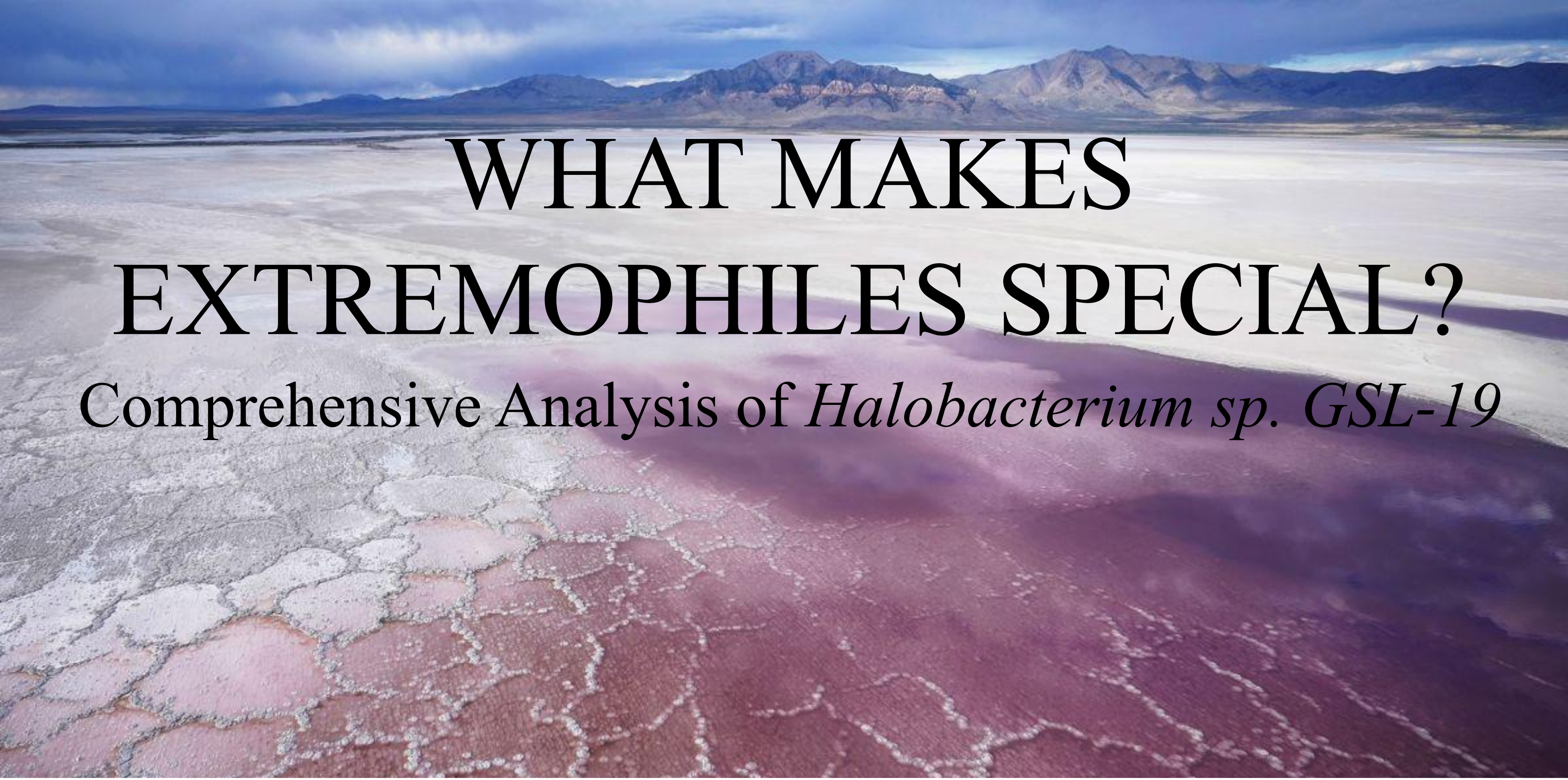


WHAT MAKES EXTREMOPHILES SPECIAL?

Comprehensive Analysis of *Halobacterium sp. GSL-19*

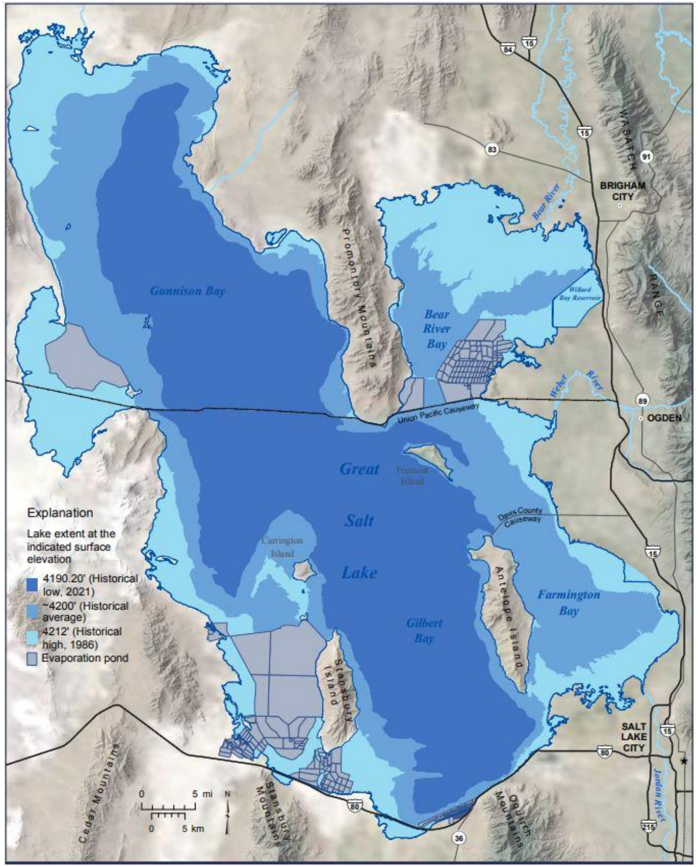
BMI 6015 Foundations in Bioinformatics, University of Utah - Fall 2024
Team: Chetan Elenki, Grace Heyborne, Matias Lee, L Weaver

Objective

Our exploration aims to gain insights into the molecular mechanisms of survival in harsh conditions.

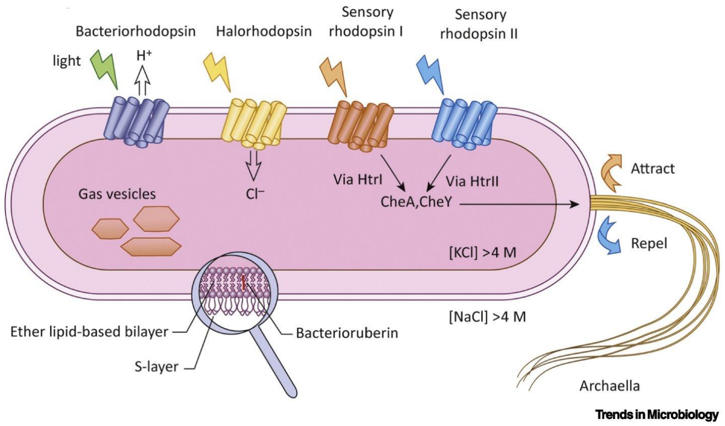
To do this, we perform a comprehensive analysis of *Halobacterium sp. GSL-19*, an organism that exhibits adaptations to extreme salinity and alkalinity.

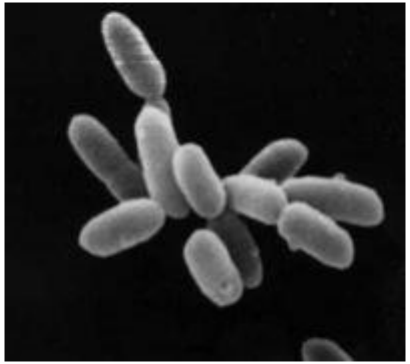
Domain: Archaea
Class: Halobacteria
Genus: *Halobacterium*
Species: *Halobacterium sp. GSL-19*
Origin: **North arm of Great Salt Lake**, Utah (41.4377°N, 112.6689°W)
Environmental Conditions: **Hypersaline** (≥300 g/L NaCl), alkaline pH



Introduction

Halobacteria are halophilic microorganisms, which means they grow in extremely high salinity environments. Comparing a halophile genome to that of other prokaryotes should give insight into microbial adaptation to extreme conditions.





Background: “what makes halophiles special?”

Compact genomes and *specialized salt tolerance mechanisms*.

- "Salt-in" strategy, where intracellular potassium compensates for external salinity, reducing osmotic stress
- Specialized protein folding pathways and acidic proteomes for structural integrity
- Modified membrane lipids enhance stability under hypersaline and alkaline conditions
- Adapted to high UV environments

(Brininger et al., 2018), (Paul et al., 2008)

2 Analyzes using GSL-19

1. **Halobacterium sp. GSL-19**: extreme halophile located in the north arm of the *Great Salt Lake*.

2. **Halanaerobium praevalen**: moderate halophile located the north arm of the *Great Salt Lake*.

3. **Halobacterium sp. NRC-1**: common halophile reference genome.

4. **Halobacterium salinarum 91 R-6**: GenBank official halophile reference genome.

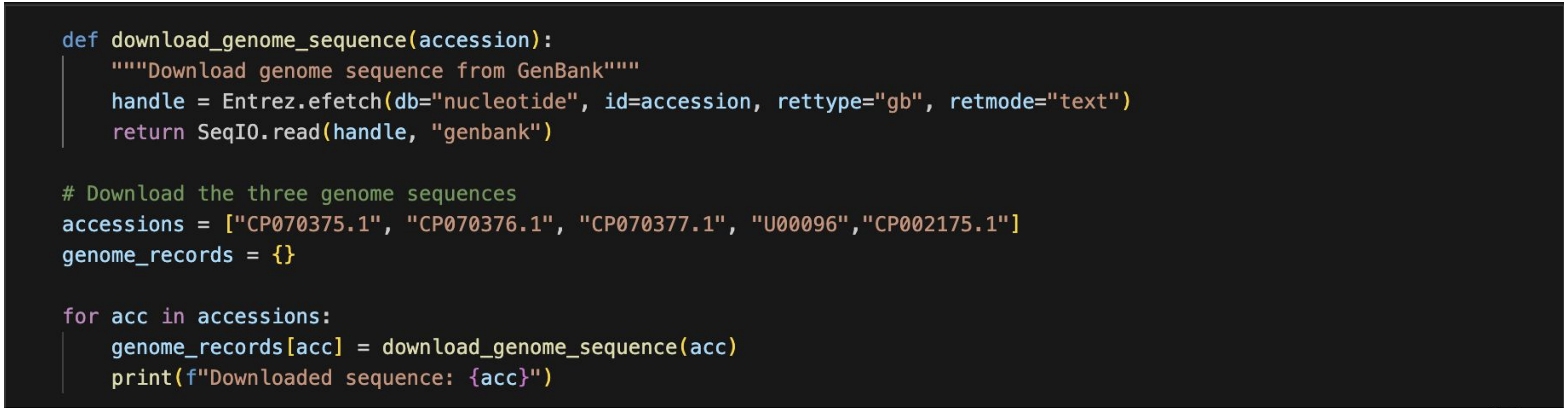
5. **Halobacterium sp. NMX12-1**: ancient halophile from the Permian Period found in a salt formation in *New Mexico*.

6. **Escherichia coli**: non-halophile.

