Fórmulas de derivación numérica

Table of Contents

Planteamiento

La derivación numérica es la computación de valores de la derivada de una función f(x) dados valores de f(x). Generalmente, la derivación numérica da valores que son mucho menos precisos que los valores de f(x). La imprecisión de la derivación numérica reside en la naturaleza de la definición de la derivada. Al ser calculada con una división de una cantidad grande entre una pequeña, puede causar errores e inestabilidad numérica. El objetivo de este proyecto es observar y analizar los errores que presentan diversas fórmulas de derivación numérica. Nos basaremos en las expansiones de la serie de Taylor de f(x) para obtener las derivadas aproximadas de diferencia finita de f(x).

Modelo matemático

La derivada de una función f en un punto x_0 se define como

$$f'(x_0) = \lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h}$$

Esta fórmula usa valores cerca de x_0. Por lo tanto, debemos escoger puntos espaciados equitativamente cerca del punto de derivación y crear un aproximación a partir de valores de f(x) en estos puntos para obtener una fórmula que aproxime la primera derivada. El error de discretización o truncamiento se encoge a medida que h se encoge. Si se quiere obtener una aproximación buena en un punto particular, entonces los errores de redondeo evitarán que tomemos valores de h excesivamente pequeños.

Se consideran tres tipos de aproximaciones: hacia adelante, hacia atrás y centradas.

Estas fórmulas que aproximaran las derivadas surgen de expansiones de f(x) en series de Taylor sobre el punto en x que queremos aproximar. La serie de Taylor de una función real infinitamente diferenciable evaluada en un punto a es la siguiente serie:

$$f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n)}(a)}{n!}(x-a)^n + \ldots$$

Por lo tanto, si queremos hacer una aproximación hacia atrás expandiendo la serie de Taylor en un punto x_0 con $x = x_0 - h$ tenemos

$$f(x_0 - h) = f(x_0) + \frac{f'(x_0)}{1!}((x_0 - h) - x_0)$$

$$\Rightarrow f'(x_0) = \frac{f(x_0) - f(x_0 - h)}{h}$$

De forma similar se obtienen las fórmulas de aproximación que usaremos a continuación.

Primera derivada

Usaremos las siguientes dos funciones en el punto x = 0.5 para probar las diferentes aproximaciones y observar los errores.

$f_1 = -0.1x^4 - 0.15x^3 - 0.5x^2 - 0.25x + 1.2$

$f_2 = x^6 - 1$

```
f1 = @(x) -(0.1.*(x.^4))-(0.15.*(x.^3))-(0.5.*(x.^2))-(0.25.*x)+1.2;
f2 = @(x) (x.^6)-1;
x = 0.5;
```

