Diversity – solution

Adrian Beker

September 2, 2021

1 The solution in the case of a single query

To begin with, let's consider the case $Q = 1$, i.e. the case of a single query. We will first determine how to efficiently compute the total diversity of a given sequence, which will be denoted by S. If we denote by K the diversity of the sequence, we may without loss of generality assume that the sequence consists of the numbers $1, 2, \ldots, K$ (by performing coordinate compression). A *gap* of a value $i \in \{1, 2, \ldots, K\}$ is a maximal (with respect to inclusion) contiguous subsequence not containing i. Let i have r_i gaps and let $\ell_{i,1}, \ldots, \ell_{i,r_i}$ be their lengths, in the order from left to right. Then it is easy to see that i occurs on a total of

$$\frac{N(N+1)}{2} - \sum_{j=1}^{r_i} \frac{\ell_{i,j}(\ell_{i,j}+1)}{2}$$

contiguous subsequences. Indeed, each contiguous subsequence not containing i is contained in a unique gap. Hence, we have (using linearity of expectation)

$$S = \sum_{i=1}^{K} \left(\frac{N(N+1)}{2} - \sum_{j=1}^{r_i} \frac{\ell_{i,j}(\ell_{i,j}+1)}{2} \right).$$

Now note that $\sum_{j=1}^{r_i} \ell_{i,j} = N - c_i$, where c_i is the number of occurrences of the value i in the sequence, so we have

$$\sum_{i=1}^{K} \sum_{j=1}^{r_i} \ell_{i,j} = \sum_{i=1}^{K} (N - c_i) = KN - \sum_{i=1}^{K} c_i = KN - N = N(K-1).$$

Hence, the above expression for S can be simplified as follows:

$$S = \frac{1}{2} \left(KN(N+1) - N(K-1) - \sum_{i=1}^{K} \sum_{j=1}^{r_i} \ell_{i,j}^2 \right).$$

We conclude that the problem reduces to the following: permute a given sequence so that the sum of the squares of the lengths of all gaps, call it T, is as large as possible. In order to solve this problem, we make the following observation:

Claim 1. In every optimal solution, the occurrences of each value $i \in \{1, 2, \ldots, K\}$ form a contiguous subsequence.

Proof. We say that a value $i \in \{1, 2, \ldots, K\}$ is *good* if its occurrences form a contiguous subsequence. We will show that if there exists $i \in \{1, 2, \ldots, K\}$ which is not good, then we can permute the sequence in such a way that T and the number of good values both increase. To this end, pick some value i which is not good and consider two adjacent blocks formed by its occurrences in the sequence. For each value $j \neq i$, let $u_j, v_j \in \{1, 2, \ldots, r_j\}$ be the indices of the gaps of the value j

containing the left and the right block respectively. Let the lengths of the left and the right block be a and b respectively. On moving the left block immediately next to the right block, T changes by

$$\Delta_1 = \sum_{j \neq i,\ a_j \neq b_j} [(\ell_{j,u_j} - a)^2 + (\ell_{j,v_j} + a)^2 - \ell_{j,u_j}^2 - \ell_{j,v_j}^2] > 2a \sum_{j \neq i,\ a_j \neq b_j} (\ell_{j,v_j} - \ell_{j,u_j}).$$

Similarly, on moving the right block immediately next to the left block, T changes by

$$\Delta_2 = \sum_{j \neq i,\ a_j \neq b_j} [(\ell_{j,u_j} + b)^2 + (\ell_{j,v_j} - b)^2 - \ell_{j,u_j}^2 - \ell_{j,v_j}^2] > 2b \sum_{j \neq i,\ a_j \neq b_j} (\ell_{j,u_j} - \ell_{j,v_j}).$$

Note that the above strict inequalities hold because there exists at least one value $j \neq i$ such that $a_j \neq b_j$, e.g. a value that occurs between the two blocks in consideration. Hence, at least one of the differences Δ_1, Δ_2 is strictly positive, so we can certainly perform one of the described transformations so that T increases. Moreover, it is clear that the number of good values increases because the value i becomes good, but no other value ceases to be good. This concludes the proof. □

