IGLOSS - Supplementary material

1 Introduction

In this Supplement, we give a detailed description of algorithms and procedures underlying IGLOSS server. Document is organized as follows: in the next Section, we describe the model building procedure. We then explain the use of the string matching algorithm from [4]. How these two are combined is described in Section 4. Finally, in the penultimate Section, we present our tests and results, while the last Section contains figures and tables.

2 Model Building

In this section we describe our model building procedure. As already mentioned, the model is computed from the input - in the first iteration - or - in subsequent iterations - from a list of positives of the previous one. The application allows for "neutral" and conserved positions in the input, where, at a neutral position - denoted by "X" - the mutation pattern corresponds to the uniform distribution of amino acids. All this complicates the model building procedure a bit, so we will deal with the first iteration separately.

In the first iteration, the input consists of one or more rows of equal width, say, N rows. We first compute sequence weights $\tilde{w}_l$ (following [3]), and then renormalize:

$$w_l = \frac{\tilde{w}_l}{\sum_l \tilde{w}_l},\ l = 1, \ldots, N. \tag{1}$$

Using $\{w_l\}$, we compute the weighted empirical distribution $\tilde{\mathcal{W}}_j$ for the column j of the input. Let m be any amino acid, and denote by $\{m_{lj}\}$ the input matrix. Then $\tilde{\mathcal{W}}_j(m)$, the position indexed by m in $\tilde{\mathcal{W}}_j$, is given by

$$\tilde{\mathcal{W}}_j(m) = \sum_{l:m_{lj}=m} w_l + \sum_{l:m_{lj}=X} \frac{1}{20} w_l. \tag{2}$$

It can be easily seen that $\tilde{\mathcal{W}}_j$ is a probability distribution (i.e. it sums to 1). We next add a small amout of pseudo-count

$$\mathcal{W}_j(m) = \frac{\tilde{\mathcal{W}}_j(m) + \frac{1}{100N}}{1 + \frac{1}{5N}}, \tag{3}$$

to obtain $\mathcal{W}_j$, a weighted distribution for each column j of the input.

We now construct evolutionary pseudo-counts. Let $\mathcal{P}$ stand for PAM120 matrix i.e. a 120-th power of the PAM matrix [1], and let $\mathcal{P}(m)$ denote the vector corresponding to the amino acid m. We use $\mathcal{W}_j$ as coefficients in a convex combination, that is, we set

$$\mathcal{N}_j = \sum_{k=1}^{20} \mathcal{W}_j(k)\mathcal{P}(k), \tag{4}$$

and obtain $\mathcal{N}_j$, an evolution-model for column j of the input. We combine this with the weighted model $\mathcal{W}_j$, where the convex combination coefficient depends on N, the size of the sample. Hence, let the function f be given by

$$f(x) = \begin{cases} 0.505514 - 0.00551429 \cdot x & x \in [0,36] \\ -9.05/x^2 + 9.5/x + 0.05 & x > 36 \end{cases} \tag{5}$$

The function f is strictly decreasing, linear on $[0,36]$, and polynomial for $x > 36$ (see Figure 1), and we define

$$\tilde{\mathcal{K}} = (1 - f(N))\mathcal{W} + f(N)\mathcal{N}. \tag{6}$$

We now deal with "neutral" and conserved positions. If the position j is considered conserved, we set

$$\mathcal{K}_j = \frac{1}{100}\tilde{\mathcal{K}}_j + \frac{99}{100}\mathcal{W}_j, \tag{7}$$

and, if the position j is considered neutral,

$$\mathcal{K}_j = \frac{1}{10}\tilde{\mathcal{K}}_j + \frac{9}{10}\mathcal{B}, \tag{8}$$

where $\mathcal{B}$ is the background amino acid distribution. Otherwise, we have

$$\mathcal{K}_j = \tilde{\mathcal{K}}_j. \tag{9}$$

Finally, we get our position specific scoring matrix for the first iteration - denoted $\mathcal{L}^1$ - as

$$\mathcal{L}^1 = \log \mathcal{K} - \log \mathcal{B}. \tag{10}$$

In subsequent iterations, we do not have X's in the input, so formula (2) is simplified, and model building proceeds in identical fashion. However, we use the PSSM from the first iteration to stabilize the model, hence

$$\tilde{\mathcal{L}}^k = \log\left(\frac{1}{2}(\exp \mathcal{L}^k + \exp \mathcal{L}^1)\right) \tag{11}$$

is the PSSM in the k-th iteration.

