NISTIR 7992

Exploring the Methodology and Utility of Standardized Latent Fingerprint Matcher Scoring

V.N. Dvornychenko
G.W. Quinn

http://dx.doi.org/10.6028/NIST.IR.7992

National Institute of Standards and Technology
U.S. Department of Commerce

NISTIR 7992

Exploring the Methodology and Utility of Standardized Latent Fingerprint Matcher Scoring

V. N. Dvornychenko
G.W. Quinn
Information Access Division
Information Technology Laboratory

http://dx.doi.org/10.6028/NIST.IR.7992

March 2014

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Under Secretary of Commerce for Standards and Technology and Director

Acknowledgements

The authors would like to thank Michael Garris and Elham Tabassi for their many valuable suggestions that went into this report.
We also like to thank the Department of Homeland Security, Science and Technology Directorate, and the Federal Bureau of Investigation's Criminal Justice Information Services Division and Biometric Center of Excellence for sponsoring this work.

Table of Contents

List of Tables

List of Figures

Abstract

Automated searches of fingerprints against a repository/database are important tools of the forensic community. Systems performing these searches are referred to as ***Automated Fingerprint Identification Systems*** *(AFISs). The output of an AFIS is a fairly small set of prospective candidates with attendant* ***matching scores*** (allowing for comparison of candidates, and frequently referred to as ***comparison scores***). *These scores provide an indication of how likely a particular candidate is a* ***true mate*** *of the search fingerprint (i.e., originates from the same individual). One difficulty in interpreting matching scores in usage is there is no accepted standard for its range and exact meaning (other than "bigger is better"). Experts need to become very familiar with the scoring of a specific system to make optimal use of the results. This report proposes that we standardize the scoring system. The standardized score becomes a number between 0 and 100 and carries two decimal places, for a total of four significant figures. Seven alternative algorithms for computing matching scores are outlined in this report, ranging from very simple to quite complex. A mid-complexity algorithm is then selected for detailed illustration/development. This report also provides a detailed analysis of scoring produced by the matchers previously tested by NIST.*

Exploring the Methodology and Utility of Standardized Latent Fingerprint Matcher Scoring

1. Introduction

Automated searches of fingerprints against a repository of known exemplars constitute an important tool of the forensic community. Systems performing such searches are referred to as ***Automated Fingerprint Identification Systems*** (AFISs). These are capable of searching large databases and returning a fairly small set of prospective candidate matches. It is then the job of the fingerprint expert to examine this list to see if any of the candidates can result in an ***identification,*** also called ***individualization***, i.e., can uniquely, and with high confidence, determine the individual that produced the print.

Central to the internal workings of an AFIS is the principle of a ***matching score*** (or ***matcher score,*** or ***comparison score***). This is a measure of similarity between the search print and any selected print from the database. The larger the score the greater the presumed similarity between the two. However, there is no standard for defining the interpretation of this score, nor even its numerical range. This can cause difficulties in interpreting candidate lists, as discussed in later sections. The purpose of this report is to explore matching score standardization for latent fingerprint systems.

1.1 Types of AFIS (Systems)

The nature and accuracy of the search process differ markedly depending upon the type of fingerprint being searched: ***ten–print*** or ***latent print***. The early history and development of AFIS is well covered in Moses [1].

Ten-print systems are the more advanced of the two types of systems, and are capable of producing high-confidence results with present technology. The result of a ten-print search will be a list containing candidates which are considered likely to be ***true mates***, i.e., fingerprints from the same subject. This list can have three variants: a) no credible mate was found, so the list is empty; b) exactly one candidate was found (deemed likely to be a true mate); or c) a small number, say two or three, candidates were found. Outcome (c) might be the result of multiple enrollments of the same subject in the database. These multiple enrollments may be accidental, and were not previously detected, or they may have been entered intentionally. At times one or more ***impostors***, i.e., ***not true mates***, will make it on to the candidate list, either alone or in addition to the true mate. The presence of one or more impostors can happen for a number of reasons, but poor image quality is often a contributing cause. With top-end systems impostors are infrequent. According to the Fingerprint Vendor Technology Evaluation (FpVTE) [18] report, AFIS can achieve a rank-one identification rate for ten-prints of more than 99.4 %, using a database of 10 000 fingerprint images. (More recent tests have verified this, and have shown even higher performance.) The important characteristic of ten-print systems to keep in mind is that **all candidates** on the candidate list are considered (at least by that AFIS) likely true mates.
For latent fingerprints the situation is markedly different. Latent fingerprints contain considerably less information than do ten-prints, and current technologies do not permit identifying the true mate with very high confidence. (Representative figures will be given later.) Therefore, it is standard practice for latent fingerprint matchers to output the results of the search in the form of a candidate list. This list can

be quite long, say twenty or even a hundred candidates. Currently most systems have a fixed, "canned," candidate list length. However, in special circumstances it is possible to override the "canned" value. The length of the list is then specified at the time the search is initiated. Candidates on the list are ranked by ***matching score***, with the highest score in top position. An important difference from ten-print is that the latent fingerprint system makes no judgment whether these are likely to be true mates or not; they are simply the highest scoring candidates, and their similarity to the search print may in fact be quite low. Further discussion on the challenges of latent AFIS are presented in Meagher [2].

Latent and ten-print systems constitute the two main types, but there are additional types. Notable types are: a) reverse latent searches, where a ten-print is searched against a database of unsolved latents; and b) latent to latent search systems, in which a latent is searched against unsolved latents. These systems are outside the scope of this report.

1.2 Candidate Lists and Candidate List Reduction

Increasing the length of the candidate list is one way of increasing the probability the true mate will appear on the list -- but this increase is surprisingly modest. Theory shows the probability a candidate is a true mate diminishes approximately inversely with its ranking on the list. (Thus the 10^{th} candidate is only half as likely to be a true mate as the fifth, etc. See [19, 20].) Therefore enhancing performance by lengthening the list has diminishing return beyond some length, while adding burden to the human expert. In addition, there is the distinct possibility a true mate is not in the database, and in fact, this is quite common. Obviously in such cases increasing the length of the list is counterproductive. This is discussed in more detail in Meagher [2].

As a result, the latent expert needs to exercise considerable judgment in selecting how many, and which, candidates to examine in detail (see [2]). In their selection the expert is assisted by the matching score. A very low scoring candidate might simply be dismissed out-of-hand. For "reasonable scores" the expert will call up the file print (image) of the candidate, and examine it. A cursory glance is often sufficient to exclude the candidate from further consideration. For example, it might be obvious the two prints have different pattern classes -- so cannot possibly be mates.

To be able to dismiss a candidate purely on score requires high familiarity with that AFIS system. Since the expert must often search a latent on several systems -- not all of which might be equally familiar -- this can cause difficulty. At a minimum, more work will be required to process multiple candidate lists; at worst, some true mates might be missed. Standardizing the matcher scores would greatly assist the process.

There are two other justifications for standardizing the score. Many candidate lists – especially the longer ones – may contain "weak" candidates, ones very unlikely to be true mates. It would be extremely helpful to the examiner if these were eliminated from the list. We refer to this process as ***candidate list reduction***, and is further discussed in Section 4.

Candidate list reduction was identified as a highly desirable goal in the ***Evaluation of Latent Fingerprint Technology (ELFT) Concept of Operations (CONOPS)***, [3]. The method for accomplishing this closely

follows the steps performed by the human expert: all very low scoring candidates are dismissed out-of-hand (except in very high value criminal cases). The remaining candidates are examined by a special kind of matcher which looks to see if the ***pattern classes***, or more generally the ***ridge flow***, are "reasonably" close. Candidates passing could be sent to an advanced matcher, capable of extracting and matching many types of features. Since this advanced matcher makes only a few comparisons, it does not need to be fast. Having a standardized score greatly assists in the first round elimination (of frivolous candidates). These ideas are further explored in Section 4.0.

1.3 Matcher Fusion

The third reason for standardizing matcher scores is to facilitate multi-matcher-fusion. Fusing (combining) the results of two or more matchers can greatly improve accuracy. Experiments have demonstrated gains of 15 % points or more in the identification rate in select cases (ref. [4]). One problem encountered in matcher fusion is that two matchers may produce ***native scores***[1] differing by orders of magnitude. Although this can be overcome by collecting statistics from a number of candidate lists (so as to calibrate the score), at the very least this requires gathering many lengthy candidate lists. This effort is justified if a dedicated fusion matcher is to be produced, which will frequently be used. However, if we consider the possibility we might have to fuse candidate lists from potentially many sources, such a procedure is impractical. Standardizing the matcher score obviates this necessity, and also allows for the fusion algorithm to be made more generic. Additional information on matcher fusion is found in ref. [4].

1.4 Why Score Standardization?

Summarizing the three reasons for score standardization:

a) To assist the human expert
b) To assist and simplify candidate list reduction
c) To assist and simplify multi-matcher fusion

Since much of the information that went into this study is based upon results of NIST's ***Evaluation of Latent Fingerprint Technology*** (ELFT) study, we next provide a brief overview of the ELFT project.

2. NIST's ELFT Project

The ELFT project was initiated by the Image Group of NIST/ITL to explore fundamental scientific and technological questions in automatic matching of latent fingerprints. The ELFT project was launched via a NIST-sponsored workshop, ref. [5]. This was followed by the publication of the CONOPS [ref. 3] detailing the goals and methodology of the project. One of the major goals was the testing and evaluation of automated latent matchers. The AFIS performance data used in this report largely comes from these tests.

[1] The native score is the score used internal to the matcher. It is not required to standardize this score. It might, or might not, appear on the candidate list in addition to any standardized score.

2.1 Outline of ELFT

The following diagram outlines the steps involved in the ELFT testing cycle.

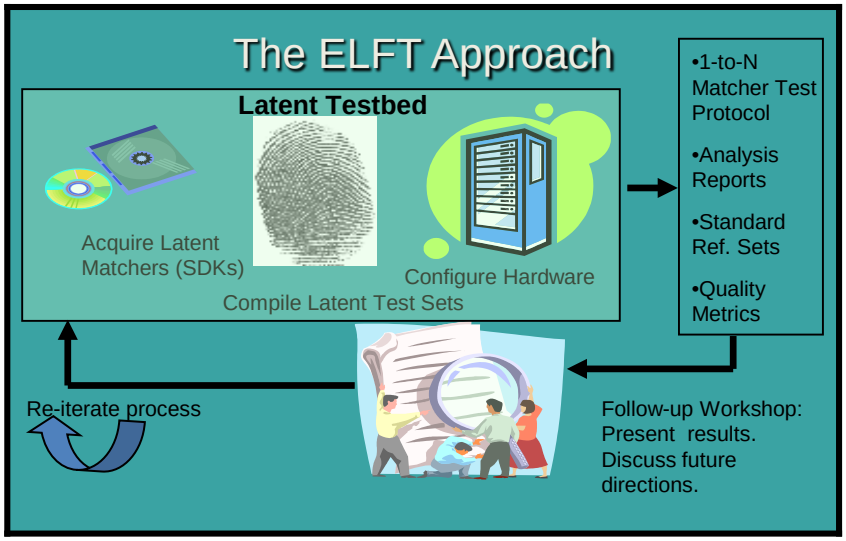

Figure 1 – Overview of ELFT Testing Approach

Matchers for testing were solicited from volunteer organizations. (These tended to be commercial AFIS developers.) Participants submitted their matchers to NIST in the form of ***Software Development Kits***, or SDKs. These SDKs are dynamic-link (.dll) library modules, linked at NIST to a main driver program. The resulting executable software was then hosted on NIST computers and run using NIST sequestered input datasets. (Here "sequestered" means the datasets are not available outside of NIST.) Additional information and best practices for SDKs are covered in Marshall, [6].

2.2 Matching Process and Output

Figure 2 provides a schematic of the actual matching process as well as the nature of the output candidate list. Candidates on this list are ranked by matching score, that is to say, by the internal or native score. (The sample scores shown in Figure 2 are representative of matchers using larger magnitude scores. Not all matchers use this range of values.) Although the diagram seems to imply a list length of twenty, the actual output of the SDKs in ELFT was 100 candidates. The candidate list also contained additional information, not shown in Figure 2, such as the finger number, and the number of minutiae extracted.

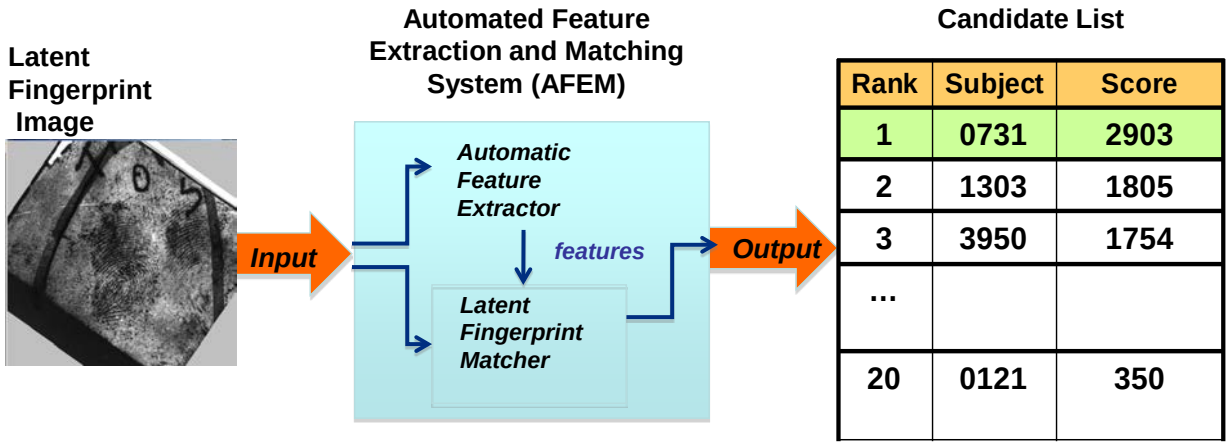

Figure 2 – Automated Candidate List Generation by an AFIS

Test data (fingerprint images and extracted features) for input to the matchers were compiled by NIST from several sources, including actual law enforcement case work, and controlled data collected from volunteers.

