False Discovery Rate for Homology Searches

Hyrum D. Carroll[1], Alex C. Williams[1],
Anthony G. Davis[1], and John L. Spouge[2]

[1] Middle Tennessee State University
Department of Computer Science
Murfreesboro, TN 37132, United States of America
Hyrum.Carroll@mtsu.edu, {acw4a,agd2q}@mtmail.mtsu.edu
[2] National Center for Biotechnology Information
Bethesda, MD 20894, United States of America
spouge@ncbi.nlm.nih.gov

Abstract. While many different aspects of retrieval algorithms (*e.g.*, BLAST) have been studied in depth, the method for determining the retrieval threshold has not enjoyed the same attention. Furthermore, with genetic databases growing rapidly, the challenges of multiple testing are escalating. In order to improve search sensitivity, we propose the use of the false discovery rate (FDR) as the method to control the number of irrelevant ("false positive") sequences. In this paper, we introduce BLAST$_{FDR}$, an extended version of BLAST that uses a FDR method for the threshold criterion. We evaluated five different multiple testing methods on a large training database and chose the best performing one, Benjamini-Hochberg, as the default for BLAST$_{FDR}$. BLAST$_{FDR}$ achieves 14.1% better retrieval performance than BLAST on a large (5,161 queries) test database and 26.8% better retrieval score for queries belonging to small superfamilies. Furthermore, BLAST$_{FDR}$ retrieved only 0.27 irrelevant sequences per query compared to 7.44 for BLAST.

1 Introduction

In response to a query, many database search algorithms (*e.g.*, BLAST [2]) return a sorted retrieval list of sequences with an E-value assigned to each sequence. Typically, each E-value is calculated from a statistical model of irrelevant database sequences and approximates the expected number of irrelevant sequences with a score equal to or better than the one calculated. Many algorithms truncate their retrieval lists at a uniform E-value threshold. We call this truncation procedure "uniform E-value thresholding". While many different aspects of BLAST have undergone rigorous examination, uniform E-value thresholding has not had the same scrutiny.

As computing potential and the sophistication of computer algorithms increase, so has the need to account for multiple testing. For homology searches, the query is compared against each sequence in the database independently, resulting in multiple tests. Performing multiple tests can give the perception of a

J.C. Setubal and N.F. Almeida (Eds.): BSB 2013, LNBI 8213, pp. 194–201, 2013.

more significant result than the data can support. False discovery rate (FDR) methods aim to control the proportion of irrelevant matches to address the issues that multiple testing introduces. They are widely used in microarray studies and virtually in all facets of genomic studies. Additionally, a FDR approach was recently used to aid in generating the DFam database [13].

Early efforts for managing the false positive rate aimed to control the Family-wise Error Rate (FWER), the likelihood of making one or more false discoveries. Due to the intrinsic nature of how the FWER is computed, FWER methods also provide control over the FDR. Four modern and traditionally-accepted FWER methods are the Bonferroni correction [4], the Holm step-down procedure [10], the Hochberg step-up procedure [9], the Hommel single-wise procedure [11]. The Bonferroni correction uses a uniform P-value threshold determined by a user-specified α (or P-value threshold) divided by the total number of performed tests. The Holm step-down procedure extends the Bonferroni correction by adding the rank of the ordered P-values to the total number of performed tests in the thresholding method. Like the Holm procedure, the Hochberg step-up process utilizes the rank in the thresholding method by looking for the P-value that is less than a user-specified α divided by the total number of performed tests in addition to the current P-value's rank. The Hommel single-wise procedure is similar in that it looks for the P-value for which all P-values with a higher rank are greater than a number proportional to α. Procedures designed to control only the FDR, such as the Benjamini-Hochberg procedure [3], are generally less conservative forms of measurement than FWER methods and never perform worse. The Benjamini-Hochberg method computes a threshold by multiplying the current P-value's rank by a user-specified α and dividing the result by the total number of performed tests.

In this paper, we explore the performance of BLAST$_{FDR}$, a BLAST variant that uses E-values to calculate the FDR. We demonstrate that BLAST$_{FDR}$ performs better than BLAST, in part by drastically decreasing the number of irrelevant sequences. The Methods section presents the implementation details of BLAST$_{FDR}$; the Results section details our testing procedures and their results. We conclude with a discussion of BLAST$_{FDR}$'s applicability.

