Typing *Staphylococcus aureus* using the *spa* gene and novel distance measures

Submitted to Special Issue on Computational Intelligence Approaches in Computational Biology and Bioinformatics

Phaedra Agius, Barry N. Kreiswirth, Steve Naidich, Kristin P. Bennett *

**Abstract**—We develop an approach for identifying groups or families of *Staphylococcus aureus* bacteria based on genotype data. With the emergence of drug resistant strains, *S. aureus* represents a significant human health threat. Identifying the family types efficiently and quickly is crucial in community settings. Here, we develop a hybrid sequence algorithm approach to type this bacteria using only its *spa* gene. Two of the sequence algorithms we used are well established, while the third, the Best Common Gap-Weighted Sequence (BCGS), is novel. We combined the sequence algorithms with a weighted match/mismatch algorithm for the *spa* sequence ends. Normalized similarity scores and distances between the sequences were derived and used within unsupervised clustering methods. The resulting *spa* groupings correlated strongly with the groups defined by the well-established Multi locus sequence typing (MLST) method. Spa typing is preferable to MLST typing which types seven genes instead of just one. Furthermore, our *spa* clustering methods can be fine-tuned to be more discriminative, to identify new strains that the MLST method may not. Finally, we performed a multi-dimensional scaling on our distance matrices to visualize the relationship between isolates. The proposed methodology provides a promising new approach to molecular epidemiology.

**I. INTRODUCTION**

*Staphylococcus aureus* is a common bacteria that often resides quite harmlessly on the skin or in the nose of a healthy person. However, a break in the skin or a weakened immune system can trigger the bacteria to cause minor skin infections or life threatening diseases such as pneumonia, septicemia and meningitis. Such infections can be passed on through skin contact or contaminated objects. It is no wonder then that *Staphylococcus aureus* is the leading cause of hospital-acquired infections in the United States. Controlling and understanding *S. aureus* in both hospital and community settings is a significant public health concern that is underscored by the continuous evolution and development of antibiotic-resistant *S. aureus* (MRSA). Recent advances in biotechnology allow molecular epidemiology to play an increasing role in the control of *S. aureus* [2]. Genotyping this bacteria is critical for the study of strain origin, epidemiology of strain outbreaks, and clonal relatedness, and is an invaluable tool for the investigation of hospital outbreaks.

Several different DNA sequencing methods for *S. aureus* strain typing have been proposed [4]. The strain typing must be efficient in terms of cost and time and must be highly reproducible. Sequencing of the full genome is too expensive. Macrogenetral analysis by pulsed-field gel electrophoresis (PFGE) distinguishes highly related strains and would therefore be suitable for an outbreak investigation. However, it is difficult to standardize and it is more time consuming than PCR-based methods since it requires culturing the bacteria. On the other hand, multi locus sequence typing (MLST), a PCR-based method, has been proven to be quite effective, reliable and reproducible. This method sequences the internal fragments of seven house-keeping genes producing an allelic profile of these genes [5].

Typing *S. aureus* by using the short sequence repeat region of the protein A (*spa*) gene has been suggested to work as well as the MLST method [4], [5]. Protein A in this bacteria acts as an immunological disguise and is considered to be a potent virulence factor. Given that *spa* typing involves the sequencing of just one gene, it has significant advantages in terms of speed, ease of use, standardization, and reproducibility as compared to the MLST method and other techniques [5].

In this work, we investigate potential math mod-
els based on *spa* typing for the molecular epidemiology of *S. aureus*. The current use of typing of *S. aureus* is limited. For example in a hospital outbreak setting, two isolates of *S. aureus* that share the exact same *spa* type are likely to be from the same source. But if the *spa* types are different, then interpretation of the *spa* type depends on expert opinion and experience. The isolates could be either minor variations of the same strain or they could represent different sources of infection. Thus the goal in this work is to create a measure of the similarity of *spa* types and use them to characterize the relatedness of different *S. aureus* strains. The eBURST algorithm has been developed to characterize and visualize the similarity of *spa* repeats that constitutes our data. The alignment and clustering methods that we discuss in this study were initially applied to the raw DNA sequences of the *spa* genes. These sequences consist of a series of repeats of approximately 24 base pairs (bp) in the short sequence repeat region. There are 90 distinct repeats identified so far, and each has been given a different alpha-numeric designation [5]. Thus each *spa* sequence can be thought of as a sequence of characters.

This paper is organized as follows. Section II describes the *spa* data and shows how the *spa* DNA sequences are preprocessed into a sequence of repeats that constitutes our data. In Sections III and IV, we show how the global alignment, the affine gap alignment and the gap-weighted subsequence algorithms can be customized to characterize *spa* similarity, and how the known biology of the sequences can be used to help generate normalized similarity scores. We also show how these similarity scores can be transformed into a distance measure in Section V. The simplest approach yields a distance function that is not a metric since the proposed similarity scores do not define a valid kernel. However using approaches from kernel methods, we can convert the similarity scores into a valid kernel that is a metric. Both types of distance measures can be successfully used in an unsupervised clustering algorithm.