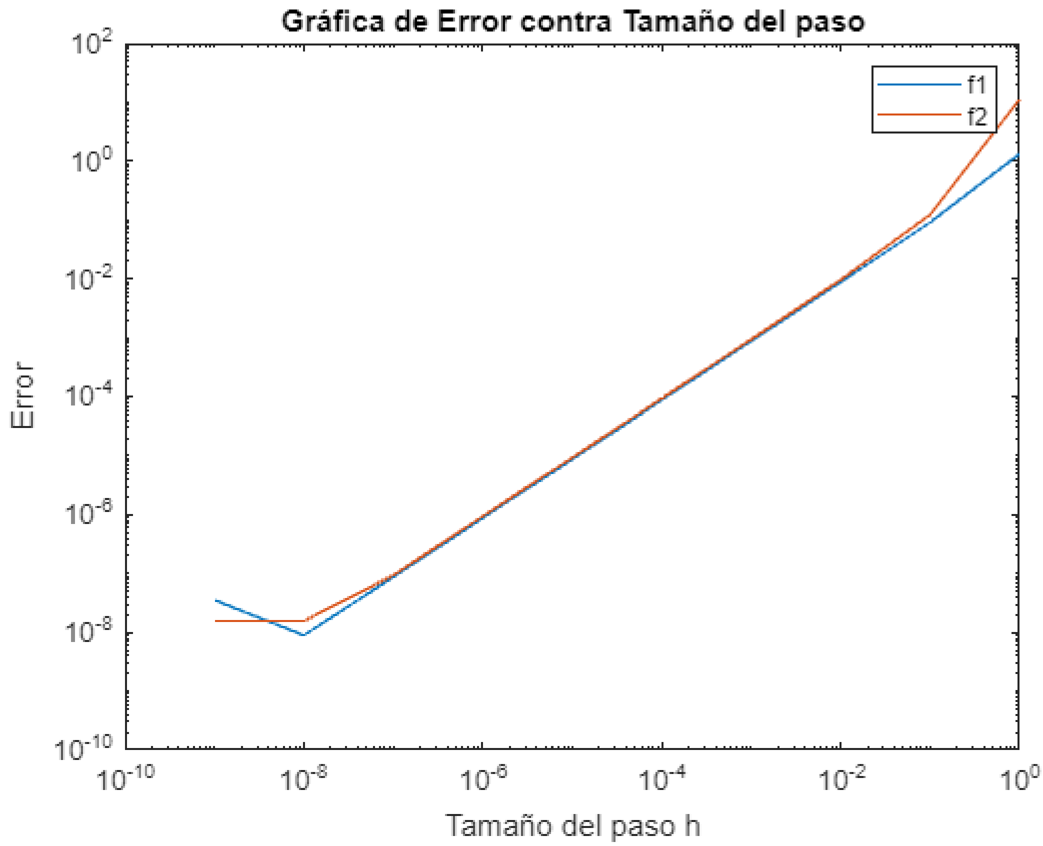
Aproximación hacia adelante

Una diferencia hacia adelante es una expresión de la forma (+) − () ya que se usa un punto por delante del valor de x que queremos aproximar. La aproximación correspondiente es

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

Esta es una aproximación de dos puntos.

```
[h, aprox, error] = firstForward(f1,x);
loglog(h,error, 'DisplayName', 'f1')
hold on
[h, aprox, error] = firstForward(f2,x);
loglog(h,error, 'DisplayName', 'f2');
hold off
xlabel('Tamaño del paso h');
ylabel('Error');
title('Gráfica de Error contra Tamaño del paso');
legend('show');
```




```
%-----------------hNuestra------------------
[valormin,indi] = min(error);
h_1AF = h(indi);
%-----------------hLibro--------------------
% n = orden del método
h_optima1AF = x*eps^(1+1);
```

Al ver la gráfica, podemos observar que la h que debemos usar para minimizar el error al usar una aproximación hacia adelante es un h de órden 10^{-8}.

