

## *genomic cartography*

The completion of a genome sequencing project is often followed by the publication of a paper detailing the process and notable observations that can be made of the newly acquired data set. More significant projects are typically presented in the prestigious scientific journals Nature and Science, as cover stories that detail what the new findings offer the scientific community.



In many such cases, the editors have seen fit to publish an image of the data itself, ostensibly to provide the reader with a high level understanding of how the new data set fits together, as a large-format ‘figure’ to accompany the paper. Of course these are also meant to be subjectively enjoyed, since they often take the form of a wall poster, which might find a place on the wall of the reader’s office or lab. Not so much as a day-to-day research tool, but an image that provides a feel for the data—a sense of what a genome ‘looks’ like.



[ gff2ps image of human chromosome 20, ref: Science ]

Considering the circulation (64,000 subscribers for Nature, 150,000 for Science) and broad scientific audience for the two publications, these maps are arguably the most widely known information graphic for genetic data. However, there has been a lack of analysis of its implementation. It would appear that the same level of rigor that is applied to the content of the magazine has not been applied to these images. Effective communication in the text of an article is tied to explicitly stated editorial principles of clarity and brevity, and the same high standard should be applied to the figures that accompany them. While perhaps not the most difficult or pressing challenge in genomic visualization, the widespread use of these large format images makes them a useful starting point for research in how to improve such data visualizations.

A useful analogy is mapping and cartography. Similar to a genome, data for geographic features is notoriously voluminous, and the resulting representation extremely dense. Yet

cartographers have mastered the ability to successfully organize geographic data in a manner that communicates effectively. Cartography is a useful model because it synthesizes illustration, information design, statistics, and most often employs technological tools for implementation.

There are two general categories of maps, one that is used to gain an understanding of high-level geographic features like the location of mountains in the United States or comparing the size of New England to the rest of the country. Conceivably, this is the goal of the current genomic maps, to understand a broad overview of the data, and perhaps pick up a few interesting parts. For instance, in the recently published mouse genome, one of the findings was a remarkable similarity to the human genome, with a few exceptions that were mostly related to ‘mouse like’ features, i.e. an improved sense of smell. The goal would be well made maps of the human and mouse genomes that could be visually compared and such differences quickly revealed.



[ref: Microsoft MapPoint 2002]

A second type of map, such as a street atlas, has a more specific purpose, in being useful to provide navigation. Genome maps could also be treated in such a way, having a more directed purpose than simply showing all the data. It is necessary here to consider what kind of task, if not navigation, is being supported. This directs the pursuit of a more relevant picture.





[ref: Microsoft MapPoint 2002]

For a kind of ‘genomic’ cartography, four additional factors require consideration. First, that the amount of data being considered necessitates computational methods for handling it. An example file listing a set of features currently known for human chromosome 20 (one of the shortest) contains 62,000 lines. It would be impossible for a designer to manually handle this much data. much less to have the resulting image quickly become irrelevant as more information becomes known.



[ Great Lakes area, circa 1778 and 2002  
ref: Library of Congress Map Collection and Microsoft MapPoint ]

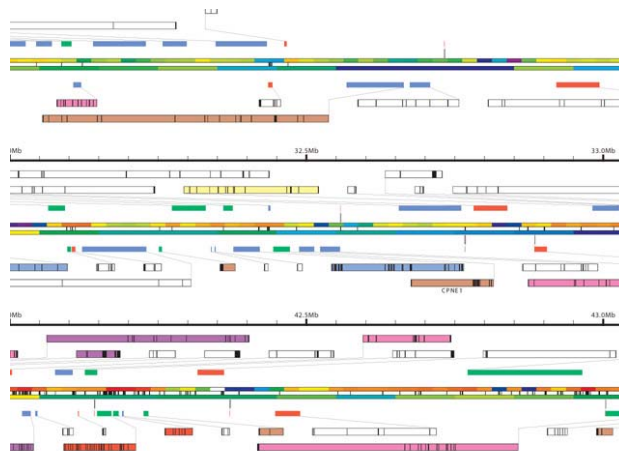
That situation leads to the second factor to be considered: that the input data is undergoing continual change. Genomic data is in a process of discovery, where new features are continually added and refined. In the months following the publication of the fruitfly genome, a series of contaminated data was discovered, causing the re-release of

the information, and rendering its published map somewhat inaccurate. The speed of discovery is accelerating, with no expectation of any kind of ‘completion’ in the near future. This expectation of change suggests that rather than focusing on a final outcome, genomic cartography should be a flexible process around a dynamic diagram that can handle change. To attain this flexibility, software presentation methods must be grounded in variability—the positioning of elements must be semi-automatic, based on rules set by the cartographer. Rather than simply applying rules of graphic design, the designer must be able to abstract some of these rules and implement them as active software elements.

Third, while it is useful to create large format printed images, the more common rendition of this work will be in interactive software. In such software, the end-user might select feature layers based on what is most important for their research, or use viewing techniques to zoom into a section, or pan along a data set. However this won’t be dealt with in depth here, instead the focus is on steps to improve the current large-format static images.

The fourth, and perhaps most difficult issue, is that these factors need to be addressed in a way that makes it possible for a designer or computational biologist to construct and/or refine a dynamic map of this kind. The previous three factors are addressed to some degree by currently available software, however their modification and refinement is not accessible to a designer, or even most programmers. On a basic level, this happens because the software code is either unavailable or too complicated. But more importantly, a clear design process and set of tools for dynamic cartography of genomic data need to be developed to augment the abilities of the designer or computational biologist.

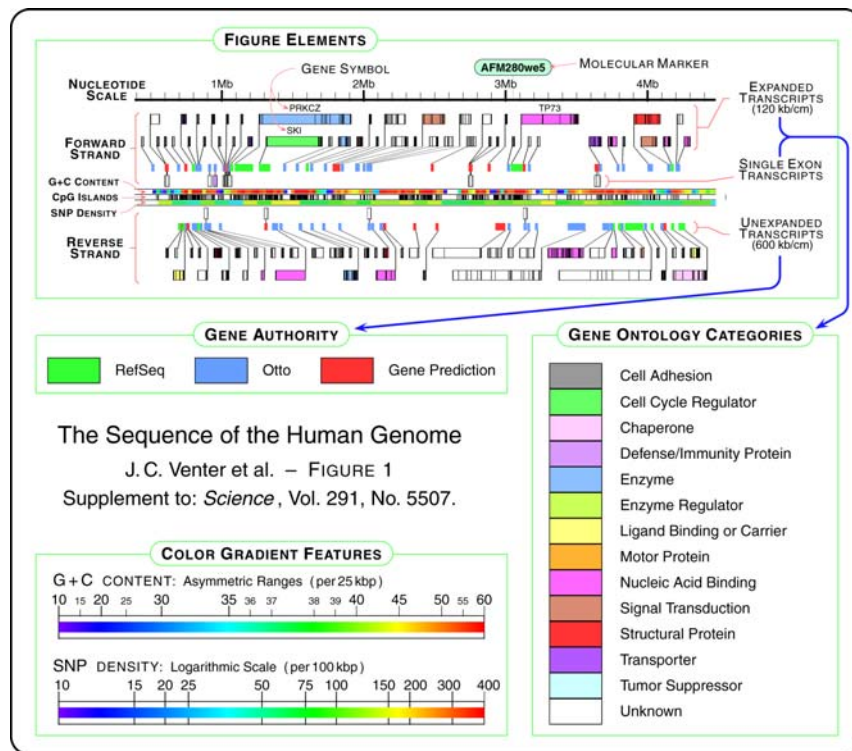
To address these needs, I have begun by first analyzing the output from the software used to create the images for Nature and Science. Next, I’ve taken steps toward a visually improved image by employing traditional graphic design methods. The redesign of the image was then implemented in software, meaning that changing the input to a new data set can modify the output, or by refining the visual design implemented in code. The process of developing that software has made clear the basis for a process and tool set for dynamic cartographic works, which will be discussed toward the conclusion.