In Section VI, we discuss how our methods can be used to identify related *S. aureus* groups or families. The distances are used within an agglomerative hierarchical clustering algorithm. The clusterings obtained are compared to the groups defined by the MLST method. In Sections VII and VIII we present our results which indicate a substantial relationship between MLST types and the *spa* groups that we obtain. Furthermore, our *spa* clusterings capture groups defined by the MLST method with finer variation. The results for our predictive experiments indicate that our methods can identify new strains and can classify known strains accurately as long as the set of sequences that define the classes a priori is large enough. In Section IX, we show how multidimensional scaling applied to our distance matrices can be used by an epidemiologist to visualize the relatedness of the strains. Potential extensions to our approaches are given in Section X.

II. THE *SPA* DATA

Constructing successful clustering methods for *S. aureus* requires fully exploiting the known structure of the *spa* genes. The alignment and clustering methods that we discuss in this study were initially applied to the raw DNA sequences of the *spa* genes in our dataset. The groupings obtained were not acceptable; they were not comparable to the groupings defined by the well-established MLST method. We therefore proceeded to work with *spa* sequences that are segmented into a series of repeats which we refer to as preprocessed *spa* sequences.

Previous studies [5] have identified common repeats and a high conservation within the amino acid sequences of the short sequence repeat region of the *spa* gene. These repeats have been alpha-numerically labelled. The alpha-numeric labels have
TABLE I
SAMPLE OF spa SEQUENCES

<table>
<thead>
<tr>
<th>spa sequence</th>
<th>MLST</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1-J1-M1-B1-M1-K1</td>
<td>5</td>
</tr>
<tr>
<td>T1-M1-D1-M1-G1-M1-M1-K1</td>
<td>5</td>
</tr>
<tr>
<td>T1-G2-M1-F1-B1-B1</td>
<td>20</td>
</tr>
<tr>
<td>U1-F1-K1-B1-P1-E1</td>
<td>1</td>
</tr>
<tr>
<td>T1-J1-F1-K1-B1-P1-E1</td>
<td>1</td>
</tr>
</tbody>
</table>

no intrinsic meaning other than that they represent different repeats. Each repeat is typically 24 base pairs (bp) long except for the repeats at the beginning of the sequences which are typically 27bp long. Table II shows a sample of the preprocessed spa sequences in our data with their corresponding MLST labels. Each MLST label represents a distinct allelic profile in the seven housekeeping genes that define that particular strain. The numeric MLST labels have no intrinsic ordering.

The DNA sequences for a sample of the repeats are as follows:

T1 AAA GAG GAA GAC AAC AAA CCT GGT
U1 AAA GAG GAA GAC AAC AAC AAA CCT GGT
F1 AAA GAA GAC AAC AAC AAG CCT GGC
G2 AAA GAA GAC GGC AAA AAA AAA AAG CCT GGC
K1 AAA GAA GAC GGC AAC AAA AAG CCT GGT
E1 AAA GAA GAC AAC AAC AAA CCT GGT

Note that repeats T1 and U1 have 27bp while the others have 24bp - these are repeats are commonly found at the beginning of the spa sequence. Table II shows the corresponding highly homologous protein sequences for the same repeats.

TABLE II
PROTEIN SEQUENCES FOR SOME REPEATS

<table>
<thead>
<tr>
<th>repeat</th>
<th>Amino Acid Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>KEEDNKKPG</td>
</tr>
<tr>
<td>U1</td>
<td>KEEDNNKPG</td>
</tr>
<tr>
<td>F1</td>
<td>K EDNNKPG</td>
</tr>
<tr>
<td>G2</td>
<td>K EDGKKPG</td>
</tr>
<tr>
<td>K1</td>
<td>K EDGNKPG</td>
</tr>
<tr>
<td>E1</td>
<td>K EDGNKPG</td>
</tr>
</tbody>
</table>

The data set used in this study contained spa sequences for 194 different isolates from the Center for Disease Control (CDC) and New York City (NYC) collections complete with 28 distinct MLST labels assembled by Dr. Nadich, Dr. Kreiswirth and the Public Health Research Institute. The MLST labels are randomly assigned numbers designated to the sequences based on their allelic profile and will be used for validating the results. Although only 94 of the sequences are distinct, each spa sequence comes from a different host. We therefore consider a weighted analysis of these sequences based on the number of occurrences of the sequence within our dataset. In this study, it is the organization of the repeats within the sequences and the high conservation of repeats at the tips of the sequences that will define the major differences between one spa type and another. The length of the variable region has been found to be non-indicative of the strain type [5] and therefore the impact of length differences in our alignment methods is reduced.

