Comparing protein data encoding in 3 common protein family representations

**Abstract**  
  
In this paper we present an analysis of 3 common encodings for protein family data. Assembling and analysing evolutionarily related protein families forms a core task in bioinformatics and and underpins a lot of molecular biology analysis. Until now little work has been present on how information is encoded in the protein family models we use and how this may impact common analyses such as protein search. Here we present a number of related experiments in characterising what 3 common protein family encodes attempt to represent, we present results for; the position specific score matrix, Hidden Markov Models and Deep Learning Transformers.

**Introduction**

The motivation for this work is an attempt to understand reasons for unwanted behaviours in protein database search [CITATION]. Established methods of protein database search include a number of iterative methods such as PSI-BLAST [CITATION], HH-BLITS[CITATION] and JACKHMMER(?)/[CITATION]. In such methods a query sequence is searched against a database of entities that encodes an entire database of proteins. The goal of such search is usually to identify the subset of proteins within the database which are evolutionarily related to the query sequence. After an initial iteration of such search, a series of accepted scoring hits, typically defined as those that pass some given threshold, is compiled. These initial hits are usually regarded as forming a putative evolutionary family with the query sequence. This set of proteins can then converted in to a model of that family which can be used as the basis for a subsequent iteration of the database search. In the case of PSI-BLAST the intermediary model that is compiled at each iteration is a called a Position Specific Score Matrix or PSSM. In the case of HHBlits [and JackHmmer] [CITATION] a new Hidden Markov Model (HMM) is produced to represent the family. The logic here is that the model represents the statistical variation within the family and will be better able to find more distant evolutionary relatives that came match the encoded family variation. Typically search is carried out for some fixed number of iterations or until the results converge and no new protein sequence hits are identified. Once search is complete a final model can be built and stored as a statistical representation of a “complete” evolutionarily related protein family. Such models can be thought of as a Sequence Family Profile.  
  
Ideally, a protein family is the representation of a cluster of proteins that forms a discrete group within the space of all known protein sequences. Typically the goal of protein search is to discover a set of evolutionarily related proteins “near” the query sequence such that they share some important or salient property such a their biochemical function or physical structure [CITATION]. However, it is well observed that membership of a search cluster changes as the iterative search progresses. Much of this is the desired behaviour of the search process as the set of proteins discovered expands. However some behaviour is unwanted such as including proteins that are not in the same evolutionary grouping or including members that move the cluster centroid away from the query sequence. These non-optimal behaviours are often termed “profile drift” [CITATION]. That is, the protein family model that is constructed at each iteration is drifting away from either query sequence or away from the putative target region of protein sequence space [CITATION].  
  
Such profile drift, and similar profile contamination has impacts across nearly all forms of bioinformatics analyses. The ability to discover coherent sets of sequence relatives is a critical part of many predictive methodologies in bioinformstics. These include protein function prediction methods such [CITATIONS], protein classification methodologies such as PRINTS, CATH, PFAM[CITATIONS] , proteins structure prediction such as AlphsFold2, DMPfold2 [CITATIONS]. When trying to make accurate predictions using protein families including “off-target” family members increases the number of false-positive predictions in classification tasks or increases the error rate for modelling tasks.  
  
However characterising drift runs in to important philosophical questions about the nature of protein families and protein evolution. Firstly protein space is not evenly populated, some types of protein family are vastly more populated and explored evoutionarily than others. Take for instance the P-loop NTPase protein domain [CITATION] this is one of the most commonly replicated proteins domains known. Additionally protein families are not spaced apart in “protein space” P-loop NTPase domains have evolutionarily close Rossmann Fold families (see CATH 3.40.50 and PFAM????). We might expect drift issues to be more prevelant in densely packed regions of protein space.

Additionally protein families are somewhat artificial human constructs and the criteria we use to define a family may not be the best way to represent the evolutionary history given protein families. For instance it is certainly useful for protein function prediction tasks that the CATH database chooses to segregate it’s Homologous Protein Families into functionally homogenous groups but in some cases this may not best represent the evolutionary trajectory of these groupings. There may be functions of evolutionary histories that are so close they need not be divided in to a separate family. The real sequence diversity in these families may “overlap” and protein search will correctly pick up sequences in both families.

