MAP2220 - Fundamentos de Análise Numérica

Prof. Pedro da Silva Peixoto
Prof. Antoine Laurain

EP1 - 2º Sem. 2018

Bacia de atração do Método de Newton
em duas dimensões e Fractais

Data de entrega: 27/09/2018
Enviar para:
ppeixoto@usp.br
laurain@ime.usp.br
paula.neves.araujo@hotmail.com
Assunto: MAP2220-EP1-diurno/noturno

Objetivos

O objetivo deste exercício-programa é resolver sistemas de equações não lineares pelo Método de Newton e determinar a bacia de atração de cada raiz encontrada. Usaremos imagens para representar a bacia de atração, e veremos que com isso é possível construir fractais. As analises, resultados obtidos e as imagens devem ser organizados em um relatório.

Instruções

- O exercício deve ser feito em Python 3.x (o mesmo usado nos cursos de MAC, veja mais detalhes em https://panda.ime.usp.br/panda/python).
- O exercício pode ser feito em duplas, sempre com alguém da mesma turma.
- Apenas um aluno deve entregar o exercício, destacando no relatório e código o nome de ambos os alunos.
- A entrega deve conter o relatório (em .pdf), contendo a análise do problema estudado, e o código usado para as simulações computacionais (arquivos .py). A entrega pode ser feita em um arquivo compactado único.

O seu código deve estar bem comentado e estruturado. A entrada e saída devem ser feitas de forma a ajudar o usuário a executar o programa e deve facilitar a análise dos resultados. Inclua qualquer arquivo adicional necessário para o seu programa no arquivo compactado a ser entregue.

O Método de Newton 2D

Considere o problema de se encontrar uma solução para um sistema de equações não lineares

$$\begin{cases} f(x,y) &= 0 \\ g(x,y) &= 0 \end{cases} \tag{1}$$

Analogamente ao caso unidimensional, o Método de Newton bidimensional é dado pela recorrência

$$\begin{bmatrix} x_{k+1} \\ y_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \end{bmatrix} - \begin{bmatrix} \frac{\partial f}{\partial x} & \frac{\partial f}{\partial y} \\ \frac{\partial g}{\partial x} & \frac{\partial g}{\partial y} \end{bmatrix}^{-1} \cdot \begin{bmatrix} f(x_k, y_k) \\ g(x_k, y_k) \end{bmatrix} \tag{2}$$

A convergência da sequência (x_k, y_k) para uma solução do sistema não linear é de análise delicada, diferente do caso unidimensional. Em geral, a aproximação inicial (x_0, y_0) precisa estar razoavelmente próxima da solução para que o método funcione.

Neste programa você vai usar 2 critérios de parada para o Método de Newton:

- Número máximo de iterações $ITMAX \in \mathbb{N}$
- Limite inferior para a variação relativa $\varepsilon > 0$. Onde a variação relativa será dada por

$$VR_k = \sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2} \tag{3}$$

As derivadas parciais serão calculadas numericamente com diferenças finitas centradas, descrita a seguir. A inversão da matriz será feira explicitamente, afinal de contas, é uma matriz 2×2!

Aproximação numérica da derivada

Dada uma função $f(x)$ diferenciável, sua derivada é o limite $f'(x) = \lim_{h \to 0} \dfrac{f(x+h) - f(x)}{h}$.

Trocando h por $-h$, a derivada é escrita também como o limite $f'(x) = \lim_{h \to 0} \dfrac{f(x) - f(x-h)}{h}$

de modo que temos também a fórmula de diferença central

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$$

Logo, podemos aproximar a derivada em um ponto x como

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Representação de imagens em arquivos texto

Esta seção descreve alguns formatos de representação de imagens em arquivos de tipo texto. Tais formatos, apesar de não serem tão populares como os formatos binários que ocupam muito menos espaço, são usados por uma série de programas de conversão e são convenientes para serem usados como formatos intermediários para saída de programas. Originalmente este formatos eram usados para transmissão de imagens por correio eletrônico.