By Claim 1, the problem reduces to the following: find a permutation π of the set $\{1, 2, \ldots, K\}$ which maximizes the function

$$f(\pi) = \sum_{i=1}^{K} \left(\left(\sum_{j=1}^{i-1} c_{\pi(j)} \right)^2 + \left(\sum_{j=i+1}^{K} c_{\pi(j)} \right)^2 \right).$$

In other words, we have to permute the sequence $c_1, \ldots, c_K$ so that the sum of the squares of the sums of all proper prefixes and suffixes is as large as possible. We can solve this problem naively in time $\mathcal{O}(K \cdot K!)$, or using dynamic programming with bitmasks in time $\mathcal{O}(K \cdot 2^K)$. However, for a solution with polynomial complexity, it is necessary to make the following observation:

Claim 2. In every optimal solution, the values in the sequence first increase and then decrease. More precisely, if π is an optimal permutation, then there exists $i \in \{1, 2, \ldots, K\}$ such that $c_{\pi(j)} \leq c_{\pi(j+1)}$ for all $1 \leq j < i$ and $c_{\pi(j)} \geq c_{\pi(j+1)}$ for all $i \leq j < K$.

Proof. Fix and index $j \in \{1, 2, \ldots, K-1\}$ and let π' be the permutation obtained by swapping the elements at positions j, $j+1$ in π, i.e. formally $\pi' = \pi \circ \tau$, where τ is the transposition which swaps j, $j+1$. Then we have

$$\begin{aligned} f(\pi') - f(\pi) &= (L_j + c_{\pi(j+1)})^2 + (R_{j+1} + c_{\pi(j)})^2 - (L_j + c_{\pi(j)})^2 - (R_{j+1} + c_{\pi(j+1)})^2 \\ &= 2(L_j - R_{j+1})(c_{\pi(j+1)} - c_{\pi(j)}), \end{aligned}$$

where we define $L_k = \sum_{l=1}^{k-1} c_{\pi(l)}$, $R_k = \sum_{l=k+1}^{K} c_{\pi(l)}$ for $1 \leq k \leq K$. Thus, if π is optimal, it follows that $L_j - R_{j+1}$ and $c_{\pi(j+1)} - c_{\pi(j)}$ have opposite sign, from which the desired conclusion follows. □

It follows from Claim 2 that an optimal solution can be built as follows: we iterate over the values $c_1, \ldots, c_K$ in increasing order and we maintain two sequences, a left one and a right one. In each step, we put the current value either at the end of the left sequence or at the beginning of the right sequence. In the end, we obtain the sought permutation by concatenating the left and the right sequence. It is now not hard to devise a solution via dynamic programming which runs in time $\mathcal{O}(K \cdot N)$, where $N = \sum_{i=1}^{K} c_i$. The state is the current position in the sorted sequence $c_1, \ldots, c_K$ and the sum of the left sequence, while the transition consists of choosing where to put the current value.

It turns out that an even stronger assertion holds (strictly speaking, it is stronger than the assertion obtained from Claim 2 on replacing the universal quantifier by an existential one):

Claim 3. If we (without loss of generality) assume that $c_1 \leq \ldots \leq c_K$, then the permutation π_0 given by

$$\pi_0(i) = \begin{cases} 2i - 1 & \text{if } 1 \leq i \leq \lceil \frac{K}{2} \rceil \\ 2(K - i + 1) & \text{if } \lceil \frac{K}{2} \rceil < i \leq K \end{cases}$$

is optimal. In other words, it is optimal to first arrange all elements with odd indices in increasing order and then all elements with even indices in decreasing order.

Proof. Let π be any permutation; we will show that $f(\pi_0) \geq f(\pi)$. To begin with, we introduce the map

$$s : S_K \to \mathbb{N}^{2(K-1)}$$

with the property that for all $\sigma \in S_K$ we have $s(\sigma)_1 \geq \ldots \geq s(\sigma)_{2(K-1)}$ and the multiset $\{s(\sigma)_i \mid 1 \leq i \leq 2(K-1)\}$ is equal to the multiset consisting of the sums of all proper prefixes and suffixes of the sequence $c_{\sigma(1)}, \ldots, c_{\sigma(K)}$ (here S_K denotes the set of all permutations of the set $\{1, 2, \ldots, K\}$). We have to show that