An upshot of this is that an analysis of protein search drift will pick up artefacts of the protein family classification methodology as well as the drift issues that arise from the statistical properties of the search methodology.

In the first part of this study we look to characterise such protein drift. Using the commonly understood tool PSI-Blast we run a number of experiments to try and characterise how iterative search works for both real and synthetic protein sequence databases. And we look at how often drift happens when we search a real protein family database. Having characterised how discretely PSI-BLAST PSSMs represent the search space around a query sequence we then compare this behaviour protein family representations generated by Hidden Markov Models (HMMs) and Large Language Models (LLMs) [CITATION]

**Method**  
  
In the first phase of the study we look at how iterative protein search typically progresses. PSI-BLAST was chosen as it is a commonly available tool for searching fasta formatted protein datasets. It is widely used and it is trivially easy to monitor the behaviour of its iterative search process between each iteration. We prepared two protein datasets; the first was a series of synthetic protein databases where the sequences are a fixed density apart. And all sequences in these databases are 120 residues in length and are generated from the same seed sequence. This database was prepared using and iterative random walk strategy which is explained in full in Appendix A. The second dataset was PFAM release N [CHECK THIS].

For the synthetic databases we searched these using PSI-BLAST with the standard parameters, for 20 iterations. At each iteration we capture the set of hits that were identified at that iteration. For the PFAM database we take a representative structure from each family and search the whole PFAM database with PSI-BLAST over 20 iterations. Once again capturing the set of hits at each iteration. The goal here is to illustrate the extent to which protein search behaviour differs in real protein space which has non-uniform clustering and density to the synthetic case.

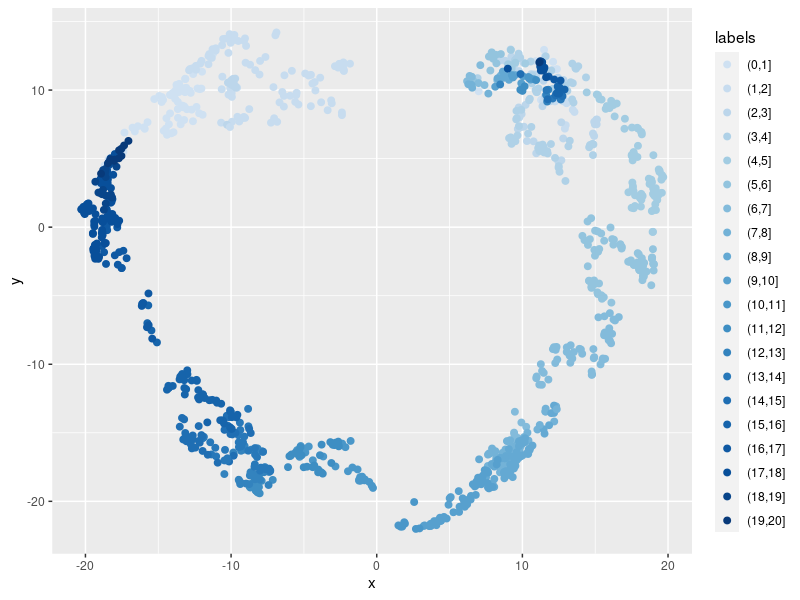
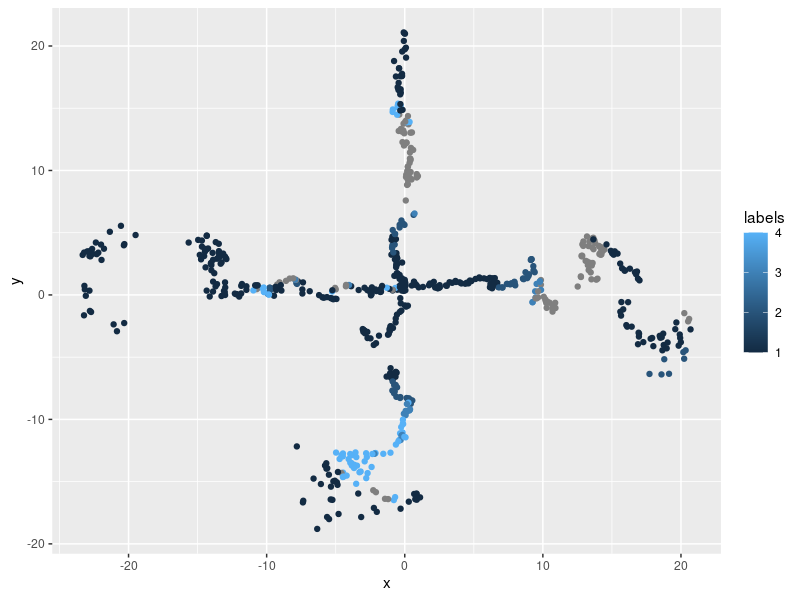
Profile drift behaviour may be impacted by methods ability to build and encode coherent, discrete protein family representations. In the next phase of the study we looked to compare the information encoded in. When searching pfam we noted that PSI-BLAST encounter drift-like behaviours in about 23% of searches for a subset N of pfam families. For each of these families we took the HMMER HMM profiles and using the HMM emit tool we generated 200 new sequences. Using fasta [CITATION] we then searched these sequences against a small database made up of the sequences from the family’s profile and the sequences PSI-BLAST identified as drift contaminants. The goal was to see if the HMM typically emits sequences within a family or is prone to emitting sequences that may be judge to be out of the family.  
  
