

זיהוי חישובי של מוטיבים רצפיים בדנ"א ורנ"א

דצמבר 2012
תשע"ג

נושא העבודה: ביולוגיה חישובית

שם התלמיד:	שם המנחה:	שם מרדכי רון	טלפון:
	ת.ז.ן:	206173379	
	תאריך לידה:	054-8196203	מספר טלפון:
	מקום העבודה/מחקר:	אהוד מנור	מנחים:
	תואר אקדמי:	אהוד מנור	מקום ביצוע העבודה:
תחומי:	מכון ויצמן למדע	מכון ויצמן למדע	
תחום הדעת של המנחה:	ביולוגיה חישובית		
כתובת:	רחוב ויצמן 11 רחובות		
מספר טלפון:	050-7519418		
דוא"ל:	ohad.manor@weizmann.ac.il		



שם בית הספר:	יח"ד מודיעין
כתובת בית הספר:	רחוב עמק בית שאן 60, מודיעין
מספר הטלפון של ביה"ס:	08-9262302

תוכן עניינים

פרק ראשון: מבוא וקירת ספרות

1	בעית המחקר והרקע למחקר
3	סקירת ספרות
8	שאלת המחקר וההשערות

פרק שני: מהלך המחקר

9	החומרים החיים הנחקרים
13	שיטות וחומרים
15	מערך המחקר

פרק שלישי: תוצאות

27	ממצאים
28	תוצאות

פרק רביעי: דיוון

33	סיכום
33	יתרונות על פני יישומים אחרים
34	שימושים
34	תוכניות לעתיד

פרק חמישי: נספחים

36	נספח 1: תוצאות הקלט הסינטטי
39	נספח 2: תוצאות קלט fqP-seq
41	נספח 3: תוצאות קלט המיקרו RNA
42	נספח 4: ביבליוגרפיה
43	נספח 5: התוכנה שכתבתי

פרק ראשון: מבוא וקירת ספרות

בעית המחבר והרקע למחקר

בעולם הביולוגי, מוטיב הוא רצף של חומצות אמינו או נוקליואוטידים שמופיע פעמים רבות ושיש חשד לכך שיש לו משמעות ביולוגית. דוגמה למוטיב של נוקליואוטידים היא: A שאחריה באה כל אות מלבד T, שאחריה באות C או G, שאחריה באה G. למוטיבים בדנ"א תפקידים שונים. חלבונים בתוך התאים נקשרים למוטיבים בדנ"א, RNA או בחלבונים אחרים כדי להתחיל פעולות שונות בתא. למשל, חלבון מסוים יכול להיקשר למוטיב בחלק מסוים בדנ"א, ולהפעיל תהליך כלשהו בתא, כמו שאיבת מימן מהסביבה, יוכל של מחוץ בתוך התא, ועוד. אם נדע מהו המוטיב שאליו נקשר אותו חלבון, נוכל בעזרה ביולוגית סינטטית לשנות אותו או למחוק אותו, ובכך לשנות תהליכיים שונים בתא, או לשנות את התנהגות התא בכלל.

מציאת מוטיבים:

מציאת מוטיבים בדנ"א היא אחת מהבעיות הקשות ביותר אליה מתמודדים מדענים העוסקים בביולוגיה חשובה. אם נקבל כמה רצפים פשוטים של אותיות, וננסה למצוא בהם את המיללים הקיימים באורך K, ספירה של כל הצירופים האפשריים באורך K תהיה הפתרון הכל' פשוט לבעה. אולם, כאשר אנו עובדים עם רצפים של דנ"א, מציאת הפתרון הופכת למסובכת הרבה יותר. יכול להיות שהמיללה שלם עברה מוציאות, נוספו לה אותיות, או שחוسرות לה אותיות, מה שմסבך הרבה יותר את החיפוש אחר המוטיבים הללו.

יש כמה גישות לתקוף את הבעיה החשובה הזאת. הגישה הראשונה היא גישה דיסקרימינטיבית (Discriminative, בעברית: מפרידה). בגישה זו אנו מקבלים שתי קבוצות של רצפים, ראשונה, יש רצפים שבובודאות מופיע בהם מוטיב מסוים, ובשנייה יש רצפים שבובודאות אותו מוטיב אינו מופיע. על ידי השוואה של שתי קבוצות אלה אנו יכולים למצוא מוטיבים שמופיעים יותר בקבוצה הראשונה בצורה משמעותית סטטיסטית. הגישה השנייה היא גישה לא דיסקרימינטיבית (-Non discriminative). בגישה זו אנו מקבלים קבוצה אחת של רצפים, ומ Chapman בה רצפים שמופיעים לעיתים קרובות. לאחר מכן אנו מפרקם את הרצפים לחלקים קטנים יותר, ובזקדים אם גם הם מופיעים לעיתים קרובות. אם לא, אזי הרצף שמצאנו הוא אכן ייחודי ומעניין אותנו. למשל, אם מצאנו שהרצף AATGAG מופיע לעיתים קרובות, נחלק אותו לחלקים בני שלוש אותיות כל אחד (, AAT, ATG, וכו') ונבדוק כמה פעמים מופיע כל שלשה. אם השלשות או חלקן יופיעו לעיתים קרובות, המסקנה היא שהרצף הגדל (AATGAG) הוא רק התרחשות אקראי של הרצפים הקטנים ולכך קרוב לוודאי שאין לו משמעות ביולוגית. לעומת זאת, אם השלשות או חלקן יופיעו לעיתים רחוקות, המסקנה היא שהרצף הגדל (AATGAG) אינו התרחשות אקראי של הרצפים הקטנים ולכך לוודאי שיש לו משמעות ביולוגית.

האלגוריתמים של מציאת מוטיבים מתחלקים לכמה סוגים. סוג אחד מtabsoo בעייר על ספירה של מילים, סוג אחר מtabsoo על חישובי הסתברות, סוג נוסף מtabsoo על למידה חישובית. האלגוריתמים SMBOSOIM על ספירה של מילים וחישובי הסתברות עושים שימוש רב בשרשראות מركוב, שהן מודל הסתברותי שבעזרתו אפשר לעקוב אחר התפתחות של תהליך ולגבה כיצד הוא יתנהג בעתיד. האלגוריתמים SMBOSOIM על למידה חישובית עושים שימוש בשיטות שונות כגון מודלים לינאריים וברשתות נוירונים (רשות של תא עצב מלאכותיים המסוגלים ללמידה ולקבל החלטות).

מוצא המוטיבים שלנו:

מטרת פרויקט המחקר שלי היא בנית כל חישובי שבעזרתו ניתן לאפיין רצפים חוזרנים קיצרים (מוטיבים) בתוך קבוצה של רצפי DN'A ארוכים יותר באופן דיסקרימינטיבי. תחילתה המטרה הייתה להוכיח שכלי שכזה אכן עובד, ולאחר מכן, המטרה הייתה להפעיל אותו על רצפים ביולוגיים אמיתיים. השאלה שעלה ניסיתי לענות במחקר היא האם באמת נתון, בעזרת אלגוריתם חישובי כלשהו, למציאת מוטיבים ברצפים ביולוגיים כלשהם. במסגרת עבודה הגמר שלי פיתחתי אלגוריתמים שונים למציאת דפוסים בתוך מחוזות גדולות של טקסט, ויישמתי את האלגוריתמים כתוכנת מחשב המונה כ-1200 שורות קוד. מוצא המוטיבים שבחرت לפתח הוא מוצא מוטיבים דיסקרימינטיבי, מפריד. המוצא שפייתחתי מקבל קבוצת רצפי DN'A, יוצר מתוכה קבוצה שלילית של רצפים, ומנסה לחפש את המוטיב. התוכנה פועלת על ידי ספירה של מילים וחישובי הסתברות. נמנעת מלהשתמש בשרשראות מרכיב מכיוון שהן מורכבות מאוד מבחינה מתמטית.

סיכום ספרות

הרעיון של חיפוש מוטיבים בדנ"א אינו חדש. מדענים מודעים לכך מוטיבים בחלוקתם שונים בדנ"א מאז גילוי הדנ"א בשנות החמישים על ידי ג'ימס ווטסון ופרנסיס קריק ולאורך השנים פיתחו כלים שונים למציאת מוטיבים. את השיטות הקיימות למציאת מוטיבים ניתן לחלק לשלווה סוגים. הסוג הראשון הוא מציאת מוטיבים בתוך גנים מסוודרים (גנום או רצף בעל משמעות בדנ"א) ביצור חי אחד. הסוג השני הוא מציאת מוטיבים באותו גנים מסוודרים בכמה יצורים חיים (ישנם גנים שמופיעים בכמה וכמה יצורים חיים. למשל, חמישים אחוז מהגנים של בני אדם מופיעים גם בדנ"א של בננה). והסוג השלישי הוא מציאת מוטיבים בתוך גנים של כמה יצורים חיים על ידי טביעות פילוגנטיות (הסביר על טביעות פילוגנטיות יבוא בהמשך). אולם, בהרבה מהכתבים על מציאת מוטיבים, חוקרים אחרים החלק את השיטות לשלווש קבוצות אחרות: מציאת מוטיבים מבוססת מילים, מציאת מוטיבים מבוססת הסתברויות, למציאת מוטיבים בעזרת מידת חישובית (רשומות נירונים, ואלגוריתמים גנטיים).

מציאת מוטיבים בגנים מסוודרים:

מצאי המוטיבים הראשונים היו כאשר התבססו על חיפוש מוטיבים בתוך גנים מסוודרים של יצור חי יחיד או כמה יצורים חיים. האלגוריתמים התבססו על רצפים שמופיעים באופן שכיח לאורכו הגנים.

מציאת מוטיבים מבוססת מילים:

ז'אק ואן הלדן ו עמיתיו פיתחו אלגוריתם מבוססת מילים שנקרא "ניתוח אוליגו" (Oligo-Analysis). האלגוריתם נבדק והראה תוצאות מאוד מדויקות. הוא הצליח לשחזר מוטיבים ידועים בדנ"א של שמרים שהתגלו על ידי ניסויים ואפילו לנבא בהצלחה מוטיבים לא ידועים. הבעה באלגוריתם זה היא שהוא מסוגל למציא רק מוטיבים פשוטים וקצרים. לפי ניתוח האוליגו, אם ניקח כמות כלשהי של רצף בסיסים באורך קבוע (נסמן את הנקודות B_n), ונסמן את ההסתברות שרצף קטן יותר s מופיע לפחות פעם אחת בכל רצף ורצף גדול $\geq n$, אז כמות המופעים הצפוי של הרצף s צריכה להיות $\approx \frac{1}{s}$. כל רצף s מקבל ציון משל עצמו M_s , שמחושב באופן הבא:

$$M_s = (N_s - Np_s) / \sqrt{Np_s(1-p_s)},$$

M_s היא בעצם כמות הפעמים שבהם ה- s יותר מופיעים של הרצף s מהציפוי (יותר מופיעים מאשר M_s). ככל M_s יותר גדול, כך גם הסיכוי שהרצף s הוא חלק מהמוטיב.

עוד אלגוריתם למציאת מוטיבים פותח על ידי האלביס ברצמה. האלגוריתם מגלה תבניות חוזרות בגנים של שמר (yeast). חוקרים בדקו את האלגוריתם וגילו שכמעט כל תבנית שהוא גילה היא חלק ממוטיב או מוטיב שלם שמוינעה בדנ"א של השמר. אלגוריתם אחר נבנה על ידי מ. סגות, והוא מבוסס על עצי סימונות. הרעיון של סגנות היה לבנות את המוטיב מהוסף להתחלה על ידי ספירה של רצפים קצרים והארכה שלהם.

מציאת מוטיבים מבוססת הסתברויות:

אחד האלגוריתמים הראשונים למציאת מוטיבים פותח על ידי מדען בשם הרץ ועמיתו. האלגוריתם קיבל קבוצה של רצפים וחיפש בהם מוטיב חוזר. האלגוריתם שופר לאורך השנים ובסיומו של דבר הוא היה מסוגל לנבא משמעות סטטיסטיות של כל מוטיב חוזר ברצפים ולתת לו ציון בעזרת סטטיסטיות סטטיסטית.

שני מדענים, דאון והאבלרד פיתחו אלגוריתם בשם NestedMICA. האלגוריתם יוצר מודל סידרתי על ידי ניתוח של רצפי DN"א ומסוגל למצוא תכניות לכמה מוטיבים בו זמניות (יתכן שתתי תכניות או יותר שייכות לאותו מוטיב). הם השתמשו גם באסטרטגיה להסקת מסקנות כדי לאחד בין המוטיבים שנתגלו למוטיב אחד ארוך (בכל הרצה נוצר לפחות מוטיב אחד זהה). המדענים השוו את האלגוריתם שלהם עם מוצאי מוטיבים שונים קיימים והגיעו לתוצאות מאד מדויקות. לעיתים אףיוו יותר טובות מהتוצאות של מוצאי המוטיבים המקוריים.

רוב האלגוריתמים מבוססים על הסתברות משתמשים ב"Expectation–maximization" ובסוג "Gibbs sampling".

בmethodology, או EM, בעברית מיקסום ציפויות, האלגוריתם בניין כך שהוא מקבל כמות מצומצמת מאוד של מידע ידוע (למשל, מוטיב מאוד חלק שנוצר על ידי ספירה של מילים וכד') וידוע שקייםות כמהות גדולה יותר של מידע לא ידוע. כדי לקבל את המידע הכי מדויק המשלב קטיעים ידועים ולא ידועים גם יחד, צריך לדעת את הקטיעים הלא ידועים (יש כאן איזו סטירה עצמית), או לדעת לנחש מהם הקטיעים הלא ידועים. אלגוריתם EM עושה בדיקות אלה. האלגוריתם עוזר לנו לנחש את החלקים הלא ידועים בצורה מאוד מדויקת.

Gibbs sampling, או בעברית דגימת גיבס, היא סוג של שרשראת מרקוב. בעזרת דגימת גיבס ניתן לחשב התפלגות כללית של כל הערכים או התפלגות חלקית של ערך אחד. כל חוליה בשרשראת היא קבוצה של ערכים. בחוליה הראשונה יהיו ערכים שנבחרו באופן אקראי, במקרה שלנו, תת רצפים שנבחרו מהתור קבוצה של רצפים ארוכים יותר. בחוליה הבאה בוחרים ערך אחד (רצף אחד) וUMB שUMB עליו פועלה שתוליה בשאר הערכים ומשנים אותו. וכך ממשיכים עד שנבחרים כל הערכים. בעזרת המודל הזה ניתן לחשב את התפלגות של הערכים ההתחלתיים ולנחש איזה מהם יכול להיות המוטיב.

הראשון שהשתמש בEM למציאת מוטיבים היה מדען בשם ס.א. לורנס. במקור בכלל השימוש לורנס בEM למציאת מוטיבים בחלבונים (חלבונים מורכבים מריצפים של חומצות וישן 22 סוגי שונים של חומצות אלו). שני מדענים בשם בייל ואלקן שיפרו את אלגוריתם EM וייצרו מוצאים מוטיבים מפורטים שנחשב לאמת המידה עבור מוצאים מוטיבים חדשים: MEME.

מתוך כל המודלים סטטיסטיים דגימת גיבס הוא הנפוץ ביותר במוצאים מוטיבים. אחד החוקרים הראשונים שיצר אלגוריתם שմבוסס על דגימות גיבס היה ס.א. לורנס (אותו לורנס שעבד עם EM). גם האלגוריתם זה היה מכון לפני החלבונים, אבל אחר כך המירו אותו לעבוד על רצפי DN"א. האלגוריתם עובר על כל הרצפים שהוא מקבל ומהפץ בהם מוטיב משותף כלשהו. עבור כל רצף עובר

האלגוריתם על כל חלק באורך W והוא מחשב את ההסתברות שהחלק הוא תצוגה כלשהיא של המוטיב. האלגוריתם מסדר את ההסתברויות בשרשראת מוקוב ושולף מתוכה את המוטיב הסופי.

מציאת מוטיבים מבוססת למידה חישובית:

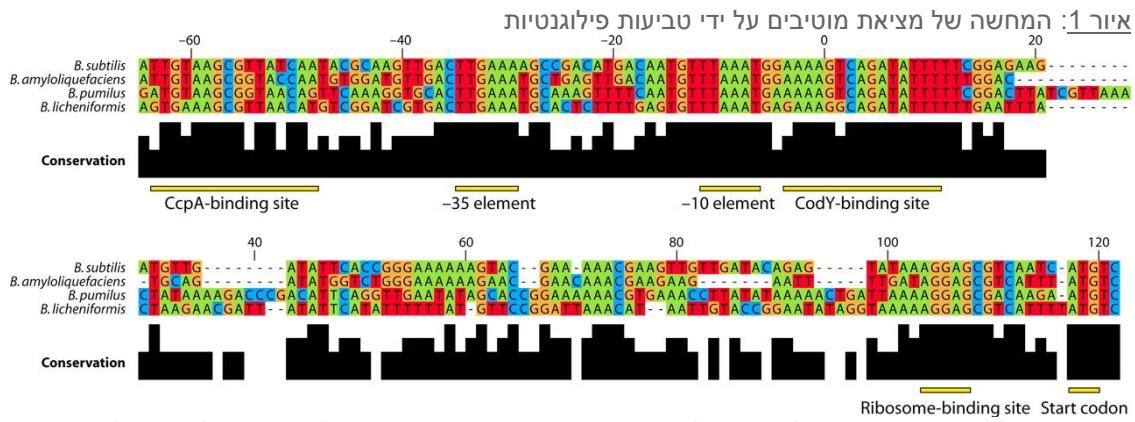
מדען אחר בשם לי (Li) ועמיתו פיתחו אלגוריתם המבוסס על למידה חישובית ואלגוריתם גנטית. אלגוריתם גנטית הוא אלגוריתם שמשתמש בתהיליכים אבולוציוניים, כמו ברירה טבעית, על מנת לפתור בעיות של ייעילות ושל דפוסים (patterns, או במקורה זהה מוטיבים). מתחילה עם אוכלוסייה בסיסית של דפוסים, מעבירים אותו "ברירה טבעית", ואז "מזרוגים" בין כל אחד אחד מהם, ווחזרים על התהיליך עד שנוצר דפו אחד. באלגוריתם של לי המוטיבים הוצגו כמטריצות PSSM (הסביר על מה זה PSSM בהמשך) והזיזוג ביניהם נעשה על ידי חיבור בין מטריצות. הברירה נעשת על ידי פסילה של PSSMs שמופיעים בהם תאים עם מידע שלילי (מידע אקראי שלא שייך למוטיב ספציפי כלשהו). האלגוריתם גם משתמש בשיטה לארגון מחדש של המטריצות על מנת להימנע מהתוצאות נקודות. המדענים קראו לאלגוריתם FMGA והראו שהוא יותר טוב בהשוואה לMEME ועוד אלגוריתמים מבוססים על דגם גיבס.