Aproximación hacia atrás

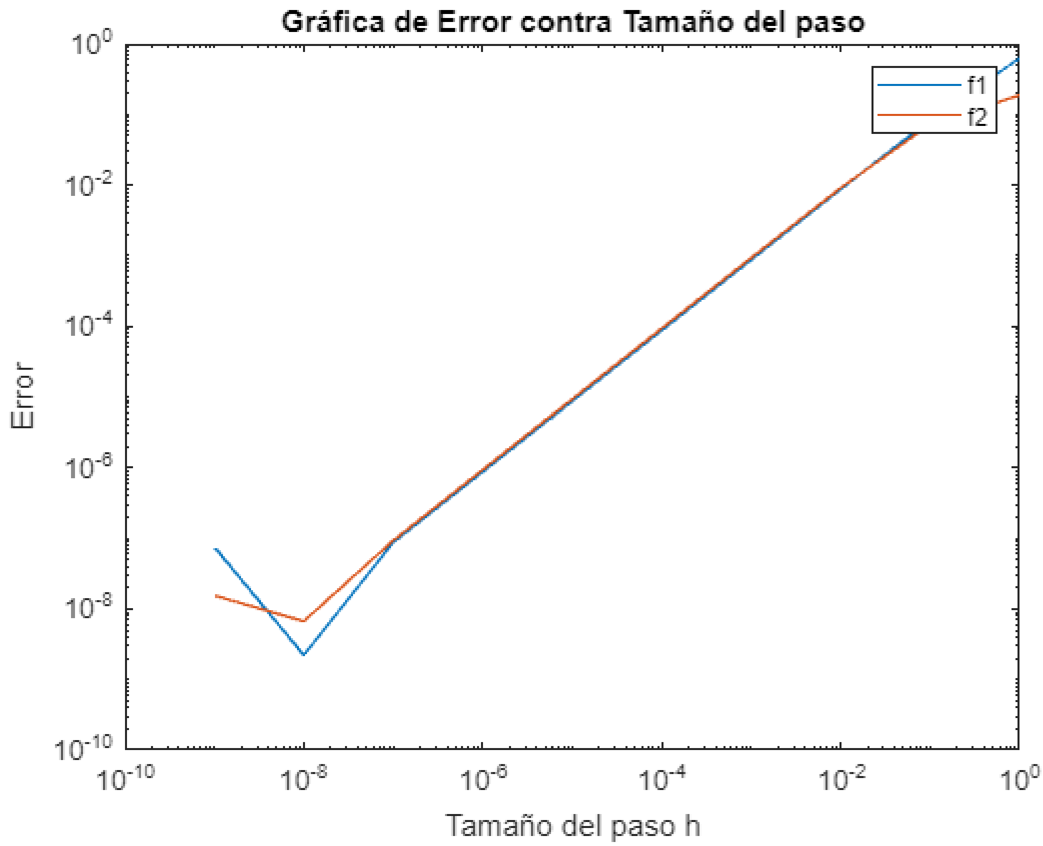
Una diferencia hacia atrás de la forma () − (-h) ya que se usa un punto por detrás del valor de x que queremos aproximar. La aproximación correspondiente es

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

Esta es una aproximación de dos puntos.

```
[h, aprox, error] = firstBackward(f1,x);
loglog(h,error, 'DisplayName', 'f1')
hold on;
[h, aprox, error] = firstBackward(f2,x);
loglog(h,error, 'DisplayName', 'f2');
hold off
```

```
xlabel('Tamaño del paso h');
ylabel('Error');
title('Gráfica de Error contra Tamaño del paso');
legend('show');
```




```
%----------------hNuestra-----------------
[valormin,indi] = min(error);
h_1AB = h(indi);
%----------------hLibro-------------------
% n = orden del método
h_optima1AB = x*eps^(1/(1+1));
```

Al ver la gráfica, podemos observar que la h que debemos usar para minimizar el error al usar una aproximación hacia atrás es un h de órden 10^{-8}.