[ close-up of output from gff2ps ]

For Science magazine's publication of the human, fruitfly and mosquito genomes, the diagrams to date have been built with a program called gff2ps, created by Abril and Guigó [ref]. As input, it reads 'General Feature Format', or GFF, a file format developed at the Sanger Center by Durbin and Haussler [ref] as a generic method for annotating genome sequences. Nature uses another piece of software, which has been used for the human [ref], and quite recently, the mouse genome [ref]. It will be addressed briefly as it suffers many of the same problems as gff2ps. For smaller genomes, such as the recent publication of the bubonic plague genome [ref], a program titled GenVision is used, which has similar issues, but for sake of brevity, it won't be covered here.

I'll begin by addressing the gff2ps software, first looking at the data behind it, pointing out a few of the visual design issues in the process, followed by discussion of a more thorough process towards its improvement. The simplest way to begin is with the map's legend, taken from the gff2ps image of the human genome, which in itself can be confusing, even for someone familiar with the data.



In order to decode this figure, it is first necessary to understand the data that's being represented and how it is acquired and stored. The building blocks of a genome are a set of bases, represented by the familiar A, C, G and T letters. The letters are each a shortened name for their chemical makeup. For instance, 'C' stands for 'cytosine', or more explicitly, 'deoxycytidine triphosphate'.

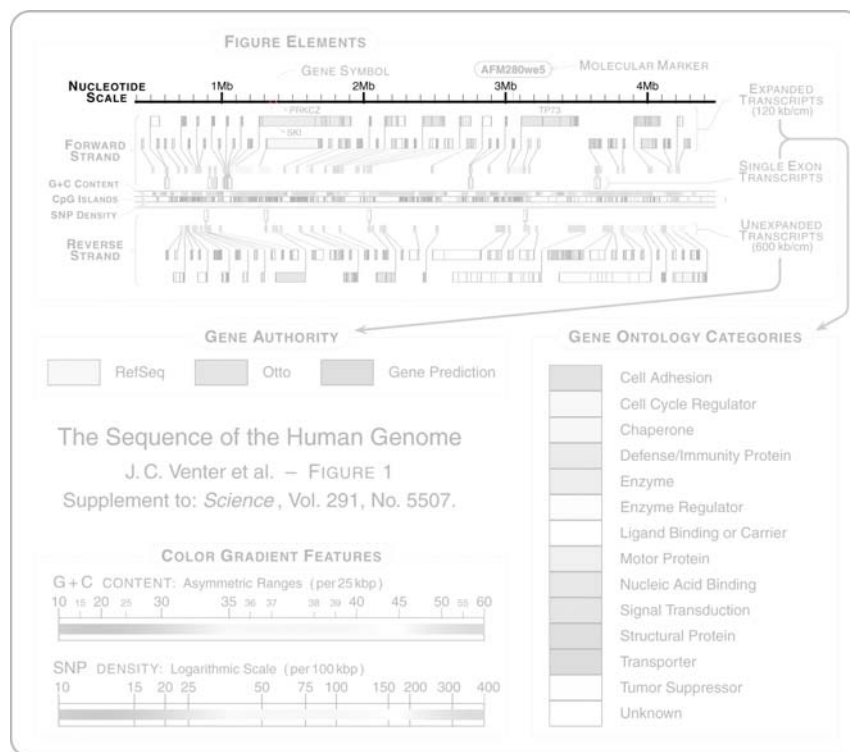
Chromosomes are composed of DNA, each a long polymer molecule, composed of many of these bases. The shortest of the human chromosomes is about 50 million letters; the longest is about 250 million. The full set of chromosomes for an organism, collectively

referred to as a genome, will be composed of anywhere from a few thousand to billions of letters. In the case of human DNA, the 24 unique chromosomes add up to 3.1 billion individual letters.

ACGTAACGCTAGCTAGCTATCGCTAGCTAGCTATCGATGATCGTAGCTGAC...

At three billion letters, however, to print the human genome as a single line of 12 point Times Roman would be more than 5,000 miles long. Clearly this is too many items to deal with directly, so it becomes necessary to present something more abstract than simply the letters themselves. Most features are not at the level of individual letters, and for those that are, they are commonly spaced hundreds or even thousands of letters apart. The need for abstraction is also necessary for a viewer who wants a macroscopic picture of the entire genome. As a result, when represented, the level of resolution is often done in thousands or millions of bases (letters), rather than 1:1 with the bases.

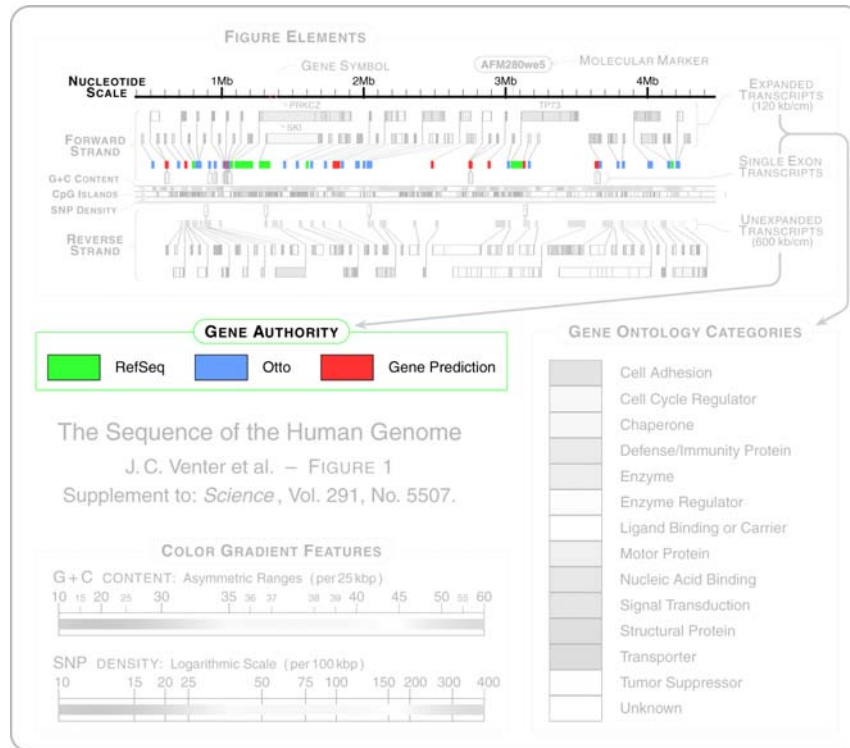
The scale shown in the legend shows major tick marks at the level of megabases, ‘2Mb’ means two million letters:



A common method for dealing with genomic data is to consider a series of ‘features’, a list of annotations assigned each with a specific location along the DNA strand. The beginning and ending point of each feature are the position of each letter involved. For instance, the gene ‘p47’ begins at letter 1,410,872 on human chromosome 20, and ends with letter 1,435,619. A gene is a subset of letters that are read by the cell’s machinery and used as blueprints for a set of amino acids, which may be used as building blocks or parts of proteins.



