Computación Numérica

Tema 3. Interpolación (II): splines y ajuste trigonométrico

Irene Parada

irene.parada@upc.edu

Departamento de Matemáticas

Universitat Politècnica de Catalunya · BarcelonaTech

22 de abril de 2024

Repaso

Breve recordatorio del Tema 3.1

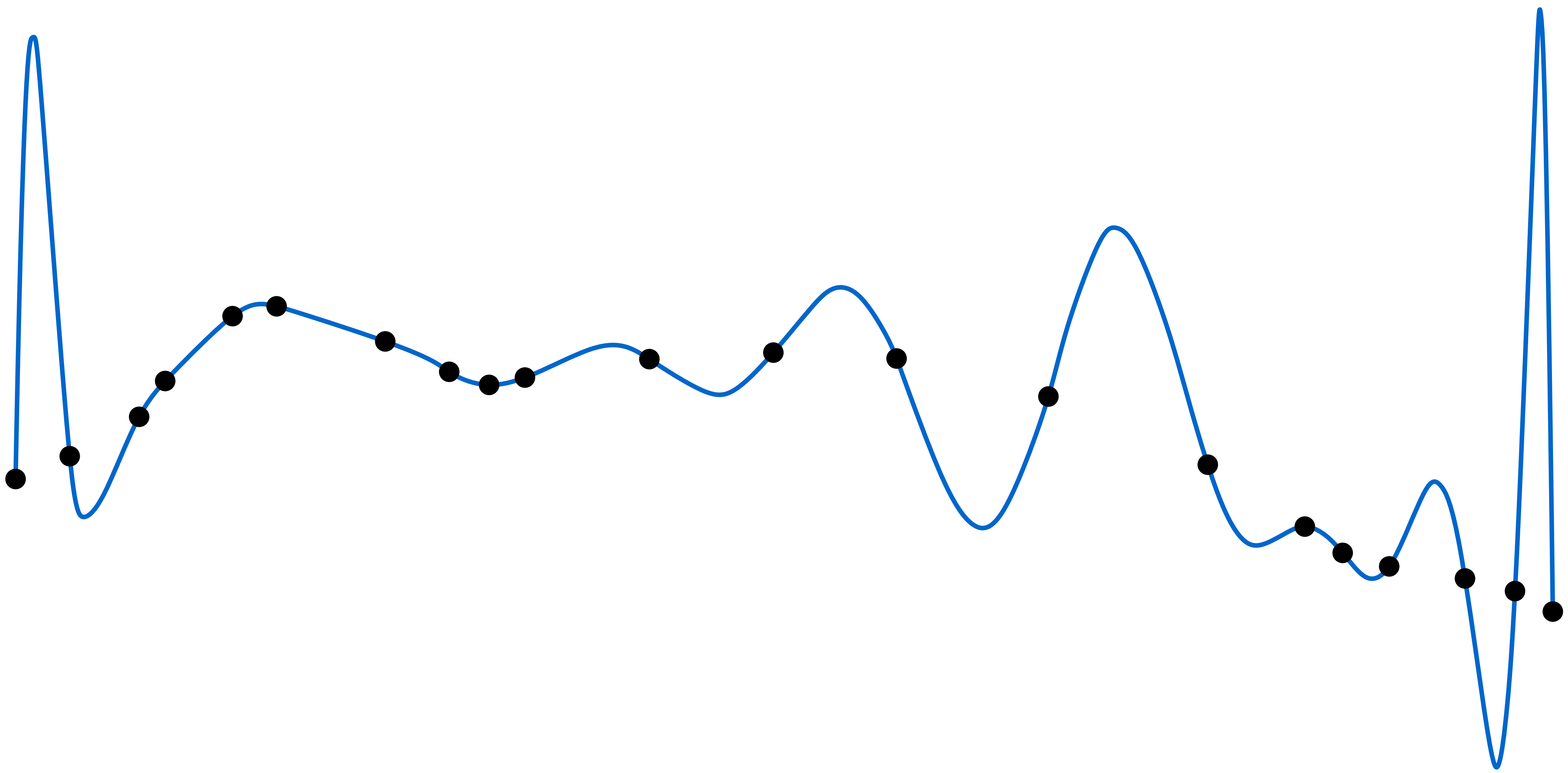
- Interpolación, extrapolación y tipos de interpolación.
- Sistema de Vandermonde. Existencia y unicidad del polinomio interpolador.
- Error en la interpolación polinómica.
- Interpolación polinómica por la fórmula de Lagrange.
- Interpolación polinómica por el métodos de las diferencias divididas de Newton.
- Fenómeno de Runge.
- Abscisas de Chevyshev.
- Interpolación de Hermite y error.
- Curvas paramétricas de interpolación.

