Il più grande mistero della Matematica

We may - paraphrasing the famous sentence of George Orwell - say that “all mathematics is beautiful, yet some is more beautiful than the other”. But the most beautiful in all mathematics is the Zeta function. There is no doubt about it. (Krzysztof Maslanka)

La serie armonica

Sia

$$H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}$$

ricordando che

$$\frac{1}{t} = \frac{d}{dt} \ln t$$

vale (la dimostrazione non è immediata)

$$H(n) > \ln(n+1)$$

e quindi H(n) diverge, inoltre H(n) cresce proprio come ln n e vale il limite fondamentale

$$\lim_{n \to \infty} H(n) - \ln n = \gamma$$

dove

$$\gamma = 0.57721566490153286060651209008240243104215933593992359880...$$

è la costante di Eulero-Mascheroni (si congettura che sia irrazionale e addirittura trascendente ma nessuno lo ha ancora dimostrato).

I numeri primi

Dimostrazione di Euclide che che i numeri primi sono infiniti

I numeri primi sono quei numeri che non possono essere scomposti in prodotto di fattori minori

A = {2, 3, 5, 7, 11, 13, 17, 19, 23, 29, ... 37,... 317, ...}.

Dimostriamo che esistono infiniti numeri primi e cioè che la successione A non termina mai.

Supponiamo che A abbia fine e che {2, 3, 5, …, p} rappresenti la successione completa dei numeri primi (per cui p risulta il massimo numero primo). Con questa ipotesi consideriamo il numero q definito dalla formula q = (2 3 5 … p) +1. E' evidente che q non è divisibile per nessuno dei numeri 2, 3, 5,…, p, perché il resto della divisione per ognuno di questi numeri sarà sempre 1. Ma, allora o q stesso è un numero primo, oppure esso è divisibile per qualche primo che supera tutti quelli di A. Questo contraddice l'ipotesi che non esista un numero primo maggiore di p, e perciò l'ipotesi che A abbia fine è falsa.

Dimostrazione di Eulero che i numeri primi sono infiniti

$$\lim_{n \to \infty} H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots =$$

$$= 1 + \sum_{1 \le i}^{\infty} \frac{1}{p_i} + \sum_{1 \le i \le j}^{\infty} \frac{1}{p_i p_j} + \sum_{1 \le i \le j \le k}^{\infty} \frac{1}{p_i p_j p_k} + \ldots =$$

$$= \left(1 + \frac{1}{p_1} + \frac{1}{p_1^2} + \ldots + \frac{1}{p_1^k} + \ldots\right)\left(1 + \frac{1}{p_2} + \frac{1}{p_2^2} + \ldots + \frac{1}{p_2^k} + \ldots\right)\ldots =$$

$$= \left[\sum_{j=0}^{\infty}\left(\frac{1}{p_1}\right)^j\right]\left[\sum_{j=0}^{\infty}\left(\frac{1}{p_2}\right)^j\right]\ldots\left[\sum_{j=0}^{\infty}\left(\frac{1}{p_k}\right)^j\right]\ldots =$$

$$= \frac{1}{1-\frac{1}{p_1}}\frac{1}{1-\frac{1}{p_2}}\ldots\frac{1}{1-\frac{1}{p_k}}\ldots = \prod_{k=1}^{\infty}\frac{1}{1-\frac{1}{p_k}} = \infty$$

Il primo termine diverge e i termini della produttoria sono tutti finiti (nel senso che sono numeri razionali), quindi la produttoria finale deve contenere infiniti termini (nel senso che non sono in numero finito). NB si noti la leggera ambiguità linguisrtica.

Questa dimostrazione dice qualcosa di più rispetto a quella di Euclide. Tenendo presente che H(n) va all'infinito come log n prendendo il logaritmo del primo e dell'ultimo termine della catena di uguaglianze si ha una espessione che Eulero scriveva come

$$\sum_{p \text{ primo}} \frac{1}{p} = \log(\log \infty)$$

e che può essere interpretata con maggiore precisione come

$$\sum_{p<x} \frac{1}{p} \approx \log(\log x)$$

formando il primo seme del Teorema dei numeri primi.

La distribuzione dei primi

La funzione π(x) rappresenta il numero di primi fino ad x, non esiste una formula chiusa semplice per π(x) ma ne esistono molte approssimazioni. Per esempio il Teorema dei numeri primi afferma che

$$\pi(x) \approx \frac{x}{\ln x}$$

(ovviamente questo non significa che la differenza tra i due termini vada a zero).

Il Teorema dei numeri primi fu congetturato da Gauss nel 1792 (a 15 anni!) e dimostrato oltre un secolo dopo (1896) da Hadamard e De la Vallée Poussin.

Gauss trovò una forma più precisa del teorema ovvero

$$\pi(x) \approx Li(x)$$

dove

$$Li(x) = \int_2^x \frac{u}{\ln u}\,du \approx \frac{x}{\ln x} + \frac{x}{(\ln x)^2} + \ldots + \frac{(k-1)!x}{(\ln x)^k} + \ldots$$

NB questo è uno sviluppo asintotico (bestia brutta e cattiva) in cui la serie non converge ma ogni troncamento finito con n termini è una approssimazione migliore di quelle con meno di n termini.

