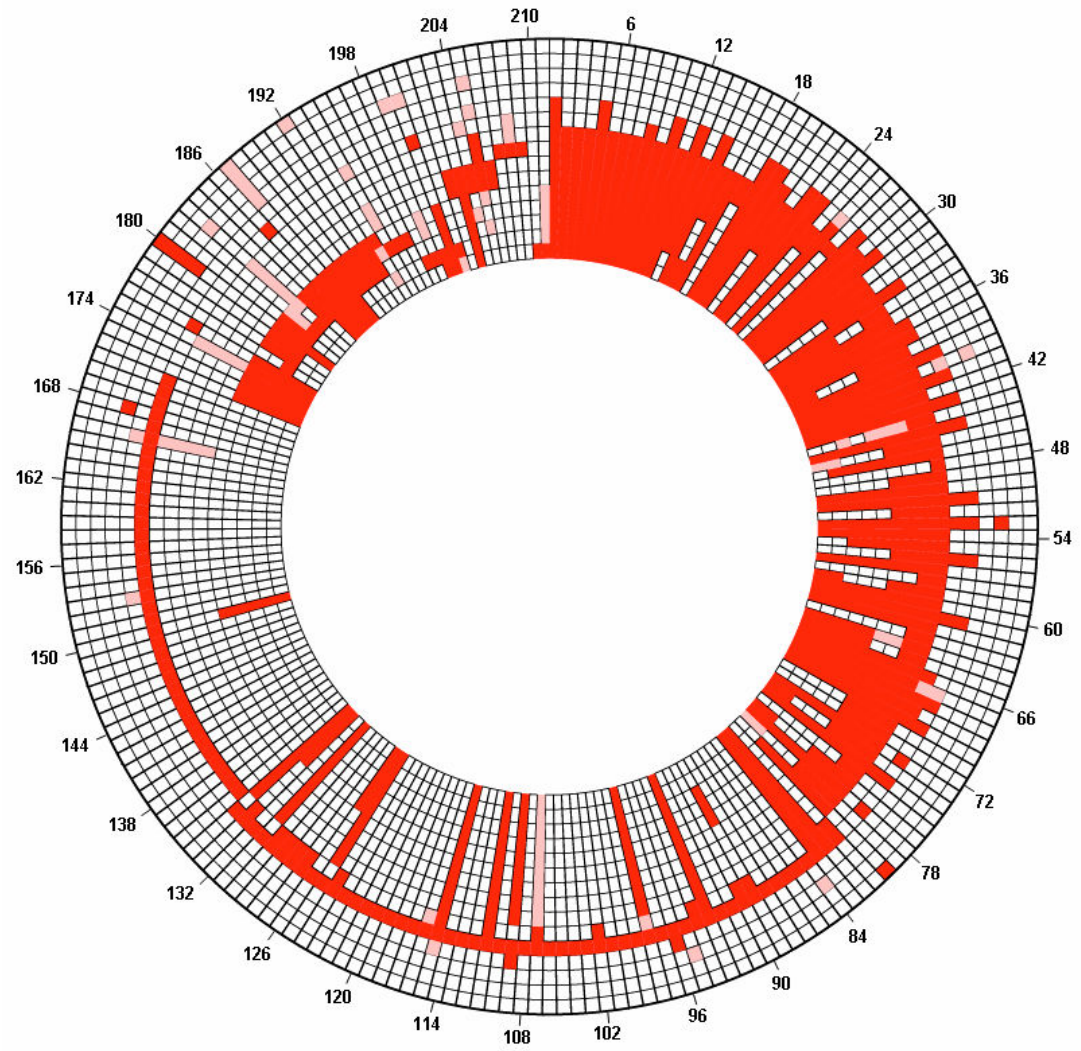
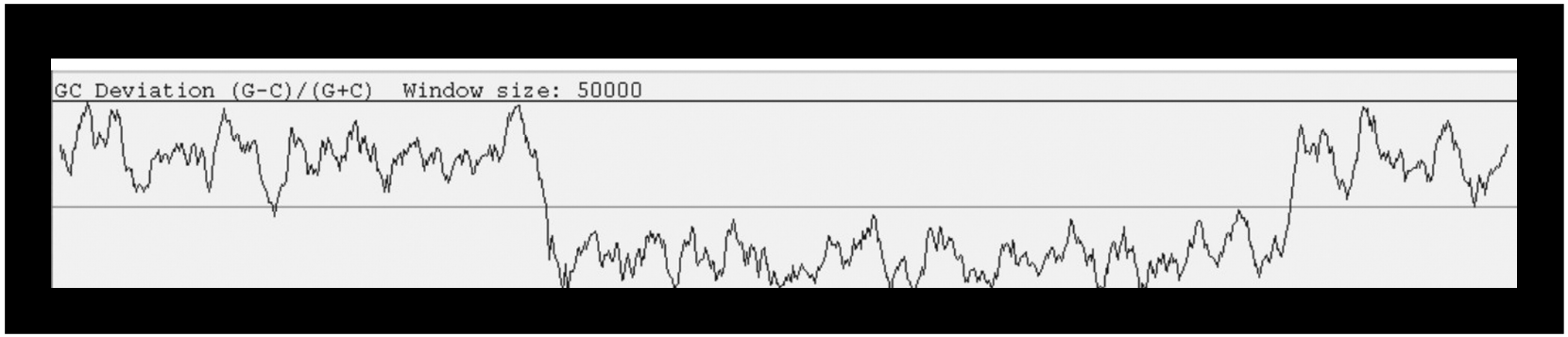
Γονιδιωματική Ρευστότητα σε Μικροβιακούς Οργανισμούς

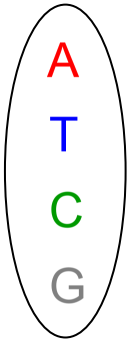
(Επαναπροσδιορίζοντας τον ορισμό του βιολογικού είδους)



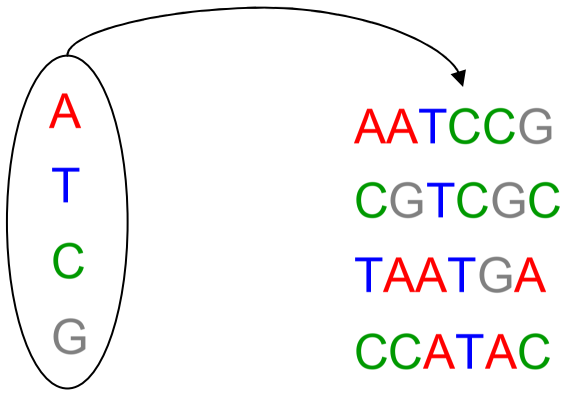
Νουκλεοτιδική Σύσταση DNA…
Τι μπορεί να μας πει?



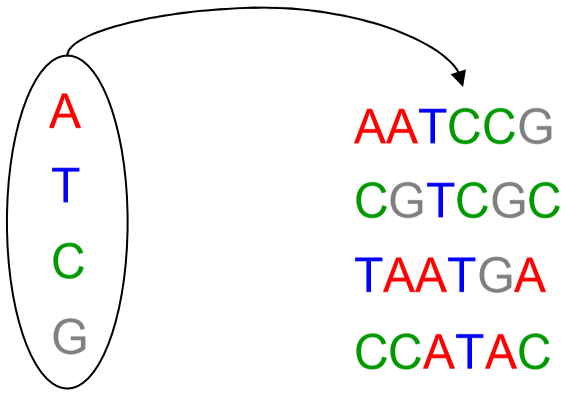
Νουκλεοτιδική Σύσταση DNA



Νουκλεοτιδική Σύσταση DNA



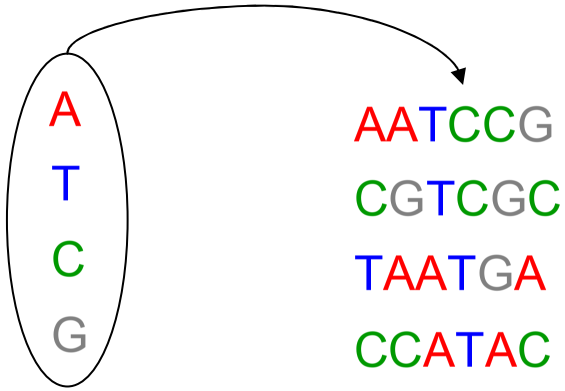
Νουκλεοτιδική Σύσταση DNA



B^1 = {A, T, C, G}
B^2 = {AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG}
B^3= {AAA, AAT, AAC, AAG, …, GGA, GGT, GGC, GGG}

B^k={…}

Νουκλεοτιδική Σύσταση DNA



B^1 = {A, T, C, G}
B^2 = {AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG}
B^3= {AAA, AAT, AAC, AAG, …, GGA, GGT, GGC, GGG}

B^k={…}

$|B|^1$=4

$|B|^2$=16

$|B|^3$=64

$|B|^4$=256

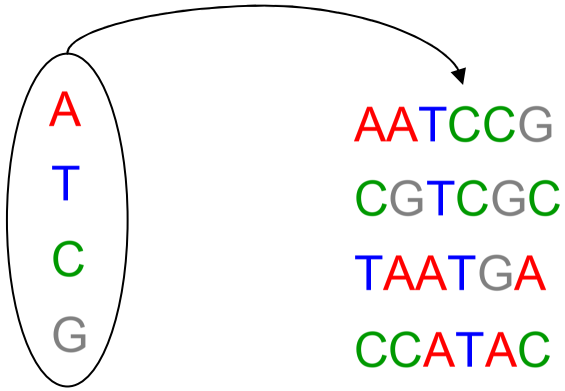
$|B|^5$=1024

$|B|^6$=4096

$|B|^7$=16384

$|B|^8$=65536

Νουκλεοτιδική Σύσταση DNA



ATCCGATGGG $\xrightarrow{1^{st}\text{ order MM}}$

B^1 = {A, T, C, G}
B^2 = {AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG}
B^3= {AAA, AAT, AAC, AAG, …, GGA, GGT, GGC, GGG}

B^k={…}

$|B|^1=4$

$|B|^2=16$

$|B|^3=64$

$|B|^4=256$

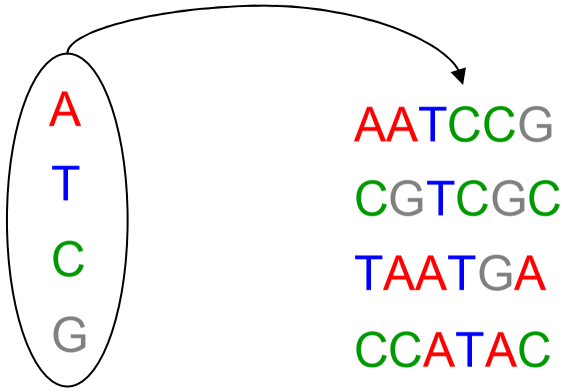
$|B|^5=1024$

$|B|^6=4096$

$|B|^7=16384$

$|B|^8=65536$

Νουκλεοτιδική Σύσταση DNA



ATCCGATGGG $\xrightarrow{1^{st}\text{ order MM}}$ p(A)= p(G) (x) p(A|G)

B^1 = {A, T, C, G}
B^2 = {AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG}
B^3= {AAA, AAT, AAC, AAG, …, GGA, GGT, GGC, GGG}

B^k={…}

$|B|^1$=4

$|B|^2$=16

$|B|^3$=64

$|B|^4$=256

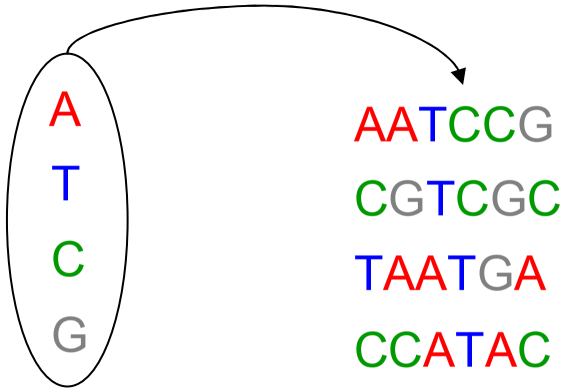
$|B|^5$=1024

$|B|^6$=4096

$|B|^7$=16384

$|B|^8$=65536

Νουκλεοτιδική Σύσταση DNA



ATCCGATGGG $\xrightarrow{1^{st}\text{ order MM}}$ $p(A)= p(G)\ (x)\ p(A|G)$

ATCCGATGGG $\xrightarrow{2^{nd}\text{ order MM}}$

$B^1 = \{A, T, C, G\}$
$B^2 = \{AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG\}$
$B^3 = \{AAA, AAT, AAC, AAG, \ldots, GGA, GGT, GGC, GGG\}$

$B^k = \{\ldots\}$

$|B|^1 = 4$

$|B|^2 = 16$

$|B|^3 = 64$

$|B|^4 = 256$

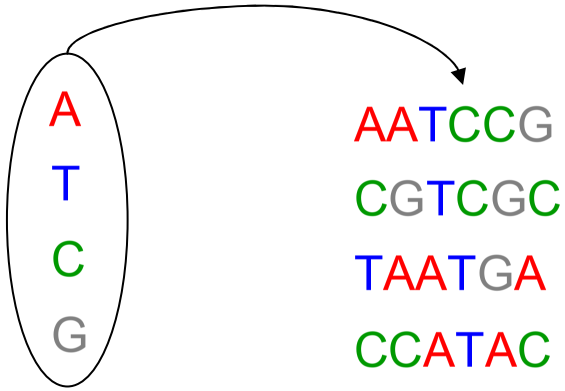
$|B|^5 = 1024$

$|B|^6 = 4096$

$|B|^7 = 16384$

$|B|^8 = 65536$

Νουκλεοτιδική Σύσταση DNA



ATCCGATGGG $\xrightarrow{1^{st}\text{ order MM}}$ p(A)= p(G) (x) p(A|G)

ATCCGATGGG $\xrightarrow{2^{nd}\text{ order MM}}$ p(A)= p(CG) (x) p(A|CG)

B^1 = {A, T, C, G}
B^2 = {AA, AT, AC, AG, TA, TT, TC, TG, CA, CT, CC, CG, GA, GT, GC, GG}
B^3= {AAA, AAT, AAC, AAG, …, GGA, GGT, GGC, GGG}