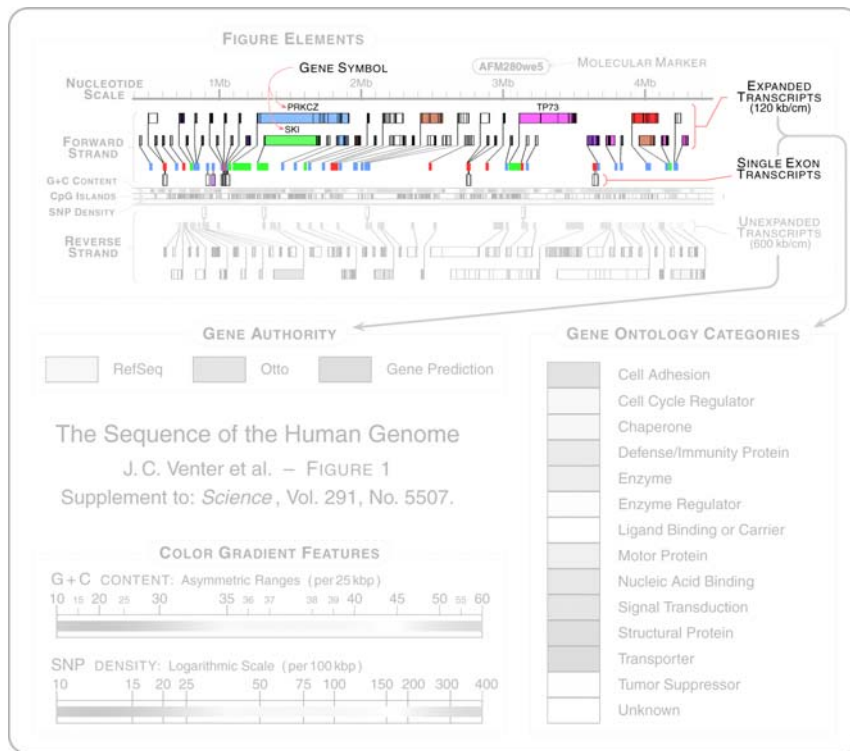
The *gff2ps* program shows the genes as boxes along the strand of DNA. Their rectangles are colored based on the *authority* from which the data for the gene was procured. This might be a sequencing center or the online database, and is considered useful because data in any given representation is often pulled from several disparate sources.



One issue to note is the amount of distance between the nucleotide scale, which is above the entire row, to the genes themselves. Since the gene rectangles are the data that actually fits along the scale, it should be closer to the scale itself for a more accurate comparison.

The gene rectangles are connected by lines to expanded versions just above them, to provide a better sense of the relative sizes of genes with one another. The expanded versions are broken into two rooms to provide for adequate space.

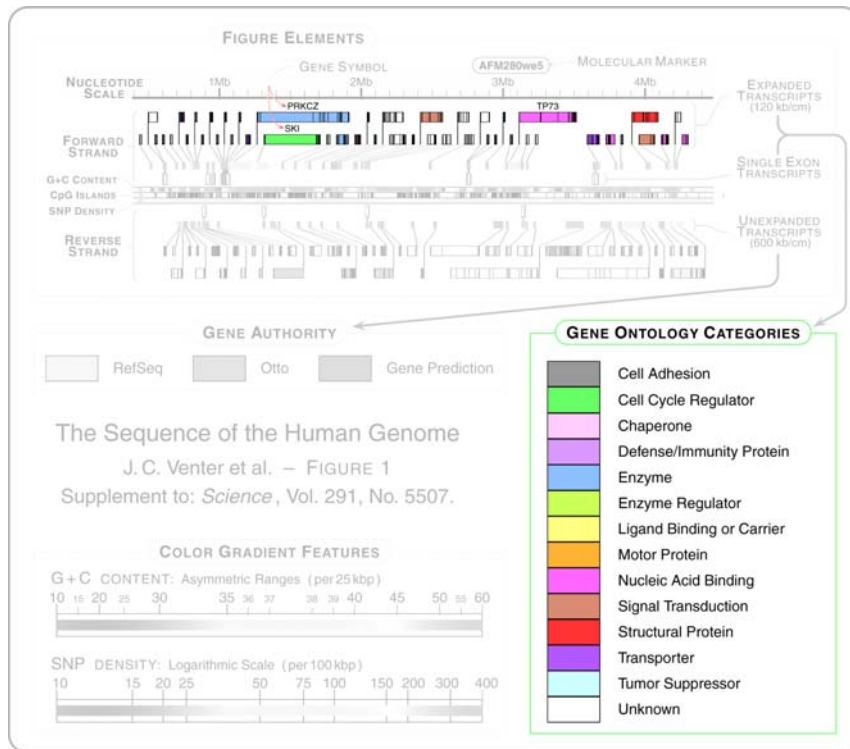
Along the genes, *exons* are shown as black stripes along the colored rectangles. Exons are subsets of a gene made up of discontinuous parts. The genes of more complex organisms, like humans, are often separated this way. Each exon is stored in a GFF file as a start and stop point that describes the portions inside the gene that are actually relevant. The entire box of the gene, save for the black stripes of the exons, represents area of the genome that is between the start and stop point of a gene, even though it is not in fact used. The use of a black strip along a colored field seems flawed because the black would appear to infer an absence of data, when in fact, it is the most essential data in the gene. This is problem is reinforced by the use of black elsewhere in the diagram to denote frames around data, rather than useful data in and of itself.



Only longer genes with multiple exons are shown expanded. Below the line of the actual size, unexpanded genes, a small rectangle is drawn for each of the single exon unexpanded genes. This additional marking seems superfluous, as it introduces no additional information or understanding about the data. If left out from the diagram, it could easily be assumed that since only multiple-exon genes are expanded, unexpanded genes would infer a single exon. Removing such a marking would help reduce clutter in the image, making it easier to understand.

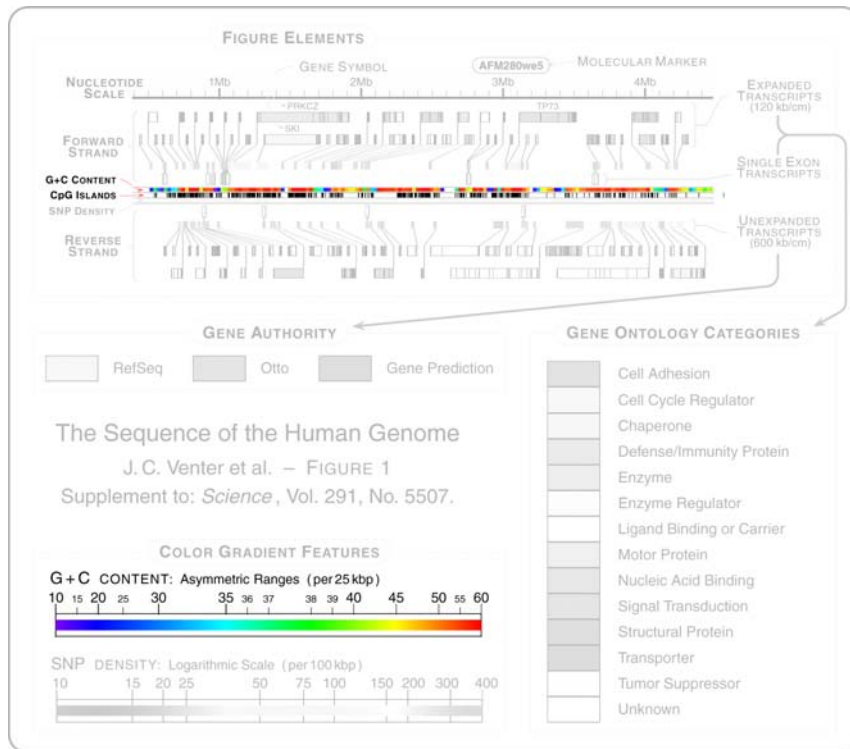
The expanded rectangles are assigned one of fourteen colors based on their *ontology* specification. Genes can be categorized in a few dozen functional categories, enumerating these known and predicted functions are important in presenting a detailed picture of how a genome ‘works’.





