An Evaluation of Intron Significance Using Bioinformatics

Senior Project Submitted by

Kelsey Byers
Biotechnology Academy

April 17, 2003

Table of Contents

Table of Contents____________________________________1

Explanation of Notation_______________________________1

Abstract____2

Introduction____2

Materials____5

Methodology____14

Results____15

Discussion ____30

References ____30

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Explanation of Notation

AF088282_1intron3 signifies gene AF088282, splice transcript 1 (this number is missing if the gene has only one transcript), intron number 3.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Abstract

The debate over whether introns carry genetic significance or are merely evolutionary remnants is ongoing. Introns share few of the conserved sequences that are such a major feature of exons and genes, and are removed during to the transcription process. The only indication that introns might carry any significance at all is their presence, which possibly indicates a past or present function.

Seven genes that exhibit alternative splicing were chosen from GenBank based on availability of intron coordinates and raw genomic DNA sequences. The introns were extracted manually and converted to FASTA format using a Perl script. Each intron was then compared to each intron (an undertaking involving 10,404 commands run by one script) and the results parsed to remove extraneous data. Results with one hundred percent identity were retained and converted into GFF format (Artemis flavor) for display in the Apollo Genome Browser. Global sequence coordinates for the matches were found using the original sequences.

Sixteen matches were found, each approximately twenty base pairs in length. The sequences were all polyAs and polyTs, but were not located in the same regions as the conserved polyAs found in introns. The presence of conserved intron sequences, albeit small ones, argues for a possible function for introns.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Introduction

With the completion of the human genome project, new information about the human genome is readily available. About 99 percent of the human genome is comprised of so-called "junk" DNA, which occurs both in and between functional genes. "Junk" DNA present within genes occurs in short stretches known as introns. More formally, introns are non-coding sections of DNA that occur within functional genes.

There are eight major classes of introns, each occurring in different locations. GU-AG introns (named for the splice sites), the most common, occur in the genomes of eukaryotes. AU-AC introns (also named for their splicing sites) are also found in eukaryotic genomes, along with some Group I introns. Group I and II introns are also found in organelle and some bacterial

RNAs. Group III introns and twintrons are found exclusively in organelle RNAs. Pre-tRNA introns are found in eukaryotic nuclear pre-tRNA, and archeal introns are found in various RNAs in archaebacteria.

Vertebrate introns contain several conserved sequences. The 5' end of the intron contains a 5' splice site, typically of the sequence GU (guanine-uracil). The 3' end of the intron contains a 3' splice site, typically of the sequence AG (adenine-guanine). Preceding the 3' splice site is a polypyrimidine tract, consisting of many repeated pyrimidine nucleotides (either cytosine or uracil). Some introns also contain "branch points," consisting of a single adenine nucleotide located between 20 and 50 base pairs upstream from the 3' splice site. When splicing occurs, the branch point binds to the 5' splice site (specifically, to the uracil) and forms a "lariat," with the loop being the sequence upstream of the branch point and the tail being the downstream sequence. Some introns also have branch point sequences that incorporate polyA signals. Few other conserved sequences are found.

There are two major hypotheses of intron evolution: "introns early" and "introns late." "Introns early" states that introns originated early in the genomic evolution process and are slowly being removed from eukaryotic genomes. "Introns late" states that introns are a recent development in the genome and are accumulating in, rather than disappearing from, eukaryotic genomes. These distinctions apply mainly to GU-AG introns, as the other introns are either rare or (as in the case of the Group I, II, and III introns) the origins are already known. The "introns early" model is also known as the "exon theory of genes" model, which states that introns were formed during the transition from RNA-based organisms to DNA-based organisms. According to this theory, modern eukaryotic genes were formed from short (mostly single-domain) genes that associated into one gene, with the intervening sequences becoming the introns. The logical consequence of this theory is the idea that all genomes possessed introns at some point. As bacterial genomes no longer contain introns, they must have been lost at some early point, which disagrees with the "introns early" hypothesis. Although evidence of similarity between Group II and GU-AG introns helps support the "introns late" hypothesis, this evidence may also be turned around to support the "introns early" hypothesis, although in a different manner than in the "exon theory of genes." Both hypotheses are still viewed as valid, as significant evidence to prove or disprove either has yet to be put forth.

Several interesting evolutionary effects have been noticed in introns found in rRNA (ribosomal RNA). Two possibilities exist for rRNA introns: that they are vertically inherited from ancestral genes, or that they are preferentially horizontally transferred to certain specific target sites. The vertical inheritance hypothesis would be sustained by evidence of closely related introns in unrelated organisms; conversely, the horizontal transfer hypothesis would be sustained by evidence of different intron classes located in the same region of the rRNA. Another question related to the horizontal transfer hypothesis is that of intron movement: are introns transposed by protein-dependent mechanisms or are they selectively retained after random insertion (i.e. introns that reduced function would be removed)? Studies showed that introns at the same position were more closely related than those at different positions, and therefore introns were retained for long periods of evolution after their initial insertion. Introns were observed to be periodically lost from certain lineages and were rarely regained or transferred to new lineages. However, a certain amount of mobility is required to ensure survival and spread of intron sequences to new organisms, otherwise the two observations would contradict one another. Finally, it was observed that frequent deletion and insertion of introns led to different phylogenetic trees based on introns or exons of rRNA. From the evidence, it appears that the vertical inheritance hypothesis is more valid than the horizontal transfer hypothesis.

Logically, like exons, intron populations and locations in the genome have changed over time and between species. Intron size itself (the typical length of each intron) is actively regulated, as changes in intron size may affect gene expression and protein export from the cell nucleus. In *Drosophila*, the gene 4f-rnp is an example of another type of intron regulation: the regulation of the splicing process. Spliceosomal intron losses within this gene create functional alternative splices in the species. When spliceosomal introns were compared between distantly related species, they were found to have undergone differential intron loss. It was observed that there was a bias in intron loss toward the 3' (AG) end of the intron. One possible explanation for this behavior is the idea of intron elimination due to genetic drift.

Of course, the most important question relating to introns, and the one that is a very recent development, is the idea of the significance or function of introns. Some argue that introns are critical to the genome, an idea supported by the "introns early" hypothesis. As introns were present in the common ancestor of all living species, and are still present in eukaryotes, they must be critical. The opposing idea is that introns are "leftovers," remnants of the evolutionary process. For example, when translated into amino acid sequences, introns do not code for meaningful proteins. Many researchers' beliefs fall in the middle of the two ideas, believing that intron function cannot be essential to translation but may provide useful regulatory functions for some organisms. Unspliced introns affect protein synthesis, as they are present in the mRNA and amino acid sequences. One idea states that introns may be required for the modification of mRNA, with the idea that introns inhibit transcriptional activity. The splicing process is thus necessary, as it activates transcription. In addition, introns affect expression levels of some genes. Minimal introns (those at the lowest end of the size distribution) have been shown to affect exportation of proteins from the cell nucleus, with the hypothesis that three export paths exist: mRNAs without introns, mRNAs with non-minimal introns, and mRNAs with minimal introns. In addition, some genes exhibit odd behavior, where the order of intron removal affects the splicing process. If one intron is removed earlier than another, certain exons are "accidentally" removed along with their flanking introns. Small clues to the possible function of introns have been discovered and described. However, the idea of intron function has not been widely researched.

Bioinformatics tools provide a useful medium for analysis of genetic sequence. Manual analysis of genetic data has become obsolete with the development of high-powered processors, data storage, and programming languages, and a variety of tools for genomic and proteomic analysis are readily available. Types of bioinformatics tools for sequence analysis include alignment programs, which compare sequence data for similarity; translation programs, which translate sequences from, for example, DNA to mRNA; gene prediction programs, which search for possible open reading frames (ORFs) in genomic sequences; annotation programs, which allow the user to manually or automatically analyze a variety of results from various tools; modeling programs, which display complex three-dimensional protein structure, et cetera. Many of these programs (and many other useful scripts) are written in Perl (Practical Extraction and Report Language), the major bioinformatics programming language. Bioinformatics programs, along with Perl, are extremely valuable to sequence analysis, and doubtless have applications in the study of intron function and significance.

In this study, conserved elements shared by several introns are discovered and located in the introns of seven genes that exhibit alternative splicing. The conserved elements are shown to be located in regions other than those corresponding to previously known conserved polyAs and polyTs. In addition, the presence of polyTs, previously not specifically noted in introns, is interesting. PolyTs are only noted in conjunction with introns because they are used to remove mRNAs from genomic DNA. The mRNA contains polyA sequences that the polyTs bind to. The uniform size of these motifs (between seventeen and twenty-five base pairs in length) is also interesting: the mean (20.9375 b.p.), median (20 b.p.), and mode (20 or 24 b.p.) are all consistent. This study can serve as a springboard for future investigation of introns, as well as demonstrating conserved elements that may indicate a function for introns.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Materials

Operating Environment

Red Hat Linux v. 7.2
Linux, one of the four major operating systems along with Windows, Macintosh, and Unix, is most similar to Unix in both operation and design. It shares Unix's text-based user interaction but includes a slightly more complex graphic user interface. Linux is available in a variety of distributions, the most well known being Mandrake, Red Hat, and Debian GNU. A full list may be found at http://www.linux.org/dist/list.html. Linux was developed by Linus Torvalds in 1991. Red Hat Linux v. 7.2 is a recent release of Red Hat's distribution. Information on acquiring Red Hat Linux may be obtained at http://www.redhat.com/software/.

Hardware details
126692K memory
257032K swap space
9550788K disk space
CPU speed unavailable. Information obtained using top (memory and swap space) and df (disk space) commands in bash.

bash

bash (the **B**ourne **A**gain **Sh**ell) is a shell, or program for text-based interaction with the computer, largely used in the Unix and Linux operating systems. It allows the user to perform tasks in the system environment by entering the command names into the shell. bash is similar in function to the MS-DOS terminal but is more integral to the user's interaction with the operating system. bash comes standard with most Linux distributions and with Unix, but it may also be found at http://ftp.gnu.org/gnu/bash/.

Perl

Perl, the **P**ractical **E**xtraction and **R**eport **L**anguage, is a programming language widely used in the field of bioinformatics. Perl can be used for many tasks: running commands, manipulating files or data, and generally facilitating data acquisition and analysis. Unlike C, Perl requires no manual compilation, making it ideal for less involved tasks. Perl utilizes regular expressions, control structures, and variable systems similar to those found in C. Perl was created by Larry Wall in 1987. Perl is available at http://www.perl.com/pub/a/language/info/software.html.

Programs and Tools

LALIGN

LALIGN, written by William Pearson as part of the FASTA bioinformatics analysis package, calculates multiple local alignments between two sequences. It utilizes the SIM algorithm developed by Huang and Miller in 1991. Unlike many current alignment programs, LALIGN displays multiple alignments with different scores, preventing the program from overlooking additional significant results. LALIGN output displays the sequence names, sizes, and matrix followed by scoring and coordinate information and a graphic base-by-base alignment for each hit. LALIGN is available from ftp://ftp.virginia.edu/pub/fasta.

Apollo

Apollo, a genome annotation and curation tool, was developed by the efforts of the FlyBase-Berkeley Informatics Group, the Howard Hughes Medical Institute, and the Sanger Institute/European Bioinformatics Institute. It provides a graphic display of genomic data and genomic annotations. In addition, annotations may be made within the program. Apollo accepts input in the Artemis flavor of GFF (Gene Finder Format) and exports to Apollo or GFF format. The program may be customized to accommodate data from various programs. Apollo is available through the Berkeley Drosophila Genome Project at http://www.fruitfly.org/annot/apollo/install.html.

XV

XV is a graphics program for acquiring, manipulating, reading, and saving graphic files in multiple formats. XV is most useful in the bioinformatics field in its capacity for acquiring data via its screenshot capability. It is also used to display these files (which may also be shown in a browser window). XV is shareware and is available from http://www.trilon.com/xv/downloads.html.

