AOL × RH — run004

Directory: `/articles/riemann-hypothesis/run004/`

Overview

Run 004 extends Runs 1–3 by adding formal goodness-of-fit tests against *Exponential(1)* on unfolded spacings and by comparing the zeros' spacing spectrum to lattice spokes `n(r)=3r²−r+c` (c∈{2..7}) via FFT overlaps. Artifacts follow the same folder layout and naming convention as previous runs.

How to reproduce locally

Pick one environment below. All produce the same `r004_*` figures & metrics using Odlyzko's `zeros6.gz` (first 100k zeros).

1) Get the zeros file

```
# macOS / Linux
curl -L -o zeros6.gz "https://www-users.cse.umn.edu/~odlyzko/zeta_tables/zeros6.gz"

# Windows (PowerShell)
Invoke-WebRequest -Uri "https://www-users.cse.umn.edu/~odlyzko/zeta_tables/zeros6.gz" -OutFile "zeros6.gz
```

Copy

2) Choose one runtime

Python (recommended)

1. Copy the "Reproducibility — Python (Run 004)" script from this page into `run004_make_all.py`.
2. Create a venv and run the script:

```
python -m venv .venv
source .venv/bin/activate   # Windows: .venv\Scripts\activate
pip install numpy pandas matplotlib
python run004_make_all.py
```

Copy

PowerShell (native, no Python)

1. Copy the "Reproducibility — PowerShell (Run 004)" block into `run004_make_all.ps1`.
2. Execute:

```
.\run004_make_all.ps1 -ZerosPath zeros6.gz
```

Copy

PHP (CLI)

1. Copy the "Reproducibility — PHP (Run 004)" block into `run004_make_all.php`.
2. Run:

```
php run004_make_all.php zeros6.gz
```

Copy

C# (.NET 6 + ScottPlot)

1. Create the project and add ScottPlot:

```
dotnet new console -o Run004Cs
cd Run004Cs
dotnet add package ScottPlot --version 5.0.19
```

Copy

2. Replace `Program.cs` with the "Reproducibility — C# (Run 004)" code, then run:

```
dotnet run -- zeros6.gz
```

Copy

3) Outputs (all runtimes)

Figures	`run004/figures/r004_ecdf_vs_exp.png`
	`run004/figures/r004_qq_exp.png`
	`run004/figures/r004_fft_corr_bar.png`
	`run004/figures/r004_fft_magnitude_overlay_best.png`

Metrics	`run004/metrics/r004_ks_cvm_summary.json` `run004/metrics/r004_lattice_metrics.csv`
Data	`run004/data/r004_fft_zeros.csv` `run004/data/r004_fft_spoke_c{2..7}.csv`

Tip: if you're seeing cached images in the browser, hard-refresh or append `?v=timestamp` to the image URL.

Run 004 — Summary

Dataset	Odlyzko zeros (`zeros6.gz`): first 100,000 ordinates → 99,999 gaps.
Spacings (unfolded)	Mean ≈ **1.00000** (by construction), σ ≈ **0.42925**, var ≈ **0.18426**.
Raw spacing mean	≈ **0.749074** (used to unfold).
GOF vs Exp(1)	KS D ≈ **0.291643** (p ≈ 0), CvM T ≈ **3050.068**.
Lattice spokes (≤ 1e6)	c=3: 88 primes → KS D≈0.1478 (p≈0.040), spec-corr≈−0.303 · c=5: 84 primes → KS D≈0.0929 (p≈0.453), spec-corr≈−0.324 · c=7: 103 primes → KS D≈0.1590 (p≈0.010), spec-corr≈−0.097. c=2,4,6: too few primes for spacing stats.
Figures	ECDF vs Exp(1) · Q–Q vs Exp(1) · FFT overlap (bar) · FFT overlay (best spoke)
Metrics	r004_ks_cvm_summary.json · r004_lattice_metrics.csv

Notes

- KS/CvM are against Exp(1) on unfolded Δγ; extremely small p-value underflows to 0 in double-precision.
- Spectral correlation uses Pearson on `log(1+magnitude)` after interpolating onto zeros' frequency grid (skip f=0).
- Best (least-negative) lattice overlap is c=7, spec-corr≈−0.097 at ≤1e6.