Il calcolo dei numeri primi

Un programma scemo

Si può pensare di calcolare tutti i numeri primi compresi tra 2 e n applicando brutalmente la definizione vista sopra. Il programma funziona nel modo seguente: dapprima si crea un *array* di n valori di booleani, assegnando `false` a 0 ed 1 e `true` a tutti gli altri numeri. A questo punto si prova a dividere ogni numero i maggiore di 2 per tutti i numeri precedenti. Se si trova che i è divisibile per un j minore di i, allora i non è primo e nella posizione corrispondente della tabella viene messo il valore `false`.

```
int primiscemo(int n) {
      boolean[] primi = new boolean[n];
      primi[0] = primi[1] = false;
      for (int i = 2; i < n; i++) {
         primi[i] = true;
         for (int j = 2; j < i; j++) if (i % j == 0) {
               primi[i] = false; break;
         }
      }
      int p = 0;
      for (int j = 2; j < n; j++) if (primi[j]) p++;
      return p;
}
```

Finito il giro, i numeri primi sono tutti e soli quelli che hanno conservato il valore `true` nella posizione corrispondente; è facile quindi scorrere la tabella e contarli.

Un programma appena meno scemo

Il programma di cui sopra può essere migliorato applicando un semplice ragionamento matematico. Se un numero non è primo allora è divisibile per un numero primo minore o uguale alla sua radice quadrata.

```
int primi2(int n) {
      boolean[] primi = new boolean[n];
      primi[0] = primi[1] = false;
      for (int i = 2; i < n; i++) {
         primi[i] = true;
         for (int j=2; j*j<=i; j++)if (primi[j] && i%j == 0) {
               primi[i] = false; break;
            }
      }
      int p = 0;
      for (int j = 2; j < n; j++)if(primi[j]) p++;
      return p;
   }
```

Quindi, quando si prova a dividere i per j, è sufficiente considerare solo quei numeri j che sono primi e tali che $j \times j \leq i$.

Un altro passo avanti consiste nel provare la primalità solo dei numeri dispari

Un programma sofisticato

I programmi visti fino ad ora sono molto lenti, esiste invece, noto fin dall'antichità, un algoritmo più sofisticato ma estremamente efficiente per il calcolo dei numeri primi. Si tratta del cosiddetto **Crivello di Eratostene**. L'idea, geniale nella sua semplicità, consiste nel creare un array di tutti i potenziali primi (come nei due casi precedenti) e poi eliminare tutti i multipli di due (marcando `false` un elemento sì e uno no); a questo punto si eliminano i multipli di 3 (marcando `false` un elemento sì e due no); si cerca il prossimo numero non marcato che sarà certamente primo (dopo 3 si trova 5 perché 4 è stato eliminato) e si tolgono i suoi multipli e così via utilizzando i primi fino alla radice della lunghezza dell'*array*. Vediamone un esempio manuale di funzionamento.

Si scrive dapprima una tabella con tutti i numeri da 2 a n (100 nel nostro caso):

```
     2   3   4   5   6   7   8   9  10
11  12  13  14  15  16  17  18  19  20
21  22  23  24  25  26  27  28  29  30
31  32  33  34  35  36  37  38  39  40
41  42  43  44  45  46  47  48  49  50
51  52  53  54  55  56  57  58  59  60
61  62  63  64  65  66  67  68  69  70
71  72  73  74  75  76  77  78  79  80
81  82  83  84  85  86  87  88  89  90
91  92  93  94  95  96  97  98  99 100
```

Il numero 2 è primo, si cancellano quindi tutti i multipli di 2, a partire da 4.

	2	3	~~4~~	5	~~6~~	7	~~8~~	9	~~10~~
11	~~12~~	13	~~14~~	15	~~16~~	17	~~18~~	19	~~20~~
21	~~22~~	23	~~24~~	25	~~26~~	27	~~28~~	29	~~30~~
31	~~32~~	33	~~34~~	35	~~36~~	37	~~38~~	39	~~40~~
41	~~42~~	43	~~44~~	45	~~46~~	47	~~48~~	49	~~50~~
51	~~52~~	53	~~54~~	55	~~56~~	57	~~58~~	59	~~60~~
61	~~62~~	63	~~64~~	65	~~66~~	67	~~68~~	69	~~70~~
71	~~72~~	73	~~74~~	75	~~76~~	77	~~78~~	79	~~80~~
81	~~82~~	83	~~84~~	85	~~86~~	87	~~88~~	89	~~90~~
91	~~92~~	93	~~94~~	95	~~96~~	97	~~98~~	99	~~100~~

Il numero 3 risulta quindi primo e si cancellano tutti i suoi multipli a partire da 9.

	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	25	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	35	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	49	~~50~~
~~51~~	~~52~~	53	~~54~~	55	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	65	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	77	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	85	~~86~~	~~87~~	~~88~~	89	~~90~~
91	~~92~~	~~93~~	~~94~~	95	~~96~~	97	~~98~~	~~99~~	~~100~~

Il numero 5 risulta primo e si provvede a cancellare tutti i suoi multipli a partire da 25.

	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	35	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	49	~~50~~
~~51~~	~~52~~	53	~~54~~	55	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	65	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	77	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	85	~~86~~	~~87~~	~~88~~	89	~~90~~
91	~~92~~	~~93~~	~~94~~	95	~~96~~	97	~~98~~	~~99~~	~~100~~

Il 7 è primo e si cancellano tutti i suoi multipli a partire da 49.

