PRACTICE 6: Numerical differentiation

Numerical differentiation is a method for approximating the derivative of a function f at a point $x \in [a, b] \subset \mathbb{R}$ using only the values of f at a finite number of points close to x. This approach is particularly useful when:

1. The values of the function f are only known at a discrete set of points x.
2. Computing the approximation is cheaper or more convenient than obtaining the exact derivative.

1 First order derivatives

The derivative of a function f at a point x is defined by the following limit:

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

However, in numerical differentiation, the step size h is chosen to be a small but finite number, allowing the derivative to be approximated by:

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

This approximation is known as the ***forward difference formula*** and is one of the simplest numerical differentiation methods. The accuracy of this approximation depends on the size of the step h and the smoothness of the function f.

Thus the three numerical differentiation schemes are as follows:

Forward Difference Formula

The *forward difference* formula approximates the derivative using the difference between the function value at a point and the function value at a small step ahead:

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

This formula is *first-order accurate*.

Backward Difference Formula

The *backward difference* formula uses the difference between the function value at a point and the function value at a small step behind:

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

This formula is also *first-order accurate*.

Central Difference Formula

The *central difference* formula uses both forward and backward differences to provide a more accurate estimate of the derivative:

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

This formula is *second-order accurate*, meaning it provides a more accurate approximation than the forward and backward difference formulas for the same step size h.

Example 1.1 *Consider the function:*

$$f(x) = x^2$$

Let us approximate the derivative at $x = 2$ using the forward difference formula with $h = 0.1$:

$$f'(2) \approx \frac{f(2+0.1) - f(2)}{0.1} = \frac{(2.1)^2 - 2^2}{0.1} = \frac{4.41 - 4}{0.1} = 4.1$$

The exact derivative is:

$$f'(x) = 2x, \quad f'(2) = 4$$

The numerical approximation is close to the exact result, with a small error due to the finite step size h.

1.1 Edge Problem

A common issue with numerical differentiation schemes is the **edge problem**, which arises when attempting to approximate derivatives at the boundaries of the interval $[a, b]$. Finite difference formulas require function values at points beyond the boundary, which are often unknown.

For example, using the forward difference formula at the right boundary $x = b$ would require the function value $f(b + h)$, which is typically unavailable. Consequently, special techniques such as backward differences or extrapolation must be employed at the boundaries.

Example 1.2 *Consider approximating the derivative of $f(x) = x^2$ at $x = 5$ in the interval $[1, 5]$ using the forward difference formula with $h = 0.1$.*

Since $5 + 0.1 = 5.1$ is outside the interval, the forward difference formula cannot be applied directly. Instead, the backward difference formula can be used:

$$f'(5) \approx \frac{f(5) - f(5-0.1)}{0.1} = \frac{25 - (4.9)^2}{0.1} = \frac{25 - 24.01}{0.1} = 9.9$$

The exact derivative is:

$$f'(x) = 2x, \quad f'(5) = 10$$

Again, the approximation is close to the exact result, with a small error due to the step size h.

In MATLAB, the forward difference formula is not directly applicable at $x = 5$, so we use the backward difference formula. The error between the approximation and the exact derivative is computed and displayed.

The code also generates a plot of the exact derivative and the approximation for a range of values from $x = 1$ to $x = 5$, and shows the approximation at $x = 5$ using the backward difference method.

MATLAB Code

```
f = @(x) x.^2; % f(x) = x^2
df_exact = @(x) 2*x; % Exact derivative f'(x) = 2x

x = 5; % Point where we are calculating the derivative
h = 0.1; % Step size

% Forward difference (x = 5 --> x + h outside the interval)
% We use the backward difference formula instead at x = 5
df_b = (f(x) - f(x - h)) / h; % Backward difference

% Exact derivative at x = 5
df_exact_value = df_exact(x);

xq = 1:0.1:5; % Points in [1, 5]

% Plot the exact derivative
figure;
plot(xq, df_exact(xq), 'k-', 'LineWidth', 2);
hold on;
plot(x, df_b(x), 'ro', 'MarkerSize', 8);
```

Exercise 1.1 *Given the function* $f(x) = \ln(x)$ *in the interval* $[1, 2]$ *and for* $h = 0.1$ *and* $h = 0.01$*:*

