένη β-πτυχωτή επιφάνεια, που περιβάλλεται εκατέρωθεν από α-έλικες”. Έτσι θα περιέγραφε κάποιος μια α/β-δομή ανοικτής πτυχωτής επιφάνειας. Το μέγεθος, ο αριθμός και η σειρά των β-κλώνων διαφέρουν σε αυτές τις δομές. Σε αντίθεση επίσης, με τα α/β-βαρέλια, οι περιοχές των ενεργών κέντρων σχηματίζονται στα σημεία εκείνα όπου δύο γειτονικοί βρόχοι κατευθύνονται προς αντίθετες πλευρές της επιφάνειας.



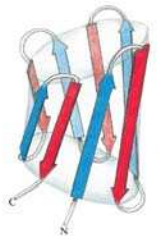
- **Πεταλοειδές δίπλωμα**: Μοτίβα πλούσια σε λευκίνη, δηλαδή αλληλουχίες που περιέχουν μία επανάληψη επτάδας καταλοίπων λευκίνης σχηματίζουν α-έλικες και β-κλώνους. Οι α-έλικες περιβάλλουν εξωτερικά β-κλώνους που σχηματίζουν μια καμπυλωτή παράλληλη β-πτυχωτή επιφάνεια.



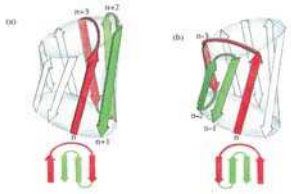
Συλλογικά, όλες οι α/β επικράτειες, δηλαδή, τα βαρέλια, οι δομές ανοικτών β-πτυχωτών επιφανειών και οι πεταλοειδείς δομές σχηματίζονται από μοτίβα β-α-β που συνδέονται με ποικίλους τρόπους.

Οι **<u>αντιπαράλληλες β-δομές</u>** αποτελούν τη δεύτερη μεγάλη ομάδα πρωτεϊνικών επικρατειών. Β-κλώνοι, που ποικίλλουν σε αριθμό, σχηματίζουν τους πυρήνες των επικρατειών αυτών. Συνήθως είναι αντιπαράλληλοι και διευθετούνται σχηματίζοντας δύο β-πτυχωτές επιφάνειες πακεταρισμένες η μία απέναντι στην άλλη.

- **άνω-και-κάτω β-πτυχωτές επιφάνειες ή βαρέλια**: Οι β-κλώνοι διευθετούνται ο ένας δίπλα στον προηγούμενο του μέχρι να σχηματιστεί μια κυκλική δομή, όπου ο τελευταίος κλώνος συνδέεται με τον πρώτο με δεσμούς υδρογόνου.

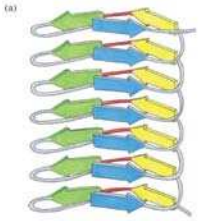


- **Μοτίβα τύπου Ελληνικό κλειδί**: Δομές αντιπαράλληλων β-βαρελιών ενσωματώνουν αντιπαράλληλους β-κλώνους που βρίσκονται σε αντίθετες πλευρές ενός βαρελιού και συνδέονται μεταξύ τους με πολύ απλό τρόπο.



- **Βαρέλια τύπου jelly roll**: Αποτελούν μια υποκατηγορία των μοτίβων τύπου Ελληνικό κλειδί στην οποία μία από τις συνδέσεις του μοτίβου διασχίζει τη μία βάση του βαρελιού.

- **Ένα πρωτότυπο μοτίβο**: Μια πολυπεπτιδική αλυσίδα περιελίσσεται με τέτοιο τρόπο, σχηματίζοντας μια πλατιά έλικα που αποτελείται από β-κλώνους που διαχωρίζονται από βρόχους. Στην απλούστερη μορφή, όταν η β-έλικα περιλαμβάνει δύο πτυχωτές επικράτειες, η κάθε στροφή της έλικας περιέχει δυο κλώνους και δυο βρόχους.



Μετά τη σύντομη αναφορά σχετικά με τα διάφορα είδη μοτίβων που κυριαρχούν στις πρωτεΐνες, μπορούμε να συνεχίσουμε με περαιτέρω ανάπτυξη του τρόπου σύγκρισης των πρωτεϊνικών μας δομών (αλληλουχικά όμοιων πρωτεϊνών), την παράθεση των αποτελεσμάτων μας και τα συμπεράσματα στα οποία καταλήξαμε.

2.2

Protein Structure Classification Database

Όπως έχουμε ήδη αναφέρει στην εισαγωγή, για να κρίνουμε τη δομική ανομοιότητα αλληλουχικά παρόμοιων πρωτεϊνών, έχουμε σκοπό να συγκρίνουμε τις καταχωρήσεις των πρωτεϊνών μας (ανά ζεύγη) στην CATH. Η βάση δεδομένων της CATH αποτελεί μια ιεραρχικά ταξινομημένη βάση επικρατειών πρωτεϊνικών δομών της PDB (PDB, Berman et al. 2003) [15]. Στην CATH περιλαμβάνονται μόνο κρυσταλλικές δομές με διακριτικότητα καλύτερη των 4 Angstroms και NMR δομές. Επίσης όλες οι μη-πρωτεϊνες, τα μοντέλα και οι δομές που παρουσιάζουν ποσοστό μεγαλύτερο από 30% της δομής για μόνο Cα ατόμα εξαιρούνται από την CATH. Υπάρχουν 4 βασικά επίπεδα: Τάξη, Αρχιτεκτονική, Τοπολογία και Ομόλογη Υπεροικογένεια (Orengo et al., 1997) [16].

Depth	Letter	Name	Clustering criteria
1	C	Class	Secondary structure content
2	A	Architecture	General spatial arrangement of secondary structures
3	T	Topology	Spatial arrangement and connectivity of secondary structures (fold)
4	H	Homologous Superfamily	Manual curation of evidence of evolutionary relationship (at least two criteria from sequence/structure/function must be observed)
5	S	Sequence Family (S35)	>= 35% sequence similarity
6	O	Orthologous Family (S60) *	>= 60% sequence similarity
7	L	“Like” domain (S95) *	>= 95% sequence similarity
8	I	Identical domain (S100)	100% sequence similarity
9	D	Domain counter	Unique domains

Εικόνα 5: Ποια η σημασία των γραμμάτων της CATHSOLID (αναπαράγεται άνευ αδείας από http://www.cathdb.info/).

- **Τάξη** (*C for class*) – οι δομές ταξινομούνται σύμφωνα με την δευτεροταγή τους δομή (περισσότερο άλφα, περισσότερο βήτα, αναμεμειγμένες άλφα/βήτα ή λίγες δευτεροταγείς δομές). Η πλειονότητα των μοτίβων δευτεροταγούς δομής έχει ήδη αναλυθεί διεξοδικά στο κεφάλαιο 2.1.

- **Αρχιτεκτονική** (*A for architecture*) – οι δομές ταξινομούνται σύμφωνα με το συνολικό τους σχήμα, όπως αυτό καθορίζεται από τις διατάξεις των στοιχείων δευτεροταγών τους δομών στον τρισδιάστατο χώρο, αλλά αγνοώντας των τρόπο σύνδεσης μεταξύ τους.

- **Τοπολογία** (*T for topology*) – σε αυτό το επίπεδο οι δομές ομαδοποιούνται σε ομάδες αναδίπλωσης βάση τόσο του ολικού σχήματος όσο και του τρόπου σύνδεσης των στοιχείων δευτεροταγών τους δομών.

- **Ομόλογη Υπεροικογένεια** (*H for homology superfamily*) – σε αυτό το επίπεδο ομαδοποιούνται μαζί πρωτεϊνικές επικράτειες που θεωρείται ότι έχουν κάποιο κοινό πρόγονο και γι αυτό περιγράφονται ως ομόλογες

- **Επίπεδα Αλληλουχικής Οικογένειας** (*S for sequence family* levels) - S,O,L,I,D

Σε αυτή την κατηγορία, επικράτειες σε κάθε *Η-επίπεδο* υποομαδοποιούνται σε αλληλουχικές οικογένειες στα παρακάτω επίπεδα:

Level	Sequence Identity	Overlap
S	35%	80%
O	60%	80%
L	95%	80%
I	100%	80%

Εικόνα 6: Επίπεδα αλληλουχικής ταξινόμησης στο Η επίπεδο (αναπαράγεται άνευ αδείας από http://www.cathdb.info/).

Το επίπεδο D συμπεριφέρεται ως μετρητής μέσα σε κάθε S100 οικογένεια έτσι ώστε κάθε επικράτεια στην CATH να διατηρεί μία μοναδική CATHSOLID καταχώρηση.

-Π*ως συνδέσαμε την CATH με την έρευνά μας*?

Στη δική μας περίπτωση επιλέγουμε να συγκρίνουμε μόνο τους τέσσερις πρώτους αριθμούς της CATH για να πραγματοποιήσουμε τις δομικές συγκρίσεις. Εφόσον αναζητούμε ανομοιότητα σε επίπεδο δομής, στην περίπτωση που οι καταχωρήσεις στην CATH διαφέρουν, αυτό θα αποτελεί δείγμα ότι υπάρχει αλλαγή μεταξύ των δομών. Ο τρίτος αριθμός της CATH που σχετίζεται με την *Τοπολογία* είναι πολύ σημαντικός, καθώς αν δύο πρωτεΐνες διαφέρουν στο μοτίβο δευτεροταγούς τους δομής ή στον τρόπο σύνδεσής τους, τότε θα έχουν αναγκαστικά διαφορετικές δομές. Αντίθετα, αν δύο δομές παρουσιάζουν ίδια *Αρχιτεκτονική* ή πόσο μάλλον αν ανήκουν στην ίδια *Τάξη*, τότε δε θα υιοθετούν υποχρεωτικά και ίδιες στερεοδιαμορφώσεις.

Τα ζεύγη πρωτεϊνών που χρησιμοποιήσαμε προέρχονται από την έρευνα των K&K. Πιο συγκεκριμένα, η αρχική βάση δεδομένων που χρησιμοποίησαν οι K&K περιελάμβανε όλες τις πρωτεϊνικές αλυσίδες της PDB (Απρίλιος 2005), οι οποίες περιείχαν πάνω από 35 κατάλοιπα και οι δομές των οποίων είχαν καθοριστεί σε διακριτικότητα καλύτερη των 2.5 A με κρυσταλλογραφία ακτίνων Χ. 38449 αλυσίδες από τις 19295 πρωτεΐνες ικανοποιούσαν αυτά τα κριτήρια. Οι αλυσίδες με αλληλουχική ομοιότητα 100% και RMSD μικρότερο του 1 A των Cα ατόμων τους θεωρήθηκαν ως όμοιες. Σε κάθε ομάδα όμοιων πρωτεϊνών κρατήθηκε αυτή με την καλύτερη διακριτικότητα. Και στην περίπτωση που υπήρχαν πολλές όμοιες αλυσίδες με ίδια διακριτικότητα, κρατήθηκε αυτή που διέθετε τα περισσότερα κατάλοιπα. Το τελικό dataset περιείχε 13193 αλυσίδες από 9906 πρωτεϊνικές δομές.

Οι αλληλουχίες όλων των παραπάνω αλυσίδων συγκρίθηκαν με το πρόγραμμα του BLAST (bl2seq). Στοιχίσεις που είχαν: (α) αλληλουχική ομοιότητα μεγαλύτερη ή ίση του 50 %, (β) e-value καλύτερο από 0.001 και (γ) τουλάχιστον 35 όμοια κατάλοιπα στη στοίχιση, επιλέχθηκαν, καταλήγοντας σε 147,186

ζευγάρια. Τέλος με τη χρήση sequence-based δομικών στοιχίσεων τα ζεύγη πρωτεϊνών που προέκυψαν χωρίστηκαν σε οχτώ (αλληλεπικαλυπτώμενες) ομάδες, βασισμένες στην αλληλουχική τους (≥ 50, 10, 99, και 100%) και στη δομική ομοιότητα (RMSD μεγαλύτερο από 3, και 6 Á) (Πίνακας 1).

Sequence Identity	Total Pairs(sequence clusters)		
	All	RMSD ≥ 3 Å	RMSD ≥ 6 Å
100%	1941	444 (184)	158 (60)
≥ 99%	12868	757 (216)	278 (69)
≥ 70%	114021	6873 (353)	1575 (126)
≥ 50%	147186	11749 (401)	2653 (138)

Πίνακας 1: Ο συγκεκριμένος πίνακας περιλαμβάνει όλα τα ζεύγη αλληλουχιών των K. & K. Ομαδοποιημένα συγκριτικά με την αμινοξική τους ομοιότητα και το RMSD που προέκυψε από τις sequence-based στοιχίσεις. Στο κόκκινο κουτάκι παρατίθεται η ομάδα η οποία περιέχει όλα τα ζεύγη για ποσοστό ομοιότητας 50 και RMSD μεγαλύτερο των 3.

Εφόσον οι ομάδες αυτές είναι αλληλεπικαλυπτώμενες, όλες θα αποτελούν υποσύνολο αυτής που περιέχει τα ζεύγη αλυσίδων για μεγαλύτερη του 50% ομοιότητα και μεγαλύτερο του 3 Å RMSD. Αυτή είναι η βάση δεδομένων που θα χρησιμοποιήσουμε εμείς στη έρευνά μας και θα την ονομάσουμε ομάδα G (Πίνακας 1, κόκκινο πλαίσιο). Η ομάδα G περιλαμβάνει 11749 ζεύγη πρωτεϊνών τα οποία χωρίζονται σε 401 υποομάδες.

2.3

'*wget_CATH_codes*'

Για να μπορέσουμε να επεξεργαστούμε την πληθώρα αλληλουχιών της ομάδας G, απαραίτητη ήταν η δημιουργία ενός προγράμματος, με το όνομα '***wget_CATH_codes***', το οποίο θα αντιπαρέλαβε τους κωδικούς CATH των αλληλουχιών του κάθε ζεύγους.

Συνοπτικά το πρόγραμμα (το οποίο παρατίθεται παρακάτω):

- διαβάζει το αρχείο που περιλαμβάνει τα ζεύγη πρωτεϊνών της ομάδας G
- επιλέγει τους κωδικούς των πρωτεϊνών του κάθε ζεύγους
- κατεβάζει τις αντίστοιχες σελίδες των πρωτεϊνικών αλυσίδων που αντιστοιχούν στην CATH
- για όσες καταχωρήσεις υπάρχουν στην CATH τυπώνει τους CATH κωδικούς με ένα επιπλέον σύμβολο (“<=”) στις περιπτώσεις που έστω και ένας κωδικός CATH διαφέρει μεταξύ των ζευγών

'wget_CATH_codes'

```
#!/usr/bin/perl -w

#
# This program downloads and compares the CATH codes
# foreach pair of sequences from K&Ks database
#



#
# Read STDIN
# while reading e.g. a log file
# get PDB codes and save them in $id1
# and $id2 variables respectively
#

while ( $line = <STDIN> )
    {
        if ( $line =~ /^(.....) (.....) / )
        {

        #
        # in the meanwhile whenever for a chain identifier
        # is given "_" symbol, replace it with space
        #

        $id1 = $1;
        $id1 =~ tr/_/ /;
        $id2 = $2;
        $id2 =~ tr/_/ /;

        #
        # download CATH pages for the PDB pairs
        # convert them to text
        # get CATH codes and sort them
        #

        $codes_1 = `wget --quiet -O - "http://www.cathdb.info/cgi-bin/SearchPdb.pl?query=$id1&type=PDB" | html2text | grep -o -P ' \\d+\\.\\d+\\.\\d+\\.\\d+ ' |
sort -u`;
        $codes_2 = `wget --quiet -O - "http://www.cathdb.info/cgi-bin/SearchPdb.pl?query=$id2&type=PDB" | html2text | grep -o -P ' \\d+\\.\\d+\\.\\d+\\.\\d+ ' |
sort -u`;



        #
        # print "<=" symbol when $code_1 code
        # differs from $code_2 code
        # else print nothing
```

```
        if ( $codes_1 ne $codes_2 )
            {
                $message = "<=";
            }

        else
            {
                $message = "";
            }

        #
        # only when both $code_1 and $code_2 are known
        # print them followed by their CATH codes in one line
        #

        $codes_1 =~ tr/\n/ /;
        $codes_2 =~ tr/\n/ /;

        if ( $codes_1 eq "" )
            {
                $codes_1 = "???????????";
            }
        elsif ( $codes_2 eq "" )
            {
                $codes_2 = "???????????";
            }
        else
            {
              printf " %6s %-50s     %6s %-50s  $message\n", $id1, $codes_1, $id2, $codes_2;
            }
        }
    }

exit;
```

2.4

CATH SIFT CRITERIA

Οι δομές που περιέχονται στην PDB δεν έχουν όλες την ίδια ποιότητα. Για να διατηρηθεί η αξιοπιστία λοιπόν κατά την ταξινόμηση και ομαδοποίηση των πρωτεϊνικών επικρατειών είναι σημαντικό να περιλαμβάνονται μόνο υψηλής ποιότητας δομές. Για το λόγο αυτό, έχει θεσπιστεί ένα σύνολο αυστηρών κριτηρίων, που ονομάζονται SIFT τα οποία καθορίζουν αν μια αλυσίδα της PDB μπορεί να γίνει δεκτή στην CATH.

-Ποια είναι λοιπόν αυτά τα κριτήρια;

Για δομές που είχαν καταχωρηθεί μέχρι και όλες τις εκδόσεις 2.x της CATH:

- η μέθοδος εύρεσης της δομής της πρωτεΐνης θα πρέπει να είναι 'κρυσταλλογραφία ακτίνων X' ή 'NMR'.
- το ποσοστό της δομής για μόνο Cα άτομα θα πρέπει να μην είναι >= 70%
- το μήκος της αλληλουχίας πρέπει να είναι >= 40 κατάλοιπα

Για τις δομές που έχουν καταχωρηθεί μετέπειτα τα κριτήρια έχουν γίνει λίγο πιο ελαστικά:

- Στην περίπτωση που η μέθοδος εύρεσης είναι άγνωστη η δομή θα γίνει αποδεκτή αν έχει προσδιοριστεί σε διακριτικότητα καλύτερη των 4.0 Å,
- και τα ο αριθμός των καταλοίπων μπορεί να είναι μικρότερος από 40

Από τα 11749 ζεύγη πρωτεϊνών που περιλαμβάνονται στη βάση δεδομένων της ομάδας G, τα 11047 ζεύγη πληρούν τις προϋποθέσεις SIFT και με αυτές θα συνεχίσουμε την έρευνά μας.

2.5

Αποτελέσματα δομικών στοιχίσεων

'**Results!** Why, man, I have gotten a lot of results.
I know several thousand things that won't work.'

Thomas A. Edison

2.5.1

Πανομοιότυπες καταχωρήσεις CATH

Όπως έχουμε τονίσει και στην εισαγωγή αυτό που αναζητούμε είναι πρωτεΐνες οι οποίες παρουσιάζουν μεγάλη ομοιότητα στην αλληλουχία και παρόλα αυτά έχουν διαφορετικές δομές. Τα περισσότερα από αυτά τα 11047 αποτελέσματα δεν αναλύθηκαν περαιτέρω καθώς παρουσίασαν πανομοιότυπους κωδικούς CATH. Η πλήρης αντιστοίχιση των κωδικών αυτών συνεπαγόταν πιθανότατα ομοιότητα των δομών αυτών.

Ορισμένα παραδείγματα (η επιλογή ήταν τυχαία, με μόνο ίσως κριτήριο να καλύπτουν όλες τις

τάξεις των καταχωρήσεων της CATH) που υποστηρίζουν ότι καλώς δεν μελετήσαμε περισσότερο αυτά τα αποτελέσματα παρατίθενται παρακάτω. Τα ζεύγη αυτά έχουν ληφθεί από την ομάδα H των K&K. Η επιλογή αυτή έγινε για το λόγο ότι η ομάδα H περιλαμβάνει μόνο τα αποτελέσματα από την ομάδα G τα οποία εμφανίζουν μεγαλύτερο του έξι RMSD. Προτιμήσαμε το μεγαλύτερο RMSD καθώς όταν το RMSD αυξάνεται αρκετά, οι αλλαγές που αναμένουμε μεταξύ των δομών αυτών θα πρέπει να είναι σημαντικές.

Στην πραγματικότητα, οι βάσεις δεδομένων που είχαν καταθέσει οι K&K δε περιελάμβαναν συγκεκριμένα τα όρια των αλληλουχιών των ζευγαριών που είχαν προκύψει από το πρόγραμμα του BLAST. Έτσι χρειάστηκε να τρέξουμε πάλι το πρόγραμμα του BLAST (συγκεκριμένα το bl2seq) με τις παραμέτρους που είχαν δηλώσει στο άρθρο τους για να πάρουμε τις σωστές στοιχίσεις. Στη συνέχεια, χρησιμοποιήσαμε τις στοιχίσεις αυτές για να απομονώσουμε τα σωστά κατάλοιπα στα αντίστοιχα αρχεία PDB. Επειδή η διαδικασία αυτή χρειάστηκε να γίνει μόνο για 40 ζευγάρια (τα οποία χρησιμοποιήσαμε ως παραδείγματα), πραγματοποιήθηκε χειρωνακτικά. Για την πραγματοποίηση των παρακάτω υπερθέσεων έγινε χρήση του προγράμματος **lovoalign** (Martinez et al. 2007) [17] και οι εικόνες προβλήθηκαν με τη βοήθεια του **vmd** (Humphrey et al. 1996) [18] και του **Raster 3D** (Merritt et al. 1997) [19]. Οι παρακάτω εικόνες περιλαμβάνουν τους κωδικούς των πρωτεϊνών (χρωματισμός με πράσινο – μοβ αντίστοιχα με τη σειρά που εμφανίζονται οι κωδικοί των πρωτεϊνών), τους κωδικούς CATH που τους αντιστοιχούν και το RMSD που έδωσε το lovoalign τόσο για όλη τη στοίχιση όσο και για τα άτομα με απόσταση λιγότερη από 3 Å (συμβολίζονται με < 3 Å RMSD). Μέσα στις παρενθέσεις αναγράφεται το μήκος της κάθε στοίχισης.