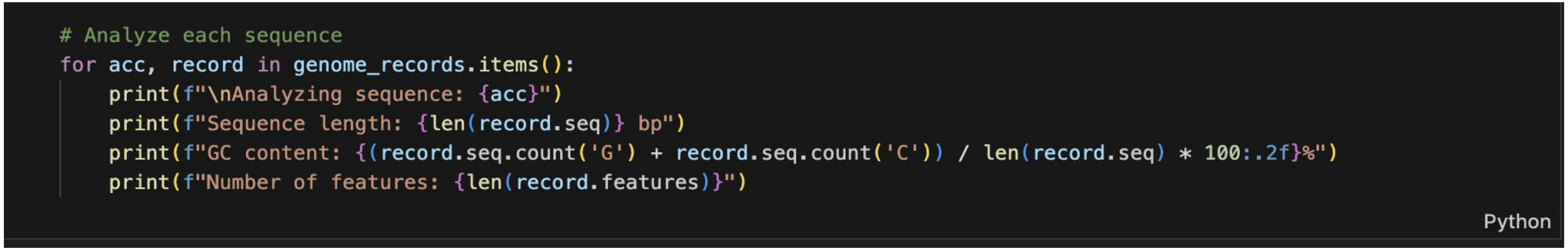
Analysis 1 with Jupyter notebooks and Galaxy

Goal: comprehensive content and feature analysis

Step 1: Sequence analysis



```python
def download_genome_sequence(accession):
    """Download genome sequence from GenBank"""
    handle = Entrez.efetch(db="nucleotide", id=accession, rettype="gb", retmode="text")
    return SeqIO.read(handle, "genbank")

# Download the three genome sequences
accessions = ["CP070375.1", "CP070376.1", "CP070377.1", "U00096","CP002175.1"]
genome_records = {}

for acc in accessions:
    genome_records[acc] = download_genome_sequence(acc)
    print(f"Downloaded sequence: {acc}")
```



```python
# Analyze each sequence
for acc, record in genome_records.items():
    print(f"\nAnalyzing sequence: {acc}")
    print(f"Sequence length: {len(record.seq)} bp")
    print(f"GC content: {(record.seq.count('G') + record.seq.count('C')) / len(record.seq) * 100:.2f}%")
    print(f"Number of features: {len(record.features)}")
```

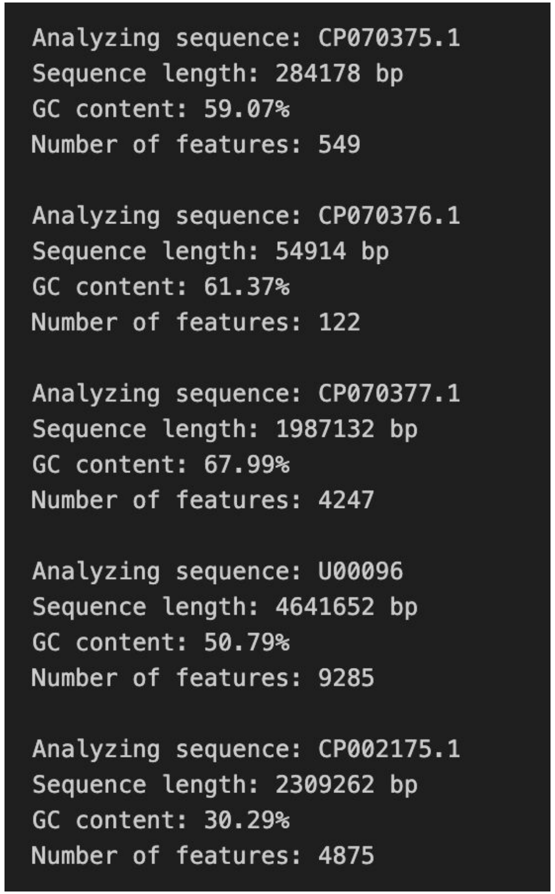
GSL-19 Plasmid 1 ➔

GSL-19 Plasmid 2 ➔

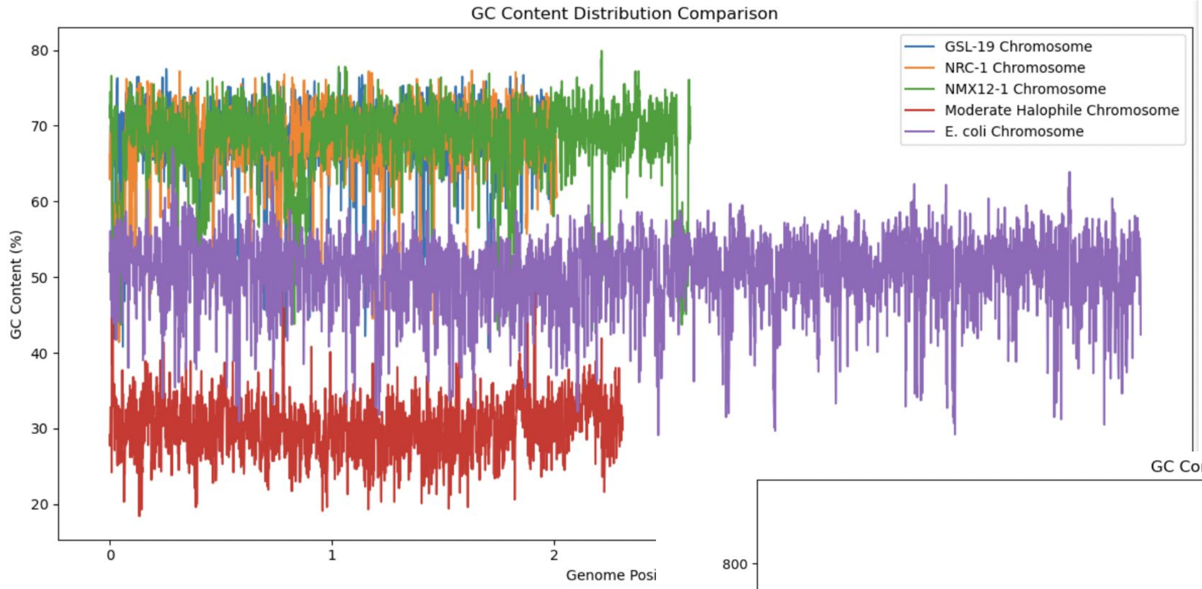
GSL-19 Chromosome ➔

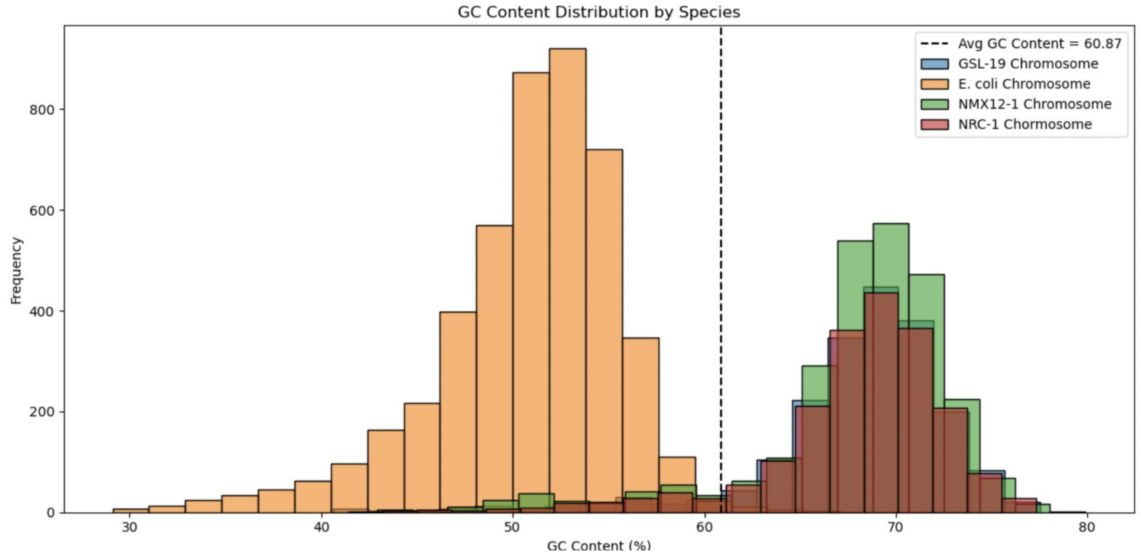
E coli. ➔

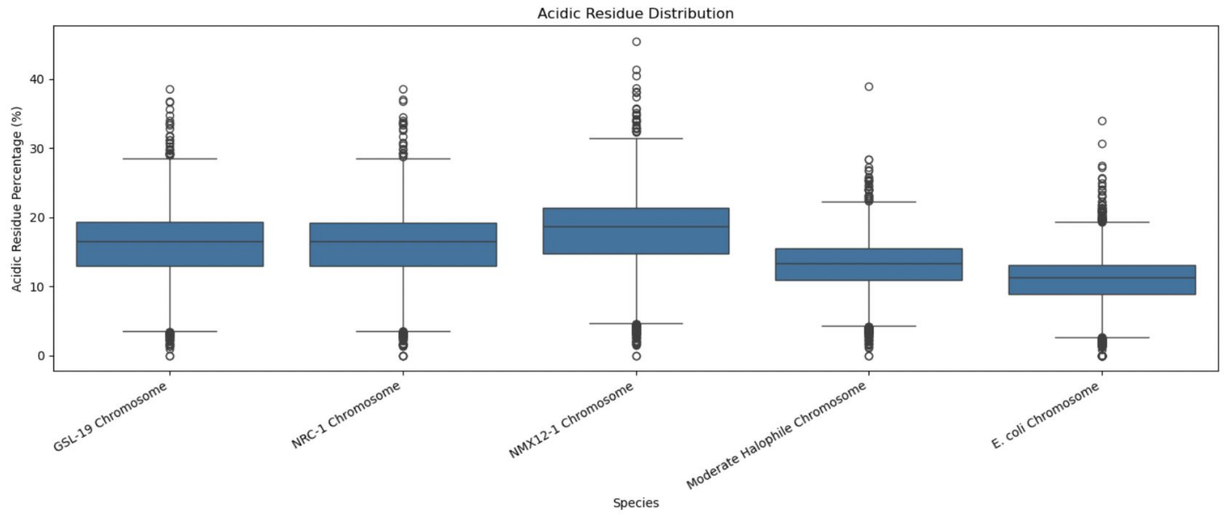
H praevalen ➔



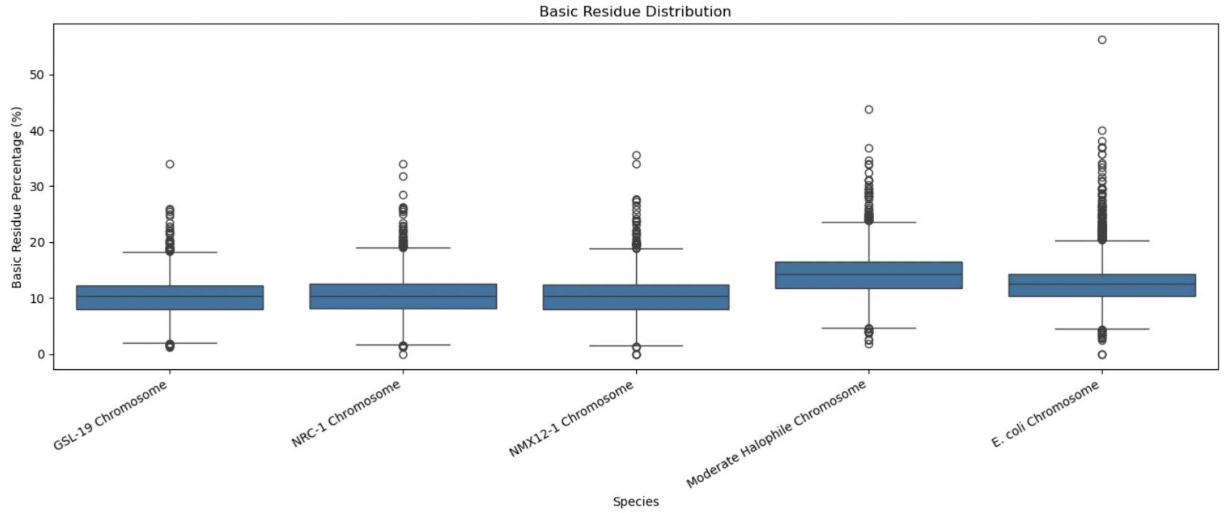
```
Analyzing sequence: CP070375.1
Sequence length: 284178 bp
GC content: 59.07%
Number of features: 549

Analyzing sequence: CP070376.1
Sequence length: 54914 bp
GC content: 61.37%
Number of features: 122

Analyzing sequence: CP070377.1
Sequence length: 1987132 bp
GC content: 67.99%
Number of features: 4247

Analyzing sequence: U00096
Sequence length: 4641652 bp
GC content: 50.79%
Number of features: 9285

Analyzing sequence: CP002175.1
Sequence length: 2309262 bp
GC content: 30.29%
Number of features: 4875
```


