ШИНЖЛЭХ УХААН ТЕХНОЛОГИЙН ИХ СУРГУУЛЬ
Мэдээлэл, Холбооны Технологийн Сургууль

F.CS213 Биоалгоритм

Probabilistic Motifs and Stochastic Algorithms

Магадлалт мотиф ба Санамсаргүй алгоритмууд

Лекц 10

Багш ***Ганбаатарын ГАНБАТ.*** *Өрөө* ***#304,*** *ganbatg@must.edu.mn,* ***99999467***

Лекцийн агуулга

- Магадлалт мотиф
  - Алгоритм
  - Визуйл дүрслэл
- Стохастик алгоритм
  - Expectation-Maximization
  - Алгоритм
- Гиббсийн түүвэрлэлтийн арга
  - Алгоритм
- BioPython магадлалт мотиф

Магадлалт мотиф

Магадлалт мотифыг ерөнхийдөө
Магадлалт жингийн матриц (Probabilistic Weight Matrices – PWM) -аар дүрсэлдэг.

- ***Магадлалт жингийн матриц*** *(Templates / Profiles)*
  - *Мөр:* Тэмдэгт (нуклеотид/амин хүчлүүд),
  - *Багана:* Мотиф дээрх байрлал.
  - P_{ij} нь P мотифын j байрлал дээр илрэх i нуклеотидын магадлал.

	1	2	…	N
A	P_{A1}	P_{A2}	…	P_{An}
T	P_{T1}	P_{T2}	…	P_{Tn}
C	P_{C1}	P_{C2}	…	P_{Cn}
G	P_{G1}	P_{G2}	…	P_{Gn}

N хэмжээтэй ДНХ-ийн мотифын PWM.

- N урттай $S = S_1 S_2 \dots S_N$ дарааллын P PWM-ээр танигдах магадлал:

$$p(S, P) = \prod_{i=1}^{N} P(S_i, i)$$

- Давтамжыг оноо руу хөрвүүлсэн ***Байрлалд суурилсан онооны матриц (Position Specific Scoring Matrices - PSSM)*** үүсгэнэ.
  - Элемент: S-ийн P-ээр танигдах магадлалын оноо:

$$score(S, P) = \sum_{i=1}^{N} log P(S_i, i)$$

Илрүүлэх тооцоолол

GATCAT
GATGAT
GAAGAA
CAAGAC
AAACTT
GGACCT
GCAAAG
CTGCAT

	1	2	3	4	5	6
A	1/8	5/8	5/8	1/8	6/8	1/8
T	0	1/8	3/8	0	1/8	5/8
C	1/8	1/8	0	4/8	1/8	1/8
G	5/8	1/8	1/8	3/8	0	1/8

- 8 дарааллын багц дээрх PWM.
- $a = GATCAT$ дарааллын P мотифт танигдах магадлалыг түүний бүх байрлалын магадлалыг үржүүлнэ.
  - $p(GATCAT|P) = \frac{5}{8} \times \frac{7}{8} \times \frac{3}{8} \times \frac{4}{8} \times \frac{6}{8} \times \frac{5}{8} = 0.03433$

S = GCGGATCATCAA

Scanned Sequence	Probability calculation	p(a|P)
GCGGATCATCAA	5/8 x 1/8 x 1/8 x 3/8 x 6/8 x 1/8	3.4×10^{-4}
G**CGGATC**ATCAA	1/8 x 1/8 x 1/8 x 5/8 x 1/8 x 1/8	3.8×10^{-5}
GC**GGATCA**TCAA	5/8 x 1/8 x 5/8 x 0 x 1/8 x 1/8	0
GCG**GATCAT**CAA	5/8 x 5/8 x 3/8 x 4/8 x 6/8 x 5/8	**0.03433**
GCGG**ATCATC**AA	1/8 x 1/8 x 0 x 1/8 x 1/8 x 1/8	0
GCGGA**TCATCA**A	0 x 1/8 x 5/8 x 0 x 1/8 x 1/8	0
GCGGAT**CATCAA**	1/8 x 5/8 x 3/8 x 4/8 x 6/8 x 1/8	1.4×10^{-3}

- S дараалал болон P PWM өгөгдсөн байг
  - S дээрх P-ээр танигдах хамгийн өндөр магадлалтай N урттай дэд дарааллыг тооцоолно.
  - N урттай шилжигч цонх ашиглан S-г сканнердана.

