PROFESSOR DE MATEMÁTICA ONLINE
Revista eletrônica da Sociedade Brasileira de Matemática

PMO v.8, n.2, 2020
ISSN: 2319-023X
https://doi.org/10.21711/2319023x2020/pmo816

Sobre a soma dos termos de progressões geométricas e métodos numéricos para determinar autovalores e raízes de polinômios

José Claudinei Ferreira

Resumo

Este texto traz uma breve apresentação da evolução de algoritmos para a aproximação de raízes de polinômios e de autovalores de matrizes, a partir de relações de recorrência, funções geradoras e da expressão para a soma dos termos de uma progressão geométrica. Com isso, pretende-se mostrar que ideias simples, quando analisadas por tempo suficiente, podem levar a resultados impressionantes, principalmente se aliadas ao pensamento e ao poder de processamento computacional. Para ilustrar os resultados incluímos um apêndice no texto, com alguns exemplos implementados em linguagem de programação R.

Palavras-chave: Relações de recorrência; Modelagem matemática; Métodos numéricos; Algoritmos.

Abstract

This article deals with a brief discussion about the origins of some algorithms to find approximations to roots of polynomial equations and the eigenvalues of matrices, using recurrence relations, generating functions, and the expression to the sum of all terms of geometric series. The basic idea is to show that, sometimes, a simple argument can give us a very interesting and profound answer, when we think about it for a long time, and use the computational power to do many calculations. To help us to understand what we mean, we have included an appendix with some examples, and R language implementations of algorithms.

Keywords: Recurrence relations; Mathematical Modeling; Numerical methods; Algorithms.

1. Introdução

Considere um número real r e um número real a_0 e defina a sequência $a_n = a_{n-1}r$, para cada número natural n. A sequência a_n é conhecida como progressão geométrica e é um dos dois exemplos clássicos de relação de recorrência[1] (ou sequências recursivas) apresentados nas aulas de Matemática, desde o Ensino Fundamental. Outro exemplo é o de progressão aritmética. Cabe destacar que o

[1]Não posso deixar de mencionar que o termo recorrência pode ser usado para trabalhar com a geração de belas imagens fractais relacionadas com o trabalho de Benoit Mandelbrot e com o conjunto com seu nome!

trabalho com relações de recorrência e algoritmos tem sido incentivado pela Base Nacional Comum Curricular.

Um exemplo comum, e útil no dia a dia das pessoas, é o uso de progressão geométrica em aplicações financeiras, em que r é uma taxa de juros e a_0 é uma aplicação inicial (ou mensal fixa, por exemplo).

Podemos definir uma nova relação de recorrência, que é a soma dos termos da progressão geométrica

$$s_n = a_0 + a_1 + \cdots + a_n = a_0(1 + r + r^2 + r^3 + \cdots + r^n)$$

ou, por exemplo, o montante após $n + 1$ aplicações mensais iguais a a_0, com taxa de juros mensais igual a r.

Imagine agora a multiplicação de s_n por r e subtraia s_n do resultado. Deve notar que

$$rs_n - s_n = s_n(r - 1) = a_0(r^{n+1} - 1).$$

Isso produz a conhecida expressão para a soma dos termos iniciais de uma progressão geométrica como

$$s_n = a_0 \frac{1 - r^{n+1}}{1 - r}, \quad n = 0, 1, \ldots.$$

Com um pouco mais de imaginação, caso $|r| < 1$, vemos que a soma de todos os termos da progressão é

$$\frac{1}{1 - r} = 1 + r + r^2 + r^3 + \cdots = \sum_{i=0}^{+\infty} r^i, \tag{1}$$

que é o ponto de partida para o que queremos discutir neste texto.

O problema torna-se um pouco mais complicado quando a taxa de juros não é conhecida. Uma forma de determinar a taxa é encontrar as raízes de um polinômio. Vejamos um exemplo disso a seguir.

Um amigo fez um empréstimo de R$ 1.200,00 de um outro conhecido nosso, para auxiliar nos gastos da festa de aniversário da filha. Esse valor deve ser pago em seis parcelas mensais de R$ 300,00, sem entrada. Meu amigo veio logo me contar do negócio e eu perguntei de imediato: Qual é a taxa de juros compostos mensais você vai pagar nesse empréstimo? O meu amigo ficou quieto, disse que nem pensara sobre isso.

Pensei que uma forma de responder a essa pergunta seria observar que, após um mês do empréstimo, o meu amigo teria uma dívida de $d_0(1 + r)$, sendo r a taxa de juros cobrados e $d_0 = 1.200,00$ reais. Fazendo o pagamento de 300 reais a dívida restante seria de $d_1 = d_0(1 + r) - 300$ reais. Quando n meses tivessem passado, o meu amigo teria uma dívida $d_n = d_{n-1}(1 + r) - 300$ reais, mas não ficaria devendo após o sexto mês. Bastaria resolver a equação polinomial $d_6 = 0$, de grau seis na variável r. Pensei, mas achei melhor não explicar isso para o meu amigo, que não gosta muito de falar em matemática com letras, com números ele suporta uma conversa, quando não tem jeito de fugir dela.

Um raciocínio semelhante ao desse exemplo pode ser aplicado a outros problemas, como no que segue.

Um amigo negociava a compra de um terreno no valor de R$ 114.900,00. Ele, porém, não dispunha do valor total e, sendo assim, a proposta em negociação era a seguinte: seria feito o pagamento de R$ 11.000,00 como entrada, com cheque para o dia 12 de agosto de 2019, um cheque de R$

30.000,00 pré-datado para o dia 12 de setembro de 2019. O restante do valor deveria ser pago em 96 parcelas fixas de R$ 1.159,00 com vencimento no dia 12 de cada mês, com início no dia 12 de outubro de 2019.

Quando meu amigo estava fazendo a negociação, ele perguntou a si mesmo: Qual é a taxa de juros compostos mensais cobrada pelo vendedor? E a taxa anual? Quanto eu ainda estarei devendo em agosto de 2025?

Problemas como esses podem ser relacionados a outros mais gerais, como o de lidar com relações de recorrência, determinar raízes de polinômios e resolver equações diferenciais e que exigem métodos numéricos. O ponto de partida para esses métodos parece ter sido influenciado pela Expressão (1).

Atualmente existem vários *softwares* para lidar com tais problemas e alguns dos algoritmos usados surgiram do conceito de função geradora. Por isso, discutimos de forma breve neste texto algumas propriedades dessas funções e suas aplicações em problemas que envolvem a determinação de autovalores e raízes de polinômios. Ilustramos isso com o uso da linguagem de programação R em alguns exemplos como Apêndice do texto.

