Nucleic Acids Research, 2005, Vol. 33, No. 11 ***3667–3677***
doi:10.1093/nar/gki669

A computer program for the estimation of protein and nucleic acid sequence diversity in random point mutagenesis libraries

Michael J. Volles* and Peter T. Lansbury, Jr

Center for Neurologic Diseases, Brigham and Women's Hospital and Department of Neurology, Harvard Medical School, 65 Landsdowne Street, Cambridge, MA 02139, USA

Received March 31, 2005; Revised May 11, 2005; Accepted June 6, 2005

ABSTRACT

A computer program for the generation and analysis of *in silico* random point mutagenesis libraries is described. The program operates by mutagenizing an input nucleic acid sequence according to mutation parameters specified by the user for each sequence position and type of point mutation. The program can mimic almost any type of random mutagenesis library, including those produced via error-prone PCR (ep-PCR), mutator *Escherichia coli* strains, chemical mutagenesis, and doped or random oligonucleotide synthesis. The program analyzes the generated nucleic acid sequences and/or the associated protein library to produce several estimates of library diversity (number of unique sequences, point mutations, and single point mutants) and the rate of saturation of these diversities during experimental screening or selection of clones. This information allows one to select the optimal screen size for a given mutagenesis library, necessary to efficiently obtain a certain coverage of the sequence-space. The program also reports the abundance of each specific protein mutation at each sequence position, which is useful as a measure of the level and type of mutation bias in the library. Alternatively, one can use the program to evaluate the relative merits of preexisting libraries, or to examine various hypothetical mutation schemes to determine the optimal method for creating a library that serves the screen/selection of interest. Simulated libraries of at least 10^9 sequences are accessible by the numerical algorithm with currently available personal computers; an analytical algorithm is also available which can rapidly calculate a subset of the numerical statistics in libraries of arbitrarily large size. A multi-type double-strand stochastic model of ep-PCR is developed in an appendix to demonstrate the applicability of the algorithm to amplifying mutagenesis procedures. Estimators of DNA polymerase mutation-type-specific error rates are derived using the model. Analyses of an alpha-synuclein ep-PCR library and NNS synthetic oligonucleotide libraries are given as examples.

INTRODUCTION

Random point mutagenesis of a nucleic acid sequence is a useful technique for probing structure and function and for directed evolution of proteins, peptides and nucleic acids. Mutagenesis libraries can be created by several methods (1), including error-prone PCR (ep-PCR) (2–4), passage through mutator *Escherichia coli* strains (5), chemical mutagenesis (6) (e.g. sodium bisulfite, nitrous acid), and oligonucleotide synthesis (7) (NNN, NNS, or arbitrary doping; N = A, G, C, or T; S = C or G). A mutagenesis library is typically characterized by its size (number of independent clones) and by how heavily it is mutated. However, statistics such as library diversity (8) (e.g. number of unique sequences, point mutations, and single point mutants), amino acid mutation bias and the distribution of the number of mutations per sequence would in some cases be of more direct interest to the experimentalist, if they were available. The reverse issue of optimizing amino acid mutation bias in synthetic oligonucleotide encoded libraries has been explored previously (7,9–11).

Prior to screening, estimates of these parameters could be used to evaluate the relative merits of preexisting libraries. They can also be used to evaluate a variety of potential mutagenesis schemes and levels, in order to determine the most

*To whom correspondence should be addressed. Tel: +1 617 768 8617; Fax: +1 617 768 8606; Email: mvolles@rics.bwh.harvard.edu

appropriate type of library to construct, for the given objective. Subsequent to selecting or preparing the most appropriate library, one would ideally screen all of its available diversity. However, if the screening process is costly in terms of time or expense, knowledge of the amount of diversity covered as a function of the number of clones screened allows the investigator to determine the optimal screen size. This is the endpoint that allows sufficient and efficient exploration of a defined portion of the library diversity while avoiding inefficient oversampling. Finally, when the screen has been completed, one would like to know how much of sequence space was actually covered, and with what amino acid mutation bias.

A number of library statistics are amenable to an analytical treatment. For example, amino acid mutation frequencies, the fraction of sequences that are wild-type, the length distribution of truncated proteins, and some simpler diversity statistics (the number of unique sequence-position-specific point mutations and single point mutants; Supplementary Appendix H) can be calculated analytically. In contrast, the distribution of the number of mutations per sequence and the overall sequence diversity (number of unique sequences) are in general not analytically tractable.

Previous analytical work has shown that probability theory can be used to estimate the overall diversity of a nucleic acid library when the equations are simplified by requiring that all sequences are equiprobable, or that all mutations occur with a single frequency, independent of wild-type base identity, mutation (e.g. transition versus transversion), and position (12). However, this requirement is generally not upheld during random mutagenesis. In the case of ep-PCR (3,13) (Table 3) and chemical mutagenesis (6), mutation frequencies vary with wild-type base, mutation (there are 3 possible changes for each of the 4 bases, and therefore 12 total mutation types), and possibly with sequence position. While equiprobability of sequences can be experimentally specified during oligonucleotide synthesis (e.g. NNS), in general the composition of every position in the sequence is arbitrarily controlled. Furthermore, the diversity of the translated protein library is often what is of prime interest, but this prediction is even more difficult to handle mathematically (14): even if the underlying nucleic acid sequence variants are equiprobable, the degeneracy of the genetic code results in non-equiprobable amino acid sequence variants (7).

Taking a numerical approach, two previously reported algorithms (14,15) use a Monte Carlo procedure and DNA translation, but allow only a single scalar value for all mutation frequencies at all positions, a small number of iterations, and do not track library diversity (these programs were written in the early 1990's when computer power was much more limiting). To our knowledge, estimates of protein/nucleic acid library diversity cannot be practically obtained with any currently available methods, analytical or numerical.

We describe here a computer program that calculates statistics for, and diversity of, nucleic acid and protein random point mutagenesis libraries. The frequency of all possible nucleic acid mutations at every position in a sequence can be specified independently, enabling one to make predictions about library composition based on the mutation frequencies derived or expected from almost any type of random mutagenesis scheme.

MATERIALS AND METHODS

Construction of a library of randomly mutated α-synuclein cDNA molecules

The ep-PCR method of Cadwell and Joyce was used (3,16). Template for the ep-PCR was generated by standard PCR (non-error generating; *Pfu* polymerase; forward primer: cgagctctccatatggatgtattcatgaaaggac; reverse primer: cgagctctcaagcttggatggaacatctgtcagc) of the α-synuclein cDNA. The template extends from 12 bp upstream of the α-synuclein start codon through ~60 bp downstream of the stop codon, and was purified using agarose gel-electrophoresis. Approximately 30 ng (100 fmol) of template was used in a 100 µl ep-PCR [10 mM Tris, pH 8.3, 50 mM KCl, 0.01% gelatin, 0.2 mM each dATP and dGTP, 1 mM each dCTP and dTTP, 7 mM $MgCl_2$, 0.5 mM $MnCl_2$, 0.3 µM forward and reverse primers (same sequences as above), 5 U *Taq* polymerase]. Thirty reaction cycles [94°C 1 min, 66°C 1 min, 72°C 75 s; this number of cycles is probably excessive, given the maximum amplification under these conditions of 300-fold, but see (3) and Supplementary Appendix D] were performed followed by product purification (Qiagen PCR purification kit).

The insert was digested with NdeI and HindIII, the product was purified again (as above), and the mixture was ligated (Takara ligation kit) into a digested and phosphatased pT7-7 *E.coli* expression vector (17). The ligated DNA was purified into a low-salt buffer for electroporation (Qiagen PCR purification kit). Ligated DNA (10 µl) was added to 30 µl of electrocompetent *E.coli* [strain DH10B prepared by the Dower method (18)] on ice, and two 20 µl electroporations were carried out at 4°C. The electroporator and cuvette were constructed in the laboratory, and provide equivalent efficiency to commercially available devices. The electroporator individually charges each of a set of 12 serially connected capacitors to several hundred volts using an electrophoresis power supply; this provides a total end-to-end voltage in the kilovolt range. The cuvette is Plexiglas with stainless steel electrodes (1.4 mm gap). An initial electric field of ~14 kV/cm with an exponential decay (time constant of 5 ms) was measured with an oscilloscope. Immediately following electroporation, the cells were diluted into 2 ml SOC medium and incubated with shaking for 1 h at 37°C. Aliquots were then plated on LB-ampicillin media to determine the number of transformants, and the remainder was grown in several hundred milliliters of LB-ampicillin media overnight for a midi-prep.