B^k={…}

$|B|^1$=4

$|B|^2$=16

$|B|^3$=64

$|B|^4$=256

$|B|^5$=1024

$|B|^6$=4096

$|B|^7$=16384

$|B|^8$=65536

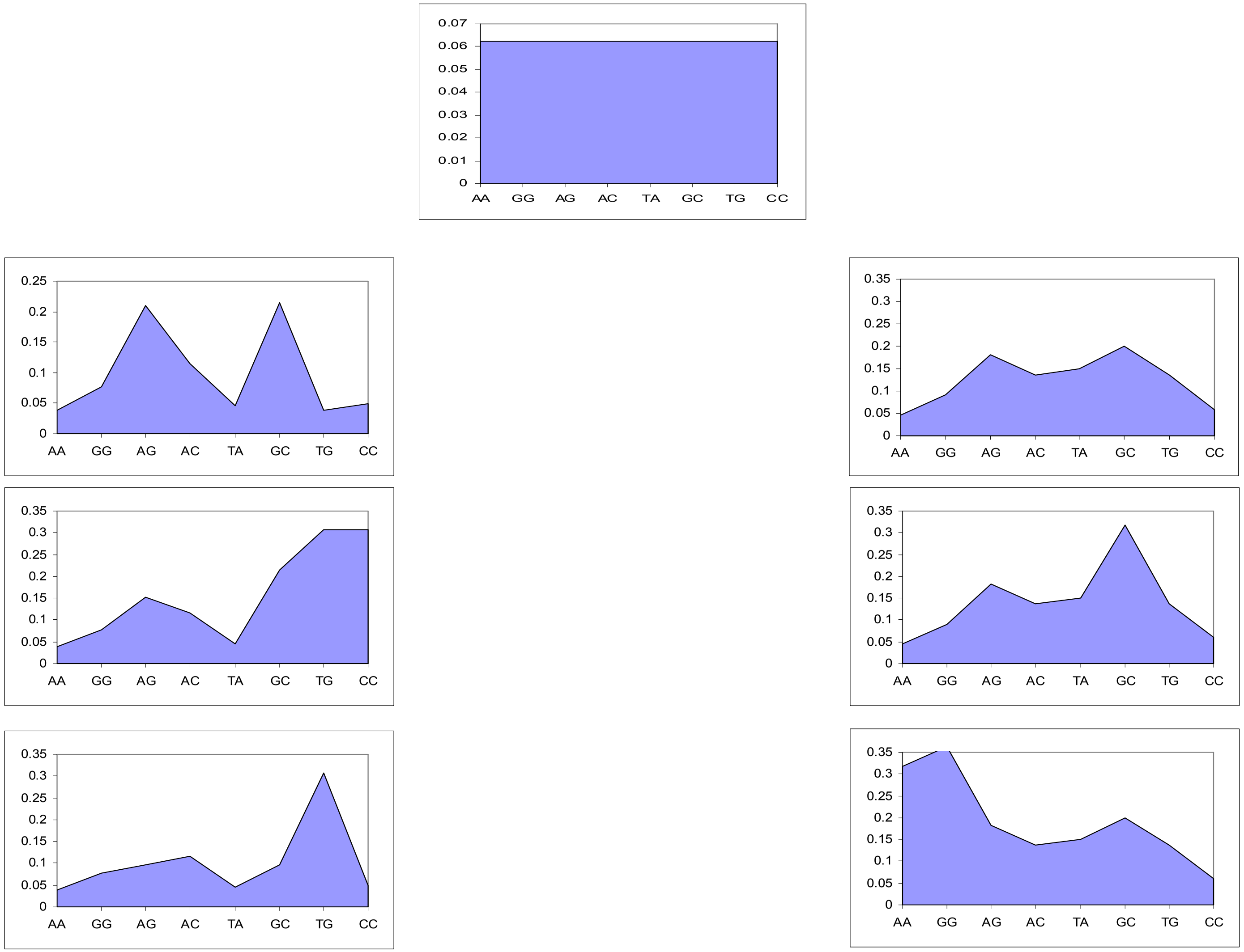
Codon Usage

Salmonella enterica subsp. enterica serovar Typhi str. CT18 **[gbbct]: 368 CDS's (89135 codons)**

fields: [triplet] [amino acid] [fraction] [frequency: **per thousand**] ([number])

UUU	F	0.51	20.3	(1811)	UCU	S	0.17	11.7	(1043)	UAU	Y	0.51	16.2	(1440)	UGU	C	0.43	5.5	(491)
UUC	F	0.49	19.7	(1759)	UCC	S	0.15	10.8	(964)	UAC	Y	0.49	15.7	(1399)	UGC	C	0.57	7.3	(648)
UUA	L	0.11	10.7	(954)	UCA	S	0.19	13.3	(1186)	UAA	*	0.50	2.1	(184)	UGA	*	0.38	1.6	(140)
UUG	L	0.13	12.5	(1114)	UCG	S	0.13	9.5	(844)	UAG	*	0.12	0.5	(44)	UGG	W	1.00	12.4	(1105)
CUU	L	0.16	15.2	(1356)	CCU	P	0.24	9.5	(849)	CAU	H	0.53	11.2	(994)	CGU	R	0.26	14.7	(1310)
CUC	L	0.14	12.8	(1144)	CCC	P	0.15	6.0	(533)	CAC	H	0.47	9.7	(865)	CGC	R	0.31	17.1	(1523)
CUA	L	0.06	5.4	(483)	CCA	P	0.25	9.9	(880)	CAA	Q	0.33	12.4	(1105)	CGA	R	0.12	6.7	(598)
CUG	L	0.39	36.6	(3263)	CCG	P	0.37	14.8	(1316)	CAG	Q	0.67	25.0	(2230)	CGG	R	0.14	7.8	(696)
AUU	I	0.43	24.8	(2212)	ACU	T	0.23	13.1	(1171)	AAU	N	0.48	21.4	(1910)	AGU	S	0.15	10.7	(951)
AUC	I	0.43	24.5	(2182)	ACC	T	0.31	17.8	(1591)	AAC	N	0.52	23.1	(2060)	AGC	S	0.21	15.0	(1333)
AUA	I	0.14	8.0	(713)	ACA	T	0.21	12.1	(1078)	AAA	K	0.59	34.5	(3074)	AGA	R	0.10	5.6	(503)
AUG	M	1.00	27.4	(2438)	ACG	T	0.26	15.2	(1351)	AAG	K	0.41	23.7	(2116)	AGG	R	0.07	4.1	(363)
GUU	V	0.31	20.7	(1847)	GCU	A	0.22	18.1	(1613)	GAU	D	0.56	31.1	(2772)	GGU	G	0.29	18.2	(1625)
GUC	V	0.24	16.2	(1447)	GCC	A	0.28	23.1	(2062)	GAC	D	0.44	24.8	(2214)	GGC	G	0.35	22.6	(2012)
GUA	V	0.17	11.6	(1033)	GCA	A	0.25	20.3	(1813)	GAA	E	0.57	37.7	(3356)	GGA	G	0.17	11.0	(982)
GUG	V	0.28	18.6	(1657)	GCG	A	0.25	20.5	(1828)	GAG	E	0.43	28.0	(2497)	GGG	G	0.19	11.9	(1060)

Compositional Distributions



G+C% content

- ✓ Έυρος: 25-75%: 72.1% Streptomyces coelicolor, 26.5% Wigglesworthia glossinidia
- ✓ Βακτήρια του εδάφους: 49%, υποχρεωτικά παράσιτα: 38%
- ✓ GTP και CTP: ενεργειακά πιο δαπανηρά από ATP και UTP
- ✓ Κεντρικός ρόλος ATP μεταβολισμό: μεγαλύτερη διαθεσιμότητα
- ✓ Περιορισμένη διαθεσιμότητα θρεπτικών: υψηλό AT
- ✓ Επιδιόρθωση DNA: μικρά γονιδιώματα – συσσώρευση μεταλλάξεων
- ✓ Η πιο συχνή μετάλλαξη: C → T (G → A)

Bentley 2004

G+C% content

GC content καταμήκος του γονιδιώματος δεν ακολουθεί ομοιόμορφη κατανομή:

1. Μικρό μέγεθος γονιδιώματος, πλούσιο σε AT:
    a. Μεταξύ γονιδίων (intergenic)
    b. Μη κωδικές περιοχές (non-coding)
    c. 3η θέση μέσα στο κωδικόνιο (3rd codon position)
    d. Ακολουθίες υποκινητών πριν από τα γονίδια (υπερελίκωση, ξεδιπλώνεται πιο εύκολα)
    e. Μερικές εκατοντάδες βάσεις γύρω από την περιοχή της έναρξης της αντιγραφής του DNA (origin of replication)

Bentley 2004

G+C% content

GC content καταμήκος του γονιδιώματος δεν ακολουθεί ομοιόμορφη κατανομή:

2. Σε επίπεδο ολόκληρου γονιδιώματος: Μικρή προτίμηση για GC προς την έναρξη και ATπρος την λήξη της αντιγραφής του DNA. Αυτό μπορεί να οφείλεται σε δομικούς περιορισμούς του μορίου του DNA ή στη φυσική και λειτουργική διαμερισματοποίηση του χρωμοσώματος
3. Λόγω εκφυλισμού του γενετικού κώδικα, πρότιμηση για AT στην 3η θέση του κωδικονίου είναι λιγότερο πιθανό να οδηγήσει σε αλλαγή του αμινοξέος που κωδικοποιεί

Bentley 2004

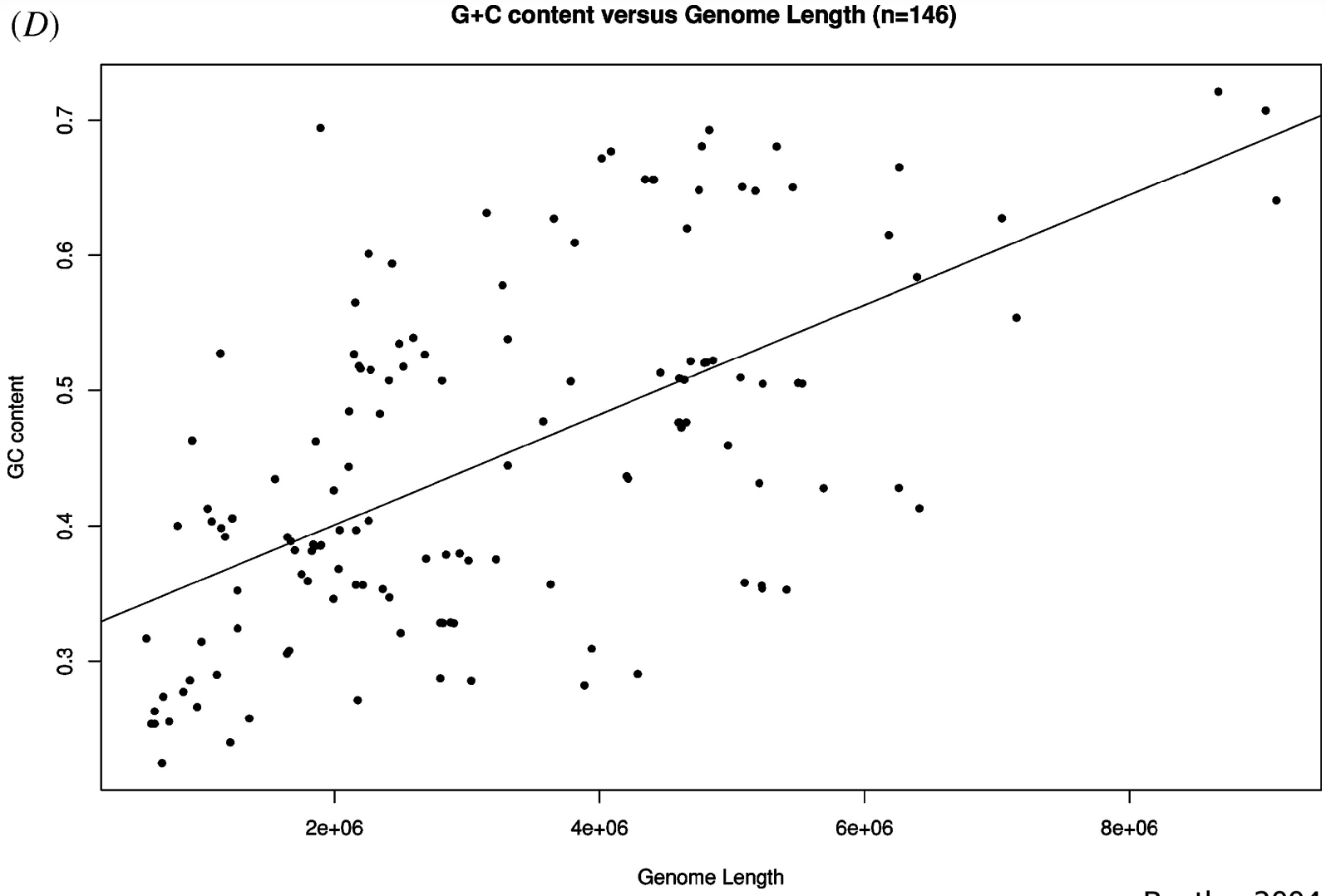
G+C% content

GC content καταμήκος του γονιδιώματος δεν ακολουθεί ομοιόμορφη κατανομή:

4. G-C/G + C στην + και την – αλυσίδα του DNA (GC skew). Στους προκαρυωτικούς υπάρχει προτίμηση για G και όχι για C στην + αλυσίδα του DNA που δημιουργεί ένα διφασικό πρότυπο κατά μήκος του γονιδιώματος το οποίο είναι χρήσιμο για τον εντοπισμό του σημείου της έναρξης και της λήξης της αντιγραφής του DNA.

Bentley 2004

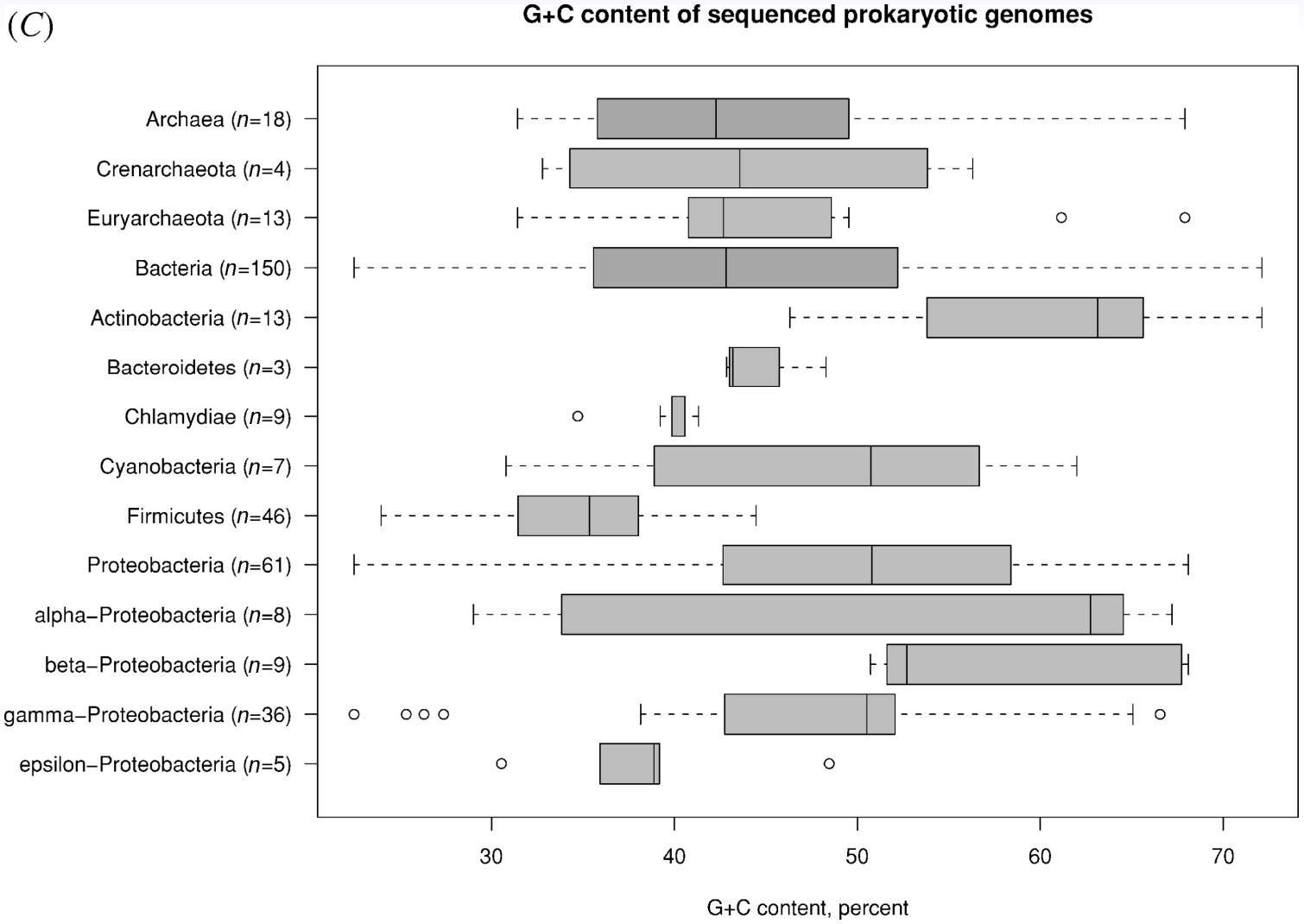
G+C% content



Bentley 2004

G+C% content

(C)