מדען אחר, שמו הוא גם לי, ועמיתו פיתחו מבנה של רשת נירונים למטרת מציאת מוטיבים בדנ"א ובחלבונים. רשת נירונים היא תוכנה המדמה חשיבה של מוח אנוש. הרשת מורכבת מחלקי, כך שכל חלק מסוגל לקבל החלטה כלשהיא לגבי הקלט שהזון לתוכו ולהעביר את הקלט לחלק אחר. הרשת לומדת את אט ומשתפרת בקבלת החלטות שלה עם הזמן (marsh כמו צור ח'). הרשת של המדענים מורכבת מכמה קומות של תת רשותות וכל רשות אחראית על חלק אחר במציאת המוטיב. הרשת מקבלת מוטיב ורצפים ומחליטה אם המוטיב שייך לרצפים. הרשת עצמה לא מורכבת בזאת החלוקה לקומות ומסוגלת לגודל בהתאם למידה שלה והחלטות קודמות. המדענים הראו שהתוכנה שלהם עובדת יותר טוב MEME ובמקומות מסוימים אפילו מסוגלת להתמודד עם רצפי דנ"א ארוכים במיוחד.

מציאת מוטיבים על ידי טביעה פילוגנטית:

טביעה פילוגנטית (Phylogenetic footprinting) היא שיטה באמצעותה ניתן למציאת מוטיבים ואלמנטים אחרים בדנ"א על ידי חיפוש של אזורים משומרים (conserved) במיוחד בדנ"א לאורך כמה שנים שונים של יצורים. היתרון בשימוש בטביעה פילוגנטית על פני חיפוש בגנים מסודרים הוא שכך לחפש מוטיב בגנים מסודרים צריך קודם כל דרך אמונה לזהות גנים מסודרים ורק אחר כך להפעיל עליהם את האלגוריתם. לעומת זאת, אם משתמשים בשיטה של טביעה פילוגנטית, ניתן לזהות מוטיבים המופיעים אפילו בגין אחד, כל עוד המוטיבים הללו נשמרים לאורך כל הרצפים המוזנים לאלגוריתם. יכולת לצבור הרבה רצפים גנטיים מספר רב של יצורים חיים באופן מהיר ויעיל מאפשרת השימוש בטביעה פילוגנטית.

השיטה הסטנדרטית באלגוריתמים מבוססים על טביעה פילוגנטית היא ליצור מערכת מיושרת (aligned) של כל הרצפים ולנסות לזהות בתוך המערכת חלקים משומרים.



באיור ניתן לראות איך עובד האלגוריתם. לוקחים את הרצפים ומישרים את כולם אחד מעל השני ולאחר מכן מחפשים איזוריים עם התאמה גבוהה (פסים שחורים מלאים, בעלי ארבע משਬצות)

קובוצה שבראשה מדען בשם ברזיקוב פיתחה אלגוריתם בשם CONREAL המבוסס על טביעה פילוגנטית. האלגוריתם משתמש במטריצות PSSM כדי לייצג מוטיבים פוטנציאליים וליצור עוגנים שליפיהם הוא מיישר את כל הרצפים. הם השוו את CONREAL לאלגוריתמים אחרים של יישור כללי (יישור של רצפים בכלל, לא רק רצפי DN'A) וראו שהם מקבלים תוצאות דומות שימושו בין יצורים דומים יחסית, כמו בני אדם ועכברים (אמנו חולקים 98% מהDN'A שלנו עם עכברים). האלגוריתם אףלו יצר תוצאות יותר טובות מאלגוריתמים אחרים בהשוואה בין DN'A של יצורים שונים, כמו בני אדם ודגים (אמנו חולקים בערך שליש מהDNA' שלנו עם דגים).

הקובוצה של המדען פ. קליפטן השתמשה בשיטה של טביעה פילוגנטית כדי למצוא מוטיבים ביצור פטריטי הדומה לשمر, *Saccharomyces*. הם חיפשו אחר טביאות פילוגנטיות בין רצפים גנטיים של שישה זנים שונים של פטרייה על ידי שימוש בכלי שנקרא W CLUSTAL. על ידי שימוש בטכניקה הפשוטה זו הם הצליחו למצוא הרבה מוטיבים משומרים, וכל זה התאפשר מכיוון שהם בין הרצפים הכל' שונים בין ששת הזנים של הפטרייה.

זוג מדענים, וואנג וסטרומו, פיתחו את האלגוריתם PHYLONET שמצאה באופן שיטתי מוטיבים משומרים באופן פילוגנטי על ידי ניתוח של איזור מסוים בהרבה גנים קשורים ויצירה של רשף של כל איזוריים הללו. התוכנה יוצרת פרופיל בשיל כל איזור ואז משתמש באלגוריתם קיימן שנקרא BLAST כדי לחפש בכל פרופיל ופרופיל מוטיבים משומרים, ואיזוריים שמקילים אותם. לאחר מכן האלגוריתם מחשב את החשיבות הסטטיסטית של כל מוטיב באמצעות עוד אלגוריתם קיימן, Karlin-Altschul. המדענים השתמשו באלגוריתם זה כדי לנתח 3524 רצפים בגנים של שמירים ולזהות רשף של 3315 איזוריים ו2962 מוטיבים. הרשף כולל כמעט את כל המוטיבים הידועים כרגע ועוד מתשעים אחוז מהאיזוריים המשומים הידועים. הם טוענים שניתן לישם את האלגוריתם גם על גנים נוספים איזוריים, כמו הגנים האנושיים (גנים הוא אוסף של כל הגנים בתא).

קרמאק ו עמיתיו פיתחו אלגוריתם סריקה, PhyloScan, המשלב בין מידע שנמצא בגנים דומים לאורך כמה בעלי חיים, למידע שנמצא בגנים דומים בבעלי חיים אחד כדי לגלו מוטיבים באופן יותר

טוב. המידע שמקורו הרבה יוצרים חיים יכול להיות מיושר, לא מיושר, או מיושר רק בחלוקת מסוימים. כאשר המידע מיושר, האלגוריתם מחשש מוטיב ולקח בחשבון גם תלות פילוגנטית בין היצורים החיים. כאשר המידע אינו מיושר, האלגוריתם מחשש את המוטיב בהינתן שהיצורים הם עצמאיים 'חלוטין' מבחינה פילוגנטית. המדענים ישוו את האלגוריתם זהה על שבע בקטניות משפחה דומה וזיהו בעזרתו מוטיבים חשובים בגנום של הבקריה ההזו.

המידע בחלק זה לקוח מחלקים [1] ו[2] בביבליוגרפיה (נספח 4).

שאלת המחקר וההשערות

המחקר שלנו הוא בתחום החישובי, ולכן אנחנו לא חקכנו שאלת ספציפית אלא רעיון כללי. התיאוריה שלנו היא שאם נספור את כל הרצפים באורך λ בשתי קבוצות של רצפי DN'A, או DN'A גדולים יותר, אחת רלוונטית למחקר, (כלומר, שאנו יודעים שקיים בה מוטיב כלשהו) והשנייה לא רלוונטית, (כלומר, ידוע שאין בה מוטיב כלשהו), נוכל למצוא את המוטיב. עבור כל רצף, חישבנו את ההסתברות שהוא יהיה שיר למוטיב, ולקחנו את רצפים שנראה שהći מתאימים. כך קיבלנו קבוצה של רצפים שמננה ניתן לבנות את המוטיב. חישבנו לקחת את הרצפים לנסوت למצוא בתוכם חפיפה (כלומר, שבשני רצפים מופיעות אותן אותיות, באחד בסוף, ובשני בתחילת). לדוגמה, AGAT וGATG, אם נמצאה חפיפה, זאת אומרת שמוטיב שלנו הוא באורך של יותר מ- λ . נאריך את רצפים שלנו ונמשיך האלה ליצור המוטיב. השאלה שעליה ניסינו לענות היא האם התיאוריה הניל'ן אכן תעבוד, והאם אפשר לישם אותה לגבי יצורים חיים שונים, ולגביהם חלקים שונים ביצורים הללו (DN'A, RNA, וכו').

במבט ראשון, נראה שהרעיון יעבד מכיוון שהרעיון הבסיסי, ספירה של רצפים וחובבי הסתבות, נראה על פניו מאד פשוט, ושאר הרעיון, החפיפה ובניית המוטיב, נראה פשוט גם הוא. אבל, יכול להיות שבמידע הביאולוגי יהיה המונח "רעש" (רצפים שמופיעים הרבה פעמים סתם), וזה האלגוריתם שלנו "יתבלבל". יחד עם זאת, אנחנו סבורים שהרעש הזה, גדול ככל שהוא, לא ישפיע על מוצא המוטיבים שלנו באופן משמעותי. ההשערה היא שנוכל עם הרעיון שלנו להגיע לתוצאות מקובלות לאילו של חוקרים אחרים.

פרק שני: מהלך המבחן

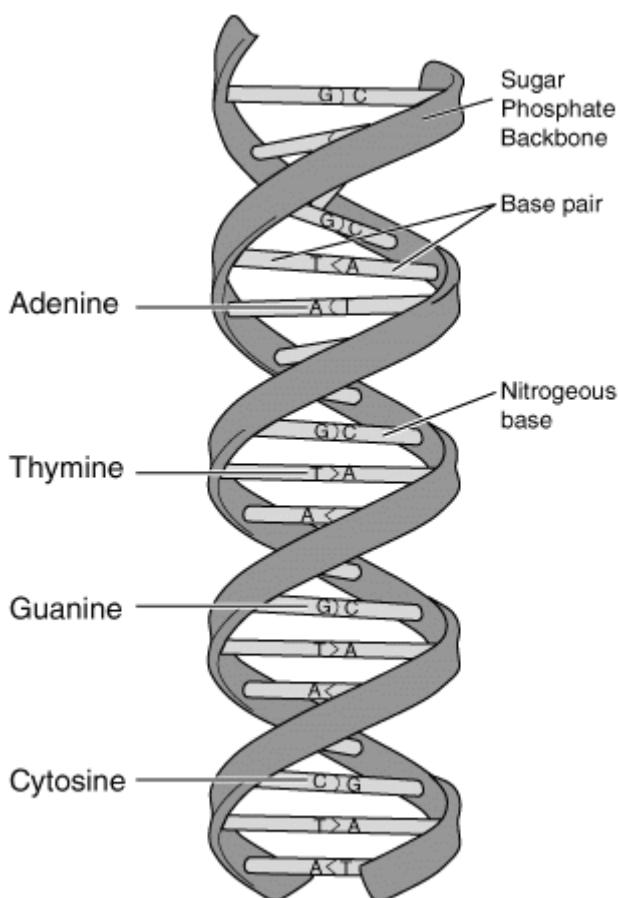
החומר החי הנחקר

החומר החי שאנו חקרנו הוא בעצם כל יצור חי. בכל תא ותא של יצור חי יש DNA שמכיל בתוכו רצפים עם מידע לגבי חלקים שונים בתא, והכל שבניו מנסה/aglowת את הרצפים הללו.

:DNA

DNA, או דנ"א (ראשי תיבות באנגלית של Acid Deoxyribonucleic, חומצה דיאוקסידיבונוקלאית) הוא מולקוללה ענקית של חומצת המורכבת ממילוי זוגות בסיסים (נוקליאוטידים) היוצרים סליל כפול. הדנ"א מכיל את כל המידע התורשתי לבניית החלבונים בתא והוא נמצא אצל כל היצורים החיים הידועים, החל מחידקים ועד לבני אדם, ואפיו בחלק מההיגנים. הדנ"א משמשו לעיתים תכופות למערכות תוכניות, מרשם או קוד מכיוון שהוא כולל את ההוראות לבניית כל רכיבי התא כמו החלבונים או מולקולות רג'ן (על הרג'ן נסbir בהמשך). כל קטעמשמעותי בדנ"א המכיל מידע לבניית חלבון או הוראות שונות אחרות נקרא גן. רצף הבסיסים המרכיב את הדנ"א יוצר גנים שונים.

הדנ"א מורכב מארבע אבני בניין, או איור 2: איור של DNA

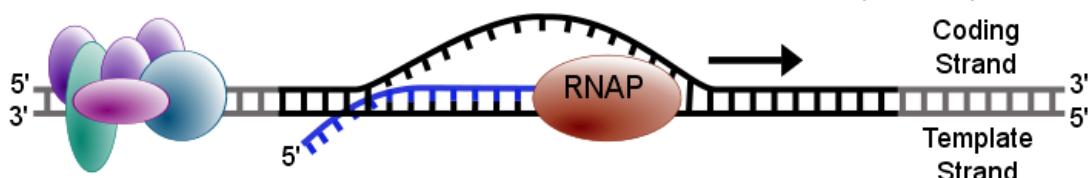


בסיום. האבניים הללו הן תת-יחידות של בסיסים חנקניים (נוקליאוטידים) שהוחולקים לשני זוגות קבועים ויוצרים קשרים כימיים אחד עם השני. הבסיסים הם: אדנין (מסומן באות A), תימין (T), גואנין (G) וציטוזין (C). אדנין יוצר קשר עם תימין, גואנין יוצר קשר עם ציטוזין. אלפיים עד מיליוןiardים של זוגות בסיסים מרכיבים את הדנ"א ויוצרים סליל כפול כך שמול כל בסיס נמצא הזוג שלו. הסליל הכפול ארוך בצורה קומפקטיבית במבנה של קרומוזומים. באירור 2 ניתן לראות סליל כפול של דנ"א. על כל סליל ישבים הבסיסים והם מחוברים בקשר כימי לבני זוגם בסליל שני. [wikipedia 4]

שיעור:

במחקר שלנו בחרנו להתמקד בתהליך השיעוטק, וברצפי שיעוטק, פרומוטרים (המווטרים שאוטם אותנו מחפשים). תהליך השיעוטק (Transcription) הוא אחד מהתהליכי הבסיסיים ביותר בתאים חיים. בתהליך השיעוטק, המידע הגנטי בדנ"א מועבר לרנ"א (RNA) על מנת שהטה תרגם מידע זה לחלבון. לפני כל גן נמצא רצף קצר הנקרא פרומוטר (Promoter, בעברית: קדם). בתהליך השיעוטק, אנזים הנקרא רנ"א פולימראז (RNA polymerase) נקשר לפרומוטר בעזרת חלבונים הנקראים פקטורי שיעוטק (Transcription Factors, בעברית: גורמי שיעוטק), ומשעטק את הדנ"א לרנ"א.

איור 3: תהליך השיעוטק



אם רואים את ה-RNAP קורא את הדנ"א (בשחור) ויוצר רנ"א (בכחול). כדי להתחיל לשעטק את הדנ"א נדרשים פקטורי השיעוטק (הבעוות בכחול וסגול משמאלי).

התהליך מתחולק לחמישה שלבים. בשלב הראשון והשני פקטורי שיעוטק נקשרים לפרומוטר ולאחר מכן הרנ"א פולימראז נקשר אליו גם הוא. לאחר מכן, הרנא"פ ("פותח" את הסליל הקפול של הדנ"א כדי שיוכל לקרוא אותו (ראה איור 3, הרנא"פ מפרק את הקשרים בין שני הסלילים לזמן קצר). בשלב השלישי הרנא"פ מטייל לאורך הסליל ומשעטק כל בסיס לבסיס חדש של רנ"א. הרנא"פ ממשיך ביצירת הרנ"א עד שהוא מגע לרצף סיום (רצף שנמצא בסוף הגן), וזה הוא מחבר חזרה את הסליל הקפול ומתרנתק ממנו.