The C++ source code for BLAST$_{FDR}$ and instructions are available at `http://www.cs.mtsu.edu/~hcarroll/blast_fdr/`.

2 Methods

BLAST accepts a sequence as a query to search for relevant matches in a specified database. Additionally, an E-value threshold may be supplied to BLAST. BLAST looks for all relevant matches between that query and the sequences in a database and then applies uniform E-value thresholding by ignoring all matches with an E-value above the specified value.

BLAST$_{FDR}$ extends version 2.2.27 of NCBI's BLAST algorithm by replacing uniform E-value thresholding with a one of the following algorithms: Bonferroni, Holm's step-down process, Hochberg's step-up process, Hommel's single-wise

process, and Benjamini and Hochberg's method. The Bonferroni method calculates a threshold value for each sequence retrieved and considers the first k ranked sequences as significant that satisfy the following criterion: $P_k \leq \frac{\alpha}{m}$, where P_k is the P-value of the k^{th} sequence and m is the size of the database searched. Because BLAST relies heavily on E-values instead of P-values, and given that E-value = P-value * m, we implemented the Bonferroni method as: $E_k \leq \alpha$ with E_k being the E-value of the k^{th} sequence. Furthermore, the Holm method considers matches significant that meet the following criterion: $E_k \leq \frac{m\alpha}{m+1-k}$. Similarly, the Hochberg method takes a different approach by starting at the least likely match and working toward the best statistical score to consider the following matches as significant: $E_k \leq \frac{m\alpha}{m+1-k}$. The Hommel method also iterates from the least significant match to find the index k such that: $E_{m-k+j} > \frac{j\alpha}{k}$ for $j = 1, \ldots, k$, then uses k to consider the following matches significant: $E_k \leq \frac{m\alpha}{k}$. Finally, the Benjamini-Hochberg method iterates from the match with the best statistical score and uses the following criterion for significant matches: $E_k \leq k\alpha$.

Each match in BLAST is called a high scoring pair (HSP). A database sequence can have multiple HSPs. BLAST organizes all of the HSPs according to the database sequence to which they belong and maintains its internal data structures sorted by the best HSP per database sequence. This is problematic for applying the methods above. Consequently, BLAST$_{FDR}$ restructures the HSPs from sorted by sequence to sorted by individual scores before applying the threshold.

To determine retrieval efficacy, we leveraged the query sequences in the ASTRAL40 database [6]. Each sequence in the ASTRAL40 database has less than 40% sequence identity to the other sequences. More importantly, each sequence has been classified into a "superfamily". We only considered the queries that have at least one other superfamily member in the database. Matches with the sequences in the same superfamily are considered relevant matches. To avoid making erroneous assignments, we ignore matches that are not in the same superfamily as the query sequence. For irrelevant matches, we augmented this database 100-fold with random sequences drawn from the distribution of amino acids residues and length of sequences found in the original ASTRAL40 database.

We partitioned the augmented database into Training and Test databases. We sorted the queries by name, and assigned the 5,162 odd sequences to the Training database and the 5,161 even sequences to the Test database [1]. Additionally, we randomly selected 103 queries (2%) from the training dataset to use to evaluate which method to use. We refer to this subset as "Training-subset".

In this study, we utilize the Threshold Average Precision (TAP) [5] method as the evaluation criterion for retrieval efficacy. The TAP method calculates the median Average Precision-Recall with a moderate adjustment for irrelevant sequences just before the threshold. TAP values range from 0.0 for a retrieval with no relevant sequences to 1.0 for a search that retrieves all of the relevant sequences and only relevant sequences.