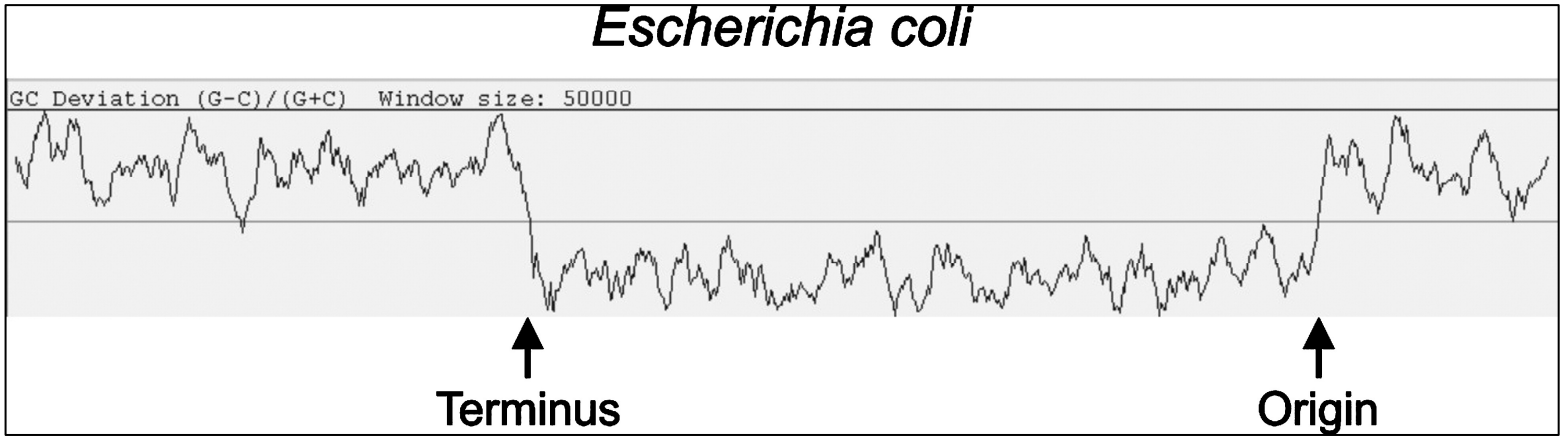


Bentley 2004

G+C content ~ phylogeny

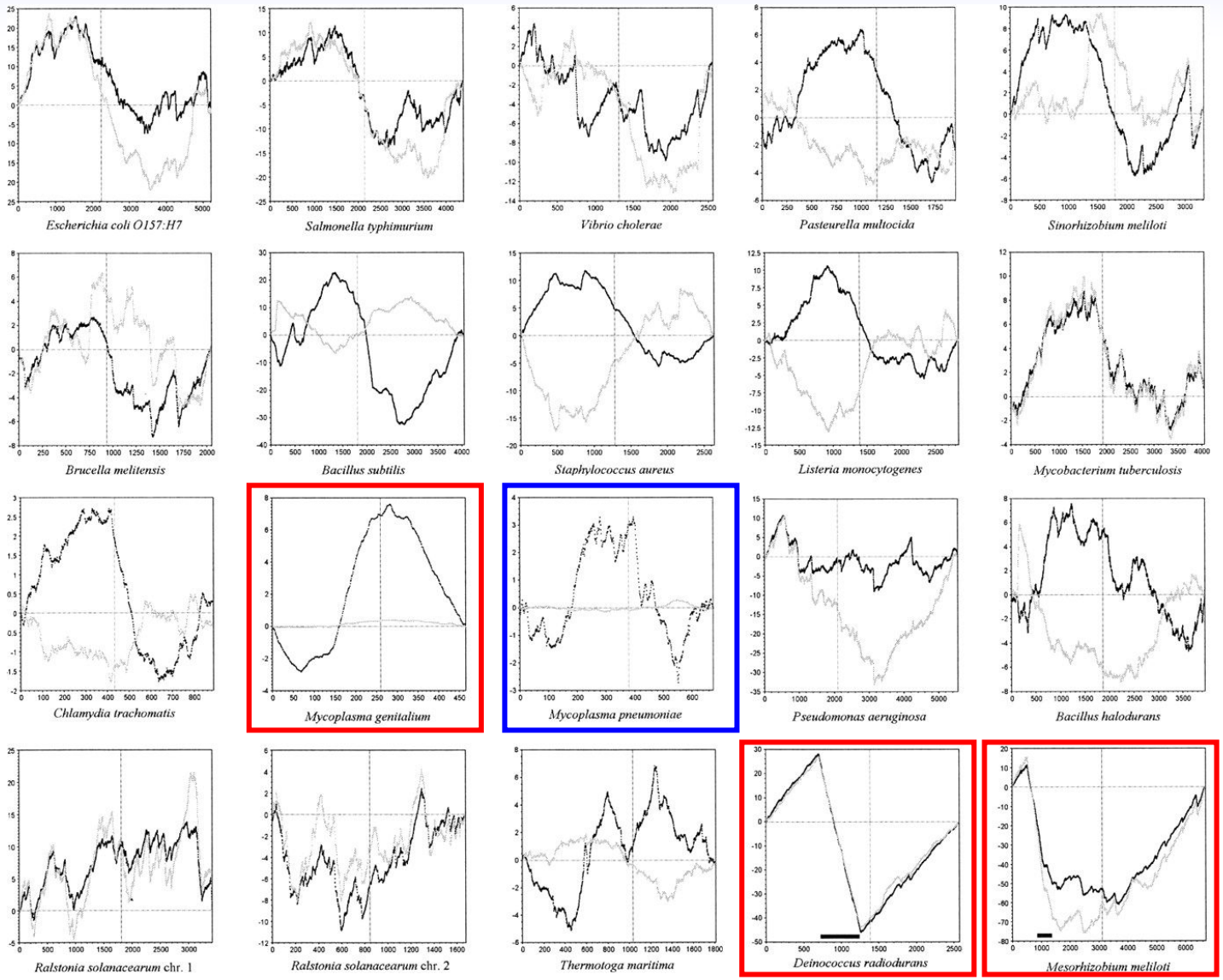
- ✓E. coli: 50%
- ✓Shigella: 51%
- ✓Salmonella: 52%
- ✓Staphylococcus: 33%
- ✓Streptococcus: 38%

G+C% content



Bentley 2004

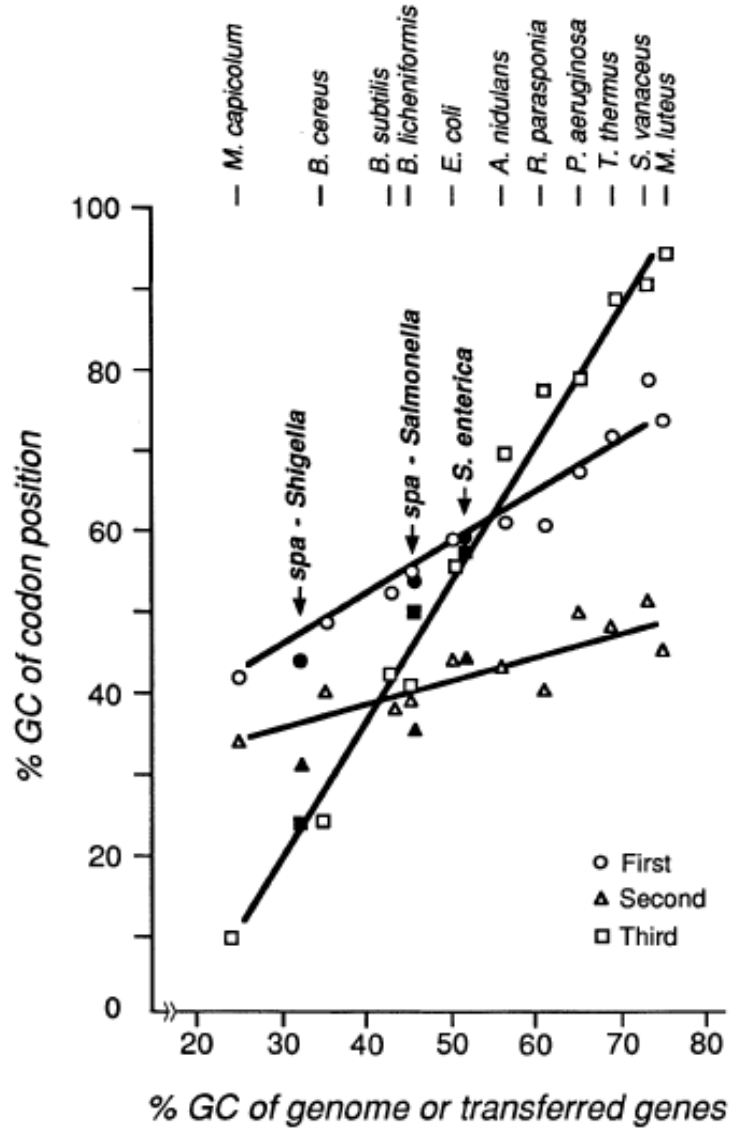
$G+C_3$ & CAI



Cumulative G+C3 (black curve) and CAI (gray curve)

Daubin 2003

$G+C_{1,2,3}$ vs G+C



$GC_1 = 0.615 \times GC + 26.9$
$GC_2 = 0.270 \times GC + 26.7$
$GC_3 = 1.692 \times GC - 32.3$

Lawrence 1997

Μέγεθος Γονιδιώματος

1. Εύρος: 12X
2. Επικάλυψη: Μεγάλοι ιοί - βακτήρια- μικροί ευκαρυωτικοί
3. Μεγάλο εύρος διακύμανσης ακόμα και μέσα στο ίδιο είδος:

   Escherichia coli, Prochlorococcus marinus, και Streptomyces coelicolor

   ποικίλουν~ 1,000,000 bp

Bentley 2004

Μέγεθος Γονιδιώματος

4. Μωσαικά πολλαπλών γενετικών γεγονότων: διπλασιασμός γονιδίων (gene duplication), οριζόντια μεταφορά (horizontal acquisition), απώλεια γονιδίων (gene loss), (ανασυνδιασμός) recombination -> καλός δείκτης της εξελικτικής μονάδας (evolutionary lineage)
5. minimal και maximal γονιδίωμα
6. Συσχετίζεται με:

   χρόνο διαίρεσης του κυττάρου, ρυθμό αντιγραφής DNA, διαθεσιμότητα ενέργειας, φυσικός χώρος μέσα στο κύτταρο

Bentley 2004

Μέγεθος Γονιδιώματος

7. Οικοσυστήματα σταθερών συνθηκών: μικρά γονιδιώματα
8. Οικοσυστήματα μεταβλητών συνθηκών: μεγάλα γονιδιώματα
9. 0,5Mb Nanoarchaeum equitans
10. 9Mb Bradyrhizobium japonicum
11. Κωδική πυκνότητα (coding density): 1 γονίδιο/kb. Μεγαλύτερα γονιδιώματα → περισσότερα γονίδια (δεν ισχύει για ευκαρυωτικούς)

Bentley 2004

Μέγεθος Γονιδιώματος

12. Περισσότερες οικογένειες πρωτεινών ή περισσότερα μέλη σε υπάρχουσες οικογένειες?
13. Ανεξαρτήτου μεγέθους:

    μετάφραση, δομή ριβοσώματος, και βιογένεση πρωτεινών
14. Εξαρτώμενο από το μέγεθος:

    Κυτταρικός μεταβολισμός (γραμμικά)

    Ρύθμιση γονιδιακής έκφρασης (μη-γραμμικά)

Bentley 2004

Τοπολογία γονιδίων

1. Τοπολογία στη φορά της αντιγραφής του DNA, μεταγραφόμενα μακρυά από το σημείο της έναρξης της αντιγραφής

2. Ελαχιστοποίηση του αριθμού των συγκρούσεων μεταξύ των συμπλόκων αντιγραφής και μεταγραφής καθώς αυτά κινούνται καταμήκος του DNA. Τέτοιες συγκρούσεις σταματούν την αντιγραφή και διακόπτουν ή ακυρώνουν τη μεταγραφή