	2	3	~~4~~	5	~~6~~	7	~~8~~	~~9~~	~~10~~
11	~~12~~	13	~~14~~	~~15~~	~~16~~	17	~~18~~	19	~~20~~
~~21~~	~~22~~	23	~~24~~	~~25~~	~~26~~	~~27~~	~~28~~	29	~~30~~
31	~~32~~	~~33~~	~~34~~	35	~~36~~	37	~~38~~	~~39~~	~~40~~
41	~~42~~	43	~~44~~	~~45~~	~~46~~	47	~~48~~	~~49~~	~~50~~
~~51~~	~~52~~	53	~~54~~	55	~~56~~	~~57~~	~~58~~	59	~~60~~
61	~~62~~	~~63~~	~~64~~	65	~~66~~	67	~~68~~	~~69~~	~~70~~
71	~~72~~	73	~~74~~	~~75~~	~~76~~	~~77~~	~~78~~	79	~~80~~
~~81~~	~~82~~	83	~~84~~	85	~~86~~	~~87~~	~~88~~	89	~~90~~
~~91~~	~~92~~	~~93~~	~~94~~	95	~~96~~	97	~~98~~	~~99~~	~~100~~

Infine, poiché 11×11 è maggiore di 100, abbiamo finito e i sopravvissuti sono i numeri primi minori di 100.

Implementazione del Crivello di Eratostene

```
boolean[] primi = new boolean[n];


public int crivello(){
   primi[0] = primi[1] = false;
   for (int i = 2; i < n; i++)primi[i] = true;
   int i = 1;
   while (i * i < n) {
      while (!primi[++i]) {}
      for (int k = i * i; k < n; k += i) primi[k] = false;
   }
   int p = 0;
   for (int j = 2; j < n; j++)if (primi[j]) p++;
   return p;

}
```

Occupazione di Memoria

Tutti i programmi visti finora possono memorizzare l'insieme di primi in un array di `boolean`, la cosa è molto veloce e semplice ma costosa (`n` byte per arrivare fino ad `n`) e il traguardo di `n` = 4.000.000.000 è irraggiungibile.

Proviamo ad usare invece la classe **`java.util.BitSet`**

- l'uso dei metodi specifici di `BitSet` semplifica molto l'implementazione
- l'occupazione di memoria è n/8 (se n=4.000.000.000 bastano 500 MegaByte).
- in ogni caso metà della memoria è sprecata in quanto si rappresentano anche i pari.

```
BitSet primi = new BitSet(n);

public int CrivelloB(){
      primi.set(2, n);
      int i = 1;
      while (i * i < n) {
         i = primi.nextSetBit(i + 1);
         for (int k = i * i; k < n; k += i) primi.clear(k);
      }
      return primi.cardinality();
   }
```

Ecco la classe `PrimeSet2` che non memorizza i numeri pari e che usa `n long`

```
import java.util.BitSet;

public class PrimeSet2 {
   private BitSet primeset;

   public PrimeSet2(long n) {
      this.primeset = new BitSet((int) (n / 2));
   }

   public void set(long fromIndex, long toIndex) {
      for (long i = fromIndex; i < toIndex; i++)
         this.set(i);
   }

   public long nextSetBit(long fromIndex) {
         long i = fromIndex;
      while (!this.get(i++)) {
      }
      return i-1;
   }

   public int cardinality() {
      return primeset.cardinality() + 1;
   }
```

```
    public void set(long i) {
        if (i % 2 == 0) return;
        primeset.set((int) i / 2);
    }

    public void clear(long i) {
        if (i % 2 == 0) return;
        primeset.clear((int) i / 2);
    }

    public boolean get(long i) {
        if (i % 2 == 0) return false;
        return primeset.get((int) i / 2);
    }
}
```

Ed ecco il crivello corrispondente

```
PrimeSet2 primi = new PrimeSet2(n);

public int CrivelloP2 (){
    primi.set(2, n);
    long i = 3;
    while (i * i < n) {
        i = primi.nextSetBit(i + 1);
        for (long k = i * i; k < n; k += i) primi.clear(k);
    }
    return primi.cardinality();
}
```

Risultati

Ecco i risultati dei vari programmi su un Power Mac Intel a 3 Ghz.

n	primiscemo	primi2	crivello	crivelloP2
10^5	2.3"/0.6M	0.02"/0.6M	0.004"/0.6M	0.02"/0.6M
10^6	165"/1.4M	0.34"/1.4M	0.008"/1.4M	0.02"/0.6M
10^7	—	5.9"/10M	0.36"/10M	0.1"/0.9M
10^8	—	—	5.24"/96M	1.8"/3.7M
10^9	—	—	58.4"/955M	28"/32M
$4\ 10^9$	—	—	—	125"/127M

La funzione Zeta di Riemann

Definizione

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}, \ \mathrm{Re}(s) > 1 \qquad (1)$$

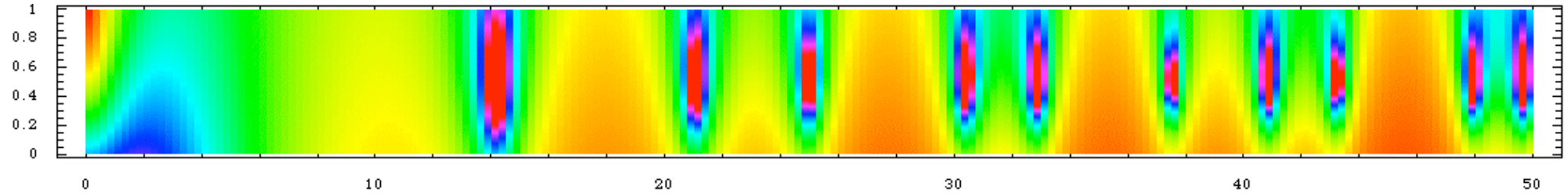
Per Re(s) ≤ 1, la serie diverge, per Re(s) > 1, ζ(s) è una funzione analitica. ζ(s) può essere estesa per continuazione analitica ad una funzione meromorfica definita su tutto il piano complesso con l'eccezione di s=1 dove ζ(1) coincide con il limite della serie armonica e ha un polo con residuo 1.

