This folder contains work towards determining which BBTM matrices ot use at which sequence identities. It very much overlaps with the work in "deriving bbtm matrices" , "automated sequence alignment", and "ziff module": these four folders are all part of my effort to create and verify the usefulness of alignments for the hmomology modeling that goes into deriving Ezβ assymmetric.

# August 29, 2012

Downloaded TMout.zip from an e-mail David jimenez-Morales sent me on August 23. This is the same TMout folder that is seen in the "deriving bbtm matrices" folder and "automated sequence alignment" folder.

Here's a quote from the 1994 ClustalW paper:

“The ranges of distances and tables used with the PAM series of matrices are: 80-100% :PAM20, 60-80%:PAM60, 40-60%:PAM120, 0-40%:PAM350. The range used with the BLOSUM series is: 80-100%: BLOSUM80, 60-80% :BLOSUM62, 30-60% :BLOSUM45, 0-30%:BLOSUM30.”

Which makes me wonder: what expected seq identities do those PAM and BLOSUM matrices correspond to? How do they choose a representative matrix from the range?

I wrote some code that, given a PAM or BBTM distance, would give the expected sequence identity at that distance:

***import numpy as np***

***import scipy***

***import matrices***

***class MatrixMapping(dict):***

***'''A dictionary such that if x is a MatrixMapping, (n\*x)[key] == n\*(x[key])***

***'''***

***def \_\_rmul\_\_(self, other):***

***output = copy.deepcopy(self)***

***for key in output.keys():***

***output[key] \*= other***

***return output***

***def \_\_lmul\_\_(self, other):***

***return self.\_\_rmul\_\_(self, other)***

***published\_bbtm\_ordering = ['A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H',***

***'I', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W',***

***'Y', 'V']***

***# Background frequencies, e-mailed to me by David Jimenez-Morales***

***# on August 20 2012. I consider it a birthday present***

***pi\_out = {'A': 0.103414, 'C': 0.000253, 'E': 0.003965, 'D': 0.010899,***

***'G': 0.071558, 'F': 0.088656, 'I': 0.064497, 'H': 0.016882,***

***'K': 0.009315, 'M': 0.018249, 'L': 0.168981, 'N': 0.016985,***

***'Q': 0.023042, 'P': 0.018898, 'S': 0.025996, 'R': 0.012083,***

***'T': 0.050352, 'W': 0.045422, 'V': 0.115135, 'Y': 0.135606}***

***# Amino acid frequencies from http://www.tiem.utk.edu/~gross/bioed/webmodules/aminoacid.htm retrieved August 27 2012***

***pi\_ver = dict({'A': 7.4e-2,***

***'R': 4.2e-2,***

***'N': 4.4e-2,***

***'D': 5.9e-2,***

***'C': 3.3e-2,***

***'E': 5.8e-2,***

***'Q': 3.7e-2,***

***'G': 7.4e-2,***

***'H': 2.9e-2,***

***'I': 3.8e-2,***

***'L': 7.6e-2,***

***'K': 7.2e-2,***

***'M': 1.8e-2,***

***'F': 4.0e-2,***

***'P': 5.0e-2,***

***'S': 8.1e-2,***

***'T': 6.2e-2,***

***'W': 1.3e-2,***

***'Y': 3.3e-2,***

***'V': 6.8e-2})***

***pam1 = 10\*\*-4 \* matrices.parse('pam1.txt')***

***def parse\_david(path):***

***'''Open the matlab format matrix files that David Jiminez-Morales***

***sent me (in the "pout" folder)'''***

***with open(path, 'r') as f:***

***output = MatrixMapping()***

***for row\_resn, line in zip(published\_bbtm\_ordering, f):***

***for col\_resn, entry in zip(published\_bbtm\_ordering,***

***line.split()):***

***output.update({(row\_resn, col\_resn): float(entry)})***

***return output***

***def p\_retrieve(t):***

***'''Return one of David Jimenez-Morales's transition probability***

***matrices, corresponding to the given time t, from the matrices***

***he sent me on August 23 2012.'''***

***return ~~matrices.parseparse\_david~~('TMout/pout/MTMout{0}.p'.format(t))***

***def id\_given\_time(t, mat\_name='bbtm'):***

***'''Given a time and the name of a matrix family (bbtm and pam***

***currently available) return the expected value of %identity between***

***a sequence and the same sequence after a time t has passed,***

***given that the amino acid frequencies in the original sequence***

***are equal to the background frequencies'''***

***if mat\_name == 'pam':***

***pam\_at\_t = matrices.matrix\_power(pam1, t)***

***return 1 - matrices.expected\_changes(pam\_at\_t, pi\_ver)***

***if mat == 'bbtm':***

***bbtm\_at\_t = p\_retrieve(t)***

***return 1 - matrices.expected\_changes(bbtm\_at\_t, pi\_out)***

(fixed an error in p\_retrieve, which was not actually used for the PAM results)

It's the last function that a user of this code actually runs, to find an expected sequence identity.