GC Content Distribution Comparison
GC Content (%)
Genome Posi
GSL-19 Chromosome
NRC-1 Chromosome
NMX12-1 Chromosome
Moderate Halophile Chromosome
E. coli Chromosome
80
70
60
50
40
30
20
0
1
2

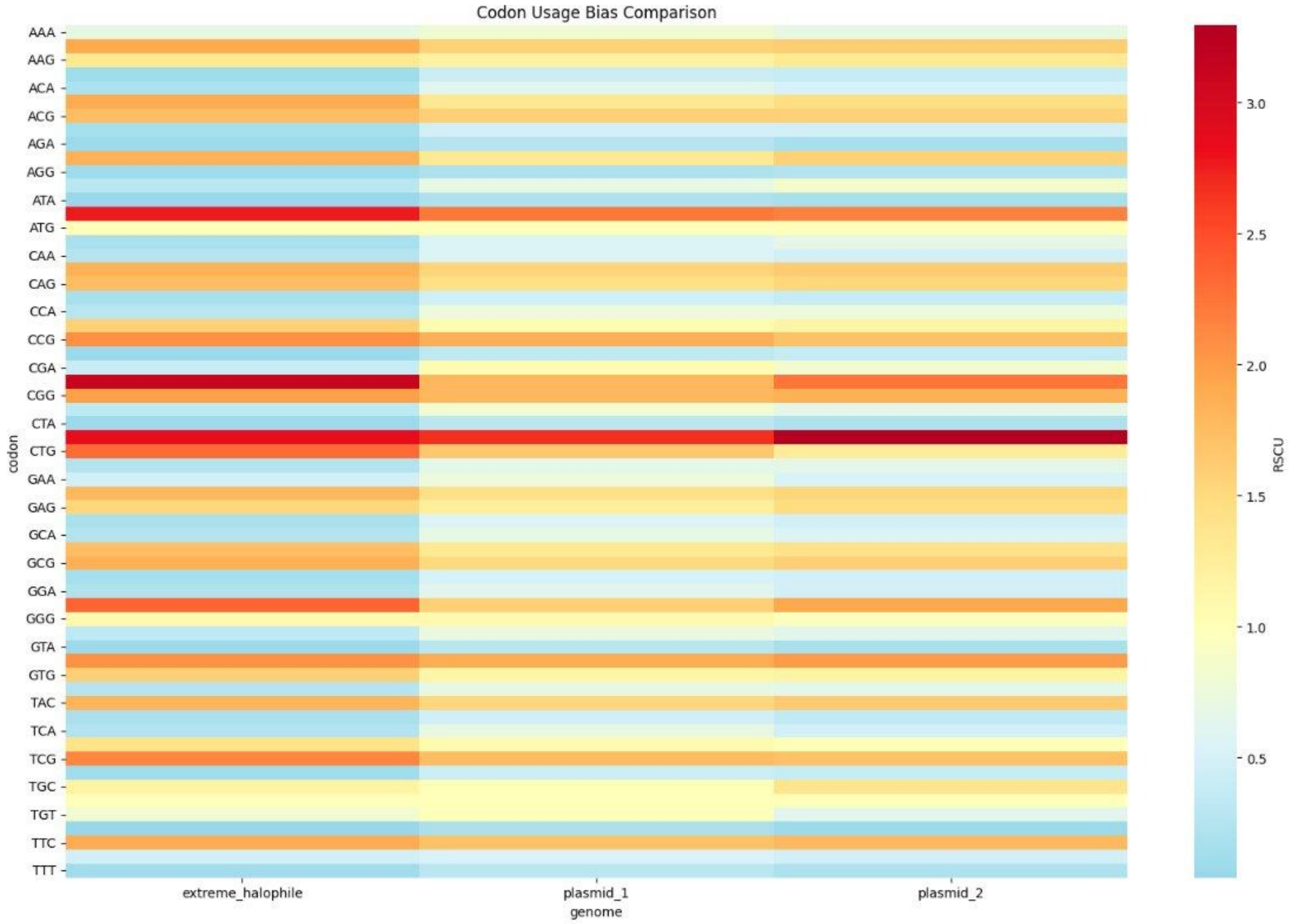

GC Content Distribution by Species
Frequency
GC Content (%)
Avg GC Content = 60.87
GSL-19 Chromosome
E. coli Chromosome
NMX12-1 Chromosome
NRC-1 Chormosome
800
600
400
200
0
30
40
50
60
70
80



Acidic Residue Distribution
Acidic Residue Percentage (%)
0
10
20
30
40
GSL-19 Chromosome
NRC-1 Chromosome
NMX12-1 Chromosome
Moderate Halophile Chromosome
E. coli Chromosome
Species

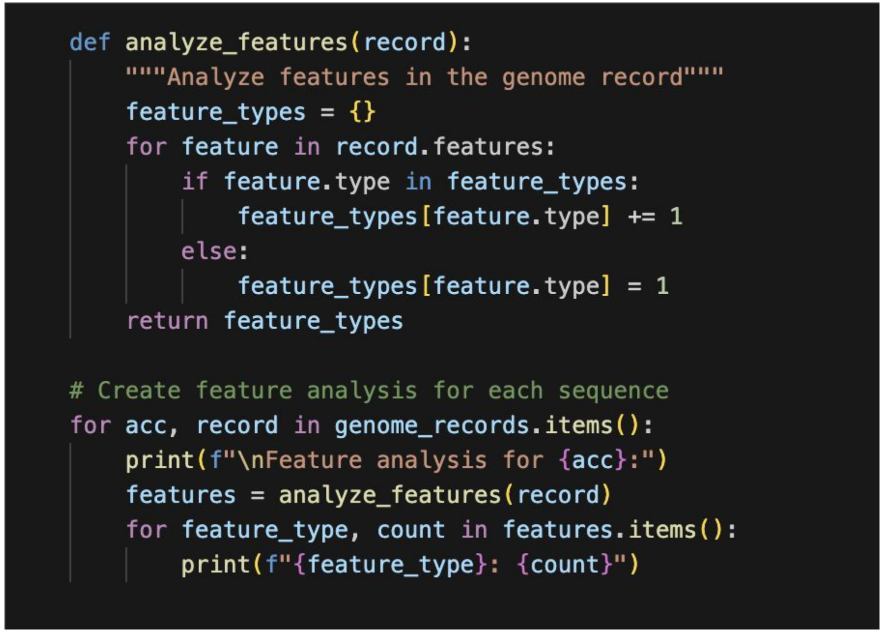


Basic Residue Distribution
Basic Residue Percentage (%)
0
10
20
30
40
50
GSL-19 Chromosome
NRC-1 Chromosome
NMX12-1 Chromosome
Moderate Halophile Chromosome
E. coli Chromosome
Species

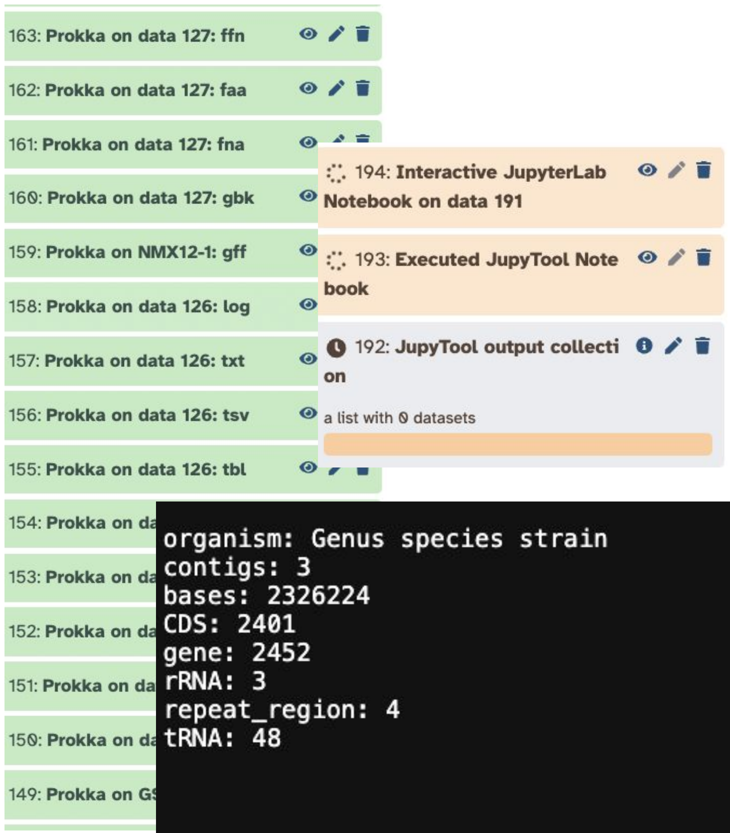


Codon Usage Bias Comparison
AAA
AAG
ACA
ACG
AGA
AGG
ATA
ATG
CAA
CAG
CCA
CCG
CGA
CGG
CTA
CTG
GAA
GAG
GCA
GCG
GGA
GGG
GTA
GTG
TAC
TCA
TCG
TGC
TGT
TTC
TTT
codon
extreme_halophile
plasmid_1
plasmid_2
genome
RSCU
3.0
2.5
2.0
1.5
1.0
0.5

Step 2: Feature analysis

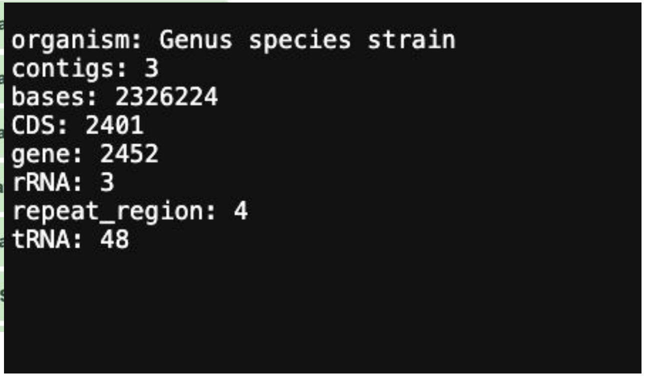


```python
def analyze_features(record):
    """Analyze features in the genome record"""
    feature_types = {}
    for feature in record.features:
        if feature.type in feature_types:
            feature_types[feature.type] += 1
        else:
            feature_types[feature.type] = 1
    return feature_types

# Create feature analysis for each sequence
for acc, record in genome_records.items():
    print(f"\nFeature analysis for {acc}:")
    features = analyze_features(record)
    for feature_type, count in features.items():
        print(f"{feature_type}: {count}")
```



163: Prokka on data 127: ffn
162: Prokka on data 127: faa
161: Prokka on data 127: fna
160: Prokka on data 127: gbk
159: Prokka on NMX12-1: gff
158: Prokka on data 126: log
157: Prokka on data 126: txt
156: Prokka on data 126: tsv
155: Prokka on data 126: tbl
154: Prokka on da
153: Prokka on da
152: Prokka on da
151: Prokka on da
150: Prokka on da
149: Prokka on G

194: Interactive JupyterLab Notebook on data 191
193: Executed JupyTool Notebook
192: JupyTool output collection
a list with 0 datasets



```
organism: Genus species strain
contigs: 3
bases: 2326224
CDS: 2401
gene: 2452
rRNA: 3
repeat_region: 4
tRNA: 48
```

