Biopython ilə BLAST: BLAST Axtarışlarını Məlumat Elmi Layihələrinə İnteqrasiya Etmək

BLAST (Basic Local Alignment Search Tool), əsas yerli hizalama axtarış aləti deməkdir və bioloji ardıcıllıqların – məsələn, müxtəlif zülalların amin turşusu ardıcıllıqları və ya DNT-nin nukleotid ardıcıllıqları – müqayisəsi üçün geniş istifadə olunan güclü bir bioinformatika alətidir. BLAST-ın əsas məqsədi verilənlər bazasında sorğu (query) ardıcıllığına oxşar olan ardıcıllıqları müəyyən etməkdir. Bu isə müxtəlif ardıcıllıqlar arasında funksional və təkamül əlaqələri barədə məlumat əldə etməyə imkan verir.

BLAST Növləri:

BLAST-ın müxtəlif növləri var və hər biri fərqli növ bioloji verilənlər bazaları üçün istifadə olunur. Biopython vasitəsilə bu növləri asanlıqla istifadə edə bilərsiniz. BLAST növləri:

1.BLASTP:

Zülal ardıcıllığını digər zülal verilənlər bazası ilə müqayisə edir.

Biopython Modulu: Bio.Blast.NCBIXML (XML nəticələri ilə işləmək üçün)

2.BLASTN:

Nükleotid ardıcıllığını nükleotid verilənlər bazası ilə müqayisə edir.

Biopython Modulu: Bio.Blast.NCBIXML

3.BLASTX:

Nükleotid ardıcıllığının altı fərqli çevirisini (frame) zülal verilənlər bazasında axtarır.

Biopython Modulu: Bio.Blast.NCBIXML

4.TBLASTN:

Zülal ardıcıllığını nükleotid verilənlər bazasına qarşı axtarır, amma çevirilmiş nükleotidlər istifadə olunur.

Biopython Modulu: Bio.Blast.NCBIXML

5.TBLASTX:

Nükleotid ardıcıllığının altı fərqli çevirisini nükleotid verilənlər bazasına qarşı axtarır.

Biopython Modulu: Bio.Blast.NCBIXML

2. Biopython ilə BLAST Axtarışı:

Biopython, BLAST axtarışlarını asanlaşdırmaq üçün NCBIWWW.qblast() funksiyasını təklif edir. Bu funksiya, BLAST serverinə sorğu göndərir, nəticələri alır və onları təhlil etmək üçün lazım olan məlumatları verir.

Əsas Modullar:

•Bio.Blast.NCBIXML: BLAST nəticələrini XML formatında oxumaq və təhlil etmək üçün istifadə olunur.

•NCBIWWW.qblast(): Bu funksiya BLAST axtarışını NCBI serverinə göndərir və nəticələri alır.

Məsələn BLASTP Axtarışı:

```
[*]: from Bio.Blast import NCBIWWW

# BLASTP axtarışı üçün bir zülal ardıcıllığı
sequence = "MTAIAIQVVMDN..."

# NCBI serverinə BLASTP sorğusu göndərmək
result_handle = NCBIWWW.qblast("blastp", "nr", sequence)

# Nəticələri saxla
with open("blast_result.xml", "w") as out_handle:
    out_handle.write(result_handle.read())
```

Nəticələrin Pars Edilməsi:

BLAST nəticələrini XML formatında saxladıqdan sonra Bio.Blast.NCBIXML modulu ilə bu nəticələri oxuyub təhlil edə bilərik.

[*]:
```
from Bio.Blast import NCBIXML

# SAX XML nəticəsini oxumaq
with open("blast_result.xml") as result_handle:
    blast_records = NCBIXML.parse(result_handle)

    for blast_record in blast_records:
        for alignment in blast_record.alignments:
            print(f"Hit: {alignment.title}")
            for hsp in alignment.hsps:
                print(f"E-value: {hsp.expect}")
```

3. BLAST Sorğularının Parametrləri:

Biopython-da BLAST sorğularını qblast() funksiyasına müxtəlif parametrlər əlavə edərək tənzimləyə bilərsiniz. Bu parametrlər sorğunun necə həyata keçiriləcəyini müəyyən edir.

Əsas Parametrlər:

•**program**: Hansı BLAST növü istifadə olunacağını göstərir (məsələn, "blastp", "blastn").

•**database**: Nəzarət ediləcək verilənlər bazası (məsələn, "nr" — ümumi zülal verilənlər bazası).

•**e_value**: Nəticələrin E-dəyərini müəyyənləşdirir (daha kiçik E-dəyəri daha əhəmiyyətli uyğunluq deməkdir).

•**word_size**: Axtarış üçün istifadə olunan "word" ölçüsünü tənzimləyir.

•**gapopen və gapextend**: Nəticə alinarkən boşluqların (gap) açılma və uzadılma cəzaları.

Məsələn Parametrlərlə BLAST Sorğusu:

```
[*]: result_handle = NCBIWWW.qblast(
         "blastp",
         "nr",
         sequence,
         e_value=0.001,
         gapopen=11,
         gapextend=1
     )
```

4. BLAST Nəticələrini Filtrləmək:

BLAST nəticələrini daha əhəmiyyətli məlumatlara ayırmaq üçün E-dəyəri və ya digər statistik göstəricilər əsasında filtr edə bilərsiniz.

