

Asymptotiquement, autant de nombres premiers de 0 à n que de n à $2n$ en Lean, dvc, septembre 2026

```
-
import Mathlib

open Filter Topology Asymptotics

/-- Théorème des nombres premiers, pris ici comme hypothèse :
 $\pi(n) \sim n / \log n$ . (À remplacer par le nom exact dans Mathlib si disponible.) -/
def PNT : Prop :=
  (fun n : ℕ => (Nat.primeCounting n : ℝ)) ~[atTop] fun n : ℕ => (n : ℝ) / Real.log n

/-- Asymptotiquement,  $(\pi(2n) - \pi(n)) / \pi(n) \rightarrow 1$  quand  $n \rightarrow +$ . -/
theorem prime_interval_equivalence (h : PNT) :
  Tendsto (fun n : ℕ =>
    ((Nat.primeCounting (2 * n) : ℝ) - Nat.primeCounting n) / Nat.primeCounting n)
    atTop (nhds 1) := by
-- 1.  $\pi(2n) \sim 2n / \log(2n)$ 
have h2n : Tendsto (fun n : ℕ => 2 * n) atTop atTop :=
  tendsto_atTop_mono (fun n : ℕ => (by omega : n ≤ 2 * n)) tendsto_id
have h2 : (fun n : ℕ => (Nat.primeCounting (2 * n) : ℝ)) ~[atTop]
  fun n : ℕ => ((2 * n : ℕ) : ℝ) / Real.log ((2 * n : ℕ) : ℝ) :=
  h.comp_tendsto h2n
-- 2.  $\pi(2n) / \pi(n) \sim (2n / \log 2n) / (n / \log n)$ 
have hdiv := h2.div h
-- 3. ce dernier quotient tend vers 2
have hlog : Tendsto (fun n : ℕ => Real.log n) atTop atTop :=
  Real.tendsto_log_atTop.comp tendsto_natCast_atTop_atTop
have hz : Tendsto (fun n : ℕ => Real.log 2 / Real.log n) atTop (nhds 0) :=
  tendsto_const_nhds.div_atTop hlog
have hlim : Tendsto (fun n : ℕ => 2 / (1 + Real.log 2 / Real.log n))
  atTop (nhds 2) := by
  have h1 : Tendsto (fun n : ℕ => 1 + Real.log 2 / Real.log n)
    atTop (nhds (1 + 0)) :=
    tendsto_const_nhds.add hz
  have h3 := (tendsto_const_nhds (x := (2 : ℝ))).div h1 (by norm_num)
  rw [show (2 : ℝ) / (1 + 0) = 2 by norm_num] at h3
  exact h3
have heq : Filter.Eventually (fun n : ℕ =>
  2 / (1 + Real.log 2 / Real.log n) =
  (((2 * n : ℕ) : ℝ) / Real.log ((2 * n : ℕ) : ℝ)) / ((n : ℝ) / Real.log n))
  atTop := by
  filter_upwards [eventually_ge_atTop 2] with n hn
  have hn0 : (0 : ℝ) < n := by exact_mod_cast (by omega : 0 < n)
```

```

have hn1 : (1 : ℝ) < n := by exact_mod_cast (by omega : 1 < n)
have h1 : 0 < Real.log n := Real.log_pos hn1
have h2pos : 0 < Real.log 2 := Real.log_pos (by norm_num)
push_cast
rw [Real.log_mul (by norm_num) hn0.ne']
field_simp
ring
have hlim' : Tendsto (fun n : ℕ =>
  (((2 * n : ℕ) : ℝ) / Real.log ((2 * n : ℕ) : ℝ)) / ((n : ℝ) / Real.log n))
  atTop (nhds 2) :=
  hlim.congr' heq
-- 4. donc  $\pi(2n) / \pi(n) \rightarrow 2$ 
have hratio : Tendsto (fun n : ℕ =>
  (Nat.primeCounting (2 * n) : ℝ) / Nat.primeCounting n) atTop (nhds 2) :=
  hdiv.symm.tendsto_nhds hlim'
-- 5.  $(\pi(2n) - \pi(n)) / \pi(n) = \pi(2n)/\pi(n) - 1$  dès que  $n \geq 2$ 
have hsub : Tendsto (fun n : ℕ =>
  (Nat.primeCounting (2 * n) : ℝ) / Nat.primeCounting n - 1) atTop (nhds 1) := by
  have h4 := hratio.sub (tendsto_const_nhds (x := (1 : ℝ)))
  rw [show (2 : ℝ) - 1 = 1 by norm_num] at h4
  exact h4
refine hsub.congr' ?_
filter_upwards [eventually_ge_atTop 2] with n hn
have hpos : 0 < Nat.primeCounting n :=
  lt_of_lt_of_le (by decide : 0 < Nat.primeCounting 2) (Nat.monotone_primeCounting hn)
have : (Nat.primeCounting n : ℝ) ≠ 0 := by exact_mod_cast hpos.ne'
field_simp

```