1.SUPERPOSITION of

1bsaA with **1yvs**A

CATH_code: 3.10.450.30
CATH_code: 3.10.450.30

RMSD:
22(107)

< 3 Å RMSD:
0.5(70)

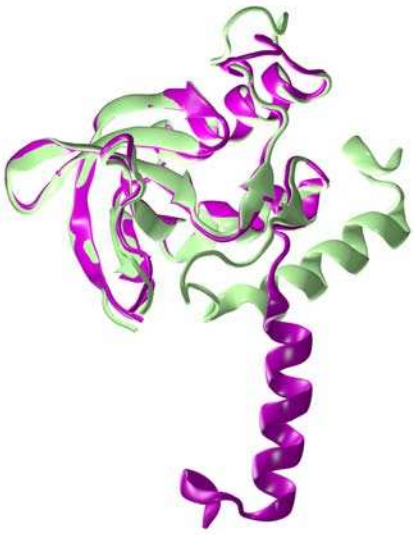
2.SUPERPOSITION of

1eyaA with **1snd**A

CATH_code: 2.40.50.90
CATH_code: 2.40.50.90

RMSD:
11(129)

< 3 Å RMSD:
0.7(104)



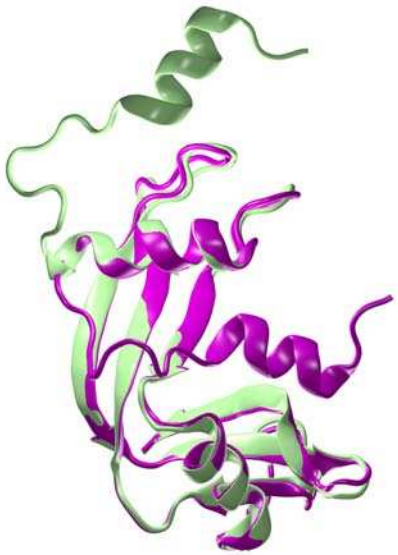
3.SUPERPOSITION of

11bgA with **1a5p**A

CATH_code: 3.10.130.10
CATH_code: 3.10.130.10

RMSD:
11(124)

< 3 Å RMSD:
0.6(103)



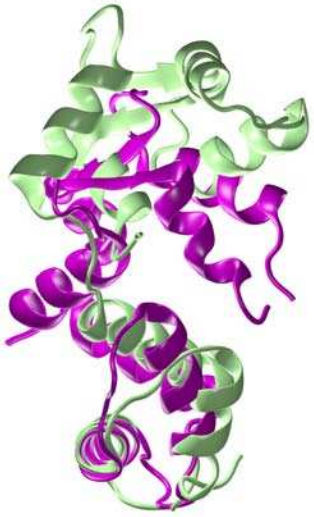
4.SUPERPOSITION of

1a2xA with **1dtl**A

CATH_code: 1.10.238.10
CATH_code: 1.10.238.10

RMSD:
9(99)

< 3 Å RMSD:
2(39)



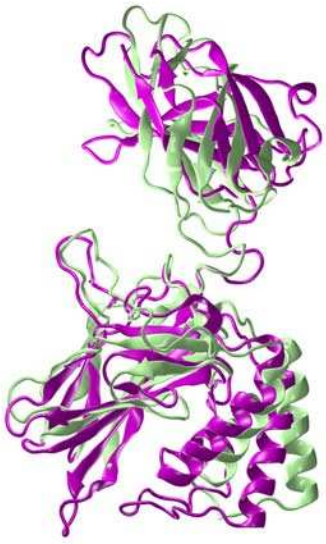
5.SUPERPOSITION of

1a3qA with **1ooa**A

CATH_code: 2.60.40.10 2.60.40.340
CATH_code: 2.60.40.10 2.60.40.340

RMSD:
5(260)

< 3 Å RMSD:
2(84)



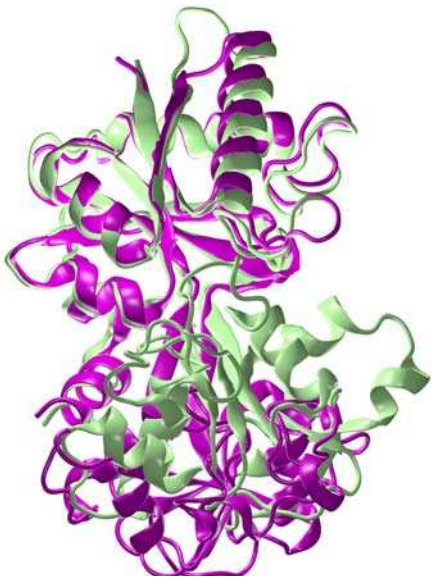
6.SUPERPOSITION of

1a8eA with **1cb6**A

CATH_code: 3.40.190.10
CATH_code: 3.40.190.10

RMSD:
7(279)

< 3 Å RMSD:
1.5(182)



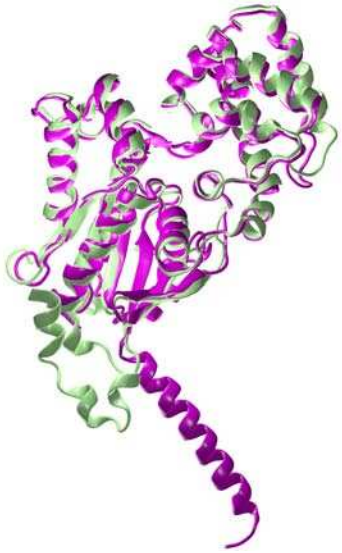
7.SUPERPOSITION of

1bofA with **1gg2**A

CATH_code: 1.10.400.10 3.40.50.300
CATH_code: 1.10.400.10 3.40.50.300

RMSD:
8(318)

< 3 Å RMSD:
1(293)



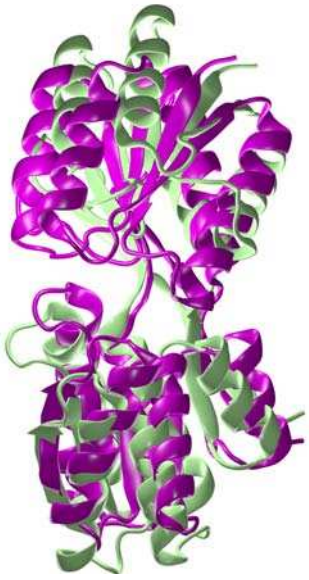
8.SUPERPOSITION of

1ba2A with **1dbp**A

CATH_code: 3.40.50.2300
CATH_code: 3.40.50.2300

RMSD:
4(240)

< 3 Å RMSD:
2(93)



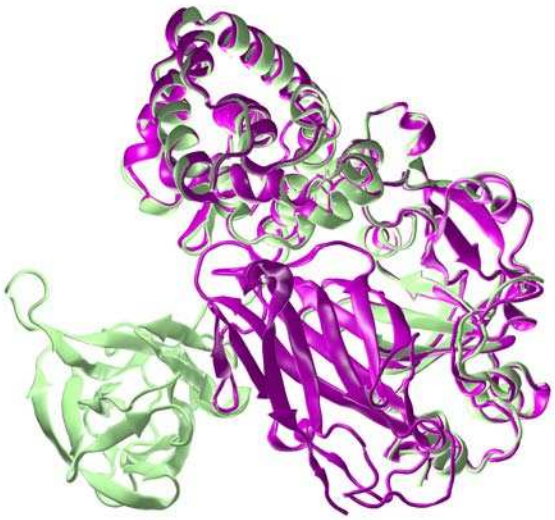
9.SUPERPOSITION of

1ddtA with **1mdt**A

CATH_code: 1.10.490.40 2.60.40.700 3.90.175.10
CATH_code: 1.10.490.40 2.60.40.700 3.90.175.10

RMSD:
20(523)

< 3 Å RMSD:
0.7(367)



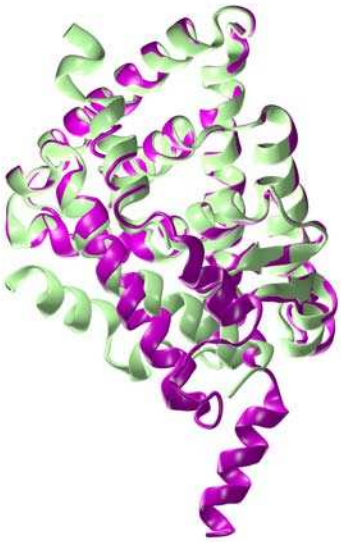
10.SUPERPOSITION of

1dkfA with **1g1u**B

CATH_code: 1.10.565.10
CATH_code: 1.10.565.10

RMSD:
8.5(211)

< 3 Å RMSD:
0.9(176)



11.SUPERPOSITION of

1dvmA with **1lj5**A

CATH_code: 2.30.39.10 3.30.497.10
CATH_code: 2.30.39.10 3.30.497.10

RMSD:
10(375)

< 3 Å RMSD:
1.5(306)

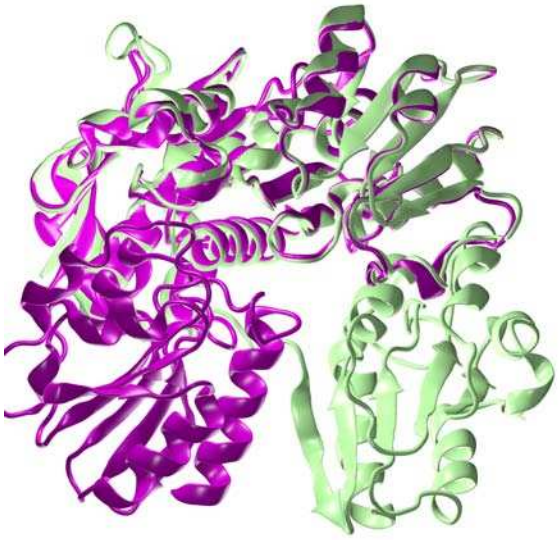
12.SUPERPOSITION of

1eehA with **1uag**A

CATH_code: 3.40.1190.10 3.40.50.720 3.90.190.20
CATH_code: 3.40.1190.10 3.40.50.720 3.90.190.20

RMSD:
20(412)

< 3 Å RMSD:
0.8(287)



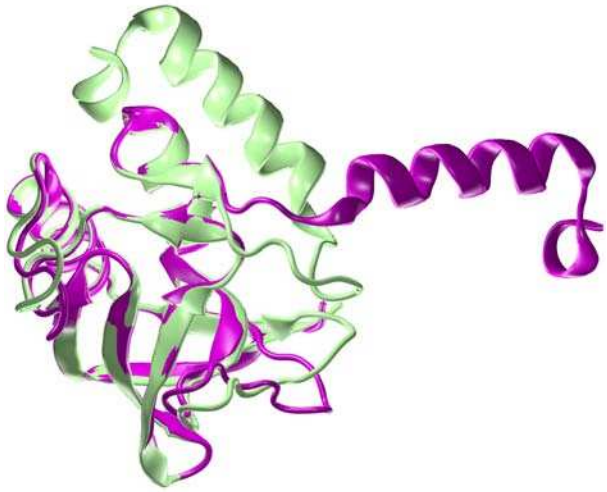
13.SUPERPOSITION of

1encA with **1snd**B

CATH_code: 2.40.50.90
CATH_code: 2.40.50.90

RMSD:
10(129)

< 3 Å RMSD:
0.8(101)



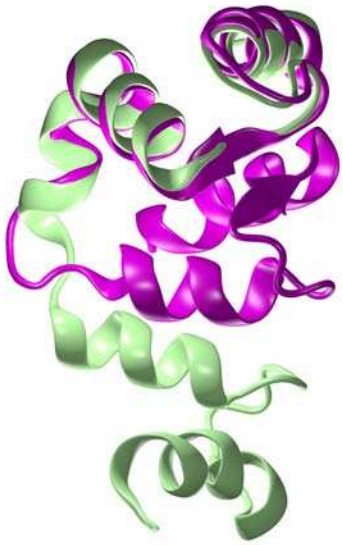
14.SUPERPOSITION of

1ht9A with **1ig5**A

CATH_code: 1.10.238.10
CATH_code: 1.10.238.10

RMSD:
13(72)

< 3 Å RMSD:
1(43)



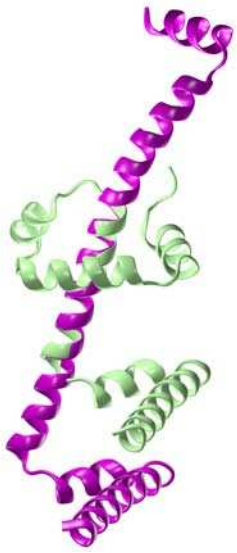
15.SUPERPOSITION of

1jhgA with **1mi7**R

CATH_code: 1.10.1270.10
CATH_code: 1.10.1270.10

RMSD:
19(86)

< 3 Å RMSD:
0.7(21)



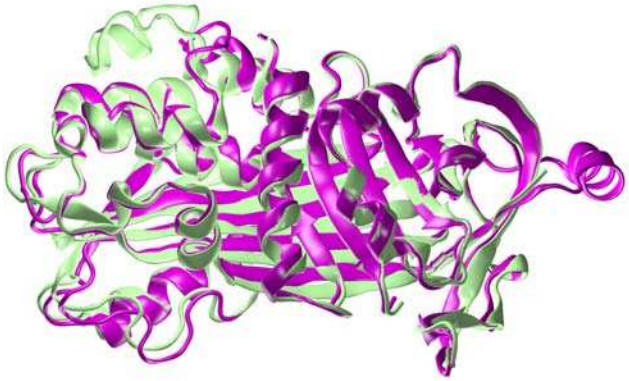
16.SUPERPOSITION of

1jtiA with **1ova**B

CATH_code: 2.30.39.10 3.30.497.10
CATH_code: 2.30.39.10 3.30.497.10

RMSD:
9(368)

< 3 Å RMSD:
1(328)



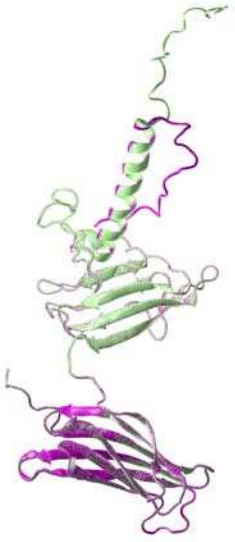
17.SUPERPOSITION of

1qexA with **1s2e**A

CATH_code: 1.20.5.960 2.60.120.640
2.60.40.1680
CATH_code: 1.20.5.960 2.60.120.640
2.60.40.1680

RMSD:
11(288)

< 3 Å RMSD:
RMSD:
0.2(271)



18.SUPERPOSITION of

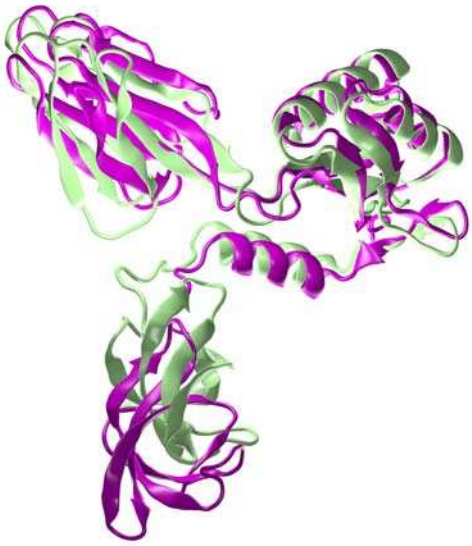
1m1gC with **1npr**A

CATH_code: 2.30.30.30 2.60.320.10 3.30.70.940

CATH_code: 2.30.30.30 2.60.320.10 3.30.70.940

RMSD:
5(226)

< 3 Å RMSD:
2(112)



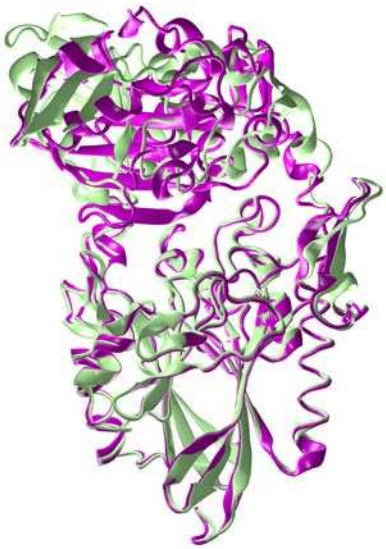
19.SUPERPOSITION of

1hpuA with **1oid**A

CATH_code: 3.60.21.10 3.90.780.10
CATH_code: 3.60.21.10 3.90.780.10

RMSD:
7(493)

< 3 Å RMSD:
0.7(342)



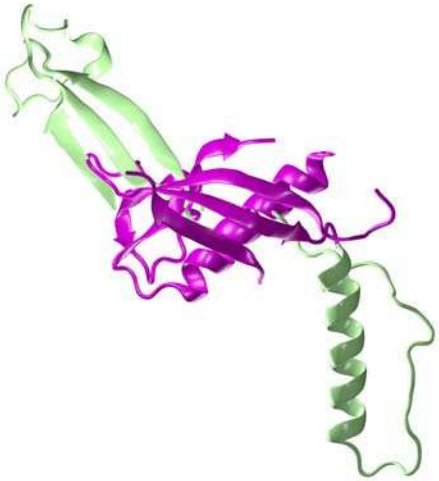
20.SUPERPOSITION of

1lxeA with **1pfp**A

CATH_code: 3.10.450.10
CATH_code 3.10.450.10

RMSD:
21(45)

< 3 Å RMSD:
2(10)



-Τι έχουν να δηλώσουν οι εικόνες αυτές;

Η πρώτη αντίδραση που θα είχε κάποιος παρατηρώντας τις δομές αυτές, θα ήταν να θεωρήσει ότι μοιάζουν σημαντικά, καθώς οι δομές με ελάχιστες εξαιρέσεις (όπως οι υπερθέσεις 9, 12 και 16) σχεδόν ταυτίζονται. Μια πιο προσεκτική παρατήρηση θα μας οδηγούσε στο συμπέρασμα ότι πολλές από τις δομές, και συγκεκριμένα αυτές που έχουν 2 τουλάχιστον επικράτειες, εμφανίζουν πλήρη δομική στοίχιση όσο αναφορά τη μία επικράτειά τους, αλλά τα πράγματα δεν είναι τόσο ξεκάθαρα για το άλλο μισό της δομής. Μπορεί να μη γίνεται εύκολα αντιληπτό με το μάτι εξαιτίας της υπέρθεσης ολόκληρης της δομής, αλλά ακόμη και οι επικράτειες που δε φαίνεται να μοιάζουν έχουν ουσιαστικά την ίδια ακολουθία μοτίβων αναδίπλωσης δευτεροταγούς δομής και τις ίδιες συνδέσεις μεταξύ των μοτίβων. Έτσι μια απλή μετατόπιση του τμήματος αυτού θα έπειθε και τον πιο καχύποπτο κρυσταλλογράφο ότι οι δομές δε διαφέρουν. Το αίτιο για την αύξηση του RMSD και για τη μη πλήρη ταύτιση των δομών αποτελεί πιθανότατα κάποια αλλαγή στο κομμάτι σύνδεσης(βρόχος) μεταξύ των επικρατειών, αλλάζοντας με αυτό τον τρόπο τη σχετική θέση της μιας επικράτειας.

Επιπλέον, στις περιπτώσεις που οι πρωτεϊνικές αλυσίδες είναι μικρότερες και δεν έχουμε πολλές επικράτειες γίνεται εύκολα αντιληπτό ότι κάποιο τμήμα της πρωτεΐνης έχει μετατοπιστεί και έχει οδηγήσει πιθανότατα σε αυτή την αύξηση του RMSD που δεν ανταποκρίνεται στη δομική πραγματικότητα.

2.5.2

Όταν οι κωδικοί CATH αλλάζουν...

Στο υποκεφάλαιο αυτό, θα συνεχίσουμε την ανάλυση των αποτελεσμάτων που έφεραν διαφορετικές καταχωρήσεις στην CATH. Η έξοδος που παρήγαγε το πρόγραμμά μας, όπως έχουμε αναφέρει επανειλημμένα, ήταν 11047 ζεύγη πρωτεϊνών με τους αντίστοιχους κωδικούς CATH για την κάθε καταχώρηση. Μέσα σε αυτά τα αποτελέσματα υπήρχαν 267 ζεύγη τα οποία δε μοιραζόντουσαν έστω και έναν κωδικό CATH μεταξύ τους. Η οποιαδήποτε διαφορά σημαινόταν από το βελάκι “<=” που τοποθετούσε το πρόγραμμά μας στην αντίστοιχη γραμμή. Εφόσον, πανομοιότυποι κωδικοί στην CATH έχουμε δείξει ότι σηματοδοτούν και ίδιες δομές, αποφασίσαμε να αναλύσουμε περαιτέρω μόνο αυτά τα 267 αποτελέσματα.

Παρατηρήστε προσεκτικά τα αποτελέσματα παρακάτω. *Σας θυμίζουν κάτι; Μήπως οι δομές δε διαφέρουν και τόσο πολύ; Μήπως οι δομικές διαφορές οφείλονται πάλι σε μετατόπιση επικράτειας λόγω αλλαγής κάποιου βρόχου;* Από την άλλη μεριά τα νούμερα των καταχωρήσεων διαφέρουν.

-Τι συμπεράσματα βγάζουμε από τις αντιπαραθέσεις αυτές;

Lets be critical! Το λεπτό σημείο που δε πρέπει να αφήσουμε να περάσει απαρατήρητο εδώ είναι το εξής: “ έστω και **<u>ένας</u>** διαφορετικός κωδικός CATH”. Το πρόγραμμά μας σημειώνει ώς διαφορετικές όλες τις καταχωρήσεις που διαφέρουν έστω και σε ένα νούμερο. Αυτό σημαίνει ότι αν μία πρωτεϊνική αλυσίδα έχει περισσότερους κωδικούς CATH από μια άλλη, γιατί η CATH έτσι έχει ορίσει για αυτή την αλυσίδα (επειδή για παράδειγμα έχει πολλές διαφορετικές επικράτειες), τότε το πρόγραμμά μας θα θεωρήσει ότι οι καταχωρήσεις της CATH διαφέρουν, κάτι το οποίο δεν ισχύει στις περιπτώσεις αυτές.

Έτσι για να αποφύγουμε παρόμοιες με την προαναφερθείσα περιπτώσεις συγκρίναμε χωρίς τη βοήθεια κάποιου προγράμματος αλλά με απλή αντιστοίχιση τα νούμερα για τα 267 αυτά ζεύγη. Οι εικόνες που ακολουθούν είναι ενδεικτικές ζευγών πρωτεϊνών που δεν αντιστοιχούν σε πραγματική διαφορά αριθμών στην CATH. Και οι δομές αυτές ανήκουν στην ομάδα Η των K. & K., σύμφωνα με τους οποίους το **RMSD** που προέκυψε από sequence-based υπερθέσεις ήταν **μεγαλύτερο ή ίσο του έξι**.