Daubin 2003 Bentley 2004

Τοπολογία γονιδίων

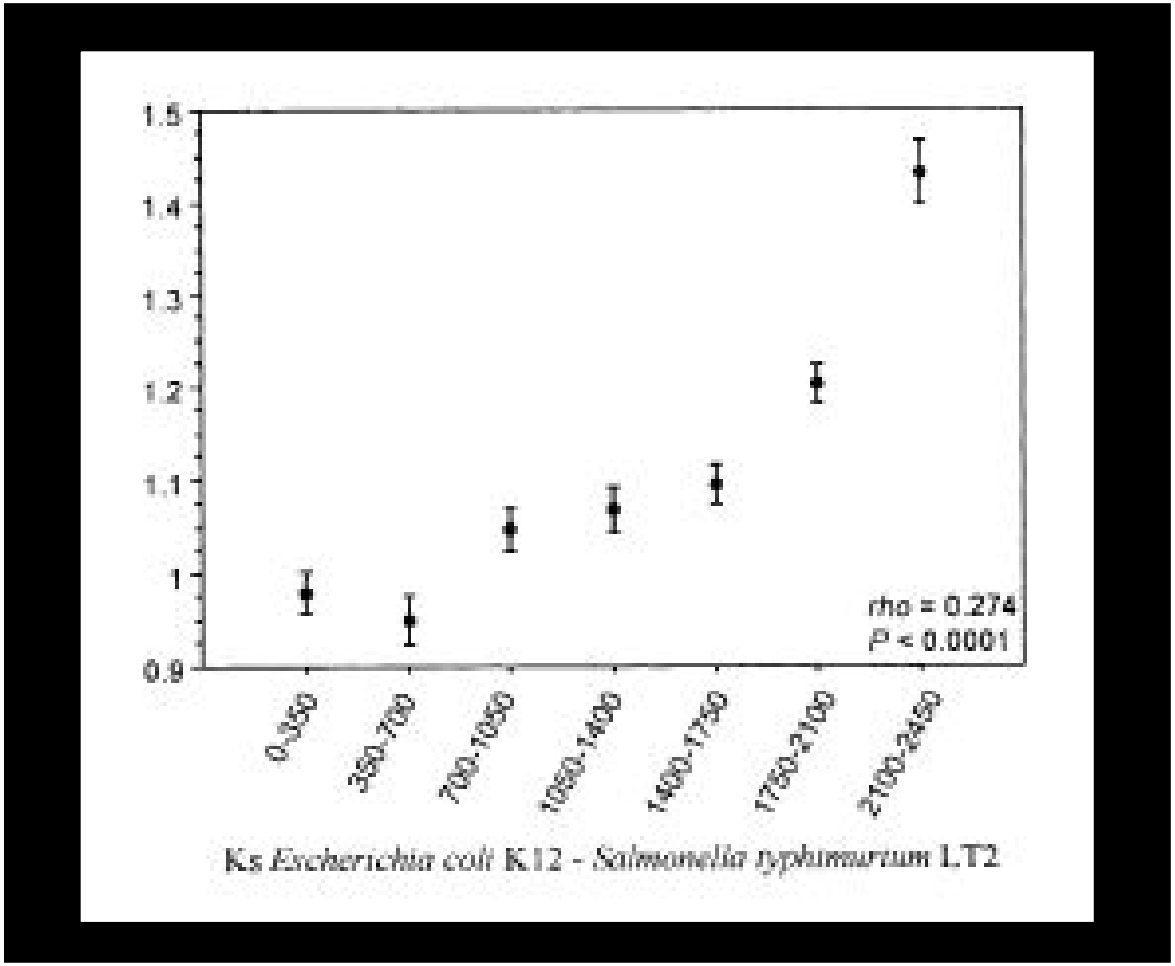
3. Συσχετίζεται με τα επίπεδα έκφρασης έτσι ώστε γονίδια με υψηλά επίπεδα έκφρασης να είναι τοποθετημένα στην οδηγό αλυσίδα του DNA το οποίο συμφωνεί με την συστηματική παρουσία του rDNA και των οπερονίων των ριβοσωμικών πρωτεινών στην ίδια αλυσίδα

4. (Πιο πρόσφατα) πιθανά η τοπολογία να συσχετίζεται με το εάν τα γονίδια είναι βασικά ή βοηθητικά. Η πλειοψηφία των πρώτων βρίσκεται μακρυά από το σημείο έναρξης της αντιγραφής (σχετίζεται πιθανά με την τοξικότητα των κουτσουρεμένων προϊόντων μετάφρασης των βασικών γονιδίων σε περίπτωση λάθους)

5. Γονίδια κοντά στο σημείο λήξης της αντιγραφής (υψηλό AT) εξελίσσονται πιο γρήγορα, ενώ γονίδια στο σημείο έναρξης είναι πιο συντηρημένα μεταξύ των ειδών

Daubin 2003 Bentley 2004

Τοπολογία γονιδίων



Daubin 2003 Bentley 2004

Δείκτες νουκλεοτιδικής σύστασης

Δείκτες	Περιγραφή
Codon Adaptation Index (CAI)	Ποσοτικοποιεί την σχετική προσαρμοστικότητα της χρήσης κωδικονίων προς την χρήση κωδικονίων που απαντούν σε γονίδια με υψηλά επίπεδα έκφρασης
Frequency of Optimal codons (Fop)	Ο λόγος των άριστων προς τα συνώνυμα κωδικόνια
Codon Bias Index (CBI)	Ποσοτικοποιεί τον βαθμό στον οποίο ένα γονίδιο χρησιμοποιεί ένα σετ από άριστα κωδικόνια
Effective number of codons (NC)	Ποσοτικοποιεί το πόσο μακρυά βρίσκεται η χρήση κωδικονίων από την ομοιόμορφη χρήση συνώνυμων κωδικονίων. Ένα γονίδιο με ένα κωδικόνιο για κάθε αμινοξύ έχει τη μεγαλύτερη πόλωση και την μικρότερη τιμή (20) αυτού του δείκτη. Γονίδιο με ομοιόμορφη χρήση κωδικονίων έχει τιμή 61
GC content	Ποσοτικοποιεί τη συχνότητα γουανίνης ή κυτοσίνης
GC_1 and GC_3 content	Ποσοτικοποιεί τη συχνότητα γουανίνης ή κυτοσίνης στην $1^η$ και $3^η$ θέση του κωδικονίου
δ* difference	Είναι η μέση, απόλυτη διαφορά της σχετικής αφθονίας των δινουκλεοτιδίων μεταξύ δυο ακολουθιών. Υπολογίζεται από τη συχνότητα των δινουκλεοτιδίων ομαλοποιημένη με το γινόμενο της συχνότητας των μονο-νουκλεοτιδίων που απαρτίζονται
Codon usage contrasts	Συγκρίνει πολώσεις (biases) στη χρήση κωδικονίων ενός γονιδίου με τη μέση πόλωση
Amino acid contrasts	Συγκρίνει πολώσεις στη χρήση αμινοξέων μιας πρωτεΐνης με τη μέση πόλωση
High order motifs	Συγκρίνει τη συχνότητα «λέξεων» μεγέθους *κ* ενός κυλιόμενου παραθύρου με την αντίστοιχη συχνότητα τους μέσα σε ολόκληρο το γονιδίωμα
Translational efficiency (P2)	Το ποσοστό των κωδικονίων που συμμορφώνονται με την ενδιαμεση ενέργεια αλληλεπίδρασης κωδικωνίων-αντικωδικονίων με βάση τον κανόνα Grosjean και Fiers.
Intrinsic codon bias index (ICDI)	Εκτιμάει πολώσεις στη χρήση κωδικονίων για γονίδια που ανήκουν σε γονιδιώματα που δεν γνωρίζουμε τα άριστα κωδικόνια. Σχετίζεται πολύ με το CBI, NC
Scaled Chi-square	Ποσοτικοποιεί το μέγεθος της παρατηρούμενης πόλωσης από την μη-ομοιόμορφη χρήση συνώνυμων κωδικονίων, χρησιμοποιώντας ως εκτιμήσεις την ομοιόμορφη χρήση. Το αποτέλεσμα ομαλοποιείται διαιρώντας με τον αριθμό των κωδικονίων ενός γονιδίου

δ*, codon usage and aminoacid contrasts

$$\{\rho^*_{xy} = f^*_{xy} / f^*_x f^*_y\} \quad (1)$$

$$\delta^*(f,g) = \frac{1}{16}\sum |\rho^*_{xy}(f) - \rho^*_{xy}(g)| \quad (2)$$

$$\sum_{(x,y,z)=a} g(x,y,z) = 1 \quad (3)$$

$$B(F/G) = \sum_a p_a(F)\left[\sum_{(x,y,z)=a} |f(x,y,z) - g(x,y,z)|\right] \quad (4)$$

$$A(F/G) = (1/20)\sum_{i=1}^{20} |a_i(F) - a_i(G)| \quad (5)$$

Karlin 2001

Codon Adaptation Index

$$w_{aa,i}(G) = \frac{f_{aa,i}(G)}{f_{aa,\max}(G)} \quad (1)$$

The relative adaptiveness of a codon is defined as its frequency relative to the most often used synonymous codon, where ***G*** is a set of highly expressed genes. ($0 \leq w \leq 1$)

$$CAI_g = \prod_{i=1}^{N} w_i^{1/N} \quad (2)$$

The CAI of a gene ***g*** is then simply the geometric average of the relative adaptiveness of all codons in a gene sequence.

$$CAI_g = \prod_{k=1}^{61} w_k^{X_{k,g}} \quad (3)$$

$$X_{k,g} = \frac{C_{k,g}}{\sum_{i=1}^{61} C_{i,g}} \quad (4)$$

http://codonw.sourceforge.net/

Jansen 2003

Effective Number of Codons

$$\hat{f} = \frac{(n\sum_{i=1}^{k} p_i^2) - 1}{n - 1} \quad (1)$$

Codon homozygosity: between 20 and 61 and tells to what degree the codon usage in a gene is biased, i.e., it approaches 20 codons for the extremely biased genes, and approaches 61 for the genes where all possible codons are used with no preference.

$$\hat{N}c(aa) = \frac{1}{\hat{F}_{aa}} \quad (2)$$

$$\hat{N}c = 2 + \frac{9}{\hat{F}_2} + \frac{1}{\hat{F}_3} + \frac{5}{\hat{F}_4} + \frac{3}{\hat{F}_6} \quad (3)$$

where F2 is the average homozygosity for the amino acids having a degeneracy of two (histidine, glutamine, etc.) and so on.

Fuglsang 2004

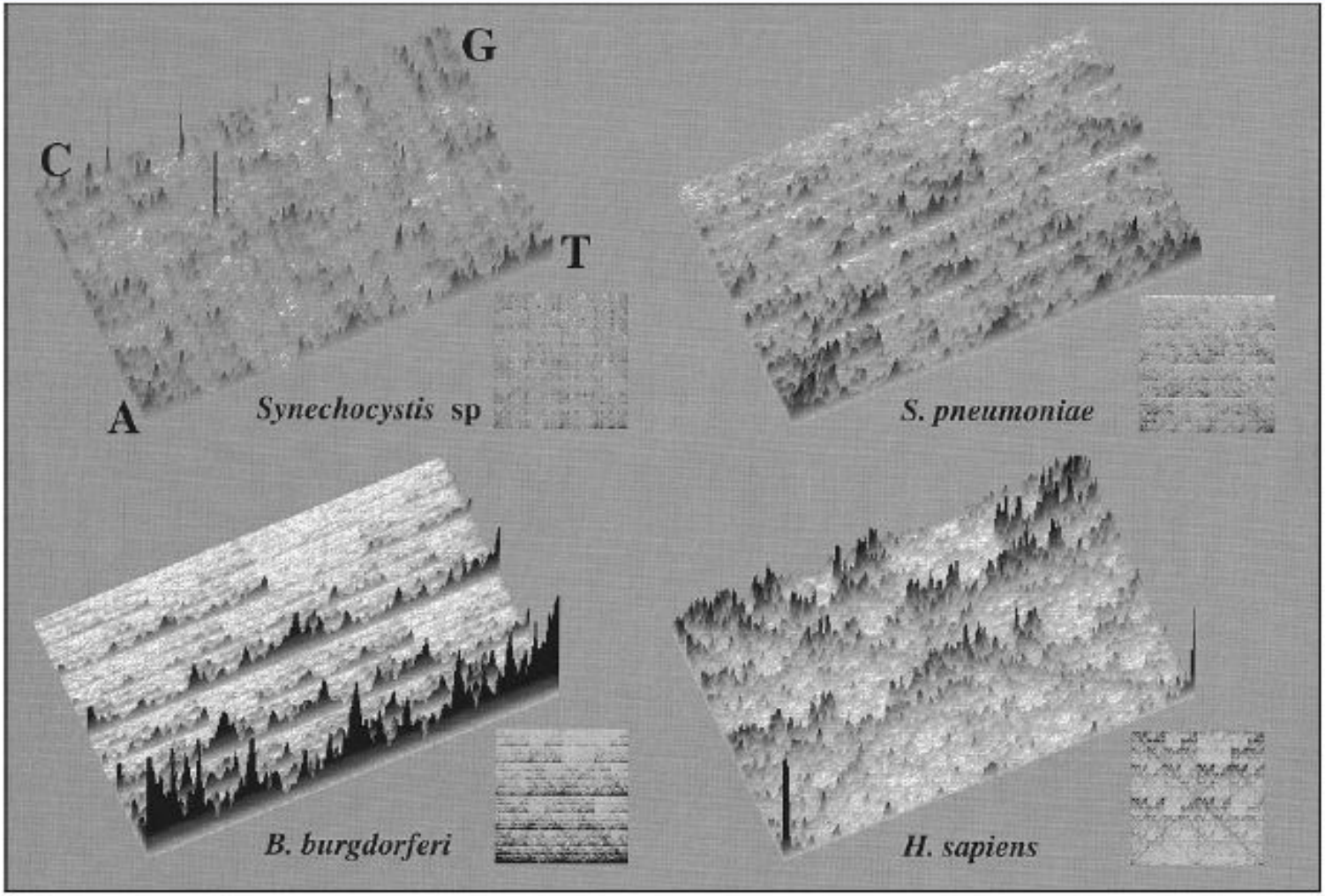
Effective Number of Codons

Salmonella enterica subsp. enterica serovar Typhi str. CT18 **[gbbct]: 368 CDS's (89135 codons)**

fields: [triplet] [amino acid] [fraction] [frequency: **per thousand**] ([number])

