

# fn permute\_and\_flip

Michael Shoemate

December 26, 2025

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

This document proves soundness of `permute_and_flip` [1] in `mod.rs` at commit `e62b0aa2` (outdated<sup>1</sup>). `permute_and_flip` noisily selects the index of the greatest score from a vector of input scores.

## 1 Hoare Triple

### Preconditions

*Types consistent with pseudocode.*

### Pseudocode

```
1 def permute_and_flip(x: list[RBig], scale: RBig, replacement: bool):
2     if scale.is_zero(): #
3         return max(range(x.len()), key=lambda i: x[i])
4
5     # begin nonzero scale
6     x_max = max(x)
7     permutation = list(range(x.len()))
8
9     sequence = range(0, len(x)) if replacement else repeat(0)
10
11     for left in sequence:
12         right = left + sample_uniform_uint_below(x.len() - left)
13         # fisher-yates shuffle up to left
14         permutation.swap(left, right)
15
16         candidate = permutation[left]
17         if sample_bernoulli_exp((x_max - x[candidate]) / scale):
18             return candidate
19
20     raise "at least one x[candidate] is equal to x_max"
```

### 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), where `scale` =  $\frac{2-\Delta}{\epsilon}$ .

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

*Proof.* By swapping elements on line 13, an online Fisher-Yates shuffle is applied up to and including index `left`.

<sup>1</sup>See new changes with `git diff e62b0aa2..9bdafa70 rust/src/measurements/noisy_top_k/mod.rs`

Substituting  $\text{scale} = \frac{2\Delta}{\epsilon}$ , the argument to `sample_bernoulli_exp` is then  $\frac{\epsilon}{2\Delta}(q_* - q_r)$ , which is non-negative, satisfying the precondition of `sample_bernoulli_exp`. Therefore by the postcondition of `sample_bernoulli_exp`, the response is a sample from  $\text{Bern}(\exp(-x))$ , where  $x = \frac{\epsilon}{2\Delta}(q_* - q_r)$ . Therefore the response is a sample from  $\text{Bern}(\exp(\frac{\epsilon}{2\Delta}(q_r - q_*)))$ , which is equivalent to Algorithm 1 in [1].

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

## References

- [1] Ryan McKenna and Daniel Sheldon. Permute-and-flip: A new mechanism for differentially private selection, 2020.