III. ALIGNMENT METHODS

Alignment algorithms and subsequence kernel methods are frequently used in bioinformatics [10], [11], [12]. The preprocessed sequences in this dataset are rather short since the actual DNA sequences have already been compartmentalized into repeat profiles. This permits the use of alignment and subsequence algorithms to generate similarity scores. We consider two well established alignment algorithms - global alignment and affine gap alignment [11], [12]. We use these algorithms to derive the optimal alignment for two sequences from which we can obtain a similarity score. We also define a third algorithm, the Best Common Gap-Weighted Subsequence (BCGS), which is a variation of the gap-weighted subsequence kernel method [7].

We initially applied these sequence algorithms to the entire spa sequences, as is customary with such algorithms. However, the distances we obtained in these preliminary experiments generated groupings that were hardly comparable to the groups defined by the MLST labels. Further experimentation revealed that a better way to generate distances would be to segment a spa sequence \( s = a_1a_2...a_m \) into three parts as follows: \((a_1)(a_2...a_{m-1})(a_m)\). We refer to the first and last segments \( a_1 \) and \( a_m \) as the sequence tips, and we refer to the remaining portion as the middle segment. We used a separate method to compare the sequence tips, and then we combined their scores with the scores for the middle segments in a flexible way that allows each segment
score to be weighted separately. This method is supported by previous research which suggests that the repeats at the tips of the *spa* sequences are a strong determinant of the family type [4]. Details on how these distance matrices are combined can be found in section IV.

We now proceed to describe the sequence algorithms used in this study, bearing in mind that they were applied only to the middle segment of the *spa* sequences.

### A. Global Alignment.

The Needleman and Wunsch algorithm [11] seeks the optimal alignment between two sequences by using a gap penalty and a substitution penalty. This dynamic programming algorithm maintains a score matrix so that that a simple traceback algorithm through the matrix yields an alignment that corresponds to the optimal score. From this optimal alignment, a similarity score may be obtained by counting the number of matching characters between the two aligned sequences. We normalize this score by dividing it by $\sqrt{mn}$ where $m$ and $n$ are the lengths of the sequences being compared. This method of normalization is less sensitive to the differences in the sequence lengths than, for instance, normalizing by the length of the alignment itself.

### B. Affine Gap Alignment.

This algorithm works much like the global alignment algorithm with the difference that the gap penalty is now an affine gap penalty which includes a gap initialization cost [12]. This encourages any gaps to clump together in the optimal alignment. The biological motivation behind this type of alignment is that the deletions or insertions of contiguous repeats occur all at once rather than in multiple events. In this dynamic programming algorithm, three score matrices are maintained. Once again, traversing through the score matrices via a traceback algorithm yields an optimal alignment. We then use the optimal affine gap alignment to obtain a similarity score by counting the number of matching characters in the alignment, and we normalize as we did for global alignment.

### C. Best Common Gap-weighted Subsequence.

The gap-weighted subsequences kernel [7] weights the occurrences of the common subsequences between two strings according to how spread out they are. For example, the occurrence of the subsequence ‘his’ in ‘this’ is more significant than it is in ‘mathematics’. We define the spread, $N$, to be the number of gaps interspersed within the common subsequence. The spread is weighted by an exponentially decaying weight $\lambda^N$ where $\lambda$ is a user-defined parameter between 0 and 1. We define the similarity score for the occurrence of a common subsequence of length $s$ with spread $N$ to be $s\lambda^N$. For example, the occurrence of ‘his’ in ‘mathematics’ is interspersed with five characters, and so the similarity score would be $3\lambda^5$. In our experiments, we set $\lambda = 0.9$ and we normalize the similarity score as we did for global and affine gap alignment.

BCGS is a dynamic programming algorithm that seeks all the common subsequences between two strings and calculates the similarity score as defined above for each one. The maximum similarity score is defined to be the BCGS similarity score between two sequences. Although this may often turn out to be the longest common subsequence, it is not always the case because of the influence of the $\lambda$ parameter (refer to Example 2). The BCGS algorithm is based on the Longest Common Subsequence (LCS) algorithm which generates a matrix $C$ that records the common subsequences. This matrix is then used to count the number of gaps for each common subsequence through the gapcount subroutine, and a matrix $L$ is generated.

### IV. Sequence Tips and Similarity Scores

In the previous section we briefly described our method of combining the distance matrices for the sequence tips and the middle segment. In this section, we discuss the details for this procedure.

Although previous research suggests that the *spa* sequence tips play a crucial role in determining the family type, a glance at the dataset indicates that this is not the only determinant of the family types. Typing the bacteria solely on the basis of matching the sequence tips would generate a grouping that is not sufficiently similar to the MLST grouping. The question that needs to be addressed is, how crucial are these sequence tips in determining the family
type? We address this question by assigning weights to the matches or mismatches of the tip repeats, and then combining these weighted similarity scores with the similarity score for the middle segments. The details are as follows.