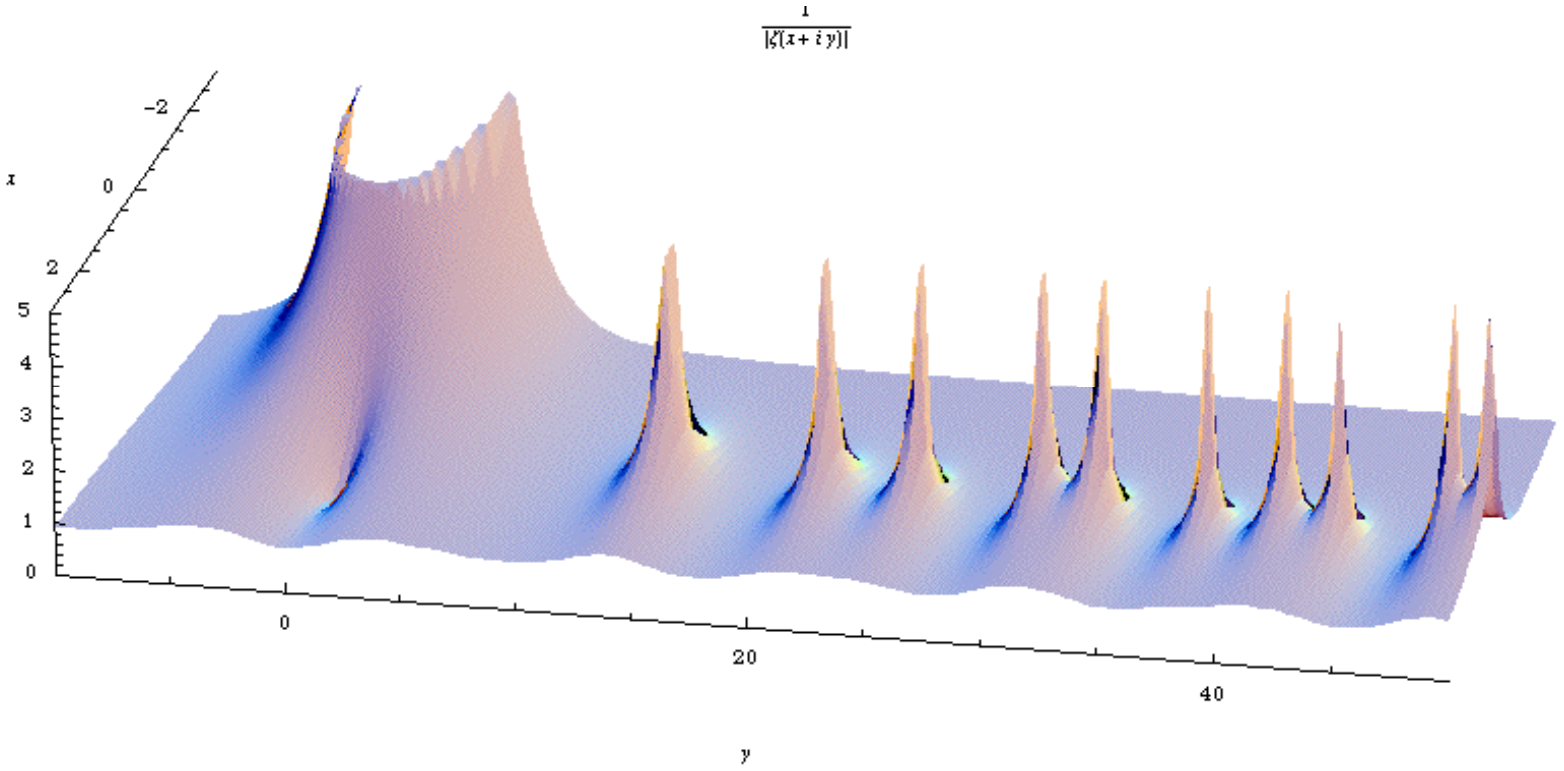
Attenzione se Re(s) ≤ 1 la ζ(s) non è più data dalla serie (1) ma, per esempio, da un'opportuna serie di Laurent (γ è la costante di Eulero-Mascheroni, gli altri γ_n le costanti di Stieltjes).

$$\zeta(s) = \frac{1}{s-1} + \gamma + \sum_{n=1}^{\infty} \frac{(-1)^n}{n!} \gamma_n (s-1)^n$$