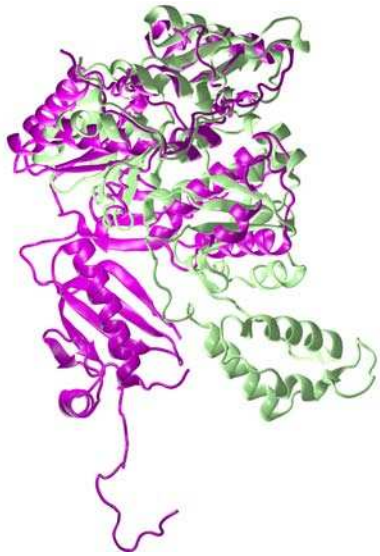
21.SUPERPOSITION of

1c1bB with **1mu2**A

CATH_code: 3.10.10.10 3.30.70.270
CATH_code: 3.10.10.10 3.30.420.10
3.30.70.270

RMSD:
6(268)

< 3 Å RMSD:
2(93)



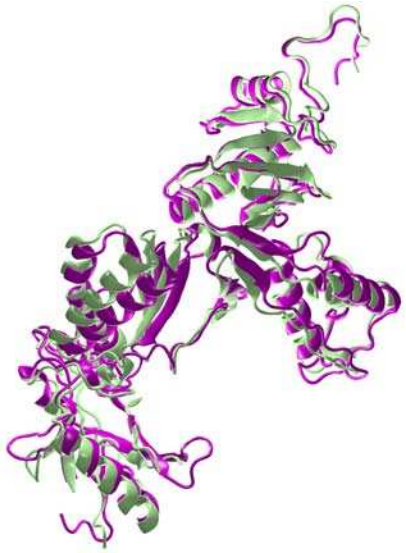
22.SUPERPOSITION of

1ep4A with **1jkh**B

CATH_code: 3.10.10.10 3.30.420.10
3.30.70.270
CATH_code: 3.10.10.10 3.30.70.270

RMSD:
3(422)

< 3 Å RMSD:
2(266)



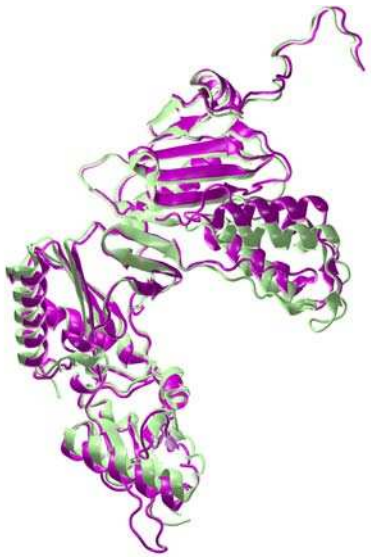
23.SUPERPOSITION of

1jlaA with **1vru**B

CATH_code: 3.10.10.10 3.30.420.10 3.30.70.270
CATH_code: 3.10.10.10 3.30.70.270

RMSD:
2.5(428)

< 3 Å RMSD:
1.5(356)



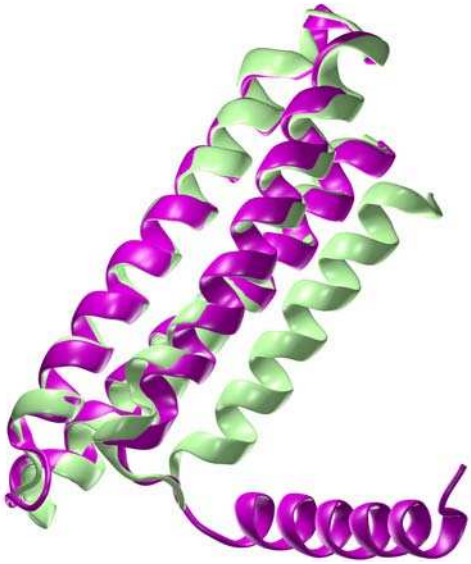
24.SUPERPOSITION of

1k40A with **1ow6**A

CATH_code: 1.20.120.330
CATH_code: 1.20.120.330 1.20.5.540

RMSD:
11(120)

< 3 Å RMSD:
0.8(95)



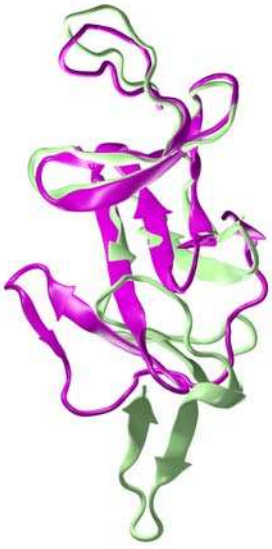
25.SUPERPOSITION of

1h8gA with **1oba**A

CATH_code: 2.10.270.10
CATH_code: 2.10.270.10 2.20.120.10 3.20.20.80

RMSD:
7(68)

< 3 Å RMSD:
1.5(43)



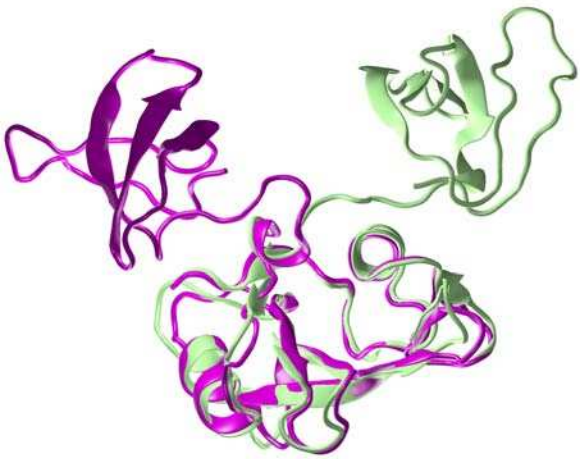
26.SUPERPOSITION of

1lckA with **2src**A

CATH_code: 2.30.30.40 3.30.505.10
CATH_code: 1.10.510.10 2.30.30.40
3.30.200.20 3.30.505.10

RMSD:
20(158)

< 3 Å RMSD:
1(104)



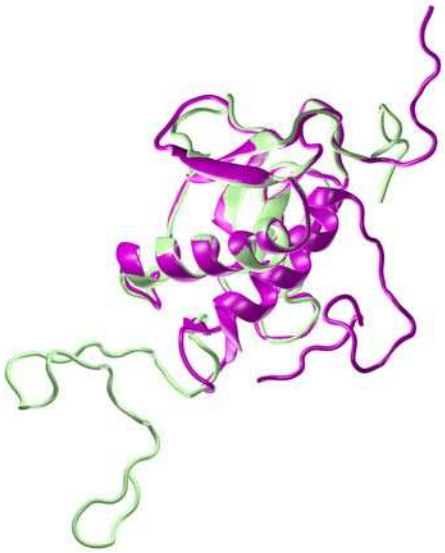
27.SUPERPOSITION of

1l6jA with **830c**A

CATH_code: 1.10.101.10 2.10.10.10
3.40.390.10
CATH_code: 3.40.390.10

RMSD:
13(134)

< 3 Å RMSD:
0.7(99)



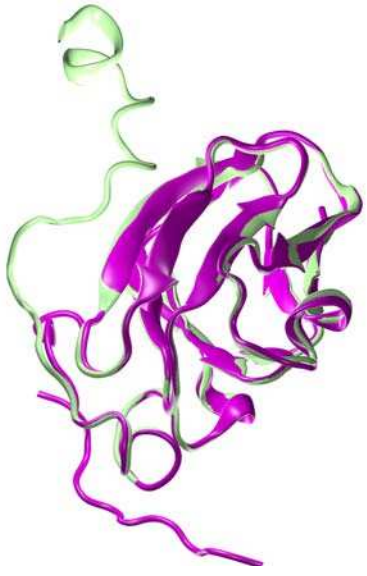
28.SUPERPOSITION of

1ehkB with **2cua**B

CATH_code: 1.20.1070.10 2.60.40.420
CATH_code: 2.60.40.420

RMSD:
10(132)

< 3 Å RMSD:
0.5(118)



Δομικές διαφορές!

-Και που καταλήξαμε;

Μόνο 10 ήταν τα αποτελέσματα που προέκυψαν από την περαιτέρω επεξεργασία της ομάδας των 267 ζευγών, οι οποίες θεωρητικά διέφεραν στις CATH καταχωρήσεις τους. Μόνο 10 εμφάνιζαν πραγματικά διαφορετικούς κωδικούς. Αποφασίσαμε λοιπόν να τα μαρκάρουμε με κίτρινο χρώμα για να τα ξεχωρίζουμε και να τα αναλύσουμε περαιτέρω και με αυτό τον τρόπο απέκτησαν το δικό τους όνομα: **the Yellow group**!

Ακόμη και σε αυτές τις εικόνες υπάρχουν περιπτώσεις όπου παρόμοια αλληλουχικά αλυσίδες εμφανίζουν όμοιες δομές. Βέβαια δε μπορούμε να παραλείψουμε το γεγονός ότι οι διαφορετικοί κωδικοί CATH αντιστοιχούν σε ολόκληρες αλυσίδες ή σε μεγαλύτερα τουλάχιστον τμήματα από τις δικές μας αλλληλουχικά όμοιες στοιχίσεις. Και αυτός είναι ο λόγος που δικαιολογεί γιατί κάποιες από αυτές τις δομικές στοιχίσεις μπορεί να μη διαφέρουν. Από την άλλη μεριά, φαίνεται ότι η σύγκριση των κωδικών CATH δεν απαντάει το ερώτημα που μας ενδιαφέρει.

Το πρώτο ζεύγος αλληλουχικά όμοιων πρωτεϊνικών αλυσίδων περιλαμβάνει τμήμα της καλσεκιεστρίνης του κουνελιού και μια κινάση αντίστοιχα. Οι δυο δομές δε σχετίζονται ούτε λειτουργικά ούτε δομικά μεταξύ τους παρά την υψηλή ομοιότητα στην αμινοξική τους αλληλουχία.

29.SUPERPOSITION of

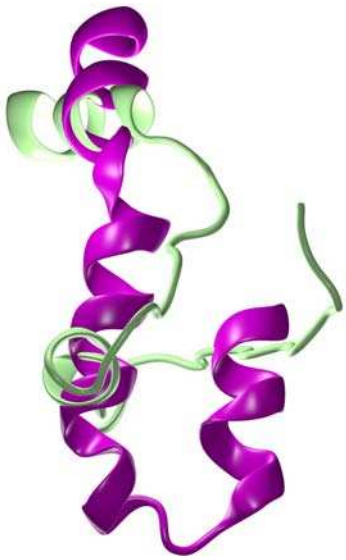
1a8yA with **1gkx**A

CATH_code: 3.40.30.10
CRYSTAL STRUCTURE OF CALSEQUESTRIN FROM RABBIT SKELETAL MUSCLE SARCOPLASMIC RETICULUM AT 2.4 A RESOLUTION

CATH_code: 1.20.140.20 3.30.565.10
BRANCHED-CHAIN ALPHA-KETOACID DEHYDROGENASE KINASE (BCK)

RMSD:
7(28)

< 3 Å RMSD:
1.5(10)



Ένα δεύτερο ζεύγος δομών που φαίνεται να διαφέρουν δομικά είναι μια επικράτεια θανάτου της πρωτεϊνης Pellex σε σύμπλοκο και η ίδια πρωτεϊνη ελεύθερη σε διαλύτη. Εφόσον η πρωτεϊνη είναι η ίδια, αυτό που προκαλεί την αλλαγή στη διαμόρφωσή της είναι πολύ πιθανόν η έκθεσή της σε 45% 2-methyl-2,4-pentanediol (MPD), καθώς υψηλές συγκεντρώσεις του MPD ή σχετικών διαλυτών μπορούν να αλλάξουν την τεταρτοταγή δομή ευαίσθητων πρωτεϊνών.

30.SUPERPOSITION of

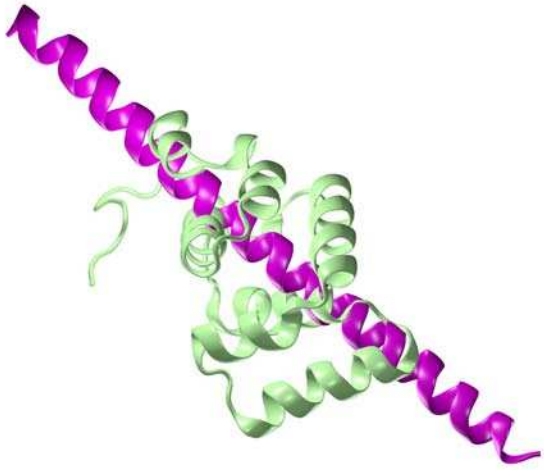
1d2zC with **1ik7**A

CATH_code: 1.10.533.10
THREE-DIMENSIONAL STRUCTURE OF A COMPLEX BETWEEN THE DEATH DOMAINS OF PELLE AND TUBE

CATH_code: 1.20.5.530
Crystal Structure of the Uncomplexed Pelle Death Domain

RMSD:
13(38)

< 3 Å RMSD:
1.5(11)



Η επόμενη περίπτωση εμπεριέχει πρωτεϊνικά κομμάτια δύο GTPασών. Οι πρωτεΐνες αυτές παρόλο που λειτουργικά είναι όμοιες, στα συγκεκριμένα τμήματα των αλυσίδων τους βλέπουμε να εμφανίζουν αλλαγή στο ένα άκρο τους εξαιτίας ενός βρόχου.

31.SUPERPOSITION of

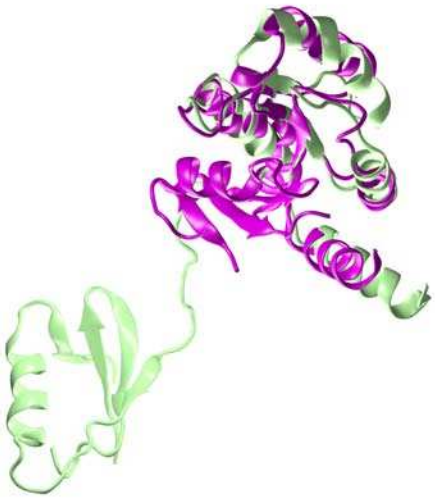
1mkyA with **1puj**A

CATH_code: 3.30.300.20 3.40.50.300
Structural Analysis of the Domain Interactions in Der, a Switch Protein Containing Two GTPase Domains

CATH_code: 1.10.1580.10 3.40.50.300
Structure of B. subtilis YlqF GTPase

RMSD:
22(142)

< 3 Å RMSD:
2(70)



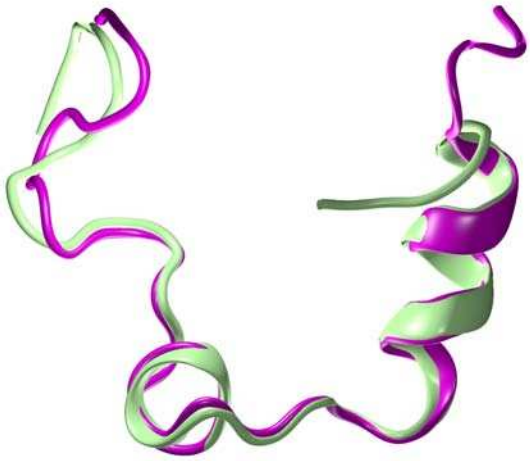
Από το σημείο αυτό και παρακάτω, αντίθετα με το γεγονός ότι οι κωδικοί CATH των πρωτεϊνών αυτών διαφέρουν, τα πρωτεϊνικά τμήματα εμφανίζουν μεγάλη δομική ομοιότητα, εκτός από μερικές αμελητέες διαφορές που εμφανίζουν στη διεύθυνση ή στο μήκος των βρόχων, οι οποίοι βρίσκονται στην αρχή και στο τέλος της αλληλουχιών αυτών. Άλλη μια απόδειξη λοιπόν ότι η σκέψη που είχαμε να αποφανθούμε για δομικές διαφορές αλληλουχικά όμοιων πρωτεϊνών συγκρίνοντας κωδικούς CATH είναι λάθος.

Στις δομές αυτές περιλαμβάνονται: η κρυσταλλική δομή της θρομβίνης σε σύμπλοκο με έναν αναστολέα ή με ένα πεπτίδιο [υπέρθεση 32], η δομή της πρωτεΐνης T-SNARE στη ζύμη [υπέρθεση 33], ένα διπλό μετάλλαγμα της εντεροτοξίνης με την τοξίνη της χολέρας [υπέρθεση 34], δύο πεπτιδάσες [υπέρθεση 35], Ν-τελικές επικράτειες πρωτεϊνών 2 διαφορετικών φάγων [υπέρθεση 36*], δύο διαφορετικές υδρογονάσες του Thermus thermophilus [υπέρθεση 37] και επικράτειες GTPασών [υπέρθεση 38].

* Στη συγκεκριμένη υπέρθεση με κόκκινες τελείες προσπαθούμε να απεικονίσουμε το κομμάτι που λείπει από την δεύτερη στη σειρά πρωτεΐνη (δηλαδή την πρωτεΐνη που φαίνεται με το μοβ χρώμα), αγνοώντας τον πραγματικό τρόπο σύνδεσης.

32.SUPERPOSITION of

1jwtA with **1ucy**L



CATH_code: 2.40.10.10
CRYSTAL STRUCTURE OF THROMBIN IN COMPLEX WITH A NOVEL BICYCLIC LACTAM INHIBITOR

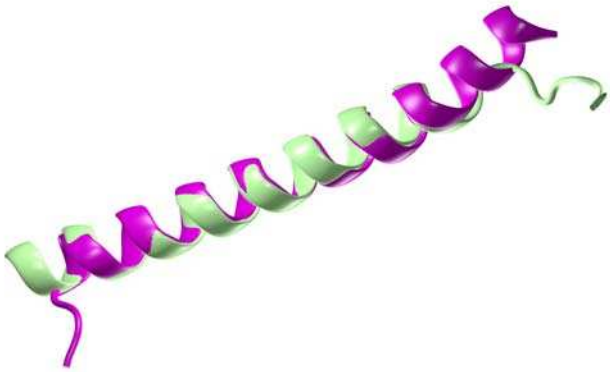
CATH_code: 4.10.140.10
THROMBIN COMPLEXED WITH FIBRINOPEPTIDE A ALPHA (RESIDUES 7-19). THREE COMPLEXES, ONE WITH EPSILON-THROMBIN AND TWO WITH ALPHA-THROMBIN

RMSD:
3.9(34)

< 3 Å RMSD:
0.9(30)

33.SUPERPOSITION of

1fioA with **1hvv**A



CATH_code: 1.20.58.70
CRYSTAL STRUCTURE OF YEAST T-SNARE PROTEIN SSO1

CATH_code: 1.20.5.110
SELF-ASSOCIATION OF THE H3 REGION OF SYNTAXIN 1A: IMPLICATIONS FOR SNARE COMPLEX ASSEMBLY

RMSD:
3.5(35)

< 3 Å RMSD:
0.8(30)

34.SUPERPOSITION of

1lt3A with **1xtc**C

CATH_code: 3.90.210.10
HEAT-LABILE ENTEROTOXIN DOUBLE MUTANT N40C/G166C

CATH_code: 1.20.5.240
CHOLERA TOXIN

RMSD:
1.5(38)

< 3 Å RMSD:
1(36)

35.SUPERPOSITION of

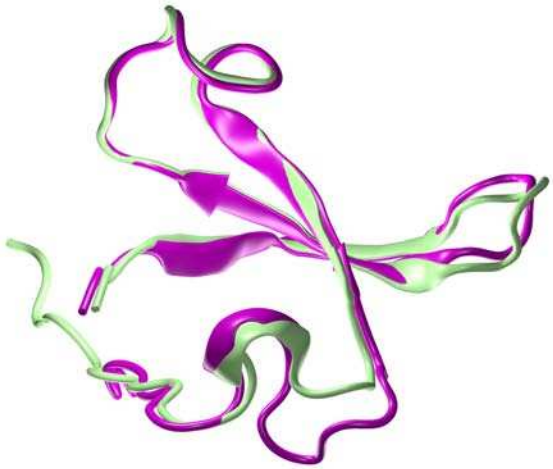
1k3bC with **1s4v**A

CATH_code: 2.40.50.170
Crystal Structure of Human Dipeptidyl Peptidase I (Cathepsin C)

CATH_code: 3.90.70.10
The 2.0 A crystal structure of the KDEL-tailed cysteine endopeptidase

RMSD:
1.5(56)

< 3 Å RMSD:
0.9(54)



36.SUPERPOSITION of

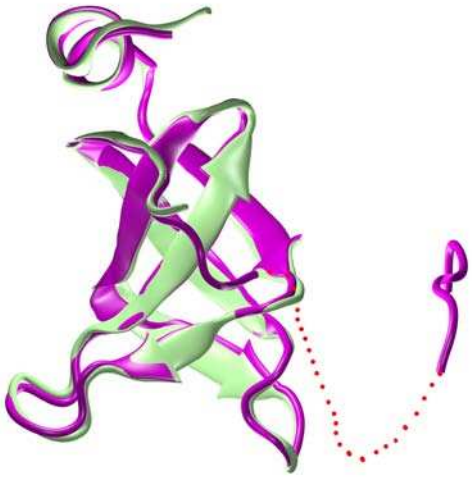
1tolA with **2g3p**A

CATH_code: 2.30.27.10 3.30.1150.10
FUSION OF N-TERMINAL DOMAIN OF THE MINOR COAT PROTEIN FROM GENE III IN PHAGE M13, AND C-TERMINAL DOMAIN OF E. COLI PROTEIN-TOLA

CATH_code: 2.30.27.10 3.90.450.1
STRUCTURE OF THE N-TERMINAL TWO DOMAINS OF THE INFECTIVITY PROTEIN G3P OF FILAMENTOUS PHAGE FD

RMSD:
0.7(64)

< 3 Å RMSD:
0.7(64



37.SUPERPOSITION of

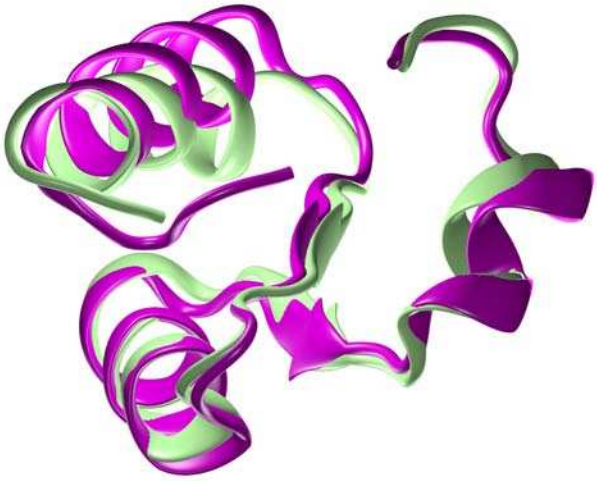
1j3vC with **1wxd**A

CATH_code: 1.10.1040.10
3.40.50.720
Structure of hydroxyisobutyrate dehydrogenase from thermus thermophilus HB8

CATH_code: 3.40.192.10
Crystal Structure of Shikimate 5-Dehydrogenase (AroE) from Thermus Thermophilus HB8

RMSD:
1.3(49)

< 3 Å RMSD:
1.3(48)



38.SUPERPOSITION of

1etsL with **1jwt**A

CATH_code: 4.10.140.10
REFINED 2.3 ANGSTROMS X-RAY CRYSTAL STRUCTURE OF BOVINE THROMBIN COMPLEXES FORMED WITH THE BENZAMIDINE AND ARGININE-BASED THROMBIN INHIBITORS NAPAP, 4-TAPAP AND MQPA

CATH_code: 2.40.10.10
CRYSTAL STRUCTURE OF THROMBIN IN COMPLEX WITH A NOVEL BICYCLIC LACTAM INHIBITOR

RMSD:
3.7(32)

< 3 Å RMSD:
0.7(27)

Επιλογικά, τα αλληλουχικά όμοια ζεύγη πρωτεϊνών της ομάδας G (υποσύνολό της αποτελεί η ομάδα H) των K.& K. τα οποία χρησιμοποιήσαμε για την πραγματοποίηση υπερθέσεων βάση της δομής των πρωτεϊνών δε μας έπεισαν ότι διαφέρουν τόσο πολύ στη δομή όσο δήλωνε το RMSD που προέκυψε από τις δικές τους sequence-based αναλύσεις. Με λίγα λόγια βλέπουμε ότι οι sequence-based υπερθέσεις δίνουν αρκετά αποτελέσματα που δεν ανταποκρίνονται στη δομική πραγματικότητα.

Παρόλα αυτά η ιδέα με την CATH δε μας έδωσε ξεκάθαρα αποτελέσματα, καθώς αποτελούσε μια πιο ποιοτική σύγκριση αποτελεσμάτων και δεν απαντούσε το ερώτημα που εμείς είχαμε θέσει. Ένας επιπλέον λόγος που μας βοήθησε να αντιληφθούμε ότι εργαζόμασταν προς τη λάθος κατεύθυνση ήταν ότι ακόμη και στην περίπτωση των 10 τελευταίων αποτελεσμάτων που έφεραν διαφορετικούς κωδικούς δεν είδαμε κάποια πραγματική διαφορά στο μοτίβο αναδίπλωσης της δευτεροταγούς δομής. Αυτό προφανώς είχε τη δική του εξήγηση, καθώς πολλές από τις περιπτώσεις που αναφέρθηκαν μέσα στα δέκα αποτελέσματα θα μπορούσαν να είναι μικρότερα κομμάτια πρωτεϊνών, τα οποία δε θα διέφεραν υποχρεωτικά στη δομή τους. Έτσι δημιουργήθηκε η ανάγκη για επιπλέον υπολογισμούς με σκοπό να απαντήσουμε το ερώτημα που πραγματικά μας ενδιαφέρει. Μια ποσοτική και πιο αντικειμενική ανάλυση γίνεται στο αμέσως επόμενο κεφάλαιο αυτής της εργασίας.

