Analysis of Universal Protein Binding Microarrays **Sept. 11, 2017**

This document is meant to serve as a practical guide for the analysis of universal PBMs synthesized by Agilent Technologies. Variants of the methods described here are acceptable for those familiar with microarray data analysis; these can be viewed mainly as general guidelines for the extraction of accurate, high-quality binding data from universal PBMs. We have supplied all programs we have written for data normalization and sequence analysis, as well as demo files for the user. Detailed notes and instructions for running these programs are contained in the following pages.

The experimental details are provided elsewhere. Briefly, single-stranded microarrays are double-stranded by a solid-phase primer extension reaction involving the incorporation of small quantities of Cy3-conjugated dUTP. These newly double-stranded arrays are bound by GST-tagged, sequence-specific DNA-binding proteins and labeled with Alexa488-conjugated anti-GST antibody. The relative amounts of protein bound to each particular probe sequence are reflected by the Alexa488 fluorescence intensity at each spot. We have designed our probe sequences utilizing a combinatorial strategy in which all *k*-mers are represented and equal number of times. Currently, we typically use three different microarray formats: 4x44K (four identical chambers of ~44,000 probes each), 8x15K (eight identical chambers of ~15,000 probes each), and 8x60K (eight identical chambers of ~60,000 probes each). The 4x44K and 8x60K arrays were designed to cover all possible 10-mer sequence variants, and the 8x15K arrays were designed to cover all possible 9-mer sequence variants. These multi-chambered formats enable multiplexing of experiments in order to examine multiple proteins or different binding conditions. Designs of each format are available from Agilent, classified according to their AMADID number: 051681 (4x44k_v1), 016060 (4x44k_v2), 015803 (8x15k_v1), 016236 (8x15k_v2), and 30265 (8x60k_v1).

The programs described here are generally applicable for all designs in the 4x44K, 8x15K, and 8x60K array formats. The main goal of the analysis is to transform the probe signal intensity measurements to a comprehensive list of scores reflecting the relative affinities for all *k*-mers. (Each probe sequence contains several overlapping *k*-mers, and each *k*-mer occurs on multiple probes.) We take advantage of the fact that every 8-mer occurs on at least 32 different probes for the 4x44K and 8x60K ‘all 10-mer’ arrays and that every 7-mer occurs on at least 40 different probes for the 8x15K ‘all 9-mer’ arrays. (Gapped *k*-mers are also represented with the same frequency as ungapped *k*-mers.) Consequently, we can derive robust *k*-mer scores based on a large ensemble of probe intensity measurements for 8-mers and 7-mers on 4x44K/8x60K and 8x15K arrays, respectively. Additionally, we can compactly represent these binding data in a more intuitive (though less informative) position weight matrix, or PWM.

Scanning Microarrays

In our PBM experiments, Cy3-labeled dUTP is used to monitor the relative concentration of DNA at each spot, and Alexa488-labeled anti-GST is used to monitor the amount of protein bound to each individual probe sequence. PBMs should be scanned with the appropriate lasers (Cy3 excitation 543 nm, emission 570 nm; Alexa488 excitation 488 nm, emission 522 nm) at a range of laser power settings in order to capture the full dynamic range of intensities. A single Cy3 scan is usually sufficient to obtain high-intensity signal measurements for all probes without any probes reaching saturation levels. The larger range in Alexa488 signal intensity (due to large variation in the amount of protein bound to different probes) typically requires 2-3 scans at different laser power settings to ensure reliable measurements of both the dimmest and the brightest spots. The lowest intensity scan should have no saturated signal intensities. Scanned images should exhibit at least 5-micron resolution and be saved as TIFF files.

Example:

```
DNAscan_Cy3_lp80pg80.TIF
PBMscan_Alexa488_lp100pg100.TIF
PBMscan_Alexa488_lp90pg90.TIF
PBMscan_Alexa488_lp80pg80.TIF
```

(In the above filenames, lp refers to laser power and pg refers to PMT gain for the detector.)

