

Prime Polar Circularity

By Luis Felipe Massena Misiec

```
n = 1000;(* Gerar a lista dos primeiros n números primos *)
primos = Prime[Range[n]];(* Calcular a diferença entre cada número primo e sua posição *)
diferencas = primos - Range[n];(* Calcular a média das diferenças *)
a=Differences[%]
b=mediaDiferencas = Mean[diferencas]
N[%,9]( * Exibir o resultado *)
mediaDiferencas =Mean[a]
N[%,9]
c=a+b
cc=N[%,9]
ListPolarPlot[a+b]
gg=Union[cc]

n = 1009;(* Gerar a lista dos primeiros n números primos *)
primos = Prime[Range[n]];(* Calcular a diferença entre cada número primo e sua posição *)
diferencas = primos - Range[n];(* Calcular a média das diferenças *)
a=Differences[%]
b=mediaDiferencas = Mean[diferencas]
N[%,9]( * Exibir o resultado *)
mediaDiferencas =Mean[a]
N[%,9]
c=a+b
cc=N[%,9]
ListPolarPlot[a+b]
gg2=Sort[Union[cc]]
gg3=Sort[gg]
(* Supondo que gg e gg2 já estejam definidos *)
gg = Union[cc]; (* Lista de gg *)
gg2 = Sort[Union[cc]]; (* Lista de gg2 *)
```

```
(* Criar a tabela de correspondência *)
```

```
tabelaCorrespondencia = Transpose[{gg, gg3}];
```

```
(* Exibir a tabela *)
```

```
tabelaCorrespondencia // TableForm
```

```
(* Supondo que gg e gg2 já estejam definidos *)
```

```
gg = Union[cc]; (* Lista de gg *)
```

```
gg2 = Sort[Union[cc]]; (* Lista de gg2 *)
```

```
(* Calcular as diferenças entre gg e gg2 *)
```

```
diferencas = gg - gg3;
```

```
(* Criar a tabela de correspondência com as diferenças *)
```

```
tabelaCorrespondencia = Transpose[{gg, gg3, diferencas}];
```

```
(* Exibir a tabela *)
```

```
tabelaCorrespondencia // TableForm
```

```
(* Função para gerar gg e gg2 e calcular diferenças *)
```

```
(* Função para gerar gg e gg3 e calcular diferenças *)
```

```
(* Função para gerar gg e gg3 e calcular diferenças *)
```

```
(* Função para gerar gg e gg3 e calcular diferenças *)
```

```
calcularDiferencas[n_, nRef_] := Module[
```

```
{primos, diferencas, a, b, c, cc, gg, gg2, gg3, diferencasGG},
```

```
primos = Prime[Range[n]];
```

```
diferencas = primos - Range[n];
```

```
a = Differences[diferencas];
```

```
b = Mean[diferencas];
```

```
c = a + b;
```

```
cc = N[c, 9];
```

```
gg = Union[cc];
```

```
gg3 = Sort[gg];
```

```
(* Calcular diferença em relação ao valor de referência *)
```

```
primosRef = Prime[Range[nRef]];
```

```
diferencasRef = primosRef - Range[nRef];
```

```
aRef = Differences[diferencasRef];
```

```
bRef = Mean[diferencasRef];
```

```
cRef = aRef + bRef;
```

```
ccRef = N[cRef, 9];
```

```
ggRef = Union[ccRef];
```

```
gg3Ref = Sort[ggRef];
```

```
diferencasGG = gg - ggRef;
```

```
{gg, ggRef, diferencasGG}
```

```
];
```

```
(* Faixa de valores de n para verificar *)
```

```
nInicio = 1000;
```

```
nFim = 1020;
```

```
(* Valor de referência *)
```

```
nRef = 1000;
```

```
(* Calcular e armazenar os resultados para cada valor de n *)
```

```
resultados = Table[
```

```
{n, calcularDiferencas[n, nRef]},
```

```
{n, nInicio, nFim}
```

```
];
```

```
(* Criar a tabela de correspondência com as diferenças *)
```

```
tabelaResultados = Table[
```

```

{
  resultados[[i, 1]],      (* n *)
  resultados[[i, 2, 1]],   (* gg *)
  resultados[[i, 2, 2]],   (* ggRef *)
  resultados[[i, 2, 3]]    (* diferenças *)
},
{i, Length[resultados]}
];

