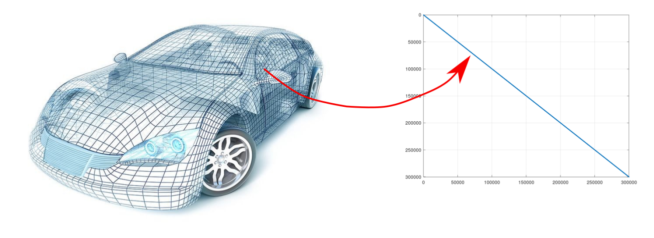
Universidad Politécnica de Cartagena

industriales etsii UPCT

Interpolación



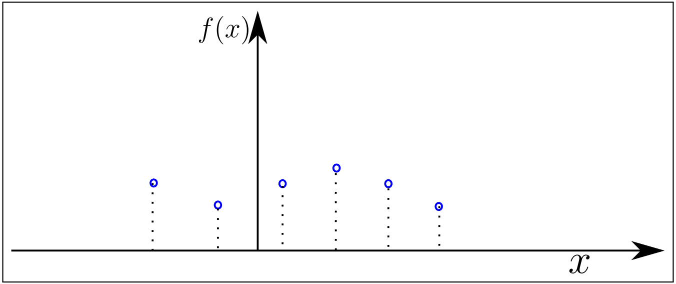
Rogelio Ortigosa Martínez
Silvestre Paredes Hernández
Departamento de Matemática Aplicada y Estadística, UPCT

Índice

Índice

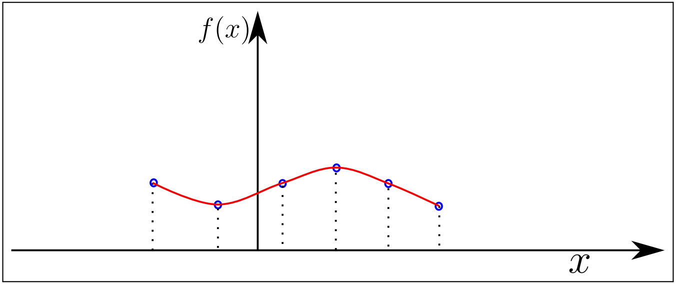
Bloque 1 (I)

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \dots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \dots, y_N\}$



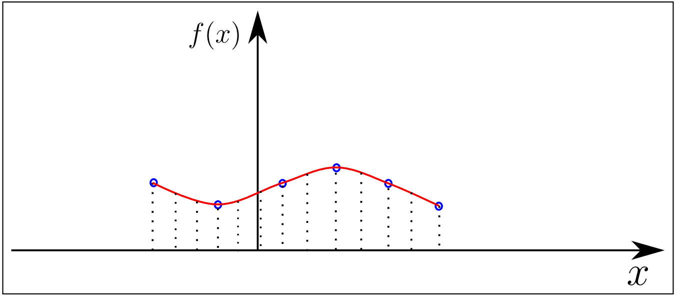
Bloque 1 (II)

- **Encontrar** polinomio de grado $N-1$ que pasa por $\boldsymbol{x} = \{x_1, x_2, \dots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \dots, y_N\}$



Bloque 1 (III)

- **Aproximar** $f(x)$ en otros puntos $\boldsymbol{x}^\star = \{x_1^\star, x_2^\star, \ldots, x_M^\star\}$ e $\boldsymbol{y}^\star = \{y_1^\star, y_2^\star, \ldots, y_M^\star\}$



Bloque 1 (IV)

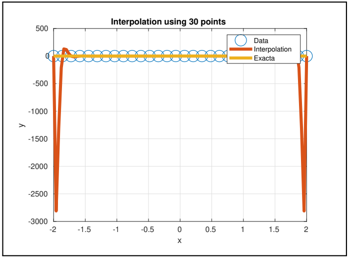
- Objetivos:

1. Generar **30 datos** entre $[-2, 2]$ **equi-espaciados** para x
2. **Generar los valores** y evaluando la función $f(x) = \frac{1}{1+16x^2}$
3. Calcular **polinomio de interpolación** $p(x)$ **de grado** 29
4. **Evaluar el** $p(x)$ en 100 puntos en $[-2, 2]$ distintos de los datos