Figure: ECDF vs Exp(1)

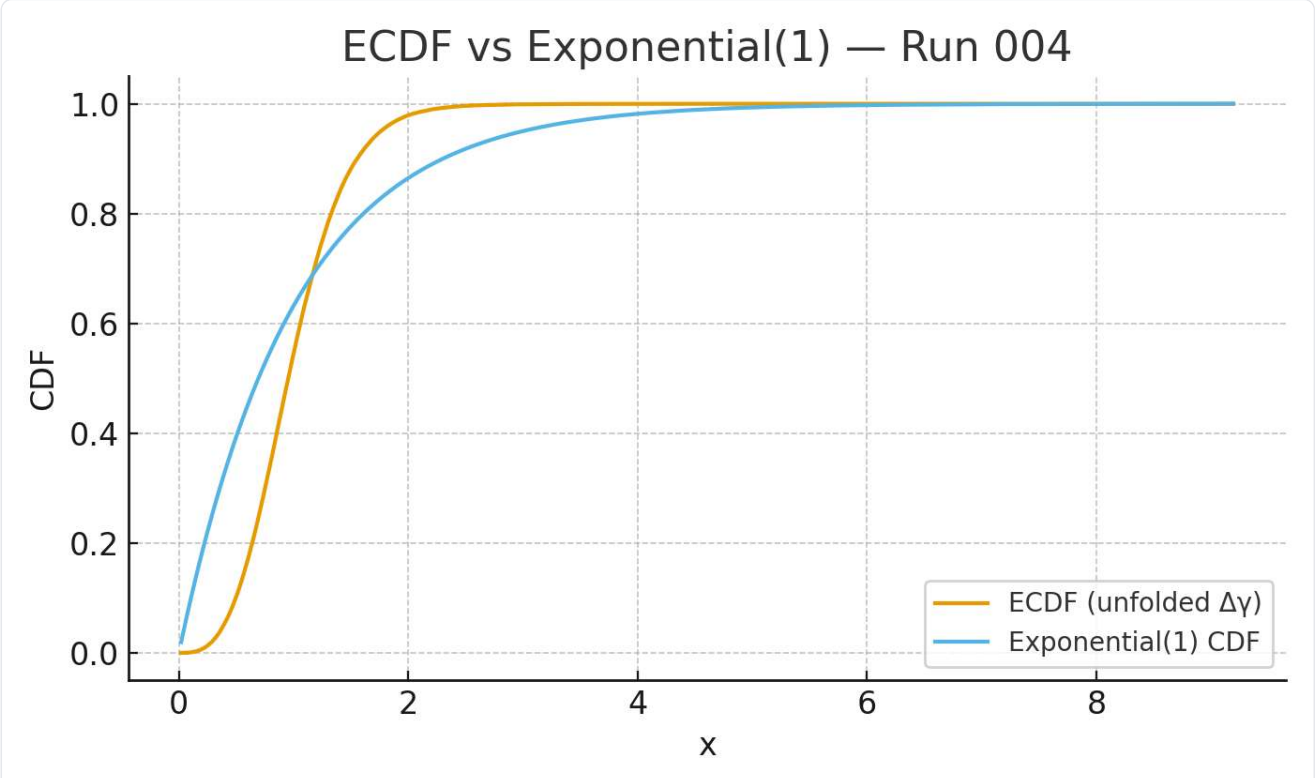


Figure: Q–Q vs Exp(1)