To determine the best performing method to use from the list above, we examined the retrieval performance for each one of them with $\alpha = \{0.0005, 0.005,$

Table 1. Average BLAST$_{FDR}$ TAP values using the Training-subset database

	α			
Method	0.0005	0.005	0.05	0.5
Bonferroni	0.163	0.170	0.198	0.199
Holm	0.163	0.170	0.198	0.199
Hochberg	0.081	0.088	0.102	0.150
Hommel	0.163	0.170	0.198	0.199
Benjamini-Hochberg	0.168	0.180	0.203	0.184

Table 2. Average BLAST$_{FDR}$ TAP values using the Training database

	α			
Method	0.0005	0.005	0.05	0.5
Benjamini-Hochberg	0.199	0.215	0.229	0.220

0.05, 0.5} using the Training-subset database. From these methods, we adopted the best performing one as the default threshold method in BLAST$_{FDR}$. We then evaluated that method with $\alpha = \{0.0005, 0.005, 0.05, 0.5\}$ using the entire training database. Finally, the best performing method with the best performing value of α was compared against BLAST using the Test database.

3 Results

To evaluate the performance of BLAST$_{FDR}$, we performed several experiments involving five different methods to account for multiple testing. We utilized an augmented version of the ASTRAL40 database (see the Methods section). We measure the performance in terms of the Threshold Average Precision (TAP) value.

First, we evaluated BLAST$_{FDR}$ with the following methods for determining the threshold for matches: Bonferroni correction, Holm step-down procedure, Hochberg step-up procedure, Hommel single-wise procedure and Benjamini-Hochberg. For each method, we set $\alpha = \{0.0005, 0.005, 0.05, 0.5\}$ on the Training-subset database (see Table 1). Of these methods, BLAST$_{FDR}$ with the Benjamini-Hochberg method received the best average TAP value of 0.203 and generally performed better than the other methods. Consequently, we adopted this method as the default for BLAST$_{FDR}$. For comparison purposes, BLAST received an average TAP value of 0.171 on the same database.

Table 3. Average TAP values for BLAST and BLAST$_{FDR}$

Database	BLAST	BLAST$_{FDR}$
Training-subset	0.171	0.203
Training	0.203	0.229
Test	0.198	0.226

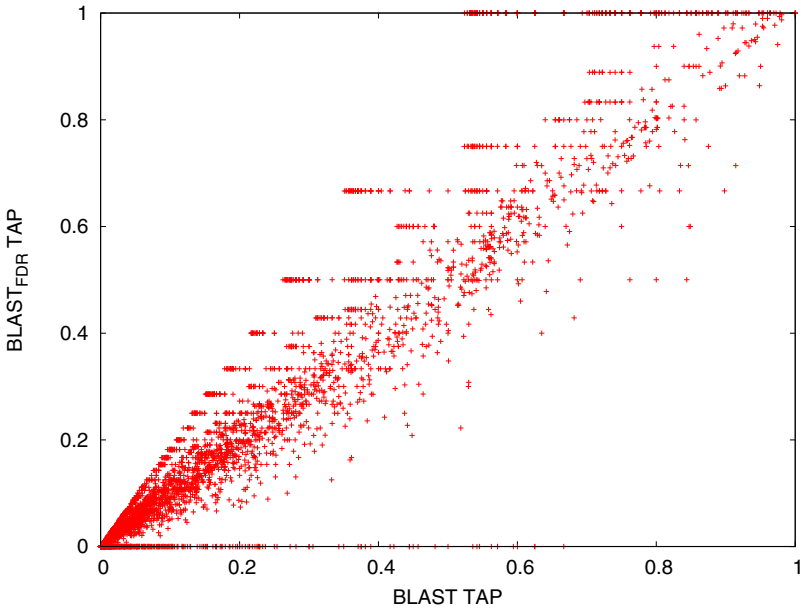

Fig. 1. TAP results for every query in the Test database

On the (full) Training database, we evaluated the same four α values for BLAST$_{FDR}$ using the Benjamini-Hochberg method (see Table 2). Of these parameters, BLAST$_{FDR}$ with $\alpha = 0.05$ received the best average TAP of 0.229 while BLAST received 0.203. Consequently, we adopted this α level as the default for BLAST$_{FDR}$.