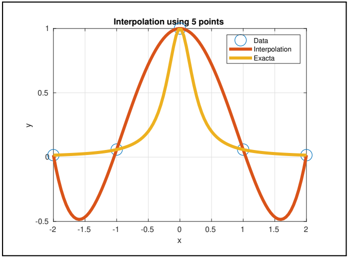
```
%-----------------------------------------------------------------------
% Generamos los datos
%-----------------------------------------------------------------------
xLeft  =  -2; % Límite izquierdo del intervalo
xRight =  2;  % Límite derecho del intervalo
NData  =  30; % Número de datos a generar
xData  =  (linspace(xLeft,xRight,NData))';  % Valor de los datos x
yData  =  zeros(NData,1);  % Valor de los datos y, generados a partir de la
                           % función y(x)=1/(1+16*x^2). Los podemos generar
                           % con loop o trabajando componente a componente
for i=1:NData
 yData(i)  =  1/(1 + 16*xData(i)^2);
end
%yData   = 1./(1+16*xData.^2);  % Opción trabajando componente a componente
%-----------------------------------------------------------------------
% Ploteamos los datos
%-----------------------------------------------------------------------
figure(1)
plot(xData,yData,'o','MarkerSize',15)
hold on
grid on
%-----------------------------------------------------------------------
% Obtenemos polinomio de interpolación de grado NData-1 (29)
% a través del comando Polyfit (utiliza matriz de Vandermonde)
%-----------------------------------------------------------------------
Coeff   = polyfit(xData,yData,NData-1);
%-----------------------------------------------------------------------
% Generamos 100 puntos de x diferentes de xData donde interpolar y con el
% polinomio de interpolación generado antes
%-----------------------------------------------------------------------
NInter  = 100;
xInter  = (linspace(xLeft,xRight,NInter))';
%-----------------------------------------------------------------------
% Interpolamos el polinomio de interpolación en esos 100 coordenadas x
%-----------------------------------------------------------------------
yInter  =  polyval(Coeff,xInter);
%-----------------------------------------------------------------------
% Ploteamos el resultado de la interpolación en esas 100 coordenadas x
%-----------------------------------------------------------------------
plot(xInter,yInter,'LineWidth',4)
%-----------------------------------------------------------------------
% Ploteamos la solución exacta en las 100 coordenadas x
%-----------------------------------------------------------------------
yExact  =  1./(1+16*xInter.^2); % siolución exacta en los 100 puntos
plot(xInter,yExact,'LineWidth',4)
legend('Data','Interpolation','Exacta')
```

Bloque 1 (V)

- Interpolación usando 30 datos:



- Interpolación usando 5 datos:

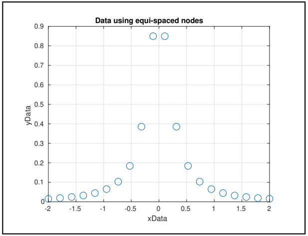


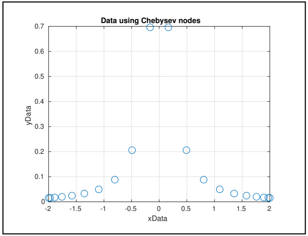
- Ilustración del **fenómeno de Runge** cuando:
  1. Utilizamos nodos distintos a los de Chebyshev (en este caso equi-espaciados)
  2. Utilizamos polinomio de interpolación de grado alto

Índice

Bloque 2 (I)

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$
- Queremos usar mismo número de datos. Si tenemos la opción de generar datos en nuevos valores de $\boldsymbol{x}$, usamos **nodos de Chebyshev**:
  1. Generamos nodos de Chebyshev en $[-1, 1]$ como sigue:
  $$\hat{x} = \{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_N\}; \qquad \hat{x}_i = \cos\left(\pi\left(\frac{i-1}{N-1}\right)\right); \qquad i = \{1, 2, \ldots, N\}$$
  2. Convertimos esos nodos al intervalo $[x_1, x_N]$, generando $\boldsymbol{x} = [x_1, x_2, \ldots, x_N]$
  $$x_i = \frac{x_N - x_1}{2}\hat{x}_i + \frac{x_N + x_1}{2}$$
- Generamos datos $\boldsymbol{y} = \{y_1, \ldots, y_N\}$ asociados con las coordenadas de $\boldsymbol{x}$ creadas