We repeated a similar process with the Meta ESM models [CITAION]. We downloaded the ESM MSA and ESM Single [CHECK] language models. These models were used to generate new sequences for a pfam family. Once again we took set of families where we had recognised drift processes with PSI-BLAST. For the single sequence ESM we took individual sequences from the pfam family and masked 25, 50 or 75 percent of the residues. This was given as input to the model and the output was a new sequence. This was done 200 times per pfam family. The goal here was to see to what extent the language model was capable of reconstructing a sequence that remains within the family of the input sequence. Created sequences  
  
**Results**

**ARE DRIFT FAMILIES ENRICHED FOR SUPER FOLDS/DOMAINS  
  
ARE HARD ALPHAFOLD TARGETS ENRICHED FOR DRIFT FAMILIES**

**Part A, probably appendix)**

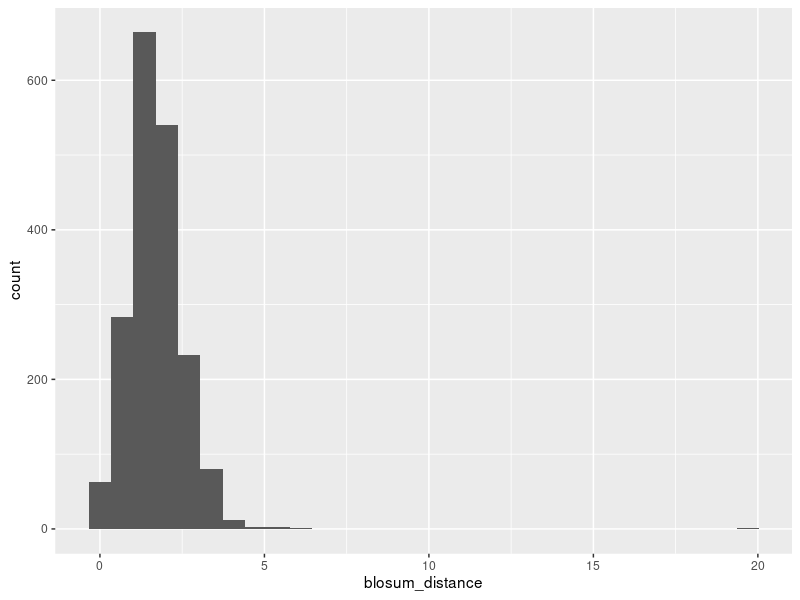
Exploring generation strategies for things that are less structured.