תהליך השיעוטק הוא תהליך חשוב במחזור החיים של התא כי הוא זה שקובע בסופו של דבר אילו גנים יקבלו ביתוי – אם גן מסויים לא משועטק לרנ"א, התא לא נותן לו ביתוי. תהליך השיעוטק הוא התהליך הראשון והכى בסיסי בהתבטאות של גנים. כדי להתחיל את תהליך השיעוטק דרוש פקטור שיעוטק שאליו יקשר הרנ"א פולימראז. פקטורי השיעוטק הם חלבון שנקשר לרצפים קצרים באורך של 6-10 בסיסים שנמצאים בפרומוטר של הגן (רצפים אלו ידועים בתור אטרי קישור לפקטורי שיעוטק, Transcription Factor Binding Sites). לכל פקטור שיעוטק יש רצפים קצרים שאוטם אותו מפעולתו, וכך לא יוכל לפעול. מכיוון שהגורמי השיעוטק הם אלו שליטהים על העברת המידע מהדנ"א לרנ"א, אם נוכל לדעת לאילו רצפים קצרים כל פקטור שיעוטק נקשר, נוכל להשפיע על קישור הפקטור אל הפרומוטר. למשל על ידי הכנסת רצף קצר צזה, או על ידי מחיקתו מהפרומוטר, נוכל להפעיל או לנטרל גנים מסוימים, דבר שיכול להיות מאוד מעוניין לדוגמה בפיתוחים של תרופות עתידיות. [wikipedia 5]

פקטור שיעוטק:

בביולוגיה מולקולרית פקטור שיעוטק (Transcription factor) הוא חלבון שנקשר לרצפים ספציפיים בדנ"א (רצפים הנקראים פרומוטורים) ובכך שולט במידע הגנטי המועבר מdna'א לרנ"א,

ומרנ"א לחלבונים אחרים. פקטורי שייעתוק יכולים לעבוד בלבד, או ביחד עם חלבונים אחרים, והם פועלים על ידי גיוס של רנ"א פולימראז לשעתוק חלקים בדנ"א או על ידי חסימה שלו מאותם חלקים בדנ"א. לפקטור שייעתוק יש כמה רצפי פרומוטור שאלייהם הוא אוהב להיקשר, כמו רצפים שהזיקה שלו אליהם גובהה, וכך כל פקטור שייעתוק אחריו על כמה גנים בדנ"א. פקטורי שייעתוק חינויים לתפקיד של התא ולהפעלת גנים שונים בתא, וכתוצאה מכך הם מופיעים בכל היצורים החיים. מספר פקטורי השיעתוק משתנה מיצור ליצור, והוא באפונן פרופורציוני לגודל הגנים (כמות הגנים) ולגנטומים גדולים יש אפילו כמה פקטורי שייעתוק שונים לכל גן. ישנו בערך 2600 חלבונים בגנים האנושיים שמיכלים סממנים של פקטורי שייעתוק, ואנחנו סבורים שהם הם אכן פקטורי שייעתוק. [6] [\[wikipedia 6\]](#)

מיקו רנ"א:

מיקו רנ"א (microRNA, או miRNA) היא מולקולה חד גידלית קצרה של רנ"א באורך של כ-22 נוקלאוטידים, המשמשת לבקרה ביוטי של גנים ביצורים רבים תאימים אוקריוטים, כמו בני אדם. מיקו רנ"א משועתק מגנים בדנ"א אבל אינו מתרגם לחלבון. הוא מתחיל כמולקולה ארוכה מאוד, אך לאחר שמנצ'ר הוא נחתך ומתקצר עד שהוא מגיע לאורך מתאים, וזה הוא מעובד למבנה לולאטי והופך למיקו רנ"א סופי. למולקولات מיקו רנ"א יש רצפים קצרים באורך של כ-6 נוקלאוטידים המתאימים לחלקים של מולקولات רנ"א אחרות, mRNA (אותו רנ"א שמתורגם לחלבון בסופו של דבר). המיקו רנ"א מתחבר לmRNA בחלקים שנקבעים 3' UTR. בתוך ה3' UTR ישנו אתרי קישור למיקו רנ"א (אתרי קישור הם רצפים שמתאים לרצפים הקצרים באורך 6 שהוזכרו לעיל). ברגע שמיקו רנ"א

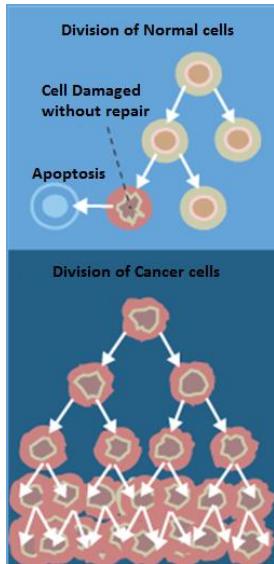
נקשר למRNA, mRNA מפסיק להיות מתרגם, והוא מתעלל חזרה לתוך התא.

למעשה, מיקו רנ"א ופקטור שייעתוק שלוטים על אותם דברים, תרגום של דנ"א לחלבון. אחד שלוט על יצירה של רנ"א שבסוףו של דבר יתורגם לחלבון, והשני קובע האם הרנ"א בכלל יתורגם לחלבון.

[[wikipedia 7](#)]

سرطان:

תא סרוני הוא תא שמתחלק בצורה מאד מהירה ופגע במערכות שונות איור 4: חלוקה של תא סרוני לאוותת תא רגיל



בגוף. תאים סרטניים מופיעים כל הזמן ביצורים חיים, ובדרך כלל הם מזוהים כתאים פגומים ומושמדים על ידי תאים שנקראים ליפוציטים. לעיתים הגוף לאצליח להזות את התאים סרטניים ואז הם מתחלים להתרפש. בימינו עוד לא נמצאה תרופה אפקטיבית נגד סרטן.

ישנו כמה תיאorias לגבי הגורמים לסרטן. תיאoria אחת היא שסרטן נגרם מווירוסים, תיאoria אחרת אומרת שהוא נגרם מביעות במערכות החיסון, עד תיאoria היא סובrat סרטן נגרם על ידי הסביבה, ועוד. תיאoria רוחות בשנים האחרונות היא שסרטן נגרם מביעות במערכות הבקרה של התא על שיעטוק ותרגום של גנים. בתאים סרטניים יש כמויות שונות של מיקרו RNA אשר בתאים אחרים, וחוקרים חוצים שזו אולי הסיבה להתקנות האלימה. [wikipedia 8]

שיטת וחומרים

מכיוון שעיקר העבודה נעשתה בתוכנות ובניה של כל חישובי מול מחשב, בהתאם את הסביבה שבה עבדתי, ואת השפה שבה כתבת.

מערכת הפעלה:

את הקוד כתבתי תחת מערכת הפעלה שנקראת לינוקס. לינוקס היא מערכת הפעלה שיצאה לאור לראשונה בשנות התשעים כפרויקט ביתי של אדם בשם לינוס טורבאלדס. לאחר מכן התפתחה מערכת הפעלה וגדלה כמות התוכנות שתמכו בה. כיום, לינוקס מותקנת על חמישה אחוזים מכלל המחשבים בעולם. השימוש העיקרי במערכת הפעלה הוא בקרב מתכנתים וכבסיס לשרתים ומחשי עלי. [wikipedia 9]

בחרתי להשתמש בלינוקס בשל הנוחות שאפשר לתכנת עליה, ומגוון השירותים שבהן היא תומכת. אני משתמש בלינוקס כבר הרבה שנים ואני מכיר את המערכת יותר טוב מכל מערכת אחרת.

שפת התוכנות:

שפת התוכנות שבחרתי לכתוב היא פרל (Perl). פרל היא שפה מפורה (Interpreted), ברמה גבוהה (High-level), מאט לרי וואל. השפה יוצאה לאור לראשונה בשנת 1987, ונהייתה פופולרית בשנות התשעים כשפה שבעזרתה ניתן לבנות אתרי אינטרנט. בנוסף לשימוש שלה בפיתוח אתרים, משתמשים בשפה מגוון תחומיים שונים כגון: תוכנות גרפי, בקרה על שירותי, חישוב של מודלים כלכליים, וביולוגיה.

בחרתי להשתמש בשפה מכיוון שמצאתי אותה נוחה לכתיבה ומכיוון שהיא שפה מפורה ולא צריכה לעבור הידור, או קומpileציה (Compilation), מה שמאפשר לבצע שינויים בקוד התוכנה באופן זריז. נכתבו בperl הרבה ספריות המאפשרות שימוש קל במודלים מתמטיים שונים לאורך השנים, דבר שמקל ומייעל את ביצוע החישובים הנדרשים בעבודה. מובנה בתוך perl גם מודל לעיבוד של כתוב שנקרא רגאקס (Regex), המאפשר חיפוש מהיר של תת רצפים בתוך רצפים גדולים יותר. רישימת התוכנות של השפה הרלוונטיות לעובדה היא כמעט אינסופית, אך אני אעוצר כאן כי יש עוד הרבה לכתוב על נושאים אחרים. [wikipedia 10]

סביבה העבודה:

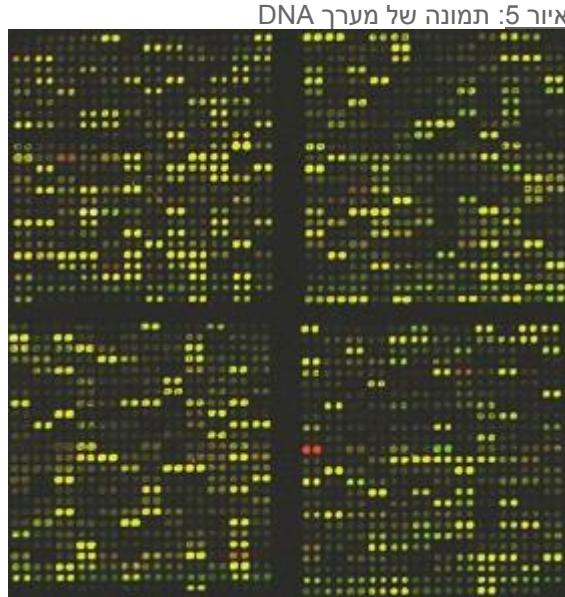
כל תוכנה נכתבת בסביבת עבודה שונה, כמו שכל מסמך נכתב בתוכנה שונה (כמו Word, Open Office, ועוד). את התוכנה שלנו בחרתי לכתוב באימקס (Emacs). אימקס היא תוכנה לכתיבה תוכנה שנכתבה על ידי ריצ'רד סטאלמן בשנת 1976. [wikipedia 11] בחרתי להשתמש באימקס מכיוון שהשתמשתי בה בעבר, היא תומכת בperl בצורה טובה, ואני מכיר אותה ומסוגל לכתוב בה מאוד מהר.

:ChIP-seq

דרך אחת להשג רצפים של דנ"א אמייתיים מטה ח' היא על ידי תהליך שנקרא ChIP-seq. ChIP-seq מתחולק לשני חלקים, הראשון, הוצאה של רצפים מאוד מסוימים מטה ח', והשני, קיראה ריצוף של הרצפים הללו. הוצאה של הרצפים מהטה נעשית על ידי ChIP-seq (Chromatin Immunoprecipitation או ChIP). קשורין פקטורי שייעתק לרצף הדנ"א באתר קישור שלהם בעזרת אור UV או בעזרת חומר שנקרו פורמליין. אחר כך שוברים את גרעין התא שבתוכו נמצא הדנ"א בעזרת גל' קול ואוספים את כל הרצפים עליהם מחוברים פקטורי השיעתק. התוצאה היא כמות רבה של שברי דנ"א עם פקטורי שייעתק קשור אליו. בסופו, משתמשים בונגדים לחלבון הפקטור שייעתק כדי להסיר אותו משברי הדנ"א. בחלק השני לוקחים את שברי הדנ"א ומרצפים אותם בעזרת מרץ גנים. התוצאה היא קבוצה של רצפים בעלי מוטיב אחד, המוטיב שלו נקשר לפקטור השיעתק. [2]

[[wikipedia](#)

:מערך דנ"א:



תמונה של מערך DNA, הצבעים השונים מייצגים את רמת הביטוי של גנים שונים

מערך דנ"א, או באנגלית DNA chip, Gene chip, DNA microarray, הוא שכבתו של גן או מחרוזות ריבועית "נקודות" דנ"א. מודענים משתמשים במערכות דנ"א כדי למדוד את רמת הביטוי של כמות גדולה של גנים בתא ובבתה אחד. כל נקודת דנ"א מכילה מאות חתיכות של רצף מסוים של דנ"א חד גידלי שנקרו גלאי. לוקחים רנ"א מתוך תאים חיים וublisherים אותו שייעתק הפור כדי להפוך אותו חזקה לדנ"א. מוסיפים לחلكי הדנ"א סימון פלוואורוסנטי ושופכים אותם על השבב. החתיכות נבדקות אל נקודות הדנ"א על השבב בדרך של זיהוג בסיסים. כשההשגב נחשף לאור, חלק הדנ"א הדבוקים יאירו,

יש יותר מחייבת כלשהיא של דנ"א, אך גם היא תאייר יותר חזק. בדרך זו, מערכי דנ"א מאפשרים לנו לראות את כמות הרנ"א בתא, וכן את כמות הגנים המפעילים בתא (ככל שיש יותר רנ"א,

כך גם יש יותר מאותו גן כשהוא מזודד). [3]

מערך המחבר

המחקר עצמו עסוק בתיאוריה שלנו והאם אפשר לישם אותה לגבי יצורים חיים שונים. בדקנו את החלק הבסיסי ביותר של התיאוריה והאם הוא עובד, ועלוי הלבשונו עוד ועוד חלקים כדי להתמודד עם מצבים שונים שעלו לאורך הדרך.

השלב הראשון: התפלגות היפרגיאומטרית ומציאת מוטיב באורך k.

התיאוריה הבסיסית ביותר שלנו בניה סביר ספירת המופיעים של כל קומבינציה של האותיות A, C, G, T באורך k (או k-mers), בקבוצה חיובית (קבוצה שבה אנחנו יודעים שיש מוטיב אחד שחוזר על עצמו), ובקבוצה שלילית (קבוצה שבה אין שם מוטיב שחוזר על עצמו). לאחר שנסקרו את כל הקימרים, התוכנה מחפשת אחר קימרים שימושיים בכמות גדולה יותר בקבוצה החיובית, מאשר בקבוצה השלילית, בזורה שהיא שימושית סטטיסטית. את המשמעותיות נחשב בעזרה התפלגות היפרגיאומטרית, שמוגדרת על ידי הנוסחה הבאה:

$$P(X = k) = \frac{\binom{m}{k} \binom{N-m}{n-k}}{\binom{N}{n}}$$

התפלגות היפרגיאומטרית היא פונקציה מתמטית העונה על סוג השאלות הבא: נניח ויש בידינו שקיי 30 אבני שחרות וכמות גדולה של אבני לבנות. אם נשלוף מהשקל 20 אבני, מה הסיכוי ש-10 מהן יהיו שחרות? הפונקציה תקבל את הפרמטרים הבאים:

1. את המספר הכללי של הרצפים (N)
 2. את מספר הפעמים שהקימר הגרפי הופיע בכל הרצפים גם יחד (m)
 3. את מספר הרצפים בקבוצה החיובית (m)
 4. את מספר הפעמים שהקימר הופיע בקבוצה החיובית הוא מקרי.
- מספר הפעמים שהקימר הופיע בקבוצה החיובית הוא מקרי. נניחו קוראים p-value. עליך שמחזירה הפונקציה, המשמעותיות, אנחנו קוראים p-value.

```
# set up a table of k-mers and their p-values
sub compute_hypergeometric_p_value {
    my ($pos_tbl, $neg_tbl, $pop_size) = @_;
    my %ret;
    foreach my $k_mer (keys %$pos_tbl) {
        my $p_value = gsl_cdf_hypergeometric_0(
            $pos_tbl->{$k_mer},
            $pop_size, $pop_size,
            $pos_tbl->{$k_mer} + $neg_tbl->{$k_mer});
        $ret{$k_mer} = $p_value;
    }
    return \%ret;
}
```

הפעולה מקבלת שתי טבלאות, אחת עם כמות המופיעים של כל קימר בקבוצה החיובית, ואחת בקבוצה השלילית, ואת הכמות הכוללת של הרצפים. הפעולה מחזירה את ערך P של כל קימר (הדגמה זו והדוגמאות שמצוירות בהמשך לקוחות כולל מתווך התוכנה שכתבת).

לקחנו אלף רצפים אקרים של A, C, T, G, ובסוף כל רצף השתלנו וריציה כלשהיא של המוטיב שלנו. הריצנו את התוכנה וראינו שהוא מזהה במדויק את כל הוריציות השונות של המוטיב:

```
ron $ ./motif.pl 6 5
{POS,NEG}.dat
I found the following k-mers
interesting:
TAGATC: 1.04422030246982e-27
GAGATC: 4.61112902167704e-29
CAGATC: 8.60385167493787e-30
AGATCC: 2.33444119100306e-28
AAGATC: 9.63100177120498e-30
AGATCT: 6.77514848803711e-66
```

הרץ' שחיפשנו היה AGATCT ועוד וריציות שלו.

השלב השני: מציאת מוטיב באורך גדול מ- a ו指点 של המוטיב.

בחלק הראשון קיבלנו שש תוצאות שונות, חלקן מתאימות למוטיב שלנו, וחלקן פחותות. התוצאות שלא מתאימות פשוט לא ישרות, לדוגמה: במקום לקביל בשורה הראשונה TAGATCT, קיבלנו C-TAGATC. שני הרצפים מאד דומים, הבעה היא שה- C , שאמור להופיע בסוף הרצף הראשון, מופיע בהתחלה שלו. הבעה כאן היא שההתוצאות ומוטיב שנמצא לא ישרים (aligned). עוד בעיה שעולה מהחלוקה הקודם היא אורך המוטיב. התוכנה בחלק הקודם לא מסוגלת למצוא מוטיבים באורך השונה מ- a (אם לדוגמה בחרנו $a = 6$, אז כל מוטיב שנחשף יהיה באורך 6). אנחנו לא יודעים את האורך האמתי של מוטיב כפי שהוא מופיע בתא, יוכל להיות שהוא באורך 7, 8 או אפילו 20.

את שתי הביעות הללו פתרנו בעזרת אלגוריתמים של יישור והארכה. פיתחנו שני אלגוריתמים שונים לשם כך, אחד קראנו **4align**, ולשני **chivalg**. האלגוריתמים מיישרים את המוטיב ובאותו זמן גם מאריכים אותו.

:4align

4align נקרא כך בשל צורת הפעולה שלו. האלגוריתם מփש תת-רצף באורך של שני שליש (מצאנו שני שליש עובד הכי טוב) מהאורך המקורי של הרצפים (אם התחילו עם אורך 6, אורך תת-רצף יהיה 4 (ומשם מגיע שם של האלגוריתם)), מיישר אותם לפי תת-רצף זהה. למשל, בששת הרצפים מהשלב הקודם, הרצף החוזר היה GATC.

```
ron $ ./motif.pl 6 5 {POS,NEG}.dat
Position of k-mer GATC in the following k-mers:
TAGATC: 2
GAGATC: 2
CAGATC: 2
AGATCC: 1
AAGATC: 2
AGATCT: 1
Average position in all k-mers: 1
```

התוכנה ממחשת את המיקום הממוצע של תת-רצף בכל אחד מהרצפים השונים. בכל הרצפים מהשלב הקודם ממוצע המיקומות היה 1 (בצד שמאל ניתן לראות את המיקום בשש מהרצפים ואת המיקום המקורי).