"The eye sees only what the mind
is prepared to comprehend."

Henri Bergson, French
Philosopher and Educator

Κεφάλαιο 3

Debugging...

"People get annoyed when you
try to debug them."

Larry Wall

3.1

CATH is not enough

Αν δε με απατά η μνήμη μου, η ερώτηση που προσπαθεί να απαντήσει αυτή η πτυχιακή εργασία σχετίζεται με την ύπαρξη πρωτεϊνικών αλυσίδων με μεγάλη ομοιότητα στην αλληλουχία οι οποίες υιοθετούν όμως διαφορετικές διαμορφώσεις. -Πώς λοιπόν φτάσαμε στο σημείο να ασχολούμαστε περισσότερο με τα αποτελέσματα που παρήγαγαν οι K&K παρά με την ουσία του ερωτήματος αυτού;

Η ιστορία έχει κάπως έτσι: Στην προσπάθειά μας να διαλευκάνουμε την παραπάνω υπόθεση χρησιμοποιήσαμε μια έτοιμη βάση δεδομένων (την ομάδα G) η οποία περιείχε ζεύγη αλυσίδων που εμφάνιζαν μεγαλύτερη του 50% ομοιότητα στην αλληλουχία και ταυτόχρονα παρουσίαζαν δομικές διαφορές (RMSD ≥ 3 Å). Αυτή η βάση δεδομένων ήταν αποτέλεσμα της εργασίας των K&K. Για λόγους που έχουν αναφερθεί στην εισαγωγή, διατηρούσαμε αμφιβολίες για την ορθότητα των αποτελεσμάτων τους όσον αφορά τον υπολογισμό του RMSD, καθώς επέλεξαν να χρησιμοποιήσουν sequence-based αλγόριθμους στην ερευνά τους. Όπως χαρακτηριστικά αναφέρουν και στο άρθρο τους:

" Sequence-based structure superpositioning will identify a larger number of structural dissimilar pairs than geometry-based structural alignments ".

Δε μπορούσαμε όμως να βασιστούμε σε αποτελέσματα τα οποία ήδη αμφισβητούσαμε. Αποφασίσαμε λοιπόν στο προηγούμενο κεφάλαιο να διασταυρώσουμε τα αποτελέσματα τους με δικούς μας υπολογισμούς. Θεωρήσαμε ότι ένας έξυπνος και γρήγορος τρόπος θα ήταν να συγκρίνουμε τους CATH κωδικούς που αντιστοιχούσαν σε κάθε ζεύγος καταχώρησης της ομάδας G. Ελπίζω να έχετε ένα *déjà vu* από το προηγούμενο κεφάλαιο καθώς έχω να δηλώσω ότι τα αποτελέσματα που προέκυψαν δεν ήταν καθόλου ξεκάθαρα. Αυτό που περιμέναμε να δούμε ήταν παρόμοιες δομές στις περιπτώσεις που οι κωδικοί ήταν πανομοιότυποι, όπως και ξεκάθαρα διαφορετικές δομές όταν οι κωδικοί διέφεραν, ειδικά στο 3ο ψηφίο τους (the yellow group). Αυτό όμως που διαπιστώσαμε ήταν ότι υπήρχαν περιπτώσεις με διαφορετικούς κωδικούς, οι οποίες είχαν παρόμοιες δομές, όπως επίσης και όμοιοι κωδικοί όπου οι δομές διέφεραν. Και παρόλο που οι περιπτώσεις αυτές δεν αποτελούσαν την πλειοψηφία των αποτελεσμάτων που αναλύσαμε περαιτέρω, ήταν αρκετές για να μας πείσουν ότι η ιδέα αυτή δεν ευδοκιμούσε και ότι δε θα μπορούσαμε να αποφύγουμε τον υπολογισμό του RMSD (βάσει geometry-based υπέρθεσης).

-Τι πήγε στραβά με την CATH και γιατί δε μπορούμε να αποφύγουμε τον υπολογισμό με το RMSD;

Τίποτα απολύτως! Η σύγκριση μεταξύ των κωδικών της CATH όπως και ο υπολογισμός του RMSD αποτελούν δύο διαφορετικές αναλύσεις και απαντούν σε διαφορετικό ερώτημα. Η CATH κοιτάει την τοπολογία και τα μοτίβα δευτεροταγούς δομής, ενώ το RMSD τοποθετεί δυο δομές τη μία πάνω στην άλλη με τέτοιο τρόπο ώστε να ελαχιστοποιεί τη μεταξύ τους απόσταση. Όταν για παράδειγμα έχουμε δύο διμερή με ίδια μονομερή, τότε στην CATH θα έχουμε όμοιους κωδικούς ενώ, αντίθετα, μια μικρή μετατόπιση ή περιστροφή του μονομερούς της μιας δομής θα μπορούσε να αυξήσει σημαντικά το RMSD. Μπορεί λοιπόν η ιδέα με την CATH να μη λειτούργησε με τον αναμενόμενο τρόπο, παρόλα αυτά ενίσχυσε τις ήδη υπάρχουσες αμφιβολίες μας για την ορθότητα των αποτελεσμάτων των K&K, καθώς τα RMSDs που έδιναν οι K&K για πολλές υπερθέσεις δεν αντιστοιχούσαν στην δομική πραγματικότητα. Επειδή έγινε αναγκαίο να επεξεργαστούμε πάλι την βάση δεδομένων των K&K και εφόσον το θέμα της πτυχιακής μας σχετίζεται με την ύπαρξη δομικών διαφορών σε όμοιες αλληλουχίες, επιλέξαμε να επεκτείνουμε λίγο την έρευνα μας και προς άλλη μία κατεύθυνση: να ελέγξουμε δηλαδή αν συγκεκριμένες αμινοξικές αλλαγές στις αλληλουχίες των στοιχίσεων οδηγούν σε σημαντικές μεταβολές στη δομή.

Στα πλαίσια λοιπόν του τρίτου και τελευταίου αυτού κεφαλαίου μας απασχολούν δύο κυρίως θέματα: (α) ο υπολογισμός του RMSD για αρκετά όμοιες αλληλουχίες (% identity ≥ 50 %) με geometry-based υπερθέσεις, ώστε να ελέγξουμε αν υπάρχουν περιπτώσεις όπου οι sequence-based υπερθέσεις αποτυγχάνουν και επίσης (β) γίνεται μια προσπάθεια εύρεσης αμινοξικών μεταλλαγών που προτιμούνται όταν οι δομές αλλάζουν.

'Always code as if the guy who ends up maintaining your code will
be a violent psychopath who knows where you live.'

Martin Golding

Lets put some code down! Για τις ανάγκες των παραπάνω ερωτημάτων γράφουμε λίγο κώδικα* ο οποίος δημιουργεί αρχείο που θα περιλαμβάνει: τους δύο κωδικούς των πρωτεϊνών από την ομάδα G των K&K, το RMSD που αυτοί δίνουν, το μήκος των αλληλουχιών που στοιχίθηκαν, τα RMSDs και τα μήκη στα οποία αντιστοιχούν οι δομικές στοιχίσεις που παράγονται από το πρόγραμμα που πραγματοποιεί την υπέρθεση, όπως και οι ίδιες αλληλουχίες των στοιχίσεων. Επίσης σε κάθε γραμμή του αρχείου θα περιλαμβάνεται το είδος και ο αριθμός των αμινοξικών αλλαγών για κάθε ζεύγος της στοίχισης. Για λόγους ευκολίας στο χειρισμό του αρχείου επιλέξαμε οι στοιχίσεις να τυπώνονται στην τελευταία στήλη. Ένα κομμάτι του αρχείου 'TMalign_results' που δημιουργείται φαίνεται στον Πίνακα 2.

* Ο συγκεκριμένος κώδικας βρίσκεται στο αμέσως επόμενο υποκεφάλαιο και αποτελεί την τελευταία και διορθωμένη από διάφορα bugs έκδοση.

Protein 1	Protein 2	Identity	K&K		TM_align		AC	AD	AE	...	Sequence 1	Sequence 2
			RMSD	Length	RMSD	Length						
102l	172l	97.6	3.34	165	2.92	154	0	1	0		MNIFEML...	MNCFEML...
11bg	1a5p	79.8	9.88	124	0.84	104	0	2	0		KESAAAK...	KETAAAK...
12e8	1ad0	57	4.76	223	3.96	208	0	0	1		EVQLQQS	EVQLLES
13pk	16pk	100	3.55	415	3.37	410	0	0	0		EKKSINE	EKKSINE
1a05	1g2u	53.4	3.06	352	2.84	343	0	0	2		KIAIFAG	KVAVLPG
1a15	1qg7	98.5	3.28	67	1.96	61	0	0	0		KPVSLSY	KPVSLSY
1a1m	1r3h	55.2	3.59	277	2.37	230	0	0	1		GSHSMRY	GSHSLRY
1a1q	1a1q	100	6.92	189	1.81	156	0	0	0		APITAYS	APITAYS
1a28	1nhz	54.4	4.86	252	2.04	227	0	0	0		QLIPPLI	QLTPTLV
1a2k	1wa5	100	3.93	176	1.8	158	0	0	0		MAAQGEP	MAAQGEP
1a2x	1avs	95.4	4.43	87	2.93	70	0	0	0		TDQQAEA	TDQQAEA
1a3q	1nfk	55.8	3.09	312	2.57	277	0	0	0		GPYLVIV	GPYLQIL
1a40	1oib	99.4	3.02	321	2.89	318	0	0	0		EASLTGA	EASLTGA
1a49	1a49	100	3.73	530	3.41	509	0	0	0		SKSHSEA	SKSHSEA
1a4r	1dpf	54	3.32	174	1.79	167	0	0	0		KCVVVGD	KLVIVGD
1a5a	1a5s	99.6	3.06	268	1.3	250	0	0	0		MERYENL	MERYENL
1a75	1bu3	88.9	3.23	108	1.4	103	0	0	0		AFAGILA	AFSGILA
1a7c	1lj5	99.7	5.32	379	1.93	357	0	0	0		VHHPPSY	VHHPPSY
1a8e	1bp5	100	6.7	329	3.93	260	0	0	0		DKTVRWC	DKTVRWC
1a8y	1gkx	50	8.99	42	2.39	22	0	0	0		LVEFLLD	LVRQLLD
1ado	1ado	100	3.63	363	1.43	353	0	0	0		PHSHPAL	PHSHPAL
1ae2	2gn5	98.9	5.26	87	1.5	81	0	0	0		MIKVEIK	MIKVEIK

Πίνακας 2: Κομμάτι του αρχείου 'Tmalign_results' που περιέχει τους κωδικούς των πρωτεϊνών, το ποσοστό ομοιότητας, RMSDs και μήκη από τους K&K και από το TMalign, το είδος και ο αριθμός των μεταλλαγών μεταξύ των πρωτεϊνών που στοιχίθηκαν και οι αντίστοιχες αλληλουχίες.

Σημείωση: Επειδή οι K&K είχαν ομαδοποιήσει τα δεδομένα τους βάσει λειτουργίας, για να είναι αρκετά αντιπροσωπευτικός ο παραπάνω πίνακας του συνόλου των αποτελεσμάτων, επιλέξαμε με τον ακόλουθο τρόπο μία καταχώρηση για κάθε υπομάδα (401 σε σύνολο) της ομάδας G. Κάθε τέτοια υποομάδα σηματοδοτείται από τη γραμμή '<multiple_blast_align_local.php' μέσα στο αρχείο τους. Έτσι, λοιπόν, με τη βοήθεια της εντολής grep* επιλέξαμε να αποθηκεύουμε σε ένα νέο αρχείο κάθε 1 καταχώρηση της ομάδας G μετά από αυτή τη γραμμή και στη συνέχεια αφαιρέσαμε όλες τις γραμμές '<multiple_blast_align_local.php', ώστε να παραμείνουν στο νέο αρχείο μόνο οι καταχωρήσεις των πρωτεϊνών.

*

```
grep -A 1 '<multiple_blast_align_local.php' | grep -v ' '<multiple_blast_align_local.php'

| grep -v '-' > 0RES
```

from grep's manual page:

-A NUM, --after-context=NUM

Print NUM lines of trailing context after matching lines. Places a line containing a group separator (--) between contiguous groups of matches. With the -o or –only-matching option, this has no effect and a warning is given.

-v, --invert-match

Invert the sense of matching, to select non-matching lines. (-v is specified by POSIX.)

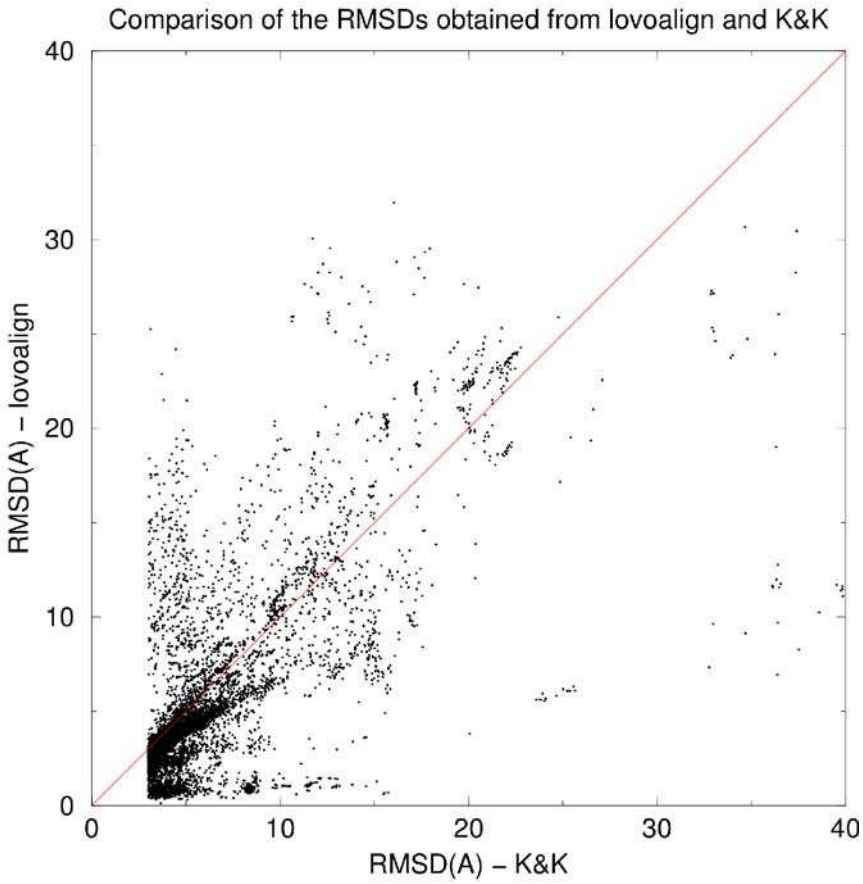
'However beautiful the strategy,
you should occasionally look at
the results'

Sir Winston Churchill

3.2

A LOVOALIGN BUG?

Μια προσεκτική παρατήρηση των αποτελεσμάτων που έχει παράγει το πρόγραμμά μας μέχρι στιγμής μας προϊδεάζει για τυχόν λάθη(bugs) που μπορεί να υπάρχουν. Εκτός από κάποια αμελητέα προβλήματα που οφειλόταν στη δική μας ασυνέπεια και τα οποία εύκολα αντιμετωπίστηκαν, ερχόμαστε αντιμέτωποι με πολύ υψηλά RMSDs που δεν ανταποκρίνονται στη δομική πραγματικότητα. -Πως αντιληφθήκαμε την ύπαρξη τους; Καθώς το πρόγραμμά μας έτρεχε, όταν πλέον είχε μαζευτεί ένας ικανοποιητικός αριθμός αποτελεσμάτων αποφασίσαμε ώστε να τις συγκρίνουμε ποιοτικά. με τη δημιουργία ενός scatterplot*, το οποίο θα περιλαμβάνει στους δυο άξονές του τα νούμερα που προέκυψαν από το lovoalign και από τους K&K αντίστοιχα (Εικόνα 6).



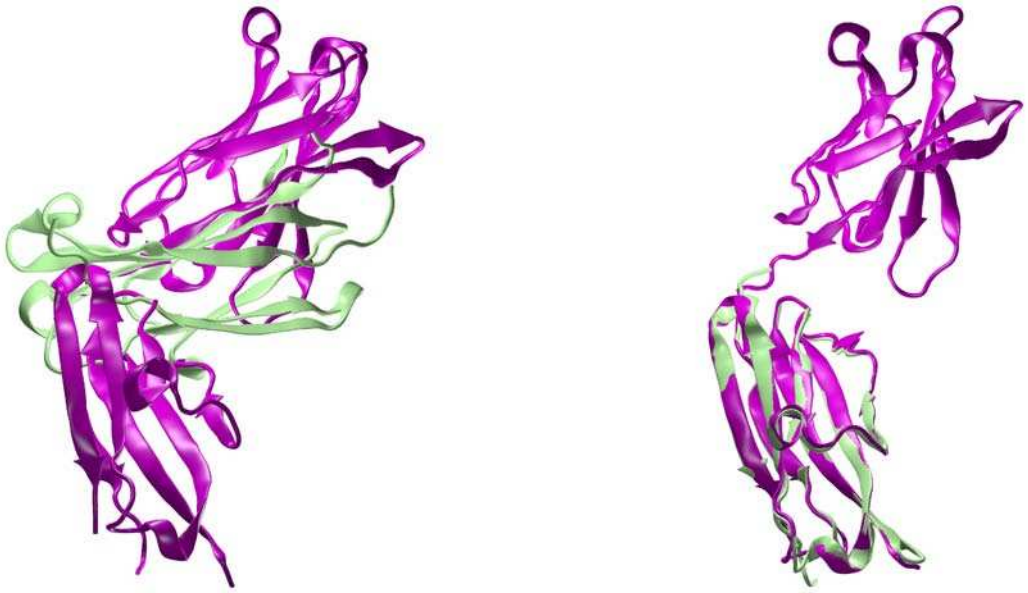
Εικόνα 6: Σύγκριση μεταξύ των RMSDs που προκύπτουν από το lovoalign και τους K&K. Η κόκκινη γραμμή αποτελεί τη διαγώνιο του γραφήματος. Τα σημεία που βρίσκονται πάνω από τη διαγώνιο αντιστοιχούν στις υπερθέσεις όπου το lovoalign έχει δώσει μεγαλύτερο RMSD από τους K&K.

* Το scatterplot το δημιουργούμε με τη χρήση του Xmgr (https://computing.llnl.gov/vis/xmgr.shtml).

Σύμφωνα με την ιδέα που συζητάμε, οι δομικές διαφορές που αναμένουμε από έναν geometry-based αλγόριθμο (lovoalign) θα έπρεπε να είναι σημαντικά μικρότερες ή έστω ίδιες με αυτές των K&K. Παρόλα αυτά στο διάγραμμα που δημιουργήθηκε πολλές καταχωρήσεις βρισκόταν στο μισό διάγραμμα που βρίσκεται πάνω από τη διαγώνιο, το οποίο αντιστοιχεί σε μεγαλύτερα RMSDs για το lovoalign και όχι για τους K&K και απαιτεί μεγαλύτερη ανάλυση ώστε να ελέγξουμε αν ανταποκρίνεται σωστά στη δομική πραγματικότητα.

Για να έχουμε μία πιο ρεαλιστική εικόνα του τι συμβαίνει, ταξινομούμε κατά φθίνουσα σειρά τα αποτελέσματά μας σύμφωνα με τη στήλη του RMSD που παρήγαγε το lovoalign. Μετά από αυτό το "sorting" πολύ μεγάλα RMSDs προέκυψαν (πιθανόν λανθασμένα) στις τελευταίες γραμμές του αρχείου, τα οποία είναι επίσης και πολύ μεγαλύτερα από τα RMSDs που δίνουν οι K&K. Ελέγχουμε με το χέρι κάποια από αυτά τα αποτελέσματα, όπως την καταχώρηση 1i7zB-1pg7H, και συνειδητοποιήσαμε ότι το lovoalign δεν παράγει την υπέρθεση που θα αναμέναμε (εικόνα της υπέρθεσης φαίνεται παρακάτω – Εικόνα 7). Πιο συγκεκριμένα, ενώ η πρωτεΐνη 1pg7 είναι ένα διμερές και η 1i7z ένα μονομερές όμοιο με την μία από τις δύο υπομονάδες του διμερούς, η υπέρθεση πραγματοποιείται εντελώς λανθασμένα (Εικόνα 7 αριστερά) προκαλώντας συντριπτική αύξηση του RMSD σε 11.67. Αντιπαραβάλουμε την εικόνα αυτή με την υπέρθεση που έδωσε το TMalign (η επιλογή του συγκεκριμένου προγράμματος αναλύεται παρακάτω).

SUPERPOSED WITH LOVOALIGN **SUPERPOSED WITH TM_align**



Εικόνα 7: Υπέρθεση της 1i7zA με την 1pg7L. Αριστερά υπέρθεση με χρήση του lovoalign ενώ δεξιά με τη βοήθεια του TMalign.

Η αλήθεια είναι ότι δεν μπορέσαμε να εντοπίσουμε για πιο λόγο το lovoalign αποτυγχάνει. Υπάρχουν παρόμοιες περιπτώσεις μονομερούς – διμερούς όπου η στοίχιση των δομών πραγματοποιείται με το βέλτιστο τρόπο. Συμπεράναμε λοιπόν ότι το lovoalign πιθανότατα αντιμετωπίζει κάποιο πρόβλημα με ορισμένα PDB αρχεία σε περιπτώσεις όπου οι αλληλουχίες εμφανίζουν σημαντικές διαφορές στο μήκος τους και στρέψαμε τις βλέψεις μας σε ένα νέο πρόγραμμα δομικής υπέρθεσης, το TM_align. Ευτυχώς οι αλλαγές στο εκτελέσιμο πρόγραμμα που χρησιμοποιούμε περιορίστηκαν στην αλλαγή των υπορουτίνων που έτρεχαν το lovoalign και επεξεργαζόταν την έξοδο που αυτό παρήγαγε, σε υπορουτίνες κατάλληλα διαμορφωμένες για το TMalign. Ο νεότερος κώδικας, όπως και η αλλαγή αυτή βρίσκονται στις τελευταίες σελίδες του υποκεφαλαίου αυτού [Appendix A].

Μια αναλυτική περιγραφή της κάθε υπορουτίνας του προγράμματος υπάρχει στα αντίστοιχα σχόλια μέσα στον κώδικα, αναφέρουμε όμως και εδώ συνοπτικά τις βασικές ιδέες:

Αρχικά συνδεόμαστε με την PDB και διατηρούμε ανοιχτή τη σύνδεση, ώστε να μεταφέρουμε όσα αρχεία είναι απαραίτητο. Για κάθε ζεύγος πρωτεϊνών που έχουν στην βάση δεδομένων τους (ομάδα G) οι K&K, τρέχουμε bl2seq για να πάρουμε τις αλληλουχικές στοιχίσεις. Επίσης κατεβάζουμε τα αντίστοιχα αρχεία PDB και τα αποσυμπιέζουμε.