Two additional libraries with similar properties were generated during optimization of this mutagenesis procedure (dNTP, Mn^{++} and Mg^{++} concentrations were never varied). Total overall mutation frequencies of these three individual libraries are similar (range of 0.006–0.009 mutations per base pair). The specific overall DNA mutation frequencies from each of these three libraries also appear to be similar [significant differences between the libraries were only detected in the mutation pair T→A, A→T (χ^2 significance level <0.05)]. Therefore, sequence data from all three sub-libraries were combined.

The computational algorithm

Inputs. First, the initial DNA sequence is read in from a text file, as are a set of mutation parameters. In the case of ep-PCR,

these parameters are 12 probabilities p_{xy} which describe the frequency that the polymerase creates a mutation by misincorporating base y across from base x. Note the distinction between these incorporation probabilities and the actual mutation which results (e.g. p_{aa} creates an A→T mutation; mutation of base x to base y is denoted by $x \rightarrow y$). The estimation of these probabilities from DNA sequencing data is discussed in Appendix B. Also, the number of PCR cycles n and the PCR efficiency λ are specified (rough estimates of these parameters are sufficient, see Supplementary Appendix C.II). In non-amplifying mutagenesis methods such as oligonucleotide synthesis, the input mutagenic parameters are the direct mutation frequencies, for example, the frequency that wild-type base A is mutated to a G. These frequencies may be different at each position in the sequence and for each of the 12 mutation types (i.e. A→G, A→C, A→T, etc.). In both cases, the number of sequences to generate is also specified, and the option is given to proceed with a numerical (see sections below) or analytical algorithm (Supplementary Appendix H). The analytical algorithm has the advantage of speed and can work with an arbitrarily large library, but cannot calculate the total library diversity, the distribution of the number of mutations per sequence, or the distribution of the number of times sequences in the library were repeatedly generated.

Numerical production of a sequence. The program generates a random number with the Mersenne Twister algorithm (19). This number is used to decide whether to accept one of the three possible mutations or to leave the base as wild-type. The decision to accept a mutation occurs with a specified probability, discussed in the next section. The program then repeats this procedure, using a new random number and the applicable acceptance frequencies, for the second base, the third base, and so on, to the end of the sequence. This single-pass mutagenesis is in contrast with the multi-pass, amplifying process of PCR. One object of Appendix A is to define a method by which the result of the latter process is simulated using the former.

How the acceptance frequencies are determined from the inputs. The method of determining the acceptance frequencies differs for non-amplifying methods (e.g. oligonucleotide synthesis, chemical mutagenesis) and amplifying methods (ep-PCR, *E.coli* mutator strain). For the case of non-amplifying methods, the acceptance frequencies are simply the inputs directly specified by the user, as described above. In the case of ep-PCR, the acceptance frequencies are determined for each new sequence as follows: First, the generation number of the sequence and the strand (top or bottom) of its zeroth generation ancestor are chosen randomly, according to their probability distributions (these terms and their probability distributions are defined in Appendix A). Inclusion of the zeroth generation (initial templates) in the probability distribution used to choose generation is optional. In some experimental procedures, these molecules are not incorporated into the library, for example, if they are lacking restriction sites (for subsequent cloning) introduced using the PCR primers. Having decided these two values, the appropriate mutation acceptance rates are derived by a series of matrix transformations on the input polymerase incorporation frequencies (see Appendix A for details). These frequencies are then held constant throughout production of a single sequence, because all bases of a sequence share the same generation and ancestor strand.

Protein translation. If protein diversity is being examined, the mutated DNA is translated. The program uses the standard genetic code by default, but by altering a single line of the program code one can specify an alternative codon translation, for example, as necessary with amber stop codon suppression or mitochondrial protein synthesis.

Further iterations. This process for mutating a wild-type sequence is then repeated exactly as above, the number of times (number of sequences) specified by the user. Each iteration begins anew with a wild-type sequence and is independent of any previous iteration.

Library storage in memory. Every mutated sequence is stored in a binary search tree (20) as it is generated. Each mutation requires two bytes of memory, the low bits (5 for protein, 2 for DNA) of which store the mutation type, while the remaining high bits are used to store the sequence position. This scheme limits protein sequences to ~2000 amino acids [$2^{(16-5)}$], but this should be adequate for almost all cases. DNA alone can be examined to lengths of ~16 kb [$2^{(16-2)}$]. Sequences are unambiguously ordered in the tree using a scheme based on the number, position, and types of mutation they contain. The number of times each sequence has been generated is also recorded. Memory is allocated dynamically for all critical parts of the algorithm, so that the problem size accessible to the program is only limited by machine hardware. For simulations which have memory requirements beyond the physical memory of the computer, an efficient disk caching routine was written which significantly extends the upper size limit of practical simulations. Reliance on operating system disk paging for this purpose would have been unacceptably slow; the non-locality of reference of the binary tree would cause continuous seeking on the hard disk.

Library analysis. Statistics of the nucleic acid and/or protein libraries are collected during and following the sequence iterations. For diversity estimation, these are the number of unique sequences, mutations, and single point mutants as a function of the total number of sequences generated, the distribution of the number of mutations per sequence, the distribution of the number of times sequences in the library were repeatedly generated, and a listing of the most often generated sequences. Additional statistics include the specific mutation frequencies generated at each position, the total mutation frequencies summed over all positions, the number of extended sequences (protein stop to sense mutation), and the distribution and number of truncations. When truncated proteins are generated, the occurrence and sequence position is recorded, but the sequence is not used in other statistics. The user has the option of discarding or keeping sequences in which a stop codon has been mutated to a sense codon.

Library size limitations. With current personal computers, physical memory is the most likely factor to set the upper practical limit on numerical simulation size, which will typically be on the order of 10^9–10^{10} sequences for average problems. Although the scratch disk function extends the upper limit well beyond that which would be possible with physical

memory alone, the nature of the disk cache routine and the slowness of disk access speed still set an upper boundary which is a function of available physical memory (the frequency with which the scratch function is used is inversely correlated with physical memory size). Modern supercomputers often provide tens of GBs of physical memory and a large amount of scratch space, which should extend the currently possible simulation size to at least 10^{11} sequences. Continual growth in available computing power may make ribosomal display size libraries (10^{12} or more sequences) accessible to the algorithm within several years.

The computer program was written in C, compiled using Microsoft Visual C++, and is portable with only minor changes (for 64-bit integer support). The program is available on our laboratory website, http://lansbury.bwh.harvard.edu/michael_volles.htm.

Table 1. Summary of α-synuclein DNA sequencing data from 89 sequences[a]

Wild-type base:	Number of times observed as: A	G	C	T
A	11 233	64	19	73
G	20	11 979	2	12
C	8	2	6296	13
T	44	8	24	5975

[a]These data were derived from the plus strand of all available sequences (89 sequences, 36 kb; see Methods section) including those selected for by expression of purifiable protein as well as unselected sequences. Significant differences between the observed mutation levels of selected and unselected sequences were not detected (all six χ^2 values with Yates correction were of P-value >0.3). Sequencing data were taken from the final base of the forward primer through the stop codon of the synuclein cDNA. We observed almost no mutations in the first 22 bp downstream of the start codon, which are complementary to the forward primer. The reverse primer is complementary to the 3′-untranslated region and, therefore, does not influence mutation frequencies in the coding sequence. The observed overall mutation frequency in the library was 0.0081 mutations per base, or an average of 3.2 mutations per DNA sequence.