Programs Written Specifically for This Project

intronlalign

This script is intended to LALIGN intron sequences from 7 alternative splice genes. Each intron sequence must be compared to each other intron sequence. The script is written in Perl.

```
#!/usr/bin/perl -w
#
# Comments:
#       This script is intended to lalign intron sequences from 7 alternative
#       splice genes.  The intron sequences have been previously separated
#       from their parent genes.  Each intron sequence must be compared to
#       each other intron sequence.

# start pseudocode

# define scalars (input, output files and databases)
# $i method (until $i>100 and such); $f array method (see examples below)


#define scalar variables

$QUERYDIR = "/home/kbyers/alternative_splices/All";
$DB = "/home/kbyers/alternative_splices/All2";
$OUTPUTDIR = "/home/kbyers/alternative_splices/intronlalign";

#put files from $db into an array
opendir (DB, $DB) || die "cannot open directory handle. $!";
while ($dbfile = readdir(DB)) {
    unless ($dbfile=~/^\./) {
        push (@intronlist,$dbfile);
        push (@querylist,$dbfile);
    }
}

#do lalign
foreach $f (@intronlist) {
    foreach $i (@querylist) {
        $cmd = "lalign $DB/$f $QUERYDIR/$i > $OUTPUTDIR/out.$f.$i.lalign";
           print $cmd,"\n";
           system ($cmd);
    }
}

close (DB);
```

lalign.parse2

A parser for lalign data to remove the actual alignment data and keep the %identity, overlap, and score. It searches for a regular expression of digits followed by a percent sign (the /\d\d.\d\%/) and prints the line containing this expression to the output file. The program requires command-line input of filenames and is written in Perl.

```
#!/usr/bin/perl -w
```

```
#
# Comments:
#       A parser for lalign data to remove the actual alignment data and
#       keep the %identity, overlap, and score.
#
# Code:

# later on this input dialog will be replaced with a foreach loop to do all
# 8000-odd parses... this is the interactive version, soon to be automated.
# Likely I will push this file to a <> mode to accept command-line file calls
# and will then write another script to execute this one automatically.
# Call with lalign.parse2 $f > $f.parsed.

# that push to full auto has already been done.

while (defined($line = <>)) {
        if ($line =~ /\d\d.\d\%/) {
                print ("$line\n");
        }
}
```

parse_lalign

This script is intended to run 'lalign.parse2' on a series of files resulting from a run of LALIGN. It is written in Perl.

```
#!/usr/bin/perl -w
#
# Comments:
#       This script is intended to run 'lalign.parse2' on a series of files
#       resulting from a run of lalign.
#
# Code:

# get files
$INPUTDIR = "/home/kbyers/alternative_splices/intronlalign";
$OUTPUTDIR = "/home/kbyers/alternative_splices/intronlalign_parse";

#put files from input into an array
opendir (INPUT , $INPUTDIR) || die "cannot open directory handle. $!";
while ($dbfile = readdir(INPUT)) {
    unless ($dbfile=~/^\./) {
        push (@lalignlist,$dbfile);
        }
}

#do parse
foreach $f (@lalignlist) {
           $cmd = "lalign.parse2 $INPUTDIR/$f > $OUTPUTDIR/$f.parsed";
           print $cmd,"\n";
           system ($cmd);
}
```

lalign_narrow100

This script is similar to lalign.parse2. However, instead of removing lines based on the presence of digits and a percent sign, it screens by removing all lines without a "100.0%" present. It's intended to remove lines resulting from lalign.parse2 that contain values of less than one hundred percent. Written in Perl.

```
#!/usr/bin/perl -w
#
# Comments:
#       A parser for lalign data to remove all data below 100% and
#       keep the %identity, overlap, and score.
#
# Code:

# later on this input dialog will be replaced with a foreach loop to do all
# 8000-odd parses... this is the interactive version, soon to be automated.
# Likely I will push this file to a <> mode to accept command-line file calls
# and will then write another script to execute this one automatically.
# Call with lalign_narrow100 $f > $f.parsed.

# that push has already been done.

while (defined($line = <>)) {
        if ($line =~ /100.0\%/) {
                print ("$line\n");
        }
    }
```

<u>lalign_narrowing100</u>

This program is a modification of parse_lalign and runs lalign_narrow100 (as opposed to lalign.parse2). Written in Perl.

```
#!/usr/bin/perl -w
#
# Comments:
#       This script is intended to run 'lalign_narrow100' on a series of files
#       resulting from a run of lalign.
#
# Code:

# get files
$INPUTDIR = "/home/kbyers/alternative_splices/intronlalign_parse";
$OUTPUTDIR = "/home/kbyers/alternative_splices/intronlalign_narrow100";

#put files from input into an array
opendir (INPUT , $INPUTDIR) || die "cannot open directory handle. $!";
while ($dbfile = readdir(INPUT)) {
    unless ($dbfile=~/^\./) {
        push (@lalignlist,$dbfile);
        }
}

#do parse
foreach $f (@lalignlist) {
           $cmd = "lalign_narrow100 $INPUTDIR/$f > $OUTPUTDIR/$f.parsed";
           print $cmd,"\n";
           system ($cmd);
}
```

counterimage100

This program counts the number of characters in all files in a directory and then outputs the filename and number of characters for each file into a separate file (countlist100). Written in Perl.

```
#!/usr/bin/perl -w
#
# Commentary:
#       This program is a perl script I've been working on lately.
#       Its purpose is to count the number of characters in each
#       file in the given directory, then save the names of the files with
#       a certain number of characters to another file.
#
# Code:

#define scalar variables
$COUNTDIR="/home/kbyers/alternative_splices/intronlalign_narrow100";

#put files from $countdir into an array
opendir (COUNTDIR, $COUNTDIR) || die "cannot open directory handle: $!";
while ($file=readdir (COUNTDIR)) {
        unless ($file=~/^\./) {

             push (@countlist,$file);
        }
}
close (DIR);

#do count
foreach $f (@countlist) {
$cmd="wc -m $COUNTDIR/$f >> countlist100";
        print $cmd,"\n";
        system ($cmd);
}
```

count.parse

This program removes the filename and number of characters of any file with 0 characters from the file resulting from countlist100. Written in Perl.

```
#!/usr/bin/perl -w
#
# Comments:
#       A parser for lalign data to remove the actual alignment data and
#       keep the %identity, overlap, and score.
#
# Code:

# later on this input dialog will be replaced with a foreach loop to do all
# 8000-odd parses... this is the interactive version, soon to be automated.
# Likely I will push this file to a <> mode to accept command-line file calls
# and will then write another script to execute this one automatically.
# Call with count.parse $f > $f.parsed.

# again, already done.
```

```
while (defined($line = <>)) {
        if ($line =~ /1\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /2\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /3\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /4\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /5\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /6\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /7\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /8\d\s/) {
                print ("$line\n");
        } elsif ($line =~ /9\d\s/) {
                print ("$line\n");
        }
}
```

<u>deltaseq</u>

This program, the most complex of this category, accepts user input or prompts for input. It takes a genomic sequence file of type GenBank, lowercase without numbers, uppercase without numbers, or FASTA and converts the file to FASTA. Written in Perl.

```
#!/usr/bin/perl

##############################################################################
#
#       Deltaseq:  A program to translate data to FASTA format.
#       Copyright 2003 by Kelsey Byers.
#       This program may be copied or modified (with credit to the original
#       owner) but NOT sold.  This message must be retained and the program's
#       source passed on.  Not to be used commercially or for profit without
#       prior permission of the owner/author.
#
##############################################################################
#
# Code:

# Do you need help?
if ($ARGV[0] eq "-h") {
        print ("Deltaseq:  the FASTA sequence conversion tool.\nWritten by
Kelsey Byers, 2003.  (kjb\@space.mit.edu)\nUsage:  deltaseq -t type
filename.\nType can range from 1 to 4.\n1 = Genbank\n2 = Lowercase text\n3 =
Uppercase text\n4 = FASTA\nEntering deltaseq -h will display this
message.\n");
        exit;
}

# Did you input a filename?  Or should I prompt you?
if ($ARGV[2] =~ /^\w/) {
        $INFILE = $ARGV[2];
} else {
        print ("Please enter an input file path.\n");
        $INFILE = <STDIN>;
}

# Processing the filename
chomp($INFILE);
open (IN, "$INFILE") || die "Cannot open $INFILE:  $!";
$TMPFILE = $INFILE . ".tmp";
open (TMP, ">$TMPFILE") || die "Cannot open temporary file $TMPFILE:  $!";
$OUTFILE = $INFILE . ".fold.fa";

# Did you input a type?  Or should I prompt you?
if ($ARGV[0] eq "-t") {
        $type = $ARGV[1];
} else {
        print ("Please enter a sequence type.\n1 = Genbank\n2 = Lowercase
text\n3 = Uppercase text\n4 = FASTA\n");
        $type = <STDIN>;
}
chomp($type);

# A few commands needed later for formatting and removing temporary files
$fold = "fold -w 60 $TMPFILE > $OUTFILE";
$rm = "rm -f $TMPFILE";

# Processing the file
if ($type == "1") {
        print TMP (">$OUTFILE\n");
        while ($fasta = <IN>) {
                $fasta =~ tr/a-z/A-Z/;
                $fasta =~ s/\d//g;
                $fasta =~ s/\s//g;
                print TMP ("$fasta");
        }
        system ($fold);
        system ($rm);
} elsif ($type == "2") {
        print TMP (">$OUTFILE\n");
        while ($fasta = <IN>) {
                $fasta =~ tr/a-z/A-Z/;
                $fasta =~ s/\s//g;
                print TMP ("$fasta");
        }
        system ($fold);
        system ($rm);
} elsif ($type == "3") {
        print TMP (">$OUTFILE\n");
        while ($fasta = <IN>) {
                $fasta =~ s/\s//g;
                print TMP ("$fasta");
        }
        system ($fold);
        system ($rm);
} elsif ($type == "4") {
        print ("$INFILE is already in FASTA format.\n");
        system ($rm);
} else {
        print ("$type is not a recognized file type.\n");
}
```

```
exit;

###############################################################################
# Command-line Options
#
# -h:  help.  Displays the help message.
# -t:  type.  Should be -t 1 (for example).
# filename to be changed.
# Should be deltaseq -t <type> <filename> or deltaseq -h
###############################################################################
```

<u>apollo</u>
The executable for Apollo is location-specific and must be run from its original directory. This script, written in shell script, executes Apollo from any location.

```
#!/bin/sh
#
# code to execute apollo
#

exec /home/kbyers/Apollo/Apollo "$@"
```

Data/Sequences

<u>GenBank</u>.
GenBank is an annotated collection of all publicly available DNA and RNA sequences. Sequence data must under process control prior to submission, ensuring accuracy. Data in GenBank is in GenBank format (flatfiles with sequence data, annotations, references, and links). Sequence data is presented in lowercase with coordinate numbers. GenBank is a database maintained by the National Center for Biotechnology Information (NCBI) and may be found at http://www.ncbi.nlm.nih.gov/Genbank/GenbankSearch.html.
<u>AF088282/OGG1</u>
OGG1 codes for an enzyme that converts 8-oxoguanine (a byproduct of exposure to reactive oxygen, also a mutagen) into a harmless product. Many alternative splice variants for this gene exist and have been described. The gene is located at 3p26.2.
<u>AF110798/IL18BP</u>
IL18BP codes for an inhibitor of IL18, an inflammatory cytokine. It inhibits interferon production and is expressed and secreted in mononuclear cells, particularly in those from patients with Crohn's disease. Four alternative splice variants have been described. Located at 11q13.
<u>AF135025/KLK12</u>
KLK12 is a serine protease. The group of proteins in which it is classified, the kallikreins, are implicated in carcinogenesis and may be of use as biomarkers. Alternative splicing encodes three variants. KLK12 is located between 19q13.3-19q13.4.
<u>AF199339/PSIP1/PSIP2</u>
PSIP1 and 2 have not been well researched. PSIP1 is thought to protect cells from cell death via apoptotic cleavage; PSIP2 is believed to bind to involucrin promoters, thereby regulating

expression of the involucrin gene. The number of splice variants is unknown. PSIP1 is located at 9p22.2; PSIP2 is located at 9p22.1.
AY052369/PPP2R5C
Information on PPP2R5C is unavailable and its function is unknown at this time. It codes for an unknown number of splice variants. The gene is located at 14q32.
L29074/FMR1
Mutations in the FMR1 gene cause fragile X syndrome, the major symptom of which is mental retardation. The syndrome is caused by an increased number of repeats in the gene. The number of splice variants is not available. FMR1 is located at Xq27.3.
M10014/FGG
FGG codes for the gamma component of fibrinogen. Fibrinogen is cleaved by thrombin to form fibrin, the major component of blood clots. Mutations in this gene lead to several disorders. Alternative splicing results in two isoforms of the gene. Located at 4q31.3.
(http://www.ncbi.nlm.nih.gov)