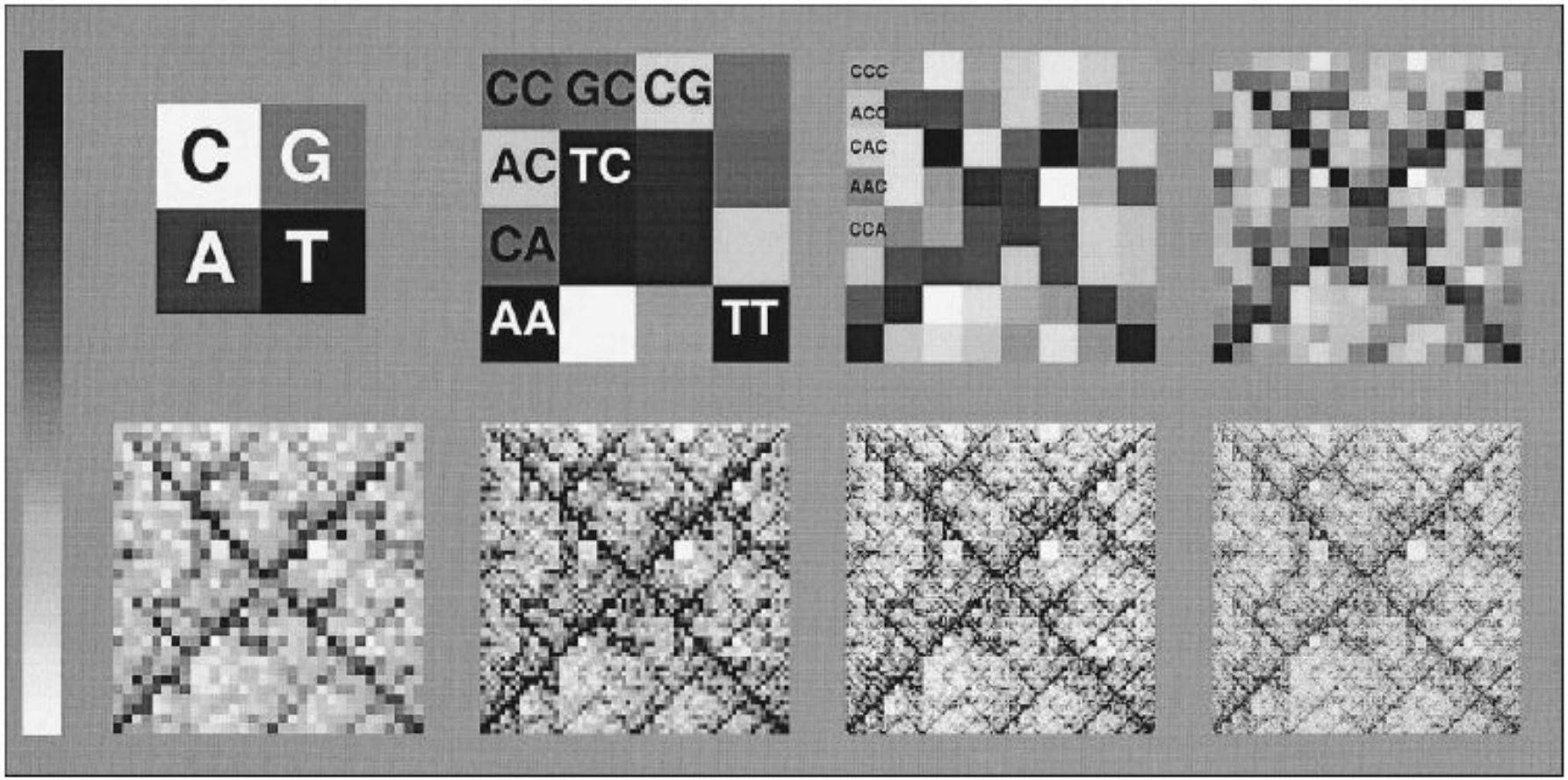
```
UUU F 0.51 20.3 (  1811)  UCU S 0.17 11.7 (  1043)  UAU Y 0.51 16.2 (  1440)  UGU C 0.43  5.5 (   491)
UUC F 0.49 19.7 (  1759)  UCC S 0.15 10.8 (   964)  UAC Y 0.49 15.7 (  1399)  UGC C 0.57  7.3 (   648)
UUA L 0.11 10.7 (   954)  UCA S 0.19 13.3 (  1186)  UAA * 0.50  2.1 (   184)  UGA * 0.38  1.6 (   140)
UUG L 0.13 12.5 (  1114)  UCG S 0.13  9.5 (   844)  UAG * 0.12  0.5 (    44)  UGG W 1.00 12.4 (  1105)

CUU L 0.16 15.2 (  1356)  CCU P 0.24  9.5 (   849)  CAU H 0.53 11.2 (   994)  CGU R 0.26 14.7 (  1310)
CUC L 0.14 12.8 (  1144)  CCC P 0.15  6.0 (   533)  CAC H 0.47  9.7 (   865)  CGC R 0.31 17.1 (  1523)
CUA L 0.06  5.4 (   483)  CCA P 0.25  9.9 (   880)  CAA Q 0.33 12.4 (  1105)  CGA R 0.12  6.7 (   598)
CUG L 0.39 36.6 (  3263)  CCG P 0.37 14.8 (  1316)  CAG Q 0.67 25.0 (  2230)  CGG R 0.14  7.8 (   696)

AUU I 0.43 24.8 (  2212)  ACU T 0.23 13.1 (  1171)  AAU N 0.48 21.4 (  1910)  AGU S 0.15 10.7 (   951)
AUC I 0.43 24.5 (  2182)  ACC T 0.31 17.8 (  1591)  AAC N 0.52 23.1 (  2060)  AGC S 0.21 15.0 (  1333)
AUA I 0.14  8.0 (   713)  ACA T 0.21 12.1 (  1078)  AAA K 0.59 34.5 (  3074)  AGA R 0.10  5.6 (   503)
AUG M 1.00 27.4 (  2438)  ACG T 0.26 15.2 (  1351)  AAG K 0.41 23.7 (  2116)  AGG R 0.07  4.1 (   363)

GUU V 0.31 20.7 (  1847)  GCU A 0.22 18.1 (  1613)  GAU D 0.56 31.1 (  2772)  GGU G 0.29 18.2 (  1625)
GUC V 0.24 16.2 (  1447)  GCC A 0.28 23.1 (  2062)  GAC D 0.44 24.8 (  2214)  GGC G 0.35 22.6 (  2012)
GUA V 0.17 11.6 (  1033)  GCA A 0.25 20.3 (  1813)  GAA E 0.57 37.7 (  3356)  GGA G 0.17 11.0 (   982)
GUG V 0.28 18.6 (  1657)  GCG A 0.25 20.5 (  1828)  GAG E 0.43 28.0 (  2497)  GGG G 0.19 11.9 (  1060)
```

Chaos Game Representation



Deschavanne 1999

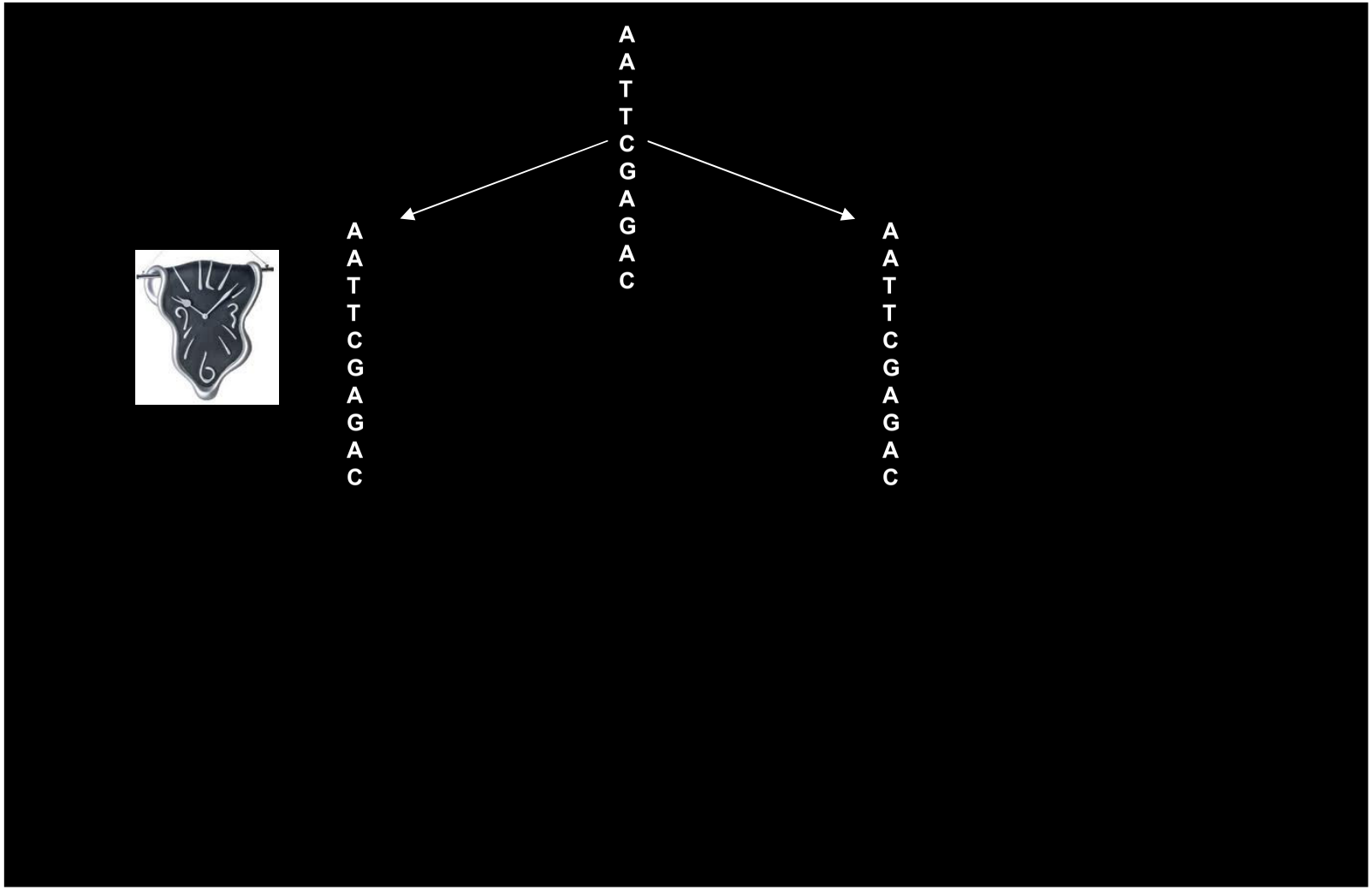
Chaos Game Representation



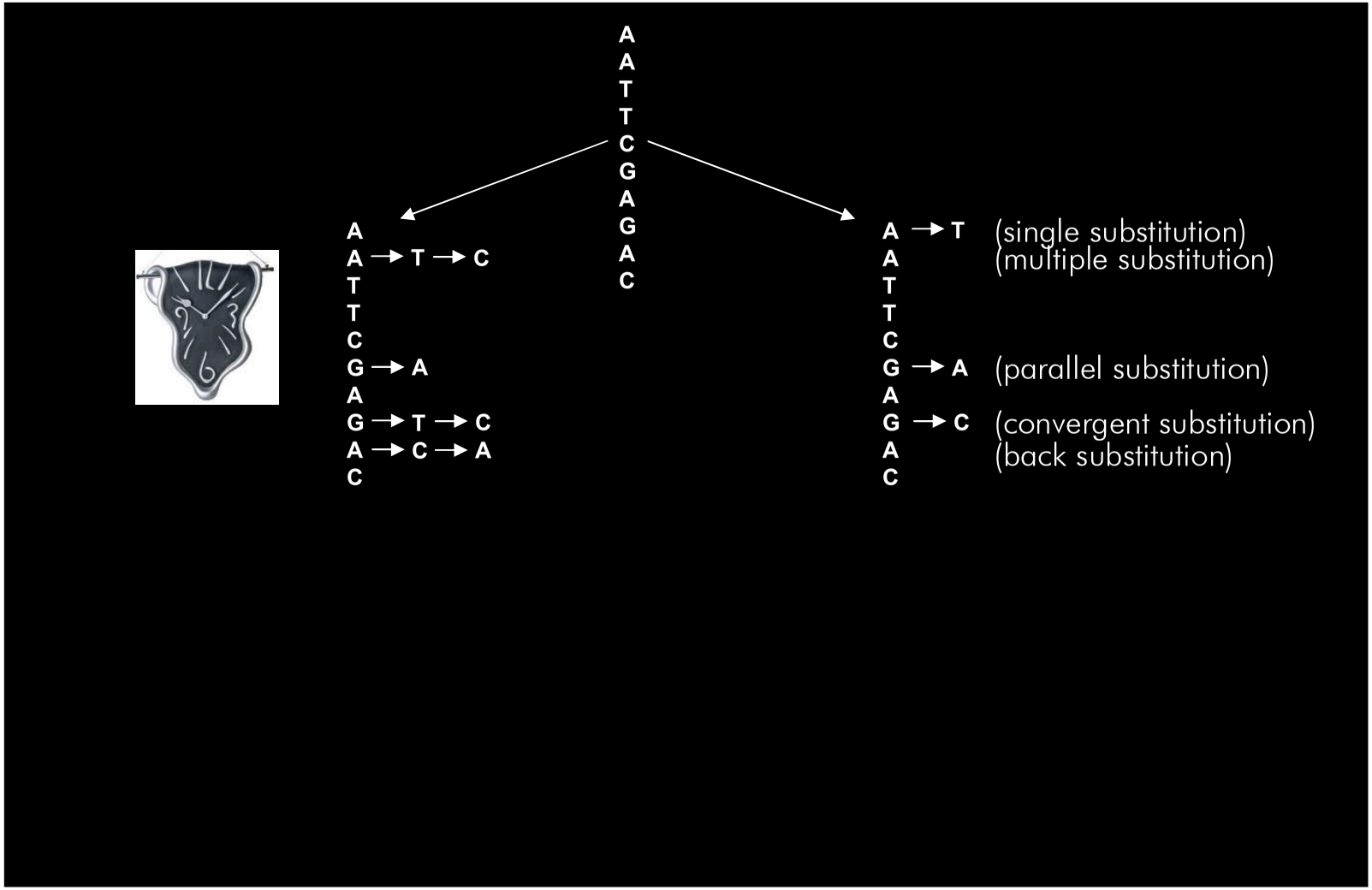
Deschavanne 1999

Οι ακολουθίες DNA δεν είναι αυτό που βλέπουμε

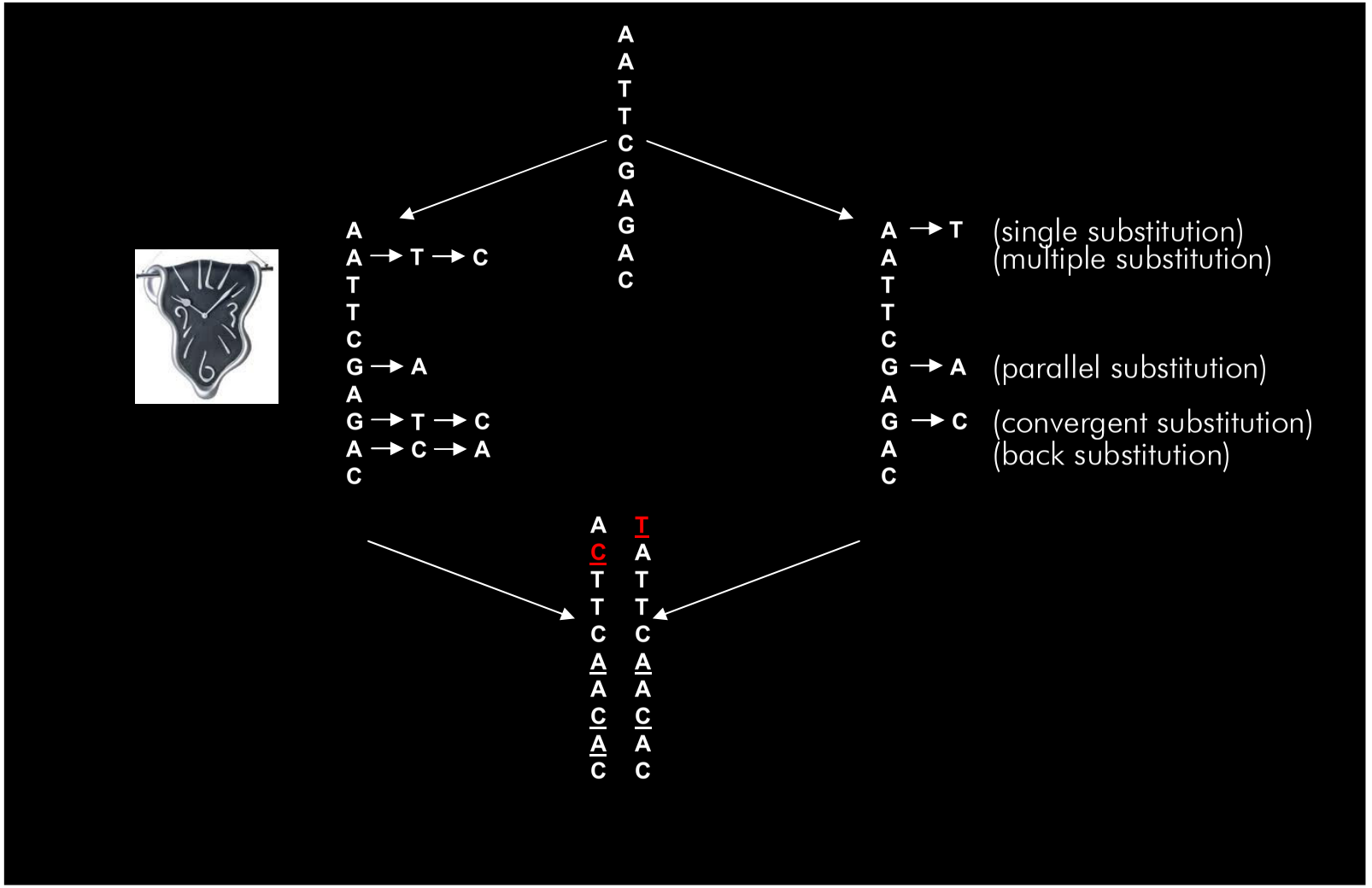
Οι ακολουθίες DNA δεν είναι αυτό που βλέπουμε