RESULTS AND DISCUSSION

Example 1: ep-PCR of the α-synuclein gene

An α-synuclein ep-PCR random mutagenesis library was created, and will be analyzed below in order to demonstrate the type of results which can be expected from the algorithm and the types of conclusions that can be drawn from them. The library contains $\sim 3.8 \times 10^6$ clones with full length, potentially expressible, inserts. A number of these were sequenced (Table 1), and the data together with the estimator of Appendix B were used to derive polymerase incorporation frequencies (Table 3).

Using these incorporation frequencies, we estimated the characteristics of the protein library (Table 2) by generating and analyzing $\sim 3.8 \times 10^6$ sequences with the computer algorithm. A larger library containing 10^9 sequences was also generated with the program (not experimentally), for the purpose of comparison. The DNA diversity and statistics will not be discussed, since we are primarily interested in the protein translation of this library. The simulation of the library required 2–3 min and 100 MB of memory for $\sim 3.8 \times 10^6$ sequences (both protein and DNA; the size of our actual library) and 10 h with 1 GB of RAM and the disk cache function enabled for 10^9 sequences (protein only), using a 1.8 GHz Pentium IV computer.

The standard deviations of the average values are quite low (Table 2); with simulations of this size or greater, random fluctuations in the output of the algorithm can be considered negligible, relative to errors from other sources. The width of the 99% confidence intervals on the estimated properties leads us to conclude that a modest amount of sequencing (e.g. Table 1) is sufficient for the algorithm to produce very useful estimates.

Figure 1A shows the number of unique sequences (black diamonds), unique point mutations (red squares), and unique single point mutants (green triangles) which were produced during the simulation, as a function of the number of

Table 2. Estimated characteristics of the α-synuclein protein library[a]

Property	Average[d]	Standard deviation[d]	99% Confidence interval[e]
Truncations (%)	15	0.02	12.0–18.3
Stop codon mutated to a sense codon[b] (%)	3	0.01	2.4–3.5
Clones producing full length α-synuclein	3.1×10^6	1.0×10^3	3.0×10^6–3.2×10^6
Protein mutation frequency per amino acid[c]	0.016	0.00001	0.013–0.018
Average number of mutations per protein[c]	2.1	0.001	1.7–2.4
Unmutated sequences (wild-type)[c] (%)	16	0.017	12.7–21.3
Number of unique proteins[c]	1.3×10^6	0.8×10^3	1.0×10^6–1.5×10^6
Number of unique point mutations[c,f]	1990	13	1766–2074
Number of unique single point mutants[c]	1566	12	1438–1660

[a]All data result from simulations of 3 770 580 sequences, which is the approximate number of independent, full-length, in-frame inserts in our library. The polymerase incorporation paired frequencies of Table 3 were used centrosymmetrically [$E(\mathbf{N})_{cen}$, see Appendix B]. As with the experimental library, initial template sequences were allowed to be candidates for incorporation into the simulated library. The values of n and λ for the simulations were identical to those used in the estimator, 9 and 0.88, respectively.

[b]As a percentage of the untruncated clones.

[c]Calculated after removing truncated proteins and proteins with the stop codon mutated to a sense codon. However, these occurrences are counted towards the total number of sequences generated (see Figure 1 legend). Data is derived from amino acid 8 through the stop codon; the forward PCR primer is complementary to the DNA corresponding to the first 7 amino acids.

[d]Averages and estimates of the SD are based on 10 independent simulations.

[e]Calculated by the method of Supplementary Appendix C.I with 2000 bootstrap replicates of 89 sequences each. The sampling distribution of the statistics showed that they were unbiased (Supplementary Appendices C and G). In rare cases (37 of 2000), the bootstrap sample contained no G→C or C→G mutations, resulting in very small negative values in the corresponding elements of $E(\mathbf{N})_{cen}$. These were adjusted to zero as discussed in Appendix B.

[f]The bootstrap sampling distribution of this property was significantly skew to the left. The other properties had sampling distributions which were quite normal.

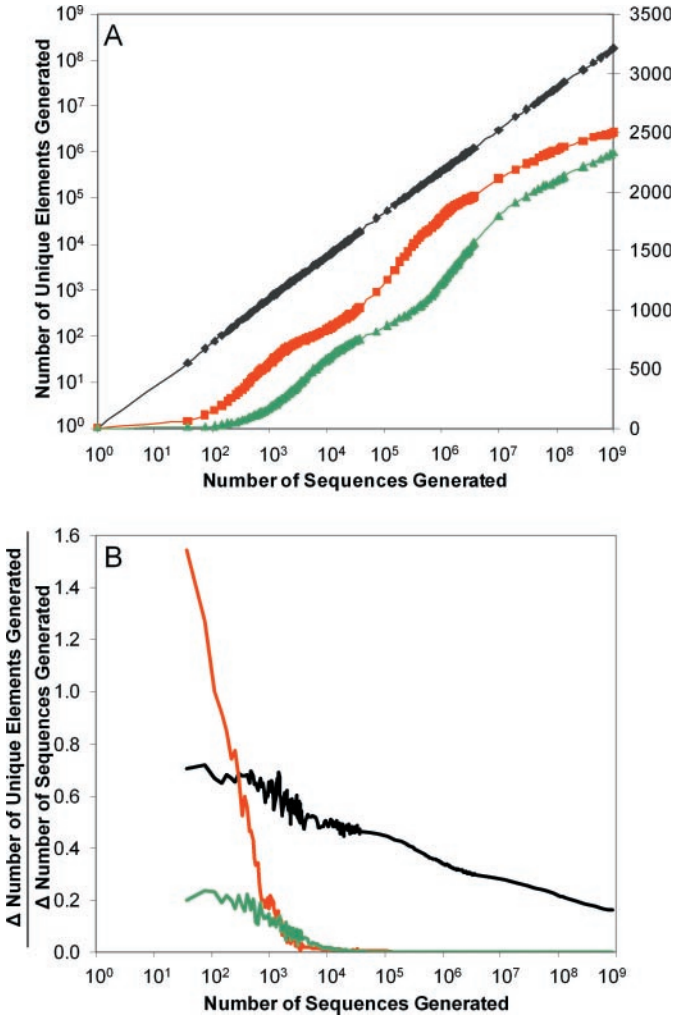

Figure 1. α-Synuclein ep-PCR protein library diversity (**A**) and screening efficiency (**B**) as a function of the library/screen size. (A) The black diamonds show the total number of unique sequences (non-wild-type) generated (left-hand *y*-axis, logarithmically spaced). The red squares indicate the number of unique point mutations generated; a unique point mutation is defined as a particular amino acid change at a particular position of the sequence, which has not occurred before. A single sequence may contain multiple unique point mutations. The green triangles refer to the number of unique single point mutants generated; a single point mutant is defined as a sequence with only a single amino-acid mutation, occurring at a specified sequence position. The general term 'unique elements' is used on the *y*-axis and encompasses all three of the above terms. The theoretical maximum number of unique point mutations and unique single point mutants is 19 times the sequence length; both are referred to the right-hand, linear, *y*-axis). Proteins with truncations or extensions (mutated stop codon) are not included on either *y*-axis, but are counted as generated sequences in the *x*-axis. This mimics an actual screen: significantly truncated or extended proteins are often not effective candidates, yet they must still be screened because they cannot be easily removed from the library. Points with abscissa values below 40 000 are averages of values from three independent simulations. Sampling without replacement is assumed (see Supplementary Appendix E). (B) The efficiency of sequence space coverage as a function of number of sequences screened. The efficiency is the expected number of unique elements [a new sequence variant (black curve), new point mutations (red curve), or a new single point mutant (green curve)] which will be covered by screening one additional clone. These curves are the derivatives of the curves in (A). Values may be greater than one for the efficiency of discovering new point mutations (red curve), because multiple new mutations can occur in a single sequence. Note that while our actual experimental library contains $\sim 3.8 \times 10^6$ sequences, the numerical simulations here were carried out for up to 10^9 sequences, for the purpose of example. The parameters used in the simulations are the same as those described in Table 2.