אחר שמצא הממוצע, התוכנה עוברת על כל רצף ובודקת את המיקום של תת-רצף ביחס למיקום הממוצע. אם תת-רצף נמצא מימין למיקום הממוצע (המיוקם שלו גדול מהממוצע), התוכנה תוסיף לרצף אות (בטיס) מצד ימין. אחרת, התוכנה תוסיף לרצף בטיס הצד שמאל. התוצאה הסופית היא קבוצה חדשה של רצפים, הפעם באורך $1 + k$ (במקרה שלנו, 7), שמיושרת יותר מהקבוצה הקודמת. התוכנה תבצע את הפעולה שוב ושוב עד שתת-רצף יופיע באותו מקום בכל אחד מהרציפים הגדולים.

```
ron $ ./motif.pl 6 5 {POS,NEG}.dat
TAGATCN: 2 - 1 = 1, rght
GAGATCN: 2 - 1 = 1, rght
CAGATCN: 2 - 1 = 1, rght
NAGATCC: 1 - 1 = 0, left
AAGATCN: 2 - 1 = 1, rght
NAGATCT: 1 - 1 = 0, left
```

למשל, אם ניקח את ששת הקיימרים מהשלב הראשוני, ונוסיף להם בטיס שמאלי או ימיני לפי ה חוקיות, נקבל שישה רצפים באורך 7 מיושרים.

```

# lengthen k-mers on either side to align them
sub ka_lengthen_k_mers {
    my ($tbl_ref, $avg_index, $file) = @_;
    my ($main_k_mer, %ret) = (get_main_k_mer($tbl_ref), %$tbl_ref);

    foreach my $key (keys %ret) {
        if (lev_index($key, $main_k_mer, 1) <= $avg_index) {
            $ret{add_base_on LEFT, $key, $file} = delete $ret{$key};
        } else {
            $ret{add_base_on RGHT, $key, $file} = delete $ret{$key};
        }
    }
    return (scalar(keys %ret) > 1) ? %ret : %$tbl_ref;
}

```

הפעולה מקבלת טבלה עם כל הרצפים, את המיקום הממוצע של תות-הרצף ואת קובץ הרצפים החיבויים. הפעולה מוצאת את תות-הרצף, ומאריכה את הקיימרים בהתאם למיקום של תות-הרצף בתוכם.

```

# push k-mers one shift into alignment
sub ka_align_once {
    my ($tbl_ref, $file) = @_;
    my $main_k_mer = get_main_k_mer $tbl_ref;

    my $avg_index = get_avg_index $tbl_ref;
    my %ret = remove_false_hits $tbl_ref, $avg_index;

    foreach my $key (keys %ret) {
        if (lev_index($key, $main_k_mer, 1) != $avg_index) {
            %ret = ka_lengthen_k_mers \%ret, $avg_index, $file;
            last;
        }
    }
    return (scalar(keys %ret) > 1) ? %ret : %$tbl_ref; # return %ret?
}

```

:pivalgn

שםו של האלגוריתם השני, pivalgn, בא מהמילה,夙同, או בעברית: ציר. האלגוריתם לוקח את כל הרצפים ומנסה להושיב אותם אחד מעל השני כך שיש חפיפה בין רוב הבסיסים (או במקרה הטוב, כל הבסיסים) של הקיימרים. אחר כך הוא מאריך כל קיימר במידה הצורך כך他会 באמת "ישבו" אחד על השני.

```

ron $ ./motif.pl 6 5
{POS,NEG}.dat
TAGATC : 2 on left
GAGATC : 2 on left
CAGATC : 2 on left
GATCCCT : 2 on right
AAGATC : 2 on left
AGATCT : 1 on right, 1 on left
...
CAGATCNN: 2 on left
NNGATCCT: 2 on right
AAGATCNN: 2 on left
NAGATCTN: 1 on right, 1 on left

```

למשל, אם נבחר שני רצפים מהתוצאות הקודמות שלו, TAGATC, וAGATCT, נראה ש כדי לחפות ביניהם צריך להזיז אחד מהם בסיס אחד שמאליה או ימינה (ראה דוגמיה משמאלי שהוא פלט מתוכנה שככetta). לאחר שהאלגוריתם בודק וראה איך צריך "להושיב" את הקיימרים, הוא הולך ומוסיף בסיס לכל קיימר בהתאם (על הוספת הבסיסים ארchip בחלק הבא). התוצאה של הפעיק pivalgn היא קבוצה חדשה של קיימרים ארוכים יותר (במקרה שלנו במקרה לקבל מוטיב באורך של, קיבלו מוטיב באורך 7), ולא צריך להריץ אותו כמה פעמים כדי לקבל תוצאה סופית (את החגיגת 4alg).

לקבל תוצאה סופית (את החגיגת 4alg) צריך להריץ וכמה פעמים עד שתות-הרצף יהיה במקום בכל קיימר).

```

# find pivot for pivalign
sub find_pivot {
    my ($left, $right) = @_;
    # !! left must be LONGER or EQUAL !!
    my ($lenl, $lenr) = (length $left, length $right);
    my ($mdev, $end) = ($lenr - 3, 0);
    my (@vals1, @valsi, @valsr);

    for (my $dev = 1; $dev <= $mdev; $dev++) {
        my $compl = substr $left, 0, $lenr - $dev;
        my $compr = substr $right, $dev;
        push @vals1, distance $compl, $compr;
    }
    for (my $diff = $lenl - $lenr; $diff >= 0; $diff--, $end++) {
        my $comp = substr $left, $end, $lenr;
        push @valsi, distance $comp, $right;
    }
    for (my $dev = 1; $dev <= $mdev; $dev++) {
        my $compl = substr $left, -( $lenr - $dev );
        my $compr = substr $right, 0, $lenr - $dev;
        push @valsr, distance $compl, $compr;
    }
    return get_index \@vals1, \@valsi, \@valsr;
}

הפעולה מקבלת שני קיימרים באורך  
מסויים. הפעולה בודקת את החפיפה  
של שני הקיימרים לצד ימין, לצד  
שמאל, ובאמצע ושולחת את  
התוצאות לפועלה אחרת  
(get_index). הפעולה שניה  
מחפשת את החפיפה הטובה ביותר  
של הקימר הראשון בשני, ומחדירה  
כמה בסיסים צריים להוסיף לקימר  
שניהם על מנת שייפר.

```

```

# align k-mers according to the pivalign algorithm
sub pa_align_k_mers {
    my ($tbl_ref, $file) = @_;
    my %ret = %$tbl_ref;
    my (@lengthen, @temp);

    my @k_mers = sort {$tbl_ref->($a) <=> $tbl_ref->($b)} keys %$tbl_ref;
    my $main_k_mer = shift @k_mers;
    foreach my $k_mer (@k_mers) {
        my ($left, $right, $pivot) =
            pa_lengthen_k_mers $main_k_mer, $k_mer, $file;
        $ret{$left} = delete $ret{$main_k_mer};
        $ret{$right} = delete $ret{$k_mer};

        if (length $main_k_mer < length $left) {
            my $diff = length($left) - length($main_k_mer);
            foreach my $k_mer (@lengthen) {
                my $new = $k_mer;
                for (my $i = $diff; $i--;) {
                    $new = add_base_on( ($pivot > 0)?RIGHT:LEFT, $new, $file);
                }
                $ret{$new} = delete $ret{$k_mer};
                push @temp, $new;
            }
            @lengthen = @temp;
            undef @temp;
        }
        push @lengthen, $right;
        $main_k_mer = $left;
    }
    return %ret;
}

```

הפעולה מקבלת טבלה
ובה כל הקיימרים, ואת
הקובץ עם הקלט
החיבוי. הפעולה בוחרת
את אחד הקיימרים
כךemer ציר, וחופפת בין
לבין שאר הקיימרים.
לאחר החפיפה הפעולה
מאריכה ומישרת את כל
הקיימרים בהתאם.

השווואה קצרה:
לאחר בדיקות וניסויים עם שני האלגוריתמים גילינו ששניהם יעילים באותה מידת alg4 טוב יותר
במציאת מוטיבים קצרים, ואילו pivalign טוב יותר במצבת מוטיבים גדולים. שני האלגוריתמים
מהירים באותה מידת כשהם מוצאים על קלט מצומצם יחסית, אך כשייר הרבה קיימרים, alg4
נוטה ללקחת הרבה יותר זמן.

הוסףת של בסיסים:

כשתיארתי את שני האלגוריתמים ציינתי שהם מօסיפים בסיסים לקיימרים על מנת להאריך אותם. כדי
להוסיף בסיס לצד קימר מסוים צריך לדעת איזה בסיס להוסיף, וכן צריך לדעת איזה בסיס יופיע
לצד אותו קימר בקובץ הקבוצה החיבוי. לשם כך כתבתי פועלה שבודקת לכל המופיעים של אותו

קיימר בקובץ, כמה פעמים מופיע לידיו כל בסיס (T, C, G, A), וכך היא בוחרת איזה בסיס להוסיף לקיימר.

למשל אם הקיימר שלנו הוא AGATCT, אנחנו רוצים להוסיף לו בסיס הצד ימין, נסתכל בקובץ החיבוי ונספור כמה פעמים מופיע כל אות מצד שמאל של הקיימר. במקרה אחר, נחפש כמה פעם מופיע כל רצף, AGATCT~~A~~, AGATCTG, AGATCTC, AGATCTA. לאחר שספרנו את המופיעים של כל רצף קיבלנו שהרצף AGATCTA מופיע הכי הרבה פעמים, ולכן נבחר להוסיף לקיימר את האות A מצד ימין.

```
# add base to k-mer according to the supplied file on either left or right
sub add_base_on {
    my ($side, $k_mer, $file) = @_;
    my ($A, $T, $C, $G, $max);
    my @lines;

    open my $fh, "<", "$file";
    while (<$fh>) {
        chomp;
        push @lines, $_;
    }
    close $fh;

    if ($side eq LEFT) {
        $A = grep /A($k_mer)/, @lines;
        $T = grep /T($k_mer)/, @lines;
        $C = grep /C($k_mer)/, @lines;
        $G = grep /G($k_mer)/, @lines;
        $max = max $A, $T, $C, $G;
        return ($max eq $A ? 'A' : $max eq $T ? 'T' : $max eq $C ? 'C' : 'G');
    }
    $k_mer;
}
```

דוגמה זו מציגה חלק מפעולות הוסףת הבסיס. הפעולה מקבלת את הצד שאלויו צריך להוסיף את הבסיס, את הקיימר ואת הקובץ החיבוי. היא סופרת את המופיעים של כל אות ומחייבת את הקיימר המודרך.

שלב שלישי: יצירת מטריצת PSSM והציג גרפית של המוטיב.

מכיוון שקיבלנו כמה רצפים המרכיבים את המוטיב שלנו (כמה רצפים עליהם נקשר פקטור השיעוטק), אנו צריכים לאחד ביניהם וליצור מודל שנוח לעבוד איתנו. לשם כך בחרנו במטריצת PSSM. PSSM הינה דרך לייצג את המידע לגבי הזיקה (affinity) של פקטור השיעוטק לרצפים שונים (הרצפים המרכיבים את המוטיב). בכל תא ותא של המטריצה נמצאים אחוזי ההסתברות שאות מסוימת (T, A, C, G) תופיע במקום מסוים לאורך הרצף. באירור 6 (בעמוד הבא), ניתן לראות שבעומדה הראשונה ברצף יש סיכוי של 68% שתופיע האות T, בעומדה השנייה ברצף ישנו סיכוי של 74% שתופיע T, בשלישית 57% שתופיע G, וכו'. לכן, לפי PSSM זה, הרצף עם הסיכוי הגבוה ביותר להקשר (נקרא גם רצף הקונצנזוס, consensus sequence) לפקטור השיעוטק הוא TTGACA.

איור 6: דוגמא לutable PSSM

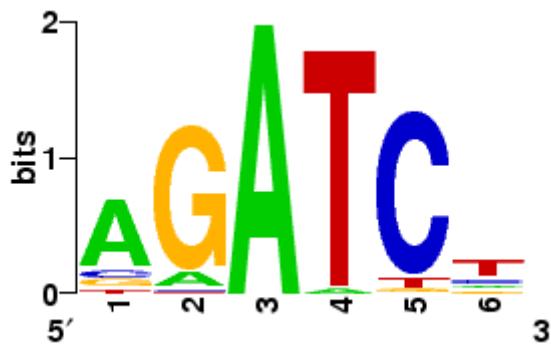
	1	2	3	4	5	6
A	0.10	0.07	0.10	0.54	0.17	0.49
C	0.10	0.09	0.13	0.19	0.54	0.14
G	0.12	0.09	0.57	0.12	0.12	0.17
T	0.68	0.74	0.20	0.15	0.18	0.20
	T	T	G	A	C	A

פעולת ייצרת ה-PSSM. הפעולה מקבלת טבלה עם כל הקיירים. הפעולה סופרת כמה פעמים כל אות מופיעה במקומות מסוימים בק'ימר (פעם במקומות הראשון, ואז בשני, וכו'). מחשבת את הסתברות שתופיע אותה אות (סה"כ האותיות / כמות המופעים). הפעולה רושמת את ההסתברות היחסית במטריצת PSSM ומחזירה אותה.

```
# make a position specific matrix out of the k-mer list
sub make_pssm_array {
    my $tbl_ref = shift;
    my ($k, $c) = (length ~~(keys %$tbl_ref)[0],
                    scalar keys %$tbl_ref);
    my @pssm;

    for (my $i = 0; $i < $k; $i++) {
        $pssm[$i] = { 'A' => 0, 'C' => 0, 'G' => 0, 'T' => 0 };
    }
    for (my $i = 0; $i < $k; $i++) {
        foreach (keys %$tbl_ref) {
            $pssm[$i]{ substr $_, $i, 1 }++;
        }
        foreach (qw(A C G T) ) {
            $pssm[$i]{$_} = $pssm[$i]{$_} / $c;
        }
    }
    return @pssm;
}
```

לאחר שייצרנו את ה-PSSM אנו צריכים דרך להציג איור 7: תוצר של seqLogo



אותה באופן גרפי (אי אפשר לתת למשתמש אוסף של הסתברויות ולצפות שהוא בין איך נראה המוטיב). לשם כך בחרנו לשימוש בתוכנה מוכנה(seqLogo). התוכנה מקבלת את ה-PSSM ויזרת ממנה תמונה יפה וצבעונית שניית באמצעוולה לראות בבירור את המוטיב.

דוגמא לתמונה שיצר seqLogo בשביל המוטיב שלנו. ניתן לראות בברור שרץ הקונצנזוס הוא AGATCT, ואת חלוקת הבסיסים בכל מקום ומקום.

<http://weblogo.berkeley.edu>

שלב רביעי: השמטה של מידע "זבל" שנוצר על ידי ההארכה, וקיצור המוטיב. במהלך היישור יתכן ויתווסף למוטיב שלנו ערכים לא רצויים בקצבות שלו. יכול להיות שהמוטיב הוא באורך של שבעה בסיסים, אבל כדי לישר את כל הקיימרים, היה צריך להאריך את המוטיב לאורכו של תשעה בסיסים. זאת אומרת שישנם שני ערכים בקצבות של PSSM (המוטיב) שהם לא רצויים. כדי לדעת אילו מהערכים לא רצויים, אנו משתמשים על הסתברויות שלהם. אם ההסתברויות מראות על מידת אנטרופיה (אקריאיות) גבוהה, אנחנו לא רוצים את הערך וניתן להשטייט אותו. למשל, להסתברויות $T = 0.25$, $C = 0.25$, $G = 0.25$, $A = 0.25$, או $A = 0.75$, $C = 0.25$, $G = 0$, $T = 0$ ושהאות שמצויה אכן משנה את הדרך שבה נראה המוטיב. כדי להבין אילו הסתברויות מראות על מידת אקריאיות נמוכה או גבוהה, בחרנו להשתמש בפונקציה שנקראת סטיית תקן (Standard deviation).

סטיית תקן:

סטיית תקן היא פונקציה מתמטית שמחשבת את המרחק בין כל אירור 8 הנוסחה לסטיית תקן $\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2}$ הערכים בקבוצה לבין ממוצע הערכים הללו. כדי לחשב סטיית תקן צריך תחילה לחשב את ממוצע הערכים של הקבוצה. לאחר חישוב הממוצע מחשבים את ההפרש של כל ערך מהממוצע, מעלים את התוצאות בריבוע, מחשבים את ממוצע הריבועים ובסוף מוציאים מהכל שורש. לדוגמה, עבור הערכים 0.45, 0.25, 0.05, 0.25, 0.45, נחשב תחילה את הממוצע: $(0.05 + 0.25 + 0.25 + 0.45) : 4 = 0.2475$

אחר כך נחשב את ההפרש של בין כל ערך והממוצע, ונעלם את ההפרש בריבוע:

$$(0.2475 - 0.05)^2 = 0.03900625$$

$$(0.2475 - 0.25)^2 = 0.00000625$$

$$(0.2475 - 0.45)^2 = 0.04100625$$

נחשב את ממוצע הריבועים ונוציא ממנו שורש:

$$(0.03900625 + 0.00000625 + 0.00000625 + 0.04100625) : 4 = 0.02000625$$

$$\sqrt{0.02000625} \approx 0.14$$

קיבלנו שסטיית התקן היא 0.14.

```
sub test_low_entropy {
    my $tbl_ref = shift;
    my @arr;
    foreach (qw(A C G T)) {
        push @arr, $tbl_ref->($_);
    }
    my $dev = stdev \@arr;
    if ($dev > 0.15) {
        return 0;
    }
    return 1;
}
```

הפעלה מקבלת עמדה במטריצת PSSM ומחשבת את סטיית התקן (stdev) שלה. אם סטיית התקן מעל הסף, הפעלה מחזירה ערך חיובי, ואם אחרת, הפעלה מחזירה ערך שלילי.