En el tema anterior...

Problemas de usar un polinomio interpolador de grado n:

- Computacionalmente costoso.
- Fenómeno de Runge.
- El grado alto del polinomio produce oscilaciones.
- Puede estar mal condicionado: pequeñas variaciones en los puntos pueden producir grandes variaciones en el polinomio interpolador.
- Los cambios no son locales: variaciones en un punto pueden producir variaciones en puntos lejanos.
- Es difícil mejorar la curva interactivamente: lo único que se puede hacer es añadir puntos, lo cual no siempre mejora el error, o moverlos, lo cual puede dar lugar a mayores errores en puntos lejanos.

Polinomio interpolador



Hoy

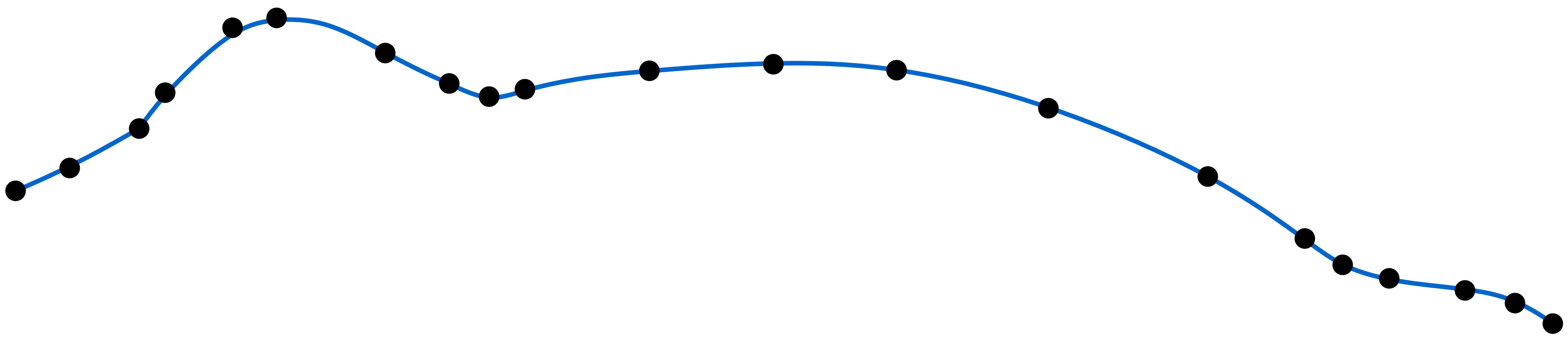
- Splines (MATLAB: `interp1()`)
  - lineales (interpolación segmentaria lineal)
  - cúbicas (interpolación segmentaria cúbica)
  - de Hermite (son cúbicas también)
- Curvas de Bézier
- B-splines

Hoy

- Splines (MATLAB: `interp1()`)
  - lineales (interpolación segmentaria lineal)
  - cúbicas (interpolación segmentaria cúbica)
  - de Hermite (son cúbicas también)
- Curvas de Bézier
- B-splines

- Ajuste trigonométrico y transformada de Fourier.

Spline cúbica



Splines

Interpolación polinómica a trozos

Interpolación polinómica a trozos: splines

Dados (x_0, y_0), (x_1, y_1), ..., (x_n, y_n), la idea es interpolar cada subintervalo $[x_i, x_{i+1}]$ formado por nodos consecutivos por un polinomio de grado bajo.