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Methodology

Preparation and Sequence Acquisition
1. Installed LALIGN, Apollo, and XV on the computer.
2. Obtained 7 alternative splice gene DNA sequences (see above for descriptions) from GenBank.
3. Removed intron sequences by hand using GenBank annotation data for coordinates and stored them on hard drive of computer.
4. Converted GenBank intron sequences to FASTA format (required by LALIGN) using deltaseq.
LALIGN
5. Wrote running scripts for LALIGN (intronlalign).
6. Tested intronlalign using a few sequences.
7. Ran intronlalign over several days, resulting in 10,404 files.
Parsing LALIGN Output
8. Wrote a parser that removes the %id line from the LALIGN files (lalign.parse2).
9. Modified lalign.parse2 to take input from command-line and write to specified output per "lalign.parse2 file > file.parsed".
10. Wrote a script to run all lalign.parse2 jobs with one call (parse_lalign).
11. Parsed all output files by running parse_lalign.
12. Modified lalign.parse2 to remove all matches with identities fewer than 100 percent (lalign_narrow100).
13. Wrote a script to run all lalign_narrow100 jobs with one call (lalign_narrowing100).
14. Got list of file sizes using counterscript100 and parsed it using count.parse to get a working list of good files with file sizes larger than 0.

Moving LALIGN Output to GFF/Apollo; Viewing Results

15. Split file size list by 1st intron (the database intron) into 36 groups, representing 36 introns that had hits.
16. Each group represented one Apollo file to display. Decided to display database intron along its length and show where the other introns matched to it.
17. Modified Apollo to accept lalign input in ~/Apollo/conf/tiers.dat file.
18. Put each file from the list into GFF format to display in Apollo.
19. Took a screenshot in XV of each file's Apollo view.
20. Noticed the presence of lots of duplicates due to repeated introns and overlaps, so screened out duplicates. Left with 22 files.
21. Had a problem with AF199339_1intron3 and 2intron3 being the same intron but hitting to different sites (6 off) on another intron- not really sure why that occurred. Sequences were exactly the same when tested in LALIGN.
22. Removed duplicate introns from good Apollo files (e.g. if intron 1 and its duplicate are both matching, remove the duplicate from the GFF file).
23. Removed hits from same genes due to overlap of transcripts rather than significant similarity between the introns.
24. Took screen shots of good introns using XV.

Obtaining Matching Sequences

25. Found global coordinates of intron motifs (the sequences that were matched by other introns).
26. Got intron sequences that were motifs of each intron (16 in total). All polyAs and polyTs of approximately 20 base pairs each.
27. Converted motif sequences to FASTA format.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Results

Format:

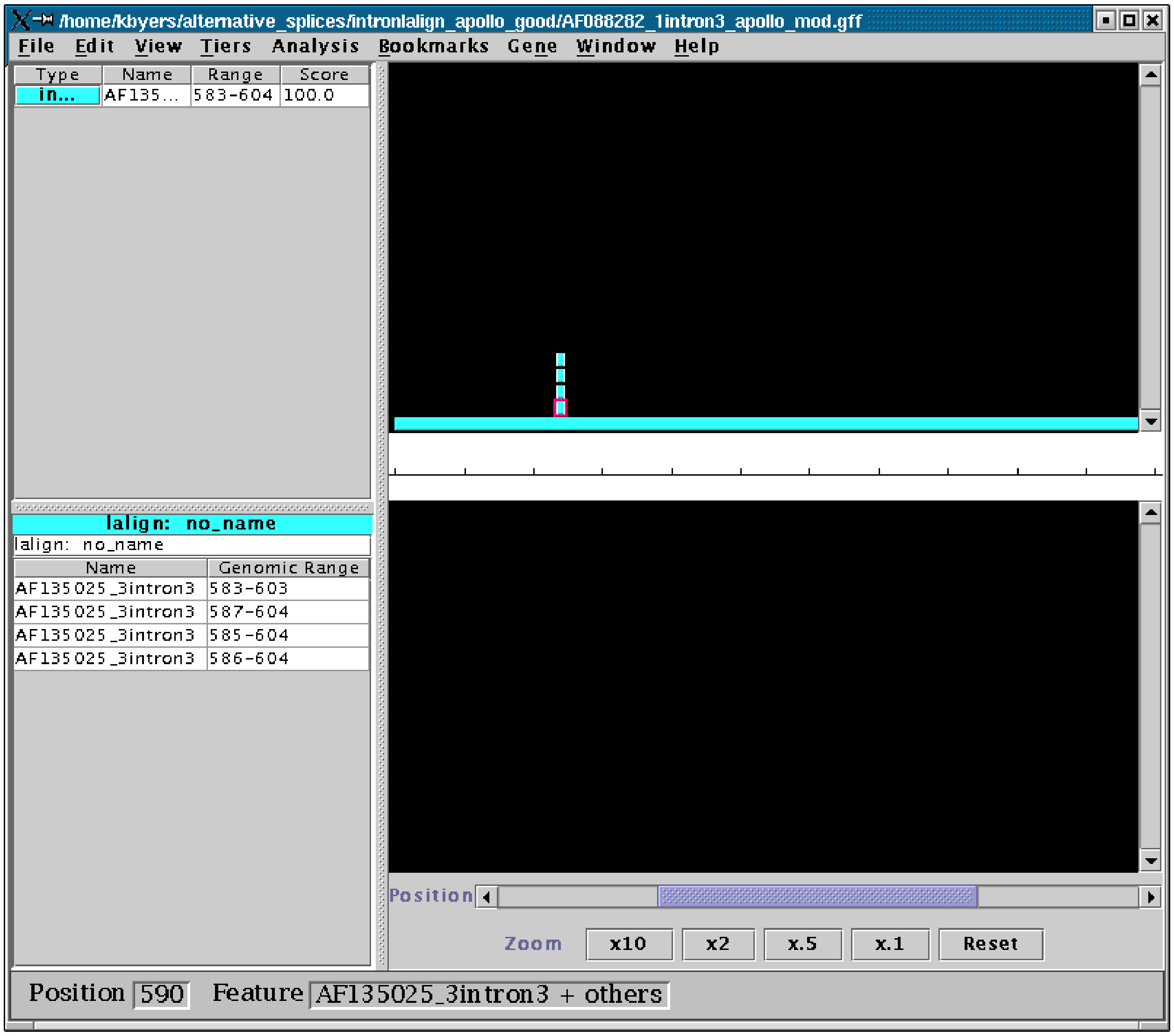
Apollo screenshot of data

Database intron: number of introns aligned to it.

Database intron		start	stop	# in local coordinates
Hit 1	start	stop		
Hit 2	start	stop		# ad nauseum
Global Coordinates: start-stop				# in global coordinates

Motif Sequence Length: Length in base pairs

Motif Sequence: DNA sequence of motif


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF088282_1intron3_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom x10 x2 x.5 x.1 Reset

Type	Name	Range	Score
in...	AF135...	583-604	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF135025_3intron3	583-603
AF135025_3intron3	587-604
AF135025_3intron3	585-604
AF135025_3intron3	586-604

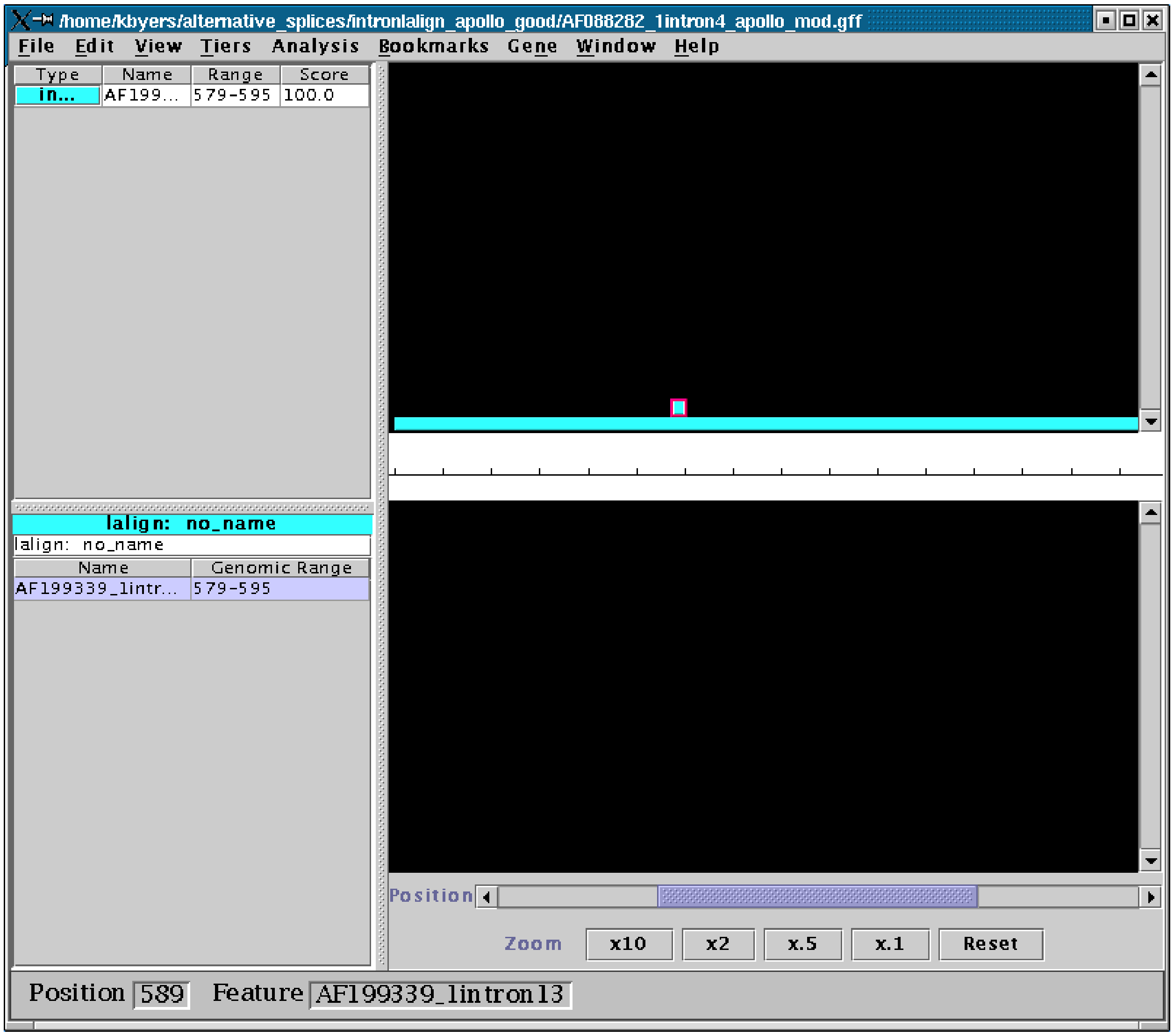
Position 590 Feature AF135025_3intron3 + others

AF088282_1intron3: 3 introns aligned.