לאחר שאנחנו מחשבים סטיית ציר לבוחר סף (threshold) שמעליו כל סטייה נחשבת לגודלה מאוד (ולכן מידת האקריאיות נמוכה מאוד). באлогוריתם שלנו בחרנו במספר 0.15 כסף סטייה. התוכנה בודקת בקצבות PSSM אם ישנים עמודות

עם אקריאיות גבוהה, אז היא משמשת אותן.

```

sub clean_pssm_array {
    my $arr_ref = shift;
    my %a = %{$$arr_ref}[0];
    while (test_low_entropy(\%a) ) {
        shift @$arr_ref;
        %a = %{$$arr_ref}[0];
    }
    my %b = %{$$arr_ref}[-1];
    while (test_low_entropy(\%b) ) {
        pop @$arr_ref;
        %b = %{$$arr_ref}[-1];
    }
    return $arr_ref;
}

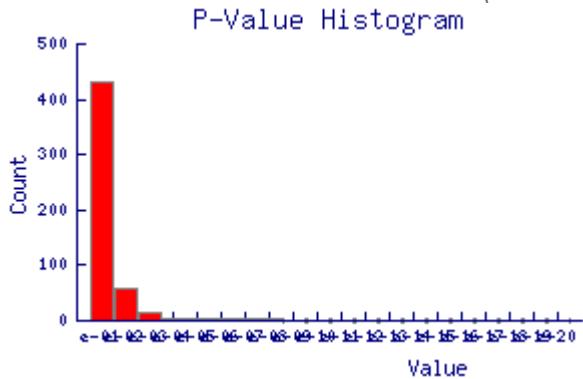
```

הפעולה מקבלת את מטריצת PSSM, היא לוקחת פעם עמודה מצד שמאל ופעם עמודה מצד ימין ובודקת אם מידת האקרαιות בעמודות גבורה. אם כן היא משנימה אותן ובוחרת עמודות חדשות. הפעולה תחזיר על הבדיקה עד שלא ימצאו עמודות עם אקרαιות נמוכה.

שלב חמישי: מציאת סף (Threshold) לרלונטיות הקיימרים.

בשלב הראשון, כשחישבנו את התפלגות ההיפרגיאומטרית של כל קיימר, קיבלנו ערכים המוראים על רלונטיות הקיימר למוטיב (לערך אנחנו קוראים value-k). ככל שהערך שקיבלנו היה יותר קטן, כך הקיימר יותר רלונטי. לפני שמדוברים את הקיימרים לפועלות היישור צריך לבחור בין הקיימרים הרלונטיים לאילו שלא. עד כה, בכל הניסויים שלנו, בחרנו את כל הקיימרים העשויים value-k שלהם קטן מ- 10^{-5} . ברור שהספ' הזה משתנה בין כל פקטור שייעtopic וביין כל מוטיב, ולכן אנחנו היינו צריכים לפתח דרך למציאת הספ' הזה.

לפני שבכלל חשבנו על דרך למציאת הספ', יצרנו [איור 9](#): ההיסטוגרמה של המוטיב שלו מההתחלת היסטוגרמה של כמות הקיימרים כפונקציה של -k value



ההיסטוגרמה של הפס הראשון יקבע על ידי כמות הקיימרים עם -k value שלהם. גובהו של הפס השני יקבע על ידי כמות הקיימרים עם value-k של הפס השני. גובהו של הפס השלישי יקבע על ידי כמות הקיימרים עם value-k של הפס הרביעי וכו'.

באיר 9 ניתן לראות שרוב הקיימרים מרכזים במקומות הראשונים, והספ' הוא בערך 10^{-4} .

הסתכלות מהירה על ההיסטוגרמה שבאיור, ועל עוד ההיסטוגרמות אחרות מורה לנו על דפוס חוזר. אם נסתכל על העמודות בעקבותיה, נראה שהעוקמה מקבלת צורה היפרבולית בתחילת ושהיא הולכת ומתיישרת מעבר לנקודת הספ'. אם נצליח להבין היכן העוקמה מתחליה את ההתישרות שלה, נוכל לדעת מה יהיה הספ'. כדי למציאת התיששות, השתמשנו בפונקציה מתחום הסטטיסטיקה שנקראת R^2 .

R^2 :

בתחום הסטטיסטיקה וניתוח גרפים משתמשים ב R^2 (הנקרא גם *Coefficient of determination*) כדי למצוא התאמה בין נתונים קיימים לנתונים חדשים, עתידיים. הערך של R^2 עונה על השאלה

הבא: אם נמתח קו ישר בין קצה אחד של הגרף שלנו, לקצה השני שלו ונבחר נקודה אקראיית על הקו, מה ההסתברות שהנקודה שבחרנו תתאים לשאר הנתונים. ככל שהנתונים יוצרים קו ישר יותר כך הערך של R^2 יתרחק יותר מ-1. מכיוון שאנו מחפשים את הנקודה שבה עיקומת היחסוטוגרמה מתוישרת, בחרנו להשתמש בפונקציית R^2 .

```
ron $ ./motif.pl POS.dat -n NEG.dat
10^-1: 0.144993273929523
10^-2: 0.706880994383275
10^-3: 0.919641802193044
10^-4: 0.961218042544887
10^-5: 0.968784826626188
10^-6: 0.983268254193052
10^-7: 0.990467221095982
...
selected t=10^-4
```

התוכנה מחשבת את ערך R^2 תחילת עבור הערכים m^{-1} עד 10^{-20} , לאחר מכן m^{-2} עד 10^{-20} , וכו', ובוחרת את הערך הראשון שעולה מעל 0.95, כפי שראים בדוגמא (לאחר בדיקות נוספת המספר שונה מ-0.95 ל-0.69 מכיוון שזאת הוא התאים לניסויים על חומר ביולוגי).

הפעולה למציאת הסוף בעדרת R בריבוע. הפעולה מקבלת את היחסוטוגרמה, מחשבת את ערכי R בריבוע כדי שמתואר לעיל, ושומרת את הערכים ברשימה. לאחר מכן הפעולה מעברת על הרשימה ומגדירה את המיקום הראשון שבו מופיע R בריבוע גדול מ-0.69.

```
# get p-value threshold using R squared - faster, less accurate?
sub get_coef_pval_threshold {
    my $arr_ref = shift;
    my ($x, $y);
    my @R2;
    for (my $i = 0; $i < scalar(@$arr_ref); $i++) {
        push @$y, scalar($arr_ref->[$i]);
        push @$x, $i + 1;
    }
    while (scalar(@$y) > 1) {
        my $ss_resid = (gsl_fit_linear($x, 1, $y, 1, scalar @$y))[-1];
        my $ss_total = gsl_stats_tss($y, 1, scalar @$y);
        push @R2, 1 - $ss_resid/$ss_total;
        shift @$y;
        shift @$x;
    }
    for (my $i = 0; $i < @R2; $i++) {
        return $i + 3 if (@R2[$i] > 0.69);
    }
    return -1;
}
```

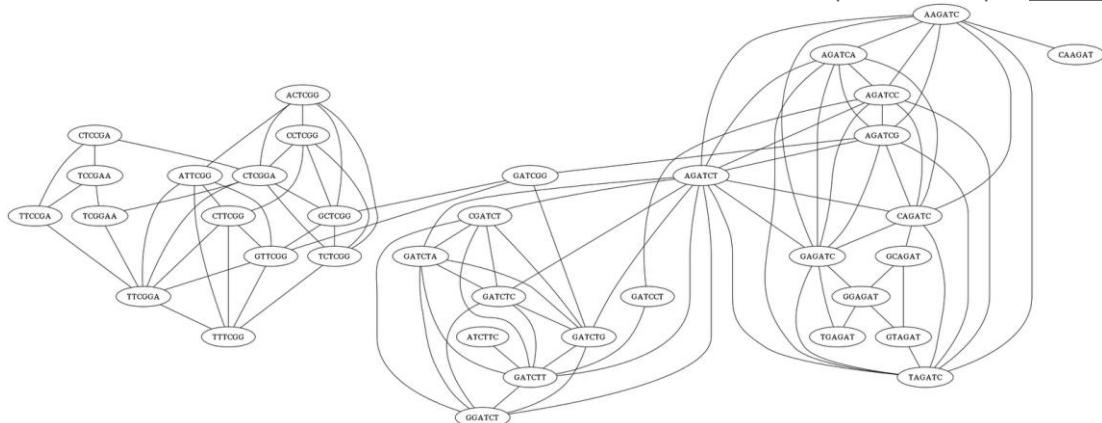
שלב שישי: קלאסטרינג (clustering) ומיצאה של יותר ממוטיב אחד.

לאחר שבנו תוכנה שמסוגלת למצוא מוטיב חוזר בתוך הרצפים שאנו נותנים לה, נשarra לנו בעיה אחת שבה צריך לטפל: מה קורה אם ישנו כמה מוטיבים בקבוצת הרצפים החביבית (יתכן שפקטור שיעתק נקשר לכמה רצפים שונים)? אם נתעלם מהבעיה זו וניתור PSS מכמה מוטיבים, נקבל מוטיב אחד משולב ומבולבל, ויהיה מדובר קשה להפיק ממנו מידע. כדי לפתור את הבעיה זו, נשתמש במושג הנקרא קלאסטרינג (Clustering, בעברית: קיבוץ באשכולות). אלגוריתם קלאסטרינג הוא אלגוריתם הממיין דוגמאות לתוך מספר קבוצות, על פי דמיון בין דוגמאות ועל פי פרמטרים וקריטריונים שונים. נסינו לבנות תחילת אלגוריתם המבוסס על גрафים מתמטיים, אבל הרעיון נכשל, אז עברנו לאלגוריתם שمبוסס על תת-רצפים.

:graphclus

הרעין המקורי שלו חשבנו למציאת כמה מוטיבים הוא לקחת את כל קיימרים הרלוונטיים ולסדר אותם בגרף (גרף על פי תורת הגרפים, קבוצה של אובייקטים המוחברים בחיצים אחד לשני). התיאוריה הייתה שם נסדר את הקיימרים בגרף ונחבר כל שני קיימרים דומים (קיימרים שההבדל ביניהם הוא אחת בחיצים נקבע פיזור שמדגיש כל מוטיב בנפרד.

איור 10: גרף של סידור הקיימרים



התוצאה לאחר סידור הקיימרים, אפשר לראות את הקיימרים ממוטיב אחד מסודרים ב>Show בצד שמאל, וקיימרים ממוטיב אחר מסודרים ב>Show בצד ימין.

הדבר היחיד שנותר לעשות עכשו הוא חלוקת הגרף לכמה חלקים (כמה קלאסטרים/מוטיבים). חשבנו שאם נבחר קיימר אקראי כנקודת מוצא, ונטיל בגרף מספר מוגבל של צעדים, נגיע בסופו של דבר לקיימר מסוים. אם נחזור על הפעולה אלף פעמים נקבל חלוקה ברורה של הקיימרים. הבועה ברעיון זהה היא שזמן הריצה שלו יהיה מאוד גדול (כשתי דקוט בשבייל קלט פשוט על שני מוטיבים), וכמוות המוטיבים שנוצרו היתה גדולה מיידי (בערך שבעה עשר מוטיבים במקום שניים).

:4clus

לאחר שהרעיון הראשון נכשל, עברנו לרעיון שմבוסס על אותו עיקנון כמו Chong 4. לפני המין לקלאסטרים, יש לנו רשימה של קיימרים (לא מיושרים) באורךSSH. אם נסתכל על הקיימרים בכל קלאסטר בגרף שלנו (באיור 10), נראה שהיא שימושת לכל קלאסטר הוא תת-רצף באורך של ארבע בסיסים. אם נצליח למיין את הקיימרים על פי תת-רצפים, אנחנו אמורים לקבל שני קלאסטרים (במקרה שלנו), אחד לכל מוטיב.

התוכנה לוקחת את הקיימרים ומסדרת אותם בטבלה. היא בוחרת כל פעם קיימר אחר ומכניסה לשורה מסוימת את כל הקיימרים שונים ממנו באות או שניים. אחר כך התוכנה מוצאת את תת-הרצף החוזר בכל שורה, ומאחדת שורות עם תת-רצפים דומים. בסופו של דבר נשאר מספר שורות כמספר הקלסלטים. התוכנה מעבירה את הקלסלטים החדשים לאלגוריתם ההארכה ומשם היא יוצרת מוטיבים.

לדוגמה, נ取 טבלה פשוטה:

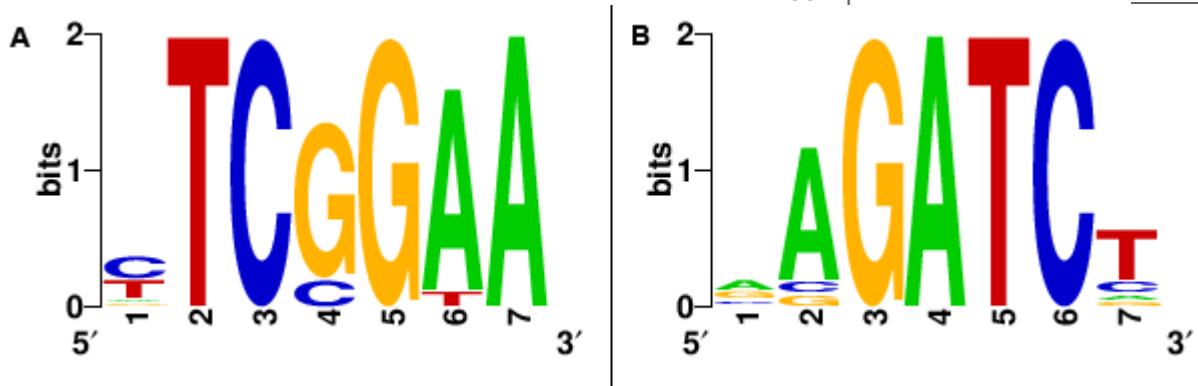
AGATCT	GGATCT	CGATCT	AGATCG
GAGATC	GTGATC	GCGATC	CAGATC
CTCGGA	ATCGGA	GTCGGA	CTCGGA

נסתכל על השורה הראשונה, כל הקיימרים בשורה זו דומים לפחות ארבע אותיות לרצף הראשון, האדום. אם נבחן את הקיימרים לעומק נראה שתת-רצף החוזר הוא GATC. געשה את אותו הדבר לשורה השנייה, נבחן את הקיימרים ונראה שגם פה תת-רצף הוא GATC. במצב זה, האלגוריתם יאחד בין שתי השורות, ונקבל שורה אחת ארכוכה. נסתכל על השורה השלישייה. בשורה זו תת-רצף החוזר הוא TCGG. הרצף זהה שונה מהרצף הראשון ולכן לא נאחד בין השורות. בסוף קיבלנו טבלה עם שתי שורות:

GAGATC	GTGATC	GCGATC	CAGATC	AGATCT	GGATCT	CGATCT	AGATCG
CTCGGA	ATCGGA	GTCGGA	CTCGGA				

כל אחת מהשורות היא קלאסטר בפני עצמה. קיבלנו שני קלסטרים, או שני מוטיבים.

איור 11: תוצאות אמיתיות לאחר הקלאסטרינג



באיזור רואים בבירור את שני המוטיבים למחרות שהם פחות מדויקים מתוציאות עם מוטיב אחד. לмотיב B (הימני) נוסף בסיס בצד השמאלי (ראה לעיל בתיאור "השלב השלישי" איך המוטיב אמרור להראות).

```

sub sort_dist {
    my $tbl_ref = shift;
    my @ret;
    foreach my $row (keys %$tbl_ref) {
        foreach my $col (keys %{$tbl_ref->($row)}) {
            $tbl_ref->($row)->($col) = distance $row, $col;
        }
    }
    while (%$tbl_ref) {
        my $row = (keys %$tbl_ref)[0];
        my %temp;
        foreach my $col (keys %{$tbl_ref->($row)}) {
            if ($tbl_ref->($row)->($col) && $tbl_ref->($row)->($col) <= 2) {
                $temp{$col} = 1;
                delete $tbl_ref->($col);
            }
        }
        $temp{$row} = 0;
        delete $tbl_ref->($row);
        push @ret, \%temp;
    }
    return @ret;
}

```

```

sub cluster {
    my $tbl_ref = shift;
    my @groups = sort_dist table_to_matrix $tbl_ref;
    foreach my $left (@groups) {
        my $left_k_mer = get_main_k_mer $left;
        foreach my $right (@groups) {
            next if ($left == $right);
            my $right_k_mer = get_main_k_mer $right;
            my $dist = distance($left_k_mer, $right_k_mer);
            if ($dist <= 2) {
                @{$left}{'keys %$right'} = values %$right;
                my $i = 0; $i++ until ($groups[$i] == $right);
                splice @groups, $i, 1;
            }
        }
    }
    foreach (@groups) { # copy the p-values over
        @{$$_}{'keys %$-' } = @{$$tbl_ref}{'keys %$-' };
    }
    return @groups;
}

```

הפעולה מקבלת רשימה של קי'רים, מסדרת אותם בטבלה ומאחדת שורות בטבלה כמו שמתואר לעיל. הפעולה מחדירה מערך ובתוכו הקלסיטרים.

שלב שביעי: ייצרת קובץ שלילי מתחום הקובץ החיווי.

עד כה הזמן לתוכנית שלו קובץ חיווי של רצפים וקובץ שלילי של רצפים, אבל מה אם אין לנו קובץ שלילי? הרו מהדנ"א אנחנו יכולים לשולף רק רצפים שידוע לנו שקיים בהם מוטיב כלשהו אבל אין לנו אפשרות להפיק מהדנ"א "קובץ שלילי" (בעצם, אנחנו יכולים רק להגיד מה יש בדנ"א, ולא יכולים לומר מה אין בו). לשם כך יצרנו עוד קטע בתוכנית שלוקח כל רצף ורכף בקובץ שלנו ומעריב אותו, כך שנוצר לנו קובץ שלילי.

```

# generate negative data from positive data by shuffling each line
sub generate_neg_file {
    my $pos = shift;
    my $rand = int rand 100;
    open my $pos_file, "<", "$pos";
    open my $neg_file, ">", ".$rand.dat";

    while (<$pos_file>) {
        my @line = split //;
        print ($neg_file) join('', shuffle(shuffle(@line))), "\n";
    }
    close $neg_file;
    close $pos_file;
    return ".$rand.dat";
}

```

פעולה ייצרת הקובץ השלילי. הפעולה מקבלת את הקובץ החיווי, וקוראת אותו שורה בשורה. הפעולה מערבבת כל שורה וcoturbת אותה לקובץ דמוי. בסוף היא מחדירה את השם של הקובץ.