E-dəyərinə görə nəticə filtrələmək:

[37]:
```python
from Bio.Blast import NCBIXML

# Nəticələri oxumaq
with open("blast_result.xml") as result_handle:
    blast_records = NCBIXML.parse(result_handle)

    for blast_record in blast_records:
        for alignment in blast_record.alignments:
            for hsp in alignment.hsps:
                if hsp.expect < 0.001:
                    print(f"Significant hit: {alignment.title}")
                    print(f"E-value: {hsp.expect}")
```

5. BLAST Təhlili və Nəticələrin İstifadəsi:

Biopython ilə BLAST axtarışlarını etdikdən sonra, əldə etdiyiniz nəticələri müxtəlif məqsədlərlə istifadə edə bilərsiniz:

•**Genetik analizlər:** Yeni genlərin annotasiyası və onların funksional analizini həyata keçirmək.

•**Evulusiya tədqiqatları:** Fərqli növlər arasındakı təkamül əlaqələrini araşdırmaq.

•**Zülal strukturu:** Yeni zülalların strukturlarını müəyyənləşdirmək və onların funksiyalarını anlamaq.

Biopython, BLAST nəticələrini təhlil edərkən məlumatların effektiv idarə olunmasını təmin edir və böyük verilənlər dəstləri ilə işləməyi asanlaşdırır.

BLAST-ın əsas xüsusiyyətləri bunlardır:

I. BLAST növləri:

Bu skriptlər BLAST sorğularını NCBI serverinə göndərir, nəticələri alır və XML formatında saxlayır. Daha sonra bu XML nəticələrini Biopython-un **Bio.Blast.NCBIXML** modulu vasitəsilə analiz edərək müvafiq məlumatları çıxarmaq mümkündür.

1. BLASTP:

Amin turşusu sorğu ardıcıllığını zülal ardıcıllıqları verilənlər bazası ilə müqayisə edir.

[*]:
```
from Bio.Blast import NCBIWWW

# Example protein sequence
query_sequence = "MVKVYAPASSANMSVGFDVLGAAVTPVDGALLGDVVTVEAAETFSLNNLGQKL"

# Run BLASTP
print("Running BLASTP...")
result_handle = NCBIWWW.qblast("blastp", "nr", query_sequence)

# Save result
with open("blastp_result.xml", "w") as out_handle:
    out_handle.write(result_handle.read())

result_handle.close()
```

```
Running BLASTP...
```

2. BLASTN:

Nukleotid sorğu ardıcıllığını nukleotid ardıcıllıqları verilənlər bazası ilə müqayisə edir.

[18]:
```
#BLASTN
from Bio.Blast import NCBIWWW
import os

# Real DNA sequence (nümunə olaraq)
query_sequence = "ATGCGTACCTGAGGAGCTTACGTGACTGAGGACTTGA"

# BLASTN sorğusunu göndər
print("BLASTN started...")
result_handle = NCBIWWW.qblast("blastn", "nt", query_sequence, format_type="XML")

# Faylı saxla
output_file = "blastn_result.xml"
with open(output_file, "w") as out_handle:
    out_handle.write(result_handle.read())
result_handle.close()

print(f"BLAST result saved as {output_file} in {os.getcwd()}")
```

```
BLASTN started...
BLAST result saved as blastn_result.xml in C:\Users\Acer
```

3. BLASTX:

Nukleotid sorğu ardıcıllığının altı mümkün oxuma çərçivəsinə əsaslanan nəzəri tərcümə məhsullarını zülal ardıcıllıqları verilənlər bazası ilə müqayisə edir.

[19]:
```
#BLASTX
from Bio.Blast import NCBIWWW
import os

# ƏSL DNA ardıcıllığı
query_sequence = "ATGGCCATTGTAATGGGCCGCTGAAAGGGTGCCCGATAG"

# BLASTX axtarışı
print("BLASTX started...")
result_handle = NCBIWWW.qblast("blastx", "nr", query_sequence, format_type="XML")

# Faylı yadda saxla
with open("blastx_result.xml", "w") as result_file:
    result_file.write(result_handle.read())
result_handle.close()

print(f"BLASTX result saved in {os.getcwd()}/blastx_result.xml")
```

```
BLASTX started...
BLASTX result saved in C:\Users\Acer/blastx_result.xml
```

4. TBLASTN:

Zülal sorğu ardıcıllığını, bütün altı oxuma çərçivəsində dinamik şəkildə tərcümə olunan nukleotid ardıcıllıqları verilənlər bazası ilə müqayisə edir.

[20]:
```
#TBLASTN
from Bio.Blast import NCBIWWW
import os

# əsl protein ardıcıllığı (nümunə olaraq)
query_sequence = "MVKVYAPASSANMSVGFDVLGAAVTPVDGALLGDVVTVEAAETFSLNNLGQKL"

# TBLASTN axtarışı - protein → nukleotid bazasında tərcümə ilə
print("TBLASTN is starting... Please wait.")
result_handle = NCBIWWW.qblast("tblastn", "nr", query_sequence, format_type="XML")

# Nəticəni fayla yaz
output_file = "tblastn_result.xml"
with open(output_file, "w") as result_file:
    result_file.write(result_handle.read())

result_handle.close()

print(f"TBLASTN result saved successfully in: {os.path.abspath(output_file)}")
```

```
TBLASTN is starting... Please wait.
TBLASTN result saved successfully in: C:\Users\Acer\tblastn_result.xml
```

- **tblastn** – protein → nukleotid məlumat bazasında (tərcümə edilmiş şəkildə) axtarış edir.
- **"nr"** – NCBI-nin bütün nukleotid/protein məlumat bazasıdır.
- **format_type="XML"** – nəticəni .xml formatında saxlamaq üçün lazımdır.
- **query_sequence** – mütləq **doğru protein ardıcıllığı** olmalıdır. Amino turşuları A-Z formatında (məsələn M, V, K, L, F, S, və s.).