The normalized similarity score between two sequences \( s_1 = a_1a_2...a_m \) and \( s_2 = b_1b_2...b_n \) is defined to be 
\[
S(s_1, s_2) = B \cdot m(a_1, b_1) + E \cdot m(a_m, b_n) + (1 - B - E) \cdot f(a_2...a_{m-1}, b_2...b_{n-1}),
\]
where \( m(a_i, b_j) = 1 \) if \( a_i = b_j \) and 0 otherwise, and \( f \) is the similarity function defined by one of the three algorithms mentioned above. Since the repeats at the beginning and end of the sequences deserve special attention, percentage weights \( B \) and \( E \) are assigned to these repeats such that \( B + E \leq 1 \). In essence, the ends of the sequences are pegged together and a separate similarity score is generated for the middle segments of the sequences.

The following example illustrates and contrasts the global and affine gap alignment methods and uses a percentage weight of 0.2 for the tip repeats. We denote the similarity scores obtained using the BCGS, Affine and Global sequence algorithms as \( S_B, S_A \) and \( S_G \) respectively. Example 2 uses the BCGS algorithm for the middle segment with the same weights for the sequence tips.

**Example 1:** The table below shows an optimal global alignment and an affine gap alignment for the middle portions of the spa sequences U1-J1-G1-F1-B1-B1-B1-B1-P1 and T1-J1-B1-B1-B1-B1-B1.

<table>
<thead>
<tr>
<th></th>
<th>J1</th>
<th>G1</th>
<th>F1</th>
<th>B1</th>
<th>B1</th>
<th>B1</th>
<th>B1</th>
<th>P1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Affine</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The scores for the begin and end repeats are \( m(U1, T1) = 0 \) and \( m(B1, B1) = 1 \). Therefore, the final global and affine similarity scores respectively are
\[
S_G(s_1, s_2) = 0.2 \cdot 0 + 0.6 \cdot \sqrt{\frac{4}{8^4}} + 0.2 \cdot 1 = 0.624
\]
\[
S_A(s_1, s_2) = 0.2 \cdot 0 + 0.6 \cdot \sqrt{\frac{3}{8^4}} + 0.2 \cdot 1 = 0.518.
\]

**Example 2:** Consider the sequences T1-K1-M1-J1-D1-D1-R1-Q1 and T1-K1-B1-J1-R1-E1. The un-normalized BCGS scores for the middle portions of these sequences with \( \lambda = 0.9 \) and \( \lambda = 0.8 \) are given in the following table.

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>( \lambda )</th>
<th>( \lambda )</th>
</tr>
</thead>
<tbody>
<tr>
<td>K1 M1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>K1 B1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>K1</td>
<td>1</td>
<td>1.8</td>
<td>1.6</td>
</tr>
<tr>
<td>K1 J1</td>
<td>3</td>
<td>1.46</td>
<td>1.02</td>
</tr>
<tr>
<td>K1 J1</td>
<td>4</td>
<td>1.97</td>
<td>1.23</td>
</tr>
</tbody>
</table>

When \( \lambda = 0.9 \) the longest common subsequence K1-J1-R1 is the optimal choice, but when \( \lambda = 0.8 \) the shorter subsequence K1-J1 is optimal.

The final similarity score for \( \lambda = 0.9 \) is
\[
S_B(s_1, s_2) = 0.2 \cdot 1 + 0.6 \cdot \frac{1.97}{\sqrt{4}} + 0.2 \cdot 0 = 0.423
\]

**V. The Distance Matrices**

From the normalized similarity scores that represent the *spa* sequences, we can now obtain a distance matrix that will represent the differences between the sequences.
In general, the similarity score $S(s_i, s_j)$ between the sequences $s_i$ and $s_j$ can be transformed to a distance measure $D(s_i, s_j)$ as follows [7]:

$$D(s_i, s_j) = \sqrt{S(s_i, s_i) - 2 \cdot S(s_i, s_j) + S(s_j, s_j)}.$$

In the case where similarity scores are normalized, $S(s_i, s_i) = S(s_j, s_j) = 1$, so that the above formulation can be reduced to

$$D(s_i, s_j) = \sqrt{2 - 2 \cdot S(s_i, s_j)}.$$

We refer to the matrix of such distances as $D$ where $D_{i,j}$ is the distance between the $spa$ sequences $s_i$ and $s_j$. Note that although this distance matrix is symmetric, it is not a valid distance metric because the triangular inequality is not observed.

We can convert the similarity scores to a valid distance metric by using an approach from kernel methods. A valid kernel is a similarity measure that forms a positive definite similarity matrix over the data. Using the kernel to induce the distances guarantees that it is a metric.

To do this we map the sequence information into a feature space defined with respect to a training set of $spa$ sequences. So, for a sequence $s_i$ in the dataset, the mapping to feature space is defined as

$$\phi(s_i) = [S(s_i, s_1), S(s_i, s_2), ... S(s_i, s_N)]$$

where $\{s_1, s_2, ... s_N\}$ is the set of sequences in the dataset with $s_i$ being a member of that set, and $S$ is the similarity function as described in section IV.