Алгоритм

```
from MySeq import MySeq
seq1 = MySeq("AAAGTT")
seq2 = MySeq("CACGTG")
seq3 = MySeq("TTGGGT")
seq4 = MySeq("GACCGT")
seq5 = MySeq("AACCAT")
seq6 = MySeq("AACCCT")
seq7 = MySeq("AAACCT")
seq8 = MySeq("GAACCT")
```

- Эхлээд мотиф үүсгэх 8 дараалллыг авч тоо ширхэг болон давтамжийн матрицыг дүрслэнэ.
- Дундаж болон максласан дундаж мотифыг олно.
- Магадлалын тооцооллыг хэд хэдэн оролтын дараалал дээр гүйцэтгэнэ.

```
lseqs = [seq1, seq2, seq3, seq4, seq5, seq6, seq7, seq8]
motifs = MyMotifs(lseqs)
print ("Counts matrix")
print_matrix (motifs.counts)
print ("PWM")
print_matrix (motifs.pwm)
print ("Sequence alphabet")
print(motifs.alphabet)

[print(s) for s in lseqs]
print ("Consensus sequence")
print(motifs.consensus())
print ("Masked Consensus sequence")
print(motifs.masked_consensus())

print(motifs.probability_sequence("AAACCT"))
print(motifs.probability_sequence("ATACAG"))
print(motifs.most_probable_sequence("CTATAAACCTTACATC"))
```

PWM үүсгэх

```
class MyMotifs:
    """Class to handle Probabilistic Weighted Matrix"""
    def __init__(self, seqs = [], pwm = [], alphabet = None):
        if seqs:
            self.size = len(seqs[0])
            self.seqs = seqs # objet from class MySeq
            self.alphabet = seqs[0].alphabet()
            self.do_counts()
            self.create_pwm()
        else:
            self.pwm = pwm
            self.size = len(pwm[0])
            self.alphabet = alphabe
```

do_counts болон **create_pwm** функцуудээр үндсэн аттрибутуудыг байгуулна.

I. PWM загварт ашигласан дарааллууд;
II. Дарааллын нийт тоо;
III. Цагаан толгой;
IV. Тоо ширхэгийн матриц
V. Мотифын байрлал бүр дээрх тэмдэгтийн давтамжын матриц.

```
def do_counts(self):
    self.counts = create_matrix_zeros(len(self.alphabet), self.size)
    for s in self.seqs:
        for i in range(self.size):
            lin = self.alphabet.index(s[i])
            self.counts[lin][i] += 1

def create_pwm(self):
    if self.counts == None: self.do_counts()
    self.pwm = create_matrix_zeros(len(self.alphabet), self.size)
    for i in range(len(self.alphabet)):
        for j in range(self.size):
            self.pwm[i][j] = float(self.counts[i][j]) / len(self.seqs)
```

MyMotifs класс

1. PWM үүсгэх,
2. Мотифын детерминистик дүрслэлүүдийг гаргаж авах,
3. Дараалал дээр мотиф илрэх магадлалыг тодорхойлох

Мотифын детерминистик дүрслэлүүд

```
def masked_consensus(self):
    """ returns the sequence motif obtained with the symbol that
occurs in at least 50% of the input sequences"""
    res = ""
    for j in range(self.size):
        maxcol = self.counts[0][j]
        maxcoli = 0
        for i in range(1, len(self.alphabet) ):
            if self.counts[i][j] > maxcol:
                maxcol = self.counts[i][j]
                maxcoli = i
        if maxcol > len(self.seqs) / 2:
            res += self.alphabet[maxcoli]
        else:
            res += "-"
    return res
```

```
def consensus(self):
    """ returns the sequence motif obtained with the most
frequent symbol at each position of the motif"""
    res = ""
    for j in range(self.size):
        maxcol = self.counts[0][j]
        maxcoli = 0
        for i in range(1, len(self.alphabet) ):
            if self.counts[i][j] > maxcol:
                maxcol = self.counts[i][j]
                maxcoli = i
        res += self.alphabet[maxcoli]
    return res
```

Мотифын байрлал бүрт хамгийн их илэрсэн тэмдэгт бүхий дундаж дүрслэлийг PWM-ээс үүсгэнэ.

