Unidade de Ensino de Matemática Aplicada e Análise Numérica
Departamento de Matemática/Instituto Superior Técnico

Matemática Experimental (LMAC) – 2º Período de 2021/2022

Trabalho Computacional

Primeira Entrega

Data de entrega: 22 de Dezembro de 2021

Introdução e Instruções

Instruções de entrega: Cada grupo deverá entregar um notebook *Mathematica* e um relatório em pdf em para o e-mail `jvideman@math.tecnico.ulisboa.pt`. O relatório deverá conter as respostas às questões, e o notebook deverá conter o código utilizado para as responder. Podem elaborar o relatório em Word, Mathematica ou Latex, desde que o submetam em pdf. Se quiserem usar Latex podem pedir um template ao professor. Se algum código demorar algum tempo a correr, devem indicá-lo, preferencialmente no notebook, com um comentário a indicar a estimativa de tempo. `(* Isto é um exemplo de um comentário. Este texto não é executado pelo processador do Mathematica.*)`

Critérios de avaliação: Os trabalhos serão cotados com base nas respostas aos exercícios. Não será avaliada a qualidade do código no notebook *Mathematica*, mas o código será lido e executado para garantir que coincide com as repostas dadas no relatório e que o raciocínio está correto. Como tal, a questão à qual cada pedaço de código está associado deverá ser indicada com um comentário. A qualidade do relatório, no entanto, será avaliada, pelo que deve estar sucinto, claro, bem-escrito e bem-organizado. O relatório não deve exceder as 15 páginas.

A não ser que indicado *explicitamente* em contrário, justificações analíticas não são necessárias. Por exemplo, para justificar que uma função é contínua seria apenas necessário fazer o gráfico e apontar que não há descontinuidades.

Alguns exercícios têm no final alguns comandos sugeridos. Não é obrigatório o seu uso, e poderá ser necessário usar outros comandos. O uso de motores de busca é permitido e encorajado.

1 Geometria e Visualização Gráfica [6 valores]

Considere as seguintes duas funções, que vão de $\mathbb{R}^2$ para $\mathbb{R}^3$, onde r e R são parâmetros reais tais que $0 < r < R$:

$$S(u,v) = (R\sin(u)\cos(v), R\cos(u)\cos(v), R\sin(v)),$$
$$T(u,v) = (\cos(u)(R + r\cos(v)), \sin(u)(R + r\cos(v)), r\sin(v)).$$

Estas funções parametrizam, respetivamente, a superfície de uma esfera e de um toro (um 'donut').

Exercício 1.1. Use o comando `ParametricPlot3D` para verificar a afirmação anterior. Use os parâmetros $R = 3$ e $r = 1$. Para parametrizar a esfera, faça variar $u \in [0, 2\pi]$ e $v \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. Para parametrizar o toro, faça $u, v \in [0, 2\pi]$.

Exercício 1.2. O que acontece à esfera se $R < 0$?

Exercício 1.3. Indique o que acontece ao toro nos seguintes casos:

a) $r = 0$,

b) $0 < R < r$,

c) $0 = R < r$.

Exercício 1.4. Defina uma função `plotTorus[R]` que retorna a imagem do toro com $r = 1$ e R dado.
Execute o seguinte código e explique o seu funcionamento.

$$\texttt{ListAnimate}[\texttt{Table}[\texttt{plotTorus}[\texttt{R}], \{\texttt{R}, 2, 0, -0.1\}]]$$

Compare o resultado com o seguinte código e explique a razão por trás da diferença.

$$\texttt{Animate}[\texttt{plotTorus}[\texttt{R}], \{\texttt{R}, 2, 0, -0.1\}]$$

Exercício 1.5. Represente graficamente:

a) As curvas na superfície da esfera com u fixo e v a variar em $[-\frac{\pi}{2}, \frac{\pi}{2}]$,

b) As curvas na superfície da esfera com v fixo e $u \in [0, 2\pi]$,

c) As curvas na superfície do toro com u fixo e $v \in [0, 2\pi]$ e

d) As curvas na superfície do toro com v fixo e $u \in [0, 2\pi]$.

Exercício 1.6. Seja C a curva na superfície do toro formada pelos pontos com coordenadas $u = v$. A curva C é um círculo em $\mathbb{R}^3$? Justifique.

Comandos sugeridos: `ParametricPlot3D`, `Show`.

2 Números k-redutíveis a 1 [7 valores]

Dados k e n naturais, definimos a operação-k aplicada a n como a soma das potências k dos algarismos de n. Por exemplo, a operação-3 aplicada a 512 retorna $5^3 + 1^3 + 2^3 = 125 + 1 + 8 = 134$. Dizemos que n é k-redutível a 1 se ao aplicarmos a operação-k um número suficiente de vezes obtemos 1. Por exemplo, 31 é 2-redutível, pois $3^2 + 1^2 = 10$ e $1^2 + 0^2 = 1$.

Exercício 2.1. Aplique a operação-5 aos números 68492, 2681 e 73921.