3 String-Matching Algorithm

Our search strategy is exhaustive - it amounts to evaluating our model at each possible position in the whole sample. In other words, the PSSM $\mathcal{L}^i$ - that was computed after the previous iteration - is in the i-th iteration evaluated at each position in each sequence. With that in mind, we construct, for each sequence x, the score-vector

$$v(x) = (v(x)_1, v(x)_2, \ldots),$$

where $v(x)_j$, the j-th component of $v(x)$, is defined as the score of $\mathcal{L}^i$ at j-th position of x. The vector $v(x)$ can easily be computed using window-sliding procedure, but in a situation where the sample is scanned over and over again, it is reasonable to consider the indexed-based method from [4]. In that case, $v(x)$ is computed with a loop over the length of the model (rather than the length of the sequence x), which speeds up the process. The algorithm was originally meant to work with a match-mismatch scoring function, so the only modification needed was to adapt it to work with a general PSSM, which - nota bene - includes dealing with multiple queries. But, that turns out to be straightforward.

4 Engine Algorithm

Once the model has been constructed - as in Section 2 - and the vector $v(x)$ calculated, for all sequences x in the sample, we can proceed to the next step. For each x, denote by $m(x)$ the maximum of $v(x)$. Then, standard results from extreme value theory [2] indicate that $\{m(x)\}$ should be approximately logistically distributed. This is, indeed, the case, as can be seen from Figure 2, and will be used to obtain an accurate estimate of distribution parameters.

Hence, we use *scipy* package, function *logistic fit*, to find the mean and scale parameters, μ and s, and set a threshold as $t = \mu + k \cdot s$, where k is the server input parameter. Now, any hit with a score greater than t is added to the list of positives from the current iteration, and used - in the next iteration - to build a new model.

5 Results

In this Section, we present and comment on our tests and results. We carried out GDSL-lipase search on five plant proteomes with three applications - IGLOSS (IG), PSI-BLAST (PB) and jackHMMER (JH). In order to assess and compare performance, we ran all three applications for various similarity levels and combined results into ROC-like curves.

First, a few words about notation: sequences in the sample that have been annotated as GDSL-lipase are marked as *condition positive* and their number is denoted as $|CP|$, while the rest are marked as *condition negative* (CN). Now, each of the three applications under consideration produces - for a specified similarity level - a list of positive hits, and their respective sequences are denoted as *positive* (P) - with the rest of the sample being *negative* (N) - while $|P|$ and $|N|$ denote the corresponding set sizes. We then have *true positives* (TP) and *false positives* (FP) as

$$TP = P \cap CP,\ FP = P \cap CN, \tag{12}$$

and likewise for *true negatives* (TN) and *false negatives* (FN)

$$TN = N \cap CN,\ FN = N \cap CP. \tag{13}$$

The usual way of assessing diagnostic ability of an application would be the ROC-curve, where one would plot *sensitivity* or *true positive rate*

$$TPR = |TP|/|CP| \tag{14}$$

against *false positive rate*

$$FPR = |FP|/|CN|. \tag{15}$$

However, in the present context, there is a serious disbalance between the sizes of the condition positive and condition negative sets: $|CN|$ is several orders of magnitude greater than $|CP|$, so, for any reasonable test outcome, $|FPR|$ will be close to 0. Consequently, we consider *precision* or *positive predictive value*

$$PPV = |TP|/|CP|, \tag{16}$$

use PPV and TPR as accuracy measures, and construct PPV-TPR ROC-like curve to illustrate performance of all three applications.

