3.2 Blast

1. 网页版blastp，过程和结果如下

输入序列，选择blastp，选择数据库mouse

确定e值范围，输入内容限制为10个，点击BLAST

结果如下

> **P值**：P值是用来判定假设检验结果的一个参数，也可以根据不同的分布使用分布的拒绝域进行比较。由Fisher首先提出。P值（P-value）就是当原假设为真时，比所得到的样本观察结果更极端的结果出现的概率。如果P值很小，说明原假设情况的发生的概率很小，而如果出现了，根据小概率原理，我们就有理由拒绝原假设，P值越小，我们拒绝原假设的理由越充分。总之，P值越小，表明结果越显著。

> **E值**: E值的概念起源于20世纪90年代初，随着基因测序技术的快速发展，生物学家面临海量序列数据的分析需求。传统的统计学方法（如卡方检验）无法高效评估序列比对结果的显著性，尤其是针对大规模数据库搜索时，亟需一种能平衡 计算效率 和 统计严谨性 的新指标。 数学家Samuel Karlin与生物信息学家Stephen Altschul合作，首次提出基于 极值分布（Extreme Value Distribution, EVD） 的比对显著性评估模型。该模型的核心思想是：局部比对的高分事件服从极值分布，而非正态分布。 E值统一了 比对质量和搜索空间规模 的双重考量，且直接反映了假阳性风险。

> 在BLAST中，p值和e值均用于评估序列比对结果的显著性，但其侧重点不同： p值（p-value）：表示在无真实关联的情况下，某次比对得分偶然出现的概率（概率越小，结果越可信）。其计算基于比对得分的极值分布，公式为 $(p = 1 - e^{-e^{-\lambda(S-\mu)}})$。 e值（E-value）：表示在相同数据库中，预期出现同等或更高得分的随机比对次数（数值越低，意义越大）。公式为 $(E = m \cdot n \cdot e^{-\lambda S})$，其中 (m,n) 分别为查询序列和数据库的规模。 p值关注单次比对的偶然性。

2. Bash脚本编程，过程和结果如下

```
#!/bin/bash

seq=MSTRSVSSSSYRRMFGGPGTASRPSSSRSYVTTSTRTYSLGSALRPSTSRSLYASSPGGVYATRSSAVRL

# remove all dir and files
```

```
if [ -d "input" ]; then
  rm -r input
fi

if [ -d "output" ]; then
  rm -r output
fi

# generate random sequence
echo "Begin generate random sequence..."
mkdir input
cd input
for i in {1..10}; do
  shuffled=$(
    echo "$seq" | fold -w1 | shuf | tr -d '\n'
  )
  echo ">> seq$i: $shuffled"
  echo $shuffled > seq$i.fasta
done
echo "Generate random sequence finished."
cd ..

total_pairs=0

# blastp each sequence pair
mkdir output
cd output
echo "Begin blastp..."
for i in {1..10}; do
    for j in {1..10}; do
        if [ $i -lt $j ]; then
            outfile="seq${i}_vs_seq${j}.txt"
            echo "seq$i: " >> $outfile
            cat ../input/seq${i}.fasta >> $outfile
            echo "seq$j: " >> $outfile
            cat ../input/seq${j}.fasta >> $outfile
            echo "\n" >> $outfile
```

```
            blastp \
                -query ../input/seq${i}.fasta \
                -subject ../input/seq${j}.fasta \
                >> $outfile
            echo "finish blastp between ../input/seq${i}.fasta and ../input/seq${j}.fasta"

            ((total_pairs++))
        fi
    done
done
echo "Blastp $total_pairs pairs"
echo "Blastp finished."
cd ..
```

选取其中一个结果文件如下（标记为#的为后期添加的注释）：

```
# 为对比序列
seq1:
SVSPFRSGTTTSSTSTGTTGPSRSGGPPYYSYRSVASYSSYSGRSRSSRLASAVSRMMLLSAARSRTVRL
seq3:
SVVGPSALTATSYYASARSSSSVRYVGSPTSRLRRYSGYASSTLSTMSTSGTRSSRRFMRSGLSPSGTRP
\n

#版本说明
BLASTP 2.6.0+

Reference: Stephen F. Altschul, Thomas L. Madden, Alejandro A.
Schaffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J.
Lipman (1997), "Gapped BLAST and PSI-BLAST: a new generation of
protein database search programs", Nucleic Acids Res. 25:3389-3402.


Reference for composition-based statistics: Alejandro A. Schaffer,
L. Aravind, Thomas L. Madden, Sergei Shavirin, John L. Spouge, Yuri
I. Wolf, Eugene V. Koonin, and Stephen F. Altschul (2001),
```

```
"Improving the accuracy of PSI-BLAST protein database searches with
composition-based statistics and other refinements", Nucleic Acids
Res. 29:2994-3005.


# 指明对比的数据库
Database: User specified sequence set (Input: ../input/seq3.fasta).
           1 sequences; 70 total letters



Query=
Length=70
                                                                    Score     E
Sequences producing significant alignments:                        (Bits)  Value

  unnamed protein product                                           9.6    7.1

# 对比得分
> unnamed protein product
Length=70

 Score =  9.6 bits (13),  Expect = 7.1, Method: Compositional matrix adjust.
 # Identities: 完全匹配度，Positives：氨基酸相似度比例，Gaps：插入和缺失占比
 Identities = 3/8 (38%), Positives = 5/8 (63%), Gaps = 0/8 (0%)

#成功比对的序列，也即hit
Query  50  LASAVSRM  57
           + S  SR+
Sbjct  26  VGSPTSRL  33


# 统计参数，用于计算得分分布
Lambda      K        H        a         alpha
   0.307    0.116    0.309    0.792     4.96
# 统计参数，用于计算带gap的得分分布
Gapped
```

```
Lambda      K        H        a         alpha    sigma
   0.267   0.0410    0.140     1.90     42.6     43.6

# 有效搜索空间，表示用于搜索的实际序列数量
Effective search space used: 4096

# 数据库信息
  Database: User specified sequence set (Input: ../input/seq3.fasta).
    Posted date:  Unknown
  Number of letters in database: 70
  Number of sequences in database:  1


# 蛋白质相似性矩阵
Matrix: BLOSUM62
Gap Penalties: Existence: 11, Extension: 1
Neighboring words threshold: 11
Window for multiple hits: 40
```