The kernel distance between two sequences $s_i$ and $s_j$ can therefore be defined as follows:

$$K(s_i, s_j) = ||\phi(s_i) - \phi(s_j)||^2$$

This approach has been applied to other non-metric sequence similarity measures to convert them to valid kernels. For example, the Smith-Waterman Kernel is not a valid kernel [8] but it was converted to one in a similar fashion and used in [9].

VI. CLUSTERING, TRAINING AND TESTING

We seek to show that $spa$ typing is at least as discriminative and accurate as MLST typing, a more costly approach. We therefore restrict the use of MLST labels to evaluative purposes and for tuning the training model. In order to be able to generalize our results, we split our dataset into a training and testing set. A training model is built on the training sequences and the test sequences are then fed into this model. The pairwise distances that now represent the sequences were used within the hierarchical clustering algorithm in Matlab to cluster the training sequences. We call the set of groups or clusters obtained a clustering. The $spa$ groups obtained were then compared to the MLST groups, and the likeness of these two groupings was measured using the Jaccard index. This is defined to be

$$\frac{N_{11}}{N_{11} + N_{10} + N_{01}},$$

where $N_{11}$ is the number of point pairs that occur in both clusterings, $N_{10}$ is the number of point pairs that occur in the first clustering but not in the second, and $N_{01}$ is the number of point pairs that occur in the second clustering but not in the first. For identical clusterings the Jaccard index is 1.

The Jaccard index was used to guide the similarity parameter choices. The choice of the number of clusters to be formed or the cutoff in the hierarchical tree, $c$, was also guided in this manner. The $B$ and $E$ parameters were varied over the ranges of 0.05 to 0.3 at 0.05 intervals and the $c$ parameter was varied from 15 to 30 at unit steps. The optimal training set clustering was defined to be the one with the highest Jaccard index with respect to MLST. In our work, the single-link hierarchical clustering method in Matlab 7.0 was found to work slightly better than the average-link or weighted-average hierarchical clustering methods, so all results reported are for single-link hierarchical clustering.

Test set sizes were varied from 5 to 50 percent of the entire dataset. The test set was extracted from the set of 94 unique $spa$ sequences. Each test sequence was then weighted by the number of occurrences of that sequence within the entire dataset. This ensured that there were no sequences common to both the training and test sets. At the same time the biological significance of the multiple occurrences of a particular sequence was maintained in our model.

The classification of a test sequence entails a comparison between that sequence and every training cluster. The distance between a test sequence and a cluster was defined to be the mean of the distance between the sequence and the members of that cluster. A condition was imposed to identify new strains in the test set that should not be absorbed by any of the training clusters. If the distance between a test sequence and the nearest cluster falls
above an outlier threshold, then the test sequence was considered to be a singleton and was assigned a new cluster by itself. The outlier threshold $T$ was defined using the mean of the distances within the training clusters $\bar{d}$ and the standard deviation $d_{\sigma}$ of those distances. A sequence was defined to be an outlier if the distance between itself and the nearest cluster was greater than $\bar{d} + T \times d_{\sigma}$. We ran experiments to examine the effects of varying the number of standard deviations $T$ that defines the threshold (refer to section VIII).

VII. EVALUATING THE RESULTS

The classification of the test sequences is evaluated using a measure which we call the accuracy score. Suppose a spa cluster $C$ defined by the training set contains sequences with different MLST labels. The label shared by the majority of the members of $C$ is defined to be the MLST label for all the sequences in that cluster and for any new sequence assigned to that cluster. The accuracy score is defined to be the proportion of correctly labeled test sequences.

A spa sequence may have a MLST label that is distinct from those of the rest of the sequences in the training set. We refer to such sequences as outliers or unknown strains. If an outlier is part of the test set, then our spa method should identify it as an outlier. We account for this in the accuracy score by maintaining an outlier class and counting how many are identified by our methods. However, the case when our methods cast a sequence as an outlier while MLST does not should not be penalized; the spa classification methods are being more discriminative than the MLST method. To make the accuracy reflect the agreement between MLST and spa without undue penalizing outlier disagreements, we do not count sequences identified as outliers by spa and not MLST in the accuracy statistic. The accuracy score only considers test sequences falling into spa clusters and that are identified as outliers by both MLST and spa.

The overall clustering of both the training and test sets also needs to be validated. We used the Jaccard score to compare the final spa clustering obtained to the MLST groups. This score is expected to decrease as the test set size increases, since the training set size decreases.

Finally, splitting the dataset into a training and test set just once is not sufficient. We therefore perform forty such splits and the Jaccard and accuracy scores that we report are averaged over those forty iterations. There should be a good amount of consistency in the final clusters formed over each partition. We measured this consistency by measuring the average Jaccard index between the spa clusterings formed across all the iterations. We refer to this measure of consistency as the cluster stability and would expect this value to fall gradually as the test set size increases.

