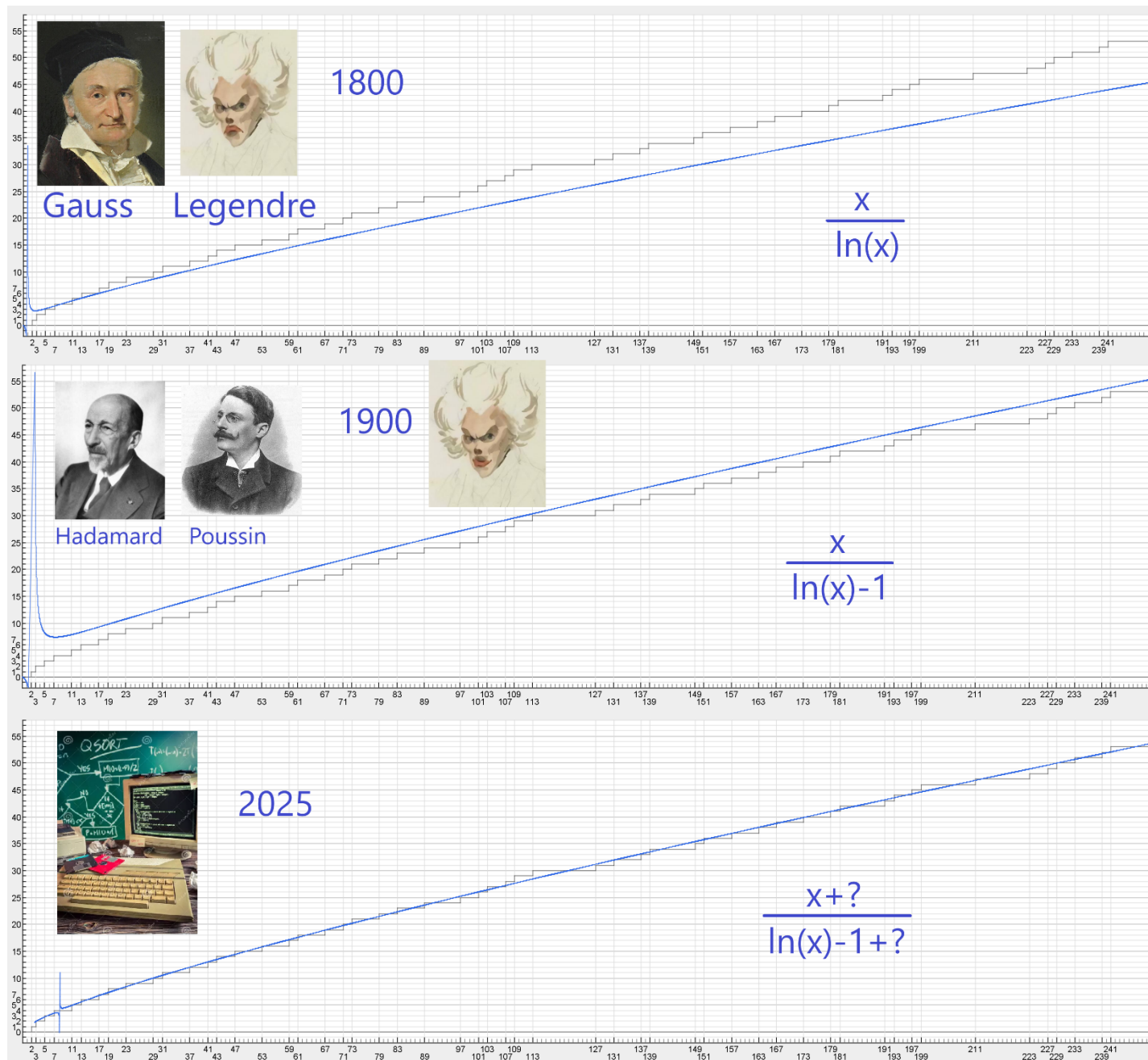


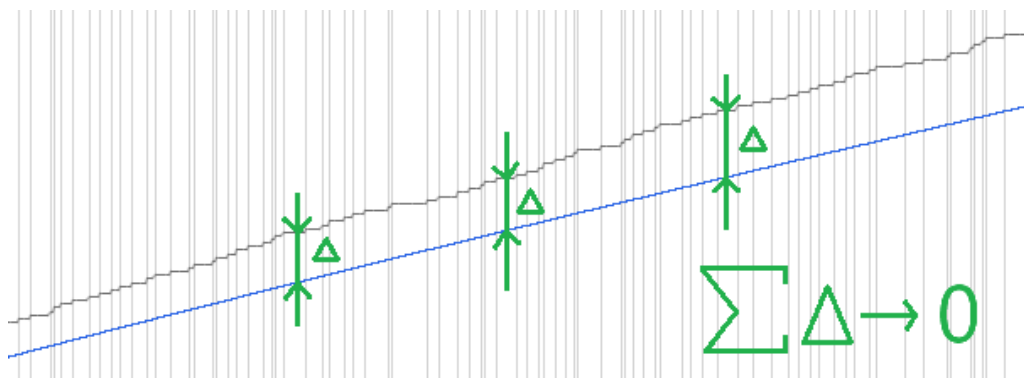
Prime number theorem

Subject: improving Gauss [prime-counting function](#) with a brute-force close research program.