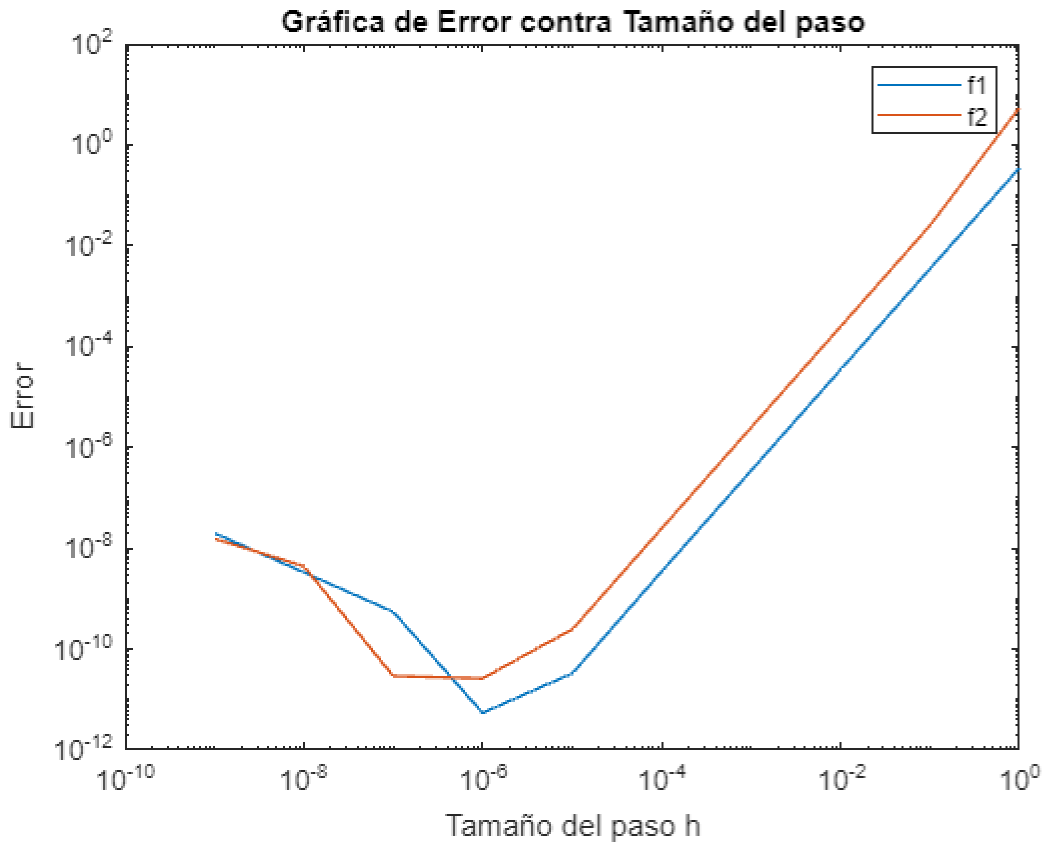
Aproximación centrada de tres puntos

Una diferencia centrada es una expresión de la forma (+ /2) − (− /2). Esta fórmula obtiene la primera derivada utilizando tres puntos cercanos para calcular una aproximación de la pendiente de la recta tangente, a la curva de una función, en cierto punto. Estos puntos están a una distancia de **h** hacia ambos lados del punto donde se quiere calcular la derivada. La aproximación correspondiente es

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

```
[h, aprox, error] = firstDerCenteredh2(f1,x);
```

```
loglog(h,error, 'DisplayName', 'f1')