VIII. COMPUTATIONAL RESULTS

In each of our experiments, we explored the performance of the three sequence algorithms to determine the most promising approach for molecular epidemiology. In our first experiment we used the entire dataset to generate a spa clustering which we then compared to the MLST groups by calculating the Jaccard scores. The three subsequence experiments involved splitting the dataset into training and test sets.

Our second experiment compared the classification results of the three sequence algorithms using the two distance methods described in Section V. The test sets used for this experiment were predominantly composed of known test sequences, that is, test sequences that shared MLST labels with at least one sequence in the training set. The third experiment explored the consequences of varying the outlier threshold when new strains were part of the test set. This experiment pointed us to a suitable outlier threshold for the fourth experiment whose test sequences were a mixture of known and new strains. The fourth experiment seeks to most closely simulate how the approach would be used for molecular epidemiology by seeing how well the algorithms simultaneously classified known strains and detected new points.

A. Using All the Data

We first examine how closely the clusters created by the spa-type corresponded to the MLST types. For this experiment, the three sequence algorithms were applied to the entire dataset. Nonmetric distances $D$ and metric distances $K$ were generated for each sequence algorithm. These distances were then used within a hierarchical clustering algorithm. Single-link and average-link hierarchical clustering was performed in each case to try to determine any
inherent differences between the methods. The $B$ and $E$ parameters were varied from 0.02 to 0.3 in 0.02 increments, while the number of clusters $c$ was varied from 15 to 30 at unit increments. The parameters with optimal Jaccard scores with respect to MLST were selected. As can be seen in Table III all variations of the $spa$ algorithm produce clusters that are very similar to the MLST types, with Jaccard scores ranging from .78 to .82. The differences in the $spa$ method across the algorithms and the hierarchical methods are slight. Used in a descriptive manner on the training data, the $D$ and $K$ distance methods exhibit little differences. However, considerable differences were evident during generalization testing with single-link hierarchical clustering performing consistently better than average-link. Therefore the results reported in the next few sections are all for single-link hierarchical methods applied to the training sets. We therefore proceed to the next experiment which examines this in a little more detail.

### Table III

<table>
<thead>
<tr>
<th></th>
<th>Single</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>BCGS</td>
<td>0.8267</td>
</tr>
<tr>
<td></td>
<td>Affine</td>
<td>0.8149</td>
</tr>
<tr>
<td></td>
<td>Global</td>
<td>0.8169</td>
</tr>
<tr>
<td>$K$</td>
<td>BCGS</td>
<td>0.8181</td>
</tr>
<tr>
<td></td>
<td>Affine</td>
<td>0.8183</td>
</tr>
<tr>
<td></td>
<td>Global</td>
<td>0.8153</td>
</tr>
</tbody>
</table>

**B. Classifying Known Strains**

In this experiment, we sought to see how well the proposed methods predicted the strains of new sequences. The data was randomly partitioned into training and test sets forty times. Each train/test iteration was subjected to each of our sequence algorithms and distance methods so that we could compare their classification results. To ensure that most strains had previously been seen, MLST types corresponding to a single $spa$ sequence were forced to be in the training set. When the test set sizes were large, there were occasions when a whole MLST family of sequences was part of the test set, rendering these strains unknown in view of the training set. This issue is discussed in more detail later on.

![Fig. 1. Comparing average Jaccard scores across the distance matrices](image1)

![Fig. 2. Comparing average Stability scores across the distance matrices](image2)

We first compared the generalization performance of the metric and non-metric distance measures. As discussed above, the data was randomly partitioned into training and test sets. All results reported are for the test sets. Figures 1, 2 and 3 show the average Jaccard, Stability and Accuracy scores respectively for the two distance matrices. The statistics for the non-metric (X-axis) and metric distance (Y-axis) are plotted for each alignment method and test set size. Points above the diagonal favor the $K$ distance, and below the line favor the $D$ distance. The two types of distance measures are roughly equivalent in terms of Jaccard and Stability scores. However, the non-metric distances performed significantly better...
on the Accuracy scores. We explored the results using both distance methods in the remaining experiments and found the non-metric distance method to consistently perform a little better. Since the non-metric distances are cheaper to compute and tend to perform slightly better, we report the results for the ensuing experiments for those distances only.

We next compared the differences between the sequence algorithms for the same experiments. Figures 4 and 5 depict average Jaccard and Stability scores for the non-metric distances in the form of bar graphs to highlight algorithmic differences. The outlier threshold was fixed at 2 standard deviations for this experiment. Clearly the Affine algorithm lags behind the other two algorithms in all the evaluative scores. The BCGS algorithm performs better than the Global algorithm on the Jaccard and Stability scores. Using a paired $t$-test across all of the results, BCGS has significantly better Jaccard and Stability scores than Global with p-values less than .001. BCGS does a better job of obtaining consistent clusterings (as measured by Stability) that correspond closest to the groups identified by MLST.