$$\sum_{i=1}^{2(K-1)} \varphi(s(\pi_0)_i) \geq \sum_{i=1}^{2(K-1)} \varphi(s(\pi)_i),$$

where

$$\varphi : \mathbb{R} \to \mathbb{R}, \quad x \mapsto x^2$$

is a convex function. To this end we will use Karamata's inequality. It suffices to show that for all $1 \leq i \leq 2(K-1)$ we have

$$\sum_{j=1}^{i} s(\pi_0)_j \geq \sum_{j=1}^{i} s(\pi)_j \tag{$\dagger$}$$

and that equality holds for $i = 2(K-1)$. The latter claim is obvious because for all $\sigma \in S_K$ we have

$$\sum_{j=1}^{2(K-1)} s(\sigma)_j = \sum_{i=1}^{K-1} \left(\sum_{j=1}^{i} c_{\sigma(j)} + \sum_{j=i+1}^{K} c_{\sigma(j)} \right) = N(K-1).$$

To prove ($\dagger$), observe that it is enough to show that for all $K \leq i \leq 2(K-1)$ we have

$$\sum_{j=i}^{2(K-1)} s(\pi_0)_j \leq \sum_{j=i}^{2(K-1)} s(\pi)_j. \tag{$\star$}$$

Indeed, then the inequality ($\dagger$) readily follows in the case $K-1 \leq i < 2(K-1)$ by subtracting the inequality ($\star$) with $i+1$ in place of i from the inequality ($\dagger$) with $2(K-1)$ in place of i. On the other hand, in the case $1 \leq i \leq K-1$, the inequality ($\dagger$) follows from the inequality ($\star$) with $2K-i-1$ in place of i because the i prefixes/suffixes with smallest sums are complementary to the i prefixes/suffixes with the largest sums.

Hence, it remains to prove the inequality ($\star$). It is enough to show that for all $1 \leq i \leq K-1$ and $0 \leq j \leq i$, the following inequality holds

$$\sum_{k=1}^{\lceil \frac{i}{2} \rceil} \sum_{l=1}^{k} c_{\pi_0(l)} + \sum_{k=1}^{\lfloor \frac{i}{2} \rfloor} \sum_{l=1}^{k} c_{\pi_0(K-l+1)} \leq \sum_{k=1}^{j} \sum_{l=1}^{k} c_{\pi(l)} + \sum_{k=1}^{i-j} \sum_{l=1}^{k} c_{\pi(K-l+1)}.$$

After grouping the same terms together, we see that this inequality is equivalent to

$$\textstyle\sum_{k=1}^{\lceil \frac{i}{2} \rceil} \left(\lceil \frac{i}{2} \rceil - k + 1 \right) c_{\pi_0(k)} + \sum_{k=1}^{\lfloor \frac{i}{2} \rfloor} \left(\lfloor \frac{i}{2} \rfloor - k + 1 \right) c_{\pi_0(K-k+1)} \leq \sum_{k=1}^{j} (j-k+1) c_{\pi(k)} + \sum_{k=1}^{i-j} (i-j-k+1) c_{\pi(K-k+1)}.$$

This inequality we can prove as follows. First, we replace the coefficients $j-k+1$ for $1 \leq k \leq j$ and $i-j-k+1$ for $1 \leq k \leq i-j$ in the expression on the right-hand side with the coefficients $\lceil \frac{i}{2} \rceil - k + 1$ for $1 \leq k \leq \lceil \frac{i}{2} \rceil$ and $\lfloor \frac{i}{2} \rfloor - k + 1$ for $1 \leq k \leq \lfloor \frac{i}{2} \rfloor$, in such a way that their relative ordering remains unchanged. Second, we permute the elements $c_{\pi(k)}$ for $1 \leq k \leq j$ and $c_{\pi(K-k+1)}$ for $1 \leq k \leq i-j$ so that their respective coefficients are oppositely sorted. Third, for all $1 \leq j \leq i$ we replace the j-th smallest element among the mentioned elements with c_j. This transforms the expression on the left-hand side into the one on the right-hand side, in such a way that its value never increases during the process. Indeed, in the first step, the coefficient of each element

doesn't increase (easy check – this boils down to the fact that the sorted version of the multiset $\{j-k+1 \mid 1 \leq k \leq j\} \cup \{i-j-k+1 \mid 1 \leq k \leq i-j\}$ for $j = \lceil \frac{i}{2} \rceil$ pointwise dominates the sorted version of the same multiset for any other value of j). In the second step, the claim is immediate from the rearrangement inequality, whereas in the third step, the claim is obvious. This concludes the proof. $\square$

