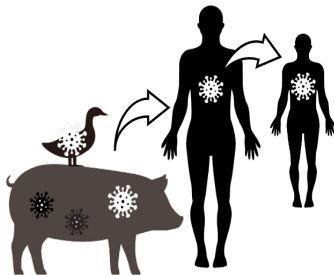# Emergenet Package Usage

ZeD Lab

University of Chicago

October 8, 2023

Emergenet computationally learns how new viral variants emerge using only viral sequence data

- Install Emergenet: `pip install emergenet --upgrade`
- Emergenet (recomputed for each time-period) is expected to automatically factor in the evolving host immunity and background environment
- Two applications:
  1. Forecasting dominant Influenza strains for future seasonal outbreaks
  2. Assessing risk of Influenza strains circulating in non-human hosts
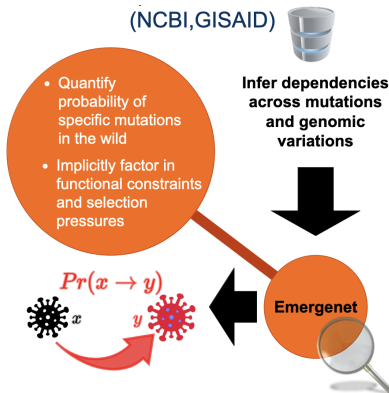
Emergenet comprises an interdependent collection of local predictors

- Each aims to predict the residue at a particular index using as features the residues at other indices

- Individual predictors (one for each sequence index) are implemented as conditional inference trees in which nodal splits have a minimum pre-specified significance in differentiating the child nodes

- Each predictor yields an estimated conditional residue distribution at each index

A sample of conditional inference trees in inferred Emergenet (H1N1 HA)
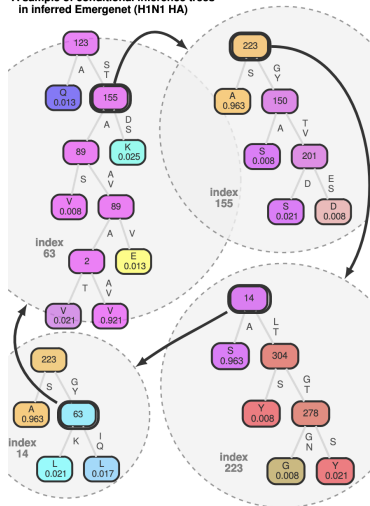
Color key (mixed colors represent distributions)

A D E G I K L Q S T V Y

Our Emergenet-inferred **E-distance** metric adapts to the background environment and approximates the log-likelihood of spontaneous change

▶ We define the E-distance as the square-root of the Jensen-Shannon divergence of the conditional residue distributions, averaged over the sequence

▶ Unlike the classic edit distance, the E-distance is informed by the Emergenet-inferred dependencies, and adapts to the specific subtype, allele frequencies, and environmental variations

▶ The E-distance approximates the log-likelihood of spontaneous change *i.e.* $\log Pr(x \to y)$

▶ Model trained on current data $\to$ predict for the near future

# Application 1: forecasting dominant strains for future seasonal outbreaks

▶ A dominant strain for an upcoming season may be identified as one which maximizes the joint probability of simultaneously arising from each (or most) of the currently circulating strains

▶ We can use the E-distance metric informed by an Emergenet constructed with hemagglutinin (HA) sequences from the previous season to predict the dominant strain for the next season

Forecast dominant strain(s) in upcoming season

H^t

Human strains previous season

dominant next season

$x_\star^{t+\delta}$
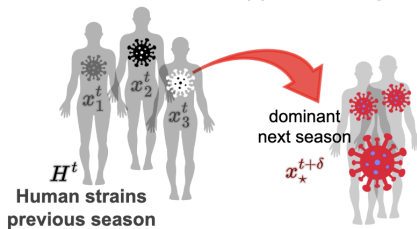
$$x_\star^{t+\delta} = \underset{x}{argmax} \prod_{x^t \in H^t} Pr(x^t \to x)$$

Emergenet recommendations are closer to the dominant strain than WHO yearly vaccine recommendations
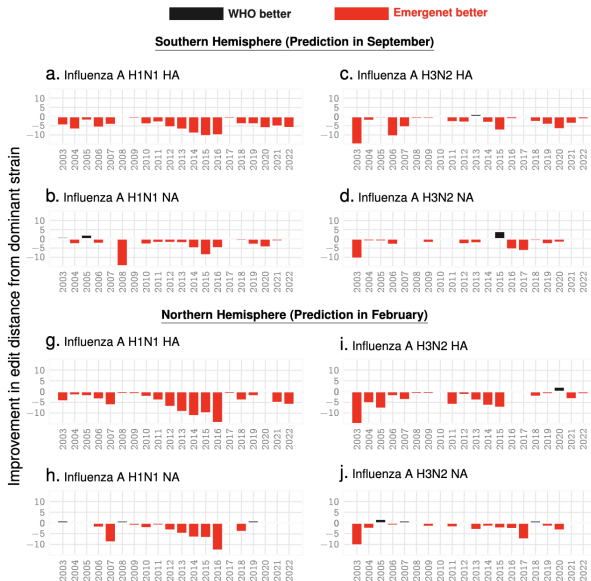
Demonstration of **emergenet.domseq** usage