- *Calculate and plot its exact derivative.*
- *Compute the approximate derivative using the forward* df_f*, backward* df_b*, and centered* df_c *approximations in equispaced points, with a space between them of* h*, in the interval* $[1, 2]$*. You cannot use points outside this interval. Plot the result alongside the exact derivative.*
- *In a new figure, compute and plot the error for each point for the approximations,* $|f'(x_i) - df_a(x_i)|$*, where* x_i *are the points and* $a = p, r,$ *or* c*.*
- *Compute the relative error for each approximation using the formula:*

$$Error_{Rel} = \frac{\|f'(x_i) - df_a(x_i)\|_2}{\|f'(x_i)\|_2} = \frac{\sqrt{\sum_{i=1}^{n} \left(f'(x_i) - df_a(x_i)\right)^2}}{\sqrt{\sum_{i=1}^{n} \left(f'(x_i)\right)^2}}$$

MATLAB Code

```
f = @(x) log(x); % f(x) = ln(x)
df_exact = @(x) 1 ./ x; % Derivada exacta f'(x) = 1/x

% Intervalo [1, 2] y valores de h
a = 1; b = 2;
h_values = [0.1, 0.01];% 2 values h1, h2

% Define the points in [a,b] where we compute:
xq_1 =a:h_values(1):b; % with h1
xq_2 =a:h_values(2):b; % with h2

figure;

for i = 1:length(h_values)
 h = h_values(i);
 % Puntos de cÃ¡lculo
 if i == 1
  xq = xq_1;
 else
  xq = xq_2;
 end
 % Derivatives Approximations:
 df_f = (f(xq + h) - f(xq)) / h; %
 df_b = (f(xq) - f(xq - h)) / h; %
 df_c = (f(xq + h) - f(xq - h)) / (2*h);
 % Derivada exacta
 df_exact_values = df_exact(xq);

 subplot(2, 1, 1),hold on
 plot(xq, df_exact_values, 'k', 'LineWidth', 2) % Exacta
 plot(xq, df_f, 'b--', 'LineWidth', 1)
 plot(xq, df_b, 'r--', 'LineWidth', 1)
 plot(xq, df_c, 'g--', 'LineWidth', 1)

 legend('Exact', 'Forward', 'Backward', 'Central'), hold off

 % Absolut errors:
 error_f = abs(df_exact_values - df_f);
 error_b = abs(df_exact_values - df_b);
 error_c = abs(df_exact_values - df_c);

 subplot(2, 1, 2), hold on
 plot(xq, error_f, 'b--', 'LineWidth', 1); % Error Forward
 plot(xq, error_b, 'r--', 'LineWidth', 1); % Error Backward
 plot(xq, error_c, 'g--', 'LineWidth', 1); % Error Central

 legend('Error Forward', 'Error Backward', 'Error Central');
 hold off;
end
```

Is it possible to get order 2 approximations for $f'(a)$ and $f'(b)$? Yes, if we use a Lagrange interpolation polynomial of f of second degree at x_0, x_1, and x_2, with $h > 0$, $x_1 = x_0 + h$, $x_2 = x_1 + h$, and $y_j = f(x_j)$.

Then we have the second-order approximation formulas:

$$\text{Forward:} \quad f'(x_0) \approx \frac{-3y_0 + 4y_1 - y_2}{2h}$$

$$\text{Centered:} \quad f'(x_1) \approx \frac{-y_0 + y_2}{2h}$$

$$\text{Backward:} \quad f'(x_2) \approx \frac{y_0 - 4y_1 + 3y_2}{2h}$$

---

2 Second order derivatives

Central Difference for Second Derivative For a function $f(x)$, the second-order derivative at a point x_0 can be approximated using the central difference formula:

$$f''(x_0) \approx \frac{f(x_0 + h) - 2f(x_0) + f(x_0 - h)}{h^2}$$

where h is the step size. This method uses function values at three points: $x_0 - h$, x_0, and $x_0 + h$.