5. TBLASTX:

Nukleotid sorğu ardıcıllığının altı oxuma çərçivəsində tərcümə olunmuş məhsullarını, nukleotid ardıcıllıqları verilənlər bazasının altı oxuma çərçivəsində tərcümə olunmuş məhsulları ilə müqayisə edir.

[21]:
```python
#TBLASTX
from Bio.Blast import NCBIWWW
import os

# əsl DNA ardıcıllığı (nümunə olaraq)
query_sequence = "ATGGCCATTGTAATGGGCCGCTGAAAGGGTGCCCGATAG"

# TBLASTX axtarışı - DNA → protein bazasında tərcümə ilə
print("TBLASTX is starting... Please wait.")
result_handle = NCBIWWW.qblast("tblastx", "nr", query_sequence, format_type="XML")

# Nəticəni fayla yaz
output_file = "tblastx_result.xml"
with open(output_file, "w") as result_file:
    result_file.write(result_handle.read())

result_handle.close()

print(f"TBLASTX result saved successfully in: {os.path.abspath(output_file)}")
```

```
TBLASTX is starting... Please wait.
TBLASTX result saved successfully in: C:\Users\Acer\tblastx_result.xml
```

- **tblastx** – DNA ardıcıllığını istifadə edərək, onun tərcümə edilmiş formalarını **protein məlumat bazasında** axtarır.
- **"nr"** – NCBI-nin protein məlumat bazasıdır.
- **format_type="XML"** – Nəticənin XML formatında alınması üçün lazımdır.
- **query_sequence** – **Əsl DNA ardıcıllığı** olmalıdır (bu, amin turşuları üçün kodlaşdırıcı DNT ardıcıllığıdır).

II. Alqoritm:

BLAST (Basic Local Alignment Search Tool) tərəfindən istifadə olunan alqoritm, sorğu ardıcıllığı ilə verilənlər bazasında olan ardıcıllıqlar arasında yerli ardıcıllıq oxşarlıqlarını sürətlə müəyyən etmək üçün hazırlanmış heuristik metodlara əsaslanır. Alqoritm ilkin olaraq Stephen F. Altschul, Warren Gish, David J. Lipman və Eugene W. Myers tərəfindən inkişaf etdirilmiş və 1990-cı ildə nəşr olunan bir məqalədə ilk dəfə təsvir edilmişdir. İllər ərzində BLAST bir neçə yeniləmə və təkmilləşdirmə keçmişdir.

Biopython-da NCBIWWW.qblast istifadə edərək BLAST axtarışı etdikdə, sorğu ardıcıllığının işlənməsi və BLAST serverinə göndərilmək üçün formatlanması avtomatik olaraq həyata keçirilir. Sorğu ardıcıllığının sözlərə və ya hissələrə necə bölünməsi ilə bağlı daxili detallar BLAST alqoritmi tərəfindən idarə olunur. BLAST alqoritminin ardıcıllığının yerinə yetirilməsi detalları BLAST alqoritmində daxil edilmişdir və Biopython istifadə edərkən bu addımı açıq şəkildə idarə etməyə ehtiyac yoxdur.

BLAST alqoritminin sadələşdirilmiş ümumi təsviri budur:

1. Sorğu Ardıcıllığının İşlənməsi:

Sorğu ardıcıllığı qısa alt ardıcıllıqlara, yəni “sözlər”ə bölünür.

Bu sözlər adətən zülal ardıcıllıqları üçün 3–4 qalıq, DNT ardıcıllıqları üçün isə 11–12 nukleotiddən ibarət olur.

2. Verilənlər Bazası Ardıcıllığının İşlənməsi:

Verilənlər bazası də sözlərə bölünür.

Verilənlər bazasında olan sözlər üçün indeks yaradılır, bu da sürətli əldə etməyə imkan verir.

3. Söz Uyğunluqları:

BLAST əvvəlcə sorğudakı sözlər ilə verilənlər bazasındakı sözlər arasında qısa, dəqiq uyğunluqları (hitlər) müəyyən edir.

Bu söz uyğunluqları yerli hizalamalar üçün potensial başlanğıc nöqtələri kimi fəaliyyət göstərir.

4. Hitlərin Uzadılması:

BLAST bu hitləri hər iki istiqamətdə uzadır və daha uzun yerli hizalamalar yaradır.

Bir uyğunluq qiymətləndirmə sistemi istifadə olunur ki, bu da uyğunluqları, uyğunsuzluqları və boşluq cəzalarını nəzərə alaraq hizalamaların keyfiyyətini dəyərləndirir.

5. Qiymətləndirmə və Dəyərləndirmə:

Uyğunluqlara və uyğunsuzluqlara xal vermək üçün qiymətləndirmə matrisi istifadə olunur.

Hizalama daxilində boşluqların yaranmasını hesablamaq üçün boşluq cəzaları tətbiq olunur.