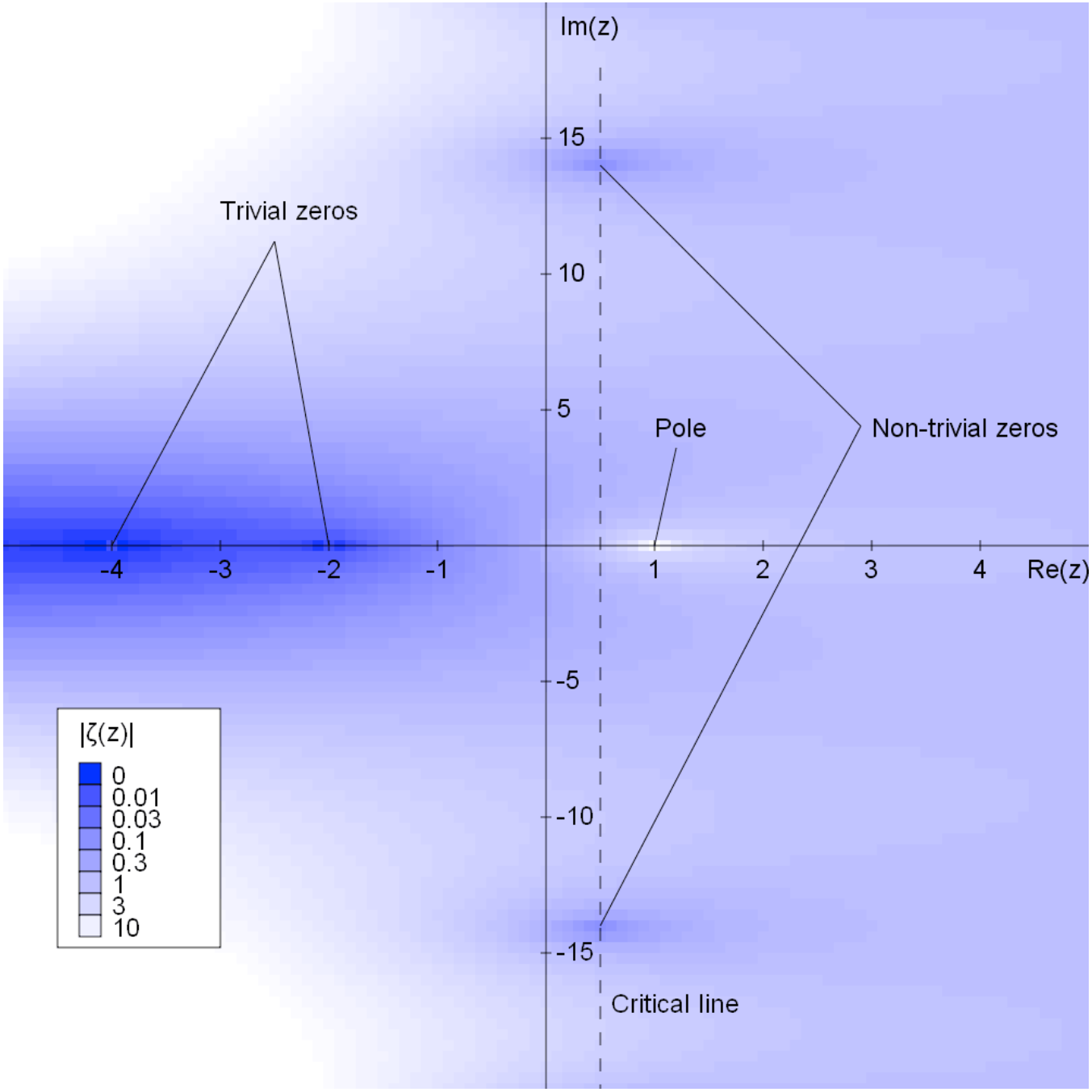
ζ(s) si annulla per s = -2, -4, ..., -2n, questi sono detti zeri banali.

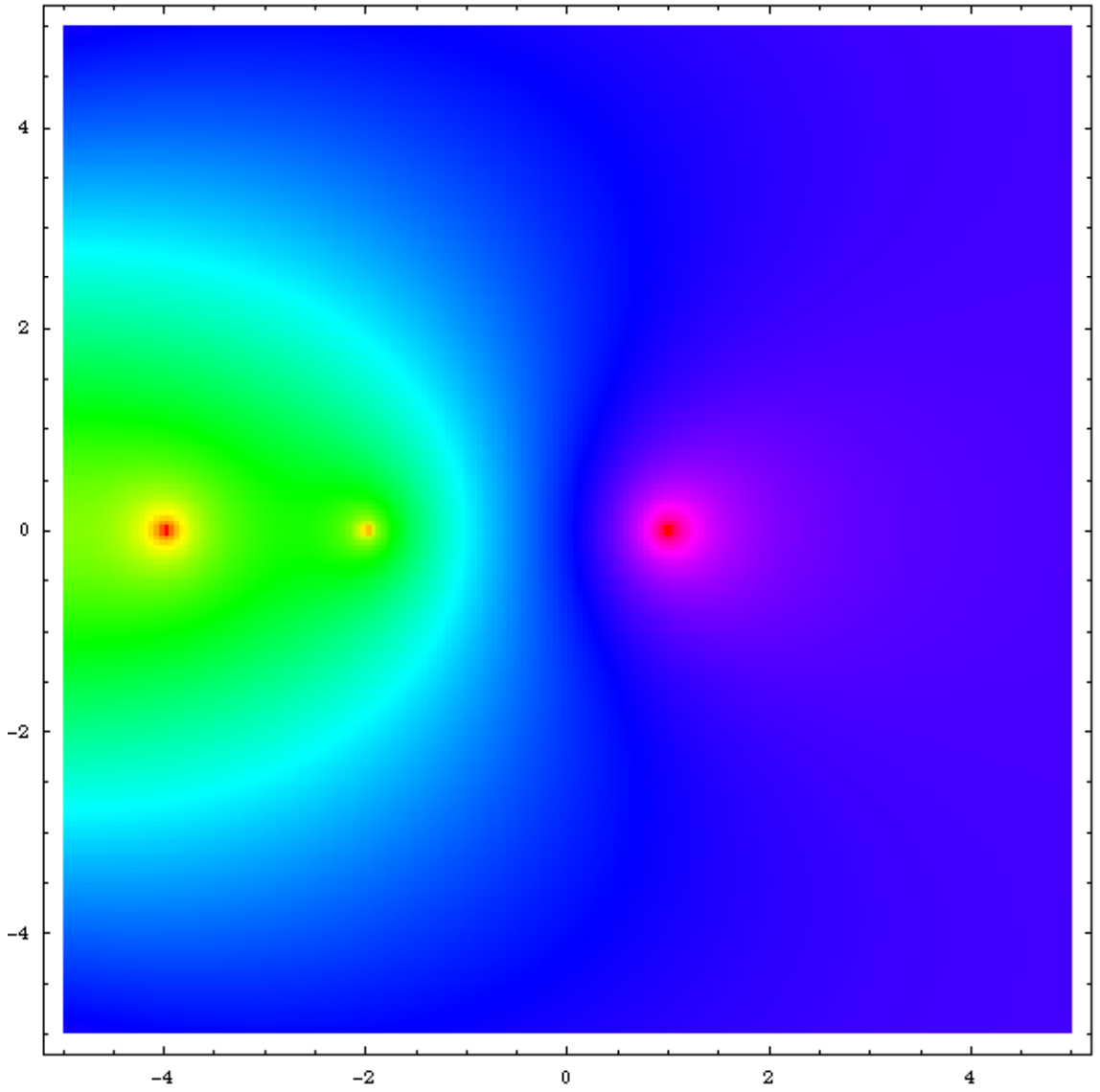
Si dimostra che tutti gli altri zeri (non banali) sono nella striscia $0 < Re(s) < 1$. In particolare vi sono infiniti zeri sulla linea critica $Re(s)=1/2$.