Exercício 2.2. Crie uma função `op[k,n]`, que calcula o valor da operação-k aplicada a n.
Verifique que para $k = 2$ e $1 \leq n \leq 10^5$ temos

$$\texttt{op}[\texttt{k}, \texttt{n}] < 500.$$

Exercício 2.3. Ao iterar a aplicação da operação-k a n, obtém-se uma sucessão de números

$$a_0 = n,\ a_1 = \texttt{op}_k(n),\ a_2 = \texttt{op}_k(\texttt{op}_k(n)),\ a_3 = \texttt{op}_k(\texttt{op}_k(\texttt{op}_k(n))),\ \ldots$$

Justifique analiticamente que para $k = 2$ e qualquer n inicial, existe N tal que

$$a_i \leq 180 \quad \forall_{i>N} \tag{1}$$

Sugestão: Sejam $d_0, \ldots, d_p$ os dígitos de n. Escreva $n = d_p 10^p + d_{p-1} 10^{p-1} + \ldots d_1 10 + d_0$, onde $0 \leq d_i \leq 9\ \forall_i$.
Calcule $\texttt{op}_2(n)$ e prove que

$$\texttt{op}_2(n) \leq 2 \times 9^2 + \frac{1}{10} n.$$

Conclua que para $n > 180$ se tem $\texttt{op}_2(n) < n$, e use este facto para justificar (1).

Exercício 2.4. Execute o seguinte código. Explique o que este código faz e use o *output* deste para descrever o comportamento no infinito da sucessão a_n. Não precisa de explicar detalhadamente o código, nomeadamente pode não mencionar os comandos estéticos. Talvez queira aumentar o tamanho do grafo clicando na imagem e arrastando a janela.

```
Select[Range[180], # == op[2, #] &]
g = Graph[Range[180], Table[k -> op[2, k], {k, 1, 180}], VertexLabels -> Placed["Name", Center],
VertexSize -> 0.1, EdgeStyle -> Directive[Pink, Arrowheads[.02]], GraphLayout -> "RadialEmbedding"];
WeaklyConnectedGraphComponents[ HighlightGraph[g, FindCycle[g, Infinity, All], GraphHighlightStyle
-> "Thick"]]
```

Exercício 2.5. Determine a percentagem dos números em $\{1, \ldots, 1000\}$ que são 2-redutíveis.

Comandos sugeridos: `IntegerDigits, NestWhile`.

3 Primos Truncáveis [7 valores]

O objectivo deste exercício é estudar primos truncáveis, isto é, primos que preservam a primalidade quando os seus dígitos são removidos um a um. Apenas definimos truncável para números primos sem nenhum algarismo igual a 0. Claro que é possível remover os dígitos de várias formas, das quais as mais conhecidas são:

- Primos truncáveis à direita, nos quais se remove sempre o dígito das unidades; Por exemplo, 233 é um primo truncável à direita, visto que 233, 23 e 2 são todos primos.
- Primos truncáveis à esquerda, nos quais se remove sempre o dígito da esquerda; Por exemplo, 137 é um primo truncável à esquerda, visto que 137, 37 e 7 são todos primos.
- Primos DE-truncáveis, nos quais se removem dois dígitos de cada vez, o da direita e o da esquerda; Por exemplo, 773 e 1373 são primos DE-truncáveis, visto que 773, 7, 1373 e 37 são todos primos.
- Primos duplamente truncáveis, ou seja, números que são simultaneamente primos truncáveis à direita e primos truncáveis à esquerda; Por exemplo, 317 é um primo duplamente truncável, visto que 317, 31 e 3 são primos, logo é truncável à direita, e 317, 17 e 7 são primos, logo é truncável à esquerda.

Neste exercício, o foco é no primeiro caso, uma vez que é o único em que se consegue determinar todos os primos em tempo útil.

Exercício 3.1. Defina uma função `nozeroprimes` que, dado um n, retorna todos os primos menores que n que não têm nenhum dígito igual a 0. Calcule `nozeroprimes[1000]`.

Exercício 3.2. Crie uma função `righttruncatableQ` e verifique se os seguintes números são primos truncáveis à direita: 1997, 1933, 2333, 71119, 17133719 e 29399999.

Exercício 3.3. Determine todos os primos truncáveis à direita menores que 500.

Exercício 3.4. Sabendo que 73 939 133 é o maior primo truncável à direita, determine todos os primos truncáveis à direita.
Nota: 73 939 133 *é um número grande, não se assuste se o seu código demorar alguns minutos a correr. Comece por testar o código para números mais pequenos se quiser verificar que está a funcionar.*

Exercício 3.5. Mostre analiticamente, com apoio computacional, que não há primos truncáveis com mais de 8 dígitos e conclua que 73 939 133 é o maior número truncável à direita, sem recorrer ao teste de todos os primos de 9 dígitos.

Exercício 3.6. Crie outra função, `lefttruncatableQ`, que determina se um número é primo truncável à esquerda. Com ela, pode-se verificar que há mais primos truncáveis à esquerda do que à direita, e que existe pelo menos um primo truncável à esquerda que é maior que todos os primos truncáveis à direita (veja a alínea (d)):

a) Determine todos os primos duplamente truncáveis.

b) Será que todos os primos duplamente truncáveis são primos DE-truncáveis? Justifique que há pelo menos mais um primo DE-truncável do que primos duplamente truncáveis. [Curiosidade: Há mais 920 720 300.]

   Nota: Repare que não é preciso (apesar de ser permitido) definir uma função que verifica se números são truncáveis à direita e à esquerda para resolver esta alínea.

c) Determine quantos números menores que 1500 são primos truncáveis à esquerda, e verifique que o número 357686312646216567629137 é um primo truncável à esquerda. [Curiosidade: este é o maior primo truncável à esquerda, havendo um total de 4260.]

Comandos sugeridos: `Prime`, `PrimePi`, `PrimeQ`, `IntegerDigits`, `IntegerLength`, `DigitCount Select`.