Son hizalama xalı, bütün hizalamayı nəzərə alaraq hesablanır.

6. Statistiki Əhəmiyyət:

BLAST, E-dəyəri (Expect Value) hesablayır, bu da müəyyən bir ölçüdə verilənlər bazasında şansa bağlı olaraq oxşar və ya daha yaxşı xala malik hizalamaların gözlənilən sayını təxmin edir.

Daha aşağı E-dəyəri daha əhəmiyyətli hizalamaları göstərir.

7. Çıxış:

BLAST, müəyyən bir həddi keçən xallara malik hizalamaları, E-dəyərləri və digər statistik ölçülərlə birlikdə bildirir.

İstifadəçilər nəticələri araşdıra bilər və homolog ardıcıllıqları müəyyən edərək funksional və təkamül əlaqələri haqqında məlumat toplaya bilərlər.

BLAST-ın heuristik təbiəti, verilənlər bazasını tam şəkildə axtarmağa ehtiyac qalmadan yerli ardıcıllıq oxşarlıqlarını sürətlə müəyyən etməyə imkan verir. Bu, BLAST-ı böyük miqyaslı ardıcıllıq verilənlər bazası axtarışları üçün uyğun edir və bioinformatika və molekulyar biologiyada geniş istifadəsinin səbəblərindən biridir.

Heuristik metodlar, adətən, mürəkkəb məsələləri sadələşdirir və tam bir axtarış və ya hesablamaya ehtiyac olmadan pragmatik və praktik həllər təklif edir. Bu yanaşma, mümkün olan ən yaxşı nəticəni təmin etməkdən çox, təxmini doğru nəticələrə tez çatmağa kömək edir.

III. Qiymətləndirmə Sistemi:

BLAST-da qiymətləndirmə sistemi, ardıcıllıq hizalamalarının müxtəlif elementlərinə, o cümlədən uyğunluqlara, uyğunsuzluqlara və boşluqlara riyazi xallar təyin etmək üçün istifadə olunur. Bu xallar daha sonra hizalamanın keyfiyyətini dəyərləndirmək üçün istifadə edilir. Qiymətləndirmə sistemi hizalama xalının hesablanmasına kömək edir və BLAST-ın hizalamanın statistik əhəmiyyətini E-dəyərlərin hesablanması yolu ilə müəyyən etməsinə kömək edir.

BLAST-da istifadə olunan qiymətləndirmə sisteminin bəzi əsas komponentləri:

1. Uyğunluq Xalı (S):

Hizalamada uyğun olan qalıqlara təyin edilən xal. Bu, müsbət bir xaldır.

2. Uyğunsuzluq Xalı (M):

Hizalamada uyğun olmayan qalıqlara təyin edilən xal. Bu, adətən mənfi bir xaldır.

3. Boşluq Cəzası (G):

Hizalamada boşluq əlavə etmək üçün təyin edilən cəza. Boşluq açmaq üçün (Boşluq Açma Cəzası) və mövcud bir boşluğu uzatmaq üçün (Boşluq Uzatma Cəzası) ayrı cəzalar ola bilər.

4. Boşluq Uzatma Cəzası (E):

Uzadılmış boşluqdakı hər bir qalıq üçün təyin edilən cəza.

Verilmiş bir hizalama üçün **hizalama xalı (S)**, uyğunluq xalları, uyğunsuzluq xalları və boşluq cəzalarının cəmi kimi hesablanır.

Yekun hizalama xalı, həmin hizalamanın keyfiyyətini qiymətləndirmək üçün istifadə olunur.

Biopython kitabxanasında, **NCBIWWW.qblast** funksiyasından istifadə edərək BLAST axtarışlarını həyata keçirərkən qiymətləndirmə sistemini fərdiləşdirmək mümkündür.

qblast funksiyası **reward**, **penalty**, **gapopen** və **gapextend** kimi parametrlərə malikdir. Bu parametrlər vasitəsilə uyğunluq və uyğunsuzluq xallarını, həmçinin boşluq cəzalarını təyin edə bilərsiniz.

[26]:
```python
from Bio.Blast import NCBIWWW

query_sequence = "MVKVYAPASSANMSVGFDVLGAAVTPVDGALLGDVVTVEAAETFSLNNLGQKL"

# Customize the scoring parameters
result_handle = NCBIWWW.qblast(
    "blastp",
    "nr",
    query_sequence,
    reward=1,    # Match reward
    penalty=-2,  # Mismatch penalty
    gapopen=2,   # Gap open penalty
    gapextend=1  # Gap extension penalty
)

# Save the result
with open("custom_scoring_result.xml", "w") as result_file:
    result_file.write(result_handle.read())

result_handle.close()
```

Qiymətləndirmə sistemini öz seçimlərinizə uyğunlaşdırmaq üçün **reward**, **penalty**, **gapopen** və **gapextend** parametrlərini tənzimləyə bilərsiniz.

IV. E-dəyər (Expect Value):

E-dəyər (Gözlənilən Dəyər), müəyyən bir ölçüdə olan verilənlər bazasında təsadüfən ortaya çıxması gözlənilən, müşahidə olunan xalla eyni və ya daha yaxşı xala malik hizalamaların sayını ifadə edir.

Daha aşağı E-dəyər, daha əhəmiyyətli bir uyğunluğu göstərir.