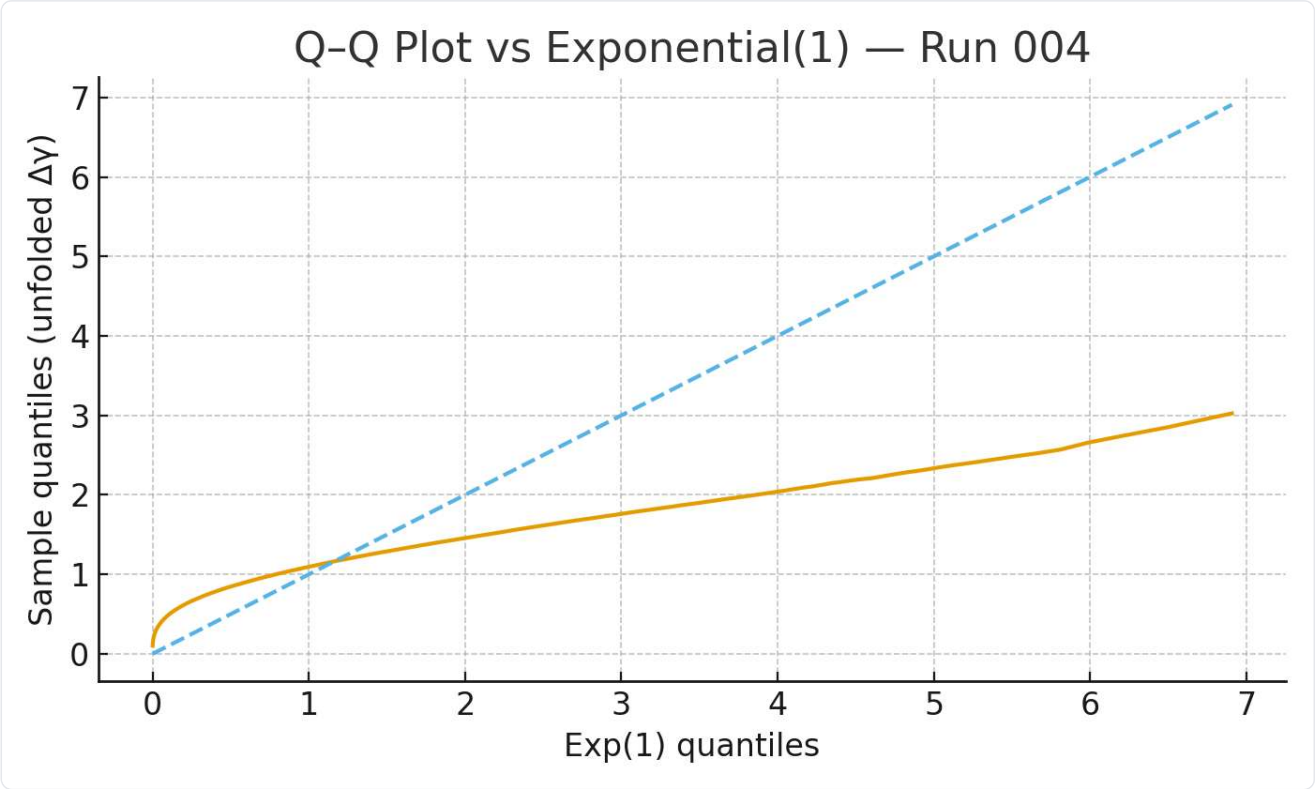


Figure: FFT overlap (bar by c)

