Developmental Biology **246,** 86–102 (2002)
doi:10.1006/dbio.2002.0619, available online at http://www.idealibrary.com on IDEAL®

New Computational Approaches for Analysis of *cis*-Regulatory Networks

C. Titus Brown,*[,1] Alistair G. Rust,†[,1,2] Peter J. C. Clarke,†[,1] Zhengjun Pan,†[,1,3] Maria J. Schilstra,†[,1] Tristan De Buysscher,*[,1] Gareth Griffin,† Barbara J. Wold,* R. Andrew Cameron,* Eric H. Davidson,*[,4] and Hamid Bolouri†

**Division of Biology 156-29, California Institute of Technology, Pasadena, California 91125; and †Science & Technology Research Centre, University of Hertfordshire, Hertfordshire AL10 9AB, United Kingdom*

The investigation and modeling of gene regulatory networks requires computational tools specific to the task. We present several locally developed software tools that have been used in support of our ongoing research into the embryogenesis of the sea urchin. These tools are especially well suited to iterative refinement of models through experimental and computational investigation. They include: BioArray, a macroarray spot processing program; SUGAR, a system to display and correlate large-BAC sequence analyses; SeqComp and FamilyRelations, programs for comparative sequence analysis; and NetBuilder, an environment for creating and analyzing models of gene networks. We also present an overview of the process used to build our model of the *Strongylocentrotus purpuratus* endomesoderm gene network. Several of the tools discussed in this paper are still in active development and some are available as open source.

Key Words: **sequence annotation; comparative analysis; macroarrays; gene network modeling; computational tools.**

Introduction

The study of gene regulatory networks that underlie embryogenesis requires a synthesis of multiple approaches. Abstract model building, gene discovery, genomic sequence annotation, and sequence comparisons are all parts of the overall investigation. Computational tools can considerably aid in the process in several specific ways. In the course of describing the endomesoderm gene regulatory network of the sea urchin embryo (Davidson *et al.*, 2002), we have developed five such tools. They are (and are meant to be) used in conjunction with the many related tools developed elsewhere, but here we limit ourselves to exposition of only the new tools that have emerged from our work. These tools support each step of our network analysis strategy and are indicated in red in Fig. 1. Note that Fig. 1 does not specify a simple linear procedure. Network modeling is used at every stage of the process to check the consistency of every experimental observation against all observed behaviors of the entire network. The gray arrows show the forward path, but equally important are the blue arrows, which indicate that often apparently contradictory observations make it is necessary to loop back one or more steps to reconsider and reformulate the putative network architecture.

Figure 1 presents an overview of our experimental approach and accompanying tools. We start at the top left corner of Fig. 1, exploiting the long and distinguished history of sea urchin experimental embryology as a starting point for our studies. Molecular studies of sea urchins and other species over the last two decades provide a rich source of data. By a process of iterative refinement, we first construct an abstract explanatory model of all pertinent observations about the network of interest. The purpose of this early model is twofold. First, the model serves to integrate disparate reported data and assists in developing a cohesive picture of the state of the art. Since relevant experimental results are usually of a mixed, qualitative and

[1] These authors contributed equally to this work.

[2] Present address: European Bioinformatics Institute, Wellcome Trust Genome Campus, Hinxton, Cambridgeshire CB10 1SD, United Kingdom.

[3] Present address: Altera European Technology Centre, Holmers Farm Way, High Wycombe, Buckinghamshire HP12 4XF, United Kingdom.

[4] To whom correspondence should be addressed. Fax: (626) 793-3047. E-mail: davidson@caltech.edu.

0012-1606/02 $35.00

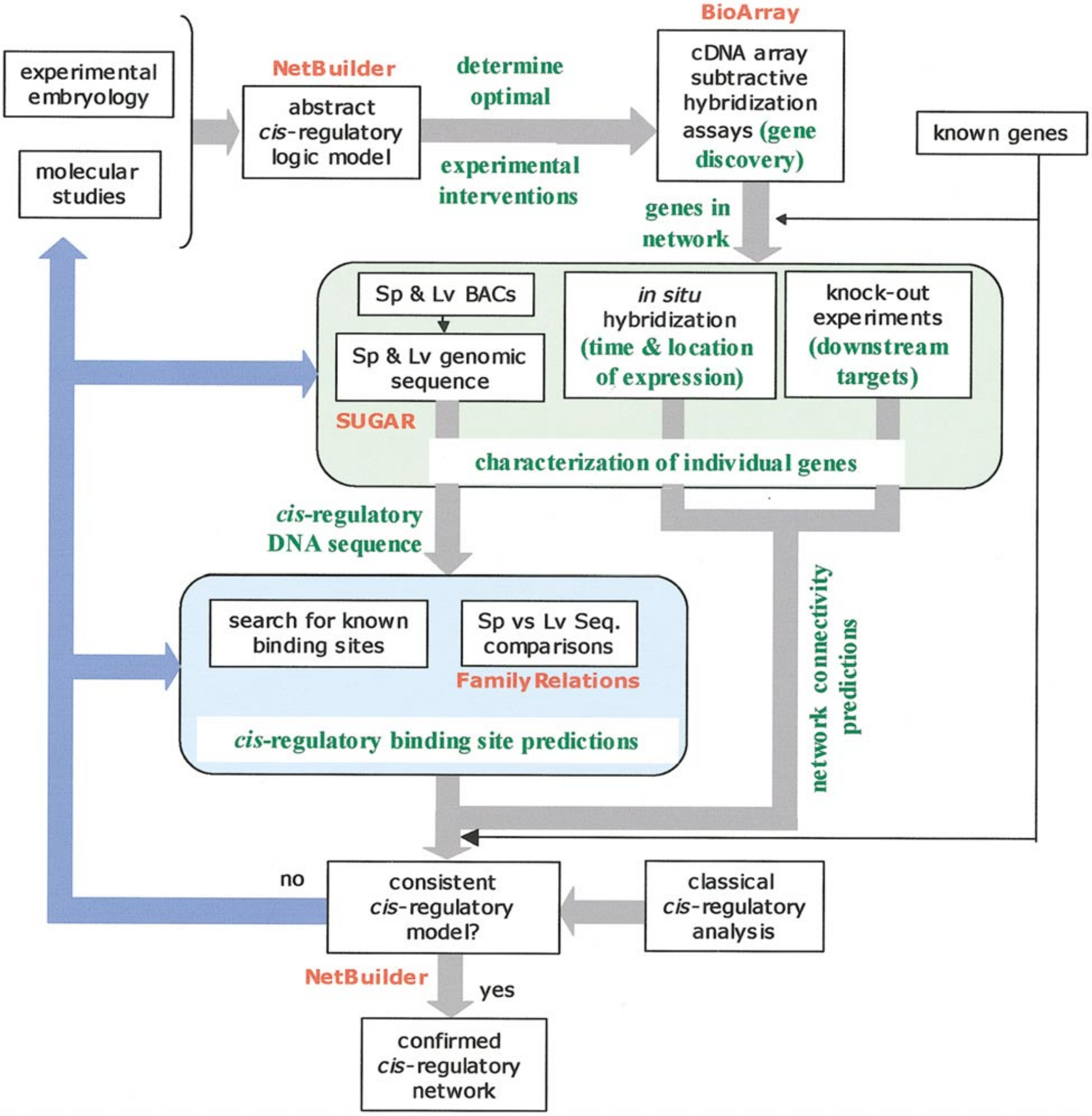

experimental embryology
molecular studies
NetBuilder
abstract cis-regulatory logic model
determine optimal
experimental interventions
BioArray
cDNA array subtractive hybridization assays (gene discovery)
known genes
genes in network
Sp & Lv BACs
Sp & Lv genomic sequence
SUGAR
in situ hybridization (time & location of expression)
knock-out experiments (downstream targets)
characterization of individual genes
cis-regulatory DNA sequence
network connectivity predictions
search for known binding sites
Sp vs Lv Seq. comparisons
Family Relations
cis-regulatory binding site predictions
no
consistent cis-regulatory model?
classical cis-regulatory analysis
NetBuilder
yes
confirmed cis-regulatory network

FIG. 1. A schematic view of our approach to revealing DNA-based *cis*-regulatory networks. The forward path is marked by the thick gray arrows. As indicated by the thick blue arrows, the actual path of discovery frequently requires reevaluation of hypotheses and results. The names of the software modules we have developed to facilitate each stage of the network reconstruction process are indicated in red. We have indicated the purpose and outcome of each step in green. Our approach starts at the top left corner of the figure with the collation of existing data into a mixed qualitative/quantitative model representing the state of knowledge at the start of a project. The model embodies a hypothesis about the genetic network thought to underlie the process of interest, and is used to plan experimental interventions for gene discovery (top right). Genes identified in this manner, combined with known genes, provide the starting point for the construction of a *cis*-regulatory model of the network. The "knock out" experiments referred to in the green "individual gene characterization" box are conducted by using morpholino oligonucleotides and engrailed domain fusion proteins, both of which have the effect of "disconnecting" genes from their downstream targets. The BACs used here to provide the DNA sequences surrounding genes of interest were prepared in earlier projects (Cameron *et al.*, 2000). Sp and Lv refer to the two sea urchin species used for comparative identification of *cis*-regulatory elements in the particular network project from which this strategy emerged (Davidson *et al.*, 2002). The sequences of homologous and potentially coregulated genes are analyzed to reveal potential transcription factor binding sites (see blue box). The results of this analysis, combined with data from *in situ* hybridizations, and knock-out experiments are combined to make predictions of *cis*-regulatory binding sites (individually and in multisite motifs) for each gene in the network. Finally, the predictions are tested with classical *cis*-regulatory experimental analysis by using artificial reporter constructs.

quantitative nature, the initial abstract models we construct use a mixture of discrete (qualitative) and continuous (quantitative) logic. That is, in some cases, the output of the system determines whether or not a gene is on in a given spatial domain, while in others its function is to control the amount of expression over time. These processes require distinct computational approaches. We are also developing a specially designed, highly intuitive tool for building and testing such models (NetBuilder) which should be generally useful for analysis of complex genetic regulatory systems.

The logical model description approach we use allows us to incrementally increase the level of detail in the model as the project proceeds. Thus, the model becomes an ongoing description of our understanding of a system. Simulation of the model provides a means of insuring against insufficient, inconsistent, or contradictory hypotheses.

In the sea urchin gene network project, we have used subtractive hybridization on cDNA to discover genes that participate in a particular developmental process, genes that were not previously suspected to be involved. This step is indicated at the top right of Fig. 1. To increase the evaluation accuracy of the radioactively labeled, posthybridization images, we have developed a new software package. BioArray allows us to look beyond heavily expressed, ubiquitous, and housekeeping genes, to identify genes whose wild-type expression levels may be as low as 10 or fewer mRNA molecules per cell (Rast *et al.*, 2000; Ransick *et al.*, 2002).

We use three complementary approaches to characterize the genes isolated by the above procedure and, where necessary, genes that are already known to be involved. These approaches are indicated in the green box in Fig. 1. *In situ* hybridization and functional knock-out experiments are used to establish the time and location of expression of each gene, and to obtain linkages, at this stage possibly either direct or indirect, between given genes and those that operate downstream of them. This information allows us to rebuild our initial abstract *cis*-regulatory network model with more detailed sets of identified genes and their connections. Additional *cis*-regulatory analysis is needed to identify the putative binding site(s) of each input incident on a gene, and establish the logical relationships between multiple inputs that control a gene's transcriptional activity.

Genetic regulatory networks must be based in the genomic DNA sequence (Bolouri and Davidson, 2002) and, in experimental terms, the relevant sequence is that which contains the genes in networks and their *cis*-regulatory control elements. Since the total genomic sequence of the sea urchin is not yet available, we have proceeded by isolating bacterial artificial chromosomes (BACs) (generally 120–150 kb) which contain the genes in the network. Their sequence was then obtained (to the point where it could be ordered completely, with only a few short gaps per BAC; see Davidson *et al.*, 2002 for summary). The gene network analysis has been carried out in *Strongylocentrotus purpuratus*, but in order to facilitate identification of *cis*-regulatory elements, we have also isolated BACs containing orthologous genes from another species, *Lytechinus variegatus*, and the sequence of these BACs was obtained as well.

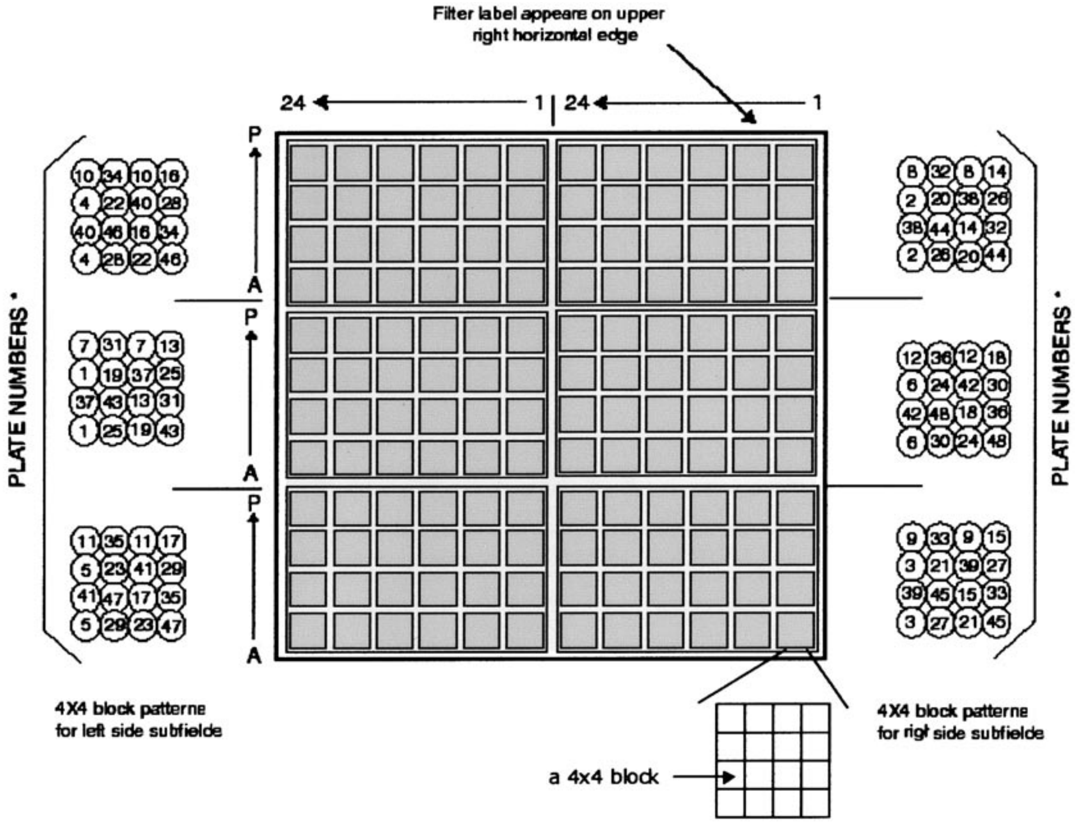
Our *cis*-regulatory analysis starts with the annotation of the sequenced BACs. As described below, we have adapted a popular annotation platform for this purpose. The Sea Urchin Genome AnnotatoR (SUGAR) allows us to search the BAC sequences with a large number of databases and DNA motif identification algorithms. Combining the results of all these searches, we identify the gene of interest within each BAC, determine the positions of all other genes or possible genes, and thereby find the DNA sequences surrounding the genes of interest, within which the regulatory sequences should lie. Then, as indicated in the blue box in Fig. 1, we compare this genomic sequence in orthologous BACs from *S. purpuratus* and *L. variegatus*, and narrow the sequence down to conserved noncoding regions, here considered as potential *cis*-regulatory modules. Our FamilyRelations program allows us to combine pairwise BLAST comparisons with more fine-grained analyses done with SeqComp, a program designed specifically for this purpose. Ultimately, a classical *cis*-regulatory analysis is carried out by using reporter constructs (Yuh *et al.*, 2002). This unambiguously tests the regions indicated by FamilyRelations for function. Finally, novel or altered predictions are reexamined within NetBuilder and a new set of experimental interventions are designed.

In the following, we describe each of these software programs in greater detail.

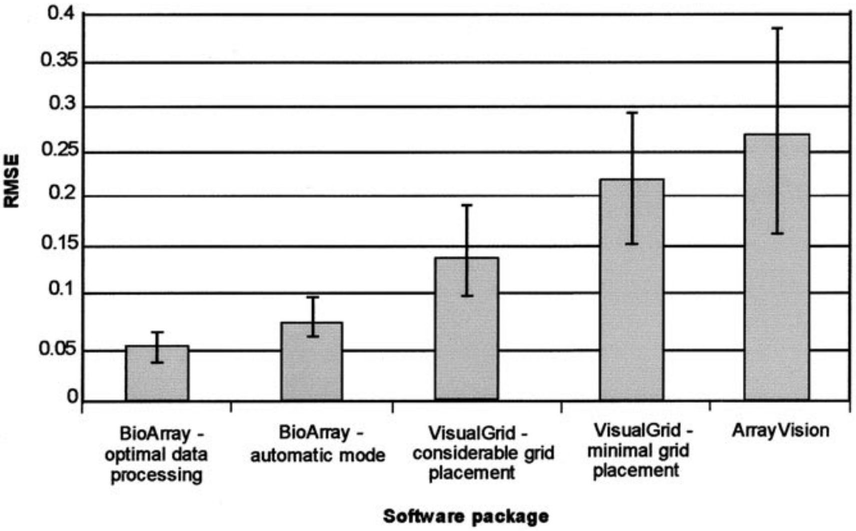
Characterizing cDNA Macroarrays with BioArray

Our gene discovery procedure centers on the use of subtractive hybridization on nylon macroarrays spotted with bacterially amplified cDNA (Rast *et al.*, 2000). The filters are approximately 22 × 22 cm and contain a total of 18,432 duplicated spots arranged in uniquely identifiable, geometrically predefined patterns in 4 × 4 blocks (Maier *et al.*, 1994.). An overview of the predefined duplicate patterns is given in Fig. 2. Further details of the filters, the sea urchin genome and DNA libraries, the hybridization procedure used, and related links can be found at http://sugp.caltech.edu.

Figure 3 compares the performance of a software package we have developed for evaluating spot hybridization intensity images (BioArray, by Z. Pan), to two of the most commonly used tools (Visual Grid: http://www.gpc-biotech.de/technologie/int_genexpress.html and Array Vision: http://www.imagingresearch.com/products/ARV_dtls.htm). All three packages were used to analyze the same images produced by a phosphorimaging scanner. All spots on 50 randomly selected filter images were used. Along the horizontal axis are distributed the different packages. The vertical axis shows the mean of the square of the difference in measured spot intensity for the duplicate spot pairs on the same filter. Comparison of the two left-most bars in Fig. 3 shows that the fully automatic procedure in BioArray produces results very close to the best we could achieve through extensive tuning (referred to as BioArray with



2



3

FIG. 2. Geometric layout of the macroarray filters used. The design was originated by H. Lehrach and his collaborators (Maier *et al.*, 1994). Each filter is approximately 22 × 22 cm. Bacterially amplified cDNA clones are spotted onto the filter in groups of 8 duplicates arranged in 4 × 4 blocks. Each gray box in the figure corresponds to a 4 × 4 arrangement of 16 such blocks. These are in turn grouped into 6 individually labeled zones for indexing purposes. The block index numbers and characters are shown along the filter periphery (top and left). The total number of spot pairs per filter is thus 18,432. The pattern of duplicate arrangements within a block is such that each pair of spots has a unique geometrical relationship. This pattern is repeated throughout the filter. The spot indexing schemes used in the 6 zones of a filter are shown to the sides of the filter. The unique relations between spot pairs are used by the BioArray software package to resolve spot pair indexing ambiguities when hybridized filter images are assessed (see text for details).

FIG. 3. Comparison of performance of BioArray and existing leading software (VisualGrid and ArrayVision). RMSE stands for Root Mean Square Error. Error is calculated as the difference in the measured intensities of a pair of spots in the same 4 × 4 block on a filter. All 18,432 spot pairs in 50 randomly selected filter images were used in the calculation. Using VisualGrid, it is possible to mark manually just the corners of a filter image ("minimal grid placement") or identify any number of corresponding spot pairs between two filters by hand. For the data shown as VisualGrid with "Considerable grid placement," we identified as many corresponding spot pairs as was visually possible for each filter. BioArray offers a similar choice: "BioArray automatic mode" refers to fully automated (single mouse button click) filter evaluation by BioArray. "BioArray optimal data processing" refers to the case where experienced users used scatter plots of spot pair intensities (see Fig. 8) to identify "noisy" spot readings and then manually corrected any mask alignment errors using BioArray's fine-tuning mask editor (see Fig. 4). Note that BioArray is not only two to five times more accurate than the other packages, it also produces the least variability in its results (compare standard deviation bars shown).

optimal data processing in Fig. 3). Note that, for the other packages, even extensive manual intervention, as illustrated by the example of manual grid placement in Visual Grid, results in much higher error than can be achieved using BioArray's fully automated filter evaluation function. Overall, BioArray evaluations are around two to five times more accurate than corresponding assessments with other packages. This greater accuracy is achieved through two measures: (1) greater accuracy in locating spots, and (2) a wider range of alternative measures of spot and background intensity. These are discussed in more detail below.

A range of factors, such as filter warping, asymmetric growth of colonies, and imprecise robotic placement, can cause spot locations to vary from a precise geometric grid. BioArray uses the predefined unique geometric relationships between spot pairs to identify unambiguous landmark features in the image, and applies nonlinear interpolation between these landmarks to correct for positional variations. The entire spot location and evaluation procedure can be performed with a single mouse button click, or the user can intervene to:

- Mark the four corners of the filter, if the image is overly warped, rotated, or otherwise distorted;
- Manually place a circular evaluation mask on top of any spot (illustrated in Fig. 4). After any such manual intervention, the user can optionally run a nonlinear interpolation algorithm to adjust the location of all other spot masks within the block, and the location of all other blocks, with respect to the manually placed masks.

By default, BioArray evaluates the intensity of each spot by evaluating pixel intensities within a circular mask placed on the spot by the above procedure. The size of the circular mask can be adjusted by the user as shown in more detail in Fig. 4. BioArray automatically adjusts the default mask size to account for different scanner resolutions. Figure 5 shows a typical evaluation mask superimposed on a filter image. For comparison, the raw image is shown in the window on the left. For the sake of legibility, only the top left corner of the filter is shown. The small blue circles are the evaluation masks placed in 4 × 4 blocks on the image. Note the extent to which each circular mask coincides with a (dark) spot. The dark blue blocks are those where BioArray was able to unambiguously allocate a spot-pair identity to detected spots. The masks in the light blue blocks were also placed automatically by BioArray, but their location was determined by an iterative process of nonlinear interpolation and error minimization using the dark blue blocks as anchors.

The input to BioArray is the raw image file produced by the hybridization intensity scanner (currently GEL and TIFF file formats are supported), an example of which was given in Fig. 5. The user interface of BioArray is similar to most Microsoft Windows applications and therefore has the same look and feel as Microsoft Word, Powerpoint, and Excel. The user interacts with BioArray by clicking the mouse on the filter image or on pull-down menus. For example, to configure the BioArray intensity evaluation procedure, the user can select from among the following options from pull-down menus such as that illustrated in Fig. 6:

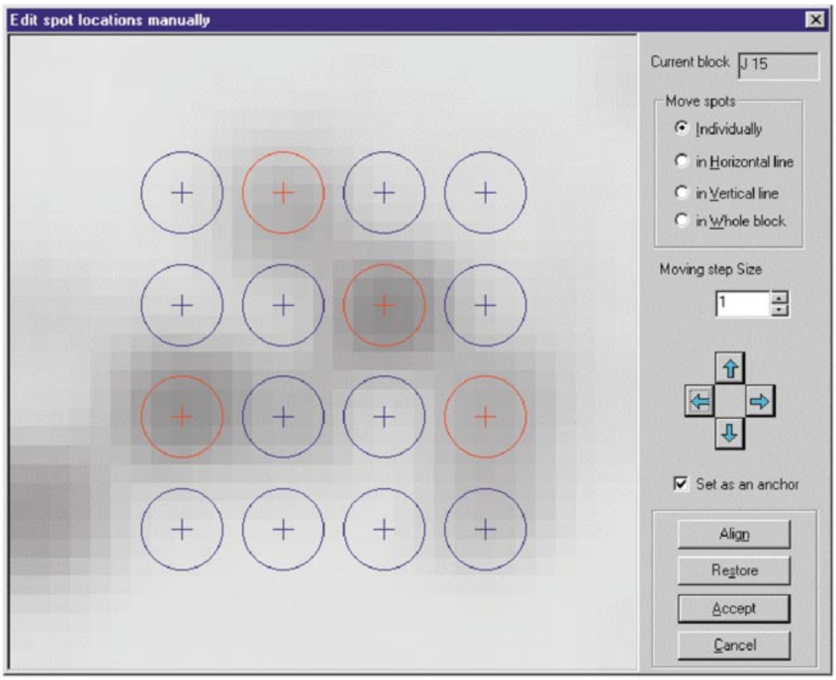
Spot Hybridization Intensity Metric

To allow the user to tune the intensity measure to particular conditions in a laboratory, and to facilitate comparison/mixing with data from other packages, BioArray offers five primary measures of spot intensity: (1) mean of all pixel intensities within the spot; (2) median of all pixel intensities within the spot; (3) 80th or 90th quantile of sorted pixel intensities within the spot; (4) the sum of all pixel intensities within the spot; and (5) the average of all pixel intensities within the spot.

The last two measures use a special algorithm to locate the boundary of a spot, then sum/average all pixel values within the boundary. They are particularly useful for identifying very large spots automatically (because their total intensity will be disproportionately larger than 1–3 above).

FIG. 4. Fine tuning the spot evaluation masks in BioArray. The blue and red circles indicate the masks superimposed on each spot. Only the pixel values inside each mask are used in the spot intensity evaluation procedure (unless the user selects the "total activity" option in BioArray; see text for details). The size of the masks can be changed by the user. BioArray calculates a default mask size based on the image resolution. The editor allows the user to move masks individually or in groups (see options menu in the top right of the figure, and the move arrows directly underneath). Red circles indicate masks which have been moved by the user. Using this approach, the user can overcome problems such as "overflow" from neighboring heavily hybridized spots, and geometric distortions in locations of spots. Unedited spot masks can be aligned to those which have been hand-tuned within the block, and the newly tuned block can be marked as an "anchor" (see the "Align" and "Set as anchor" buttons on the bottom right of the figure). If this option is selected, BioArray will then adjust the locations of all other block masks on the filter by adding the new block's location to the list of definitive landmarks and recomputing the interpolations for difficulty to locate blocks.

FIG. 5. Example views of a filter image before (left) and after fully automatic evaluation by BioArray (right). The top left corner of the same filter is shown in both windows. The small circles are the individual spot evaluation masks placed by BioArray. The dark blue 4 × 4 blocks are those to which BioArray could unambiguously assign filter location indices. The light blue blocks were placed by BioArray using nonlinear interpolation between the dark blue blocks. Note how well both types of spot masks coincide with the actual spots seen as dark areas on the filter. The red letters and numbers to the left and above the evaluated filter image are the position indices assigned by BioArray. BioArray's pull-down menu options and short cut keys can be seen just above the filter images (see text for description).



Edit spot locations manually

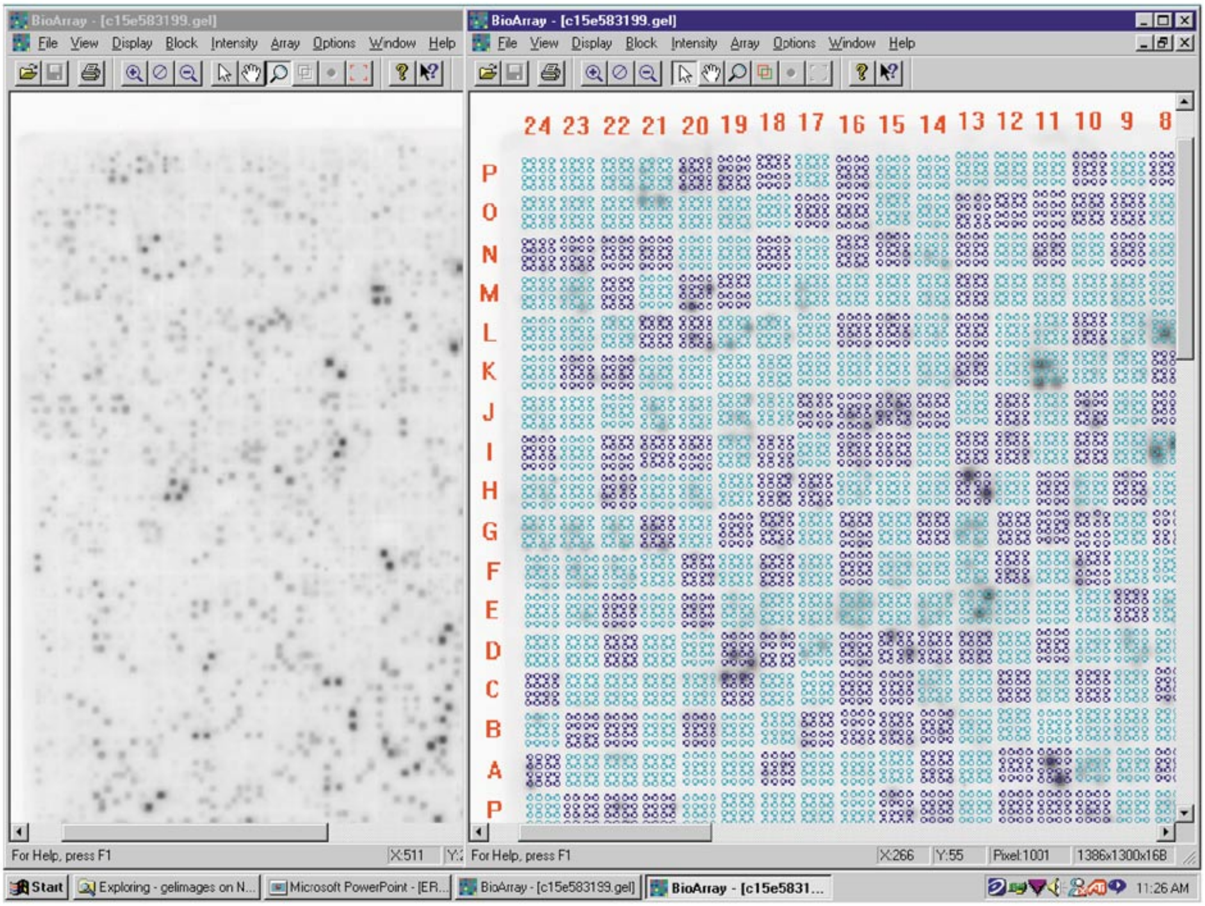
Current block: J 15

Move spots: ● Individually ○ in Horizontal line ○ in Vertical line ○ in Whole block

Moving step Size: 1

☑ Set as an anchor

4



BioArray - [c15e583199.gel]

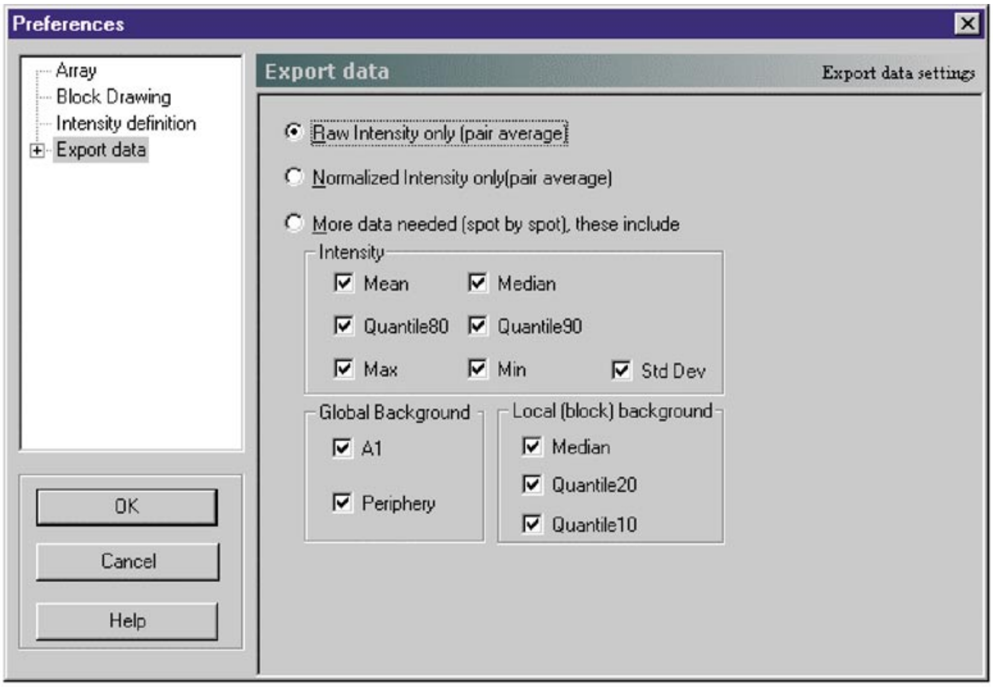
For Help, press F1 — X:511 Y:

BioArray - [c15e583199.gel]

For Help, press F1 — X:266 Y:55 Pixel:1001 1386x1300x16B

File View Display Block Intensity Array Options Window Help
24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8
P
O
N
M
L
K
J
I
H
G
F
E
D
C
B
A
P
Start
Exploring - gelimages on N...
Microsoft PowerPoint - [ER...
BioArray - [c15e583199.gel]
BioArray - [c15e5831...
11:26 AM

5



Preferences

- Array
- Block Drawing
- Intensity definition
- Export data

Export data — Export data settings

● Raw Intensity only (pair average)

○ Normalized Intensity only(pair average)

○ More data needed (spot by spot), these include

Intensity

☑ Mean ☑ Median

☑ Quantile80 ☑ Quantile90

☑ Max ☑ Min ☑ Std Dev

Global Background

☑ A1

☑ Periphery

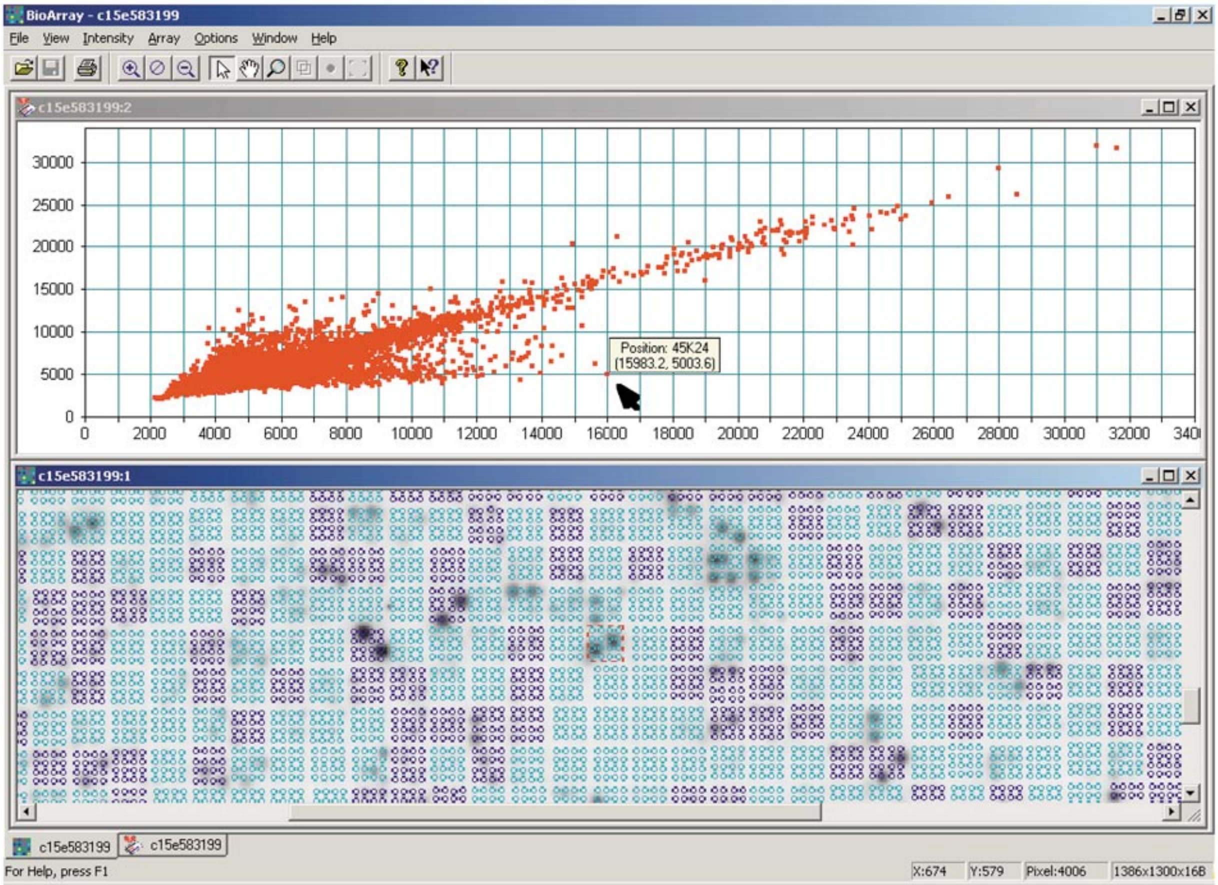
Local (block) background

☑ Median

☑ Quantile20

☑ Quantile10

6



BioArray - c15e583199

Position: 45K24
(15983.2, 5003.6)

For Help, press F1 X:674 Y:579 Pixel:4006 1386x1300x16B

7

Background Metric

Background hybridization levels can vary from filter to filter and also within filters. Global background measures do not allow for variations within filters, while local background measures, because they are based on limited local information, can be noisy. The optimum measure of background will vary according to the degree of variation of background activity within a filter, and the choice of spot intensity measure. For this reason, BioArray allows the user to measure background levels either locally (per block) or globally (for the whole filter). The options for local background measurement are the median, 10th, or 20th quantile of the spot intensity histogram for the block of interest. The global measures are either the average intensity of the filter periphery or the intensity of a predefined pair of spot locations on the filter, which are usually left blank for this purpose. The background value can be automatically subtracted from spot intensities, or the user can elect to view/analyze only subsets of spots with intensities above some multiple of background.

Normalization Metric

To make the spot intensity measure comparable across filters, the user can choose to normalize measured intensities by: (1) background measure (local or global, as defined above); (2) average activity of control spots on the filter (selected by clicking on the pertinent spots in the image); and (3) total spot activity in the filter (the user can specify any subset of spots to be excluded from this measure, for example if a portion of the filter is smudged).

Additional pull-down menus allow the user to carry out other tasks such as:

Image Viewing

While all BioArray data-processing operations are performed on raw images to maximize information extraction, the image presented to the human eye can be enhanced for ease of viewing. The user can sharpen edges in the viewed image, introduce false coloring by changing the levels of red, blue and green in the image, invert the image, zoom in/out, or use a magnifying glass to inspect the image in detail.

Exclude Subsets of Spots from the Data Set

For example: (1) duplicate spot pairs whose intensity difference is greater than a user-defined threshold; (2) duplicate spot pairs whose activity is less than a user-specified multiple of the background intensity; and (3) manually selected blocks of spots.

Produce a Scatter Plot of Selected Duplicate Spot-Pair Intensities

Duplicate spots should ideally exhibit the same level of hybridization and hence image intensity. Plotting the intensities of duplicate spot pairs against each other produces a scatter plot around the 45-degree line representing the ideal case. In BioArray, the user can draw such a scatter plot with a single mouse click and inspect outliers (i.e., duplicate spot pairs whose intensities are dissimilar) by double clicking the mouse on any point in the scatter plot.

Figure 7 shows the scatter plot for the filter shown in Fig. 5. The user has double clicked the mouse on a point in the scatter plot indicated by the black arrowhead. In the BioArray filter image window below the scatter plot (here zoomed in for legibility), BioArray has automatically highlighted the block containing the spot pair by placing a dotted red square around the block. Note that the unique filter coordinates of the block are also given in the scatter plot window. In this example, one of the two circular spot evaluation masks appears to be only partially overlapping the corresponding spot. If these spots are of sufficient interest, the user can now manually edit the location of the errant spot mask in this block to improve the readings and

FIG. 6. Example configuration menu in BioArray. The data export configuration menu is displayed. The user can choose to export data out of BioArray into a text file or a Microsoft Excel spreadsheet. The amount and type of data to be exported can be specified by selecting the appropriate menu items. The first two items in the menu provide short cut means of outputting commonly sought data (raw intensities, or intensities normalized by whatever measure was chosen in the normalization properties menu). The third menu item gives the user the option to specify a number of statistical measures of spot and background intensity. In the spot intensity field, Mean, Median, Quantile 80/90, Max, and Min all refer to the histogram of pixel intensities for the pixels within the evaluation of mask of each spot. There are two options for calculating background. In the global background field, A1 refers to a reserved location which is left blank on all filters. The "periphery" option outputs the average pixel intensity for all pixels falling outside the rectangle containing all spots on the filter. In the local background field, the median and quantile measures are calculated for the histogram of all pixel values of all the spots in the block.

FIG. 7. Scatter plot of spot pair intensities generated by BioArray for the filter shown in Fig. 5. Each red dot in the plot has the measured intensity of one spot in a pair as its X coordinate and the measured intensity of the other spot in the pair as its Y coordinate. In the ideal case, spot pairs would exhibit equal intensities and the dots of the scatter plot would all fall on a 45-degree line. Dots far away from this line indicate spot pairs with noticeably different measured intensities. This could be due to hybridization differences, or due to evaluation errors, such as misplaced masks or overflow from neighboring spots. To check the cause of the error, the user need only click the mouse on a dot. BioArray will then (1) display the unique index address of the spot pair, and (2) highlight the relevant block on the filter by putting a dotted red line around it (see filter view below the scatter plot). If desired, the user can then use the fine-tuning editor shown in Fig. 4 to improve the spot intensity evaluations in the block.

reduce the error for this particular spot pair. In this manner, it is possible to iteratively refine the spot evaluation procedure for best results.

Last, but not least, BioArray also allows automatic comparison of two individually assessed filters, as illustrated in Fig. 8 (filters must be normalized in the same manner to be comparable). The user can select the differential display colors for each filter. We usually use green for control (e.g., presubtraction image) and red for "treatment" (e.g., postsubtraction image) to produce results similar to microarray fluorescence images. It is also possible to select subsets of spot pairs by level of activity above background and other intensity measures. Double clicking the mouse on a spot in the differential activity display highlights the location of the two spots in the original filter images (dotted red squares in the filter images). Using this facility, the user can iterate through the filter assessment and comparison process to optimize readings for particularly interesting spots.

Finally, all data generated by BioArray can be output automatically to Microsoft Excel, should the user wish to perform further analysis.

BioArray was developed specifically with Genetix QBot robotic technology (http://www.genetix.co.uk/) in mind, and has been licensed to Genetix so that it can be bundled and distributed with that technology in the future. The version described in this paper is freely available to academic researchers upon request. See Availibility of Tools, below.

Annotation of BAC Sequences with SUGAR

The Sea Urchin Genome AnnotatoR (SUGAR, by Alistair Rust) is built on top of the publicly available Genotator annotation workbench (http://www.fruitfly.org/∼nomi/genotator/) (Harris, 1997), which in turn was built on the bioTkPerl widget library originally developed by Gregg Helt at UC Berkeley. Apart from some minor differences (for example, in SUGAR, reverse strand features are mapped onto the forward strand and displayed in a different color), most of the look and feel of Genotator is retained. The main difference with Genotator is that SUGAR includes several additional analyses to address our specific needs. By summarizing all related search results in a single graphical window, SUGAR provides a convenient overview which can be interactively interrogated for more detail.

SUGAR typically displays one BAC at a time, typically 120–150 kb of concatenated contigs of different lengths (individual contigs are shown in alternating colors in the main SUGAR window, see wide horizontal bar labeled "contigs" in the center of Fig. 9). Data are stored in two formats: ACEdb (http://www.acedb.org/) and GFF (General Feature Format, http://www.sanger.ac.uk/Software/formats/GFF/), which is a generic format useful for data exchange (for example, with our FamilyRelations program described below).

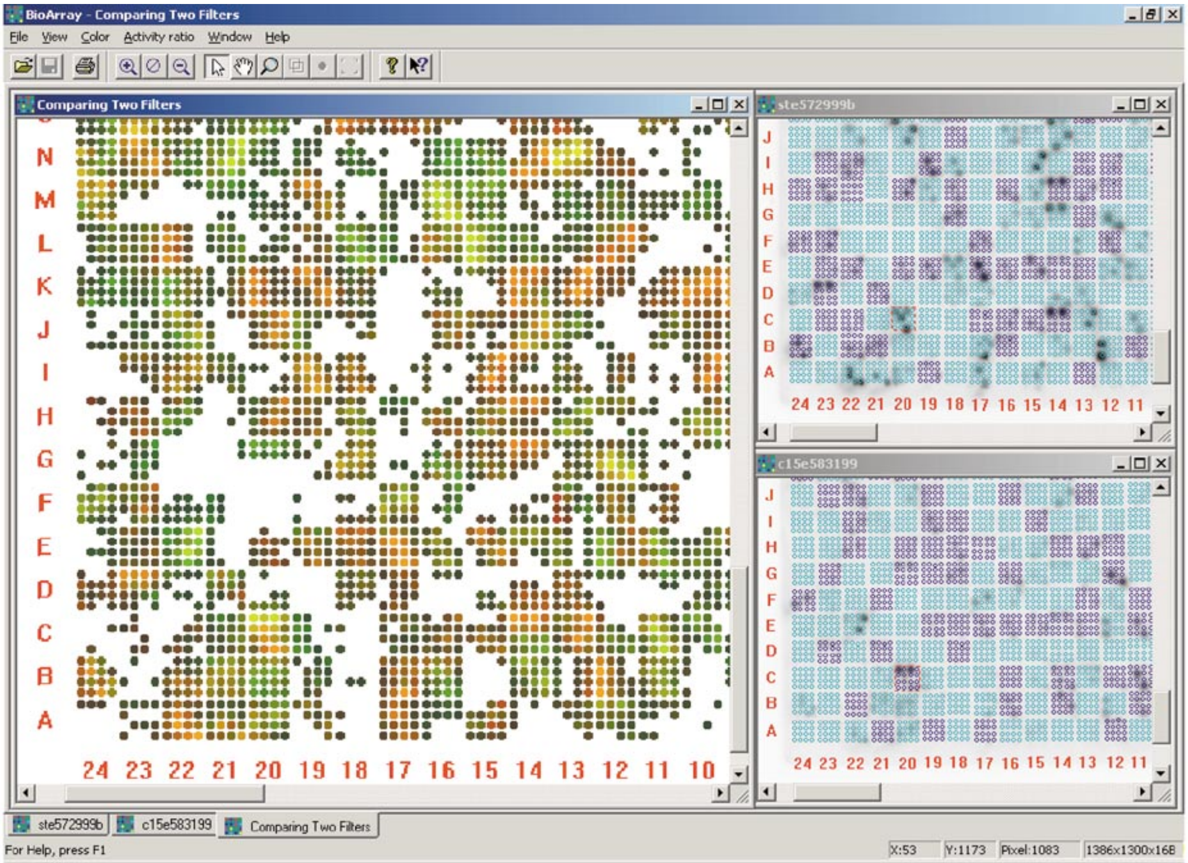
SUGAR presents the user with a great deal of useful information simultaneously (Fig. 9). BLAST search results against a number of databases are displayed in graphical form aligned against the BAC sequence. These include Swiss-Prot, GenBank, known cDNAs, sea urchin ESTs, *S. purpuratus* repeat sequences, and BAC ends. All of the sea urchin databases are accessible from the Sea Urchin Genome Project, http://sugp.caltech.edu.

Swiss-Prot hits are displayed at two different levels of significance for ease of visual analysis. Forward strand hits are colored red, while reverse strand hits are shown in green. Clicking the mouse on a colored bar representing a Swiss-Prot hit opens up another browser window in which details of the sequence alignment and other information are displayed. In addition, the top 25 hits are listed in descending order of significance. Clicking on an item in this table takes the browser to the more detailed alignment view.

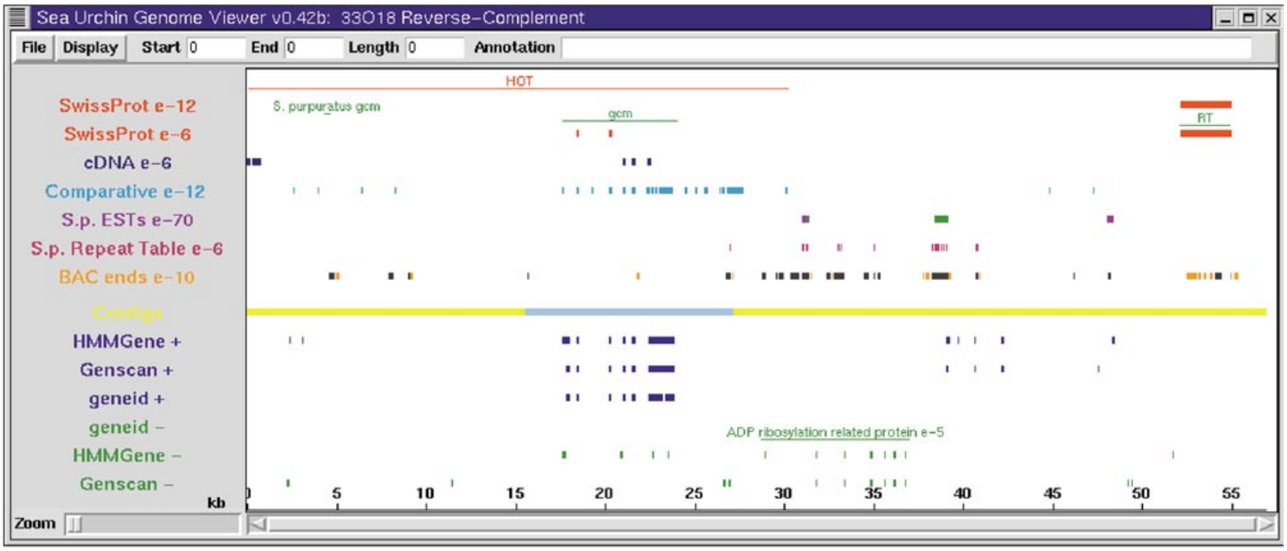
The protein matches and exon detection markers are color-coded to distinguish between forward and reverse strand hits. The BAC-end hits (this database contains 76,000 BAC end sequences; Cameron *et al.*, 2000) are color-coded to distinguish between "unique" matches (colored orange, these are in fact matches to three or fewer BAC-ends) and "repeats" (colored black, these are sequences matching larger numbers of BACs). Again, clicking on a block opens a new browser display which shows the results in more detail. At the top of this display is a more a detailed graphic of the BAC-end alignments. This is followed by a table of unique back end accession names, the individual alignment information, and histograms of the number of BAC-ends in a repeat block.

Note that, in the main SUGAR display window (e.g., Fig. 9), below the line representing the BAC sequence, SUGAR displays hits by a number of gene/exon identification programs. These are (1) HMMgene (http://www.cbs.dtu.dk/

FIG. 8. Comparison of two filters using BioArray. The bottom left-hand corners of two filters are shown in the windows on the right. The window on the left shows the visual comparison of the evaluated spot intensities in the two filters. Clicking the mouse on any block in the comparison window highlights the corresponding blocks in the individual filter images (dotted red rectangles). For this example, activity in the top filter is shown in red and activity in the bottom filter is shown in green. Thus, taking block 20C as an example, we note that spots which are more active in the top filter have a red hue (e.g., second from right on the bottom row of the block), while spots which are more active in the bottom filter have a green hue (e.g., top left spot in the block). The brightness of the colors is proportional to the level of hybridization intensity in the contributing spots. Note that many spots have a dark brown color resulting from superposition of low intensity spots in both filters. Various menu options allow the user to tailor and configure the filter comparison process. For example, here, the user has elected to remove all spots whose activity was less than a threshold value (blank areas in the figure, see text for details). Data from a filter comparison can be exported to text or Microsoft Excel files.



8



9

FIG. 9. An annotation of the *S. purpuratus* BAC sequence containing *glial cells missing* gene (*gcm*) (see Ransick *et al.*, 2002) displayed in SUGAR. The *gcm* coding sequence is at the center of the 56-kb BAC, oriented so that the 5′ end of the coding sequence is to the left. The SUGAR display is divided in two by the line indicating the orientation and location of the contigs: above this line the matches to experimentally obtained sequence data are displayed, including (from top to bottom) matches to Swiss-Prot, the location of the cDNA sequence, matches to the *Lytechinus variegatus* sequence, and the locations of ESTs, BAC ends, and repeats from the Sea Urchin Genome Project. The stringencies at which the matches were selected are displayed next to the database resource used. Below the contigs line are forward and reverse exons predicted by several programs, including HMMGene, Genscan, and Geneid. These are of use in predicting genes for which there is no extant sequences in either the Swiss-Prot or the Sea Urchin Genome Project databases.

services/HMMgene/); (2) Genscan (http://genes.mit.edu/GENSCAN.html), and (3) Geneid (http://www1.imim.es/software/geneid/index.html).

Forward-strand hits are displayed in blue, and reverse strand hits in green. Slider controls allow the user to zoom in and out and pan right and left as desired.

Identification of putative coding or regulatory regions using SUGAR is very much a matter of reviewing and weighing the evidence on a case-by-case basis. For example, a region marked with many "SU repeat table" hits and displaying black (repeat) BAC-end markings would probably not be considered as a significant regulatory or coding segment, even if other searches such as Swiss-Prot or any of the exon finders also show hits in that exact region (as might occur in the case of a recently inactivated pseudogene). Similarly, a region in which multiple results concur can be assumed to be more likely to be a true positive.

SUGAR is available under the same license as Genotator from the Web page http://sea-urchin.caltech.edu/software/SUGAR/.

Selection of Putative Regulatory Sequences with FamilyRelations

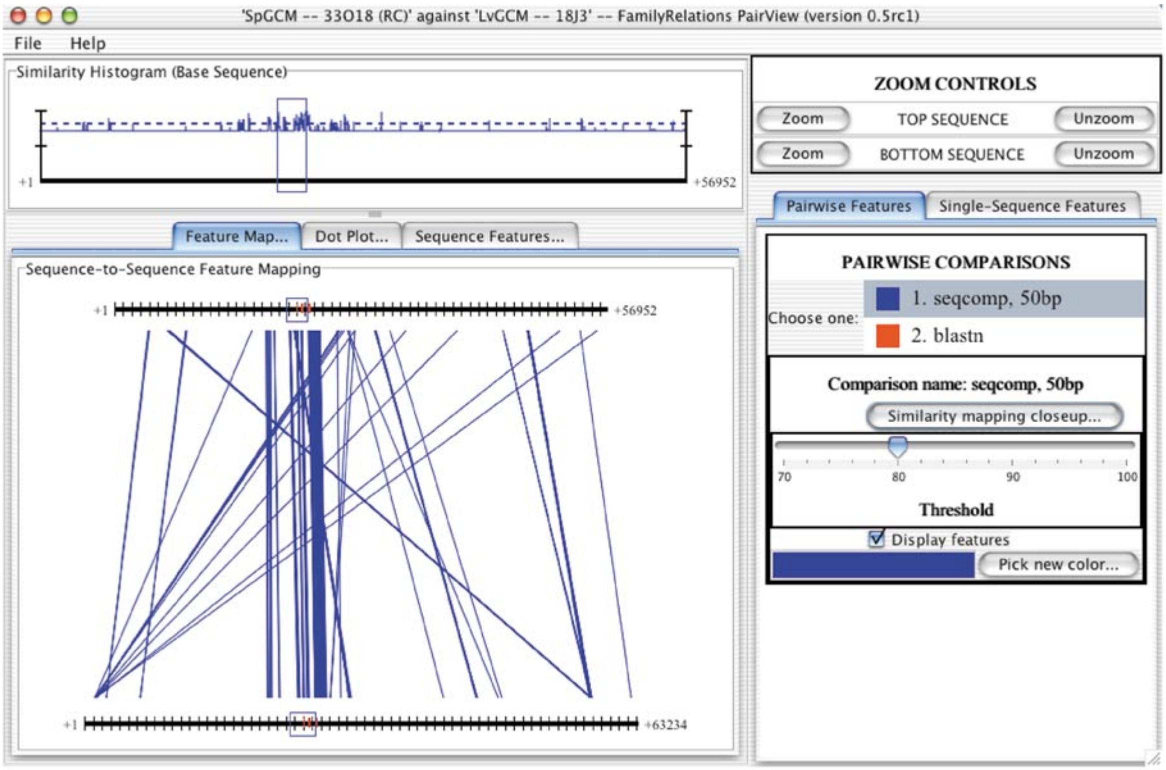
Comparison of BAC sequences containing homologous genes from *L. variegatus* and *S. purpuratus* provides a powerful evolutionary means for identifying *cis*-regulatory domains of genes. Our sequence comparison algorithm (SeqComp, by Tristan De Buysscher) compares two BACs using small sliding windows of programmable length (usually from 20 to 50 bp). It is similar in function to the Dotter algorithm by Sonnhammer and Durbin (1995). For each "word" (contiguous sequence of bases in a window) in sequence 1, a list of the 10 best matches is identified and marked with a similarity index. The main differences between our evolutionary comparison and other approaches to phylogenetic footprinting (Wasserman *et al.*, 2000) are that we perform the search exhaustively using the small sliding window on very large sequences (10–50 kb per run) excised from BACs after annotation; we allow matches between any 2 windows, irrespective of their relative locations on the BACs; and we detect reverse complement matches. A separate program (FamilyRelations, by C. Titus Brown) presents a graphical user interface to SeqComp and also offers a number of additional features.

The main view of FamilyRelations is shown in Fig. 10. The display is divided into three windows. To the right are the user controls. These include menus for selecting the sequences to be compared, and a slider with which the user specifies the threshold for displaying SeqComp matches. The top left window displays the SeqComp similarity histogram. The horizontal axis represents one of the two compared sequences (the "reference" sequence). The vertical axis, which displays the match quality, is automatically truncated to show matches above a preselected analysis threshold, usually calculated from the background expectation level for random sequences. The dotted blue line shown in the histogram window marks the user-specified display threshold. The histogram peaks correspond to regions of highest similarity between the two sequences. The regions corresponding to peaks that exceed the user-specified threshold are displayed in more detail in the bottom left window. The horizontal black lines at the top and bottom of this window represent the total length of each of the two compared sequences. The bundles of blue lines connecting the two sequences link corresponding matches in the two DNA strands. Selecting any region in this window opens up a more detailed view of the relationship between the two sequences.

An example of a detailed view, matching the region selected by the blue boxes in Fig. 10, is shown in Fig. 11. The top window (not expanded) contains a flattened repre-

FIG. 10. The FamilyRelations graphical user interface for comparing regulatory domains of homologous genes from evolutionarily close species. In this case, the analysis compares BAC sequences surrounding the *glial cells missing* gene (*gcm*) in *S. purpuratus* and *L. variegatus*, two sea urchin species that are approximately 50 My diverged (Gonzales and Lessios, 1999). The sequences are oriented to match the direction of the sequence in Fig. 9, with the 5′ end of the gene to the left. On the top left is a histogram display of the SeqComp matches between the entire *S. purpuratus* BAC (57 kb) and the corresponding region of probable orthology in the *L. variegatus* BAC (100 kb, of which 62 kb is shown). On the bottom left is a display of the regions that map between the two genomic sequences (*Spgcm* on top, *Lvgcm* on bottom) with a similarity of 80% or greater over a 50-bp window; the threshold is displayed as a dotted line on the histogram in the top left. Note that the computed similarities are bounded from below at 70%, which was user-selected at the time of analysis, as was the window size of 50 bp. On the right side of the display are user controls that allow zooming and some control of the display, including comparison color and selection threshold. Any number of pairwise comparisons can be loaded, including BLAST comparisons; we have turned off the BLASTN comparison here, to avoid confusion in the display. On the lines representing the BAC sequences, we have graphed TBLASTX matches to the cDNA sequence of *Spgcm* in red. Note that the regions of high similarity extend well beyond the red exons. In addition to the matches displayed "in register," that is, as parallel lines, there are a number of matches that fall out of register, including several that correspond to more than one region on the other BAC. These show elements duplicated within and between the BACs; in this case, they represent simple sequence repeats.

FIG. 11. A close-up view of the region enclosed by the blue boxes in Fig. 10. On the top (minimized to show only the sequence line) is a window display of the pairwise features "flattened" against the top sequence, used for selecting and manipulating the features. The middle window is a close-up view of the mapping; the red rectangles on the sequence line show the cDNA matches from Fig. 10 in more detail. The bottom window displays the sequences selected by the blue box in the middle window, with matching base pairs drawn in red and connected with lines.



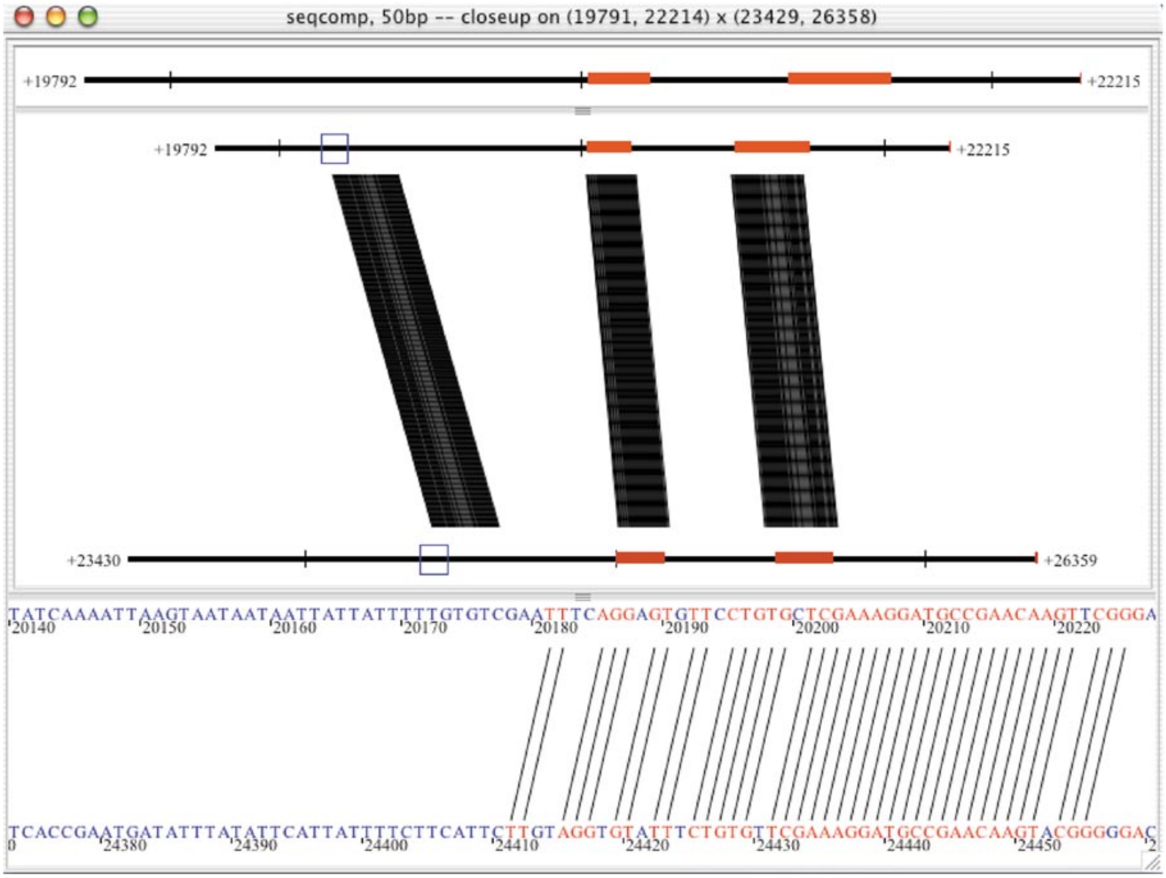
'SpGCM -- 33018 (RC)' against 'LvGCM -- 18J3' -- FamilyRelations PairView (version 0.5rc1)

Choose one: 1. seqcomp, 50bp; 2. blastn

Comparison name: seqcomp, 50bp

☑ Display features

10



seqcomp, 50bp -- closeup on (19791, 22214) x (23429, 26358)

+19792
+22215
+19792
+22215
+23430
+26359
TATCAAAATTAAGTAATAATAATTATTATTTTTGTGTCGAATTTCAGGAGTGTTCCTGTGCTCGAAAGGATGCCGAACAAGTTCGGGA
20140
20150
20160
20170
20180
20190
20200
20210
20220
TCACCGAATGATATTTATATTCATTATTTTCTTCATTCTTGTAGGTGTATTTCTGTGTTCGAAAGGATGCCGAACAAGTACGGGGGAC
0
24380
24390
24400
24410
24420
24430
24440
24450
2

11

sentation of the matches that is used for manipulating them. The middle window is a zoomed-in version of the bottom left window in Fig. 10. The actual sequences of the regions inside the blue rectangles in this window are displayed in the bottom window. Matching regions are displayed with red text, and lines connect the matching base pairs.

In addition to the above, FamilyRelations can also be used to simply display annotation data, or generate a "dot plot" of similarity regions between two sequences. Again, the user can zoom in, zoom out, or pan sideways, as required.

Because FamilyRelations is written in the machine-independent Java language, the software can run in Windows, Unix, and Mac OS X environments. SeqComp and FamilyRelations, and an accompanying tutorial, are available freely from http://family.caltech.edu/.

Building Logical Models of Regulatory Networks with NetBuilder

NetBuilder (by Maria Schilstra) is an environment for constructing mixed, qualitative and quantitative logical models of Genetic Regulatory Networks (GRNs). It is based on principles commonly used in electronic engineering to model mixed, analog–digital integrated circuits. The user interface is entirely graphical and does not require any specialist knowledge or training. NetBuilder allows modelers to build graphical representations of GRNs by placing predefined network components on a canvas, and drawing connections between components to represent interactions. NetBuilder should in the future be generally useful for analysis of complex gene regulatory systems, though it has only just begun to be used for actual research on DNA-based GRNs.

As shown in Fig. 12, the main NetBuilder screen (the network editor) has the look and feel of common Microsoft Windows applications and offers many of the same menus and tools. In the center is the user work-space or canvas. There is a line of pull-down menus at the top left of the screen. Below it are a set of common, general-purpose action icon buttons (e.g., save, print, zoom, and select). A set of general-purpose drawing and annotation tools is provided in the tool bar below the canvas window. Because the tools in these bars are essentially the same as those found in Microsoft PowerPoint, Word, and other packages, and for the sake of brevity, we will not describe them here. The column of icons to the left of the user workspace provides short cuts to the NetBuilder network symbol library (these can also be accessed through the pull-down menus). To draw a new network, the user places the appropriate symbols on the canvas by pressing the symbol buttons. The buttons are grouped into four categories. Starting from the top and taking each category in turn (refer to left margin in Fig. 12), we provide an overview of these features below.

The top category defines four types of symbols for network elements. From top to bottom (see Fig. 12), these are:

Scalars. These are used to model effects, such as the amplification of the effect of one transcription factor by another, or the existence of a maternally inherited factor. The user can specify a linear amplification coefficient and a "Hill coefficient" to specify the degree of nonlinearity of the effect of the scalar factor as appropriate.

Genes. The symbol for a gene comprises a 90-degree bent arrow symbolic of the transcriptional apparatus, and a horizontal line in two parts: the portion behind the bent arrow symbolizes the regulatory region, while the portion forward of the bent arrow symbolizes the coding region.

The convention we use in NetBuilder is to describe interactions outside of the nucleus above genes and to describe *cis*-regulatory interactions on DNA as arrows or bars which impinge on the line representing the *cis*-regulatory DNA. This convention usually leads to a convenient arrangement of genes in a horizontal line, with the space above this line representing interactions between gene products, and the space directly below each gene representing the consequences of the *cis*-regulatory interactions governing the transcriptional activation of that gene.

Interaction Symbols

Interactions can be modeled as logical And, logical Or, and algebraic addition and multiplication. The logical operators can be Boolean or continuous-valued. For a Boolean logical Or, if any one of the inputs is active, the output will be active. For a Boolean logical And, unless all inputs are active, the output will be inactive.

Switch Elements

These elements can be constructed from the interaction elements above. The following commonly encountered elements are available as NetBuilder library elements to speed up network capture: (1) A bistable switch. This element is used to indicate processes which, once they are activated, remain active until overridden by another event (for example, conformational changes in some proteins). (2) Threshold and comparison symbols. These are used to describe the relative significance of an interaction effect or a gene product. They include symbols for comparing two interaction effects or concentrations, and for determining whether an effect or concentration is above some threshold.

The second group of symbols provides a variety of connectors with which to link up the components of a network. The connectors simply transmit the outcome of interactions or states of genes from one point to another. Different symbols are provided for the sake of diagrammatic clarity. The last of the four connectors provides a generic shorthand symbol for negation. Depending on the functionality of the receiving component, this can mean a logical Not, a change in the sign of input, or the reciprocal of the input.

The two buttons that make up the third category of NetBuilder icons (see Fig. 12) offer drawing features useful

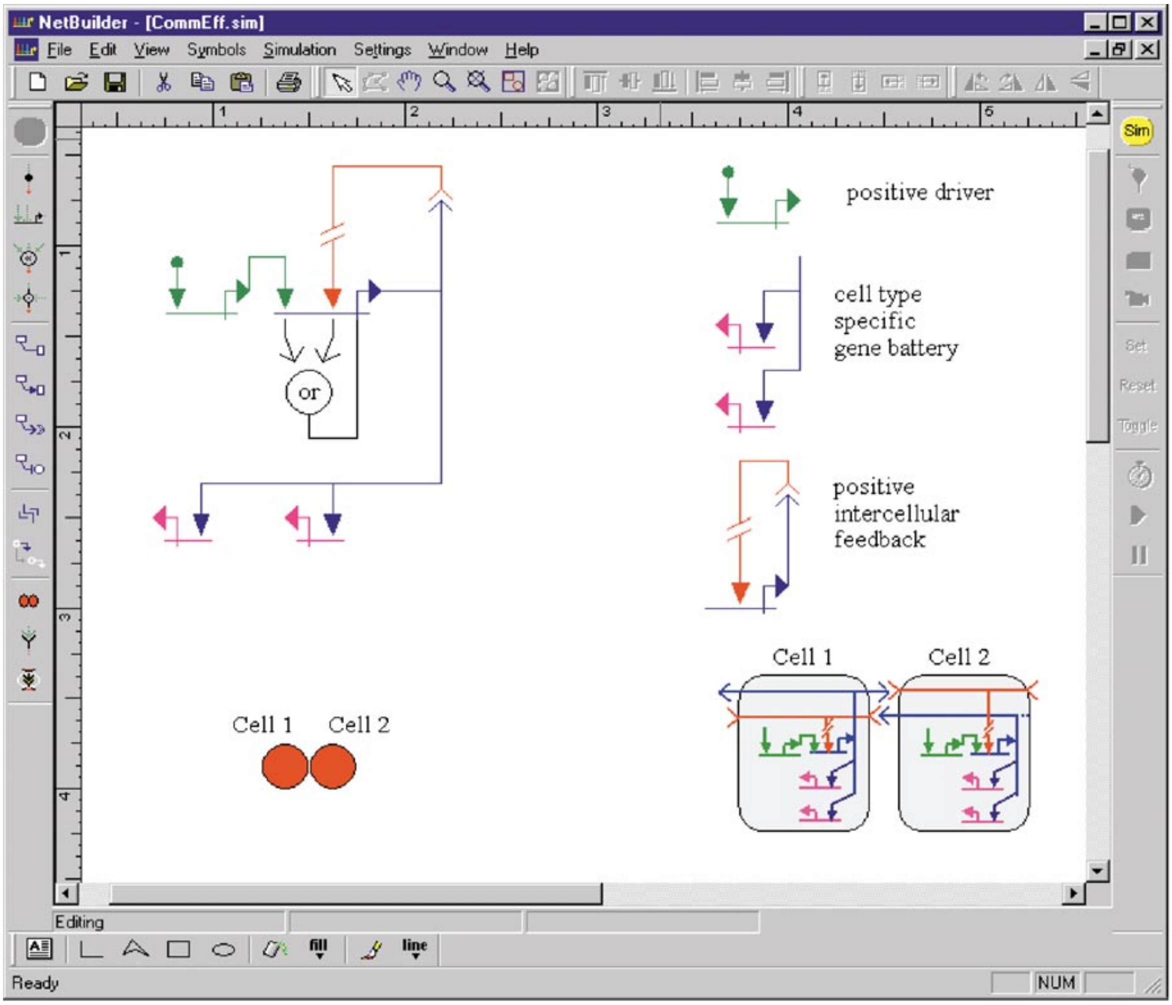
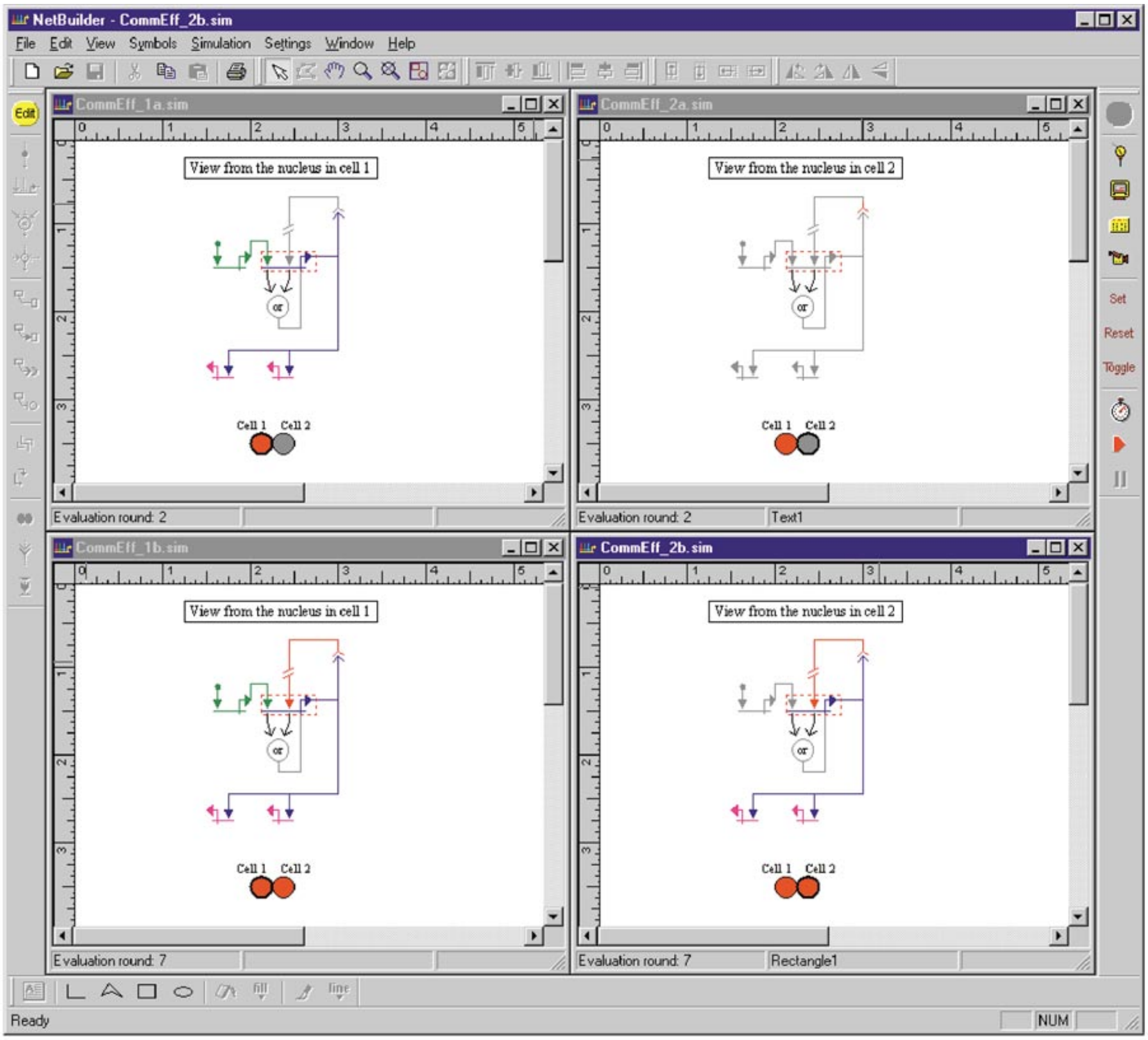

NetBuilder - [CommEff.sim]
File Edit View Symbols Simulation Settings Window Help
positive driver
cell type specific gene battery
positive intercellular feedback
or
Cell 1
Cell 2
Cell 1
Cell 2
Sim
Set
Reset
Toggle
Editing
fill
line
Ready
NUM

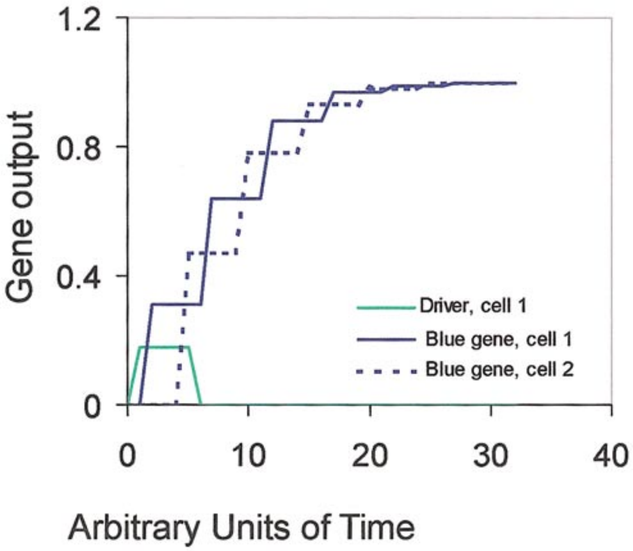
FIG. 12. The NetBuilder network editor. The figures on the left-hand side of the canvas represent the NetBuilder model. The two red circles below the network model represent the cells modeled. The modeled network element in each cell is shown at the top left of the canvas. The figures on the right-hand side of the canvas are annotations the user has added optionally to facilitate understanding, and to document the model. The example of the network displayed implements an intercellular positive feedback loop between two cells. The green gene is assumed to be activated by a maternal or other localized factor. The blue genes are engaged in self-reinforcing intercellular signaling and are modeled as driving downstream gene batteries (pink genes). The network has the characteristic that arbitrarily small levels of activity driving the green gene in either cell will cause the gene batteries in both cells to become activated and remain locked on. The two horizontal tool bars just above and below the editor canvas provide general purpose file management and figure annotation facilities. The two vertical tool bars to the left and right of the editor canvas provide NetBuilder editing and simulation facilities as described in the text.

FIG. 13. View from the nuclei of cells during simulation with NetBuilder. The state of each of the two cells (defined in Fig. 12) is shown at two points in time. Top left, cell 1 just after activation of the green gene. The blue gene has become active, but has not yet received a signal from cell 2. Top right, cell 2 at the same moment in time. The receptor on the cell has become active (signal from blue gene in cell 1), but nothing else is yet activated (gray lines indicate inactivity). Note the cartoon cells at the bottom of these images. A thick black border indicates which cell is being viewed. The color of the "cell" is red if the gene being monitored (in this case, the blue gene) is active, gray otherwise. The two bottom views show the same two cells some time later, when all components of the intercellular feedback network have become active in both cells. Note that in all these views, the simulation icons are colored (active) and the editor icons are gray (inactive). The opposite is true when NetBuilder is in edit mode (see Fig. 12).

FIG. 14. Example of a time-course simulation with NetBuilder. Time-course activities for the circuit of Figs. 12 and 13 are shown, for the green gene in cell 2, and the blue genes in both cells. Note the activity of the green gene is low and short lived, but is nonetheless sufficient to activate the blue genes in both cells to saturation (Y axis normalized to show maximum activity as 1) due to the positive intercellular feedback.


NetBuilder - CommEff_2b.sim
File Edit View Symbols Simulation Settings Window Help
Edit
Set
Reset
Toggle
CommEff_1a.sim
View from the nucleus in cell 1
or
Cell 1 Cell 2
Evaluation round: 2
CommEff_2a.sim
View from the nucleus in cell 2
or
Cell 1 Cell 2
Evaluation round: 2
Text1
CommEff_1b.sim
View from the nucleus in cell 1
or
Cell 1 Cell 2
Evaluation round: 7
CommEff_2b.sim
View from the nucleus in cell 2
or
Cell 1 Cell 2
Evaluation round: 7
Rectangle1
fill
line
Ready
NUM

13


Gene output
1.2
0.8
0.4
0
0
10
20
30
40
Arbitrary Units of Time
Driver, cell 1
Blue gene, cell 1
Blue gene, cell 2

14

for editing GRNs. When the top button is pressed, lines are automatically drawn with horizontal and vertical segments only. The button underneath this automatically updates the connector colors; i.e., when the button is pressed, the connectors assume the same color as the symbol whose output they transmit.

The fourth category of symbols ("Cells," "Receptors," and "Define contacts") relates to what is probably the most unique feature of NetBuilder. NetBuilder has been specially designed to allow the modeling of GRNs in multicellular embryos. Clicking in the Cell button (red filled circles) places a symbolic cell on the canvas (a red circle). The user can place a collection of such "cells" together to draw cartoons of a growing embryo at different stages of development (see, for example, the cartoon underneath the network drawing in Fig. 12). Different groups of cells (in time and space) can be grouped together and specified to have different initial/inherited conditions. The receptor library element is used as a generic intercellular signaling symbol. The user can edit the attributes of receptors (e.g., location, delay, degree of activation nonlinearity) using a menu that pops up when the "Define contacts" button is clicked.

As shown in the right-hand side of the canvas in Fig. 12, the user can place any number of additional comments and annotations on the canvas without interfering with the simulation model (the network and cells drawn on the left-hand side of the canvas). All of the NetBuilder symbols described can be enlarged, shrunk, or elongated by the user as necessary.

As described by Bolouri and Davidson (2002), we refer to the network diagram in which all interactions between genes and their products are illustrated as the View From the Genome (VFG). The circuit in Fig. 12 represents an example of a VFG. The user need only define such a "View from the Genome" and start the simulation. NetBuilder will automatically track the different states of the GRN in different cells. The simulation outcome can be viewed in two ways: (1) The "View from the Nucleus" for any particular cell type can be viewed by simply clicking on the cell of interest and then selecting the probe symbol (see top of simulation toolbar to the right of the canvas); the outputs of those genes which are active in the cells of interest are given user-defined colors, while inactive outputs are shown in gray. An example of the set of VFNs generated by the intercellular circuit of Fig. 12 is shown in Fig. 13. Presentations using those VFG and VFN conventions can be seen for the real regulatory model in the paper of Davidson *et al.* (2002). (2) A time-course plot of the level of expression of any gene in any cell. As an example, see Fig. 14, which shows a simulation outcome for the circuit illustrated in Figs. 12 and 13.

As with the network editor, the simulation facilities provided by NetBuilder use graphical icons to provide the user with a simple, intuitive means of controlling the simulation. The simulation icons are listed in a column to the right of the editor canvas. The simulation icons allow the user to set the start and end time of simulations, set or reset the state of any gene or switch in the network, plot the time course activity of any gene, and save the simulation result.

During the course of the *S. purpuratus* endomesoderm network project, we have modeled the complete network and individual subcircuits of interest in NetBuilder at various points. Some example files and the current version of NetBuilder (all in a continual state of flux) are available from our Web pages at: http://strc.herts.ac.uk/bio/Maria/NetBuilder.

Conclusion

We have outlined a set of software tools to aid the process of understanding genetic regulatory networks. Although the description is necessarily linear, in reality, the process is both nonlinear and iterative. These software tools aid in the process of building and testing experimentally falsifiable hypotheses. Of course they do not obviate the need for human intervention and clear thinking. Nor are they a comprehensive tool set, but are intended to be used in conjunction with the many other useful tools developed elsewhere. Moreover, the popular saying "If it works, it is out of date" is particularly apt here. All software tools described in this paper are in a continual state of flux and development. Nonetheless, where possible, all current versions of our algorithms are available freely via the Internet (see individual listings above). As with calculators, when tools mature, their use becomes transparent. It is our hope that with use, the tools we have presented will mature to the point that their use will be transparent and taken for granted. However, even in their present state, it can be said that the packages described have proved their mettle: the sea urchin endomesoderm network analysis described by Davidson *et al.* (2002) could never have been assembled without them.

Availability of Tools

The executable binary for the version of BioArray described in this paper is available to academic users upon request; contact titus@caltech.edu to obtain a copy. BioArray has been licensed to Genetix for further development; please contact Genetix (http://www.genetix.co.uk/.) for more information.

The SUGAR source code and analyses are available at http://sea-urchin.caltech.edu/software/SUGAR/. We cannot redistribute the programs used to generate the analyses, but they are available at the URLs given in the text. SeqComp and FamilyRelations are available, along with complete source code and a tutorial, at http://family.caltech.edu/. NetBuilder is available at http://strc.herts.ac.uk/bio/Maria/NetBuilder/. These links can also be found at http://sea-urchin.caltech.edu/software/ (please be aware that these Web addresses are case sensitive!).

ACKNOWLEDGMENTS

We thank Drs. Rod Adams, Henry Brzeski, and Jonathan Rast for numerous discussions and helpful feedback. Data for Fig. 3 were generated by Toni Snow and Yinjian Xiong. We also thank the robotics macroarraying facility, supported by NIH Grant RR15044 (to E.H.D.) and by the Caltech Beckman Institute. The software development work reported in this paper was supported in part by UK BBSRC Grant 310/BI012024 to HB, and US NIH Grant GM61005 (to E.H.D. and H.B.). C.T.B. is a participant in the Initiative in Computational Molecular Biology, which is funded by an award from the Burroughs Wellcome Fund Interfaces program.

REFERENCES

Bolouri, H., and Davidson, E. H. (2002). Modeling DNA sequence-based *cis*-regulatory gene networks. *Dev. Biol.* **246,** 2–13.

Cameron, R. A., Mahairas, G., Rast, J. P., Martinez, P., Biondi, T. R., Swartzell, S., Wallace, J. C., Poustka, A. J., Livingston, B. T., Wray, G. A., Ettensohn, C. A., Lehrach, H., Britten, R. J., Davidson, E. H., and Hood, L. (2000). A sea urchin genome project: Sequence scan, virtual map, and additional resources. *Proc. Natl. Acad. Sci. USA* **97,** 9514–9518.

Davidson, E. H. (2001). "Genomic Regulatory Systems: Development and Evolution." Academic Press, San Diego, CA.

Davidson, E. H., Rast, J. P., Oliveri, P., Ransick A., Calestani, C., Yuh, C.-H., Minokawa, T., Amore, G., Hinman, V., Arenas-Mena, C., Otim, O., Brown, C. T., Livi, C. B., Lee, P. Y., Revilla, R., Schilstra, M. J., Clarkes, P. J. C., Rust, A. G., Pan, Z., Arnone, M. I., Rowen, L., Cameron, R. A., McClay, D. R., Hood, L., and Bolouri, H. (2002). A provisional regulatory gene network for specification of endomesoderm in the sea urchin embryo. *Dev. Biol.* **246,** 162–190.

Gonzalez, P., and Lessios, H. A. (1999). Evolution of sea urchin retroviral-like (SURL) elements: Evidence from 40 echinoid species. *Mol. Biol. Evol.* **16,** 938–952.

Harris, N. L. (1997). Genotator: A workbench for sequence annotation. *Genome Res.* **7,** 754–762.

Maier, E., Meier-Ewert, S., Ahmadi, A. R., Curtis, J. and Lehrach, H. (1994). Application of robotic technology to automated sequence fingerprint analysis by oligonucleotide hybridisation. *J. Biotechnol.* **35,** 191–203.

Mayor, C., Brudno, M., Schwartz, J. R., Poliakov, A., Rubin, E. M., Frazer, K. A., Pachter, L. S., and Dubchak, I. (2000). VISTA: Visualizing global DNA sequence alignments of arbitrary length. *Bioinformatics* **16,** 1046.

Ransick, A., Rast, J. P., Minokawa, T., Calestani, C., and Davidson, E. H. (2002). New early zygotic regulators of endomesoderm specification in sea urchin embryos discovered by differential array hybridization. *Dev. Biol.* **246,** 132–147.

Rast, J. P., Amore, G., Calestani, C., Livi, C. B., Ransick, A., and Davidson, E. H. (2000). Recovery of developmentally defined gene sets from high-density cDNA macroarrays. *Dev. Biol.* **228**, 270–286.

Schwartz, S., Zhang, Z., Frazer, K. A., Smit, A., Riemer, C., Bouck, J., Gibbs, R., Hardison, R., and Miller, W. (2000). PipMaker: A web server for aligning two genomic DNA sequences. *Genome Res.* **10,** 577–586.

Sonnhammer, E. L. L., and Durbin, R. (1995). A dot-matrix program with dynamic threshold control suited for genomic DNA and protein sequence analysis. *Gene* **176,** GC1–GC10.

Wasserman, W. W., Paulumbo, M., Thompson, W., Fickett, J. W., and Lawrence, C. E. (2000). Human-mouse genome comparisons to locate regulatory sites. *Nat. Genet.* **26,** 225–228.

Yuh, C.-H., Brown, C. T., Livi, C. B., Rowen, L., Clarke, P. J. C., and Davidson, E. H. (2002). Patchy interspecific sequence similarities efficiently identify positive *cis*-regulatory elements in the sea urchin. *Dev. Biol.* **246,** 148–161.

Received for publication December 21, 2001
Revised February 7, 2002
Accepted February 7, 2002