sequences generated. The slopes of the lines in Figure 1A are shown with an identical color scheme in Figure 1B. As discussed in the introduction, this information is important for judging the appropriate screen size to use, and the efficiency of the screening procedure in examining sequence space. For example, if one were interested in looking at as many point mutations as possible, without regard for whether multiple point mutations existed per sequence, the red curves in Figure 1 would be appropriate. In the first several thousand clones examined, the efficiency ranges from >1.6 new mutations examined per clone, down to ~0.05, and a total of ~700 different mutations can be expected after 2500 clones. Screening for new point mutations beyond this first several thousand sequences is very inefficient; only infrequently will a new point mutation be looked at. In this system, there are 2527 possible single point mutations (133 mutatable amino acids times 19). Therefore, if the screening procedure is not trivial in terms of time and expense, and if one wished to look beyond the ~25% of those mutations which can be efficiently examined in this library, a different experimental approach to creating the library might be necessary. However, almost all of the possible point mutations would be accessible in a full screen of a library with the mutation frequencies shown in Table 3, and which contained 10^9 independent clones. This is relevant if the screen/selection size is not limiting (e.g. phage display). If one wishes to consider only unique single point mutants, the situation is similar (Figure 1, green), but somewhat higher numbers of clones must be screened relative to the unrestricted case (Figure 1, red) in order to get equal coverage. A similar analysis of the number of unique sequence variants (Figure 1, black), shows that a library of 10^9 clones can be mined for new variants with comparative efficiency throughout its entirety (at least one in five clones examined are new sequences).

The relative frequencies of each type of protein mutation from each wild-type amino acid are shown in Figure 2. As expected, some mutations are much more probable than others. One reason for this is that many protein mutations require two or three DNA base changes in a single codon, which is an infrequent event in ep-PCR. Other contributing

Table 3. Estimated polymerase paired incorporation frequencies for the α-synuclein ep-PCR DNA library[a]

Specific event	Probability
p_{aa}, p_{tt}	0.00161
p_{ag}, p_{tc}	0.00037
p_{ac}, p_{tg}	0.00121
p_{ga}, p_{ct}	0.00026
p_{gg}, p_{cc}	0.00005
p_{gt}, p_{ca}	0.00043

[a]This set of six paired mutagenic incorporation frequencies was derived using the estimator of Appendix B with the sequencing data of Table 1, and $n = 9$ and $\lambda = 0.88$. The minimum value of k in the estimator was taken as zero, because this library can potentially incorporate initial template sequences (see Appendix B). The values of n and λ were chosen as reasonable estimates that also coincide with the experimental amplification factor of ~300. The accuracy of this set of probabilities, as measures of the paired incorporation frequencies of a real polymerase, depend strongly on the accuracy of the estimated values of n and λ. In contrast, the results of the simulation/analysis are relatively insensitive to these two parameters (Supplementary Appendix C.II). Statistical uncertainty in these values is reflected in the confidence intervals of Table 2.

factors are that certain DNA mutations are much more frequent than others (Table 3) (3,13), and that amino acids have varying degrees of degeneracy with respect to the genetic code. This mutation bias underlies, in part, the limitations on screening efficiency shown in Figure 1. The bias is further underscored by the output of the program on the most frequently generated protein sequences, which are a pool of several hundred single point mutants. The most common, single point mutant F87L, exists more than 10^6 times in a library of size 10^9, and the top 100 sequences make up 7% of all 10^9 generated sequences. The computer program also reports the number of occurrences of each mutation type at each position of the sequence (data not shown). This allows for the examination of region or codon specific bias.

Another way to examine the data in Figure 2 is by grouping amino acids with similar properties. Specifically, the *x*- and *y*-axes in Figure 2 are arranged in order of increasing hydropathy index (21). Consider Figure 2 divided into four quadrants: hydrophobic→hydrophobic, hydrophobic→ hydrophilic, hydrophilic→hydrophobic, hydrophilic→hydrophilic. All four are reasonably well populated; by this measure, the bias is not nearly as great as in the above analysis. In certain cases, e.g. optimization of structural stability, this latter approach may be the most appropriate bias indicator. In others, such as optimization of a catalytic site, the former may be more relevant.

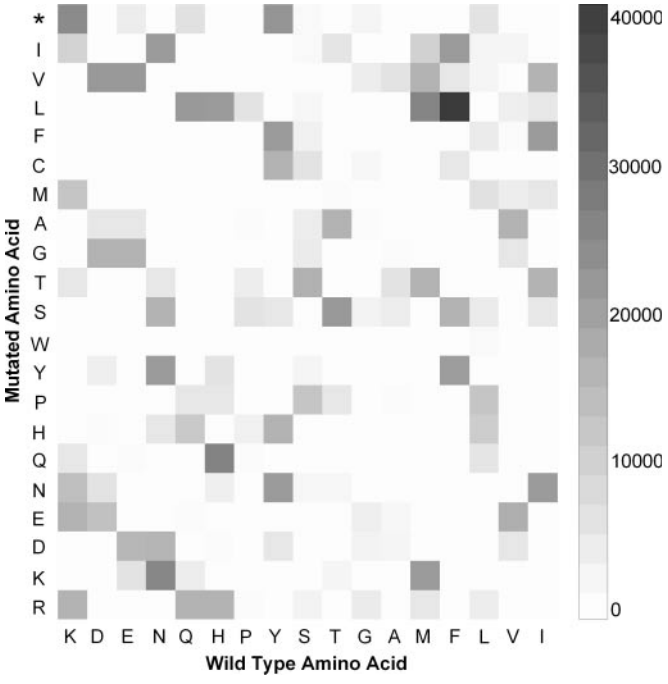

Figure 2. Mutation frequencies and bias as a function of wild-type amino acid in an α-synuclein ep-PCR library. A grey-scale is used to show the number of times (labeled every 10 000 units) a particular wild-type amino acid (*x*-axis) was observed to be changed to a particular mutated amino acid (*y*-axis) after numerical simulation of 3 770 580 sequences (see Table 2 legend for details), normalized (divided) by the number of times the wild-type amino acid appears in the sequence. Sense mutations are recorded only in untruncated, unextended sequences. Positions in the graph corresponding to wild-type (no mutation) have values of zero. Both the *y*- and *x*-axes are arranged in order of increasing hydropathy index (21). Mutation to a stop codon is represented by an asterisk. α-Synuclein does not contain cysteine, arginine, or tryptophan, therefore, these amino acids do not appear on the *x*-axis. The graph was created using HeatMap Builder (http://mozart.stanford.edu/heatmap.htm).

Example 2: synthetic NNS oligonucleotides

Simulations were performed of synthetic NNS oligonucleotides of various lengths, in multiples of three, and after