Interpolación polinómica a trozos: splines

Dados (x_0, y_0), (x_1, y_1), ..., (x_n, y_n), la idea es interpolar cada subintervalo $[x_i, x_{i+1}]$ formado por nodos consecutivos por un polinomio de grado bajo.

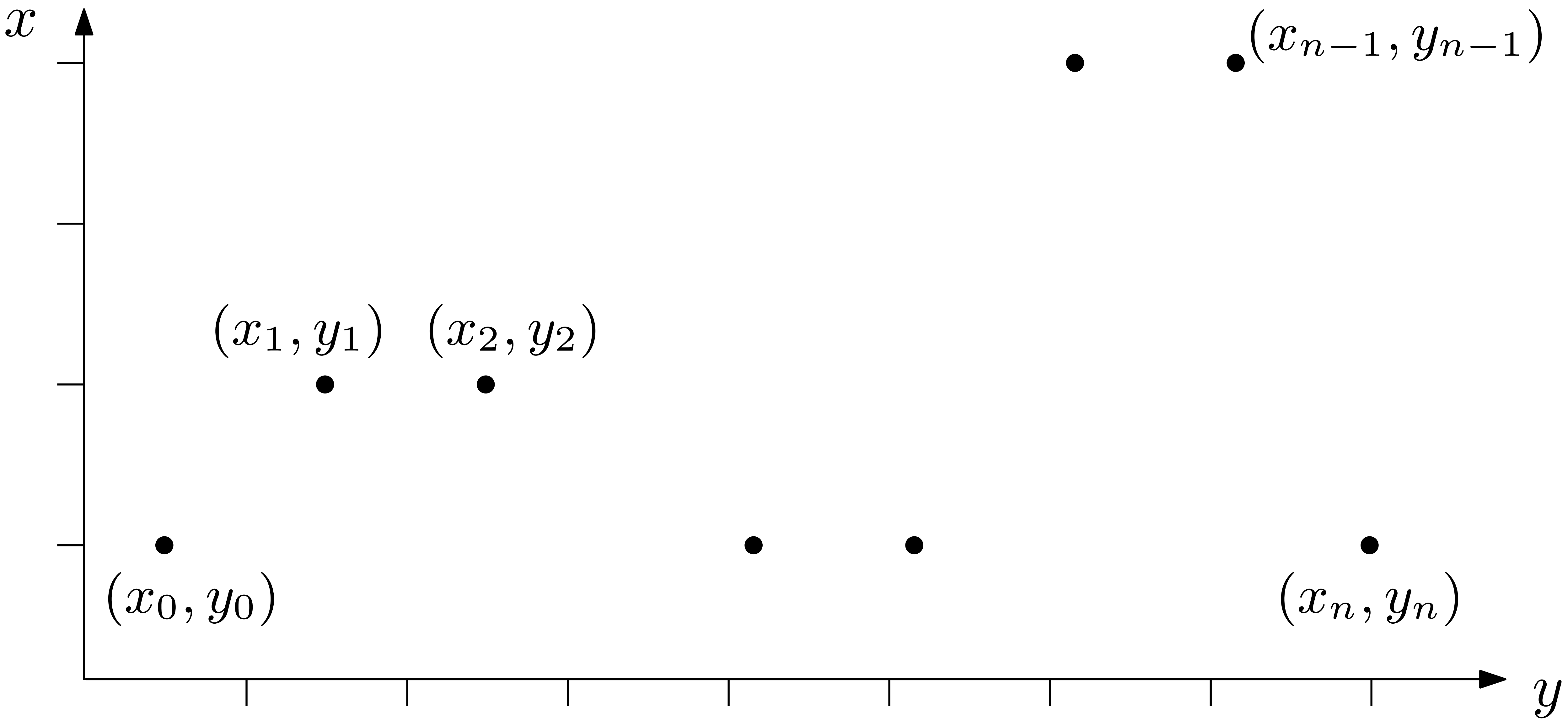
- especialmente útil cuando hay muchos puntos.

