Sequence Specific Effects on the Incorporation of Dideoxynucleotides by a Modified T7 Polymerase

Thesis by Alan-Philippe Blanchard

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1993

(Defended February 17, 1993)

Acknowledgements

It would be an undertaking in itself to acknowledge all the people who contributed, in all the various ways, to making this work a reality. So let me simply extend my thanks

- To Jim Kajiya, who had faith in me when others didn't.
- To Yaser Abu-Mostafa, who succeeded, where many others had failed, in teaching me what entropy *really* is.
- To John Hopfield, who rarely needed more than a sentence or two to explain the flaws in my thinking.
- To Lee Hood, whose eclectic lab provided an incomparable environment for doing this work.
- And to all the Old Darbs who made Caltech an oasis in the vast desert of Los Angeles.

Abstract

While incorporating nucleotides onto the end of a DNA molecule, DNA polymerases selectively discriminate against dideoxynucleotides in favor of incorporating deoxynucleotides. The magnitude of this discrimination is modulated by the template DNA sequence near the incorporation site. This effect has been characterized by analyzing the raw data from a large number of DNA sequencing experiments. It is shown that, for bacteriophage T7 polymerase, the 5 contiguous bases extending from 3 bases 3′ (on the template strand) from the incorporation site to 1 base 5′ of the incorporation site are the most important in modulating dideoxynucelotide discrimination. A table of discrimination ratios for 1007 different 5-mer contexts is presented.

Contents

Chapter 1

Introduction

Polymerases are one of the ubiquitous enzymes of Life. Their function is to duplicate a DNA molecule by synthesizing its complementary strand from deoxynucleotides. *In vivo*, this is one of the essential steps of reproduction. *In vitro*, these enzymes are one of the essential tools of the modern molecular biologist.

Like most things in Life, polymerases are not perfect. They will, on occasion, incorporate the wrong nucleotide. For living creatures, trying to propagate their genome as faithfully as possible, this is a problem necessitating more enzymes to correct the mistakes polymerases make. For the molecular biologist or clinical pharmacologist, this is an opening to be exploited for all it's worth.

The mechanisms behind the ability of DNA polymerase to accurately copy a DNA molecule are poorly understood at best. One avenue of attack is to study the cases where the polymerase makes a mistake. The exponential growth of DNA sequencing in recent years provides a wealth of data with which to study a particular type of mistake made by a particular DNA polymerase.

Nucleotide analogues are an important class of anti-viral agents (e.g., acyclovir, dideoxycytosine, and dideoxyinosine, all of which have clinical applications in the

treatment of AIDS) by virtue of the fact that polymerases will incorporate these unnatural molecules sufficiently often to disrupt viral reproduction. The molecular biologist exploits the same effects of the same molecules (the dideoxynucleotides and their analogues) to determine the sequence of DNA molecules.

The present work is essentially a characterization of how the local DNA sequence affects the ability of a modified T7 polymerase to distinguish between deoxy- and dideoxy-nucleotides.

The primary data for this study was generated by others for the purpose of sequencing murine T-cell receptor genes. The enzyme and reaction conditions were chosen on the basis of what would be best for sequencing, not a study of enzyme properties. Their analysis of the data was directed solely towards extracting sequence data. I have re-analyzed the primary data with less emphasis on determining sequence (which is now known) and more emphasis on extracting quantitative incorporation data. Much of this work is concerned with reducing the voluminous primary data (digitized color images of electrophoresis gels) down to a measure of dideoxynucleotide incorporation. The last part of this thesis is devoted to a statistical analysis of the data and examining the role of sequence context in determining the dideoxynucleotide incorporation.

1.1 Previous Work

There has been very little work done in this area. The fact that certain sequences lead to reproducibly anomalous dideoxynucleotide incorporation was noted in Sanger's original paper on enzymatic DNA sequencing [Sanger et al. 1977]. Comparisons of this effect between different polymerases (Klenow, T7 and Taq) were done by Sanders [Sanders et al. 1989]. Sanders presents statistics from a total of 903 bases sequenced by each of the 3 polymerases, but does not relate this to sequence context other than

the base being incorporated.

Kristensen presents the most detailed analysis of all [Kristensen et al. 1988]. Using T7 polymerase, he extracted the normalized amplitude of $\approx$ 1500 bases and published a list of 41 sequence-contexts (3 bases on either side of the incorporated base, which was always C) that produce low peaks. No data was published concerning the other 3 bases or of contexts that gave rise to high peaks.

Another effect that might be related is the tendency for certain sequences to slow down the replication of DNA ("pause sites"). Evidence that a pause site is also a site for increased nucleotide mis-incorporation is given by Fry [Fry and Loeb 1992]. A possible theoretical connection between pausing and mis-incorporation is given by Hopfield [Hopfield 1974]. Pause sites have been mapped in mouse mitochondrial DNA using *Drosophila* polymerase α [Kaguni and Clayton 1982], phage $\phi X174$ DNA using monkey polymerase α [Weaver and DePamphilis 1982] and bacteriophage fd DNA using T4 polymerase [Bedinger et al. 1989], among others. Pausing has also been observed in RNA transcription. The sum total of all this pausing data has been insufficient to find a consensus pattern [Yager and von Hippel 1987].

1.2 This Work

Chapters 2,3 and 4 provide an introduction to the theory and practice of DNA sequencing. Details of the experimental set-up are provided, along with the details of the experiments themselves that relate to the interpretation of the experimental data.

Chapter 5 is concerned with the first stage of data reduction; reducing the megabytes of image data down the more useful and manageable fragment files. This includes the novel use of an image processing step to mitigate the effects of certain experimental artifacts. Chapter 6 details the process of extracting quantitative information about all the products of the sequencing reactions. The novel aspect of both stages of

data reduction is the attention paid to extracting the best quantitative information possible from the original measurements. The previous work in this field was never concerned with extracting anything other than the raw sequence information.

Chapter 7 presents a statistical analysis of the data from over 100,000 sequenced bases. Certain patterns in the effect of sequence context on dideoxynucleotide incorporation stand out in statistically important magnitude. This raises the level of characterization of this effect from the merely anecdotal and qualitative to that of statistically defensible quantitative measurement.

1.3 An Aside

In the process of doing this project I had to write software that could automatically reduce raw sequencer data to DNA sequence. Such software is becoming more and more important as a tool as the amount of DNA sequencing grows. Indeed, the present work started out as a project to improve such tools. But my conclusion, after studying the whole process of dideoxy sequencing, is that the old computer adage, Garbage In, Garbage Out applies here. No amount of sophisticated data analysis will ever bring back information that was lost at the lab bench. Present technology will yield a consistent $400-450$ bases of sequence per fragment on a fluorescent sequencer. After looking at the raw data from hundreds of fragments, I estimate, even with the most elaborate computer processing, that that data wouldn't yield more than an extra ≈ 50 bases per fragment. On the other hand there are examples of data, from a different experimental set-up (longer, thinner gels, higher fields, etc.), that even the most simplistic software would have no trouble calling out to 800+ bases. One can argue that this beautiful data is not representative of day-to-day experiments, but it is proof, by existence, that there are much greater gains to be had by improving the experimental technique than by relying on computer magic. For this reason I decided

not to pursue the problem of base calling further than necessary for the project at hand.

Chapter 2

DNA, Polymerases and Dideoxy Sequencing

2.1 Chemistry of DNA and its Polymerases

Deoxyribonucleic acid (DNA) is the name given to the class of molecules formed by linking together molecules of the 4 deoxynucleotides in linear chains. Each of the 4 deoxynucleotides consists of a phosphorylated (at the 5′ position) sugar moiety attached to a base; one of adenine (A), thymine (T), cytosine (C), or guanine (G). The sugar moieties of two deoxynucleotides are linked by phosphodiester bridges between the 3′-OH of one deoxynucleotide and the 5′ phosphate of the next (Figures 2.1 and 2.2). This gives the chain an orientation. By convention, a DNA molecule is identified by writing out the sequence of its bases in order from the 5′ to the 3′ end. DNA molecules can range in size from a single deoxynucleotide to chains 200 million deoxynucleotides long. There is no restriction on the order of the bases and there are known procedures that can, in principle, synthesize any given sequence of any given length. Commercial machines exist which automatically, as a routine matter,

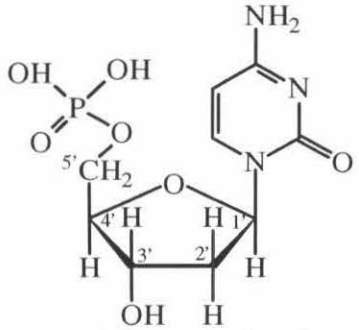
synthesize arbitrary sequences of 100 deoxynucleotides.

When there is no danger of confusion, deoxynucleotides are often refered to simply as nucleotides or by the initial letter of their associated base (A,T,C or G).

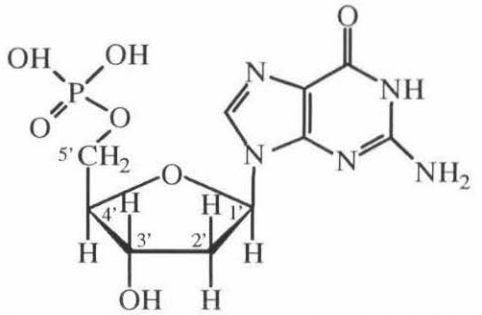
In 1953 Watson and Crick showed that molecules of DNA occur naturally as double helices, consisting of two complementary DNA molecules held together by hydrogen bonds between the bases. The two strands of DNA in a double helix are oriented oppositely from each other with the 5′ end of one strand adjacent to the 3′ end of the other. Opposing bases are "complementary," with A and T forming a complementary pair as do C and G. Thus, the sequence and orientation of one strand determine the sequence and orientation of the other strand.

It was not lost on Watson and Crick that an obvious way of duplicating a molecule of DNA would be to "unzipper" a double helix, use each strand as a template for constructing its complementary strand and end up with two identical double helices. This is exactly how Nature does it.

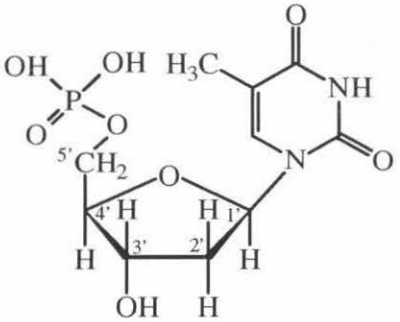
While the full process of DNA replication is rather complicated, the main process involves a class of enzymes known as DNA polymerases. A typical polymerase requires three main things: a piece of template DNA, a short piece of DNA (or RNA) complementary to the 3′ end of the template (the "primer"), and a supply of the 4 deoxynucleotide triphosphates. Under conditions of physiological pH and temperatures, two pieces of complementary single-strand DNA will spontaneously line up and form a double helix in solution. Thus the primer DNA will attach itself to the template DNA without help. A polymerase molecule will then attach itself to the template-primer complex near the 3′ end of the primer and extend it by covalently attaching the appropriate deoxynucleotide. The extended primer thus remains complementary to the template DNA. At this point the polymerase molecule slides down one deoxynucleotide on the template-primer complex and repeats the process. The reaction is driven by the fact that splitting off the pyrophosphate group from the



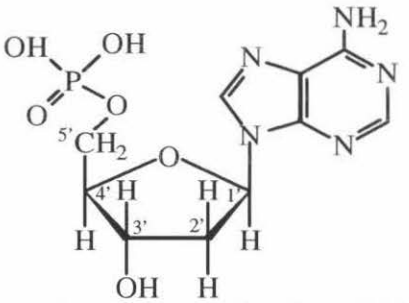
Deoxycytidine monophosphate (dCMP)



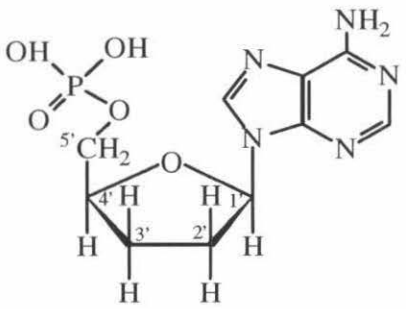
Deoxyguanosine monophosphate (dGMP)



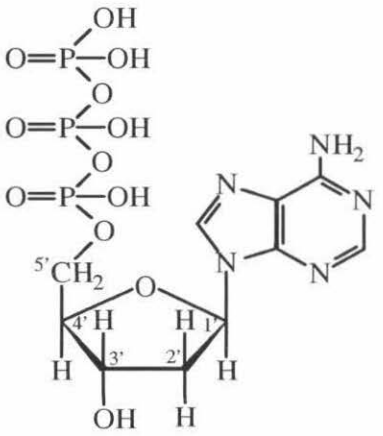
Deoxythymidine monophosphate (dTMP)



Deoxyadenosine monophosphate (dAMP)



Dideoxyadenosine monophosphate (ddAMP)



Deoxyadenosine triphosphate (dATP)

Figure 2.1: Molecular structures of the four deoxynucleotides and their dideoxy and triphosphate analogs.

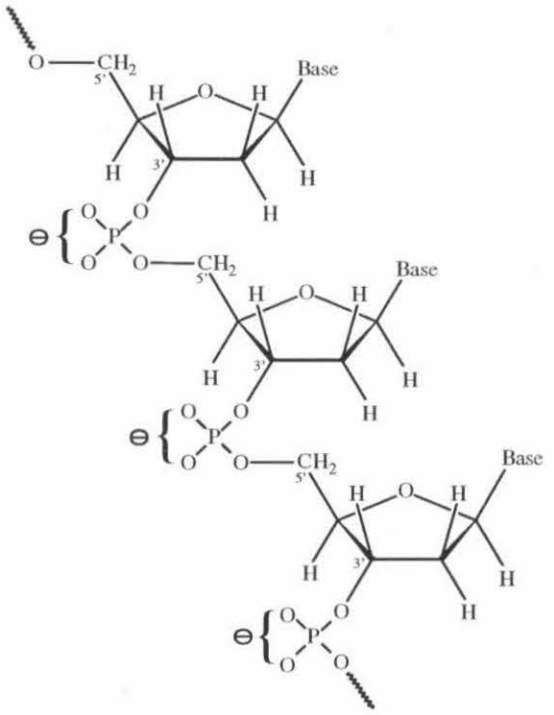


Figure 2.2: Backbone structure of DNA. Note that, under alkaline conditions, each base is associated with one negative charge on the phosphate group.

deoxynucleotide triphosphates releases energy. The reaction is, of course, reversible, and an excess of pyrophosphate will, in fact, cause the polymerase to catalyse the stepwise destruction of the primer [Tabor and Richardson 1990].

Under optimal conditions, one polymerase molecule can extend a primer at the rate of > 300 deoxynucleotides per second [Tabor and Richardson 1987a]. The polymerase molecule can, and does, periodically fall off the primer-template complex, allowing another (or the same) polymerase molecule to attach itself and continue the process. The mean number of deoxynucleotides a single polymerase molecule will extend a primer before falling off is known as its "processivity." Different polymerases have processivities from < 10 to $> 10,000$ [Tabor et al. 1987].

2.2 Sanger Dideoxy Sequencing Reactions

In 1977 Sanger [Sanger et al. 1977] demonstrated an elegant method of sequencing DNA based on polymerase chemistry. The technique relies on selectively terminating primer extension with dideoxynucleotides. Dideoxynucleotides are exactly like normal deoxynucleotides, except that they lack the 3′-OH group essential to chain extension (Figure 2.1). Hence, if a dideoxynucleotide is incorporated into a growing primer, it will effectively block any further growth.

If a reaction mixture is prepared consisting of a template- primer complex, sufficient quantities of the 4 deoxynucleotide triphosphates, and one percent of, say, dideoxythymidine triphosphate (ddT) then, assuming the polymerase doesn't distinguish between dT and ddT (often an invalid assumption, it turns out), every A (A being complementary to T) in the template will have a one percent chance of terminating the extension process. The products of this reaction (the "T" reaction) will be dominated by segments of DNA ending in ddT, and complementary to the initial sequence of the template (Figure 2.3). If these short pieces of DNA are now

separated and their sizes measured (in units of deoxynucleotides), then we know the relative positions of all the A's in the original template. If this procedure is repeated for the three remaining deoxynucleotides ("A", "G" and "C" reactions), we can then determine the complete sequence of the the original template.

Even ignoring the problems of measuring the fragment sizes, there are several mechanisms by which this procedure can yield incorrect, or difficult to interpret results:

1. **Straightforward errors in translation.** Polymerases are not perfect and may incorporate the wrong deoxynucleotide, which, if it happens to be a dideoxynucleotide, will terminate the chain at the wrong length. For example, if in the "T" reaction, a polymerase molecule falls off the primer-template complex with the primer terminating in a G, this will introduce a "T" signal where there should have been none.

2. **Processivity problems.** If the polymerase molecule falls off in the middle of extending the primer, and neither it nor another polymerase molecule takes up where it left off, then the fragment may terminate at an inappropriate position.

3. **Template secondary structure.** If the template DNA contains two short complementary stretches near each other, it can loop back on itself and form a short stretch of double-stranded DNA. This loop will inhibit the action of the polymerase, increasing the chance of it falling off at the wrong place and mis-terminating the primer.

4. **Sequence specific variations in dideoxy incorporation ratios.** As noted above, polymerase molecules are not completely oblivious to the differences between deoxynucleotides and dideoxynucleotides. The probability that a polymerase will incorporate a dideoxynucleotide is dependent on the relative proba-

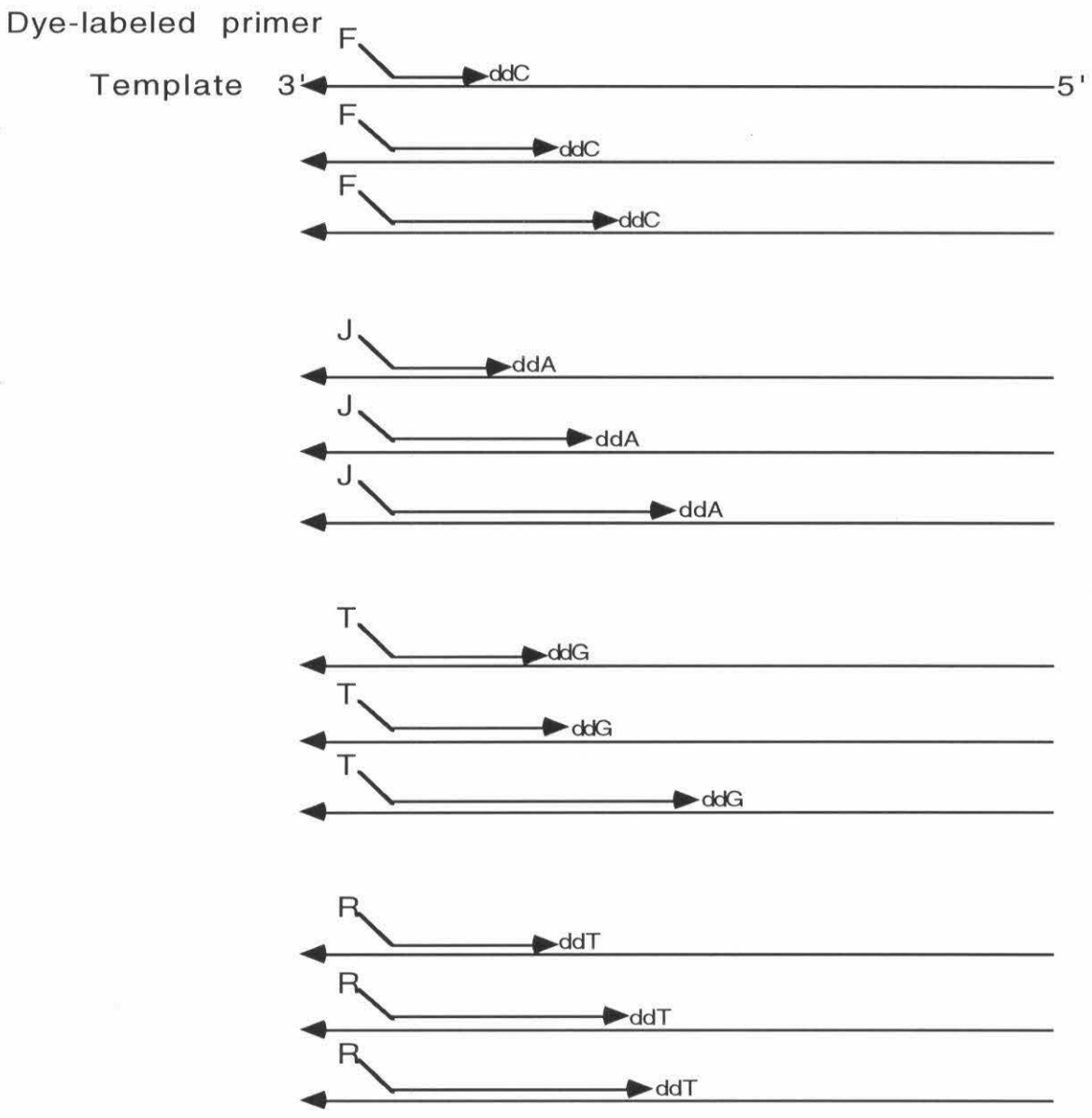


Figure 2.3: Schematic diagram of the four sequencing reactions. Each primer is labeled with one of the four dyes, FAM, JOE, TAMRA or ROX. These dyes are used in conjuction with ddC, ddA, ddG and ddT respectively. The primers are extended by the polymerase until it happens to incorporate a dideoxynucleotide, which terminates the chain. The labeled products are then sorted by size to get the sequence of the template. Arrows point in the $5' \rightarrow 3'$ direction.

bilities that the polymerase encounters a deoxynucleotide or a dideoxynucleotide at the proper moment, and the discrimination the polymerase has between the two. The first is dependent solely on the relative concentrations of the two in the reaction mixture, while the second has two components, global and sequence specific. A global discrimination between deoxynucleotides and dideoxynucleotides can be balanced by adjusting the relative concentrations of the two, while a sequence specific discrimination shows up as a local variation in the number of molecules of each size fragment. This variation is often observed to be repeatable given the same template sequence and, in some cases, can be large enough to turn what would otherwise be an unambiguous signal into noise.

2.3 Gel Electrophoresis

The second part of this sequencing method, separating and measuring the length of all the fragments, is accomplished by gel electrophoresis. DNA molecules, by virtue of their phosphate linkages, carry a charge and will thus move under the influence of an electric field. If one puts a sample of DNA at one end of an aqueous gel and applies an appropriately oriented electric field, the DNA will drift along the length of the gel. Smaller molecules generally drift faster than small ones, hence a mixture will be separated into distinct bands on the gel consisting of molecules of identical size. Various methods exist for imaging the bands and thus determining the relative sizes of their constituent molecules.

The rate at which a particular molecule will drift depends on the local electric field, the charge carried by that molecule, and how easy it is for the molecule to find its way through the microscopic channels in the gel. Ideally, one would hope that the drift velocity of a DNA molecule would be strictly proportional to its length in bases, thus simplifying the measurement of its size. Both the electric field and the

charge on DNA molecules behave rather well. It is not hard to arrange a uniform and constant electric field across the gel, and the charge on a DNA molecule (at the appropriate pH) is proportional to its length in bases independent of the specific base composition.

For most sequences, a DNA fragment N bases long will migrate slower than a fragment $N - 1$ bases long and faster than a fragment $N + 1$ bases long. This results in fragments ending up in "bands" oriented perpendicular to the electric field with the contents of each band being fragments of a single length. The distance between bands (in the direction parallel to the electric field) is a measure of the difference in size between fragments in the two bands. There are other sequence-specific effects, to be discussed below, that interfere with this monotonic relationship between fragment size and mobility, but these are rare enough to be treated as special cases later. In experienced hands, gel electrophoresis is capable of resolving a fragment 999 bases long from one 1000 bases long. This sort of performance is often thwarted by temperature gradients along the width of the gel, which causes the bands to tilt and curve. Bubbles in the gel also warp the bands. Later on I will discuss how to mitigate both these effects.

2.4 Detection

Sanger's original experiments were designed to yield DNA fragments that incorporated radioactive phosphorus (^{32}P). The results of the four reactions were then separated by gel electrophoresis in adjacent lanes on a gel. By placing the gel against a piece of photographic film, which turns black where the electrons emitted from the ^{32}P hit it, it is possible to visualize the position of each band. By studying the relative positions of the bands in the four lanes, it is possible to determine the original template DNA sequence.

This procedure has 2 main problems. One is the presence of radioactivity, and the other is the fact that the fragments in the four lanes do not migrate at exactly the same rate. The latter effect produces difficulties in aligning fragment bands across all four lanes. The solution to both these problems is the use of fluorescently labeled primers. The original four reactions are carried out as before, except that each reaction uses a primer labeled with a fluorescent dye, a different dye for each of the four reactions. Radioactivity is no longer necessary, as detection is done by detecting the fluorescence of the primers. The gel electrophoresis is now made self-aligning by mixing the products of the four reactions and separating them out in a single lane. The different colors emanating from the four different dyes are used to distinguish between products from the four reactions. This also makes it possible to run four times as many samples on a given gel.

A further refinement is real-time detection. Instead of running the electrophoresis for a fixed time and examining the gel afterwards, it is possible to scan across the gel (perpendicular to the electric field) with a laser focused to a small spot. The fluorescence is then detected as the fragments migrate across the line scanned by the laser. This is the principle behind the Applied Biosystems (ABI) model 373 DNA sequencer from which all the primary data for this thesis was generated.

It is important to keep in mind that this real-time detection scheme records the amount of time it took a fragment to migrate a fixed distance, whereas the original radioactive-photographic film method records the distance a fragment migrates in a fixed time.

Chapter 3

Technologies

3.1 Sequenase

A successful sequencing reaction requires that the growing chain terminate by the incorporation of a dideoxynucleotide, and that it remain intact through the electrophoresis. Chains that terminate at random or that are cut back after termination will lead to errors in interpreting the final sequence, e.g., by producing primer strands in the T reaction that end in G. Thus it is important that the enzyme both finish the job by incorporating nucleotides until it incorporates a dideoxynucleotide, and that once the primer strand is thus terminated, that it not be subsequently modified.

Most DNA polymerases have a $3' \rightarrow 5'$ exonuclease activity. This removes nucleotides from the end of the growing chain and is thought to be part of a "proofreading" process *in vivo*. This exonuclease activity is not simply the reverse of the polymerase reaction as it does not require pyrophosphate and is usually associated with a different part of the enzyme. Such exonuclease activity will result in cutting back an already terminated primer strand. This will result in spurious peaks in the final data.

Different polymerases also have different processivities. An enzyme with a low processivity has a higher chance of releasing the primer-template complex before incorporating a chain-terminating dideoxynucleotide. If this happens, the primer strand has a good chance of ending in the wrong nucleotide.

The first enzyme used for sequencing was the large fragment of *E. Coli* DNA polymerase ("Klenow"). Klenow has the disadvantage of having large sequence-dependent variations in dideoxynucleotide incorporation which makes interpretation of the data difficult [Sanders et al. 1989]. In the late 1980's, Tabor and Richardson created a much improved enzyme system for sequencing by selectively oxidizing away the $3' \rightarrow 5'$ exonuclease activity of T7 polymerase [Tabor and Richardson 1987b]. It was also discovered that the T7 polymerase binds tightly (5×10^{-9}M) to *E. Coli* thioredoxin in a one-to-one stoichiometry. The thioredoxin-polymerase complex has a greater processivity than the polymerase itself [Tabor et al. 1987] [Huber et al. 1987]. The use of manganese instead of magnesium ions in the reaction mix lowers the discrimination of T7 polymerase against dideoxynucleotides to the point where deoxynucleotides and dideoxynucleotides are incorporated at essentially the same rate [Tabor and Richardson 1989]. The modified T7 polymerase-thioredoxin complex is marketed under the trade name "Sequenase." All the sequencing reactions that generated the data for this work used Sequenase with manganese as the metal ion.

Although there is presently no crystal structure for T7 polymerase, there is a crystal structure for Klenow [Ollis et al. 1985] which reveals that Klenow is about as long as 20 DNA bases. There is a strong homology between the DNA sequences encoding T7 and Klenow [Himawan and Richardson 1992], hence I assume T7 polymerase is also about 20 bases long. This gives an upper bound on how far away from the incorporation site one might reasonably expect sequence context to have any effect given a template without significant secondary structure.