Pam20, 80% to 100%:

***>>> id\_given\_time(20, mat\_name='pam')***

***0.81964237460071687***

Pam60, 60% to 80%:

***>>> id\_given\_time(60, mat\_name='pam')***

***0.57486688878682646***

Pam120, 40% to 60%:

***>>> id\_given\_time(120, mat\_name='pam')***

***0.37126323726146515***

Pam350, 0% to 40%:

***>>> id\_given\_time(350, mat\_name='pam')***

***0.13774262286766792***

I am confused, because the expected identities are *outside the ranges for which they are meant to be used. Maybe they use different amino acid frequencies? The ones I'm using are the vertebrate amino acid frequencies. PAM is used for plant, bacterial, everything.*

*I don't care a whole lot. This was an interesting exercise and I'm glad I got to see and interpret the results, but the importance of this was really to put me in a good position to find the correct BBTM matrices for a variety of ranges of sequence identity, and I think I can do that now.*

september 28 - merged this update with mainstream log

# September 17, 2012

Wrote some new code, and repurposed some old code, to check the expected number of changes given a particular transition matrix and starting distribution. To test it, I checked to see whether PAM1 would really give 1% expected positions different.

I used the PAM1 matrix downloaded on August 27 2012 from http://www.icp.ucl.ac.be/~opperd/private/pam1.html. I used amino acid frequencies in vertebrates from http://www.tiem.utk.edu/~gross/bioed/webmodules/aminoacid.htm retrieved August 27 2012.

Using the following code:

**import** numpy as np

**import** scipy

**import** itertools

**import** copy

**class** MatrixMapping(dict):

'''A dictionary such that if x is a MatrixMapping, (n\*x)[key] == n\*(x[key])

    '''

**def** \_\_rmul\_\_(self, other):

output = copy.deepcopy(self)

**for** key **in** output.keys():

output[key] \*= other

**return** output

**def** \_\_lmul\_\_(self, other):

**return** self.\_\_rmul\_\_(self, other)

published\_bbtm\_ordering = ['A', 'R', 'N', 'D', 'C', 'Q', 'E', 'G', 'H',

'I', 'L', 'K', 'M', 'F', 'P', 'S', 'T', 'W',

'Y', 'V']

# Background frequencies, e-mailed to me by David Jimenez-Morales

# on August 20 2012. I consider it a birthday present

pi\_out = {'A': 0.103414, 'C': 0.000253, 'E': 0.003965, 'D': 0.010899,

'G': 0.071558, 'F': 0.088656, 'I': 0.064497, 'H': 0.016882,

'K': 0.009315, 'M': 0.018249, 'L': 0.168981, 'N': 0.016985,

'Q': 0.023042, 'P': 0.018898, 'S': 0.025996, 'R': 0.012083,

'T': 0.050352, 'W': 0.045422, 'V': 0.115135, 'Y': 0.135606}

# Amino acid frequencies from http://www.tiem.utk.edu/~gross/bioed/webmodules/aminoacid.htm retrieved August 27 2012

pi\_ver = dict({'A': 7.4e-2,

'R': 4.2e-2,

'N': 4.4e-2,

'D': 5.9e-2,

'C': 3.3e-2,

'E': 5.8e-2,

'Q': 3.7e-2,

'G': 7.4e-2,

'H': 2.9e-2,

'I': 3.8e-2,

'L': 7.6e-2,

'K': 7.2e-2,

'M': 1.8e-2,

'F': 4.0e-2,

'P': 5.0e-2,

'S': 8.1e-2,

'T': 6.2e-2,

'W': 1.3e-2,

'Y': 3.3e-2,

'V': 6.8e-2})

**def** expected\_changes(mat, pi):

'''Expected fraction of positions different from original, after update

    with mat, given original frequency distribution pi.

    Very different from expected number of TRANSITIONS.'''