Image Analysis with GenePix Pro v6.0

We use GenePix Pro v6.0 software (Molecular Devices) to analyze the microarray spot intensities of our image files. GenePix Pro requires both a TIF image file and a GAL (GenePix Array List) or GPS (GenePix Settings) file that contains information regarding the coordinates and identities of all spots within a subgrid. The output is a large table in a GPR (GenePix results) file. When opening a scan, specify that the image is 5-micron resolution. Additionally, when prompted to enter the wavelength of the scan, use "1" for both Alexa488 and Cy3. (This value makes no difference for the image analysis, but it appears in the headings of the output GPR file. Both Masliner (below) and our program for normalizing PBM signal search for "1" as the default wavelength.) Then, open the GAL or GPS file. Align each block of spots over the corresponding subgrids, and then allow GenePix to automatically fine-tune the alignment. Manually flag problematic spots as "bad" (i.e., spots with scratches, dust flecks, etc.); this assigns a flag of "-100" to such spots. These adjustments can be saved in a new GPS file. After all subgrids have been aligned and flagged, the scan can be analyzed to produce a large table with various quantitative measures for each spot. This table can be saved as a GPR file (specify that each subgrid should be saved as a separate file). Each scan at each intensity should be analyzed and saved separately, although the same GPS file may be used for multiple intensity scans of the same experiment.

A GAL file is provided by Agilent with each microarray. As an alternative, we have provided four GPS files corresponding to the five designs we commonly use (v1 and v2 for 4x44k, v1 and v2 for 8x15K, and v1 for 8x60K).

Example: If the microarray contains four subgrids (4x44K) and the option to save each subgrid in a separate file is specified, the following files will be written automatically.

```
DNAscan_Cy3_lp80pg80_1-4.GPR
DNAscan_Cy3_lp80pg80_2-4.GPR
DNAscan_Cy3_lp80pg80_3-4.GPR
DNAscan_Cy3_lp80pg80_4-4.GPR

PBMscan_Alexa488_lp100pg100_1-4.GPR
PBMscan_Alexa488_lp100pg100_2-4.GPR
PBMscan_Alexa488_lp100pg100_3-4.GPR
PBMscan_Alexa488_lp100pg100_4-4.GPR

PBMscan_Alexa488_lp90pg90_1-4.GPR
PBMscan_Alexa488_lp90pg90_2-4.GPR
PBMscan_Alexa488_lp90pg90_3-4.GPR
PBMscan_Alexa488_lp90pg90_4-4.GPR

PBMscan_Alexa488_lp80pg80_1-4.GPR
PBMscan_Alexa488_lp80pg80_2-4.GPR
PBMscan_Alexa488_lp80pg80_3-4.GPR
PBMscan_Alexa488_lp80pg80_4-4.GPR
```

Combining Multiple Scans of Different Intensities with Masliner

Masliner is a Perl script written by members of the Church lab (Dudley, Aach, Steffen, Church. *PNAS*, 2002.) in order to combine multiple scans of different intensities from the same microarray. High intensity scans are often necessary to capture reliable above-background measurements of probes with weaker signals but may result in many bright spots exceeding saturation levels. Lower intensity scans are consequently required to resolve the relative intensity differences among these brightest spots. For a pair of scans, Masliner performs a linear regression using the spots that fall within the linear range of the scanner in both scans in order to extrapolate the signal intensities of any saturated spots in the brighter scan. As output, Masliner returns the GPR file of the brighter scan, with additional columns appended to the end including the adjusted background-subtracted intensity (ADJBSI) of each probe.