- ▶ Download the example data
- ▶ This tutorial analyzes H1N1 from 2021-2022, northern hemisphere
- ▶ Data downloaded from GISAID and NCBI, and parsed, merged, and cleaned with BioPython (see notebook)
- ▶ Compute current season (2021-2022) dominant strains and predict future dominant strain (2022-23) using HA sequences

```python
import pandas as pd
from emergenet.domseq import DomSeq, save_model, load_model

DATA_DIR = 'example_data/domseq/'

# initialize the DomSeq, truncating HA to 565
# all sequences must be same length to train and use an Emergenet
domseq = DomSeq(seq_trunc_length=565, random_state=42)
```

Demonstration of **emergenet.domseq** usage

- *north_h1n1_21_22.csv* contains HA sequences from 02/15/2021 - 02/14/2022
- The sequences must be stored in a *sequence* column
- If you have a FASTA file, use *domseq.load_data* to load and format automatically

```python
# load current season sequences
df = pd.read_csv(DATA_DIR+'north_h1n1_21_22.csv')
print('Number of sequences:', len(df))
# Number of sequences: 735

# if you have a fasta file, you can parse it using load_data
df = domseq.load_data(self, 'file.fasta', outfile='sequences.csv')
```

Demonstration of **emergenet.domseq** usage

- ▶ *domseq.compute_domseq* does not rely on the Emergenet at all; it returns the cluster-wise dominant sequences of the input sequences, against which we evaluate our predictions from the previous season

- ▶ *domseq.compute_domseq* automatically clusters sequences and returns the centroid strain and cluster size for each cluster

```
1  # compute dominant sequences for 2021-2022
2  dom_seqs = domseq.compute_domseq(seq_df=df)
3
4  # save dominant sequences
5  dom_seqs.to_csv(DATA_DIR + 'dom_seqs_21_22.csv', index=False)
6  print(dom_seqs[['name','sequence','cluster_size']])
```

	name	sequence	cluster_size
15	A/Togo/0172/2021	MKAILVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	531
197	A/Bangladesh/9004/2021	MKAILVVMLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	193
165	A/Wisconsin/04/2021	MKAVLVVLLYTVTNANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	2
89	A/Mecklenburg-Vorpommern/1/2021	MEAKLFVLFCVFTALKADTICVGYHANNSTDTVDTIMEKNVTVTHS...	1
60	A/Gansu-Xifeng/1194/2021	MKARLFILFCAFTALKADTICVGYHANNSTDTVDTILEKNVTVTHS...	1
82	A/Wisconsin/03/2021	MKAVLVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	1
516	A/Denmark/36/2021	MKAILVALLYTFATANADTLCIGYHANNSTDTVDTILEKNVTVTHS...	1
372	A/SouthAfrica/PET20744/2021	MKAILVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	1
24	A/Gansu-Xifeng/1143/2021	MKARLFILFCAFTALKADTICVGYHANNSTDTVDTILEKNVTVTHS...	1
9	A/Parana/10835/2021	MKAILVVLLYTFATTNADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	1
281	A/SouthAfrica/PPET20447/2021	MKAILVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	1
90	A/Mecklenburg-Vorpommern/1-A/2021	MEAKLFVLFCVFTALKADTICVGYHANNSTDTVDTIMEKNVTVTHS...	1

Demonstration of **emergenet.domseq** usage

- ▶ Next, we predict dominant strains for 2022-2023
- ▶ First train the Emergenet using sequences from 2021-2022
- ▶ We use 3000 to demonstrate the *sample_size* parameter, but it does nothing here as *df* contains 735 sequences

```
1  # train enet
2  enet = domseq.train(seq_df=df, sample_size=3000, n_jobs=1)
3  # save enet
4  save_model(enet=enet, outfile=DATA_DIR+'enet.joblib')
```

