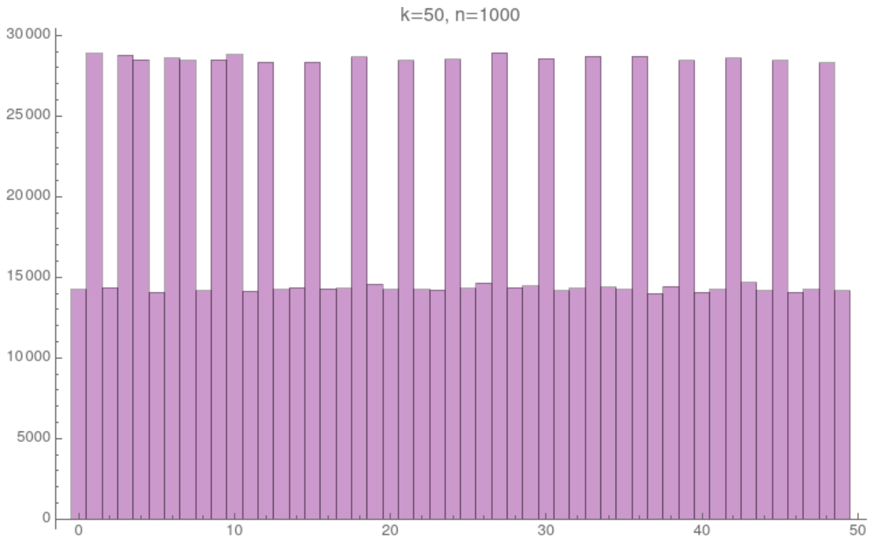
Seven algorithms for the same task

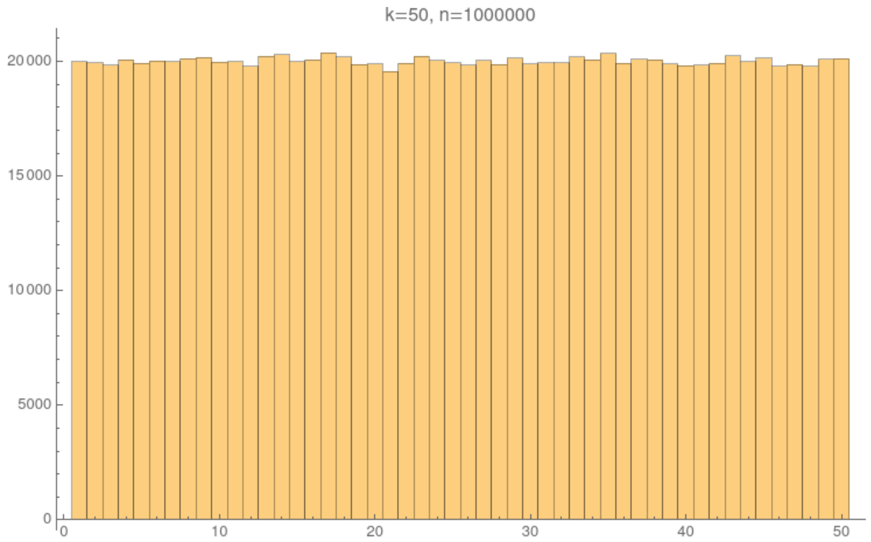
Testing if your data is uniformly distributed