3.2 The ABI 373 Sequencer

The ABI 373 sequencer is the commercial version of a machine described by Smith et al. [Smith et al. 1986]. It both runs the electrophoresis and detects the fluorescently labeled DNA as it migrates down the gel. The gel, sandwiched between two glass plates, is held vertically against a thermostated metal plate. Both top and bottom of the gel are immersed in an aqueous buffer solution. This forms the electrical contacts for the electrophoresis. Approximately 25 cm below the top of the gel a slot is cut into the metal plate. Through this slot a movable optical assembly both illuminates and looks at a point on the gel.

The ABI 373 uses a laser focused down to a 300 micron spot which, as the optical assembly moves back and forth, sweeps across the width of the gel in 1.0 second. It then takes 0.50 second to turn around before sweeping back to the other edge of the gel. Mounted along with the laser optics is a photomultiplier tube focused on the volume element illuminated by the laser. A filter wheel with four filters is placed in the optical path between the photomultiplier tube and the gel. The filter wheel is advanced to the next position every time the optical assembly reverses direction. It thus takes 6.0 seconds (four sweeps across the gel, each taking a total of 1.5 seconds) for a complete 4-color scan of that 300 micron line across the gel.

The signal from the photomultiplier is amplified and fed to an integrator. The signal, as the optical assembly is still moving, is integrated for 4.2 ms, then fed to a 12-bit analog-to-digital converter. During the next 1.0 ms, the integrator is re-zeroed. There are thus 194 samples in the 1.0 second it takes to sweep across the gel. The following data are recorded for each sweep: the 194 digitized samples, which filter was in place during that sweep, the temperature of the gel and the voltage and current of the electrophoresis. A typical experiment lasts for about 12 hours, yielding almost 29,000 sweeps. This is what I call the primary, or image data. It is essentially a

4-color image of the fluorescence of the gel, with one axis of the image being time.

3.3 Running A Gel

The electrophoresis gel is cast between two glass plates held 400 microns apart by plastic spacers along their right and left edges. The reaction products are then loaded into "wells" formed by the glass plates and a plastic "comb." Once the DNA enters the gel due to the electric field, there is no mechanical barrier between different DNA samples. Nor are the different DNA samples mechanically registered with the pixels generated by the scanning optics. There is some play in the horizontal position of the comb, hence the lanes are never in the same position on the gel from run to run. The different DNA samples migrate reasonably parallel to one another down the gel, in "lanes"; there is rarely mixing between the samples. But, lanes do not run exactly straight down the gel. Small ions in the reaction mixture create a focusing effect that makes the lanes thinner at the beginning of a run than they are at the end. This focusing effect can also bend adjacent lanes towards each other at the beginning of a run.

Because the lanes move about and are not in the same absolute positions from run to run, the software that analyses the gel images must figure out, just by examining the image, where the edges of the lanes are and how they move around as the run progresses.

The reaction products are dissolved in formamide, which is denser than the buffer solution. Thus the reaction products sink down onto the surface of the gel. The surface of the gel often has a bump in the middle of a well due to the teeth of the comb pushing down the edges. The dense DNA sample therefore collects more in the edges of the well and less in the middle. This shows up in the image as an increased brightness at the edges of the lanes.

Chapter 4

Generating the Data; A Sequencing Project

4.1 Cosmids and M13

Since it is not yet possible to directly sequence entire chromosomes all at once, the DNA is divided up in several stages. I will describe the last couple stages, as they relate to the interpretation of the final data.

We start at the point where the DNA of interest has been cloned into cosmids in the bacterium *E. Coli.* Cosmids are autonomously replicating, extrachromosomal, circular DNA that are engineered to allow easy insertion of foreign DNA. The bare cosmid used in these experiments is 8213 bases long [Seto et al. 1992]. 35-40Kb of foreign DNA can be inserted into a cosmid, which is then placed back into an *E. Coli* where it replicates along with the *E. Coli.* A cosmid is still far too big to sequence directly, so it is divided down again in a process known as "shotgunning." The cosmid DNA from a large number of identical *E. Coli* is purified, dissolved and subjected to high amplitude ultrasonic sound waves. This mechanically shears the cosmid DNA

in random places. As the process continues, the DNA gets sheared into smaller and smaller pieces. These small pieces are then inserted into the genome of a small virus known as M13.

M13 is a virus with an interesting genome. Only one strand (the (+)strand), of the viral DNA exists in the virus particle itself. When the virus infects its host bacterium, the host's DNA replication mechanism forms the other (-)strand. In the bacterium there exists a double-stranded, circular, replicative-form (RF) of the viral DNA. When the time comes to produce more viral particles, the (+)strand of the RF DNA dissociates, is covered with a virus-encoded coat protein, and is ejected from the cell. The (-)strand of viral DNA is then used, by the host's DNA replication mechanisms, as a template to produce a copy of the (+)strand. It is possible to isolate single-stranded M13 DNA from the virus particles themselves, or the double-stranded RF DNA from infected *E. Coli.* This is helpful because splicing DNA is done with double-stranded DNA while the sequencing reactions prefer single-stranded templates.

A variant of M13, known as M13mp18 has been engineered with the recognition site for the SmaI restriction endonuclease. SmaI recognizes the hexanucleotide $5' - CCCGGG - 3'$ of double-stranded DNA and cuts both strands between the C and G in the middle. M13mp18 contains exactly one occurrence of $5' - CCCGGG - 3'$ in its circular genome, hence when the RF DNA is cut by SmaI, it results in a linear piece of double-stranded DNA with blunt ends.

The small bits of double stranded cosmid DNA are then ligated onto the linearized M13 DNA which is then recircularized and transfected into *E. Coli.* Monoclonal colonies of *E. Coli* infected with these recombinant M13 are then grown up and the resulting single-stranded M13 DNA is recovered from viral particles.

In terms of sequencing, this subcloning into M13 serves two purposes. It produces single stranded DNA for the sequencing reactions and, by subcloning into a

well-defined position in the M13 DNA, it allows one to use a common primer, complementary to a section just 5′ of the SmaI site, for all sequencing reactions. The one drawback with this technique is that it is completely random (hence the term "shotgunning"). The sonication treatment cuts the cosmid DNA at random positions. Also, since the small bits of cosmid DNA can ligate onto the M13 DNA in either orientation, either strand can end up on the (+)strand of the M13.

All this is sorted out in the end by computer. After 700-1000 M13 clones have been sequenced (yielding $\approx$ 400 bases each), the whole cosmid, including the native cosmid sequence, is reconstructed from the pattern of overlaps, much like a jigsaw puzzle. Each base of the cosmid ends up being sequenced several times, at different positions from the primer and in both orientations.

4.2 Cosmid 2-47

All the data used in this thesis came from one cosmid that was sequenced as part of a project to sequence part of the murine T-cell receptor locus; part of the immune system. This cosmid is referred to as "2-47" and its 34,476 base insert has Genbank accession number M94080. It was mostly sequenced between January and May of 1992 by Don Seto, et al., using four ABI 373 sequencers at Caltech.

The sequencing effort went on independently of my work. The original image files from the sequencers were compressed and saved on magnetic tape for me to re-analyze later.

The detailed protocol for the sequencing reactions is set out in [Koop et al. 1990]. The enzyme used was Sequenase, Version 1.0 (USB) with the buffer containing manganese instead of magnesium. The primers used (-21M13) had sequence 5′ – TGTAAAACGACGGCCAGT – 3′ which ends 41 bases before the SmaI site. The four dyes on the primers are known as FAM, JOE, TAMRA and ROX, they are used

in conjunction with dideoxy- C, A, G and T respectively.

Chapter 5

Image Processing

The raw data from the sequencing experiments consists of the digitized output of the photomultiplier tube as it scans back and forth across the gel. The gel contains the information from up to 24 DNA fragments. There is no mechanical separation between the fragments, they have to be separated on the basis of information within the image itself. There are also experimental artifacts related to the electrophoresis that are irrelevant to the study of the enzyme that have to be factored out of the image. This is a multi-step process along the following lines:

1. Common time re-sampling.
2. Gel straightening.
3. Lane finding.
4. Horizontal averaging.

5.1 Common-Time Resampling

Since the sequencer does a complete sweep across the gel with one filter before switching filters for the next sweep, the data does not align temporally across colors for any particular pixel. It will be important later, when the four-color data for each pixel will be transformed to four-dye data, that the different colors be temporally aligned. To do this a simple resampling scheme is employed.

For each group of 4 sweeps, a reference time is created (the beginning of the third sweep). For each pixel, in each color, the 2 values before the reference time plus the value after the reference time are fit to a quadratic function. The value of this quadratic at the reference time is used as the value of that pixel in that color for this four-color scan (see Figure 5.1). This effectively resamples all the data in a four-color scan at the common reference time. Because the peaks in this data will be of interest later on, quadratic interpolation was chosen as the lowest order interpolation that could reproduce a peak.

5.2 Gel Straightening

What we are after in the end is the integrated signal of each individual band. Somehow we have to figure out the boundaries of each band so we know what to integrate over. One way to do this would be to directly try and find the boundary of each band by various image processing techniques, such as computing various spatial derivatives and looking for zero-crossings. This approach works to some extent, but ignores some of the inherent structure in these particular images that can provide clues as to what the real band boundaries are. I chose instead, to warp the gel image so as to straighten the bands then find the left and right edges of the lanes. The bands are now rectangular areas with known left and right boundaries. Finding the vertical

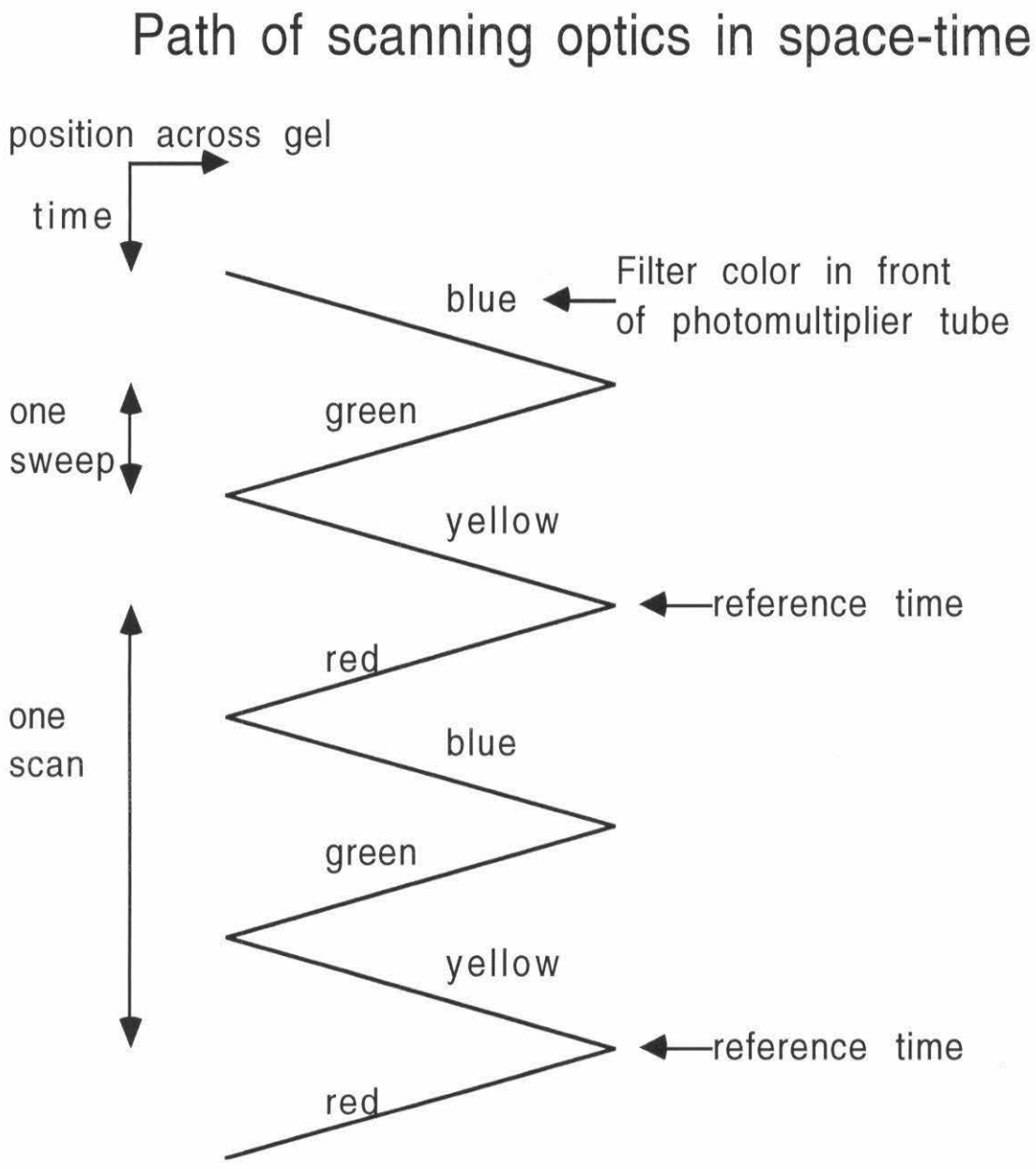


Figure 5.1: Space-Time diagram of gel scanning. Quadratic interpolation is used to resample all the data to make it appear as if all the data for a single scan had been sampled at the reference time.

(i.e., temporal) limits is deferred to a later stage.

The first signals to appear in a gel image are the "primer peaks." These are due to excess dye-modified primer left over from the reactions. Being the shortest molecules in the mix, they come down first. They are also present in substantial excess over the template, so there are a lot of them left over. This high concentration of primer produces a broad, bright peak at about 600 scans (1 hour) from the beginning of the run. I detect the primer peaks as a group across the gel by just integrating the total signal over a rectangular window spanning the width of the gel and 50 scans. The presence of a sharp decrease in this value is taken as the trailing edge of the primer peak.

Directly after the primer peaks is a region where the lanes can be rather curved due to the focusing effects of small ions. I ignore this region, which I define as an empirically determined number of scans after the falling edge of the primer peak, until later.

Observe that the bands just after this region are fairly straight and get increasingly more warped as the run progresses. Instead of trying to figure out the shape of each band in isolation, I exploit this continuity of deformation and deal with several bands at once. I divide the gel into non-overlapping "chunks" 50 scans high starting with chunk 0 at the beginning of the straight region and working up (i.e., later in time). Imagine the image decomposed into thousands of segments, 50 scans high and one pixel wide. The process of straightening the bands consists of displacing these segments vertically to maximize the correlation between segments.

Mathematically, the correlation computation goes as follows. Consider each color individually for the moment. Define x_i and y_i, where i is the scan number ranging over 50 values, as the pixel values of adjacent segments in one particular color. Compute

$X_i = x_i - M_x$ where M_x is the mean of the x_i. Likewise for Y_i. Define

$$C_{x,y} = \frac{\sum(X_i Y_i)}{\sqrt{\sum X_i^2}\sqrt{\sum Y_i^2}}.$$

This is a dimensionless quantity that can vary from 1, when $x_i = y_i$ to -1 when $x_i = -y_i$. This quantity is independent of the relative scaling or offsets between x_i and y_i. I define the correlation between two segments as the sum of the $C_{x,y}$ taken over the 4 colors.

The segment displacements are determined first for chunk 0. Starting from the left edge, the correlations are computed between adjacent segments for a range of displacements of the right-hand segment. The right-hand segment is then displaced by the amount that gives the maximum correlation. The algorithm then shifts over one segment to the right and repeats the process using the newly displaced segment as the left hand segment of the pair. The displacements calculated for one chunk are then used as the starting point for the next chunk. This minimizes the range of displacements that must be evaluated during the maximization procedure hence improving the execution speed of the algorithm and minimizing the chance of falling into spurious band alignments.

The left-hand most pixel column in the image is left untouched by this procedure. As the alignment procedure crosses a lane boundary, it is going to generate essentially random displacements. These displacements accumulate across the gel giving rise to a global warping, amounting, in some cases, to hundreds of scan lines from one side of the gel to the other. Later on, when we have found the lane boundaries, I correct for this global warping by resetting the left most pixel column in each lane to have zero absolute displacement (i.e., unchanged from the original image), while preserving the relative displacements within a lane.

The image is assumed to be straight below chunk 0. In practice, the narrowing and

curving of the lanes as one proceeds down from the chunk 0 confuses the straightening algorithm, so that region is just left alone.

Once all the displacements have been calculated, they are interpolated between chunks to produce a smooth deformation map of the image. This map takes the form of lines running horizontally across the gel that I call "warp contours." If the image is deformed so that the warp contours form straight, horizontal lines, then the bands will be straight.

5.3 Lane Finding

After the gel is straightened, the bands now form horizontal rectangles. The purpose of the lane finding procedure is to find the left and right boundaries of the bands. Observe that all the bands lie in lanes with more-or-less straight vertical boundaries. Lanes often abut one another as the run progresses, thus any good algorithm for finding the edges of the lanes will have to take into account the actual correlations in the data and not rely on there being any dark space between the lanes. (This was a real problem with trying to find individual bands one at a time. The algorithm failed whenever two lanes abutted and had the same color band at the same position. There was no way of knowing this was really two bands.)

Beyond chunk 0, lanes tend to be 6 or 7 pixels wide, may abut one another and may drift around at the rate of about 1 pixel per 1000 scans. Going down from chunk 0, the lanes tend to get narrower (down to 2 or 3 pixels wide), have blank spaces between them and can curve quite substantially (1 pixel per 50 scans).

Lane finding starts by computing correlations between segments of the straightened gel. The computations are identical to those used for straightening the gel and are performed between adjacent segments and segments separated by 1 or 2 pixels. Auto-correlation computations are also done to give an offset-independent measure

of signal strength.

Just as the gel-straightening algorithm starts with the best data, the lane-finding algorithm starts at chunk 19 and works both directions from there. The first step is to identify the center of each lane at chunk 19, find the edges of each lane, then follow the edges up and down the gel.

To find the center of each lane, I compute a quantity for each segment I call the "center strength." This is just the average of the correlations between a segment and its six neighbors (3 on each side). Only segments near the center of a lane will have a high center strength. I average the center strengths over a window 20 chunks high centered at chunk 19. Scanning across the gel, I look for local maxima in the averaged center strengths with a value ≥ 0.6. These points are recorded as the lane centers. This procedure misses any lanes that have died out by lane 19, but I consider those too short to be worth worrying about.

I also compute a "signal strength," which is just the auto-correlation of a segment divided by the lowest auto-correlation of any segment in the same chunk, minus 1.0. This is a non-negative quantity which is zero on segments with no signal (there are always such segments at one side of the gel) and is a measure of the signal strength of that segment.

To find the edges of the lanes I compute two quantities, the "right strength" and the "left strength" analogous to the center strength. The left strength of a segment is the average of that segment's correlations with its three neighbors to the right. The right strength is defined similarly. Right, left and signal strengths are also averaged over a 20 chunk window centered on the chunk in question. Given the center (or at least an interior) point of a lane, the left edge is found by scanning left from the center of the lane until one of three things occurs:

1. The left strength of the segment is < 0.55. That segment is taken to be just

outside the lane.

2. The center strength hits a local minimum. That segment is taken to be the left edge of the lane.

3. The left strength times the signal strength is less than 3.0 AND less than 0.3 times the average signal strength of all the segments already known to be within the lane. That segment is taken to be just outside the lane.

The first condition dictates a minimum correlation between a segment and its neighbors to the right in a lane. The second condition says that a segment minimally correlated with its neighbors must on an edge or between lanes. The third condition dictates that a segment must make a certain minimal contribution to the signal in a lane.

The right-hand edge is found similarly.

The edges are extended up the gel similarly except that the edges are constrained to be within one pixel of where they were in the last chunk, and the values of the minimum left strength and minimum relative signal strength are relaxed somewhat to 0.5 and 0.25, respectively. The edges for the last 10 chunks are just extended straight from where they were 10 chunks from the end.

To extend the edges down the gel, certain modifications are necessary due to the tendency of the lanes to curve more. Going down from chunk 19 to chunk 1, the algorithm is the same, except that instead of averaging the various strengths over 20 chunks, they are averaged over only 9 chunks. Below chunk 1, the tendency to curve becomes even more pronounced with the lanes becoming narrower with blank spaces between them. From chunk 0 to chunk -6, the lane width is fixed at 3 pixels with the lane center being determined solely by local maxima in the signal strength. The lanes are also allowed to drift without constraint. Below chunk -6, the lane edges are just extended straight down.

A consistency check is then done on all the lane boundaries to make sure lanes don't overlap. If they do, the boundaries are arbitrarily re-adjusted and a warning message is printed. This doesn't happen often, and when it does it's usually in the first few chunks where the lanes are curving sharply.

5.4 Horizontal Averaging

Once the lanes have been found, the warp contours are adjusted to put the left-hand-most pixel columns in each lane as they were in the original image, while preserving the appropriate warping between segments within a lane. This gives the minimum distortion of the gel consistent with straightening the bands.

Now that we have the gel straightened and the lanes found, we can simplify the problem by noting that the intensity variations across a lane are purely experimental artifacts; they tell us nothing about the properties of the enzyme. So, I reduce the two-dimensional image to several one-dimensional data sets by averaging across the lanes, along the warp contours. Each lane thus gives rise to a separate "fragment file." Each fragment file contains 4 graphs, one for each color, of the averaged intensity versus scan number. I call these 4 graphs the "raw data." Included in the fragment files are a wealth of ancillary information, such as the original lane boundaries, the warp contours, book-keeping information about the DNA fragment, which machine the gel was run on, etc.

5.5 Comparison with ABI Software

After going through all the trouble of straightening the image and finding the actual edges of the lanes, the question is: does this really improve the data? For good, clean, straight image data the difference is small. Data which suffers from gross band

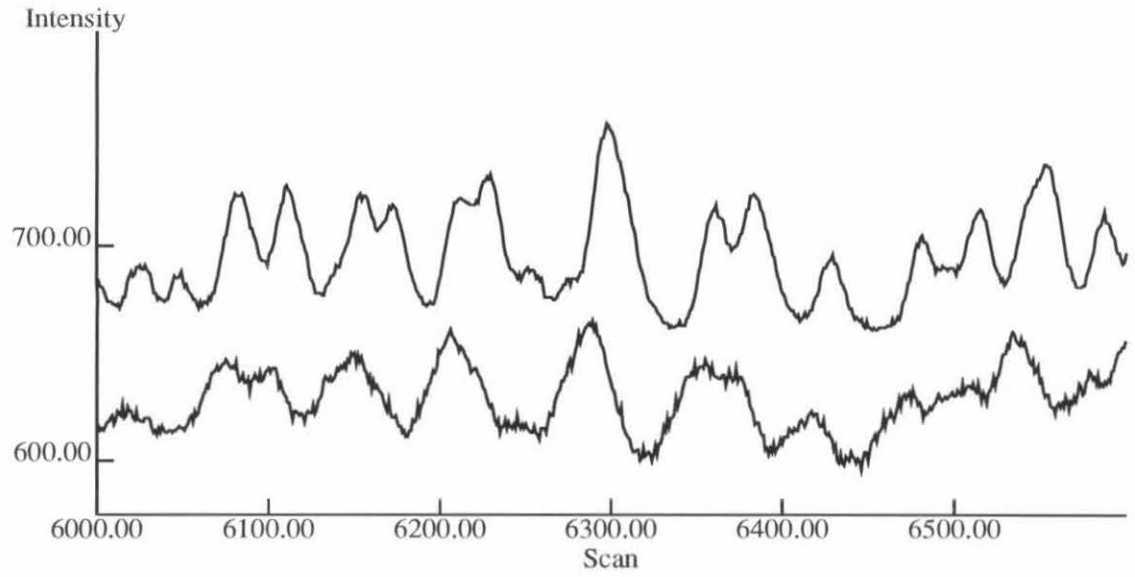


Figure 5.2: Comparison of data extracted by my software (upper trace) and ABI's Data Collection software (Version 1.0.1) (lower trace). The traces have been vertically offset for clarity. The upper trace comes out shifted to the right with respect to the lower trace because I reference the band straightening to the left edge of the lane, while the ABI software averages across the middle 3 pixels. This data is in filter-space (filter 0, blue) from scans 6000-6600 of the eighth lane from the left of the image in Figure 5.3 (pixels 61-67).

---

deformation is improved markedly by all this processing. Figure 5.2 shows a side-by-side comparison of the graphs extracted by the ABI Data Collection software and my software.

One would expect peaks to be better resolved after straightening because averaging horizontally across warped bands combines data belonging to adjacent bands. The ABI software only averages the middle (or what it thinks are the middle) three pixels of a band. If it averaged across more pixels, it would smear the data even more. My software averages across the entire lane, usually $6-7$ pixels. One would expect the signal/noise ratio to improve by a factor of $\sqrt{2}$ by virtue of averaging across twice as

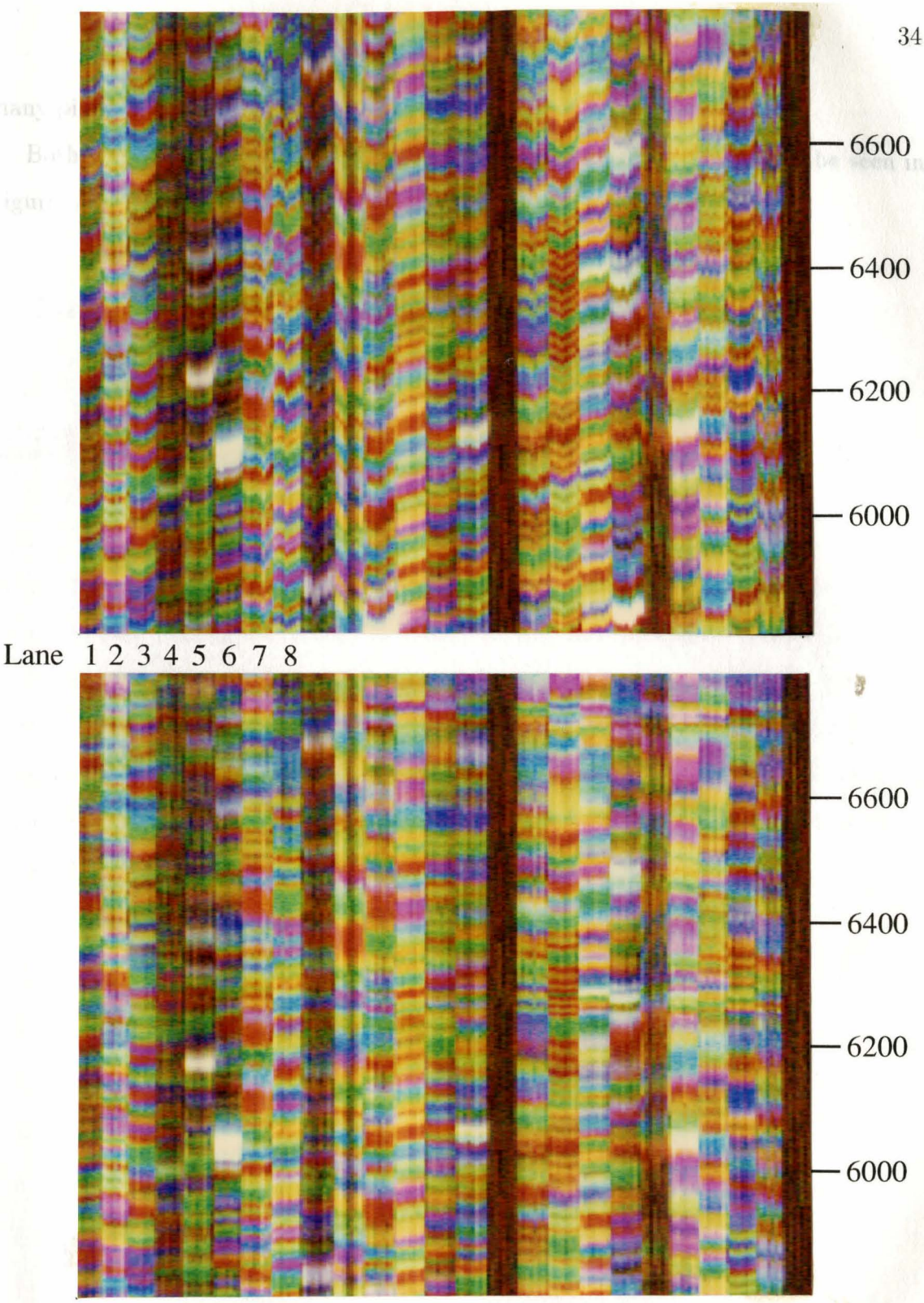


Figure 5.3: Gel images showing the effect of band straightening. Top photo is raw data, bottom is after straightening. Both photos represent scans 5800 (bottom) to 6800 (top). See also Figure 5.2.

many pixels.