- consensus: Мотифын байрлал бүрийг сканнердаж, байрлал бүрт хамгийн олон давтамжтай тэмдэгтийг сонгоно.
- masked_consensus: Төстэй байдлаар ажилладаг боловч байрлал бүрийн давтамж нь 50% -иас бага бол "-", үгүй бол цагаан толгойн тэмдэгтийг гаргадаг.

Дараалал дээрх мотиф магадлалыг тооцоолох

```
def probability_sequence(self, seq):
    res = 1.0
    for i in range(self.size):
        lin = self.alphabet.index(seq[i])
        res *= self.pwm[lin][i]
    return res

def probability_all_positions(self, seq):
    maximum = -1.0
    maxind = -1
    for k in range(len(seq)-self.size):
        p = self.probability_sequence(seq[k:k+ self.size])
        if(p > maximum):
            maximum = p
            maxind = k
    return maxind

def create_motif(self, seqs):
    from MySeq import MySeq
    l = []
    for s in seqs:
        ind = self.most_probable_sequence(s.seq)
        subseq = MySeq ( s[ind:(ind+self.size)], s. get_seq_biotype() )
        l.append(subseq)
    return MyMotifs(l)
```

- probability_sequence болон probability_all_positions
  - Тухайн байрлалд илрэх бүх тэмдэгтийн магадлалуулын үржвэр.
  - Урт дараалал даар илрэх N урттай дэд дараалал бүр дээр мотиф илрэх магадлал.
  - Эхний индексээс $|S| - N + 1$ байрлалд хүртэл скан хийнэ.
  - Дэд дараалал бүрийн магадлалыг жагсаалтад хадгална.

- most_probable_sequence
  - Мотифт тохирох хамгийн өндөр магадлалтай дэд дарааллыг олно.
  - Энэ нь мотифыг шинэчлэх, сайжруулах боломжийг олгоно.

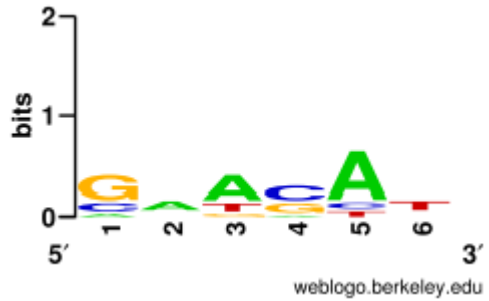
- Create_motif
  - Оролтын дарааллуудыг сканнердаж, MyMotifs классын объектыг буцаана
  - Шинэ мотифыг байгуулах хамгийн боломжит дэд дарааллуудыг сонгоно.

Визуал дүрслэл

- PWM загвар нь дарааллын байрлал дээрх тэмдэгт бүрийн давтамжийг тооцон мотифын магадлалыг дүрсэлдэг.
  - Ороолтын дарааллууд доторх мотифуудын шинэ илрэлүүдийн хайхад PWM-ийг ашиглах
  - Өндөр оноотой илрэлүүдийг мотиф загварт нэгтгэн сайжруулж болно.
- PWM загварт тохирох дэд дарааллын тоо харьцангуй бага бол зарим тэмдэгт зарим байрлалд тооцогдохгүй байх боломжтой, ө.х давтамж нь 0 байж болно.
  - Энэ нь үүсэх мотифын магадлал 0 байх нөхцөлд хүргэнэ.
  - Тиймээс, тоо ширхэгийн нөлөөг ихэсгэхийн тулд PWM утгууд дээр *псевдо-тооолт (pseudo-count)*-ыг нэмдэг.
- PWM-ийг дүрслэх нэг нийтлэг арга бол *Дарааллын лого*
  - 1990 онд Шнайдер, Стефенс нар танилцуулсан.
  - Мотифын байрлалын хадгалалтын түвшинтэй пропорционал өндөртэй үсгүүдийн стекээр дүрсэлдэг.
  - *Weblogo* хэрэгслэл нь PWM-ийн “гоё” дүрслэлийг хялбархан үүсгэнэ.

$$I_i = 2 + \sum_b P_{b,i} \times log_2 P_{b,i}$$

Дараалсан логоны b тэмдэгт бүрийн i байрлал дээрх мэдээллийг тооцоолох томьёо.