Masliner takes two scans as input. To combine a series of more than two scans, Masliner must be run multiple times, first taking the two lowest intensity scans as input, then taking the output of that iteration with the next lowest intensity scan, etc. Masliner must be run in an environment that can interpret the Perl programming language. The Masliner program can be downloaded here:
http://arep.med.harvard.edu/masliner/supplement.htm

Example: To combine the PBM Alexa488 scans for chamber 1, multiple iterations of Masliner are required.

```
perl masliner.pl -g1 PBMscan_Alexa488_lp80pg80_1-4.GPR -g2
PBMscan_Alexa488_lp90pg90_1-4.GPR -o output1.GPR -ll 200 -lh 40000 -f1
1 -f2 1

perl masliner.pl -g1 output1.GPR -g2 PBMscan_Alexa488_lp100pg100_1-
4.GPR -o PBMscan_Alexa488_MaslinerOutput_1-4.GPR -ll 200 -lh 40000 -f1
1 -f2 1
```

This creates a single file: `PBMscan_Alexa488_MaslinerOutput_1-4.GPR`.

(In the command line, –g1 and –g2 denote the input GPR files, –o denotes the output file, –ll and –lh denote the lower and upper bounds of the linear range of the scanner, and –f1 and –f2 denote the "wavelength" printed in the GPR file headings, as described above.)

Normalizing and Spatially Detrending PBM Signal Intensities

Our Perl script `normalize_agilent_array.pl` performs two important functions: (1) normalize the protein binding signal by the relative concentration of DNA at each spot, and (2) adjust the normalized signal to account for spatial non-uniformities across the microarray. Currently, it can operate for 4x44K, 8x15K, or 8x60K Agilent microarray formats. It is meant to be run on one single subgrid at a time, but the user can write a shell script to sequentially analyze all subgrids in the microarray. To run properly, the script requires two additional files to be stored in the same directory: `analyze_agilent.pm` and `Regression.pm`, provided here.

The program takes three files as input: (1) the GPR file (raw or Masliner-adjusted) for the Alexa488 PBM scan(s), (2) the GPR file (raw or Masliner-adjusted) for the Cy3 DNA scan(s) [optional], and (3) a file containing the probe ID's, sequences, and coordinates. This file is specific to each individual array design; we provide sequence files for each format: `4x44k_v1_sequences.txt`, `4x44k_v2_sequences.txt`, `8x15k_v1_sequences.txt`, `8x15k_v2_sequences.txt`, and `8x60k_v1_sequences.txt`. Each file contains five columns: Column / Row / Probe Name / Probe ID / Probe Sequence. This file must be in UNIX format. Probe ID's for the combinatorial spots (collectively containing all *k*-mers) must begin with "dBr", indicating that they were designed from a de Bruijn sequence. Additional probes include Agilent-specific proprietary control sequences and our own control sequences that we designed for a variety of purposes. Based on the number of probes in the file, the program determines whether the microarray is a 4x44K, 8x15K, 8x60K, or an indeterminate format.

The program takes <5 minutes to run on a 44K or 60K subgrid and <1 minute to run on a 15K subgrid. The Cy3 normalization step requires GPR files from both the Alexa488 PBM scan(s) and the Cy3 DNA scan(s). It will automatically read in ADJBSI values from Masliner-adjusted GPR files; if those do not exist because Masliner was not run, values from the column labeled "F1 Median – B1" will be used. As stated above, if the wavelength was entered as a value other than "1" when opening the scan in GenePix, this column will have an alternate name (e.g. "F488 Median – B488"). In this case, rather than starting GenePix from the beginning, the code can be manually changed at lines 96 and 97 to read in alternate GPR files. Spots flagged as bad in GenePix ("-100") will be eliminated from the analysis. Cy3 normalization begins by performing a linear regression over all combinatorial probes to calculate coefficients representing the contributions to the total Cy3 signal of each trinucleotide. This is necessary because we have found the incorporation of Cy3-dUTP to be dependent on the local sequence context of each adenine in the template strand. Once these coefficients have been determined, the expected Cy3 signal of each probe is calculated, and the Alexa488 PBM signal is normalized by the ratio of the observed-to-expected Cy3 signal. (Probes with observed Cy3 signals two-fold smaller or larger than expected are removed from further consideration.) The quality of the fit of the regression (R^2) reflects the fidelity of the primer extension reaction. Typically, we observe $R^2 \geq 0.65$. The spatial detrending step then uses these normalized Alexa488 intensities to compute the median intensity for the "local neighborhood" of each spot (15 x 15 block centered on each spot) and divides signal at each spot by the ratio of the local median to the global (whole-array) median. The local neighborhood of any spot near the corners and margins of the grid is adjusted so that it is the same size as that for any interior spot. If a Cy3 GPR file is not entered, the program will skip the Cy3 normalization step and proceed directly to spatial detrending.