2. Funções geradoras

Lembramos que uma sequência de números reais é uma função $a : \mathbb{N} \to \mathbb{R}$ que denotamos como $a(n) = a_n$. Vamos associar essa sequência ao que chamamos de função geradora (veja [6]), dada por

$$G(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \cdots \tag{2}$$

sem a preocupação com a convergência da série e nem preocupação com o que é o domínio de $G(x)$. O interesse principal é no caso em que a_n é uma relação de recorrência que depende linearmente $a_{n-1}, \ldots, a_{n-j}$.

Vejamos isso em uma situação real, em que desejamos saber a quantidade de formas possíveis para fazer uma faixa $1 \times n$ utilizando ladrilhos de tamanho 1×1 e 1×2.

O diagrama da Figura 1 relaciona a sequência $a_0 = 1$, $a_1 = 1$ e $a_n = a_{n-1} + a_{n-2}$, para $n = 2, 3, \ldots$, conhecida como sequência de Fibonacci, com o problema de contar formas de se obter os ladrilhamentos.

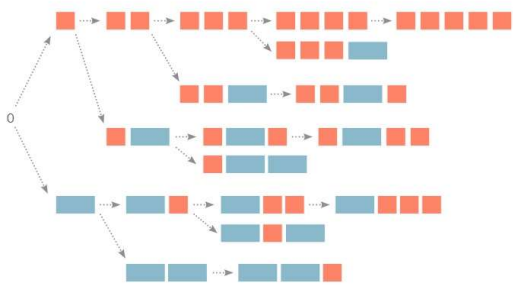

Figura 1: Ladrilhamento $1 \times n$, com peças 1×1 e 1×2. Fonte: [2]

Isso quer dizer que, para se obter o valor de a_{100}, por exemplo, precisamos conhecer todos os a_n, para n entre 2 e 99. Nesse caso, temos a função geradora

$$\begin{aligned} G(x) &= 1 + x + 2x^2 + 3x^3 + 5x^4 + \cdots \\ &= 1 + x + \sum_{n=2}^{+\infty} (a_{n-1} + a_{n-2})x^n \\ &= 1 + x \sum_{n=0}^{+\infty} a_n x^n + x^2 \sum_{n=0}^{+\infty} a_n x^n \\ &= 1 + xG(x) + x^2 G(x) \end{aligned} \tag{3}$$

Segue que, a menos de indeterminações ou polos, temos que

$$G(x) = \frac{1}{1 - x - x^2}.$$

Como $1 - x - x^2 = (x - r_1)(x - r_2)$, sendo são $r_1 = \frac{-\sqrt{5}-1}{2}$ e $r_2 = \frac{\sqrt{5}-1}{2}$, utilizando frações parciais obtemos

$$\begin{aligned} G(x) &= \frac{B_1}{x - r_1} + \frac{B_2}{x - r_2} \\ &= \frac{A_1}{1 - \frac{x}{r_1}} + \frac{A_2}{1 - \frac{x}{r_2}} \\ &= \sum_{n=0}^{+\infty} \left(A_1 \left(\frac{1}{r_1} \right)^n + A_2 \left(\frac{1}{r_1} \right)^n \right) x^n. \end{aligned}$$

Uma comparação com a Expressão (3) produz a igualdade

$$a_n = A_1 \left(\frac{1}{r_1} \right)^n + A_2 \left(\frac{1}{r_1} \right)^n,$$

em uma forma não recorrente. Observe que obtemos uma representação de a_n como uma combinação linear de termos de uma progressão geométrica e que para isso utilizamos a Expressão (1).

3. Resoluções de recorrência lineares

O exemplo que acabamos de discutir sugere que podemos utilizar as m raízes da equação polinomial ([3])

$$p(x) = 1 - \sum_{i=1}^{m} b_i x^i = 0, \tag{4}$$

para encontrar uma expressão para a sequência a_n definida pela relação de recorrência

$$a_n - \sum_{i=1}^{m} b_i a_{n-i} = 0, \qquad n \geq 1, \tag{5}$$

com a_0 dado (e não nulo) e $a_j = 0$, se $j < 0$.

Isso de fato ocorre, e para verificar rapidamente o procedimento, consideramos a função geradora

$$G(x) = a_0 + b_1a_0x + (b_1a_1 + b_2a_0)x^2 + (b_1a_2 + b_2a_1 + b_3a_0)x^3 + \cdots + \left(\sum_{i=1}^{m} b_i a_{m-i}\right) x^m + \cdots$$

Rearranjando os termos em função de b_i concluímos que

$$G(x) = a_0 + b_1xG(x) + b_2x^2G(x) + \cdots + b_mx^mG(x),$$

e segue que

$$G(x) = \frac{a_0}{1 - \sum_{i=1}^{m} b_ix^i}.$$

Supondo ainda que

$$p(x) = c(x - r_1)(x - r_2)\cdots(x - r_m), \qquad r_i \neq r_j, \qquad i, j = 1, 2, \ldots, m, \tag{6}$$

e tendo em mente a Expressão (1) mais uma vez, podemos escrever

$$\begin{aligned} G(x) &= \frac{A_1}{1 - \frac{x}{r_1}} + \frac{A_2}{1 - \frac{x}{r_2}} + \cdots + \frac{A_m}{1 - \frac{x}{r_m}} \\ &= \sum_{n=0}^{+\infty} \left(A_1 \left(\frac{1}{r_1}\right)^n + A_2 \left(\frac{1}{r_2}\right)^n + \cdots + A_m \left(\frac{1}{r_m}\right)^n \right) x^n. \end{aligned} \tag{7}$$

Isso produz uma representação para a_n, dada por

$$a_n = A_1 \left(\frac{1}{r_1}\right)^n + A_2 \left(\frac{1}{r_2}\right)^n + \cdots + A_m \left(\frac{1}{r_m}\right)^n. \tag{8}$$

Esse raciocínio levou a grandes avanços na elaboração de algoritmos para determinar as raízes de $p(x) = 0$, ou os polos de $G(x)$, uma vez que está demonstrada a não existência de métodos diretos para a resolução de equações polinomiais de grau maior que 4 e que Gauss demonstrou a existência dessas raízes em 1799.

Se puder, veja um breve histórico do estudo de equações polinomiais em [5, 7, 15] e leia sobre o trabalho de Jacob Bernoulli (1654-1705). Cabe notar que as relações de Girard foram publicadas em 1629.

Observação 1. Para o caso em que $a_0 = a_1 = \cdots = a_{p-1} = 0$, com $a_p \neq 0$, pode ser facilmente resolvido colocando o termo x^p em evidência na expressão de $G(x)$ e procedendo como fazemos anteriormente para a função geradora $G(x)/x^p$ no lugar de $G(x)$.

O caso em que algum $r_i = r_j$, na Expressão (6), ele pode ser tratado usando expressões um pouco mais complicadas que a Expressão (7), mas vamos omitir isso neste texto.