Both these improvements, better peak resolution and lower noise, can be seen in Figure 5.2.

Chapter 6

Extracting Incorporation Ratios

Once the image has been reduced to a number of fragment files, each representing a separate DNA fragment, it is then necessary to find, interpret and measure the peaks on the 4-color graphs. The peak heights are normalized against their neighbors to produce a dimensionless measure of dideoxynucleotide incorporation. Once a base assignment has been made to each peak, the resulting sequence is then aligned to the known sequence of that cosmid to detect errors and confirm the base calling. The process follows the following outline:

1. Dye-space transformation.
2. Baseline subtraction.
3. Mobility-shift corrections.
4. Base calling.
5. Peak height normalization.
6. Consensus alignment.
7. Quantitating peaks.

Functions 4,5 and 6 are applied iteratively as there are, in fact, three base calling algorithms each of which uses information derived from the previous base callings to completely re-call the bases.

6.1 Dye-Space Transformation

The data in the fragment file represents the signal from the photomultiplier tube looking through a colored filter at the gel. Since the transmission spectra of each filter overlaps the fluorescence spectra of each of the 4 dyes, the 4 signals each represent a linear combination of the fluorescence from the dyes. If one considers the 4 color signals at each pixel as the components of a 4-dimensional vector (in "filter-space"), and the fluorescence intensities of the 4 dyes as another vector (in "dye-space"), then one can convert between the two with a linear transformation. This transformation varies a little bit between machines due to variations in the exact transmission spectra of the filters. The manufacturer supplies the components of the transformation matrix for each individual machine.

6.2 Baseline Subtraction

In addition to the fluorescence from the four dyes, there is also a background emission from the gel material itself and scattering of the laser that contribute to the detected signal. This gives more than four sources of light from the experiment (the four dyes plus various backgrounds). Unfortunately, the sequencer only has four filters. Hence, it is impossible to sort out the various emissions by just applying a linear transformation to the four components of each pixel. Since the various background emissions are fairly constant on the time scale of a few bands, one is tempted to estimate the background by some sort of spectral technique.

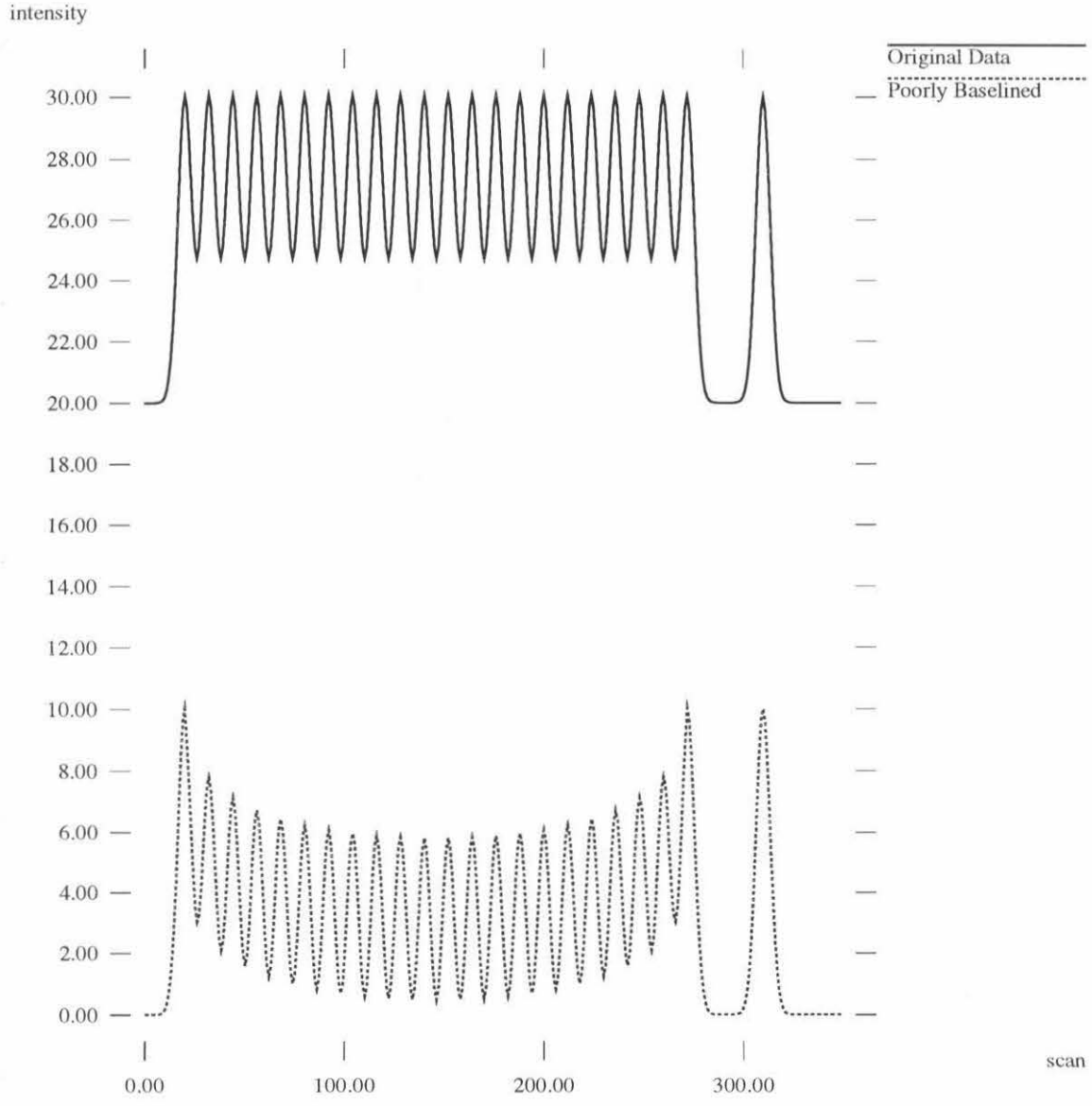


Figure 6.1: Removing the baseline by subtracting a low-pass filtered version of local minima results in peaks in the middle of a run of identical bases being anomalously low. This type of artifact is seen in data processed by ABI's base-calling software.

One must be careful with the details in order to avoid artifacts that will be important later on. An obvious, but poor way to determine the background would be to locate all the local minima, apply a wide smoothing filter and use that as the baseline. This leads to the artifact (seen in the graphs generated by the ABI software) shown in Figure 6.1, where the middle bases of a run of identical bases are lower than they should be. This artifact probably has no effect on the accuracy of base calling, but it does obscure the relevant enzyme effect.

The problem with any pure spectral technique is that the signal and background both have a DC component. Given that no pixel-by-pixel transformation will work, and the spectra of the signal and background overlap, some other property of the signal must be used to separate it out. That property is that the signal is always non-negative. That and the weak spectral overlap between signal and background at low frequencies led me to the simple algorithm that works well.

The background is taken to be the minimum, over a sliding window 200 scans wide (100 in each direction). 200 scans is longer than any run of identical bases in the data set, but is still short enough to capture the slow variation in the baseline. This background subtraction is carried out independently for each color, after transforming the data to dye-space.

This algorithm is unsatisfying in that it relies on the hypothesis that the fluorescence due to each dye falls to exactly zero at least once every 200 scans. Without the means to independently confirm this, I consider baselining to be the weak link in the signal processing part of this work. Anyone planning to repeat these experiments would be well advised to make measurements at enough different points in the spectrum to unambiguously resolve all the sources of fluorescence and backscatter.

6.3 Mobility-Shift Corrections

The mobility of DNA is not determined completely by the number of bases it contains, there is a dependence on the exact sequence involved. To a large extent this sequence dependence can be ignored because we are comparing mobilities of molecules that are identical up to the point where one is longer than the other. Two adjacent peaks represent identical molecules except that one has one extra base. The separation of peaks is determined by the change in mobility due to the addition of that extra base. This change in mobility is dependent, in part, on the identity of that last base. If the average (over the four bases) peak separation is normalized to 1.0, then this effect changes the peak separation from ≈ 0.84 for C to ≈ 1.17 for T [Bowling et al. 1991]. The presence of the dye moiety on the 5′ end of the DNA molecules also changes the mobility of the molecule. This mobility shift is not constant as the length of the molecule varies. Because the identity of the dye is correlated with the identity of the last base, both these effects are compensated for at the same time by shifting the four graphs (in dye-space) with respect to one another. These mobility shift corrections are empirically determined and expressed as the number of scans to shift the data given the position in scans (Figure 6.2).

This is crude in the sense that it does not take into account the variations in running conditions for different gels, but perfect correction for these effects is not important.

There are at least two other effects that modify peak-to-peak spacing. One is the identity of the penultimate base; this is always a small effect [Bowling et al. 1991]. Another is secondary structure ("hairpins") forming at the 3′ end of the molecule. This effect can be substantial; even to the point of reversing the proper order of peaks. This is the second most important reason for mis-calling bases, but correcting for it would require an *a priori* knowledge of the sequence.

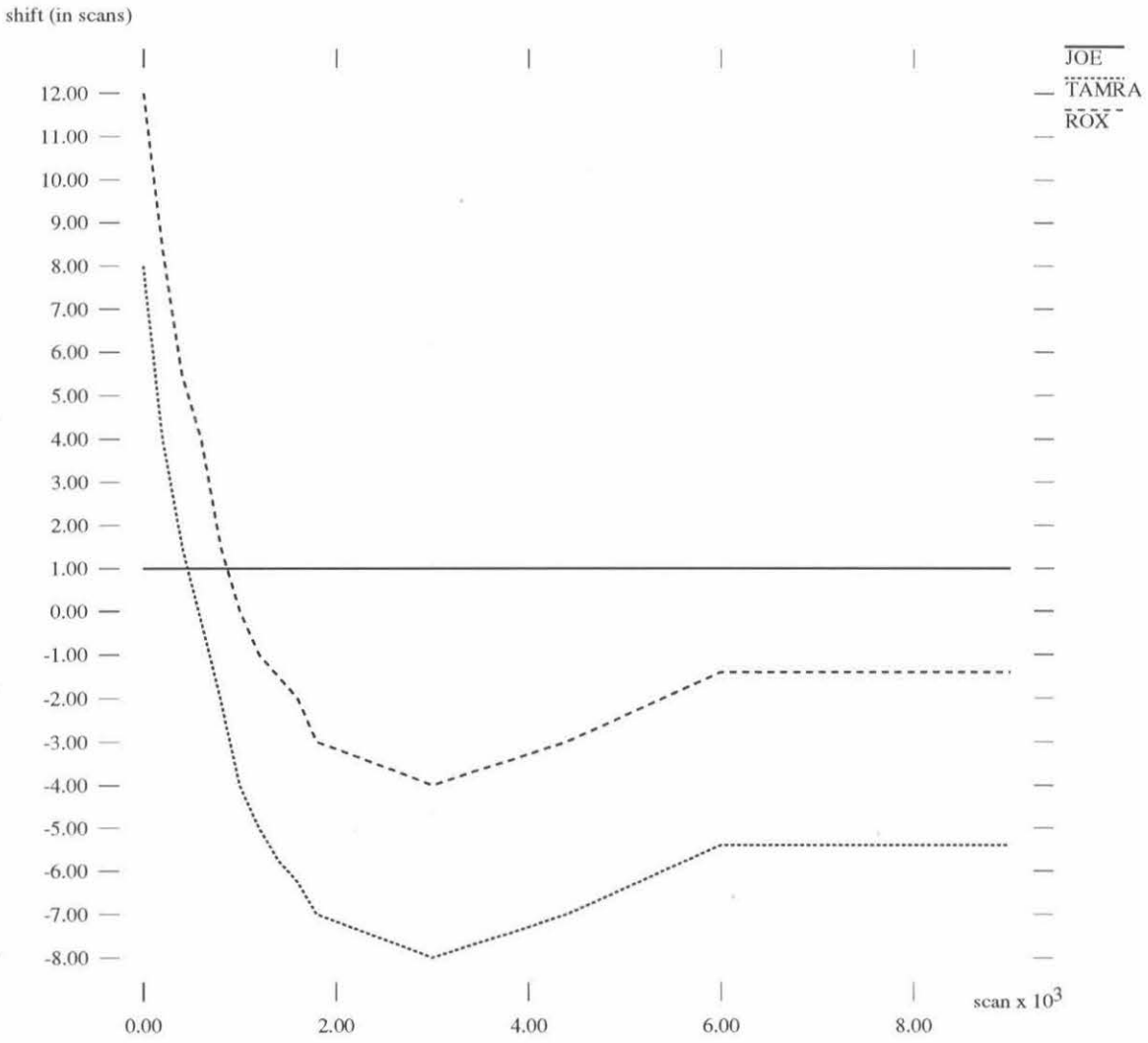


Figure 6.2: Graphs showing the mobility shifts associated with the dyes. The dye FAM is taken as the reference. These shifts incorporate both the effect of the dye and the end-base effect.

6.4 Base Calling (first two passes)

At this point, after the filter-to-dye space transformation, baselining and mobility shifting, the data is ready to be interpreted as a DNA sequence. That sequence is reflected in the order of the different color peaks in the data. In particular, the sequence is that of the primer strand reading in the $5' \rightarrow 3'$ direction with increasing scan number. As with many things, the first 90% is easy, the next 5% can be had with some work, and the last 5% is an unsolved problem. As I noted in the Introduction, I am not interested in solving that last 5%, so I have not gone to extremes in calling bases. There is enough data to be able to just ignore questionable data and since I compare my base calling with the final consensus, I am confident of what is left.

Most of the bases can be called just by finding peaks. Runs of identical bases sometimes require a knowledge of what the peak spacing is to call the proper number of bases. The major constraint on base calling is lack of resolution. When the peak width gets large relative to the peak spacing, it becomes impossible to sort out the peaks. The second most important problem is "compressions." This is where the sequence near the end of a fragment has the appropriate symmetry to form a hairpin loop, thus altering its mobility. The signature of this effect is a compression of the peaks (hence the name), sometimes even altering their order, followed by a rarefaction as the hairpin structure becomes less stable as it gets farther from the end. Another, less common, effect that is also, unfortunately, known as a compression, occurs during the sequencing reaction itself. This is where, for some reason, the enzyme tends to terminate the extension in all four reactions. In the data, this manifests itself as peaks in all four colors on top of each other.

I ignore both types of compressions. In practice, they are resolved by resequencing using different experimental techniques that are less prone to such artifacts.

The base calling algorithm is a three-pass algorithm. The first pass is a very

simple peak-finder that has a very loose pre-conception of what the appropriate peak spacing should be and loose rules concerning which peaks to accept as potential real base peaks. The second pass normalizes the data so that the average peak is 100 units high. This allows comparisons of peak heights between colors. The third pass computes an estimate of what the peak spacing is based on the positions of the peaks found in the second pass, and uses that information to try to call some of the more ambiguous peaks. There are two functions common to all three passes, one is filtering and the other is finding the primer peak.

6.4.1 Filtering

It is necessary to filter the data a bit in order to robustly find peaks without getting confused by high frequency noise. I filter the data by convolving it with the difference of two gaussians. The parameters of the two gaussians vary as a function of scan number in order to compensate a bit for the increasing peak width as the run progresses. This filtering scheme tends to sharpen up peaks, but it is not meant to be interpreted as a "corrected" view of the data. It is used in conjunction with, not as a replacement for, the original data.

The filter kernels are 31 scans wide. Four parameters are used to describe a kernel, they change every 500 scans. They change linearly over the following ranges as the scan number goes from 0 to 10000. These values were chosen through the rather *ad hoc* procedure of looking at the graphs from various fragments and playing with the parameters until the result looked "OK."

Variable	Range
`peak`	$2.0 \rightarrow 2.0$
`width`	$2.0 \rightarrow 4.0$
`width_ratio`	$2.0 \rightarrow 2.0$
`scale`	$0.5 \rightarrow 0.3$

The following code fragment computes the filter kernel from the four parameters:

```
filter_coef(scale,peak,width,width_ratio,filter)
float scale,peak,width,width_ratio,*filter;
{
   int i;
   float x,y,t1[31],t2[31],s1,s2,c1,c2;
   s1=width;
   s2=s1*width_ratio;
   sum1=sum2=0.0;
   for(x=-15; x<=15; x++){
      y=exp((-x*x)/(s1*s1));
      sum1+=y;
      t1[(int)(x-(-15))]=y;
      y=exp((-x*x)/(s2*s2));
      sum2+=y;
      t2[(int)(x-(-15))]=y;
   }
   c2=peak/((sum2/sum1)-1.0);
   c1=c2*sum2/sum1;
   for(i=0; i<=30; i++){
      filter[i]=scale*((c1*t1[i])-(c2*t2[i]));
   }
}
```

6.4.2 Finding the Primer Peak

The primer peak is the very large, broad peak at the beginning of a run due to excess dye-modified primer. It's exact location varies a bit from run to run and, due to the way the samples are loaded on the gel, it also varies from lane to lane within a single gel. It is usually around 600 scans (one hour) from the beginning of a run. Representing, as it does, the beginning of the real data for a lane, it is a major landmark. Several things later on are located with respect to the primer peak. Later on, after the second-pass base calling, I will locate the SmaI site of the fragment as a more precise landmark. The following code finds the primer peak given the filtered data. The filtered data is used because what we really want to look at is the signal strength without the very low frequency background.

```
/* This compares the ratio of the sum of a 50 scan window with
   the sum of a 100 scan window 50 scans in front of the 50 scan
   window.  When this ratio is lowest, the primer peak is in the
   middle of the 100 scan window.

   fdata[color][scan] is the two-D array of the filtered data.
   dcnt is the number of scans in fdata.
   prpos is the (returned) value of the primer position.
*/
int find_primer2(fdata,dcnt,prpos)
float **fdata;
int dcnt,*prpos;
{
   int i,j,k,pos,color;
   float ratio,max_ratio,sum50,cnt50,sum100,cnt100;
   if(dcnt<1600){
      *prpos=850; /* the default */
      return(0);
   }
   max_ratio=0.0;
   k=850;
   for(pos=150; pos<=1550; pos+=25){
      sum50=0.0;  cnt50=0.0;
      sum100=0.0; cnt100=0.0;
      for(i=pos-150; i<pos-100; i++){
         for(color=0; color<4; color++){
            if(fdata[color][i]>=0.0){
               sum50+=fdata[color][i];
               cnt50++;
            }
         }
      }
      sum50/=cnt50;
      for(i=pos-50; i<pos+50; i++){
         for(color=0; color<4; color++){
            if(fdata[color][i]>=0.0){
               sum100+=fdata[color][i];
               cnt100++;
            }
         }
      }
      sum100/=cnt100;
```

```
      ratio=sum100/sum50;
      if(ratio>max_ratio){
         max_ratio=ratio;
         k=pos;
      }
   }
   *prpos=k;
}
```

6.4.3 Finding Peaks

This is a very simple algorithm that just finds local maxima in the filtered data that are greater than 5 units high. This is a very permissive algorithm, it is up to later stages to throw out the spurious peaks.

```
/* a couple of structure definitions */
typedef struct _pkloc {
   int color;
   float scan;
}Pkloc;

typedef struct _peak {
   int pcnt;
   Pkloc data[1];
}Peak;

/* fdata[color]scan] is the two-D array of the filtered data.
   dcnt is the number of scans in fdata[][].
   peaks is the structure array where the scan-sorted list of
   peaks is returned.
*/

find_peaks(fdata,dcnt,peaks)
float **data,**fdata;
int dcnt;
Peak *peaks;
{
   int i,j,k,scan,color,pcnt,pkent,prpos,low,index[MAX_SCANS];
   float x,y,z,*dat,*histptr;
```

```
    float ftmp[MAX_SCANS];
    Pkloc *pklocks,tpeaks[MAX_SCANS];
    FILE *outfp;
    pklocks=peaks->data;
    pcnt=0;
    for(color=0; color<4; color++){
       dat=fdata[color];
       /* start at scan 2 and work up to within 2 scans of the end */
       scan=2;
       while(scan<dcnt-3){
          x=dat[scan];
          if(x<=5.0){goto no;}/* reject really small peaks */
          /* has to be a local max */
          if(dat[scan-1]>x || dat[scan+1]>x){goto no;}
          tpeaks[pcnt].color=color;
          tpeaks[pcnt].scan=scan;
          scan+=2; pcnt++;
          continue;
        no:
          scan++;
       }
    }
    /* sort peaks in order of increasing scan */
    for(i=0; i<pcnt; i++){
       ftmp[i]=tpeaks[i].scan;
    }
    fsort(pcnt,ftmp,index);
    for(i=0; i<pcnt; i++){
       pklocks[i].scan=tpeaks[index[i]].scan;
       pklocks[i].color=tpeaks[index[i]].color;
    }
    peaks->pcnt=pcnt;
}
```

6.4.4 First-Pass Base Calling

This is the first-pass algorithm for calling bases. It knows nothing about spacing rules, it just has some cheap heuristics for deciding whether a peak is likely to be a real base peak or a noise peak. The main purpose of this is to call the bases well

enough to get a rough average of real peak heights in order to then normalize the 4 graphs to an average peak height of 100.

```
/* some structure definitions */
typedef struct _pkloc {
   int color;
   float scan;
}Pkloc;

typedef struct _peak {
   int pcnt;
   Pkloc data[1];
}Peak;

typedef struct _bloc {
   int base;    /* C=0, A=1, G=2, T=3, N=4 */
   float scan;
}Bloc;

typedef struct _abloc {
   int cnt;
   Bloc data[1];
}Abloc;

find_bases1(data,fdata,dcnt,abloc,peaks)
float **data,**fdata;
int dcnt;
Abloc *abloc;
Peak *peaks;
{
   int i,j,k,dir,scan,nscan,color,tc,bcnt,bcnt1;
   int low,high,index[MAX_SCANS];
   int pcnt,prpos,use,lcnt;
   float min_peak_height=100.0;
   float abs_min_peak_height=30.0;
   float amp_ratio=1.5;
   float max,x,y,z,space,new_shift,spacings[MAX_SCANS];
   float areas[4],ftmp[MAX_SCANS];
   int sp_cnt,max_sp_cnt,min_sp_cnt;
   Abloc *abloc1;
   Bloc *blocs,*blocs1;
```

```
Pkloc *pklocs;
/* abloc1 and blocs1 are temporary structures. */
abloc1=(Abloc *)Mymalloc(MAX_SCANS*sizeof(Bloc),"BLOCS");
abloc1->cnt=0;
blocs1=abloc1->data;
pcnt=peaks->pcnt;
pklocs=peaks->data;
blocs=abloc->data;
bcnt1=0;
for(i=0; i<pcnt; i++){
   scan=pklocs[i].scan;
   /* if this color has the highest peak for +-3, and
      (the original data is greater than min_peak_height,
      or the integral over +-2 scans is more than
      amp_ratio greater than any other color with non-neg
      peakiness) use it */
   max= -1.0;
   tc=0;
   use=1;
   color=pklocs[i].color;
   if(data[color][scan]<min_peak_height){
      /* have to check the area ratios */
      if(data[color][scan]<abs_min_peak_height){
         use=0;
      }else{
         for(k=0; k<4; k++){
            areas[k]=0.0;
            for(j=scan-2; j<=scan+2; j++){
               areas[k]+=data[k][j];
            }
         }
         for(k=0; k<4; k++){
            /* if any color with non-neg peakiness is
               greater than peak color, flush it */
            if(k!=color){
               if(((amp_ratio*areas[k])>areas[color]) &&
                  fdata[k][scan]>=0){
                  use=0;
               }
            }
         }
      }
```

```
      }
      /* here we check to make sure a peak is the highest
         peak within 3 scans
      */
      for(color=0; color<4; color++){
         x=0.0;
         for(j= -3; j<=3; j++){
            x=MAX(x,fdata[color][scan+j]);
         }
         if(x>max){
            tc=color;
            max=x;
         }
      }
      if(tc!=pklocs[i].color){
         use=0;
      }
      if(use){
         blocs1[bcnt1].scan=scan;
         blocs1[bcnt1].base=tc;
         bcnt1++;
      }
   }
   abloc1->cnt=bcnt1;
   abloc->cnt=abloc1->cnt;
   for(i=0; i<abloc1->cnt; i++){
      blocs[i].scan=blocs1[i].scan;
      blocs[i].base=blocs1[i].base;
   }
}
```

6.4.5 Graph Normalization

The data, as it comes out of the sequencer, has no reference intensity. The value of the dye-space data is determined by the amount of DNA loaded, the laser power, the transmission of the optical filters, the geometry of the lens system, the voltage on the photomultiplier tube, the gain of the amplifier electronics, and the scaling of the filter-to-dye space matrix. Most of these effects are completely uncalibrated, hence the

scaling of the data is effectively random. After the first-pass base calling it is possible to re-scale the data by the average peak height in a certain window. This allows me to use intensity data as a criterion in calling bases. It also allows comparisons between colors that were not possible before. The following code renormalizes the data given the first-pass base calling. It also applies the same normalization coefficients to the filtered data.

```
/* This routine looks at a section of the data to try and figure out the
   relative strengths of the different color signals.  It then evens them
   out.  fdata gets normalized the same way data gets it.
*/

norm_data(data,fdata,dcnt,abloc)
float **data,**fdata;
int dcnt;
Abloc *abloc;
{
   int i,j,k,scan,color,base;
   float x,y,z,avg_peak[4],num_peak[4];
   int scan_low=1500;
   int scan_high=5000;
   if(dcnt<=scan_high){\* a too-short chromatogram */
      printf("Norm_data:  Only %d scans.",dcnt);
      return(-1);
   }
   /* for each color, accumulate the average peak heights for
      all bases within the window scan_low to scan_high
   */
   for(color=0; color<4; color++){
      avg_peak[color]=0.0;
      num_peak[color]=0.0;
   }
   for(i=1; i<abloc->cnt-1; i++){
      scan=abloc->data[i].scan;
      if((scan>=scan_low) && (scan<scan_high)){
         base=abloc->data[i].base;
         x=data[base][scan];
         num_peak[base]++;
```

```
            avg_peak[base]+=x;
         }
      }
      for(color=0; color<4; color++){
         if(num_peak[color]==0.0){
            printf("base %d has no peaks\n",color);
            avg_peak[color]= -1.0;
         }else{
            avg_peak[color]/=num_peak[color];
         }
      }
      /* now go through and normalize everything to an even 100.0 average */
      for(color=0; color<4; color++){
         if(avg_peak[color]>0.0){
            x=100.0/avg_peak[color];
            for(scan=0; scan<dcnt; scan++){
               data[color][scan]*=x;
               if(fdata!=NULL){
                  fdata[color][scan]*=x;
               }
            }
         }
      }
}
```

6.4.6 Second-Pass Base Calling

The second pass at base calling assumes that the data has been normalized to an average peak height of 100. This allows me to use peak height as a criterion in determining whether a peak is a real base peak or just noise.

```
find_bases2(data,fdata,dcnt,abloc,peaks)
float **data,**fdata;
int dcnt;
Abloc *abloc;
Peak *peaks;
{
   int i,j,k,dir,scan,nscan,color,tc,bcnt,bcnt2;
   int low,high,index[MAX_SCANS];
```

```
int pcnt,prpos,use,lcnt;
float fheight,pheight;
float min_peak_height=50.0;
float abs_min_peak_height=30.0;
float min_sum_height=80.0;
float min_fheight=25.0;
float amp_ratio=1.5;
float max,x,y,z,space,new_shift,spacings[MAX_SCANS];
float areas[4],ftmp[MAX_SCANS];
int sp_cnt,max_sp_cnt,min_sp_cnt;
Abloc *abloc1;
Bloc *blocs,*blocs1;
Pkloc *pklocs;
/* abloc1 is a temporary structure */
abloc1=(Abloc *)Mymalloc(MAX_SCANS*sizeof(Bloc),"BLOCS");
abloc1->cnt=0;
blocs1=abloc1->data;
pcnt=peaks->pcnt;
pklocs=peaks->data;
blocs=abloc->data;
bcnt2=0;
for(i=0; i<pcnt; i++){
   scan=pklocs[i].scan;
   color=pklocs[i].color;
   pheight=data[color][scan];
   fheight=fdata[color][scan];
   if(pheight<abs_min_peak_height || fheight<min_fheight){
     continue;
   }
   if(pheight+fheight<min_sum_height){continue;}
   /*if there is another peak within +-3 scans, we must be
     at least half as tall both in pheight and fheight
   */
   max= -1.0;
   tc=0;
   use=1;
   if(data[color][scan]<min_peak_height){
      /* have to check the area ratios */
      if(data[color][scan]<abs_min_peak_height){
         use=0;
      }else{
         for(k=0; k<4; k++){
```

```
            areas[k]=0.0;
            for(j=scan-2; j<=scan+2; j++){
               areas[k]+=data[k][j];
            }
         }
         for(k=0; k<4; k++){
            /* if any color with non-neg peakiness is
               greater than peak color, flush it */
            if(k!=color){
               if(((amp_ratio*areas[k])>areas[color]) &&
                  fdata[k][scan]>=0){
                  use=0;
               }
            }
         }
      }
   }
   tc=pklocs[i].color;
   /* the +-3 scan condition */
   for(color=0; color<4; color++){
      if(tc==color){continue;}
      for(j= -3; j<=3; j++){
         if(pheight<(0.5*data[color][scan+j]) ||
            fheight<(0.5*fdata[color][scan+j])){
            use=0;
            break;
         }
      }
   }
   if(use){
      blocs1[bcnt2].scan=scan;
      blocs1[bcnt2].base=tc;
      bcnt2++;
   }
}
abloc1->cnt=bcnt2;
abloc->cnt=abloc1->cnt;
for(i=0; i<abloc1->cnt; i++){
   blocs[i].scan=blocs1[i].scan;
   blocs[i].base=blocs1[i].base;
}
Myfree(abloc1);
```

}

6.5 Base Calling (third pass)

The third pass at base calling is where I use spacing information. Part of this involves finding a precise, base-position related landmark. The primer position is too crude for the final pass, in fact the major use I make of the primer position is in finding the SmaI site.