The program outputs four files: (1) a "raw data" file, containing a summary of the un-normalized probe intensities, flags, and sequences extracted from the GPR files; (2) an "all data" file, containing all the information above plus the calculated values for expected Cy3, observed/expected Cy3, Cy3-normalized Alexa488, and spatially-detrended Alexa488; (3) a "combinatorial" file, containing a ranked list of the normalized intensities and sequences of all combinatorial 'all *k*-mer' probes (with control spots removed); and (4) a "regression" file, containing the coefficients determined from the linear regression over Cy3 probe intensities and sequences, including the R^2 value indicating the quality of the fit. The most important of these is the "combinatorial" file, which can be input directly to our sequence analysis program (described below) or used by other motif finding algorithms. The other output files are mainly intended for users more familiar with microarray analysis who wish to apply additional normalization or filtering steps beyond those implemented here.

`normalize_agilent_array.pl` must be run in an environment that can interpret the Perl programming language. Running the program without further command line inputs will reveal syntax requirements and parameter options.

Example: This command will initiate the normalization of the Masliner-adjusted PBM GPR file by the Cy3 raw GPR file from the corresponding chamber of the microarray, spatially adjust the normalized intensities, and output four files beginning with the prefix "proteinA". The sequence file must correspond to the microarray design that was used for this PBM experiment. (The Cy3 GPR file can be omitted, in which case only the spatial detrending operation will be performed.)

```
perl normalize_agilent_array.pl -i
PBMscan_Alexa488_MaslinerOutput_1-4.GPR -c
DNAscan_Cy3_lp80pg80_1-4.GPR -s 4x44k_v1_sequences.txt -o
proteinA
```

Output files:

```
proteinA_rawdata.txt
proteinA_alldata.txt
proteinA_combinatorial.txt
proteinA_regression.txt
```

Sequence analysis

Our Perl script `seed_and_wobble.pl` performs two functions: (1) calculates scores for all individual *k*-mers (including enrichment scores, median signal intensities, and Z-scores), and (2) constructs mononucleotide position weight matrices (PWMs) reflecting the relative preference of each nucleotide at each position within the binding site motif. The main requirement is a two-column ranked list of probe sequences and normalized intensities. This program is meant only for the analysis of data from universal PBM experiments (where all sequence variants of a given length are represented an equal number of times) and is not a general-purpose motif finder for other data sets. To run properly, the script requires the additional file `seed_and_wobble_modules.pm`, provided here, to be stored in the same directory.

The name "Seed-and-Wobble" describes our algorithm for PWM construction, in which the *k*-mer with the greatest enrichment score (over all possible contiguous and gapped *k*-mers covered an equal number of times on the array) is chosen as a "seed", and the relative preference of each nucleotide variant is calculated by "wobbling" each position within and outside of the seed. In the process of choosing candidate seeds, a comprehensive look-up table containing scores for each *k*-mer is created. Often, this may be all that is desired, as it provides a much richer source of information than a condensed mononucleotide PWM. For instance, by comprehensively calculating scores for all *k*-mers, information regarding nucleotide interdependencies (and also high and low affinity classes of binding sites) is retained. Our *k*-mer scoring methods and Seed-and-Wobble algorithm are described in detail in the Supplementary Methods of Berger *et al*., *Nature Biotechnology* (2006) and will not be duplicated here. In short, the median intensity scores and Z-scores are useful because they retain information regarding relative differences in signal intensity, and thus probe occupancy and relative affinity as well. However, we have found our separate rank-based, non-parametric enrichment scores (E-scores) to be preferable because they are invariant to differences in protein concentration, they facilitate the comparison of all TFs on the same scale, and they importantly enable the data from replicate experiments on separate microarray designs to be combined. E-scores range from -0.5 (lowest enrichment) to +0.5 (highest enrichment). Our method for PWM construction uses these E-scores and takes advantage of the fact that all sequence variants occur an equal number of times on the microarray, and it considers all features without applying any arbitrary cutoffs.