Note que, se

$$q(x) = x^m - \sum_{i=1}^{m} b_ix^{m-i} \tag{9}$$

e $\lambda_i = r_i^{-1}$, então, $q(\lambda_i) = 0$, uma vez que $x^m p(x^{-1}) = q(x)$.

Por fim, como $G(x)$ é uma função racional, não é difícil verificar que existem todas as derivadas $G^{(j)}(x)$, para $j = 0, 1, 2, \ldots$, quando $p(x) \neq 0$. Isso quer dizer que a série de Taylor (veja [13]) de $G(x)$ é da forma

$$G(x) = G(0) + G'(0)x + \frac{G''(0)}{2!}x^2 + \cdots + \frac{G^{(n)(0)}}{n!}x^n + \cdots. \tag{10}$$

Em particular, segue que

$$\frac{G^{(n)}(0)}{n!} = \sum_{i=1}^{m} b_i a_{n-i} = a_n.$$

3.1. Sistemas de equações a diferenças lineares

Muitas vezes temos problemas envolvendo uma relação de recorrência vetorial linear, e podemos aplicar a ideia de funções geradoras. Neste texto apresentamos apenas um exemplo.

Considere a relação de recorrência que está relacionda com um problema de ladrilhamento $2 \times n$ (veja [2]),

$$\begin{cases} T_n &= 2T_{n-1} + T_{n-2} + 2R_{n-1} \\ R_n &= T_{n-1} + R_{n-1} \\ T_0 &= 1 \\ T_1 &= 2 \\ R_0 &= 1 \\ R_1 &= 1 \end{cases}$$

Podemos observar que $T_n - T_{n-2} = 2(T_{n-1} + R_{n-1}) = R_n$ e isso quer dizer que

$$\begin{cases} T_n &= 3T_{n-1} + T_{n-2} - T_{n-3} \\ T_0 &= 1 \\ T_1 &= 2 \\ T_2 &= 7 \end{cases}$$

Basta determinar as raízes da equação $1 - 3x - x^2 + x^3 = 0$ para determinar T_n e R_n (veja [12]).

4. Equações diferenciais ordinárias

Equações a diferença ou relações de recorrência são usados para problemas discretos. Para problemas descritos no tempo contínuo é comum o uso de equações diferenciais (veja [1] para um tratamento numérico).

Um caso particular desse tipo de equação contém derivadas de ordem até $m \geq 1$ e pode ser escrito como

$$y^{(n)}(t) - \sum_{i=1}^{m} b_i y^{(n-i)}(t) = 0, \qquad n = m, m+1, \ldots \tag{11}$$

em que b_i, $t \in \mathbb{R}$, sendo $y^{(0)}(t) = y(t)$ e $y^{(i)}(t)$ a derivada (ou taxa de variação) de $y^{(i-1)}(t)$ no tempo t, para $i = 1, 2, \ldots, m$. Esse tipo de equação é útil em circuitos elétricos, por exemplo. Notou alguma semelhança com a Expressão (5)?

Denotando $y(t) = y_1(t)$, $y'(t) = y_2(t)$, $y'''(t) = y_3(t)$, ..., $y^{(m-1)}(t) = y_m(t)$, segue que

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_{m-1} \\ y_m(t) \end{bmatrix}' = \begin{bmatrix} 0 & 1 & 0 & \ldots & 0 \\ 0 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \ldots & 1 \\ b_m & b_{m-1} & b_{m-2} & \ldots & b_1 \end{bmatrix} \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_{m-1}(t) \\ y_m(t) \end{bmatrix}. \tag{12}$$

A matriz $A_{m\times m}$ desse sistema de equações é a matriz companheira do polinômio $p(x)$ dado em (4).

Podemos notar agora que o sistema de equações diferenciais (12) produz uma nova relação de recorrência dada por

$$Y^{(n+1)}(t) = AY^{(n)}(t) = A^nY(t), \qquad n = 0, 1 \ldots, \tag{13}$$

em que n denota a ordem da derivada. Note que aqui basta que a matriz $A_{m\times m}$ seja real ou complexa. Uma aplicação dessa equação é na modelagem matemática simplificada do movimento de pêndulos.

Tomando a série de Taylor da função $Y(t)$, e denotando a matriz identidade $n \times n$ por I, temos

$$\begin{aligned} Y(t) &= Y(t_0) + Y'(t_0)(t-t_0) + Y''(t_0)\frac{(t-t_0)^2}{2!} + \cdots + Y^{(m)}(t_0)\frac{(t-t_0)^m}{m!} + \cdots \\ &= \left(I + A(t-t_0) + A^2\frac{(t-t_0)^2}{2!} + \cdots + A^m\frac{(t-t_0)^m}{m!} + \cdots\right)Y(t_0) \end{aligned}$$

Com vislumbres da série de Taylor de e^t definimos a expressão

$$e^{At} = I + At + A^2\frac{t^2}{2!} + \cdots + A^m\frac{t}{m!} + \cdots \tag{14}$$

E, se conhecermos o vetor $Y(t_0)$, obtemos uma solução da Equação (13) como

$$Y(t) = e^{A(t-t_0)}Y(t_0).$$

Com um pouco mais de manipulações, dado algum vetor v, podemos resolver também a equação

$$Y^{(n+1)}(t) = AY^{(n)}(t) = A^nY(t) + v.$$

Note ainda que a junção das expressões (11) e (13) produz a expressão

$$A^n - \sum_{i=1}^{m} b_iA^{n-i} = 0, \qquad n = m, m+1, \ldots \tag{15}$$

que é uma versão particular do teorema de Caylay-Hamilton, de 1858, cuja demonstração completa foi feita em 1878 por Ferdinand Georg Frobenius. Esse teorema garante que, para toda matriz $A_{n\times n}$, com coeficientes reais ou complexos, existem $b_1, b_2, \ldots, b_n$ tais que a Equação (15) seja verdadeira. Isso produz o polinômio (9), chamado de polinômio característico de A (veja [3] para um pouco da teoria existente sobre esse tema).

Podemos usar as Identidades de Newton, que seguem das relações de Girard, para determinar os coeficientes b_i da Expressão (15). Para isso, defina

$$p_k = \sum_{i=1}^{m} \lambda_i^k = \mathrm{tr}(A^k),$$

sendo $\mathrm{tr}(A)$ o traço de A, que é a soma dos elementos da diagonal principal, e λ_i uma raiz do

polinômio (9), chamado de autovalor de A. Dessa forma, definimos

$$\begin{cases} d_0 &= 1 \\ d_1 &= -p_1 \\ d_2 &= -\left(\frac{d_0p_2+d_1p_1}{2}\right) \\ d_3 &= -\left(\frac{d_0p_3+d_1p_2+d_2p_1}{3}\right) \\ \vdots & \vdots \quad \vdots \\ d_m &= -\left(\frac{d_0p_m+d_1p_{m-1}+\ldots+d_{m-1}p_1}{m}\right) \end{cases} \tag{16}$$

para qualquer que seja a matriz $A_{m\times m}$ (veja o Toerema de Newton em [4]). Tome $b_i = -d_i$, para $i = 1, 2, \ldots, m$ (veja um exemplo no Apêndice A.3).

