fn peel_permute_and_flip

Michael Shoemate

September 8, 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This document proves soundness of `peel_permute_and_flip` in `mod.rs` at commit e62b0aa2 (outdated[1]). `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(x: list[RBig], scale: RBig, k: usize):
2      natural_order = []
3      sorted_order = BTreeSet.new()
4
5      for _ in range(min(k, x.len())):
6          index = permute_and_flip(x, scale)   #
7          x.remove(index)   #
8
9          # map index on modified x back to original x (postprocessing)
10         for del_ in sorted_order:   # |
11             if del_ <= index:
12                 index += 1
13             else:
14                 break
15
16         sorted_order.insert(index)
17         natural_order.push(index)
18
19     return natural_order
```

Postcondition

Theorem 1.1. Returns the index of the max element z_i, where each $z_i \sim \text{Exp}(\text{shift} = x_i, \text{scale} = \texttt{scale})$, k times without replacement.

Proof. The pseudocode applies `permute_and_flip` on line 6 up to k times (no greater than the number of scores), each time removing the selected score on line 7.

Since the index in subsequent selections may be offset by an earlier removal, a data-independent postprocessing of the index is performed on line 10. □

---

[1]See new changes with `git diff e62b0aa2..d61e97a rust/src/measurements/noisy_top_k/exponential/mod.rs`