The results for outlier detection also favor BCGS. Recall that test sequences whose distance from the nearest cluster exceeds the outlier threshold are called outliers. Figure 6 shows the number of outliers detected by each algorithm with the white bars indicating the number of correctly identified outliers. A correctly identified outlier is a novel MLST type in the testing set that was not included in the training set. The other outliers may be errors,
but since spa is more discriminative we expect a some outliers predicted by spa and not MLST to be correct. BCGS does a good job of detecting the known outliers while simultaneously detecting less total outliers than the other two algorithms. We considered two possibilities for this: either the outlier threshold affects BCGS in a different way than it does the other two algorithms or BCGS is not as good at identifying outliers as it is at classifying known strains. We explore these ideas a little further in the Section C, where we vary the outlier threshold.

C. Identifying New Strains

Ultimately, the goal of our model is to accurately identify existing and novel strains of S. aureus. The identification of novel strains or outliers is dependent on the choice of parameters. The following two experiments investigate the behavior of the algorithms as the outlier threshold is varied in the hopes of determining an appropriate outlier threshold to use for an experiment with a test set composed of both known and new strains.

Recall that we define a new strain to be a spa sequence whose MLST label is distinct from the rest of the sequences in the training set. There are 12 such sequences in our data set and these sequences were designated to be the test set. Figure 8 shows the proportion of outliers identified by each algorithm as the outlier threshold is varied. These results suggest that an outlier threshold of 2 may be a little too lax with a considerable number of outlier test points slipping the net and going undetected. Note that BCGS is not as adept at identifying these outliers as Global and Affine. One may surmise that, in contrast to Global and Affine, BCGS is eager to classify strains and reluctant to cast them as outliers. But this aspect may pertain only to this small set of outlier sequences and their corresponding training set.

The average Accuracy scores shown in Figure 7 are much more variable. Notice that with respect to Stability, Jaccard scores and outliers (Figures 4, 5, 6) the results vary smoothly across test set sizes. A paired t-test across the Accuracy scores for all the experiments indicates that Global has a higher Accuracy score than BCGS with a p-value of 0.07. But it is difficult to make conclusions based on on Accuracy alone due to the influence of outliers. The Accuracy score favors methods the global and affine methods since they identify more outliers. We tried alternative methods of measure of accuracy, but all are skewed by the fact that the MLST labels may not exactly correspond to S. Aureus strains, so we cannot know for sure the true status of outliers. Thus we consider the Jaccard, Stability, and outlier results to be more indicative of the true performance of the algorithm. These all favor BCGS.

In the next experiment, we wished to explore the way the algorithms classified known strains and outliers as the outlier threshold was varied. A fixed 30 percent test set was used across the different outlier thresholds. The results were averaged over 10 trials. Figure 9 shows the Jaccard scores aver-
aged across the 10 trials for each outlier threshold. The Jaccard scores indicate an upward trend as the threshold increases. When the threshold is too low our methods pick up too many outliers, and this is penalized by the Jaccard score. The Jaccard scores start to peak at outlier thresholds between 1.5 and 2. Figure 9 indicates a subtle lead for BCGS Jaccard scores as the outlier threshold increases. As was discussed in Section B, the Global and Affine algorithms identified more incorrect outliers than did the BCGS algorithm. The Global and Affine algorithms are forced to classify sequences that they originally incorrectly labelled as outliers. In contrast, BCGS does not have this problem because it did not identify as many known sequences as outliers.

**Fig. 9.** Average Jaccard scores when the outlier threshold is varied

![Jaccard Scores Graph](image)

**D. Classifying Known and New Strains**

We now examine how the proposed methods might be used in more realistic scenario for molecular epidemiology in which we seek to correctly identify both existing and novel strains. In this experiment we investigated the performance of our algorithms on test sets that comprised known sequences and outliers. As discussed in the previous section, we lowered the outlier threshold to 1.5 standard deviations to better capture outlier sequences. Figures 10, 11 and 12 show the results obtained for the non-metric distances. The results are not too different from those obtained when the outlier threshold was set to 2 and the test sets were composed of known strains. Once again, the Affine algorithm performs worst, and BCGS is in the lead for Jaccard and Stability scores but lags behind Global alignment for the Accuracy scores. A paired t-test indicates that the differences are significant with p-values of almost 0. Figure 13 partially explains this discrepancy - as before, Affine and Global detect more outliers than BCGS causing their Accuracy scores to become artificially higher than those for BCGS. The white bars indicate little difference between the algorithms with regards to the correct detection of outliers.