While perhaps one of the most important features, the ontology coloring fails to be as useful as it could because of its overuse of color. A rule of thumb in visual design is that categorization with more than 4-6 colors fails to work, because of limitations of the brain to categorize and differentiate solely based on color. This suggests that a reduced number of colors would be useful, which will be discussed later.

The information found on the next track, G+C content, is the percentage of data that is Gs and Cs. An above average level can be an indicator of ‘gene rich’ areas. In the diagram, the level of G+C Content is shown as a single line colored based on a full spectrum color scale:

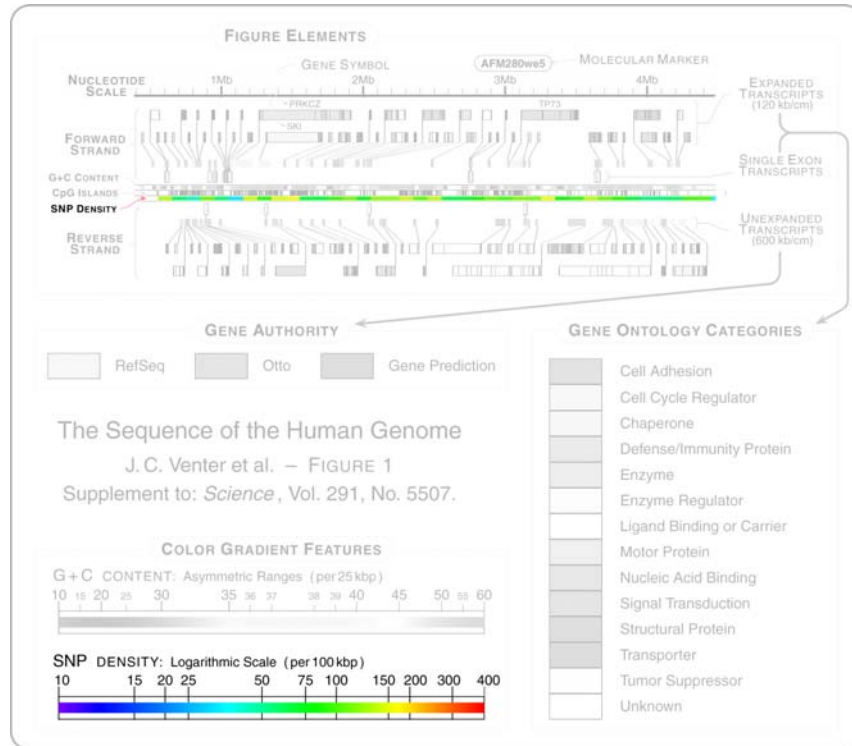


Full spectrum scales can be difficult to read, and in this case, it becomes a problem because the amount of color used for other kinds of features in its immediate vicinity. If one were to approach the diagram looking for areas of high G+C content, they must first internalize the color scaling, which is itself asymmetric, then apply that to reading the image. Rather than so much detail, the data would be better suited with a simpler scale that gives an immediate impression of low, medium, and high without necessitating its comparison to the scale.

Just below the G+C content track is another denoting *CpG islands*, which are GC rich areas commonly found before a gene. In the human genome, more than half of genes are ‘downstream’ or further along the chromosome just past a CpG island. A visual inspection reveals that the areas of high G+C content (those in red) are commonly nearby markings for the islands for CpG islands.

The use of a framed white box presents additional visual noise, because the line thickness of the border around the box is often larger than the span of the data it represents, which causes much overlap to occur. As a result, the data becomes obscured. For instance, longer regions have a white interior, and smaller regions have none. A few larger regions, wouldn’t command as much visual attention as many smaller ones, because the smaller ones attract much more visual attention with the weight of the black markings, even though the larger regions would arguably be more important. Removing the black border and simply filling the interior of the rectangles would show the underlying pattern with less obscurity.

Another track details the relative density of SNPs, or single nucleotide polymorphisms, similarly to the G+C Content track. SNPs are locations of single-letter changes, generally brought on by the slow paced mutations that occur over many generations of an organism. Such mutations are useful in understanding populations, because they can be used as markers to track inheritance over many years. For the human genome, it is believed that a SNP can be found roughly every thousand letters.



The track has many of the same issues as the G+C Content track, where the full spectrum scale distracts, rather than showing a straightforward reading. In addition, in spite of its identical coloring to the G+C Content, it employs another numeric scale, which is proportioned differently (it is logarithmic, instead of asymmetric or even linear), and uses different units (the numbers shown are per 100,000 base pairs; while G+C is per 25,000 base pairs). Such discrepancies make it difficult to read the diagram and lengthen the amount of time it takes to familiarize oneself with its content.

Returning to the genome itself, it's next important to consider that DNA is double stranded, where the second strand is paired to the first in what is referred to as 'base pairs'. An A on one strand is always found adjacent to a T on the other; a C on the first with a G on the next (and vice versa). For example:

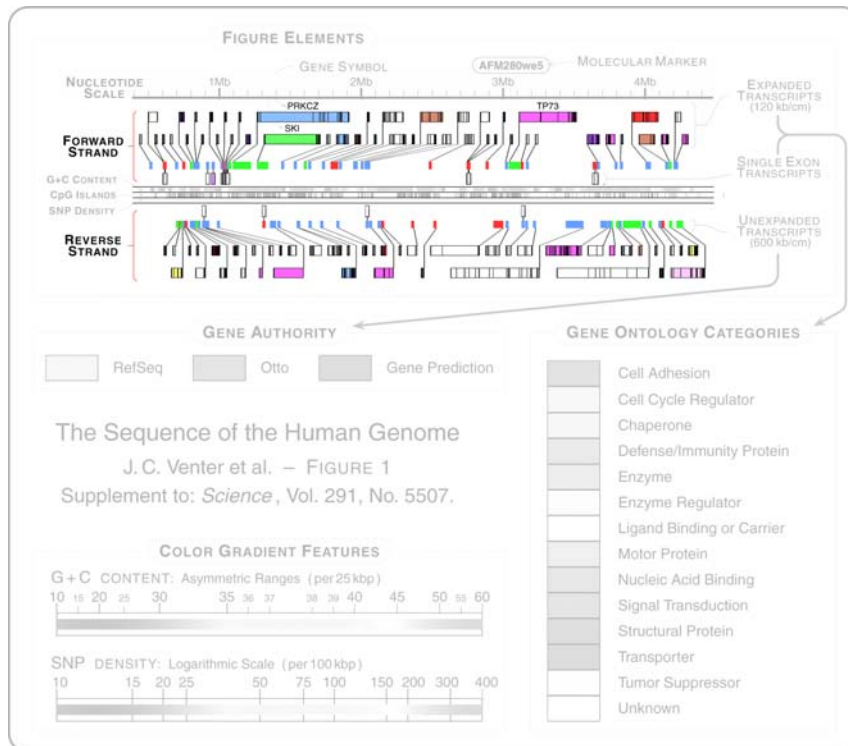
```
ACGTAACGCTAGCTAGCTATCGATCGTAGCTGAC
TGCATTGCGATCGATCGATAGCTAGCATCGACTG
```

The start of a sequence is called 5', the opposite is the 3' end, the names referring to the carbon atom to which it is connected. The two strands are wrapped around one another, in the familiar shape of a double helix.