Interpolación polinómica a trozos: splines

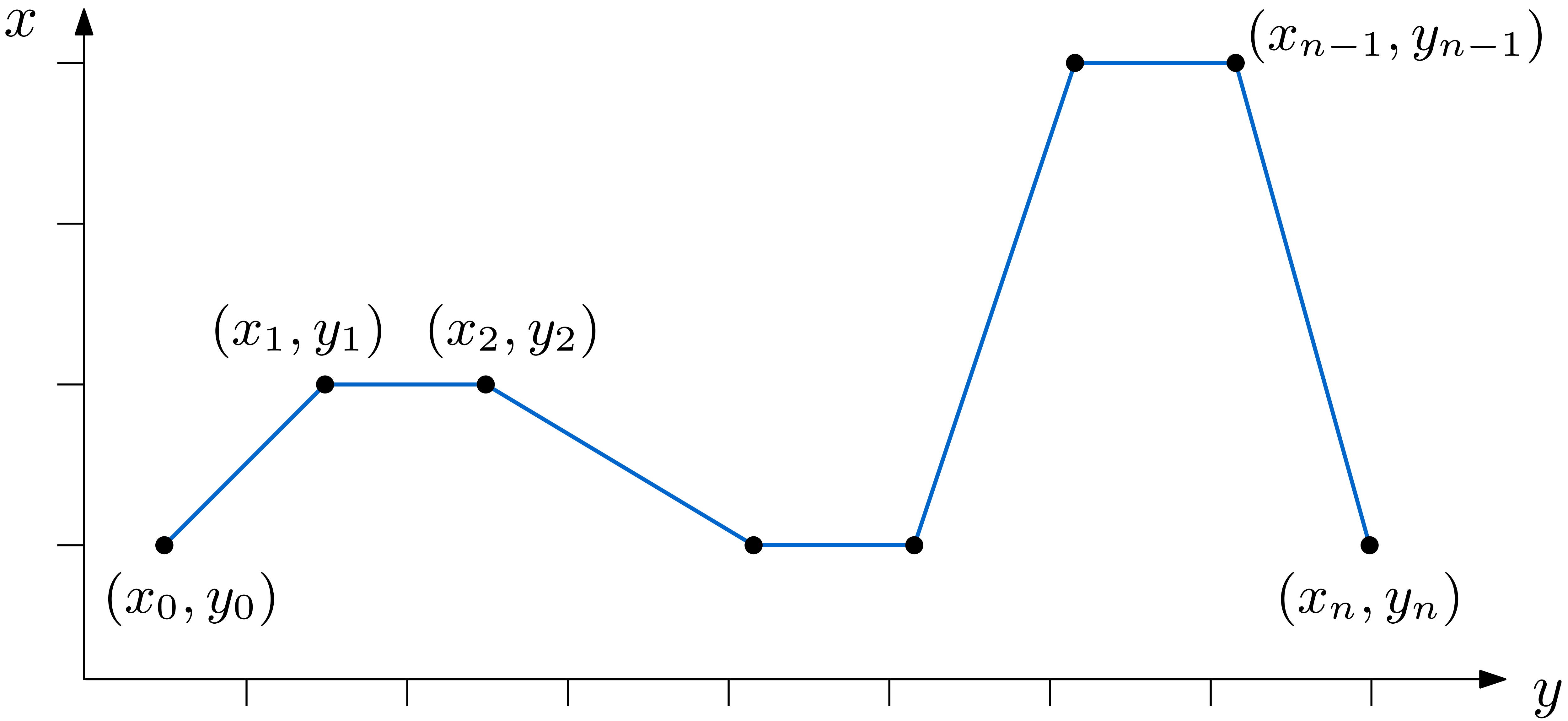
Dados (x_0, y_0), (x_1, y_1), …, (x_n, y_n), la idea es interpolar cada subintervalo $[x_i, x_{i+1}]$ formado por nodos consecutivos por un polinomio de grado bajo.