6.5.1 The SmaI Site

Recall from Chapter 4 that each fragment is identical up to the middle of the SmaI site, where the insert is placed; every insert is preceded by 4 C's in a row. This makes a fine landmark which I find by searching the second-pass base calling, starting at the primer position, for three C's in a row. I then declare the SmaI site to be the second base after the third C. This is because it is not unusual for the erratic base spacing in this region to confuse the base calling software; only three C's can reliably be expected to be found. This algorithm can be off by a base if the software happens to miss the first C and only find the last three. Thus I might miss the first base in the insert, but I never confuse the final C from M13mp18 with an insert base. From here on, base positions are referenced to the SmaI site.

6.5.2 Peak Spacing

My first attempts at using peak spacing involved an adaptive technique whereby I would extrapolate the nominal peak spacing from the bases I had already called. This turned out to be unstable, in spite of my attempts to stabilize it. As soon as the algorithm started to make any errors in the number of bases it called, this error

was incorporated into its idea of the proper base spacing, which then caused more errors to be made later on. This error feedback is, by its nature, positive and hence unstable. I could adjust the gain of this feedback, but there did not seem to be any gain setting that was both high enough to track the actual variations in peak spacing and low enough to keep the instabilities within reasonable bounds. I decided the solution to this problem was to measure, in some run-condition invariant way, the expected peak spacing as a function of base position. I could then use this reference graph to extrapolate the peak spacing farther out, thus allowing me to use islands of regular base spacing to anchor future spacing estimates. This presented something of a chicken-and-egg problem as I could not measure the expected peak spacing without calling bases, and I needed the spacing graph to call those very bases. The solution was an iterative process, which I describe below. For the moment, let's assume such a spacing graph exists.

There is still the question of how to define this graph in the most run invariant way. A simple graph of absolute peak spacing versus absolute scan number would be unsatisfactory since the run conditions vary too much from run to run. The first improvement is to graph the spacing against base position relative to the SmaI site. The second improvement is to graph not the absolute peak spacing, but the spacing relative to a region near the beginning. The graph is normalized so that the spacing at peak 100, from the SmaI site, is exactly 1.00. The final refinement is to utilize the voltage and temperature information that the sequencer records.

To a first approximation, the velocity with which a molecule travels through the gel is proportional to the electric field and inversely proportional to the viscosity of the buffer solution in the gel. I make the assumptions that the electric field is proportional to the recorded voltage and that the viscosity of the buffer solution is proportional to the viscosity of water at the recorded temperature. The viscosity of water changes by about 2% per °C at normal gel temperatures. Using these assumptions and the

recorded voltages and temperatures, I compute a "corrected scan" position for each peak, i.e., the scan where that peak would have been if the gel had been run at a constant 1500 volts at a constant temperature of 40°C. Figure 6.3 shows the resulting graph. More will be said about it below.

6.5.3 Reference Spacing

Part of using the spacing graph is getting the best possible estimate for the peak spacing somewhere. It doesn't matter where, since I just need it to scale the spacing graph to match it to this particular fragment. Given a base position range, the routine `best_spacing()` finds the most regularly spaced 11 contiguous bases in that range and reports back the average spacing of those 11 bases and where they are in the fragment.

```
/*
   this returns the most regular spacing over the range lowbase->hibase.
   spacing returns the spacing, where returns what base position of the
   best spacing.
*/
best_spacing(bases,lowbase,hibase,spacing,where)
Basedata *bases;
int lowbase,hibase,*where;
float *spacing;
{
   int i,j,k;
   int bestloc;
   float x,y,z,bestreg,avg;
   bestreg=100000.0; /* big number */
   for(i=lowbase; i<hibase-10; i++){
      x=(bases[i+10].scan-bases[i].scan)/10.0;
      y=0.0;
      for(j=1; j<=10; j++){
         y=MAX(y,ABS(x-(bases[i+j].scan-bases[i+j-1].scan)));
      }
      if(y<bestreg){
```

```
            bestreg=y;
            bestloc=i;
        }
    }
    *where=bestloc;
    *spacing=(bases[bestloc+10].scan-bases[bestloc].scan)/10.0;
    if(*spacing==0.0){\* an error condition */
        printf("BS: lowbase=%d  hibase=%d\n",lowbase,hibase);
        printf("BS: bl=%d bases[bl+10].scan=%d  bases[bl].scan=%d\n",
                bestloc,bases[bestloc+10].scan, bases[bestloc].scan);
    }
}
```

6.5.4 Third-Pass Base Calling

This final pass at base calling requires that the data is normalized to an average peak height of 100, that the position of the first insert base is known and that a reference spacing graph exists. It also requires the results of the second-pass base calling. The expected spacing is initially determined by calling `best_spacing()` on the second-pass base calling in the region from base 50 to base 150. The position returned by `best_spacing()` is called `fbase`. `nbcnt` is the position of the currently called base. When `nbcnt` gets to be 50 bases ahead of `fbase`, I call `best_spacing()` on the third-pass base calling in the region from `nbcnt`−150 to `nbcnt`. `comp_nom_spacing()` returns the expected spacing according to extrapolations based on the reference spacing graph, the spacing returned by `best_spacing()` and the recorded temperature and voltage of this particular fragment.

```
/* find_bases3 tries to fix up the preliminary base calling by studying
   the spacing.  The following are required inputs:
   fragd->bases is filled in with the result of the second-pass base
   calling.
   fragd->insert_base is filled in with  the location of the first
   insert base.
   fragd->dspdata is the data normalized to a mean peak height
```

```
   of 100
   fragd->fildata is the normalized filtered data.
   Results returned in freag->bases1.
*/
#define INSERT_THRESH (1.60)
find_bases3(fragd)
Fragdata *fragd;
{
   int i,j,k,bcnt,bpos,excess,color,tscan,maxcolor,fbase;
   float x,y,z,scan1,scan2,nomsp,avgval,maxval,fspacing;
   float *dspdata[4],*fildata[4];
   Basedata *bases,*nbases,*abases,tmpbases[MAX_BASES];
   int nbcnt,*abcnt;
   abases=fragd->fraux->bases2;
   nbases=malloc(MAX_BASES*sizeof(Basedata));
   abcnt=&(fragd->fraux->baselen2);
   /* load_nom_spacings() loads a global array with spacing
      information derived from the universal spacing graph
      adjusted according to temperature and voltage information
      recorded for this particular fragment.  This array will
      be used later by the routine comp_non_spacing().
   */
   load_nom_spacings(nomspacings);
   bases=&(fragd->bases[0]);
   bcnt=fragd->basecnt;
   for(i=0; i<4; i++){
      dspdata[i]=fragd->dspdata+(i*fragd->vnum);
      fildata[i]=fragd->fildata+(i*fragd->vnum);
   }
   /* first, remove double peaks, I.e. peaks of the same color
      separated by less than 0.5* nomsp, and replace them by one
      peak between the two. */
   for(i=0; i<fragd->insert_base; i++){
      tmpbases[i].base=bases[i].base;
      tmpbases[i].scan=bases[i].scan;
   }
   nbcnt=i;
   /* find the best spot between bases 50 and 150
      NOTE THAT FBASE IS RELATIVE TO FRAGD->INSERT_BASE !
      THIS MAKES FBASE COMPATIBLE WITH COMP_NOM_SPACING
   */
   best_spacing(bases+fragd->insert_base,50,150,&fspacing,&fbase);
```

```
/* record the computed spacing at base 100 as fragd->nom_sp */
nomsp=comp_nom_spacing(nomspacings,fragd,fbase,fspacing,
                       bases[100+i].scan,100);
fragd->nom_sp=nomsp;
for(bpos=i; bpos<bcnt-1; bpos++){
   if(nbcnt%10==0){
      x=comp_nom_spacing(nomspacings,fragd,fbase,fspacing,
                         bases[bpos].scan,nbcnt-fragd->insert_base);
      if(x>1.0){nomsp=x;}/* a reality check, should be an error */
   }

   /*this gets rid of double-called bases.
     I.e. less than 0.5*nomsp apart */
   if(bases[bpos].base==bases[bpos+1].base &&
      ((bases[bpos+1].scan-bases[bpos].scan)<(0.5*nomsp))){
       /* a winner! */
      tmpbases[nbcnt].base=bases[bpos].base;
      tmpbases[nbcnt].scan=
        (0.5*(bases[bpos+1].scan+bases[bpos].scan))+0.5;
      nbcnt++;
      bpos++;
   }else{
      tmpbases[nbcnt].base=bases[bpos].base;
      tmpbases[nbcnt].scan=bases[bpos].scan;
      nbcnt++;
   }

}
if(bpos<bcnt){
   tmpbases[nbcnt].base=bases[bpos].base;
   tmpbases[nbcnt].scan=bases[bpos].scan;
   nbcnt++;
}
bcnt=nbcnt;
/* ignore the first 100 bases for now, the peak information is
   probably a better set of criteria than spacing information
   there, anyway. */
for(i=0; i<100+fragd->insert_base; i++){
   nbases[i].base=tmpbases[i].base;
   nbases[i].scan=tmpbases[i].scan;
}
nbcnt=i;
```

```
for(bpos=i; bpos<bcnt; bpos++){
   /* if were getting far from the last fbase, recompute fspacing
      and fbase from the last 150 bases  */
   j=nbcnt-fragd->insert_base; /* j is now relative to insert */
   if(j>fbase+50){
      x=fspacing;
      best_spacing(nbases+fragd->insert_base,
                   MAX(j-150,0),
                   j-1,&fspacing,&fbase);
   }
   /* compute the expected spacing here */
   scan1=tmpbases[bpos-1].scan;
   scan2=tmpbases[bpos].scan;
   z=scan2-scan1;
   x=comp_nom_spacing(nomspacings,fragd,fbase,fspacing,
                      (int)scan2,nbcnt-fragd->insert_base);
   if(x<1.0){/* this is an error condition */
      printf("NOMSP=%f  bpos=%d  nbcnt=%d fbase=%d fspacing=%f\n",
             nomsp,bpos,nbcnt,fbase,fspacing);
   }else{
      nomsp=x;
      excess=(z/nomsp)+1.0-INSERT_THRESH;
      if(excess>=1){/* candidate for multiple insertions */
         y=z/(excess+1.0);
         /*  fill in the interval with the base with the highest
             value in dspdata
         */
         for(i=1; i<=excess; i++){
            tscan=scan1+(i*y)+0.5;
            maxval= -10000.0;
            for(color=0; color<4; color++){
               avgval=(dspdata[color][tscan-1]+
                       dspdata[color][tscan]+
                       dspdata[color][tscan+1])/3.0;
               if(avgval>maxval){
                  maxval=avgval;
                  maxcolor=color;
               }
            }
            nbases[nbcnt].base=maxcolor;
            nbases[nbcnt].scan=tscan;
```

```
                nbcnt++;
            }
        }
    }
    nbases[nbcnt].base=tmpbases[bpos].base;
    nbases[nbcnt].scan=tmpbases[bpos].scan;
    nbcnt++;
  }
  fragd->bases1=nbases;
  fragd->basecnt1=nbcnt;
}
```

6.5.5 Consensus Alignment

Since the sequence of the cosmid from whence all these fragments came is known, it would be foolish to rely solely on my basecalling software. (One may wonder why, if the sequence is already known, did I bother with writing base calling software at all. The answer is that while the sequence was known, the knowledge of where each base appears in the raw data was lost. The ABI software has a peculiar way of manipulating the data during the base calling process that makes it impossible to correlate the final sequence with the original data.)

I know where the peaks are in the data, and I have a reasonably good idea, from my own base calling, as to what the sequence around those peaks are. What I am after from the consensus sequence is the definitive sequence context. There are two basic types of mistakes in base calling, miscalls and indels (short for *in*sertion or *del*etion). An optimal alignment of two similar sequences can be defined by assigning different weights to matches,miscalls and indels and maximizing the sum of these weights over all possible combinations of adding spaces to the sequences and shifting them with respect to one another. There is a widely used dynamic programming algorithm for finding this optimal alignment which is described in detail in [Waterman 1988]. I used the following weights in all my alignments:

Matches:	1.0
Mismatches:	−1.0
Indels:	−1.5

To determine where a fragment belongs in the consensus, I find the best alignment between the second one-hundred bases (generally the most accurately called region of a run) in my base calling of the fragment and the consensus. It is necessary to compare the fragment both with the consensus as given and its complement, since the fragment could have been from either strand of the cosmid. I then use that rough positioning information to do an alignment of the full fragment with the appropriate strand of the consensus. That optimal alignment, and which strand it was from, is then recorded in the fragment file. This makes it possible to determine, from information in the fragment file, exactly where each base in my base calling actually came from in the original cosmid. It also reveals which bases were called incorrectly.

6.6 Computing The Spacing Graph

The spacing graph was computed as an average from 629 fragments from the 2-47 cosmid. The process was as follows:

1. Call the bases for each fragment.
2. Align each fragment with the consensus.
3. Using the consensus alignment to get accurate base positions and peak spacings, produce a spacing graph (normalized and corrected for voltage and temperature effects).
4. Average all 629 graphs, smooth and extrapolate by eye.
5. Go back to step 1, using the new spacing graph.

The first iteration of this procedure applied only the first two passes of the base calling algorithm, since I didn't have a spacing graph yet. I went through 3 iterations of this procedure. The graph in Figure 6.3 is the result. The manual smoothing and extrapolation in step 4 should not have introduced any bias, since I always compare the resulting base calling to the consensus. The final graph is from the actual data, it has not been smoothed or extrapolated.

One final caveat: if two sequences are not sufficiently similar locally, the alignment algorithm will sometimes find spurious local alignments. This is probably the case with the data beyond base position 600. I tried to minimize this by only considering bases that were correctly called and farther than 2 bases from the nearest miscall.

6.7 Quantitating Peaks

The final phase of this process is to quantitate the peaks representing the called bases. What I am after is a measure of the deoxy/dideoxy incorporation ratio. This is proportional to the number of fragment molecules of a given size generated by the sequencing reaction. This in turn is equal to the number of dye moieties in a band, which is presumably proportional to the fluorescence signal picked up by the photomultiplier tube, integrated over the entire band. Ideally, one would want to measure the peak area over the entire band. In practice this is complicated by small spurious peaks of unknown origin. It is not uncommon for there to be three peaks, a real one and two spurious ones, within 12 scans. Attempting to fit 3 gaussians, each with 3 free parameters to 12 noisy points turned out to be a mistake. I discovered that trying to fit gaussians to this data is not a robust procedure, the results were often clearly wrong.

I only need the peak areas relative to the surrounding peaks, since I'm going to normalize with respect to them anyway to get rid of gradual changes in signal

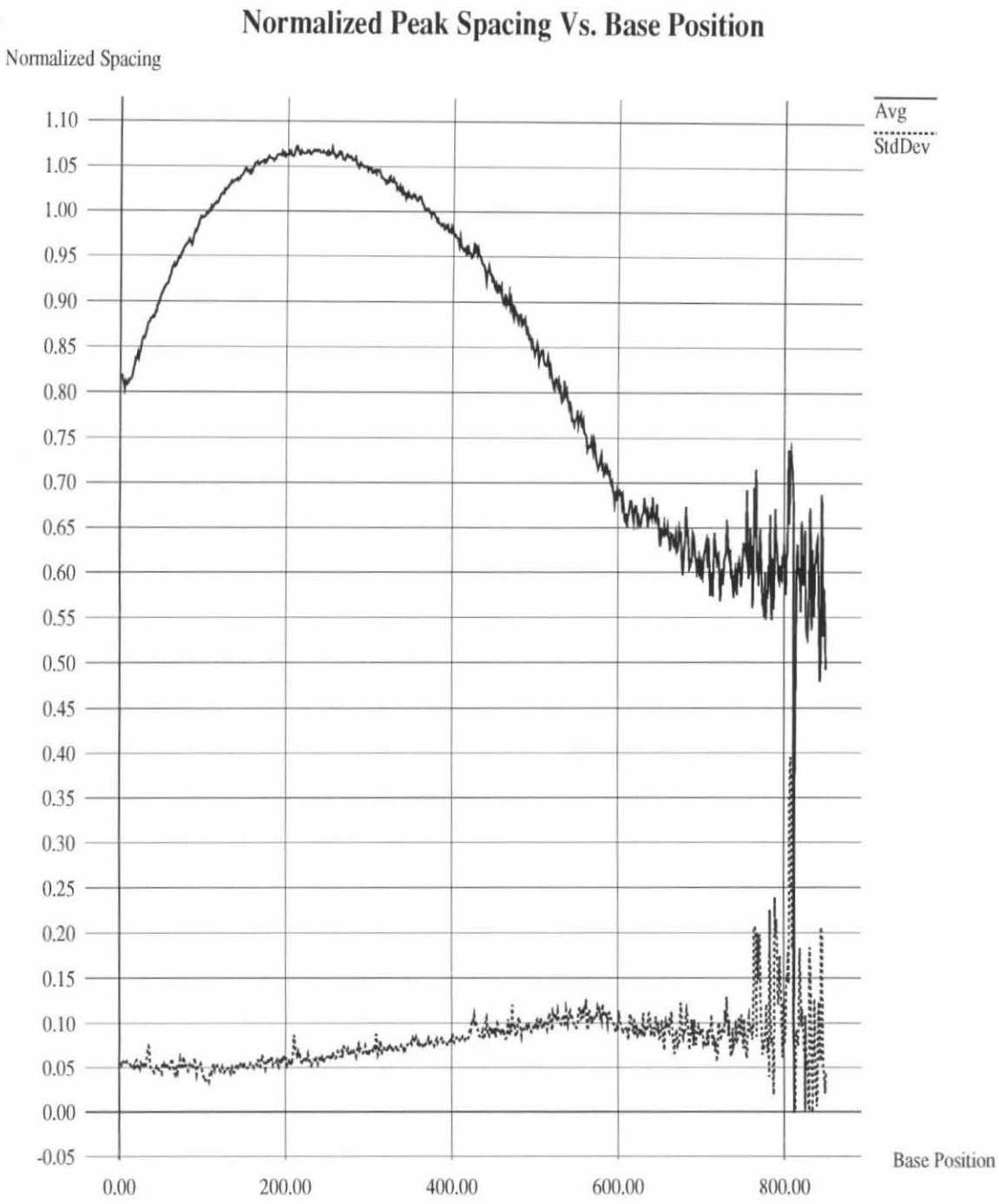


Figure 6.3: Graph of peak-to-peak spacing vs. base position. The spacing is normalized to 1.0 at base 100. This data was averaged over 629 fragments from the 2-47 cosmid. Corrections for variations in electrophoresis voltage and temperature have been made. Base counts are relative to the SmaI site. Data after base 600 is suspect due to the high number of errors that occur in sequencing that far out.

strength. So I make the assumption that the peak width changes slowly enough that I can use the peak height as a surrogate for the area. In practice, I average the height over the 3 scans centered on the peak.

One of the criteria used in the base calling is peak height; peaks that are too low are rejected as real base peaks. This potentially introduces a bias in the data, low peaks will be culled out. In order to correct for this I use my knowledge of the true sequence. If my base calling has left a gap (according to the consensus alignment) and if the spacing of the gap is consistent with the idea that a base is missing and if the 2 bases on both sides of the gap were called correctly, I assume there really is a base in the middle of the gap.

```
/* a new structure for storing all the info about quantitated peaks */
typedef struct _pkinfo{
   char base;    /* the consensus base for this peak */
   int  called;  /* bool. 1 if correctly called in fragment */
   int  fcnt;    /* base pos in fragment. Rel to fragd->true_base */
   int  conpos;  /* pos in consensus, neg implies comp strand */
}Pkinfo;

Pkval pkvals[1000]; /* place to put the 3-point sum stats */
int pkvalcnt;
Gfit_cmd(fragd)
Fragdata *fragd;
{
   int i,j,k,l,ent,seqlen,useit,color,evalcnt,called,vnum;
   float x,y,z;
   float **bldata,*bldata_[4];
   char entname[100],seq[2000],con[2000];
   int sptr,cptr;
   Fragdata *fragd;
   Basedata *fbases,tbases[10];
   Pkinfo  pks[2000];
   int pkcnt;
   vnum=fragd->vnum;
         fbases=fragd->bases1;
   decode_align(fragd,seq,con,&seqlen);
```

```
bldata_[0]=(float *)Mymalloc(4*vnum*sizeof(float),"BLDATA_TMP");
for(i=1; i<4; i++){
   bldata_[i]=bldata[0]+(i*vnum);
}
bldata=fragd->dspdata;
/* now that we have the alignment decoded, we can chose peaks to
 quantitate.
 The selection rules are:
 1) if a correct peak has 2 correct peaks on both sides, use it.
 2) if an uncalled peak has 2 good peaks on both sides & the spacing
    is reasonable, use it.
*/
sptr=fragd->true_base+fragd->fstart;  /* sptr keeps track of where
                                         in the fragment sequence
                                         i-2 is */
cptr=fragd->cstart; /* cptr keeps track of the consensus (i-2) */
pkcnt=0; pkvalcnt=0;
for(i=2; i<MIN(fragd->alignlen-2,600); i++){
   useit=0;
   if(i>=2 && seq[i-2]==con[i-2] && seq[i-1]==con[i-1] &&
      seq[i+1]==con[i+1] && seq[i+2]==con[i+2] &&
      (seq[i]==con[i] || seq[i]=='-')){/* a possibility */
      for(j=0; j<2; j++){
         tbases[j].base=fbases[sptr+j].base;
         tbases[j].scan=fbases[sptr+j].scan;
      }
      if(seq[i]==con[i]){/* the easy case */
         called=1;
         for(j=0; j<3; j++){
            tbases[j+2].base=fbases[sptr+j+2].base;
            tbases[j+2].scan=fbases[sptr+j+2].scan;
         }
         useit=1; /* a winner */
      }else{/* we know seq[i]=='-'. Here we have to look at
               the spacing */
         for(j=0; j<2; j++){
            /* remember the offset going across seq[i] */
            tbases[j+3].base=fbases[sptr+j+2].base;
            tbases[j+3].scan=fbases[sptr+j+2].scan;
         }
         x=(tbases[4].scan-tbases[0].scan)/4.0;
         /* x=avg spacing over all 5 */
```

```
            y=(tbases[3].scan-tbases[1].scan)/2.0;
            /* y=avg gap spacing */
            if(x/y<1.25 && x/y>0.75){
            /* if the avg gap spacing is within range use it */
               tbases[2].scan=(tbases[1].scan+tbases[3].scan)/2.0;
               tbases[2].base=base_from_char(con[i]);
               called=0;
               useit=1;
            }
         }
      }
      cptr++; sptr++;
      if(seq[i-2]=='-'){sptr--;}
      if(con[i-2]=='-'){cptr--;}
      if(useit){/* enter its stats into the list  */
         pkvals[pkvalcnt].base=tbases[2].base;
         pkvals[pkvalcnt].scan=tbases[2].scan;
         pkvals[pkvalcnt].basenum=cptr+1;
         pkvals[pkvalcnt].called=called;
         k=tbases[2].scan;
         color=tbases[2].base;
         pkvals[pkvalcnt].val=(bldata[color][k-1]+bldata[color][k]+
                               bldata[color][k+1])/3.0;
         pkvalcnt++;
      }
   }
   fragd->pkval0=pkvals;
   fragd->pkval0_cnt=pkvalcnt;
}
```

Chapter 7

The Data

7.1 Intro

After re-analyzing the image files and culling out fragments that were, according to the sequencing records, sequenced with nucleotide analogs (to resolve compressions), there remained 629 fragments that aligned with the insert from 2-47. These fragments were used to compute the spacing graph. In keeping with my philosophy of rejecting poor quality data instead of trying to fix it up with software, I rejected all fragments with more than 10 base calling errors in the first 300 bases. This left 545 fragments.

I also wanted to minimize the effect that secondary structure might have on the incorporation ratio. I did a search of the 2-47 consensus sequence to find potential hairpin structures and rejected everything within 20 bases of them. I used the following criteria to decide if a region had a potential hairpin:

1. A stem of 4 or more bases.

2. A loop of 3 to 7 bases.

3. At least two GC pairs.

Bases in a fragment closer than 20 bases to the SmaI site were excluded because their actual sequence context on the 5′ side is that of the M13mp18 genome, not 2-47. This implies the caveat that a base's context is indeterminate further than 20 bases away, it could be that of 2-47 or of M13mp18. Bases in a fragment further than 320 bases from the SmaI site were excluded because the peaks beyond there start to overlap. Bases in a fragment that were not called correctly by my software, or were within 2 bases of a miscalled base were also excluded. All these exclusions still left over 100,000 usable bases.

7.2 Statistical Analysis

The 110,549 usable fragment bases have a mean of 0.9992 and a standard deviation of 0.1007 (variance of 101.4×10^{-4}). The density function is graphed in Figure 7.1. The relative base frequencies are:

Base	Frequency
C	0.1913
A	0.3078
G	0.1908
T	0.3100

The 38% G+C content is typical of mammalian DNA.

Figure 7.2 is a graph of the mean intensity of the data versus position on the sequencing fragment. There are 110,549 fragment bases distributed over 300 positions, or $\approx$ 368 fragment bases per position. This should reduce the standard deviation by a factor of $\sqrt{368} \approx 19$. Thus one would expect one standard deviation to be $\approx 5.2 \times 10^{-3}$. The graph shows that the peak height normalization has removed any significant bias due to distance from the primer.

7.2.1 How Noisy is the Data?

The first goal is to get an idea of how noisy the data is. The number of fragments that include the base at position P in the consensus is called the "depth" at P (D_P). The depth for this data set varies from 0 to 11. If we assume that measuring an incorporation ratio at a consensus position P is equivalent to sampling a normal random variable with mean m_P and variance σ^2 (assumed to be the same for all P), then we can combine statistics for all base positions with $D_P \geq 2$ to give an estimate for σ^2. This is a measure of the intrinsic noise in the measurements since each measurement at position P has exactly the same sequence context for at least 20 bases in both directions.