hold on;
[h, aprox, error] = firstDerCenteredh2(f2,x);
loglog(h,error, 'DisplayName', 'f2');
hold off
xlabel('Tamaño del paso h');
ylabel('Error');
title('Gráfica de Error contra Tamaño del paso');
legend('show');
```




```
%-----------------hNuestra------------------
[valormin,indi] = min(error);
h_1AC = h(indi);
%----------------hLibro-------------------
% n = orden del método
h_optima1AC = x*eps^(1/(2+1));
```

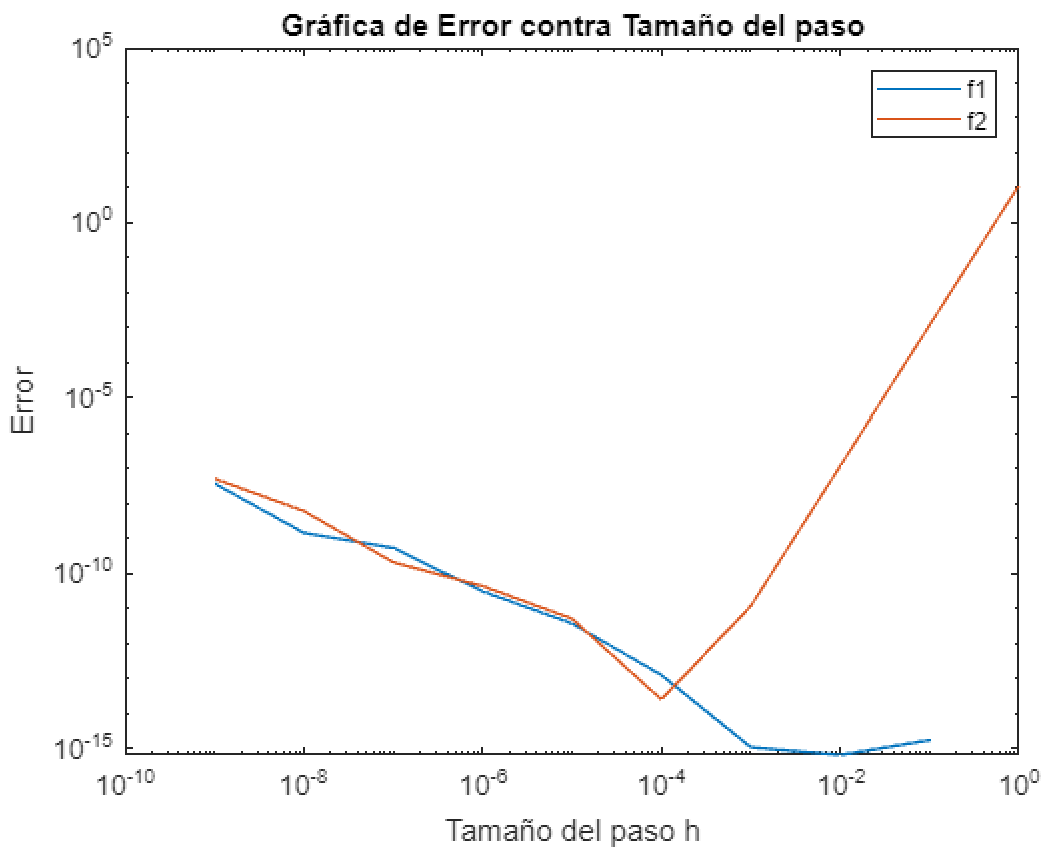
Al ver la gráfica, podemos observar que la h que debemos usar para minimizar el error al usar una aproximación centrada de tres puntos es un h de órden 10^{-6}.