Estaremos interessados nos chamados formatos PNM (portable anymap), em particular no tipo PPM (portable pixmap file format), que permite imagens coloridas.

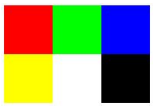
A definição de um arquivo PPM é a seguinte:

- Um "número mágico" para identificar o tipo de arquivo. Um arquivo PPM usa os caracteres "P3".
- Espaço ou troca de linha
- Uma largura, em caracteres texto decimais.
- Espaço ou troca de linha
- Uma altura, em caracteres texto decimais.
- Espaço ou troca de linha
- O máximo nível de componente de cor, em caracteres texto decimais.
- Largura*Altura triplas de níveis de cor entre 0 e o valor máximo especificado separados por espaço, começando do canto superior esquerdo da imagem. Cada valor das triplas corresponde, respectivamente, aos níveis de vermelho, verde e azul. O valor 0 significa que a cor não aparece enquanto que o valor máximo significa que a cor está no nível máximo.

- Linhas que começam com "#" são comentários e portanto são ignoradas.
- Nenhuma linha pode ser mais longa que 70 caracteres.

Por exemplo:

```
P3
# The P3 means colors are in ASCII, then 3 columns and 2 rows,
# then 255 for max color, then RGB triplets
3 2
255
255   0   0     0 255   0     0   0 255
255 255   0   255 255 255     0   0   0
```

corresponde à imagem (ampliada)  .

Para evitar problemas com o número máximo de caracteres por linha você pode imprimir cada pixel em uma nova linha.

Há diversos programas que abrem imagens PPM, para o Windows você pode usar o software gratuito chamado IrfanView (`http://www.irfanview.com/`).

Bacia de atração de uma raiz

Considere um sistema de equações como em (1). Dizemos que um ponto (x_0, y_0) está na *bacia de atração* de uma solução $(\bar{x}, \bar{y})$ se a sequência correspondente do Método de Newton (2) for convergente a $(\bar{x}, \bar{y})$.

Dado um sistema (1) e um retângulo $R = [a, b] \times [c, d]$, o objetivo deste exercício-programa é mapear a bacia de atração das soluções do sistema em R. Usaremos cores para diferenciar as bacias para cada raiz.

Para isso você deverá escrever um programa que cria um arquivo de imagem colorida PPM de tamanho $N \times M$ correspondente ao retângulo R. Assim teremos os pontos (x_i, y_j) dados por

$$x_i = a + (b - a)(i - 1)/M,$$

$$y_j = c + (d - c)(j - 1)/N.$$

Você deve associar a cada raiz obtida com o método de Newton uma cor. Cada ponto de malha (x_i, y_j) será um chute inicial do método de Newton, e deve ser pintado com a cor relativa da raiz para qual o método converge com esse chute inicial.

Pinte de preto o pixel que tiver:

- Chegado em erro no método de Newton por excesso de iterações
- Chegado em erro no método de Newton por zeros na matriz de derivada (determinante nulo)
- Levar a pontos fora do retângulo

Para podermos identificar mais facilmente as raízes, pinte de branco $RGB = (255, 255, 255)$ os pontos muito perto das raízes (2 ou menos pixels de distância).

Os outros pontos podem ser pintados com o esquema que você preferir, mas tente não usar cores muito parecidas. O nível máximo de cada componente deve ser 255. Como sugestão, você pode usar a tabela abaixo. Acrescente mais cores se você achar necessário.

