Ayudantía 6

1. Cómo utilizar SageMath para los informes

SageMath es un sistema de álgebra computacional (System for Algebra and Geometry Experimentation). Su software es libre por lo que es una herramienta accesible (aunque la instalación puede ser engorrosa si utilizas Windows). Este sistema agrupa más de 100 paquetes de software matemáticos (por ejemplo, Pari/GP, Numpy, etc...) y basta sólo manejar el lenguaje de python para ocuparlos.

Para acceder a SageMath recomiendo inicialmente utilizar la página CoCalc (puedes hacer click en la palabra para acceder a la página); aquí puedes crear tus proyectos sin necesidad de instalar SageMath. Aunque siempre es mejor instalarlo, pero toma tiempo hacerlo y procura leer bien las instrucciones que ofrece la página principal de SageMath.

2. Cómo adjuntar código

Cuando ya tengas tu proyecto, puedes tomar el código y agregarlo a tu informe en LaTeX. Para ello puedes usar el paquete `minted` y adjuntar tu código como sigue

```
\begin{minted}{python}
def rad(x):
        return prod(prime_factors(x))
def coprime(a, b):
    return (gcd(a, b) == 1)
\end{minted}
```

3. Ejemplo

Realizaremos un ejemplo de experimento numérico de la conjetura *abc*. Para ello debemos recordar lo siguiente:

Para un entero positivo n definimos su ***radical*** como

$$\mathrm{rad}(n) = \prod_{p|n} p.$$

donde p recorre todos los divisores primos de n sin repetición.

***Conjetura 1** (abc):* Sea $\varepsilon > 0$. Existe una constante $\kappa_\varepsilon > 0$ que solo depende de ε tal que ocurre lo siguiente: Sean a, b, c enteros positivos coprimos con $a + b = c$. Entonces,

$$c \leq \kappa_\varepsilon \cdot \mathrm{rad}(abc)^{1+\varepsilon}. \tag{3.1}$$

Tras haber recordado el enunciado de la conjetura definiremos en SAGE las siguientes funciones

```
def rad(x):
        return prod(prime_factors(x))
def coprime(a, b):
    return (gcd(a, b) == 1)
```

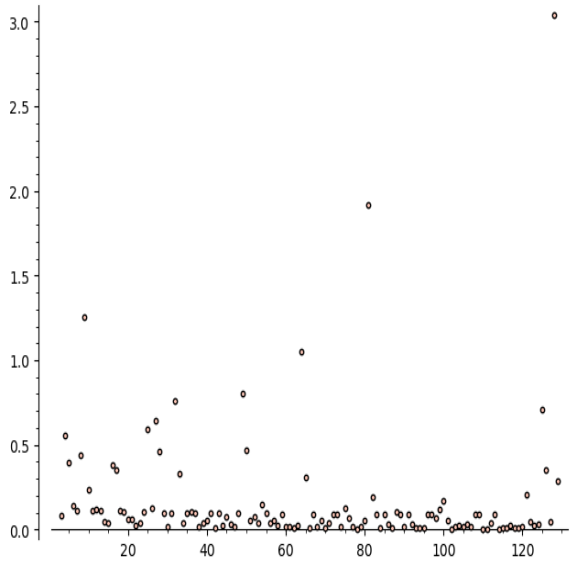
Notemos que lo interesante de la conjetura es la existencia de una constante universal κ_ε que no depende de a, b, c. Así que intentaremos encontrar esta constante universal para casos pequeños.

Primero, reescribamos la desigualdad (3.1) como

$$\frac{c}{\mathrm{rad}(abc)^{1+\varepsilon}} \leq \kappa_\varepsilon.$$

Para ver qué valores toma κ_ε nuestro experimento iniciará fijando un valor de c y un valor para ε y a partir de ellos, buscaremos a, b coprimos y calcularemos κ_ε como sigue

```
def minrad_abc(c, epsilon):
    r = c**(2+3*epsilon) # = (c^3)^{1+epsilon}/c
    if c <= 2:
        return (2**epsilon, 1, 1, 2)
    A = 1
    for a in range(1, c//2):
        if not coprime(a, c):
            continue
        m = rad(a*(c-a)*c)**(1+epsilon)/c
        if m < r:
            r = m
            A = a
    return (r, A, c-A, c) # aquí se retorna kappa, a, b y c
```



Y finalizaremos graficando los valores que nos entregue dentro de cierto rango

```
epsilon = 0.1
scatter_plot([ (c, 1/minrad_abc(c, epsilon)[0]) for c in range(3, 130) ], marker='o', markersize=10)
```

Por último, vamos a realizar una tabla de tuplas *abc* de los resultados

```
m, a, b, c = minrad_abc(2, epsilon)
counter = [(2, m)]
bad = [(a, b, c, 1/m)]
for c in range(3, 1000):
    M, a, b, _ = minrad_abc(c, epsilon)
    if M < m:
        m = M
        counter.append((c, m))
        bad.append((a, b, c, 1/m))
plot_step_function(counter)
for tupla in bad:
    print( " & ".join([ str(e) for e in tupla ]) + " \\\\" )
```

Obteniendo de manera manual el código LaTeXpara la tabla:

```
1 & 1 & 2 & 0.933032991536807 \\
1 & 8 & 9 & 1.25393820311691 \\
```

```
1 & 80 & 81 & 1.92154977483732 \\
3 & 125 & 128 & 3.03652310097749 \\
```

Así que lo copiamos y lo colocamos en nuestro informe:

a	b	c	κ_ε
1	1	2	0,933032991536807
1	8	9	1,25393820311691
1	80	81	1,92154977483732
3	125	128	3,03652310097749

Referencias

1. The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 10.4)* `https://www.sagemath.org` (2024).

Correo electrónico: `rseplveda@uc.cl`