לאחר שסיימנו את שבעת השלבים קיבלנו תוכנה שצפינו שתיהיה מסוגלת למצוא מוטיבים בתחום רצפי דנ"א שימושיים לתוכה. עד כה הרצינו את הקוד על רצפים סינטטיים, שיצרנו בעצמנו ושהשתלנו בהם מוטיבים. כתע נותרה לנו רק ההרצה של הקוד על רצפים ביולוגיים.

פרק שלישי: תוצאות

ממצאים

בחרנו להזין שלושה סוגיים שונים של קלט לתוכנה שלנו, ובכל אחד מהם קיבלנו תוצאות פחות או יותר צפוי. הסוג הראשון של הקלט היה קלט סינטטי. בקלט הסינטטי ניסינו לבדוק אם הקוד שלנו באמת עובדות, והאם באמת ניתן ליצור מוטיבים לפיהן. לקחנו קבוצות של אלף רצפים ו"שתלנו" בתוכן מופיע של PSSM, המופיע היה שונה מקום למקום אבל הצורה הכללית של PSSM. מצאנו שניית באמת למצוא את המוטיב ששתלנו בעזרת התוכנה. תחילה, היה דבר תלוי במשתנים כלשהם, למשל, האורך של המוטיב היה חייב להיות 5, אבל ככל שייפרנו את התוכנה שלנו התלות במשתנים נעלמה.

הסוג השני של הקלט שבחרנו היה קלט שנלקח מניסוי *seq-PChI*. בסוג הקלט זהה ניסינו לראות אםמצא המוטיבים שלנו מסוגל למצוא מוטיב בודד למורთ שיש בדגימות שלנו ריש ביוווגי. לקחנו מספר של בין 400 ל-1500 דגימות לכל פקטורי שיעתק, והרצינו עליו את התוכנית. בחרנו בכוונה פקטורי שיעתק שהמוטיב שלהם ידוע, כדי שנוכל לבדוק את עצמינו. מצאנו שמצא המוטיבים שלנו יכול, בעזרת אלגוריתם הקלסטרינג למצוא את המוטיב. התוכנה משתמשת בקלסטרינג כדי לנוקת את המוטיב מריש (ארחיב על התופעה זו בהמשך). הוספנו עוד ועוד חלקים לתוכנה עד שהגענו לתוצאות מאוד דומות לתוצאות הצפויות.

הסוג השלישי והאחרון של הקלט הוא קלט של מדידות ביוטי גנים בתאים בריאים וسرطניים. לקחנו מדידות של ביוטי גנים בכמה סוגים של תאים. לכל סוג תא לקחנו מדידות מתא בריא ומטא סרטני מאותו רקמה. המדידות נעשו בעזרת מערך דנ"א. עברו כל סוג תא הסתכמנו על ביוטי הגנים בתא הבריא ובתא הסרטני, ולאחר מכן את רצפי ה-3' UTR של גנים שהביוטי שלהם שונה באופן משמעותי בין התא הבריא לסרטני, ולאחר מכן את רצפי ה-2 או יותר, או חוסר של פי 2 או פחות). חישבנו האם מופיע ברצפי ה-3' UTR שאספנו אתר קישור של מיקרו רנ"א כלשהו, אם כן, יש סיכוי שהוא מיקרו רנ"א שנקשר לאთר הקישור הזה משפייע על התפתחות הסרטן בסוג התא זהה. לקחנו תאים של סרטן שלפוחית השתן, סרטן השד, סרטן המעי הגס, סרטן הכליה, סרטן הריאה סרטן הערמוני וسرطان הרחם, והרצינו עליהם את הבדיקה. מצאנו שברצפים שאספנו מسرطן השד, סרטן מעי הגס, וسرطان הערמוני ישן אתרי קישור לכמה מיקרו רנ"א שונים. בשאר הבדיקות לא מצאנו אתרי קישור לאף מיקרו רנ"א.

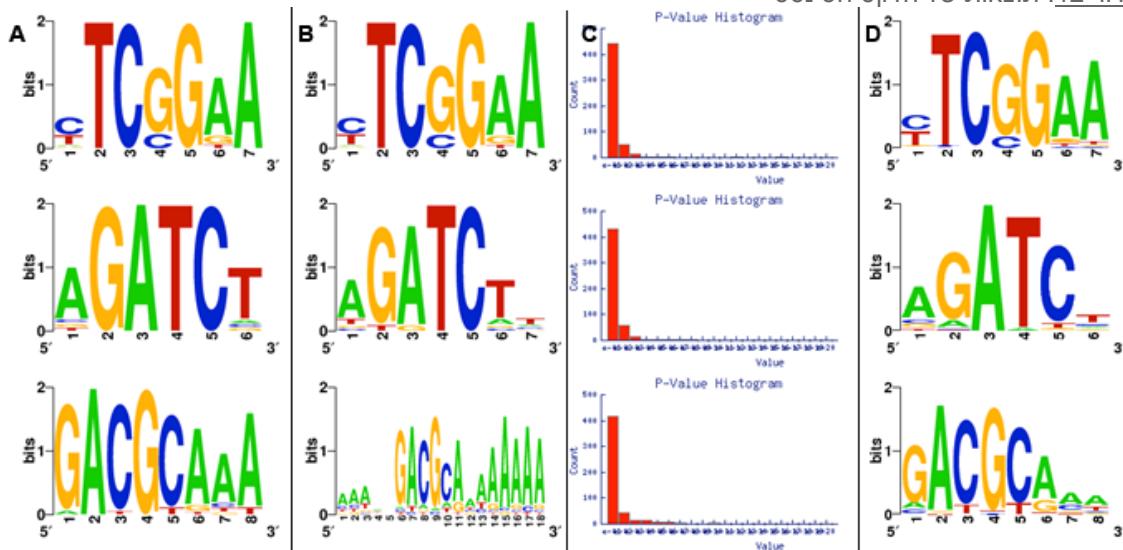
תוצאות

קלט סינטטי:

כפי שציינו בחלק הקודם, הרצינו את התוכנה שלנו על שלושה סוגים שונים של קלט. הסוג הראשון הוא הקלט הסינטטי. הקלט הסינטטי הוא אוסף של רצפי אקראיים לחולוטין באורך של אלף בסיסים. בסוף כל אחד מהרצפים האלה השתלנו מוטיב קצר. הקלט הסינטטי מאפשר לנו לבדוק אם התיאוריה שלנו אכן עובדת, ולראות האם ניתן לשחזר את המוטיב שהשתלנו.

הסיבה שבגלה השתמשנו בקלט סינטטי ולא ישר קפכנו לעבוד על דנ"א אמיתי היא רعش ביולוגי. בדנ"א אמיתי יש רעש ביולוגי מאד גדול. מכיוון המידע בדנ"א הוא לא אקראי, יכול להופיע מוטיב חוזר בכל הר齊פים שהוא חסר משמעות, או פשוט לא שייך למוטיב שאחנו מחפשים.

איור 12: תוצאות של הלקט הסינטטי



באир אנחנו רואים חלק מהתוצאות של הקלט הסינטטי. בעמודה A נמצאת התוצאה של `4align`, בעמודה B התוצאה של `4align`, בעמודה C ההיסטוגרמה, ובעמודה D המוטיב האמיתי (השאיפה שלנו).

לפי התוצאות באיר 12 נראה כי התוכנה עצמה עובדת. ישנו מקומות בהם אלגוריתם אחד עובד והשני לא (כמו בשורה השלישית - `4align` יוצר מוטיב די מדויק ואילו `ChIP-seq` מובילן), ומקומות בהם אחד מהאלגוריתמים יותר מדויק מהשני (כמו בשורה השנייה למשל, `4align` יוצר מוטיב יותר מדויק), אך ניתן לראות שהמוטיב זווה כפוי. בעזרתה התוצאות הללו הצלחנו ליצור בסיס לתוכנה שלנו, שעליו הוספנו עוד ועוד חלקי כדי שתוכל להתמודד עם השלבים הבאים. את שאר התוצאות ניתן למצוא ב��פה 1.

קלט ChIP-seq:

סוג קלט שני שהוא לתוכנה הוא קלט שהתקבל מ-`ChIP-seq`. בקלט זהה נמצא רק מוטיב אחד כמו בסוג הקלט הקודם, אך מכיוון שהקלט הזה נלקח מדנ"א של תא אמיתי, יש בתוכו הרבה רעש

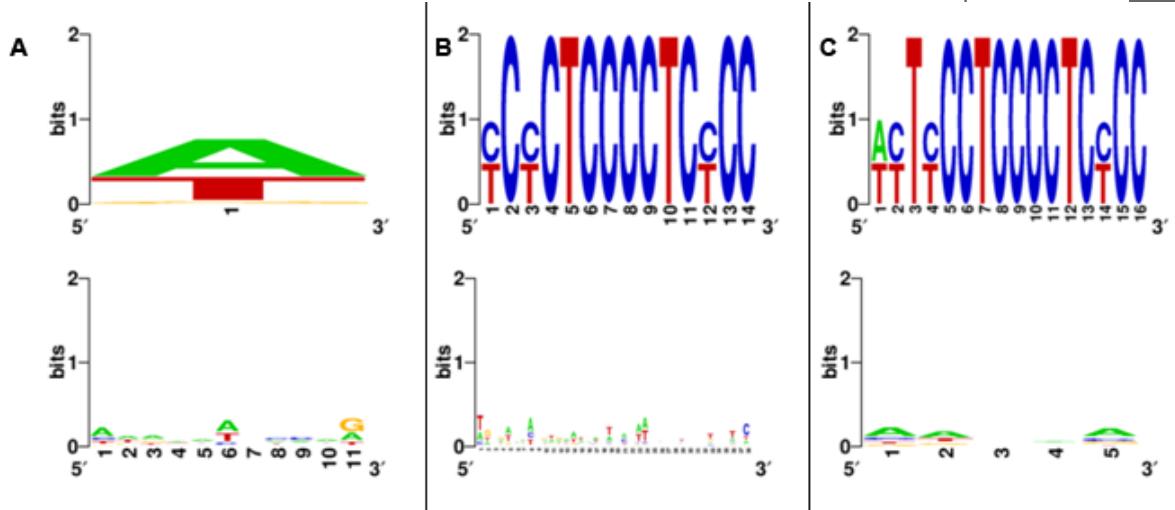
ביולוגי. בשלב זה בחרנו בפקטורי שייעתוק שכבר מצאו את המוטיבים שלהם כדי שנוכל לבדוק האם התוצאות שלנו נכונות. את המוטיבים הידועים לקחנו מהאתר של ד"ר ארן סגל:
<http://genie.weizmann.ac.il/pubs/fmm08/index.html>

בחרנו להשתמש בקלט ChIP-seq כדי לבחון האם התוכנה שלנו מסוגלת להתמודד עם רעש ביולוגי ועם מוטיב אחד. לו היינו בוחרים להשתמש ברכפי דב"א עם כמה מוטיבים ותוכנה לא הייתה מצליחה להתמודד איתם, לא יכולנו לדעת אם הבעיה קשורה ברעש, או במספר הרב של המוטיבים.

תוצאות ללא סף:

לפני שכתבנו את הקטע שמחפש את סף רלוונטיות הקיימרים, הרצינו את התוכנה 3 פעמים, פעם אחת לקחנו את 20 הקיימרים עם השעון-value-k הכי גבוה, בפעם השנייה לקחנו 30 קיימרים, ובפעם השלישית לקחנו 40 קיימרים.

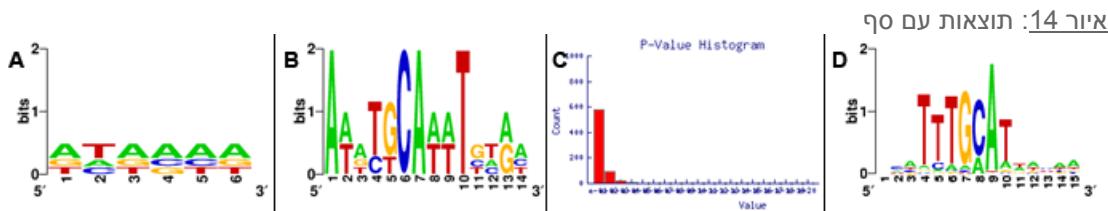
איור 13: תוצאות ללא סף



באир זה אפשר לראות את התוצאות של `4align` ו-`pivalign` עבור פקטור השיעותוק שננקה *SOX2_Boyer* עם הבחירה של 20, 30, ו-40 קיימרים היכי רלוונטיים. בשורה העליונה נמצאות התוצאות של `4align` ובשורה התחתונה נמצאות התוצאות של `pivalign`. בעמודה A נמצאות התוצאות לאחר בחירה של 20 קיימרים, בעמודה B לאחר בחירה של 30, ובC לאחר 40.

קל מאד לראות שהתוצאות האילו לא טובות. `4align` יצר מוטיבים עם המון C ו-D, ובמוטיב ראשוני התקבל משחו באורך של בסיס (ברור שמוטיב לא יכול להיות באורך של בסיס אחד). `4align` יצר מוטיבים מאוד אקריאים (האותיות נמצאות מאוד) וארוכים. בכלל התוצאות הללו הוספנו את הקטע שמחפש סף רלוונטיות.

תוצאות עם סף:

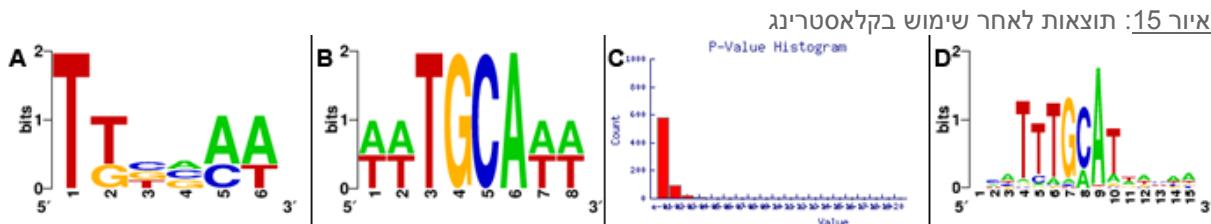


תוצאות האלגוריתמים לאחר שימוש במצב סף עבור פקטור שייטוק בשם *OCT4_Boyer*. העמודות: *A* תוצאות של 4align B, *B* 4align A, *C* היסטוגרמם P-value, *D* סגל ("האמתית").

התוצאות במצב זהה הן כבר יותר טובות. 4align הצליח ל特派 חלק קטן מהмотיב, TGCA, אך ככל זאת יש מקום לשיפור. המוטיב של 4align מוביל וחסר בו חלק עם שלושת ה-*T*, והמוטיב של 4align לא מזכיר את המוטיב האמתי בכלל.

שימוש בקלוסטרינג כדי לנוקות רעש:

לאחר מאמצים רבים לנסوت להבדיל בין קיימרים שיעיכם למוטיב וקיימרים שהם סתם רעש חסר משמעות, גילינו שהוא מאוד מעניין. אם נפעיל על הקלט שלנו את הקלוסטרינג, נראה שאנו מקבלים כמה מוטיבים, כאשר אחד מהם הוא המוטיב האמתי והשאר שיעיכם לרעש (את הסינון עשינו בעין). המוטיב שמתקיים הוא הרבה יותר מדויק מהמוטיב שקיבלנוearlier (הקדם).



תוצאות עבור אותו פקטור שייטוק, *OCT4_Boyer*, הפעם עם קלוסטרינג. העמודות הן כמו באירור הקודם.

הפעם התוצאות של 4align נראות טובות יותר, אבל עדין לא משקפות את המוטיב האמתי. התוצאות של 4align נקיות יותר ושוב האלגוריתם הצליח למצוא חלק מהמוטיב, הפעם, הפעם. בחלק הזה של הניסוי לא ניסינו לקבל תוצאות מדויקות בהשוואה לאילו של ד"ר סגל. התוצאות של סגל הן גם תוצאות של תוכנה למציאת מוטיבים ולכן הן לא בהכרח מדויקות בעצמן. הרעיון בניסוי ChIP-seq היה לקבל תוצאות דומות לתוצאות קיימות (במקרה הזה התוצאות של ד"ר סגל).

את שאר תוצאות ChIP-seq ניתן למצוא בנוסףštayim.

קלט בטוי גנים סרטניים ובריאים:

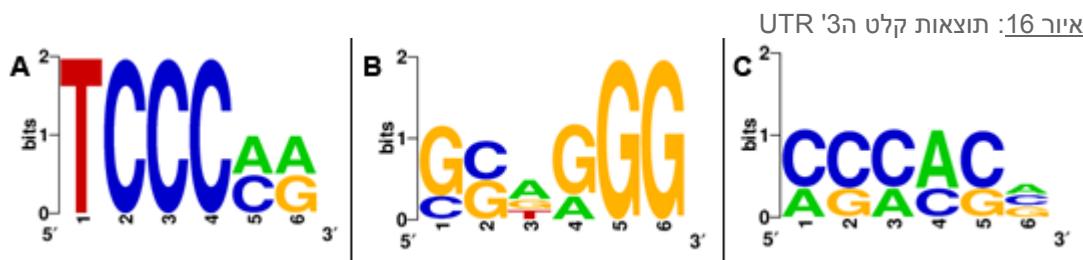
הסוג השלישי והאחרון של הקלט הוא קלט של מדידות בטוי גנים בתאים בריאים וסרטניים. לקחנו מדידות של בטוי גנים בכמה סוגים של תאים. לכל סוג תא לקחנו את רצף ה3' UTR של גנים ושברו בטוי בלתי רגיל בתא סרטני לאומת התא הבריא והזנו אותם קלט לתוכנה. בחרנו לעבוד

עם תאים של סרטן שלפוחית השטן, סרטן השד, סרטן המעי הגס, סרטן הכליה, סרטן הרירא סרטן הערמוני וסרטן הרחם.

```
ron $ . ./gen_gene_table.sh
      BLDR    BRST    COLON   KID     LUNG    PAN     PROST   UT
found: 1237    1115    612     1006    530     818     141     1334
```

בדוגמה לעיל ניתן לראות טבלה שהדפיסה את התוכנות שכתבותי. הטבלה מסכמת את תוצאות חיפוש הגנים. בשורה הראשונה רשומים סוג' התאים, ובשורה השנייה רשומה כמות הגנים שנמצאו שעברו ביטוי בלב רגיל (למשל, ניתן לראות שבסרטן שלפוחית השטן (BLDR) נמצאו 1237 גנים).