2.3 Objectives of ELFT

The principal objectives of ELFT testing, ref. [3], were:

a) Determine the overall state-of-the-art in automated latent fingerprint searching. By "automated" we mean that searches are performed without any human assistance. (This is strictly true for the ***image-only*** mode, where the matcher was given the latent image, and no other info.)
b) Performance of a matcher was mainly assessed by the percentage of hits (true mates) found in first place on the candidate list.
c) Some searches also included human-extracted features, either by themselves or in conjunction with the latent image. The purpose was to determine how much performance would increase using these additional features.
d) Measurements were also made on how much performance was affected by latent image quality.
e) Experiments were performed to study how much performance could be boosted by fusing two or more matchers. (See ref. [4].)

The output of each match was a candidate list, nominally 100 long. Following the ranking number, the candidate list contained the matcher score, the subject ID[2] in database, and the finger number.

[2] An alphanumeric identifier pointing to the fingerprint image, etc. No actual personal identification data is kept on file.

In addition, test participants (SDK submitters) were asked to supply supplemental data, including: the number of minutiae the matcher found on the latent; the number of minutiae the matcher was able to match; and perhaps most important for this paper, a ***probability-like matcher score*** (PLMS). The PLMS was intended as a prototype standard score. However, as no details were specified, each developer was free to develop their own PLMS. The only firm specification (in the API) was that the PLMS be an integer between 0 and 100. It was strongly recommended that the score reflect the likelihood that the candidate is a true mate, but the method of measuring this "likelihood" was unspecified.

Restricting the PLMS to integer values may have been a minor mistake, as this provided insufficient resolution, and resulted in excessive ties. A detailed analysis of PLMS submitted to NIST will be presented in Section 5.

2.4 **Representative Information Obtained from ELFT**

The series of ELFT tests conducted by NIST/ITL resulted in considerable insight on latent AFIS performance, and the state-of-the-art in general. The following graph is a representative example.

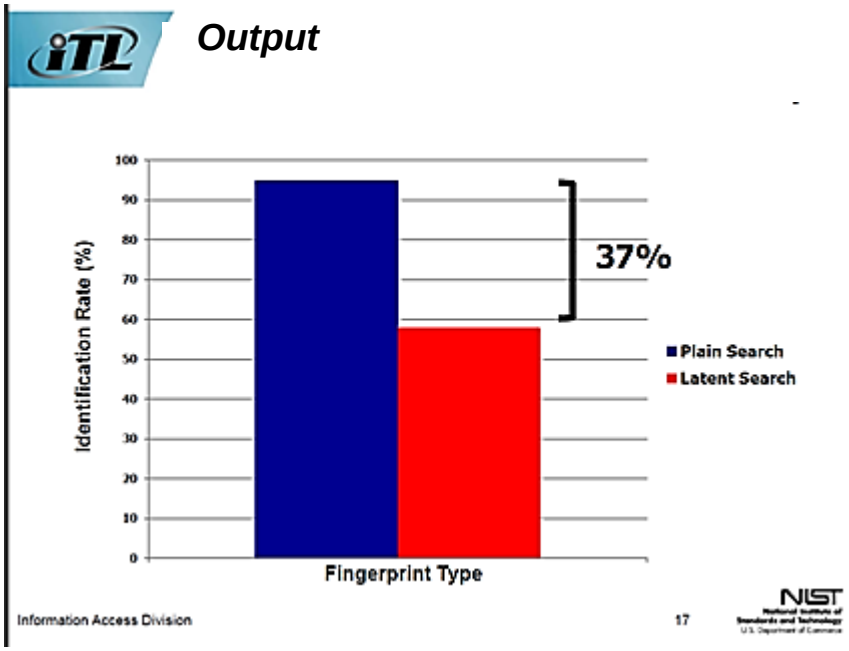

Figure 3 – Comparison of Achieved Latent Performance with that of Plain-impressions using current AFIS technology

The graph shows the difference in performance between: a) searches using a ***plain impression*** (a type of controlled capture – sometimes called a "dab" or "slap"); versus, b) an actual latent. The graph shows that while the plain impression has a 95 % chance of appearing at the top of the candidate list, the latent has only a 58 % chance. This is a very significant performance difference, and confirms the earlier statements that latents present a much greater challenge.

The following graph shows another kind of performance comparison.

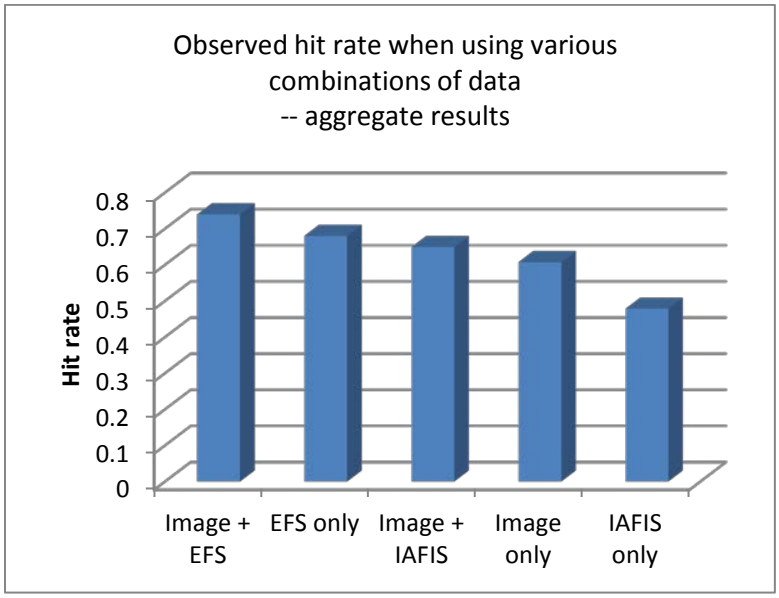

Figure 4 – Comparison of Achieved Latent AFEM Performance Using Different Data Modes; Aggregate of Four Matchers

Figure 4 shows that when operating in image-only mode (only the latent image is supplied, and no other information) about 58 % of the time the correct mate will be in first place. Using IAFIS[3]-type features only as input, and no image, produces a lower result of about 45 %. Adding IAFIS-type features (extracted manually) to the image increases performance by about 4 % from image-only. Highest performance achieved, using all available features, is about 71 %. For definitions of the ***extended features*** see ref. [7] and [8]. Supplemental information on ELFT is found in ref. [9, 10, & 11].

3. Matcher Scoring as Currently Practiced

In Section 2 we provided some background on NIST's ELFT project and the type of information we obtained. Another example of information garnered is the range of scores used by various matchers. These are summarized in the table below.

Matcher & Data Set Pair	Max Observed Score	Matcher & Data Set Pair	Max Observed Score
1	2 635 000	10	394.9

[3] IAFIS = Integrated Automated Fingerprint Identification System; formerly the FBI's principal fingerprint search and identification system; recently replaced by the newer Next Generation Identification System, or NGI

2	2 133 000	11	100
3	34 765	12	100
4	9 966	13	85.7
5	8 894	14	1
6	4 022	15	0.991
7	3 000	16	0.218
8	1 474	17	0.147
9	1 208	18	0.128

Table 1 – Some Representative Scores Output by Submitted Automated Matchers

Table 1 was compiled from candidate lists generated during NIST's testing of automated latent fingerprint matchers. (Not all came from the same test. Some of these data can be gleaned from references [12, 13, and 14].) Examination of the above data suggests there are basically three schools of thought regarding score magnitude:

1) Scores adhering to a "scientific" scale. These scores fall into the unit interval; alternatively, they can be expressed as a percentage.
2) Scores subscribing to the school of thought that larger numbers have more resolution. These scores tend to range from a few hundred, upwards to many thousands.
3) Scores which strike a compromise, and have values around a hundred. We prefer this option.

What we termed "scientific scale" is similar to what mathematicians adopted long ago for mathematical probability, namely [0, 1]. This is an obvious extension of the convention that 0 = FALSE and 1 = TRUE. Numbers between zero and unity represent various degrees of truth or likelihood. Correlation coefficients have a similar scale, but here the range is [-1, +1]. Negative scores are not normally used by matchers – they could be used for identifying incomplete data, very poor images, or similar types of problems.

The wide spectrum of prospective score values causes practical difficulties. When dealing with an unfamiliar matcher, human experts must familiarize themselves with the range of possible outputs for that specific matcher, so as to know what constitutes a "significant" value. Using normalized scores largely obviates these problems.

NIST Recommendation for Scoring Range

In this report we explore methods of standardizing the score. Details of how a score might be computed and its interpretation are deferred to Section 7. The proposed score is a variant of the second (mid-range) category, and has values between 0 and 100. To provide finer discriminating power, two decimal places are always carried, for example, 0.08, 37.08, or 97.43.

The score should provide a clear indication of the likelihood the candidate is a true mate. Thus, (using the previously quoted scores) 0.08 would indicate "very unlikely to be a true mate;" 37.08 would indicate "potential for a true mate, and should be examined;" while 97.43 would indicate "very likely to be a true mate."

4. Preliminaries to Algorithms for PLMS

Recall that the PLMS is a type of normalized matcher score with a strong connection to the likelihood a candidate is actually a true mate. This candidate list will originally be ranked by the native score produced by the matcher. For our studies we began with a candidate list of 100, and reduced this to smaller size, for example 20.

The algorithms considered fall into two basic classes: 1) algorithms which always preserve the original ordering; and 2) those potentially capable of increasing the score of certain candidates so as to move it ahead of others which previously outranked it. We consider both types here. The first is simpler, and can be implemented without significant additional information, computation, or software. The second type is the more desirable, as it potentially allows us to reclaim true mates (potential identification, or individualizations, or so-called "hits") which might be relatively far down the candidate list, and unlikely to be examined if not promoted upward on the list.

Consider for example, a case where in the original (native) score placed the true mate in 17^{th} place. It is unlikely that this candidate would survive candidate list reduction, and probably would not be looked at by the expert. But if additional information can be applied to the scoring (e.g., PLMS), it is possible that this candidate might be elevated to higher place. We have encountered many such cases during our investigation of matcher fusion [4]. Fusion accomplishes this by introducing new information. Analyzing the candidate list can also provide additional information, though generally of lower quality.

Figure 5 shows one of the simplest algorithms for implementing PLMS. It is of the first category, which preserves the original ordering of candidates. It uses a global point transformation to remap the native score into a PLMS. It is shown in Appendix B that such a transformation cannot change the order, and the only gain is that the score is now standardized, so more easily interpreted.

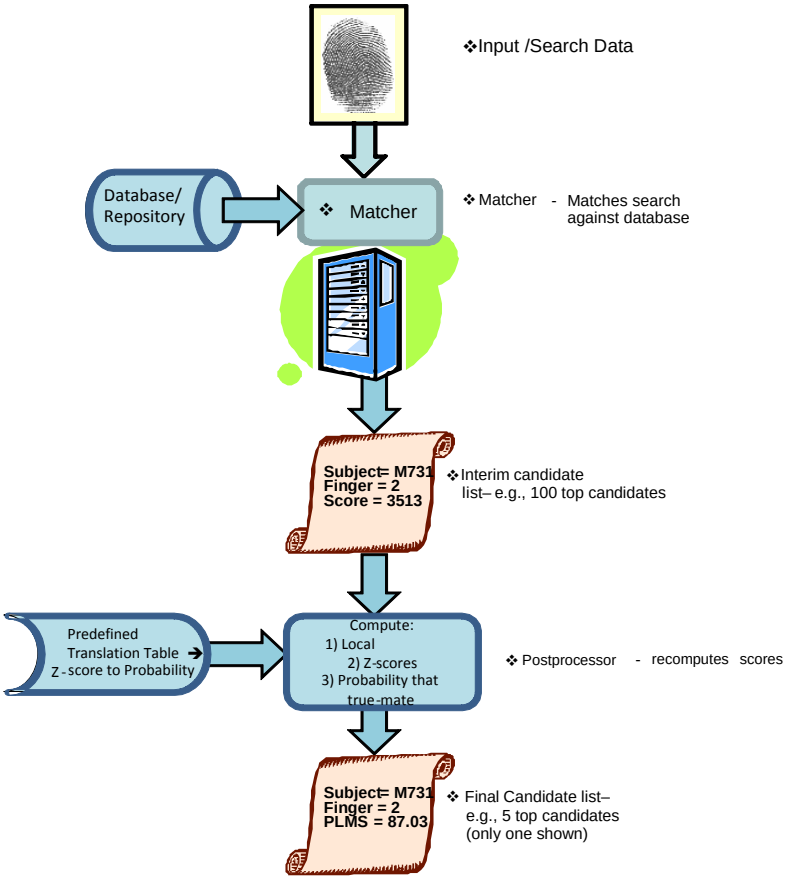

Figure 5 – Baseline Method for Producing PLMS; the Final Output Score appears at Bottom of Diagram (labeled "PLMS")

The following "walk through" might be helpful:

1) An intermediate score is computed based on the familiar Z-score:

Z_score = (native_score – mean_impostor_score)/standard_error_impostor … (1)

2) The mean impostor score is ideally a global estimate of impostor scores. It would therefore be a stored, predefined value. (Should this be not available, a local value, based on the 99 candidates beginning with candidate two, can be used; but this will introduce more uncertainty. Also, a larger database/background will generally result in a larger mean, because it will be based on the **largest** 99 values obtained. It might be necessary to adjust for this effect using extreme value theory.)
3) Finally, using either a look-up-table or a pre-computed analytic approximation (based on statistics collected from the development set) we compute the PLMS based on the algorithm selected. See for example Table 2. (In the above diagram this appears as 87.03.)
4) Note that as the transformation to the PLMS is based on a global point transformation, and so cannot improve performance (see Appendix B), other than making the score easier to interpret.