Segue das expressões (14) e (15) que

$$e^{At} = c_0(t)I + Ac_1(t) + A^2c_2(t) + \cdots + A^{m-1}c_{m-1}(t),$$

em que $c_i(t)$ é um número complexo que depende de t e das raízes de $p(x)$. Essa expressão segue do cálculo funcional, que permite escrever

$$f_t(z) = e^{zt} = p(z)q_t(z) + s_t(z),$$

sendo $s_t(z)$ um polinômio em z e de grau menor que $p(z)$, que pode ser determinado pelas relações $f_t(\lambda_i) = e^{\lambda_i t} = s_t(\lambda_i)$. Isso determina as funções $c_i(t)$ e cria mais uma motivação para determinarmos os valores de λ_i, uma vez que $f_t(A) = e^{At} = s_t(A)$.

Outra motivação, bem mais conhecida, é dada pela resolução de equações diferenciais por meio da transformada de Laplace, frações parciais e a soma (1).

5. Métodos iterativos para aproximação de raízes de polinômios

Quando observamos a Expressão (8) para a_n dada em função da raízes do polinômio $p(x)$, podemos justificar o método de Bernoulli, que pode ser formalizado pelo resultado seguinte ([7, 10, 15]).

Teorema 1. *Seja*

$$p(x) = 1 - \sum_{i=1}^{m} b_i x^i,$$

e a_n *uma sequência definida pela relação de recorrência*

$$a_n - \sum_{i=1}^{m} b_i a_{n-i} = 0, \qquad n \geq 1,$$

com a_0 *dado (e não nulo) e* $a_j = 0$, *para* $j < 0$. *Se*

$$p(x) = c(x - r_1)(x - r_2)\cdots(x - r_m),$$

com $|r_1| < |r_2| \leq \cdots \leq |r_m|$. *Então,*

$$\frac{a_{n+1}}{a_n} \to \frac{1}{r_1}.$$

Demonstração. Basta observar que, nas condições do enunciado e para n grande,

$$a_n = \left(\frac{1}{r_1}\right)^n \left(A_1 + A_2\left(\frac{r_1}{r_2}\right)^n + \ldots + A_m\left(\frac{r_1}{r_m}\right)^n\right) \approx A_1\left(\frac{1}{r_1}\right)^n.$$

□

Jacques Hadamard (1865 - 1963), em sua tese de doutorado de 1892, demonstrou uma versão desse resultado que pode fornecer uma forma de aproximar todas as raízes de p(x) ao mesmo tempo.

Corolário 1. *Defina* $H_0^n = 1$, $H_1^n = a_n$ *e* H_k^n *como determinante de Hankel*

$$H_k^n = \begin{vmatrix} a_n & a_{n+1} & \cdots & a_{n+k-1} \\ a_{n+1} & a_{n+2} & \cdots & a_{n+k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n+k-1} & a_{n+k} & \cdots & a_{n+2k-2} \end{vmatrix}, \qquad k = 2, 3, \ldots m.$$

i) Se $|r_k| < |r_{k+1}|$, *então,*

$$\frac{H_k^{n+1}}{H_k^n} \to \frac{1}{r_1 r_2 \ldots r_k}, \qquad n \to +\infty.$$

ii) Se $|r_{k-1}| < |r_k| < |r_{k+1}|$, *então,*

$$q_k^n = \frac{H_k^{n+1}}{H_k^n} \cdot \frac{H_{k-1}^n}{H_{k-1}^{n+1}} \to \frac{1}{r_k} = \lambda_k, \qquad n \to +\infty.$$

Ideia da Demonstração: Para concluir esse resultado, precisamos utilizar a Expressão (8) e propriedades do determinante que define H_k^n. Apresentamos aqui apenas uma ideia da demonstração, supondo, por simplicidade, que m = 2 e que $|r_1| < |r_2|$, segue da Expressão (8) que

$$H_2^n = \begin{vmatrix} a_n & a_{n+1} \\ a_{n+1} & a_{n+2} \end{vmatrix}$$

e, assim,

$$\begin{aligned} H_2^n &= a_n a_{n+2} - a_{n+1}^2 \\ &= \frac{1}{r_1^{n+1} r_2^{n+1}}\left(A_1 r_1 + A_2 r_1\left(\frac{r_1}{r_2}\right)^n\right)\left(\frac{A_1}{r_2}\left(\frac{r_2}{r_1}\right)^{n+2} + \frac{A_2}{r_2}\right) \\ &\quad - \frac{1}{r_1^{n+1} r_2^{n+1}}\left(A_1 + A_2\left(\frac{r_1}{r_2}\right)^{n+1}\right)\left(A_1\left(\frac{r_2}{r_1}\right)^{n+1} + A_2\right) \\ &= \frac{1}{r_1^{n+1} r_2^{n+1}} A_1 A_2\left(\frac{r_1}{r_2} + \frac{r_2}{r_1} + 2\right). \end{aligned}$$

Logo,

$$\frac{H_2^{n+1}}{H_2^n} \to \frac{1}{r_1 r_2}.$$

□

A presença do determinante sugere que esse método é computacionalmente complexo e caro, o que é verdade. Pode-se ainda verificar que esse método é bastante sensível a pequenas perturbações nas raízes do polinômio, e é, então, numericamente instável. Entretanto, Hadamard obteve também uma relação de recorrência (não linear) para H_k^n, que reduz muito o custo computacional do método, mas não deu muita importância a ela, talvez porque não houvesse computador naquela época e as calculadoras mecânicas eram pouco desenvolvidas.

Como o determinante de uma matriz surge de um processo de análise combinatória, não é estranho que resultados envolvendo H_k^n e os resultados finais desta seção tenham sido demonstrados mais de 100 anos depois por meio de uma interpretação combinatória envolvendo frações contínuas (veja [14] e referências).