We evaluated the efficacy of BLAST and BLAST$_{FDR}$ using the 5,161 query sequences in the Test database. Table 3 summarizes the results and Figure 1 details the TAP values for BLAST plotted against the TAP values for BLAST$_{FDR}$ for each of the queries. While BLAST received an average TAP value of 0.198, BLAST$_{FDR}$ earned an average TAP value of 0.226. In terms of irrelevant sequences, BLAST$_{FDR}$ retrieves an average of only 0.27 irrelevant sequences per query whereas BLAST retrieves 2,780% more with 7.44 per query. Finally, Figure 2 is a histogram of the E-values of sequences retrieved by BLAST that

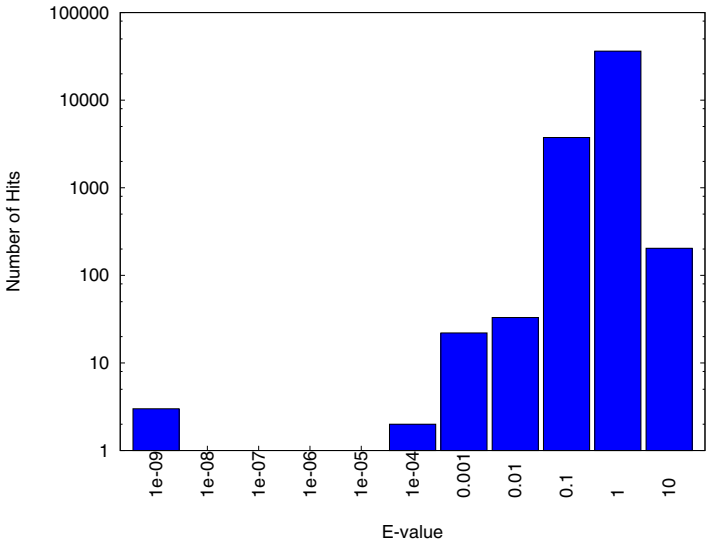

Fig. 2. Histogram of the E-values of sequences in the Test database declared significant by BLAST but not by BLAST$_{FDR}$

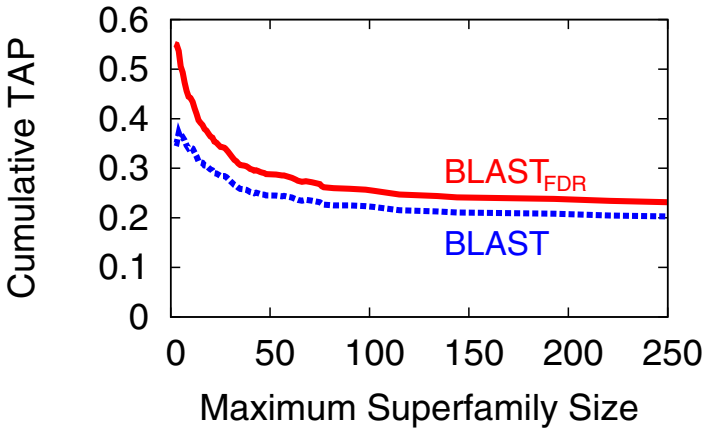

Fig. 3. Cumulative BLAST$_{FDR}$ TAP and BLAST TAP versus aggregate superfamily size for the Test database

were not retrieved by BLAST$_{FDR}$. For every dataset in the Test database, the retrieval list for BLAST$_{FDR}$ was shorter than the respective list for BLAST.

Furthermore, BLAST$_{FDR}$ performs notably better on datasets that belong to small superfamilies. Figure 3 illustrates this with the cumulative average TAP for both BLAST$_{FDR}$ and BLAST for ascending superfamily sizes. For example, for superfamilies with a size of twelve or fewer members, BLAST$_{FDR}$ has a TAP of 0.421 and BLAST a TAP of 0.332.

Similar results are obtained by using each of the ASTRAL40 database queries and searching in the NR database for up to five iterations and then using the resulting PSSM on the augmented database (data not shown).

4 Discussion

In this article we discussed an observed deficiency in the control of the proportion of irrelevant records in retrieval algorithms. Including too many irrelevant sequences has been shown to corrupt searches in a genetic database search algorithm [7]. To address this issue, we propose BLAST$_{FDR}$, an implementation of BLAST that exercises a false discovery rate method, for finer control over the percentage of irrelevant sequences.