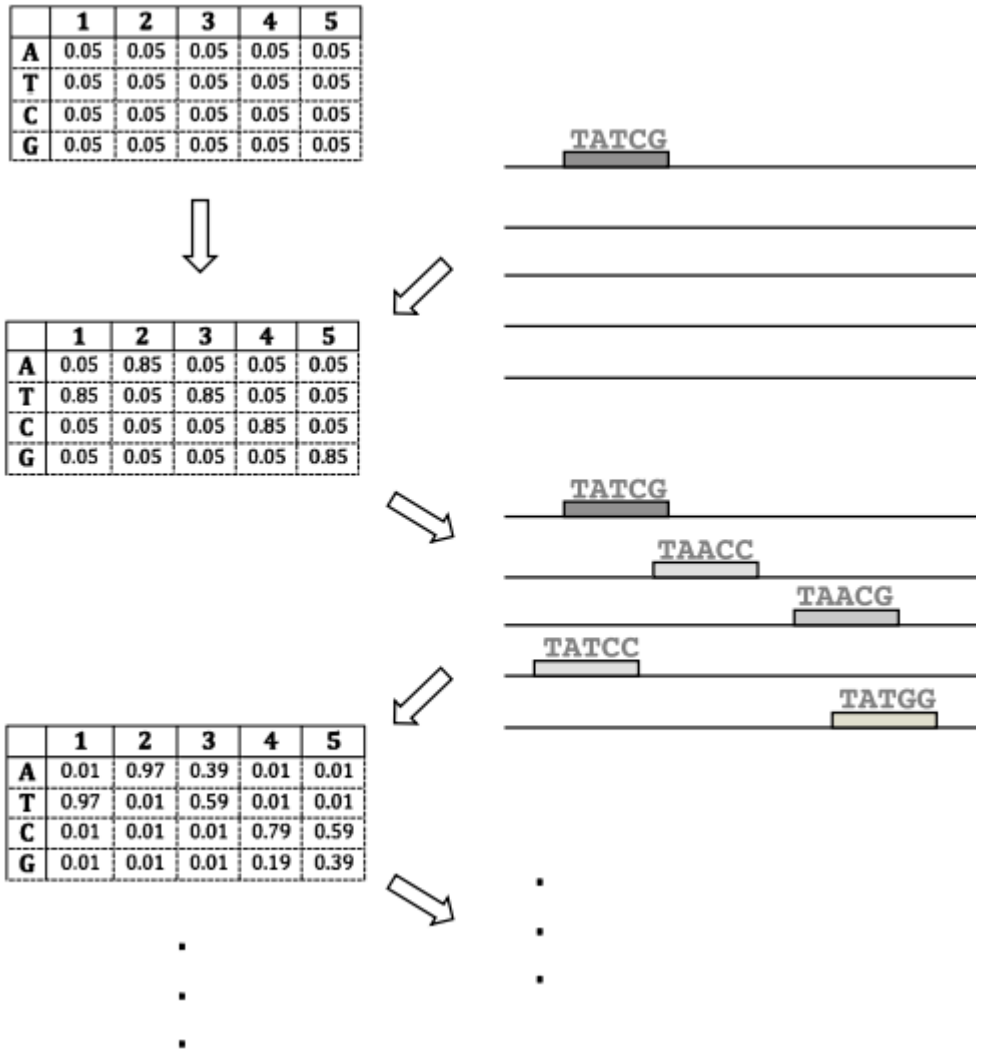
8 дэд дарааллын Weblogo

Стохастик алгоритм

- Оролтын дарааллууд дээрх мотифын хамгийн сайн байршилд зоруулагдсан сайн таамаглалыг илэрхийлсэн дэд дарааллын багц өгөгдсөн байг.
  - Эдгээр дэд дарааллаас мотифын онцлогийг агуулсан PWM-ийг гаргаж авна.
  - Гэхдээ энэ мэдээлэл нь ихэндээ урьдчилж мэдэгддэггүй.
  - Тиймээс мотиф илрүүлэх аргыг хэрэгжүүлсэн байх шаардлагатай.
- Мотифыг илрүүлэх асуудлын тодорхойлолтоос харахад
  - *Ролт:* L урттай t ширхэг дараалал бүхий D олонлог, Мотифын урт N.
  - *Гаралт:* $score(s, D)$ функцийн утгыг хамгийн их байлгах D доторх дэд дарааллуудын анхдагч байрлалуудыг агуулсан t урттай s вектор (мотифын чанарыг хамгийн сайн байлгах зорилготой функц).
- Детерминистик болон магадлалт мотифын хувьд тавигдах нөхцөл нь ижил ч хайлтын боломжийг шүүх арга нь өөр.
  - Exhaustive хайлтанд тооллогын аргыг хэрэглэх, зарим хьюристик хандлага нь уг хайлтын үр ашгийг нэмэгдүүлдэг.
  - Мотифын хамгийн боломжит шийдлүүдийг олоход чиглэдэг
  - Хайлтын тоологдом орон зайг баримжаалахын тулд загварын тусгай хийсвэрлэлийг шаарддаг
  - Тиймээс бүдэг (илүү гажсан) мотифыг орхиж болно.
- *Expectation-Maximization (EM)* дээр суурилсан алгоритмууд нь хайлтын орон зайг шалган нэвтрэх болон мотифыг оновчлох аргаар уг асуудлыг шийдэх боломжтой.
  - Давталтын явцад хамгийн сайн дүрслэгдсэн дэд дарааллуудыг тодорхойлон мотифын загварыг сайжруулах, шинэчлэхэд хэрэглэдэг.