Ένα πολύ δύσκολο κομμάτι, αφορούσε το πως θα επεξεργαστούμε τα αρχεία της PDB, ώστε να κόψουμε τα σωστά τμήματα που αντιστοιχούσαν στις στοιχίσεις από το BLAST (bl2seq). Η ιδέα είναι να χρησιμοποιήσουμε την γραμμή DBREF από τα PDB αρχεία ώστε να πάρουμε τα αριθμητικά όρια ολόκληρης της αλληλουχίας στο αρχείο PDB και να κόψουμε άμεσα τις ενδιάμεσες γραμμές που μας ενδιαφέρουν. Ο ορισμός του εναρκτήριου αριθμού καταλοίπου στα αρχεία PDB είναι εντελώς υποκειμενική, οπότε χρειαζόμαστε τα πραγματικά όρια κάθε φορά για να κόψουμε απευθείας τις γραμμές που θέλουμε. Δυστυχώς, παρατηρήσαμε ότι σε ορισμένα αρχεία υπάρχουν γραμμές SEQADV*, οι οποίες χαλάνε την αριθμητική αντιστοίχιση και μπορεί να μας οδηγήσουν σε απώλειες καταλοίπων. Διορθώνουμε όμως τον κώδικα με τέτοιο τρόπο ώστε η ύπαρξη SEQADV να μην επηρεάζει τα αποτελέσματα. Επομένως, οι μοναδικές περιπτώσεις στις οποίες το πρόγραμμα μας αποτυγχάνει να παράγει αποτελέσματα είναι όταν δεν υπάρχει καθόλου αναφορά για DBREF ή στην περίπτωση που η συγκεκριμένη πρωτεϊνη έχει αντικατασταθεί από νεότερη καταχώρηση στην PDB. Και στις δυο αυτές περιπτώσεις τυπώνεται σε αρχείο LOG, το όνομα της πρωτεΐνης και ο λόγος για τον οποίο δε βρέθηκε, ώστε εν κατακλείδι να μπορούμε να ελέγξουμε αν το πρόγραμμά μας ορθώς παράγει τα αποτελέσματα αυτά.

Με τη βοήθεια του TMalign που έχουμε ήδη αναφέρει υπερθέτουμε τα κομμένα αρχεία που δημιουργήσαμε προηγουμένως. Στη συνέχεια από την έξοδο που παράγει κρατάμε το RMSD και το μήκος στο οποίο αντιστοιχεί. Ακολουθεί η καταμέτρηση του αριθμού των διαφόρων μεταλλαγών και τέλος η εκτύπωση όσων έχουν αναφερθεί σε ένα αρχείο με όνομα 'TMalign_results'.

* SEQADV: Η καταγραφή SEQADV αναγνωρίζει διαφωνίες μεταξύ της πληροφορίας σχετικά με την αλληλουχία στις καταχωρήσεις ATOM της PDB και της πληροφορίας που δίνεται στο DBREF. Σημειώνεται επίσης ότι οι συγκεκριμένη καταγραφή έχει σχεδιαστεί για να αναγνωρίζει διαφορές και όχι λάθη. Δεν υπάρχει κάποια υπόθεση για το ποια βάση περιέχει τα σωστά δεδομένα. Το PDB αρχείο ίσως περιλαμβάνει στην εισαγωγή καταχωρήσεις REMARK, οι οποίες αντικατοπτρίζουν την άποψη του καταθέτη για το ποια βάση δεδομένων θεωρεί αυτός σωστή.

Appendix A

```
#!/usr/bin/perl -w

##########################################################################################################################################################
#                                                                                                                                                        #
#    This program downloads and cuts the PDB files that correspond to the pairs                                                                          #
#    given in K&K's dataset in order to superpose their structures based on the                                                                          #
#    proteins alignments. It also calculates the number and the kind of mutations                                                                        #
#    between these alignments. Finally, it prepairs a new file which includes: the                                                                       #
#    protein codes of each pair, RMSD and length given from K&K and TMalign and                                                                          #
#    an amino acid mutation array counting the mutations presented in the current                                                                        #
#    dataset.                                                                                                                                            #
#                                                                                                                                                        #
##########################################################################################################################################################


#
# FTP stands for File Transfer Protocol
# It allows files to be sent to or fetched from a server
#
# GUNZIP is provided to carry out "one-shot" uncompression between buffers and/or files.
#

use Net::FTP;
use Net::FTP::A;
use IO::Socket::INET;
use IO::Uncompress::Gunzip qw(gunzip $GunzipError) ;
use warnings;

#
# Initialize variables
#

my $code1 = "";
my $chain1 = "";
my $code2 = "";
my $chain2 = "";
my $KK_RMSD = 0;

#
# open 2D_matrix for writing
#

open ( SUM, ">>2D_matrix" ) || die ("Could not open SUM for writing");




#
```

```
# connect to the pdb server
# give password (whatever password is accepted --> entry is free)
# ftp will not be closed untill EOF
#


$ftp = Net::FTP->new("198.202.122.61", Debug => 0, Passive => 1, Timeout => 720) or die "Cannot connect to ftp.wwpdb.org: $@";
$ftp->login("anonymous",'-anonymous@') or die "Cannot login ", $ftp->message;



#
# prepare a hash which includes
# all possible capital letter combinations
#

%aadif = ();

print SUM "                                                  ";

for ( $i=65; $i<=90; $i++ )
    {
       for ( $j=$i+1; $j<=90; $j++ )
           {
              $a_a = chr($i).chr($j);
              printf SUM ("%5s", $a_a);
           }
    }

print SUM "\n";


while ( $line = <STDIN> )
    {
          #
          # keep two codes and the RMSD from K&K's database
          # save them as upper case characters in the following
          # scalars
          #

          if ( $line =~ /^(....)(.) (....)(.) .+ .+ .+ .+ .+ .+ .+ (.+) .+ .+$/ )
              {
                $code1 = $1;
                $uc_code1 = uc($code1);
                $chain1 = $2;
                $code2 = $3;
                $uc_code2 = uc($code2);
                $chain2 = $4;
                $KK_RMSD = $5;
                $have_PANIC = 0;

                #
                # wget Fasta files for these specific codes
                #
```

```
 `wget --quiet -O fasta_1 "http://www.pdb.org/pdb/download/downloadFile.do?fileFormat=FASTA&compression=NO&structureId=$uc_code1"`;
 `wget --quiet -O fasta_2 "http://www.pdb.org/pdb/download/downloadFile.do?fileFormat=FASTA&compression=NO&structureId=$uc_code2"`;

 #
 # run fasta subroutines
 # to keep the right chains from the fasta files
 #

 my $chain_def1 = &fasta($uc_code1, $chain1, 1);
 my $chain_def2 = &fasta($uc_code2, $chain2, 2);

 #
 # Run bl2seq
 # $query, $Sbjct and $blast_length
 # are returned to the @seqs
 #

 my @seqs = &bl2seq();

 #
 # transfer the required files in our server
 #

 $ftp->get("pub/pdb/data/structures/all/pdb/pdb$code1.ent.gz") or ( print("FTP ERROR:$code1\n") && next );
 $ftp->get("pub/pdb/data/structures/all/pdb/pdb$code2.ent.gz") or ( print("FTP ERROR:$code2\n") && next );


 #
 # unzip $input file --> "pdb$code1.ent.gz"
 # and rename it as $output file --> "pdb$code1.ent"
 # or print that it could not be unzipped
 # (respectively for $input2-$output2)
 #

 $input1="pdb$code1.ent.gz";
 $output1="pdb$code1.ent";
 gunzip $input1 => $output1 or die "gunzip failed: $GunzipError\n";

 $input2="pdb$code2.ent.gz";
 $output2="pdb$code2.ent";
 gunzip $input2 => $output2 or die "gunzip failed: $GunzipError\n";

 #
 # Run the keep lines subroutine
 # to keep the pdb lines respectively
 # to the bl2seq hit
 #

@SEQ_ADV1 = &keep_lines( $uc_code1, $chain_def1, $seqs[0], $seqs[2], 1);
@SEQ_ADV2 = &keep_lines( $uc_code2, $chain_def2, $seqs[1], $seqs[2], 2);
```

```
#
# delete pdb.ent.gz and pdb.ent files
#

unlink ($input1) or warn ("Could not erase file");
unlink ($input2);

unlink ($output1) or warn ("Could not erase file");
unlink ($output2);


#
# PANIC checks if DBREF exist in the PDB file
# in order to continue with the superposition
#

if ($have_PANIC == 0)
{


&superpose_the_PDBs($code1, $code2);

@RMSDs = &keep_RMSD($code1, $code2);


#
# create a two dimentional MATRIX
# with amino acid changes RMSDs and
# the respective structures
#

$id = 0;
$seq1 = "";
$seq2 = "";


#
# initialize all matrix elements to zero
#

for ( $i=65; $i<=90; $i++ )
    {
        for ( $j=$i+1; $j<=90; $j++ )
            {
                $a_a = chr($i).chr($j);
                $aadif{ "$a_a" } = 0;
            }
    }

#
# compare each residue between query and Subject
#
```

```
              for ( $x=0; $x < $seqs[2] ; $x++ )
                  {
                     $seq1 = substr($seqs[0], $x, 1);
                     $seq2 = substr($seqs[1], $x, 1);

                     if ( ($seq1 ne $seq2) && ($seq1 ne '-' && $seq2 ne '-') && ($seq1 ne 'X' && $seq2 ne 'X'))
                         {

                            #
                            # whether there is a aminoacid change AG or GA
                            # increase only AG by one
                            #

                            if ( $seq2 gt $seq1 )
                                 {
                                   $aadif{$seq1.$seq2}++;
                                 }
                            else
                                {
                                   $aadif{$seq2.$seq1}++;
                                }
                         }

                     if ( $seq1 eq $seq2 && ($seq1 ne '-' && $seq2 ne '-') && ($seq1 ne 'X' && $seq2 ne 'X'))
                         {
                            $id++;
                         }
                  }


              printf SUM ("%s %s %5.1f %5.2f %5d %5.2f %4d %5.2f %4d %d %d", $code1, $code2, (100.0*$id)/$seqs[2], $KK_RMSD, $seqs[2], @RMSDs, $SEQ_ADV1[0],
$SEQ_ADV2[0]);
              for ( $i=65; $i<=90; $i++ )
                  {
                     for ( $j=$i+1; $j<=90; $j++ )
                         {
                            $a_a = chr($i).chr($j);
                            printf SUM ("%5d", $aadif{"$a_a"});
                         }
                  }

              print SUM (" $seqs[0] $seqs[1]");
              print SUM "\n";

              }
              unlink ("$code1-1.pdb");
              unlink ("$code2-2.pdb");

           }
    }
```

```
$ftp->quit;
close(SUM);




#--------------------#
#--- SUBROUTINES ---#
#--------------------#



############################
#
#
#   fasta subroutine
#
#
############################


sub fasta
    {
      $/='>';

      $code_sub = $_[0];
      $chain_sub = $_[1];
      $identifier = $_[2];



      open ( FILE, "<fasta_$identifier" ) || die ("Could not open fasta_$identifier");
      open ( NEWFILE, ">new$identifier.fasta" ) || die ("Could not open new$identifier.fasta for writing");

      #
      # when K&K do not give chain identifier ("_")
      # save chain identiffrom the fasta file --> in a new file
      # (in that case only one chain exist)
      #
      # else write the sequence of the correct chain
      # in a new file
      #

      while ( $chains = <FILE> )
          {
           if ( $chain_sub eq "_")
               {
                  if ( $chains =~ /^$code_sub\:(.)\|PDBID\|CHAIN\|SEQUENCE/)
                      {
                      $chain_defined = $1;
                      chomp($chains);
                      print  NEWFILE ">", $chains, "\n";
```

```
                      ######}#
              }#
            elsif ( $chains =~ /^$code_sub\:($chain_sub)\|PDBID\|CHAIN\|SEQUENCE/)
                {
                   $chain_defined = $1;
                  chomp($chains);
                  print  NEWFILE ">", $chains, "\n";
                }
          }
        close(FILE);
        close(NEWFILE);
        $/="\n";
       return ($chain_defined);
   }



############################
#
#
#   bl2seq subroutine
#
#
############################


sub bl2seq
    {


       #
       # run bl2seq and read the results file
       # check whether no hits found
       # if no hits are found, find the best alignment (first one)
       # and keep its alignment
       #

       $query = "";
       $Sbjct = "";
       $num_of_hits = 0;

       `bl2seq -p blastp -F F -i new1.fasta -j new2.fasta -o bl2seq.out`;

       open ( BL2S, "bl2seq.out" ) || die ("Could not open file BL2S");

       while ( $bl2s = <BL2S> )
           {
              if ( $bl2s =~/\s\*\*\*\*\* No hits found \*\*\*\*\*\*/ )
                  {
                     print " ***** No hits found ******\n";
                  }

              if ( $bl2s =~ / Score/ )
```

```
                    {
                        $num_of_hits++;
                    }

                if  ( $num_of_hits == 1 )
                    {
                        if ( $bl2s =~/ Identities = \d+\/(\d+)\s/)
                            {
                                $blast_length = $1;
                            }
                        if ( $bl2s =~/Query:\s*\d+\s*([GVALISTCFMNPQWYKERDHX-]+)/ )
                            {
                                $query = $query.$1;
                            }
                        if ( $bl2s =~/Sbjct:\s*\d+\s*([GVALISTCFMNPQWYKERDHX-]+)/ )
                            {
                                $Sbjct = $Sbjct.$1;
                            }
                    }
            }

        $que = length($query);
        $Sub = length($Sbjct);

        if ( $blast_length != $que || $blast_length != $Sub )
            {
                print "ERROR! --> different blast length!\n";
            }

        close(BL2S);
        return($query, $Sbjct, $blast_length);
    }


############################
#
#
#   keep lines subroutine
#
#
############################


sub keep_lines
    {
        $code = $_[0];
        $_code = lc($code);
        $chain = $_[1];
        $subseq = $_[2];
        $identity = $_[4];
        $seq = "";

        open ( FASTA, "<new$identity.fasta" ) || die ("Could not open FASTA for reading");
```

```
open ( PDB, "<pdb$_code.ent" ) || die ("Could not open PDB for reading");
open ( NEWPDB, ">>$_code-$identity.pdb" ) || die ("Could not open NEWPDB for writing");


#
# save in a scalar the whole fasta sequence
#


while ( $fasta_line = <FASTA> )
    {
       if ( $fasta_line =~ /^>/ )
           {
              next;
           }
       chomp($fasta_line);
       $seq = $seq.$fasta_line;
    }


 $sub_seq= "";
@each_residue = split("", $subseq);

#
# for each amino of the @each_residue array
# unless $amino = "-"
# save amino in @unique_residue array
#


foreach $amino ( @each_residue )
    {
       unless ( $amino eq "-" )
           {
              $sub_seq = $sub_seq.$amino;
           }
    }

$pure_length = 0;
$SEQADV=0;
$sumSEQADV = 0;
$tot_dbref = 0;

#
# while reading PDB lines
# keep the two numbers given in DBREF line
# and cut the lines in the pdb between
# these residues ( including the first & the last )
#


while ( $PDB_line = <PDB> )
    {
```

```
        if ( $PDB_line =~ /^DBREF  $code $chain\s*(\d+)\s*(\d+)/ )
            {
                $db1[ $tot_dbref ] = $1;
                $db2[ $tot_dbref ] = $2;
                $tot_dbref++;
             }
          }
seek( PDB, 0, 0 );

if ( $tot_dbref == 0 )
    {
        print "PANIC $code$chain !!\n\n";
        $have_PANIC = 1;
    }

 if ( $tot_dbref == 1 )
     {
         $first_res = $db1[0];
     }

 #
 # just in case that there are more than one DBREF
 # calculate pure length by subtracking
 # all first and last residues given
 #

 if ( $tot_dbref > 1 )
   {
         for ( $a=0; $a < $tot_dbref; $a++ )
             {
                 $first_res = $db1[0];
                 $pure_length = $pure_length + ($db1[$a] - $db2[$a] + 1 );
             }
   }

 if ( $have_PANIC == 0 )
 {

 #
# $index shows in what place the subseq exists
# whereas $begin gives the accurate residue number in the PDB
# where the subseq begins
#

$index = index ($seq, $sub_seq, 0 );    # +1
$begin = $index + $first_res;      # -1
$last = length($sub_seq) + ($begin - 1);


while ( $PDB_line = <PDB> )
    {
        if ( $PDB_line =~ /^SEQADV/ )
            {
```

```
                        $SEQADV=1;
                       $sumSEQADV++;
                   }

               if ( $PDB_line =~ /^(ATOM|HETATM)\s*\d+\s+\w+\d*\s*\w+\s+$chain\s*(\d+)[\s|\w+]\s/ )
                   {
                       if ( $2 >= $begin && $2 <= $last )
                           {
                              print NEWPDB $PDB_line;
                           }
                   }
           }
       print NEWPDB "END";


       }

       close(FASTA);
       close(PDB);
       close(NEWPDB);
       return( $SEQADV, $sumSEQADV);
    }


############################
#
#
#   superpose_the_PDBs subroutine
#
#
############################


sub superpose_the_PDBs
    {

       #
       # run TMalign to get the superposition of the PDBs
       # save results in TMalign_results
       my $code_1 = $_[0];
       my $code_2 = $_[1];


       open ( TMALIGN, ">TMalign_results" ) or die ("Cannot open TMALIGN");

       #
       # run TMalign for the superposition
       #

       $running_tmalign = `TMalign $code_1-1.pdb $code_2-2.pdb`;

       print TMALIGN "$running_tmalign";
```

```
        close (TMALIGN);

#       unlink ("SEMI-SUPERPOSITIONS/$code1-1_$code2-2.pdb");
    }


############################
#
#
#   keep_RMSD subroutine
#
#
############################


#
# from TMalign results
# keep the RMSDs and the structure alignments
#

sub keep_RMSD
    {
       my $code_1 = $_[0];
       my $code_2 = $_[1];

       open ( TMALIGNTEST, "TMalign_results" ) || die ("Could not open file");

       while ( $TMline = <TMALIGNTEST> )
           {
              if ( $TMline =~ /^Aligned length=\s+(\d+), RMSD=\s+(.+), TM/)
                  {
                     $coverage = $1;
                     $RMSD = $2;
                  }
                     $closer_RMSD = 0;
                     $coverage_closer = 0.0;
           }

       @alignments = ( $RMSD, $coverage, $closer_RMSD, $coverage_closer );
       return( @alignments );
     }
```

Στη σελίδα αυτή θα παραθέσουμε τις δύο τελευταίες υπορουτίνες που επεξεργάζονταν το lovoalign πριν αυτό αντικατασταθεί από τις υπορουτίνες του TMalign:

```
#
# superpose the PDBs with lovoalign
# and keep the results in lovoalign_results file
#

sub superpose_the_PDBs
    {
       my $code_1 = $_[0];
       my $code_2 = $_[1];

       open ( LOVO, ">lovoalign_results" ) or die ("Cannot open LOVO");

       #
       # run lovoalign for the superposition
       #

       $running_lovoalign = `lovoalign -p1 $code_1-1.pdb -p2 $code_2-2.pdb -o SEMI-SUPERPOSITIONS/
$code_1-1_$code_2-2.pdb`;

       print LOVO "$running_lovoalign";

       my  $combine_files  =  `cat  SEMI-SUPERPOSITIONS/$code_1-1_$code_2-2.pdb  $code_2-2.pdb  >
SUPERPOSITIONS/$code_1-1_$code_2-2.pdb`;

       close (LOVO);

       unlink ("SEMI-SUPERPOSITIONS/$code1-1_$code2-2.pdb");
    }


#
# from lovoalign_results
# keep the RMSDs and the structure alignments
#

sub keep_RMSD
    {
       my $code_1 = $_[0];
       my $code_2 = $_[1];

       open ( LOVOTEST, "lovoalign_results" ) || die ("Could not open file");

       while ( $lovoline = <LOVOTEST> )
           {
              if ( $lovoline =~ /FINAL SCORE:.+COVERAGE:\s+(\d+)\s+RMSD:\s+(.+)\s+GAPS/)
                  {
                     $coverage = $1;
                     $RMSD = $2;
                  }

              if ( $lovoline =~ /ATOMS CLOSER THAN\s+3.0000 Ang: RMSD:\s+(.+)\s+COVERAGE:\s+(\d+)/)
                  {
                     $closer_RMSD = $1;
                     $coverage_closer = $2;
                  }
           }

       @alignments = ( $RMSD, $coverage, $closer_RMSD, $coverage_closer );
       return( @alignments );

    }
```

Πριν όμως βάλουμε το ανανεωμένο πρόγραμμα να παράγει 11749 αποτελέσματα, αποφασίσαμε να το δοκιμάσουμε για κάποια από αυτά τυχαία και να συγκρίνουμε τα επιμέρους αποτελέσματα με αυτά από τους K&K και το lovoalign. Η τυχαία επιλογή των δειγμάτων γίνεται με την παρακάτω εντολή, η οποία επιλέγει κάθε εκατοστή γραμμή του αρχείου:

```
perl -ne 'print ((0 == $. % 100) ? $_ : "")' multiple_databaseG > selected.dat
```

Εκτός από αυτά, στο τέλος προσθέσαμε τις παρακάτω καταχωρήσεις (Πίνακας 3), στις οποίες το lovoalign είχει παράγει ανεξήγητα μεγάλα RMSDs σε σχέση με τις sequence-based υπερθέσεις των K&K, όπως και με το TMalign, το νέο πρόγραμμα που επιλέξαμε για να πραγματοποιήσουμε geometry-based υπερθέσεις. Σημαντικό είναι να αναφέρουμε ότι για κάθε RMSD που προκύπτει είναι απαραίτητο να ελέγχουμε και το μήκος της στοίχισης στην οποία αντιστοιχεί. Για να έχουμε βέβαια μια αντικειμενική άποψη του ποιος έχει δίκιο σύμφωνα με το δομικό κόσμο, απαραίτητο ήταν να κοιτάξουμε πιο από τα δύο RMSD αντιπροσώπευε καλύτερα τις διαφορές των δομών. Έτσι παρατηρήσαμε λίγες εικόνες από τις καταχωρήσεις που αναφέρονται στον πίνακα και καταλήξαμε ότι το TMalign δίνει ορθότερα αποτελέσματα (data not shown).