Aproximación centrada de cinco puntos

Esta fórmula obtiene la primera derivada utilizando cinco puntos cercanos para calcular una aproximación de la pendiente de la recta tangente, a la curva de una función, en cierto punto. Estos puntos están a una distancia de **h** hacia ambos lados del punto donde se quiere calcular la derivada y los puntos x-h y x+h. La aproximación correspondiente es

$$f'(x) = \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h}$$

```
[h, aprox, error] = firstDerCenteredh4(f1,x);
loglog(h,error, 'DisplayName', 'f1')
hold on;
[h, aprox, error] = firstDerCenteredh4(f2,x);
loglog(h,error, 'DisplayName', 'f2');
hold off
xlabel('Tamaño del paso h');
ylabel('Error');
title('Gráfica de Error contra Tamaño del paso');
legend('show');
```




```
%-----------------hNuestra------------------
[valormin,indi] = min(error);
h_1AC5 = h(indi);
%-----------------hLibro--------------------
% n = orden del método
h_optima1AC5 = x*eps^(1/(4+1));
```

Al ver la gráfica, podemos observar que la h que debemos usar para minimizar el error al usar una aproximación centrada de tres puntos es un h de órden entre 10^{-2} y 10^{-4}.

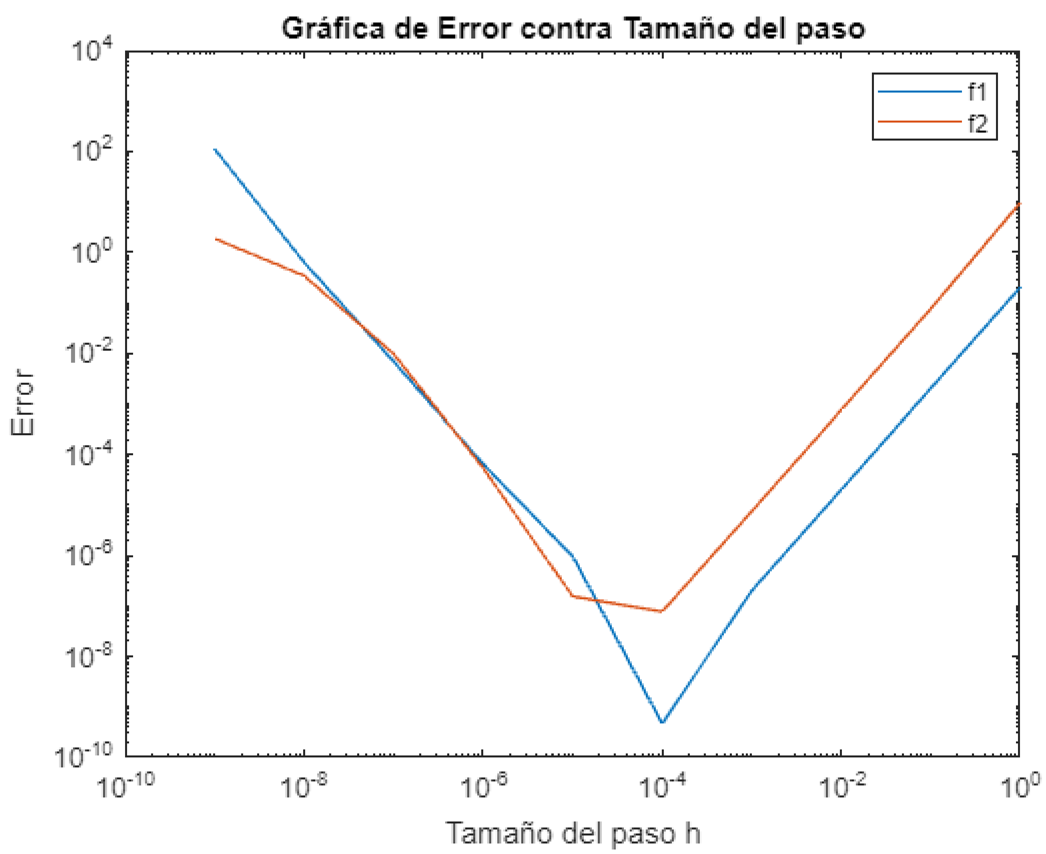
Segunda derivada

Aproximación centrada de tres puntos

Esta fórmula obtiene la segunda derivada utilizando tres puntos cercanos para calcular una estimación de la curvatura de la función en cierto punto. Estos puntos están a una distancia de **h** hacia ambos lados del punto donde se quiere calcular la derivada. La aproximación correspondiente es

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

```
[h, aprox, error] = seconDerCenteredh2(f1,x);
loglog(h,error, 'DisplayName', 'f1')
hold on;
[h, aprox, error] = seconDerCenteredh2(f2,x);
loglog(h,error, 'DisplayName', 'f2');
hold off
xlabel('Tamaño del paso h');
ylabel('Error');
title('Gráfica de Error contra Tamaño del paso');
legend('show');
```




```
%-----------------hNuestra------------------
```

```
[valormin,indi] = min(error);
h_2AC = h(indi);
```

```
h_2AC = 1.0000e-04
```

```
%-----------------hLibro--------------------
% n = orden del método
h_optima2AC = x*eps^(1/(2+1));
```

Al ver la gráfica, podemos observar que la h que debemos usar para minimizar el error al usar una aproximación centrada de tres puntos es un h de órden 10^{-4}.

Resultados

Libro --> $q* = x\sqrt[n+1]{\text{eps}}$

```
format long
H = [h_1AF h_optima1AF;h_1AB h_optima1AB;h_1AC h_optima1AC;h_1AC5
h_optima1AC5;h_2AC h_optima2AC]
```

```
H = 5×2
10^-3 ×
    0.0000    0.0000
    0.0000    0.0000
    0.0010    0.0030
    0.1000    0.3700
    0.1000    0.0030
```

```
% Primera columna "nuestras" y segunda columna "libro"
```

Conclusiones

La virtud del uso de la expansión de series de Taylor para aproximar la derivada es su simplicidad y facilidad de uso. Su falla es la falta de generalidad y el hecho de que se usa a través de instancias concretas de la función.

¿Qué fórmulas presentan menos errores?

Dada la naturaleza de las fórmulas, todas presentan un error en la aproximación. Al usar las aproximaciones que usan más puntos, se llega a una aproximación muy cercana a la derivada real de forma más rápida. Por lo tanto, no necesitamos un h tan pequeña para obtener un resultado deseable y minimizar el error obtenido.

¿Cómo dependen de *h*?

El valor de h define que tan equitativamente espaciados deben estar los valores de x al ser evaluados por f(x). Dado que en la teoría, la derivada define el cambio inmediato en puntos de f(x), necesitamos una h pequeña en la práctica para aproximar ese cambio inmediato.