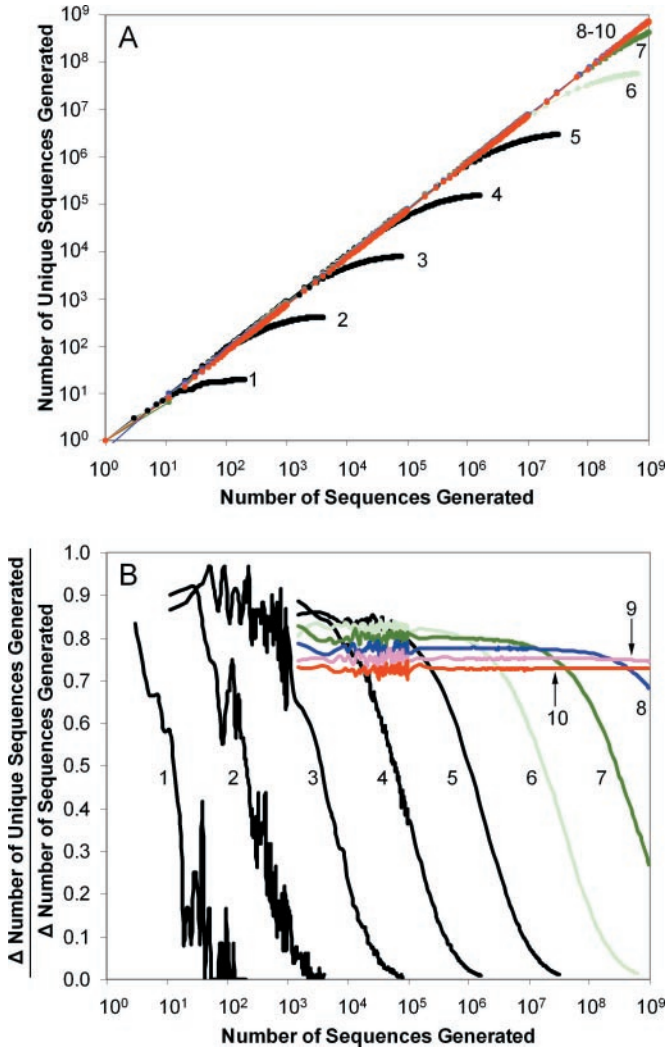

Figure 3. Peptide diversity and screening efficiency of NNS (S = G or C) oligonucleotide libraries. (**A**) Number of unique peptide sequences generated (logarithmically spaced *y*-axis) as a function of the number of NNS DNA sequences produced (log-spaced *x*-axis). Sequences which contained a stop codon were discarded during the simulation. Results with varying numbers of NNS triplets are labeled with amino acid length. Data for all 1 through 10mer peptides are shown, but 8–10, which have a much higher diversity than the maximum number of simulated sequences, are not separately discernible. Data points below 1000 represent average values from three separate runs, to reduce the noise inherent in very small simulations. Sampling without replacement is assumed (see Supplementary Appendix E). (**B**) Efficiency (*y*-axis; see Figure 1 legend for description) as a function of number of sequences screened (log-spaced *x*-axis). Colors are used to distinguish some of the overlapping curves (6mer, light green; 7mer, dark green; 8mer, blue; 9mer, pink; 10mer, red). The colors in (A) follow an identical scheme. The abrupt decrease in noise level at 100 000 in (B) is due to an increase in the Δx used to calculate slope from the points in (A), and is not inherent in the data. Note the decrease in initial efficiency [visible in (B), especially in the less noisy, colored, curves], as the size of the peptide increases: longer sequences have a greater chance of being eliminated due to incorporation of a nonsense codon.

translation the peptide libraries were examined. The wild-type DNA sequence was specified as the appropriately sized poly-A, with 25% mutation frequencies to each of the three other bases (for N), or a 50% chance each of mutating to G or C (for S). Protein diversity, in terms of number of unique sequences, as a function of library size, is shown in Figure 3A. We simulated the lesser of 10 times the theoretical number of unique sequences or 10^9 sequences, per peptide length; the latter typically required 8 h on the machine described above. These statistics are useful for estimating the number of peptides which need to be screened in order to approach a certain coverage of the available diversity. The slopes of the curves in Figure 3A are shown in Figure 3B, describing the efficiency of continued screening, as a function of number of clones screened. Note that because the NNS library is well-distributed, with relatively little mutation bias, the curves of Figure 3A are regular and evenly spaced (compare Figure 1A; this suggests that an analytic fit of this empirical data may be usefully extrapolated in order to solve larger, otherwise inaccessible problems). Still, in most cases the point at which efficiency drops off severely (Figure 3B) would not be entirely predictable without some sort of diversity sampling calculation. On the other hand, with a very large unbiased library, such as the NNS 10mer library, diversity predictions are superfluous; we know that essentially every additional clone examined will be unique.

SUPPLEMENTARY MATERIAL

Supplementary Material is available at NAR Online.

ACKNOWLEDGEMENTS

We thank Alina M. Vrăbioiu and Allegra A. Petti for critical reading of the manuscript and helpful discussions. This work was supported by a Morris K. Udall Parkinson's Disease Research Center of Excellence Grant (NS38375). M.J.V. is supported by a postdoctoral fellowship from the American Parkinson Disease Association. Funding to pay the Open Access publication charges for this article was provided by NS38375.

Conflict of interest statement. None declared.

REFERENCES

1. Neylon,C. (2004) Chemical and biochemical strategies for the randomization of protein encoding DNA sequences: library construction methods for directed evolution. *Nucleic Acids Res.*, **32**, 1448–1459.
2. Cadwell,R.C. and Joyce,G.F. (1992) Randomization of genes by PCR mutagenesis. *PCR Methods Appl.*, **2**, 28–33.
3. Cadwell,R.C. and Joyce,G.F. (1995) In Dieffenbach,C.W. and Dveksler,G.S. (eds), *PCR Primer: A Laboratory Manual*. Cold Spring Harbor Laboratory Press, Plainview, NY, pp. 583–589.
4. Leung,D.W., Chen,E. and Goeddel,D.V. (1989) A method for random mutagenesis of a defined dna segment using a modified polymerase chain reaction. *Technique*, **1**, 11–15.
5. Greener,A., Callahan,M. and Jerpseth,B. (1997) An efficient random mutagenesis technique using an *E. coli* mutator strain. *Mol. Biotechnol.*, **7**, 189–195.
6. Myers,R.M., Lerman,L.S. and Maniatis,T. (1985) A general method for saturation mutagenesis of cloned DNA fragments. *Science*, **229**, 242–247.
7. Arkin,A.P. and Youvan,D.C. (1992) Optimizing nucleotide mixtures to encode specific subsets of amino acids for semi-random mutagenesis. *Biotechnology*, **10**, 297–300.
8. Makowski,L. and Soares,A. (2003) Estimating the diversity of peptide populations from limited sequence data. *Bioinformatics*, **19**, 483–489.
9. Tomandl,D., Schober,A. and Schwienhorst,A. (1997) Optimizing doped libraries by using genetic algorithms. *J. Comput. Aided Mol. Des.*, **11**, 29–38.
10. Jensen,L.J., Andersen,K.V., Svendsen,A. and Kretzschmar,T. (1998) Scoring functions for computational algorithms applicable to the design of spiked oligonucleotides. *Nucleic Acids Res.*, **26**, 697–702.
11. Wolf,E. and Kim,P.S. (1999) Combinatorial codons: a computer program to approximate amino acid probabilities with biased nucleotide usage. *Protein Sci.*, **8**, 680–688.
12. Patrick,W.M., Firth,A.E. and Blackburn,J.M. (2003) User-friendly algorithms for estimating completeness and diversity in randomized protein-encoding libraries. *Protein Eng.*, **16**, 451–457.
13. Vanhercke,T., Ampe,C., Tirry,L. and Denolf,P. (2005) Reducing mutational bias in random protein libraries. *Anal. Biochem.*, **339**, 9–14.
14. Ophir,R. and Gershoni,J.M. (1995) Biased random mutagenesis of peptides: determination of mutation frequency by computer simulation. *Protein Eng.*, **8**, 143–146.
15. Siderovski,D.P. and Mak,T.W. (1993) RAMHA: a PC-based Monte-Carlo simulation of random saturation mutagenesis. *Comput. Biol. Med.*, **23**, 463–474.
16. Wilson,D.S. and Keefe,A.D. (2001) Random mutagenesis by PCR. In Ausubel,F.M., Brent,R., Kingston,R.E., Moore,D.D., Seidman,J.G., Smith,J.A. and Struhl,K. (eds), *Current Protocols in Molecular Biology Online*. John Wiley & Sons, Inc., New York, pp. 8.3.
17. Tabor,S. (2001) Expression using the T7 RNA polymerase/promoter system. In Ausubel,F.M., Brent,R., Kingston,R.E., Moore,D.D., Seidman,J.G., Smith,J.A. and Struhl,K. (eds), *Current Protocols in Molecular Biology Online*. John Wiley & Sons, Inc., New York, pp. 16.2.
18. Dower,W.J., Miller,J.F. and Ragsdale,C.W. (1988) High efficiency transformation of *E.coli* by high voltage electroporation. *Nucleic Acids Res.*, **16**, 6127–6145.
19. Matsumoto,M. and Nishimura,T. (1998) Mersenne Twister: A 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans on Modeling and Computer Simulation*, **8**, 3–30.
20. Deitel,H.M. and Deitel,P.J. (1994) *C: How to Program, 2nd edn*. Prentice Hall, Englewood Cliffs, NJ.
21. Kyte,J. and Doolittle,R.F. (1982) A simple method for displaying the hydropathic character of a protein. *J. Mol. Biol.*, **157**, 105–132.
22. Miller,J.H. (1996) Spontaneous mutators in bacteria: insights into pathways of mutagenesis and repair. *Annu. Rev. Microbiol.*, **50**, 625–643.
23. Krawczak,M., Reiss,J., Schmidtke,J. and Rosler,U. (1989) Polymerase chain reaction: replication errors and reliability of gene diagnosis. *Nucleic Acids Res.*, **17**, 2197–2201.
24. Sun,F. (1995) The polymerase chain reaction and branching processes. *J. Comput. Biol.*, **2**, 63–86.
25. Weiss,G. and von Haeseler,A. (1995) Modeling the polymerase chain reaction. *J. Comput. Biol.*, **2**, 49–61.
26. Wang,D., Zhao,C., Cheng,R. and Sun,F. (2000) Estimation of the mutation rate during error-prone polymerase chain reaction. *J. Comput. Biol.*, **7**, 143–158.
27. Weiss,G. and von Haeseler,A. (1997) A coalescent approach to the polymerase chain reaction. *Nucleic Acids Res.*, **25**, 3082–3087.
28. Piau,D. (2002) Mutation-replication statistics of polymerase chain reactions. *J. Comput. Biol.*, **9**, 831–847.
29. Moore,G.L. and Maranas,C.D. (2000) Modeling DNA mutation and recombination for directed evolution experiments. *J. Theor. Biol.*, **205**, 483–503.
30. Piau,D. (2004) Immortal branching Markov processes: averaging properties and PCR applications. *Ann. Probab.*, **32**, 337–364.