One important point to understand in scoring *k*-mers and constructing PWMs is that both contiguous and gapped *k*-mers are covered an equal number of times. Our 4x44K and 8x60K 'all 10-mer' microarrays were designed to ensure that all possible contiguous 8-mers and gapped 8-mers up to 12 total positions (with one exception) occur on at least 16 different probes (32 probes when reverse complements are considered). Thus, we are able to reliably estimate the relative preference of a TF for 22.3 million gapped and contiguous 8-mers (4^8 sequence variants of 341 patterns up to 8-of-12) based on a large ensemble of probe intensity measurements. (All 8-mers of the pattern 4-gap-4, with gaps of up to 20 positions, are also uniformly covered.) Our 8x15K 'all 9-mer' microarrays are similar in that all contiguous 7-mers and gapped 7-mers up to 12 total positions are also uniformly covered. This coverage of gapped *k*-mers of all different patterns enables us to consider them as candidate "seeds" and also to systematically test all nucleotide variants at positions outside the chosen seed during the "wobble" step. Further, by considering different gapped patterns as candidate seeds, it is possible to construct long PWMs anchored on the most informative positions within the motif. Due to runtime and memory considerations, it is not always practical to score every sequence variant of every single pattern covered on the array. For instance, even though all 8-of-12-mers are covered in our 4x44K and 8x60K arrays, we typically exhaustively score only 8-of-10-mers. However, once the initial seed is chosen, the complete list of covered patterns are used to determine which additional flanking positions can be added to the final PWM.

`Seed_and_wobble.pl` is meant to be run for a single experiment on a single microarray design. (Combining data from multiple microarray designs is discussed below.) It requires three input files. The first is a two-column ranked list of the normalized intensities and sequences for all combinatorial 'all *k*-mer' probes (*i.e.*, the "_combinatorial" file generated in the normalization procedure). The second is a "query file" listing all contiguous and gapped patterns to be considered as candidate seeds. As stated above, we often use the set of all 8-mer patterns up to 10 total positions for our 4x44K and 8x60K arrays. Each one is entered as a series of "1"s and "."s (e.g., 11111111, 1.1111111, 11.111111, etc.). These are the patterns for which *k*-mer scores will be exhaustively calculated and printed. The third is a "total file" listing all contiguous and gapped patterns covered in the array design, used to determine which flanking positions can be tested during the PWM construction stage. The syntax is the same as the query file. All entries in the query file must also be represented in the total file. Because *k*-mers and reverse complements are considered together, the reverse of a pattern already included in the file does *not* need to be listed as well. (For instance, 1111111.1 can be left out if 1.1111111 is already present.) To generate scores for only contiguous 8-mers, the query files should contain only a single line: 11111111 . Sample query files and total files are provided for our 4x44K and 8x60K 'all 10-mer' format and our 8x15K 'all 9-mer' format: `patterns_8of10.txt`, `patterns_8of12.txt`, `patterns_7of9.txt`, `patterns_7of12.txt`. Also, the files `patterns_4x44k_all_8mer.txt`/`patterns_8x60k_all_8mer.txt` and `patterns_8x15k_all_7mer.txt` contain lists of all the patterns covered (with 16-fold redundancy) on the 4x44K/8x60K (same files, different names for consistency's sake) and 8x15K arrays, respectively.