Qualche altro grafico della $\zeta(s)$




4
2
0
-2
-4
-4
-2
0
2
4

Eulero aveva studiato la sommatoria (1) un secolo prima di Riemann e aveva notato una importante connessione tra questa funzione e la sequenza dei numeri primi.

$$\zeta(s) = \prod_{p \text{ primo}} \frac{1}{1 - p^{-s}}, \quad \mathrm{Re}(s) > 1$$

Dimostrazione

$$\prod_{k=1}^{\infty} \frac{1}{1 - p_k^{-s}} = \frac{1}{1 - p_1^{-s}} \frac{1}{1 - p_2^{-s}} \cdots \frac{1}{1 - p_k^{-s}} \cdots =$$

$$= \left[\sum_{j=0}^{\infty} p_1^{-sj}\right]\left[\sum_{j=0}^{\infty} p_2^{-sj}\right] \cdots \left[\sum_{j=0}^{\infty} p_k^{-sj}\right] \cdots =$$

$$= \left(1 + \frac{1}{p_1^s} + \frac{1}{p_1^{2s}} + \ldots + \frac{1}{p_1^{ks}} + \ldots\right)\left(1 + \frac{1}{p_2^s} + \frac{1}{p_2^{2s}} + \ldots + \frac{1}{p_2^{ks}} + \ldots\right) \cdots =$$

$$= 1 + \sum_{1 \le i}^{\infty} \frac{1}{p_i^s} + \sum_{1 \le i \le j}^{\infty} \frac{1}{p_i^s p_j^s} + \ldots + \sum_{1 \le i \le j \le k}^{\infty} \frac{1}{p_i^s p_j^s p_k^s} + \ldots =$$

$$= 1 + \frac{1}{2^s} + \frac{1}{3^s} + \ldots \frac{1}{k^s} + \ldots = \zeta(s)$$

Ipotesi di Riemann

Tutti gli zeri non banali sono sulla linea critica.

Non è stata dimostrata ma risulta verificata per i primi 10.000 miliardi di zeri.

È il problema aperto più importante della matematica, ha infiniti legami con la teoria dei numeri (in particolare con la distribuzione dei primi), con la fisica quantistica, con la teoria della relatività, con problemi di algoritmica statistica, di crittografia, ecc.

Per dimostrarla falsa, basterebbe trovare uno zero non banale fuori dalla linea critica ½ + i t.

Il calcolo degli zeri della funzione di Riemann sulla linea critica

Year	n	Author
1903	15	J. P. Gram [6]
1914	79	R. J. Backlund [1]
1925	138	J. I. Hutchinson [7]
1935	1,041	E. C. Titchmarsh [22]
1953	1,104	A. M. Turing [24]
1956	15,000	D. H. Lehmer [12]
1956	25,000	D. H. Lehmer [11]
1958	35,337	N. A. Meller [14]
1966	250,000	R. S. Lehman [10]
1968	3,500,000	J. B. Rosser, J. M. Yohe, L. Schoenfeld [21]
1977	40,000,000	R. P. Brent [2]
1979	81,000,001	R. P. Brent [3]
1982	200,000,001	R. P. Brent, J. van de Lune, H. J. J. te Riele, D. T. Winter [25]
1983	300,000,001	J. van de Lune, H. J. J. te Riele [8]
1986	1,500,000,001	J. van de Lune, H. J. J. te Riele, D. T. Winter [9]
2001	10,000,000,000	J. van de Lune (unpublished)
2004	900,000,000,000	S. Wedeniwski [26]
2004	10,000,000,000,000	X. Gourdon and Patrick Demichel [5]

Tabella dei record

La cosa importante non è tanto calcolare molti zeri sulla linea critica quanto dimostrare che non ve ne sono altri. Per fare questo lo schema classico del calcolo, sino dai tempi di Turing (pur con infinite varianti) è organizzato come segue.

Si imposta un algoritmo per il calcolo della funzione Z(t) (la funzione di Riemann-Siegel) che è reale, analitica e per cui vale

$$|Z(t)| = \left|\zeta(\frac{1}{2} + it)\right|, t \text{ reale}$$

Si calcola il numero di zeri N(T) di $\zeta(x+iy)$ nel rettangolo $-1 < x < 2$, $-iT < i\,y < iT$, per esempio con l'integrale chiuso sui bordi del rettangolo R.

$$N(t) - 1 = \frac{1}{2\pi i}\int_{\partial R} \frac{\zeta'(s)}{\zeta(s)} ds$$

Poiché Z(t) è pari, si cercano esattamente N(T)/2 intervalli di separazione di Z(t) tra 0 e T, se si ha successo l'ipotesi è verificata fino a T.