Teorema 2. *Se* H_k^n *é definida como no Teorema 1, então,*

$$(H_k^n)^2 - H_k^{n-1}H_k^{n+1} + H_{k+1}^{n-1}H_{k-1}^{n-1} = 0.$$

Essa relação foi redescoberta na década de 1920 por Aitken, que estava trabalhando no problema de determinação de raízes de polinômios (veja [7]). Ela nos ajuda a calcular H_k^n, usando apenas os termos a_j, sem a necessidade do cálculo de determinantes, através da tabela

$$\begin{array}{cccccc}
1 & & & & & \\
1 & H_1^0 & & & & \\
1 & H_1^1 & H_2^0 & & & \\
1 & H_1^2 & H_2^1 & H_3^0 & & \\
1 & H_1^3 & H_2^3 & H_3^1 & H_4^0 & \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

cujo preenchimento da coluna k pode ser feito usando as colunas $k-1$ e $k-2$, ou seja,

$$H_{k+1}^{n-1} = \frac{H_k^{n-1}H_k^{n+1} - (H_k^n)^2}{H_{k-1}^{n-1}},$$

caso $H_{k-1}^{n-1} \neq 0$, que é um quociente de uma diferença.

Heinz Rutishauser[2], na década de 1950, manipulou a relação na forma anterior e na forma

$$H_{k+2}^n = \frac{H_{k+1}^n H_{k+1}^{n+2} - (H_{k+1}^{n+1})^2}{H_k^n},$$

para obter um método mais estável e mais leve computacionalmente, o que deu origem ao método quociente de diferenças (veja [8, 14]). Ele introduziu o termo

$$e_k^n = \frac{H_{k-1}^{n+1}}{H_k^n} \cdot \frac{H_{k+1}^n}{H_k^{n+1}}$$

e observou que o Corolário 1 pode ser reescrito, e produz um algoritmo mais estável.

[2]Para ver mais sobre a importância de Rutishauser no surgimento e desenvolvimento da analise numérica e da computação leia o texto "My years with Rutishauser", de Friedrich L. Bauer.

Teorema 3 (o algoritmo qd)**.** *Suponha que* $b_i \neq 0$. *Seja* $q_0^1 = -\frac{b_{m-1}}{b_m}$, $e_0^n = e_m^n = q_k^0 = 0$, *para* $k = 2, 3, \ldots, m$, *e* $e_k^0 = \frac{b_{m-k-1}}{b_{m-k}}$, *para* $k = 1, 2, \ldots, m-1$. *Então,*

$$q_k^{n+1} + e_{k-1}^{n+1} = q_k^n + e_k^n, \qquad e_k^{n+1} = e_k^n \frac{q_{k+1}^n}{q_k^{n+1}}.$$

Se e_k^n *convergir para 0, então,* q_k^n *converge para* λ_k.

O que é mais interessante e surpreendente no trabalho de Rutishauser é que ele observou que os valores de q_k^n propostos por Hadamard (sessenta anos antes) poderiam ser calculados utilizando o processo de decomposição LU (ou LR em Alemão) de uma matriz

$$A_n = L_n R_n = R_{n-1} L_{n-1}, \qquad n = 1, 2, \ldots, \tag{17}$$

sendo

$$L_n = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ e_1^n & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & e_{m-1}^n & 1 \end{bmatrix}, \qquad R_n = \begin{bmatrix} q_1^n & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & q_{m-1}^n & 1 \\ 0 & 0 & \ldots & 0 & q_m^n \end{bmatrix}.$$

Isso permite também uma estimativa de λ_i através do simples belo teorema dos círculos de Gershgorin.

O algoritmo qd pode ser usado na determinação de polos de funções e de autovalores generalizados, em problemas de interpolação, entre outros, como pode ser visto no trabalho [9], de 2007, e em suas citações.

5.1. O problema de determinação de autovalores

A Expressão (1) pode ser usada para o cálculo aproximado da matriz inversa de uma dada matriz $A_{m\times m}$, quando os seus autovalores possuírem módulo menor que 1. Isso pode ser feito através da identidade

$$(I - A)^{-1} = I + A + A^2 + A^3 + \cdots = \sum_{n=0}^{+\infty} A^n,$$

em que $I_{m\times m}$ denota a matriz identidade. Isso permitiu a análise do problema de determinar os autovalores de A sem a determinação do polinômio característico, tanto por Hadamard quanto por Aiken e Rutishauser. Para isso eles trabalharam com a sequência $a_n = u^t A^{n+1} v$, para algum $u, v \in \mathbb{R}^m$, e com a função geradora

$$F(x) \quad = \quad u^t (I - xA)^{-1} v,$$

em que u^t denota o vetor coluna u. Isso é uma forma de obter uma a explicação para o método das potências para determinação de um autovetor de A, o que ajuda no entedimento do Algoritmo PageRank do Google. Note que quando A for uma matriz simétrica, com um autovalor dominante λ_1, então,

$$\frac{a_{n+1}}{a_n} = \frac{u^t A^{n+1} v}{u^t A^n v} \to \lambda_1.$$

Rutishauser também descobriu uma evolução do algoritmo qd para matrizes, que, em alguns casos, os autovalores de A poderiam ser determinados pela Expressão (17), iniciando com a matriz tridiagonal

$$A_0 = \begin{bmatrix} \alpha_1 & 1 & \cdots & 0 & 0 \\ \beta_1 & \alpha_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \alpha_{m-1} & 1 \\ 0 & 0 & \dots & \beta_{m-1} & \alpha_m \end{bmatrix},$$

que pode ser obtida de A por processo de eliminação de Gauss (veja também o algoritmo de Lanczos para a determinação de autovalores de A). Em muitos casos isso funciona aplicando a Expressão (17), tomando $A_0 = A$. É claro que isso pode ser usado para a matriz companheira de p(x) ([7]).

Enquato trabalhava na programação de um agoritmo computacional para aproximar autovalores de matrizes (para simulações de aviões), adaptando as ideias de Rutishauser, John Francis criou em 1959, a partir do método LR, o algoritmo QR, e o algoritmo QR com shifts, para o cálculo de autovalores complexos usando apenas aritmética real. Esse último está entre os 10 mais importantes algoritmos da Análise Numérica que ainda está em uso e ainda gera discussões sobre o porquê do seu funcionamento. As tentativas de explicações utilizam as mais avançadas ferramentas de Álgebra Linear disponíveis (veja [15], de 2011).

Com a criação do algoritmo QR e o avanço nas aplicações de autovalores e autovetores (sugiro leitura sobre a LAPACK - Linear Álgebra PACKage), a determinação de raízes de polinômios, sem o uso de matrizes, ficou em segundo plano, mas tem grande valor histórico e gera muitas ideias!

Autores como [7] acreditam que novos argumentos e técnicas envolvendo computação paralela podem fazer com que os algoritmos LR ou qd tornem-se mais vantajosos que os ligados ao algoritmo QR.

5.2. O cálculo de autovalores como um problema de valor inicial

Um outro olhar para o problema de determinação de autovalores aparece no trabalho [11], de 1985. Esta seção é dedicada a tal forma. Para isso, considere uma matriz $L_{m\times m} = L(t)$ com entradas reais e contínuas reais ou complexas, para t número real, e denote por L^* a sua adjunta, que no caso é a sua transposta conjugada. Defina

$$B = B(t) = L^*(t)_+ - L(t)_-,$$

em que A_+ denota a parte triangular superior e A_- denota a parte triangular inferior de uma matriz A, sem a diagonal principal. Segue que $B^* = -B$.