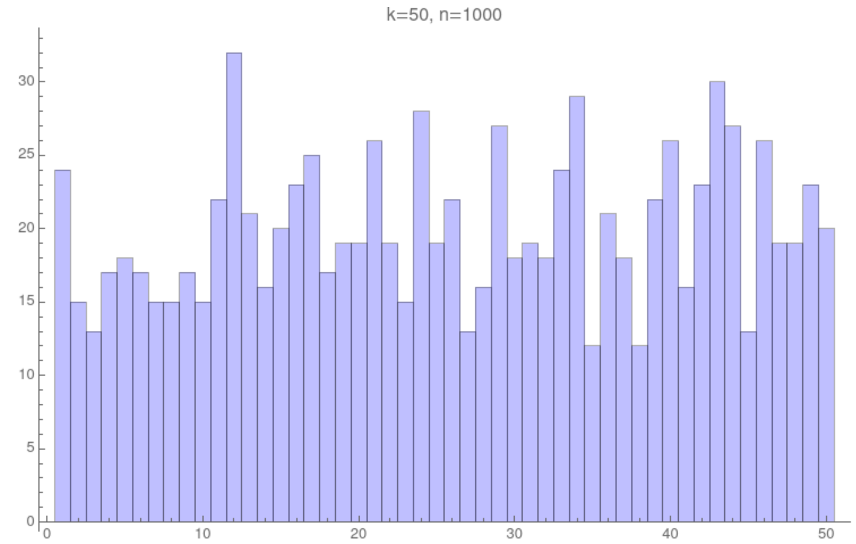
You have n i.i.d. samples from some unknown distribution over

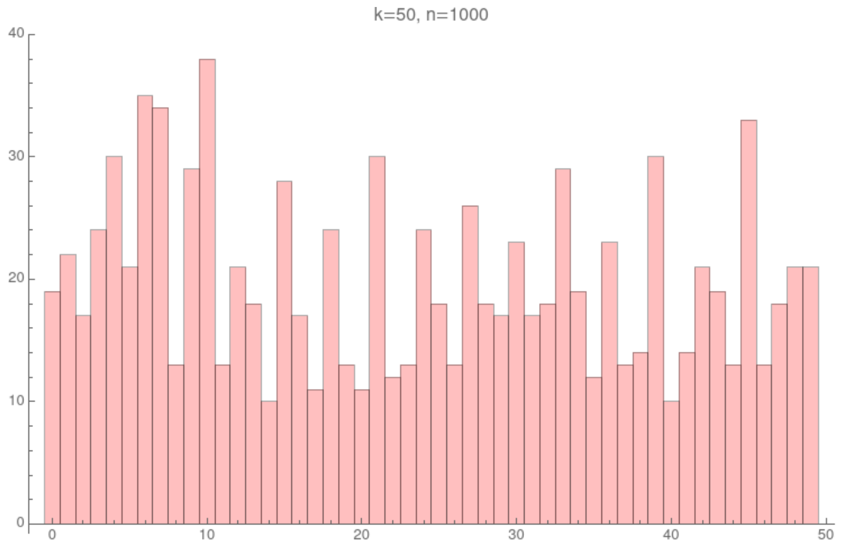
$$[k]=\{1,2,\ldots,k\}$$

and want to know: is it *the* uniform distribution? Or is it **statistically far** from it, say, at total variation distance ε?


k=50, n=1000
30000
25000
20000
15000
10000
5000
0
0
10
20
30
40
50


k=50, n=1000000
20000
15000
10000
5000
0
0
10
20
30
40
50


k=50, n=1000
30
25
20
15
10
5
0
0
10
20
30
40
50


k=50, n=1000
40
30
20
10
0
0
10
20
30
40
50

Total variation distance:

$$\mathrm{d}_{\mathrm{TV}}(\mathbf{p}, \mathbf{q}) = \sup_{S \subseteq [k]} (\mathbf{p}(S) - \mathbf{q}(S)) = \frac{1}{2} \|\mathbf{p} - \mathbf{q}\|_1 \in [0, 1]$$

"a measure of *how distinguishable* two distributions are given a single sample"

Uniformity testing algorithm:

Input: ε in [0,1], n i.i.d. samples from unknown p over [k]

Output: **accept** or **reject**

- If p=u, accept with probability ≥ .99
- If TV(p,u)>ε, reject with probability ≥ .99

Uniformity testing $\Leftrightarrow$ Identity testing

.99 is arbitrary

Optimal n is $\Theta(\sqrt{k}/\varepsilon^2)$

Nice, but **how**?

(Some ideas?)

Nice, but **how**? And also, **what**?

- **Data efficiency:** does the algo achieve optimal sample complexity?
- **Time efficiency:** how fast is the algo to run ?
- **Memory efficiency:** how much memory does the algo require ?
- **Simplicity:** is the algo simple to describe and implement?
- **Simplicity':** is the algo simple to *analyse*?
- **Robustness**: how "tolerant" is the algo to noise?
- **Elegance:** OK, that's a bit subjective, but you get it
- **Generalizable**: Does the algo have useful "bonus features"?