Ancora sulla distribuzione dei primi

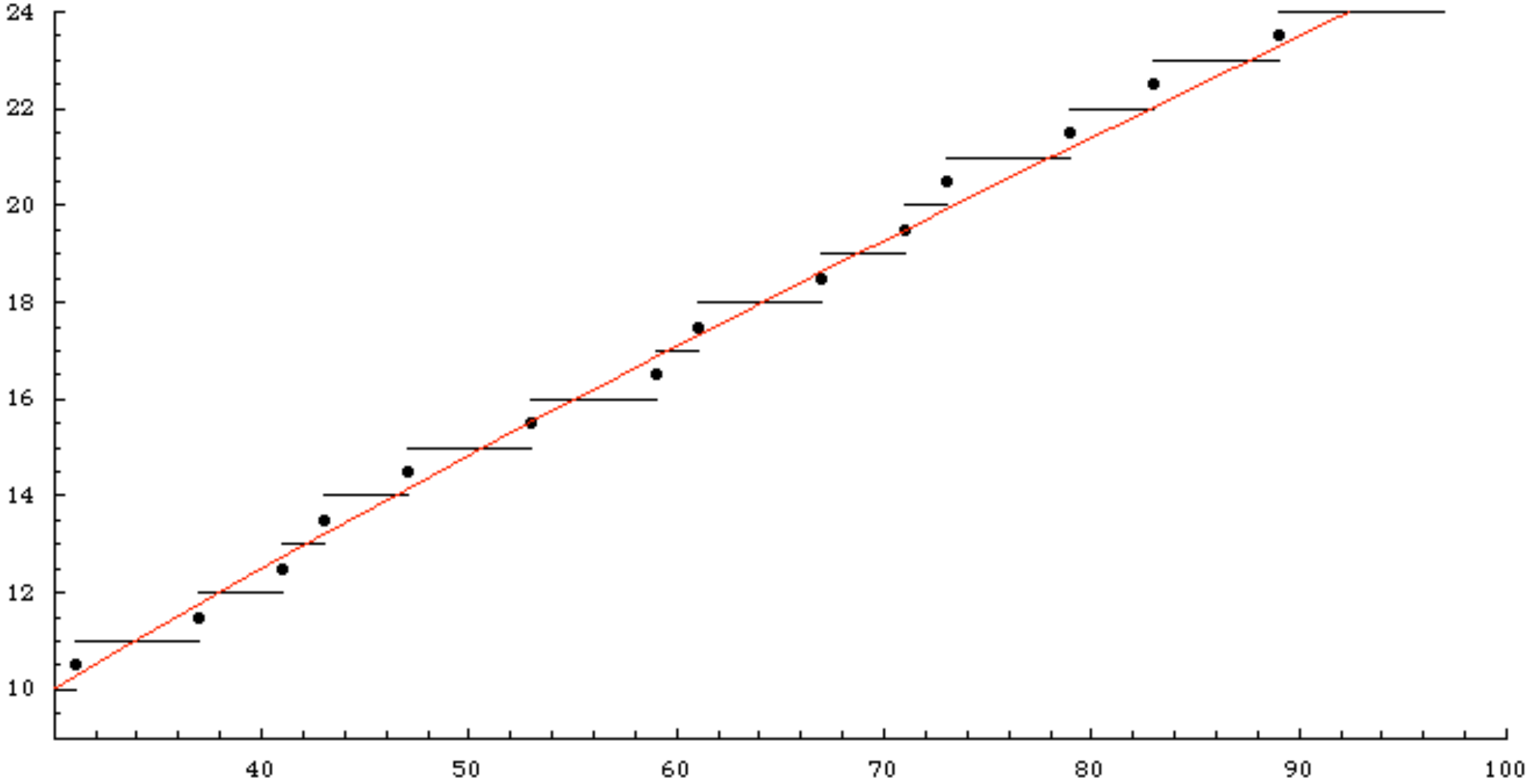
Si dimostra che la seguente relazione è equivalente alla Ipotesi di Riemann

$$\pi(x) = Li(x) + O\left(\sqrt{x}\ln x\right)$$

Ancora più accurata di Li(x) è la Formula di Riemann per la distribuzione dei primi ottenuta usando valori della funzione Zeta

$$R(x) = 1 + \sum_{m=1}^{\infty} \frac{(\log x)^m}{m!\, m\, \zeta(m+1)}$$

Questa approssimazione (in rosso) è una funzione continua e monotona crescente



Approssimazione a $\pi(x)$ con la formula di Riemann .

Purtroppo, come si vede anche dalla figura, la $\pi(x)$ è invece una funzione discontinua costante a tratti.

Esiste una approssimazione molto più sofisticata che fa entrare in gioco gli infiniti zeri non banali che la $\zeta(x)$ possiede sulla linea critica $x = 1/2 + i\,t$.