AF088282_1intron3	1	2750
AF135025_1intron3	583	603
AF135025_1intron3	585	604
AF135025_1intron3	586	604
AF135025_1intron3	587	604
AF135025_3intron3	583	603
AF135025_3intron3	585	604
AF135025_3intron3	586	604
AF135025_3intron3	587	604
AF199339_1intron2	585	604
AF199339_1intron2	583	602
AF199339_1intron2	586	604
L29074_intron12	586	604
L29074_intron12	585	603

L29074_intron12	586	604
L29074_intron12	585	603

Global Coordinates: 3515-3534
Motif Sequence Length: 20 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAAA


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF088282_1intron4_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom
x10
x2
x.5
x.1
Reset

Type	Name	Range	Score
in...	AF199...	579-595	100.0

lalign: no_name
lalign: no_name

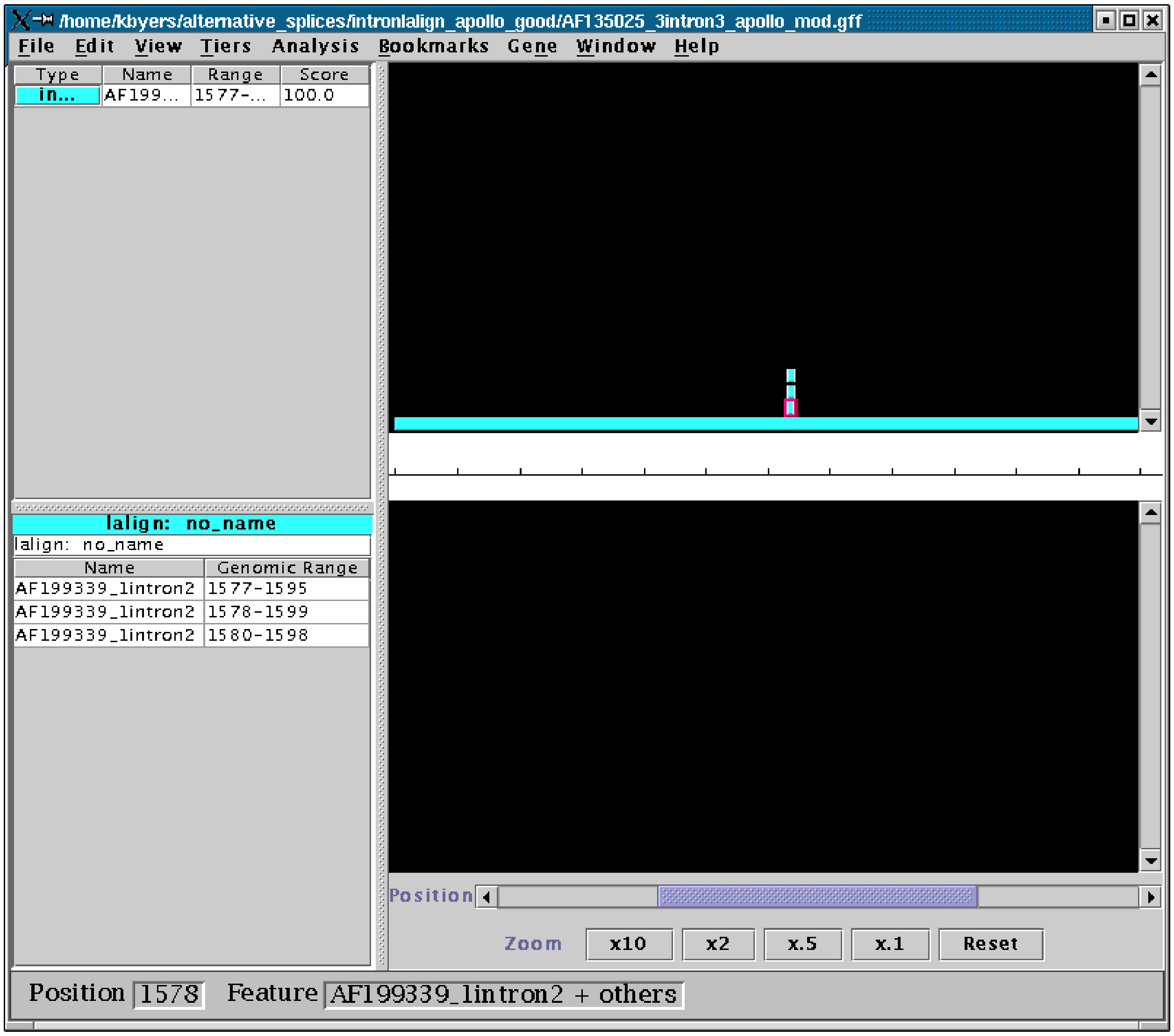
Name	Genomic Range
AF199339_1intr...	579-595

Position 589   Feature AF199339_1intron13

AF088282_1intron4: 1 intron aligned.

AF088282_1intron4	1	1574
AF199339_1intron13	579	595

Global Coordinates: 6441-6457
Motif Sequence Length: 17 b.p.
Motif Sequence: CTTTTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF135025_3intron3_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom
x10
x2
x.5
x.1
Reset

Type	Name	Range	Score
in...	AF199...	1577-...	100.0

lalign: no_name

lalign: no_name

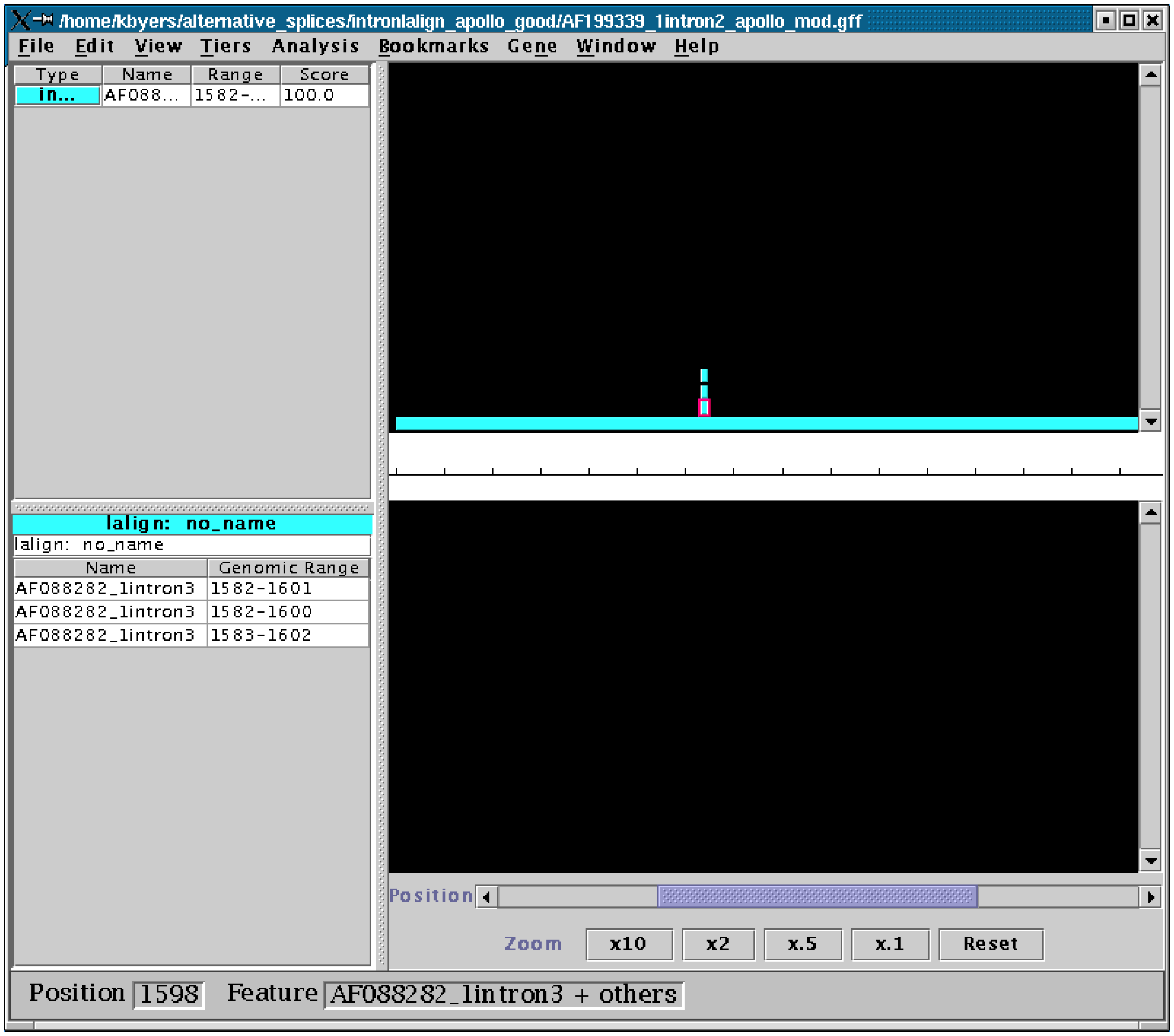
Name	Genomic Range
AF199339_1intron2	1577-1595
AF199339_1intron2	1578-1599
AF199339_1intron2	1580-1598

Position 1578 Feature AF199339_1intron2 + others

AF135025_3intron3: 3 introns aligned.

AF135025_3intron3	1	3058
AF199339_1intron2	1578	1599
AF199339_1intron2	1580	1598
AF199339_1intron2	1577	1595
AY052369_intron8	1580	1599
L29074_intron12	1580	1599
L29074_intron12	1579	1597
L29074_intron12	1579	1597

Global Coordinates: 6235-6254
Motif Sequence Length: 20 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAAG


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF199339_1intron2_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom
x10
x2
x.5
x.1
Reset

Type	Name	Range	Score
in...	AF088...	1582-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF088282_1intron3	1582-1601
AF088282_1intron3	1582-1600
AF088282_1intron3	1583-1602

Position 1598 Feature AF088282_1intron3 + others

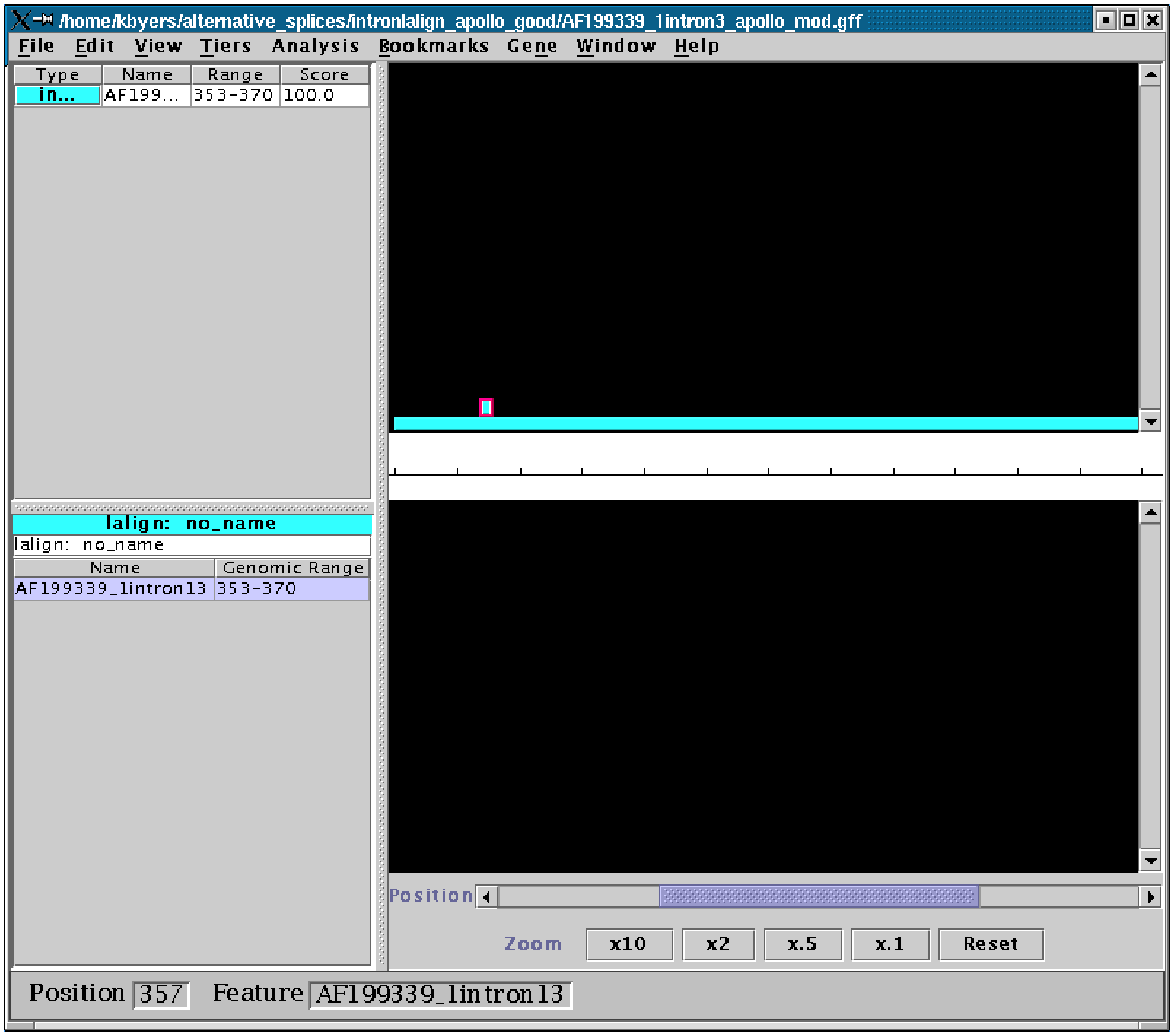
AF199339_1intron2: 3 introns aligned.

