1406–1410 Nucleic Acids Research, 1995, Vol. 23, No. 8

The application of numerical estimates of base calling accuracy to DNA sequencing projects

James K. Bonfield and Rodger Staden*

MRC Laboratory of Molecular Biology, Hills Road, Cambridge CB2 2QH, UK

Received November 29, 1994; Revised and Accepted February 17, 1995

ABSTRACT

During DNA sequencing projects one of the most labour intensive and highly skilled tasks is to view the original trace descriptions of gels and to adjudicate between conflicting readings. Given the current methods of calculating a consensus, the majority of the time employed in viewing traces and editing readings is actually devoted to making the poorer data fit the good data. We propose new consensus calculation algorithms that employ numerical estimates of base calling accuracy and which when used in conjunction with an automatic detector of contradictory data should greatly reduce the time spent checking and editing readings and hence improve DNA sequencing productivity.

INTRODUCTION

It is not always appreciated that one of the major advances introduced by DNA sequencing instruments such as the ABI 373A and Pharmacia A.L.F. is that they produce machine readable data. The availability of this data has meant that programs used for checking and editing sequence readings from these instruments (1–3) can display their original traces on the computer screen. Having instant access to the experimental data in this way saves a great deal of time and has hence increased the productivity of sequencing groups. However, we have always felt that the most useful outcome of having a sequence reading determined by a computer-controlled instrument would be that each base was assigned a numerical estimate of its probability of having been called correctly. Although being able to show the original traces is necessary, having numerical estimates of base accuracy is the key to further automation of data handling for sequencing projects.

Previous work in this area has concentrated on methods to correct the original base calls by locating possible errors. One group has described (4) a method of assigning position-specific error probabilities to sequence readings and their application to reconstructing the target sequence and to estimation of the accuracy of the final sequence. Another group has devised (5) a method of calculating confidence values for sequence readings and their use for 'automatic trace editing'. However, our approach is somewhat different. We assume that most of the effect of these useful error detection methods is absorbed into the base calling software to produce more accurate readings. In addition, we assume that new base calling algorithms will produce numerical estimates of the accuracy of each base.

The simple procedure that we propose in this paper is a method of using the numerical estimates of base calling accuracy to obviate much of the tedious and time consuming trace checking currently performed during a sequencing project. In summary, we propose that the numerical estimates of base accuracy should be used by software to decide if conflicts between readings require human expertise to help adjudicate. We argue that if the accuracy estimates are reasonably reliable then the majority of conflicts can be ignored (i.e. the traces do not need to be examined by eye) and so the time taken to check and edit a contig will be greatly reduced.

In part to prepare for this, in 1992 we published a file format (SCF) for data from DNA sequencing instruments (6). Files in this format can store the called sequence, the trace descriptions and for each called base four numerical values to represent the probability of the base being A, C, G or T. (The current specification of this format is obtainable via anonymous ftp from ftp.mrc-lmb.cam.ac.uk/pub/staden/docs/text/scf.) We are aware of several groups working on improved base calling algorithms that will produce numerical estimates of base accuracy. In the expectation that the values will be available in the near future, we have been making further preparations for applying them and this is what we describe in this paper.

MATERIALS AND METHODS

The methods described here are used in the Genome Assembly Program (GAP), which runs on DEC, SUN and SGI UNIX computers. The program can handle data from ABI, Pharmacia A.L.F. and LI-COR fluorescence-based sequencing machines and the Amersham Film Reader. The LI-COR and Amersham instruments write their data in SCF format and, although it is not essential, we recommend users to employ one of our utility programs (makeSCF) to convert 373A and A.L.F. data to SCF files (for 373A files this can give a 70% saving in disk space if the original files are deleted). As we described in a previous paper (1) the traces from these instruments can be displayed on the user's screen. The original digitized images from the Amersham Film Reader can also be displayed. In order to calculate the simple estimates of base accuracy that GAP currently employs, the

* To whom correspondence should be addressed

program EBA (Estimate Base Accuracy) is applied to files in SCF format. The programs described here are available as part of our sequence analysis package.

RESULTS

Consensus algorithms

Consider a contig of length n. We want to calculate a consensus sequence s_i, $i = 1,\ldots, n$ composed of the symbols in the set $X = \{A, C, G, T, *, -\}$ from the aligned readings that also contain symbols in the set X. Here the symbol * means a padding character introduced to achieve alignment and the symbol – means an indeterminate base.