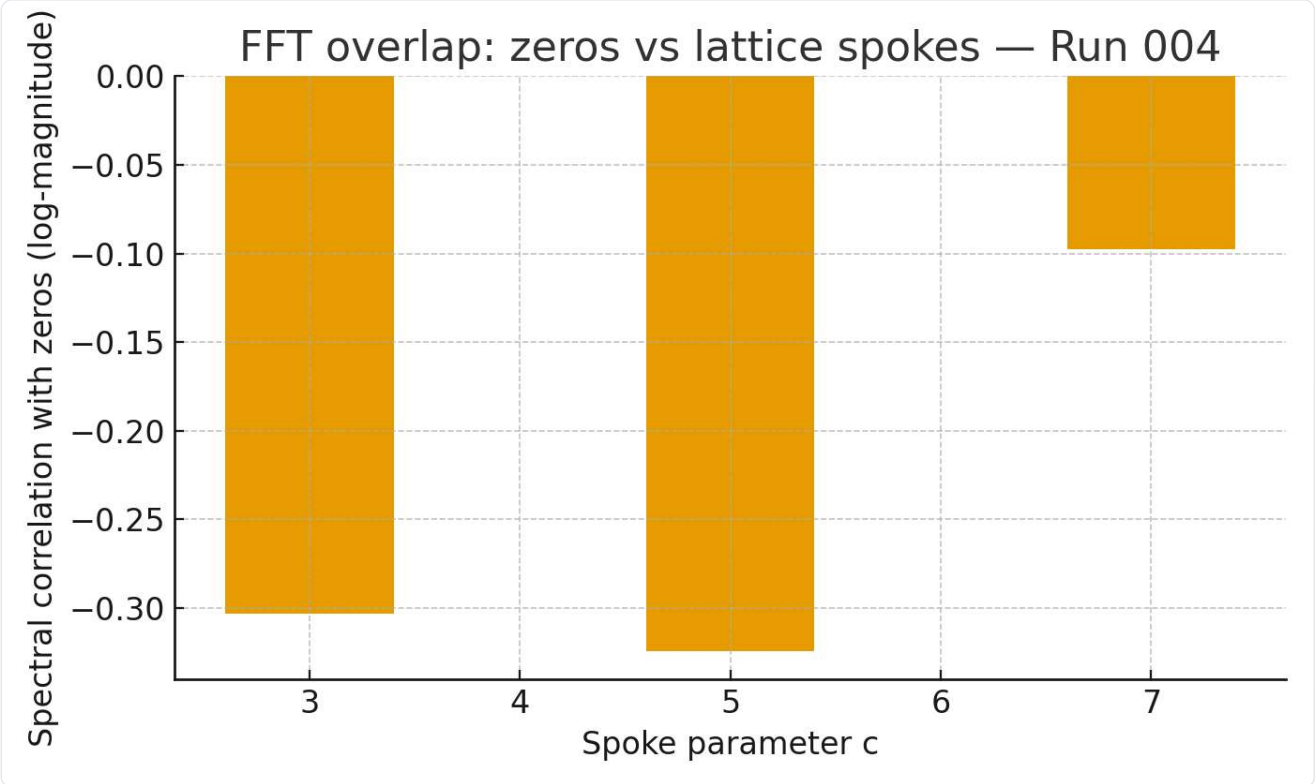
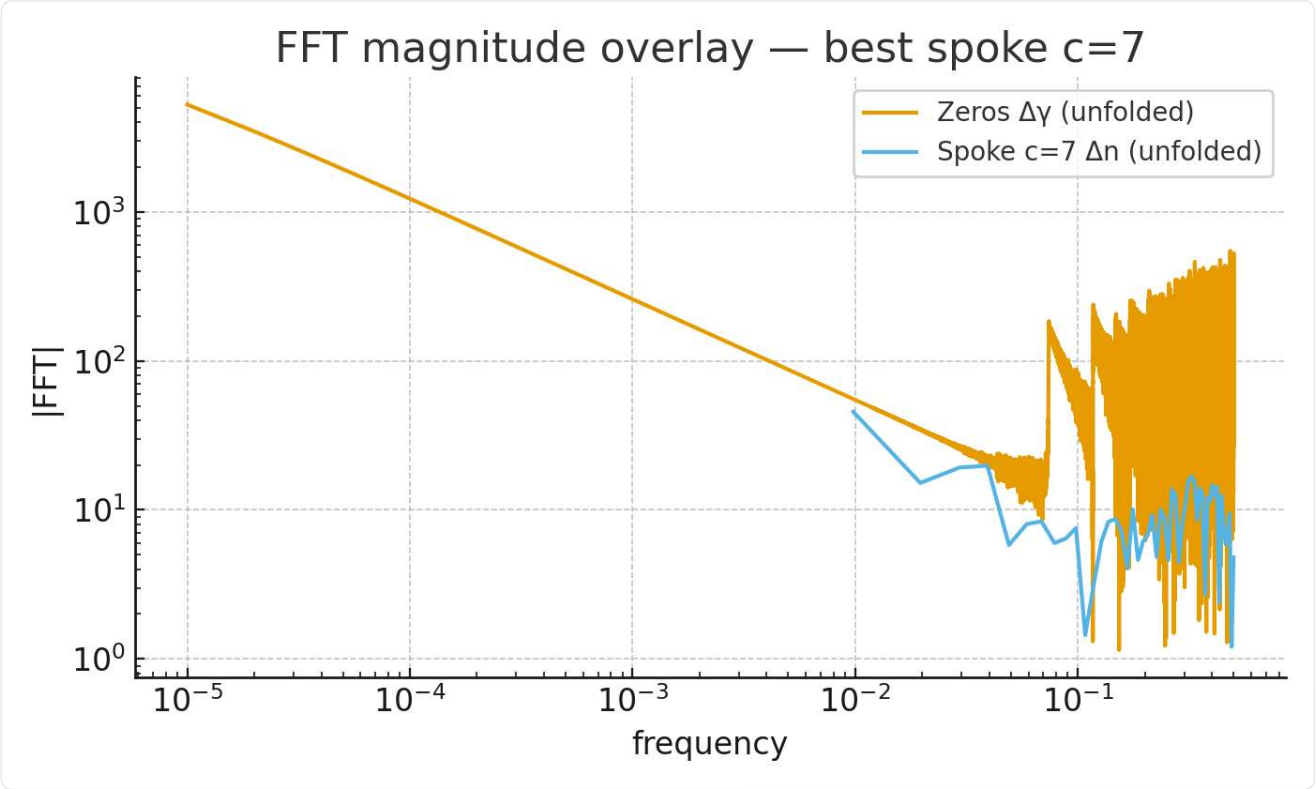