**return** sum((1 - mat[i,i])\*pi[i] **for** i **in** pi.keys())

**def** avg\_rate(q, pi):

**return** -1 \* sum(q[i,i] \*pi[i] **for** i **in** pi.keys())

**def** parse(mat\_filename):

'''Take filename of a file in the format in which matrices are presented

    in "Patterns of Amino Acid Substitutions..." Jimenez-Morales, Jie Liang

    and return a dictionary that can be used like q['A']['T'] to find

    the entry in row A, column T.'''

with open(mat\_filename, 'r') as mat\_file:

found\_resns = False

**for** line **in** mat\_file:

# Ignore comments

**if** line[0] == '#':

**continue**

# Find which resn corresponds to which column number

**if** **not** found\_resns:

found\_resns = True

col\_names = line.split()

# Check to make sure they're all there

**for** resn **in** ['C', 'N', 'H', 'D', 'S', 'Q', 'K', 'M', 'P',

'T', 'F', 'A', 'G', 'I', 'L', 'R', 'W', 'E',

'Y', 'V']:

**try**:

**assert** resn **in** col\_names, "missing " + resn

**except** AssertionError:

**print**(line)

**raise**

# Make the matrix that will be returned

output = MatrixMapping((tuple\_, None)\

**for** tuple\_ **in** itertools.product(\

col\_names, col\_names))

**continue**

row = line.split()

row\_name = line[0]

**for** rate, col\_name **in** zip(row[1:], col\_names):

output[row\_name,col\_name] = float(rate)

**return** output

pam1 = 10\*\*-4 \* parse('pam1.txt')

**def** to\_mat(m, order):

'''The parser returns a dictionary. This function turns one of

    those dictionaries into a matrix, with the elements in the

    specified order. "order" should be a list of one-letter

    residue names.'''

# I know more about manipulating lists than matrices,

# so the output is constructed as a list, then converted into a

# matrix right before the return statement

mat\_as\_list = list()

**for** row\_name **in** order:

# Append a row:

mat\_as\_list.append([m[row\_name,col\_name] **for** col\_name **in** order])

mat = scipy.matrix(mat\_as\_list)

**return** mat

**def** parse\_david(path):

'''Open the matlab format matrix files that David Jiminez-Morales

    sent me (in the "pout" folder)'''

with open(path, 'r') as f:

output = MatrixMapping()

**for** row\_resn, line **in** zip(published\_bbtm\_ordering, f):

**for** col\_resn, entry **in** zip(published\_bbtm\_ordering,

line.split()):

output.update({(row\_resn, col\_resn): float(entry)})

**return** output

**def** david\_changes(n):

**return** expected\_changes(parse\_david('TMout/pout/MTMout{0}.p'\

.format(n)), pi\_out)

I checked the expected fraction of amino acids different after update with the PAM1 matrix, using the vertebrate amino acid frequencies:

>>> expected\_changes(pam1, pi\_ver)

0.010248299999999943

Then, using the transition probability matrices and amino acid frequences for the BBTMOUT dataset that David Jimenez-Morales sent me, copied from "bbtm derivation" (the date of their retrieval can be found in that log), I checked MTMout1. I would have thought, since the units are, according to one of his papers, evolutionary time units, that I would get the same 1% change (that's how 1 evolutionary time unit was defined in a paper of Jie Liang's). However, instead, I got about 1.7%:

>>> david\_changes(1)

0.017431194030000024

# September 21, 2012

In Molecular Biology and Evolution 2006, Tseng and liang write "Here 1 time unit represents the time required for 1 substitution per 100 residues (Dayhoff, Schwartz, and Orcutt 1978)". This is the paper that is cited as describing the methods used for rate matrix estimation in Jimenez-Morales, Liang PLoS One 2011. In the PLoS One paper, the idea of an evolutionary time unit is also referenced: "Figure S3... Scoring matrix BBTMout. Scoring matrix derived from Qout at evolutionary time unit of 40." So, I expected the matrices David Jimenez-Morales sent me to be labeled by evolutionary time unit. They apparently are not (see Sebtember 17 results, this log), which confuses me. I don't think David Jimenez-Morales would mislabel them, but I also think I calculated the expected number of substitutions correctly.

However, I will proceed under the assumption that my calculations are correct.

My goal is to produce a mapping f between ranges of sequence identities and BBTMout scoring matrices, to command ClustalW that when two sequences have sequence identity i, and i is in range R, to use the matrix f(R). I am not sure whether to use matrices with expected changes within the range, or with expected changes times two in the range.

Dan Gusfield at UC Davis has, on a website for one of his classes, notes that say the following (<http://cs124.cs.ucdavis.edu/lectures/scoringmatrices.pdf> retrieved today):

*"The two sequences*

*Si and Sj have each diverged from some common ancestor Sij , and the*

*molecular clock theory implies that the expected PAM distance (number*

*of PAM units of divergence) between Sij and Si equals the expected PAM*

*distance between Sij and Sj . So one uses half the number of diﬀerences in*

*the alignment of Si*

*to Sj to calculate the PAM distance between Sij and its*

*two derived sequences Si and Sj ."*

In a 1994 paper in *Nucleic Acids Research*, the ClustalW authors give their mapping from sequence identity ranges to PAM matrices. I can check to see whether I understand this correctly by looking at the expected changes of the PAM matrices they assigned ot each range.