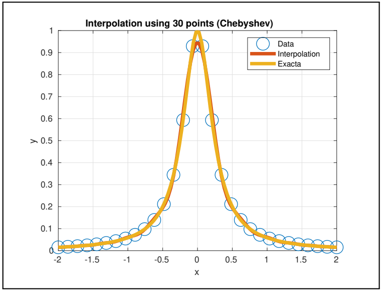
Bloque 2 (II)

- Implementación:

```
%--------------------------------------------------------------------------
% Generamos los nodos de Chebyshev
%--------------------------------------------------------------------------
xC  =  zeros(NData,1);
for i=1:NData
   xC(i)    =  cos(pi*(i-1)/(NData-1)) ; % nodos de Chebyshev en [-1,1]
end
xC         =  flip(xC);

xDataCheb     =  (xRight-xLeft)/2*xC + (xRight+xLeft)/2; % nodos en [-2,2]
yDataCheb     =  zeros(NData,1);
for i=1:NData
 yDataCheb(i)  =  1/(1 + 16*xDataCheb(i)^2);
end
%yData   = 1./(1+16*xData.^2);  % Opción trabajando componente a componente
%--------------------------------------------------------------------------
% Ploteamos los datos
%--------------------------------------------------------------------------
figure(2)
plot(xData,yData,'o','MarkerSize',15)
hold on
grid on
%--------------------------------------------------------------------------
% Obtenemos polinomio de interpolación de grado NData-1 (29)
% a través del comando Polyfit
%--------------------------------------------------------------------------
CoeffCheb   = polyfit(xDataCheb,yDataCheb,NData-1);
%--------------------------------------------------------------------------
% Generamos 100 puntos de x diferentes de xData donde interpolar y con el
% polinomio de interpolación generado antes
%--------------------------------------------------------------------------
NInter  = 100;
xInter  = (linspace(xLeft,xRight,NInter))';
%--------------------------------------------------------------------------
% Interpolamos el polinomio de interpolación en esos 100 coordenadas x
%--------------------------------------------------------------------------
yInterCheb  =  polyval(CoeffCheb,xInter);
%--------------------------------------------------------------------------
% Ploteamos el resultado de la interpolación en esas 100 coordenadas x
%--------------------------------------------------------------------------
plot(xInter,yInterCheb,'LineWidth',4)
%--------------------------------------------------------------------------
% Ploteamos la solución exacta en las 100 coordenadas x
%--------------------------------------------------------------------------
yExact  =  1./(1+16*xInter.^2); % solución exacta en los 100 puntos
plot(xInter,yExact,'LineWidth',4)
legend('Data','Interpolation','Exacta')
xlabel('x')
ylabel('y')
title('Interpolation using 30 points (Chebyshev)')
```

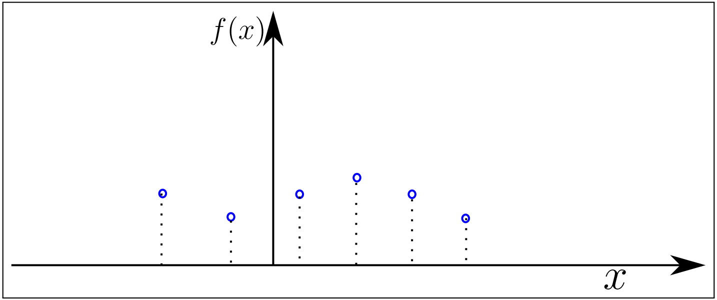
- Interpolación libre de **fenómeno de Runge**:



Índice

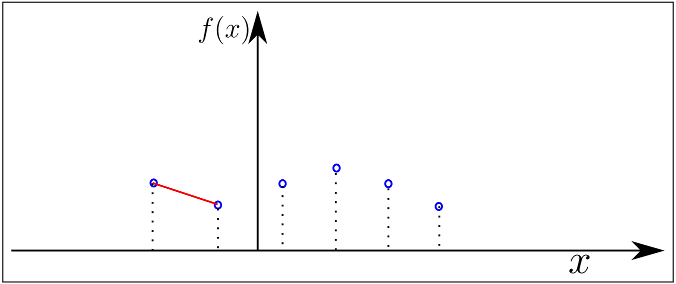
Bloque 3 (I): subdivisión e interpolación con polinomios lineales

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$. **Podemos usar nodos equi-espaciados** (no son necesarios nodos de Chebishev!)