When paired, the second strand is in the 3' to 5' direction:

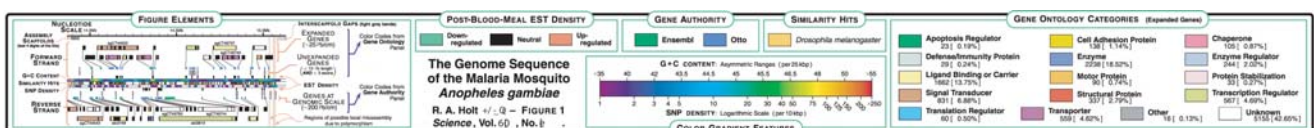
```
5' ACGTAACGCTAGCTAGCTATCGATCGTAGCTGAC 3'
3' TGCATTGCGATCGATCGATAGCTAGCATCGACTG 5'
```

The second strand may also be read by the cell's machinery (but in the 5' to 3' direction, which would be backwards of what is seen above) this second reading provides another unique sequence of genetic code. As a result, features may proceed in either direction. In the *gff2ps* image, the features (genes, mostly) in the opposite direction (the 'negative' strand) are shown below the center tracks.



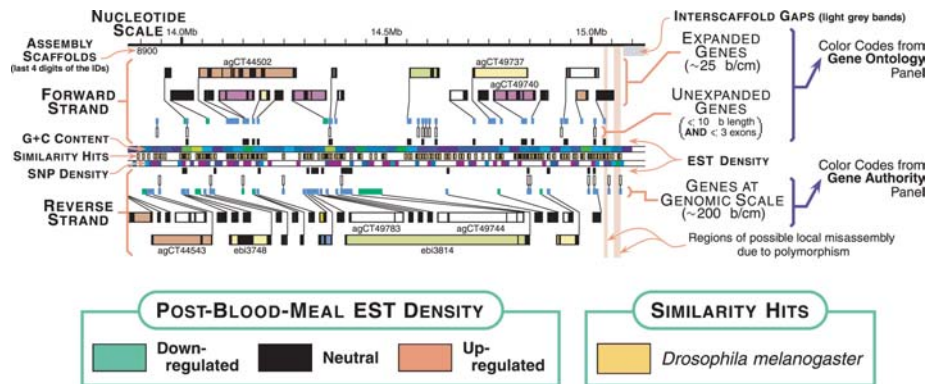
In addition, features may overlap themselves, for instance the beginning of a gene may begin before another gene has terminated.

Part of the value in having a generic feature format is that it lends itself well to the application of additional layers of annotation, or 'tracks'. Some tracks are more common than others, or when doing research, an individual may add their own tracks that are relevant to their particular work. For the post of the mosquito genome, for instance, additional tracks were added that were relevant to that data set. The full legend follows:



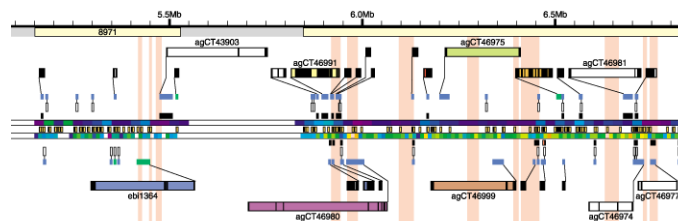


Most of the features are identical to those used for the human genome poster, so a simplified legend with just the new features follows:



It is possible to tell what genes are in use, or ‘expressed’ by measuring how much a particular gene is being read and copied. cDNA are stretches of sequence that are part of the process of copying DNA that can be easily sequenced and catalogued as expressed sequence tags. *Post-blood-meal EST density* indicates after a mosquito has bitten, which ‘expressed sequence tags’ are found more, less, or remain unchanged. For this track, yet another set of colors is used to differentiate the three states; although an improvement would be to use another symbol, such as small symbols indicating up, down or neutral.

Another track, for ‘similarity hits’, shows sections of the mosquito genome that have similarity to the genome of the fruitfly. Such comparisons are useful in understanding patterns of evolution, as well as quickly identifying features in a less analyzed genome by inferring its function based on its structural similarity to a well-studied genome. The fruitfly is considered a ‘model organism’, meaning it was selected as uniquely useful for biological study because of useful attributes like short generation span, a range of measurable physical attributes (i.e. eye color), and small size for storage in large amounts. The mosquito has not been studied as thoroughly, but much can be learned by simply comparing its genome to the fruitfly. Sections of similarity are marked with orange boxes and a black outline. These boxes have the same problem as the unexpanded single exon genes, that the black border overpowers the useful data—the orange section. Simply removing the border would go a long way towards removing the clutter created by many hits found as a group.



One more interesting feature in the mosquito representation is sections that are marked as having possible misassembly. The process of sequencing a genome is done in parts, because it is not yet possible to sequence more than about a thousand letters at a time.

The general approach is to break a chromosome randomly into many pieces of a usable length. The result is a mess of many thousands of unordered pieces. To put everything back in the correct lineup, this is done several times, so that the pieces begin to overlap one another, increasing the probability of having the proper order. The pieces are referred to as 'contigs' and are a by-product of the process, but can be useful to verify the integrity of the data. The contigs are shown above with the gray and pale yellow bands at the top of the diagram. Regions in the diagram where the background is marked with stripes of a salmon color are it's not yet verified whether the re-assembling of the individual pieces was done correctly.

For the salmon color to be in the background seems mostly appropriate, as it suggests a zone of possible confusion in the data. However, the markers should not be as large vertically, because when they sit behind the rectangles of the expanded genes, their positioning is no longer accurate, because they are not subject to the same scaling.

## *evolving gff2ps*

### *introduction*

Returning to the original line of study, the focus of this work was to improve upon a widely used diagram, and in the process, begin to outline a process for addressing similar problems in information visualization as it relates to genomics.

The redesign is meant as an evolutionary step, a diagram that could replace the gff2p-produced images, but still makes sense in the same context of publication. A more complete revision would begin with an in-depth study of the data, the problem, and the goals of the audience. In such a case it is most likely that a very different representation with more clearly specified goals than simply being able to ‘see’ the data would be more sensible. For the time being, a simpler revision is more appropriate as it sufficiently illustrates the broader aims of my research.

This analysis will focus on just chromosome 20 of the human genome, because of its availability and reasonable size (being one of the shorter human chromosomes).

Clearly, having published the gff2ps images several times over the past few years, there is some attraction in this software. At the minimum, the editors of Science have found it to be at least subjectively attractive enough that it should be included. Since this is not intended as a diagram for day-to-day research, perhaps that’s all that is needed. However it seems a shame to not push further, when so much could be improved.

Its consistent inclusion points out a lack of basic visualization tools for genetic data. The authors have successfully filled an obvious gap for software tools in this area. They have also made it freely available, along with its original source code, meaning that it can conceivably be modified and used at will, unlike most commercial visualization tools.

### *overview of problem areas*

Perhaps most dominating in the gff2ps image is its lack of a clear visual hierarchy. Visual hierarchy is the level relative importance the mind assigns to various aspects of an image. Nearly all the shapes in the gff2ps image are of a similar size; most are filled with a wide range of colors, and bordered with thick black lines. A wider range of visual elements is needed. Most elements are similarly sized colored rectangles, when they could be lines of varying thicknesses, patterned shapes, or even symbols like arrows or other glyphs. Elements have implicit meanings in terms of hierarchy—a circle may draw more attention than a line, and text carries more weight than a background filled with color. By exploiting the qualities of a range of elements, a hierarchy can be established in the image.

Related to the visual hierarchy are issues of sizing and placement. Most of the elements in the gff2ps image have a similar spacing to one another, which becomes confusing when several rows of genetic data are shown together, because it is difficult to know immediately how the rows are separated. The size of elements deserves further attention, because relative sizes are perceived pre-attentively, meaning that generally, no conscious work is done by the brain to determine the relative size of two objects. So with size being understood so quickly, it can be a high priority consideration for quickly establishing visual hierarchy in the image.

