

fn peel_permute_and_flip

Michael Shoemate

November 30, 2025

This proof resides in “**contrib**” because it has not completed the vetting process.

This document proves soundness of `peel_permute_and_flip` in `mod.rs` at commit `e62b0aa2` (outdated¹). `peel_permute_and_flip` noisily selects the index of the greatest score from a vector of input scores k times without replacement.

1 Hoare Triple

Preconditions

Types consistent with pseudocode.

Pseudocode

```
1 def peel_permute_and_flip(
2     x: list[RBig],
3     scale: RBig,
4     k: usize,
5     replacement: bool,
6 ):
7     natural_order = []
8     sorted_order = BTreeSet.new()
9
10    for _ in range(min(k, x.len())):
11        index = permute_and_flip(x, scale, replacement)  #
12        x.remove(index)  #
13
14        # map index on modified x back to original x (postprocessing)
15        for del_ in sorted_order:  # |
16            if del_ <= index:
17                index += 1
18            else:
19                break
20
21        sorted_order.insert(index)
22        natural_order.push(index)
23
24    return natural_order
```

Postcondition

Theorem 1.1. • If replacement is set, returns a sample from \mathcal{M}_{EM} (as defined in MS2023 Definition 4), otherwise returns a sample from \mathcal{M}_{PF} (as defined in MS2023 Lemma 1), k times by peeling, where $scale = \frac{2\cdot\Delta}{\epsilon}$.

¹See new changes with `git diff e62b0aa2..5907d96 rust/src/measurements/noisy_top_k/mod.rs`

- Errors are data-independent, except for exhaustion of entropy.

Proof. The pseudocode applies `permute_and_flip` on line 11 up to k times (no greater than the number of scores), each time removing the selected score on line 12.

Since the index in subsequent selections may be offset by an earlier removal, a data-independent post-processing of the index is performed on line 15.

The only source of error is due to entropy exhaustion. □