הזנו את רצפי ה3' UTR לתוכנה וקיבלו כמה מוטיבים לכל תא ותא, מוטיב אחד שהוא אתר קישור לרנ"א מסוים, ועוד מוטיבים של רעש. ניתן להבדיל בין רצפי רעש וrzcfps אמייתים Ci רצפים אמייתים מתאימים לנר"א אחד או שניים, ואילו רצפי רעש מתאימים להרבהRN"אים או לא מתאימים לאףRN"א. היו לנו גם כמה תאים שלא נמצא בהם אף מוטיבים שהם רצפים של אתרי קישור.



באיור 16 רואים את התוצאות הסופיות של התהילה. בעמודה A המוטיב לסרטן השד. נמצא שהוא מתאים למיקרוRN"א 5p-miR-423a/423b. בעמודה B המוטיב לסרטן מעי הגס. נמצא שהוא מתאים למיקרוRN"א miR-296-5p-miR-744i/miR-633RN"אים 551ab. המוטיב לסרטן הערמוני. נמצא שהוא מתאים למיקרוRN"אים miR-423bRN"א.

באיור 16 אנו רואים את תוצאות קלט המיקרוRN"א. התוצאות הללו מלמדות שבסוג סרטן מסוים, ישנו מאזור לא תקין של מיקרוRN"א מסוים, וייתכן שהמאזור הזה הוא אחד הגורמים הסרטן. למשל, מצאנו שבסרטן השד ישנו מאזור שלילי חזק של מיקרוRN"א מסווג 423-miR. חיפשנו מאמרים שימושיים את הממצאים שלנו, ומיצאנו שני מאמרים מהשנתים האחרונות שמראים שיש קשר בין מחסור miR-423b לבין חולות (חולות?) בסרטן השד. לא ידענו שאילו יהיו התוצאות, אפילו לא ידענו אם יש באמת קשר בין מיקרוRN"א וسرطان, אך התוצאות הללו והמאמרים מראים שיש. את המאמרים ניתן למצוא בקישורים הבאים, ובמקומות 14 ו-15 בביבליוגרפיה (נספח 4).

<http://www.ncbi.nlm.nih.gov/pubmed/22593246>

<http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3129492/>

התוצאות של שני סוג' הסרטן האחרים קצר מוזרót. קיבלו מוטיב אחד שמתאים לשני מיקרוRN"אים שונים. לדעתי הסיבה לכך היא בגלל שהמוטיב הזה בעצם משלב בין שני רצפים שונים. לדוגמה, נראה שלמוטיב שהתקבל בסרטן הערמוני ישן שתי "קומות", הקומה העליונה היא CCCAC, והתחתונה היא AGACG. אחת מהקומות הללו נוצרה כתוצאה מהרעש שבקלט שגרם לאלגוריתם הקלאסטרינג שלנו לא להבדיל בין שני המוטיבים. לא מצאנו מחקרים שמאשימים את התוצאות לגבי

שני סוגים של סרטן הללו, אך מכיוון שצדקנו בתוצאות שלנו לגבי סרטן השד, יתכן שגם בסרטנים האחרים יש קשר בין המיקרו רנ"אים שמצאנו להתקפות הסרטן.

פרק רביעי: דיוון

סיכום

בעובדה זו ניסינו ליצור כל חישובי שבuzzרטו ניתן למצאו מוטיבים בדנ"א. הרעיון שלנו התבסס על ספירה של מיללים. התחלנו בספירה של כל המיללים באורך 6 בשתי קבוצות רצפים שאויתן ה зан לתוכנה. לאחר מכן חישבנו את התפלגות ההיפרגיאומטרית של כל אחת מהמיללים, וסידרנו את המיללים בהיסטוגרמם. בעזרה ההיסטוגרמם חישבנו ערך סף ובחרנו את כל המיללים שהערך שלהם בתחום. לאחר מכן חישבנו את התפלגות ההיפרגיאומטרית, ערך הק, היה פחות מהסף. לאחר שאספנו את קיימרים ששיערנו שישיכים למוטיב, נותר לנו לבנות מהם את המוטיב עצמו. בשלב זה צפינו בעיה: יתכן שקיים יותר ממוטיב אחד באוסף הקיימרים הללו, ולכן צריך לסדר אותם בכמה קבוצות, או קלאלסטרים. סידרנו את המיללים בטבלה על פי תת רצפים באורך 4 שימושיים בתוכן, וכך חילקו אותן לקבוצות (אלגוריתם שמחולק אותן לקבוצות קראנו 4asus). לאחר שהקיימרים חולקו לקבוצות, התחלנו לבנות מכל קבוצה מוטיב. בכל קבוצה היה לנו בהתחלה אוסף של קיימרים באורך 6, אך יתכן שהמוטיב אותו הם מרכיבים הוא באורך שונה מ-6 (אולו 8, ואולי אפילו 15), לכן בנינו שני אלגוריתמים שמאrics את הקיימרים, ובאותו זמן גם מסדרים אותם האחד מעל השני. האלגוריתם הראשון שבנוינו עבד על פי תת רצפים בתוך הקיימרים ונקרא alg4, ואילו האלגוריתם השני עבד לפי ציר של אותיות ברצפים, וקראנו לו algival. מצאנוalg4 שalgival טוב יותר במצב מוטיבים קצרים, ואילו algival טוב יותר במצב מוטיבים ארוכים. החלק האחרון במבנה המוטיב היה הניקון. לאחר שבנוינו את המוטיב, נותר לנו לנוקות אותו מרעש שנוצר על ידי ההארכה. כדי לעשות זאת השתמשנו בפונקציה שנקרהת סטיית תקן. בסוף יצרנו לוגו ויזואלי כדי לייצג את המוטיב שלנו.

הזמן לתוכנית שלנו שלושה סוגים שונים של קלט ובחנו את התוצאות. הסוג הראשון של הקלט שהזנו לתוכנית היה קלט סינטטי. הבדיקה היחיד של הקלט היה לראות שככל החלקים בתוכנה עובדים כצפי, והרעיון שלנו פועל באופן מעשי, ולא רק באופן תיאורטי. אחר כך בחרנו קלט שונה לחלווטין, קלט שנוצר על ידי ChIP-seq. בקלט זה ישנו רק מוטיב אחד אבל המוטיב הזה מלאה בהמון רעש ביולוג. רצינו לראות האם התוכנה שלנו מסוגלת להתמודד עם רעש שכזה. השווינו את התוצאות שלנו עם תוצאות של אנשים אחרים (שמצאו את המוטיבים בדרכים שונות), וראינו שההתוצאות אכן דומות. השתמשנו בקלסטרינג כדי לנוקות את הרעש מההתוצאות שלנו, וקיבלו תוצאות די מדיוקנות. סוג הקלט השלישי גם היה שונה מהקלטים הקודמים, זהו קלט מדדיות של ביוטי גנים בתאים שונים. קלט זה הכיל בתוכו ביוטי גנים באים מסוימים, ובעזרתו בנינו קלט לתוכנה שמקיל בתוכו אתרים קישור למיקרו RNA. בעזרה קלט זה מצאנו שבסוגי סרטן שונים יש מחסום במיקרו RNA מסוים, וייתכן שהחוסר הזה הוא אחד הגורמים לסרטן.

יתרונות על פני יישומים אחרים

מצוא המוטיבים שלנו הוא אחד מטור מספר תוכנות אחרות. אם כך, מה מייחד את מצוא המוטיבים שלנו מאשר היישומים של הרעיון? ובכן, סיבוכיות הזמן של מצוא המוטיבים שלנו נמוכה מאוד, ככלומר, זמן הריצה של התוכנה נמוך מאוד. זמן הריצה עם הקלט הסינטטי היה קצר יותר מדקה, וזמן

היריצה עם קלט fqIP-seqP היה ב ממוצע 7 דקות, כאשר דקה מהזמן זהה מוקדשת להתפלגות היפרגיאומטרית. היריצה נעשתה על מחשב בן 10 שנים (4 Pentium חד-לבתי). זמן היריצה של הרבה ממצאים המוטיבים האחרים יכול להגיע אפילו לחצי שעה. יתרון נוסף של התוכנה שלנו הוא שהיא מסוגלת לבצע את החישובים שלה על פני כמות גדולה יחסית של רצפים. עוד יתרון של המוצא שלנו לעומת מוצאים אחרים הוא הקלאלוסטרינג. בזכות האלגוריתם הזה התוכנה שלנו מסוגלת לזהות כמה וכמה מוטיבים בו-זמןית. חלק מהתוכנות האחרות אין מסוגלות לכך.

שימושים

עתה שהצלחנו למצוא מוטיב בתוך דנ"א של תא, אנחנו יודעים לאילו רצפים נקשר פקטור שייעתקם, ואנחנו יכולים לשער רצפים מסוימים למקור רנ"א. אבל מה עכשו? מה ניתן לעשות עם התוכנה שלנו מעבר להגיד שאנו יכולים למצוא מוטיבים בדנ"א? מסתבר שיש לתוכנה שימושים בעולם המדע. דוגמא אחת לכך היא הנדסה גנטית. אם אנחנו רצים למשל, לצורך מחקר ופיתוח של תרופה, וכו' לגרום לתא לשנות את התנהוגות שלו (מדענים עושים זאת לעיתים תכופות), אפשר להחדיר לתא חומרים כימיים שונים ולקווות שאוותם חומרים יגרמו לשינוי שאנו רוצים. לעומת זאת, אפשר לבטל גן שנמצא בתוך התא וshaworm לתנהוגות שאוותה אנחנו רוצים. את ביטול הגן אפשר להשיג באמצעות מציאת מוטיבים. אם נמצא שרץ' כלשהו הוא זה שמקשר בין פקטור שייעתקם לגן שלנו, ובסבבו של דבר גורם להפעלה של הגן, אפשר יהי ליצור חומר שנקשר לרצף זהה וחוסם את הקישור עם פקטור השיעתק. או ליצור חומר שמחסל את הפקטור. בדרך זו נצליח לבטל את הגן. דוגמא שנייה הצורך בתיקון של מוטיציות שנגארמות לעיתים לגנים בתאים. כתוצאה מהמוטיצה לגן כלשהו מהגנים שבתא ישנה השפעה שלילית על שאר המרכיבים בתא. אם נדע מה רץ' הקדם של אותו גן פגום, נוכל באופן עקרוני לפתח תרופה שחווסמת את הרץ' זהה ומונעת שייעתק. כתוצאה גם חסימה של הפעלה השלילית של הגן הפגום. דוגמא שלישית, עשויה שימוש במיקרו רנ"א. ברגע שגילינו שבתא סרטי מסויים יש מחסור או עודף של מיקרו רנ"א כלשהו, אנחנו יכולים לפתח תרופה שתצאן את כמות המיקרו רנ"א בתא זהה ובכך יכול להיות שהוא יופיע לתא רגיל ובריא.

תוכניות לעתיד

התוכנה שפיתחנו לא עובדת באופן מדויק מספיק. כפי שראינו בתוצאות עם fqIP-seqP, אפשר להגיעה לדיקק הרבה יותר גבוהה. הבעיה בתוכנה שלנו היא "שהרחלוצה" של התוצאות שלנו נמוכה מדי. כמו, התוכנה מסוגלת למצוא את המוטיב הכללי אבל עדין לא מדיקת פרטים קטנים (למשל, אנחנו יכולים לראות שהמוטיב ATAGCT מופיע במוטיב, אבל אם יש סיכוי של 10 אוחז שבמקום ה-A השני יופיע T, לא נדע את זה). יש מקום לשיפור בקוד, ובאמצעות עבודה נוספת אנחנו סבורים שניתן יהיה להביא את האלגוריתם לדיקק גבוהה יותר. דבר נוסף שניתן להוסיף הוא התייחסות למקדים פרטיים. קיבלנו ללא מושג תוצאות שבהן יש במוטיב אחד תת-מוטיב חזק מצד שמאל, ותת-מוטיב חזק מצד ימין, ורצף אקראי של אותיות ביניהם. לעיתים גם מופיעה האות T

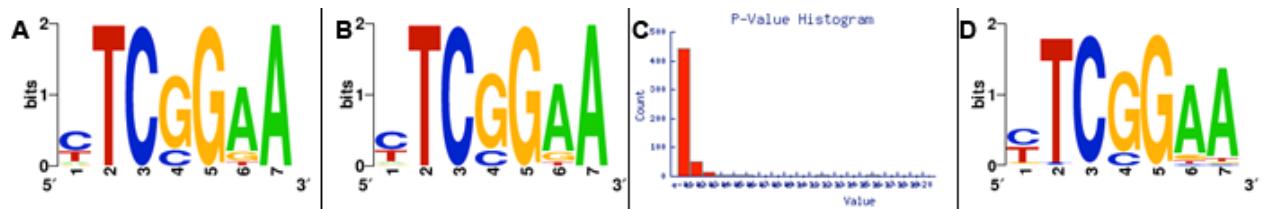
במקומות לא רצויים במושגך. אם ניקח בחשבון מקרים פרטיים כדוגמת אלה ונמצא דרך לטפל בהם,
אני בטוח שנגיע לתוצאות אפילו יותר טובות.

פרק חמישי: נספחים

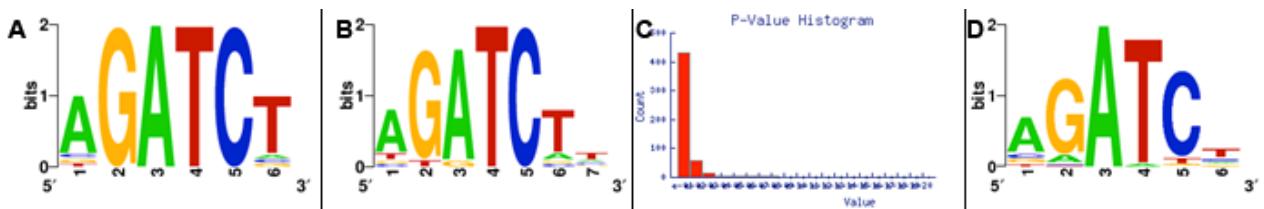
נספח 1: תוצאות הקלט הסינטטי

לכל קטע מהקלט הסינטטי נתנו שם, באירורים למטה מוצגים התוצאות עבור כל קטע. משמעות העמודות: A התוצאה עבור chgval, B התוצאה עבור chgval0, C ההיסטוגרמה, DI ה-PSSM אוטו שתלנו בקלט ("השאיפה" שלנו).

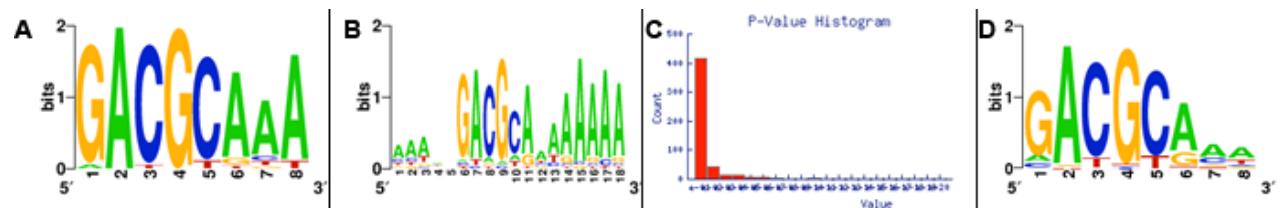
CEP3:



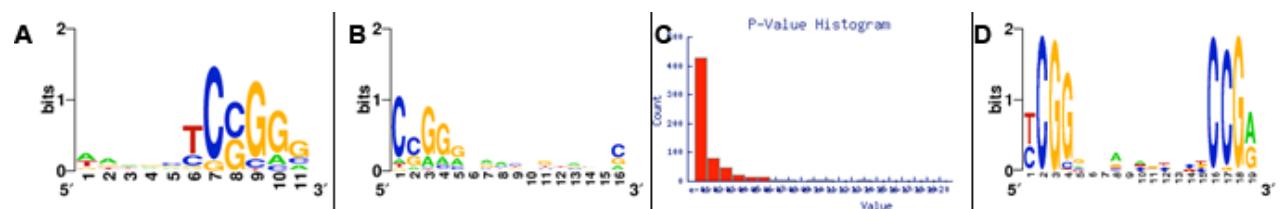
ECM23:



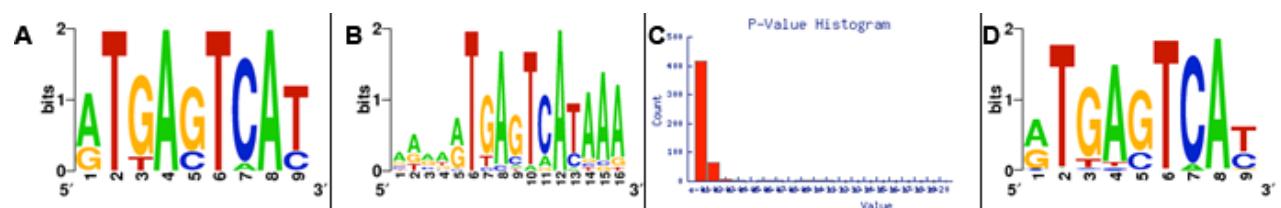
FHL1:



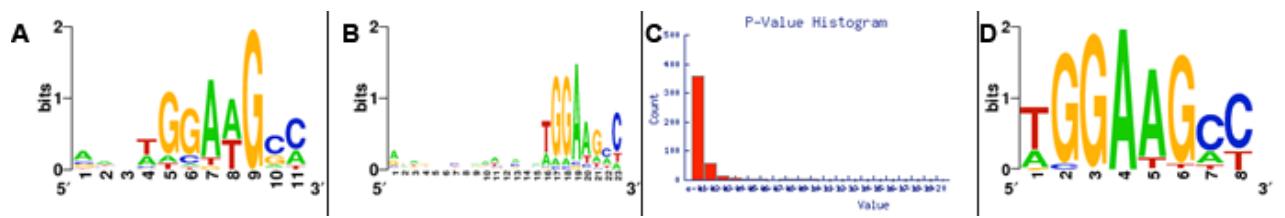
GAL4:



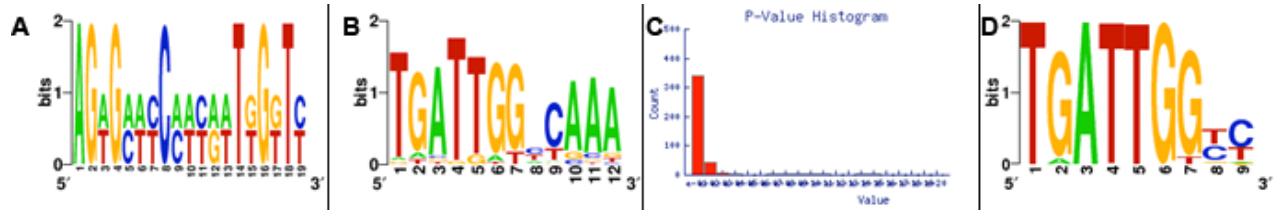
GCN4:



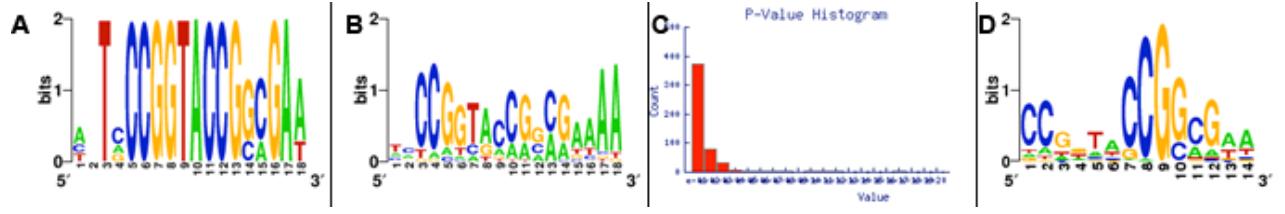
GCR1:



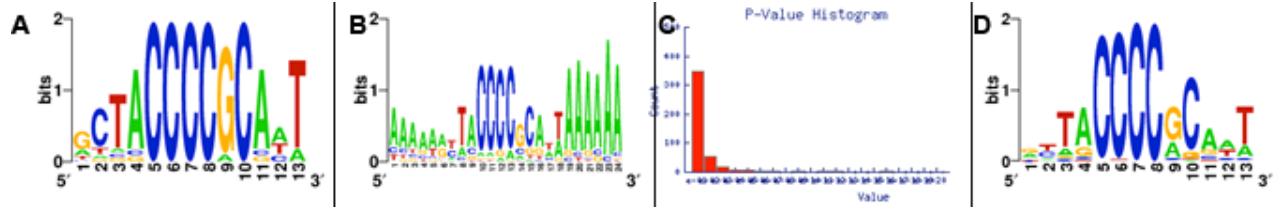
HAP4:



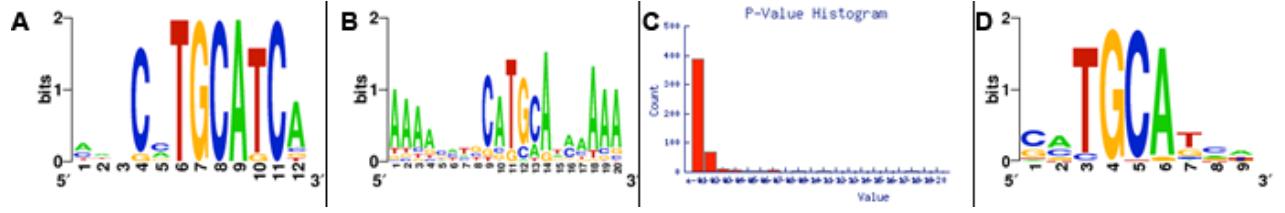
LEU3:



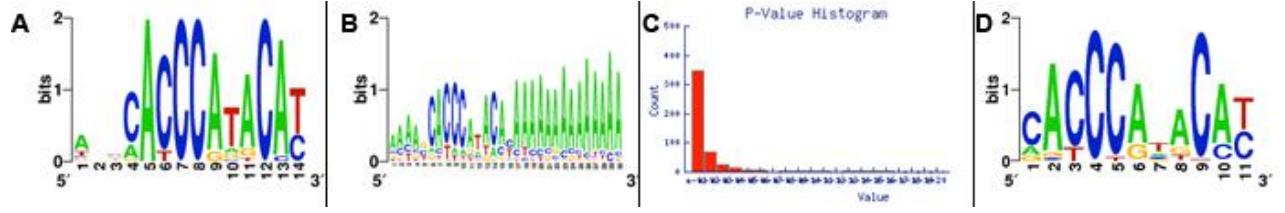
MIG2:



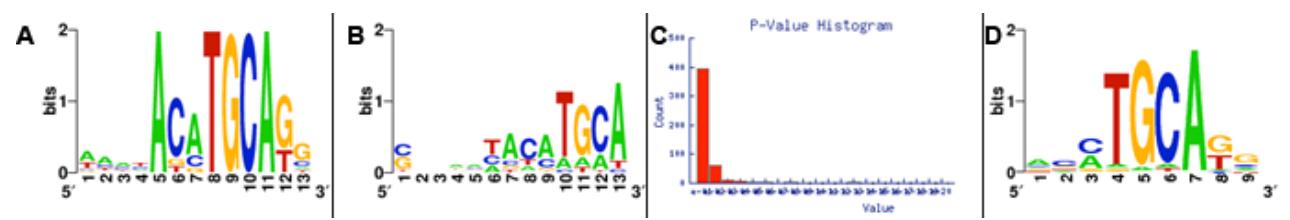
PHD1:



RAP1:



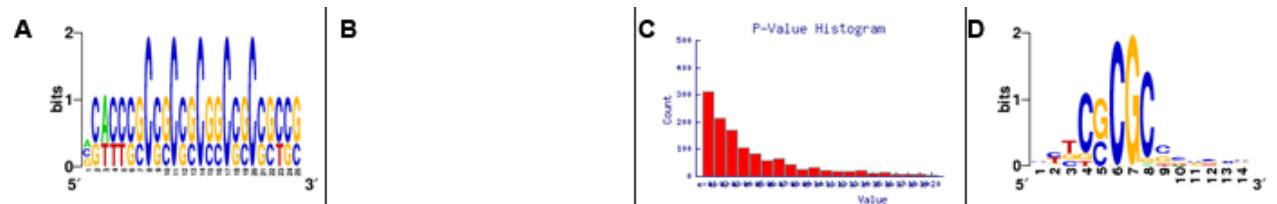
SOK2:



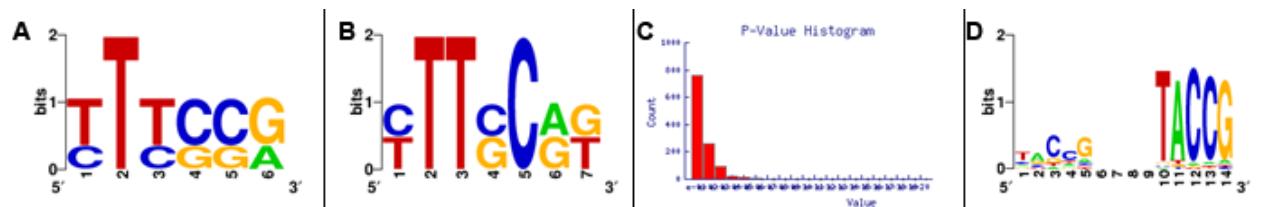
נספח 2: תוצאות קלט ChIP-seq

לקחנו רצפים של פקטורי שייעתוק שונים שהועגו על ידי ניסוי ChIP-seq והזנו אותם לתוכנה שלנו. באירועים להלן ניתן למצוא תוצאות עבור כל פקטור שייעתוק. משמעות העמודות: A התוצאה עבור B 4align, התוצאה עבור align, C ההיסטוגרמה, וD התוצאה של ד"ר סגל ("השאיפה" שלנו).

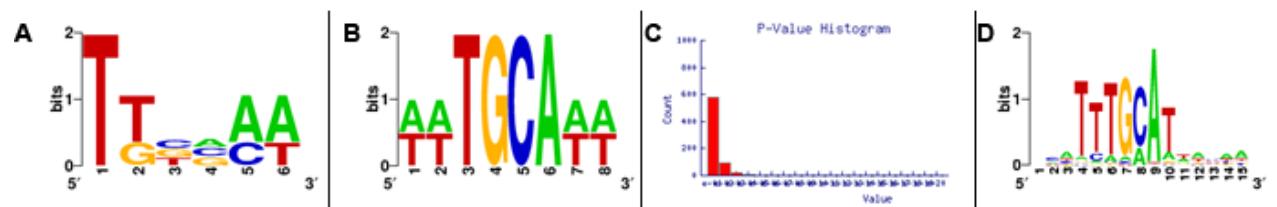
E2F4 Boyer:



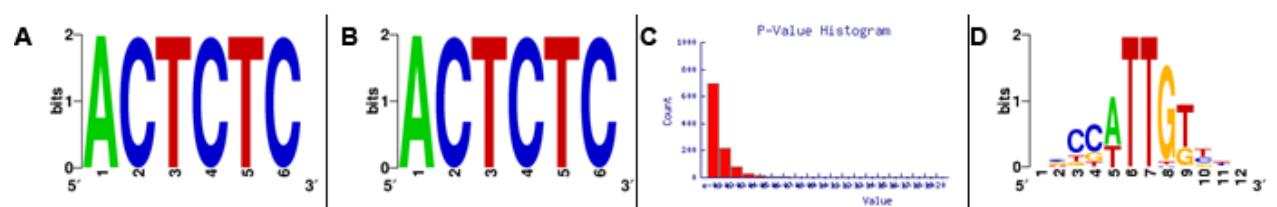
NANOG Boyer:



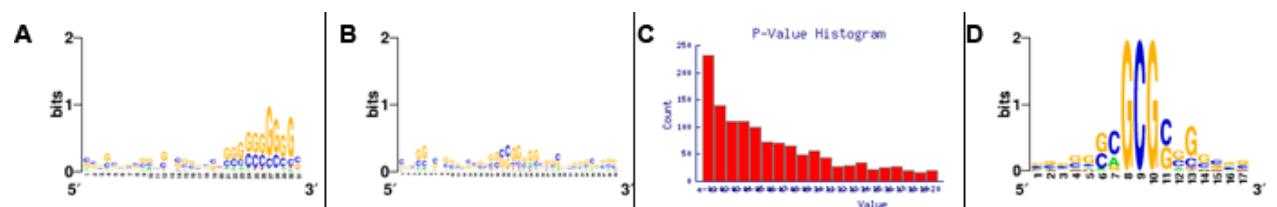
OCT4 Boyer:



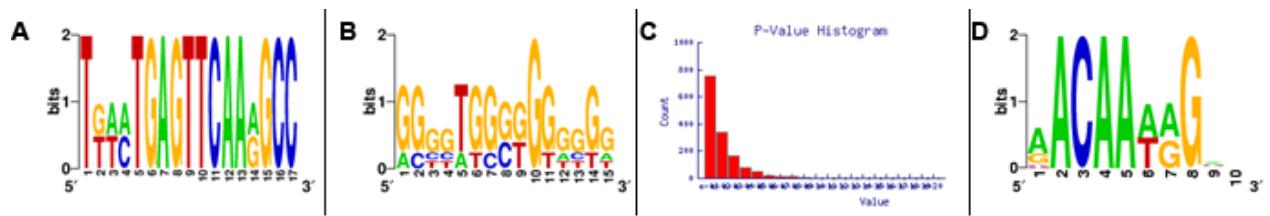
SOX2 Boyer:



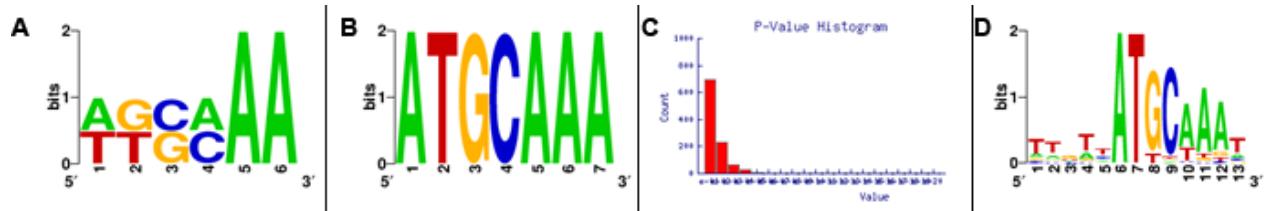
PRC2 SUZ12:



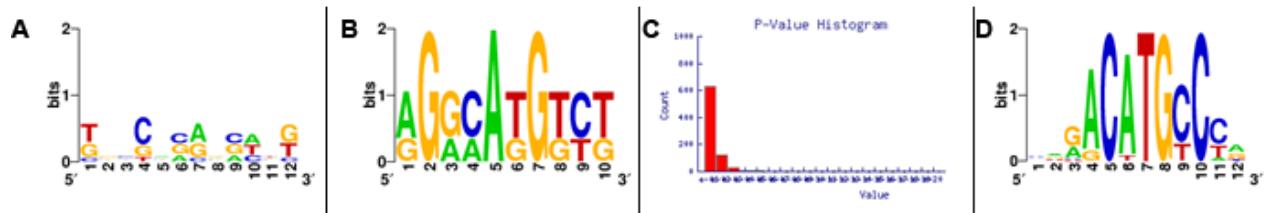
NANOG Loh:



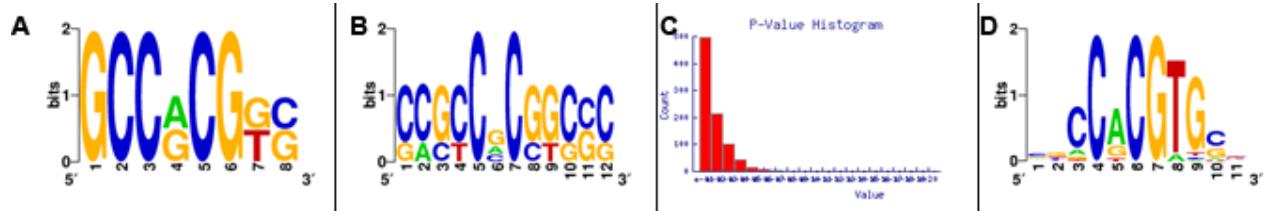
OCT4 Loh:



P53 PET3:



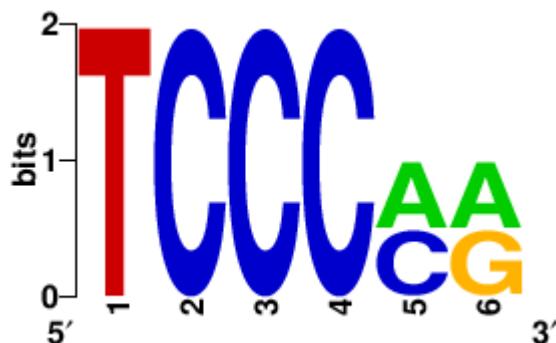
c-Myc PET3:



נספח 3: תוצאות קלט המיקרו RNA

אפשר לראות באיזור את אתר הקישור של מיקרו RNA מסוים לגן עבור כל סרטן.

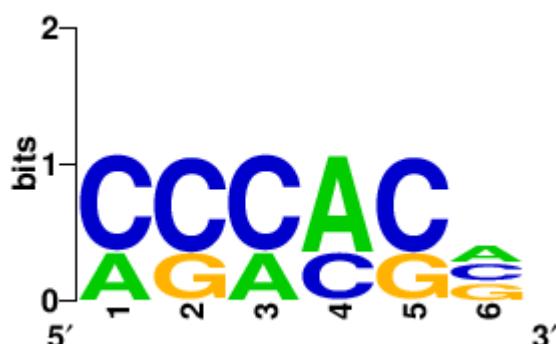
Breast: (miR-423)



Colon: (miR-296, miR-551)



Prostate: (miR-663, miR-744)



נספח 4:ביבליוגרפיה

1. A survey of DNA motif finding algorithms,
<http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2099490/>
2. Fitting a mixture model by expectation maximization to discover motifs in biopolymers, <http://www.sdsc.edu/~tbailey/papers/ismb94.pdf>
3. Phylogenetic footprinting, http://en.wikipedia.org/wiki/Phylogenetic_footprinting
4. DNA, <http://en.wikipedia.org/wiki/DNA>
5. Transcription, [http://en.wikipedia.org/wiki/Transcription_\(genetics\)](http://en.wikipedia.org/wiki/Transcription_(genetics))
6. Transcription factor, http://en.wikipedia.org/wiki/Transcription_factor
7. MircoRNA, <http://en.wikipedia.org/wiki/MicroRNA>
8. Cancer, <http://en.wikipedia.org/wiki/Cancer>
9. Linux, <http://en.wikipedia.org/wiki/Linux>
10. Perl, <http://en.wikipedia.org/wiki/Perl>
11. Emacs, <http://en.wikipedia.org/wiki/Emacs>
12. ChIP-sequencing, <http://en.wikipedia.org/wiki/ChIP-sequencing>
13. DNA microarray, http://en.wikipedia.org/wiki/DNA_microarray
14. A genetic variant located in miR-423 is associated with reduced breast cancer risk, <http://www.ncbi.nlm.nih.gov/pubmed/22593246>
15. MicroRNA sequence and expression analysis in breast tumors by deep sequencing, <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3129492/>

נספח 5: התוכנה שכתבתי

צירפתי לעובדה CD ובו קבצי התוכנה שכתבתי. בתחום הCD צורבים קבצי התוכנה, .om.kalgן, .pmi.motif, .pml.mirnai, .lib.data, .pml.mirnai. תיקיות data מכילה את קבצי הקלט הסינטטי, תיקיות lib מכילה קבצים שדרושים כדי שהתוכנה תעבור, ותיקיות mirnai מכילה את הקוד לעיבוד המידע הקשור למיקרו רנ"א.