2025-07-16_workflow

July 21, 2025

1 Protein folding

vpSAT can be found here: https://github.com/jnoms/vpSAT

[ ]:
```
# Make MSAs for colabfold
SEARCH=colabfold_search
REF_DB=path/to/colabfold
$SEARCH infrared_megaphages_clust_rep_seq.fasta $REF_DB msas

# Fold
nextflow run $CODE/vpSAT/main.nf \
    -profile lw \
    --entry_point "colabfold" \
    --in_files "msas/*a3m" \
    -resume \
    --COLABFOLD_num_recycles 3 \
    --COLABFOLD_num_models 3 \
    --COLABFOLD_stop_at_score 80 \
    --COLABFOLD_stop_at_score_below 40
```

2 DALI searches

[ ]:
```
# Import the viral 2Hs into DALI database format
mkdir -p query_db

for FILE in query_structures/*pdb ; do

ABREV=$(basename ${FILE%.pdb})

import.pl \
--pdbfile $FILE \
--pdbid $ABREV \
--dat query_db \
--clean

done
```

```
# Running the DALI searches.
# The phage databases are split into 3 folders, and we use an SGE array to run three searches.
QUERY_DB=query_db
TARGET_DB=path/to/dali_db/batch${SLURM_ARRAY_TASK_ID}

$CODE/vpSAT/bin/dali.sh \
-q $QUERY_DB \
-t $TARGET_DB \
-o result/res_${SLURM_ARRAY_TASK_ID} \
-n 28

# Parse the DALI results
KEY=path/to/dali_db_key.txt

for FILE in result/*/*txt ; do

BASE=${FILE%.txt}

sat.py aln_parse_dali \
-a $FILE \
-s $KEY \
-o ${BASE}.m8

done
```

[ ]:
```
# Sequence searches (jackhmmer, mmseqs2)
```

[ ]:
```
# First, convert the phage 2H structure files into fasta files
STRUCS=path/to/query_structures

mkdir -p query_fastas

for STRUC in $STRUCS/*pdb ; do

BASE=$(basename ${STRUC%.pdb})

sat.py struc_to_seq \
-s $STRUC \
-H $BASE \
-o query_fastas/${BASE}.fasta

done

cat query_fastas/*fasta > query.fasta

# Now, run jackhmmer
```

```
conda activate hmmer

jackhmmer \
--tblout ligT_search.hmmer.m8.tmp \
--cpu 5 \
-N 3 \
query.fasta \
path/to/all_phage_seqs.fasta


# Parse the jackhmmer output into a better file...
# colnames are target,query,evalue,bits
awk 'NR > 3 {print $1, $3, $5, $6}' FS=" " OFS="\t" ligT_search.hmmer.m8.tmp >␣
 ↪ligT_search.hmmer.m8.tmp2

# There are a few lines starting with #, need to remove them
grep -v "^#" ligT_search.hmmer.m8.tmp2 > ligT_search.hmmer.m8


# Filter on evalue <= 0.001
sat.py aln_filter \
-a ligT_search.hmmer.m8 \
-o ligT_search.hmmer.eval0.001.m8 \
-f "target,query,evalue,bits" \
-x evalue \
-m 0 \
-M 0.001

# MMseqs2 as a comparitor
#-----------------#
# conda activate vpSAT
TARGET=path/to/all_phage_seqs.fasta


mkdir -p tmp

mmseqs easy-search \
query.fasta \
$TARGET \
ligT_search.mmseqs.m8 \
tmp \
--threads 5 \
 --format-mode 4
```

3 Phylogenetics

[ ]:
```
# Run foldmason.
# structures contains the 2H structures from phage and eukaryotic viruses
mkdir -p tmp
foldmason easy-msa \
structures \
ligt_virus_only_foldmason.fasta \
tmp \
--report-mode 1

# Run iqtree
iqtree \
-s ../foldmason/ligt_virus_only_foldmason.fasta_aa.fa \
-m TEST \
-B 1000 \
--prefix ligt_virus_only_foldmason \
--seqtype AA \
-T AUTO \
--threads-max 10 \
--redo \
-v
```