There is some subtlety in this calculation. Given D samples from a random variable $\mathbf{X}$, the unbiased estimator for σ^2 of $\mathbf{X}$ is

$$s^2 = \frac{1}{(D-1)} \sum_{i=1}^{D} (x_i - \bar{x})^2. \tag{7.1}$$

s^2 is also a random variable with mean σ^2 and variance $\frac{2\sigma^4}{(D-1)}$. If there are N_D positions with depth D, then we can average over the N_D samples of s^2 to get a new random variable s_D^2 which also has mean σ^2 but variance $\frac{2\sigma^4}{N_D(D-1)}$. Now that we have the s_D^2, how do we compute a statistic from them that best estimates σ^2? Obviously, any

$$S^2 = \sum_{D=2}^{11} (\alpha_D s_D^2) \tag{7.2}$$

with the condition that

$$\sum_{D=2}^{11} \alpha_D = 1 \tag{7.3}$$

will have a mean of σ^2, but which choice of α_D's will give the statistic with the

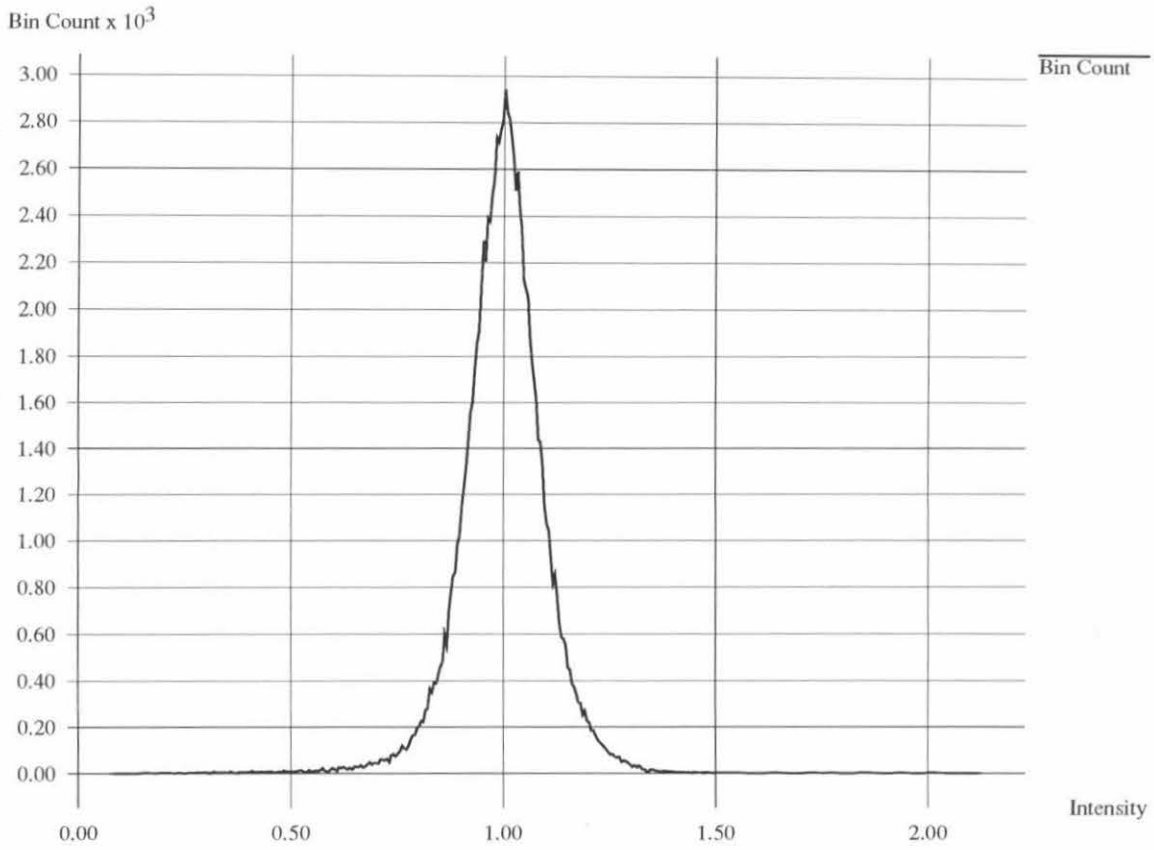


Figure 7.1: Intensity density of the data. Each bin spans 0.005 intensity units.

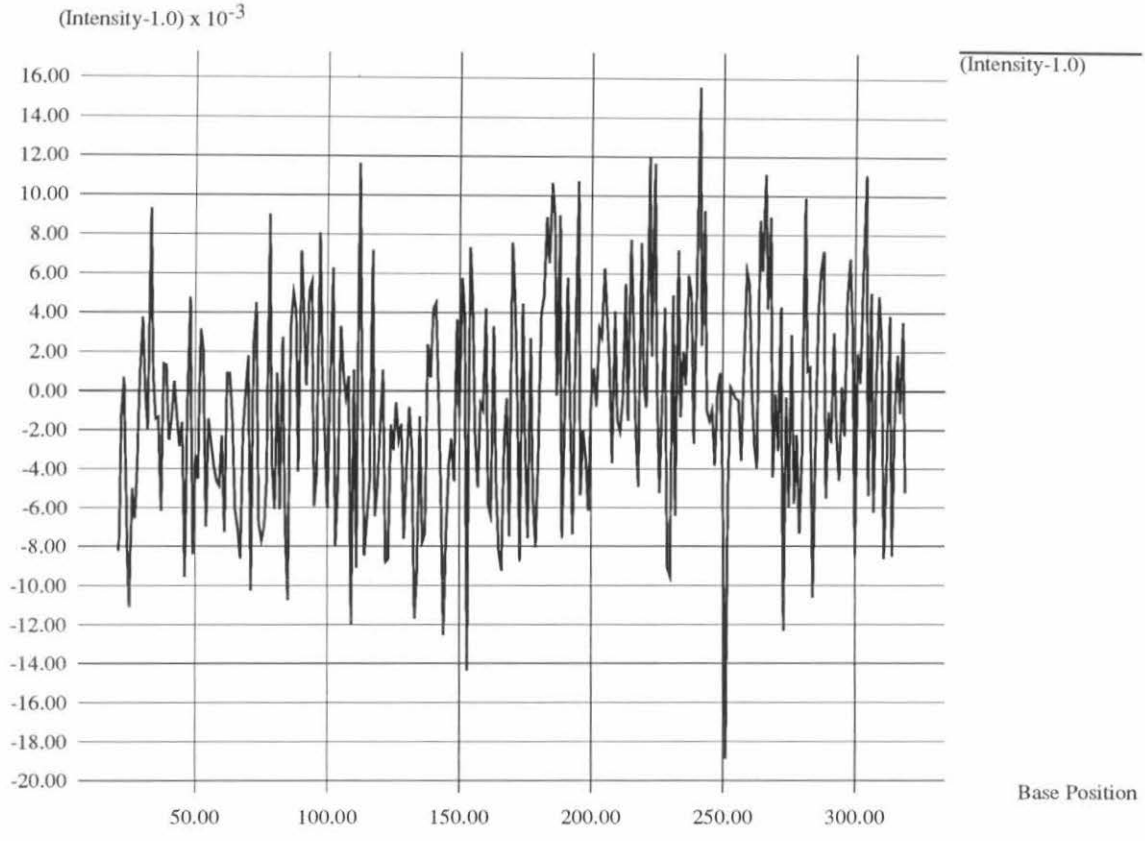


Figure 7.2: Graph of mean intensity vs. position in the sequencing fragment. Note the vertical scale and that the graph represents the deviation of the positional mean from the mean of the data as a whole. The expected standard deviation for this data is $\approx 5.2 \times 10^{-3}$.

D	N_D	$s_D^2 \times 10^{-4}$
1	11440	0.0
2	12389	35.3
3	8573	34.6
4	4763	41.7
5	2304	35.9
6	1173	44.3
7	780	54.4
8	340	36.2
9	224	45.5
10	63	33.4
11	16	66.3

Table 7.1: Partial results in calculating the noise variance. Weighting the s_D^2 appropriately yields $\sigma^2 = 39.0 \times 10^{-4}$.

smallest variance? We know

$$\mathrm{Var}(\mathrm{S}^2) = \sum_{\mathrm{D}=2}^{11} (\alpha_{\mathrm{D}}^2 \mathrm{Var}(\mathrm{s}_{\mathrm{D}}^2)). \tag{7.4}$$

Under the constraint of Equation 7.3, this is minimized when

$$\alpha_i s_i^2 = \alpha_j s_j^2, \qquad 2 \leq i, j \leq 11 \tag{7.5}$$

which happens when the α_D are proportional to $N_D(D-1)$. The calculations are detailed in Table 7.1. The final result is $\sigma^2 = 39.0 \times 10^{-4}$. This leaves a maximum variance of 62.4×10^{-4} to be explained by sequence context.

7.2.2 Which Base Positions Are Important?

The next question is, which base positions, with respect to the incorporation site, are important in determining the incorporation ratio? The crystallographic evidence implies that any sequence effect must be due to bases within 20 of the incorporation

sites. As a matter of convention, I call the incorporation site position 0, with position numbers increasing in the $5' \rightarrow 3'$ direction on the primer strand, so one would expect any sequence dependent effect to be confined to positions -20 to 20.

In calculating the noise variance, I grouped together fragment bases that represented the same consensus position. In order to estimate the effect that sequence context has, I group together fragment bases that have identical sequence at specified positions relative to the incorporation site. One can also consider this from the viewpoint of prediction accuracy. Grouping according to consensus position is equivalent to calculating the average intensity at each consensus position, then calculating the mean squared error one gets by using this as a predictor. Similarly, one can tabulate the mean intensity over all fragment bases that have the same 5 base sequence centered on the incorporation site for all 1024 different 5 base combinations. The difference between the total variance of the data and the mean squared error of predictions based on those 5 bases is a measure of how important those 5 bases are in determining the incorporation ratio.

Table 7.2 gives the variances when the data is grouped by 5-mers. Note that predictions based on positions -20 through -16 are almost equal to the total variance (100.7×10^{-4} vs. 101.4×10^{-4}), hence these base positions have almost no effect on the incorporation ratio. Positions -3 through 1 give the best predictions, lowering the variance by 39.6×10^{-4}.

In order to see if some non-contiguous set of base positions might make better predictions, I evaluated the variances for all 3003 ways of chosing 5 positions from the 15 positions from -8 to 6. Table 7.4 shows the 11 best combinations of base positions. Positions -3 through 1 still come out as the most influential.

Tables 7.3 and 7.5 show analogous results for predictions based on 6 bases. Positions -4 through 1 are the most influential. The fact that positions -3 through 1 appear in the top 10 combinations of 6 bases also attests to the importance of those

5 positions.

Table 7.6 shows predictions bases on 7 bases. Note that the variance for positions -20 through -14 is significantly lower than the total variance (85.3×10^{-4} vs. 101.4×10^{-4}). This is probably because of spurious correlations due to the fact that there are 16,384 7-mers and only 42,065 distinct consensus (and its complement) positions represented in the data.

Positions Used	Var $\times 10^{-4}$
XXXXX----------+----------	100.7
-XXXXX---------+----------	100.4
--XXXXX--------+----------	99.2
---XXXXX-------+----------	99.6
----XXXXX------+----------	99.9
-----XXXXX-----+----------	100.0
------XXXXX----+----------	100.3
-------XXXXX---+----------	99.9
--------XXXXX--+----------	101.8
---------XXXXX-+----------	97.7
----------XXXXX+----------	99.0
----------XXXXx----------	100.0
-----------XXXXx----------	95.9
------------XXXXx----------	101.8
-------------XXXXx----------	91.0
--------------XXXXx----------	90.2
---------------XXXxX----------	82.9
----------------XXxXX----------	**61.8**
-----------------XxXXX----------	69.3
------------------xXXXX----------	80.1
-------------------+XXXXX----------	90.2
--------------------+-XXXXX----------	95.3
---------------------+--XXXXX----------	98.2
----------------------+---XXXXX----------	100.7
-----------------------+----XXXXX----------	99.8
------------------------+-----XXXXX----------	100.0
-------------------------+------XXXXX----------	99.2
--------------------------+-------XXXXX----------	99.5
---------------------------+--------XXXXX----------	100.0
----------------------------+---------XXXXX----------	101.8
-----------------------------+----------XXXXX----------	101.9
------------------------------+-----------XXXXX----------	100.5
-------------------------------+------------XXXXX----------	99.9
--------------------------------+-------------XXXXX----------	99.8
---------------------------------+--------------XXXXX----------	99.6
----------------------------------+---------------XXXXX----------	99.8
-----------------------------------+----------------XXXXX----------	100.4

Table 7.2: Prediction variances for 5-mers from positions -20 (far left) to 20 (far right). "+" and "x" represent the incorporation site. The best predictions come from using positions -3 to 1, which gives a variance of 61.8×10^{-4}.

Positions Used	Var $\times 10^{-4}$
XXXXXX----------+----------	96.8
-XXXXXX----------+----------	95.7
--XXXXXX----------+----------	96.0
---XXXXXX----------+----------	96.5
----XXXXXX----------+----------	96.1
-----XXXXXX---------+----------	96.9
------XXXXXX--------+----------	95.9
-------XXXXXX-------+----------	98.4
--------XXXXXX------+----------	94.3
---------XXXXXX-----+----------	95.5
----------XXXXXX----+----------	96.5
-----------XXXXXX---+----------	92.4
------------XXXXXX--+----------	97.9
-------------XXXXXX-+----------	87.4
--------------XXXXXX+----------	87.1
--------------XXXXXx----------	79.2
---------------XXXXxX----------	**58.5**
----------------XXXxXX----------	59.3
-----------------XXxXXX----------	65.9
------------------XxXXXX----------	76.5
-------------------xXXXXX----------	87.2
--------------------+XXXXXX----------	90.9
---------------------+-XXXXXX----------	95.0
----------------------+--XXXXXX----------	96.2
-----------------------+---XXXXXX----------	96.6
------------------------+----XXXXXX----------	95.9
-------------------------+-----XXXXXX----------	95.9
--------------------------+------XXXXXX----------	96.3
---------------------------+-------XXXXXX----------	98.0
----------------------------+--------XXXXXX----------	98.2
-----------------------------+---------XXXXXX----------	96.7
------------------------------+----------XXXXXX----------	96.3
-------------------------------+-----------XXXXXX----------	96.2
--------------------------------+------------XXXXXX----------	96.1
---------------------------------+-------------XXXXXX----------	96.2
----------------------------------+--------------XXXXXX----------	97.1

Table 7.3: Prediction variances for 6-mers from positions -20 (far left) to 20 (far right). "+" and "x" represent the incorporation site. The best predictions come from using positions -4 to 1, which gives a variance of 58.5×10^{-4}.

Positions Used	Var $\times 10^{-4}$
-----XXXxX-----	61.8
----X-XXxX-----	67.4
-X----XXxX-----	67.9
X-----XXxX-----	67.9
--X---XXxX-----	68.0
---X--XXxX-----	68.0
------XXxXX----	69.3
------XXxX----X	69.5
------XXxX---X-	70.2
------XXxX--X--	70.3
------XXxX-X---	70.6

Table 7.4: 11 best ways to chose 5 bases out of the 15 positions from -8 (far left) to 6 (far right). The positions marked "x" represent the incorporation site. The best predictions come from using positions -3 to 1, which gives a variance of 61.8×10^{-4}.

Positions Used	Var $\times 10^{-4}$
----XXXXxX-----	58.5
-----XXXxXX----	59.3
-X---XXXxX-----	59.3
X----XXXxX-----	59.7
--X--XXXxX-----	59.9
---X-XXXxX-----	59.9
-----XXXxX----X	60.6
-----XXXxX--X--	60.8
-----XXXxX---X-	60.9
-----XXXxX-X---	61.3

Table 7.5: 10 best ways to chose 6 bases out of the 15 positions from -8 (far left) to 6 (far right). The positions marked "x" represent the incorporation site. The best predictions come from using positions -4 to 1, which gives a variance of 58.5×10^{-4}.

Positions Used	Var $\times 10^{-4}$
XXXXXXX----------+-------------------	85.3
-XXXXXXX----------+-------------------	86.0
--XXXXXXX----------+-------------------	85.3
---XXXXXXX----------+-------------------	85.6
----XXXXXXX---------+-------------------	86.6
-----XXXXXXX--------+-------------------	85.6
------XXXXXXX-------+-------------------	87.6
-------XXXXXXX------+-------------------	84.2
--------XXXXXXX-----+-------------------	85.2
---------XXXXXXX----+-------------------	86.9
---------XXXXXXX---+-------------------	82.7
----------XXXXXXX--+-------------------	87.0
-----------XXXXXXX-+-------------------	78.3
------------XXXXXXX+-------------------	78.2
-------------XXXXXXx-------------------	72.3
-------------XXXXXxX-------------------	54.3
--------------XXXXxXX------------------	**53.9**
----------------XXXxXXX----------------	54.8
-----------------XXxXXXX---------------	60.0
------------------XxXXXXX---------------	69.9
-------------------xXXXXXX--------------	77.1
--------------------+XXXXXXX------------	81.5
--------------------+-XXXXXXX-----------	84.1
--------------------+--XXXXXXX----------	86.5
--------------------+---XXXXXXX---------	85.7
--------------------+----XXXXXXX--------	85.6
--------------------+-----XXXXXXX-------	86.2
--------------------+------XXXXXXX------	87.2
--------------------+-------XXXXXXX-----	87.9
--------------------+--------XXXXXXX----	86.3
--------------------+---------XXXXXXX---	86.0
--------------------+----------XXXXXXX--	86.3
--------------------+-----------XXXXXXX-	85.7
--------------------+------------XXXXXXX	85.9
--------------------+-------------XXXXXXX	86.4

Table 7.6: Prediction variances for 7-mers from positions -20 (far left) to 20 (far right). "+" and "x" represent the incorporation site. The best predictions come from using positions -4 to 2, which gives a variance of 53.9×10^{-4}.

7.3 Effect of Particular Bases

From this point on, I consider the data after averaging the intensities of the N_D fragment bases at each consensus position. Table 7.7 shows the 30 lowest and highest intensities of the 42,065 usable consensus position. There are a number of patterns to see in this table; I will point out two. In the 30 lowest contexts, at base position -2 there are 25 A's; at position -3 there are 22 T's. This suggests a way to visualize the data. I sorted all the consensus positions by average intensity and split the resulting list into bins, each of which has at least 30 points and spans at least 0.005 intensity units. For each bin I computed the fraction of the entries with a given base at a given position. I then graphed this against the average intensity of the bin.

Figures 7.3 to 7.17 are the resulting graphs from positions -7 to 7. These have not been corrected for the non-uniform distribution of bases (see page 70).

Positions $-7, -6$ and -5 show almost no relationship between intensity and base identity. Position -4 has a weak effect while positions -3 through 2 have very strong influences on intensity. Positions 3 and 4 again have a small effect with positions further out having no effect. This is consistent with the variance vs. base position tables.

Sequence Context	Intensity	Sequence Context	Intensity
CTTCCCATAC T CTGAGTGCTA	0.079	CCCCCAGCAG G TGCATCTTTT	2.016
AGCCAACTAT T CTTTAATTAT	0.146	AGTCGCTGAG G CTGCACTACC	1.856
TTTGGGCTAC G TTTGGAATGT	0.152	AGTATTCATG G TCTCAAGTCA	1.811
GTAAGGAGAC G CGTTTAACAT	0.196	CCAGCTTAGC G GTTATTTTGT	1.803
TCTCCTCCTT T CTGTAATGTC	0.216	TCCCGGAAAG G CCTGTGTCTG	1.732
CAGCTTAGCG G TTATTTTGTC	0.240	GCCATGCTAG G CTCCTGACTG	1.668
AAACACATAT G CCTTACTCAG	0.260	TCCAGTGGAG G CCCTGTCTAC	1.656
AACAGTCCTT T CTTTTTCCTA	0.281	TCTTTTGTGG G TGACTCAGTT	1.616
TGTTATTTAT G CTGGGGTGGG	0.286	ACAACTGGAG G TAACCTCTTC	1.595
TCTTTTATAT T CTGTTAGTGA	0.289	AGCAATTAGC G GGGGTTTTTG	1.590
TATCATATAC T CAAAATGCTT	0.295	TCCTCCAGTG G AGGCCCTGTC	1.581
TTTAATGTAC G ATTTTGTTGT	0.301	GTTTTTGAAG G TTGGTTGGTG	1.528
GTTTGAATAT G CTTGGCCCAG	0.309	TTCCTAGTCC C CCCCACACAA	1.479
CATTACATAC A CTAGCAAGAT	0.327	AAATGTTATC C CCTTTCCTGG	1.479
GCTCCTACAC T CTTTCCACCA	0.330	TTCCTATGAG G TTGTTATCCC	1.476
TATGCCTTAC T CTGGTATAGG	0.341	ATTATTTACA G AATCTCAATA	1.447
AGTAATGTAT G CAGCTTGAAT	0.343	CCTCTCAAAG G ACAGCCATGC	1.423
TCCAGAATAC G TGACTCACGG	0.344	GGTCCTCTGG G CTTCTCTCCT	1.423
TTTCCTTTAT T CTCATTACAC	0.348	GCATCTGCCG C CACCACTTCT	1.421
AGCCAACTAA G CCCTCCTGGT	0.352	AATGGCTGAG C ATGGACCATG	1.406
TTTATGACTT T CTTTGTTCTG	0.356	TCCTGTCTCC C CCTATACCTG	1.398
CAGAAAATAC G TGCTGCACTT	0.362	GTGACTCACG G TCTACAACAA	1.391
TTTCCTATAC A ATGTATTCAT	0.369	TTCTGTAATG T CACCAAGGAG	1.388
TTTAGACTAT G TTCACTGTGA	0.375	CATTACAACT T TCAGGATTGT	1.385
GGTACCGACA G GTTCCTCTTC	0.376	TCCCCATGTT T TTGAGTAATA	1.382
CCATCCATAC T CATGTACCAA	0.377	TATGTATGTG T TGAATCACTA	1.366
CCAAGACAAT G CTGAAAAGGA	0.380	CCTTGTTGGA T CACTTGTACC	1.364
AATGTGATAC T CGCCCATCCA	0.386	TTTTTAAAAA A AGAAAGAAGA	1.359
TTTTAAATAT T CACAGCTAAG	0.387	CTTCCCATTG T TGAACATTTC	1.358
CAAATATTAT T CACTTTCCAG	0.391	TGGTTGTATT T TGTTGTAGAC	1.358

Table 7.7: Table of 30 highest and lowest intensities with their contexts. Shown are positions −10 to 10.