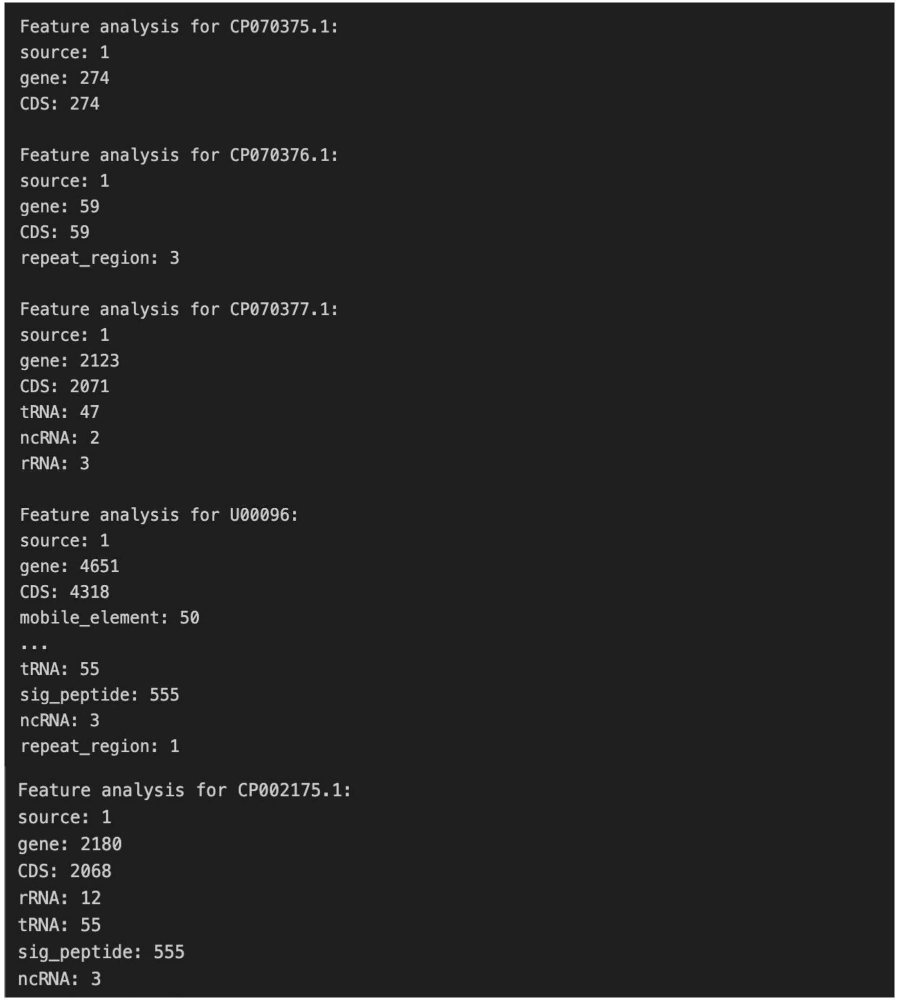
GSL-19 Plasmid 1 ➔

GSL-19 Plasmid 2 ➔

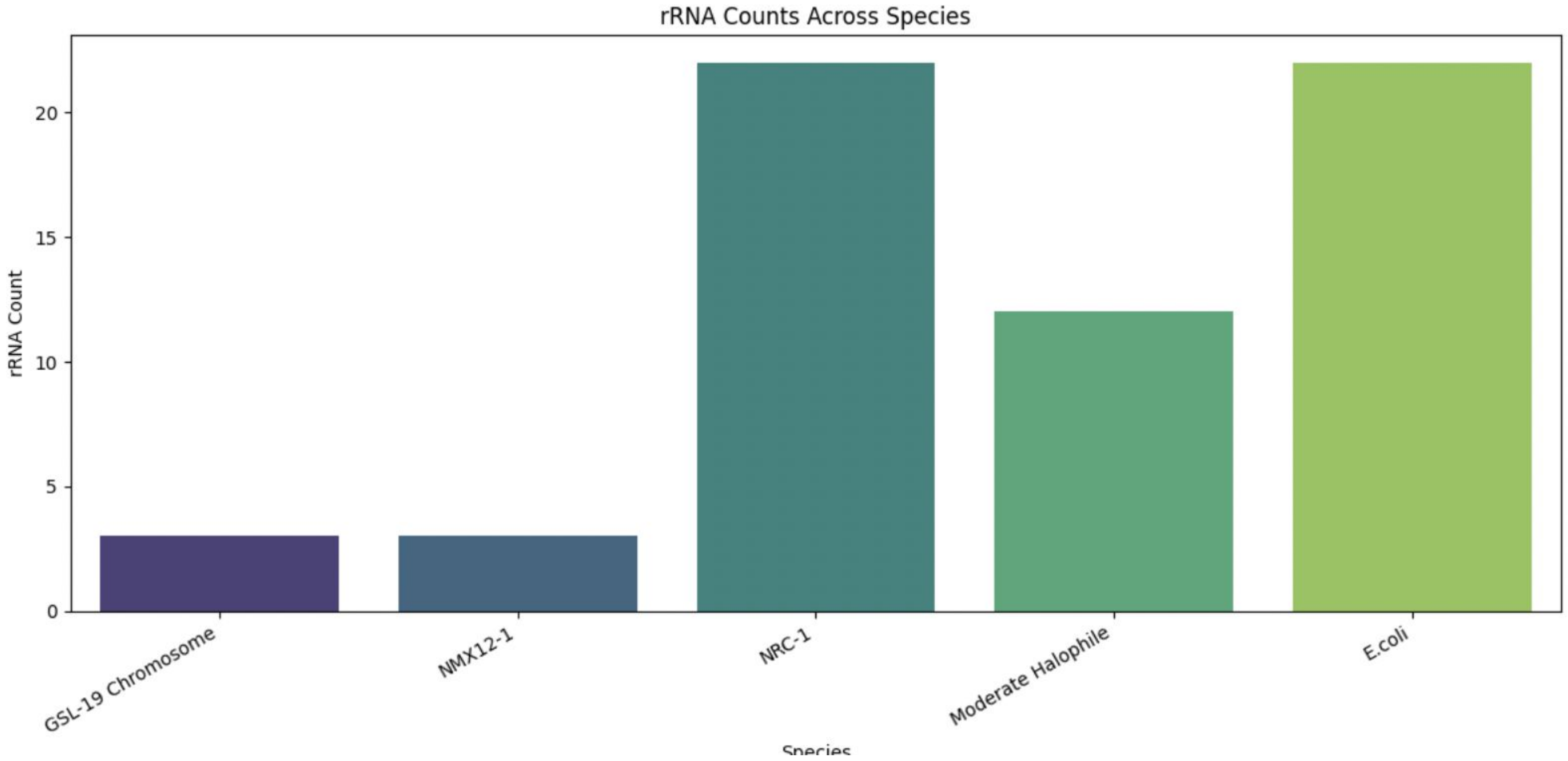
GSL-19 Chromosome ➔

E coli. ➔

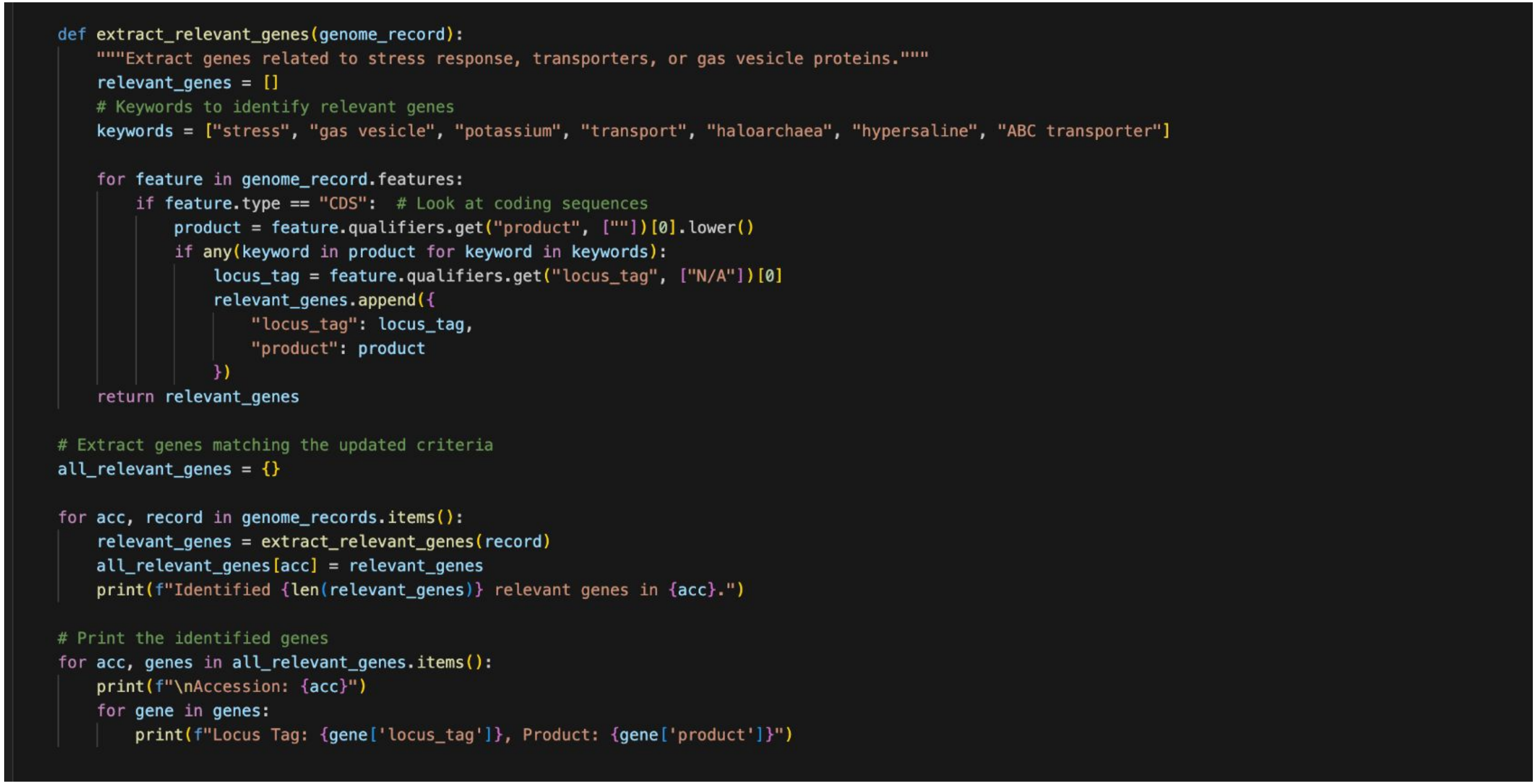
H praevalen ➔



```
Feature analysis for CP070375.1:
source: 1
gene: 274
CDS: 274

Feature analysis for CP070376.1:
source: 1
gene: 59
CDS: 59
repeat_region: 3

Feature analysis for CP070377.1:
source: 1
gene: 2123
CDS: 2071
tRNA: 47
ncRNA: 2
rRNA: 3

Feature analysis for U00096:
source: 1
gene: 4651
CDS: 4318
mobile_element: 50
...
tRNA: 55
sig_peptide: 555
ncRNA: 3
repeat_region: 1
Feature analysis for CP002175.1:
source: 1
gene: 2180
CDS: 2068
rRNA: 12
tRNA: 55
sig_peptide: 555
ncRNA: 3
```