To potentially change the order of candidates ("bubble up" the true mate) we need to inject additional information. This can be done by processing the candidates on the list through a new matcher, called Matcher B, as shown in Figure 6. (This may be considered a type of fusion.)

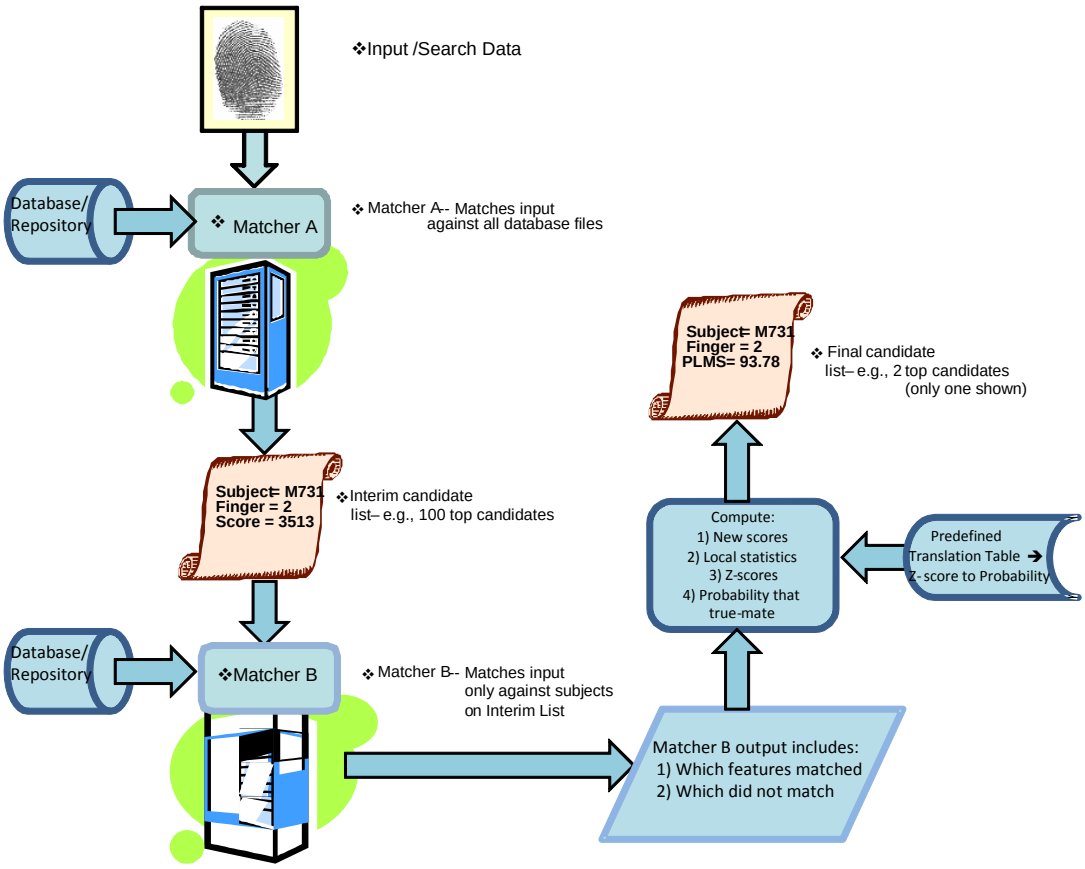

Figure 6 – A More Advanced Algorithm for Producing PLMS; this Algorithm is Capable of Re-ordering Candidates; the Specimen Output Score appears at Top-right of Diagram, and is labeled PLMS

Surprisingly, it is possible to reduce "matcher B" to a minimalist matcher employing only information already on the original candidate list. There is then no need to go back to the repository (as implied in the above figure). The principle behind this is that the distribution of scores on the candidate list carries additional information, which was not used in the original scoring (native score). In Section 6 we give an example of such score improvement.

In the more sophisticated version shown in Figure 6, Matcher B is an actual matcher and is a variant of the primary matcher, Matcher A. Matcher B looks only at the candidates output by Matcher A. The objective is to use the information available to Matcher A more fully and without incurring the cost of computing additional features.

The third and final algorithm, this time using additional features, is shown in Figure 7.

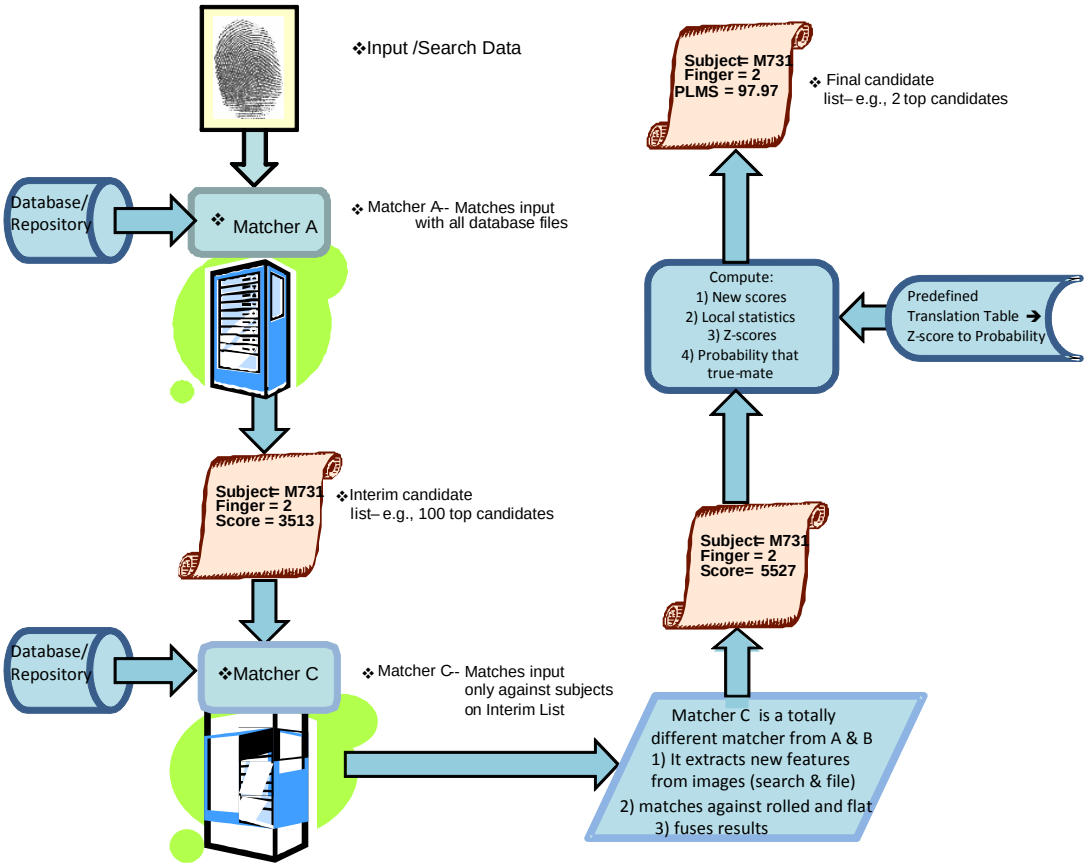

Figure 7 – Highly Advanced Algorithm for Producing PLMS; this Algorithm is Capable of Re-ordering Candidates; the Specimen Output Score appears at Top Right of Diagram (and is labeled PLMS)

The principal difference from the prior diagram (Figure 6) is the inclusion of a more advanced matcher, labeled Matcher C. This matcher has the option of going back to the images and extracting additional features. There is also the option of matching against both the rolled and the plain impression, then fusing the two results. This has been shown to be quite powerful [refs. 4, 12, 14].

5. Analysis of Scores Provided on Candidate Lists

In Section 2 we provided a synopsis of the ELFT project. As stated, a major objective of the ELFT project was testing of matchers provided by participants (principally commercial AFIS vendors). These matchers were run on NIST compiled and sequestered datasets, and produced output candidate lists. The candidate lists contained two types of scores: 1) native score, and 2) a PLMS-like score, referred to as "probability of true mate," or simply "prob" The exact algorithm for computing this "prob" was not specified, except that (i) it should be an integer between 0 and 100; (ii) it should have significant correlation with whether the candidate is a true mate or not; finally (iii) it was encouraged (but not required) that the calculation for "prob" should take supplemental information into account, beyond the native score. The reason for (iii) was to encourage the development of a score more powerful than the native score.

Analysis of candidate lists (produced by submitted matchers) led to significant insights, many of which found their way into this report. First, analysis showed that developers had adhered to instructions, but not all matchers produced enhanced scoring. Some matchers produced values more or less spanning the range, while others tended to cluster: a low cluster with values around 10, and a high one around 90. It also became apparent that limiting the output to 101 values resulted in too many ties. These tied scores caused, in some cases, a slight loss of performance (due to the tied scores being presented (sorted) in the arbitrary order).

In this section we examine to what extent stipulation (iii) (see above) was in fact achieved. If "prob" is computed by means of a global point transformation of the native score, and no new information is added, then it will be no more powerful than the native score, though perhaps easier to interpret. The figure-of-merit (FOM) we used to gauge the efficacy of "prob" was the ***minimum-total-error***, or MTE. This is essentially the sum of type I and type II errors, taken at the threshold which minimizes this sum. (Types of errors and methods for gauging these are discussed in the Appendixes, and in ref. [15, 16, and 17].) If it turns out "prob" is no more powerful than the native score, its MTE should be the same or larger than for the native score. The following figure shows the results for the five matchers, called A-D. For each matcher three datasets were used: LA, LE, and LG. The first (LA) consists of only the image of the latent; the second (LE) consists of the image plus selected and pre-extracted feature; the third (LG) of IAFIS-style features only. Figure 8 provides a comparison of the MTE achieved by the four different matchers. Each matcher uses three types of input data, for a total of twelve cases.

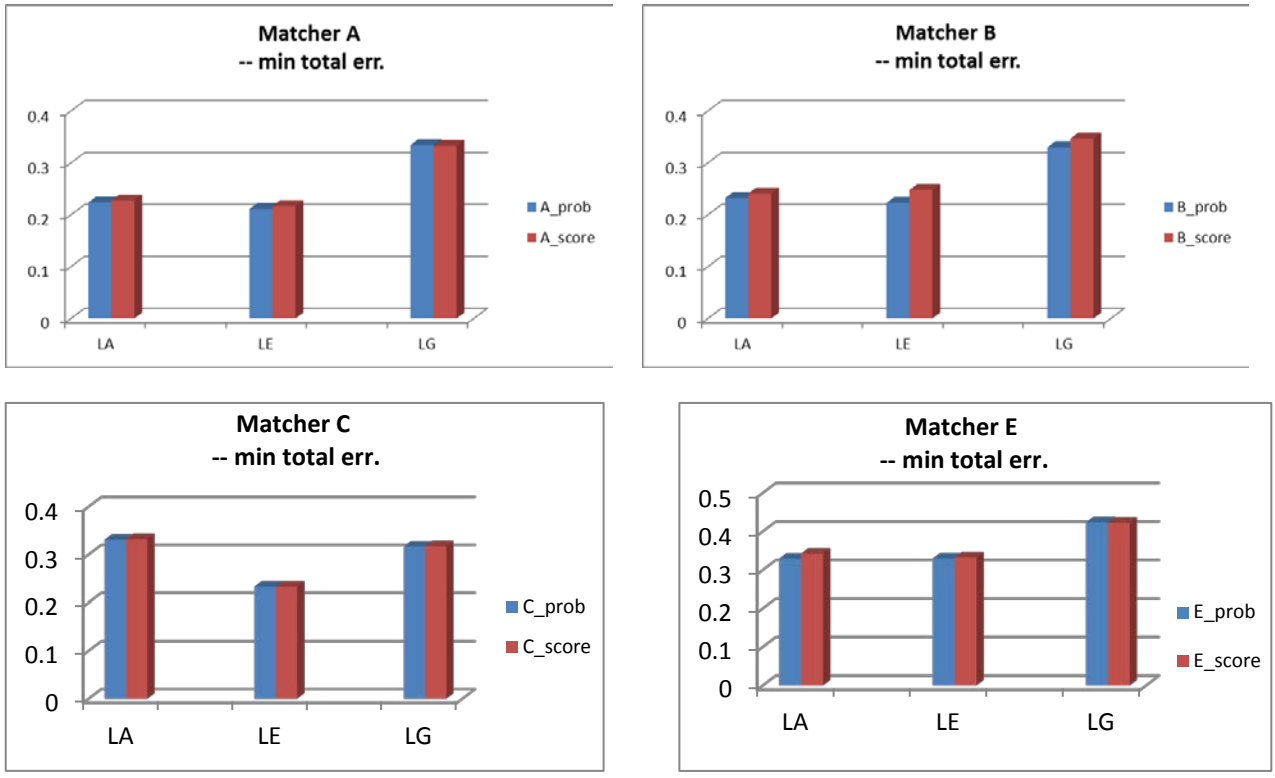

Figure 8 – Comparison of Achieved MTEs for Four Matchers

The bar on the right (red) of each pair of bars represents the MTE based on the native score (internal score used by matcher); the bar on the left (blue) is obtained when using "prob." In interpreting Figure 8 keep in mind that "smaller is better," so shorter bars represent better performance. In some cases the two bars appear to be nearly the same height; in other cases there is a marked difference. A more sensitive gauge than casual visual inspection is required to make these differences clear. To achieve this, we first compute the standard error of the MTE calculation. This is defined as the root-mean-square (RMS) variation of the MTE when arbitrary (but reasonable) changes are made in the numerical calculations. These variations (or errors) are primarily due to bin boundaries used in computing the histogram, and secondarily to the quadratic interpolation for estimating the minimum. (Additional information appears in Appendix F.)