The program generates several output files. (Since these files can be very large, there are built-in options to limit what is printed, discussed below.) First, for each pattern in the query file, a separate file is generated containing the enrichment scores, median intensities, and Z-scores of all 4^k sequence variants. Second, a single integrated file is created for all *k*-mers with an E-score above a specified cutoff. Third, a file is created with PWMs generated from the top *N* seeds, each listed in various formats. (There are several ways to create graphical logos of these PWMs. The energy-based matrix can be recognized by enoLOGOS (http://chianti.ucsd.edu/cgi-bin/enologos/enologos.cgi), and the probability-based matrix can be made into a graphical logo by a variety of web-based programs.)

In addition to the three input files, the command line requires two additional parameters: the width of the sequences to be examined (*k*) and the prefix for all output files. For example, to run Seed-and-Wobble on the normalized data described above (from a 4x44K array), the following command would be used:

```
perl seed_and_wobble.pl proteinA_combinatorial.txt 8
patterns_8of10.txt patterns_4x44k_all_8mer.txt proteinA
```

The following output files would be generated:

```
proteinA_8mers_11111111.txt
proteinA_8mers_1.1111111.txt
proteinA_8mers_11.111111.txt
proteinA_8mers_111.11111.txt
proteinA_8mers_1111.1111.txt
proteinA_8mers_1..1111111.txt
proteinA_8mers_1.1.111111.txt
proteinA_8mers_1.11.11111.txt
proteinA_8mers_1.111.1111.txt
proteinA_8mers_1.1111.111.txt
proteinA_8mers_1.11111.11.txt
proteinA_8mers_1.111111.1.txt
proteinA_8mers_11..111111.txt
proteinA_8mers_11.1.11111.txt
proteinA_8mers_11.11.1111.txt
proteinA_8mers_11.111.111.txt
proteinA_8mers_11.1111.11.txt
proteinA_8mers_111..11111.txt
proteinA_8mers_111.1.1111.txt
proteinA_8mers_111.11.111.txt
proteinA_8mers_1111..1111.txt
proteinA_8mers_top_enrichment.txt
proteinA_8mers_pwm.txt
```

Several parameters influencing the number and size of the output files can be manually adjusted by editing the code. These parameters are clearly marked at the beginning of the file.

$Escore_cutoff [default 0.25]: This value determines the size of the integrated "top_enrichment" file by setting the E-score cutoff for *k*-mers to be included. To include all k-mers in a single file, this can be set to -0.5.

$print_seeds [default "yes"]: If "yes", a separate file is created for each of the patterns in the query file. If "no", only a single file for contiguous *k*-mers is created. (The top_enrichment file will still contain gapped k-mers with high E-scores.)

$print_top_seeds [default "yes"]: If "yes", the "top_enrichment" file will be created. In "no", it will not.

$topN [default 3]: This value sets the number of independent PWMs constructed. By default, separate PWMs will be generated for the top 3 *k*-mers with the highest E-scores.

When two PBM experiments have been performed, the combined *k*-mer scores and a single combined PWM can be determined using the Perl script `seed_and_wobble_twoarray.pl`. (The two experiments can come from the same or different array designs, as long they are the same format, *i.e.* 4x44K, 8x15K, or 8x60K.) This program also requires the file `seed_and_wobble_modules.pm` to be stored in the same directory. It can be run almost exactly as the single-array version described above, but it requires a second ranked list of normalized intensities and probe sequences as input. One other difference is that *k*-mer median intensities and Z-scores cannot always be directly combined from different experiments, so the only scores reported here are E-scores (which we devised specifically so that we could combine multiple experiments from different array designs).