Example 2.1 *Let's consider a simple function for which we want to approximate the second derivative:*

$$f(x) = x^2$$

The exact second derivative of $f(x) = x^2$ *is:*

$$f''(x) = 2$$

Now, let's approximate $f''(x)$ *at* $x_0 = 2$ *using the central difference method with* $h = 0.1$.

MATLAB Code

```
% Define the function f(x)
f = @(x) x.^2; % f(x) = x^2

% Choose the point x0 and step size h
x0 = 2;
h = 0.1;

% Calculate the second-order derivative approx.
f_double_prime_approx = (f(x0 + h) - 2*f(x0) + f(x0 - h)) / h^2;

% Exact second derivative
f_double_prime_exact = 2;


% Plot the function and its approximation
x_vals = linspace(1, 3, 100); % Points for plotting
y_vals = f(x_vals);
f_prime_approx = (f(x_vals + h) - f(x_vals - h)) / (2 * h);

figure;
subplot(2,1,1);
plot(x_vals, y_vals, 'b-', 'LineWidth', 2);
title('Function f(x) = x^2');
subplot(2,1,2);
plot(x_vals, f_prime_approx, 'r-', 'LineWidth', 2);
title('First Derivative Approximation (Central Difference)')
```

Exercise 2.1 *Given the function:*

$$f(x) = \cos(2\pi x)$$

in the interval $[-1, 0]$*, with step size* $h = 0.001$*, we want to:*

1. *Compute and plot the exact second derivative.*
2. *Approximate the second derivative using the central difference formula:*

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

3. *Calculate the relative error of the approximation in the interval* $[-1 + h, -h]$*.*

The exact second derivative of the function is:

$$f''(x) = -4\pi^2 \cos(2\pi x)$$

The relative error is defined as:

$$\text{Relative Error} = \frac{|f''(x) - \tilde{f}''(x)|}{|f''(x)|}$$

where $\tilde{f}''(x)$ is the numerical approximation.

MATLAB Code

```
% Parameters
h = 0.001;
x = -1:h:0;
f = @(x) cos(2*pi*x);
d2f_exact =@(x) -4*pi^2*cos(2*pi*x);

% Numerical approximation using central difference
f_approx = (f(x + h) - 2*f(x) + f(x - h)) / h^2;
f_approx(1) = 0; % Boundary conditions
f_approx(end) = 0;

% Relative error in the interval [-1+h, -h]
error = abs(f_exact - f_approx) ./ abs(f_exact);
error(1) = NaN; % Exclude boundary points
error(end) = NaN;

% Plotting
figure;
subplot(3, 1, 1);
plot(x, f_exact, 'b', 'DisplayName', 'Exact'),hold on
plot(x, f_approx, 'r--', 'DisplayName', 'Approximate')
title('Second Derivative')

subplot(3, 1, 2);
plot(x, error, 'k');
title('Relative Error')

subplot(3, 1, 3);
plot(x, f, 'g'), title('Original Function')
```

Exercise 2.2 *Given the function:*

$$f(x) = \sin(3\pi x)$$

in the interval $[0, 1]$*, with step size* $h = 0.002$*, we want to:*

1. *Compute and plot the exact second derivative.*
2. *Approximate the second derivative using the central difference.*
3. *Calculate the relative error of the approximation in the interval* $[h, 1 - h]$*.*

3 Numerical Differentiation of Functions of Several Variables

For the domain $\Omega = [a, b] \times [c, d] \subset \mathbb{R}^2$, consider uniform partitions:

$$a = x_1 < x_2 < \cdots < x_{M-1} < x_M = b$$

and

$$c = y_1 < y_2 < \cdots < y_{N-1} < y_N = d$$

Denote a point in the grid by (x_m, y_n). Then, its neighbor points are:

$$(x_{m-1}, y_n), \quad (x_{m+1}, y_n), \quad (x_m, y_{n-1}), \quad (x_m, y_{n+1})$$

For numerical differentiation, we use finite differences to approximate the partial derivatives of a function $f(x, y)$ at the point (x_m, y_n). The most common formulas for the numerical derivatives are:

$$\frac{\partial f}{\partial x} \approx \frac{f(x_{m+1}, y_n) - f(x_{m-1}, y_n)}{2h_x}$$
$$\frac{\partial f}{\partial y} \approx \frac{f(x_m, y_{n+1}) - f(x_m, y_{n-1})}{2h_y}$$

where $h_x = x_{m+1} - x_{m-1}$ and $h_y = y_{n+1} - y_{n-1}$ represent the step sizes in the x and y directions, respectively.