Demonstration of **emergenet.domseq** usage

- *north_h1n1_21_22_pred.csv* contains human HA sequences from 09/15/2001 - 02/14/2022 (all candidate sequences from the past)
- Get predictions from the three largest clusters
- Since our candidate sequence dataframe is large (18,057 sequences), this will take a while

```
1  # load candidate sequences for recommendation
2  candidate_df = pd.read_csv(DATA_DIR+'north_h1n1_21_22_pred.csv')
3  print('Number of sequences:', len(candidate_df))
4  # Number of sequences : 18057
5
6  # compute prediction sequences (predictions from 3 largest clusters)
7  pred_df = domseq.predict_domseq(seq_df=df, pred_seq_df=candidate_df,
        enet=enet_model, n_clusters=3, sample_size=3000)
8
9  # save predictions
10 pred_df = pred_df.sort_values(by=['cluster_size'], ascending=False)
11 pred_df.to_csv(DATA_DIR+'predictions_for_22_23.csv', index=False)
12 print(pred_df[['name','sequence','cluster_size']])
```

	name	sequence	cluster_size
0	A/Netherlands/00475/2020	MKAILVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	527
1	A/Bangladesh/9004/2021	MKAILVVMLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	193
2	A/North_Dakota/12226/2021	MKAILVVLLYTFTTANADTLCIGYHANNSTDTVDTVLEKNVTVTHS...	3

# Application 2: assessing risk of Influenza strains circulation in animals

- ▶ The Center for Disease Control's (CDC) current method is the Influenza Risk Assessment Tool (IRAT)
- ▶ Scalability issue: IRAT depends on multiple experimental assays, possibly taking weeks to compile for a single strain
- ▶ We estimate the pandemic potential of novel animal strains, via a time-varying **E-risk** score, computed using the E-distance informed by the Emergenet trained on recent human hemagglutinin (HA) and neuraminidase (NA) strains

Estimate pandemic potential of animal strains
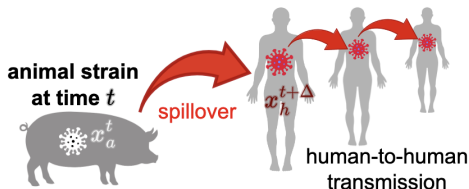
$$\mathrm{risk}(x_a^t) \propto \langle \log Pr(x_a^t \to x_h^{t+\Delta}) \rangle$$

Emergenet scores are correlated with IRAT emergence scores, R = 0.707
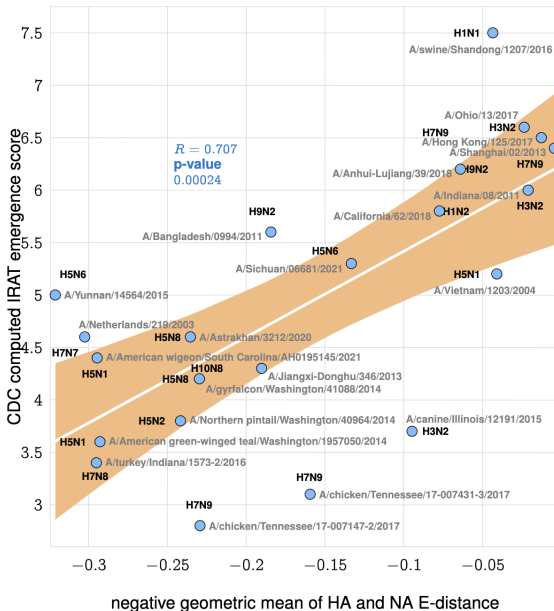
Demonstration of **emergenet.emergenet** usage

- ▶ Download the example data
- ▶ This tutorial analyzes the risk of our target sequence, A/Ohio/13/2017, assessed by IRAT in July 2019
- ▶ Initialize the HA Enet with a FASTA file containing only the HA segment of A/Ohio/13/2017; initialization with a Python string is also possible (see notebook)

```python
import numpy as np
from emergenet.emergenet import Enet, save_model, load_model,
    irat_risk

DATA_DIR = 'example_data/emergenet/'

# initialize the Enet with A/Ohio/13/2017 HA
enet_ha = Enet(seq=DATA_DIR+'ha_target_sequence.fasta',
    seq_trunc_length=550, random_state=42)

print(enet_ha.seq_metadata, '\n')
print(enet_ha.seq, '\n')
```