СТОХАСТИК АЛГОРИТМ Expectation-Maximization

- EM алгоритм нь L урттай мотифын хувьд :
  - Оролтын дарааллууд дээрх L урттай дэд дараалал бүрийг сканнердана
  - Тухайн дэд дарааллын тэмдэгтүүдийг маш бага суурь давтамжтай байх PWM-ийг эхлүүлнэ
  - Эдгээр дэд дараалал бүрийн хувьд мотифоор үүсгэгдсэн байх магадлалыг тооцоолдог.
  - Ингэсэн нь оролтын дарааллууд дээрх санамсаргүй загвараас илүү байдаг
  - Сонгосон бүх дэд дарааллын давтамжийг дундажлан мотифыг сайжруулахад ашигладаг.
  - Онооны өсөлт нь 0 (хамгийн бага) бол процедур зогсдог.
- Энэ арга нь ерөнхийдөө оновчтой шийдэлд ойртдог.
  - Мотифын хамгийн сайн тохиолдлын дараалал дээр маск хэрэглэн (тэмдэгтүүдийг солих), давтан ажиллуулах замаар бусад мотифыг илрүүлнэ.
  - *The MEME Suite: Motif-based sequence analysis tools.*
- Зурагт мотифыг эхлүүлэх EM процессийг үзүүлсэн
  - Нуклеотидын жигд тархалтын 20%-ийг авч үзнэ.
  - Мотифын загвар нь шинэ сайн тохиодлыг тооцсон давталтаар сайжирдаг.



Алгоритм

- EM алгоритм нь оролтын дээрх L урттай бүх дэд дарааллыг сканнердсанаар детерминистик оновчлолыг гүйцэтгэдэг.
  - Ерөнхийдөө оновчтой шийдэд ойртдог ч энэ нь бүх анхдагч дад дарааллуудыг exhaustive тоочихыг шаарддаг.
  - Хайлтанд стохастик компонентийг оруулснаар хайлтын процессыг оновчтой болгож болно.
  - Жнь, мотифыг үүсгэхэд ашигласан анхдагч дэд дарааллуудыг санамсаргүй сонгох.