Aşağı E-dəyər, uyğunluğun statistik baxımdan daha əhəmiyyətli olduğunu göstərir. **E-dəyər** tez-tez BLAST nəticələrini filtirləmək üçün hədd kimi istifadə olunur.

Məsələn, əgər E-dəyər həddi olaraq **0.01** təyin etsəniz, bu o deməkdir ki, yalnız axtarışların 1%-ində təsadüfi olaraq baş verə biləcək uyğunluqları nəzərə alırsınız.

Biopython-da, **E-dəyər** BLAST nəticə çıxışında təqdim edilən məlumatın bir hissəsidir və siz bu dəyəri proqram vasitəsilə əldə edə bilərsiniz.

[*]:
```python
from Bio.Blast import NCBIWWW
from Bio.Blast import NCBIXML

# Əsl protein ardıcıllığını daxil et
query_sequence = "MVKVYAPASSANMSVGFDVLGAAVTPVDGALLGDVVTVEAAETFSLNNLGQKL"

# BLASTP sorğusu göndər
print("Sending BLAST request...")
result_handle = NCBIWWW.qblast("blastp", "nr", query_sequence)

# XML faylına yaz (opsional)
with open("blast_result.xml", "w") as out_handle:
    out_handle.write(result_handle.read())
result_handle.close()

# Faylдan yenidən oxumaq
with open("blast_result.xml") as result_handle:
    blast_record = NCBIXML.read(result_handle)

# İlk uyğunluğu göstər
first_alignment = blast_record.alignments[0]
e_value = first_alignment.hsps[0].expect

print(f"E-value: {e_value}")
```

Bu nümunədə, BLAST axtarışı aparıldıqdan sonra nəticə **NCBIXML.read()** funksiyası ilə pars edilir. Daha sonra ilk hizalamanın **E-dəyəri**, **hsps** (Yüksək Xallı Seqment Cütü) atributu vasitəsilə əldə edilir. **HSP** obyektinin **expect** atributu E-dəyərini təqdim edir.
Siz **E-dəyərdən** istifadə edərək BLAST nəticələrinizi ardıcıllıqlar arasındakı oxşarlıqların statistik əhəmiyyətinə əsaslanaraq süzə və prioritetləşdirə bilərsiniz. **Daha aşağı E-dəyərlər** adətən daha etibarlı hesab olunur və bioloji baxımdan əhəmiyyətli uyğunluqları göstərir.

V. Nəticə (Output):

BLAST çıxış nəticələrini xallarına görə sıralanmış hizalamalar siyahısı şəklində təqdim edir.

Çıxışda ardıcıllıqlar arasındakı oxşarlıqlar, **E-dəyərlər** və hizalama detalları haqqında məlumatlar yer alır.

VI. Tətbiqlər (Applications):

BLAST bioinformatika və molekulyar biologiya sahəsində müxtəlif məqsədlər üçün geniş istifadə olunur, o cümlədən:

- Homoloji ardıcıllıqları müəyyən etmək.
- Yeni sekanslaşdırılmış genlərin annotasiyası.
- Zülal funksiyalarını proqnozlaşdırmaq.
- Növlər arasında təkamül əlaqələrini öyrənmək.
- Mühafizə olunan domenlər və motivləri müəyyən etmək.

VII. Onlayn və Local Versiyalar:

BLAST həm NCBI veb saytında onlayn xidmət olaraq, həm də kompüterə quraşdırılaraq yerli olaraq işlədilə bilən müstəqil proqram olaraq mövcuddur. Local versiya, tədqiqatçılara böyük verilənlər toplusunu internetə qoşulmağa ehtiyac olmadan analiz etməyə imkan verir.

BLAST bioinformatikada çox yönlü və vacib bir alətdir, tədqiqatçılara bioloji ardıcıllıqları müqayisə etməyə və genlərin və zülalların strukturu və funksiyası haqqında dəyərli məlumatlar əldə etməyə imkan verir.

VIII. Nümunə İş Prosesi (Example Workflow):

1.Veriləri Yükləmək:

pd.read_csv funksiyasını istifadə edərək DNA ardıcıllıqlarını CSV faylından Pandas DataFrame-ə yükləyirik.

2.Ardıcıl Sequensiyalar Üzərində Döngü:

Verilənlər toplusundakı hər bir DNA ardıcıllığı üzərində iterasiya edirik.

3.BLASTN Axtarışı Aparmaq:

Hər bir ardıcıllıq üçün, **NCBIWWW.qblast** funksiyasını istifadə edərək nt (nukleotid) verilənlər bazasında BLASTN axtarışı aparırıq.

4.BLAST Nəticəsini Pars Etmək:

NCBIXML.read funksiyasını istifadə edərək BLAST nəticəsini pars edirik və hizalamalar haqqında məlumat əldə edirik.

5.Ən Yaxın Uyğunluq Məlumatlarına Çatmaq:

Ən yaxşı uyğunluğun məlumatlarını çıxarırıq, məsələn, uyğunluq başlığı və **E-dəyəri**.

6.Nəticələri Çap Etmək və ya Saxlamaq:

Nəticələri çap edə bilərsiniz və ya daha sonra analiz üçün uyğun bir formatda saxlayırsınız.