Let the depth or number of readings covering position i in the contig be d.

Let the symbol in a reading at position i, $i = 1,\ldots, n$ and depth j, $j = 1,\ldots,d$ be x_{ij}, $x \in X$.

Let the probability distribution for the contig position i, $i = 1,\ldots, n$, for the reading at depth j, $j = 1,\ldots,d$ be $p_{ij}(x)$, where x is a member of $Y = \{A, C, G, T, *\}$.

Let q_{ij} be the $p_{ij}(x)$ for the called base x_{ij}, i.e. q_{ij} is the accuracy estimate for the called base.

Let the frequencies of the called bases at position i be $f_i(x)$, $x \in Y$

For every position i there will be a $u \in Y$ such that $f_i(u) \geq f_i(x)$, $x \neq u$.

Let the sums of the $p_{ij}(x)$ for each $x \in Y$ be $t_i(x) = \sum_{j=1}^{d} p_{ij}(x)$ (1)

For every position i there will be $v,w \in Y$ such that:
$t_i(v) \geq t_i(w) \geq t_i(x)$, $x \neq v,w$

Let the sums of the q_{ij} for each $x \in Y$ be $z_i(x) = \sum_{j=1}^{d} q_{ij}$ (2)

For every position i there will be a $y \in Y$ such that:
$z_i(y) \geq z_i(x)$, $x \neq y$.

Using these definitions we define four consensus rules:

Rule I If $f_i(u) / \sum_{x \in Y} f_i(x) \geq C_I$ then $s_i = u$, else s_i= '–'

Rule II If $t_i(v) / \sum_{x \in Y} t_i(x) \geq C_{II}$ then $s_i = v$, else s_i = '–'

Rule III If $t_i(v)/t_i(w) \geq C_{III}$ then $s_i = v$, else s_i = '–'

Rule IV If $z_i(y) / \sum_{x \in Y} z_i(x) \geq C_{IV}$ then $s_i = y$, else s_i = '–'

Here C_I, C_{II}, C_{III} and C_{IV} are minimum values set by the user. Notice that we would need to apply a correction for small samples and that this would avoid division by zero occurring. Rule II defines the probability that the highest scoring base is correct and Rule III returns the relative likelihood of the most probable to the next most probable.

For each reading r of length r_l starting at position r_b in the contig, we can calculate a measure of its overall quality Q_r by summing its q_{ij} values and dividing by its length. Note that for any reading r the depth j is fixed.

$$Q_r = 1/r_l \sum_{k=r_b}^{r_b+r_l-1} q_{kj} \quad (3)$$

Alternatively, following (7) we might use the average entropy for the reading given by

$$E_r = -1/r_l \sum_{k=r_b}^{r_b+r_l-1} \sum_{a \in X} p_{kj}(a)\log_2(p_{kj}(a)) \quad (4)$$

The rules defined above can be clarified by a simple example. Suppose we had four readings covering some position in a contig and they had the values shown in Table 1. We do not know the form the numerical estimates of accuracy will finally take, but here, for simplicity, we have assumed that each base position is given four values that usually sum to 1.0.

Table 1. Example data to demonstrate the definitions

j	p(A)	p(C)	p(G)	p(T)	p(*)	q_{ij}	x_{ij}
1	0.9	0.1	0.0	0.0	0.0	0.9	A
2	0.8	0.0	0.1	0.1	0.0	0.8	A
3	0.8	0.1	0.1	0.0	0.0	0.8	A
4	0.3	0.2	0.1	0.4	0.0	0.4	T
$t(x)$	2.8	0.4	0.3	0.5	0.0		
$z(x)$	2.5	0.0	0.0	0.4	0.0		

Here f_i(A) = 3, f_i(T) = 1 and f_i(C) = f_i(G) = f_j(*) = 0; $\Sigma t_i(x) = 4.0$; $\Sigma z_i(x) = 2.9$

Rule I If 0.75 (3/4) $\geq C_I$ then s_i = A
Rule II If 0.7 (2.8/4.0) $\geq C_{II}$ then s_i = A
Rule III If 5.6 (2.8/0.5) $\geq C_{III}$ then s_i = A
Rule IV If 0.86 (2.5/2.9) $\geq C_{IV}$ then s_i = A

Checking and editing assembled readings

We assume the readings have been correctly assembled into contigs (8) and that all bases in the consensus are covered by data from both strands. To finish the sequencing project the contig must be checked and edited. Editing a base achieves several things. It removes a problem, it helps to produce the correct consensus sequence and, if it is done in a different case to the original data (for example original data in upper case and edits in lower case), as we would recommend, it marks the changed character as having been edited by hand. Below we describe the current editing strategy and then the proposed strategy.