- Стохастик хьюристик хайлт дээр суурилсан алгоритмын үндсэн алхмууд:

1. Оролтын D дарааллын дагуу $s = (s_1, \ldots, s_t)$ анхны байрлалуудыг санамсаргүй байдлаар сонгож эхэлнэ.
2. 1-р алхамд үүсгэсэн дарааллаас P PWM профайл үүсгэгдэнэ.
3. D-д хамгийн их магадлалтай n дэд дарааллыг хайхад P мотифыг хэрэглэнэ. n-ийн дагуу s вектор дээрх шинэ анхдагч байрлалуудад өөрчлөлт орсон байна.
4. 3-р алхамд тооцсон байрлалуудыг ашиглан шинэ P PWM профайлыг үүсгэсэн. 3 ба 4-р алхмуудыг $score(s, D)$ өсөхөө больтол давтана.

СТОХАСТИК АЛГОРИТМ > EM Хэрэгжүүлэлт

```
def create_motif_from_indexes(self, indexes):
    pseqs = []
    for i,ind in enumerate(indexes):
        pseqs.append( MySeq(self.seqs[i][ind:(ind+self.motif_size)],self.seqs[i].get_seq_biotype()) )
    return MyMotifs(pseqs)
def heuristic_stochastic (self):
        from random import randint
        s = [0]* len(self.seqs)
        for k in range(len(s)):
            s[k] = randint(0, self.seq_size(k)- self.motif_size)
        motif = self.create_motif_from_indexes(s)
        motif.create_pwm()
        sc = self.score_multiplicative(s)
        bestsol = s
        improve = True
        while(improve):
            for k in range(len(s)):
                s[k] = motif.most_probable_sequence(self.seqs[k])
            if self.score_multiplicative(s) > sc:
                sc = self.score_multiplicative(s)
                bestsol = s
                motif = self.create_motif_from_indexes(s)
                motif.create_pwm()
            else: improve = False
        return bestsol
```

MotifFinding класс (Бүлэг 10).

create_motif_from_indexes

- Мотифын загварыг бүтээхэд ашигласан дэд дарааллын анхдагч байрлалыг агуулсан индексүүдийн жагсаалтыг авч MyMotifs төрлийн магадлалын мотиф үүсгэдэг

heuristic_stochastic

- Ороолтын дарааллууд дээрх санамсаргүй сонгосон анхдагч байрлалууд болон харгалзах PWM (үржүүлсэн оноо) –ээр мотиф үүсгэдэг.