APPENDICES

Appendix A: a multi-type double-strand mathematical model of independent sequences from an ep-PCR

The algorithm described here stochastically mutates the bases of a wild type-sequence, stores the finished sequence, and

then repeats the process, beginning again with a new wild type-sequence. No replication of previously mutated sequences is involved. In contrast, ep-PCR entails multiple rounds of mutagenesis with amplification. We here define the method by which our single-pass algorithm accurately models experimental ep-PCR libraries. The treatment should also apply to *E.coli* mutator strain mutagenesis, whose mechanism similarly involves uncorrected errors of replication, with amplification (22).

The type of model. Previous analytical work (23–28) has modeled the mutagenic PCR process using a generic single DNA base type, a single mutation type, and a single DNA strand. An additional report considered a multi-type PCR model, but did not address the issue of DNA double-strandedness (29). We, therefore, developed for our purposes a mathematical model of ep-PCR that is multi-type and double-stranded; it considers the distinctness of the 4 nucleotides, and explicitly treats the double-stranded nature of DNA. We assume that replication of a strand occurs with constant probability (efficiency) λ. Note that this number expresses the probability that a replication is completed, the alternative being that a replication is not begun; partially completed replications are not considered, nor are insertion and deletion mutations. The appropriateness and goodness-of-fit to the data of this type of model is considered in Supplementary Appendix C.III.

DNA polymerase in the model. The polymerase activity is parameterized by 16 base incorporation probabilities (e.g. the probability the polymerase will install a T across from a G, causing a G→A mutation) which are conveniently expressed as p_{ij} = the probability that the polymerase pairs base j with template base i. These probabilities are one of the inputs specified by the user, and are considered constant throughout the PCR. They are also assumed to be sequence-context independent [for experimental justification, see (2,13); estimators for these parameters are derived in Appendix B]. In theory one could account for sequence context dependence, but this will not be considered here. It is currently impractical to gather enough sequencing data to routinely determine position dependent mutation frequencies. Since in our model a polymerase always pairs a base i with exactly one other base,

$$\sum_{j=1}^{4} p_{ij} = 1$$

Twelve of the sixteen possible incorporations are mutagenic, and the other four are the standard Watson–Crick base pairs. Using the above equation, we are free to set each Watson–Crick base pair equal to 1 minus the sum of the other three mutagenic pairings; there are 12 degrees of freedom.

We now introduce a 4×4 random matrix $\mathbf{N}$, which may intuitively be thought of as representing the stochastic operation of a polymerase incorporating a single base. The members of the set of matrices which $\mathbf{N}$ can realize (the individual members of this set we designate $\mathbf{n}_x$, where x ranges from 1 to 4^4) each have the following three properties: (i) Every row contains a single element 1, and three 0 elements. A 1 at some position ij of a particular realization of $\mathbf{N}$ indicates that given a base i as template, the polymerase will copy a base j across from it. (ii) The probability that element $\mathbf{N}_{ij}$ is a 1 is given by the probability p_{ij}. (iii) Each row is completely independent of the other 3 rows.

DNA sequences in the model. We can specify the sequence of a single-stranded DNA molecule of length L bases using an $L \times 4$ matrix $\mathbf{Z}$. The four elements in any row of $\mathbf{Z}$ consist of three zeros and a single 1. The coordinate that contains the 1 defines the base type at that position according to the order (A,G,C,T; the same arbitrary order must also be used in the rows and columns of matrix $\mathbf{N}$). For example, a base G is represented by [0,1,0,0].

The notation $\mathbf{Z}_{kr}$ will be used to refer specifically to a sequence which is a generation k PCR product whose initial strand ancestor in the PCR was a 'top' strand ($r = 0$), or 'bottom' strand ($r = 1$). Generation number is defined as the number of polymerase copying events separating the molecule from a zeroth generation (initial template) sequence. Of course, all bases of a single strand of DNA are the same generation k and have the same value of r.

For convenience, we will deal exclusively with $\mathbf{Z}$ matrices expressed in terms of an arbitrarily designated 'top' strand composition (as opposed to its bottom-strand complement); we transform bottom-strand composition to top by right operating with the exchange matrix $\mathbf{T}$:

$$\mathbf{T} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Note that for $x \geqslant 0$:

$$\mathbf{T}^x = \begin{cases} \mathbf{T} & x \text{ odd} \\ \mathbf{I} & x \text{ even} \end{cases}$$

where $\mathbf{I}$ is the identity matrix.

A PCR typically begins not only with top strand templates (designated $\mathbf{Z}_{00}$), but also with their bottom strand complements ($\mathbf{Z}_{00}\mathbf{T}$). $\mathbf{T}$ is also applied to expressions as necessary, in order that final product sequences are always referenced to top strand composition. This matches the reality of ep-PCR: reaction products are cloned in *E.coli* prior to DNA sequencing, thereby converting (we assume without error) all bottom strand PCR products to top plus bottom strand (a related issue is discussed in Supplementary Appendix D). One can think of matrix $\mathbf{T}$ as the particular realization of $\mathbf{N}$ that produces mutation-free copies.

First generation products. We operate with $\mathbf{N}$ on the right to transform a zeroth generation top strand template sequence ($\mathbf{Z}_{00}$) into a first generation, random, bottom strand copy ($\mathbf{Z}_{10}$):

$$\mathbf{Z}_{10} = (\mathbf{Z}_{00}\mathbf{N})\mathbf{T}.$$

As discussed above, $\mathbf{T}$ is used in order that the first generation product, random matrix $\mathbf{Z}_{10}$ (which exists as a bottom strand), is given in terms of top strand composition.

Similarly, beginning with a bottom strand template, we generate a top strand:

$\mathbf{Z}_{11} = (\mathbf{Z}_{00}\mathbf{T})\mathbf{N}$.