BLASTP 比对结果显示，seq1 和 seq3 在第 50-57 位和第 26-33 位之间有一段短的相似区域。但是对比的期望值较高（7.1），基本不具有不具有统计学意义，也即二者基本不存在明显的相似关系。这和我们随机生成序列得到的预期比对结果一致。

3. 常用的方法有下面几种

- Local Alignment

> 局部比对通过寻找序列中的局部相似区域，而不是进行全局比对，减少了需要计算的区域，避免了对整个序列进行耗时的全局比对，从而提高了效率。

- 预计算的相似性矩阵

> BLAST使用预计算的相似性矩阵（如BLOSUM矩阵）来快速评估氨基酸或核苷酸之间的相似性。预计算的矩阵允许快速查找相似性得分，避免了在每次比对时重新计算，节省了时间。

- PSI-BLAST

> PSI-BLAST（Position-Specific Iterated BLAST）通过迭代搜索和构建位置特异性评分矩阵（PSSM）来提高灵敏度。迭代搜索允许在每次迭代中发现更多的相似序列，从而在较少的迭代次数内达到较高的灵敏度，减少了总体计算时间。

- Pruning

> Pruning是一种优化策略，用于减少搜索空间。在BLAST搜索过程中，Pruning通过删除与当前查询序列不相关的序列来减少搜索空间，从而提高搜索效率。

4. 对称与不对称PAM250矩阵的原理及应用比较

PAM矩阵的背景与基本概念

> PAM（Percentage Accepted Mutation）矩阵是蛋白质序列比对中常用的相似性评分矩阵，用于衡量两个氨基酸在进化过程中被接受为相似的概率。PAM250表示在250个单位的进化时间中，两个氨基酸被接受为相似的概率。这一矩阵基于进化模型计算，反映了氨基酸在进化过程中的替换频率。

对称PAM250矩阵

> 对称的PAM250矩阵假设氨基酸之间的相似性是相互的，即矩阵中的值在i和j位置上与j和i位置上是相同的。这种对称性基于早期的假设，认为氨基酸之间的替换是相互的，没有方向性差异。 因此显得简单直观：对称矩阵结构简单，易于理解和使用，计算效率高：由于矩阵对称，计算时可以减少一半的存储和计算量。 缺点： 可能不够精确：在某些情况下，氨基酸之间的替换可能具有方向性，对称性可能无法准确反映真实的替换概率。

不对称PAM250矩阵

> 不对称的PAM250矩阵不再假设氨基酸之间的相似性是相互的，而是根据实际的进化数据来调整每个方向的相似性。这种矩阵考虑了氨基酸在不同方向上的替换概率差异，更加贴近真实的进化过程。因此可以带来更高的准确性：不对称矩阵能够更好地反映氨基酸在不同方向上的替换概率，提供更精确的相似性评分；灵活性：适用于需要高精度比对的场景，能够更好地适应复杂的进化关系。

应用上的不同

> 在大多数情况下，对称PAM250已经足够准确，适用于一般的蛋白质序列比对需求。计算资源有限的情况：由于其计算效率高，适合在计算资源有限的环境下使用。 而在需要极高比对精度的场景，如关键蛋白质结构预测或功能分析中，不对称PAM250能够提供更准确的结果；对于涉及复杂进化路径的研究，不对称矩阵能够更好地捕捉氨基酸替换的方向性；在一些高级的生物信息学工具和算法中，不对称PAM250被用于提升比对的灵敏度和特异性。

总结

> 对称PAM250：简单直观，计算效率高，适用于常规比对和资源有限的场景。 不对称PAM250：更精确，能够反映氨基酸替换的方向性，适用于高精度比对和复杂进化分析。