**Fig. 10.** Average Jaccard scores for test sets comprising known and unknown strains, Outlier threshold=1.5

![Jaccard Scores Graph](image)

**Fig. 11.** Average Stability scores for test sets comprising known and unknown strains, Outlier Threshold=1.5

![Stability Scores Graph](image)

**IX. Visualization of Strain Relatedness**

The proposed methods can also be used to visualize the relatedness of isolate. Graphic portrayals of how related isolate are would be a valuable tool for understanding and controlling *S. Aureus*
Fig. 12. Average Accuracy scores for test sets comprising known and unknown strains, Outlier threshold=1.5

Fig. 13. Average Outlier Detection, Outlier threshold=1.5

Fig. 14. MDS picture of nine MLST types

Fig. 15. MDS of MLST types 20 and 59

in the hospital and community settings. Also we can use it to validate the models. A two or three dimensional representation of the sequence can be created by using multidimensional scaling (MDS) on our distances. MDS finds the representation of the data in two dimensions such that the stress, the squared difference between the distances in the two dimensional representation and the given distances, is minimized. The 
\texttt{cmdscale} Matlab function was used to do a MDS of the distance matrix and generate a two-dimensional plot with each sequence labeled with its MLST type.

In Figures 14 and 15, the BCGS algorithm was applied to the entire dataset and non-metric distances were generated using the optimal choice of the $B$ and $E$ parameters as reported in Table III. Figure 14 is a plot of the \textit{spa} sequences for nine MLST types visualized using \textit{spa} distances. The shapes of the points indicate the MLST type. This picture confirms that the \textit{spa} clusters appear to be well defined and conform to the MLST labels. It enabled us to note that some \textit{spa} clusters are tight while others, such as MLST type 109, are not so closely knit. The \textit{spa} sequences for this MLST type are rather short sequences, and this feature tends to magnify the differences between them. Figure 15 shows MLST types 20 and 59 using the same distances. This picture shows that these sequence distances are quite discriminative, with the sequences of MLST type 20 almost disparate enough to be split into two separate groups. In fact, when the $c$ parameter is increased, this group does get split. This is a small example of the splitting phenomenon that makes the \textit{spa} clustering a little more discriminative than MLST. The dis-
criminative power of \textit{spa} clustering can be tuned at will by tweaking the \textit{c} parameter accordingly. These MDS plots provide a viewable contrast of the \textit{spa} clustering to the MLST labels. They allow the epidemiologist to see and understand the genomic similarity and differences between isolates.

X. CONCLUDING REMARKS

The development of novel genotyping methods for molecular epidemiology represents an important new application of machine learning with great potential. Collections of data for many types of bacteria are being gathered and new methods for exploiting this data are needed. Methods are needed to define how isolates are related when they do not yield an exact match. This work represents a first effort in this regard for \textit{S. aureus}. Specifically we demonstrated how normalized similarity measures can be transformed to distance measures which can then be used to group and visualize isolates.

The practical use of genotyping information for epidemiology and control of disease requires the existence of quick inexpensive DNA finger printing methods. MLST and \textit{spa} are two competing methods for finger printing \textit{S. aureus} isolates with the \textit{spa} having the advantage that it requires sequencing of only one gene instead of seven. This work investigate developed the first similarity/distance measures for \textit{spa} and showed that they are at least as discriminative as MLST types. We developed a hybrid algorithm that combined sequence algorithms with weighted match/mismatch algorithms for the sequence tips. In this way we incorporated the known biology of \textit{S. aureus} into our similarity measures. Distances between the isolates were then deduced from the similarity scores and used within an unsupervised hierarchical clustering algorithm.

We investigated two existing and one novel sequence algorithms for use within our hybrid algorithms. Our experiments showed that all three algorithms worked well on these preprocessed sequences. Overall, the novel BCGS algorithm performed best, despite its slight inclination to include outliers within its established groups. As the availability of \textit{spa} data increases, we would like to be able to explore the discriminative features of the sequence algorithms in more detail using larger datasets. Since the MLST definition of what constitutes an outlier is limited, further work using alternative genotyping methods is needed to more clearly defined which sequences are truly outliers. With the added data and information, the proposed families and parameter choices can be suitably refined.

The evaluation process we used depended on the MLST labels associated with each sequence. Although we used an unsupervised method, we showed that MLST typing can be cheaply reproduced by \textit{spa} typing with high accuracy. Furthermore, as the number of clusters \textit{c} is increased, the \textit{spa} grouping becomes more discriminative than the MLST grouping. This indicates that these \textit{spa} sequences may be capturing some information that is not captured in the MLST allelic profiles.

We have tested our methods on dataset of 194 \textit{spa} sequences. Future work will entail the use of larger datasets, that we are assembling. The evaluation of our results rested heavily on the MLST labels. We plan to validate our results against other typing methods as well. Potentially our results could be improved by incorporating different measures for synonymous and non-synonymous DNA changes in our similarity measures. We would also like to explore the application of the BCGS algorithm and those tools to other bacteria.

REFERENCES