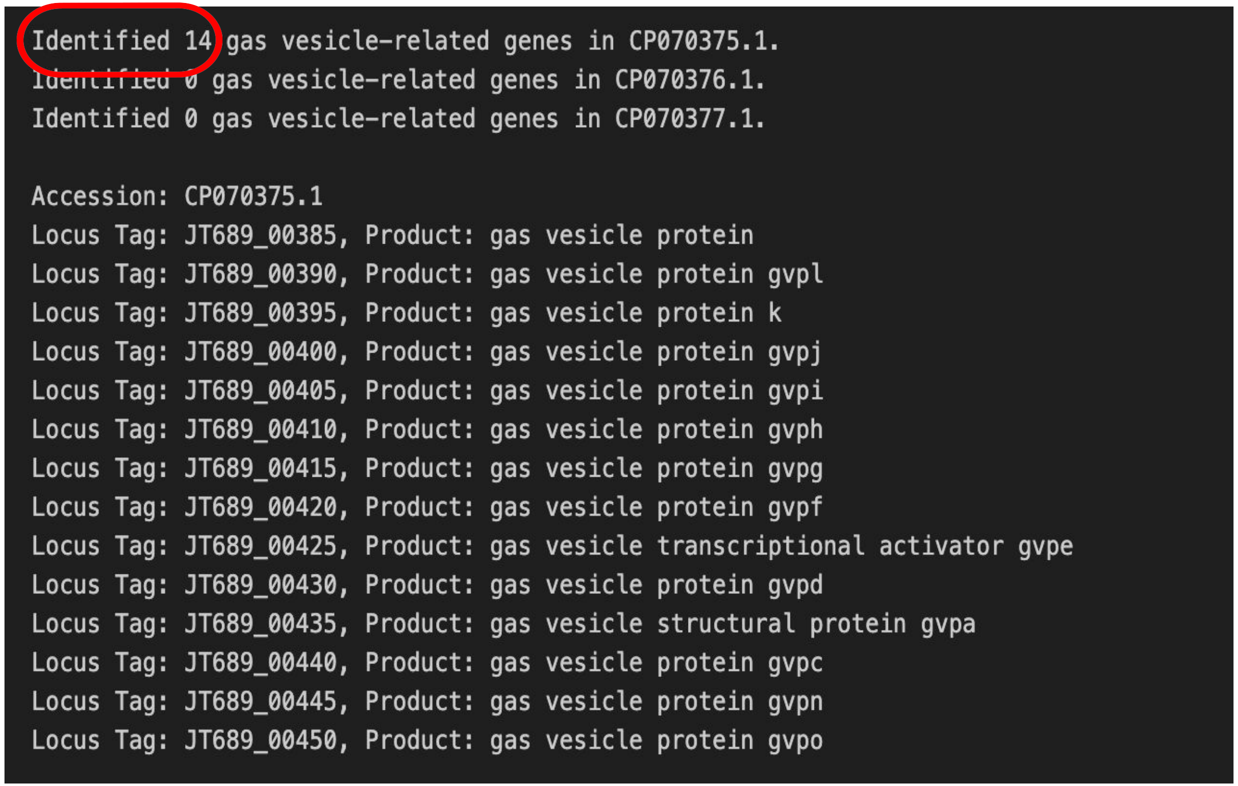

rRNA Counts Across Species
rRNA Count
0
5
10
15
20
GSL-19 Chromosome
NMX12-1
NRC-1
Moderate Halophile
E.coli
Species

Step 3: Gene extraction



```python
def extract_relevant_genes(genome_record):
    """Extract genes related to stress response, transporters, or gas vesicle proteins."""
    relevant_genes = []
    # Keywords to identify relevant genes
    keywords = ["stress", "gas vesicle", "potassium", "transport", "haloarchaea", "hypersaline", "ABC transporter"]

    for feature in genome_record.features:
        if feature.type == "CDS":  # Look at coding sequences
            product = feature.qualifiers.get("product", [""])[0].lower()
            if any(keyword in product for keyword in keywords):
                locus_tag = feature.qualifiers.get("locus_tag", ["N/A"])[0]
                relevant_genes.append({
                    "locus_tag": locus_tag,
                    "product": product
                })
    return relevant_genes

# Extract genes matching the updated criteria
all_relevant_genes = {}

for acc, record in genome_records.items():
    relevant_genes = extract_relevant_genes(record)
    all_relevant_genes[acc] = relevant_genes
    print(f"Identified {len(relevant_genes)} relevant genes in {acc}.")

# Print the identified genes
for acc, genes in all_relevant_genes.items():
    print(f"\nAccession: {acc}")
    for gene in genes:
        print(f"Locus Tag: {gene['locus_tag']}, Product: {gene['product']}")
```

We found many instances of osmotic balance genes, protein adaptations, and membrane adaptations in GSL-19.



```
Identified 14 gas vesicle-related genes in CP070375.1.
Identified 0 gas vesicle-related genes in CP070376.1.
Identified 0 gas vesicle-related genes in CP070377.1.

Accession: CP070375.1
Locus Tag: JT689_00385, Product: gas vesicle protein
Locus Tag: JT689_00390, Product: gas vesicle protein gvpl
Locus Tag: JT689_00395, Product: gas vesicle protein k
Locus Tag: JT689_00400, Product: gas vesicle protein gvpj
Locus Tag: JT689_00405, Product: gas vesicle protein gvpi
Locus Tag: JT689_00410, Product: gas vesicle protein gvph
Locus Tag: JT689_00415, Product: gas vesicle protein gvpg
Locus Tag: JT689_00420, Product: gas vesicle protein gvpf
Locus Tag: JT689_00425, Product: gas vesicle transcriptional activator gvpe
Locus Tag: JT689_00430, Product: gas vesicle protein gvpd
Locus Tag: JT689_00435, Product: gas vesicle structural protein gvpa
Locus Tag: JT689_00440, Product: gas vesicle protein gvpc
Locus Tag: JT689_00445, Product: gas vesicle protein gvpn
Locus Tag: JT689_00450, Product: gas vesicle protein gvpo
```

Gas vesicle proteins

Gas vesicles are a **crucial adaptation** in halophilic archaea that allow these microorganisms to regulate their buoyancy in hypersaline environments.

Gvp’s can be used as contrast agents, delivery carriers, and immunology boosters for disease prevention, diagnosis, and treatment. Largely due to their tiny size, strong stability and non-toxic advantages.

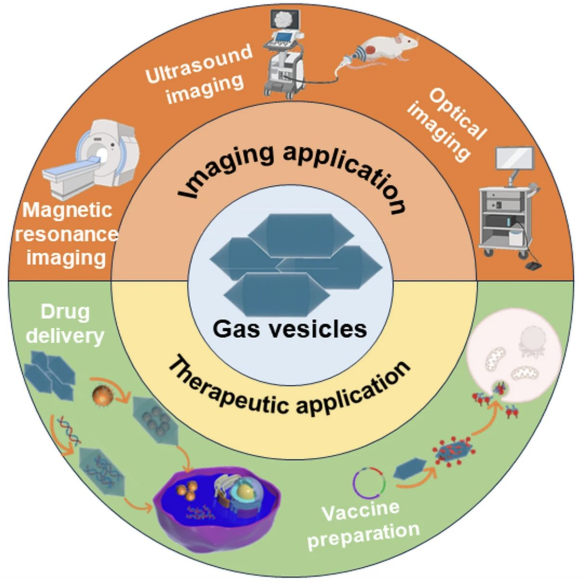
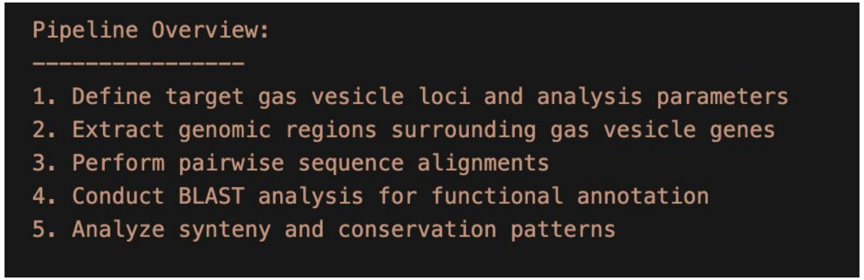
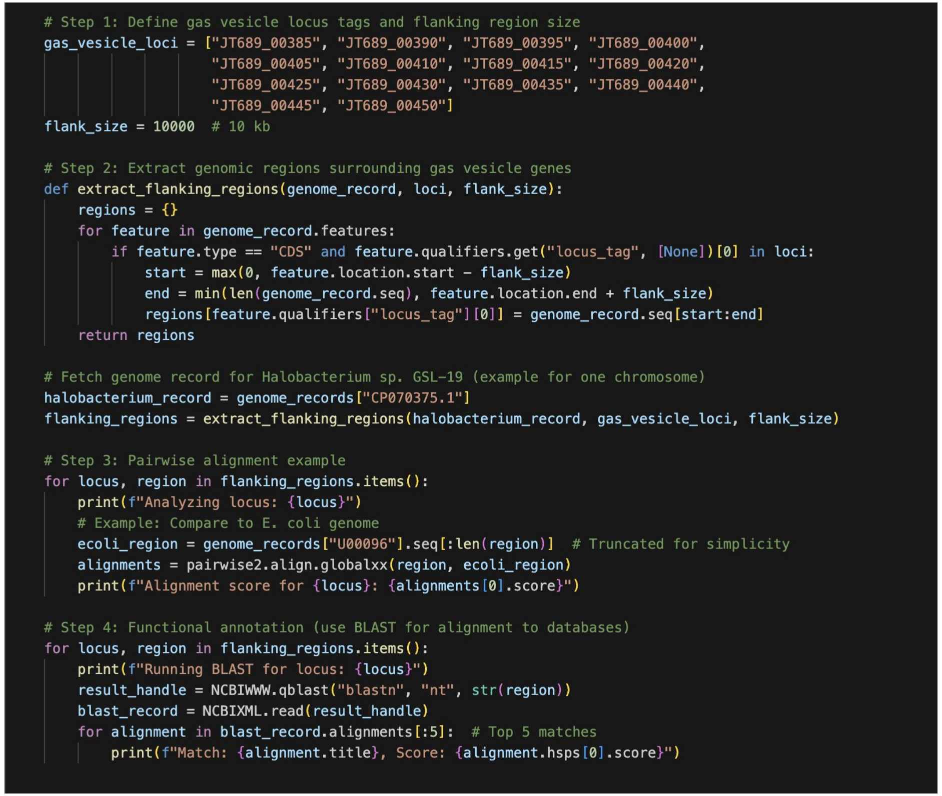


Figure 1 (Feng et al. 2024), (Jost and Pfeifer 2022)

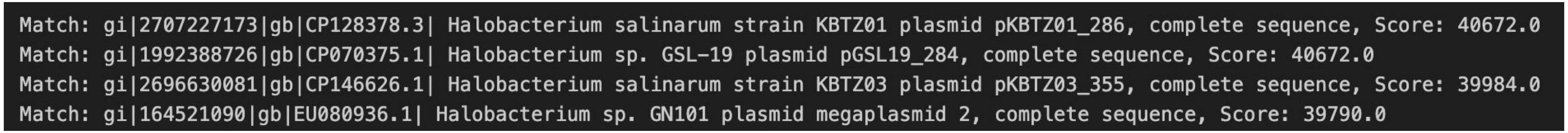
Step 4: Analysis of gas vesicle proteins using BLAST