Finally, the problem can be solved by sorting the sequence $c_1, \ldots, c_K$ and evaluating the function f on the permutation π_0 from Claim 3, in time $\mathcal{O}(K \log K)$ (after the initial step of determining the sequence $c_1, \ldots, c_K$, which takes $\mathcal{O}(N \log N)$ time).

2 The solution in the case of many queries

Once we have established the solution of the problem in the case $Q = 1$, we may move on to the version with more than one query. The main idea is to use the so-called Mo's algorithm, which is nothing else but a way of ordering the queries which yields a small sum of sizes of the symmetric differences of intervals in neighbouring queries. More concretely, if we denote the intervals by (l_i, r_i) (0-based) for $1 \leq i \leq Q$, then we sort the queries according to the lexicographic ordering of the respective pairs $(\lfloor \frac{l_i}{B} \rfloor, r_i)$, where we set $B = \lfloor \sqrt{N} \rfloor$. In other words, we split the sequence into $\lceil \frac{N}{B} \rceil$ blocks of size B, and we split the queries into groups depending on the block to which the left endpoint of the interval belongs. In each group, we sort the queries in increasing order according to the right endpoint of the interval. It is easy to see that the aforementioned sum of sizes of the symmetric differences of intervals doesn't exceed $\mathcal{O}((Q+N)\sqrt{N})$.

How does this observation help us to solve the problem? If we process the queries in the described order, then for all $0 \leq i \leq N-1$ we can maintain a counter $freq[i]$ which denotes the number of occurrences of the value i in the current interval (under the assumption that we have previously performed coordinate compression on the whole sequence). When moving on to the next query, we can refresh the counters by simply iterating over the symmetric difference of the current interval and the next one.

It is clear that the sequence $c_1, \ldots, c_K$ from Section 1 consists precisely of the non-zero elements of the array $freq$. However, there can be many such elements, so we have to keep track of the array $freq$ by storing it in a compressed form. This can be done by storing for each $1 \leq i \leq N$ a counter $comp[i]$ which denotes the number of occurrences of the value i in the array $freq$. We are now interested in the non-zero elements of the array $comp$. How many such elements can there be? Observe that the sum of $comp[i] \cdot i$ over all $1 \leq i \leq N$ equals the sum of $freq[i]$ over all $0 \leq i \leq N-1$, which is at most N. Thus, if the array $comp$ has L non-zero elements, then the sum of their indices is at the same time at least $1 + 2 + \ldots + L = \frac{L(L+1)}{2}$ and at most N. Therefore, we have $L \leq \sqrt{2N}$, so $comp$ has at most $\mathcal{O}(\sqrt{N})$ non-zero elements.

It remains only to efficiently evaluate the function f on the permutation π_0, for the sequence $c_1, \ldots, c_K$ given in the described compressed form. This can be done in time $\mathcal{O}(L)$ by performing a summation which involves the sum of squares of the elements of an arithmetic progression, on the condition that we can efficiently obtain the non-zero elements of the array $comp$ in the order in which they appear in the array. This can be achieved by either storing the array $comp$ in a map, or by storing the indices of the non-zero elements in a separate vector, which is then sorted as needed. This approach has time complexity $\mathcal{O}((Q+N)\sqrt{N} \log N)$. For the purposes of sorting, we can also use radix sort in order to recover the complexity $\mathcal{O}((Q+N)\sqrt{N})$. Even though the latter approach is faster in theory, it is the former approach that gives better results in practice. For more details concerning implementation, please consult the attached source code of the official solution.