[*]:
```
import pandas as pd
from Bio.Blast import NCBIWWW
from Bio.Blast import NCBIXML

# Load your dataset
df = pd.read_csv("sequences.csv")

# Iterate over each DNA sequence in the dataset
for index, row in df.iterrows():
    query_sequence = row["DNA_sequence"]

    # Perform BLASTN search
    result_handle = NCBIWWW.qblast("blastn", "nt", query_sequence)

    # Parse the BLAST result
    blast_record = NCBIXML.read(result_handle)

    # Access information about the top hit
    top_hit = blast_record.alignments[0]
    hit_title = top_hit.title
    e_value = top_hit.hsps[0].expect

    # Print or store the results as needed
    print(f"Query: {query_sequence}")
    print(f"Top Hit: {hit_title}")
    print(f"E-value: {e_value}")
    print("=" * 30)

    # Close the result handle
    result_handle.close()
```

COVID-19 virusunun sekansı

```
from Bio.Blast import NCBIWWW, NCBIXML
```

2. COVID-19 Sekansını Yükləyirik:

COVID-19 virusunun genetik məlumatlarını GenBank-dan və ya digər etibarlı bioloji verilənlər bazalarından əldə edirik.

Məsələn, SARS-CoV-2 (COVID-19) virusunun sekansını GenBank-dan (ID: MN908947) yükləyin.

2. BLASTN Sorğusu və Sekansların Yüklənməsi

COVID-19 virusunun genetik ardıcıllığını GenBank-dan (məsələn, SARS-CoV-2, GenBank kodu: MN908947) əldə edə bilərsiniz. Bu ardıcıllığı Biopython-da istifadə edərək analiz edəcəyik.

Sekansın Yüklənməsi:

SARS-CoV-2 sekansını GenBank-dan yükləyin. Məsələn, bu sekansı bir dəyişəndə saxlayın:

[5]:
```
from Bio import Entrez, SeqIO

Entrez.email = "your-email@example.com"  # Buraya real email ünvanını yaz

# GenBank formatında məlumatı çəkirik
handle = Entrez.efetch(db="nucleotide", id="MN908947", rettype="gb", retmode="text")

# SeqIO ilə parse edirik
record = SeqIO.read(handle, "genbank")

# Sekansı çıxarırıq
sequence = record.seq

print(sequence)
```

```
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCTGTTCTCTAAACGAACTTTAAAATCTGTGTGGCTGTCACTCGGCTGCATGCTTAGTGCACTCACGCAGTATAATTAATAACTAATTACTG
CGTTGACAGGACACGAGTAACTCGTCTATCTTCTGCAGGCTGCTTACGGTTTCGTCCGTGTTGCAGCCGATCATCAGCACATCTAGGTTTCGTCCGGGTGTGACCGAAAGGTAAGATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAA
ACACGTCCAACTCAGTTTGCCTGTTTTACAGGTTCGCGACGTGCTCGTACGTGGCTTTGGAGACTCCGTGGAGGAGGTCTTATCAGAGGCACGTCAACATCTTAAAGATGGCACTTGTGGCTTAGTAGAAGTTGAAAAAGGCGTTTTGC
TCAACTTGAACAGCCCTATGTGTTCATCAAACGTTCGGATGCTCGAACTGCACCTCATGGTCATGTTATGGTTGAGCTGGTAGCAGAACTCGAAGGCATTCAGTACGGTCGTAGTGGTGAGACACTTGGTGTCCTTGTCCCTCATGTGG
CGAAATACCAGTGGCTTACCGCAAGGTTCTTCTTCGTAAGAACGGTAATAAAGGAGCTGGTGGCCATAGTTACGGCGCCGATCTAAAGTCATTTGACTTAGGCGACGAGCTTGGCACTGATCCTTATGAAGATTTTCAAGAAAACTGGA
CACTAAACATAGCAGTGGTGTTACCCGTGAACTCATGCGTGAGCTTAACGGAGGGGCATACACTCGCTATGTCGATAACAACTTCTGTGGCCCTGATGGCTACCCTCTTGAGTGCATTAAAGACCTTCTAGCACGTGCTGGTAAAGCTT
ATGCACTTTGTCCGAACAACTGGACTTTATTGACACTAAGAGGGGTGTATACTGCTGCCGTGAACATGAGCATGAAATTGCTTGGTACACGGAACGTTCTGAAAAGAGCTATGAATTGCAGACACCTTTTGAAATTAAATTGGCAAAGA
ATTTGACACCTTCAATGGGGAATGTCCAAATTTTGTATTTCCCTTAAATTCCATAATCAAGACTATTCAACCAAGGGTTGAAAAGAAAAAGCTTGATGGCTTTATGGGTAGAATTCGATCTGTCTATCCAGTTGCGTCACCAAATGAAT
CAACCAAATGTGCCTTTCAACTCTCATGAAGTGTGATCATTGTGGTGAAACTTCATGGCAGACGGGCGATTTTGTTAAAGCCACTTGCGAATTTTGTGGCACTGAGAATTTGACTAAAGAAGGTGCCACTACTTGTGGTTACTTACCCC
AAATGCTGTTGTTAAAATTTATTGTCCAGCATGTCACAATTCAGAAGTAGGACCTGAGCATAGTCTTGCCGAATACCATAATGAATCTGGCTTGAAAACCATTCTTCGTAAGGGTGGTCGCACTATTGCCTTTGGAGGCTGTGTGTTCT
TTATGTTGGTTGCCATAACAAGTGTGCCTATTGGGTTCCACGTGCTAGCGCTAACATAGGTTGTAACCATACAGGTGTTGTTGGAGAAGGTTCCGAAGGTCTTAATGACAACCTTCTTGAAATACTCCAAAAAGAGAAAGTCAACATCA
```