One of the more striking qualities of the gff2ps image is its overuse of color, which diminishes its meaning. Color-coding is used for almost every element in the image, undermining the ability for the viewer to learn how to read the image, because the same colors are assigned to entirely different tracks of data, even tracks within a close vicinity of one another. It's not necessary to use color so heavily when other visual elements, like lines, patterns, or glyphs could instead be used to differentiate between varied kinds of data.

### *implementation methods*

The revision has been built using Proce55ing, a programming environment, language, and set of software libraries developed by Casey Reas and myself. This software works as a kind of sketchbook for the computational designer, with a focus on rapidly developing visual images and interactive works. Simplified versions of many routine methods are available, such as reading all the lines of a file into a list, but its underpinnings are based on Java, meaning that the more powerful features of a full-fledged language are available.

As used here, the shortcomings of software development in Proce55ing as it relates to dynamic cartography experiments makes clear what kind of additional methods and features are required of a more full-fledged system. For instance, simpler methods for constraints on placement of shapes so that elements don't overlap.

It is also the hope that the software itself might be more maintainable than gff2ps, which is implemented in awk, a shell-scripting language for Unix systems. Awk is not known for its readability or maintainability, and the talent pool of programmers who can effectively develop large programs in awk is significantly smaller than that of Java.

The gff2ps program is entirely standalone, meaning that it can read GFF files and produce Postscript output that can be sent directly to a printer. Proce55ing or Java applications have a set of high-level libraries that simplify programming by offering a set of useful building blocks. For this reason, the software to produce the new image is a quarter of the size of the gff2ps program, weighing in at only about 1000 lines of code, versus 4000.



## *analysis and redesign*

The redesign begins with the scale, a single track to take the center position of the image. The scale provides a way to ground the several tracks and tie them together. This is also an attempt at having the scale closer to the data itself (in this case beneath it), rather than having it sit high above all the tracks as in the previous design. On top of the scale is a gray bar that will be used for additional layers of data, but its gray background acts as a visual anchor to draw the eye along the line.

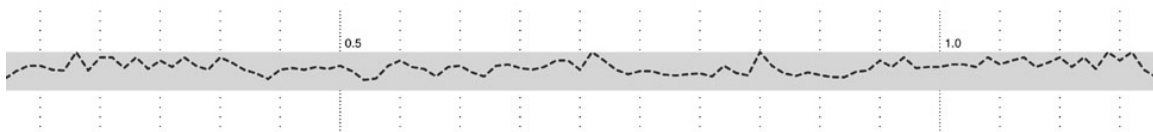


With the units unchanging throughout the diagram, it's not necessary to include the “Mb” distinction for every marked location. That is, instead of “12.0Mb” followed by “12.5Mb” and so forth, a simple “12.0” and “12.5” should be sufficient. This should make the scale itself easier to read, “12.5” being more readable without the “Mb” aside it. Additionally, the “.0” is superfluous to adjacent to a number, because no notion of ‘significant figures’ comes into play,

In addition, for the tick marks shown on the megabase scale, the tick marks are modified to relate to the numbers represented. That meaning, rather than 10 or 15 tick marks between 12.0 and 12.5, that 5 marks are used in the redesign.

Text in gff2ps output is wedged between shapes with little breathing room for readability. So a 10% reduction is employed, without much loss of clarity.

Replacing the full spectrum scale, the relative levels for the SNPs have been replaced with a dotted line graph. The minimal line graph is clearer at conveying values of low, medium, and high; or comparing two regions than the spectrum scale. The numeric specifics that are intended to be conveyed by the full spectrum scale are lost in the previous diagram because it's too much detail.

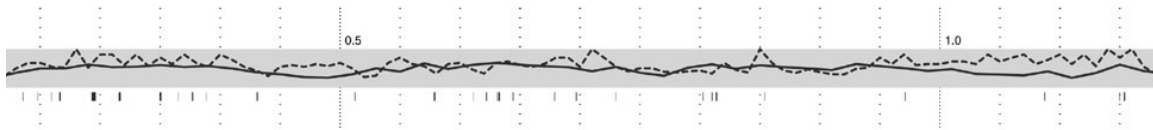


A more functional version of the diagram might instead point out which SNPs are useful, like showing which occur inside a coding region and cause a change in the amino acid that's created as a result.

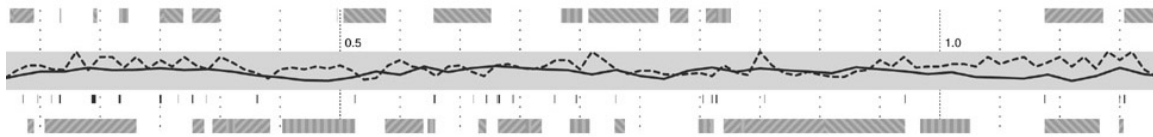
Another consideration would be to bin and smooth the values. Binned, in the sense that only 3 distinct values – representing low, medium, and high might be represented, or even 5 values, to show slightly more nuance. This is a case where more can be conveyed by limiting the amount of information. Because the previous rainbow scale and

enormously varying levels produce so much visual noise, it's questionable whether they convey any information at all. The bar graph representation is often used for circular genomes and is an improvement, however the enormous amount of variation would seem irrelevant, and could rather be smoothed to reinforce a general overview rather than specific details. The details should be left to other representations anyways, where even more quantitative specifics (numbering for exact positions, etc) are required.

Replacing the band of full-spectrum scale color, GC content is now shown with a simple line graph. The middle of the graph is the average content level  
Not much attention is paid



Next are the genes, in a similar position to their location in the gff2ps image. The three colors previously used for gene authority have been replaced with one of three patterns (left leaning, right leaning, or straight), because the importance of this particular bit of data is not high enough to warrant the use of color. They are intended to be mildly low contrast and not draw too much attention, however if one were curious as to which authority were used, it would be easy to discern based on a quick glance at the legend.

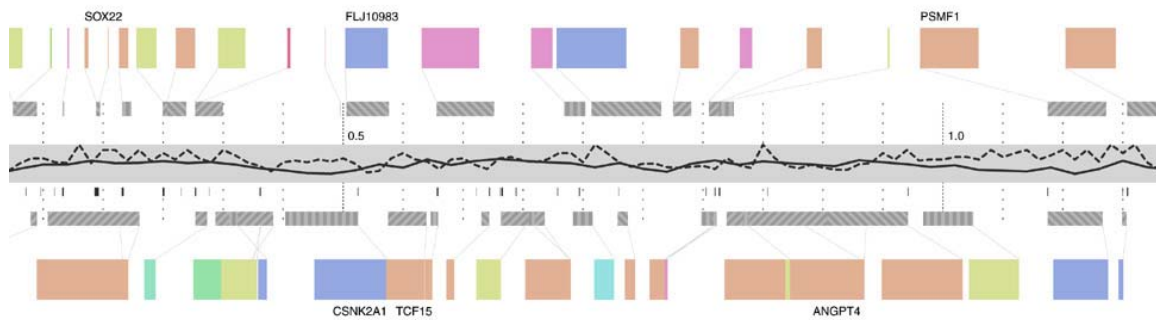


Note also that the genes are in a closer position to the scale, making location information, and the size of individual genes more apparent.

The expanded version of the genes are again colored using their ontology information, as this is likely one of the most important features being shown on the diagram, because it is one of the clearest indicators of how the genome 'works'. The expanded genes are also considerably thicker vertically, so as to take more visual prominence in the image. The black border on the boxes from the previous image has been removed, since it had no specific use, and became instead a visual distraction.

As a subtle point, the gene names have been moved to the side of the gene where it starts. This allows the eye to follow the line to or from the main track and directly to the name, and reinforces the direction in which the genes are 'read'.