Higher generation products. To generate a random second generation sequence, we operate with $\mathbf{N}$ on a first generation sequence. This process may be repeated indefinitely to generate higher generation sequences. In general, random matrix $\mathbf{Z}_{kr}$, which is a generation k (≥ 1) sequence with r-strand zeroth generation ancestor, will result from operating on the right of $\mathbf{Z}_{00}\mathbf{T}^r$ with k independent random matrices $\mathbf{N}_l$:

$$\mathbf{Z}_{kr} = \mathbf{Z}_{00}\mathbf{T}^r\left(\prod_{l=1}^{k}\mathbf{N}_l\right)\mathbf{T}^{k+r}.$$

The final $\mathbf{T}^{k+r}$ in the equation converts sequences to top strand composition, where necessary. A matrix to the zeroth power is to be interpreted as the identity matrix. Note that the possible realizations of the product of k independent random $\mathbf{N}$ matrices is the set of $\mathbf{n}_x$. The subscript l is used to emphasize that each of the matrices $\mathbf{N}$ (total of k) are independent and identically distributed; each row of $\mathbf{Z}_{kr}$ is also generated with a completely independent set of $\mathbf{N}_l$. In fact, every $\mathbf{N}$ of every equation in this article is independent and is constant for only a single base; a polymerase has no history in our model. Consequently, only a single row from any given $\Pi(\mathbf{N}_l)$ is ever used when generating sequences, and in practice it is sufficient to generate this random vector alone.

Overall distribution of ep-PCR products. We now consider the total products of a complete ep-PCR, and how one should computationally model an ep-PCR library. We can use the above equation to simulate a random sequence chosen from the library, once we have specified a realization of K and R according to their probability distributions. Sun (24) derived the probability distribution of K after n PCR cycles, assuming a large number of initial templates relative to the numerical value of the PCR amplification factor squared. This is always satisfied in typical ep-PCRs, though the assumption is not stringent (24,28).

$$\mathrm{P}(K=k) = \binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}$$

The sequence will also have originated from either a top strand or a bottom strand, described by random variable $R = 0$ or 1; the probability distribution of R will be divided equally between the strands.

All the bases of a single strand share the same generation (value of $K = k$) and initial template strand (value of $R = r$), but are mutated (the $\mathbf{N}_l$) independently. Mathematically, the random variables R and K are chosen once, and kept constant throughout a single sequence.

When computationally producing a base in a sequence, as stated above, we only need to generate a realization of the single relevant row vector of $\Pi(\mathbf{N}_l)$. Since the vector will contain three zeros and a 1, its probability distribution is completely defined if one knows the probability of a one being in each of the four positions. Given that $K = k$,

$$\mathrm{P}\left(\left(\prod_{l=1}^{k}\mathbf{N}_l\right)_{ij} = 1\right) = \left(\mathrm{E}\left(\prod_{l=1}^{k}\mathbf{N}_l\right)\right)_{ij},$$

$$\mathrm{E}\left(\prod_{l=1}^{k}\mathbf{N}_l\right) = \sum_{\substack{\text{all } 4^{4k} \text{ possible length-}k \\ \text{products of } \mathbf{n}_x}}\left[\prod_{x=1}^{k}(\mathbf{n}_x\mathrm{P}(\mathbf{N}=\mathbf{n}_x))\right]$$

$$= \left[\sum_{x=1}^{4^4}(\mathbf{n}_x\mathrm{P}(\mathbf{N}=\mathbf{n}_x))\right]^k = \mathrm{E}(\mathbf{N})^k.$$

Note that:

$$\mathrm{E}(\mathbf{N}) = \begin{bmatrix} p_{aa} & p_{ag} & p_{ac} & 1-(p_{aa}+p_{ag}+p_{ac}) \\ p_{ga} & p_{gg} & 1-(p_{ga}+p_{gg}+p_{gt}) & p_{gt} \\ p_{ca} & 1-(p_{ca}+p_{cc}+p_{ct}) & p_{cc} & p_{ct} \\ 1-(p_{tg}+p_{tc}+p_{tt}) & p_{tg} & p_{tc} & p_{tt} \end{bmatrix}.$$

We have defined how to computationally model one sequence from the total PCR products, by specifying the appropriate mutation acceptance frequencies for a single-pass Monte Carlo mutagenesis. In general, subsequent sequences from a library cannot be considered independent from the first, since they may share a common ancestor. However, the number of initial templates will almost always be significantly greater than the number of individuals screened or selected against. For example, in the α-synuclein ep-PCR, 6×10^{10} initial template molecules were used, and the final library contained $<6\times10^6$ clones (the bottleneck being ligation and transformation). Under these conditions, all sampled sequences can be considered to be independent, and the equations above can be used repeatedly in order to simulate all of the sequences in the library. For this discussion to hold in mutator strain mutagenesis, an appropriate bottleneck would need to be artificially imposed. Alternatively, a brute force realistic simulation of the entire branching process may be feasible, but only if the number of initial templates were very low, and the number of generations not too large [also note that some work with finite population models has been reported (27,30)].

Appendix B: an estimator of the DNA polymerase incorporation frequencies

Our goal is to computationally produce a set of PCR products with a distribution of mutated sequences as near as possible to those in an experimental PCR library of interest. However, our only insight into that experimental library is limited DNA sequencing information, estimates of n, λ, and library size, and the model presented above. To be able to generate sequences computationally, using the method of Appendix A, we must first derive an estimate of $\mathrm{E}(\mathbf{N})_{\mathrm{real}}$, the true state of nature. We use the method of moments to derive an estimator.

The expectation of a sequence drawn at random from the library is given by:

$$\mathrm{E}(\mathbf{Z}) = \sum_{\substack{\text{All possible} \\ \text{sequences } i}}\mathbf{Z}_i\mathrm{P}(\mathbf{Z}_i),$$

$$\mathrm{E}(\mathbf{Z}) = \sum_{k=0}^{n}\sum_{r=0}^{1}\left[\left(\sum_{\substack{\text{All possible} \\ \text{sequences } i}}\mathbf{Z}_i\mathrm{P}(\mathbf{Z}_i|K=k,R=r)\right)\mathrm{P}(K=k)\mathrm{P}(R=r)\right],$$

$$\mathrm{E}(\mathbf{Z}) = \sum_{k=0}^{n}\sum_{r=0}^{1}(\mathrm{E}(\mathbf{Z}|K=k,R=r)\mathrm{P}(K=k)\mathrm{P}(R=r)),$$

$$\mathrm{E}(\mathbf{Z}) = \sum_{k=0}^{n}\left(\frac{\mathbf{Z}_{00}}{2}\left(\mathrm{E}(\mathbf{N})^k\mathbf{T}^k + \mathbf{T}\mathrm{E}(\mathbf{N})^k\mathbf{T}^k\mathbf{T}\right)\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right).$$

Note that E(**Z**), unlike **Z**, may have non-zero elements in every position, corresponding to the expected level of that mutation type at that position. Additionally, complementary mutations (e.g. A→C and T→G) are expected to occur in the library with equal probability. This is intuitively clear from the double-stranded complementary nature of DNA, and is embodied in the equations by the centrosymmetry of $\mathrm{E}(\mathbf{N})^k\mathbf{T}^k + \mathbf{T}\mathrm{E}(\mathbf{N})^k\mathbf{T}^k\mathbf{T}$.