4 random walks



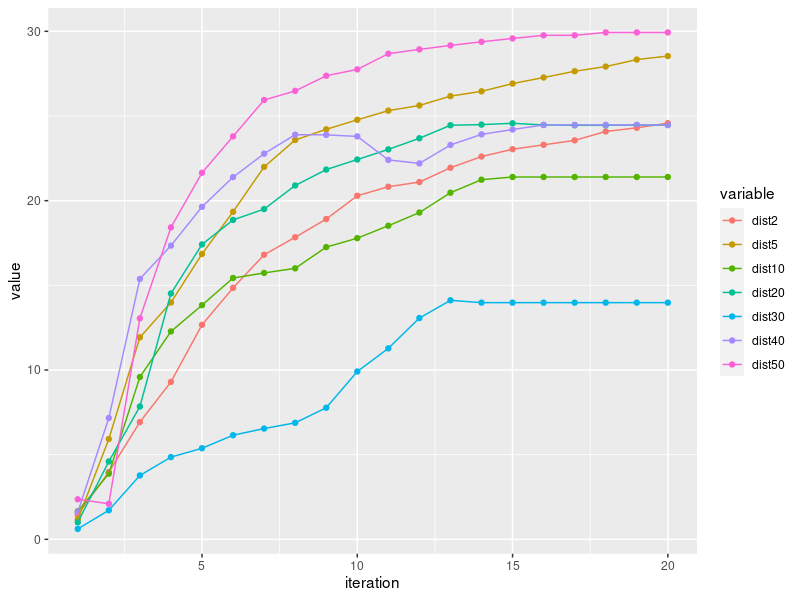
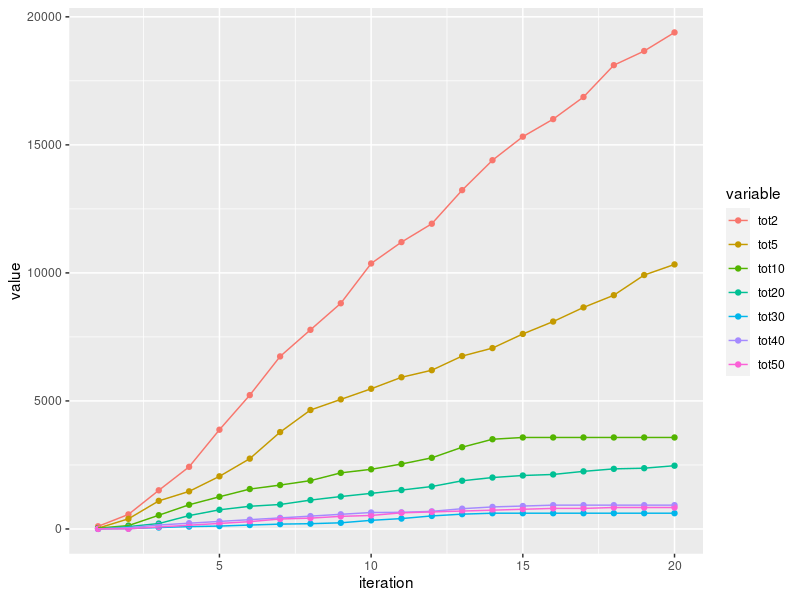
**Part B maybe also appendix)**

Exploring how far apart cath families are



Part C, distance experiment)

Given a database density how rapidly are sequence recruited to a psi-blast search



Part D/E, drift experiment)

Given a background density can we see drift in artificial databases. Compare random substitutions and BLOSUM substitutions. How does it compare to a real PFAM/CATH search.

PART E CATH/PFAMBLAST GROWTH EXPERIMENT)

What does PSI-BLAST growth look like with CATH or PFAM

PART F ESM Transformer encoding MSA & SINGLE)

IF WE PICK SOME NON DRIFT FAMILIES WHAT DO THEY LOOK LIKE

PART G HHBlits encoding)

IF WE PICK SOME NON DRIFT FAMILIES WHAT DO THEY LOOK LIKE

APPENDIX A

A number of synthetic protein datasets were constructed using two protocols. The goal here was to build protein datasets where we could control the density of the protein sequence space. This would allow us to observe the performance of iterative search away from the complexities of the real evolutionary sequence space. The space of real proteins is known not to be uniformly populated, some families have a great many closely related members and other regions of the protein space are very sparsely populated. For the synthetic protein datasets we attempt to make uniformly populated sequence datasets or differing degrees of sparsity with 100,000 members. In to these datasets we could insert densely populated regions to mimic the presence of “discrete” protein families and we could control exactly how far apart these were.  
  