Our assembly program GAP contains an editor (1) that allows the user to scroll rapidly along a contig to view and edit the aligned readings. (A typical display of the editor, which is described later, is shown in Figure 1.) The contig editor calculates a consensus for the aligned readings and displays it at the bottom of the window. One of the commands available in the editor is 'Next Problem'. If the user selects this command the editing cursor moves to the next position in the contig where the consensus is unresolved (and hence contains '–'). The user would then double click on a mouse button and the traces for the aligned readings would immediately appear in a window on the computer screen. Having examined the traces the user decides what the correct sequence should be and edits the readings accordingly. The consensus sequence will change and the

Figure 1. A view of the contig editor in which high quality bases are shown in black, poorer quality data in red and edited bases in lower case characters.

'Next Problem' command will shift the cursor to the next unresolved base in the consensus.

With this strategy the user can quickly move from problem to problem to adjudicate between conflicts. However, for a cosmid sized project, it still takes a long time, is very tiring and hence prone to error.

One reason this task takes so much time is that in order to be sure of checking all possible errors the consensus cut-off C_I used by the 'Next Problem' command is set to require 100% agreement between the readings at each point. This level of caution (and consensus rule I) is employed because with these methods, until the traces are viewed, the user has no way of knowing the relative quality of the different readings.

Using the new consensus rules

We propose the use of consensus rules II or III and believe that by effectively weighting the contributions of each reading in proportion to their accuracy estimates, the number of conflicts that need checking will be greatly diminished. The actual level of diminution achieved will naturally depend directly on the reliability of the accuracy estimates. If the estimates are reasonably reliable, conflicts should be caused by readings with low accuracy values and, if they are aligned with readings with high values, then use of the weighted consensus calculation will mean they are ignored by the 'Next Problem' command (assuming the cut-off is not set to 100%). Users should find that they are only called upon to adjudicate conflicts between apparently high quality readings.

Every base in every reading is given an estimate of its accuracy. In the long-term these estimates will be produced by the base calling software, but at present we calculate our own rough values for data in SCF files. Although we do not know how the traces relate to the original band intensities on the gel, the following calculation has provided values suitable for software development and demonstration purposes. We divide the area under the peak for the called base by the area under the next highest peak at the same position. This returns only a single number (normalized to lie between 1 and 99) for each base position and is the equivalent of the q_{ij} figures defined above. We would expect base calling software to provide four values, represented above by p_{ij}, for each base position. These values are stored in SCF files and copied into GAP databases during assembly and are then used in the consensus calculation. Because we only have the q_{ij} values, we currently use consensus Rule IV. Note that if the consensus is being displayed in the contig editor, any padding characters it includes are shown to maintain alignments, but if the sequence is being written to a file for analysis, the padding characters are removed.

The main programing effort in the work reported here is the redesign of the sequence assembly database to include the accuracy estimates, writing the routines for reading and manipulating the values and their use by the consensus calculation and the 'Next Problem' command. Now that this work has been completed, the changeover to values more reliable than our present crude estimates should be relatively straightforward, once these become available.

In addition to the consensus rules and their use by the 'Next Problem' command defined above, we have devised two further variations. The first is that the consensus calculation can be made strand sensitive. Some problems, such as compressions, may only be revealed by data from one strand of the DNA and so it is essential to have readings from both. Here the consensus rule is applied separately to each strand and then only if both strands return the same symbol is the consensus not '–'. This helps find regions without good data on both strands of the sequence and which therefore may contain hidden errors. Again, this new consensus rule can be used by the 'Next Problem' command. The other variation is the use of 'Next Problem' in a mode in which the cursor is moved sequentially to each previous edit so that, if necessary, it can be double checked as part of a quality control system.