Nice, but **how**? And also, **what**?

	Sample complexity	**Notes**	**References**
Collision-based	$\frac{k^{1/2}}{\varepsilon^2}$	Tricky	[GR00, DGPP19]
Unique elements	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/4}$	[Pan08]
Modified χ^2	$\frac{k^{1/2}}{\varepsilon^2}$	Nope	[VV17, ADK15, DKN15]
Empirical distance to uniform	$\frac{k^{1/2}}{\varepsilon^2}$	Biased	[DGPP18]
Random binary hashing	$\frac{k}{\varepsilon^2}$	Fun (+ fast, small space)	[ACT19]
Bipartite collisions	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/10}$	[DGKR19]
Empirical subset weighting	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/4}$	

Key Insight (4 of the Dwarfs)

Forget about TV distance, ℓ_2 distance is a good proxy:

$$\mathrm{d}_{\mathrm{TV}}(\mathbf{p}, \mathbf{u}_k) = \frac{1}{2}\|\mathbf{p} - \mathbf{u}_k\|_1 \leq \frac{\sqrt{k}}{2}\|\mathbf{p} - \mathbf{u}_k\|_2$$

so if p is at TV ≥ ε, it is at $\ell_2 \geq 2\varepsilon/\sqrt{k}$.

Key Insight (4 of the Dwarfs)

Also,

$$\|\mathbf{p} - \mathbf{u}_k\|_2^2 = \sum_{i=1}^{k}(\mathbf{p}(i) - 1/k)^2 = \sum_{i=1}^{k}\mathbf{p}(i)^2 - 1/k = \|\mathbf{p}\|_2^2 - 1/k$$

so it suffices to estimate $\|p\|_2$. How?

Collisions

Fact.

$$\Pr_{x,y\sim\mathbf{p}}[x = y] = \sum_{i=1}^{k} \mathbf{p}(i)^2 = \|\mathbf{p}\|_2^2$$

I.e., the squared ℓ_2 norm is the "collision probability."

Collisions

Natural idea.

$$Z_1 = \frac{1}{\binom{n}{2}} \sum_{s \neq t} \mathbb{1}_{\{x_s = x_t\}}$$

Take n samples $x_1,\ldots x_n$. For each of the $\binom{n}{2}$ pairs, check if a *collision* occurs. Count those collisions, and use the result as unbiased estimator for $\|p\|_2^2$; threshold appropriately.

Collisions

Natural idea.

$$Z_1 = \frac{1}{\binom{n}{2}} \sum_{s \neq t} \mathbb{1}_{\{x_s = x_t\}}$$

Take n samples $x_1, \ldots x_n$. For each of the {n choose 2} pairs, check if a collision occurs. Count those collisions, and use the result as unbiased estimator for $\|p\|_2^2$; threshold appropriately.

✔ Simple ✔ Fast ✔ Intuitive ✔ Elegant

Not so **simple'**

Collisions

More detail:

We want to threshold Z_1 at $(1+2\varepsilon^2)/k$ or so, to distinguish **uniform** ($\mathbb{E}[Z_1] = 1/k$) from **far from uniform** ($\mathbb{E}[Z_1] = \|p\|_2^2 \geq (1+4\varepsilon^2)/k$).

So we want to bound the variance of Z_1 and use Chebyshev's inequality. This gets... messy.

(Getting $\Theta(\sqrt{k}/\varepsilon^4)$ is not hard. The optimal $\Theta(\sqrt{k}/\varepsilon^2)$ is challenging.)

Unique elements

Take n samples, count the number Z_2 of elements that appear exactly **once**.