Considere agora a equação diferencial matricial dada por

$$\frac{dU}{dt} = BU, \qquad U(0) = I, \tag{18}$$

em que I denota a matriz identidade de ordem $n \times n$.

Podemos utilizar o teorema do ponto fixo de Banach para garantir que existe uma única função U(t) que satisfaz a equação diferencial (18), para todo número real $t \geq 0$.

Utilizando a regra do produto temos que

$$\frac{dU^*U}{dt} = \left(\frac{dU}{dt}\right)^* U + U^*\frac{dU}{dt} = (BU)^*U + U^*BU = -U^*BU + U^*BU = 0,$$

e segue que $U^*(t)U(t) = U^*(0)U(0) = I$, ou seja, $U(t)$ e $Q(t) = U^*(t)$ são matrizes limitadas e ortogonais (unitárias) definidas para todo número real $t \geq 0$.

Isso garante que a equação diferencial

$$\frac{dL}{dt} = BL - LB, \qquad L(0) = L_0, \tag{19}$$

tem solução única dada por

$$L(t) = Q^*(t)L_0Q(t),$$

para U satisfazendo a Equação (18). Em particular, $L(t)$ é limitado, para todo $t \geq 0$.

A Equação (19) produz uma forma de aproximar autovalores de matrizes autoadjuntas (veja [11]).

Teorema 4. *Seja L_0 uma matriz autoadjunta. Então, a função $L(t)$ possui os mesmos autovalores que L_0 e converge para uma matriz diagonal, quando t cresce.*

Demonstração. Como $L(t)$ é solução da Equação (19), vemos que é limitada, autoadjunta e uniformemente contínua. Alguns cálculos mostram que

$$\sum_{k=1}^{i} \frac{dL_{kk}}{dt} = 2\sum_{k=1}^{i}\sum_{j=k}^{m} |L_{kj}|^2 - 2\sum_{k=1}^{i}\sum_{j=1}^{k} |L_{kj}|^2 = 2\sum_{k=1}^{i}\sum_{j=i+1}^{m} |L_{kj}|^2, \quad 1 \leq k < m.$$

Disso, e da integração de ambos os lados da expressão anterior, segue que

$$\int_0^s |L_{kj}(t)|^2 dt \leq \sum_{k=1}^{m} (L_{kk}(s) - L_{kk}(0)) \leq M < +\infty.$$

Isso implica que

$$\lim_{t\to+\infty} L_{kj}(t) = 0, \quad k \neq j.$$

Então, a derivada de $L_{kk}(t)$ converge para 0 e $L_{kk}(t)$ converge, quando $t \to +\infty$. □

Com argumentos um pouco mais sofisticados podemos mostrar que, se k autovalores de L_0 tiverem a forma $\mathrm{Re}(\lambda_1) > \mathrm{Re}(\lambda_2) > \cdots > \mathrm{Re}(\lambda_k)$, então, para $|t|$ grande, a menos de permutação na diagonal,

$$L(t) \approx \begin{bmatrix} \lambda_1 & \cdot & \cdots & \cdot \\ 0 & \lambda_2 & \cdots & \cdot \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & L_k(t) \end{bmatrix}_{m\times m}.$$

Esses argumentos podem ser usados em uma interpretação para a convergência ou não do algoritmo QR, em alguns casos ([11]).

6. Comentários finais

Este texto tem um tema definido pelo seu título, mas o tema não é o que mais importa para o autor. Na realidade, o que se pretende é discutir como ideias simples podem evoluir para ideias revolucionárias, se investirmos nelas e as deixarmos seguir os caminhos aleatórios que surgem quando da sua divulgação e análise. Muito do que gostaria de dizer aqui tem a ver com a necessidade de mais flexibilidade e abertura para ideias novas ou diferentes, além de conhecimento e cultura matemática, as quais um estudante ou pesquisador da área de matemática precisa ter, principalmente se estiver interessado no que chamam por aí de Matemática Aplicada. A matemática sempre esteve ligada aos cálculos que surgem no dia a dia, depois nos modelos matemáticos e nos computadores (que eu vejo como um modelo matemático também). Enfim, escrevo este texto com muita coisa que parece matemática pura, mas quero que busquem voltar o olhar, sempre que possível, para o fato de que alguém construiu toda a tecnologia que usamos, o GeoGebra, o WolframAlpha, o Google (e o PageRank), o Facebook,...., e antes de construírem, eu garanto, fizeram muitos cálculos e projetos e usaram muitos teoremas, mas o que fez e faz a coisa funcionar são os algoritmos.

Agradecimentos

Agradeço ao professor Marcelo Moreira pela leitura da versão inicial deste texto, que utilizei em minicurso que ministrei no III Workshop do Programa de Pós-Graduação em Estatística Aplicada e Biometria da Universidade Federal de Alfenas. Agradeço também minha esposa Rejane Siqueira Julio que sempre me apoia quando decido escrever algo.

Referências

[1] Bezerra, F. D. M. e Ramos, M. W. A. *Métodos de Euler e Runge-Kutta de 4ª ordem por meio de um applet do Geogebra*. Professor de Matemática Online, v.8, nº1, 21–42, 2020.

[2] Fabbi, M. *Brincando com Ladrilhos*. Revista do Professor de Matemática, nº84, pp.11–17, 2014.

[3] Ferreira, J. C. e Silva, N. R. *Sobre autovalores e zeros de polinômios e problemas de otimização: uma síntese e demonstrações*. Sigmae, v. 8, nº 1, pp. 1–15, 2019.

[4] Franco, N. B. *Cálculo numérico*, Pearson, 2006.

[5] Garbi, G. G. *O romance das equações algébricas*. Livraria da Física, São Paulo, 4ª ed., 2010.

[6] Graham, R. L., Knuth, D. E. and Patashnik, O. *Concrete Mathematics*. Reading, MA: Addison-Wesley, 1994.

[7] Gutknecht, M. and Parlett, B. *From qd to LR, or, How were the qd and LR algorithms discovered?* IMA J. Numer. Anal., 2009.

[8] Henrici, P. and Watkins, B. O. Finding zeros of a polynomial by the Q-D algorithm. Communications of the ACM, Volume 8, Issue 9, 1965.

[9] Lee, W.-S. From quotient-difference to generalized eigenvalues and sparse polynomial interpolation. In SNC'07, pages 110–116. ACM, New York, 2007.

[10] Mekwi, W. R. *Iterative Methods for Roots of Polynomials*. Exeter College, University of Oxford, Master Degree Thesis, 2001.

[11] Nanda, T. Differential Equations and the QR Algorithm. SIAM Journal on Numerical Analysis Vol. 22, No. 2, 310–321, 1985.