AF199339_1intron2	1	3930
AF088282_1intron3	1583	1602
AF088282_1intron3	1582	1601
AF088282_1intron3	1582	1600
AF135025_1intron3	1582	1603
AF135025_1intron3	1584	1602
AF135025_1intron3	1582	1600
AF135025_1intron3	1585	1602
AF135025_3intron3	1582	1603
AF135025_3intron3	1584	1602
AF135025_3intron3	1582	1600

Global Coordinates: 5201-5221

Motif Sequence Length: 21 b.p.

Motif Sequence: AAAAAAAAAAAAAAAAAAAAA


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF199339_1intron3_apollo_mod.gff

File Edit View Tiers Analysis Bookmarks Gene Window Help

Type	Name	Range	Score
in...	AF199...	353-370	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF199339_1intron13	353-370

Position 357 Feature AF199339_1intron13

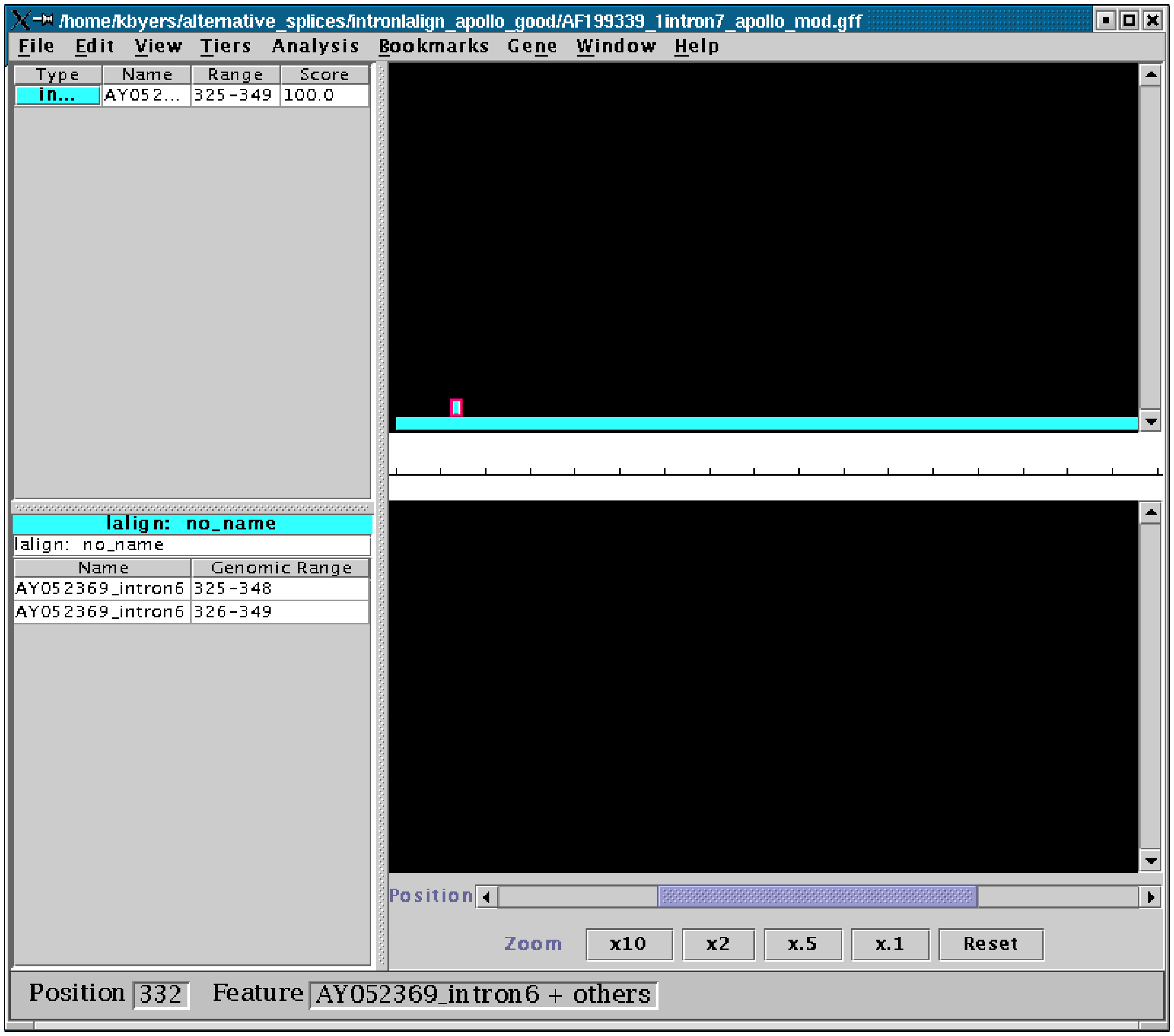
AF199339_1intron3: 1 intron aligned.

AF199339_1intron3	1	3053
AF199339_1intron13	353	370

Global Coordinates: 8341-8358

Motif Sequence Length: 18 b.p.

Motif Sequence: ATTTCTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/AF199339_1intron7_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom
x10
x2
x.5
x.1
Reset

Type	Name	Range	Score
in...	AY052...	325-349	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AY052369_intron6	325-348
AY052369_intron6	326-349

Position 332 Feature AY052369_intron6 + others

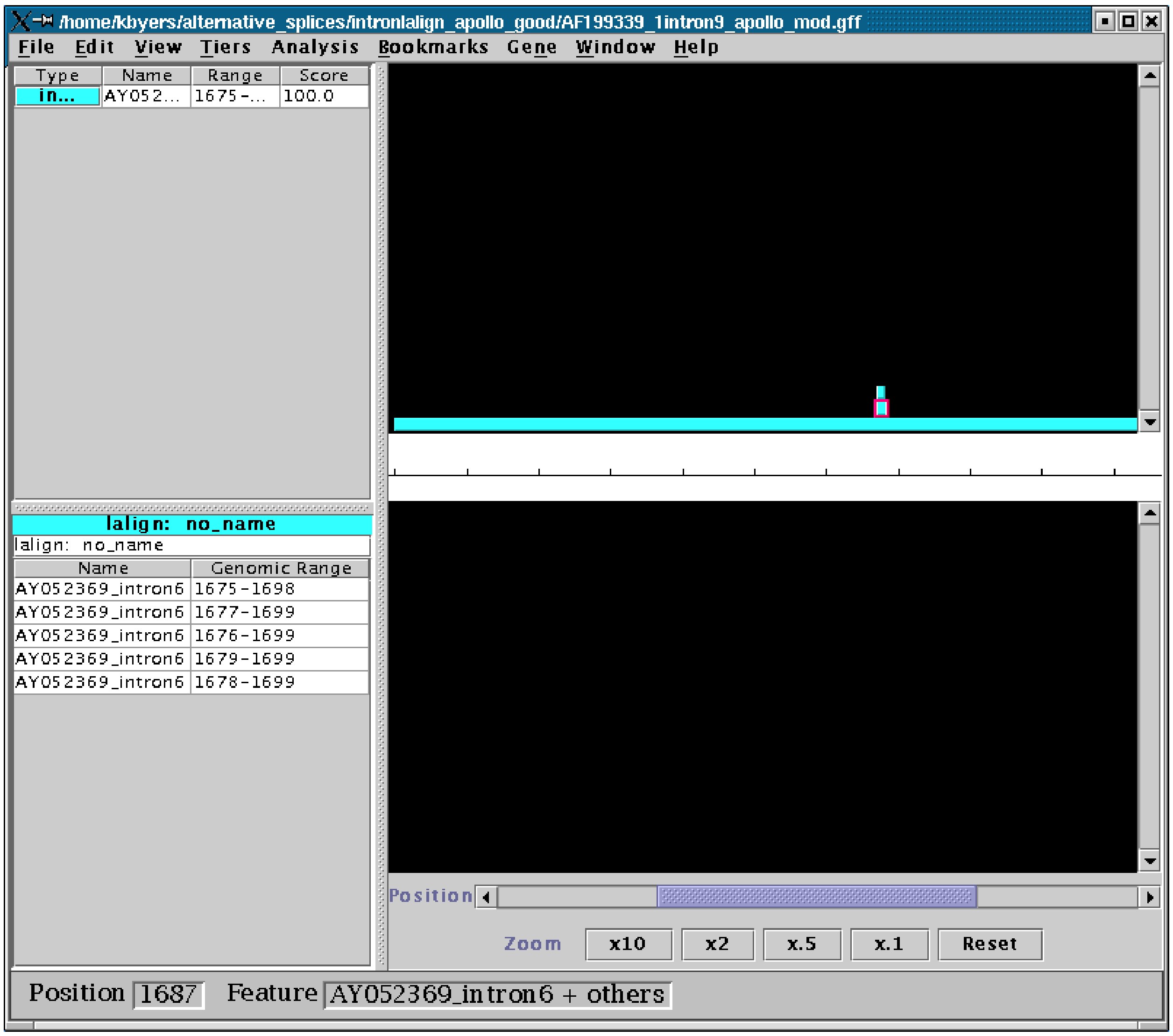
AF199339_1intron7: 1 intron aligned.

AF199339_1intron7	1	4238
AY052369_intron6	325	348
AY052369_intron6	326	349

Global Coordinates: 17132-17156

Motif Sequence Length: 25 b.p.