Formula di Riemann-Mangoldt per la distribuzione dei primi

$$\pi_0(x) = \sum_{n\geq 1} \frac{\mu(n)}{n} f\left(x^{1/n}\right)$$

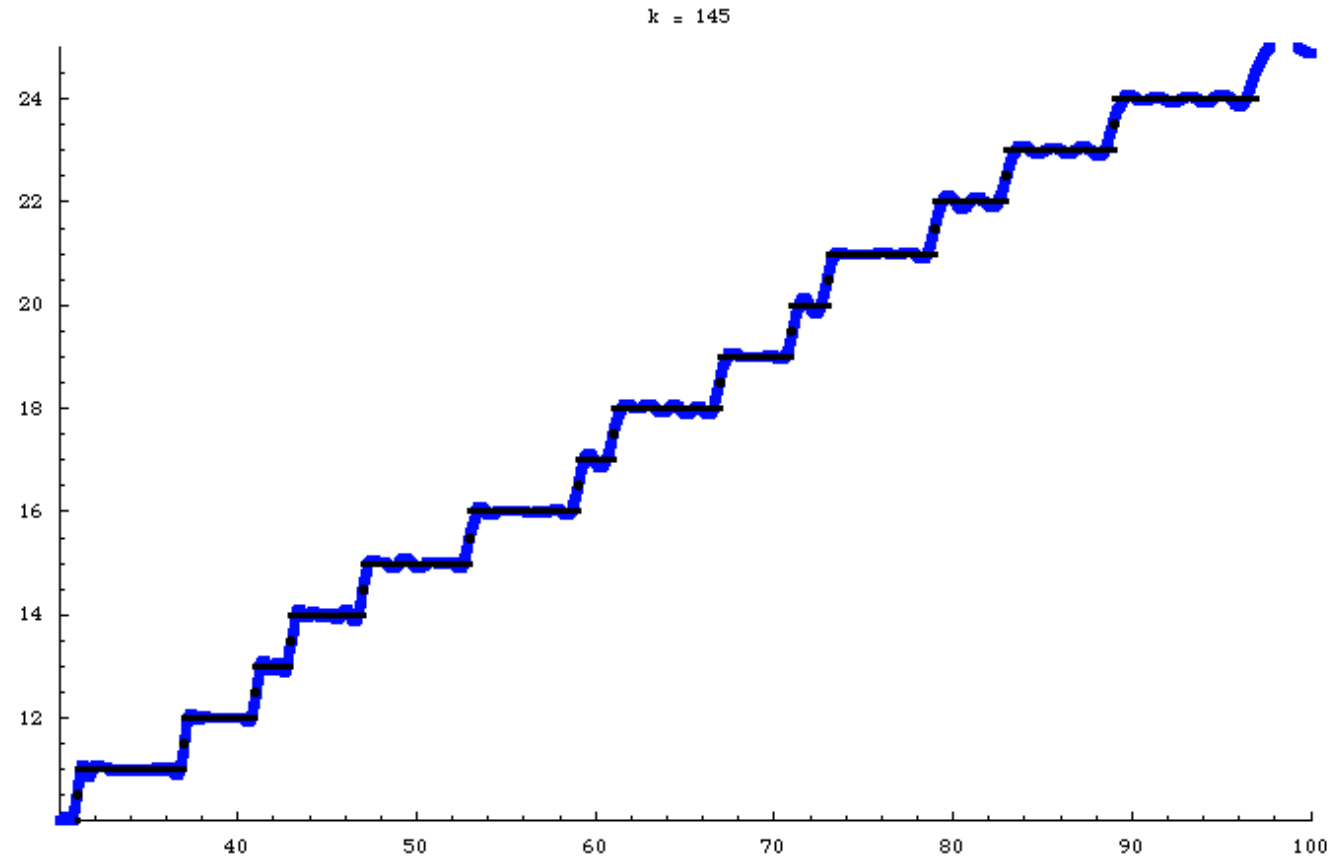
$$f(x) = \mathrm{li}(x) - \sum_{\zeta(\rho)=0} \mathrm{ei}(\rho \log x) - \log 2 + \int_x^\infty \frac{dt}{t(t^2-1)\log t}$$

ovvero, con pochi semplici passaggi

$$\pi_0(x) = \sum_{n=1}^{\log_2 x} \frac{\mu(n)}{n}\left[\mathrm{li}\left(x^{1/n}\right) - \log 2 + \int_{x^{1/n}}^\infty \frac{dt}{t(t^2-1)\log t}\right] +$$

$$-2 \sum_{\substack{\zeta(\rho)=0 \\ \mathrm{Im}(\rho)>0}} \sum_{n=1}^{\log_2 x} \frac{\mu(n)}{n} \mathrm{Re}\left(\mathrm{ei}\left(\frac{\rho}{n}\log x\right)\right)$$

Questa formula insieme ad un buon programma di calcolo numerico e simbolico (io ho usato *Mathematica*), permette di generare una bellissima animazione che evidenzia l'influenza della collocazione degli zeri della $\zeta(x)$ sulla distribuzione dei primi.



Approssimazione di Riemann-Mangoldt a $\pi(x)$ ottenuta con 145 zeri della funzione Zeta

Bibliografia

La Bibliografia sulla Zeta di Riemann è sterminata e la sua lettura richiede profonde conoscenze di analisi complessa.

Qui ci limitatiamo a citare il testo storico-divulgativo

M. du Sautoy, L'enigma dei numeri primi. BUR saggi, 2005

e alcuni link di internet

mathworld.wolfram.com/RiemannZetaFunction.html
mathworld.wolfram.com/PrimeNumberTheorem.html
mathworld.wolfram.com/RiemannHypothesis.html

en.wikipedia.org/wiki/Prime_number_theorem
en.wikipedia.org/wiki/Riemann_hypothesis
en.wikipedia.org/wiki/Prime_counting_function
en.wikipedia.org/wiki/Riemann_zeta_function

web.mala.bc.ca/pughg/RiemannZeta/RiemannZetaLong.html
numbers.computation.free.fr/Constants/Miscellaneous/zetazeroscompute.html