[12] Silva Filho, J. F. e Pereira, O. E. S. Revisitando as equações de terceiro grau. Professor de Matemática Online, 205–214, 2019.

[13] Thomas, G. B. *Cálculo*. Vol. 2. São Paulo: Pearson Addison-Wesley, 2002.

[14] Viennot, X. G. *A Combinatorial Interpretation of the Quotient-Difference Algorithm*. In: Krob D., Mikhalev A.A., Mikhalev A.V. (eds) Formal Power Series and Algebraic Combinatorics. Springer, Berlin, Heidelberg, 2000.

[15] Watkins, D. S. *Francis's Algorithm*. Journal The American Mathematical Monthly 118, 5, 387-403, 2011.

A. Exemplos numéricos

Ilustramos os temas discutidos no texto com o uso da linguagem de programação R em alguns exemplos.

A.1. Resolução numérica do Problema do empréstimo

Após um mês do empréstimo, o meu amigo terá uma dívida de $d_0(1 + r)$, sendo r a taxa de juros cobrados e $d_0 = 1200$ o valor inicial da dívida. Fazendo, então, o pagamento de 300 reais, a dívida restante será de $d_1 = d_0(1 + r) - 300$. De forma análoga, o meu amigo vai ter uma dívida $d_n = d_{n-1}(1 + r) - 300$, quando n meses tiverem passado. Quando, porém, pagar a sexta prestação já não ficará mais devendo, ou seja, $d_6 = f(r) = 0$.

Segue o código em linguagem R da função f(r), que descreve o valor de sua dívida d_6, em função do valor da taxa de juros mensal r. Essa função é um polinômio de grau 6 escrito como

```
f<-function(r){ p=1200*(1+r)-300; for ( i in 1:5){p=p*(1+r)-300};p }
```

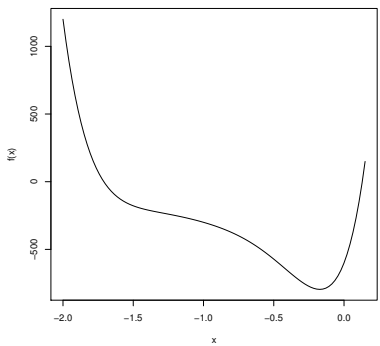

Figura 2: Gráfico da função f(x), gerado pelo comando curve (f,-2,0.15).

Como r é uma taxa de juros, observamos que $0 < r < 1$. Utilizando o método da bissecção

```
a=0; b=1; i=1; while(f(a)*f(b)<0){ i=1+i; r=(a+b)/2;
if (f(a)*f(r)<0){b=r} else{a=r}; if (i==100){break}}; r; f(r)
```

obtemos $r \approx 0.12978$ e $d_6 \approx 0$. Pode-se dizer que taxa de juros mensal é aproximadamente 13%.

Se preferir pode copiar e colar os comandos em linguagem R em https://rextester.com e fazer as simulações mais simples *online*. Se precisar de muitas iterações pode ser necessário istalar o *software* R, disponível em https://www.r-project.org/.

A.2. Resolução numérica do problema da compra do terreno

O raciocínio nesse problema é bem parecido com o do problema anterior. Aqui temos $d_0 = 103900$ e $d_1 = d_0(1 + r) - 30000$, e a partir daí vale a relação $d_n = d_{n-1}(1 + r) - 1159$, para $n = 2, 3, \ldots, 98$, sendo que $d_{98} = f(r) = 0$, que é uma equação polinomial de grau 98. O código de f(r) é

```
f<-function(r){p=103900; p=p*(1+r)-30000; for ( i in 1:97){p=p*(1+r)-1159};p}
```

Aplicamos o método da bissecção novamente para descobrir a raiz de f(r) entre 0 e 1. Segue que a taxa de juros cobrada pelo vendedor é $r \approx 0.8982\%$ mensal e $(1 + r)^{12} - 1 \approx 11.32\%$ anual.

Para determinar o valor da dívida depois de 72 meses reescrevemos a função f trocando o argumento r, pelo número de meses n, e assim definimos a função g(n). O valor da dívida será g(72) = 27677.82 reais.

```
g<-function(n){p=103900; p=p*(1+r)-30000; for ( i in 1:(n-2)){p=p*(1+r)-1159};p}
```

A.3. O teorema de Newton

Aplicamos a Expressão (16) para determinar os coeficientes do polinômio característico da matriz

$$A = \begin{bmatrix} 8 & 1 & 1 & 0 & 2 \\ 0 & 9 & 2 & 1 & 1 \\ 2 & 1 & 11 & 2 & 0 \\ 3 & 2 & 1 & -8 & 1 \\ 2 & 1 & 2 & 3 & 10 \end{bmatrix}. \tag{20}$$

O código de entrada da matriz é dado por

```
c=c(8, 1, 1, 0, 2, 0, 9, 2, 1, 1, 2, 1, 11,2, 0, 3, 2, 1, -8, 1, 2, 1, 2, 3, 10)
A=matrix(c,5,5,,byrow=TRUE); A
```

Definimos a função que calcula a soma dos elementos da diagonal principal ou o traço de uma matriz como

```
tr<-function(A){p=0; n=length(A[1,]); for ( i in 1:n ){p=p+A[i,i]}; p}
```

O cálculo dos coeficientes do polinômio é feito através das linhas de comando em R:

```
coef<-function(A){ n=length(A[1,]); a=0*1:(n+1);s=0*1:n; s[1]=tr(A)
a[1]=1;a[2]=-s[1]; B=A; for ( i in 2:n){B=B%*%A; s[i]=tr(B); q=0
for (j in 1:i){q=q+a[j]*s[i+1-j]}; a[i+1]=-q/i;}; a=(-1)^n*a;a}; coef(A)
```

Os valores determinados pelo algoritmo são -1, 30, -219, -1164, 19532, -59758. Portanto,

$$p(x) = -x^5 + 30x^4 - 219x^3 - 1164x^2 + 19532x - 59758.$$

A.4. Implementação do método de Bernoulli

Apresentamos um exemplo de aplicação do método de Bernoulli para aproximar a menor raiz do polinômio

$$p(x) = 128x^4 - 256x^3 + 160x^2 - 32x + 1.$$

Utilizamos os coeficientes de p(x) para definir a relação de recorrência do Teorema 1.

```
a=0*1:4; a[4]=1; n=50; for (i in 1:n){
a[i+4]=-(128*a[i]-256*a[i+1]+160*a[i+2]-32*a[i+3])}; a; b=a[n-1]/a[n]; b; p(b)
```

Obtemos $r_1 \approx 0.038060233744356624$. Trocando agora $p(x) = 0$ por $x^4p(1/x) = 0$ temos

```
a=0*1:4; a[4]=1; n=50; for (i in 1:n){ a[i+4]=-(-256*a[i+3]+160*a[i+2]-32*a[i+1]
+a[i])/128}; a; b=a[n-1]/a[n]; print(b, digits=22); p(1/b)
```

Isso produz $r_2 \approx 1.0395660380719571$. Para as demais raízes veja método de Newton Bairstow em [4, 10].