Using accepted evaluation procedures, BLAST$_{FDR}$ had an average TAP value 14.1% higher than BLAST on the ASTRAL40 Test datasets. This difference is significant given the extremely wide use that BLAST enjoys. Furthermore, BLAST$_{FDR}$ is particularly appropriate for queries with small superfamily sizes as evidenced by it obtaining an average TAP value 26.8% higher than BLAST for superfamilies with sizes up to and including 12. For queries in larger superfamilies, if the goal is to assign function to a query, then adequately identifying the superfamily is sufficient. For example, retrieving 50% of a large superfamily clearly indicates which superfamily the query belongs. This objective is not currently captured in retrieval evaluation metrics and may make evaluation values misleading for large superfamilies.

While BLAST$_{FDR}$ does show significant performance improvements over BLAST, the increase was not seen for all queries. For example, Figure 1 illustrates that there are several datasets in the Test database that BLAST$_{FDR}$ receives a TAP value of 0.0 but BLAST achieves a non-zero TAP value. Clearly some improvements can be made to BLAST$_{FDR}$ to improve its performance.

Traditionally, the Receiver Operating Characteristic (ROC$_n$) [8] method has served as an evaluation criterion for retrieval efficacy. The ROC$_n$ method ignores the threshold implied by a homology search algorithm and truncates a list of matches after the n^{th} irrelevant match. The resulting list of matches is plotted with the number of irrelevant matches on the x-axis and the proportion of relevant matches on the y-axis. A ROC$_n$ score is then the normalized area under the curve. Typically, $n = 50$. The ROC$_n$ method was not suitable for this study as it generally requires the threshold imposed by the algorithm to be artificially modified to allow for n irrelevant matches, thus erasing the affect of the threshold method.

While we used BLAST as an example in this study, other retrieval algorithms that use uniform thresholding could also benefit from the implementation of a FDR controlled threshold. Furthermore, employing more advanced false discovery rate methods, such as the Q-value method [12] could also yield improvements. Implementation of the Q-value, because it requires the entire distribution of statistical scores, is inherently challenging for a heuristic algorithm like BLAST.

References

1. Altschul, S., Gertz, E., Agarwala, R., Schäffer, A., Yu, Y.: PSI-BLAST pseudocounts and the minimum description length principle. Nucleic Acids Research 37(3), 815–824 (2009)
2. Altschul, S.F., Madden, T.L., Schäffer, A.A., Zhang, J., Zhang, Z., Miller, W., Lipman, D.J.: Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Research 25(17), 3389–3402 (1997)
3. Benjamini, Y., Hochberg, Y.: Controlling the False Discovery Rate: a Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society, Series B 57, 289–300 (1995)
4. Bonferroni, C.E.: Il calcolo delle assicurazioni su gruppi di teste. Tipografia del Senato (1935)
5. Carroll, H.D., Kann, M.G., Sheetlin, S.L., Spouge, J.L.: Threshold Average Precision (TAP-k): A Measure of Retrieval Efficacy Designed for Bioinformatics. Bioinformatics 26(14), 1708–1713 (2010)
6. Chandonia, J., Hon, G., Walker, N., Lo Conte, L., Koehl, P., Levitt, M., Brenner, S.: The ASTRAL Compendium in 2004. Nucleic Acids Research 32(Database Issue), D189–D192 (2004)
7. Gonzalez, M., Pearson, W.: Homologous over-extension: a challenge for iterative similarity searches. Nucleic Acids Research 38(7), 2177–2189 (2010)
8. Gribskov, M., Robinson, N.: Use of receiver operating characteristic (ROC) analysis to evaluate sequence matching. Computers and Chemistry 20(1), 25–33 (1996)
9. Hochberg, Y.: A sharper Bonferroni procedure for multiple tests of significance. Biometrika 75(4), 800–802 (1988)
10. Holm, S.: A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics, 65–70 (1979)
11. Hommel, G.: A stagewise rejective multiple test procedure based on a modified Bonferroni test. Biometrika 75(2), 383–386 (1988)
12. Storey, J.: A direct approach to false discovery rates. Journal of the Royal Statistical Society: Series B (Statistical Methodology) 64(3), 479–498 (2002)
13. Wheeler, T.J., Clements, J., Eddy, S.R., Hubley, R., Jones, T.A., Jurka, J., Smit, A.F., Finn, R.D.: Dfam: a database of repetitive DNA based on profile hidden Markov models. Nucleic Acids Research 41(D1), D70–D82 (2013)