Unique elements

Take n samples, count the number Z_2 of elements that appear exactly **once**.

$$\mathbb{E}[Z_2] = n \sum_{i=1}^{k} \mathbf{p}(i)(1 - \mathbf{p}(i))^{n-1}$$

Unique elements

Take n samples, count the number Z_2 of elements that appear exactly **once**.

$$\mathbb{E}[Z_2] = n \sum_{i=1}^{k} \mathbf{p}(i)(1 - \mathbf{p}(i))^{n-1}$$

Under uniform: $\approx n - n^2/k$

Under "far" p: $\approx n - n^2\|p\|_2^2 \leq n - n^2/k - 2n^2\varepsilon^2/k$

Unique elements

More detail:

Assuming the variance is small enough,

the $n^2\varepsilon^2/k$ gap in expectation
+ Chebyshev (again)
+ all approximations from the previous slide holding

let us test as long as $n=\Omega(\sqrt{k}/\varepsilon^2)$.

Unique elements

More detail:

Assuming the variance is small enough,

the $n^2\varepsilon^2/k$ gap in expectation
+ Chebyshev (again)
+ all approximations from the previous slide holding

let us test as long as $n=\Omega(\sqrt{k}/\varepsilon^2)$.

✔ Simple ✔ Fast ✔ Intuitive ✔ Elegant

Unique elements

More detail:

Assuming the variance is small enough,

the $n^2\varepsilon^2/k$ gap in expectation
+ Chebyshev (again)
+ all approximations from the previous slide holding

let us test as long as $n=\Omega(\sqrt{k}/\varepsilon^2)$.

Problem: can't work for $\varepsilon \gg 1/k^{1/4}$, since then $n \gg k$ (but we can't have that many distinct elements...)

Next stop: χ^2

Idea: the χ^2 divergence between distributions is a ~~metric~~ thing, related to KL divergence and others. Pearson's χ^2 test is a staple of Statistics. Can we have a test inspired by that?

Next stop: χ^2

Idea: the χ^2 divergence between distributions is a ~~metric~~ thing, related to KL divergence and others. Pearson's χ^2 test is a staple of Statistics. Can we have a test inspired by that?

$$Z_3 = \sum_{i=1}^{k} \frac{(N_i - n/k)^2 - N_i}{n/k}$$

where N_i = # times we see i among the n samples.

Next stop: χ^2

Idea: the χ^2 divergence between distributions is a ~~metric~~ thing, related to KL divergence and others. Pearson's χ^2 test is a staple of Statistics. Can we have a test inspired by that?

$$Z_3 = \sum_{i=1}^{k} \frac{(N_i - n/k)^2 - N_i}{n/k}$$

where N_i = # times we see i among the n samples. It works.*

($\mathbb{E}[Z_3] = nk\|p\|_2^2$ and, again, Chebyshev.)

Plugin estimator: why are we doing all this?

We've been doing a lot of specific stuff, with ad hoc estimators. **Why?**

Plugin estimator: why are we doing all this?

We've been doing a lot of specific stuff, with ad hoc estimators. **Why?**

Can't we just:

1. take our n samples
2. compute the empirical distribution $\hat{p}$
3. see if the "plugin" distance $TV(\hat{p},u)$ is large
4. be done