3 Appendix: Inequalities used in the proofs

In the proof of Claim 3, we made use of the following (more or less standard) results:

Theorem 1. (Karamata's inequality) Let $f : \mathbb{R} \to \mathbb{R}$ be a convex function, that is to say, such that

for all $x, y \in \mathbb{R}$, $t \in [0,1]$ the following inequality holds

$$f((1-t)x + ty) \leq (1-t)f(x) + tf(y).$$

Let $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ be real numbers such that $x_1 \geq \ldots \geq x_n$, $y_1 \geq \ldots \geq y_n$ and for all $1 \leq i \leq n$ we have

$$\sum_{j=1}^{i} x_j \geq \sum_{j=1}^{i} y_j,$$

with equality in the case $i = n$. Then the following inequality holds

$$\sum_{i=1}^{n} f(x_i) \geq \sum_{i=1}^{n} f(y_i).$$

Proof in the case $f(x) = x^2$. Let $p_i = \sum_{j=1}^{i} x_j$, $q_i = \sum_{j=1}^{i} y_j$ for $0 \leq i \leq n$. Then we have $p_0 = q_0 = 0$, $p_n = q_n$ and $p_i \geq q_i$ for all $0 \leq i \leq n$. Using summation by parts, we obtain that

$$\begin{aligned}
\sum_{i=1}^{n} f(x_i) - \sum_{i=1}^{n} f(y_i) &= \sum_{i=1}^{n} (x_i - y_i)(x_i + y_i) \\
&= \sum_{i=1}^{n} [(p_i - p_{i-1}) - (q_i - q_{i-1})](x_i + y_i) \\
&= \sum_{i=1}^{n} [(p_i - q_i) - (p_{i-1} - q_{i-1})](x_i + y_i) \\
&= \sum_{i=1}^{n} (p_i - q_i)(x_i + y_i) - \sum_{i=0}^{n-1} (p_i - q_i)(x_{i+1} + y_{i+1}) \\
&= \sum_{i=1}^{n-1} (p_i - q_i)(x_i + y_i) - \sum_{i=1}^{n-1} (p_i - q_i)(x_{i+1} + y_{i+1}) \\
&= \sum_{i=1}^{n-1} (p_i - q_i)[(x_i - x_{i+1}) + (y_i - y_{i+1})] \geq 0,
\end{aligned}$$

where the last inequality holds because $p_i - q_i, x_i - x_{i+1}, y_i - y_{i+1} \geq 0$ for $1 \leq i \leq n-1$. □

Theorem 2. (Rearrangement inequality) Let $x_1, \ldots, x_n$ te $y_1, \ldots, y_n$ be real numbers such that $x_1 \leq \ldots \leq x_n$, $y_1 \leq \ldots \leq y_n$. Then for all permutations σ of the set $\{1, 2, \ldots, n\}$ the following inequalities hold

$$\sum_{i=1}^{n} x_i y_{n-i+1} \leq \sum_{i=1}^{n} x_i y_{\sigma(i)} \leq \sum_{i=1}^{n} x_i y_i.$$

Proof. Note that the first inequality follows by applying the second inequality to the sequence $-y_n, \ldots, -y_1$ in place of $y_1, \ldots, y_n$. Hence, it suffices to prove the second inequality. Choose a permutation σ which maximizes the value of the sum $\sum_{i=1}^{n} x_i y_{\sigma(i)}$. If there is more than one such permutation, choose the lexicographically smallest one. We claim that σ is the identity permutation, i.e. that we have $\sigma(i) = i$ for all $1 \leq i \leq n$. Indeed, if this is not the case, then there exists $i \in \{1, 2, \ldots, n-1\}$ such that $\sigma(i) > \sigma(i+1)$. Then let $\sigma' = \sigma \circ \tau$, where τ is the transposition that swaps i, $i+1$. Then we have

$$\sum_{i=1}^{n} x_i y_{\sigma'(i)} - \sum_{i=1}^{n} x_i y_{\sigma(i)} = x_i y_{\sigma(i+1)} + x_{i+1} y_{\sigma(i)} - x_i y_{\sigma(i)} - x_{i+1} y_{\sigma(i+1)} = (x_{i+1} - x_i)(y_{\sigma(i)} - y_{\sigma(i+1)}) \geq 0.$$

Furthermore, σ' is lexicographically smaller than σ, which is a contradiction. □