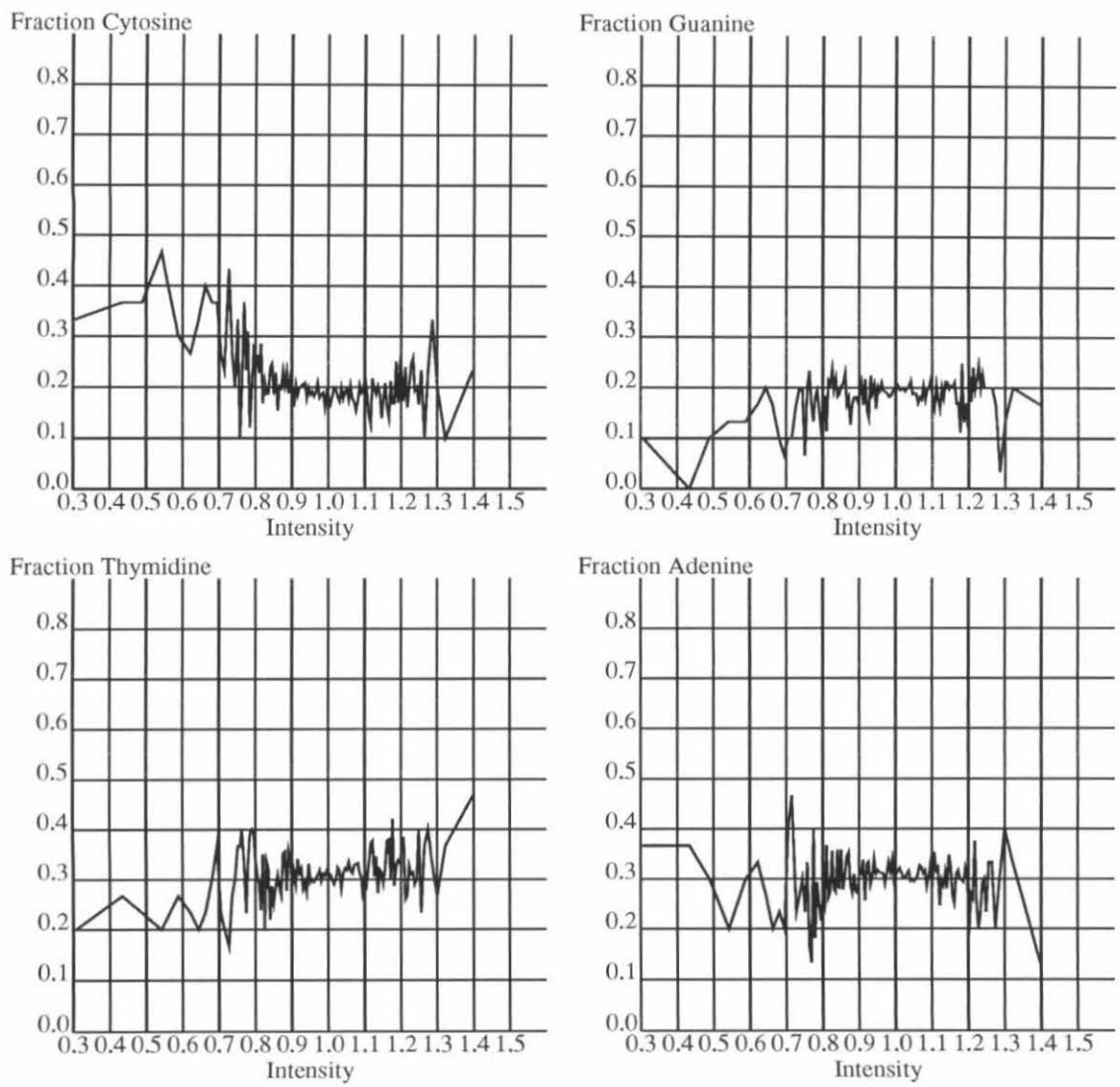


Figure 7.3: Position -7

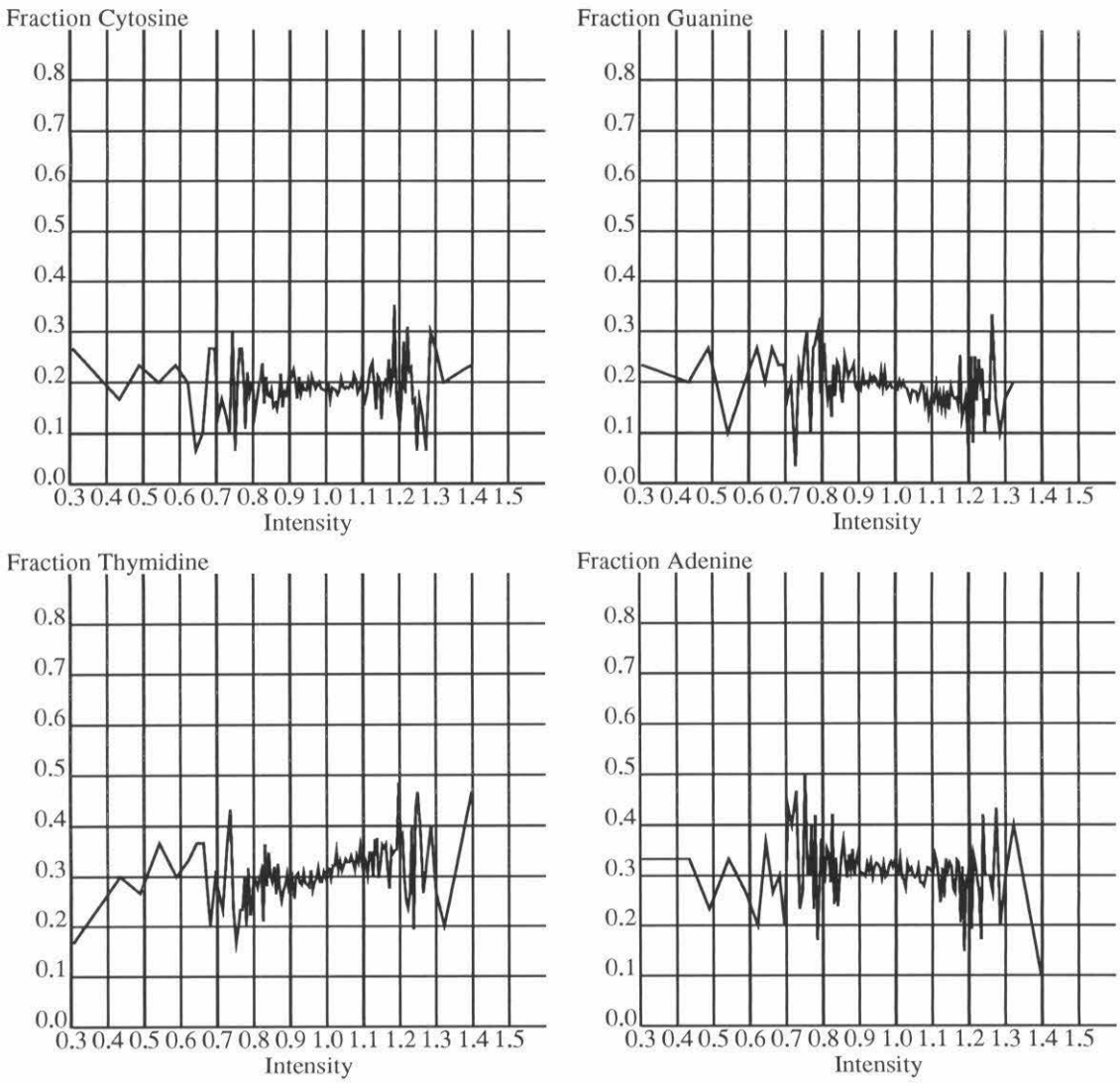


Figure 7.4: Position -6

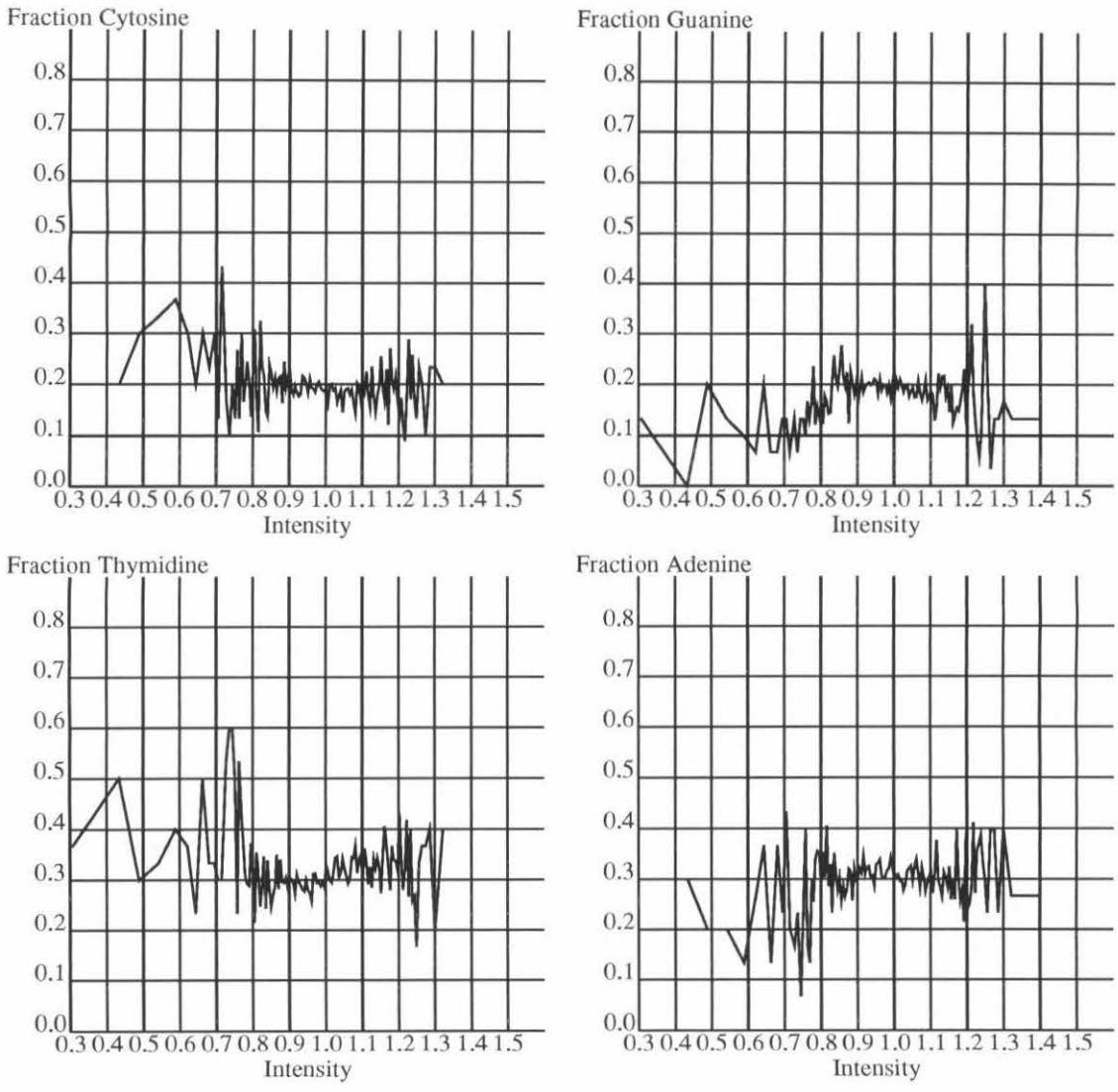


Figure 7.5: Position -5

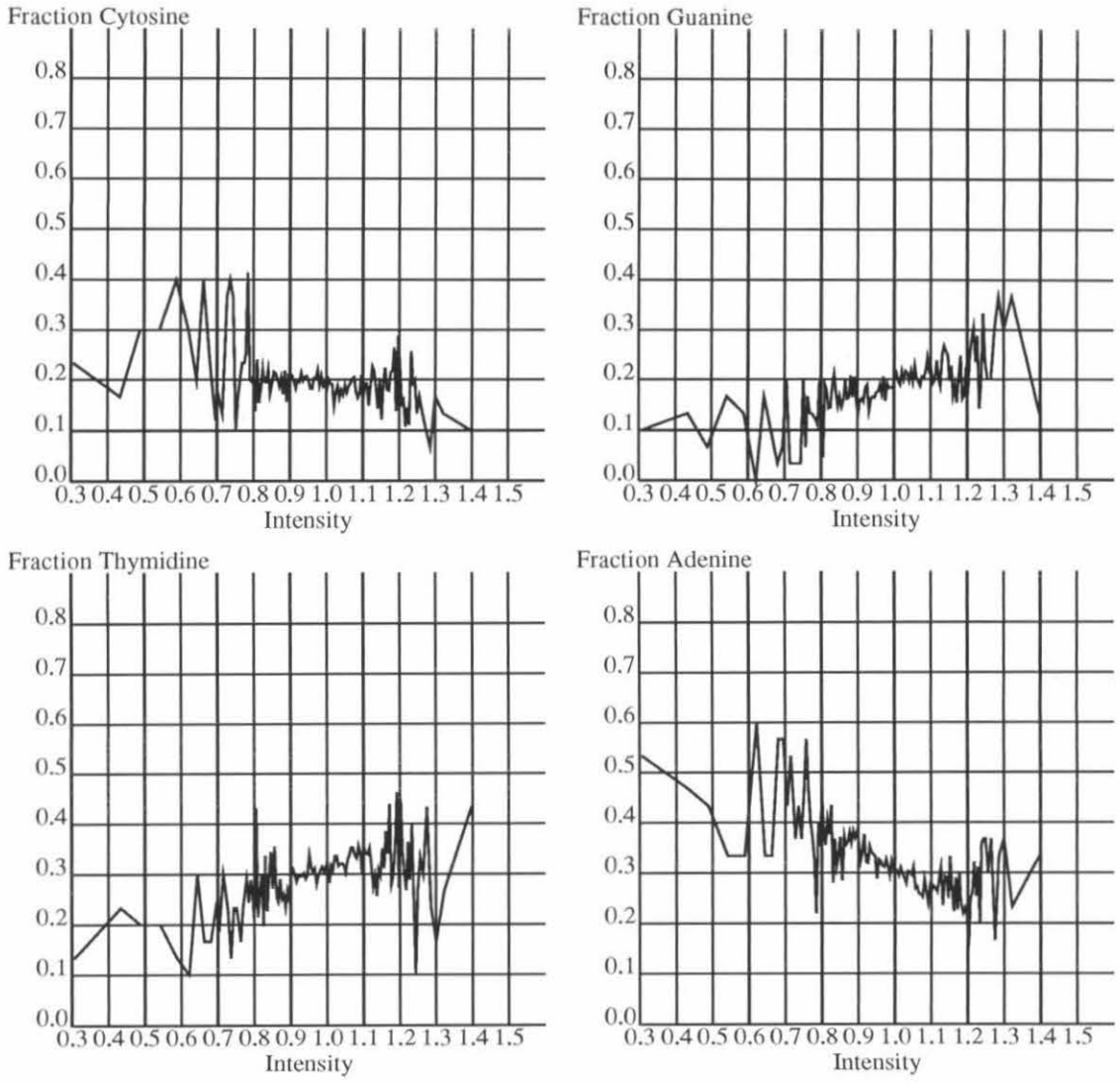


Figure 7.6: Position -4

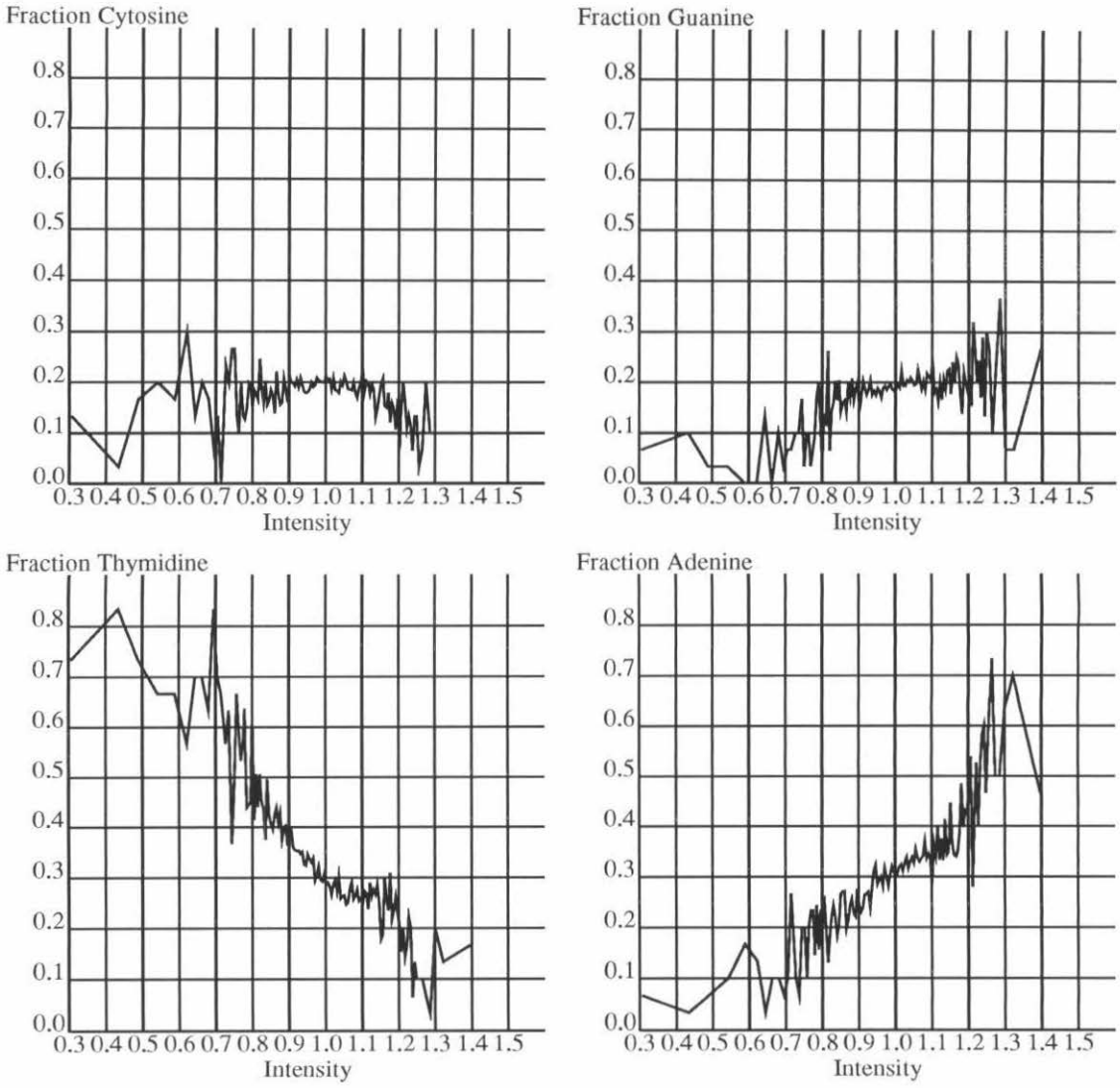


Figure 7.7: Position -3

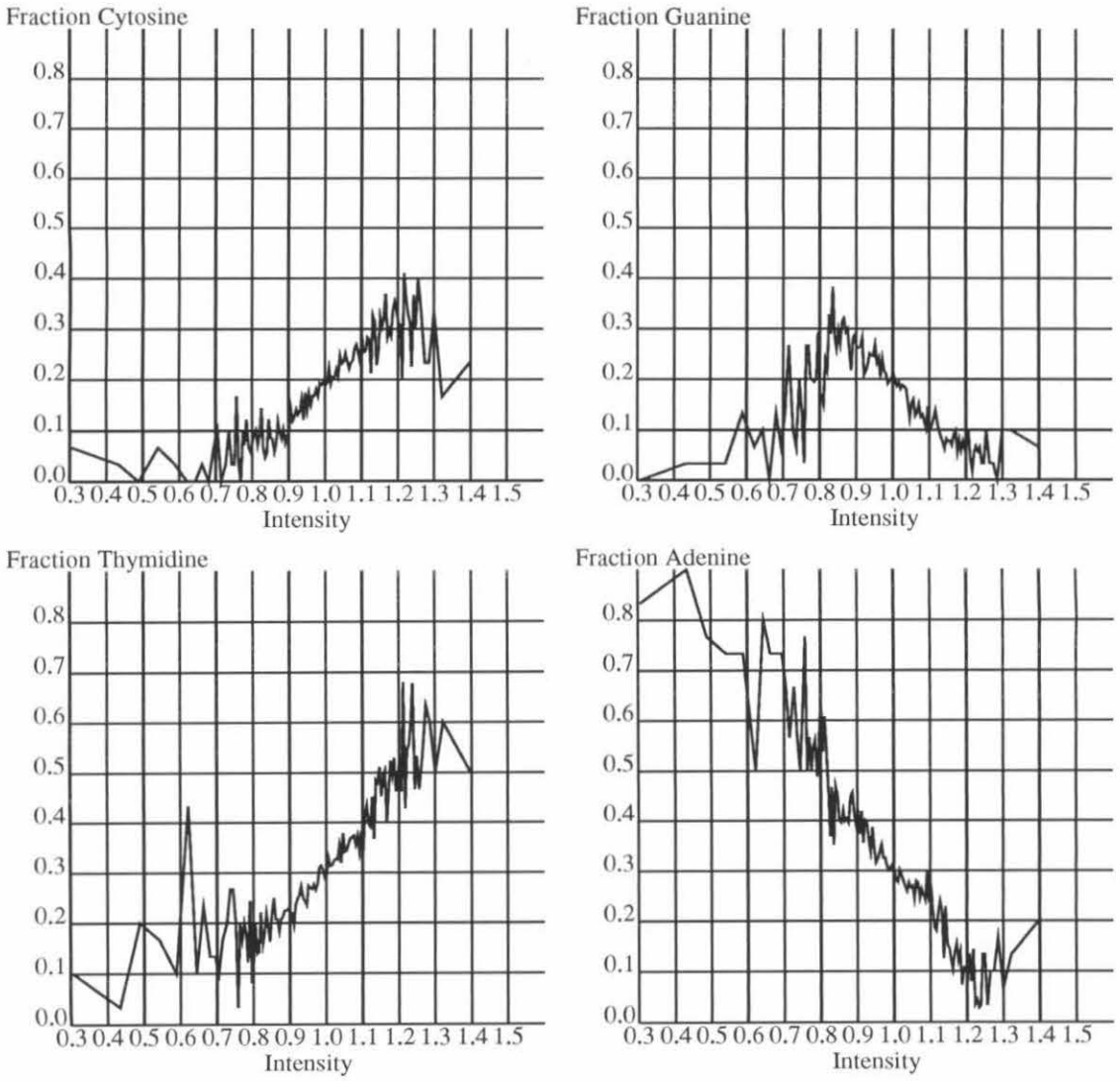


Figure 7.8: Position -2

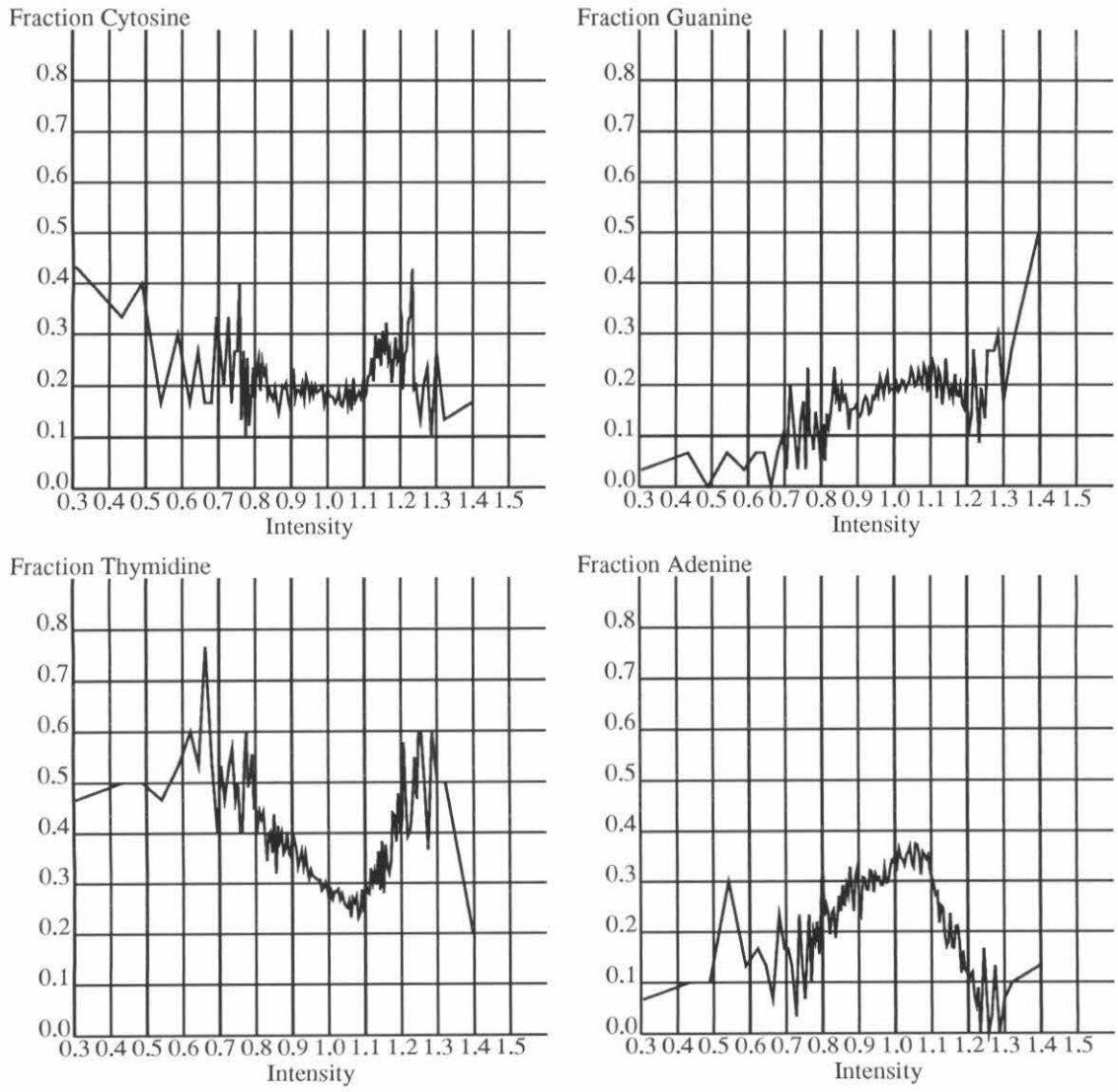


Figure 7.9: Position -1

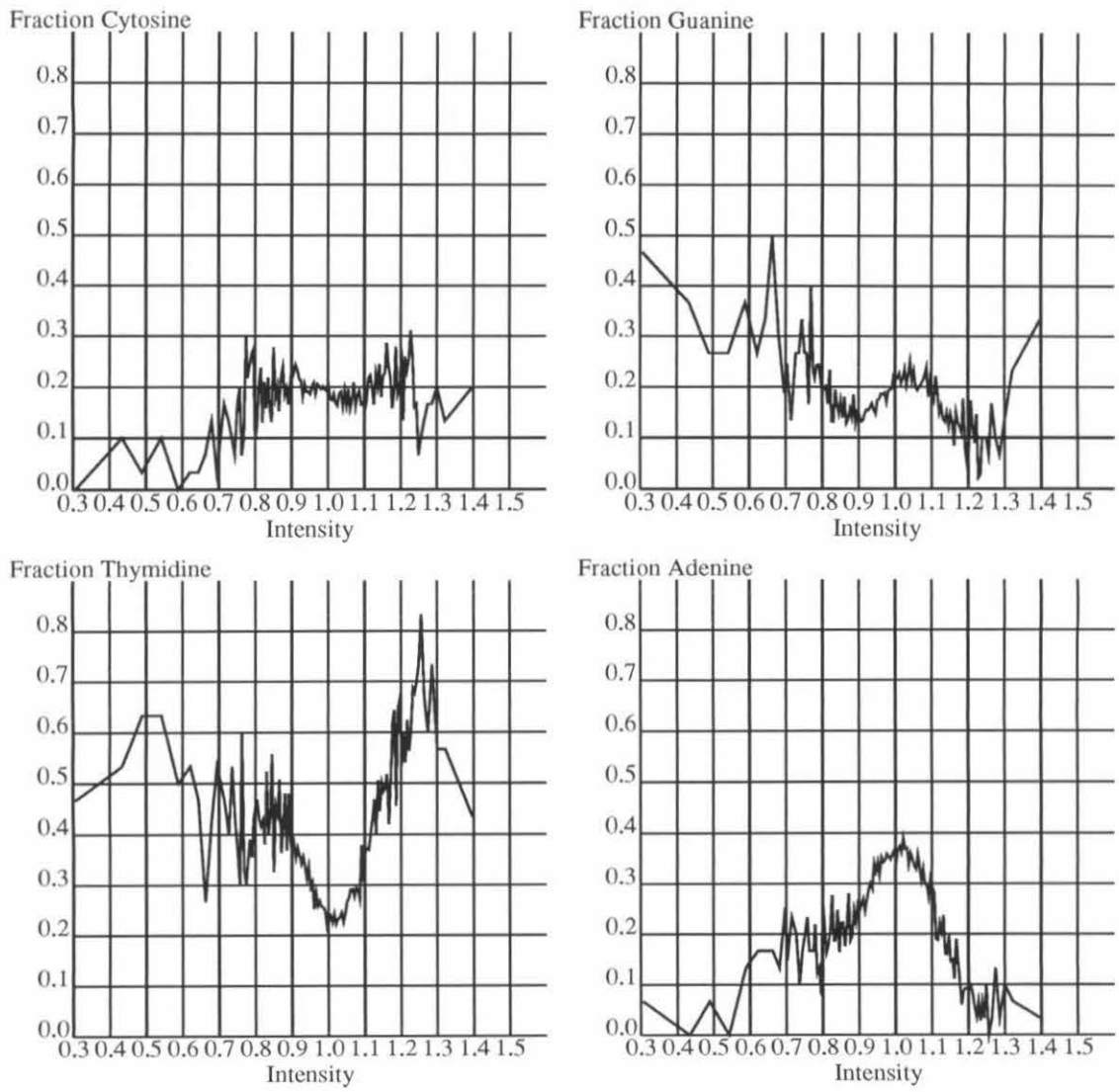


Figure 7.10: Position 0

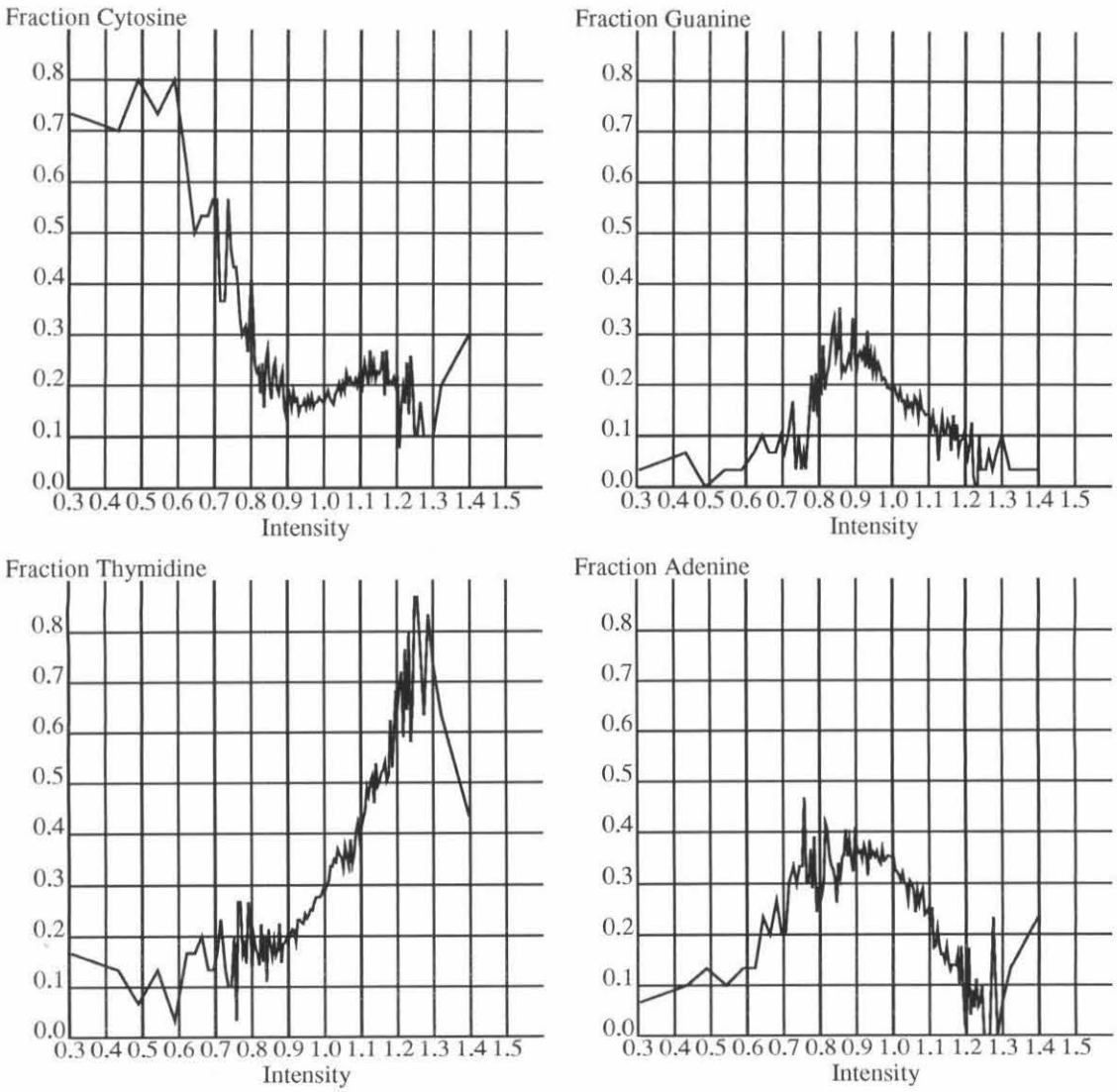


Figure 7.11: Position 1

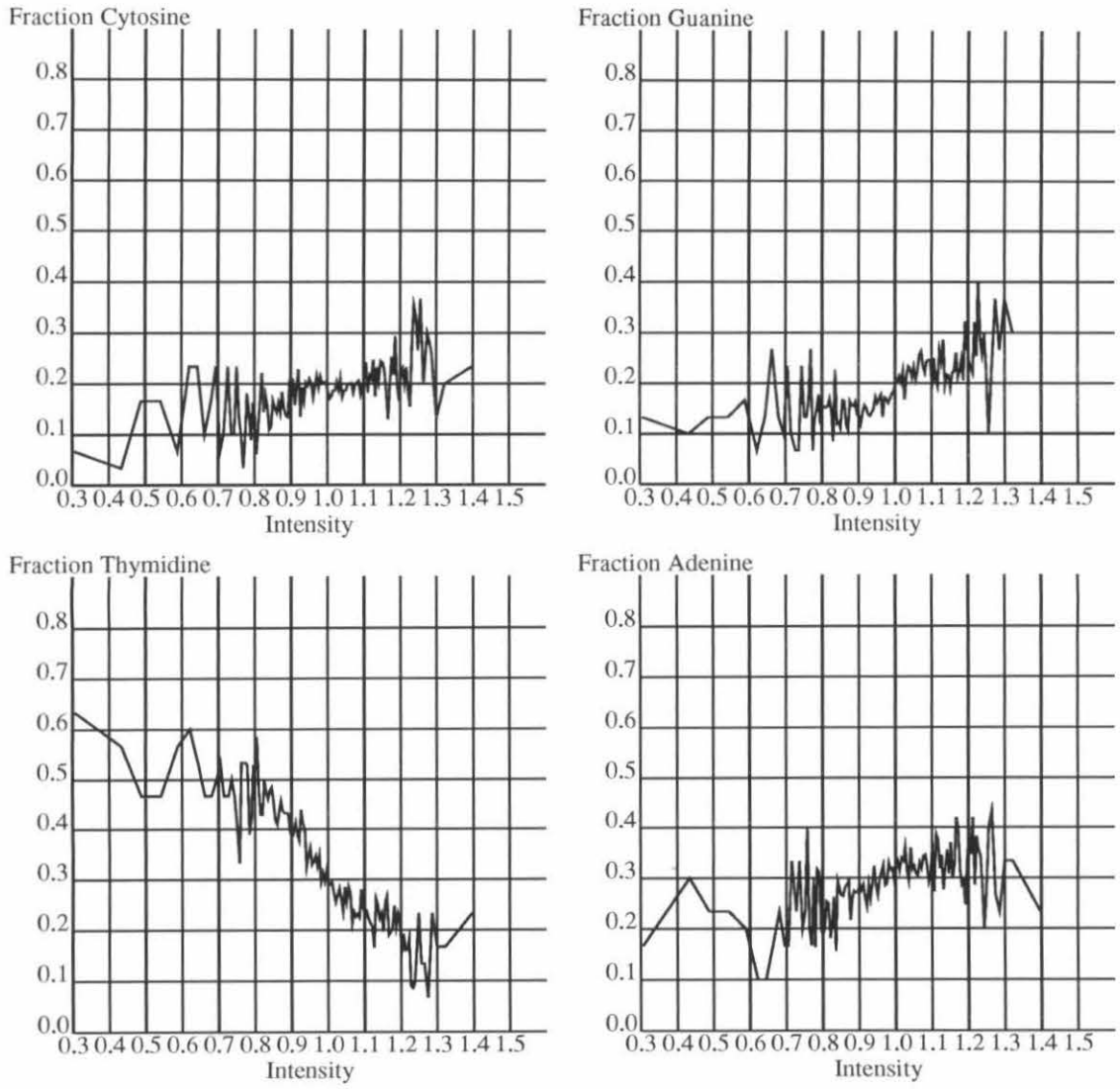


Figure 7.12: Position 2

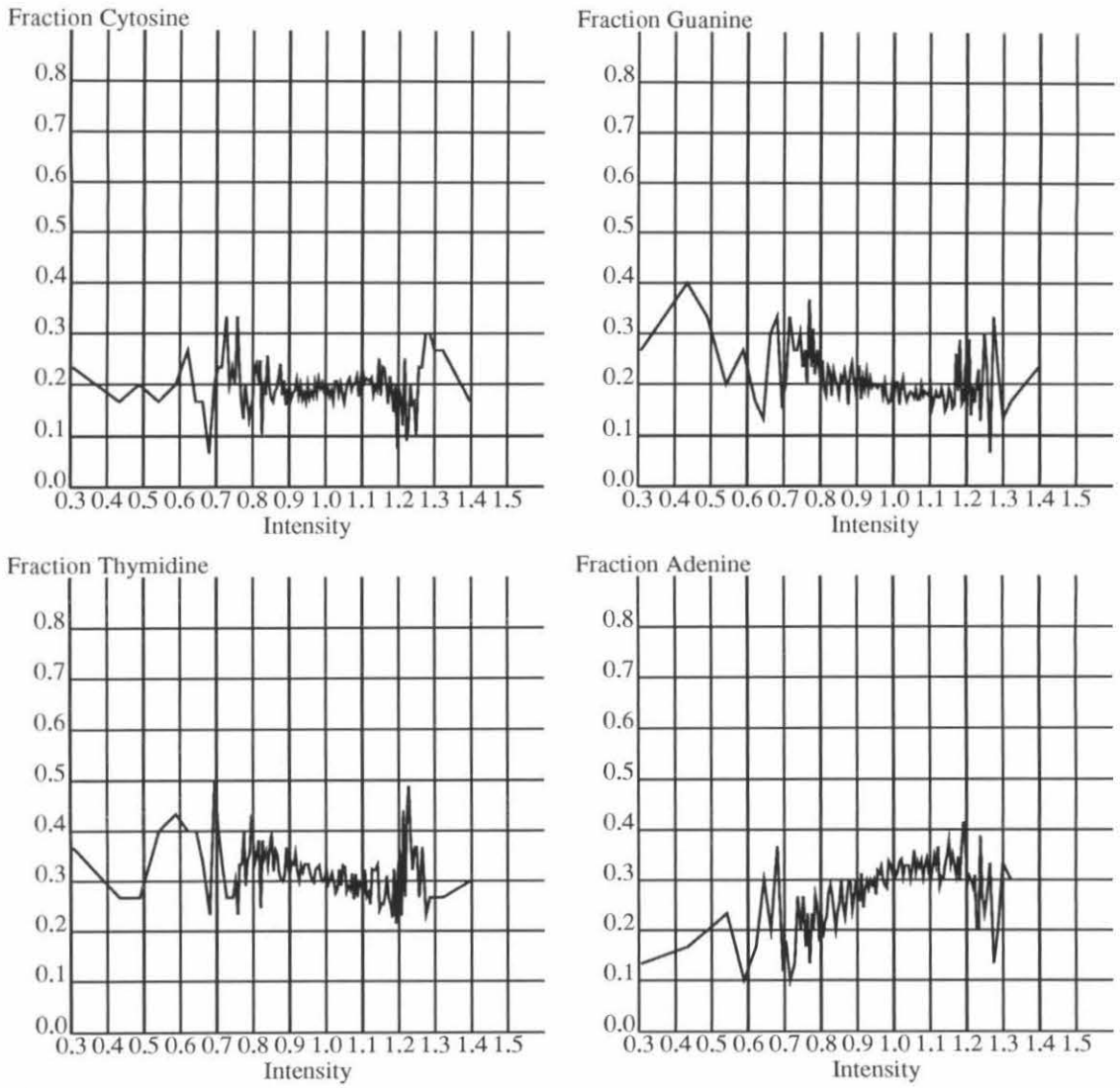


Figure 7.13: Position 3

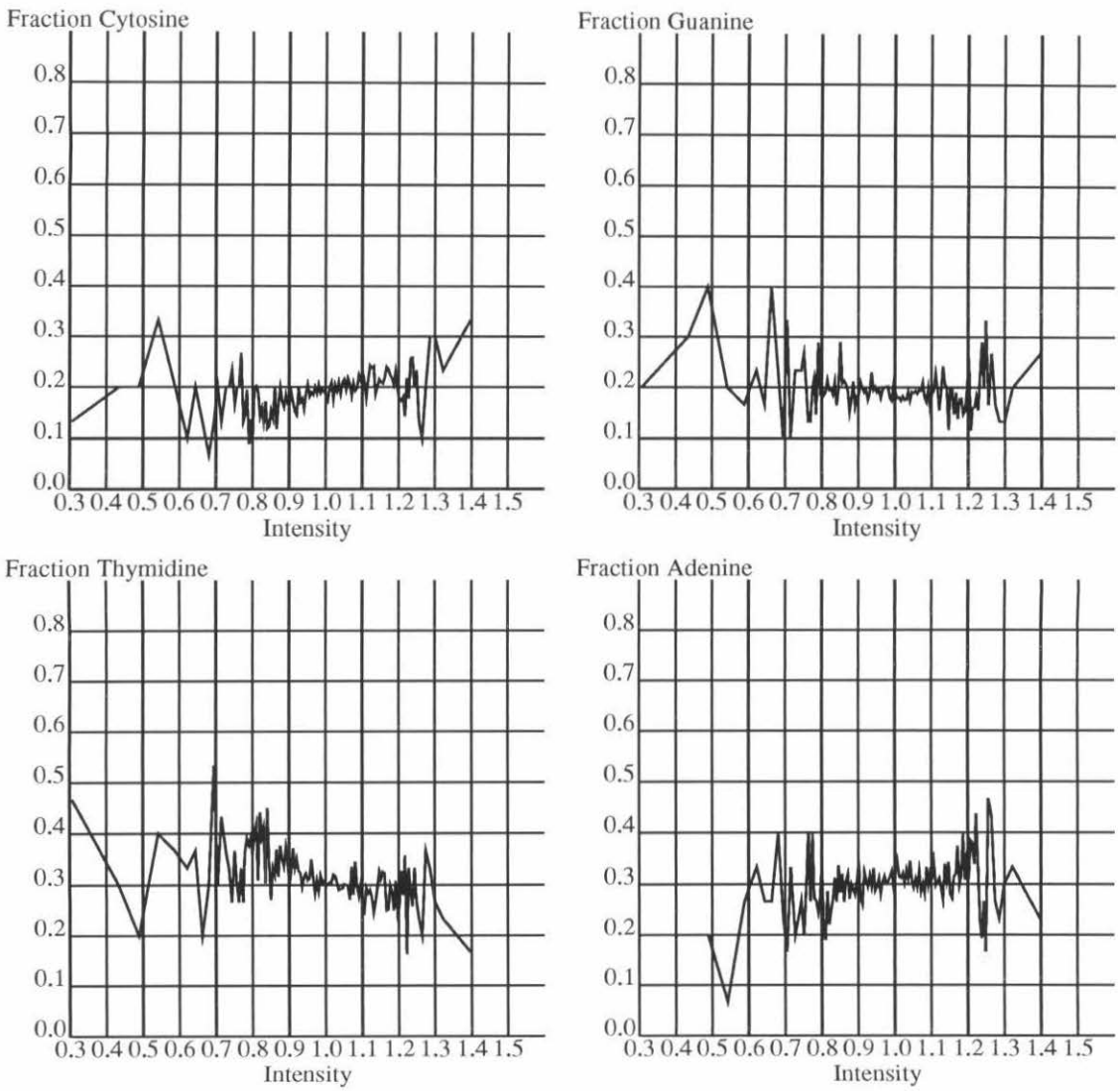


Figure 7.14: Position 4

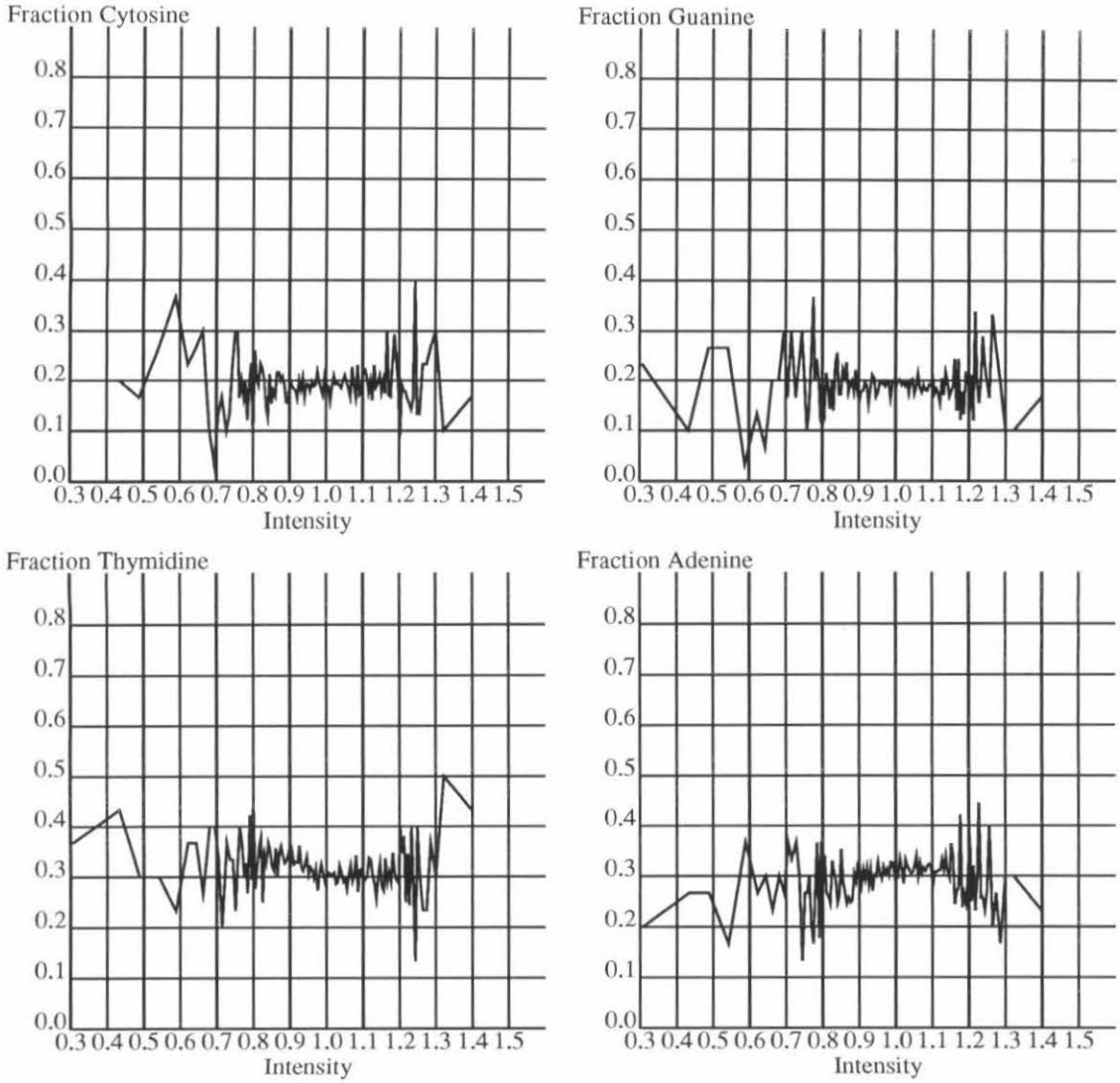


Figure 7.15: Position 5

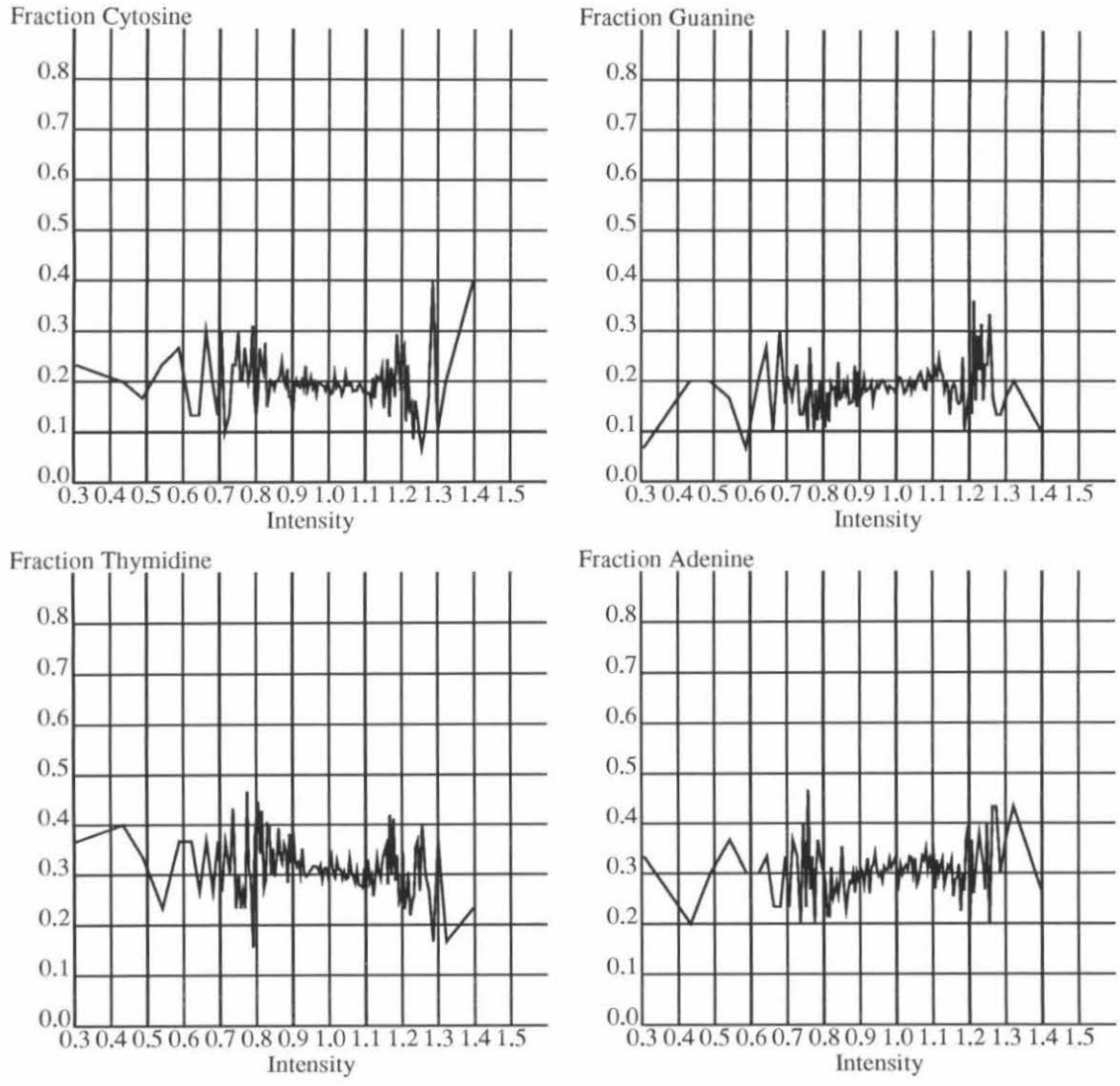


Figure 7.16: Position 6

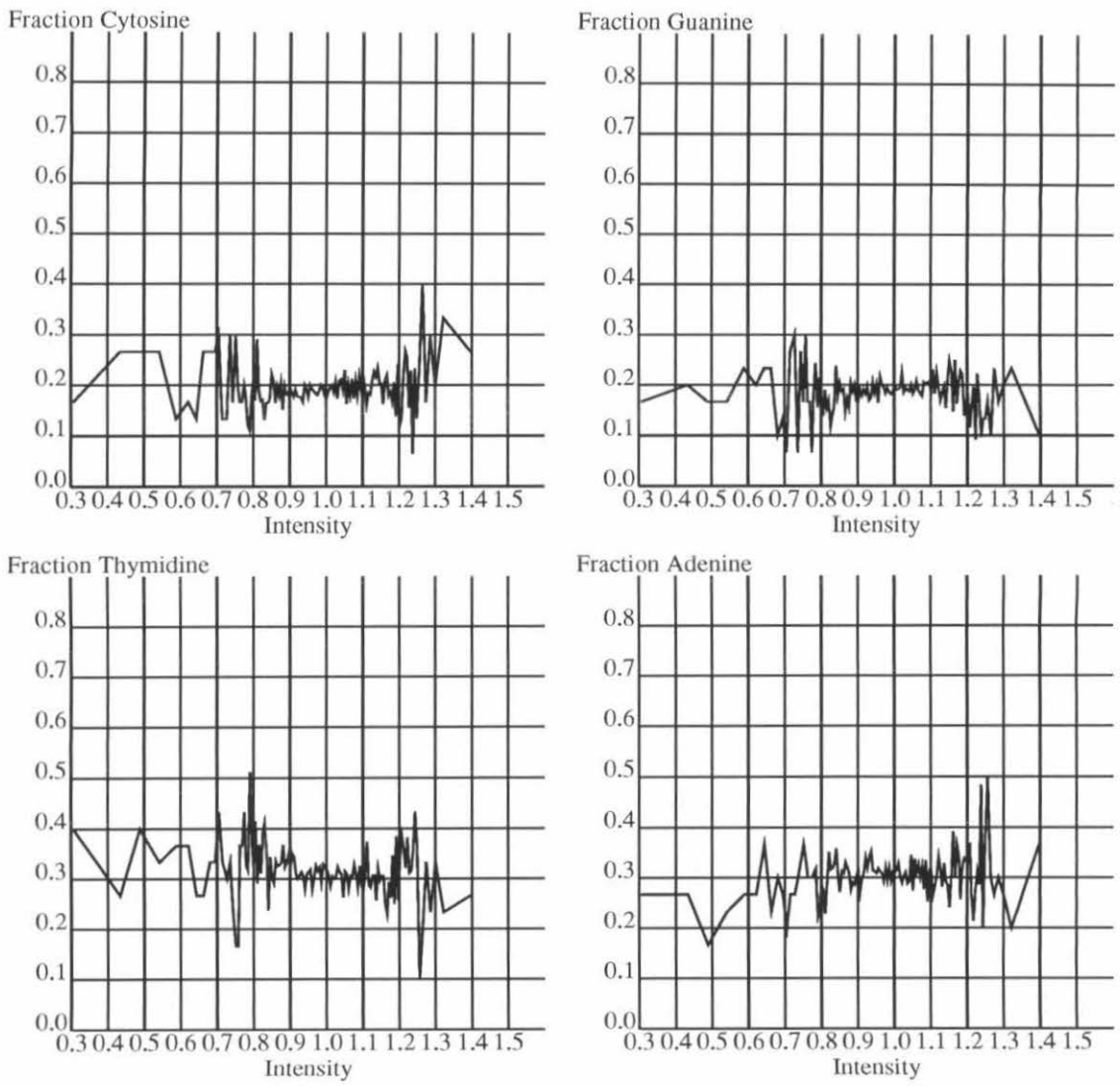


Figure 7.17: Position 7

Chapter 8

Conclusions

The observation that sequence affects the ability of a polymerase to discriminate between deoxy- and dideoxy-nucleotides, first alluded to by Sanger, has been shown to be a real, quantifiable effect. The important base positions have been shown to be those near the incorporation site, particularly positions -3 to 1. Positions further away than 4 bases have no discernible effect. Bases both $5'$ and $3'$ of the incorporation site are influential with the effect decreasing faster with distance from the incorporation site in the $3'$ direction than in the $5'$ direction. The effect of particular nucleotide depends on which position it occupies; a T at position -3 has an effect opposite to that of a T at position -2. The effect is not completely separable; a T at position -1 has opposite effects depending on what the rest of the sequence is.