We performed motif-scanning on five organisms - *Arabidopsis thaliana* (AT) (TAIR9), *Oryza sativa* (OS) (MSU v7), *Solanum tuberosum* (ST) (ITAG1), *Solanum lycopersicum* (SL) (ITAG2.3) and *Beta vulgaris* (BV) (KWS2320). That consisted, for each plant, of 35 IG, 25 PB and 35 JH tests for various similarity levels, so, in total, we have carried out 95×5 tests. This enabled us to obtain a full range of $PPV—TPR$ points. Clearly, for PB and JH, similarity level is controlled by e-value, while in case of our server, scale is the required input.

Summary of our tests is presented in Figures 3 and 4, some of the results were compiled into tables in the next Section, while a complete set of results is available from the website. Figures 3 and 4 represent cumulative results, where, for example, the set CP is obtained as an union of CPs for each organism, i.e.

$$CP_{total} = CP_{AT} \cup \ldots \cup CP_{BV}, \tag{17}$$

and likewise for the rest. This gives sufficient density to obtain an (almost) smooth PPV-TPR curve for each application in Figure 3, and PPV-F1 curve in Figure 4. Here, F1 stands for the harmonic mean between PPV and TPR, hence

$$F1 = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR} \tag{18}$$

Finally, a short comment regarding the Figure 3: it can be seen that, in terms of PPV-TPR trade-off, IG outperforms PB for the whole range of PPV values, more-or-less matches JH in the lower range, and gives an improvement for values of PPV over 0.7. Note also that, in the upper range, IG registers a simultaneous improvement in PPV and TPR, which is - essentially - a natural consequence of a careful model building procedure in this context: an increase in PPV will yield a better profile, which will increase both PPV and TPR in the next iteration.

6 Figures and Tables

In this Section, we collected figures and tables that have been commented upon earlier. In Tables 1 to 5, you will find motif scanning results for GDSL Block I consensus query FVFGDSLSDA. As already mentioned, these tables represent a sample of the tests that we carried out. We assembled results of the three applications in a single table by (approximately!) matching the number of positives. The full size of the family (per organism) is in the top row, the size with isoforms counted once is in brackets, while the execution time is in seconds (tests have been carried out on an HP ProBook, intel CORE i5). Tables are ordered by increasing joint PPV values.