Figure 1 shows an example of the contig editor in the program GAP. Ranged along the top of the editor window are a set of command buttons (for example 'Next Problem') and two 'repeater' buttons. One (labelled C and here set to 70%) is used to change the consensus cut-off (used by the current consensus Rule) to values between 1 and 100% and the other, labelled Q, is used to change the accuracy cut-off (here set to 81%). Below this are a set of scroll buttons and a scrollbar. The main part of the window contains the data for the readings covering this region (2841–2920) of the contig. Each line contains a reading number,

its name and a segment of its sequence. Negative reading numbers indicate readings that are in the reverse orientation. The bottom line contains the consensus for the data aligned above.

A number of relevant points are illustrated by the data in this figure. The characters in black are the ones whose accuracy estimate is ≥81%, those in red are <81%. These accuracy estimates have been calculated by program EBA using the simple peak area measure defined above. The data shown is for a finished cosmid that has been edited using consensus Rule I and for which all conflicts have been checked and changes made to give a 100% consensus. Altered bases are in lower case letters and are automatically assigned an accuracy value of 100% (and hence are written in black).

In this figure there are 26 lower case characters, but originally there are likely to have been several padding characters inserted to achieve alignment, which have been deleted during the editing process, i.e. there would have been more editing performed than is now evident. Notice that 73% (19) of the lower case characters are within or adjacent to red segments and so are judged to be of low quality.

If this 80 character wide segment of the cosmid is typical (and we believe it is) and if the user could confidently ignore these 19 changes, then this translates to a saving of 7600 ($19 \times 32\,000/80$) sets of trace viewings and edits over the full 32 000 characters of this cosmid.

DISCUSSION

In summary, we have devised what we believe is an important use for numerical estimates of base accuracy. We have already (6) introduced a file format for storing the accuracy values along with their trace descriptions. Now we have redesigned our assembly database to be able to store these values and have written the code to transfer them from the SCF files and to employ them during weighted consensus calculations. The consensus calculation is used by an error detection command that should find only positions for which the evidence is poor and hence will pass over regions where, although there are conflicts between readings, there is good data. The level of accuracy for the readings in a contig can be displayed in the contig editor. In fact, by setting the accuracy value (denoted by Q in the editor) to 0 and then using the repeater button to gradually increase Q to 100%, the user can observe as each of the characters in the contig editor window change from black to red.

In the worked example of the consensus calculations we say that we assume for simplicity that the numerical estimates of accuracy sum to 1.0. The assumption is made only to explain the numbers in the table and is not a component of the consensus calculations. Indeed, in general we would not expect this to be the case: rather, the sum of the values would reflect the quality of the data from each region of the gel. Our SCF format and the calculations described here have assumed that four accuracy estimates for each called base are sufficient. However, it is possible that accuracy estimates may be produced with four values for each base plus a further number to define the likelihood of an insertion or deletion. In this event we would need either to combine the extra component with the other four or to extend the SCF and GAP file formats. Both formats are designed to be readily modified. Separate indel values would also require changes to the consensus calculations. Most people use the base calling software that accompanies the commercial sequencing instruments. As far as we know, of these only the Amersham Film Reader produces confidence values and there is only one per called base. The only non-commercial base calling software we are aware of (9) also produces a single value for each called base. This group also report plans to use confidence values to aid assembly.

Whatever form the numerical estimates of accuracy finally take, once they are available one important task will be to compare them to the edits made during real sequencing projects. That is to ask, for each level of accuracy estimate, what is the frequency at which edits are made. Such figures can provide confidence for each level of accuracy estimate. The GAP program is ideal for performing such a survey, as it stores the numerical estimates and also records all changes made to sequences.

Note that the consensus algorithm currently employed in GAP has a further refinement on the methods described above, the legitimacy of which we are uncertain. Instead of using all readings to produce a weighted consensus, the algorithm only uses those characters whose accuracy value reaches the cut-off value Q, i.e. not only is poor data weighted down, but the data from the very worst readings is ignored altogether. This refinement of the consensus calculation fits with the general notion that there are good readings and bad readings and that only good data should be employed when deciding the final sequence.

A key point about the editing performed under our previous strategy is that in order to produce the desired consensus, disagreements needed to be edited and most editing was done to make poor data agree with good data, i.e. it is the poor data that takes up the time. For example, if there are several poor readings and one good reading covering a particular position and the poor ones disagree with the good one, they would all need changing to produce the correct consensus. By employing the numerical estimates of base calling accuracy in the way outlined above it is likely that none of the disagreeing bases would need to be edited to produce a good consensus and so less work would be required.