From a number of trials we determined the standard error (SE) is approximately 0.005. We then took the difference between the MTE of "prob" and the MTE of native-score and divided this by the above SE, specifically:

$$\text{Normalized\_delta} = (\text{MTE\_prob} - \text{MTE\_score})/\text{SE} \qquad \ldots (2)$$

The results are shown in Figure 9.

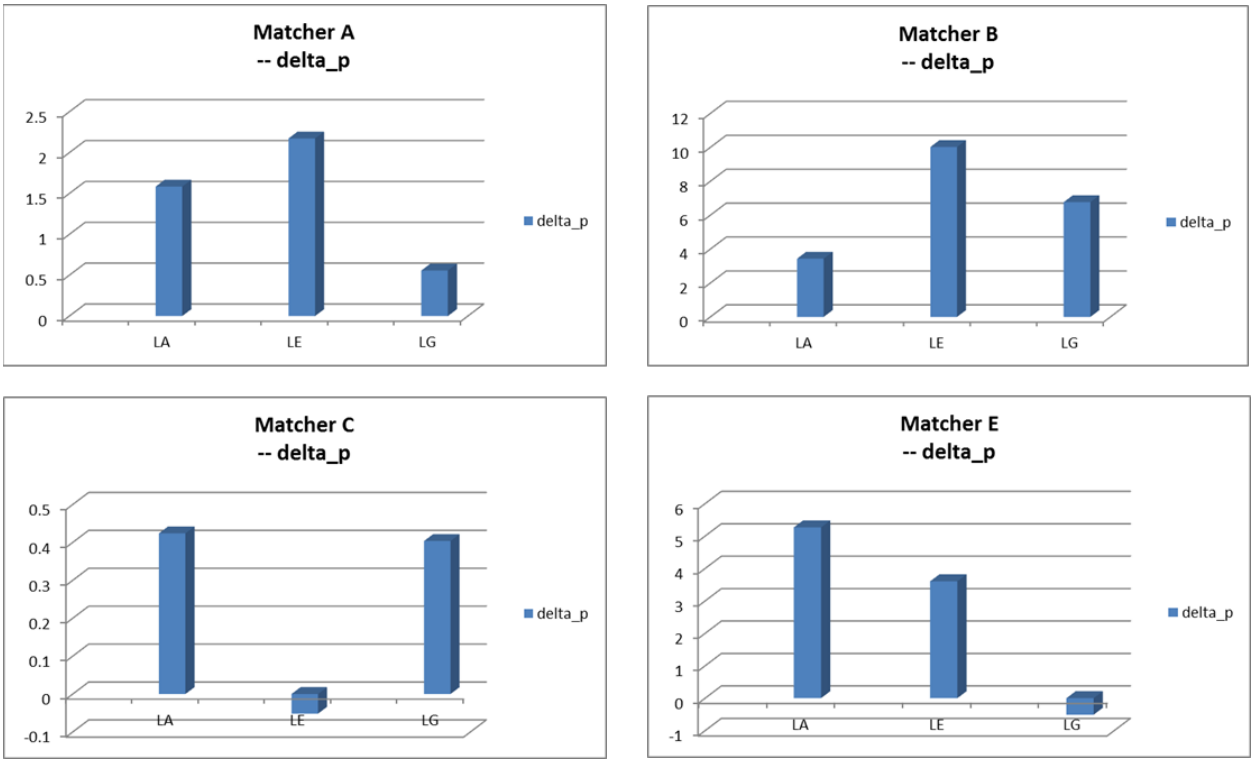

Figure 9 – Normalized Deltas

Note that the scales on the graphs are different. In assessing the results we take a conservative approach: an absolute delta less than unity is considered insignificant. (Why? It is common practice to consider a one-sigma variation as insignificant, partly because experience shows that most standard

error estimates are too small.) A value between one and two units is considered marginal; while a change exceeding 2 units is considered significant. It then appears that for Matcher C, and possibly also for Matcher A, the differences are not significant. For Matcher B they are definitely significant. (2 sigma errors are associated with 95 % confidence interval.) We conclude that Matcher B (and possibly E) used additional information beyond the native score.

6. Improving the Native Score

The previous section showed it is possible to improve upon native score by judiciously adding information. In this section we show that this can be done with minimal additional resources. In fact, by using only the native scores on the candidate list and repackaging it, it is already possible to produce a degree of improvement.

A number of variant algorithms were tried. Some were unsuccessful. One method producing positive results is related to a principle used by latent experts: a large increase in score from the adjacent candidate immediately beneath might indicate a true mate. We used this observation in the following way:

1) Sort the candidates using the native score and retain the top 100.
2) Modify each score on the list by subtracting from it a (fixed) fraction of the score immediately below it.
3) Reorder the resulting scores.

(This algorithm fails for the 100^{th} candidate, because there is no candidate below it – we simply ignore the last candidate.) Explicitly, if S_i denotes the original "native score," and S^*_i the "improved score," then

$$S^*_i = S_i - k\, S_{i+1} \qquad \ldots (3)$$

where k is a constant. This multiplicative constant (for the subtraction term) was determined experimentally so as to minimize the MTE. Based upon a small number of trials, a value of 0.3 was selected for k. (No attempt was made to optimize the algorithm of eq. (3) as the purpose was only to demonstrate a concept.) The following figure shows the resulting improvement in MTE when using S* vs. S. For comparison, the corresponding deltas based upon p/"prob"[4] are also shown.

[4] See first para. of Section 5; p and "prob" will be used interchangeably.

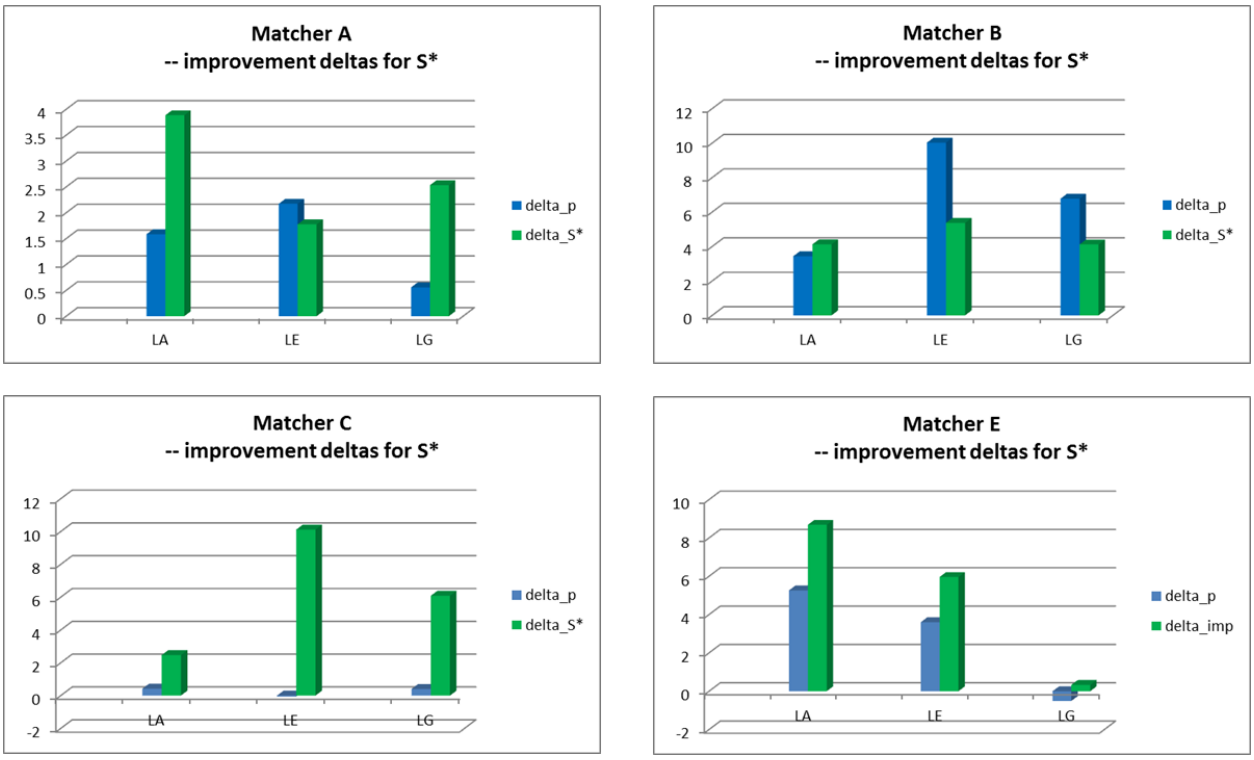

Figure 10 – Normalized Deltas Using S Compared to those Using "p/prob"*

In the above, the green bars, labeled "delta_S*," represent the improvement in MTE due to using S* in lieu of p (= prob). Also shown for reference, are the blue bars resulting from "prob." (These are the same as those of Figure 9.) It will be seen that in all cases the green bars demonstrate improved performance of S*, and except for Matcher B, the improvement is greater than that of prob. (We found it somewhat surprising that such a simple algorithm could produce such results.)

The overall results are summarized in the following table. The numbers shown are the averages over the three input datasets.

Matcher	Delta-p	Delta-S*
A	1.5 (marginal gain)	2.7
B	6.7	4.5
C	0.3 (no gain)	6.2
E	2.8	5.0

Table 2 – Summary of Performance Gains

(The improvement in performance in terms of reduction of MTE is obtained by multiplying these values by .005.)

7. Detailed Definition of the Standardized Score/PLMS

In Section 3.0 we established some desirable characteristics for a standardized score. One of these was that it should correlate with the likelihood of the candidate being a true mate. We referred to a score having this characteristic as a PLMS. We first define the standardized score/PLMS in terms of an intuitive interpretation, and follow this with an algorithm for calculating this score. First the simple, intuitive explanation:

The standardized score is a positive number between 0. and 100, and carried to two decimal places (e.g., 93.76).

7.1 Survey of Potential Algorithms

The following table outlines nine algorithms for computing the PLMS. These range from simplest (alg. 1), to the most complex (alg. 7).