Protein 1	Protein 2	KK		LOVO_align		TM_align	
		RMSD	Length	RMSD	Length	RMSD	Length
1clo	1lo3	4.52	221	18.12	86	1.03	99
1bz7	1i7z	4.13	211	18.30	78	0.53	98
1kcu	1mj8	4.24	219	18.63	91	0.67	101
1i7z	1pg7	5.35	221	19.05	87	1.57	100
1igj	1nd0	5.04	218	21.47	78	1.17	91
1dqq	1iqw	3.72	218	22.86	107	1.73	115
1bgx	1lo0	3.12	216	25.26	77	0.82	103

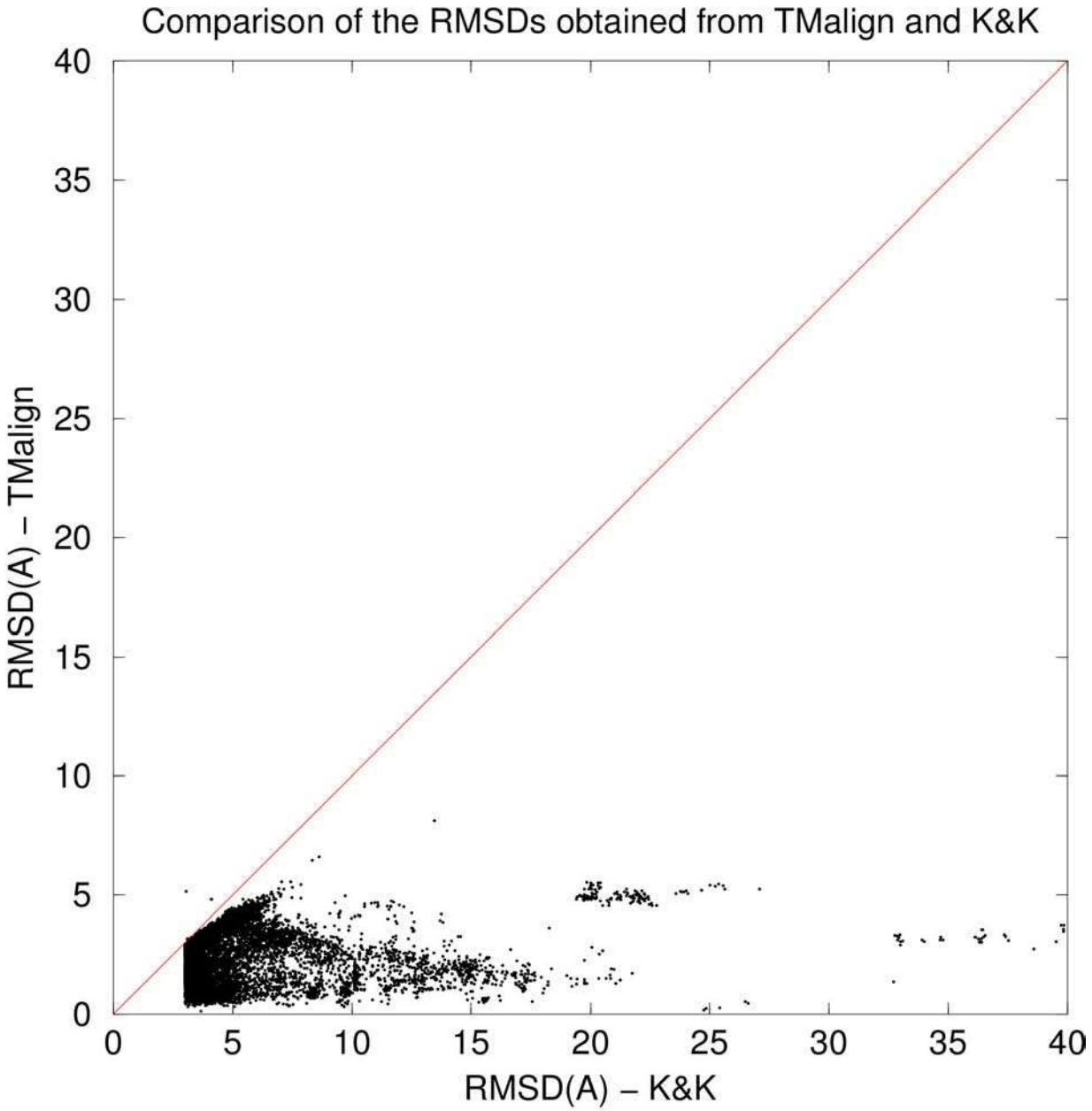
Πίνακας 3: Καταχωρήσεις όπου το lovoalign έχει παράγει πολύ μεγάλο RMSD σε σχέση με τους K&K, όπως και με το νέο πρόγραμμα δομικής υπέρθεσης που χρησιμοποιούμε, το Tmalign.

Όταν πλέον έχει ολοκληρωθεί η δημιουργία του αρχείου και βασιζόμενοι στα αποτελέσματά που αναφέρθηκαν προηγουμένως, τα οποία συγκλίνουν στο μη αποτελεσματικό τρόπο του lovoalign να πραγματοποιήσει την υπέρθεση κάποιων δομών, μπορούμε να δούμε συλλογικά πόσο έχει αποκλίνει η έξοδος του lovoalign από τον νέο αλγόριθμο που χρησιμοποιούμε.

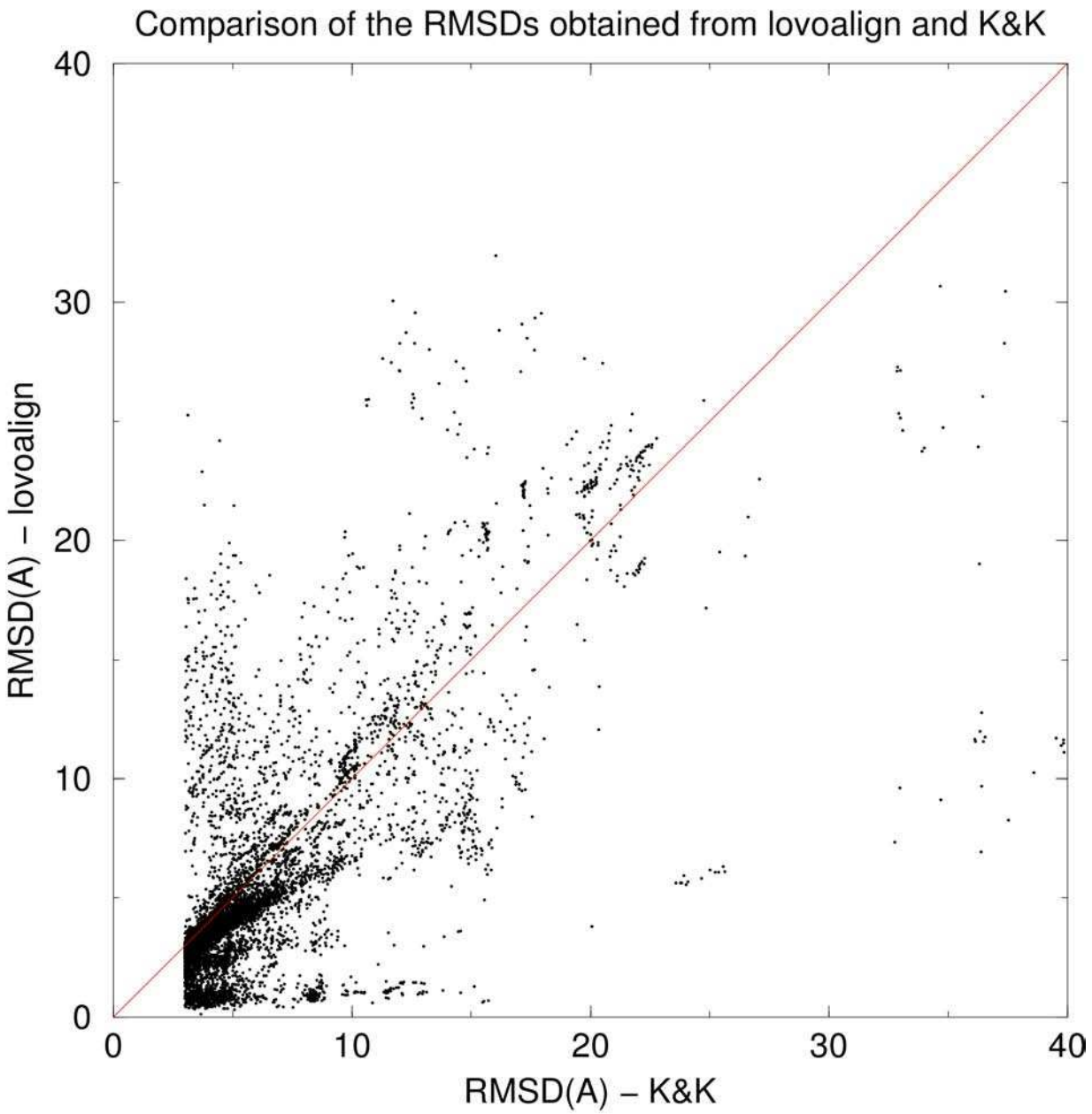
Είναι εύλογο να αναρωτηθεί κανείς πως χρειάστηκαν τόσοι μήνες για να συνειδητοποιήσουμε ότι κάτι πάει στραβά με το lovoalign. Αναφέρομαι σε μήνες καθώς το συγκεκριμένο πρόγραμμα το είχαμε χρησιμοποιήσει όχι μόνο για τη δημιουργία των εικόνων του δεύτερου κεφαλαίου αλλά και σε προηγούμενη εργασία. Σε εκείνη όμως την προσπάθεια οι αλληλουχίες με τις οποίες είχαμε ασχοληθεί ήταν μικρές σε μέγεθος. Επίσης το μήκος των αλληλουχιών ανά ζεύγη ήταν πάντα το ίδιο. Έτσι είχαμε αποκλείσει εμμέσως όλες εκείνες τις περιπτώσεις στις οποίες το lovoalign αποτυγχάνει. Επιπροσθέτως, ακόμη και στην παρούσα μελέτη, πέρασαν αρκετές ώρες μέχρι να διακρίνουμε κάποια μη αναμενόμενα αποτελέσματα και ίσως να αργούσαμε ακόμη περισσότερο αν δεν είχε προηγηθεί το sorting.

'Any sufficiently advanced bug is indistinguishable from a feature.'

Rich Kulawiec



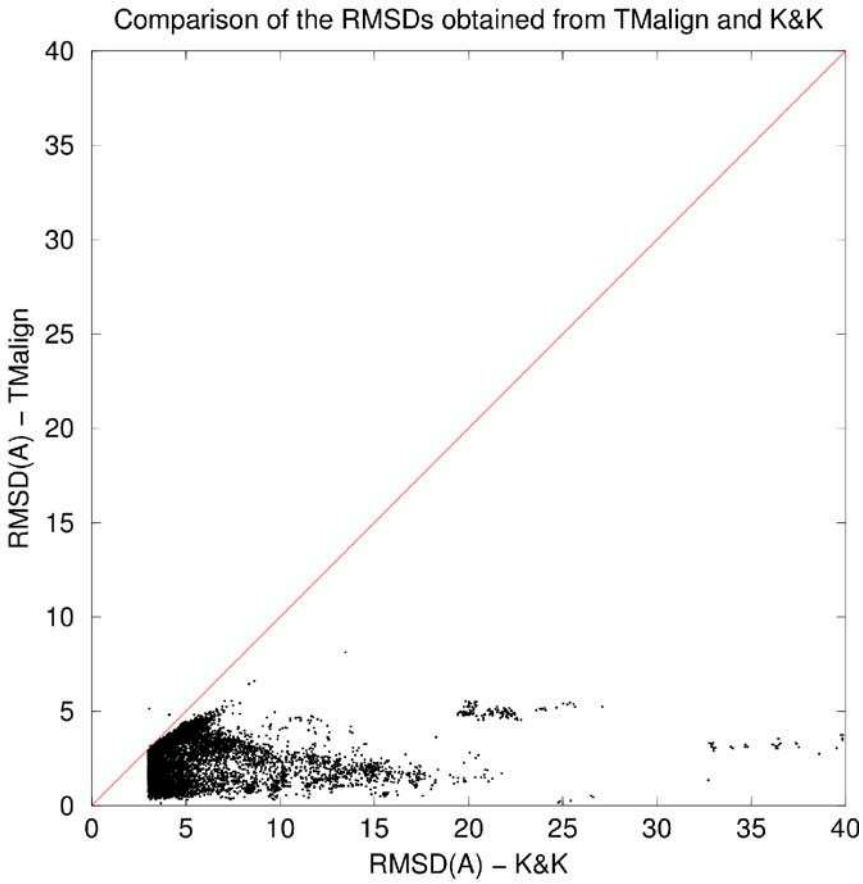
* Η δημιουργία του συγκεκριμένου διαγράμματος αναλύεται στο αμέσως επόμενο υποκεφάλαιο


Comparison of the RMSDs obtained from lovoalign and K&K
RMSD(A) – lovoalign
RMSD(A) – K&K
0
10
20
30
40

3.3

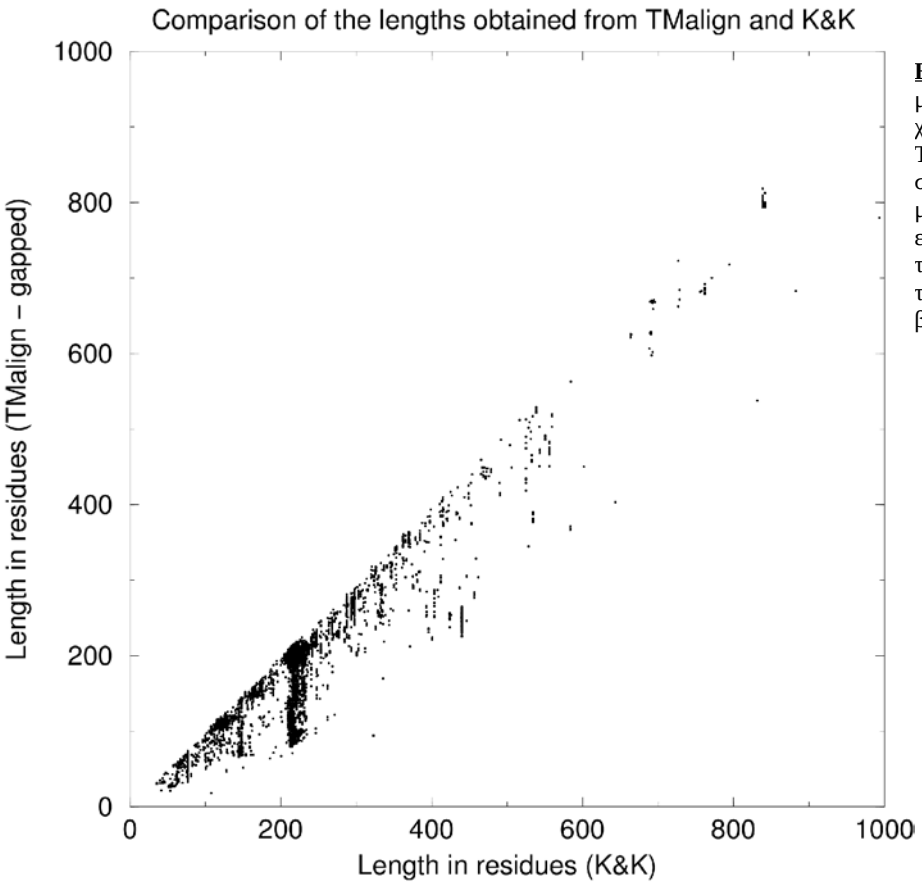
K&K 'BUG'

Μετά από 46 περίπου ώρες εκτέλεσης του προγράμματος, αρκετές διορθώσεις και πολύ υπομονή το 'TMalign_results' συγκέντρωσε 11371 αποτελέσματα επιτρέποντάς μας επιτέλους να συνεχίσουμε την ανάλυσή μας. Η πρώτη υπόθεση που θέλουμε να ελέγξουμε είναι αν τα RMSDs που έχουν υπολογίσει οι K&K με τις sequence-based στοιχίσεις τους είναι παρόμοια με αυτά από το TMalign. Για τον έλεγχο αυτό αποφασίσαμε να δημιουργήσουμε ένα scatterplot (Εικόνα 8), το οποίο θα συγκρίνει τις αντίστοιχες τιμές των RMSDs. Είναι εύκολο να αντιληφθεί κανείς ότι με την εξαίρεση 3 μόνο περιπτώσεων όπου το TMalign έχει δώσει λίγο μεγαλύτερο RMSD από τους K&K, σε όλες τις υπόλοιπες φαίνεται ότι οι sequence-based υπερθέσεις παρουσιάζουν μεγαλύτερο αν όχι ίσο RMSD.



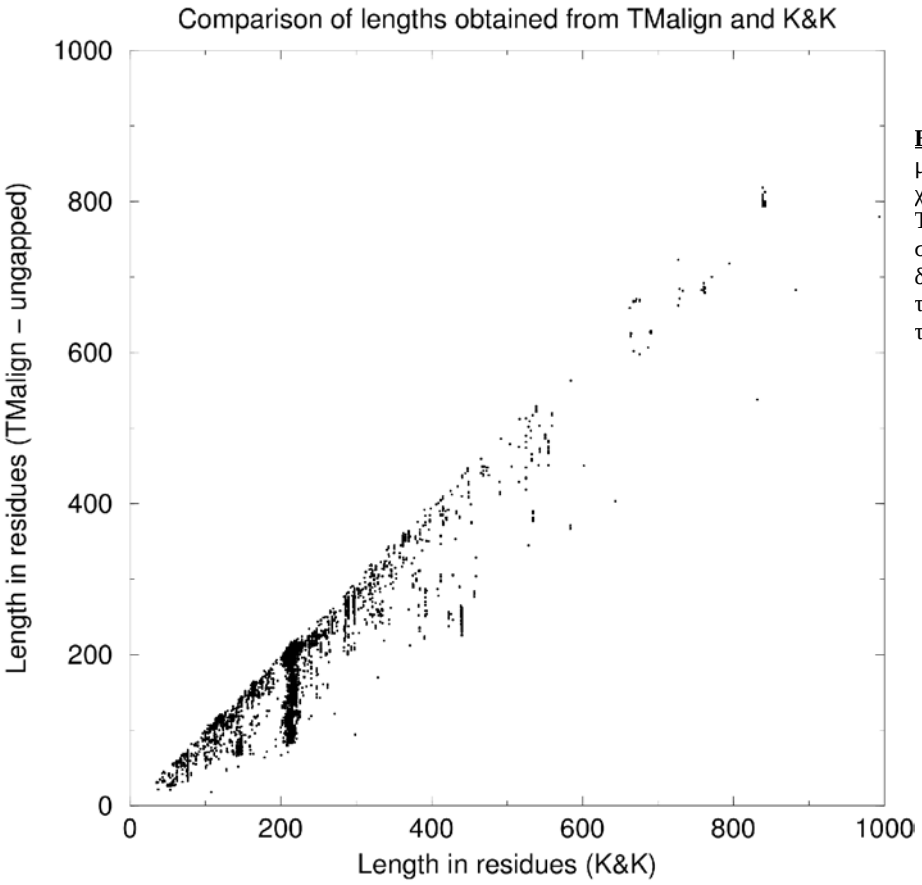
Εικόνα 8: Σύγκριση μεταξύ των RMSDs που προκύπτουν από το TMalign και τους K&K. Η κόκκινη γραμμή αποτελεί τη διαγώνιο του γραφήματος. Τα σημεία που βρίσκονται κάτω από τη διαγώνιο αντιστοιχούν στις υπερθέσεις όπου το TMalign έχει δώσει μικρότερο RMSD από τους K&K.

Προφανώς δε μπορούμε να εξαγάγουμε συμπεράσματα με μόνο κριτήριο το παραπάνω scatterplot, χωρίς να ελέγξουμε αν και τα μήκη των αντίστοιχων στοιχίσεων παραμένουν ίδια. Έτσι με τον ίδιο τρόπο δημιουργούμε ένα νέο γράφημα, το οποίο θα συγκρίνει τα μήκη των αλληλουχιών. Η ιδανική γραφική παράσταση που θα αναμέναμε είναι όλα τα σημεία να βρίσκονταν πάνω στη διαγώνιο (Εικόνα 9).



Εικόνα 9: Σύγκριση μεταξύ του μήκους των καταλοίπων που χρησιμοποιεί για την υπέρθεση το TMalign και του μήκους που έχουν στις στοιχίσεις τους οι K&K. Το μήκος που δίνεται για το TMalign εμπεριέχει κενά. Αν όλα τα μήκη για τις αντίστοιχες στοιχίσεις ήταν όμοια τότε όλα τα σημεία θα έπρεπε να βρίσκονται πάνω στη διαγώνιο.

Επειδή όμως στον πραγματικό κόσμο τίποτα δεν είναι τόσο ιδανικό, προσπαθούμε να βελτιώσουμε το scatterplot με την ακόλουθη διαδικασία: το μήκος για τις στοιχίσεις των K&K δίνεται από το πρόγραμμα του bl2seq με αποτέλεσμα στο μήκος που αρχικά έχουμε εμείς να περιλαμβάνονται και τα κενά της στοίχισης. Με λίγο κώδικα (NO_gaps, επόμενη σελίδα) λοιπόν αφαιρούμε αυτά τα κενά και ξανασχεδιάζουμε το γράφημα (Εικόνα 10).



Εικόνα 10: Σύγκριση μεταξύ του μήκους των καταλοίπων που χρησιμοποιεί για την υπέρθεση το TMalign και του μήκους των στοιχίσεων των K&K. Το μήκος που δίνεται για το TMalign έχει τροποποιηθεί καθώς αφαιρέσαμε όλα τα κενά από τις στοιχίσεις.

'NO_gaps'

```
#!/usr/bin/perl -w


#
# This program removes all gaps from the sequence pairs,
# calculates again the sequence length and adds a new column
# to the previous 2D_matrix file with the new length
#



$no_gaps = 0;

$line = <STDIN>;
print "$line";

while ( $line = <STDIN> )
  {
    if ( $line =~ /(....\s+....\s+.....\s+.....\s+)(\d+)\s+(.*\s)([A-Z\-]+)\s+([A-Z\-]+)\s/)
      {
        $sub_line1 = $1;
        $blast_length = $2;
        $sub_line2 = $3;
        $seq1 = $4;
        $seq2 = $5;
        $no_gaps = 0;

        if ( length($seq1) != length($seq2) || length($seq1) != $blast_length )
          {
            print "PANIC: ", length($seq1), " ", length($seq2), " ", $blast_length, "\n";
            exit;
          }

        for ( $x = 0; $x < $blast_length; $x++ )
            {
               if ( substr($seq1, $x, 1) ne "-" && substr($seq2, $x, 1) ne "-" )
                   {
                      $no_gaps ++;
                   }
            }

   printf ("%s%d %5d   %s  %s %s\n", $sub_line1, $blast_length, $no_gaps, $sub_line2, $seq1, $seq2);

      }
    else
      {
        print "?????? : $line\n\n";
        exit;
      }
  }
```

Τα νέα δεδομένα απεικονίζουν ότι σε αρκετές περιπτώσεις υπέρθεσης που έχουν πραγματοποιηθεί με τη βοήθεια του TMalign το μήκος των αλληλουχιών της στοίχισης είναι πάλι μικρότερο σε σχέση με αυτό για τους K&K (το διάγραμμα είχε αμελητέες αλλαγές όταν αφαιρέσαμε τα κενά από τις στοιχίσεις). Για κάτι ανάλογο έχει γίνει αναφορά και στο άρθρο των K&K. Οι τελευταίοι στα πλαίσια της έρευνάς τους είχαν πραγματοποιήσει κάποιες geometry-based στοιχίσεις με τη βοήθεια των προγραμμάτων Ska (Petrey et al., 2003) [20] και CE (Shindyalov et al., 1998) [21] για να τις συγκρίνουν με τις sequence-based. Βάσει αποτελεσμάτων λοιπόν υποστήριξαν ότι οι μέθοδοι δομικής στοίχισης συστηματικά στοιχίζουν ίσο ή μικρότερο αριθμό καταλοίπων από τις sequence-based στοιχίσεις.