Another consequence of the new strategy is that as fewer bases need changing to produce the correct consensus, most of what appears on the screen will be the original called bases. Indeed, we have taken this a step further and suggest that if a base needs changing because it has a high accuracy estimate and is conflicting with other good data, then rather than change the character shown on the screen, the user should lower its accuracy value. By so doing more of the original base calls are left unchanged and hence are visible to the user. As stated above, the current accuracy estimates produced are normalized to lie between 1 and 99. This allows us to reserve values 0 and 100 for hand edited bases. There is a function within the contig editor to reset the accuracy value for the current base to 0. Alternatively, the accuracy value for the base that is thought to be correct can be set within the contig editor to 100. The policy of changing as little as possible of the original data and yet still achieving a good consensus fits with the comments of Churchill and Waterman (7), who point out that in order to be able to estimate the accuracy of the final consensus the minimum number of changes should be made.

A natural and important outcome of the new strategy, though one which may necessitate more rather than less work, is that the 'Next Problem' function will not only find places where there are disagreements between good readings, but also places where there is no data of sufficient accuracy. Previously some groups

may have overlooked such regions if they relied only on locating regions of conflict and were satisfied if the sequence was covered by data on both strands.

We have evaluated the usefulness of the accuracy estimates we currently calculate by examining several finished cosmids sequenced using the ABI 373A instrument. As expected, our accuracy estimates are negatively correlated with the number of edits made. Figure 1 also shows that 73% of edits were within the lower accuracy values calculated by this method. In addition, it can be seen that, as would be expected, the ends of the readings are predominantly coloured red. However, our reason for calculating the values was not that we want to use them to edit real data, rather we needed some sensible figures to allow us to implement our ideas in GAP. Assuming that accuracy estimates in which we can be more confident soon emerge from new base calling software, we see no reason to explore our crude measures further. However, if there is a delay in the provision of better values, there is now probably sufficient finished data accumulated by users of our software to permit an analysis of the reliability of our accuracy estimates.

We have described what we believe to be the most important application of base calling accuracy measures, which is to reduce human intervention during sequence assembly, but there are other areas where the values could be applied. Earlier (equation 3) we defined a measure of the average quality of a reading. These average quality measures are stored along with the readings and so a temporal order of assembly could easily be set up by sorting the readings on this value. The outcome would be that the highest quality readings could be sent to the assembly program first and the lowest quality ones last. This should assist correct assembly. Another problem that can be better addressed using base accuracy estimates is that of deciding how much of the data at the 3′-end of a reading should be ignored during assembly. In addition, the accuracy estimates can be used in weighted alignment algorithms during the assembly process.

Although the work described here is necessarily incomplete because the final implementation will depend on the properties of the numbers produced by new base calling software, we believe it is valuable, because it shows a way of increasing sequencing productivity and, in so doing, strengthens the case for numerical estimates of base calling accuracy to be made available as soon as possible.

ACKNOWLEDGEMENTS

We thank Molly Craxton, Andrew McLachlan and Tony Crowther for critical reading of the manuscript. We are also grateful to the users, particularly those at the Sanger Centre, who have helped in the development of our software. This work was partly supported by an MRC HGMP grant to RS.

REFERENCES

1 Dear,S. and Staden,R. (1991) *Nucleic Acids Res.*, **19**, 3907–3911.
2 Gleeson,T. and Hillier,L. (1991) *Nucleic Acids Res.*, **19**, 6481–6483.
3 Smith,S., Welch,W., Jakimcius,A., Dahlberg,T., Preston,E. and Van Dyke,D. (1993) *BioTechniques*, **14**, 1014–1018.
4 Lawrence,C.B. and Solovyev,V.V. (1994) *Nucleic Acids Res.*, **22**, 1272–1280.
5 Lipshutz,R.J., Taverner,F., Hennessy,K., Hartzell,G. and Davis,R. (1994) *Genomics*, **19**, 417–424.
6 Dear,S. and Staden,R. (1992) *DNA Sequence*, **3**, 107–110.
7 Churchill,G.A. and Waterman,M.S. (1992) *Genomics*, **14**, 89–98.
8 Staden,R. (1980) *Nucleic Acids Res.*, **8**, 3672–3694.
9 Giddings,M.C., Brumley,R.L.,Jr, Haker,M. and Smith,L.M. (1993) *Nucleic Acids Res.*, **21**, 4530–4540.