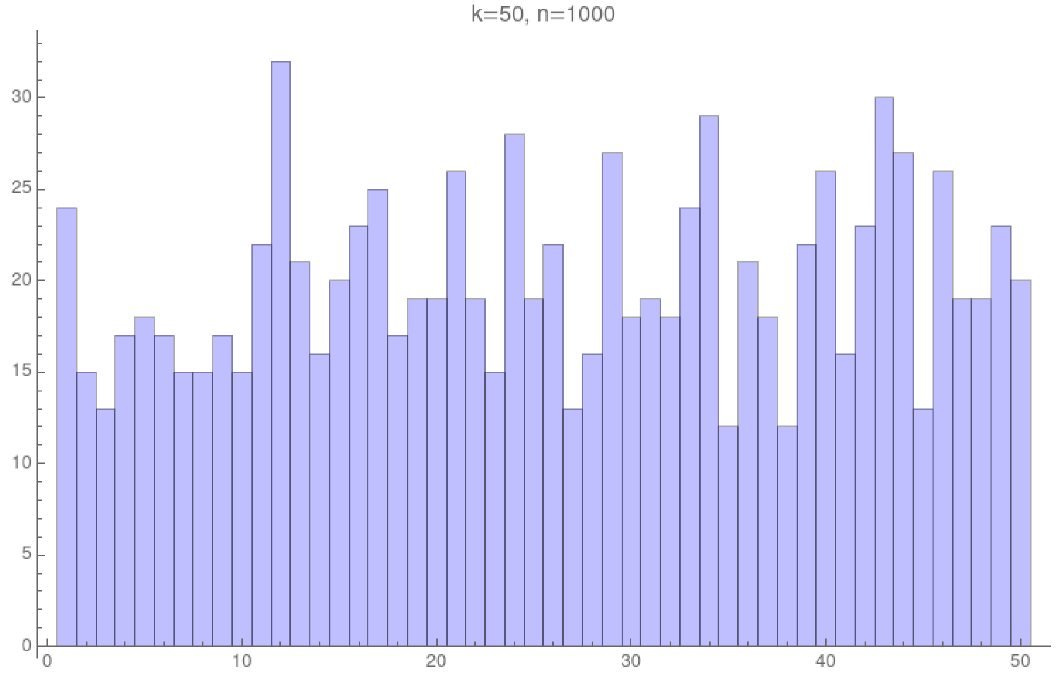
?

Plugin estimator: why are we doing all this?

Of course not: the empirical distance TV($\hat{p}$,u) will be very large

$$TV(\hat{p},u) = 1-o(1)$$

even if p **is** uniform, for any n ≪ k.



Plugin estimator: why are we doing all this?

But still yes: the empirical distance TV($\hat{p}$,u) will be very large

$$TV(\hat{p},u) = 1-o(1)$$

even if p **is** uniform, for any $n \ll k$, indeed.

But that "o(1)" is not the same if p=u and if TV(p,u) > ε. And somehow that's enough!

Plugin estimator: why are we doing all this?

But still yes: the empirical distance $TV(\hat{p},u)$ will be very large

$$TV(\hat{p},u) = 1-o(1)$$

even if p **is** uniform, for any $n \ll k$, indeed.

But that "o(1)" is not the same if p=u and if $TV(p,u) > \varepsilon$. And somehow that's enough!

Need more than Chebyshev for that one.

Plugin estimator: why are we doing all this?

✔ Simple ✔ Fast Intuitive?!? ✔ Elegant ✔ **Generalises**

Also, the first one we see not relying on ℓ_2 norm as a proxy.

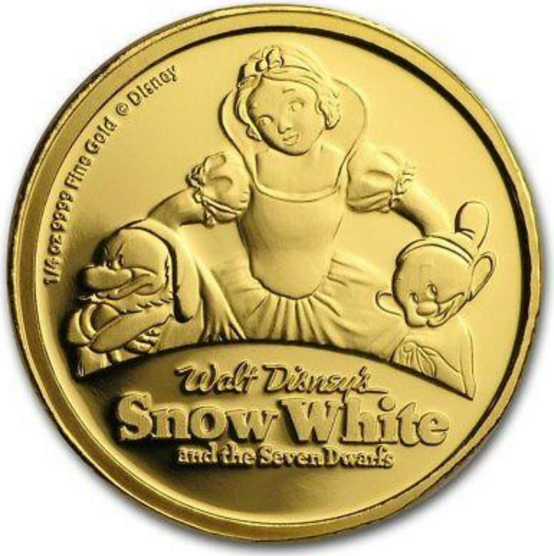
Binary hashing

I don't like big numbers, like k.

Binary hashing

I don't like big numbers, like k.

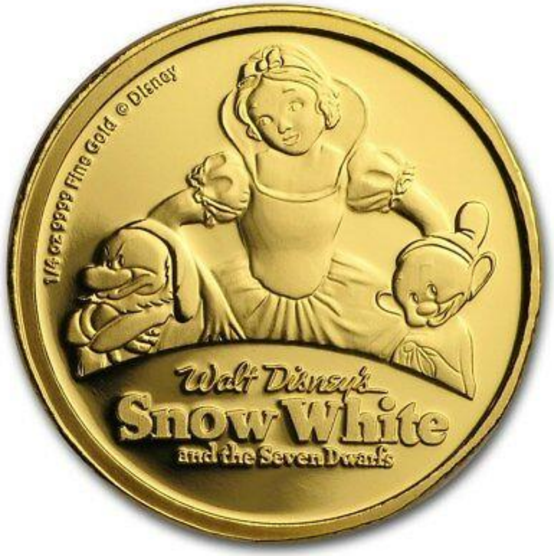
Fact. Distinguishing between a fair coin (Bernoulli(½)) and a coin with bias α (Bernoulli(½±α)) can be done with $\Theta(1/\alpha^2)$ samples.