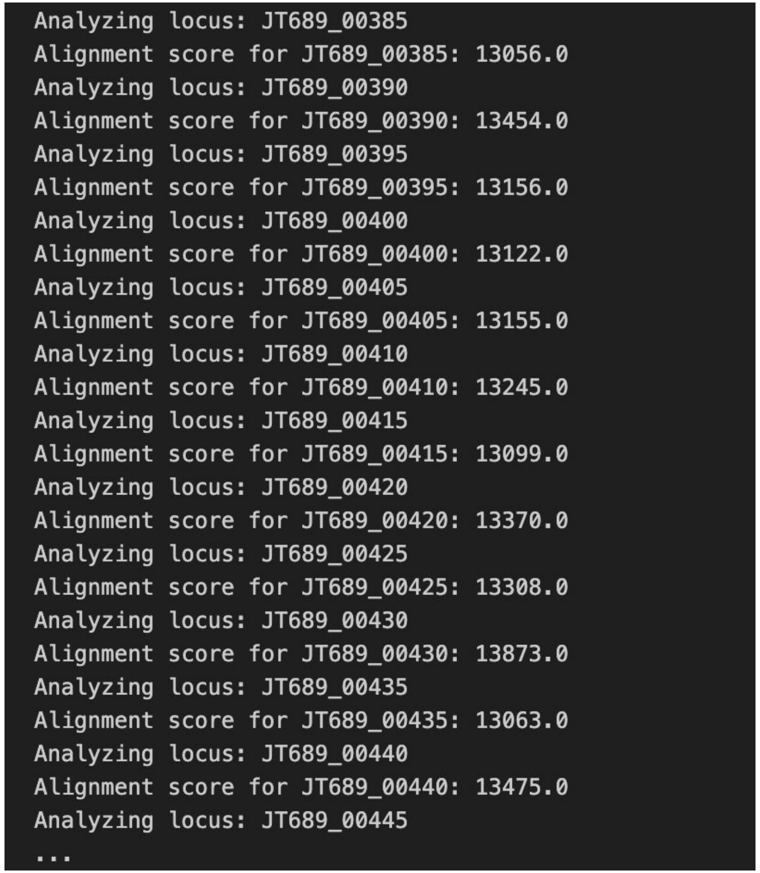
```
Pipeline Overview:
-----------------
1. Define target gas vesicle loci and analysis parameters
2. Extract genomic regions surrounding gas vesicle genes
3. Perform pairwise sequence alignments
4. Conduct BLAST analysis for functional annotation
5. Analyze synteny and conservation patterns
```



```python
# Step 1: Define gas vesicle locus tags and flanking region size
gas_vesicle_loci = ["JT689_00385", "JT689_00390", "JT689_00395", "JT689_00400",
                    "JT689_00405", "JT689_00410", "JT689_00415", "JT689_00420",
                    "JT689_00425", "JT689_00430", "JT689_00435", "JT689_00440",
                    "JT689_00445", "JT689_00450"]
flank_size = 10000  # 10 kb

# Step 2: Extract genomic regions surrounding gas vesicle genes
def extract_flanking_regions(genome_record, loci, flank_size):
    regions = {}
    for feature in genome_record.features:
        if feature.type == "CDS" and feature.qualifiers.get("locus_tag", [None])[0] in loci:
            start = max(0, feature.location.start - flank_size)
            end = min(len(genome_record.seq), feature.location.end + flank_size)
            regions[feature.qualifiers["locus_tag"][0]] = genome_record.seq[start:end]
    return regions

# Fetch genome record for Halobacterium sp. GSL-19 (example for one chromosome)
halobacterium_record = genome_records["CP070375.1"]
flanking_regions = extract_flanking_regions(halobacterium_record, gas_vesicle_loci, flank_size)

# Step 3: Pairwise alignment example
for locus, region in flanking_regions.items():
    print(f"Analyzing locus: {locus}")
    # Example: Compare to E. coli genome
    ecoli_region = genome_records["U00096"].seq[:len(region)]  # Truncated for simplicity
    alignments = pairwise2.align.globalxx(region, ecoli_region)
    print(f"Alignment score for {locus}: {alignments[0].score}")

# Step 4: Functional annotation (use BLAST for alignment to databases)
for locus, region in flanking_regions.items():
    print(f"Running BLAST for locus: {locus}")
    result_handle = NCBIWWW.qblast("blastn", "nt", str(region))
    blast_record = NCBIXML.read(result_handle)
    for alignment in blast_record.alignments[:5]:  # Top 5 matches
        print(f"Match: {alignment.title}, Score: {alignment.hsps[0].score}")
```



```
Match: gi|2707227173|gb|CP128378.3| Halobacterium salinarum strain KBTZ01 plasmid pKBTZ01_286, complete sequence, Score: 40672.0
Match: gi|1992388726|gb|CP070375.1| Halobacterium sp. GSL-19 plasmid pGSL19_284, complete sequence, Score: 40672.0
Match: gi|2696630081|gb|CP146626.1| Halobacterium salinarum strain KBTZ03 plasmid pKBTZ03_355, complete sequence, Score: 39984.0
Match: gi|164521090|gb|EU080936.1| Halobacterium sp. GN101 plasmid megaplasmid 2, complete sequence, Score: 39790.0
```



```
 Analyzing locus: JT689_00385
 Alignment score for JT689_00385: 13056.0
 Analyzing locus: JT689_00390
 Alignment score for JT689_00390: 13454.0
 Analyzing locus: JT689_00395
 Alignment score for JT689_00395: 13156.0
 Analyzing locus: JT689_00400
 Alignment score for JT689_00400: 13122.0
 Analyzing locus: JT689_00405
 Alignment score for JT689_00405: 13155.0
 Analyzing locus: JT689_00410
 Alignment score for JT689_00410: 13245.0
 Analyzing locus: JT689_00415
 Alignment score for JT689_00415: 13099.0
 Analyzing locus: JT689_00420
 Alignment score for JT689_00420: 13370.0
 Analyzing locus: JT689_00425
 Alignment score for JT689_00425: 13308.0
 Analyzing locus: JT689_00430
 Alignment score for JT689_00430: 13873.0
 Analyzing locus: JT689_00435
 Alignment score for JT689_00435: 13063.0
 Analyzing locus: JT689_00440
 Alignment score for JT689_00440: 13475.0
 Analyzing locus: JT689_00445
 ...
```

BLAST hits reveal interesting patterns: most matches are to plasmid sequences, with a consistent presence on large plasmids. Suggests potential for horizontal gene transfer.

1. GSL-19 (consistently highest scores)
2. H. salinarum strains KBTZ02 and KBTZ01 (very similar scores)
3. H. salinarum strain KBTZ03 (slightly lower scores)
4. Halobacterium sp. GN101 (consistently lowest scores)

Analysis 2 Genome Comparison in Galaxy

Goal: compare gas vesicle protein genes across
6 bacteria species/strains

Gas vesicle protein variations

GvpA and C are particularly important as they (respectively) form the ribs of the vesicle and stabilize the structure, while **gvpN and O** are found in lower amounts.

The functions of **gvpEFGHIJKLM** are not completely understood, but some are required.

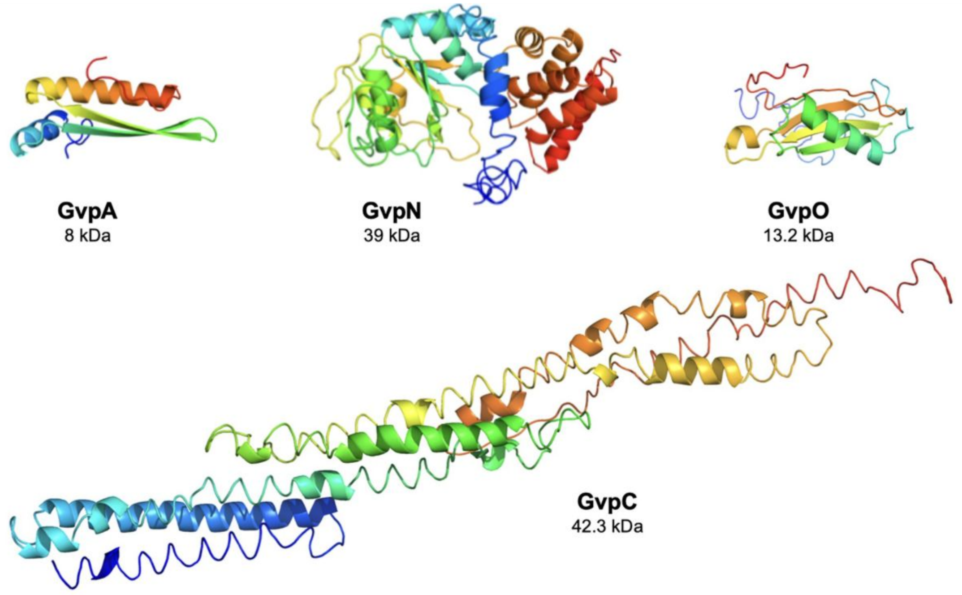
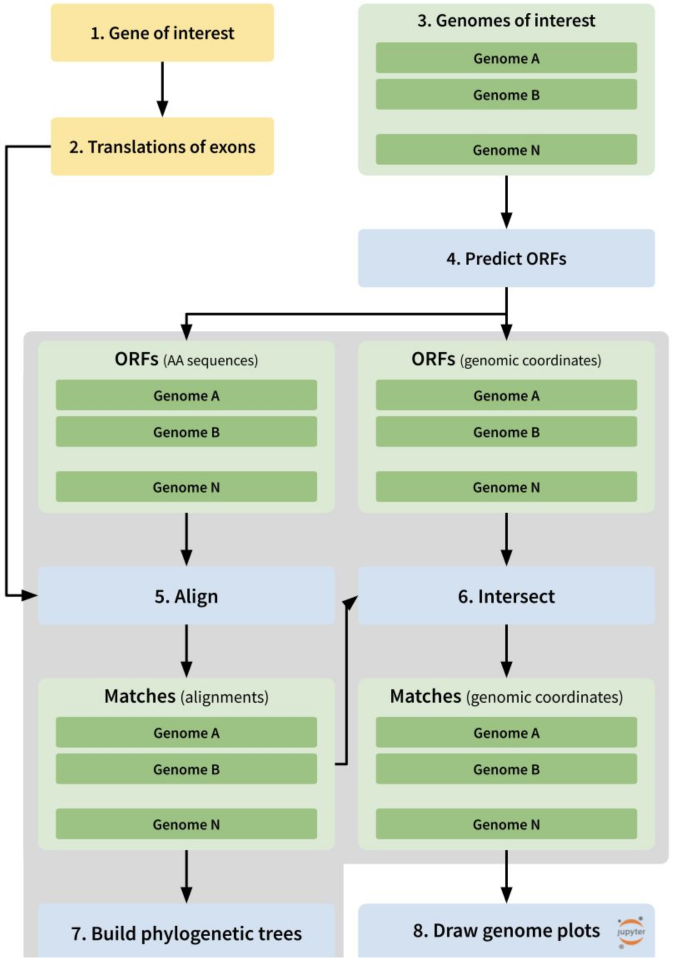


Figure 1 (Jost and Pfeifer 2022)

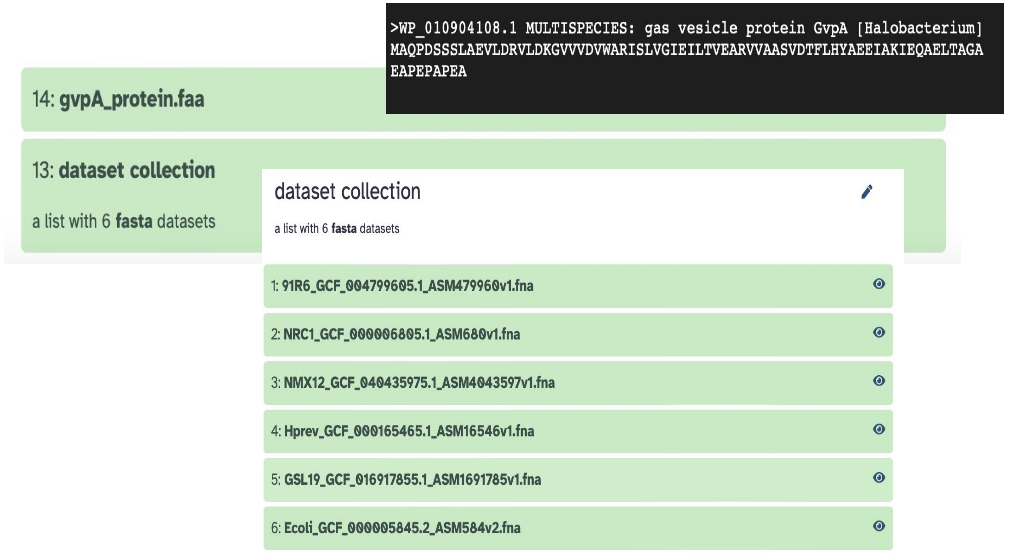
Comparative Gene Analysis Workflow

1. upload **gvpA** amino acid translation
2. upload **the 6 species** genomes as a dataset
3. extract ORF's from the genomes
4. create a diamond database
5. run tutorial workflow
6. Jupyter notebooks visuals

Output: phylogenetic trees and gene location graphs



Step 1 and 2: Upload data



14: **gvpA_protein.faa**

```
>WP_010904108.1 MULTISPECIES: gas vesicle protein GvpA [Halobacterium]
MAQPDSSSLAEVLDRVLDKGVVVDVWARISLVGIEILTVEARVVAASVDTFLHYAEEIAKIEQAELTAGA
EAPEPAPEA
```