Alg. No.	Algorithms synopsis	Positive attributes/advantages	Negative attributes/disadvantages
1	Scale the "native" score (internal matcher score) of all candidates, both genuine and impostor, linearly. Lowest score becomes zero, highest 100. (Zero to 10 000 has also been suggested, with no decimals carried.)	Simplest possible implementation. Might be adequate. Can be used in conjunction with other algorithms/scores. This transformation retains the matcher's natural distribution.	Too "lumpy" – scores not uniformly distributed. Not clear where transition from impostors to true mates occurs.
2	Use only non-mate scores (impostors). Scale these based on the inverse of the cumulative distribution function (CDF). Again, result will be between zero and 100, or zero and 10 000 depending upon preference.	Simple to implement. Also, there is always an abundance of non-mate scores.	True mates will be crowded into a small range of top scores. Many true mates will simply score 100, and there will be no discrimination between "ordinary" hits, and "monster" hits. [Monster hits are typically several magnitudes larger than mean impostor scores.] Most of the range of scores will be of little use.
3	Same as (2) but use a logarithmic scale to stretch the top-most portion of score distribution. Rescale this to fit into the interval [0, 100].	Simple to implement. Also, there is always an abundance of non-mate scores. Gives better separation at top than does (2). Worth further investigation.	Somewhat more complicated in the details. Unclear where the breakpoint is. Some searches produce higher scoring non-mates than others. These might produce the bulk of impostors. [By "breakpoint" we mean values high enough that a candidate deserves further scrutiny.]
4	Similar to (2), but use only true mate scores.	Simple, and puts the emphasis on true mates – which is what we want to identify.	Since it does not take non-mates into account, it gives no indication of breakpoint.
5	Hybrid system. Similar to (4), but counts only true mates which appear in first place. This method is developed in detail in the report. May be generalized by counting all hits up to rank L. (Prior version has L=1.)	Takes impostor distribution into account. Provides good indication of what is a strong hit.	May be limited by availability of large background and sufficient searches with mates. Tends to de-emphasize lower non-mates. Breakpoint still somewhat dependent on size of background. (But only minimally so.)
6	Computes likelihood-ratio (LR) – ratio of	Theoretically one of the best	Difficult to compute: may require

	probabilities: a) that candidate comes from true mate population, to b) candidate comes from impostor population.	indicators -- if accurate information exists.	extreme value theory as foundation. Maybe misleading if information upon which it is based is unstable. Also, the dynamic range of the output can be very large, say 1E-6 to 1E6.
7	Same as 8, but take log() of LR, then remap to 0 to 100 (or 10,000).	Better than 8 because numbers are more user friendly. May be theoretically the best if information it is based on is sufficiently stable.	Difficult to compute. [We do not say it should be abandoned, but the extra effort might be unproductive.]

Table 3 – Comparison of Potential PLMS Algorithms

At this point in our investigation it is not possible to definitively select the optimum algorithm. Our impression is that algorithm 1 is too simple, and does not take sufficiently into account the differences between systems. Algorithms 6 and 7, on the other hand, are somewhat on the complicated side, and require detailed information, which might not be available, or even unstable.

In view of the above, we have selected an algorithm in the middle, algorithm 5, for further development.

7.2 Detailed Development of Algorithm 5

The standardized score, or PLMS, is a positive number between 0 and 100, and is carried to two decimal places. For algorithm 5 the exact interpretation of this score is that it represents the percent of true mates in first (top) position that have equal or lower native scores than the candidate being considered. For example, a score of 93.76 would indicate that 93.76 of true mates appearing in first position will have less or equal score -- and this will of course be true whether using the standardized score or the native score. The same transformation (score remapping) is used regardless of position on ***the*** candidate list. For example, a PLMS of 12.77 for a candidate in third place (third position on list) means that only 12.77 % of true mates in first place have equal or lower scores. Note that this definition depends to an extent on the size of the background. We propose to use one million (fingers) as the standard size.

There are several approaches to arriving at the PLMS. Below we outline a stepwise approach, which is intuitively easy to understand. Later on in this section we outline a more terse mathematical approach. We begin with the optimized native score, S*. (See Section 6.) From S* we compute an intermediate score, of the Z-score type:

$$Z = (S^* - \mu_{imp})/\sigma_{imp} \qquad \ldots (4)$$

Here μ_{imp} is the mean of all imposter (non-mate) scores (S*), and σ_{imp} is the standard deviation of these scores. These values are computed from the development set. As it stands, eq. (4) already encompasses some normalization properties: a) the mean value is close to zero; b) the standard deviation is close to unity; and exceptionally high scores (say produced by true mates) have values on the order of ten or twenty.

The next step is to transform eq. (4) into the final desired normalized score (PLMS). This can be done in many ways. We selected a simple approach, via the exponential function:

$$Z^* = 100.*(1 – \exp((Z + 2)/c)) \qquad \ldots (5)$$

The constant c is experimentally chosen. We found c = 18 to work reasonably well. The purpose of the +2 in eq.(4) is to eliminate (or at least minimize negative) values. (For example, a value of Z = -2 will produce Z* = 0, while a value of Z = -3 will result in Z* = -5.71.) If negative values are to be avoided entirely, this can be done via a clamp function, such as:

$$Z^{**} = \max(Z^*, 0) \qquad \ldots (6)$$

To test the result, we collect a histogram (of Z* or Z**) for all true mates in first position using the development set. This requires choosing bin boundaries for the histogram. To illustrate, we might have a bin with boundaries at 0.90 and 0.95. All scores ≥ 0.90 but < 0.95 will fall into this bin, and be so counted. Normalizing this count by the total number of hits in first place (regardless of score) will give a number which represents a probability. This value ideally should be near 0.925 (the center of the bin). Suppose that after conducting the above procedure we obtain 0.96. This might be deemed accurate enough, but if a more precise value is desired we can correct eq. (4) using a low-order polynomial (say a cubic). Calling this polynomial p(Z*), we then arrive at an equation of the form:

$$Z^{**} = Z^* + p(Z^*) \qquad \ldots (7)$$

The corrections produced by p(x) should be fairly small, on the order 10 % or less. (In the previous example p(92.5) ≈ - 3.5.)

The following figure shows a plot of the standardized score derived from the above algorithm, using the exponential plus a quadratic correction term. (This uses Matcher B and dataset LE – but the choice does not appear critical.)

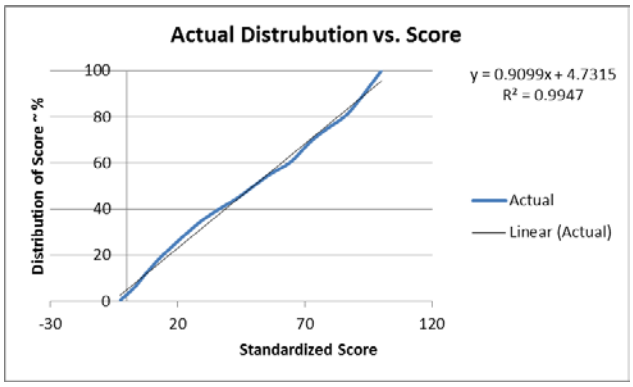

Figure 11 – Actual Distribution of Scores Based on PLMS

Ideally, the blue line should follow the principal diagonal. The actual curve shown has noticeable "squiggles." We could attempt to refine this further, using eq. (7), but a limit is soon reached due to uncertainties in the histogram data. Probably Figure 11 is close to what can easily be achieved.

The following table provides some interesting statistics based on the standardized score derived above.

Statistic	Observed Value
Max score obtained (true mate)	95.3
Min true mate score (in first place)	3.7
Max impostor score (in first place)	80.85
Min impostor score (in first place)	2.1
Mean true mate score (in first place)	49.9
Standard dev. of true mate score	31.6
Mean impostor score	5.3
Standard dev. of impostor score	5.2
Mean hit position (mean rank on list, assuming on list of 100 candidates)	3.7

Table 4 – Selected Statistics Based on PLMS

In the above we deliberately took a roundabout procedure to make it intuitively clear. Mathematically, a much more compact method is to: a) empirically compute the cumulative distribution function (based on the development set; b) find a good analytic approximation to this histogram (this depends upon specifics of the native score, and might require the error function, erf(x), the Rayleigh function, etc.); c) finally, we take the resulting analytic cumulative distribution and scale it by 100 to arrive at the PLMS. Details are found in Appendices H and J.

We have not addressed the question: what specific threshold should alert us of a possible hit? Insight is provided by the point at which the minimum total error occurs. This is generally near 10, suggesting that a value somewhat larger than 10 would be a reasonable threshold for declaring a "hit." This topic is examined in detail in Appendix F.

Summary: This report shows the need for standardizing the matcher score. Three specific reasons are given: i) To assist the human expert in examining candidate lists; ii) for candidate list reduction (eliminating obviously weak candidates); and iii) for multi-matcher fusion. We propose that the score should range from zero to 100, with two additional decimal places; or else from zero to 10 000, with no decimals carried. Seven potential algorithms were listed, ranging from very simple to quite complex. A mid-range algorithm was then selected for detailed development. This report expresses the opinion that additional evaluation and input from human experts is required for a definitive selection. Seven appendices are provided so support the mathematical underpinnings.

8. References

[1] Moses, K., et al, "Automated Fingerprint Identification System (AFIS)," Chapter 6 & 7 of *The **Fingerprint Sourcebook***; McRoberts, A., McRoberts, D., Eds.; National Institute of Justice: Washington, D.C.; 2011.

➔ https://www.ncjrs.gov/pdffiles1/nij/225326.pdf

[2] S. Meagher, V. Dvornychenko, M. Garris, "Characterization of Latent Print "Lights-Out" Modes for Automated Fingerprint Identification Systems (AFIS)," 2013, in publication

[3] V.N. Dvornychenko, P. Grother, M. Indovina, "Concept of Operations (CONOPS) for Evaluation of Latent Fingerprint Technologies", NIST Publication 2007 (NIST website)

➔ http://biometrics.nist.gov/cs_links/latent/elft07/elft07_concept.pdf

[4] Dvornychenko, V. N., "Evaluation of Fusion Methods for Latent Fingerprint Matchers," 5th IAPR International Conference on Biometrics, New Delhi, India, April 2012

➔ http://www.nist.gov/manuscript-publication-search.cfm?pub_id=910369
➔ http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6199806

[5] V. Dvornychenko, M. Garris, "Summary of NIST Latent Testing Workshop," 2006,
➔ http://www.nist.gov/customcf/get_pdf.cfm?pub_id=50876

[6] K. Marshall, et al., "Incorporating Biometric Software Development Kits into the Development Process," NISTIR 7929, April 2013.
➔ http://dx.doi.org/10.6028/NIST.IR.7929

[7] Brad Wing, Editor, "Data Format for the Interchange of Fingerprint, Facial, and Other Biometric Information," NIST SP 500-290, September, 2011.

➔http://niatec.info/ViewPage.aspx?id=242
➔http://www.nist.gov/customcf/get_pdf.cfm?pub_id=51174

[8] A. Hicklin, "Instructions for Extended Friction Ridge Features", version 1.0, March 2012
➔ http://www.noblis.org/interop

[9] Dvornychenko, V. N. and Indovina, M., "Roadmap of ELFT – Past, Present, and Future," Presentation dated March 2009
➔http://biometrics.nist.gov/cs_links/latent/workshop09/proc/latent_nist_dvornychenko.pdf

[10] Indovina, M., "NIST Forensic Science Activities: Latent Fingerprint," NIST/ITL/IAD presentation material, March 2012
➔ http://www.nist.gov/director/vcat/upload/12-INDOVINA-Fingerprints-10_14-FINAL.pdf

[11] Evaluation of Latent Fingerprint Technologies (ELFT) (Website Homepage)

➔ http://www.itl.nist.gov/iad/894.03/latent/

[12] M. Indovina, V. Dvornychenko, E. Tabassi, G. Quinn, P. Grother, S. Meagher, M. Garris, "ELFT Phase II - An Evaluation of Automated Latent Fingerprint Identification Technologies", NISTIR 7577, April 2, 2009
➔ http://www.nist.gov/customcf/get_pdf.cfm?pub_id=901870

[13] M. Indovina, A. Hicklin, G. I. Kiebuzinski, "ELFT-EFS Evaluation of Latent Fingerprint Technologies: Extended Feature Sets [Evaluation #1]," NISTIR 7775, March 2011
➔ http://biometrics.nist.gov/cs_links/latent/elft-efs/NISTIR_7775.pdf

[14] M. Indovina, et al., "ELFT-EFS Evaluation of Latent Fingerprint Technologies: Extended Feature Sets [Evaluation #2]," NISTIR 7859, May 2012
➔ http://biometrics.nist.gov/cs_links/latent/elft-efs/NISTIR_7859.pdf

[15] M. Garris, C. Wilson, "NIST Biometrics Evaluations and Developments," NISTIR 7204, Feb. 2005

> A Receiver Operator Characteristic (ROC) analysis measures the trade-off of TAR and FAR. A threshold is swept across the range of scores such as those in Fig. 3. At each step, the percentage of match scores above the threshold is recorded as TAR, and the percentage of non-match scores above the threshold is recorded as FAR. Plotting these (TAR, FAR) points creates an ROC curve like the ones shown in Fig. 4. This serves as a primary measurement of verification performance.

➔ ftp://sequoyah.nist.gov/pub/nist_internal_reports/ir_7204.pdf

[16] A. Martin, et al., "THE DET CURVE IN ASSESSMENT OF DETECTION TASK PERFORMANCE," NIST/ITL web publication

⇨ http://www.itl.nist.gov/iad/mig/publications/storage_paper/det.pdf

[17] V. Sravya, et al., **"**A Survey on Fingerprint Biometric System," **International Journal of Advanced Research in Computer Science and Software Engineering,** April 2012
➔ www.ijarcsse.com
⇨ http://www.ijarcsse.com/docs/papers/April2012/Volume_2_issue_4/V2I40016.pdf

[18] C. Wilson *et al., "*Fingerprint vendor technology evaluation 2003: Summary of results and analysis report," NISTIR7123, 2004

➔ http://fpvte.nist.gov/report/ir_7123_analysis.pdf

[19] V. N. Dvornychenko, "Latent Fingerprint System Performance Modeling," Proc. of SPIE-IS&T, 2008

➔ http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=811697

[20] Lael Rayfield, "Simplified Analysis of Fingerprint Matchers, NIST Internal Presentation," 2012

Appendix A – Metrics for Evaluating Matcher Performance

This appendix covers the most common and useful metrics for evaluating matcher performance. While these are generally well known, we include them for completeness, and as an introduction to the other appendices.