Figure: FFT magnitude overlay (best spoke)



Best here = highest correlation among c∈{3,5,7}, which is c=7.

Methods & Parameters

- **Zero set:** Odlyzko's `zeros6.gz`, first 100,000 ordinates.
- **Spacings:** nearest-neighbor Δγ; *unfolded* to mean 1.0 by dividing by raw mean.
- **GOF tests:** one-sample KS and Cramér–von Mises vs CDF $F(x)=1-e^{-x}$.
- **Lattice spokes:** $n(r)=3r^2-r+c$, c∈{2..7}; primes ≤ N_{max}=1,000,000 via sieve.
- **Spectral comparison:** magnitude of rFFT on zero-mean spacings; Pearson corr on $\log(1+|FFT|)$ after interpolating lattice spectrum onto zeros' frequency grid (excluding f=0).

Results

- **Unfolded spacings:** mean = 1.00000, σ ≈ 0.42925, var ≈ 0.18426 (n = 99,999).
- **KS vs Exp(1):** D ≈ 0.291643, p ≈ 0.0.
- **CvM vs Exp(1):** T ≈ 3050.068.
- **Lattice spokes (≤1e6):**
  - c=3 — 88 primes, 87 gaps; KS D≈0.1478 (p≈0.040), CvM≈0.5829, spec-corr≈−0.303.
  - c=5 — 84 primes, 83 gaps; KS D≈0.0929 (p≈0.453), CvM≈0.1955, spec-corr≈−0.324.
  - c=7 — 103 primes, 102 gaps; KS D≈0.1590 (p≈0.010), CvM≈0.4156, spec-corr≈−0.097.

Discussion

The unfolded zeros strongly reject Poisson spacing (as expected) under KS and CvM. First-pass spectral overlaps with AOL spokes at ≤1e6 are weak/negative; the "best" spoke (c=7) is still slightly negative. These results are consistent with non-Poisson repulsion for zeros and do not provide evidence of a simple lattice-spoke spectral match at this scale.

Limitations

- Spoke samples are small (≈80–100 spacings), limiting spectral power.
- Simple periodogram correlation is sensitive to interpolation and leakage; coherence or band-specific statistics would be more stable.
- GOF vs Exp(1) is a sanity check; a direct comparison to GUE/Wigner is planned for a follow-on run.