Sekansın Göstərilməsi:

Sekansı əldə etdikdən sonra bu sekansı çap edərək baxa bilərsiniz:

```
[7]: print(sequence[:1000])  # İlk 1000 nukleotidi göstəririk
```

```
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCTGTTCTCTAAACGAACTTTAAAATCTGTGTGGCTGTCACTCGGCTGCATGCTTAGTGCACTCACGCAGTATAATTAATAACTAATTACTGTCGT
TGACAGGACACGAGTAACTCGTCTATCTTCTGCAGGCTGCTTACGGTTTCGTCCGTGTTGCAGCCGATCATCAGCACATCTAGGTTTCGTCCGGGTGTGACCGAAAGGTAAGATGGAGAGCCTTGTCCCTGGTTTCAACGAGAAAACACACGT
CCAACTCAGTTTGCCTGTTTTACAGGTTCGCGACGTGCTCGTACGTGGCTTTGGAGACTCCGTGGAGGAGGTCTTATCAGAGGCACGTCAACATCTTAAAGATGGCACTTGTGGCTTAGTAGAAGTTGAAAAAGGCGTTTTGCCTCAACTTGA
ACAGCCCTATGTGTTCATCAAACGTTCGGATGCTCGAACTGCACCTCATGGTCATGTTATGGTTGAGCTGGTAGCAGAACTCGAAGGCATTCAGTACGGTCGTAGTGGTGAGACACTTGGTGTCCTTGTCCCTCATGTGGGCGAAATACCAGT
GGCTTACCGCAAGGTTCTTCTTCGTAAGAACGGTAATAAAGGAGCTGGTGGCCATAGTTACGGCGCCGATCTAAAGTCATTTGACTTAGGCGACGAGCTTGGCACTGATCCTTATGAAGATTTTCAAGAAAACTGGAACACTAAACATAGCAG
TGGTGTTACCCGTGAACTCATGCGTGAGCTTAACGGAGGGGCATACACTCGCTATGTCGATAACAACTTCTGTGGCCCTGATGGCTACCCTCTTGAGTGCATTAAAGACCTTCTAGCACGTGCTGGTAAAGCTTCATGCACTTTGTCCGAACA
ACTGGACTTTATTGACACTAAGAGGGGTGTATACTGCTGCCGTGAACATGAGCATGAAATTGCTTGGTACACGGAACGTTCT
```

3. BLASTN Sorğusu İcra Etmək

BLASTN, nukleotid ardıcıllıqları üçün xüsusi bir BLAST versiyasıdır. NCBI-nin web xidmətini istifadə edərək bu sekansı BLAST-a göndərə bilərik.

```
[*]: from Bio.Blast import NCBIWWW

# BLAST sorğusunu göndəririk (bu bir neçə dəqiqə çəkə bilər)
result_handle = NCBIWWW.qblast("blastn", "nt", sequence)

# Fayla yazırıq
with open("blast_result.xml", "w") as out_handle:
    out_handle.write(result_handle.read())
result_handle.close()
```

```
[*]: from Bio.Blast import NCBIXML

# XML faylını oxuyuruq
with open("blast_result.xml") as result_handle:
    blast_records = NCBIXML.parse(result_handle)

    for blast_record in blast_records:
        for alignment in blast_record.alignments:
            print(f"Title: {alignment.title}")
            print(f"Length: {alignment.length}")
            for hsp in alignment.hsps:
                print(f"E-value: {hsp.expect}")
                print(f"Identity: {hsp.identities}")
                print(f"Match:\n{hsp.query}\n{hsp.match}\n{hsp.sbjct}\n")
                print("-" * 40)
```

4. BLAST Nəticələrini Analiz Etmək

BLAST nəticələrini analiz etmək üçün blast_records istifadə edəcəyik. Bu, nəticələri pars etmək və ilk uyğunluğu çıxarmaq üçün istifadə ediləcək.

[*]:
```
# İlk nəticəni alırıq
blast_record = next(blast_records)

# İlk nəticənin hitlərini yazdırırıq
for alignment in blast_record.alignments:
    for hsp in alignment.hsps:
        print(f"Hit title: {alignment.title}")
        print(f"E-value: {hsp.expect}")
        print(f"Score: {hsp.score}")
        print(f"Query sequence: {hsp.query[0:100]}")  # İlk 100 nukleotidi göstərin
        print(f"Match sequence: {hsp.sbjct[0:100]}")  # İlk 100 nukleotidi göstərin
        print("-----------")
```

5. Nəticələrin Təhlili və Yekunlaşdırılması

Nəticələri analiz edərək, E-value və score kimi statistik göstəricilərdən istifadə edə bilərsiniz. Məsələn:

•**E-value**: Aşağı E-value, daha etibarlı nəticələri göstərir. Bu, uyğunluğun şanslı olub olmadığını bildirir.

•**Score**: Yüksək skorlar, daha yaxşı uyğunluqları göstərir.

