**Joel K Weltman**

## SUPPLEMENTARY INFORMATION

# **Sample code. Python 2.7**

# **Task[1]** Compute amino acid entropy (**H**).
# **Task[2]** Sort for amino acid positions where **H**>0 in human but not in mosquito
# **Task[3]** Compute cumulative mutual information (**cMIxx**) within the Exclusive human subset (**Hx**)

**Task[1]** Compute Shannon entropy (H) in bits. Input probability of occurrence of each amino acid.

```
# a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y = amino acid probabilities

from cmath import *

def main(a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y):

    if a==0:
        Ha=0
    else: Ha=a*log(1/a,2)

    if c==0:
        Hc=0
    else: Hc=c*log(1/c,2)

    if d==0:
        Hd=0
    else: Hd=d*log(1/d,2)

    if e==0:
        He=0
    else: He=e*log(1/e,2)

    if f==0:
        Hf=0
    else: Hf=f*log(1/f,2)

    if g==0:
        Hg=0
    else: Hg=g*log(1/g,2)
```

```
if h==0:
    Hh=0
else: Hh=h*log(1/h,2)

if i==0:
    Hi=0
else: Hi=i*log(1/i,2)

if k==0:
    Hk=0
else: Hk=k*log(1/k,2)

if l==0:
    Hl=0
else: Hl=l*log(1/l,2)

if m==0:
    Hm=0
else: Hm=m*log(1/m,2)

if n==0:
    Hn=0
else: Hn=n*log(1/n,2)

if p==0:
    Hp=0
else: Hp=p*log(1/p,2)

if q==0:
    Hq=0
else: Hq=q*log(1/q,2)

if r==0:
    Hr=0
else: Hr=r*log(1/r,2)

if s==0:
    Hs=0
else: Hs=s*log(1/s,2)

if t==0:
    Ht=0
else: Ht=t*log(1/t,2)

if v==0:
    Hv=0
```

```
    else: Hv=v*log(1/v,2)

    if w==0:
        Hw=0
    else: Hw=w*log(1/w,2)

    if y==0:
        Hy=0
    else: Hy=y*log(1/y,2)

    H=Ha+Hc+Hd+He+Hf+Hg+Hh+Hi+Hk+Hl+Hm+Hn+Hp+Hq+Hr+Hs+Ht+Hv+Hw+Hy

    H=abs(H)

    return H

# Task[2] sort for positions where H>0 in human origin polyproteins and H=0 in Aedes origin polyproteins
#   Hx = exclusive human subset with H>0; n=3423 polyprotein amino acid positions

Hx=zeros(n)
for ndx2 in range(0, n, 1):
   x1 = H_human[ndx2]
   x2 = H_aedes[ndx2]
   if x1 > 0.0:
      if x2 == 0.0:
         Hx[ndx2] = x1

# Task[3] compute cumulative MI (cMIxx) in paired members of Human Exclusive Subset

cMIxx = zeros(n)
for ndx1 in range(0, n, 1):
   if Hx[ndx1] > 0.0:
      seqx1 = make_seq_human(ndx1)
      seqx1 = seqx1.replace('X', '-') # 'X' represents gaps
      for ndx2 in range(0, n, 1):
         if Hx[ndx2] > 0.0:
            seqx2 = make_seq_human(ndx2)
            seqx2 = seqx2.replace('X', '-') # 'X' represents gaps
            pairs = zip(seqx1, seqx2)
            list_out2=[]
            M = zeros((20, 20))
            M = matrix(M, float)
            for ndx3 in range(0, 20, 1):
               x1 = aa_list1[ndx3]
```

```
    for ndx4 in range(0, 20, 1):
        x2 = aa_list2[ndx4]
        ref_pair = (x1, x2)
        ref_pair2 = aa_list1[ndx3], aa_list2[ndx4]
        if ref_pair in pairs:
            cx=pairs.count(ref_pair)
            M[ndx3, ndx4] = float(cx)

sum_M =sum(M)
probs = M/sum_M
M2 = zeros((20, 20))
M2 = matrix(M, float)
for ndx5 in range(0, 20, 1):
    for ndx6 in range(0, 20, 1):
        x = probs[ndx5, ndx6]
        if x > 0.0:
            Hjoint = -x*log2(x)
            M2[ndx5, ndx6] = Hjoint
MI =  Hx[ndx1] + Hx[ndx2] - sum(M2)
if MI>0.0:
    if ndx1 != ndx2:   # correction for autocorrelation
        cMIxx[ndx1] = cMIxx[ndx1] + MI
```