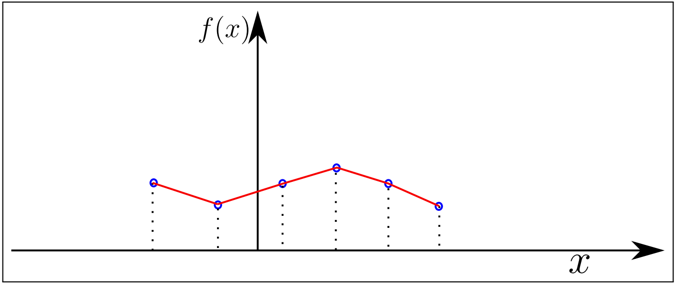
Bloque 3 (II): subdivisión e interpolación con polinomios lineales

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$



Bloque 3 (III): subdivisión e interpolación con polinomios lineales

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \dots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \dots, y_N\}$



- Hemos creado una función (polinomio de interpolación) en el intervalo $[x_1, x_N]$ de **clase** $\mathcal{C}^0([x_1, x_N])$, por tanto con **derivadas discontinuas**

Bloque 3 (IV): subdivisión e interpolación con splines

- **Datos:** partimos de $\boldsymbol{x} = \{x_1, x_2, \ldots, x_N\}$ e $\boldsymbol{y} = \{y_1, y_2, \ldots, y_N\}$
- Interpolación con **splines cúbicos naturales:**

1. Imponemos en cada sub-intervalo $[x_i, x_i + 1]$

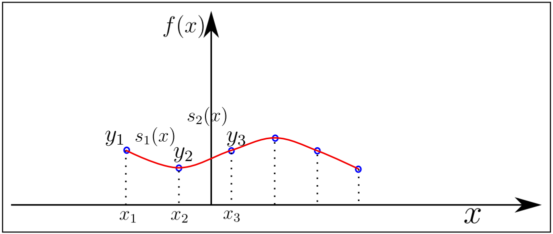
$$y_i = s_i(x_i); \qquad y_{i+1} = s_i(x_{i+1})$$

2. Imponemos en cada nodo intermedio x_i continuidad de primeras y segundas derivadas (función de **clase** $\mathcal{C}^2([x_1, x_N])$**:**)

$$s_i'(x_{i+1}) = s_{i+1}'(x_{i+1}); \qquad s_i''(x_{i+1}) = s_{i+1}''(x_{i+1})$$

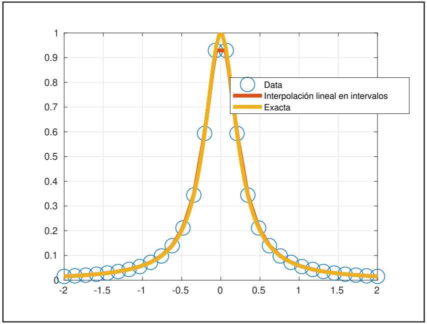
3. El sistema de ecuaciones generado se cierra imponiendo

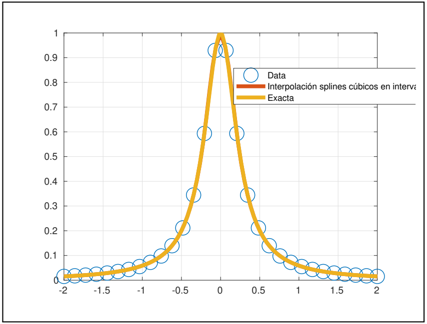
$$s_1''(x_1) = s_N''(x_N) = 0$$



Bloque 3 (V)

- **Interpolación lineal** en cada intervalo $[x_i, x_{i+1}]$
- **Interpolación splines cúbicos** en $[x_i, x_{i+1}]$





- Interpolaciones de grado bajo sin **fenómeno de Runge**