Motif Sequence: TTTTTTTTTTTTTTTTTTTTTTTTT



/home/kbyers/alternative_splices/intronlalign_apollo_good/AF199339_1intron9_apollo_mod.gff

Type	Name	Range	Score
in...	AY052...	1675-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AY052369_intron6	1675-1698
AY052369_intron6	1677-1699
AY052369_intron6	1676-1699
AY052369_intron6	1679-1699
AY052369_intron6	1678-1699

Position 1687 Feature AY052369_intron6 + others

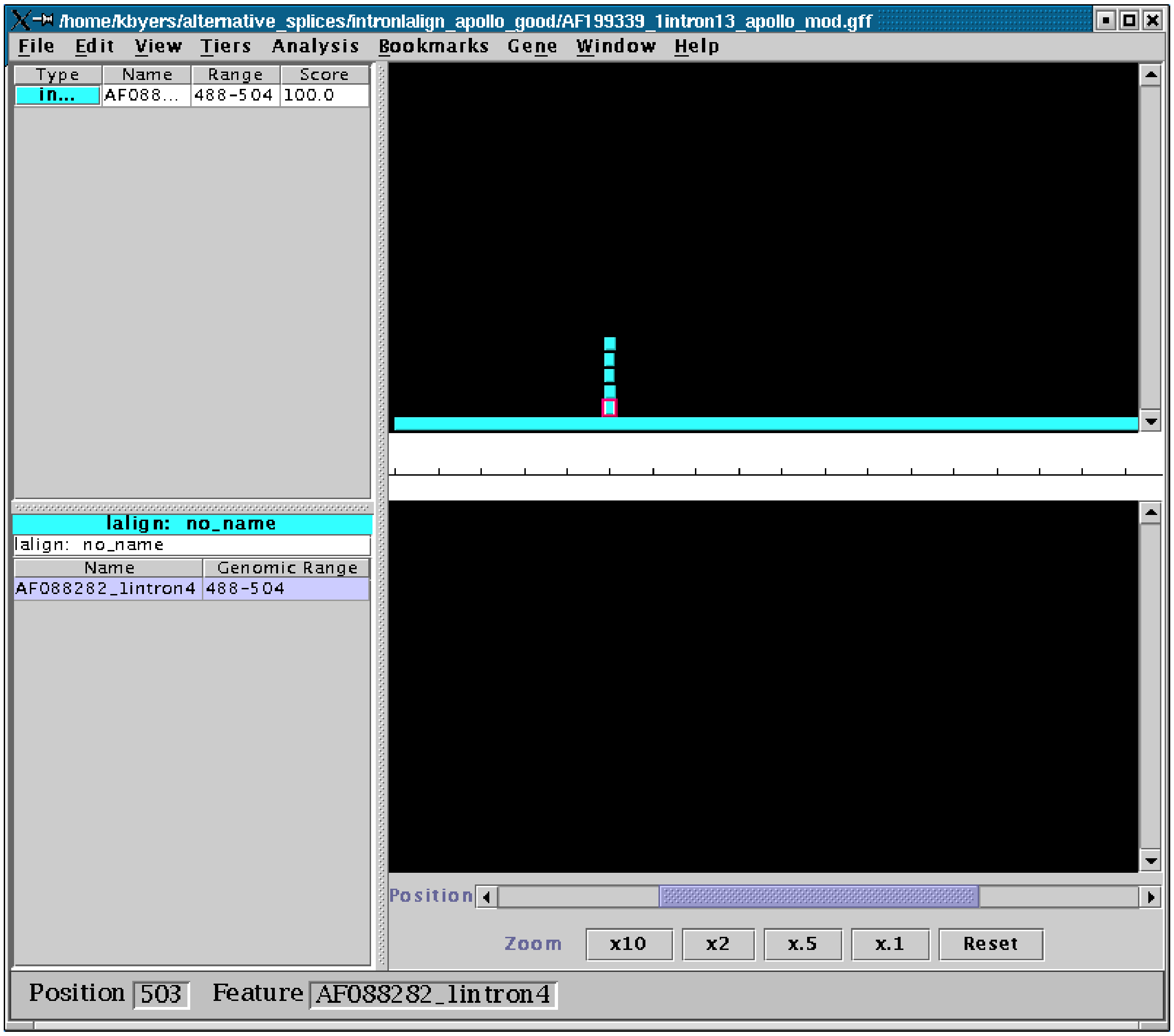
AF199339_1intron9: 2 introns aligned.

AF199339_1intron9	1	2640
AF199339_1intron13	1677	1696
AF199339_1intron13	1676	1695
AY052369_intron6	1675	1698
AY052369_intron6	1676	1699
AY052369_intron6	1677	1699
AY052369_intron6	1678	1699
AY052369_intron6	1679	1699

Global Coordinates: 24327-24350

Motif Sequence Length: 24 b.p.

Motif Sequence: TTTTTTTTTTTTTTTTTTTTTTTT



/home/kbyers/alternative_splices/intronlalign_apollo_good/AF199339_1intron13_apollo_mod.gff

Type	Name	Range	Score
in...	AF088...	488-504	100.0

lalign: no_name

lalign: no_name

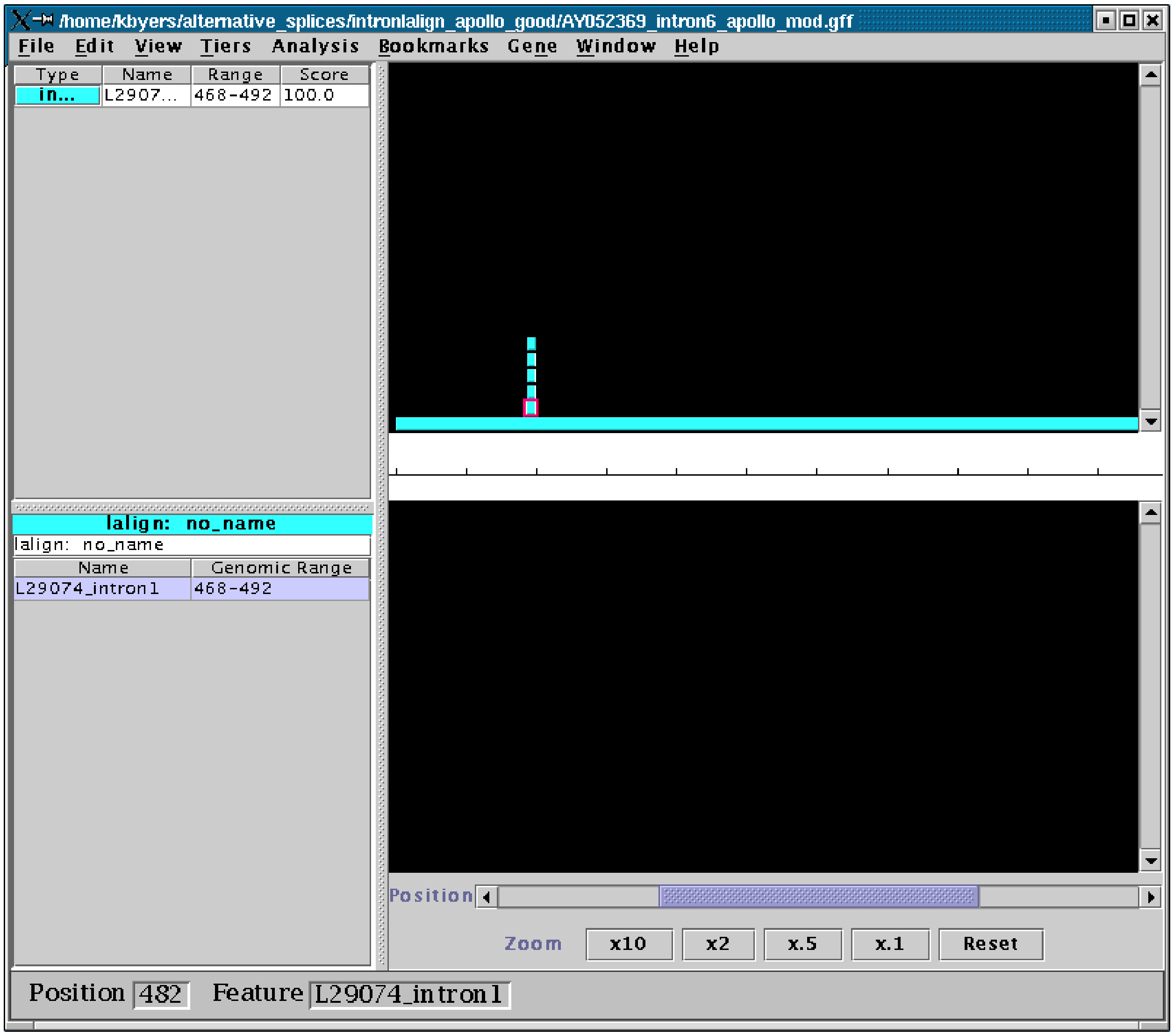
Name	Genomic Range
AF088282_1intron4	488-504

Position 503 Feature AF088282_1intron4

AF199339_1intron13: 4 introns aligned.

AF199339_1intron13	1	1769
AF088282_1intron4	488	504
AF199339_1intron3	489	506
AF199339_1intron9	489	508
AF199339_1intron9	489	508
AY052369_intron6	489	508
AY052369_intron6	489	507
AY052369_intron6	491	508
AY052369_intron9	489	506

Global Coordinates: 27143-27162
Motif Sequence Length: 20 b.p.
Motif Sequence: TTTTTTTTTTTTTTTTTTTT



/home/kbyers/alternative_splices/intronlalign_apollo_good/AY052369_intron6_apollo_mod.gff

Type	Name	Range	Score
in...	L2907...	468-492	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
L29074_intron1	468-492

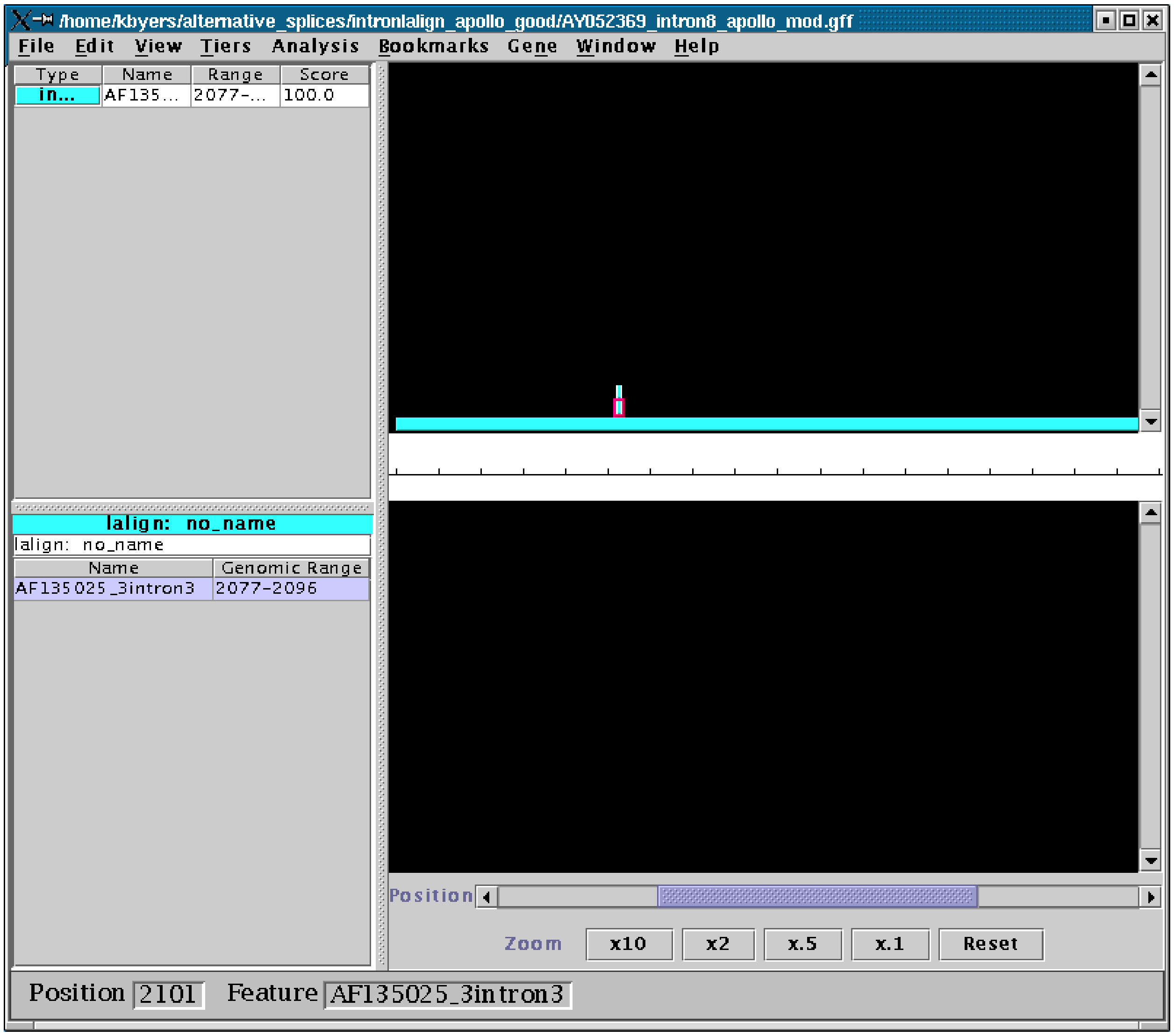
Position 482 Feature L29074_intron1

AY052369_intron6: 5 introns aligned.