Binary hashing

I don't like big numbers, like k.

Fact. Distinguishing between a fair coin (Bernoulli(½)) and a coin with bias α (Bernoulli(½±α)) can be done with $\Theta(1/\alpha^2)$ samples.

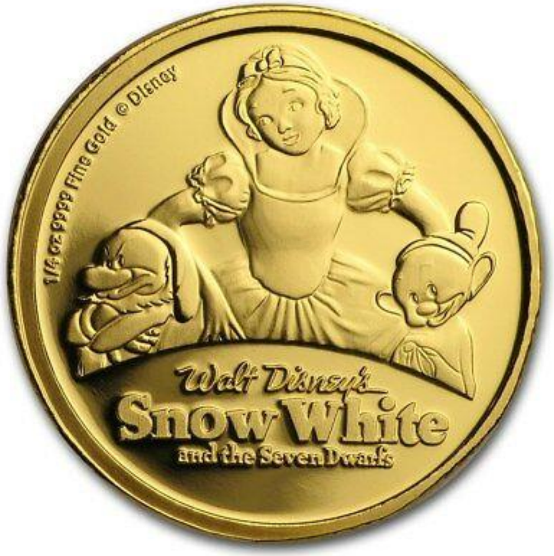


If we had k=2, we could use that.

Binary hashing

I don't like big numbers, like k.

Fact. Distinguishing between a fair coin (Bernoulli(½)) and a coin with bias α (Bernoulli(½±α)) can be done with $\Theta(1/\alpha^2)$ samples.



If we had k=2, we could use that. So let's **make** k=2.

Binary hashing

Partition the domain [k] in two equal parts at random, S and [k]\S. Then if a sample is in S, it's *tails*; otherwise, it's *heads*.

Binary hashing

Partition the domain $[k]$ in two equal parts at random, S and $[k]\setminus S$. Then if a sample is in S, it's *tails*; otherwise, it's *heads*.

- Of course, if $p=u$, then $p(S)=|S|/k=½$. Fair coin!

Binary hashing

Partition the domain [k] in two equal parts at random, S and [k]\S. Then if a sample is in S, it's *tails*; otherwise, it's *heads*.

- Of course, if p=u, then p(S)=|S|/k=½. **Fair** coin!
- If TV(p,u) ≥ ε, however...

$$\Pr_{S\subseteq[k]}\left[\,|\mathbf{p}(S)-\mathbf{u}_k(S)|=\Omega(\varepsilon/\sqrt{k})\,\right]=\Omega(1)$$

Biased coin! (With constant probability over choice of S)

Binary hashing

Now we can use our fact, with $\alpha := \varepsilon/\sqrt{k}$. Give sample complexity

$$\Theta(1/\alpha^2) = \Theta(k/\varepsilon^2)$$

Binary hashing

Now we can use our fact, with $\alpha := \varepsilon/\sqrt{k}$. Gives sample complexity

$$\Theta(1/\alpha^2) = \Theta(k/\varepsilon^2)$$

✔ Simple ✔ Fast ✓**Fun** ✔ Elegant ✔ **Generalises** **Not optimal**

(**"Sometimes optimal":** very useful in some settings!**)**

Bipartite collision tester

This one is a bit... boring: like the collision-based, but you divide the n samples in two sets S_1, S_2 and count collisions between S_1 and S_2 only.

$$Z_5 = \frac{1}{n_1 n_2} \sum_{(x,y)\in S_1 \times S_2} \mathbb{1}_{\{x=y\}}$$

Bipartite collision tester

This one is a bit... boring: like the collision-based, but you divide the n samples in two sets S_1, S_2 and count collisions between S_1 and S_2 only.

$$Z_5 = \frac{1}{n_1 n_2} \sum_{(x,y)\in S_1\times S_2} \mathbb{1}_{\{x=y\}}$$

Advantage? Also sample-optimal, sometimes useful (more flexible), e.g., with **memory** constraints.