The mechanisms whereby the base sequence of DNA affects the ability of T7 polymerase to discriminate between deoxy- and dideoxy-nucleotides remain unknown. A full explanation awaits further data on the three-dimensional structure of the enzyme and how it interacts with its template DNA. Any model of this effect will be able to be judged by how well it can reproduce the data presented in this work.

It is interesting to note that *in vivo* a polymerase is presumably surrounded by adenosine triphosphate (ATP), thus making it essential to discriminate between ATP and dATP, two molecules which differ only by a single Oxygen atom at the 2′ position. Is the same mechanism responsible for the discrimination between dATP and ddATP, two molecules which differ by a single Oxygen atom at the 3′ position?

Kristensen [Kristensen et al. 1988] suggests that knowledge of how sequence affects peak height might be useful in calling bases. A glance at Figure 7.1 shows that only a minute fraction of peaks are less than half normal height. The utility of this approach is questionable. Recent work by D. Seto [Seto 1992] suggests that sequence effects can be substantially mitigated by the addition of 10% Dimethyl Sulfoxide to the sequencing reactions.

Appendix A

Table of 5-mer Contexts

The following is a table of all 5-mer contexts of bases -3 to 1, i.e., the most influential 5 bases. Each 5-mer is listed with the average normalized peak height over all occurrences of that 5-mer context in the 110,549 fragment base data set. There are 17 5-mers for which there was no data (*N.D.*), they all contain at least one occurrence of CG, which is known to be an under-represented 2-mer in vertebrate DNA. It should be remembered that this data is rather noisy, the standard deviation on entries in this table averages ≈ 0.079 (see Table 7.4).