13: **dataset collection**

a list with 6 **fasta** datasets

dataset collection

a list with 6 **fasta** datasets

1: 91R6_GCF_004799605.1_ASM479960v1.fna

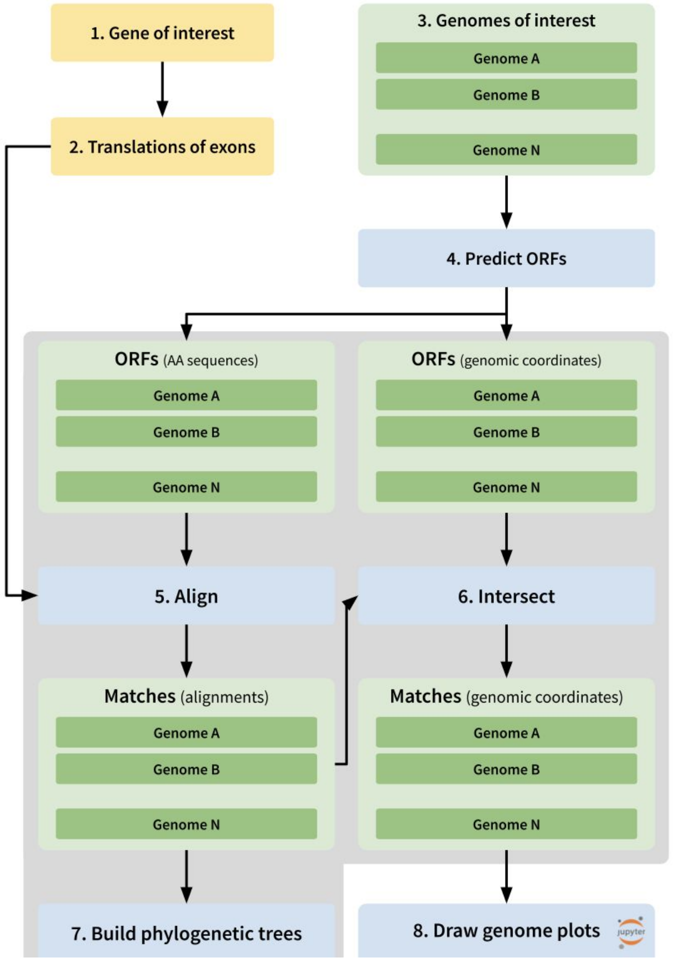
2: NRC1_GCF_000006805.1_ASM680v1.fna

3: NMX12_GCF_040435975.1_ASM4043597v1.fna

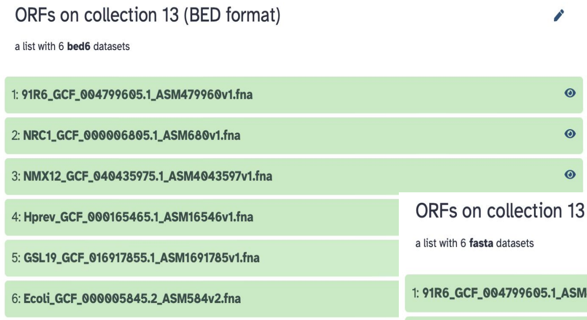
4: Hprev_GCF_000165465.1_ASM16546v1.fna

5: GSL19_GCF_016917855.1_ASM1691785v1.fna

6: Ecoli_GCF_000005845.2_ASM584v2.fna



Step 3: ORFiPy



ORFs on collection 13 (BED format)

a list with 6 **bed6** datasets

1: 91R6_GCF_004799605.1_ASM479960v1.fna

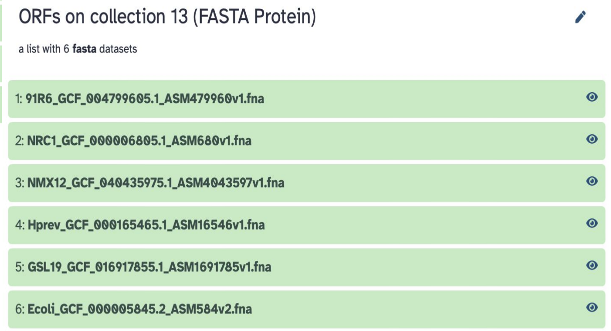
2: NRC1_GCF_000006805.1_ASM680v1.fna

3: NMX12_GCF_040435975.1_ASM4043597v1.fna

4: Hprev_GCF_000165465.1_ASM16546v1.fna

5: GSL19_GCF_016917855.1_ASM1691785v1.fna

6: Ecoli_GCF_000005845.2_ASM584v2.fna



ORFs on collection 13 (FASTA Protein)

a list with 6 **fasta** datasets

1: 91R6_GCF_004799605.1_ASM479960v1.fna

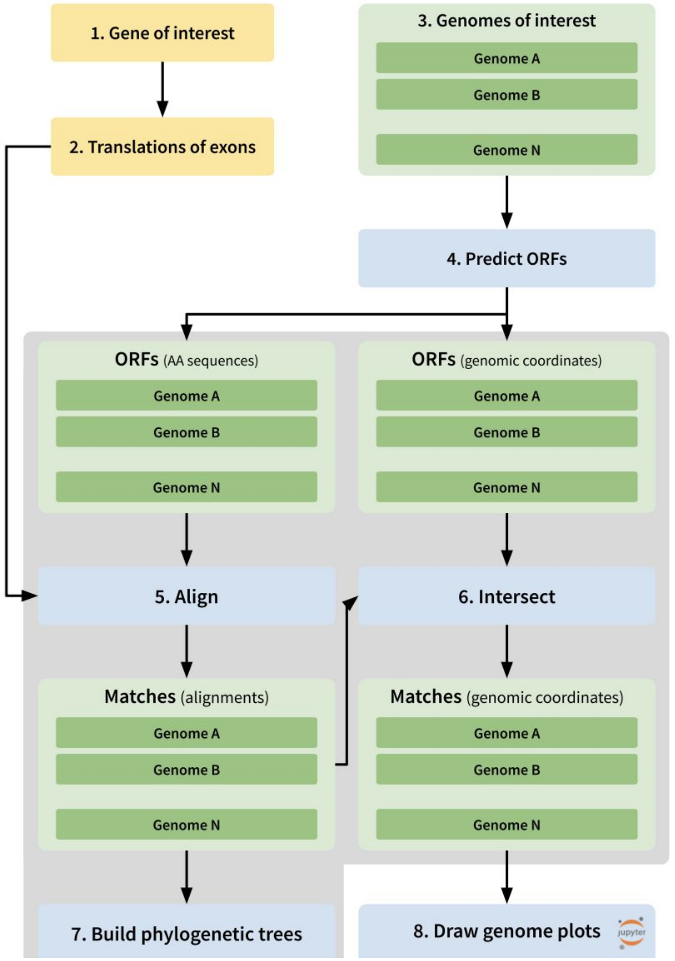
2: NRC1_GCF_000006805.1_ASM680v1.fna

3: NMX12_GCF_040435975.1_ASM4043597v1.fna

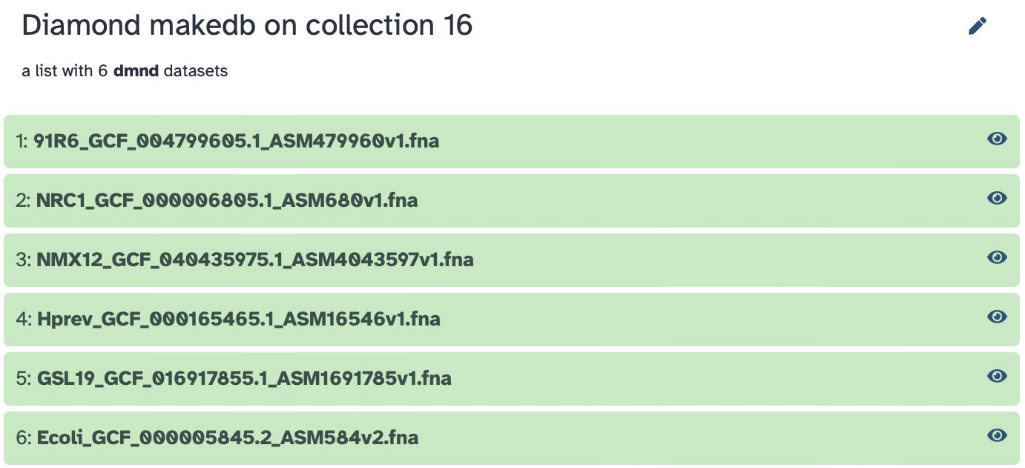
4: Hprev_GCF_000165465.1_ASM16546v1.fna

5: GSL19_GCF_016917855.1_ASM1691785v1.fna

6: Ecoli_GCF_000005845.2_ASM584v2.fna



Step 4: Diamond makedb


Diamond makedb on collection 16

a list with 6 **dmnd** datasets

1: **91R6_GCF_004799605.1_ASM479960v1.fna**

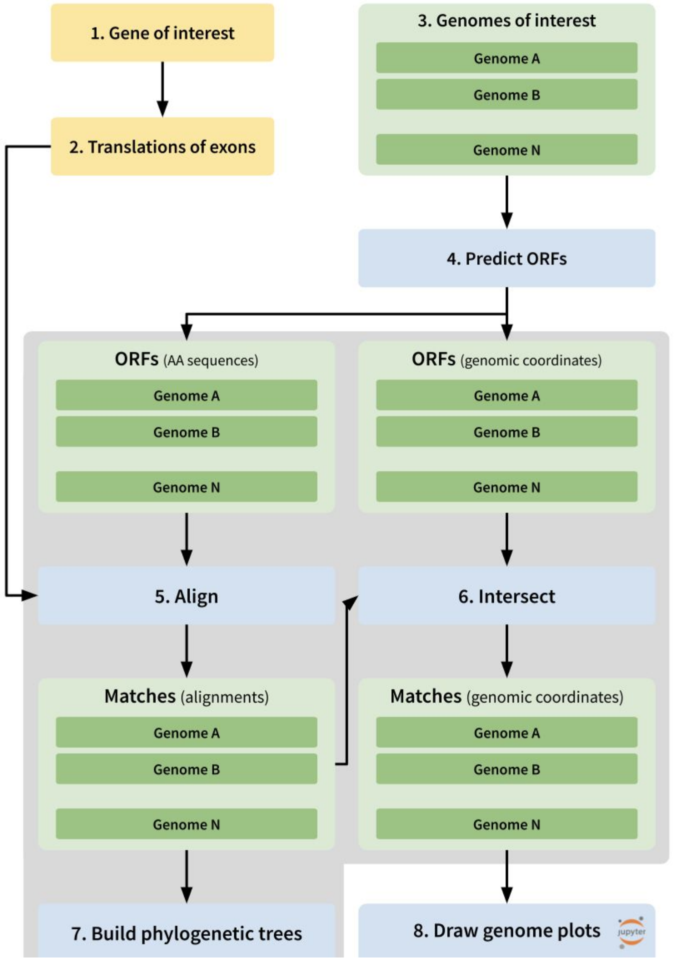
2: **NRC1_GCF_000006805.1_ASM680v1.fna**