Two metrics/statistics are of prime importance in gauging the performance of an AFIS: a) the ***true acceptance rate***, TAR; and b) the ***false acceptance rate***, FAR. The first, TAR, is a measure of the probability that the true mate will appear on the candidate list (output list) given that the true mate is in the database that is searched. The FAR is a measure of the probability that a false mate, or impostor, will appear on the candidate list. A seminal report by Garris of NIST, ref. [15], explains this concept in detail.

There are several variants of TAR and FAR, depending on how long of a candidate list is under consideration. When counting impostors (for FAR) we might only include impostors outranking the true mate (if present). Alternatively, we might only consider impostors appearing in first position.

For TAR we often count only candidates in first position, as is typically the case in this paper. This simplifies the statistics but may prevent the detection of multiple mates in the database (multiple enrollments).

An important performance measure employing TAR is the ***cumulative match characteristics***, or CMC. This is defined as the probability that the true mate will appear on a candidate list of specified length. Later on we give some examples.

Closely related to the CMC, but more powerful, is what is termed the ***receiver operating characteristics***, or ROC. The name betrays the radar origins of the concept, and is something of an anachronism. A more up-to-date name is ***relative operating characteristics***, or even simply ***operating characteristics***, or OC. The ROC concept is described in detail in [15]. The mathematical expression for the ROC is developed in Appendix B. A closely related concept, the DET, is detailed in [16]. Ref. [17] provides a more recent overview of these concepts.

In essence, the OC shows the dependence of TAR upon FAR. The OC was used in this report for computing performance, principally in the form of MTE (viz.).

Appendix B – Invariance of the ROC to Global Point Transformation.

We derive the fundamental equation for the Operating Characteristics curve, or OC,[5] based on the distribution of true mate scores (genuine scores) and non-mate scores (impostors). We have several reasons for doing this, but ultimately we wish to show that the result is independent of simple score transformations.

Let S denote the (native) matching score produced by a system; let N denote the size of the database/background/repository; next, let $f_1(S)$ denote the probability distribution of scores (density function) when matching against a non-mate, and let $f_2(S)$ denote the corresponding function when matching against a true mate. Finally, let $F_1(S)$ and $F_2(S)$ denote the respective cumulative distribution functions, i.e., the integrals of f_1 and f_2.

Suppose we assign a score threshold, τ, with the idea that only scores exceeding or equaling this threshold will be counted. The fraction of the database expected to exceed this threshold (when matched against the search) is $1 - F_1(\tau)$. Or put another way

$$N(1 - F_1(\tau)) = FAR \qquad \ldots (B\text{-}1)$$

(If there is exactly one true mate in the repository, then N should really be replaced by N-1 – but this is obviously unimportant for large N.)

From eq. (B-1) we can solve for τ, obtaining:

$$\tau = F_1^{-1}(1 - FAR/N) \qquad \ldots (B\text{-}2)$$

The quantity FAR/N is just the normalized FAR, that is, the fraction of the repository passing the threshold.

The probability that a true mate will exceed the threshold τ is given by

$$p = 1 - F_2(\tau) \qquad \ldots (B\text{-}3)$$

Combining (B-2) and (B-3) we arrive at

$$TAR = 1 - F_2(F_1^{-1}(1 - FAR/N)) \qquad \ldots (B\text{-}4)$$

Suppose now we wish to use a different score, S*. We assume S* is derived from S by means of a global point transformation, $S^* = G(S)$. (By a global point transformation, we mean a function of a single variable. Also the function may not contain "hidden" variables which are allowed to change from case to case.)

[5] Also called ROC

Assuming that the function G() is invertible, the distribution functions will now be given by $F_1(G^{-1}(S^*))$ and $F_2(G^{-1}(S^*))$. When these expressions are substituted into (B-4) we obtain the following expression:

$$TAR = 1 - F_2G^{-1}GF_1^{-1}(1 - FAR/N) \quad \ldots (B\text{-}5)$$

Provided that the function G has a regular inverse, G will drop out of (B-5), and we are left with the same result as before, namely (B-4). If however the function G exhibits jumps or flat spots, then this argument does not apply, because now there is information loss, and the new OC will be lower.

The important conclusion to take away is:

A point transformation of the score cannot improve performance. If the transformation is regular, it will leave performance the same; otherwise performance will diminish. Note that the transformation of eq. (3), Section 6, is not a point transformation since it involves two adjacent values.

Appendix C – Examples of OC Functions

It is often claimed that actual distribution functions for the scores do not closely follow any classical distributions (e.g., normal, log-normal, etc.), and therefore OC functions derived from such distributions are of no use. There is a measure of truth to this, but the conclusions are overstated.

Sometimes the distributions can be represented with good accuracy by classical functions for most of the score range. The following graph is such an example.

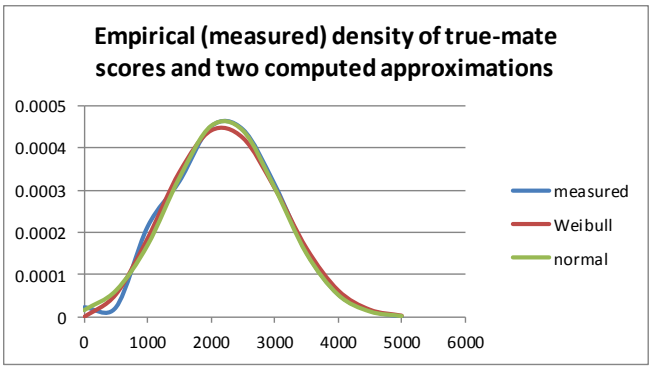

Figure C-1 – Experimental True Mate Score Distribution and Two Approximations Using Classical Functions

The fit is generally quite good, except at the very low end, say 0 to 700. Some of the deviation is almost certainly due to "sampling errors," i.e., peculiarities of the specific test data, but it remains true that such approximation often fail at the tails of the distributions.

It is possible to improve the fit of the tails by using a min-max-error criterion. This will improve the error in the tails at the expense of slightly increasing errors elsewhere.

A second observation is that many distribution functions lead to the identical OC function. (See for example eq. (B-5).) In principle, it is therefore possible for non-standard empirical distributions to lead to an OC function derived from classical distribution functions.

The third observation is: even allowing that we have a classical OC function which is not sufficiently accurate in certain sections of its domain, we can consider this function to be a first approximation, then add corrections terms. These "not quite accurate enough" OC functions often can lead to essentially correct conclusions, and so are useful.

One of the easiest classes of distribution functions to work with are the Weibull distributions. Here the general equation is

$$F_i(s) = 1 - \exp(-(s/\lambda_i)^{n_i}) \quad , s \geq 0 \qquad \ldots (C\text{-}1)$$

Note that there are two parameters, or degrees-of-freedom, λ_i and n_i. Assuming that both the true mate and the impostor distribution follow the form of (C-1), and using eq. (B-4), we obtain

$$TAR = \exp(-\alpha \circ \ln^{1+\beta}((N/FAR)) \quad \ldots(C\text{-}2)$$

where

$$\alpha = (\lambda_2 / \lambda_1)^{n1} \quad \text{and} \quad \beta = (n_1 - n_2)/n_2 \quad \ldots(C\text{-}3)$$

When $n_1 = n_2$ this simplifies to what is known as the ***power-law form***:

$$TAR = (FAR / N)^{\alpha} \quad \ldots(C\text{-}4)$$

The power law is very simple and is often a very good first approximation, not only for OC curves, but for CMC curves as well.

The following provides an example of a good fit between a classically derived OC curve and an empirical one.

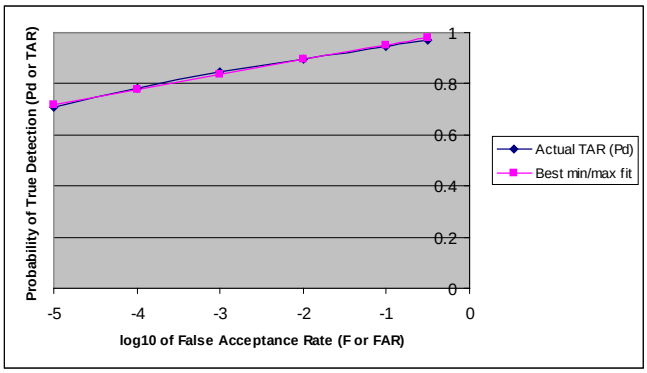

Figure C-2 – Sample Experimental OC Curve and Best Fit using Extended Power-law

Conclusion: "Classically derived" OC function may give good approximations to empirical data providing the parameters are properly selected. Remaining deviations (from empirical data) may then provide a clue to the matcher architecture.

Appendix D – CMC Curves

The CMC was briefly mentioned in Appendix B. It is a kind of distribution function, and provides the probability the true mate will appear on the candidate list (given there is exactly one true mate in the database). It is normally presented as a cumulative distribution, so that P(L) is the probability that the true mate will appear somewhere on a candidate list of length L.

The expression for P(1), the probability the true mate is in first place, is given by

$$P(1) = P_{first} = \int_0^{\infty} (F_1(s))^{N-1} f_2(s) ds \qquad \ldots (D\text{-}1)$$

Where the probability distributions are as defined in Appendix B.

This expression is more complicated than eq. (B-4) for the OC, although there is a close connection between the two. For most functions the above integral **cannot** be evaluated exactly. However, for Weibull distributions with the same n (see eq. (C-1)) a closed-form solution can be obtained:

$$P(1) = \alpha \circ B(N, \alpha) \qquad \ldots (D\text{-}2)$$

In the above, B denotes the beta function, and α is as defined by (C-3). For large N (say a million) (D-2) can be shown to be well approximated by

$$P(1) \approx \Gamma(\alpha+1)/N^{\alpha} \qquad \ldots (D\text{-}3)$$

This can be generalized to arbitrary candidate list lengths, L:

$$P(L) \approx \Gamma(\alpha+1)*(L/N)^{\alpha} \qquad \ldots (D\text{-}4)$$

Since α needs to be small for a reasonably performing system (say 0.05 or smaller), we can approximate Γ(α+1) as 1 – 0.56 α, to arrive at

$$P(L) \approx (1 - 0.56\,\alpha)*(L/N)^{\alpha} \qquad \ldots (D\text{-}5)$$

Further simplification gives

$$P(L) \approx (L/N)^{\alpha} \qquad \ldots (D\text{-}6)$$

Note that this is just the power law, analogous to eq. (C-4). This analogy is not coincidental, as shown by the following argument. A candidate list of length L allows up to L impostors to appear on the list; that is to say, FAR = L if the true mate is not on the list, and FAR = L-1 otherwise. This will be seen to be very similar to the OC of (C-4).

Conclusion: The power law often provides a good approximation to both the CMC and OC.

Appendix E – Methods of Ranking OC Curves

Given two OC curves, we often need to determine which of the two represents superior performance. There are four methods in common use: 1) dominance; 2) area above curve; 3) minimum total error; and 4) equal error rate. Some of these concepts are discussed in ref. [17].

1. Dominance

This simply means that one curve is always higher than the other. Whatever FAR we pick, the superior curve will always have higher TAR. But in the general case the two curves might cross, in which case dominance does not apply.

The following graph illustrates these concepts.

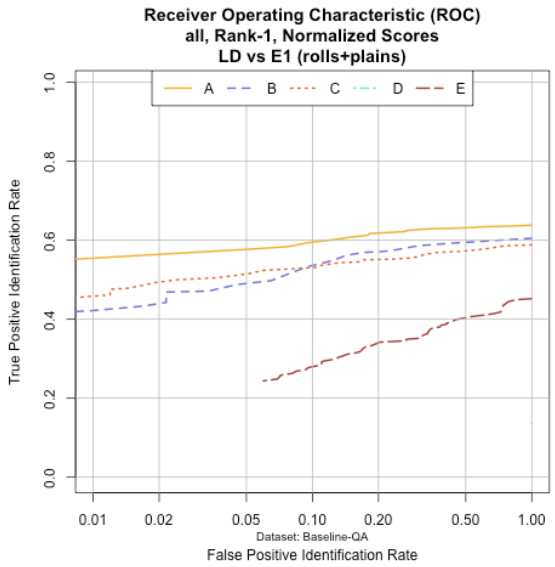

Figure E-1 – Representative Experimental OC Curves

Four curves, A, B, C and E are shown. (D is not shown.) Curve A dominates the other three. Curves B and C cross, so neither dominates the other.

2. Area above curve

The first quantitative measure of goodness, or figure-of-merit (FOM), for an OC curve is the area-above-curve. It represents the area above the OC curve, and below the line y =1. It also equals: 1 - area_below_curve. Note that smaller values indicate the better system, and for very high performance this will be a small number. For a random-guessing system it will be ½.

3. Minimum total error (MTE)

This is the second FOM, and is given by the minimal value of the sum of the two errors: i) the miss rate, 1 – TAR, and ii) the FAR. A factor of ½ is included to make the result consistent with a random-guessing system. The equation is therefore MTE = min{ (1 – TAR + FAR)/2}.

4. Equal error rate (EER)

The point along the OC curve where the two types of errors are equal is known as the equal error point. At this point 1 – TAR = FAR. The EER is then the value of FAR at this point. This is our third FOM. (See ref. [17].)