Η χρήση συκεκριμένου structure-based αλγόριθμου, δηλαδή του TMalign, δεν είναι καθόλου τυχαία, καθώς συγκεντρώνει πολλά πλεονεκτήματα σε σχέση με άλλους geometry-based αλγόριθμους. Το εν λόγο πρόγραμμα για να υπολογίσει την καλύτερη δομική στοίχιση μεταξύ ζευγών πρωτεϊνών συνδυάζει τον πίνακα περιστροφής (TM score) με το δυναμικό προγραμματισμό. Ο αλγόριθμος είναι 4 φορές πιο γρήγορος από τον DE που χρησιμοποίησαν οι K&K και κατά μέσο όρο οι στοιχίσεις που προκύπτουν έχουν μεγαλύτερη ακρίβεια και μήκος στοίχισης. Όπως γνωρίζουμε το πιο κοινό μέτρο σύγκρισης πρωτεϊνών είναι το RMSD το οποίο αποτελεί την τυπική απόκλιση μεταξύ των αντίστοιχων καταλοίπων μετά από μια βέλτιστη περιστροφή της μίας πρωτεΐνης πάνω στην άλλη. Εφόσον το RMSD αντιπροσωπεύει τις αποστάσεις μεταξύ όλων των ζευγών καταλοίπων με τον ίδιο τρόπο, ένας μικρός αριθμός τοπικών δομικών αποκλίσεων μπορεί να οδηγήσει σε πολύ υψηλό RMSD, ακόμη και όταν οι ολικές τοπολογίες των συγκρινόμενων δομών είναι παρόμοιες. Επίσης το μέσο RMSD τυχαία σχετιζόμενων πρωτεϊνών σχετίζεται με το μήκος των συγκρινόμενων αλληλουχιών που στοιχίζονται. Η χρήση του TM score στο συγκεκριμένο αλγόριθμο ξεπερνάει τα προβλήματα αυτά αξιοποιώντας μια παραλλαγή του Levitt-Gerstein weight factor (1998) [22] που θεωρεί τα ζεύγη καταλοίπων σε κοντινές αποστάσεις πιο σημαντικά από αυτά που απέχουν μεγαλύτερες αποστάσεις. Έτσι το TM score είναι πιο ευαίσθητο στην συνολική τοπολογία της πρωτεΐνης σε σχέση με τις τοπικές αλληλεπιδράσεις. Επιπλέον, η τιμή του TM score κανονικοποιείται από με τέτοιο τρόπο ώστε το μέγεθος του score να μην επηρεάζεται από το μέγεθος της πρωτεΐνης, με τιμή 0.17 για ένα μέσο ζεύγος τυχαία στοιχιζόμενων αλληλουχιών(Zhang et al. 2004) [23].

Τα πλεονεκτήματα του TMalign σε συνδυασμό με το ότι μια επίσης μεγάλη πλειοψηφία των στοιχίσεων βρίσκεται πάνω στη διαγώνιο του γραφήματος μας ενθαρρύνουν όσον αφορά τη σημαντικότητα των αποτελεσμάτων μας. Το γεγονός ότι αρκετά δεδομένα ανήκουν στη διαγώνιο συνεπάγεται ότι τα μήκη που χρησιμοποιήθηκαν στις στοιχίσεις αυτές ήταν σε ένα σημαντικό ποσοστό τα ίδια και έτσι μπορούμε να αποφανθούμε ότι υπάρχουν περιπτώσεις στις οποίες το TMalign που χρησιμοποιούμε έχει δώσει σημαντικά μικρότερο RMSD. Αυτό σε συνδυασμό με το ότι το TMalign στοιχίζει μεγαλύτερες αλληλουχίες από άλλους geometry-based αλγόριθμους και επιπροσθέτως υπολογίζει το RMSD βάσει της ολικής τοπολογίας της πρωτεΐνης, θέτει πολλά ερωτηματικά σε αυτό που υποστηρίζουν οι K&K: ότι δηλαδή οι sequence-based αλγόριθμοι υπερθέσεων υπολογίζουν καλύτερα τις δομικές διαφορές σε πρωτεΐνες με μεγάλη ομοιότητα στην αλληλουχία τους.

Επειδή όμως στον κόσμο των πρωτεϊνών αυτό που έχει πραγματικά σημασία είναι οι ίδιες οι δομές , αποφασίσαμε να δείξουμε επιλεκτικά ορισμένες αντιπροσωπευτικές περιπτώσεις στις οποίες για το ίδιο μήκος στοίχισης το TMalign και οι K&K συμφωνούν και δίνουν ίδια RMSDs (υπερθέσεις 1, 2, 3). Στη συνέχεια, επιλέγουμε κάποια παραδείγματα (υπερθέσεις 4, 5, 6, 7) όπου πάλι για ίδιο μήκος στοίχισης το TMalign υπολογίζει συντριπτικά μικρότερα RMSDs από τους K&K. Είναι αρκετά εμφανές πιστεύω ότι τα αρκετά μεγάλα RMSDs που δίνουν οι K&K για τις υπερθέσεις 4, 5 και 6 δεν έχουν το ανάλογο αντίκτυπο στη δομική πραγματικότητα. Η επιλογή των καταχωρήσεων έχει γίνει έτσι ώστε να μη διαφέρουν τα μήκη που δίνουν οι K&K και το TMalign (τα μήκη των στοιχίσεων φαίνονται μέσα στις παρενθέσεις), ώστε να έχει πραγματοποιηθεί η υπέρθεση γ στη σύγκριση. Επίσης δίπλα από τους κωδικούς των πρωτεϊνών υπάρχει το ποσοστό ομοιότητας για ίδιο αριθμό καταλοίπων. Με τις εικόνες αυτές εύκολα μπορεί να αντιληφθεί κανείς ποιο από τα δύο είδη υπέρθεσης φαίνεται να αντικατοπτρίζει με καλύτερο τρόπο τις διαφορές μεταξύ των πρωτεϊνών από άποψη δομικού περιεχομένου.

1.SUPERPOSITION OF

1cunA with **1u4qB (99.1%)**

TMalign RMSD:

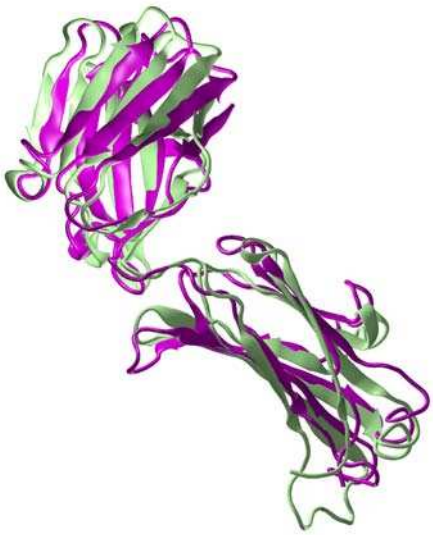
3.03(209)

K&K RMSD:

3.11(212)

2.SUPERPOSITION OF

12e8P with **1fvdD (60.2%)**



TMalign RMSD:

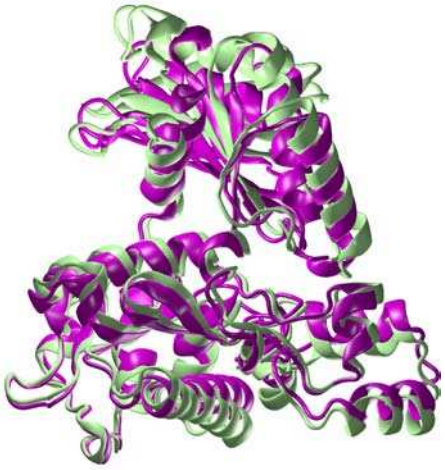
2.96(219)

K&K RMSD:

3.03(221)

3.SUPERPOSITION OF

1g99A with **1tuuA (100.0%)**



TMalign RMSD:

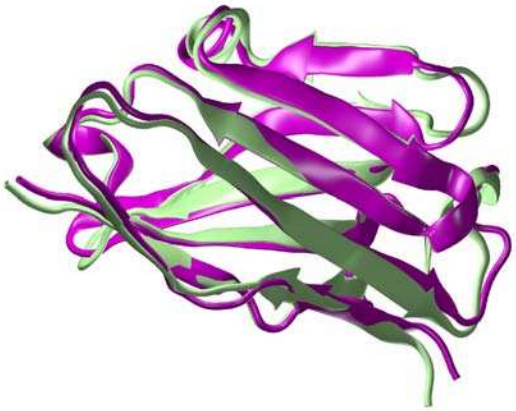
3.38(394)

K&K RMSD:

3.46(399)

4.SUPERPOSITION OF

1dzbA with **1p2cE (100.0%)**



TMalign RMSD:

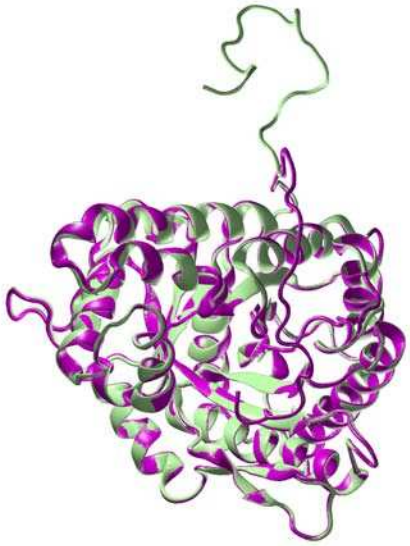
1.45(116)

K&K RMSD:

12.07(116)

5.SUPERPOSITION OF

1fdjA with **1fdjD (100.0%)**



TMalign RMSD:

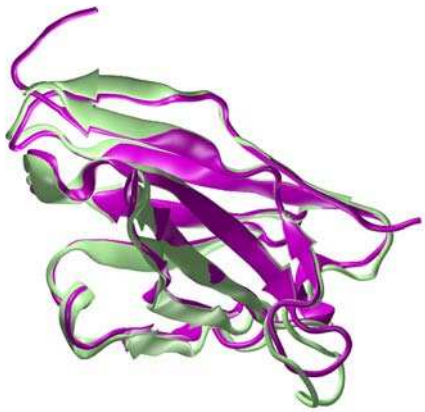
0.47(346)

K&K RMSD:

8.10(363)

6.SUPERPOSITION OF

1iqbB with **1nqbA (67.0%)**



TMalign RMSD:

1.03(116)

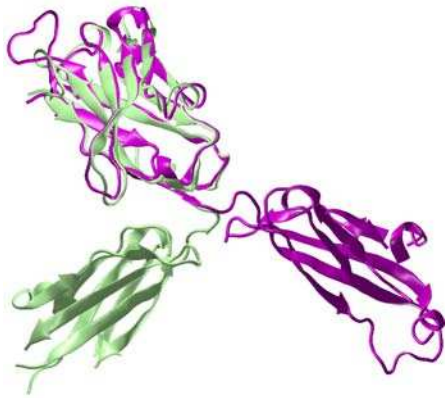
K&K RMSD:

8.28(120)

Το να προσπαθούμε να επιβάλουμε ολική υπέρθεση μέσω sequence-based στοίχισης μπορεί να οδηγήσει σε λάθος συμπεράσματα και σε περιπτώσεις όπου υπάρχει Domain movement. Μια πιο ποσοτική ανάλυση για όλες αυτές τις περιπτώσεις θα μπορούσε να δοθεί με διαχωρισμό των επιμέρους μονομερών και υπολογισμό του RMSD για το καθένα. Όταν τα RMSDs είναι μικρά και ταυτοχρόνως περίπου ίδια σημαίνει ότι η αύξηση του RMSD κατά την πραγματοποίηση sequence-based υπερθέσεων οφείλεται σε κάποια αλλαγή στο βρόχο που τα συνδέει. Χαρακτηριστικό παράδειγμα η εικόνα [υπέρθεση 7] που ακολουθεί.

7.SUPERPOSITION OF

1iqdB with **1op3M (63.8%)**



TMalign RMSD:

1.70(118)

K&K RMSD:

13.70(221)

Από τις λίγες αυτές εικόνες φαίνεται ότι στις περιπτώσεις που οι K&K δίνουν διαφορετικά RMSDs από το πρόγραμμα δομικής στοίχισης που χρησιμοποιούμε εμείς και ταυτόχρονα τα μήκη των στοιχίσεων δεν διαφέρουν, το TMalign ανταποκρίνεται καλύτερα στην δομική πραγματικότητα. Οι εικόνες αυτές μας επιτρέπουν να εμπιστευτούμε το αρχικό διάγραμμα που είχαμε δημιουργήσει (Εικόνα 8) και ορθά να μεταφράσουμε τη διαφορά μεταξύ των RMSDs σε μη σωστή επιλογή των K&K στις περιπτώσεις αυτές να χρησιμοποιήσουν sequence-based υπέρθεση για να υπολογίσουν διαφορές σε επίπεδο δομής.

Η αλήθεια είναι ότι υπάρχουν περιπτώσεις όπου τόσο οι structure-based αλγόριθμοι όσο και οι sequence-based παράγουν ίδια αποτελέσματα τα οποία αντικατοπτρίζουν με αρκετή σαφήνεια τη δομική πραγματικότητα. Επίσης δε μπορούμε να αποκλείσουμε προφανώς ότι οι sequence-based υπερθέσεις έχουν και αυτές τη δική τους χρησιμότητα και ότι μερικές φορές μπορεί να έχουν σημαντικότητα για τις διαφορές μεταξύ των δομών βάσει πλήρης αλληλουχίας. Όμως, η ανάλυση που έχει πραγματοποιηθεί στα δύο αυτά τελευταία κεφάλαια και η παράθεση των διαφόρων εικόνων μας έχει πείσει για το πλεονέκτημα που εμφανίζουν οι structure-based στοιχίσεις έναντι των sequence-based.

Εν κατακλείδι, παρατηρούμε ότι αλληλουχικά όμοιες πρωτεΐνες μπορεί να διαφοροποιούνται σε επίπεδο δομής, τα RMSDs όμως που δίνουν οι K&K μας δίνουν μια εικόνα πολύ διαφορετική από αυτήν που πραγματικά ισχύει στον τρισδιάστατο κόσμο των πρωτεϊνών.

3.4

Υπάρχουν αμινοξικές μεταλλαγές που συνδέονται με αλλαγή δομής;

Εφόσον έχουμε αποφανθεί για το βαθμό διαφοροποίησης δομών με σχετικά όμοιες αλληλουχίες, προσπαθούμε να ελέγξουμε αν αυτές οι αλλαγές στη δομή συνδέονται με συγκεκριμένες μεταλλαγές* των αμινοξέων.

Για το σκοπό αυτό έχουμε ήδη δημιουργήσει έναν πίνακα κατανομής των μεταλλαγών μεταξύ των ζευγών των στοιχίσεων που δίνουν οι K&K. Στο σημείο αυτό έπρεπε να αυθαιρετήσουμε λίγο έτσι ώστε να χωρίσουμε (με τη βοήθεια κώδικα, 'RMSD_separator') τα αποτελέσματα μας σε δύο υποομάδες, μία που θα αντιστοιχεί σε χαμηλά και μία σε υψηλά RMSDs αντίστοιχα. Επιλέξαμε λοιπόν ότι δομές που εμφανίζουν μεγαλύτερο του 2.5 Å ανήκαν στην τελευταία ομάδα και οι υπόλοιπες στην πρώτη. Στη συνέχεια, με λίγο κώδικα ('frequencies_calc') υπολογίζουμε το άθροισμα της κάθε μεταλλαγής στην εκάστοτε ομάδα. Να αναφέρουμε στο σημείο αυτό ότι επιλέξαμε να αυξάνουμε τον αριθμό μιας μεταλλαγής είτε αυτή συμβαίνει από το ένα κατάλοιπο προς το άλλο, είτε αντίστροφα. Για παράδειγμα αν μια Αλανίνη μετατραπεί σε Προλίνη και επίσης μια προλίνη γίνει Αλανίνη, τότε αυξάνουμε τη μεταλλαγή AP (δηλαδή Ala-Pro) κατά 2.

'RMSD_seperator'

```
#!/usr/bin/perl -w

#
# This program separates our Tmalign_results
# into two distinct groups based on the RMSD
# (lower or equal to 2.5 RMSD and higher )
#


$line = <STDIN>;

while ( $line = <STDIN> )
    {
      if ( $line =~ /^(....)\s+(....)\s+.....\s+(.....)\s+\d+\s+(.....)/ )
         {
            $code1 = $1;
            $code2 = $2;
            $KK_RMSD = $3;
            $MY_RMSD = $4;

                   if ( $4 <= 2.5 )
                       {
                           print $line;
                       }
         }
     }
```

* Με τον όρο μεταλλαγές αναφερόμαστε σε όλες τις αλλαγές αντικατάστασης καταλοίπου, καθώς μόνο αυτές μπορούμε να ελέγξουμε.

'frequencies_calc'

```
#!/usr/bin/perl -w

#
# This program calculates the frequency
# of each mutation in the 'Tmalign_results'
# matrix
#


$line = <STDIN>;
@legends = split ( ' ', $line );
$all = @legends;

for ( $i=0 ; $i < 1000 ; $i++ )
  {
    $res[ $i ] = 0;
  }

while ( $line = <STDIN> )
    {
      chomp($line);
      @words = split ( ' ', $line );
      $len = @words;

      if ( $len != $all )
        {
          print "Unexpected line : $line. Goodbye.\n";
          exit;
        }

      for ( $i=0 ; $i < $len ; $i++ )
        {
          $res[ $i ] += $words[ $i ];
        }
    }



for ( $i=0 ; $i < $len ; $i++ )
  {
    print $legends[ $i ]," ",$res[$i], "\n";
  }
```

Πολύ σημαντικό είναι επίσης να υπολογίσουμε το σύνολο εμφάνισης των μεμονωμένων καταλοίπων στις αλληλουχίες που χρησιμοποιούμε. Αυτό είναι απαραίτητο καθώς η εμφάνιση κάθε μεταλλαγής ενός καταλοίπου προφανώς εξαρτάται και από τον αριθμό εμφάνισης του συγκεκριμένου καταλοίπου στη βάση δεδομένων που χρησιμοποιούμε. Για να γίνει πιο κατανοητό αυτό θα αναφέρουμε το εξής απλό παράδειγμα: σε μια βάση δεδομένων όπου απουσιάζει για παράδειγμα εντελώς το αμινοξύ Τρυπτοφάνη, η συχνότητα μεταλλαγής της Τρυπτοφάνης θα είναι σχεδόν αμελητέα σε σύγκριση με τα άλλα αμινοξέα, καθώς θα είναι πολύ πιο σπάνιο να πραγματοποιηθεί κάποια τέτοια μεταλλαγή. Αυτό όμως δεν ανταποκρίνεται στην πραγματική συχνότητα μεταλλαγής της Τρυπτοφάνης. Έτσι, αφού επιλέξουμε μόνο τις στήλες που περιέχουν τις στοιχίσεις του κάθε ζεύγους, υπολογίζουμε την συχνότητα εμφάνισης του κάθε καταλοίπου με τον παρακάτω τρόπο('count_residues'):

'count_residues.c'

```
/* This program calculates the total
   number of each residue found in
   our database */


#include <stdio.h>


main()
{
    int       i;
    char character;
    int aminos[129];

   for ( i=0 ; i <= 128 ; i++ )
      {
         aminos[i] = 0;
      }


    while ( scanf("%c", &character) == 1)
       {
           aminos[character]++;
       }


    for ( i=65 ; i <= 90 ; i++ )
      {
        printf( "%c %d\n", i, aminos[i] );
      }

}
```

Διαθέτουμε πλέον τον αριθμό εμφάνισης κάθε μεταλλαγής, όπως και το σύνολο κάθε καταλοίπου στην κάθε υποομάδα (για χαμηλά και υψηλά RMSD). Στον πίνακα που ακολουθεί έχουμε επιλέξει να δείξουμε τις δέκα πρώτες σε εμφάνιση μεταλλαγές όπως και τα δέκα πρώτα αμινοξέα που εμφανίζονται στις δύο αυτές υπομάδες.

LOW RMSD (≤2.5) (total 5.113)			
residues		mutation	
L	185860	**ST**	11623
S	177953	**IV**	7503
A	156496	**LV**	6331
T	149829	**KR**	5926
G	148557	**AS**	5890
V	146188	**DE**	5718
K	129475	**IL**	5488
E	119533	**FY**	4435
D	110730	**NS**	4355
I	96839	**KQ**	4147

HIGH RMSD (>2.5) (total: 6238)			
residues		mutation	
S	295865	**ST**	23607
T	226497	**AS**	10357
L	214270	**LV**	10114
V	195563	**KR**	9520
G	194280	**IV**	9413
A	181183	**DE**	7466
K	173508	**AV**	7281
E	138921	**AT**	6993
P	138135	**NS**	6909
D	129856	**KQ**	6868

Πίνακας 4: Αθροίσματα εμφάνισης καταλοίπων και μεταλλαγών σε δύο υποομάδες, οι οποίες δημιουργήθηκαν σύμφωνα με την τιμή του RMSD (μικρότερο-ίσο ή μεγαλύτερο του 2.5) που υπολόγισε το TMalign για τα ζεύγη της ομάδας G.

LOW RMSD	
normalized mutation f	
ST	0.0355
IV	0.0309
FY	0.0276
KR	0.0263
DE	0.0248
IL	0.0194
KQ	0.0193
LV	0.0191
AS	0.0176
NS	0.0163

HIGH RMSD	
normalized mutation f	
ST	0.0452
IV	0.0312
DE	0.0278
LV	0.0247
KQ	0.0237
AS	0.0217
KR	0.0213
AV	0.0193
NS	0.017
AT	0.0109

Πίνακας 5: Οι ίδιες συχνότητες μεταλλάξεων κανονικοποιημένες ως προς το άθροισμα εμφάνισης των αντίστοιχων καταλοίπων και τοποθετημένες κατά φθίνουσα σειρά.

Μετά την κανονικοποίηση των αποτελεσμάτων ως προς το άθροισμα εμφάνισης των αντίστοιχων καταλοίπων και την τοποθέτησή τους σε φθίνουσα σειρά, παρατηρούμε ότι οι συχνότερα εμφανιζόμενες μεταλλαγές ανεξάρτητα αν το RMSD αλλάζει, είναι η μεταλλαγή Ser – Thr(Thr - Ser) και Val – Ile (Ile -Val). Επιπλέον, συγκρίνοντας τα νούμερα για τα δύο διαφορετικά RMSDs φαίνεται να μην υπάρχει κάποια προτίμηση καταλοίπου όταν οι δομές αλλάζουν διευθέτηση.

Βέβαια, ακόμη και με τον συνολικό αριθμό εμφάνισης των καταλοίπων δε μπορούμε να έχουμε στατιστικά σημαντικά αποτελέσματα. Αυτό συμβαίνει γιατί η ομάδα G των K&K περιλαμβάνει πολλές επαναλήψεις πρωτεϊνών οι οποίες μοιράζονται μεγάλη ομοιότητα μεταξύ τους (χαρακτηριστικό παράδειγμα οι λυσοζύμες) ενώ κάποιες άλλες πρωτεΐνες τις συναντάμε ελάχιστες φορές. Με λίγα λόγια, είναι σαν να θέλουμε να συγκρίνουμε την εμφάνιση ενός καταλοίπου που υπάρχει σε όλες τις λυσοζύμες με κάποιο κατάλοιπο που υπάρχει μόνο σε κάποια μεμονωμένη πρωτεΐνη. Ακόμη κι αν η πρωτεΐνη αυτή διέθετε 100 επαναλήψεις μιας Γλυκίνης, και κάθε λυσοζύμη μόνο μία επανάληψη αυτού του καταλοίπου, το αποτέλεσμα θα μας δημιουργούσε την υποψία ότι οι Γλυκίνες συναντόνται συχνότερα στις λυσοζύμες παρά στην άλλη πρωτεΐνη.