Search for the arithmetic function « Logarithm asymptotic » $La(x)$ as an approximation of $\pi(x)$ or anti-derivative form of $Li(x)$.

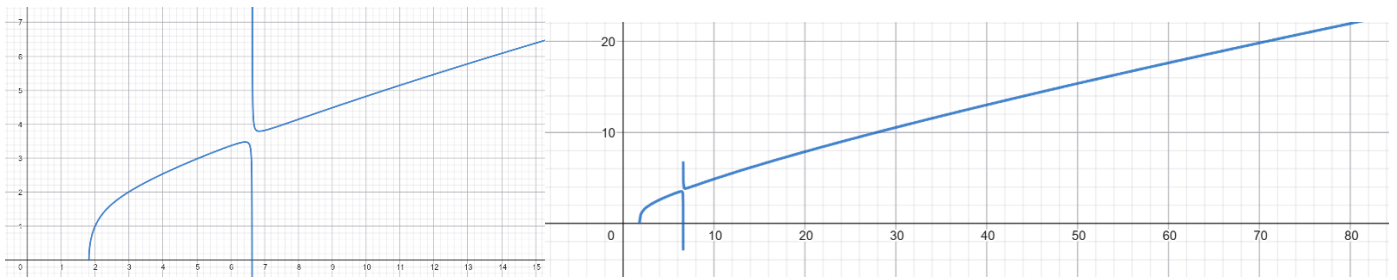


Methodology: Sum the deltas between prime stairs $\pi(x)$ and $La(x)$ and get the closest to zero.



Best result found after months of random search:

$$La(x) \approx \frac{x - 2.67 \sqrt{\frac{x}{\ln(x) - 1 + 1.23 \frac{\ln(x)}{x}}} + 0.35 \sqrt[3]{\ln(x)} - 1}{\ln(x) - \frac{1.11}{\ln(x) - 1 + 1.23 \frac{\ln(x)}{x}} - 1} \approx \pi(x)$$

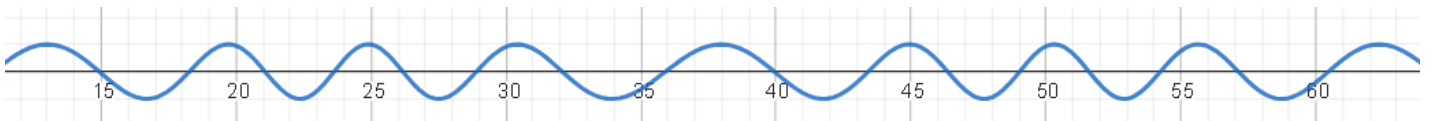


www.geogebra.org/graphing:

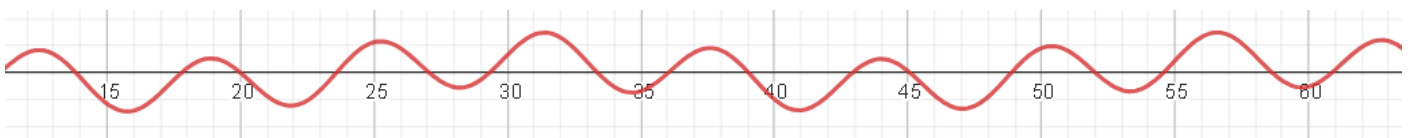
$$(x - 2.67 * \sqrt{x / (\ln(x) - 1 + 1.23 * \ln(x) / x)} + 0.35 * \ln(x)^{(2/3)} - 1) / (\ln(x) - 1.11 / (\ln(x) - 1 + 1.23 * \ln(x) / x) - 1)$$

The constants are approximative but the structure is clear-cut. The structure obviously suggests the primes repartition does not follow a random probability but a precise logarithm.

Conjuncture: we could use $la(x)$ to identify primes with the zeros of a trigonometric function, spring witch tends and relax at $la(x)$ pace could be a solution: $\cos(x + \cos(lax))$



Another one could be the skiing function: $\cos(x) + \cos(lax)$



Brute-force java program is open source here: [github](https://github.com)

$\pi(x)$ stair's dataset is generated in [Eratosthene](#) -> `pixRefMap()` and stored in [PrimesUtils](#).

$Li(x)$ values are used for $x > 10^{30}$ until $x = 10^{300}$.

Run [IdentifyLaxTest](#) to search for $La(x)$ function.

Run [PrimeTest9](#) to try to identify primes with a skiing function.

Run [PrimeTest11](#) to try to identify primes with a spring function.