3: **NMX12_GCF_040435975.1_ASM4043597v1.fna**

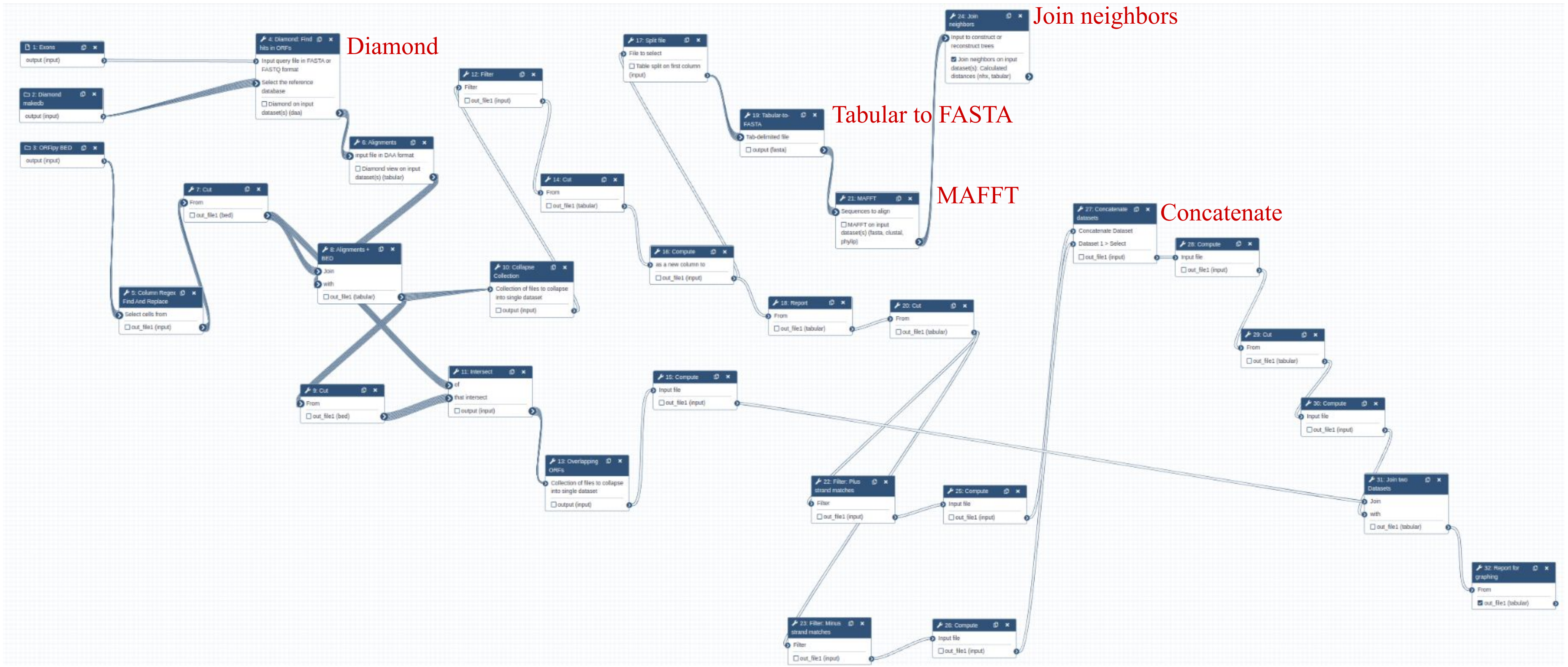
4: **Hprev_GCF_000165465.1_ASM16546v1.fna**

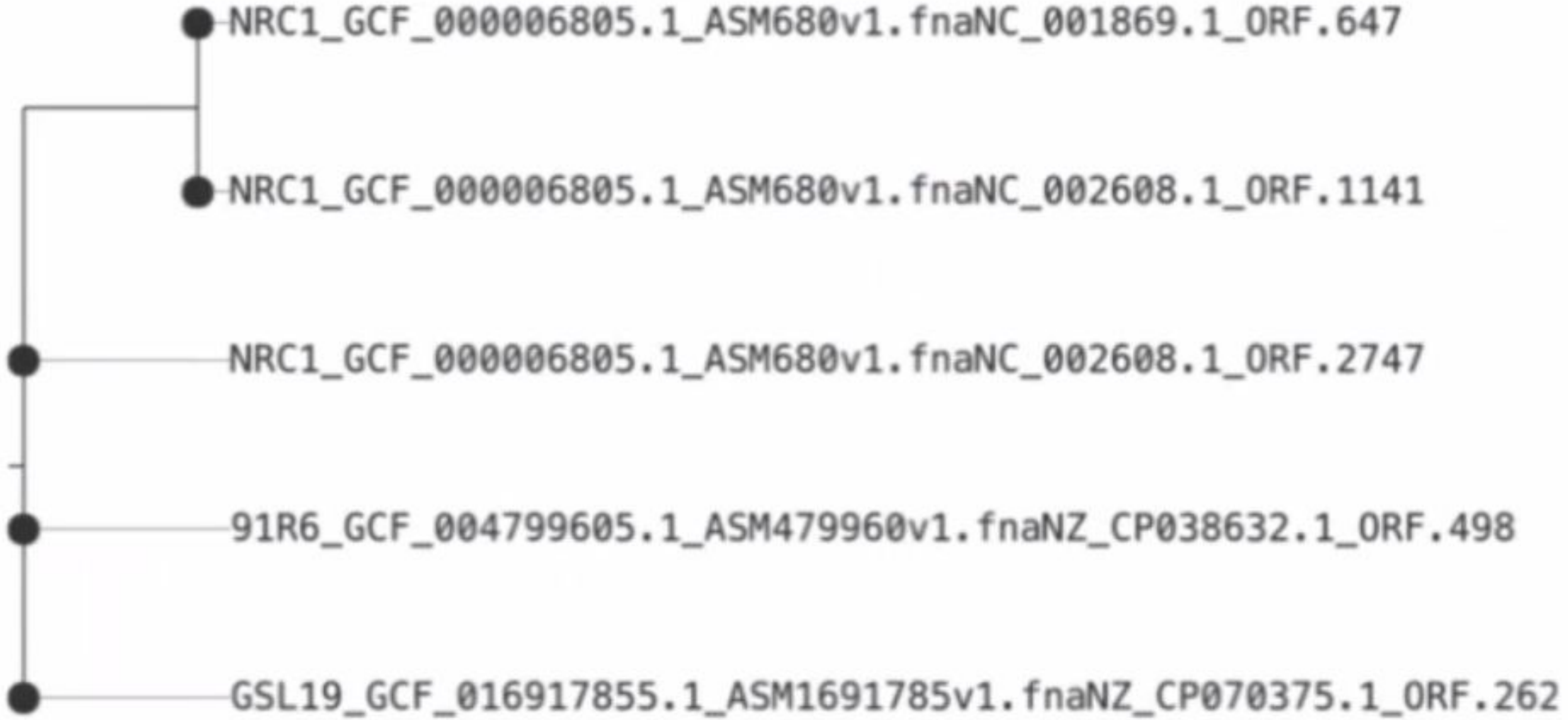
5: **GSL19_GCF_016917855.1_ASM1691785v1.fna**

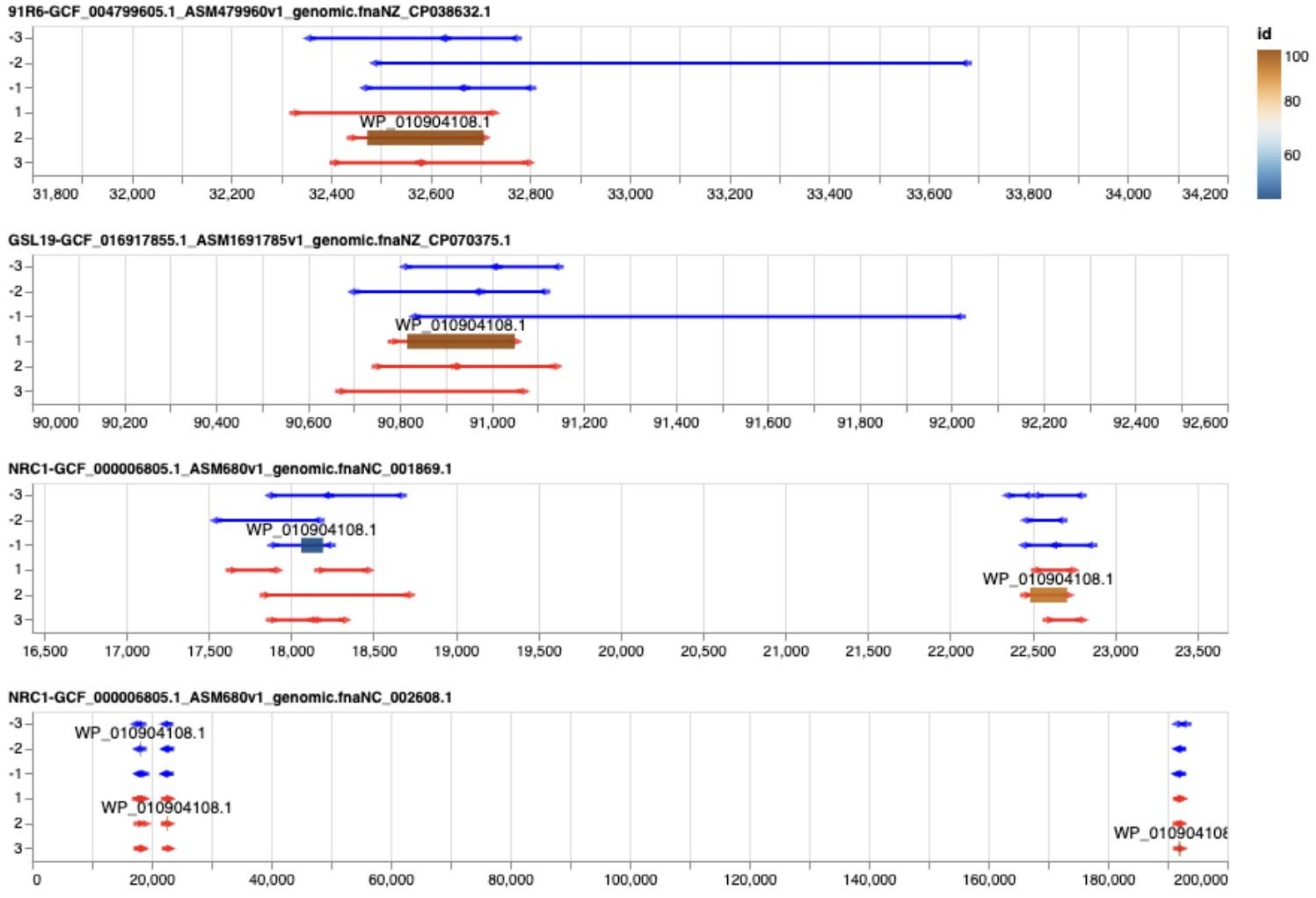
6: **Ecoli_GCF_000005845.2_ASM584v2.fna**



Run Workflow




NRC1_GCF_000006805.1_ASM680v1.fnaNC_001869.1_ORF.647
NRC1_GCF_000006805.1_ASM680v1.fnaNC_002608.1_ORF.1141
NRC1_GCF_000006805.1_ASM680v1.fnaNC_002608.1_ORF.2747
91R6_GCF_004799605.1_ASM479960v1.fnaNZ_CP038632.1_ORF.498
GSL19_GCF_016917855.1_ASM1691785v1.fnaNZ_CP070375.1_ORF.262


91R6-GCF_004799605.1_ASM479960v1_genomic.fnaNZ_CP038632.1
-3
-2
-1
1
2
3
WP_010904108.1
31,800
32,000
32,200
32,400
32,600
32,800
33,000
33,200
33,400
33,600
33,800
34,000
34,200
id
100
80
60
GSL19-GCF_016917855.1_ASM1691785v1_genomic.fnaNZ_CP070375.1
WP_010904108.1
90,000
90,200
90,400
90,600
90,800
91,000
91,200
91,400
91,600
91,800
92,000
92,200
92,400
92,600
NRC1-GCF_000006805.1_ASM680v1_genomic.fnaNC_001869.1
WP_010904108.1
WP_010904108.1
16,500
17,000
17,500
18,000
18,500
19,000
19,500
20,000
20,500
21,000
21,500
22,000
22,500
23,000
23,500
NRC1-GCF_000006805.1_ASM680v1_genomic.fnaNC_002608.1
WP_010904108.1
WP_010904108.1
WP_010904108
0
20,000
40,000
60,000
80,000
100,000
120,000
140,000
160,000
180,000
200,000

Conclusion

Gas vesicle proteins are one of the features that make GSL-19 and other halophiles so special!

GSL-19 is a particularly interesting halophile. It exhibits strong halophilic characteristics such as high GC content/acidity, abundant genes showcasing osmotic balance and membrane adaptations, and (of course) gas vesicle protein genes making it an ideal halophile to study.

Implications

The study of *Halobacterium sp. GSL-19* provides valuable insights into extremophile adaptations.

- **Biotechnology**: enzymes and biopolymers stable in high salt.
- **Astrobiology**: models for extraterrestrial life.
- **Evolutionary Biology**: understanding genome reduction and adaptation mechanisms.