¿El error siempre disminuye si *h* se hace más pequeño? ¿Qué otro factor influye en el error?

Hasta cierto punto, el tamaño del error disminuye si h decrementa, como se puede observar en las gráficas; sin embargo, llegado a ese punto, el error aumenta debido al error de truncamiento y de redondeo. El error de truncamiento se refiere al error causado por la falta de términos de una serie infinita. En el caso de las fórmulas utilizadas, esto ocurre en la serie de Taylor. Por otro lado, el error de redondeo se refiere al error provocado por el dispositivo donde se realiza el cálculo. Las computadoras tienen un límite definido de dígitos después del punto decimal, por lo que es necesario redondear. En conclusión, la fórmula más precisa para calcular una derivada, en general, será la fórmula centrada, pero dependerá de los puntos mencionados anteriormente: el tamaño de h, los puntos que se utilice para calcular la derivada y el dispositivo que realice dicho cálculo.

Implementación

```
function [H, aprox, error] = firstForward(f, x)
    fsym = sym(f);
    dfs = diff(fsym);
    df = matlabFunction(dfs);
    h=1;
    H(1) = h;
    aprox(1) = (f(x+h)-f(x))/h;
    error(1) = abs(aprox(1)-df(x));
    for i=2:10
        h=h/10;
        H(i) = h;
        aprox(i) = (f(x+h)-f(x))/h;
        error(i) = abs(aprox(i)-df(x));
    end
end

function [H, aprox, error] = firstBackward(f, x)
    fsym = sym(f);
    dfs = diff(fsym);
    df = matlabFunction(dfs);
    h=1;
    H(1) = h;
    aprox(1) = (f(x)-f(x-h))/h;
    error(1) = abs(aprox(1)-df(x));
    for i=2:10
        h=h/10;
        H(i) = h;
        aprox(i) = (f(x)-f(x-h))/h;
        error(i) = abs(aprox(i)-df(x));
    end
end

function [H, aprox, error] = firstDerCenteredh2(f,x)
    fsym = sym(f);
```

```
    dfs = diff(fsym);
    df = matlabFunction(dfs);
    h=1;
    H(1) = h;
    aprox(1) = (f(x+h)-f(x-h))/(2*h);
    error(1) = abs(aprox(1)-df(x));
    for i=2:10
        h=h/10;
        H(i) = h;
        aprox(i) = (f(x+h)-f(x-h))/(2*h);
        error(i) = abs(aprox(i)-df(x));
    end
end

function [H, aprox, error] = firstDerCenteredh4(f,x)
    fsym = sym(f);
    dfs = diff(fsym);
    df = matlabFunction(dfs);
    h=1;
    H(1) = h;
    aprox(1) = (-f(x+(2*h))+(8*f(x+h))-(8*f(x-h))+f(x-(2*h)))/(12*h);
    error(1) = abs(aprox(1)-df(x));
    for i=2:10
        h=h/10;
        H(i) = h;
        aprox(i) = (-f(x+(2*h))+(8*f(x+h))-(8*f(x-h))+f(x-(2*h)))/(12*h);
        error(i) = abs(aprox(i)-df(x));
    end
end

function [H, aprox, error] = seconDerCenteredh2(f,x)
    fsym = sym(f);
    dfs = diff(fsym);
    dffs = diff(dfs);
    dff = matlabFunction(dffs);
    h=1;
    H(1) = h;
    aprox(1) = (f(x+h)-(2*f(x))+f(x-h))/(h^2);
    error(1) = abs(aprox(1)-dff(x));
    for i=2:10
        h=h/10;
        H(i) = h;
        aprox(i) = (f(x+h)-(2*f(x))+f(x-h))/(h^2);
        error(i) = abs(aprox(i)-dff(x));
    end
end
```

Referencias

- Ascher, U. M., & Greif, C. (2011). *A first course in numerical methods*. The Universityof British Columbia.
- Chapra, S. C. (2018). *Applied Numerical methods with MATLAB for engineers and scientists* (Fourth Edition). Mc Graw Hill.