Cálculo Numérico

Em um exercício anterior, fomos capazes de calcular derivadas para funções polinomiais (representadas como listas de coeficientes). Neste exercício, exploraremos algumas opções para aproximar derivadas e integrais para funções arbitrárias numericamente.

1) Derivados

Uma maneira de aproximar a derivada de uma função ff em um ponto xx é com a seguinte fórmula:

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

para algum valor pequeno de δ.

Observe que esta é uma aproximação linear. Ela encontra dois pontos na função original que estão próximos um do outro (e simétricos em relação ao ponto onde queremos calcular a derivada) e retorna a inclinação da linha que conecta esses dois pontos. Embora existam maneiras melhores de aproximar as derivadas, na prática esse método pode funcionar bem.

Defina uma função `approx_derivative(f, x, delta=1e-6)` que recebe uma função `f` e um número `x` (e, opcionalmente, um segundo número, `delta`) e retorna a derivada aproximada de `f` em `x` (usando `delta` como o valor δ na expressão se for fornecido, e usando 10^{-6} como o valor, do contrário).

No mesmo arquivo, vamos experimentar um pouco com uma forma ligeiramente diferente. Para este fim, defina uma segunda função `approx_derivative_2 (f, delta = 1e-6)`. Esta função deve receber uma função `f` e um número `delta` conforme descrito acima. Ao contrário do acima, porém, `approx_derivative_2` deve retornar uma *função* que representa f'. Esta nova função deve receber um único argumento representando x, e sua saída deve ser uma aproximação de $f'(x)$ calculado usando a aproximação acima.

Tente agora:

O que há de diferente nessas duas formas e o que é igual? Quando podemos preferir um em vez do outro?
Mostrar/Esconder

Você vai querer testar alguns valores em sua própria máquina antes de enviar seu código. Você pode testar seu código computando algumas derivadas (de polinômios, funções trigonométricas, etc) simbolicamente e avaliando-as em pontos específicos. Ao usar essas mesmas funções em seu código, você deve obter um resultado semelhante.

2) Integrais

Existem várias maneiras diferentes de aproximar numericamente a integral de uma função arbitrária, mas não iremos examiná-las aqui. Em vez disso, ficaremos com a boa e velha soma de Riemann. Em particular, usaremos a formulação que sabemos ser uma aproximação decente: usando a *regra*

trapezoidal.

Defina uma função `approx_integral(f, lo, hi, num_regions)` que aproxima a integral da função `f` entre os limites `lo` e `hi`. Seu código deve dividir esta região em várias regiões de tamanhos iguais (na dimensão x), com número especificado por `num_regions`. Você deve então aproximar a área sob a curva em cada região usando a regra trapezoidal e somar essas áreas para produzir sua aproximação.

Como antes, é uma boa ideia tentar alguns casos de teste em sua própria máquina antes de enviar seu código aqui. Você pode testar seu código computando algumas integrais definidas (de polinômios, funções trigonométricas, etc.) simbolicamente e avaliando-as em pontos específicos. Ao conectar essas mesmas funções em seu código, você deve obter um resultado semelhante.

Submissão

Quando estiver pronto (depois de ter simulado manualmente e testado em sua própria máquina e estiver convencido de que seu programa fará a coisa certa), faça upload do seu arquivo Python no **Problema 3.5** no Gradescope. Lembre de nomear seu arquivo `p3 5.py`.