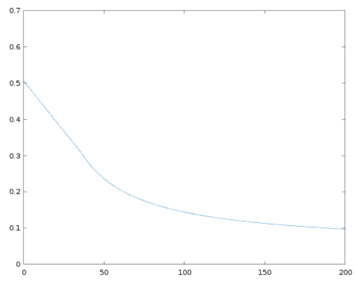

Figure 1: Graph of function f

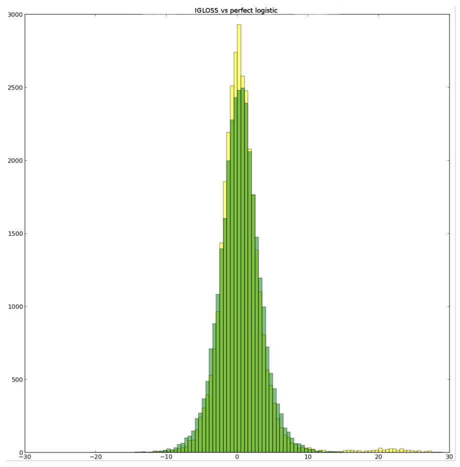

Figure 2: IGLOSS results in yellow, perfect logistic in green

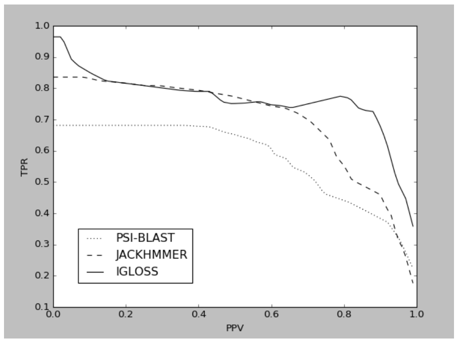

Figure 3: PPV-TPR curve

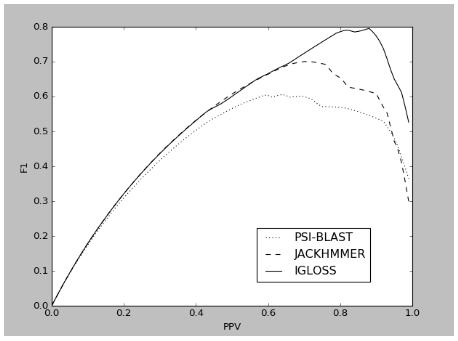

Figure 4: PPV-F1 curve

		AT	OS	ST	SL	BV	UNION
	CP	118(104)	155(116)	123(123)	108(108)	86(82)	590(533)
PSI-BLAST	TP/P	82/172	84/161	84/168	85/161	58/123	393/785
	PPV	0.48	0.52	0.50	0.53	0.47	0.50
	TPR	0.69	0.54	0.68	0.79	0.67	0.67
	EVALUE	1022	1421	845	720	860	
	TIME	0.39	0.56	0.35	0.35	0.34	
JACKHMMER	TP/P	103/172	100/162	92/169	90/161	64/123	449/787
	PPV	0.60	0.62	0.54	0.56	0.52	0.57
	TPR	0.87	0.65	0.75	0.83	0.74	0.76
	EVALUE	327	445	335	283	285	
	TIME	0.16	0.26	0.14	0.15	0.13	
IGLOSS	TP/P	80/172	120/162	92/169	93/161	62/123	447/787
	PPV	0.47	0.74	0.54	0.58	0.50	0.57
	TPR	0.68	0.77	0.75	0.86	0.72	0.76
	SCALE	6.7	8.2	6.7	6.7	6.7	
	TIME	12.18	33.86	15.66	11.32	10.16	

Table 1: Scanning results I, for various e-values and scale; joint PPV is in the last column

		AT	OS	ST	SL	BV	UNION
	CP	118(104)	155(116)	123(123)	108(108)	86(82)	590(533)
PSI-BLAST	TP/P	74/119	84/156	76/123	76/108	54/86	364/592
	PPV	0.62	0.54	0.62	0.70	0.63	0.61
	TPR	0.63	0.54	0.62	0.70	0.63	0.62
	EVALUE	640	1417	580	465	560	
	TIME	0.32	0.55	0.30	0.28	0.27	
JACKHMMER	TP/P	90/118	99/156	85/124	82/108	57/86	413/592
	PPV	0.76	0.63	0.69	0.76	0.66	0.70
	TPR	0.76	0.64	0.69	0.76	0.66	0.70
	EVALUE	172	426	224	163	183	
	TIME	0.17	0.27	0.14	0.14	0.13	
IGLOSS	TP/P	86/113	120/158	89/122	90/109	61/85	446/587
	PPV	0.76	0.76	0.73	0.83	0.72	0.76
	TPR	0.73	0.77	0.72	0.83	0.71	0.76
	SCALE	7.8	8.3	7.7	8.1	7.7	
	TIME	16.67	33.82	14.68	11.56	13.44	

Table 2: Scanning results II, for various e-values and scale; joint PPV is in the last column