To generate one of these synthetic datasets we selected a protein sequence with 120 [CHECK] residues, this length was chosen as it is the typical length for a protein domain. This sequence was our seed sequence which we then mutated by randomly selecting residues and making substitutions. Ideally we would like to generate all sequences that are a uniform distance apart form one another. However our possible sequence space is 120x20 dimensions in size and exhaustively sampling sequences would generate sequence datasets that are prohibitively large to work with. One parsimonious solution was to make n random walks away from the seed sequence such that each newly generated sequence is a fixed distance from the last.  
  
To do this two substitution protocols were used. In the first, unrealistic protocol, we take the seed sequence and select a random residue to substitute. The substitution is randomly selected from the remaining set of 19 amino acid residues. In the second protocol we randomly select residues in the seed sequence and but the new residue is selected given the residue propensities in the BLOSUM62 matrix [CITATION]. This ensures that substitutions are biased towards what is empirically observed in real protein datasets. To generate one new sequence of a given distance from the seed sequence we keep making substitutions until we reach a given BLOSUM62 score threshold. For instance if we wanted the new sequence to be a BLOSUM62 distance of 10 away from the seed sequence we would typically make around 4 substitutions. If that value was 20 we would make approximately 8 substitutions. By controlling this threshold we can adjust how far apart a new sequence is from a previous sequence and in turn control the density of the sequence space in our synthetic protein sequence databases. Multiple N(CHECK HOW MANY) data sets for the “Random” and “BLOSUM” generation protocol were adjusted this threshold with average BLOSUM distances of N, N, N, N and N (CHECK).  
  
To actually generate the synthetic database we take the initial seed sequence and generate 20 separate new sequences, this gives us the first step in 20 random walk trajectories. We then iterate this process by selecting the previously generated sequences as new seeds (20 at i+1) and generating 5 new sequences per seed. We hold this at 5 for all subsequent iterations. Sequence generation was terminate when we reached approximately 100,000 sequences. Sequences are generated such that they “fan out” from the initial seed sequence. This process was repeated for each BLOSUM threshold and for the “Random” and “BLOSUM” residue substitution protocols, generating a total of N synthetic datasets.  
  
To check that the sequences appear approximately uniformly distributed raxML [CITATION] was used to build an all-against-all distance matrix for each synthetic database. This was then visualised in 2D using MDS in R [CHECK and CITE]. This iterative sequence generation protocol was developed by adjusting the number of new sequences per “ancestor” and visualising the distribution of the sequences using MDS. This protocol, generating 20 initial sequences followed by 5 per seed in subsequent iterations, was selected as the first set of parameters which produces synthetic datasets which had near uniform densities in a 2D MDS projection. VISUALISATIONS IN THE SUPPLEMENTARY MATERIAL. PSI-BLAST, using default parameters and 20 search iterations, was then used to search these synthetic databases using the initial seed sequence. Hits discovered were record at each iteration.

Next we took the CATH structural domain database VERSION and extract N representative sequence for each S95 rep. We used raxML on this set of sequence to calculate the average distance between and with homologous superfamilies. Which gave us values of M and N. This gives us some measure of how densely populated an evolutionary family of proteins sequences is in sequence space and an estimate of the typical distance between protein sequence families.  
  
Next we took the synthetic databases where the average step distance was X. We randomly select 2 sequences separated by increasing evolutionary distances. For these two sequences we generate 500 sequences with a distance of only N (see above). Creating 10 [CHECK] new protein databases. This allows us to create sequence databases that have a background uniformly distributed set of sequences but with two densely populated sequence regions, which represent a pseudo evolutionary family. For each of the 10 databases we then take one of the two sequences and search the dataset with using PSI-BLAST for 20 iterations. At each iteration we record which sequences are discovered.