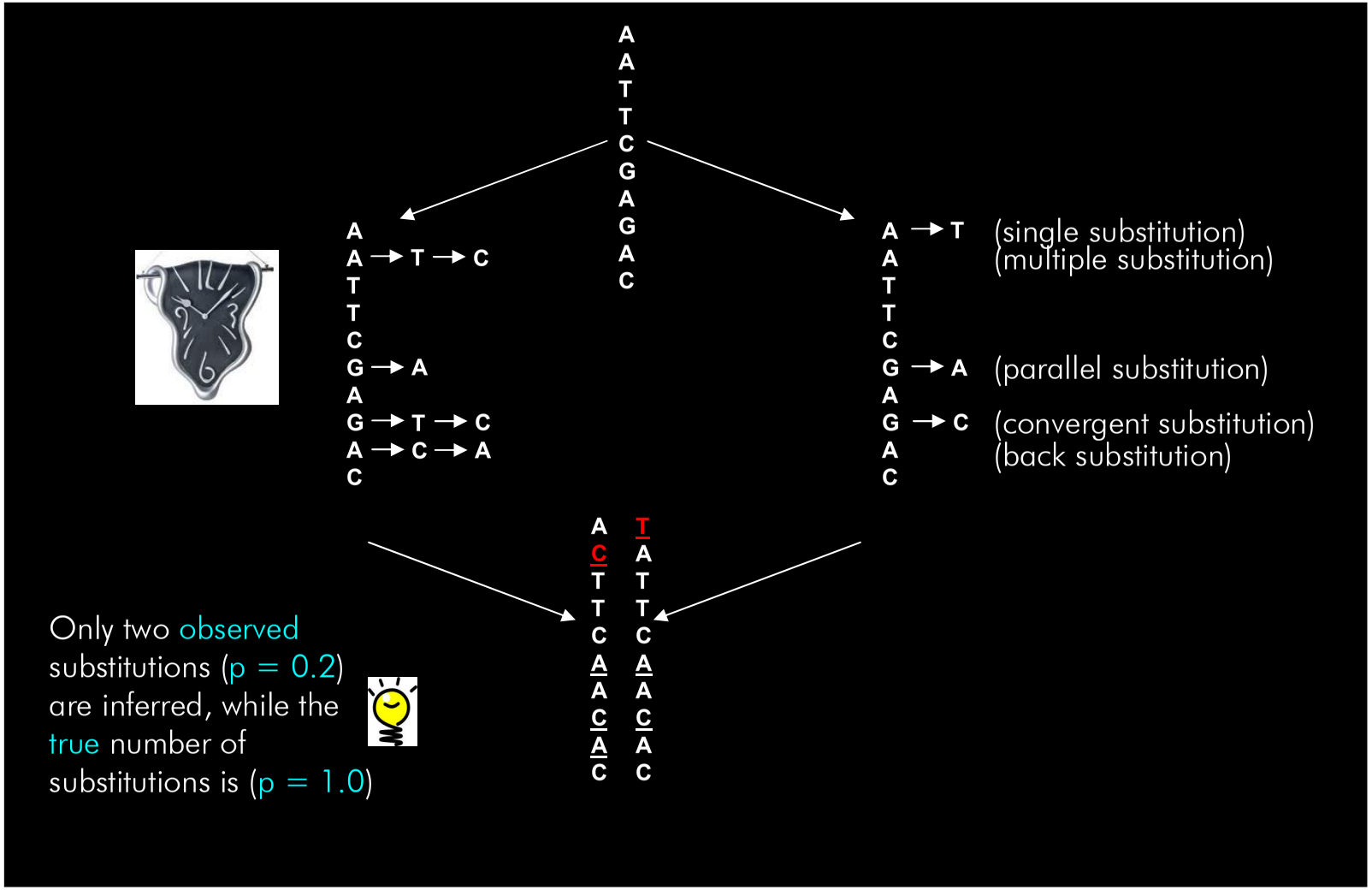
Οι ακολουθίες DNA δεν είναι αυτό που βλέπουμε



Οι ακολουθίες DNA δεν είναι αυτό που βλέπουμε



Οι ακολουθίες DNA δεν είναι αυτό που βλέπουμε



Μέγεθος αλφαβήτου και συχνότητα εμφάνισης

k-mer	counts	$\|4\|^k$
GCCAGCGC	24	4^8
CCAGCGC	49	4^7
CAGCGC	116	4^6
AGCGC	238	4^5
GCGC	738	4^4
CGC	2452	4^3
GC	9784	4^2
C	33854	4^1

k-mer	counts	$\|4\|^k$
AAAACATG	1	4^8
AAACATG	6	4^7
AACATG	26	4^6
ACATG	81	4^5
CATG	474	4^4
ATG	2110	4^3
TG	9243	4^2
G	32499	4^1

Διαφορετικότητα

seq1: GCCGCCCCCGCGCGCG

seq2: GCCGCCCGCCGCGCCG

Διαφορετικότητα

seq1: GCCGCCCCCGCGCGCG

seq2: GCCGCCCGCCGCGCCG

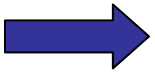
0th order
 # states: 4

seq1

A	0
T	0
G	6
C	10

seq2

A	0
T	0
G	6
C	10



"typical"

Διαφορετικότητα

seq1: GCCGCCCCCGCGCGCG

seq2: GCCGCCCGCCGCGCCG

0th order
 # states: 4

seq1	
A	0
T	0
G	6
C	10

seq2	
A	0
T	0
G	6
C	10

→ “typical”

1st order
 # states: 16

GC	5
CC	5
CG	5

GC	5
CC	5
CG	5

→ “typical”

Διαφορετικότητα

seq1: GCCGCCCCCGCGCGCG

seq2: GCCGCCCGCCGCGCCG

0th order
 # states: 4

seq1	
A	0
T	0
G	6
C	10

seq2	
A	0
T	0
G	6
C	10

→ "typical"

1st order
 # states: 16

GC	5
CC	5
CG	5

GC	5
CC	5
CG	5

→ "typical"

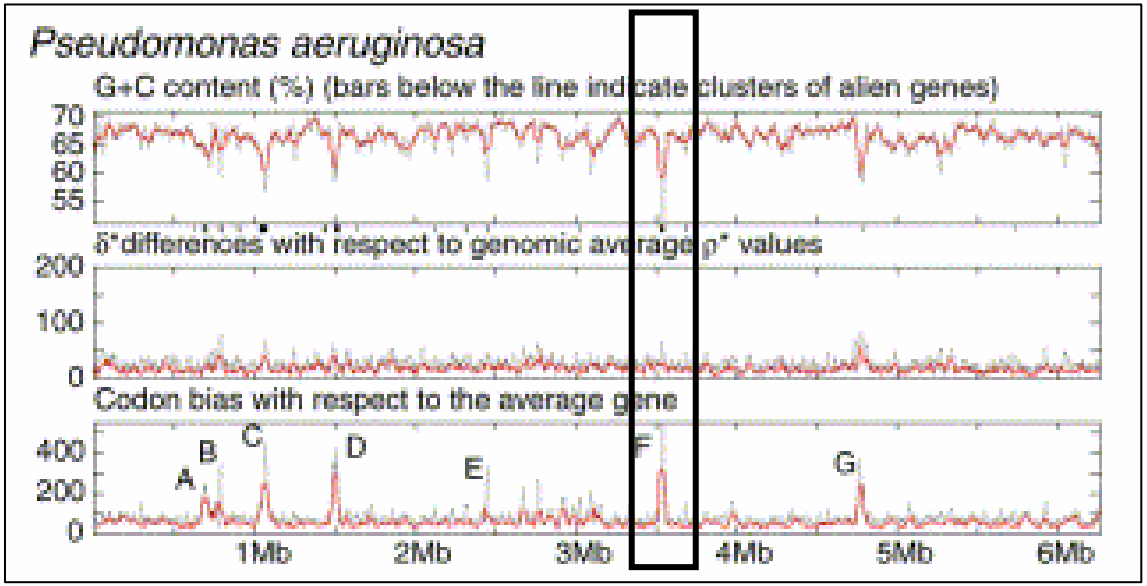
2nd order
 # states: 64

GCC	2
CCG	2
CGC	4
CCC	3
GCG	3

GCC	4
CCG	4
CGC	4
CCC	1
GCG	1

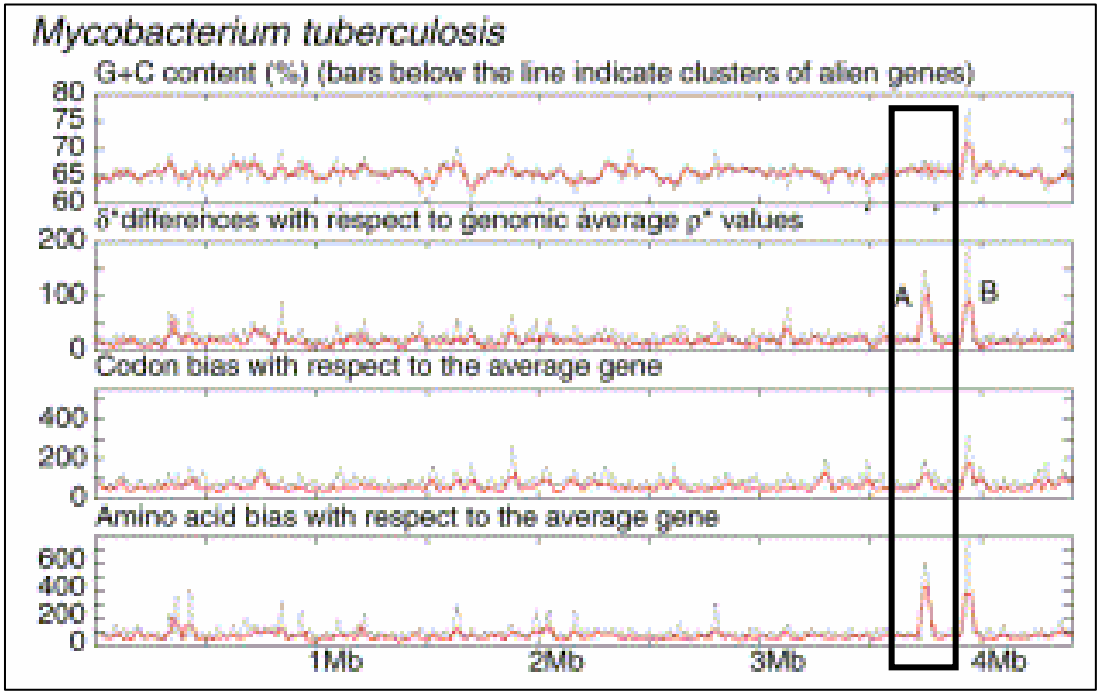
→ "atypical"