		AT	OS	ST	SL	BV	UNION
	CP	118(104)	155(116)	123(123)	108(108)	86(82)	590(533)
PSI-BLAST	TP/P	73/113	79/113	73/110	76/107	50/73	351/516
	PPV	0.65	0.70	0.66	0.71	0.68	0.68
	TPR	0.62	0.51	0.59	0.70	0.58	0.59
	EVALUE	605	1200	510	460	460	
	TIME	0.32	0.53	0.29	0.27	0.26	
JACKHMMER	TP/P	86/113	87/113	80/110	82/107	52/73	387/516
	PPV	0.76	0.77	0.73	0.77	0.71	0.75
	TPR	0.73	0.56	0.65	0.76	0.60	0.66
	EVALUE	165	265	180	160	159	
	TIME	0.16	0.26	0.14	0.14	0.12	
IGLOSS	TP/P	86/113	107/113	89/110	89/107	63/73	434/516
	PPV	0.76	0.95	0.81	0.83	0.86	0.84
	TPR	0.73	0.69	0.72	0.82	0.73	0.74
	SCALE	7.9	10.8	8.5	8.5	8.2	
	TIME	16.35	36.79	17.58	11.38	10.12	

Table 3: Scanning results III, for various e-values and scale; joint PPV is in the last column

		AT	OS	ST	SL	BV	UNION
	CP	118(104)	155(116)	123(123)	108(108)	86(82)	590(533)
PSI-BLAST	TP/P	75/106	86/115	76/123	76/107	52/78	365/529
	PPV	0.71	0.75	0.62	0.71	0.67	0.69
	TPR	0.64	0.55	0.62	0.70	0.60	0.62
	EVALUE	480	900	600	470	470	
	TIME	0.29	0.36	0.27	0.29	0.26	
JACKHMMER	TP/P	80/107	88/116	86/125	82/107	56/85	392/540
	PPV	0.75	0.76	0.69	0.77	0.66	0.73
	TPR	0.68	0.57	0.70	0.76	0.65	0.66
	EVALUE	150	270	230	160	180	
	TIME	0.17	0.26	0.14	0.15	0.12	
IGLOSS	TP/P	85/106	108/114	89/124	89/108	61/85	432/537
	PPV	0.80	0.95	0.72	0.82	0.72	0.80
	TPR	0.72	0.70	0.72	0.82	0.74	0.73
	SCALE	8.5	10	7.5	8	7.7	
	TIME	16.30	31.41	13.63	15.26	13.46	

Table 4: Scanning results IV, for various e-values and scale; joint PPV is in the last column

		AT	OS	ST	SL	BV	UNION
	CP	118(104)	155(116)	123(123)	108(108)	86(82)	590(533)
PSI-BLAST	TP/P	46/61	65/77	62/71	52/56	43/48	268/313
	PPV	0.75	0.84	0.87	0.93	0.90	0.86
	TPR	0.39	0.42	0.50	0.48	0.50	0.45
	EVALUE	265	705	280	160	270	
	TIME	0.25	0.38	0.24	0.22	0.21	
JACKHMMER	TP/P	53/61	70/77	66/71	55/57	42/49	286/315
	PPV	0.87	0.91	0.93	0.96	0.86	0.91
	TPR	0.45	0.45	0.54	0.51	0.49	0.48
	EVALUE	82	172	72	51	75	
	TIME	0.15	0.26	0.13	0.14	0.12	
IGLOSS	TP/P	52/61	73/77	67/71	56/56	49/49	297/314
	PPV	0.85	0.95	0.94	1.00	1.00	0.95
	TPR	0.44	0.47	0.54	0.52	0.57	0.50
	SCALE	11.0	12.5	11.0	11.0	11.0	
	TIME	17.51	46.32	18.93	12.28	15.92	

Table 5: Scanning results V, for various e-values and scale; joint PPV is in the last column

References

[1] Dayhoff, M. and Schwartz, R. (1978) A Model of Evolutionary Change in Proteins, *In Atlas of protein sequence and structure*, pages 345-352.

[2] de Haan, L. and Ferreira, A. (2006). *Extreme Value Theory: An Introduction*. Springer.

[3] Henikoff, S. and Henikoff J. (1994) Position-Based Sequence Weights, *Journal of Molecular Biology*, **243**, pages 574-578.

[4] Ristov, S. (2016). A fast and simple pattern matching with hamming distance on large alphabets. *Journal of Computational Biology*, **23**(11), 874–876.