```

(* Exibir os resultados em uma tabela *)

```
tabelaResultados // TableForm
```

3182.41300

3183.41300

3185.41300

3187.41300

3189.41300

3191.41300

3193.41300

3195.41300

3197.41300

3199.41300

3201.41300

3203.41300

3205.41300

3207.41300

3209.41300

3211.41300

3213.41300

3215.41300

3182.41300

3183.41300

3185.41300

3187.41300

3189.41300

3191.41300

3193.41300

3195.41300

3197.41300

3199.41300

3201.41300

3203.41300

3205.41300

3207.41300

3209.41300

3211.41300

3213.41300

3215.41300

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

0.*10⁻⁶

3186.15285

3187.15285

3189.15285

3191.15285

3193.15285

3195.15285

3197.15285

3199.15285

3201.15285

3203.15285

3205.15285

3207.15285

3209.15285

3211.15285

3213.15285

3215.15285

3217.15285

3219.15285

3182.41300

3183.41300

3185.41300

3187.41300

3189.41300

3191.41300

3193.41300

3195.41300

3197.41300

3199.41300

3201.41300

3203.41300

3205.41300

3207.41300

3209.41300

3211.41300

3213.41300

3215.41300

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3.73985

3189.89022

3190.89022

3192.89022

3194.89022

3196.89022

3198.89022

3200.89022

3202.89022

3204.89022

3206.89022

3208.89022

3210.89022

3212.89022

3214.89022

3216.89022

3218.89022

3220.89022

3222.89022

3182.41300

3183.41300

3185.41300

3187.41300

3189.41300

3191.41300

3193.41300

3195.41300

3197.41300

3199.41300

3201.41300

3203.41300

3205.41300

3207.41300

3209.41300

3211.41300

3213.41300

3215.41300

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

7.47722

From the fact that the circular polar prime's position can be divided by 360 degrees and as it seems to be the order of appearance of the position in the circular polar plot, then it emerges the following relation:

$360/1020(\text{position})=0.352941176$ that if multiplied by 10000 gives 3529.4117 , then $3529.4117/360 = 9.80392156$ that if multiplied by 1020 gives 10000...

```
n = 1000; (* Number of primes to generate *)
primos = Prime[Range[n]]; (* List of first n prime numbers *)
diferencas = primos - Range[n]; (* Differences between each prime number and its position *)
a = Differences[diferencas]; (* Differences between consecutive differences *)
b = Mean[diferencas]; (* Mean of the differences *)
```

```
Animate[
  c = Take[a + b, k]; (* Take the first k elements of the list *)
  ListPolarPlot[c, PlotRange -> All, ImageSize -> Large,
    PolarAxes -> True, PolarTicks -> {"Degrees", Automatic}],
  {k, 1, n, 1},
  DefaultDuration -> 1000 (* Duration in seconds for the whole animation *)
]
```

$$a/b=(x*c)/a=(y*b)=c$$

$a/b=(x*c)/a=(y*b)=c$ will always be true ?

To determine if the equation $ab=x \cdot ca=y \cdot b=c \frac{a}{b} = \frac{x \cdot c}{a} = y \cdot c$ will always be true, we need to analyze it carefully. Let's break it down step by step.

First, let's consider the equation $ab=x \cdot ca \frac{a}{b} = \frac{x \cdot c}{a}$:

1. Multiply both sides by a to get: $a^2=x \cdot c \cdot b$
2. This implies that $x=a^2 \cdot c \cdot b$

Now, let's look at $x \cdot ca=y \cdot b \frac{x \cdot c}{a} = y \cdot b$:

1. Substitute x from the previous result: $(a^2 \cdot c \cdot b) \cdot ca=y \cdot b \frac{a^2 \cdot c \cdot b}{a} = y \cdot b$
2. Simplify to get: $a^2 \cdot a=y \cdot b \frac{a^2}{b} = y \cdot b$
3. Further simplify to get: $ab=y \cdot b \frac{a}{b} = y$
4. This implies that $y=ab^2y = \frac{a}{b^2}$

Lastly, the equation $y \cdot b = c y \cdot b = c$:

1. Substitute y from the previous result: $a b^2 \cdot b = c \frac{a}{b^2} \cdot b = c$
2. Simplify to get: $a b = c \frac{a}{b} = c$

The equation $a b = c \frac{a}{b} = c$ must hold true for the original equation to be true.

```
Clear[a, b, c, x, y]
```

```
(* Solve for x *)
```

```
eq1 = a/b == x*c/a;
```

```
sol1 = Solve[eq1, x]
```

```
(* Solve for y *)
```

```
eq2 = (x*c)/a == y*b;
```

```
sol2 = Solve[eq2 /. sol1, y]
```

```
(* Validate the final equation *)
```

```
eq3 = y*b == c;
```

```
eq3 /. sol2 /. sol1
```

results:

```
{{x->a^2/(b c)}}
```

```
{{y->a/b^2}}
```

```
{{a/b==c}}
```

This code clears any existing definitions, solves for x and y in terms of a , b , and c , and then validates the final equation. The key is ensuring $a b = c \frac{a}{b} = c$ for the original equation to be true.