Raiz	RGB	Cor
1	$(255, 0, 0)$	Vermelho
2	$(0, 255, 0)$	Verde
3	$(0, 0, 255)$	Azul
4	$(255, 255, 0)$	Amarelo
5	$(255, 165, 0)$	Laranja
6	$(159, 95, 159)$	Violeta

Para que a figura fique mais interessante, vamos pintar os pixels de forma proporcional ao número de iterações do método de Newton. Logo você deve multiplicar a cor (cada uma das componentes) por um fator que será dado por

$$\lambda = \frac{ITMAX - IT}{ITMAX}, \tag{4}$$

onde $ITMAX$ é número máximo de iterações definida para o método de Newton, e IT é o número de iterações obtida para aquele chute inicial (pixel). Note que se o método atingir o número máximo de iterações (o método diverge), então o fator será zero, e logo a cor será preta.

Testes

Para as derivadas numéricas, utilize $h = 10^{-4}$. Para o método de Newton, use $ITMAX = 40$ e $\varepsilon = 10^{-5}$. Considere o retângulo com $N = 800 \times M = 800$.

Teste seu programa para:

1. $$\begin{cases} x^3 - 3xy^2 - 1 &= 0 \\ 3x^2y - y^3 &= 0 \end{cases}$$

   no retângulo $[-0.7, 1.1] \times [-1, 1]$.

2. $$\begin{cases} x^4 - 6x^2y^2 + y^4 - 1 &= 0 \\ 4x^3y - 4xy^3 &= 0 \end{cases}$$

   no retângulo $[-3, 3] \times [-3, 3]$.

3. $$\begin{cases} \cos(3x^2)y &= 0 \\ \cos(3y^2)x &= 0 \end{cases}$$

   no retângulo $[-1, 1] \times [-1, 1]$.

4. Experimente outras funções. Observe e comente sobre a relação do que foi feito nos testes anteriores com o descrito neste site: `https://en.wikipedia.org/wiki/Newton_fractal` . Construa pelo menos um Fractal adicional aos ítens anteriores.

Entrada

O seu programa deve ler os os parâmetros de uma arquivo de entrada, demoninado "entrada.txt". este arquivo deve conter as seguintes informações, nesta ordem e por linhas:

- Número do teste (função a ser testada)
- Os extremos dos intervalos de definição da função (a, b, c, d), onde f está definida em $[a, b] \times [c, d]$
- O espaçamento para a derivada discreta (h)
- O número máximo de iterações ($ITMAX$) e a tolerância (ε) do método de Newton
- Número de linhas e colunas da imagem

Um arquivo de exemplo para o teste 1 é:

```
1                    # Numero do teste (funcao)
-1.0 1.0 -1.0 1.0    # a, b, c, d (caixa do problema)
1e-4                 # Espaçamento para derivada discreta
40 1e-5              # Num max de iterações e tolerância (Newton)
800 800              # Num linhas, num colunas
```

Saída

A saída do seu programa deve conter uma impressão de um cabeçalho com seu nome e número usp, e para cada teste, a impressão de cada uma das raízes encontradas e qual a cor que será atribuída a ela. Além disso, o seu programa deve gerar a figura, em formato PPM, correspondente ao teste executado (ex: saida.ppm).

As imagens obtidas terão características bem interessantes, sendo que os dois primeiros testes geram fractais. Um exercício interessante é rodar o método para sub-retângulos da imagem, e verificar que os padrões se repetem infinitamente (pode ser preciso ajustar os parâmetros para ver os detalhes dos sub-níveis do fractal). Fique à vontade para usar sua criatividade e criar outras imagens além dessas!

Dicas

- A princípio você não conhece o número de raízes de cada problema, portanto você deve ir armazenando em uma lista (vetor) cada nova raiz encontrada. Você pode assumir que o seu método acha no máximo 20 raízes, para poder definir um tamanho máximo inicial para essa lista.
- Para saber para qual raiz o seu chute inicial convergiu, você pode testar se a distância do ponto para o qual você teve convergência até uma das raízes já encontradas é suficientemente pequeno. Pode usar uma tolerância de $\varepsilon = 10^{-5}$.
- Vocês podem mostrar suas imagens na monitoria ou em aula para verificar se seus resultados fazem sentido.

Bons Estudos!