The three FOMs are related, though generally do not give identical results. As will appear shortly, area-above-curve is the "easiest grader," while EER is the toughest. To investigate the relationship between the three we assume a power-law for the OC. This of course only represents special cases, but was shown in Appendix D to be often a good approximation. Also it allows for analytic expressions for the three error measures (FOMs). The power-law assumes

$$TAR = FAR^{\alpha} \qquad \ldots (E\text{-}1)$$

Where α is a positive constant between zero and unity. Zero represents a perfect system; and unity represents a random guessing ("clue-less") system (this is essentially (B-4) with $F_1 \equiv F_2$.)

For the above functional form the value of area-above-curve (FOM #1) is given by

$$area\_above\_curve = \alpha/(1+\alpha) \approx \alpha - \alpha^2 \qquad \ldots (E\text{-}2)$$

and is therefore very nearly given by α for small α.

The expression for MTE (FOM #2) can be shown to be

$$MTE = (1 - \alpha^{\alpha/(1-\alpha)} + \alpha^{1/(1-\alpha)})/2 \qquad \ldots (E\text{-}3)$$

This cannot be given a power expansion around zero, as was done for (E-2), because there is a singularity at zero. However, a good approximation is provided by

$$MTE \approx 0.5\,\alpha\,(1 - \ln(\alpha)) \qquad \ldots (E\text{-}4)$$

No simple analytic expression appears to exist for EER, but a good analytic approximation – along the lines of (D-4) – is provided by

$$(-0.458*\ln(\alpha) + 1.1911)*\alpha^{1/(1-\alpha)} \qquad \ldots (E\text{-}5)$$

The following graph compares the three types of error measures.

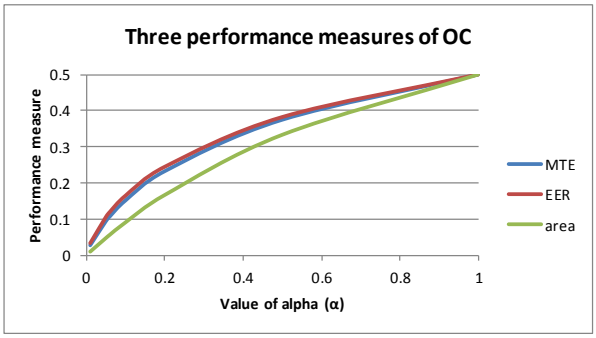

Figure E-2 – Comparison of Three Ways of Measuring Quality of OC curves

As is seen from this figure all three measures are about the same. Area-above-curve is the "easiest grader" (gives most favorable (lowest) scores). MTE and EER are nearly the same, but EER is slightly "tougher."

These results were used in this report to assess the accuracy of MTE estimates.

Appendix F -- Numerical Methods for Computing MTE

In Appendix E we provided algorithms for computing MTE, EER, and area-above-curve. However, these were based on idealized forms of the OC curve. This appendix provides practical algorithms applicable to more general cases. We emphasize MTE as this is extensively used in this report. The algorithms are based on three points taken from the OC. In the first algorithm we used these three points to fit a power-law to the arc spanned by these three points. Several algorithms can be used to find α. Among the best is min-abs-error. That is, we pick that α which makes the errors at the end-points of equal magnitude but opposite sign. We then use eq. (E-3) to actually compute the MTE.

The second algorithm uses quadratic approximation, and does not depend on the power law. Assume the three points are given by (x_1, y_1), (x_2, y_2) and (x_3, y_3), where $x_1 = FAR_1$, and $y_1 = TAR_1$, etc.. Next compute a, b, and c using the following expressions:

$$a = y_1/((x_1 - x_2)(x_1 - x_3)) + y_2/((x_2 - x_1)(x_2 - x_3)) + y_3/((x_3 - x_1)(x_3 - x_2)) \quad \text{… (F-1)}$$

$$b = -y_1(x_2 + x_3)/((x_1 - x_2)(x_1 - x_3)) + -y_2(x_1 + x_3)/((x_2 - x_1)(x_2 - x_3)) + -y_3/(x_1 + x_2)/((x_3 - x_1)(x_3 - x_2)) \quad \text{… (F-2)}$$

$$c = y_1(x_2\, x_3)/((x_1 - x_2)(x_1 - x_3)) + y_2(x_1\, x_3)/((x_2 - x_1)(x_2 - x_3)) + y_3/(x_1\, x_2)/((x_3 - x_1)(x_3 - x_2)) \quad \text{… (F-3)}$$

(These are not the simplest expressions for a, b, c, but are the most symmetric.)

From these we can then compute

$$MTE = (1 + 4a - 2b + b^2 - 4ac)/(8a) \quad \text{… (F-4)}$$

The expression for EER is slightly more complicated (in agreement with the analysis of Appendix D):

$$EER = (-1 - b - \sqrt{((b+1)^2 - 4a(c - 1))})/(2a) \quad \text{… (F-5)}$$

As a check we take three points check these equations against the results of Figure D-2. We take $\alpha = 0.2$, and using this α we compute the following three data points : (0.1, 0.631), (0.2, 0.725), (0.3, 0.786). The assumed FAR values are arbitrary; the TAR value follow from the assumed α. From these three points we compute a = -1.65, b = 1.44, and c = 0.504. These result in MTE = 0.234, and EER = 0.244. Both values are in excellent agreement with the graph of Appendix E.

These results were used in estimating MTE and the associated standard error.

Appendix G – Likelihood and Likelihood-ratios

In Section 7.0 we discuss seven alternative candidates for standardizing the matcher score. These range from simple to quite complex. Simplicity has definite advantages. Such algorithms are easily understood, easily implemented, and the calculated values tend to be more stable. However, they may not provide an accurate estimate of the likelihood that a subject is a true mate. For example, consider a zero-information, or random guessing system. The distribution of scores for true mates and impostors is the same. Assuming there is one true mate in the database, of size N, the probability it will appear in first place is just 1/N – regardless of the score indicated.

Of course AFIS are not random guessing systems -- so they should do a lot better. The simplest assumption is that systems of the same generation perform about the same. This assumption by itself goes a long way toward calibrating and interpreting the standard score.

The term ***likelihood*** is used in statistics to indicate an ***estimate of a probability***. For all practical purposes this is a probability, but there are technical reasons for differentiating the two. ***Likelihood ratio*** is used to denote the ratio of two likelihoods, ususally in the form of favorable outcome to unfavorable. This is similar to the gambler's "odds." When a gambler states the odds of something happening are 3:2 say, what he means is that 3/5 = 0.6 the outcome will occur, and 2/5 = 0.4 it will not occur (and the "opposite" will occur).

Using the same matcher (Matcher B) and dataset (LE) as was used for Figure 11 (Section 7.0), and using algorithm 5, we obtain the distribution of first-place impostor scores as shown in the following figure.

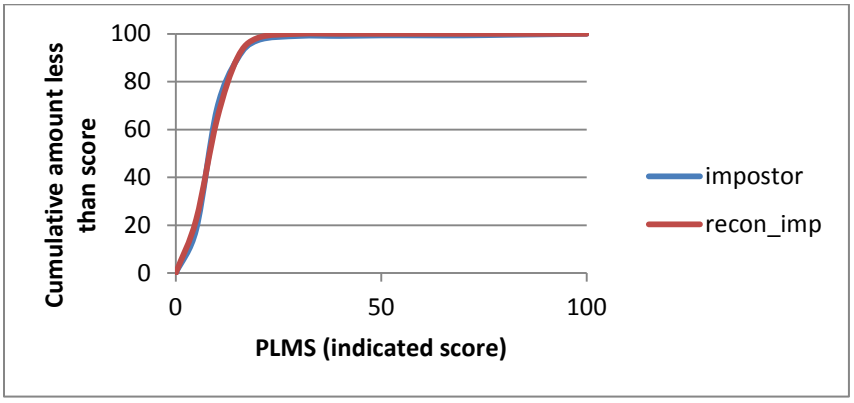

Figure G-1—Cumulative Distribution of Impostor Scores (Matcher B/LE)

The blue line (labeled "impostor") gives the experimentally obtained distribution. The red line (labeled "recon_imp") provides an empirically derived fit to the experimental data. The equation for the reconstruction is a Rayleigh diatribution with parameter 9.9.

The following figure provides the derivative of Figure G-1. This is required for computing the likelihood-ratio.

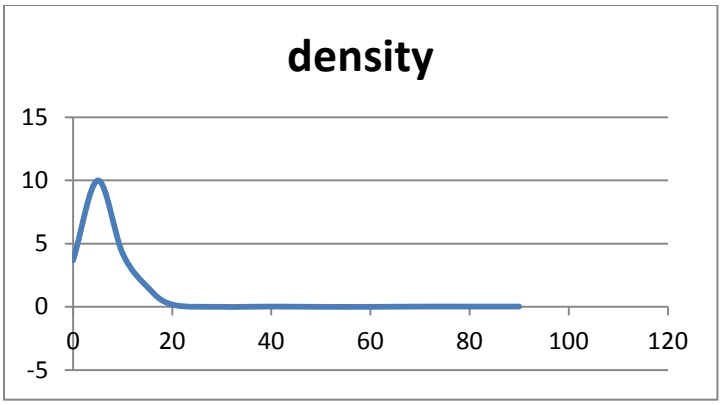

Figure G-2 – Derived Density (based on G-1)

The density function for true mates, by design, is uniform and equal to unity. Suppose a score of S was obtained for the candidate in first place. The probability of a true mate score falling in the interval [S – δ/2, S+δ/2] is simply δ. (Delta (δ) is arbitrary but of course assumed small.) For impostors, the probability is the value indicated in the above graph times δ. The ratio of these probabilities is then the likelihood ratio. It is given below.

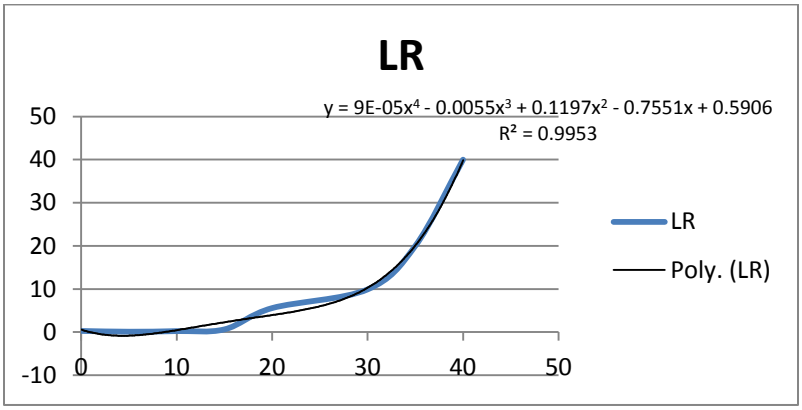

Figure G-3 – Derived Likelihood Ratio (LR) with Smooth Fit

A likelihood-ratio of 10 is achieved at a score of about 30, and implies a probability of 10/11 = 0.91. The next figure shows a smoothed version of the above graph for small scores.

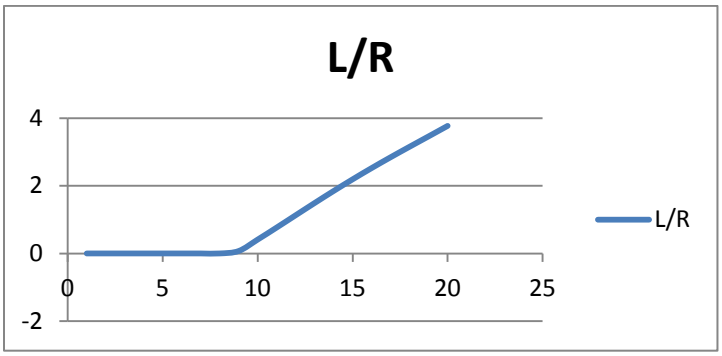

Figure G-4 – Simplified Smooth Approximation for Small PLMS

From this figure we see that a ratio of 1:1 is achieved at about 12, and a ratio of 2:1 at about 14. Note that this is in agreement with the concusion at the end of Section 7.0.

The mean value for impostors in first place for this particular matcher and dataset was about 5.3. A different system will have a somewhat different mean value. Provided the difference is not too great, we can use the mean impostor score to estimate the critical likelihood ratio (= 1). ***Assume the mean impostor score is 10; then it is not unreasonable to assume a likelihood ratio of 2:1 would be achieved at 20.***

Appendix H -- Probability Distributions of Transformed Scores

Suppose we start with the native score, S, and wish to transform this to a new score, S*; furthermore, we would like the new score, S*, to obey a specified distribution function (which might be quite different from that of S). How can this best be accomplished?

Following Appendix B, we assume that the relationship between S and S* is given by

$$S^* = G(S) \qquad \ldots (H\text{-}1)$$

Assume further that the cumulative distribution function for S is given by $F_2(S)$, and that for S* is given by $F_2^*(S^*)$. (We follow the notation of Appendix B, wherein the subscript 2 indicates a true-mate distribution.) We must then have

$$F_2(S) = F_2^*(S^*) \qquad \ldots (H\text{-}2)$$

In view of (H-1) this can be written as

$$F_2(G^{-1}(S^*)) = F_2^*(S^*) \qquad \ldots (H\text{-}3)$$

($G^{-1}()$ indicates the inverse function of G().) Solving for G we obtain

$$G() = F_2^{*-1}(F_2()) \qquad \ldots (H\text{-}4)$$

(For simplicity we omitted the argument in the above. The superscript -1 indicates we are to take the inverse of $F_2^*()$.)