4 Finding co-associated domains

[ ]:
```
# Running Chainsaw to split domains
#--------------------------------#
STRUCTURE_DIR=/path/to/2H/structures
OUTPUT_DIR=result

for FILE in "$STRUCTURE_DIR"/*.pdb; do
    BASENAME=$(basename "$FILE" .pdb)
    OUTFILE="${OUTPUT_DIR}/${BASENAME}.txt"

    echo "Processing $BASENAME at $(date)"

    if [[ ! -f "$FILE" ]]; then
        echo "  [ERROR] File not found: $FILE"
        continue
    fi

    python path/to/chainsaw/get_predictions.py \
        --structure_file "$FILE" \
        --output "$OUTFILE"

    STATUS=$?
```

```
    if [[ $STATUS -ne 0 ]]; then
        echo "  [ERROR] Python script failed for $FILE with exit code $STATUS"
    else
        echo "  [OK] Finished $BASENAME"
    fi
done


# Concatenate the chainsaw output
#--------------------------------#
dir=$OUTPUT_DIR
first=1
for f in "$dir"/*.txt; do
    if [ $first -eq 1 ]; then
        cat "$f"
        first=0
    else
        tail -n +2 "$f"
    fi
done > combined_chainsaw_file.txt

# Extract the domains
#--------------------------------#
CHAINSAW_FILE_PATH=/path/to/combined_chainsaw_file.txt
PDB_PATH=path/to/structures
OUTPUT=extracted_domains
for PDB_FILE in $PDB_PATH/*.pdb; do
    sat.py struc_get_domains -s $PDB_FILE -c $CHAINSAW_FILE_PATH -m 80 -o␣
  ↪$OUTPUT
done

# Run DALI to identify which domains are the 2H domains
#--------------------------------#
dali_format_inputs.sh \
-d extracted_domains \
-o databases/domain_2H_db \
-s DALI_search/struc_2H_key.txt

QUERY_DB=/path/to/acb1/
TARGET_DB=databases/domain_2H_db
KEY=DALI_search/struc_2H_key.txt

/wynton/home/doudna/nprice/vpSAT/bin/dali.sh \
-q $QUERY_DB \
-t $TARGET_DB \
-o 2H_domain_dali_result/unparsed \
-n 20
```

```
conda activate SAT

for FILE in 2H_domain_dali_result/unparsed/*txt ; do

BASE=$(basename ${FILE%.txt})

sat.py aln_parse_dali \
-a $FILE \
-s $KEY \
-o 2H_domain_dali_result/${BASE}.m8

done

cat 2H_domain_dali_result/*m8 > 2H_domain_dali_result/all_2H_domains.m8

# Pull out domains that didn't align to the 2H domains
#-------------------------------#
DOMAIN_PDBS=path/to/extracted_domains

ALN=2H_domain_dali_result/all_2H_domains.m8

mkdir -p non2H_domains

for FILE in "${DOMAIN_PDBS}"/*.pdb; do
  BASE=$(basename "${FILE%.pdb}")

  if ! grep -q "$BASE" "$ALN"; then
    cp $FILE non2H_domains
  fi
done

# Run Foldseek against the CATH structures
#-------------------------------#
foldseek easy-search \
non2H_domains \
cath_foldseek_db/cath \
foldseek_non2H_vs_cath/aln.m8 \
tmp \
 --format-mode 4 \
--tmscore-threshold 0.5 \
--format-output␣
 ↪"query,target,fident,alnlen,mismatch,gapopen,qstart,qend,tstart,tend,qcov,tcov,evalue,bits,

# Filter for alignments > 60 reisudes
sat.py aln_filter \
-a foldseek_non2H_vs_cath/aln.m8 \
```

```
-o foldseek_non2H_vs_cath/aln.lenFilt.m8 \
-x alnlen \
-m 60 \
-M 100000
```

[ ]: