7908–7915 *Nucleic Acids Research, 2010, Vol. 38, No. 22* *Published online 6 August 2010*
doi:10.1093/nar/gkq661

SEWAL: an open-source platform for next-generation sequence analysis and visualization

Jason N. Pitt[1,2,*], Indika Rajapakse[2,3] and Adrian R. Ferré-D'Amaré[1,2]

[1]Howard Hughes Medical Institute, [2]Division of Basic Sciences, Fred Hutchinson Cancer Research Center, 1100 Fairview Avenue North, Seattle WA 98109-1024 and [3]Biostatistics and Biomathematics, Public Health Sciences, Fred Hutchinson Cancer Research Center, 1100 Fairview Avenue North, Seattle WA 98109-1024, USA

Received June 18, 2010; Revised July 12, 2010; Accepted July 13, 2010

ABSTRACT

Next-generation DNA sequencing platforms provide exciting new possibilities for *in vitro* genetic analysis of functional nucleic acids. However, the size of the resulting data sets presents computational and analytical challenges. We present an open-source software package that employs a locality-sensitive hashing algorithm to enumerate all unique sequences in an entire Illumina sequencing run ($\sim10^8$ sequences). The algorithm results in quasilinear time processing of entire Illumina lanes ($\sim10^7$ sequences) on a desktop computer in minutes. To facilitate visual analysis of sequencing data, the software produces three-dimensional scatter plots similar in concept to Sewall Wright and John Maynard Smith's adaptive or fitness landscape. The software also contains functions that are particularly useful for doped selections such as mutation frequency analysis, information content calculation, multivariate statistical functions (including principal component analysis), sequence distance metrics, sequence searches and sequence comparisons across multiple Illumina data sets. Source code, executable files and links to sample data sets are available at http://www.sourceforge.net/projects/sewal.

INTRODUCTION

Recent advances in DNA sequencing technology afford new opportunities and new challenges in nucleic acid sequence analysis (1). The ability routinely to generate gigabases of data in laboratories outside of sequencing centers has necessitated the advent of computationally and monetarily inexpensive software. We present here an open-source software package entitled SEWAL (**S**equence **E**volution **W**ith **A**daptive **L**andscapes) designed to analyze Illumina deep-sequencing data and display it in the form of interactive three-dimensional (3D) scatter plots. To generate these plots, SEWAL also features a variety of sequence manipulation and analysis options designed to run with high speed on desktop computers. SEWAL is currently compiled for the 64-bit Apple Macintosh operating system; however the code is open-source (http://www.sourceforge.net/projects/sewal), written in gnu C++, and should be easily compiled on any 64-bit operating system that supports OpenGL.

SEWAL (Figure 1) analyzes deep-sequencing data sets generated from *in vitro* selection pools of functional nucleic acids and performs nearest-neighbor analysis of all sequences in a sequencing run using a locality sensitive hashing (LSH) algorithm (2,3). SEWAL then tabulates the frequency of every observed sequence and displays the data from the entire sequencing run in the form of a 3D scatter plot hereafter referred to as mutant spectrum (4). The goal in this type of analysis is to leverage massive sequencing as a tool to perform functional genetics of nucleic acids *in vitro*. This idea is based on the assumption that the frequency of a sequence in an *in vitro* selection experiment is proportional to the fitness of that sequence. Using SEWAL and high-throughput biochemical analysis, we have recently shown this assumption to be true for one class of catalytic RNA molecules (J. N. Pitt and A. R. Ferré-D'Amaré, submitted for publication). Therefore, data from SEWAL can be used to construct empirical fitness landscapes. A fitness landscape as originally conceived by Wright is analogous to a topographic map, depicting differing allele combinations in the horizontal axes with the height of any given allele combination representing Darwinian fitness of that allele combination in the vertical axis (5). The concept was extended to

*To whom correspondence should be addressed. Tel: +1 206 667 3603; Fax: +1 206 667 3331; Email: jpitt@fhcrc.org

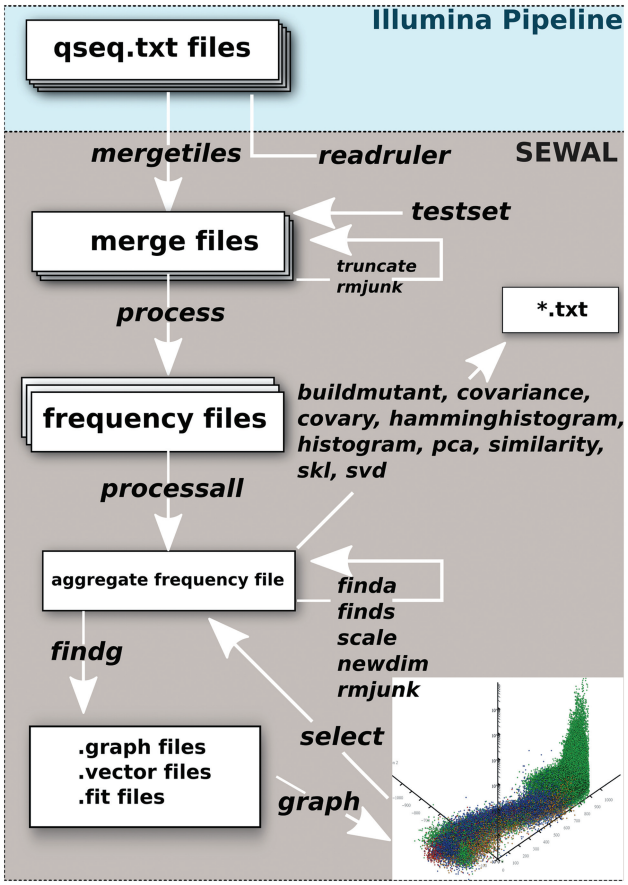

Figure 1. SEWAL pipeline for sequence analysis. SEWAL currently reads qseq.txt files generated by the Illumina pipeline. SEWAL commands are shown in italics. File types are shown by white boxes. All files are tab-delimited text files with defined formatting. Inset graphic is a screenshot of the SEWAL interactive graphics window.

macromolecules by Maynard Smith (6). In this case, a given macromolecule sequence (allele), is characterized by specific combination of nucleotide or amino acid substitutions (i.e. a haplotype) relative to a reference sequence. These fitness landscapes have attracted considerable theoretical interest, but constructing them experimentally involves phenotypic analysis of all of alleles in the mutant spectrum, which can be technically challenging (7). In molecular evolution, the fitness landscape is a hyper-dimensional object with its dimensionality being a function of the length of the polymer in question (8). If one were able to generate a fitness landscapes for a sequence of interest under a variety of environmental conditions, these could be used to infer accessible evolutionary paths and even predict the course of evolution for that sequence. To this end, SEWAL can compare the frequency of all unique sequences across multiple next-generation sequencing runs. While our application (J. N. Pitt and A. R. Ferré-D'Amaré, submitted for publication) has been limited to analyzing *in vitro* selection pools of a catalytic RNA (9), SEWAL should prove useful for any assay where the copy number of unique nucleic acids in a complex data set may be informative, for instance, viral quasispecies evolution in response to immune selection or drug therapy, or differential mRNA expression (10).

MATERIALS AND METHODS

The primary computational task with our analysis is the *de novo* comparison of every sequence in a data set to every other sequence in the data set in order to determine a frequency for each observed sequence. To determine the copy number of individual sequences in the data set, sequences first pass quality filtering and are then compared to each other to determine the frequency of each unique sequence. A brute-force comparison of all members in the set would result in a worst-case complexity of $O(\nu\, n^2)$, where ν is the length of the sequence and n is the total number of sequences in the data set. In practice, these searches become time limiting on single-processor systems for data sets exceeding 2×10^5 sequences ($\nu = 58$). LSH is a technique that uses LSH functions to preprocess the set using k number of hash functions (h_i) and hash tables (3). (A hash function is a deterministic function that reduces a high dimensional data set, such as a long polynucleotide sequence, to a single data point, such as an integer.) The hashing step can be performed with random strings or with a search string of interest (q), with the latter hash value being a distance metric to q (defined in the sewal.prefs file or subsequently using the *newdim* command). Hashing is performed ($k = 4$) and the set sequentially sorted using a stable sort for all stored hash values $O[k\nu n + kn\ log(kn)]$ (11). The probability of collision of non-equal sequences during hashing is a function of ν and k (Figure 2). Following hashing, the set is then traversed and copy number determined using pairwise comparison [limited to $h_i\,(q) = h_i\,(n)$] to prevent equating of non-equal sequences that collided during hashing. We determined empirically that for our typical Illumina data sets with ($n \sim 10^7$, $\nu = 58$) a pairwise comparison after sorting with $k = 4$ was faster than allowing the LSH to converge without pairwise comparison ($k \sim 20$). The Q_{phred} scores (12) of identical sequences are summed and stored. After the list is traversed it is pruned, and mean Q_{phred} scores for each sequence are calculated as: $[\Sigma_{i=1}^{\text{frequency}}(Q_{\text{phred}})_i]$/frequency.

RESULTS AND DISCUSSION

We chose to focus on the Illumina platform because its short read length and higher read density were advantageous in analyzing populations of small RNAs (13). The current release of SEWAL is designed to accept qseq.txt files from the Illumina Genome Analyzer v1.4 (Illumina, San Diego, CA, USA). These files are merged, filtered for quality, hashed, sorted, tabulated, merged with similar data sets generated under differing conditions and then formatted as a SEWAL aggregate frequency file, which can be analyzed using several sequence and graphic analysis features built into SEWAL. Experimentally generated test data sets are available for download at

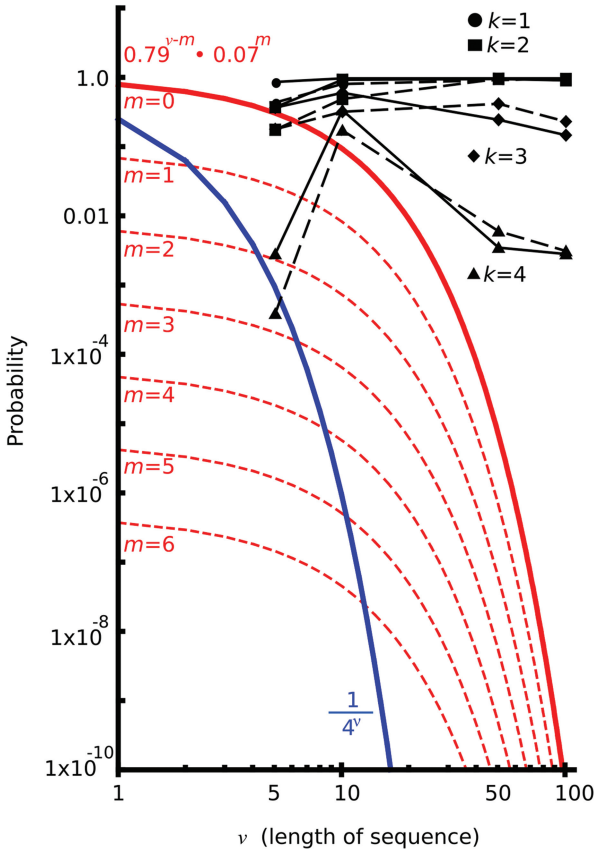

Figure 2. Expected mutant frequencies and empirically determined algorithm performance. SEWAL is designed to analyze pools of nucleic acids to determine the frequency of specific mutant sequences in the pool. The worst-case comparison of all observed mutants to all other observed mutants would have a complexity of $O(\nu\ n^2)$ where O is the upper bound of the growth of the function, n is the total number of sequences, and ν is the length of each sequence. The blue line is the probability of encountering any mutant in a pool composed of random RNAs as a function of ν. The red line is the probability of encountering a reference sequence with a pool mutagenized at a rate of 21% (7% each possible point mutation). The dashed red lines correspond to the probability of encountering a reference sequence with m number of specific point mutations. Black lines show the performance of the SEWAL sorting algorithm on test data sets ($n = 10^7$) using k number of LSH functions expressed as the probability of a collision between any two nonidentical sequences with equal hash values. Solid black lines represent random test data sets; dashed lines, *in silico* generated data sets mutagenized at 21% per position.

the NCBI Trace Archives (accession number SRA020870.1). An overview of the SEWAL computational pipeline is shown in Figure 1 and a detailed explanation is presented below. All SEWAL text commands are hereafter presented in italics.

Process

Illumina pipeline qseq files are first merged into a single data set using the SEWAL commands *mergetiles* and *truncate* to generate a fixed number of clusters that pass quality filtering for cross-sample or cross-lane comparison. This step is critical when inferring changes in the frequency of a particular sequence in multiple sequencing runs or flow-cell lanes. Similar comparisons can be achieved using the *scale* command to scale two data sets relative to each other based on the sequencing of a loading control sequence following construction of a SEWAL frequency file (*readruler* command). Quality filtering is user-controllable using a configuration file (sewal.prefs; see file header for details). Default quality-control settings ignore Illumina quality flags, but remove clusters containing one or more of the following errors: base calls flagged as gaps, clusters with homo-polymer runs longer than six residues, and more than six positions with Q_{phred} scores of 3 or higher. A Q_{phred} score (12) is the probability that a base call is incorrect. The Illumina pipeline encodes Q_{phred} scores as an ASCII string (ASCII value $-64 = Q_{\text{phred}}$; one char per base) as: $Q_{\text{phred}} = -10 \log (ep)$, where (ep) is the error probability for that position. A Q_{phred} score equal to 3 is a 50% chance that the base is incorrectly assigned.

Locality sensitive hashing

We have solved the problem of comparison of every sequence in the data set to every other sequence using the approach of LSH to reduce the dimensionality of the data from ν sequence positions to 4 LSH values. Using four successive table sorts of the LSH values, the frequency of each sequence is then determined in quasilinear time (Table 1). This approach has the added advantage that the data set has been preprocessed, accelerating subsequent nearest-neighbor searches (*finds*). We set the number LSH functions (k) to 4 because of the nature of our data; namely, mutant pools with high similarity ($n \sim 10^7$, $\nu = 58$). The primary trade-off between small and large values of k is between preprocessing/sorting time and pair-wise comparison time. In practice, all sorts with $k > 1$ resulted in efficient tabulation of data of the scale currently generated by the Illumina platform (Table 1). A k of 4 translated into brute-force searches for ~0.001% of our data set or 10 000 sequences (Figure 2). For larger data sets ($>>n$) a greater number of LSH values could be computed, increasing the preprocessing time and space linearly, and incorporating additional sorts, but decreasing the collision probability and requisite pair-wise comparisons. A useful feature of the LSH algorithm used in this context is that as the length of a random polymer (ν) increases, the performance of the algorithm also increases (Table 1 and refs 2 and 3). This behavior results from the LSH function and is dependent on both k and ν. Therefore, as Illumina read lengths improve, this sorting procedure will remain highly efficient.

Processall

Following LSH, SEWAL generates a frequency file, sorted by copy number, which is equivalent to the tab-delimited qseq file with the Q_{phred} string replaced by the mean Q_{phred} string, and a unique ID tag, sequence copy number and hash values appended. The *processall* command compares the frequency of all sequences between two or more frequency files (maximum number of experiments/Illumina lanes is currently 7). SEWAL generates an aggregate frequency file by concatenating the individual frequency files and then sorting the sequences based on the

Table 1. Performance of the SEWAL LSH sorting algorithm as a function of k (number of hash functions) and ν (length of each sequence) for *in silico* generated populations ($n = 10^7$)

k	ν	Processing time (s)	Library type	Collision probability	Total collisions	Total comparisons
1	5	25	Random	0.8837	8 838 480	10 001 677
2	5	55	Random	0.3819	3 819 630	10 000 378
3	5	75	Random	0.3768	3 779 490	10 029 480
4	5	106	Random	0.0029	29 165	10 029 165
1	5	27	Doped	0.446	4 460 760	10 001 360
2	5	48	Doped	0.1805	1 805 160	10 000 409
3	5	63	Doped	0.1845	1 853 660	10 048 838
4	5	83	Doped	0.0004	4474	10 004 474
1	10	21	Random	0.9986	15 475 100	15 496 479
2	10	46	Random	0.9512	11 955 500	12 568 309
3	10	75	Random	0.6172	6 715 260	1 088 0872
4	10	103	Random	0.3455	3 600 590	10 422 430
1	10	25	Doped	0.8302	8 892 800	10 711 457
2	10	53	Doped	0.5035	5 183 970	10 295 630
3	10	76	Doped	0.3278	3 304 510	10 082 137
4	10	103	Doped	0.1769	1 775 630	10 040 221
1	50	78523	Random	0.9999	$1.61\,607 \times 10^{11}$	$1.61\,617 \times 10^{11}$
2	50	93	Random	0.9813	525 142 000	535 141 971
3	50	104	Random	0.2522	3 373 350	13 373 353
4	50	140	Random	0.0036	35 779	10 035 779
1	50	72806	Doped	0.9999	$1.77\,183 \times 10^{11}$	$177\,193 \times 10^{11}$
2	50	118	Doped	0.9878	811 783 000	821 773 604
3	50	106	Doped	0.4301	7 545 680	17 544 589
4	50	140	Doped	0.0062	62 612	10 062 405
1	100	58	Random	0.9987	78 376 300	78 481 204
2	100	63	Random	0.9359	33 599 300	35 900 313
3	100	108	Random	0.1513	1 690 980	11 174 301
4	100	129	Random	0.0029	29 433	10 027 028
1	100	91994	Doped	0.9999	$1.44\,037 \times 10^{11}$	$1.44\,047 \times 10^{11}$
2	100	81	Doped	0.9793	473 721 000	483 721 331
3	100	88	Doped	0.2376	3 116 040	13 116 042
4	100	122	Doped	0.0032	32 331	10 032 331

Library type is random (a 25% probability of each nucleotide at each position) or doped (a 79% probability of a wild-type residue at each position and a 7% probability of each possible mutant residue.

stored LSH values. This sorting of data sets from multiple Illumina lanes is memory-intensive, and in the current implementation requires a 64-bit OS with >20 GB physical memory. After sorting, the duplicates in the list are collapsed as above, and a SEWAL aggregate frequency file created. This file contains each sequence with a unique ID number, a mean Q_{phred} score string, the copy number of the observed sequence in each lane, the four LSH scores, the observed copy numbers from each lane, and the lane number containing the maximal copy number.

Data analysis

Histogram, hamminghistogram. These commands generate histograms of a user-defined bin size and bin number. The output is a tab-delimited text file containing the limits for each bin and the number of sequences in each bin. Both the number of unique sequences and the total number of observations of all members in the bin are given in the table. The difference between the *hamminghistogram* and *histogram* commands is the distance metric used to determine the bin placement for each sequence. Bins for the *histogram* command are measured using one of the four LSH values (user-defined), while *hamminghistogram* takes an input string for distance comparison and the outputted histogram bins are based on the Hamming distance from the query string (14).

Finda, findg, finds. These three search commands, *finda, findg* and *finds,* find sequences in the aggregate frequency file based on specific search queries. The *finda* (find all) and *finds* (find sequence) commands return a new aggregate frequency file containing sequences that existed in a specified region of the mutant spectrum or are similar to a specified search sequence, respectively. The *findg* (find and graph) command is similar to the *finda* command but returns either a .graph, .vector or .fit file. The .graph files are used to generate 3D scatter plots using the *graph* command. In these plots, each sequence is represented by a single dot with the axes being the copy number of the sequence and the other two axes being user-selected LSH values (Figure 3). The .vect files have these same axes but contain vectors showing the changes in frequency of each sequence in each analyzed lane (Figure 4b), while the .fit files depict the magnitude of the vector between any two Illumina lanes as a scatter plot.

Covary, buildmutant. One of the main uses of SEWAL is to understand the mapping between the sequence of a nucleic acid and its fitness. To display this mapping, SEWAL can generate mutation frequency tables based on defined regions of sequence space using the *buildmutant* command. The *buildmutant* command outputs a tab-delimited text file that contains a table of observed point mutant frequencies from a user provided reference sequence. The table also contains positional information content, in bits (15). This information content calculation can be adjusted to account for sequence bias in the starting population (16). In a structured nucleic acid, many positions are not conserved at the primary sequence level but are required to maintain Watson–Crick base pairing (17). At these positions, a mutation can be said to 'covary' with a mutation at another position in the RNA to maintain base pairing. These data can be used to infer the secondary structure of the nucleic acid. SEWAL can analyze the data set to look for instances of covariation using the *covary* command. The *covary* command takes a reference sequence and a defined region of the mutant spectrum and uses the sequence data from the deep sequencing run to output a covariation matrix for all of the sequenced positions that are separated by a minimum of three residues.

Singular value decomposition and principal component analysis

A specified region of the genotype frequency plot may be expressed in terms of a matrix A, consisting of m rows of

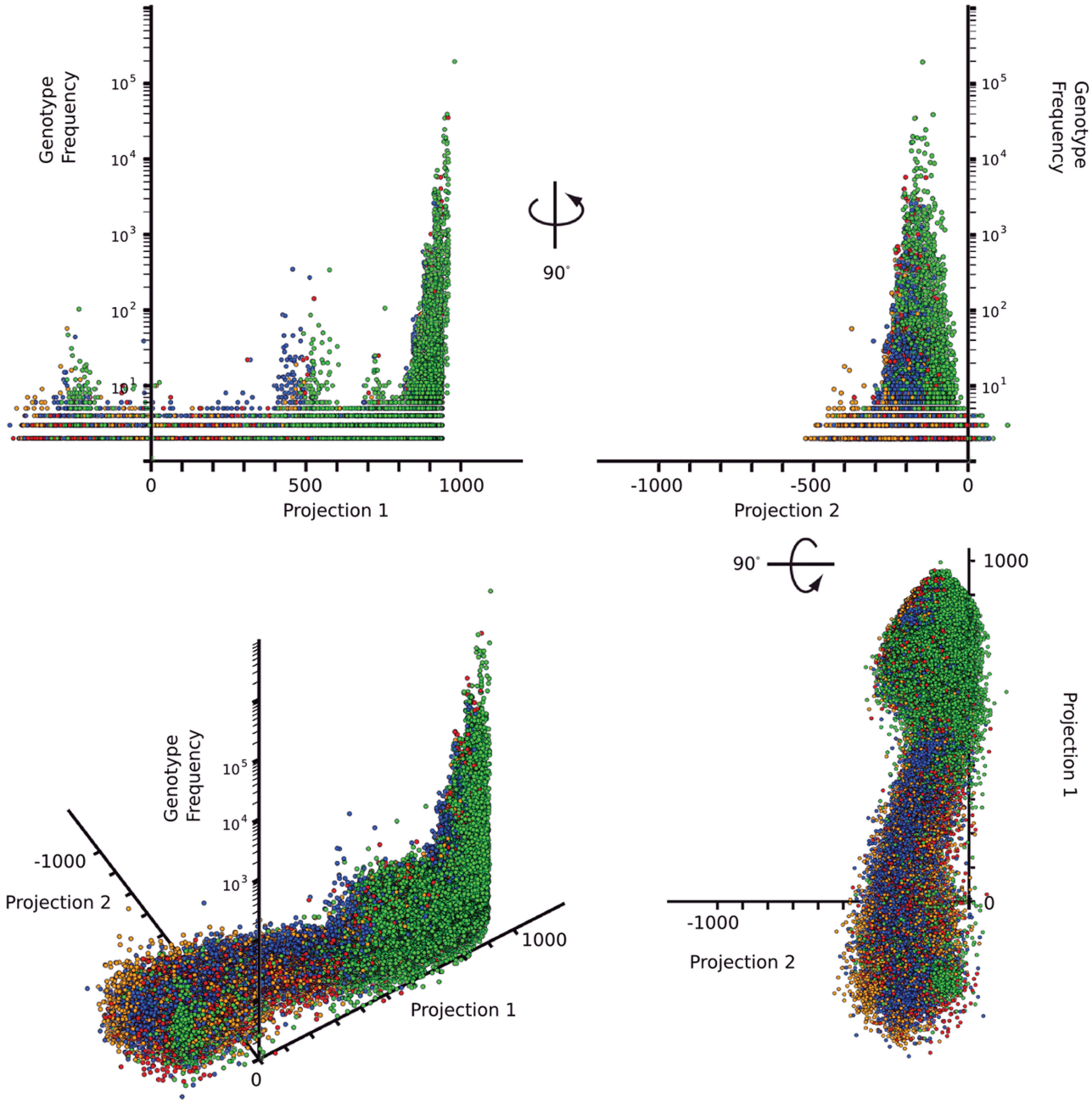

Figure 3. Orthogonal views of 3D mutant spectra generated by SEWAL. Each colored ball represents a unique sequence in the pool. Vertical axis is the $\log_{10}$ of the observed frequency of the sequence. Horizontal axes are the SEWAL LSH values using a wild-type sequence (Projection 1) or a sequence with no similarity to the data set (a segment of the *Ornithorhynchus anatinus* lysozyme gene; Projection 2). Sequences have been colored using the *colorbase* command based on the identity of the base at position 21 in the sequence (G, green, A, red, T,U, blue, C, orange). Curved arrow depicts the relative orientation of the views.

positional nucleotide frequencies or information content, over z columns of differing sequencing lanes (e.g. z differing selective pressures). Any $m \times z$ matrix A can be factored into $A = UDV^T$ (18). The columns of $U(m \times m)$ are eigenvectors of AA^T, and the columns of V $(z \times z)$ are eigenvectors of A^TA. The r singular values on the diagonal of $D(m \times z)$ are the square roots of the nonzero eigenvalues of both AA^T and A^TA. The matrix A^TA is known as the covariance matrix of A, and it has familiar interpretations in statistics and other fields (18). When a data matrix has been *centered*, by subtracting the mean of each column from the entire column, this process is known as 'principal component analysis' (PCA). The right singular vectors are the components, and the scaled left singular vectors are the *scores*. PCAs are described in terms of the eigenvalues and eigenvectors of the covariance matrix, A^TA (19). The first singular vectors or principal components, when applied to positional nucleotide frequencies for a single defined peak in the mutant spectrum, provides eigenvectors corresponding to the reference sequence of

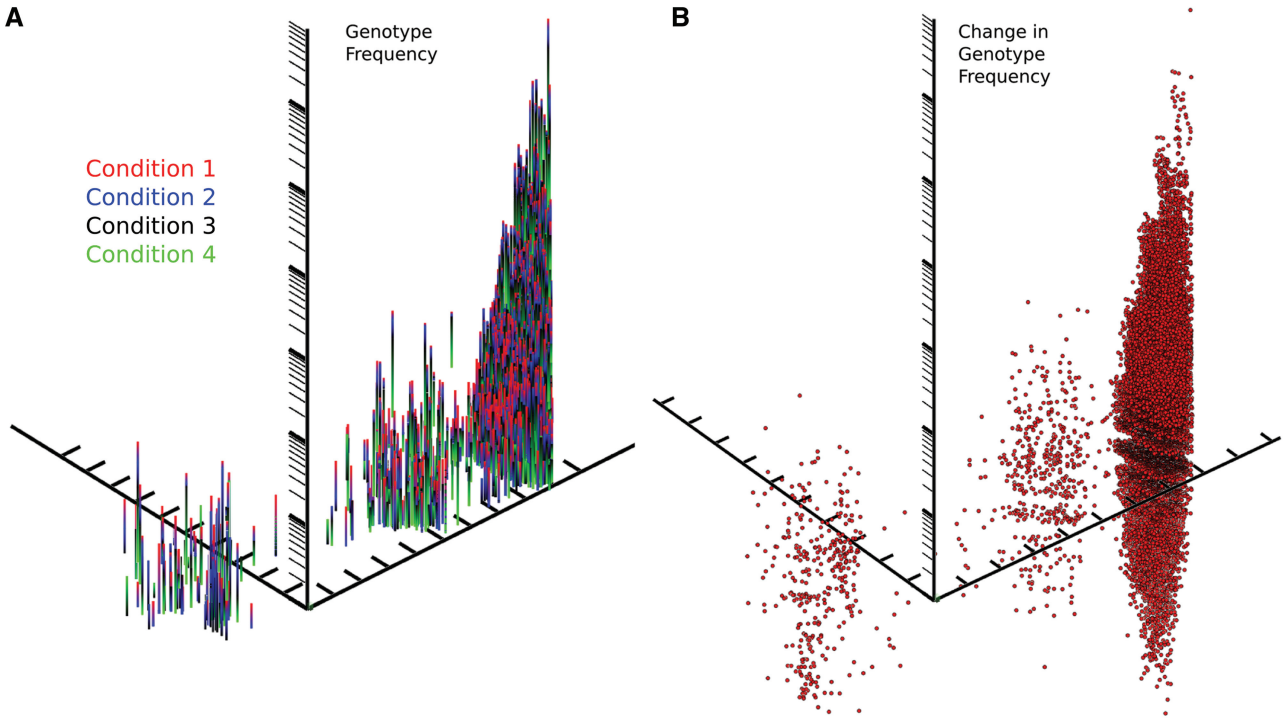

Figure 4. 3D Vector and fitness plots of genotype distributions in four separate sequencing experiments generated by SEWAL. (**A**) Each vertical line represents a unique sequence present in four separate sequencing experiments. Vectors are color coded to show the frequency of each sequence in each experimental condition. Sequences sorted using the SEWAL *findg* command to display only sequences that have a maximal distribution in condition 1 (red). (**B**) Sequences from the same data set plotted to display the magnitude of the difference between condition 1 (high stringency selection) and condition 4 (low stringency selection).

the spectrum (J. N. Pitt and A. R. Ferré-D'Amaré, submitted for publication, 20). The *svd* and *pca* commands return matrixes UDV^T for $A(m,z)$ where m is specified by the user as either the information content of a position or the library bias adjusted base frequency at each position over z sequencing lanes. The commands *svddmax* or *pcadmax* perform identical calculations; however, sequences are presorted depending on their maximum observed distribution across lanes, and are binned in terms of their maximal distribution ($z(i)$ = sequences with a maximal distribution in lane $d_{\max}(z)$:$1,\ldots,z$).

Covariance. The *covariance* command can be used to generate AA^T or A^TA for a specified region of the mutant spectrum. AA^T provides the correlation of each possible mutant residue in the sequence with every other possible mutant residue. For mutant spectra containing RNA secondary structures, this could be interpreted as Watson–Crick base-pairing covariance or any other structural feature dependent on more than one position. The magnitude of the covariance indicates critical (i.e. invariant) interactions, in that they are independent of the selective condition, or in terms of the matrix, highly correlated. Conversely, A^TA provides the dependence or independence of a region of the mutant spectrum across sequencing lanes or selective conditions, indicating the correlation between selective regimes as indicated by the mutant spectra.

Symmetrized Kullback–Leibler divergence

The relative entropy or Kullback–Leibler (KL) divergence is a measure in statistics (21,22) that quantifies how close a probability distribution $x(i)$ is to a model (or candidate) distribution $y(i)$:

$$\mathrm{KL}(x,y) = \sum_i x(i)\log_2\frac{x(i)}{y(i)}.$$

KL divergence is often used as a measure of the difference between two distributions (21,22). The KL divergence is not symmetric, thus we define the symmetrized Kullback–Leibler divergence (SKL) (23) as:

$$\mathrm{SKL}(x,y) = \tfrac{1}{2}\left(\sum_i x(i)\log_2\frac{x(i)}{y(i)}\right) + \tfrac{1}{2}\left(\sum_i y(i)\log_2\frac{y(i)}{x(i)}\right).$$

SKL is a useful metric for quantifying the difference between mutant spectra sequenced under differing conditions. When the selective pressures are equal, SKL should approach 0 and under differing selective pressures SKL is much greater than 0. SKL can be performed on specified regions of the mutant spectrum to determine regions of maximal or minimal divergence. The SKL command can also bin the sequences depending on their $d_{\max}$ across multiple sequencing lanes providing a measure of the relative entropy in bits between these populations:

$$A = \begin{pmatrix} 0 & . & . & . \\ 1.56 & 0 & . & . \\ 4.04 & 0.7 & 0 & . \\ 8.63 & 3.47 & 1.42 & 0 \end{pmatrix}$$

Where A is the symmetric matrix produced by SKL between the d_{max} binned populations from four example *in vitro* selection experiments with decreasing selective stringency.

Similarity. The similarity command returns several traditional population-based similarity indexes (Jaccard, Søresen and Mountford) frequently used in ecology to quantify the total relationship between two different populations (in this case, Illumina lanes) (24).

Graph, fit, select, colorbase. SEWAL consists of two windows, a text-based interface launched from the terminal/command line and an interactive 3D graphical interface that is activated by running the *graph* command from the SEWAL command line. Sequences are displayed as a mutant spectrum (Figure 3) using the *open* command to load a .graph (Figure 3), .vector (Figure 4A) or .fit file (Figure 4B) generated using *findg* (above). Multiple files can be loaded simultaneously, and the software has successfully produced graphs of as many as 7×10^6 sequences. Alternatively, using the .vector file format, the frequency of each sequence in different sequencing experiments is displayed visually as a color-coded vector (Figure 4). The .fit files are particularly useful when the difference in the frequency of a genotype across lanes is known (or hypothesized) to be correlated with a difference in that genotype's phenotypic fitness, because this plot then represents an empirical fitness landscape (J. N. Pitt and A. R. Ferré-D'Amaré, submitted for publication).

From the graphical window, sequences can be selected using the mouse and exported using the *select* command to generate a new aggregate frequency file. Sequences can be colored using specified sequence parameters using the *colorbase* command (Figure 3). The graphical window contains a menu system to adjust how elements are displayed, and the 3D plot can be rotated and zoomed using the mouse and captured and exported as a .tif file. The graphical window also generates a histogram for each of the three axes that can be displayed.

CONCLUSIONS

Visualization of hyperdimensional objects is not trivial; however, several software applications have been written to visualize multivariate data (25,26). We chose to build a new interactive 3D interface into SEWAL for several reasons. First, the standard software graphing packages available in most laboratories simply cannot graph more than $10^5 - 10^6$ points. Illumina data sets routinely generate $10^7 - 10^8$ sequences in a single run, and therefore all of these data cannot be visualized simultaneously. SEWAL has been used to graph (albeit slowly) data sets approaching 10^7 points, and there is no practical limit to the number it could graph at reduced frame rates. Second, using the 3D frequency plots to identify patterns or regions of interest in the data and exporting those DNA sequences is a useful feature. SEWAL supports this point-and-click methodology. Third, graphical representation of DNA sequence information in multiple colors and dimensions reveals interesting patterns in the data not provided by other applications.

SEWAL provides a new platform for DNA sequence analysis and visualization of large data sets generated by massive sequencing to perform functional *in vitro* genetics. The visualization incorporates the underlying LSH information in a visually intuitive manner to generate mutant spectra. When the differences in these plots resulting from altered selective pressures are visualized, they can represent projections of the hyperdimensional fitness landscape, if the frequency of the sequences has been demonstrated to correlate with phenotypic fitness (8).

ACKNOWLEDGEMENTS

SEWAL contains data processing code from the ALGLIB library, written by Sergey Bochkanov (http://www.alglib.net). We thank R. Basom, A. Marty, and the staff of the Fred Hutchison Cancer Research Center Genomics Shared Resource for performing the Illumina sequencing required for testing the software, and D. Bartel, B. Fritz, R. Knight, H. Malik, W.S. Noble, H. Robins, A. Roll-Mecak, and M. Tewari, for discussions. A.R.F. is an Investigator of the HHMI. We have released the SEWAL as an open source project on SourceForge (http://www.sourceforge.net/projects/sewal). This should allow the software to be adapted to other sequencing platforms and other applications not foreseen at the time of writing.

FUNDING

The W.M. Keck Foundation; Howard Hughes Medical Institute (HHMI); Mentored Quantitative Research Career Development Award (K25) from National Institutes of Health grant 1K25DK082791-01A109 (to I.R.). Funding for open access charge: Howard Hughes Medical Institute.

Conflict of interest statement. None declared.

REFERENCES

1. Rusk,N. (2009) Focus on next-generation sequencing data analysis. *Nat. Methods*, **6**, S1.
2. Buhler,J. (2001) Efficient large-scale sequence comparison by locality-sensitive hashing. *Bioinformatics*, **17**, 419–428.
3. Andoni,A. and Indyk,P. (2008) Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, **51**, 117–122.
4. Domingo,E., Escarmis,C., Sevilla,N., Moya,A., Elena,S., Quer,J., Novella,I. and Holland,J. (1996) Basic concepts in RNA virus evolution. *FASEB J.*, **10**, 859–864.
5. Wright,S. (1932) The roles of mutation, inbreeding, crossbreeding and selection in evolution. *Proceedings of the Sixth International Congress of Genetics*, **1**, 356–366.
6. Maynard Smith,J. (1970) Natural selection and the concept of a protein space. *Nature*, **225**, 563–564.
7. Poelwijk,F.J., Kiviet,D.J., Weinreich,D.M. and Tans,S.J. (2007) Empirical fitness landscapes reveal accessible evolutionary paths. *Nature*, **445**, 383–386.
8. Schuster,P. (2009) Genotypes and phenotypes in the evolution of molecules. *Eur. Rev.*, **17**, 281–319.

9. Pitt,J.N. and Ferré-D'Amaré,A.R. (2009) Structure-guided engineering of the regioselectivity of RNA ligase ribozymes. *J. Am. Chem. Soc*, **131**, 3532–3540.
10. Domingo,E. and Wain-Hobson,S. (2009) The 30th anniversary of quasispecies. Meeting on 'Quasispecies: past, present and future'. *EMBO Rep*., **10**, 444–448.
11. Musser,D.R. (1997) Introspective sorting and selection algorithms. *Software Pract. Exper*., **27**, 983–993.
12. Cock,P.J.A., Fields,C.J., Goto,N., Heuer,M.L. and Rice,P.M. (2010) The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res*., **38**, 1767–1771.
13. Quail,M.A., Kozarewa,I., Smith,F., Scally,A., Stephens,P.J., Durbin,R., Swerdlow,H. and Turner,D.J. (2008) A large genome center's improvements to the Illumina sequencing system. *Nat. Methods*, **5**, 1005–1010.
14. Hamming,R. (1950) Error detecting and error correcting codes. *Bell Sys. Tech. J*., **29**, 147–160.
15. Schneider,T.D., Stormo,G.D., Gold,L. and Ehrenfeucht,A. (1986) Information content of binding sites on nucleotide sequences. *J. Mol. Biol*., **188**, 415–431.
16. Carothers,J.M., Oestreich,S.C., Davis,J.H. and Szostak,J.W. (2004) Informational complexity and functional activity of RNA structures. *J. Am. Chem. Soc*., **126**, 5130–5137.
17. Eddy,S.R. and Durbin,R. (1994) RNA sequence analysis using covariance models. *Nucleic Acids Res*., **22**, 2079–2088.
18. Strang,G. (2009) *Introduction to Linear Algebra*, 4th edn. Wellesley-Cambridge Press, Wellesley.
19. Anderson,T.W. (2003) *An Introduction to Multivariate Statistical Analysis*, 3rd edn. Wiley-Interscience, New York, USA.
20. Kurtovic,S., Shokeer,A. and Mannervik,B. (2008) Emergence of novel enzyme quasi-species depends on the substrate matrix. *J. Mol. Biol*., **382**, 136–153.
21. Kullback,S. (1997) *Information Theory and Statistics*. Courier Dover Publications, New York.
22. Cover,T.M. and Thomas,J.A. (2006) *Elements of Information Theory*, 2nd edn. Wiley-Interscience, New York.
23. Rajapakse,I., Perlman,M.D., Scalzo,D., Kooperberg,C., Groudine,M. and Kosak,S.T. (2009) The emergence of lineage-specific chromosomal topologies from coordinate gene regulation. *Proc. Natl Acad. Sci. USA*, **106**, 6679–6684.
24. Wolda,H. (1981) Similarity indices, sample size and diversity. *Oecologia*, **50**, 296–302.
25. Hubisz,M.J., Falush,D., Stephens,M. and Pritchard,J.K. (2009) Inferring weak population structure with the assistance of sample group information. *Mol. Ecol. Resour*., **9**, 1322–1332.
26. Lawrence,M., Wickham,H., Cook,D., Hofmann,H. and Swayne,D. (2009) Extending the GGobi pipeline from R. *Computat. Stat*., **24**, 195–205.