A.5. Aproximação pelo método de Hadamard

No que segue implementamos a ideia de Hadamard para determinar raízes isoladas do polinômio

$$p(x) = 24 - 50x + 35x^2 - 10x^3 + x^4.$$

Para obter a sequência recorrente e os determinates de Hankel usamos o código a seguir

```
m=50; a=0*1:(m+4); a[4]=1
for ( i in 5:(m+3)){a[i]=-(-50*a[i-1]+35*a[i-2]-10*a[i-3] +a[i-4])/24}
a=a[4:(m+3)]; a; H=matrix(0,m,5); q=0*1:4; q[1]=a[m-1]/a[m]
for ( i in 1:m){H[i,1]=1}; H[,2]=a
for (k in 1:3){ for (n in 1:(m-3)){H[n,k+2]=(H[n+2,k+1]*H[n+1,k+1]
-H[n+3,k+1]^2)/H[n,k]}; q[k+1]=H[n-1,k+2]/H[n,k+2]; m=m-3 }
for ( k in 2:4){q[k]=q[k]/q[k-1]}; q
```

Verificamos que $q_1^{50} = 1$, $q_2^{50} = 2$ e $q_4^{50} = 3.92143242853595408 \approx 4$ são aproximações para raízes de p(x), e que o método não aproxima a quarta raiz, que é 3. A propagação de erros de arredondamento faz com que o método não produza resultado para $n \geq 63$ porque H_1^{64} fica muito próximo de 0.

Adaptamos esse código para o caso em que $a_n = u^*A^nu$, sendo A a matriz companheira de p(x). Nesse caso, apenas a raiz de maior módulo foi aproximada corretamente.

A.6. O algoritmo qd

Aplicamos o algoritmo qd ao mesmo polinômio do Apêndice A.5, e nesse caso obtemos uma ótima aproximação para as quatro raízes. Segue o código em linguagem R:

```
a=c(24,-50,35,-10,1); n=length(a); e=0*1:n; q=e; q[2]=-a[n-1]/a[n]
for (i in 2:(n-1)){e[i]=a[n-i]/a[n-i+1]}; for (j in 1:100){
for (i in 2:n){q[i]=e[i]-e[i-1]+q[i]}
for (i in 2:(n-1)){e[i]=q[i+1]*e[i]/q[i]}}; q[2:n]; e[2:n]
```

A.7. O método LR

Implementamos uma versão do algoritmo LR aplicado diretamente na Matriz (20), mesmo conhecendo o seu polinômio característico. Grande parte do código é para obter a fatoração LR de A, que é simplesmente o processo de eliminação de Gauss, sem trocar linhas ou colunas:

```
LR<-function(A){ n=length(A[,1]); B=diag(1,n,n); for (j in 1:(n-1)){
for ( i in (j+1):n){ m= A[i,j]/A[j,j]; A[i,]=A[i,]-m*A[j,];A[i,j]=0; B[i,j]=m }}
C=matrix(0,n,2*n);C[,1:n]=B;C[,(n+1):(2*n)]=A; C}
n=length(A[,1]); for (i in 1:1000){C=LR(A);L=C[,1:n];R=C[,(n+1):(2*n)];A=R%*%L};A
```

Os elementos da diagonal da matriz A_{1000} aproximam os autovalores de A, com 7 dígitos exatos, sendo

$$A_{1000} \approx \begin{bmatrix} 13.82553 & 2.498656 & 9.705029 & -0.3816629 & 2.0000000 \\ 0 & 9.644720 & -24.76705 & 1.186106 & -0.9752351 \\ 0 & 0 & -8.249550 & 0.7496636 & -1.1952827 \\ 0 & 0 & 0 & 7.921257 & -25.3361919 \\ 0 & 0 & 0 & 0 & 6.8580457 \end{bmatrix}.$$

A.8. Um problema de valor inicial para determinar autovalores

Para terminar este texto utilizamos método Runge-Kutta de ordem 4 ([1]),

```
RungK4<-function(f,u0,b,n){h=b/n; m=length(u0); Y=u0; for (i in 1:n){ k1=f(Y)
k2=f(Y+h*k1/2); k3=f(Y+h*k2/2); k4=f(Y+h*k3); Y=Y+( k1+ 2*k2 + 2*k3+ k4)*h/6}; Y}
```

para resolver as Equação (18), e obter uma aproximação para L(20), com condição inicial

$$L_0 = \begin{bmatrix} 8 & 1 & 1 & 0 & 2 \\ 20 & 9 & 2 & 1 & 1 \\ 2 & 1 & 11 & 2 & 0 \\ 3 & 2 & 1 & -8 & 1 \\ 2 & 1 & 2 & 3 & 10 \end{bmatrix},$$

para a Equação (19). Usamos os comandos

```
f<-function(U){ m=length(U[,1]); B=U%*%A%*%t(Conj(U)); C=0*B
for( i in 1:m){ for( j in i:m){C[i,j]=Conj(B[j,i]); C[j,i]=-B[j,i]};C[i,i]=0}
p=C%*%U; p} ; Y=RungK4(f,diag(1,5,5),20,10000); Y%*%A%*%t(Conj(Y))
```

e obtemos

$$L(20) \approx \begin{bmatrix} 16.04517 & 4.610904 & -3.075265 & 16.24276. & 0.6358197 \\ 0 & 8.563399 & 2.827115 & -7.692009 & -2.5597661 \\ 0 & -0.03971360 & 8.918768 & 3.321055 & 0.7019277 \\ 0 & 0 & 0 & 4.781560 & 1.2611216 \\ 0 & 0 & 0 & -0 & -8.3088918 \end{bmatrix}.$$

Observamos que essa matriz não é triangular superior. Isso sugere que L_0 possui dois autovalores complexos.

Escolhemos, aleatoriamante, $\alpha = 1 + i$ e trocamos L_0 por αL_0 no nosso código. Isso faz com que αL_0 não possua autovalores com parte real igual e os valores 16.045166, 8.741083+0.284083i, 8.741083-0.284083i, 4.781560 e -8.308892 como aproximações para a diagonal de $L(20)/\alpha$, que são aproximações para os autovalores de L_0.

José Claudinei Ferreira
Universidade Federal de Alfenas
Departamento de Matemática
Alfenas, MG, 37130-000, Brasil
<jose.ferreira@unifal-mg.edu.br>

Recebido: 16/10/2019
Publicado: 02/06/2020