Ο χρόνος που διαθέταμε όμως δεν αρκούσε για να πραγματοποιήσουμε αυτούς τους επιπλέον υπολογισμούς και γι αυτό περιοριστήκαμε στο να εξαγάγουμε όποια συμπεράσματα μπορούσαμε με όσα δεδομένα διαθέταμε. Χωρίς την κανονικοποίηση τα συμπεράσματα είναι δύσκολα. Αυτό που μπορούμε να συμπεράνουμε είναι ότι όλες οι προτιμώμενες αντικαταστάσεις είναι συντηρητικές και επίσης παρατηρούμε ότι και οι Γλυκίνες συντηρούνται. Αυτό είναι αρκετά λογικό καθώς είναι πολύ πιο εύκολο να υιοθετηθεί ένα νέο κατάλοιπο στην θέση που βρισκόταν κάποιο άλλο με παρόμοια φυσικοχημικά χαρακτηριστικά. Δηλαδή η μεταλλαγή ενός υδρόφοβου καταλοίπου με ένα υδρόφιλο θα έφερνε αντιμέτωπο το τελευταίο κατάλοιπο με το υδρόφοβο περιβάλλον στο οποίο συνυπήρχε αρμονικά το προηγούμενο. Έτσι, οι αλληλεπιδράσεις μεταξύ των μορίων δε θα ήταν τόσο σταθερές με αποτέλεσμα να αυξηθεί η ελεύθερη ενέργεια του συστήματος. Με λίγα λόγια, τόσο στις υπερθέσεις που εμφανίζουν μικρό RMSD όσο και σε αυτές που οι δομές διαφέρουν σημαντικά παρατηρούμε γενικά το ίδιο μοτίβο αντικαταστάσεων μεταξύ των αμινοξέων.

Στο σημείο αυτό πρέπει να σημειώσουμε ότι το RMSD που προκύπτει από το TMalign δεν αντιστοιχεί στη συνολική στοίχιση που διαθέτουμε, καθώς δεν στοιχίζει το σύνολο τον καταλοίπων που δίνουν οι K&K. Παρόλα αυτά, επιλέξαμε να διατηρήσουμε και τα κατάλοιπα στις “μη στοιχισμένες περιοχές” καθώς μπορεί να ευθύνονται και μεταλλαγές στις περιοχές αυτές για τις αλλαγές δομής. Συνεπώς θεωρούμε καλύτερο να μη εξαιρέσουμε κανένα τμήμα της στοίχισης.

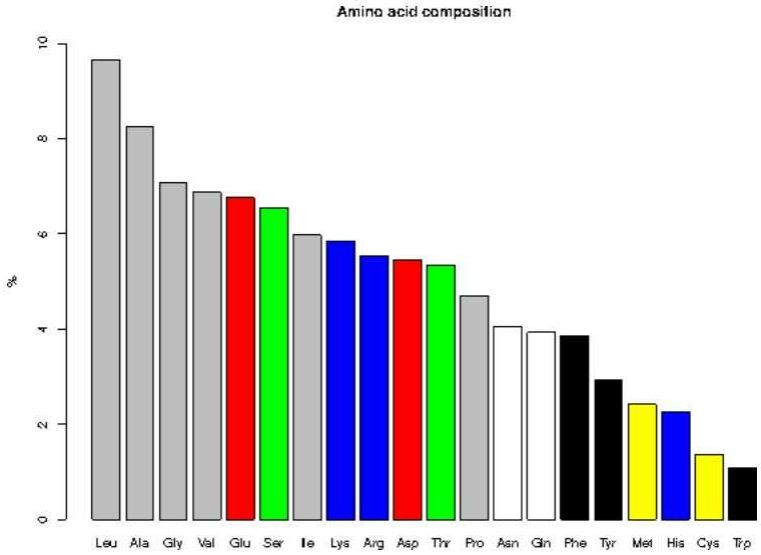
Επίσης παρακάτω έχουμε παραθέσει τις συχνότητες εμφάνισης των διαφόρων αμινοξέων στη βάση δεδομένων της Swissprot/Uniprot, ώστε να υπάρχει ένα κριτήριο του τι προτιμάται στον κόσμο των αμινοξέων:

6.1 Composition in percent for the complete database

Ala (A) 8.26	Gln (Q) 3.93	Leu (L) 9.66	Ser (S) 6.55
Arg (R) 5.53	Glu (E) 6.75	Lys (K) 5.85	Thr (T) 5.34
Asn (N) 4.06	Gly (G) 7.08	Met (M) 2.42	Trp (W) 1.08
Asp (D) 5.46	His (H) 2.27	Phe (F) 3.86	Tyr (Y) 2.92
Cys (C) 1.36	Ile (I) 5.97	Pro (P) 4.70	Val (V) 6.87

Εικόνα 11: Ποσοστά εμφάνισης των αμινοξέων στη Swissprot/Uniptot (αναπαράγεται άνευ αδείας από http://web.expasy.org/docs/relnotes/relstat.html

Asx (B) 0.000 Glx (Z) 0.000 Xaa (X) 0.00



Legend: gray = aliphatic, red = acidic, green = small hydroxy,
blue = basic, black = aromatic, white = amide, yellow = sulfur

Εικόνα 12: Στην εικόνα αυτή εκτός από το ποσοστό εμφάνισης του κάθε αμινοξέος φαίνονται επίσης και οι ιδιότητές του (αναπαράγεται άνευ αδείας από http://web.expasy.org/docs/relnotes/relstat.html)

Σύμφωνα με τις εικόνες αυτές φαίνεται ότι η Λευκίνη (9.66%) διατηρεί με διαφορά την πρώτη θέση ως αναφορά τη συχνότητα εμφάνισης και ακολουθούν η Αλανίνη(8.26%) και η Γλυκίνη(7.08%). Τέταρτη με πολύ μικρή διαφορά από την Γλυκίνη βρίσκεται η Βαλίνη(6.87%). Τα δεδομένα αυτά μας βοηθούν να δούμε κατά πόσο ο αριθμός του κάθε καταλοίπου στις δύο ομάδες που μελετάμε διαφοροποιείται από αυτό που επικρατεί σε γενικότερο πλαίσιο.

Η αλήθεια είναι ότι τα αποτελέσματά μας δεν διαφωνούν απόλυτα με τα δεδομένα τις Swissprot, παρατηρούμε εναλλαγή των ίδιων αμινοξέων στις μεγαλύτερες θέσεις. Παρόλα αυτά δεν συγκλίνουν κιόλας με μεγάλη ακρίβεια. Έτσι λόγω περιορισμού χρόνου θα παραμείνουμε στο αμφίβολο αυτό σημείο με ότι έχουμε αναφέρει μέχρι στιγμής.

Κεφάλαιο 4

Επίλογος

Φτάνοντας λοιπόν στο τέλος αυτής της μελέτης ας κάνουμε έναν απολογισμό του τι μας δίδαξε το ίδιο το “ταξίδι”.

Το πρώτο λοιπόν συμπέρασμα μας υποστηρίζει ότι μεγάλη ομοιότητα στην πρωτεϊνική αλληλουχία δεν αντιστοιχεί και απαραίτητα σε παρόμοιες τρισδιάστατες δομές. Η φυσική επιλογή έχει συμβάλει σημαντικά στην “παρερμηνεία” του πειράματος του Anfinsen, καθώς υπάρχει ένας συντηρητισμός στη διατήρηση των δομών πρωτεϊνών χάρη της λειτουργικότητας και της επιβίωσης στον πλανήτη. Απουσία όμως φυσικής επιλογής, βρισκόμαστε σε πλήρη άγνοια στο έλεος της αναδίπλωσης των πρωτεϊνών. Τα τελευταία αποτελέσματα που προέκυψαν από το TMalign δείχνουν ξεκάθαρα ότι ένα μεγάλο πλήθος αλληλουχικά όμοιων πρωτεϊνών (ποσοστό ομοιότητας > 50) παρουσιάζουν RMSD μεγαλύτερο του 2.5, όπως και έχουν RMSD μεγαλύτερο του 5, το οποίο συνεπάγεται διαφορές σε επίπεδο δομών.

Το δεύτερο συμπέρασμα αφορά τις sequence-based στοιχίσεις, βάσει των οποίων πραγματοποίησαν την έρευνά τους οι K&K. Όταν βασιζόμαστε στην αλληλουχία μιας πρωτεΐνης για να συγκρίνουμε τη δομή της, είναι σαν να προσπαθούμε να συγκρίνουμε δύο τρισδιάστατα αντικείμενα, στηριζόμενοι στη μονοδιάστατη μορφή τους. Είναι πολύ εύκολο μια μικρή αλλαγή(μια ένθεση ή μια έλλειψη) στην αλληλουχία να μετατοπίσει την στοίχιση των πρωτοταγών τους δομών και αυτό να οδηγήσει σε αύξηση του RMSD που δεν ανταποκρίνεται στο δομικό κόσμο, καθώς εξαναγκάζουμε τις δομές να στοιχιθούν, όπως εμείς ορίσαμε, αμελώντας να κοιτάξουμε τη στοίχιση που ορίζουν οι ίδιες οι δομές τους. Αυτό που θέλουμε να αναφέρουμε είναι ότι επειδή υπάρχουν αυξημένες πιθανότητες οι sequence-based αλγόριθμοι να αποτύχουν, απαραίτητο θα ήταν κάθε φορά να παρατηρούμε και τις αντίστοιχες δομές των πρωτεϊνών που στοιχίζουμε, ώστε να οι υπερθέσεις αυτών να συνάδουν με τη δομική πραγματικότητα.

Η τρίτη εικασία που θέλουμε να αναφέρουμε (καθώς δεν υπήρξε αρκετός χρόνος για να έχουν στατιστική σημαντικότητα τα αποτελέσματά μας) σχετίζεται με το αν υπάρχουν αμινοξικές αλλαγές που προτιμούνται όταν αλλάζει η στερεοδιαμόρφωση μιας πρωτεΐνης. Οι μόνοι ισχυρισμοί που διαθέτουμε υποστηρίζουν πως υπάρχει μεγάλη προτίμηση στης μεταλλαγές αντικατάστασης μεταξύ αμινοξέων με παρόμοια χαρακτηριστικά, κάτι το οποίο είναι εύκολο να αντιληφθεί κανείς, αφού τέτοιου είδους αλλαγές μπορούν πολύ εύκολα να υιοθετηθούν χωρίς σπατάλη ενέργειας.

Κεφάλαιο 5

"Some things... you should never forget!"

Η σημασία της βιβλιογραφίας

Σε αυτό το μάλλον πρωτότυπο κεφάλαιο περιγράφεται ουσιαστικά η εφαρμογή του νόμου του Μέρφυ " **Αν κάτι μπορεί να πάει στραβά, θα πάει** " (Arthur Bloch) [27] στα πλαίσια της πτυχιακής αυτής εργασίας.

Η ιδέα με την οποία ξεκίνησε η συγκεκριμένη μελέτη, διαφέρει σημαντικά από αυτήν στην οποία εξελίχθηκε. Με τίτλο "Are reversed sequences still foldable?" η αρχική ιδέα της συγκεκριμένης πτυχιακής εργασίας αφορούσε τη μελέτη των αποτελεσμάτων αντιστροφής της αμινοξικής αλληλουχίας των πρωτεϊνών. Στα πλαίσια αυτής της μελέτης αναζητούσαμε ολιγοπεπτίδια στις ανεστραμμένες πρωτεϊνικές αλληλουχίες τα οποία εμφάνιζαν μεγάλη ομοιότητα με φυσικές πρωτεϊνικές περιοχές.

Για την πραγματοποίηση της έρευνας αυτής απαραίτητη ήταν η δημιουργία ενός προγράμματος (παρατίθεται παρακάτω στο Appendix B), το οποίο θα ανέστρεφε όλες τις αμινοξικές αλληλουχίες της PDB, θα αναζητούσε ομοιότητες μεταξύ των ανεστραμμένων και των φυσικών αλληλουχιών και θα αποθήκευε μόνο εκείνες που παρουσίαζαν ποσοστό ομοιότητας μεγαλύτερο από 40%.

Ο υπέρογκος αριθμός αποτελεσμάτων του πρώτου προγράμματος φανέρωσε την ανάγκη δημιουργίας ενός δεύτερου προγράμματος (ο κώδικας αυτός λίγο τροποποιημένος, βρίσκεται στο Appendix A, του κεφαλαίου 3), για την επεξεργασία των αποτελεσμάτων αυτών. Το εν λόγο πρόγραμμα, σε γενικές γραμμές, δημιουργεί δύο αρχεία PDB τα οποία περιέχουν μόνο τις γραμμές εκείνες οι οποίες αντιστοιχούν στα ολιγοπεπτίδια που προκύπτουν από τη στοίχιση μεταξύ της ανεστραμμένης και της φυσικής πρωτεΐνης, πραγματοποιεί υπέρθεση των αρχείων αυτών και εντοπίζει τα στοιχεία δευτεροταγούς τους δομής. Το τελικό αρχείο που δημιουργείται περιέχει: το όνομα της πρωτεΐνης που αντιστρέφουμε (template), το όνομα της φυσικής πρωτεΐνης που προκύπτει από το BLAST (target), τις αλληλουχίες και τα όριά τους, το RMSD που προκύπτει από την υπέρθεση και τα στοιχεία δευτεροταγούς δομής τους.

Μελλοντικές μας βλέψεις ήταν:

- Η σύγκριση των στοιχείων δευτεροταγούς δομής για τον έλεγχο προτίμησης κάποιου μοτίβου
- Η δημιουργία δύο διαγραμμάτων (ένα πρώτο για RMSDs – hits, και ένα δεύτερο για RMSDs – IDs) έτσι ώστε να ελέγχαμε αν υψηλά IDs αντιστοιχούσαν σε χαμηλά RMSDs)
- Σύγκριση διαγραμμάτων με "cut-off" 40% και χωρίς "cut-off" για να δούμε αν χάνουμε σημαντικά αποτελέσματα με το συγκεκριμένο "cut-off"

Η πορεία της εργασίας αυτής διακόπηκε αναπάντεχα, καθώς κατά τη διάρκεια της έρευνάς μας, τυχαία και προς απογοήτευσή μας, εντοπίσαμε ένα άρθρο που περιείχε βασικές ιδέες του θέματός μας. Περαιτέρω έρευνα σε σχετικά άρθρα, αποκάλυψε ακόμη περισσότερες δημοσιεύσεις, οι οποίες στο σύνολό τους κάλυψαν όλη την έκταση της δικής μας μελέτης. Έτσι με την συλλογή κυρίως των παρακάτω άρθρων,

> '*Inverse sequence similarity of proteins does not imply structural similarity.*' (Lorenzen et al. 2003) [25]
>
> 'A tale of two symmetrical tails: Structural and functional characteristics of palindromes in proteins' (Sheari et al. 2008) [26]

μπήκε μια τελεία στο συγκεκριμένο θέμα της πτυχιακής εργασίας και αποφασίσαμε να στρέψουμε τα σχέδιά μας στο θέμα που αναλύθηκε εν τέλει.

-Τι πήγε στραβά;

Πριν από περίπου ένα χρόνο, όταν επιλέξαμε να μελετήσουμε την αντιστροφή των πρωτεϊνικών αλυσίδων, αμελήσαμε να δώσουμε την απαραίτητη προσοχή στην βιβλιογραφία που υπήρχε για το συγκεκριμένο θέμα. Έγινε μια αρχική έρευνα σε πιο γενικό επίπεδο, κατά την οποία αναζητήσαμε άρθρα τα οποία σχετίζονταν με το θέμα που θέλαμε να αναπτύξουμε, αλλά γρήγορα εγκαταλείφθηκε καθώς στραφήκαμε στην υλοποίηση της έρευνας. Το αποτέλεσμα έδειξε ότι προφανώς δεν αφιερώσαμε τον απαραίτητο χρόνο για τον έλεγχο των δημοσιεύσεων που υπήρχαν, όπως επίσης και ότι ίσως πραγματοποιήσαμε με λάθος τρόπο την αναζήτησή μας. Βέβαια, παρόλο που συνειδητοποιήσαμε τα λάθη μας, το αποτέλεσμα παρέμεινε αμετάβλητο.

Και επειδή εκτός από το νόμο του Μέρφυ ισχύει ότι:

“ **Αφού τα πράγματα έχουν πάει ήδη από το κακό στο χειρότερο, ο κύκλος θα επαναληφθεί** ” (το συνακόλουθο του Farnsdick στο πέμπτο συνακόλουθο), πρώτη μας έγνοια κατά την επιλογή του επόμενου θέματος διπλωματικής εργασίας ήταν ο πολύ καλός έλεγχος της βιβλιογραφίας, έτσι ώστε να μειώσουμε συντριπτικά την πιθανότητα να εντοπίσουμε τυχαία τις βασικές ιδέες της έρευνάς μας δημοσιευμένες και πάλι!

'Appendix B'

```
#!/usr/bin/perl -w
use Bio::Seq;
use Bio::SeqIO;
use Bio::Search::HSP::GenericHSP;
use Bio::Search::HSP::HSPI;

# load test.txt in $seqio_obj variable
# (test.txt includes our protein sequences in fasta format)

my $seqio_obj /= Bio::SeqIO->new(-file => "pdbaanr", -format => "fasta" );
```

```
# for each sequence
# reverse it

while (my ($seq_obj) = $seqio_obj->next_seq)
  {
       my $reversed = reverse $seq_obj->seq;

        # save the results of blast's subroutine in $report_obj variable

        $report_obj = &do_blast($reversed);



          # foreach sequence of the $report_obj, foreach $hit of the sequence, foreach $hsp of the
$hit
        # if percent identity is above 30

       while( $result = $report_obj->next_result )
         {
              while( $hit = $result->next_hit )
                 {
                     while( $hsp = $hit->next_hsp )
                       {
                          if ( $hsp->percent_identity > 40 )
                                {
                                $maxname=$hit->name;
                                $querylength=$result->query_length;
                                $maxhsp=$hsp->length('total');
                                $per_ident=$hsp->percent_identity;
                                $query_string = $hsp-> query_string;
                                $hit_str=$hsp->hit_string;
                                $query_beg = $hsp->start('query');
                                $query_end = $hsp->end('query');
                                $subject_beg = $hsp->start('sbjct');
                                $subject_end = $hsp->end('sbjct');

                                print $seq_obj->display_id();
                                print " ", $querylength;
                                print " ", $maxname;
                                print " ", $maxhsp;
                                print " ", $per_ident;
                                print " ", $query_beg;
                                print " ", $query_end;
                                print " ", $subject_beg;
                                print " ", $subject_end;
                                print " ", $query_string;
                                print " ", $hit_str, "\n";
                                }
                        }
```

```
                }
            }
  }


# blast's subroutine

sub do_blast
  {
        use Bio::Tools::Run::StandAloneBlast;
        use Bio::SearchIO;

        # we define a matrix @params where blast's results are saved

        @params = (program  => 'blastp', database => 'pdbaanr');
        $blast_obj = Bio::Tools::Run::StandAloneBlast->new(@params);

        # we define Filter query sequence -F=F
        # initially default=T

        $expectvalue = "F";
        $blast_obj->F($expectvalue);

        # $seq_obj1 takes each reversed sequence

        my $seq_obj1 = Bio::Seq->new(-seq =>$_[0]);

        # blast with $seq_obj1 variable
        # save blast results in $report_obj variable

        my $report_obj = $blast_obj->blastall($seq_obj1);
  }
```

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Anfinsen C B '*Principles that govern the folding of protein chains*' Proteins 1973 20;181(96):223-30

2. Chothia C and Lesk A M '*The relation between the divergence of sequence and structure in proteins*' EMBO J. 1986;5:823-826

3. Sander C and Schneider R '*Database of homology-derived protein structures and the structural meaning of sequence alignment.*' Proteins 1991;9:56-68

4. Rost B. '*Twilight zone of protein sequence alignments*' Protein Eng 1999;12:85-94

5. Krieger E, Nabuurs S B and Vriend G '*Homology Modeling*' Struct Bioinformatics 2003;25:507-521

6. Rose G '*Protein folding and the Paracelsus challenge*' Nat Struct Biology 1997;4:512-514

7. Dalal S., Balasubramanian S. and Regan L. '*Protein Alchemy: Changing β-sheet into α-helix.' Nat Struct Biol 1997;4(7): 548-52*

8. Kosloff M. and Kolodny R. *'Sequence-similar, structure-dissimilar protein pairs in the PDB'* Proteins 2008;1;71(2):891-902

9. Kabsch W. and Sander C. '*How good are predictions of protein secondary structure*' FEBS Lett 1983 8;155(2):179-82

10. Gan HH, Perlow RA, Roy S, Ko J, Wu M, Huang J, Yan S, Nicoletta A, Vafai J, Sun D, Wang L, Noah JE, Pasquali S, Schlick T. '*Analysis of protein sequence/structure similaritiy relationships.*' Biophys J 2002;83(5):2781-91

11. Knudsen M and Wiuf C '*The CATH database*' Hum Genomics 2010;4(3):207-12

12. Zang Y. and Skolnick J. '*TM-align: a protein structure alignment algorithm based on the TM-score*' Nucleic Acids Research 2005 Apr 22; 33: 2302-2309

13. Branden C and Tooze J '*Εισαγωγή στη Δομή των Πρωτεϊνών*' 1991 (ελληνική έκδοση 2006)

14. Dickerson R E, Kendrew J C and Strandberg B E '*The crystal structure of myoglobin: Phase determination to a resolution of 2 Å by the method of isomorphous replacement*' Acta Cryst. 1961;14:1188-1195

15. Berman H M, Westbrook J, Feng Z, Gilliland G, Bhat T N, Weissig H, Shindyalov I N and Bourne P E *'The Protein Data Bank*' Oxford Journals 1999;28(1):235-242

16. Orengo C A, Michie1 A D, Jones S, Jones DT, Swindells MB and Thornton J M '*CATH — a hierarchic classification of protein domain structures*' Structure 1997;5(8):1094-1109

17. Martinez L, Andreani R, Martinez J M '*Convergent algorithms for protein structural alignment*' BMC Bioinformatics, 2007;8:306

18. Humphrey W Dalke and Schulten A '*VMD Visual Molecular Dynamics*' J Mol Graphics 1996;14.1:33-38

19. Merritt E A and Bacon D J '*Raster3D photorealistic molecular graphics*' Methods Enzymol 1997;277: 505-524

20. Petrey D. and Honig B. '*GRASP2: visualization, surface properties, and electrostatics of macromolecular structures and sequences*.' Methods Enzymol 2003; 374: 492-509

21. Shindyalov IN. and Bourne PE. '*Protein structure alignment by incremental combinatorial extension (CE) of the optimal path*.' Protein Eng 1998;11:739-747

22. Levitt M and Gerstein M '*A unified statistical framework for sequence comparison*' Proc Natl Acad Sci USA 1998;95:5913-5920

23. Zhang Y and Skolnick '*Scoring function for automated assessment of protein structure template quality*' Proteins 2004;57:702-710

24. Panchenko AR, Wolf YI, Panchenko LA, Madej T. '*Evolutionary plasticity of protein families: coupling between sequence and structure variation.*' Proteins 2005 15;61(3):535-44.

25. Lorenzen S. et al. '*Inverse sequence similarity of proteins does not imply structural similarity.*' FEBS Lett 2003 Apr 15; 545: 105-1092

26. Sheari A. et al. '*A tale of two symmetrical tails: Structural and functional characteristics of palindromes in proteins.*' BioMed Central 2008 Jun 11;

27. Arthur Bloch '*Ο νόμος του Μέρφυ ή και άλλοι λόγοι για τους οποίους πάνε όλα στραβά.*' Γράμματα; βιβλίο 2