Grid Scheme

We can visualize the grid of points and their neighbors in the following diagram:

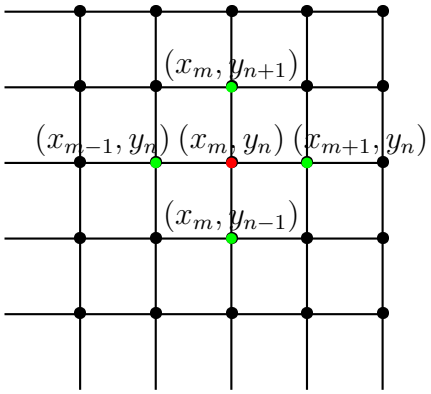

Figure 1: Grid of points and neighboring points for numerical differentiation.

x-Partial Derivative (Forward)

The forward difference formula for the partial derivative with respect to x is given by:

$$\frac{\partial x}{\partial f}(x_m, y_n) = \frac{f(x_m + 1, y_n) - f(x_m, y_n)}{h_x}$$

where h_x is the step size in the x-direction.

Gradient (Centered)

The gradient of $f(x, y)$ at a point (x_m, y_n) is given by the vector of partial derivatives with respect to both x and y:

$$\nabla f(x_m, y_n) = \left(\frac{\partial x}{\partial f}(x_m, y_n), \frac{\partial y}{\partial f}(x_m, y_n) \right)$$

where the centered differences for the partial derivatives are:

$$\frac{\partial x}{\partial f}(x_m, y_n) = \frac{f(x_m + 1, y_n) - f(x_m - 1, y_n)}{2h_x}$$

$$\frac{\partial y}{\partial f}(x_m, y_n) = \frac{f(x_m, y_n + 1) - f(x_m, y_n - 1)}{2h_y}$$

Examples in 2D

We will solve two simple examples in 2D and plot the results using MATLAB.

Example 1: $f(x, y) = x^2 + y^2$

The function $f(x, y) = x^2 + y^2$ is a simple quadratic function. Compute the gradient, using the above formulas for $\frac{\partial f}{\partial x}$ and $\frac{\partial f}{\partial y}$.

Example 2: $f(x, y) = \sin(x) \cdot \cos(y)$

For the function $f(x, y) = \sin(x) \cdot \cos(y)$, the gradient is calculated similarly using the central difference method.

3.1 Edge Detection Using the Gradient

To detect edges in an image, we calculate the gradient of the image. The gradient at each point in the image highlights areas where there is a significant change in intensity, which typically corresponds to edges.

1. **Load the Image**: The image is first loaded and converted to grayscale if it is in color.
2. **Compute the Gradient**: We compute the gradient of the image in both the x- and y-directions. The gradient in the x-direction can be approximated using the forward difference formula:

$$\frac{\partial f}{\partial x}(x_m, y_n) = \frac{f(x_m + 1, y_n) - f(x_m, y_n)}{h_x}$$

Similarly, the gradient in the y-direction is computed as:

$$\frac{\partial f}{\partial y}(x_m, y_n) = \frac{f(x_m, y_n + 1) - f(x_m, y_n)}{h_y}$$

where h_x and h_y are the step sizes in the x- and y-directions, respectively.

3. **Compute the Gradient Magnitude**: The magnitude of the gradient is calculated using the following formula:

$$\|\nabla f(x_m, y_n)\| = \sqrt{\left(\frac{\partial f}{\partial x}(x_m, y_n)\right)^2 + \left(\frac{\partial f}{\partial y}(x_m, y_n)\right)^2}$$

This represents the strength of the edges at each point in the image.

4. **Edge Detection**: The gradient magnitude highlights areas where there are significant changes in pixel intensity, which correspond to edges. High gradient magnitudes typically correspond to strong edges in the image.

5. **Visualize the Result**: The resulting edge-detected image can be visualized by plotting the gradient magnitude.