The colors have been shifted to use a fixed level of saturation and brightness, and hues that are evenly spaced along the scale so that the individual colors are maximally different from one another. The largest ontology category covers 40% of the data and so the orange was chosen as the least jarring of the colors, and also works well with the orange in not taking too much prominence.



The fourteen colors used for the gene ontology categories are too many, because for categorization, the mind can only handle about five colors to at a time. Because the function of a gene is likely to be one of the more telling aspects of a genome-wide map, color would seem appropriate, though even symbols or even text could be used as well.

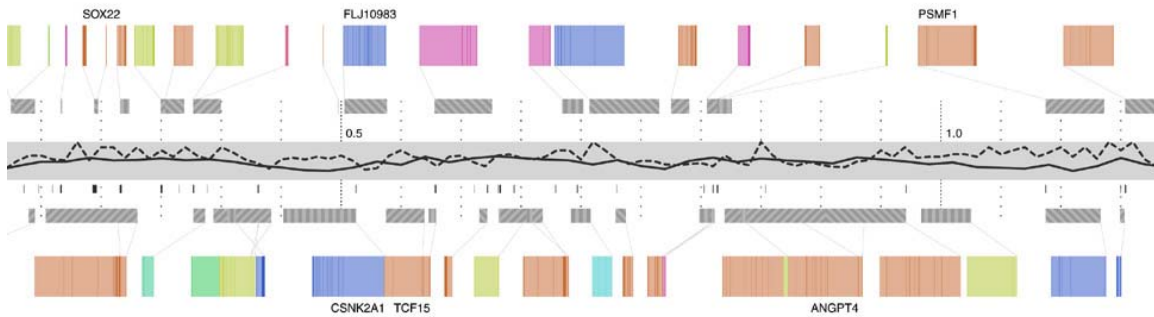
One solution is that all fourteen categories can be grouped, i.e. 'enzyme' with 'enzyme regulator' as mentioned, or that all the regulators fit together. In meeting with Dr. Altshuler, his suggested categories can be broken into five as such:

1. metabolism, biochemistry, physiology (enzyme, enzyme regulator, transporter)
2. signaling (nucleic acid binding, signal transduction)
3. structure (structural protein, motor protein, cell adhesion)
4. cell division, replication (cell cycle regulator, tumor suppressor)
5. immunity (defense/immunity protein)

with 'other' as a sixth to be treated differently. Sub-categories within each might be treated differently, for instance with a cross-hatching or pattern.

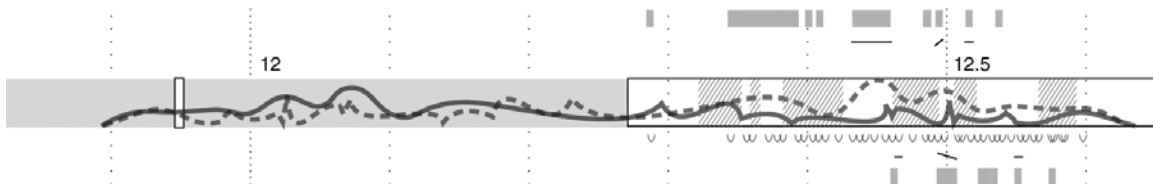
However, he also pointed out that the categories themselves are poor, and probably not useful in breaking down function. Because this is one of the strongest elements in the image, this clearly needs more work in the future.

In the gff2ps output, exons, the actual coding regions within the genes, were shown as black stripes on the gene's colored rectangle. Since the black was also used for the border around the rectangle, it gave the appearance that the stripes represented data that was not important, resulting in the relationship being reversed. In the new image, the exons are shown as a more intense variation of the coloring used for the background of the gene box, highlighting 'in use' data.



The row of single exon unexpanded genes has been removed because it is superfluous, relative to the amount of other data that needs to be included in the image. In the current representation, all genes are expanded and shown larger in the upper row. If this makes the row seem too crowded, the single exon genes could be removed, and a notation made in the legend that only genes of two or more exons are shown expanded. That way the viewer could safely assume that an unexpanded gene is made of a single exon.

Four additional features which are not included with the data for the human genome, but were included with the mosquito poster, and offer an extension of the direction for the redesign.



*Assembly scaffolds and Interscaffold gaps* this information has been moved to the center of the row, along the chromosome band. This information is more accurately grouped with the gridlines and base scale, as this is the data it directly affects.

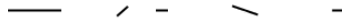
*Regions of possible misassembly* The vertical strips of a pale salmon pink are replaced with another indicator, hatched lines in the case of the schematic shown above, to denote 'problematic', without calling as much attention to the area as the vertical bands. It represents a kind of metadata--a comment about the accuracy of the data it envelops. The notion of varying degrees of accuracy in information visualization is important, especially in the case of biological data, because the field by its very nature rarely deals with numerical absolutes (for instance, two different data sets show human chromosome 20 as offset by half a megabase, or 500,000 letters of code). So to represent ranges of probability in quantitative 'samplings' of biological data requires much attention, and should be dealt with carefully.



*Fly similarity Hits* Replacing the orange boxes obscured by their own border, are a set of concave shape glyphs that envelop areas of possible similarity. Another glyph might



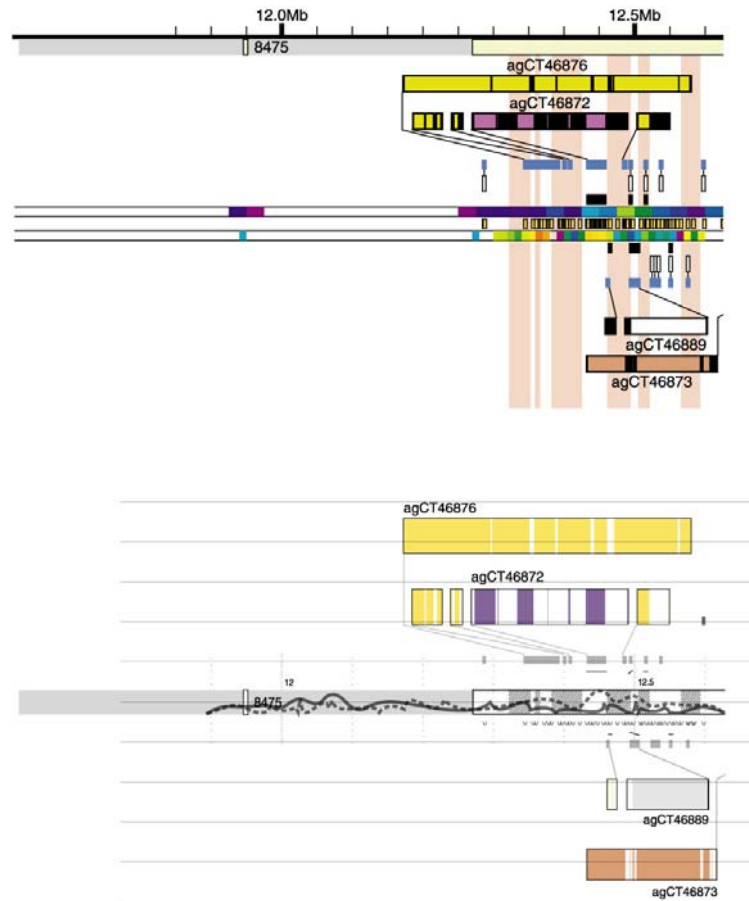
be better in other cases where the regions of similarity are larger, but the small curve is a useful indicator of possession.