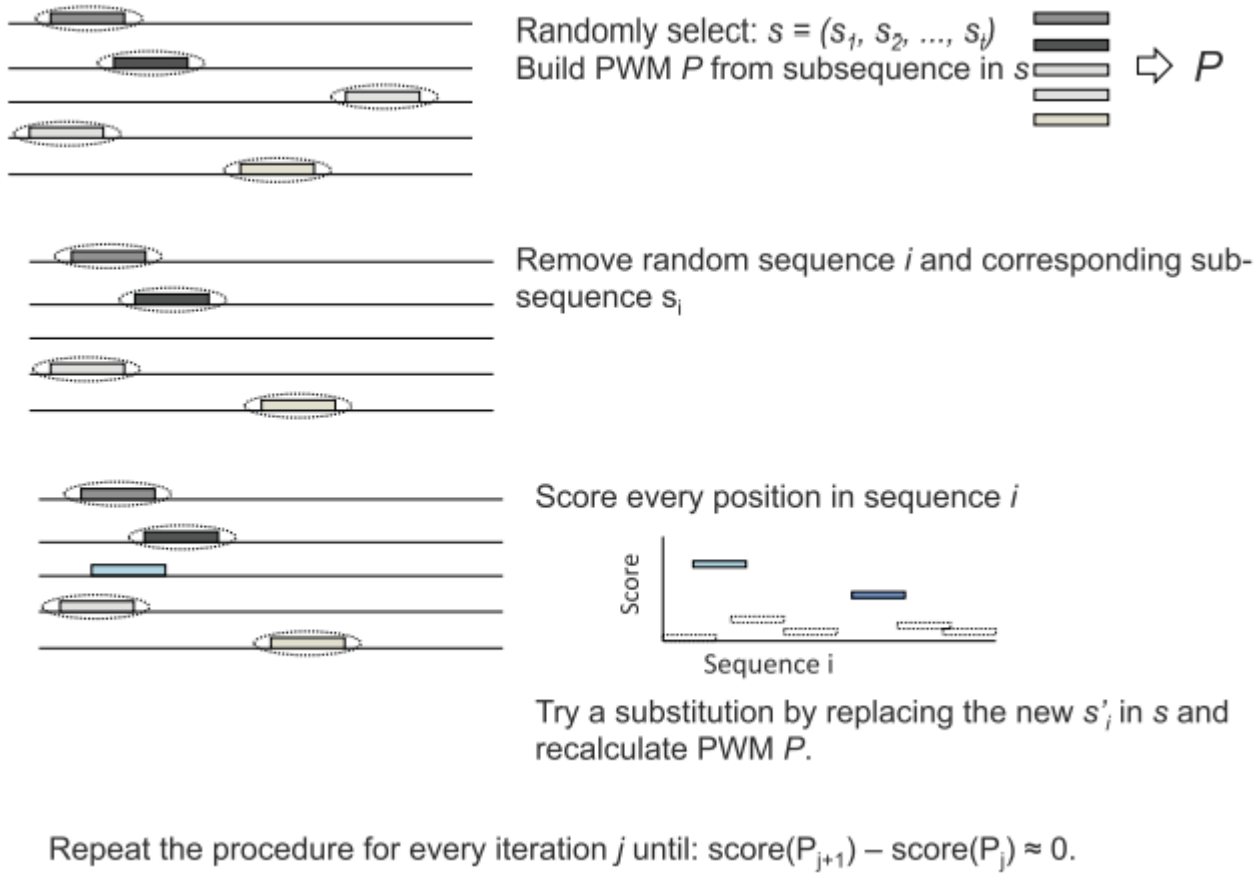
Гиббсын түүвэр

- Лоуренс нар: Гиббсын түүвэр (Gibbs sampling)
  - Хэд хэдэн мотиф олох алгоритмд хьюристик аргачлал болгон ашигладаг
  - Хамгийн сайн шийдлийг баталгаажуулдаггүй ч практикт давталтаар ажиллуулахад сайн үр дүнтэй.
- Мотиф загварыг санамсаргүй байдлаар сонгосон дэд дарааллаар эхлүүлж, улмаар анхны загвартай харьцуулсан оноог тооцдог.
- Давталт бүрийн хувьд
  - Мотифын аль нэг илрэлийг шинэчлэх эсэх магадлалаар локал хайлт хийдэг.
  - Өгөгдсөн оролтын дарааллуудаас мотифыг загварчлахад ашигласан дэд дарааллыг арилгана.
  - Сүүлд нь өөр дэд дарааллаар солихыг оролдох, Мотифын оноог тооцоолох, Сайжруулалтыг хадгалах эсэхийг шийддэх.

Схемчилсэн дүрслэл

Гиббсийн түүвэрийн аргын алхмууд.

1. Оролтын D дарааллуудаас $s = (s_1, \ldots, s_t)$ анхдагч байрлалуудыг санамсаргүйгээр сонгоно.
2. D-ээс i гэсэн дарааллыг санамсаргүй байдлаар сонгоно.
3. 2-р алхам дээр сонгосон дэд дарааллыг s-дарааллаас хасч P PWM үүсгэнэ.
4. i дарааллын p байрлал бүрийн хувьд, P-ээр үүсгэгдэж буй L урттай p байрлалаас эхэлсэн байх дэд дарааллын магадлалыг тооцоолно.
5. 4-р алхамд тооцсон магадлалын дагуу p-г стохастик аргаар сонгоно.
6. P мотиф дэх s-ийн оноо улам сайжрах хүртэл 2-5-р алхамуудыг давтана.



АНХААРАЛ ТАВЬСАНД БАЯРЛАЛАА