AY052369_intron6	1	2703
AF199339_1intron13	471	490
AF199339_1intron13	474	492
AF199339_1intron13	469	486
AF199339_1intron7	469	492
AF199339_1intron7	469	491
AF199339_1intron9	469	492
AF199339_1intron9	469	492
AF199339_1intron9	469	491
AF199339_1intron9	469	490
AF199339_1intron9	469	489
AY052369_intron13	469	492
L29074_intron1	468	492

Global Coordinates: 83368-83391

Motif Sequence Length: 24 b.p.
Motif Sequence: TTTTTTTTTTTTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/AY052369_intron8_apollo_mod.gff

Type	Name	Range	Score
in...	AF135...	2077-...	100.0

lalign: no_name
lalign: no_name

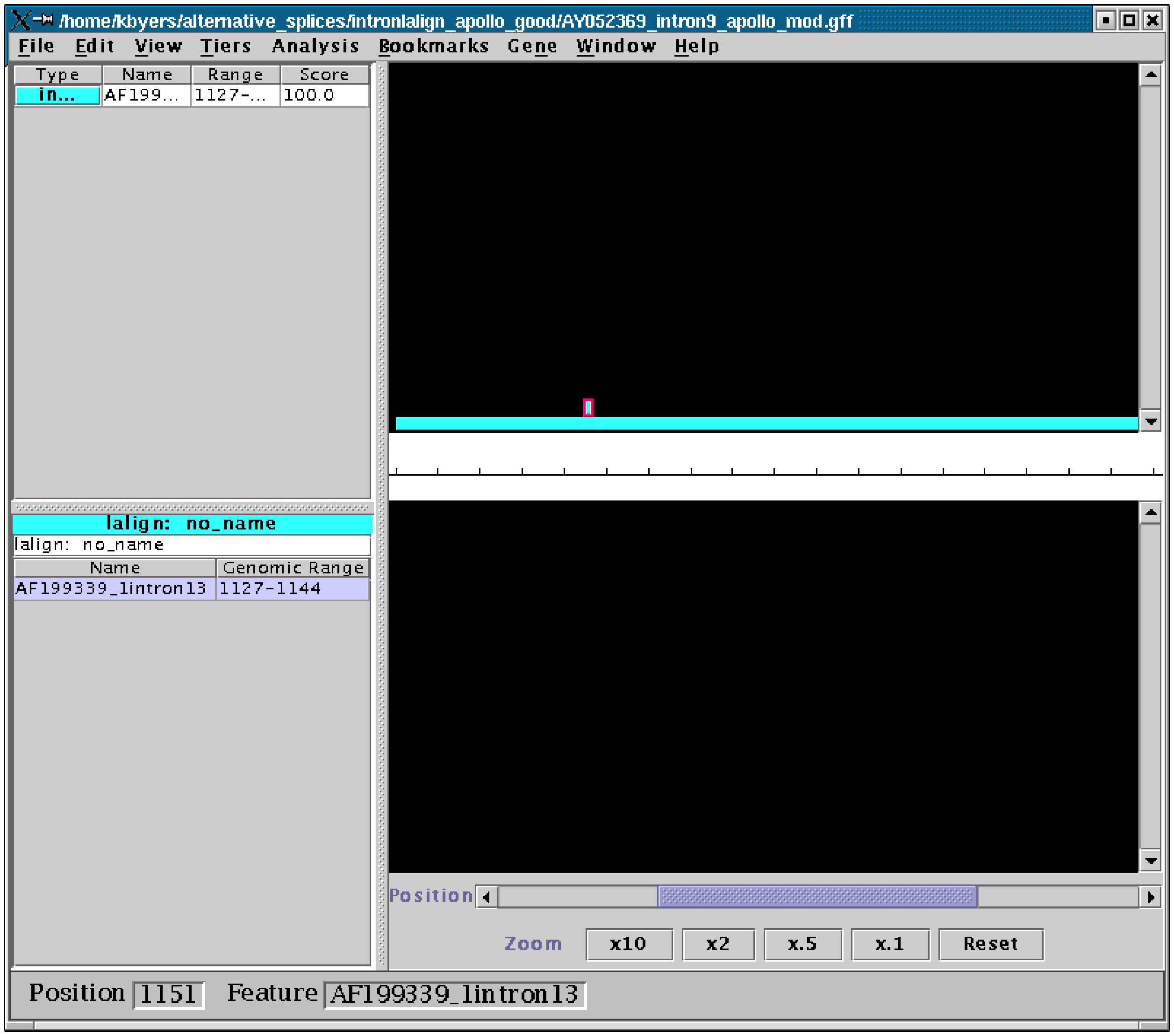
Name	Genomic Range
AF135025_3intron3	2077-2096

Position 2101   Feature AF135025_3intron3

AY052369_intron8: 2 introns aligned.

AY052369_intron8	1	7158
AF135025_1intron3	2077	2096
AF135025_1intron3	2077	2095
AF135025_1intron3	2077	2095
AF135025_3intron3	2077	2096

Global Coordinates: 89240-89259
Motif Sequence Length: 20 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAAG


/home/kbyers/alternative_splices/intronlalign_apollo_good/AY052369_intron9_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom x10 x2 x.5 x.1 Reset

Type	Name	Range	Score
in...	AF199...	1127-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF199339_1intron13	1127-1144

Position 1151 Feature AF199339_1intron13

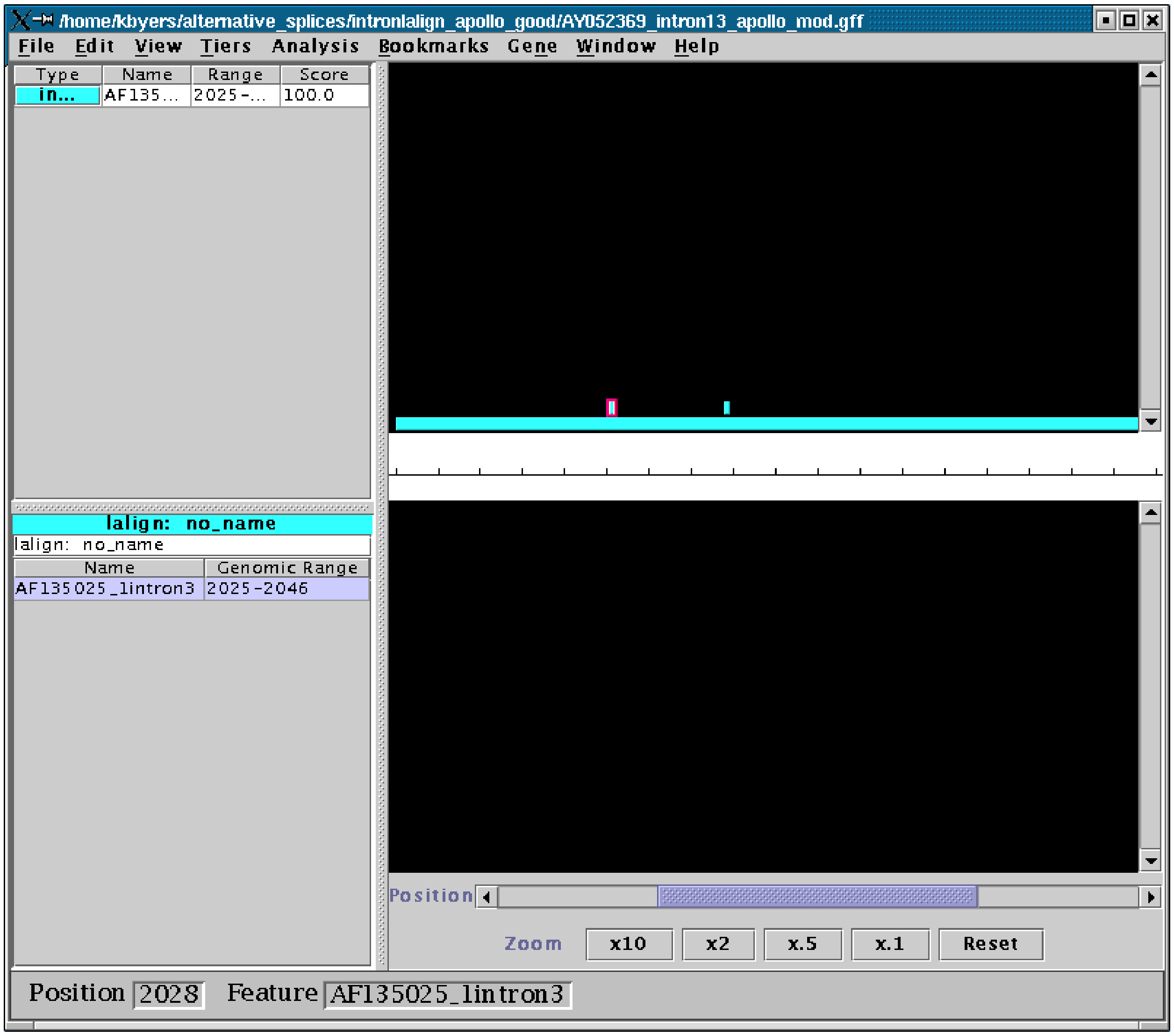
AY052369_intron9: 1 intron aligned.

AY052369_intron9	1	4512
AF199339_1intron13	1127	1144

Global Coordinates: 95619-95636

Motif Sequence Length: 18 b.p.

Motif Sequence: TTTTTTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/AY052369_intron13_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom x10 x2 x.5 x.1 Reset

Type	Name	Range	Score
in...	AF135...	2025-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF135025_1intron3	2025-2046

Position 2028 Feature AF135025_1intron3

AY052369_intron13_1: 1 intron aligned.