*Post Blood-Meal EST Density* The row of colored boxes that indicated up, down, or unchanged expression levels are replaced with simple line glyphs for lines tilted upwards, downwards and remaining neutral. The length of the line is still related to the size of the region, and it is tilted in the direction of up or down regulation. Other glyphs, such as triangles or small arrows might be another option, depending on the amount of room in the image.

## *looking at process*

The redesign is only a band-aid for the problems of dealing with large data sets, so what follows is an outline of how to improve the process. The diagram began as a small section being redesigned in Adobe Illustrator, followed by writing software that implemented the new design:



The redesign part was tedious because of the large number of elements that had to be cleaned up. The programming was tedious because the parts to be implemented are repetitive and don't involve much problem solving. If the two parts of the process were brought closer together, they could complement one another well, but their separation simply highlights the worst aspects of both. The simple shape drawing and tweaking of Illustrator needs to be combined with the ability of specific-purpose software to handle many, many elements.

Once in software, the process of iteration continues. As can be seen from the early sketch, the coloring has not yet been worked out. While selection of colors can be done in Illustrator, the implementation of the proper scaling is done in software, and revised over several iterations of running the software, tweaking the colors, and fixing the program.

The exons still have the inverted relationship as in the original diagram, which is an example of how during implementation it became clear that the data was saying something very different than the diagram being built.

The type of tedious code in software first has to do with generic functions that deal with handling data, such as this one, which reads a file that contains a tab-separated list of all the feature information for the G+C content strip.

```
String gcPercentStr[] = loadStrings("gcPercent.chr20.tab");
int gcc = gcPercentStr.length;
gcStart = new long[gcc];
gcStop = new long[gcc];
gcLevel = new float[gcc];
gcMin = Float.MAX_VALUE;
gcMax = 0;
for (int i = 0; i < gcc; i++) {
    int values[] = splitInts(gcPercentStr[i], '\t');
    gcStart[i] = values[1];
    gcStop[i] = values[2];
    gcLevel[i] = (float)values[4] / 1000.0f;
    if (gcLevel[i] != 0) {
        if (gcMin > gcLevel[i]) gcMin = gcLevel[i];
        if (gcMax < gcLevel[i]) gcMax = gcLevel[i];
    }
}
gcSpan = gcMax - gcMin;
```

There is nothing unique in this piece of code, yet this type of code is useful for a wide range of problems. This type of code relates to the ‘data acquisition’ part of the process, that needs to be generalized, so that it is only necessary to tell the program, “load tab separated data and normalize it by its max and min in columns 3 and 4.”

The following code handles drawing the line for G+C content:

```
ai.strokeColorGray(0.3f);
ai.strokeWidth(1);
ai.beginPath();
first = true;

ai.fillColorGray(0.3f);
for (int i = 0; i < gcStart.length; i++) {
    long featureStart = gcStart[i];
    long featureStop = gcStop[i];

    if (!((featureStart > lineStart) && (featureStart < lineStop)) ||
        ((featureStop > lineStart) && (featureStop < lineStop)) ||
        ((featureStart < lineStart) && (featureStop > lineStop))) {
        continue;
    }
    if (gcLevel[i] == 0) {
        // don't bother drawing the line until good values available
        if (first) continue;

        // else it's mid-line, so end the path and reset 'first'
        ai.endPath();
        ai.beginPath();
        first = true;
    }
}
```

```

}

//double startx = (double) (featureStart - lineStart) * HSCALE;
//double stopx = (double) (featureStop - lineStart) * HSCALE;
long featureMiddle = (featureStart + featureStop) / 2L;
//long featureMiddle = featureStart;
float x = (float) (LEFT + (featureMiddle - lineStart) * HSCALE);
float weighted = (gcLevel[i] - gcMin) / gcSpan;
float y = (float) (Y + CHROM_BAND_RADIUS -
                 weighted * CHROM_BAND_RADIUS*2);

if (first) {
    ai.moveTo(x, y); first = false;
} else {
    ai.lineTo(x, y);
}

//ai.fillRect(LEFT + startx, Y + CPG_Y1, LEFT + stopx, Y + CPG_Y2);
//ai.fillRect(LEFT + startx, Y - LOCUS_Y1, LEFT + stopx, Y + LOCUS_Y2);
}
ai.endPath();

```

For something as simple as a line graph, this is too much code. Too much code because more code is a negative in any project: it means more bugs, and more difficulty in future maintenance.

Considering the half page of code required to draw a line graph, it should be pointed out that more complex shapes are even more difficult. One option for the gene boxes would be small curved boxes, or boxes tilted in the direction of how the sequence is read. The tilted version might be simple, but the curved version might be avoided because of its difficulty in implementing in software. This is because the algorithm is not necessarily simple; for instance how does one specify that the curve radius must get smaller when the size of the gene is smaller than the curve radius? Such algorithms can be difficult, but the process of developing them needs to be more like designing a multiple master font, where the outlines for multiple sizes are created, then interpolated between. The completely mathematical rendition is very difficult to visually design or maintain.

This also points out a side negative about minimalism in computer graphics. Often it is not simply because minimal forms are better, but rather that more complex forms are far more difficult to design or control in software.

Changing visual qualities like stroke width, color, or font size should be done as they are in Illustrator (or simpler), through the use of tools that allow for simple modifications. Currently, in using the Illustrator output as a guide, the values must be entered numerically in software, and tweaked the same way. This fits poorly with the design process, which is heavily oriented to tweaking and refining small details quickly.

Another missing point from the integration of design and programming is the ability for elements to have some awareness of one another. These are often called ‘design knowledge’, and can be as simple as handling the layout and spacing of the expanded versions of the gene boxes.



Even other parameters such as thickness should, when modified, update adjacent elements to move them out of the way. The movement can be tweaked by the designer afterwards, but the basics should be available.

In addition, when outputting the same data image to multiple formats, factors like layout and resolution play a role, and should be built into the design knowledge of a system. For instance, the images that follow show the same program creating output for chromosome 14 and for the entire genome. In fitting with the notion that the underlying data is continually changing, this representation must always work, i.e. fit the page properly, given a particular page size and dimensions. That sizing information needs to be fed back into the system. Currently, the method for handling such data often requires much tweaking of the software by hand, in non-general ways. If such layout knowledge could be used as inputs to the design system, it would increase the flexibility of such images.

These are a few simple starting points for a software tool. There are many more to be considered, but even if these could be addressed, it would go a long ways towards rapid creation of more sophisticated images for complex data.

## *conclusion*

Cartographers have mastered the ability to successfully organize data in a manner that communicates effectively. The field synthesizes illustration, information design, statistics, and employs technological tools for implementation, making it a useful starting point for visualization work with extremely large data sets, genomics in the case of this research.

Genomic cartography expands on geographic in four ways. First, that it must be software oriented because of the large volume of data. Second, that the representations must be flexible in a way that allows the data underneath to undergo continuous change or update. Third, that the representations themselves will in most cases be software based interactive works, which allow the user to modify what layers of features are being shown. Finally, the process needs to be specified and made accessible to a general audience of designers, followed by others who have need to represent large data sets in genomics.

This is not to say that the work will be developing a specific, tested, generally usable software application, but rather will be usable enough to demonstrate how it might be made more general. The process will be supported by a new tool set that combines the features of illustration and mathematical software. The audience for the scope of this research is myself, over the next months using it to develop representations for specific genetics work, such as the Haplotype Map project that was recently launched at the Whitehead Institute. The tools will need to evolve in a way that supports a set of useful examples in this area.

With a better set of tools to make it easier to handle and represent data, the focus can become learning to ask more relevant questions of what needs to be expressed through the visual representation of the data. Rather than strictly data driven representations that include some features simply because they're easy to measure, or strictly image driven representations that look nice but are useless, such representations gain elegance in simplicity by their ability to cleanly communicate specific ideas.