Διαφορετικοί δείκτες – διαφορετικά αποτελέσματα



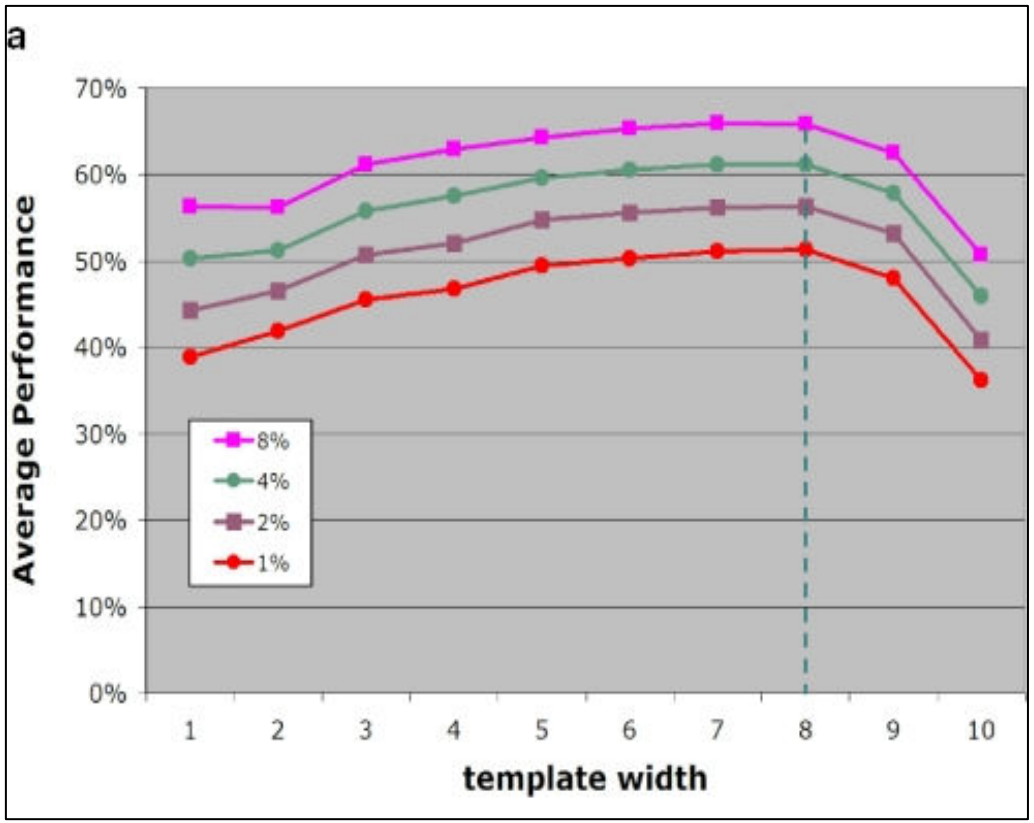
Karlin 2001

Διαφορετικοί δείκτες – διαφορετικά αποτελέσματα



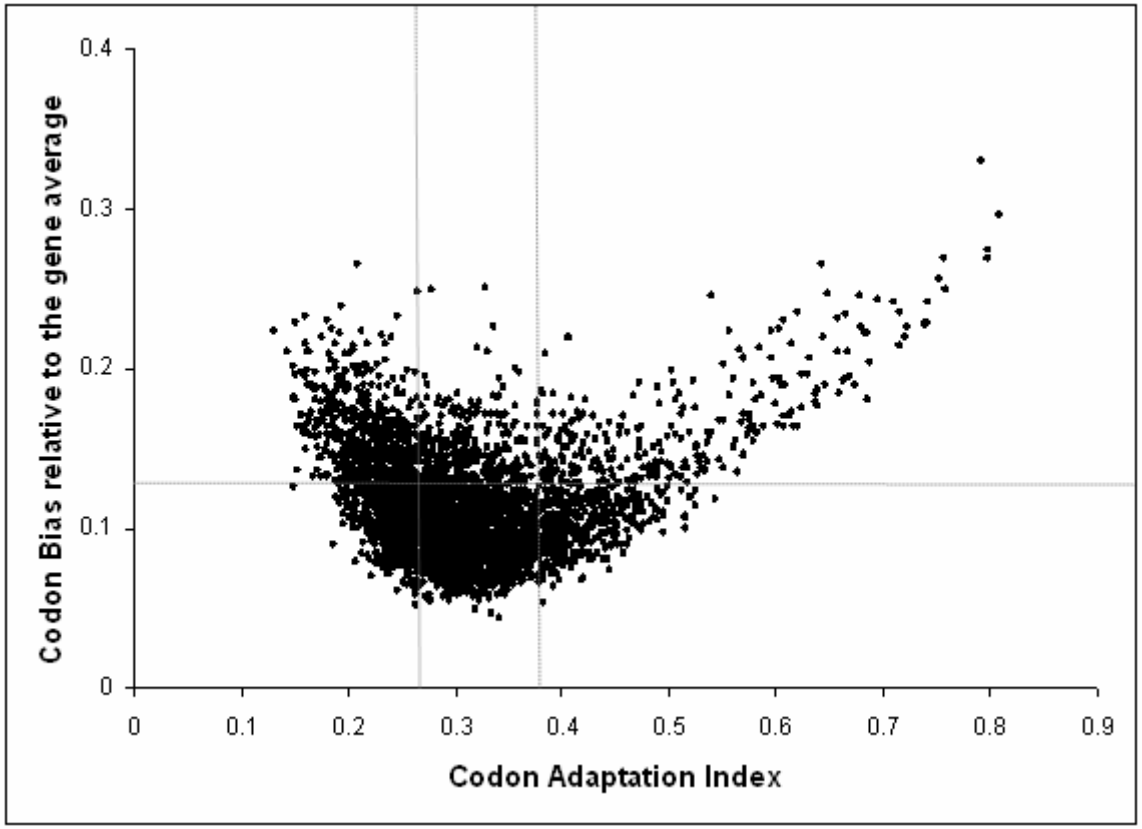
Karlin 2001

Μέγεθος και απόδοση δεικτών



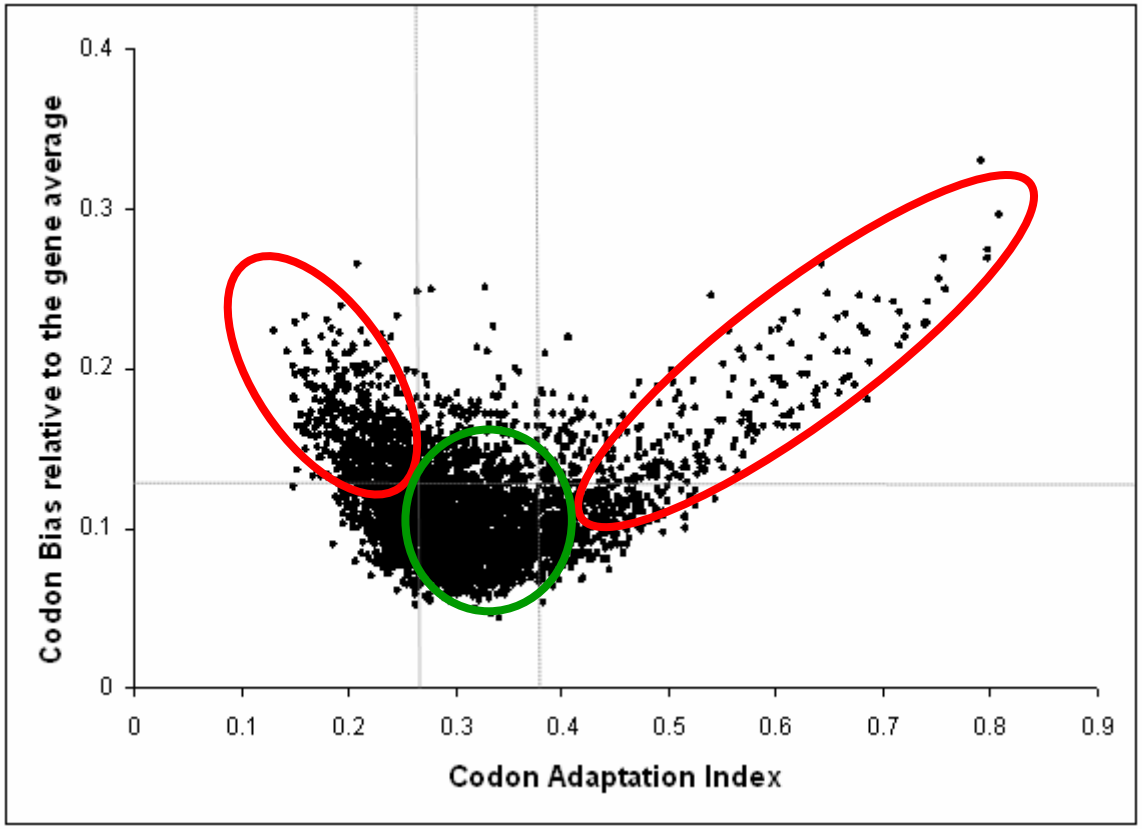
Tsirigos 2005

Συνδυάζοντας δείκτες: ”The rabbit-like plot”



Karlin 1998

Συνδυάζοντας δείκτες: ”The rabbit-like plot”



Karlin 1998

Interpolated Variable Order Motifs (IVOMs)

1. Γραμμικός συνδυασμός πιθανοτήτων «λέξεων» διαφορετικής τάξεως
2. Εξερευνεί δυναμικά μεγαλύτερο αλφάβητο «λέξεων» όταν είναι δυνατό → μεγαλύτερη ακρίβεια
3. Σε περίπτωση «φτωχού» περιεχομένου «λέξεων» υψηλής τάξεως εξερευνεί δυναμικά μικρότερο αλφάβητο

Vernikos 2006

Counts vs Dimensionality

4. Συχνότητα εμφάνισης (**counts**)
5. Μέγεθος αλφαβήτου διαφορετικής τάξης (**dimensionality**)

Παράδειγμα

3mer → counts **128**

5mer → counts **8**

} **"Ίδιο βαθμό αξιοπιστίας"**

Vernikos 2006

Interpolated Variable Order Motifs (IVOMs)

$$B = \{a, t, c, g\} \quad (1)$$

$$P_m(S) = \frac{A_m(S)}{N - k + 1} \quad (2)$$

$$W_m(S) = \frac{A_m(S) \cdot |B|^k}{\sum_{j=1}^{8} A_j(S) \cdot |B|^j} \quad (3)$$

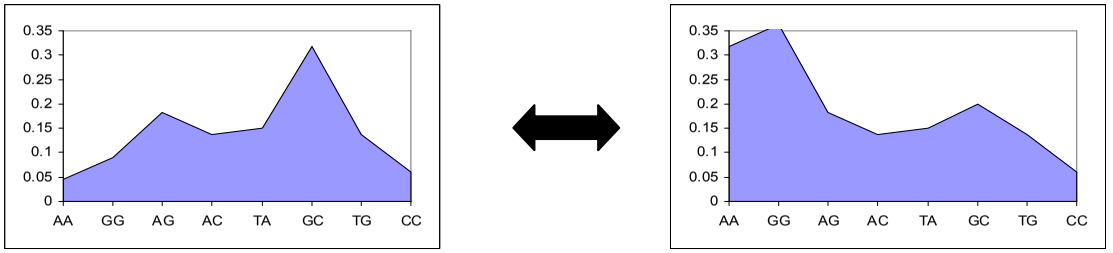
$$\mathrm{IVOM}(S,m) = \begin{cases} W_m(S) \cdot P_m(S) + [1 - W_m(S)] \cdot \mathrm{IVOM}(S, m_{2,|m|}) & if\ |m| \geq 2 \\ W_m(S) \cdot P_m(S) & if\ |m| = 1 \end{cases} \quad (4)$$

Vernikos 2006

Παράδειγμα

8mer	Interpolated *k*-mer	$A_m(S)$	$\|B\|^k$	$A_m(S) \times \|B\|^k$	$W_m(S)$
GCCAGCGC	GCCAGCGC	24	4^8	1572864	42.10
	CCAGCGC	49	4^7	802816	21.51
	CAGCGC	116	4^6	475136	12.73
	AGCGC	238	4^5	243712	6.53
	GCGC	738	4^4	188928	5.06
	CGC	2452	4^3	156928	4.20
	GC	9784	4^2	156544	4.19
	C	33854	4^1	135416	3.63
AAAACATG	AAAACATG	1	4^8	65536	7.38
	AAACATG	6	4^7	98304	11.08
	AACATG	26	4^6	106496	12.00
	ACATG	81	4^5	82944	9.35
	CATG	474	4^4	121344	13.67
	ATG	2110	4^3	135040	15.21
	TG	9243	4^2	147888	16.66
	G	32499	4^1	129996	14.65

Relative Entropy

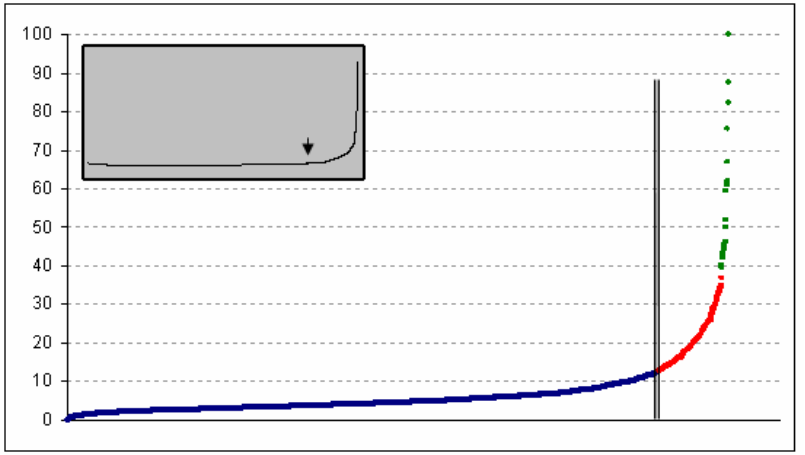


$$\overrightarrow{\mathbf{IVOM}(\boldsymbol{S},\boldsymbol{m})} = \{\mathrm{IVOM}(S,m) \mid m \in B^8\} \quad (5)$$

$$d_G(w) = \sum_{m \in B^8} \mathrm{IVOM}(w,m) \log_2 \frac{\mathrm{IVOM}(w,m)}{\mathrm{IVOM}(G,m)} \quad (6)$$

Vernikos 2006

Score Threshold



Algorithm: K-means clustering.

C: number of re-initializations.

F: objective function.

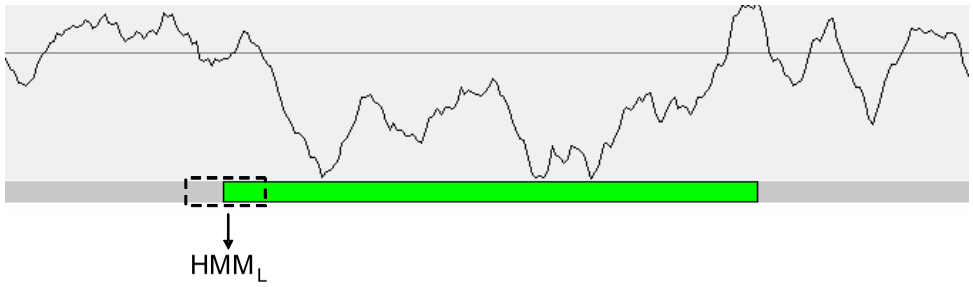
i = 1.

1. Determine the number of clusters, K = 3.
2. Initialize the value of the 3 centroids.
3. Assign each point to the cluster with the nearest centroid value.
4. When all points have been assigned to one of the 3 clusters, update the new centroid values.
5. Re-iterate steps 3 and 4 until the 3 centroids do not change; convergence criteria: $Last\_F_i - Current\_F_i < 0.1$.
6. **If** $i < C$ **do**
   if $F_i > F_{i\_max}$ then $F_{i\_max} = F_i$
   i++
   goto step 2 re-initializing the 3 centroids with different values.
7. Set the score threshold to the value where the transition from cluster 1 → 2 occurs, for the iteration with F_{i_max}.

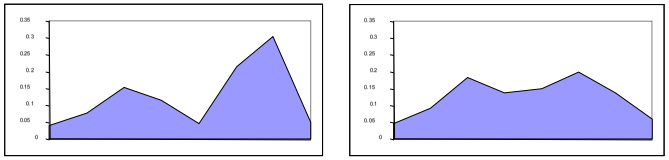
end

$$F = \sum_{j=1}^{K} \sum_{i=1}^{n} \left\| x_i - c_j \right\|^2 \quad (7)$$

Change-point

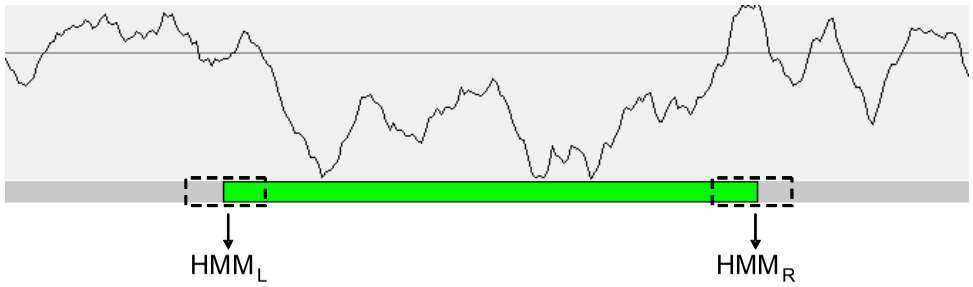


Emission_Prob

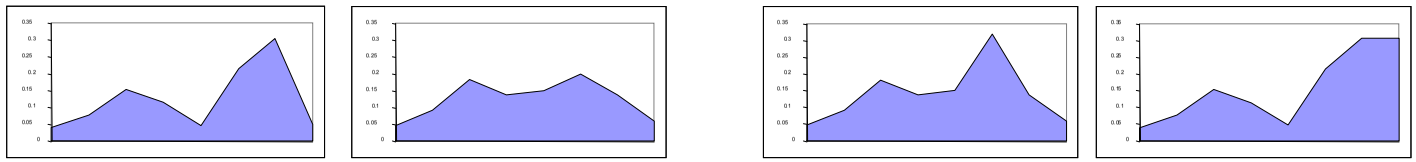


Left out ***Left in***

Change-point



Emission_Prob



Left out ***Left in*** ***Right in*** ***Right out***

Change-point

Algorithm: Change-point detection.