For normalized data from array v1 and array v2, the command line would be as follows:

```
perl seed_and_wobble_twoarray.pl proteinA_v1_combinatorial.txt
proteinB_v2_combinatorial.txt 8 patterns_8of10.txt
patterns_4x44k_all_8mer.txt proteinA
```

The following output files would be generated:

```
proteinA_8mers_11111111_combined.txt
proteinA_8mers_1.1111111_combined.txt
proteinA_8mers_11.111111_combined.txt
proteinA_8mers_111.11111_combined.txt
proteinA_8mers_1111.1111_combined.txt
proteinA_8mers_1..1111111_combined.txt
proteinA_8mers_1.1.111111_combined.txt
proteinA_8mers_1.11.11111_combined.txt
proteinA_8mers_1.111.1111_combined.txt
proteinA_8mers_1.1111.111_combined.txt
proteinA_8mers_1.11111.11_combined.txt
proteinA_8mers_1.111111.1_combined.txt
proteinA_8mers_11..111111_combined.txt
proteinA_8mers_11.1.11111_combined.txt
proteinA_8mers_11.11.1111_combined.txt
proteinA_8mers_11.111.111_combined.txt
proteinA_8mers_11.1111.11_combined.txt
proteinA_8mers_111..11111_combined.txt
proteinA_8mers_111.1.1111_combined.txt
proteinA_8mers_111.11.111_combined.txt
proteinA_8mers_1111..1111_combined.txt
proteinA_8mers_top_enrichment_combined.txt
```

`proteinA_8mers_pwm_combined.txt`

The same parameters as described above exist to curtail the number and size of the output files and can be manually adjusted by editing the code.

Additional Programs

It may occasionally be desirable to generate a PWM from a manually chosen "seed" besides the *k*-mer with the greatest enrichment score. This alternate seed could be chosen to better represent low to moderate affinity binding sites or to capture an additional component of a transcription factor's binding specificity. We provide two Perl scripts: `wobble_single_seed.pl` and `wobble_single_seed_twoarray.pl`.

`wobble_single_seed.pl`, like `seed_and_wobble.pl`, takes as input a "combinatorial" file containing a ranked list of intensities and sequences for the combinatorial 'all k-mer' probes, and a "total" file listing all contiguous and gapped patterns covered in the array design. But instead of inputting a "query" file with a list of patterns to test, this program takes a single *k*-mer chosen by the user (*e.g.*, TCA.G.GTTG). `wobble_single_seed_twoarray.pl` runs the same way except it takes two "combinatorial" files as input, corresponding to the same protein tested on two different array designs (analogous to `seed_and_wobble_twoarray.pl`). Both programs also require the file `seed_and_wobble_modules.pm` to be contained within the same directory. Running the program without further command line inputs will display syntax requirements.

Example:
```
perl wobble_single_seed.pl proteinA_combinatorial.txt TCA.G.GTTG
patterns_4x44k_all_8mer.txt > proteinA_pwm.txt

perl wobble_single_seed_twoarray.pl proteinA_v1_combinatorial.txt
proteinA_v2_combinatorial.txt TCA.G.GTTG
patterns_4x44k_all_8mer.txt > proteinA_pwm.txt
```

Finally, we have developed a method for systematically searching for multiple PWMs from a single universal PBM experiment. This is especially useful in cases where the entire PBM signal cannot adequately be explained by a single PWM model. Briefly, this procedure entails (1) running `seed_and_wobble.pl` on the list of probe sequences ranked by intensity, (2) using the discovered "primary" PWM to calculate the expected occupancy of each probe, (3) re-ranking the probes according to the ratios of their observed and expected ranks, (4) running `seed_and_wobble.pl` on the re-ranked list of probes sequences to discover a "secondary" PWM. This procedure can be iterated several times to search for multiple PWMs.

The re-ranking can be achieved using the program `rerank.pl`. This program takes as input the "combinatorial" file of probe sequences and intensities (which was input to `seed_and_wobble.pl`) and the "pwm" file (which was output by `seed_and_wobble.pl`). It outputs a re-ranked list of probe sequences and intensities that can be directly used as input in a second iteration of `seed_and_wobble.pl`.

Example:

```
perl rerank.pl proteinA_combinatorial.txt proteinA_pwm.txt
proteinA_reranked.txt
```