CGCAC	*N.D.*	CACTC	0.849	TGTCA	0.900	TAATT	0.921	GAATA	0.934
CGCGA	*N.D.*	TATGT	0.850	GGATA	0.902	GGGAG	0.921	CCATG	0.934
CGCCG	*N.D.*	TACTA	0.856	CACGT	0.902	AATCG	0.921	TGCGG	0.935
TGCGC	*N.D.*	GGACG	0.860	TGATG	0.902	AAATC	0.922	AATTC	0.935
CGTCG	*N.D.*	CTTGC	0.861	TAATG	0.903	GGCGG	0.922	CTTAA	0.935
ATCGG	*N.D.*	TAAGC	0.862	GGTCT	0.904	GGATT	0.922	TGTAG	0.935
GTGCG	*N.D.*	GACGA	0.865	GACGG	0.904	AGGTA	0.922	GAATG	0.935
TCGCG	*N.D.*	CGTCT	0.868	GGCGA	0.905	AGATG	0.923	GGTCC	0.936
CGCGC	*N.D.*	GACTC	0.868	GACTA	0.905	TATTA	0.923	AATCT	0.936
CCGAT	*N.D.*	CGTCA	0.868	AGTCA	0.905	TTTAA	0.923	AAGCG	0.937
CGACG	*N.D.*	TAATA	0.869	CCTCG	0.905	GGCGT	0.924	GGCTG	0.937
GCGCG	*N.D.*	GGGTG	0.870	AACTC	0.905	TTATG	0.924	TTTCG	0.937
GTCGT	*N.D.*	TGGTG	0.873	AGGTC	0.907	AATCC	0.924	GGTAG	0.938
CGGCG	*N.D.*	TATCA	0.874	CCCGC	0.908	GGGCT	0.925	TGCTC	0.938
CGCGG	*N.D.*	CTTTC	0.875	TAGTC	0.909	CATCA	0.925	AAGTG	0.939
CCGCG	*N.D.*	GAATC	0.878	AGTGC	0.910	CTTGT	0.925	TTATA	0.939
GCGCA	*N.D.*	TATAA	0.878	TATAC	0.910	TACAG	0.926	TCTGC	0.939
TACGC	0.494	AGTCG	0.879	CACTG	0.910	CGTCC	0.926	GATTC	0.939
TACTC	0.541	GGTCA	0.879	TGATA	0.911	TTTTC	0.926	GGGAT	0.939
TACGA	0.571	AATGC	0.880	TACGG	0.911	CTTCA	0.926	CCTCA	0.940
GACGC	0.590	CTCGC	0.881	TCATG	0.911	GGGCG	0.926	GGTAC	0.940
TAATC	0.640	TACTG	0.882	CTTCC	0.913	AAACG	0.927	TGTCC	0.941
TACGT	0.666	CCGGG	0.882	AGGTG	0.913	TAACA	0.927	AGGAG	0.941
TATTC	0.697	CCGTG	0.883	CGCTG	0.914	CTTCG	0.928	TGGAG	0.941
TATGC	0.699	CGATG	0.884	GGATG	0.915	GGACT	0.929	GGACA	0.941
AACGC	0.712	GGGTA	0.886	GGGTT	0.915	TCATC	0.929	GATCA	0.942
TACAC	0.779	AGCGT	0.889	TTTGC	0.915	TGTCT	0.929	AAATG	0.942
GCGGT	0.791	AACGT	0.890	TGGTA	0.916	ACGAG	0.929	TCATA	0.942
CACGC	0.798	TACAT	0.891	CGCTC	0.916	TGGTC	0.929	CGACA	0.942
CACGA	0.814	CTTGA	0.893	TTTAG	0.917	CCGTA	0.929	CTTAC	0.943
AACGA	0.821	TATCG	0.893	CACTA	0.917	TTTCA	0.930	TTCAG	0.943
TATGA	0.823	ACGTG	0.894	TGTTC	0.918	AACGG	0.932	GGGAA	0.943
GGTCG	0.826	AATCA	0.895	AGATA	0.918	TAACG	0.932	GGCTC	0.943
TATCC	0.831	GATGC	0.896	AGCGA	0.918	CGATC	0.932	CTATG	0.943
GGCGC	0.840	CCGCA	0.896	AATGA	0.919	GGGCA	0.933	GACGT	0.943
TACTT	0.841	CATGC	0.897	CGTAG	0.919	TGTCG	0.933	GACTG	0.944
CGGTG	0.841	CGCAG	0.897	CATTC	0.920	TCCGG	0.933	AGTTC	0.944
TGCGT	0.843	TAACC	0.897	CGGAT	0.920	GGTGA	0.933	AGCTC	0.944
TACAA	0.845	GGGTC	0.897	CAATC	0.920	CATCC	0.933	TCCGT	0.944
AGCGC	0.849	CGGTC	0.900	CCCGA	0.921	TACCC	0.934	TCCGC	0.944

CGAAG	0.945	TATCT	0.953	CCCTA	0.960	GCATG	0.967	CAGAG	0.975
AGTCC	0.945	CAATG	0.953	TTTGT	0.960	GTCGC	0.967	CCACT	0.975
TATTG	0.945	TGCTG	0.953	AGGCT	0.960	GTCAG	0.967	TGCAA	0.975
TGTGC	0.946	AGATT	0.954	TGTAA	0.961	CGACT	0.967	GGTGT	0.976
TTGTG	0.946	GCACG	0.954	AGACA	0.961	AGATC	0.967	CAAGC	0.976
ATTCG	0.947	AAACA	0.954	CTGAG	0.961	AACAA	0.968	CACAT	0.976
AACTA	0.947	GGCAG	0.954	CAATA	0.961	TCCTC	0.968	ATTAG	0.976
TGATT	0.947	CTACA	0.954	CGCTA	0.962	CCTGA	0.968	TGGAC	0.976
TTTAC	0.947	TGACA	0.954	TTATC	0.962	GGTGG	0.968	AGGAA	0.976
TAAGA	0.947	ACATG	0.954	TGATC	0.962	GTACA	0.969	TCCTG	0.976
GGCTA	0.948	TGTAC	0.954	CGGCT	0.962	TTCGC	0.970	TATTT	0.976
GATCC	0.948	TATAG	0.955	CTTAG	0.963	AGGAC	0.970	TCTAG	0.976
CATCG	0.948	AGTGA	0.955	GTATG	0.963	CACCG	0.970	GATCT	0.977
CATGT	0.948	GAACA	0.955	ATTGC	0.963	AGCTA	0.970	CCGGT	0.977
ATGAG	0.949	AGGCA	0.955	TGACT	0.963	GTGAG	0.970	TATAT	0.977
AATGT	0.949	CATGA	0.955	CCCTG	0.963	TCATT	0.971	CGAAC	0.977
TGCTA	0.949	TGGTT	0.955	CAGCA	0.963	ACCGT	0.971	AATGG	0.977
TATGG	0.949	GACAG	0.956	CGGTT	0.964	GAGTG	0.971	CTCGG	0.977
CCGAA	0.950	GGTGC	0.956	CAACA	0.964	AGTAC	0.971	GTTAG	0.977
CAGCG	0.950	GAGTA	0.956	GCGTC	0.964	TTACA	0.971	CATGG	0.977
GATGA	0.950	TTAAG	0.956	AGGCG	0.964	ATTCA	0.971	GAGCA	0.977
AGTCT	0.950	AAGAG	0.957	GACTT	0.964	CATAA	0.972	GCGTG	0.977
TTTGA	0.950	GTTCG	0.957	CATCT	0.964	GGTAA	0.972	GGGGC	0.977
GCGAG	0.950	TCTGA	0.958	ATGCG	0.964	AACAC	0.972	GAACG	0.977
AACTG	0.951	TGGCA	0.958	TCCTA	0.964	GAGTC	0.972	CGTTG	0.977
AAGTA	0.951	TGTTG	0.958	GCATA	0.964	GACAA	0.972	AGACT	0.978
ACGCG	0.951	AGCAG	0.958	CCCGG	0.964	CAACG	0.972	CAGTG	0.978
GACAC	0.951	CCATC	0.958	TTCGT	0.964	TCTCG	0.972	TTCTG	0.978
GAGCG	0.951	CACAC	0.959	TTCGA	0.964	TGAAG	0.973	AAACC	0.978
CATAG	0.951	TAAGT	0.959	AGCTG	0.965	GTTAA	0.973	TAGAG	0.979
TGCGA	0.951	TCGTA	0.959	CGATA	0.965	CACTT	0.973	GGGGT	0.979
TGACG	0.952	TCGTG	0.959	GATAA	0.965	CGTAT	0.973	TGTGA	0.979
AAATA	0.952	GGGCC	0.959	TTTAT	0.965	TGGAA	0.974	GCATC	0.979
TCGTC	0.952	CGTGC	0.959	GCTCG	0.965	AGTAA	0.974	TTTCT	0.979
CTCAG	0.952	TGCAG	0.959	ACCGA	0.965	TCTCA	0.974	CTATA	0.979
ACGAT	0.952	AAGCA	0.959	TACCA	0.965	TTGAG	0.974	CGTAC	0.980
CGGAA	0.952	AACAG	0.959	TCACG	0.965	AGAAG	0.974	ACGAA	0.980
CGGCA	0.953	CCACA	0.959	TAACT	0.966	TCAGA	0.974	TAGTA	0.980
CACAG	0.953	GGGAC	0.960	AGTGT	0.966	TGGAT	0.974	CCCTC	0.980
TAGTG	0.953	CACAA	0.960	CCACG	0.967	CCATA	0.974	TTTCC	0.980

TGCAC	0.980	CCAGT	0.985	GCGGG	0.991	CTAGG	0.995	TCGAT	1.001
TAGCA	0.980	CCTGT	0.986	TCCAG	0.991	CAAGA	0.995	ACTGA	1.001
AAGAA	0.981	AACCG	0.986	CTGAA	0.991	CAGAA	0.995	TTCTC	1.001
TGTTA	0.981	CTACT	0.986	ACGGA	0.991	CTTAT	0.995	GGAGG	1.001
GATGT	0.981	ATCGC	0.986	TGACC	0.991	GCTAG	0.995	GGAGC	1.001
CGACC	0.981	CCAGA	0.986	CGTGA	0.991	TCTGG	0.995	CAGTC	1.002
GATTA	0.981	GGTTC	0.986	AGTGG	0.991	CTGTG	0.995	TCACA	1.002
GGTTA	0.981	GATAG	0.986	GGTAT	0.991	CTGAC	0.995	ATACG	1.002
CGTTC	0.981	CCGAG	0.987	CTATC	0.991	TGCCA	0.995	AGCCA	1.002
TTGCG	0.981	TAGCG	0.987	AGCAC	0.992	CGCGT	0.996	GTTAC	1.002
TTCAA	0.981	GCGTA	0.987	GAGAA	0.992	TGGCG	0.996	TTAGA	1.002
CGGCC	0.981	CTCAA	0.988	ACCGG	0.992	GAGAC	0.996	CGAAT	1.002
CAACC	0.982	CGGAG	0.988	AAGCT	0.992	CGTAA	0.997	CTGCA	1.002
ATATG	0.982	GCCGA	0.988	TCGGT	0.992	TCAGC	0.997	TCGCT	1.003
AAACT	0.982	GTATA	0.988	AAGTC	0.992	TCTAA	0.997	TTGTA	1.003
AGGTT	0.982	AATAC	0.988	GATAC	0.992	CATAC	0.997	TTAAT	1.003
GAATT	0.982	GAGAG	0.988	GCGAC	0.992	TAGAA	0.998	TTACT	1.003
CTTGG	0.982	GATCG	0.988	ATTGA	0.993	GGAGA	0.998	TTGAA	1.003
AGTAG	0.982	CTCAC	0.988	TCTTC	0.993	GGCAC	0.998	CTCGA	1.003
ATCAG	0.982	ACTCG	0.989	AACAT	0.993	CTACG	0.998	GTATC	1.003
ACTAG	0.982	GAACT	0.989	GGGGA	0.993	AAGAC	0.998	CTCGT	1.003
GCCAG	0.983	CGATT	0.989	GAGCT	0.993	CGTGT	0.998	GTAGA	1.004
GTTCA	0.983	AGACG	0.989	GTTGC	0.993	AAATT	0.998	CCACC	1.004
AACTT	0.983	AGCAA	0.989	ATCAA	0.993	CGCAA	0.998	AGTTA	1.004
GGAAG	0.983	CCTAG	0.989	GCCTA	0.993	TCGCA	0.998	AGCAT	1.004
GGCAA	0.983	ACGTA	0.989	GAGCC	0.994	TTATT	0.999	AGGAT	1.004
CGCCT	0.983	GGAGT	0.989	GACAT	0.994	GTGAA	0.999	CTACC	1.004
GGTTG	0.983	CACGG	0.989	ACATA	0.994	CCGCT	0.999	GTCAA	1.004
AATAA	0.984	ACGCT	0.990	AAAGC	0.994	TCCGA	0.999	ACAGA	1.005
AATAG	0.984	CCTGC	0.990	AAAGT	0.994	ATGAA	0.999	CGGAC	1.005
GGACC	0.984	CTCTA	0.990	TCGGA	0.994	GACCA	0.999	CAGTA	1.005
CTCTG	0.984	TTCTA	0.990	TCGAA	0.994	ATGAC	0.999	CAGAC	1.005
CTTCT	0.984	ATACA	0.990	AGACC	0.994	GTCTA	0.999	GAAGC	1.005
TGTAT	0.984	TTTGG	0.990	GCTCA	0.994	ACTCA	1.000	AAAAG	1.005
GCCTG	0.984	CATTA	0.990	TCTGT	0.994	CAAGT	1.000	TTGAC	1.005
GAACC	0.985	TTCAT	0.990	CATTG	0.995	GGCCA	1.000	ATGCA	1.005
TCAGT	0.985	ACACA	0.990	CCCAG	0.995	AAAGA	1.000	GTGTG	1.005
TTCAC	0.985	ATTAA	0.991	CCAGC	0.995	GATGG	1.000	GTGCA	1.006
GTACT	0.985	CACCA	0.991	CCCCA	0.995	ATAAG	1.001	CGGGA	1.006
CCGTC	0.985	ACCAG	0.991	AAGCC	0.995	TCTAC	1.001	GATTG	1.006

CCTCC	1.006	ACTGG	1.011	GTGAC	1.016	CACCC	1.021	GCTGC	1.028
TCTTG	1.006	TCAAG	1.011	GTAGT	1.016	AAAGG	1.021	TCTCT	1.028
TTAGT	1.006	GCTAA	1.011	GTTGT	1.016	TCTTA	1.021	AGGGA	1.029
CCTCT	1.006	AGAGA	1.011	ACGCA	1.016	CCATT	1.021	GCCGT	1.029
TTTTA	1.007	CTCTC	1.011	GTTGA	1.017	ACAGC	1.022	ATAGT	1.029
AACCA	1.007	ATCTG	1.011	ATAGA	1.017	GGAAC	1.022	TCGGC	1.029
CGGTA	1.007	GCAGA	1.011	GCAGT	1.017	CGGGT	1.022	CCTAC	1.029
CCAGG	1.007	ATCAC	1.012	ATTGT	1.017	GCAGG	1.022	CGAAA	1.029
GAGAT	1.007	AGTTG	1.012	ATCGA	1.017	CCGTT	1.022	CTCCA	1.029
GAAGA	1.007	CTGAT	1.012	TCCAC	1.017	TGCTT	1.022	CAAGG	1.030
GCATT	1.007	ACAGT	1.012	GCACA	1.017	CTTTG	1.022	AACCC	1.030
TAGAC	1.007	CAGCT	1.012	ACCGC	1.017	ACGCC	1.023	ATTCT	1.030
TCGAC	1.007	TGGCC	1.012	CCAAG	1.017	TCCCA	1.023	GTAGC	1.030
CATAT	1.008	GTAAG	1.012	ATTAT	1.017	GAAAG	1.023	TTAGG	1.031
CCGAC	1.008	ACGAC	1.013	TAGAT	1.018	GTTAT	1.023	GCCTC	1.031
TTTTG	1.008	TAGCC	1.013	CCTAA	1.018	ACTAA	1.024	GTACG	1.031
CGAGA	1.008	GCCGG	1.013	GATAT	1.018	CGTTA	1.024	AGAGG	1.031
AAGAT	1.008	GGCAT	1.013	TGAAT	1.018	GACCT	1.024	ATCGT	1.031
ACGGC	1.008	AGCCG	1.013	TTGCA	1.019	TGAAA	1.024	GTTGG	1.031
GTCTG	1.008	CTAGA	1.013	TGAGA	1.019	CCCCT	1.024	GCTGA	1.032
ATGAT	1.008	TAAGG	1.013	ACAGG	1.019	AGAGC	1.024	TTGTC	1.032
GTCAC	1.008	CTAAG	1.014	CCCTT	1.019	CGCTT	1.024	CTGCT	1.032
TTACG	1.008	AGAGT	1.014	CAATT	1.019	CCGGA	1.025	CAAAG	1.032
GGCTT	1.009	TAAAC	1.014	GAAGT	1.019	CGTGG	1.025	GTACC	1.032
CCTGG	1.009	GTCAT	1.014	GGAAT	1.019	GAAGG	1.025	ACGGT	1.032
ATTAC	1.009	AGGCC	1.015	GTTCC	1.019	TCCCG	1.025	GCCCG	1.033
TAGCT	1.009	GCCAA	1.015	CCCAA	1.019	GCCAC	1.025	CAGCC	1.033
TGCAT	1.009	CCCCG	1.015	GCGAT	1.019	CCCAC	1.025	GCTAT	1.033
TTGAT	1.009	GTCGA	1.015	GTAGG	1.020	TTAAA	1.026	CCCGT	1.033
TGTGT	1.009	ATGTG	1.015	GCTAC	1.020	GCAGC	1.026	ACCAA	1.033
GGAAA	1.009	TCAGG	1.015	AGTAT	1.020	ACTGC	1.026	CCTAT	1.034
ACATC	1.009	TAAAA	1.015	AGAAT	1.020	GTGAT	1.027	ACCTG	1.034
ATTGG	1.010	GGCCG	1.015	GTTCT	1.021	TCCAA	1.027	AGAAC	1.034
CAGAT	1.010	ACCTA	1.015	TAAAG	1.021	GCGGA	1.027	TGAGT	1.035
CTCAT	1.010	CTGTA	1.015	TACCT	1.021	GCAAG	1.027	CCCAT	1.036
ATACT	1.010	AATTA	1.015	AGAAA	1.021	CTATT	1.027	GCACT	1.036
CAACT	1.010	CTAGT	1.016	ACAAG	1.021	AGGGC	1.028	TGAGC	1.036
ATATA	1.011	CTGCG	1.016	TGTGG	1.021	TCTAT	1.028	GTCGG	1.036
TGGCT	1.011	ATCTA	1.016	GTCTC	1.021	ATCAT	1.028	TCGCC	1.036
AATTG	1.011	CTTTA	1.016	TCGAG	1.021	CACCT	1.028	AGGGT	1.036

AATAT	1.036	CGAGC	1.044	GTAAC	1.055	ATATT	1.065	CCGCC	1.080
GCTGG	1.037	AAAAT	1.044	TGTTT	1.055	AAGGT	1.066	CATTT	1.081
TTAGC	1.037	ACCAC	1.044	CCCCC	1.055	TGCCG	1.066	GAGGG	1.082
AGCTT	1.037	CGTTT	1.044	GCCAT	1.055	TGCCT	1.066	ATCCA	1.083
CCTTA	1.037	TCCAT	1.045	GTGCC	1.055	TACCG	1.067	AATTT	1.083
CGGGC	1.037	GCGCT	1.045	CAAAA	1.056	TTGGG	1.067	ACTCT	1.084
TTACC	1.038	GTGTA	1.045	GTGCT	1.056	CCAAC	1.067	AGTTT	1.085
TCACT	1.038	AAAAA	1.046	TTGGC	1.057	AAGGA	1.067	GGCCC	1.085
TGAAC	1.039	TCAAT	1.046	TGGGC	1.057	GTGGA	1.067	ATGGG	1.086
CCTTG	1.039	AAGTT	1.046	GTGGC	1.057	ACCCG	1.068	GAGGC	1.086
ACTGT	1.039	GCGAA	1.046	TTGCC	1.058	TTGGT	1.068	ATGGT	1.087
ACTAC	1.039	ATGCT	1.046	GCTCT	1.058	CCAAA	1.068	GAGGT	1.087
GGATC	1.039	CTAAT	1.047	CAAAC	1.058	GCTCC	1.068	ATCCG	1.089
TTAAC	1.039	GTTTC	1.047	CAAAT	1.058	ACTTG	1.069	GTGGG	1.089
ATAGC	1.039	CGCAT	1.048	CCAAT	1.059	CTCCG	1.069	ACTCC	1.090
TGAGG	1.039	TCACC	1.048	GTCCG	1.059	GCAAC	1.069	ACCTC	1.090
ATTCC	1.040	GAGGA	1.048	ATACC	1.059	TCGGG	1.070	ACACC	1.090
TTCGG	1.040	GACCG	1.048	CTAAC	1.059	TTGTT	1.070	TGCCC	1.092
ACACG	1.040	GACCC	1.049	CGAGG	1.059	CTGCC	1.070	CTGGG	1.092
ATAGG	1.041	GAGTT	1.050	ACCCA	1.059	CAGGT	1.070	ACGGG	1.093
AGCCT	1.041	TTCCA	1.050	ATGTA	1.060	TAGGC	1.070	ACGTC	1.093
CTGGA	1.041	TGGGA	1.050	GAAAC	1.060	CTGGC	1.071	GCGTT	1.093
ATAAT	1.041	GTAAT	1.051	GTTTA	1.061	ACAAA	1.071	GCTTC	1.094
CTGTC	1.041	CTGGT	1.051	AAAAC	1.061	ATTTA	1.073	CAGGG	1.094
GCGCC	1.042	GGGGG	1.051	TCAAA	1.061	CAGGC	1.073	AGCCC	1.095
GCTTG	1.042	GCTGT	1.051	GTGGT	1.061	GCACC	1.073	GTGTC	1.097
CTAAA	1.042	ACTAT	1.051	GAAAT	1.062	ATCTC	1.073	ACAAC	1.098
ATAAA	1.042	GGCCT	1.051	GGTTT	1.062	AAGGC	1.074	TCCCT	1.099
ACATT	1.042	CGCCC	1.051	TTGCT	1.062	ACCAT	1.075	TGGGG	1.099
GTCCA	1.043	CGAGT	1.051	TCGTT	1.062	TAGTT	1.076	GTCTT	1.099
GTATT	1.043	GCCGC	1.052	ACACT	1.063	ATGCC	1.076	GCGGC	1.101
TGGGT	1.043	TAGGT	1.052	GATTT	1.063	CTCTT	1.078	CTGTT	1.101
ACGTT	1.043	AACCT	1.052	ATTTC	1.063	TTCTT	1.078	CGCCA	1.102
CTAGC	1.043	ATATC	1.053	GCTTA	1.063	AGGGG	1.078	CTTTT	1.105
TTGGA	1.044	CCTTC	1.053	TCAAC	1.064	GCAAT	1.078	TAGGG	1.107
TAAAT	1.044	TCCTT	1.053	ACAAT	1.065	ACTTA	1.078	TCTTT	1.107
GAAAA	1.044	ATGGA	1.053	CAGTT	1.065	ATTTG	1.079	CTCCT	1.111
CAGGA	1.044	GCCCA	1.054	ATAAC	1.065	AAGGG	1.079	CTCCC	1.113
TCTCC	1.044	TAGGA	1.054	GTAAA	1.065	ATGGC	1.079	TTTTT	1.117
GTTTG	1.044	TTCCG	1.054	CGGGG	1.065	GCAAA	1.080	TCCCC	1.121

GCCTT	1.122	GTCCT	1.140	GTGTT	1.155	ATGTT	1.163	ACCCT	1.190
CCGGC	1.128	ACTTC	1.141	CCTTT	1.157	GCCCC	1.164	GTTTT	1.199
ATGTC	1.133	ATCTT	1.143	ACCTT	1.158	GCTTT	1.174	ATTTT	1.202
TTCCC	1.134	GCCCT	1.147	GTCCC	1.158	ACCCC	1.178	ACTTT	1.208
TTCCT	1.135	ATCCT	1.150	ATCCC	1.163	AGCGG	1.183		

References

[Bedinger et al. 1989] Bedinger, P., Munn, M., and Alberts, B. (1989). Sequence-specific pausing during *in vitro* DNA replication on double-stranded DNA templates. *J. Biol. Chem.*, 264(28):16880–16886.

[Bowling et al. 1991] Bowling, J. M., Bruner, K., Cmarik, J., and Tibbetts, C. (1991). Neighboring nucleotide interactions during DNA sequencing gel electrophoresis. *Nucleic Acids Research*, 19(11):3089–3097.

[Fry and Loeb 1992] Fry, M. and Loeb, L. (1992). A DNA polymerase α pause site is a hot spot for nucleotide misinsertion. *Proc. Natl. Acad. Sci. U.S.A.*, 89:763–767.

[Himawan and Richardson 1992] Himawan, J. and Richardson, C. (1992). Genetic analysis of the interaction between bacteriophage T7 DNA polymerase and *escherichia coli* thioredoxin. *Proc. Natl. Acad. Sci. U.S.A.*, 89:9774–9778.

[Hopfield 1974] Hopfield, J. (1974). Kinetic proofreading: A new mechanism for reducing errors in biosynthetic processes requiring high specificity. *Proc. Natl. Acad. Sci U.S.A.*, 71(10):4135–4139.

[Huber et al. 1987] Huber, H., Tabor, S., and Richardson, C. (1987). *Escherichia Coli* thioredoxin stabilizes complexes of bacteriophage T7 DNA polymerase and primed templates. *J. Biol. Chem.*, 262(33):16224–16232.

[Kaguni and Clayton 1982] Kaguni, L. and Clayton, D. (1982). Template-directed pausing in *in vitro* DNA synthesis by DNA polymerase α from *Drosophila Melangaster* embryos. *Proc. Natl. Acad. Sci. U.S.A.*, 79:983–987.

[Koop et al. 1990] Koop, B., Wilson, R., Chen, C., Halloran, N., Sciamas, R., Hood, L., and Linelian, J. (1990). Sequencing reactions in microtiter plates. *Biotechniques*, 9(1):32–37.

[Kristensen et al. 1988] Kristensen, T., Voss, H., Schwager, C., Stegemann, J., Sproat, B., and Ansorge, W. (1988). T7 DNA polymerase in dideoxy sequencing. *Nucleic Acids Research*, 16(8):3487–3496.

[Ollis et al. 1985] Ollis, D., Brick, P., Hamlin, R., Xuong, N., and Steitz, T. (1985). Structure of large fragment of *escherichia coli* DNA polymerase I complexed with dTMP. *Nature*, 313:762–766.

[Sanders et al. 1989] Sanders, J. Z., MacKellar, S., Otto, B., Dodd, C., Heiner, C., Hood, L., and Smith, L. (1989). Peak height variability and accuracy in automated DNA sequencing. In Sarma, R. and Sarma, M., editors, *Proceedings of the Sixth Conversation in Biomolecular Stereodynamics.* Adenine Press.

[Sanger et al. 1977] Sanger, F., Nicklen, S., and Coulson, A. (1977). DNA sequencing with chain terminating inhibitors. *Proc. Natl. Acad. Sci. USA*, 74:5463–5467.

[Seto 1990] Seto, D. (1990). An improved method for sequencing double stranded plasmid DNA from minipreps using DMSO and modified template preparation. *Nucleic Acids Research*, 18(19):5905.

[Seto 1992] Seto, D. (1992). Personal Communication.

[Seto et al. 1992] Seto, D., Koop, B., Seto, J., and Hood, L. (1992). Complete nucleotide sequence of the cosmid vector pWE15A. *Nucleic Acids Research*, 20(14):3786.

[Smith et al. 1986] Smith, L., Sanders, J., Kaiser, R., Hughs, P., Dodd, C., Connell, C., Heiner, C., Kent, S., and Hood, L. (1986). Fluorescence detection in automated DNA sequence analysis. *Nature*, 321:674–679.

[Tabor et al. 1987] Tabor, S., Huber, H., and Richardson, C. (1987). *Escherichia Coli* thioredoxin confers processivity on the DNA polymerase activity of the gene 5 protein of bacteriophage T7. *J. Biol. Chem.*, 262(33):16212–16223.

[Tabor and Richardson 1987a] Tabor, S. and Richardson, C. (1987a). DNA sequence analysis with a modified bacteriophage T7 DNA polymerase. *Proc. Natl. Acad. Sci. USA*, 84:4767–4771.

[Tabor and Richardson 1987b] Tabor, S. and Richardson, C. (1987b). Selective oxidation of the exonuclease domain of bacteriophage T7 DNA polymerase. *J. Biol. Chem.*, 262(32):15330–15333.

[Tabor and Richardson 1989] Tabor, S. and Richardson, C. (1989). Effect of manganese ions on the incorporation of dideoxynucleotides by bacteriophage T7 DNA polymerase and *Escherichia Coli* DNA polymerase I. *Proc. Natl. Acad. Sci. U.S.A.*, 86:4076–4080.

[Tabor and Richardson 1990] Tabor, S. and Richardson, C. (1990). DNA sequence analysis with a modified bacteriophage T7 DNA polymerase (effect of pyrophosphorolysis and metal ions). *J. Biol. Chem.*, 265(14):8322–8328.

[Waterman 1988] Waterman, M. (1988). Sequence alignments. In Waterman, M., editor, *Mathematical Methods for DNA Sequences*, chapter 3, pages 54–90. CRC Press, Boca Raton, Fla.

[Weaver and DePamphilis 1982] Weaver, D. and DePamphilis, M. (1982). Specific sequences in native DNA that arrest synthesis by DNA polymerase α. *J. Biol. Chem*, 257(4):2075–2086.

[Winship 1989] Winship, P. (1989). An improved method for directly sequencing PCR amplified material using dimethyl sulphoxide. *Nucleic Acids Research*, 17(3):1266.

[Yager and von Hippel 1987] Yager, T. and von Hippel, P. (1987). Transcription elongation and termination in *escherichia coli*. In Neidhardt, F., editor, *E. Coli and S. Typhimurium: Cellular and Molecular Biology*, chapter 76, page 1255. Am. Soc. Microbiology, Washington, D.C.