A particularly simple case occurs when $F_2^*(x) = x$, $0 \leq x \leq 1$. That is to say, $F_2^*(x)$ is the ramp (identity) function on [0, 1]. In this case eq. (H-4) reduces to

$$G() = F_2() \qquad \ldots (H\text{-}5)$$

The cumulative distribution of S*is now a ramp function, while the density function is now a constant equal to unity.

Suppose however, that – as suggested in the body of this report – we would like S* to take on values from zero to 100, rather than from zero to 1. Then we need to modify eq. (H-5) to be

$$G() = 100*F_2() \qquad \ldots (H\text{-}6)$$

The resulting density function is again constant, but this time it has value 1/100.

In the above we assumed that it is the distribution of ***true mates*** that we would like to modify. Of course exactly the same procedure can be used to modify the ***impostor*** distribution to some specified functional form. But it is not generally possible to specify both distributions at the same time. For example, it is not generally possible to find a transformation which reduces both true-mates and impostors to a uniform distribution, except for very special cases, such as a random guessing system.

Appendix J – Numerical Example of Distribution Modification

This appendix comprises a numerical example employing the theory of Appendix H. We begin with empirically collected data in the form of a histogram. This generally has the appearance of "unkempt hair," as in the following graph.

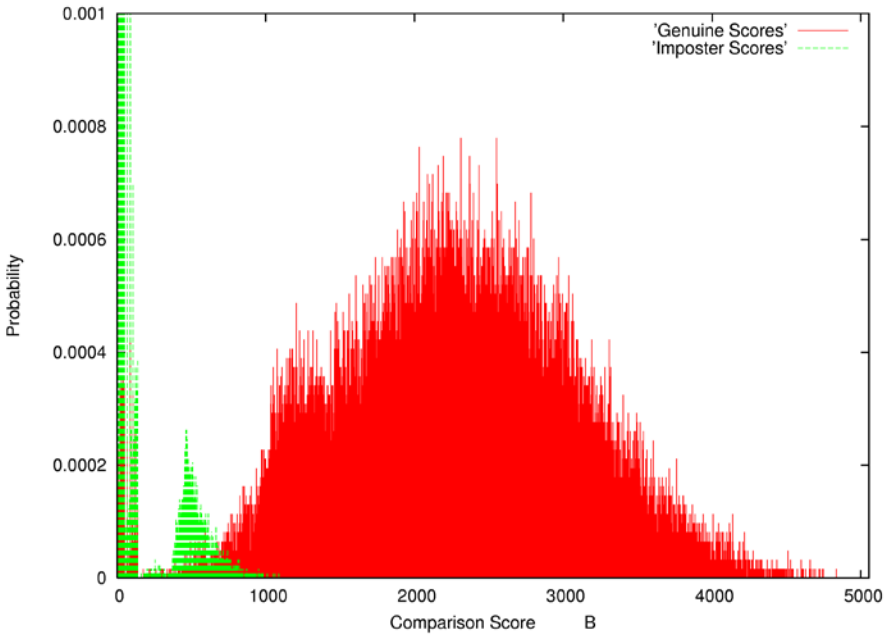

Figure J-1 – Histogram of Score Distributions (raw data)

The next step is to smooth this data by averaging over short intervals taken symmetrically about a point. The resulting function will now be smooth, as in the following diagram, and hence easier to work with.

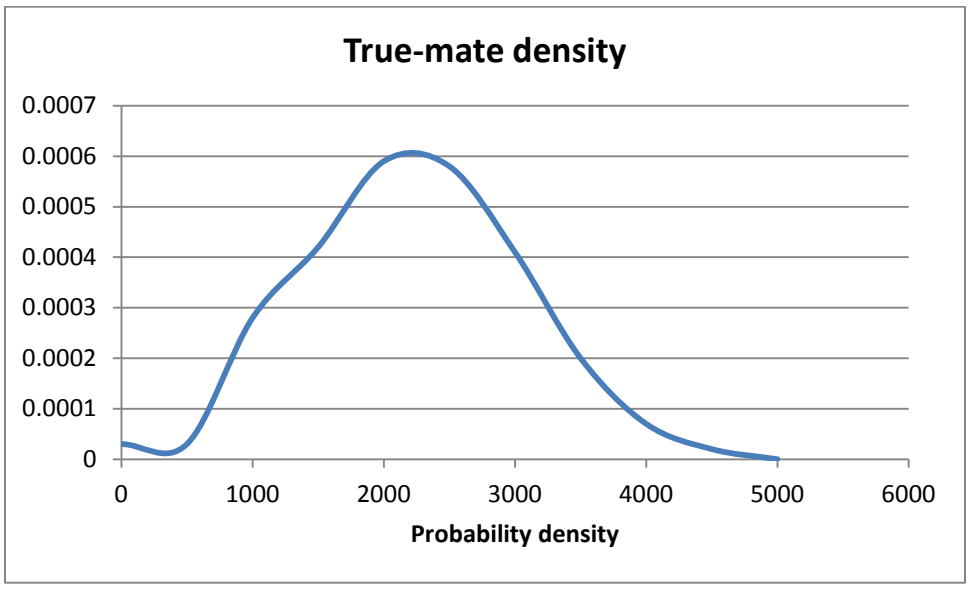

Figure J-2 – Smoothed Score Distribution – True-Mate/Genuine-Scores Only

The deviations between the rough empirical data and its smoothed version are largely due to sampling errors (i.e., random fluctuations). These sampling errors do not contribute to our understanding, so they will be removed, and we work with the smoothed version. The next step is to fit the empirical curve with classical distributions. Two approximating functions were tried: 1) Weibull, and 2) normal (Gaussian). Note that as the empirical distribution has aspects of being tri-modal: there is an anomalous bump at low values, around [0, 400], and another at about 1100. These bumps will appear as errors to the fit. The optimality criterion for the fit was min-max-error. The results of the fit are shown below.

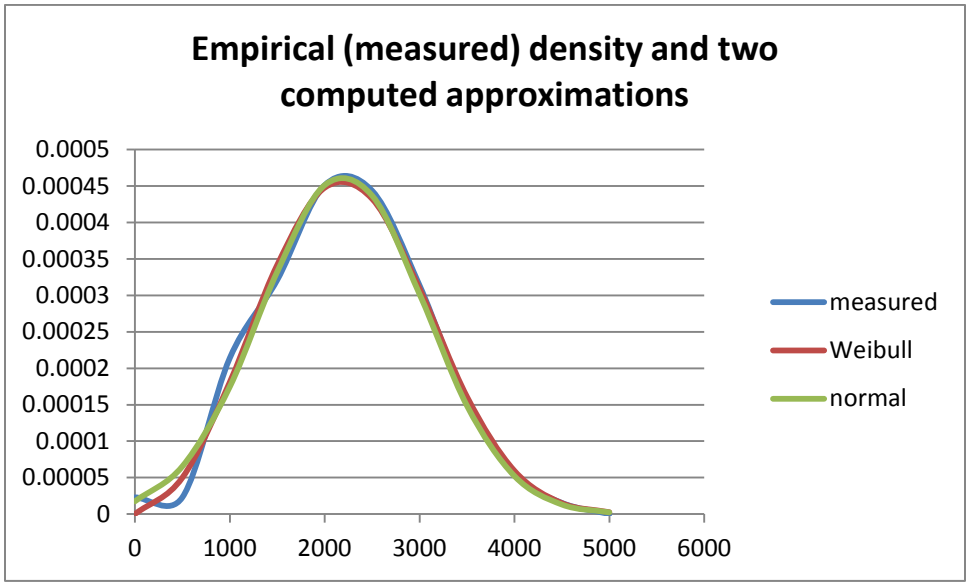

Figure J-3 – Empirical Data with Two Candidate Curve Fits

The errors between the curve fit and the data are shown below.

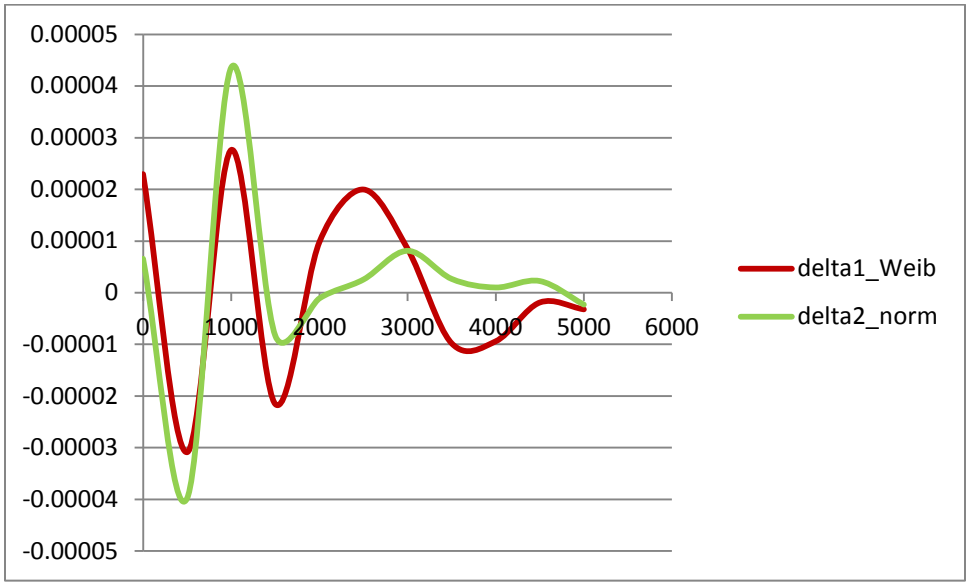

Figure J-4 – Errors in Curve Fit (Deltas)

Note that the max positive error and max (absolute) negative error are about equal in magnitude – a hallmark of the min-max-error method. Also note these maximal excursions occur at about 400 and 1100, as anticipated. The table below shows the values of the parameters produced by the curve fit.

Weibull	
λ	2525
α	2.88
Normal	
μ	2213
σ	855

The Weibull distribution exhibits the smaller errors (Figure J-4), and also is analytically more tractable; for these reasons we select it as our working approximation. The equation for the cumulative distribution is then given by

$$F_2(S) = 1 - \exp(- (S/\lambda)^{\alpha}) = 1 - \exp(- (S/2525)^{2.88}) \qquad \ldots (J\text{-}1)$$

If we want S* to take on values in the interval [0, 100] we need to set

$$S^* = 100^* F_2(S) \qquad \ldots (J\text{-}2)$$

The resulting cumulative distribution for S* is then given by

$$F_2^*(S^*) = S^*/100 \qquad \ldots (J\text{-}3)$$

And the density function is now

$$f_2^*(S^*) = 1/100 \qquad \ldots (J\text{-}4)$$

Strictly speaking, (J-3) and (J-4) apply to the Weibull approximating function, and not to the (smoothed) empirical approximation. The deviation of the cumulative distribution for the empirical data from a uniform distribution is shown below. In general, these deviations are quite small, averaging less than 0.5 %.

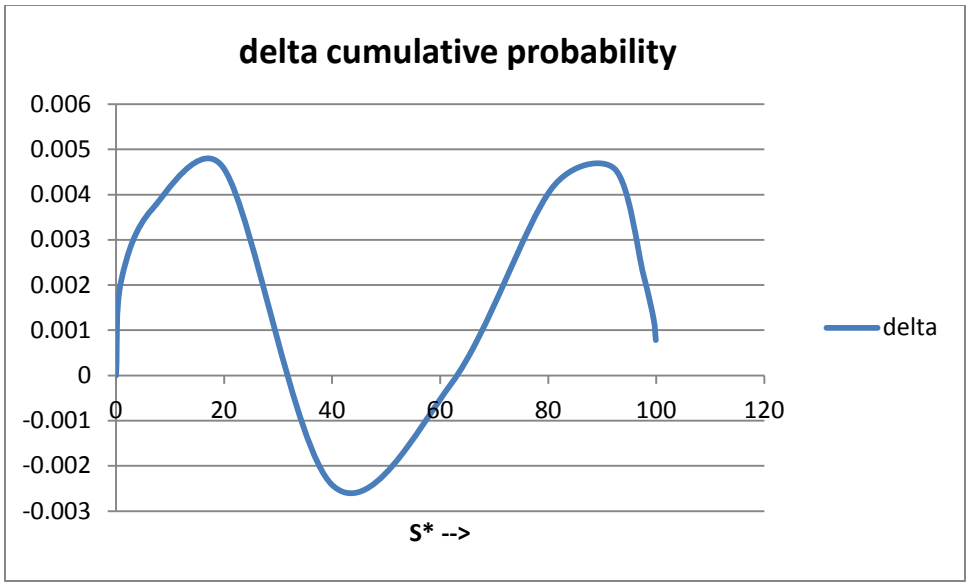

Figure J-5 – Difference (delta) Between Actual Cumulative Distribution and Ideal Ramp Function