[*]:
```python
from Bio import Entrez
from Bio.Blast import NCBIWWW, NCBIXML

# NCBI-yə daxil olmaq üçün email adresinizi yazın
Entrez.email = "your-email@example.com"

# GenBank-dan SARS-CoV-2 sekansını yükləyirik
handle = Entrez.efetch(db="nucleotide", id="MN908947", rettype="gb", retmode="text")
record = Entrez.read(handle)

# Sekansı çıxarırıq
sequence = record[0]["ORIGIN"]

# BLASTN sorğusunu göndəririk
result_handle = NCBIWWW.qblast("blastn", "nt", sequence)

# Nəticələri XML formatında oxuyuruq
blast_records = NCBIXML.parse(result_handle)

# İlk nəticəni alırıq və analiz edirik
blast_record = next(blast_records)

# Nəticələri çap edirik
for alignment in blast_record.alignments:
    for hsp in alignment.hsps:
        print(f"Hit title: {alignment.title}")
        print(f"E-value: {hsp.expect}")
        print(f"Score: {hsp.score}")
        print(f"Query sequence: {hsp.query[0:100]}")  # İlk 100 nukleotidi göstərin
        print(f"Match sequence: {hsp.sbjct[0:100]}")  # İlk 100 nukleotidi göstərin
        print("-----------")
```

Homologluqların Tapılması (Homology Search)

Homologluq, iki və ya daha çox bioloji ardıcıllığın ortaq bir əcdaddan gəldiyini göstərir. BLAST aləti, SARS-CoV-2 virusunun genetik ardıcıllığını digər viruslarla **qarşılaşdıraraq** onların nə dərəcədə homolog olduğunu aşkar etməkdə istifadə olunur.

SARS-CoV-2 genomunu istifadə edərək GenBank bazasında onunla **ən çox oxşarlığı olan** virusları tapmaq və onların **genetik fərqlərini** müşahidə etmək.

Addım-addım Homolog Tapılması

1. SARS-CoV-2 Genomunun Əldə Edilməsi

[18]:
```python
from Bio import Entrez

Entrez.email = "your-email@example.com"
handle = Entrez.efetch(db="nucleotide", id="MN908947", rettype="fasta", retmode="text")
sequence_data = handle.read()
handle.close()

print(sequence_data[:500])  # Genomun ilk 500 bazını göstəririk
```

```
>MN908947.3 Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1, complete genome
ATTAAAGGTTTATACCTTCCCAGGTAACAAACCAACCAACTTTCGATCTCTTGTAGATCTGTTCTCTAAA
CGAACTTTAAAATCTGTGTGGCTGTCACTCGGCTGCATGCTTAGTGCACTCACGCAGTATAATTAATAAC
TAATTACTGTCGTTGACAGGACACGAGTAACTCGTCTATCTTCTGCAGGCTGCTTACGGTTTCGTCCGTG
TTGCAGCCGATCATCAGCACATCTAGGTTTCGTCCGGGTGTGACCGAAAGGTAAGATGGAGAGCCTTGTC
CCTGGTTTCAACGAGAAAACACACGTCCAACTCAGTTTGCCTGTTTTACAGGTTCGCGACGTGCTCGTAC
GTGGCTTTGGAGACTCCGTGGAGGAGGTCTTATCAGAGGCACGTCAACA
```

2. BLASTN ilə axtarış

[*]:
```
from Bio.Blast import NCBIWWW

result_handle = NCBIWWW.qblast(
    program="blastn",  # nukleotid axtarışı üçün
    database="nt",     # GenBank nukleotid bazası
    sequence=sequence_data,
    hitlist_size=5     # ən yaxşı 5 uyğunluğu göstər
)
```

3. BLAST Nəticələrinin Pars Edilməsi

[9]:
```python
from Bio.Blast import NCBIXML

blast_record = NCBIXML.read(result_handle)

for alignment in blast_record.alignments:
    print("==================================")
    print("Başlıq (Title):", alignment.title)
    print("Uzunluq:", alignment.length)

    for hsp in alignment.hsps:
        print("Oxşarlıq (Identity):", hsp.identities)
        print("Score:", hsp.score)
        print("E-value:", hsp.expect)
        print("Uyğunluq hissəsi (Query):", hsp.query[:75])
        print("Baza uyğunluğu (Subject):", hsp.sbjct[:75])
        print("Uzunluğu:", hsp.align_length)
```

Homolog Tapıldıqdan Sonra Nə Etmək Olar?

1. Fərqlərin Müqayisəsi:

•Uyğunluq (alignment) hissələrini müqayisə edərək hansı nukleotidlərdə fərqlilik olduğunu görə bilərsiniz.

2. Filogenetik Analizə Başlamaq:

•Homolog ardıcıllıqları ClustalW və ya MAFFT ilə çoxlu ardıcıllıq düzülməsi (multiple sequence alignment) üçün istifadə edə bilərsiniz.

3. Yeni Virus Variantlarının Tanınması:

•Əgər fərqliliklər çoxdursa, bu yeni bir **variant** ola bilər.

Məsələn:

SARS-CoV-2 genomu ilə BLASTN etdikdə, aşağıdakı oxşar viruslar tapıla bilər:

•Bat coronavirus RaTG13 (təxminən 96.2% homolog)

•Pangolin coronavirus (təxminən 90% homolog)

•SARS-CoV (təxminən 80% homolog)

Bu, virusun **mümkün mənşəyi** və **təkamül yolu** barədə önəmli ipucları verir.