C: number of iterations

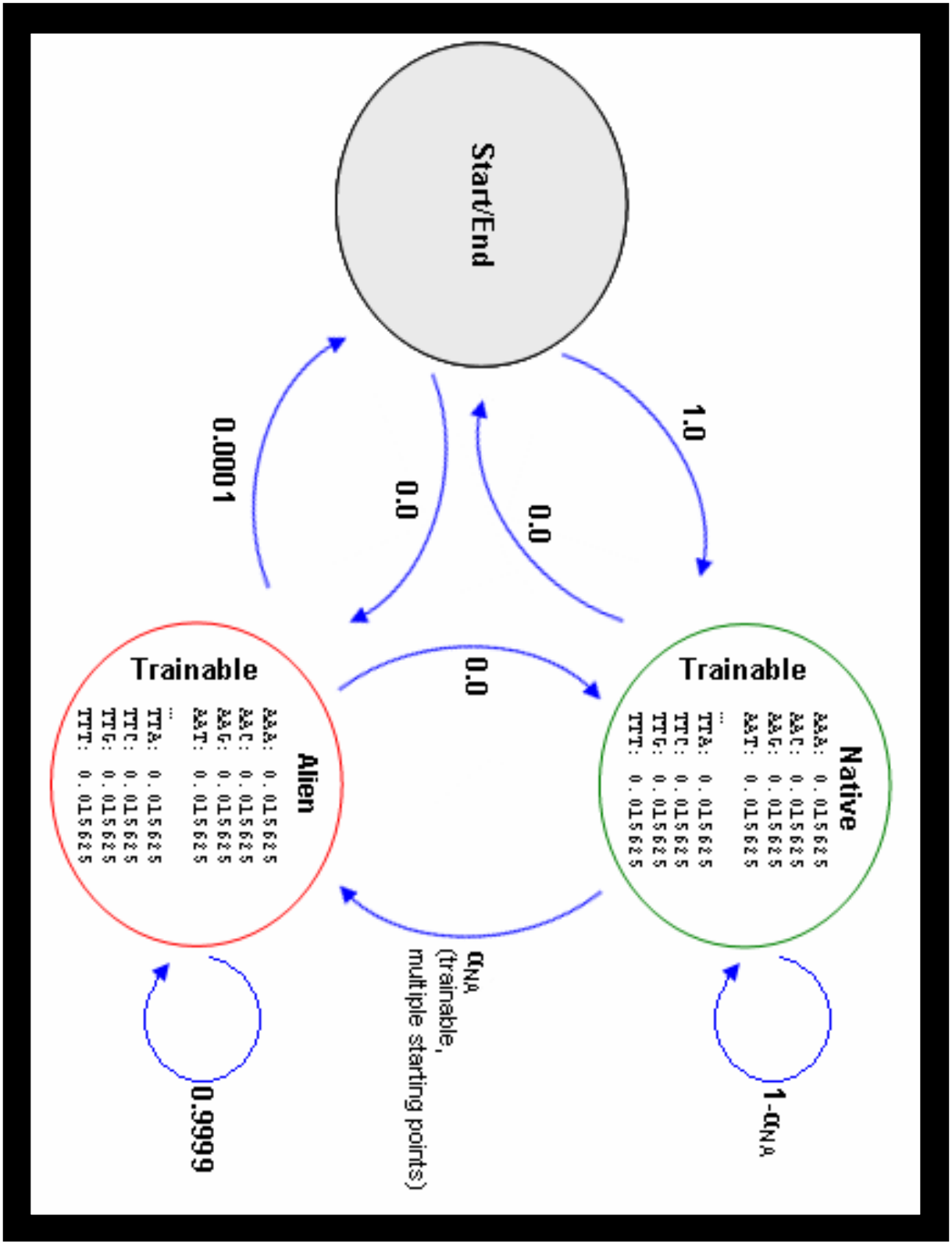
Init: i = 1;

a'_{NA}: initial starting point for a_{NA}

1. extend the predictions upstream and downstream
2. set initial model:
   - 2.1. *prior* distribution for the emission probabilities:
     - 2.1.1. *N* state: trainable second order uniform (e_N) distribution
     - 2.1.2. *A* state: trainable second order uniform (e_A) distribution
   - 2.2. *prior* transition probabilities:
     - 2.2.1. $a_{NA} = a'_{NA}$ (multiple starting points - trainable)
     - 2.2.2. $a_{AN} = 0$ (untrainable)
3. BW training until convergence:
   - 3.1. stopping criteria: LastScore - CurrentScore < 0.001
   - 3.2. updated-trained emission, transition probabilities
4. Viterbi: most probable path π^*, with score S_i
   - 4.1.1. **if** $S_i > S_{imax}$ then $S_{imax} = S_i$
5. **if** i < C **do**
   - 5.1.1. i++;
   - 5.1.2. new starting point a'_{NA}
   - 5.1.3. **goto** step 2
6. report the path π^* with S_{imax}
7. set predicted boundary = transition point in the path π^* with S_{imax}

end

Change-point … HMM architecture



Change-point … Viterbi

iteration	score S_i of path π^*	*prior* over α_{NA}	change-point (bp)
1	-9643.868804	500^{-1}	1720
2	-9643.868873	1000^{-1}	1720
3	-9627.033373	2000^{-1}	4870
4	-9627.033077	2500^{-1}	4870
5	-9627.033131	3000^{-1}	4870

Change-point - BioJava

```
import java.io.*;
import org.biojava.bio.symbol.*;
import org.biojava.bio.seq.*;
import org.biojava.bio.seq.io.*;
import org.biojava.bio.dp.*;
import org.biojava.bio.*;
import org.biojava.bio.seq.db.*;
import org.biojava.bio.seq.impl.*;
import org.biojava.bio.dist.*;
import org.biojava.utils.*;
import java.util.*;

class ChangepointLeft{

public static SymbolList seqL;
public static int order;
public static int flatOrRandom;
public static int trainOrUntrain;
public static Distribution dist;
public static int duration;
public static ModelTrainer mt;
public static int transition_point=0;
public static int count=0;

        //make alphabets
        static FiniteAlphabet DnaAlphabet = DNATools.getDNA();

        public static void main (String args[]) throws Exception{

        if(args.length != 5) {
        throw new Exception("Use: sequence.fa order.int flatD.bin trainableTrans.bin duration.int");
        }

        try{
```

Change-point - BioJava

```
File seqFile = new File(args[0]);
               order = Integer.parseInt(args[1]);
               flatOrRandom = Integer.parseInt(args[2]);
               trainOrUntrain = Integer.parseInt(args[3]);
               duration = Integer.parseInt(args[4]);

               if((flatOrRandom != 0) & (flatOrRandom != 1)) {
               throw new Exception("Use flatD.bin: only binary i.e. 0 or 1: . . 1/0 . .");
               }
               if((trainOrUntrain != 0) & (trainOrUntrain != 1)) {
               throw new Exception("Use trainableTrans.bin: only binary i.e. 0 or 1: . . . 1/0 .");
               }

               SymbolTokenization rParser = DnaAlphabet.getTokenization("token");

               SequenceBuilderFactory sbFact = new FastaDescriptionLineParser.Factory(SimpleSequenceBuilder.FACTORY);
               FastaFormat fFormat = new FastaFormat();

               SequenceIterator seqI = new StreamReader(new FileInputStream(seqFile),
                                       fFormat,
                                       rParser,
                                       sbFact);
               seqI.hasNext();

               Sequence seq2 = seqI.nextSequence();
               SequenceDB seqs = new HashSequenceDB();
               seqL = seq2;

               MarkovModel island = createModel();
               DP dp=DPFactory.DEFAULT.createDP(island);

              Sequence seq = new SimpleSequence(
                  SymbolListViews.orderNSymbolList(seq2, order),
                  null,
                  seq2.getName() + "-o" + order,
                  Annotation.EMPTY_ANNOTATION
               );

               seqs.addSequence(seq);
```

Change-point - BioJava

```
TrainingAlgorithm ta = new BaumWelchTrainer(dp);

        ta.train(
                seqs,
                0.01,
        new StoppingCriteria() {
                public boolean isTrainingComplete(TrainingAlgorithm ta) {

                try {
                // XmlMarkovModel.writeModel(ta.getDP().getModel(), System.out);
                //out2.write(ta.getCycle() + "\t" + ta.getCurrentScore() + "\n");
                }catch (Exception ex) {ex.printStackTrace();}
                //System.out.println(ta.getCycle() + "\t" + ta.getCurrentScore());
                //return (ta.getCycle() >=2);
                return Math.abs(ta.getLastScore() - ta.getCurrentScore()) < 0.001;
                }
        }
        );
```

Change-point - BioJava

```
//Viterbi

            SymbolList [] rl = {SymbolListViews.orderNSymbolList(seq2, order)};

            StatePath statePath = dp.viterbi(rl, ScoreType.PROBABILITY);

            for(int i = 0; i <= statePath.length() / 60; i++) {

                  for(int j = i*60; j < Math.min((i+1)*60, statePath.length()); j++) {
                        //System.out.print(statePath.symbolAt(StatePath.STATES, j+1).getName().charAt(0));
                        char state=statePath.symbolAt(StatePath.STATES, j+1).getName().charAt(0);
                       count++;
                        //it prints the states in binary mode for art user_graph
                        if(state == 'a'){
                        //out.write("0 1");
                        }
                        else{
                        transition_point=count;
                        //out.write("1 0");
                        }

                  }

            }

            System.out.print(transition_point + " " + statePath.getScore());

            }catch (Exception e) {
            e.printStackTrace();
      }
}
```

Change-point - BioJava

```
//creates the model
    public static MarkovModel createModel() {

        List l = Collections.nCopies(order, DNATools.getDNA());
        Alphabet alpha = AlphabetManager.getCrossProductAlphabet(l);

        int [] advance = { 1 };
        Distribution typicalD;
        Distribution atypicalD;

        try{

                //check if higher order; else normal dist
                if(order >1){
                typicalD = OrderNDistributionFactory.DEFAULT.createDistribution(alpha);
                atypicalD = OrderNDistributionFactory.DEFAULT.createDistribution(alpha);
                }
                else{
                typicalD = DistributionFactory.DEFAULT.createDistribution(alpha);
                atypicalD = DistributionFactory.DEFAULT.createDistribution(alpha);
                }


        }catch (Exception e){
        throw new AssertionFailure("Can't create distributions", e);
        }
```

Change-point - BioJava

```
EmissionState typicalS = new SimpleEmissionState("typical", Annotation.EMPTY_ANNOTATION, advance, typicalD);
EmissionState atypicalS = new SimpleEmissionState("atypical", Annotation.EMPTY_ANNOTATION, advance, atypicalD);

        SimpleMarkovModel island = new SimpleMarkovModel(1, alpha, "Island");

        try{
                island.addState(typicalS);
                island.addState(atypicalS);
        }catch (Exception e){
        throw new AssertionFailure("Can't add states to model", e);
        }

        //set up transitions between states
        try {
                island.createTransition(island.magicalState(),typicalS);
                island.createTransition(island.magicalState(),atypicalS);
                island.createTransition(typicalS,island.magicalState());
                island.createTransition(atypicalS,island.magicalState());
                island.createTransition(typicalS,atypicalS);
                island.createTransition(atypicalS,typicalS);
                island.createTransition(typicalS,typicalS);
                island.createTransition(atypicalS,atypicalS);
        }catch (Exception e){
        throw new AssertionFailure("Can't create transitions", e);
        }
```

Change-point - BioJava

```
//set up emission probabilities
      try {
            SymbolList highOrderSeq = SymbolListViews.orderNSymbolList (seqL, order);
            Hashtable symbol= new Hashtable();

           for (Iterator i = highOrderSeq.iterator(); i.hasNext(); ) {
                  Symbol sym = (Symbol) i.next();

                  if(!symbol.containsKey(sym)){
                  //uniform weights for atypical emmision probs
                  atypicalD.setWeight(sym,0.25);
                  typicalD.setWeight(sym, 0.25);
                  symbol.put(sym, new Integer(1));
                  }
            }

            if(flatOrRandom == 0){
            //it randomizes the atypical emission probs
            DistributionTools.randomizeDistribution(atypicalD);
            DistributionTools.randomizeDistribution(typicalD);
            }

      }catch (Exception e) {
      throw new AssertionFailure("Can't set emission probabilities", e);
      }
```

Change-point - BioJava

```
//set up transition scores.
      try {
            {
            //if user option =1 then it trains ; if 0 then untrained
            if(trainOrUntrain ==0){
            //it keeps the transition probs untrainable
            dist = new UntrainableDistribution (island.transitionsFrom(island.magicalState()));
            }
            else{
            dist = island.getWeights(island.magicalState());
            }
            dist.setWeight(typicalS,          1.0);
            //since it will always start at start at state typicalS
            dist.setWeight(atypicalS,         0.0);
            island.setWeights(island.magicalState(), dist);
            }

            {
            // always trainable
            dist = island.getWeights(typicalS);
            float T_A = (float)1/duration;
            float T_T = (float)1-T_A;
            //1/region = 1/7500
            dist.setWeight(atypicalS,           T_A);
            //1-1/7500
            dist.setWeight(typicalS,            T_T);
            //zero since it will always end at atypical
            dist.setWeight(island.magicalState(), 0.0);
            island.setWeights(typicalS, dist);
            }
```

Change-point - BioJava

```
            {
            // always trainable
            dist = island.getWeights(typicalS);
            float T_A = (float)1/duration;
            float T_T = (float)1-T_A;
            //1/region = 1/7500
            dist.setWeight(atypicalS,           T_A);
            //1-1/7500
            dist.setWeight(typicalS,            T_T);
            //zero since it will always end at atypical
            dist.setWeight(island.magicalState(), 0.0);
            island.setWeights(typicalS, dist);
            }

            {
            //always untrainable
            dist = new UntrainableDistribution (island.transitionsFrom(atypicalS));
            //when it changes it persists for ever.
            dist.setWeight(typicalS,            0.0000000000000000000000000000001);
            dist.setWeight(atypicalS,           0.9999);
            //it was 0.0001  but it throwed NaNs
            dist.setWeight(island.magicalState(), 0.00009999999999999999999999999999);
            island.setWeights(atypicalS, dist);
            }
      }catch (Exception e) {
      throw new AssertionFailure("Can't set transition probabilities", e);
      }

   return island;
  }

}
```

Viterbi … online DEMO

Source: http://www.cs.umb.edu/~srevilak/viterbi/

Target sequence: "ATGCATGCATGGGGCC"

Alphabet: [A, T, G ,C]

# of states: 2

Transition: There is 0.2 probability of switching from state1 to state2. There is 0.9 probability of switching from state2 to state1.

Emission: In state1 the frequency of observing A, T, G, C is their expected frequencies assuming a zero-th order alphabet. In state2 $P_A = P_T = 0.1$ and $P_G = P_C$.

Initial probabilities: The probability of the model starting in state1 is 0.6.

Deliverables:

A. Build the model.

B. Run the prediction.

C. Record the most probable state path.

D. Design the HMM architecture.