Bipartite collision tester

This one is a bit... boring: like the collision-based, but you divide the n samples in two sets S_1, S_2 and count collisions between S_1 and S_2 only.

$$Z_5 = \frac{1}{n_1 n_2} \sum_{(x,y)\in S_1\times S_2} \mathbb{1}_{\{x=y\}}$$

Advantage? Also sample-optimal, sometimes useful (more flexible), e.g., with **memory** constraints.

[Analysis: Chebyshev returns.]

Final tester: The Weird One

Final tester: The Weird One

Take the first s of your n samples. That's a set S.

Final tester: The Weird One

Take the first s of your n samples. That's a set S.

$$\mathbb{E}[\mathbf{p}(S)] = \sum_{i=1}^{k} \mathbf{p}(i)(1 - (1 - \mathbf{p}(i))^s)$$

which is $\approx s\|p\|_2^2$.

Final tester: The Weird One

Take the first s of your n samples. That's a set S.

$$\mathbb{E}[\mathbf{p}(S)] = \sum_{i=1}^{k} \mathbf{p}(i)(1 - (1 - \mathbf{p}(i))^s)$$

which is $\approx s\|p\|_2^2$. Then use the remaining $n'=n-s$ samples to estimate p(S): is it $\approx s/k$, or $\geq s(1+2\varepsilon^2)/k$?

Final tester: The Weird One

Then use the remaining $n'=n-s$ samples to estimate $p(S)$:

is it $\approx s/k$, or $\geq s(1+2\varepsilon^2)/k$?

For it to work, we need

$$n - s \gtrsim \frac{k}{s\varepsilon^4}$$

and we can optimise: $s=n/2$ gives us the optimal $n=O(\sqrt{k}/\varepsilon^2)$.

Final tester: The Weird One

Then use the remaining $n'=n-s$ samples to estimate $p(S)$:

is it $\approx s/k$, or $\geq s(1+2\varepsilon^2)/k$?

For it to work, we need

$$n - s \gtrsim \frac{k}{s\varepsilon^4}$$

and we can optimise: $s=n/2$ gives us the optimal $n=O(\sqrt{k}/\varepsilon^2)$.

✔ Simple ✔ Fast ✓**Fun** ✔ Elegant ✔ **Generalises** but needs $\varepsilon \gg 1/k^{1/4}$

Final tester: The Weird One

Then use the remaining $n'=n-s$ samples to estimate $p(S)$:

is it $\approx s/k$, or $\geq s(1+2\varepsilon^2)/k$?

For it to work, we need

$$n - s \gtrsim \frac{k}{s\varepsilon^4}$$

and we can optimise: $s=n/2$ gives us the optimal $n=O(\sqrt{k}/\varepsilon^2)$.

✔ Simple ✔ Fast ✔**Fun** ✔ Elegant ✔ **Generalises** but needs $\varepsilon \gg 1/k^{1/4}$

Can you see why?

Thank you. (Questions?)

	Sample complexity		**References**
Collision-based	$\frac{k^{1/2}}{\varepsilon^2}$	Tricky	[GR00, DGPP19]
Unique elements	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/4}$	[Pan08]
Modified χ^2	$\frac{k^{1/2}}{\varepsilon^2}$	Nope	[VV17, ADK15, DKN15]
Empirical distance to uniform	$\frac{k^{1/2}}{\varepsilon^2}$	Biased	[DGPP18]
Random binary hashing	$\frac{k}{\varepsilon^2}$	Fun (+ fast, small space)	[ACT19]
Bipartite collisions	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/10}$	[DGKR19]
Empirical subset weighting	$\frac{k^{1/2}}{\varepsilon^2}$	$\varepsilon \gg 1/k^{1/4}$	