AY052369_intron13	1	7194
AF135025_1intron3	2025	2046

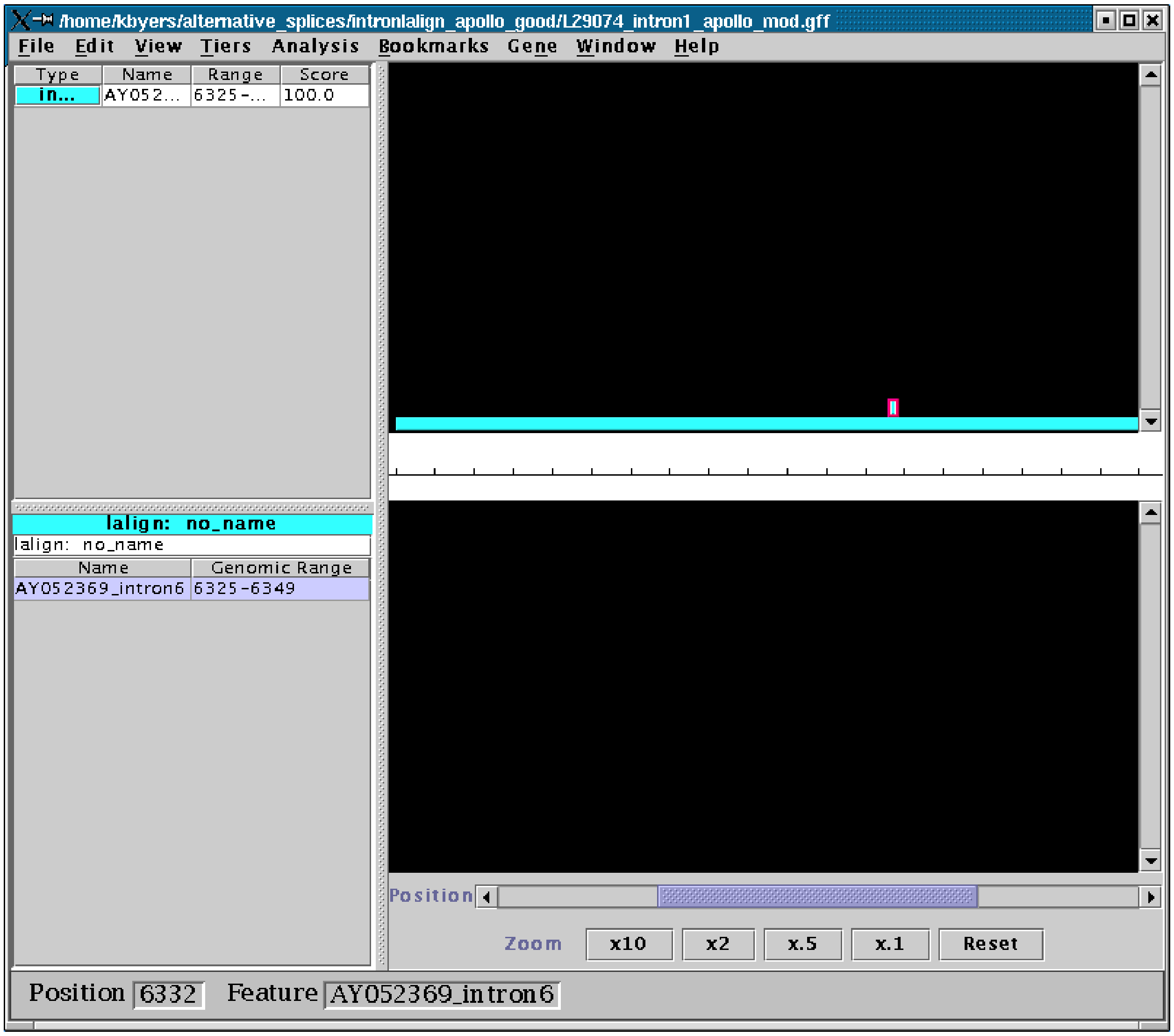
Global Coordinates: 112575-112596
Motif Sequence Length: 22 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAAAAA

These two motif sequences share the same Apollo file.

AY052369_intron13_2: 1 intron aligned.

AY052369_intron13	1	7194
AF135025_1intron3	3116	3139

Global Coordinates: 113666-113689
Motif Sequence Length: 24 b.p.
Motif Sequence: TTTTTTTTTTTTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/L29074_intron1_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom
x10
x2
x.5
x.1
Reset

Type	Name	Range	Score
in...	AY052...	6325-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AY052369_intron6	6325-6349

Position 6332 Feature AY052369_intron6

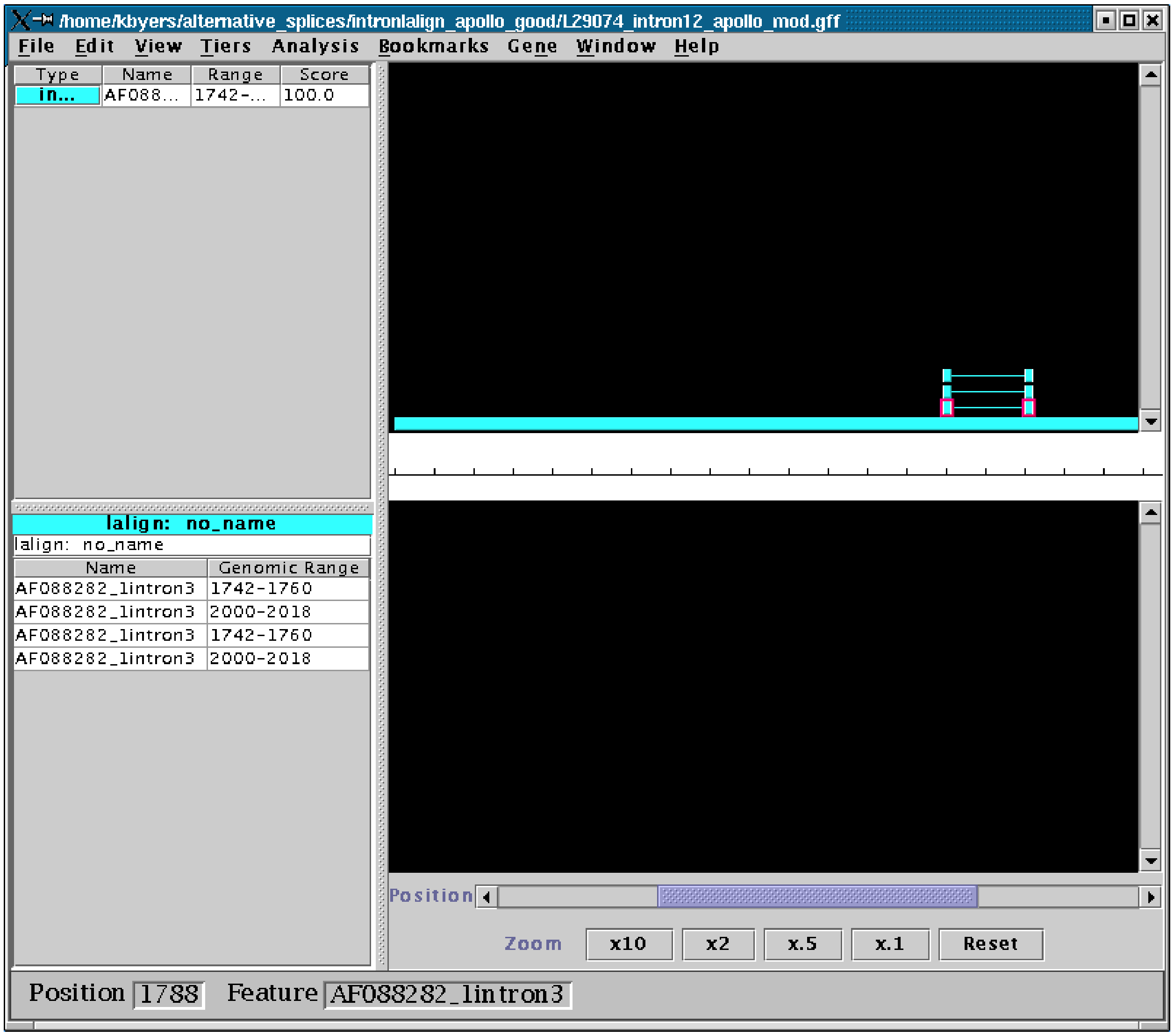
L29074_intron1: 1 intron aligned.

L29074_intron1	1	9702
AY052369_intron6	6325	6349

Global Coordinates: 20337-20361

Motif Sequence Length: 24 b.p.

Motif Sequence: CTTTTTTTTTTTTTTTTTTTTTTT


/home/kbyers/alternative_splices/intronlalign_apollo_good/L29074_intron12_apollo_mod.gff
File Edit View Tiers Analysis Bookmarks Gene Window Help
Position
Zoom x10 x2 x.5 x.1 Reset

Type	Name	Range	Score
in...	AF088...	1742-...	100.0

lalign: no_name

lalign: no_name

Name	Genomic Range
AF088282_1intron3	1742-1760
AF088282_1intron3	2000-2018
AF088282_1intron3	1742-1760
AF088282_1intron3	2000-2018

Position 1788 Feature AF088282_1intron3

L29074_intron12_1: 3 introns aligned.

L29074_intron12	1	2414
AF088282_1intron3	1742	1760
AF088282_1intron3	1742	1760
AF135025_1intron3	1742	1761
AF135025_1intron3	1742	1760
AF135025_3intron3	1742	1761
AF135025_3intron3	1742	1760

Global Coordinates: 41686-41704
Motif Sequence Length: 19 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAA

These two motif sequences share the same Apollo file.

L29074_intron12_2: 3 introns aligned.

L29074_intron12	1	2414
AF088282_1intron3	2000	2018
AF088282_1intron3	2000	2018
AF135025_1intron3	2000	2018
AF135025_3intron3	2000	2018

Global Coordinates: 41944-41962
Motif Sequence Length: 19 b.p.
Motif Sequence: AAAAAAAAAAAAAAAAAAA

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

Discussion

The presence of sixteen completely conserved sequences shared between seven completely unrelated genes cannot be attributed to coincidence. Neither can the statistical values shared by the sequences in terms of length and type. Of the sequences, seven were determined to be polyA sequences; the remaining nine were polyT sequences. The majority of each type were pure sequences with no corrupting residues; some sequences of each group had one or more intervening cytosine or guanine residues.

The presence of polyAs is noted in introns only 20 to 50 base pairs upstream from the 3’ end of the intron. None of these sequences are located in this region, so it may be assumed that they represent a different motif occurring at a different site in each intron. The presence of polyTs is not noted within introns, so these sequences are novel as well.

Since conservation of motif sequences requires regulatory action, and regulatory action is only performed on necessary sequences (observe the chaotic state of intergene regions), it may be hypothesized through logic that the sequences containing motifs must have (or have had) a function now or in the past. It’s possible that so-called “junk DNA” might not be completely irrelevant when it comes to the evolution, development, and maintenance of an organism.

Future research confirming this phenomenon must be completed. It will be interesting to note whether these motifs are common to all introns from one species or across species boundaries. In addition, any possible role these sequences play in alternative splicing, the one common element among them, should be explored.

xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
x

References/Citations

"Apollo User Guide." http://www.fruitfly.org/annot/apollo/userguide.html (February 26, 2003)

Baxevanis, Andreas D., and B. F. Francis Ouellette. Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins (Second Edition). New York: Wiley-Interscience, 2001. (p. 187-212)

Brown, T. A. Genomes. New York: Wiley-Liss, 1999. (p. 212-219, 221-223, 385-387)

Feiber, A. L., Rangarajan, J., and Vaughn, J. C. (2002) The evolution of single-copy Drosophila nuclear 4f-rnp genes: spliceosomal intron losses create polymorphic alleles. *Journal of Molecular Evolution*, **55**, 401-413.

Jackson, S., Cannone, J., Lee, J., Gutell, R., and Woodson, S. (2002) Distribution of rRNA introns in the three-dimensional structure of the ribosome. *Journal of Molecular Biology*, **323**, 35.

Schwartz, Randal L., and Tom Christiansen. Learning Perl. Sebastopol, California: O'Reilly and Associates, Inc., 1997.

Takahara, K., Schwarze, U., Imamura, Y., Hoffman, G.G., Toriello, H., Smith, L.T., Byers, P.H., and Greenspan, D.S. (2002) Order of intron removal influences multiple splice outcomes, including a two-exon skip, in a COL5A1 acceptor-site mutation that results in abnormal pro-alpha1(V) N-propeptides and Ehlers-Danlos syndrome type 1. *American Journal of Human Genetics*, **71**, 451-465.

Vromans, Johan. Perl Pocket Reference (Fourth Edition). Sebastopol, California: O'Reilly and Associates, Inc., 2002.

Watson, James. Molecular Biology of the Gene (Fourth Edition). New York: Addison-Wesley Publishing, 1997. (p. 91-95)

Watts, Giles D. J. (1998) Analysis of genes within human chromosome region 11q22-23. *University of Birmingham Thesis Paper.*

Yu, Jun, Yang, Zhiyong, Kibukawa, Miho, Paddock, Marcia, Passey, Douglas A., and Wong, Gane Ka-Shu. (2002) Minimal introns are not "junk." *Genome Research*, **12**, 1185-1189.