In our model, all bases of a single type in a single sequence are equivalent. We may, therefore, contract all $L \times 4$ matrices to 4×4 matrices, by setting each row in the 4×4 matrix (there is one for each, A,G,C,T) equal to the sum of all rows in the $L \times 4$ matrix that originate with that wild-type base. Mathematically, this is equivalent to multiplying the matrix on the left with the transpose of matrix $\mathbf{Z}_{00}$.

$$\mathrm{E}(\mathbf{Z}_{00}^T\mathbf{Z}) = \sum_{k=0}^{n}\left(\frac{\mathbf{Z}_{00}^T\mathbf{Z}_{00}}{2}\left(\mathrm{E}(\mathbf{N})^k\mathbf{T}^k + \mathbf{T}\mathrm{E}(\mathbf{N})^k\mathbf{T}^k\mathbf{T}\right)\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right).$$

Our experimental estimate of this quantity, based on DNA sequencing of s sequences, will be a matrix with element ij equal to:

$$\frac{(b+d)}{s}\,\frac{f}{(f+h)}$$

where b = number of observed $i \rightarrow j$ changes in s sequences
d = number of observed $\bar{i} \rightarrow \bar{j}$ changes in s sequences
f = number of i bases in one wild type sequence
h = number of $\bar{i}$ bases in one wild type sequence

Numbers of bases in the above are to be taken from a single arbitrarily chosen DNA strand. A barred variable indicates Watson–Crick complement, $\bar{\mathrm{A}} = \mathrm{T}$, $\bar{\mathrm{G}} = \mathrm{C}$, $\bar{\mathrm{C}} = \mathrm{G}$, $\bar{\mathrm{T}} = \mathrm{A}$. The term $b + d$ divided by s gives our estimate of the average number of mutations of this pair type per sequence and the second term takes the appropriate fraction of this value, based on wild-type sequence composition. This formula also removes any differences in frequency between the two complementary members of a mutation pair, since these are due solely to random sampling, as discussed above.

To formulate the estimate in matrix terms, we begin with matrix **X** of the sequencing data. Specifically, from $s = 89$ sequences of the α-synuclein library (Table 1):

$$\mathbf{X} = \begin{bmatrix} 11233 & 64 & 19 & 73 \\ 20 & 11979 & 2 & 12 \\ 8 & 2 & 6296 & 13 \\ 44 & 8 & 24 & 5975 \end{bmatrix}.$$

Then the experimental estimate equals,

$$\frac{1}{s}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\left(\mathbf{Z}_{00}^T\mathbf{Z}_{00} + \mathbf{T}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\mathbf{T}\right)^{-1}(\mathbf{X} + \mathbf{T}\mathbf{X}\mathbf{T}).$$

Setting the experimental estimate equal to the expected value given above, and canceling terms:

$$\frac{1}{s}\left(\mathbf{Z}_{00}^T\mathbf{Z}_{00} + \mathbf{T}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\mathbf{T}\right)^{-1}(\mathbf{X} + \mathbf{T}\mathbf{X}\mathbf{T}) = \sum_{k=0}^{n}\left(\frac{\left(\mathrm{E}(\mathbf{N})^k\mathbf{T}^k + \mathbf{T}\mathrm{E}(\mathbf{N})^k\mathbf{T}^k\mathbf{T}\right)}{2}\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right).$$

There is not a unique solution to this equation. We choose to solve for an E(**N**) that is centrosymmetric, designated $\mathrm{E}(\mathbf{N})_{\mathrm{cen}}$. This reduction in degrees of freedom is motivated by the analysis in Supplementary Appendix F, and allows us to proceed algebraically towards a unique solution. The resulting estimator is statistically analyzed in Supplementary Appendix G. The equation above simplifies to,

$$\frac{1}{s}\left(\mathbf{Z}_{00}^T\mathbf{Z}_{00} + \mathbf{T}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\mathbf{T}\right)^{-1}(\mathbf{X} + \mathbf{T}\mathbf{X}\mathbf{T}) = \sum_{k=0}^{n}\left(\mathrm{E}(\mathbf{N})_{\mathrm{cen}}^k\mathbf{T}^k\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right).$$

Furthermore, $\mathrm{E}(\mathbf{N})_{\mathrm{cen}} = \mathbf{T}\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}$, and therefore $\mathrm{E}(\mathbf{N})_{\mathrm{cen}}{}^k\mathbf{T}^k = [\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}]^k$.

$$\frac{1}{s}\left(\mathbf{Z}_{00}^T\mathbf{Z}_{00} + \mathbf{T}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\mathbf{T}\right)^{-1}(\mathbf{X} + \mathbf{T}\mathbf{X}\mathbf{T}) = \sum_{k=0}^{n}\left(\left(\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}\right)^k\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right)$$

Assume that the left hand side of the above equation is real-diagonalizable (to matrix $\mathbf{D}_y$) by a matrix **C**. We multiply both sides on the left by $\mathbf{C}^{-1}$, and both sides on the right by **C**.

$$\mathbf{D}_y = \mathbf{C}^{-1}\left(\frac{1}{s}\left(\mathbf{Z}_{00}^T\mathbf{Z}_{00} + \mathbf{T}\mathbf{Z}_{00}^T\mathbf{Z}_{00}\mathbf{T}\right)^{-1}(\mathbf{X} + \mathbf{T}\mathbf{X}\mathbf{T})\right)\mathbf{C} = \sum_{k=0}^{n}\left(\mathbf{C}^{-1}\left(\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}\right)^k\mathbf{C}\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right)$$

A matrix shares identical eigenvectors with all of its powers. Following from this, the overall eigenvectors of a power series of a matrix must equal the eigenvectors of its terms. Therefore, the matrix **C** must diagonalize $\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}$ to matrix $\mathbf{D}_q = \mathbf{C}^{-1}\,\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}\mathbf{C}$, and,

$$\mathbf{D}_y = \sum_{k=0}^{n}\left(\mathbf{D}_q^k\binom{n}{k}\frac{\lambda^k}{(1+\lambda)^n}\right).$$

Each diagonal entry of $\mathbf{D}_q$ can be found independently by solving an nth-degree polynomial in the field of real numbers. There may be multiple real solutions for each entry in $\mathbf{D}_q$. In our data, real solutions existed for each entry at $\sim$1 and -3. To choose the appropriate values, we note that $\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}$ and $\mathbf{D}_q$ are similar matrices, and therefore share identical eigenvalues. Because $\mathrm{E}(\mathbf{N})_{\mathrm{cen}}\mathbf{T}$ will always be nearly the identity matrix in ep-PCR, its eigenvalues will all be near unity and, therefore, the appropriate solutions are close to 1.

Finally, the unique physically sensible solution for matrix $E(\mathbf{N})_{cen}$ can be found using the relationship,

$$E(\mathbf{N})_{cen} = \mathbf{C}\mathbf{D}_q\mathbf{C}^{-1}\mathbf{T}.$$

The quality of $E(\mathbf{N})_{cen}$ so derived, in terms of the accuracy and precision of the resulting estimated library properties, is analyzed in Supplementary Appendix G and summarized in the confidence intervals (see Supplementary Appendix C) of Table 2. A Mathematica notebook is available from the authors as a template for computation of the matrix estimator. As a technical point, if any of the elements of $\mathbf{X}$ is zero, the solution $E(\mathbf{N})_{cen}$ may contain very small negative values. This reflects the discreteness and limited size of our actual observations. Such negative values should be changed to the most acceptable substitute, zero, before proceeding. When the experimental library excludes the template sequences (see above), the summation in this set of equations should begin at $k = 1$, and the right hand side should be divided by $1 - P(K = 0)$, to normalize the summation of the probabilities $P(K = k)$ to a total of one.

Note the close approximation between the preliminary incorporation frequencies obtained with the Sun estimator under hypothesis II (Supplementary Appendix F) and those in $E(\mathbf{N})_{cen}$ (Table 3). The reason that these initial estimates are so good is that the mutation rate during ep-PCR is low enough that multiple mutations at a single site are very rare. We use the terms 'low' and 'high' in this respect, throughout this article. It has previously been implied that the error rate of ep-PCR is quite high in this respect ['$\sim$0.7%' per base per PCR cycle (26), citing Cadwell and Joyce (2)]. However, the rate in the Cadwell and Joyce paper is over the entire reaction, and is not given per cycle: 'we used this method to mutagenize the gene... with a mutation rate of 0.66% $\pm$ 0.13% (95% C.I.) per position per PCR' (2).