Metrics — inline preview

KS/CvM summary (JSON)

File: metrics/r004_ks_cvm_summary.json


```
{
  "raw mean spacing first100k": 0.7490744184827048,
  "ks D vs exp1": 0.2916428404789309,
  "ks p vs exp1": 0,
  "cvm_T_vs_exp1": 3050.068367113704
}
```


Lattice spokes metrics (CSV)

File: metrics/r004_lattice_metrics.csv

c	n_primes	n_spacings	mean	ks_D	ks_p	cvm_T	spec_corr
2	1	0					
3	88	87	1.0	0.1477870625117912	0.04014121085725448	0.582903018849898	-0.3033435540438732
4	0	0					
5	84	83	1.0000000000000002	0.09288220939924219	0.4525515628138149	0.19551432035338034	-0.3240356854460898
6	0	0					
7	103	102	0.9999999999999999	0.15896631367819897	0.010087269908838934	0.41555350558922294	-0.0973952627107658

Reproducibility — Python (end-to-end)

Pipeline mirrors Run 002/003; only run id and filenames change to `r004_*`.

Expand Copy

```
# Key differences for r004:
# run base = "run004"; output filenames use r004 *.
# Additional steps: KS/CvM vs Exp(1); lattice spokes (c=2..7) primes ≤ 1e6; FFT overlap correlation.
# (You can reuse the r003 code blocks, swapping run_base and adding the new analyses.)
```

Reproducibility — Python (Run 004)

Generates ECDF vs Exp(1), Q–Q vs Exp(1), FFT overlap bar, FFT overlay (best spoke), and all CSV/JSON metrics.

Environment

Copy

```
python -m venv .venv
source .venv/bin/activate  # Windows: .venv\Scripts\activate
pip install numpy pandas matplotlib
```

Script

Expand Copy

```
import gzip, os, json, math, numpy as np, pandas as pd
import matplotlib.pyplot as plt

run base = "run004"
zeros_path = "zeros6.gz"

fig dir = os.path.join(run base, "figures")
met dir = os.path.join(run base, "metrics")
data dir = os.path.join(run base, "data")
os.makedirs(fig dir, exist ok=True)
os.makedirs(met dir, exist ok=True)
os.makedirs(data_dir, exist_ok=True)
```

Reproducibility — PowerShell (Run 004, native)

Windows PowerShell 5+/PowerShell 7+. Uses .NET Charting for PNGs; naive DFT for spectra (dependency-free).

```
# run004_make_all.ps1
param(
  [string]$ZerosPath = "zeros6.gz",
  [string]$RunBase = "run004",
  [int]$Take = 100000,
  [int]$ParseCap = 120000,
  [int]$Nmax = 1000000
)
$ErrorActionPreference = "Stop"
Add-Type -AssemblyName System.IO.Compression.FileSystem
Add-Type -AssemblyName System.Windows.Forms
Add-Type -AssemblyName System.Windows.Forms.DataVisualization
```

Reproducibility — PHP (CLI, Run 004)

CLI usage: `php run004_make_all.php zeros6.gz` . Outputs CSV/JSON and `run004/report.html` with inline SVG figures.

```
<?php
// run004_make_all.php
$zeros = $argv[1] ?? 'zeros6.gz';
$run = 'run004';
@mkdir("$run/metrics", 0777, true);
@mkdir("$run/figures", 0777, true); // (unused by this PHP; figures go to HTML report)
@mkdir("$run/data", 0777, true);

// --- parse zeros (~120k lines); analyze first 100k
$vals = [];
$f = gzopen($zeros, 'r'); if(!$f) die("Cannot open $zeros\n");
while(!gzeof($f) && count($vals) < 120000){
  $line = trim(gzgets($f)); if($line==='') continue;
  $tok = preg_split('/\s+/', $line)[0];
  if(is_numeric($tok)) $vals[] = (float)$tok;
```

Reproducibility — C# (.NET 6 + ScottPlot, Run 004)

Setup

```
dotnet new console -o Run004Cs
cd Run004Cs
dotnet add package ScottPlot --version 5.0.19
```

Program.cs

```
using System.IO.Compression;
using ScottPlot;

string zerosPath = args.Length > 0 ? args[0] : "zeros6.gz";
string runBase = "run004";
Directory.CreateDirectory(Path.Combine(runBase,"metrics"));
Directory.CreateDirectory(Path.Combine(runBase,"figures"));
Directory.CreateDirectory(Path.Combine(runBase,"data"));

// Parse zeros (~120k lines), analyze first 100k
List<double> vals = new();
using (var fs = File.OpenRead(zerosPath))
```

Run

```
dotnet run -- zeros6.gz
```