A/Ohio/13/2017|A_/_H3N2|$SEGMENT_NAME|2017-07-14|EPI1056653|

MKTIIALSHILCLVFAQKLPGNDNNMATLCLGHHAVPNGTIVKTITNDQIEVTNATELVQSFSTGEICNSPYQILDGENCTLIDALLGDPQCDGFQNNKWDLFVERSKAHSNCYP
YDVPDYASLRSLVASSGTLEFNNESFNWTGVTQDGASSSCKRRSSNSFFSRLNWLTHLNFKYPALEVTMPNNEQFDKLYIWGVHHPATDKDQISLYAQAAGRIIVSTKRNQQAVI
PNIGSRPRVRDIPSRISIYWTIVRPGDILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSACITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNIPEKQTR
GIFGAIAGFIENGWEGMVDGWYGFRHQNSEGRGQAADLKSTQAAIDQINGKLNRLIGKTNEKFHQIEKEFSDVEGRIQDLEKYVEDTKIDLWSYNAELLVALENQHTIDLTDSEM
NKLFEKTKKQLRENAEDMGNGCFKIYHKCDNACIGSIRNGTYDHDVYRNEALNNRFQIKGVELKSEYKDWILWISFAISCFLLCVALLGFIMWACQKGNIKCNICI

Demonstration of **emergenet.emergenet** usage

- *ha_sequences.fasta* contains human HA sequences from 7/1/2018 - 6/30/2019 (one year prior to IRAT assessment)
- Train the HA Emergenet using these sequences; it will take a couple minutes

```python
1  # load fasta data
2  df_ha = enet_ha.load_data(filepath=DATA_DIR+'ha_sequences.fasta')
3  print('Number of sequences:', len(df_ha))
4  # Number of sequences: 12389
5
6  # train enet (automatically includes target sequence with df)
7  enet_ha1 = enet_ha.train(seq_df=df_ha, sample_size=1000, n_jobs=1)
8  # save enet
9  save_model(enet=enet_ha1, outfile=DATA_DIR+'ha_enet.joblib')
```

Demonstration of **emergenet.emergenet** usage

- ▶ Compute the HA emergence risk score and variance under the trained HA Emergenet model
- ▶ This will take a couple seconds

```
1  # compute risk score
2  risk_score_ha, variance_ha = enet_ha.emergence_risk(seq_df=df_ha,
       enet=enet_ha1, sample_size=1000)
3
4  print('Emergenet Risk Score:', risk_score_ha)
5  print('Variance:', variance_ha)
6  # Emergenet Risk Score: 0.020315896450399634
7  # Variance: 1.673019568927247e-05
```

Demonstration of **emergenet.emergenet** usage

▶ We repeat the last three slides, but with NA

```
1  # initialize the Enet with A/Ohio/13/2017 NA
2  enet_na = Enet(seq=DATA_DIR+'na_target_sequence.fasta',
       seq_trunc_length=449, random_state=42)
3
4  # load fasta data
5  df_na = enet_na.load_data(filepath=DATA_DIR+'na_sequences.fasta')
6  print('Number of sequences:', len(df_na))
7  # Number of sequences: 12388
8
9  # train enet (automatically includes target sequence with df)
10 enet_na1 = enet_na.train(seq_df=df_na, sample_size=1000, n_jobs=1)
11 # save enet
12 save_model(enet=enet_na1, outfile=DATA_DIR+'na_enet.joblib')
13
14 # compute risk score
15 risk_score_na, variance_na = enet_na.emergence_risk(seq_df=df_na,
       enet=enet_na1, sample_size=1000)
16
17 print('Emergenet Risk Score:', risk_score_na)
18 print('Variance:', variance_na)
19 # Emergenet Risk Score: 0.03050091990424366
20 # Variance: 1.564767759048388e-05
```

Demonstration of **emergenet.emergenet** usage

- ▶ Finally, we compare our scores to IRAT
- ▶ The geometric mean of HA and NA scores is 0.024893, as seen in the IRAT vs. Emergenet figure
- ▶ *irat_risk* uses our GLM models (see S-Tab. 4-5 in the paper) to predict the IRAT emergence and impact risk scores: 6.3 and 6.4, respectively
- ▶ IRAT's scores are 6.6 and 5.8, respectively

```python
geom_mean_risk_score = np.sqrt(risk_score_ha * risk_score_na)
irat_emergence_prediction, irat_impact_prediction = irat_risk(
    risk_score_ha, risk_score_na)

print('Geometric Mean of HA and NA risk scores:', round(
    geom_mean_risk_score, 6))
print('Emergenet prediction of IRAT emergence estimate:', round(
    irat_emergence_prediction, 1))
print('IRAT emergence estimate: 6.6')
print('Emergenet prediction of IRAT impact estimate:', round(
    irat_impact_prediction, 1))
print('IRAT impact estimate: 5.8')
# Geometric Mean of HA and NA risk scores: 0.024893
# Emergenet prediction of IRAT emergence estimate: 6.3
# IRAT emergence estimate: 6.6
# Emergenet prediction of IRAT impact estimate: 6.4
# IRAT impact estimate: 5.8
```

Links to additional resources

- To download the package, visit our package page on PyPI
- For the theoretical framework of Emergenet and full results of our experiments, see our paper
- For source code and examples, visit our GitHub repository