- especialmente útil cuando hay muchos puntos.
- Una spline es una curva definida a trozos por polinomios de grado k, con continuidad hasta la derivada $\ell \leq k - 1$.

Spline lineal

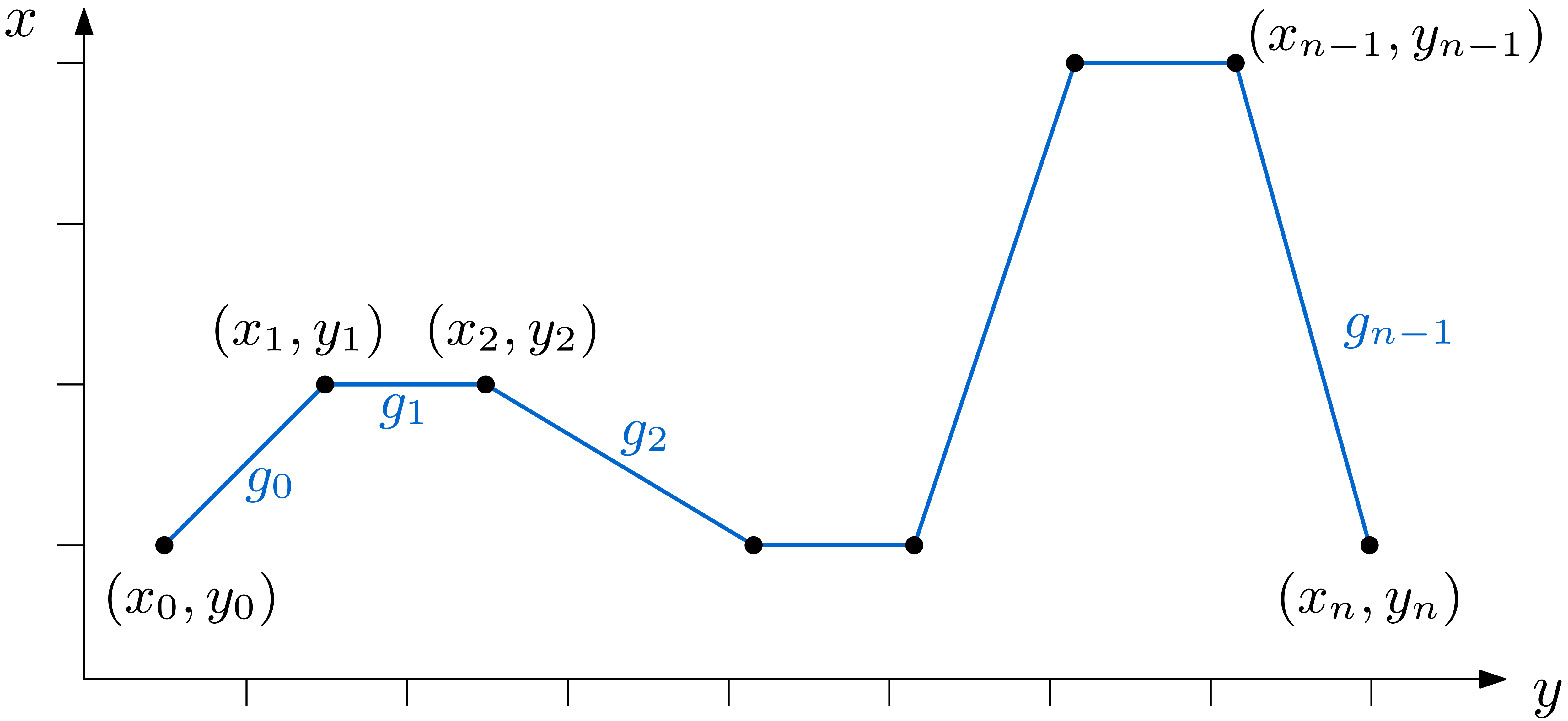


Spline lineal



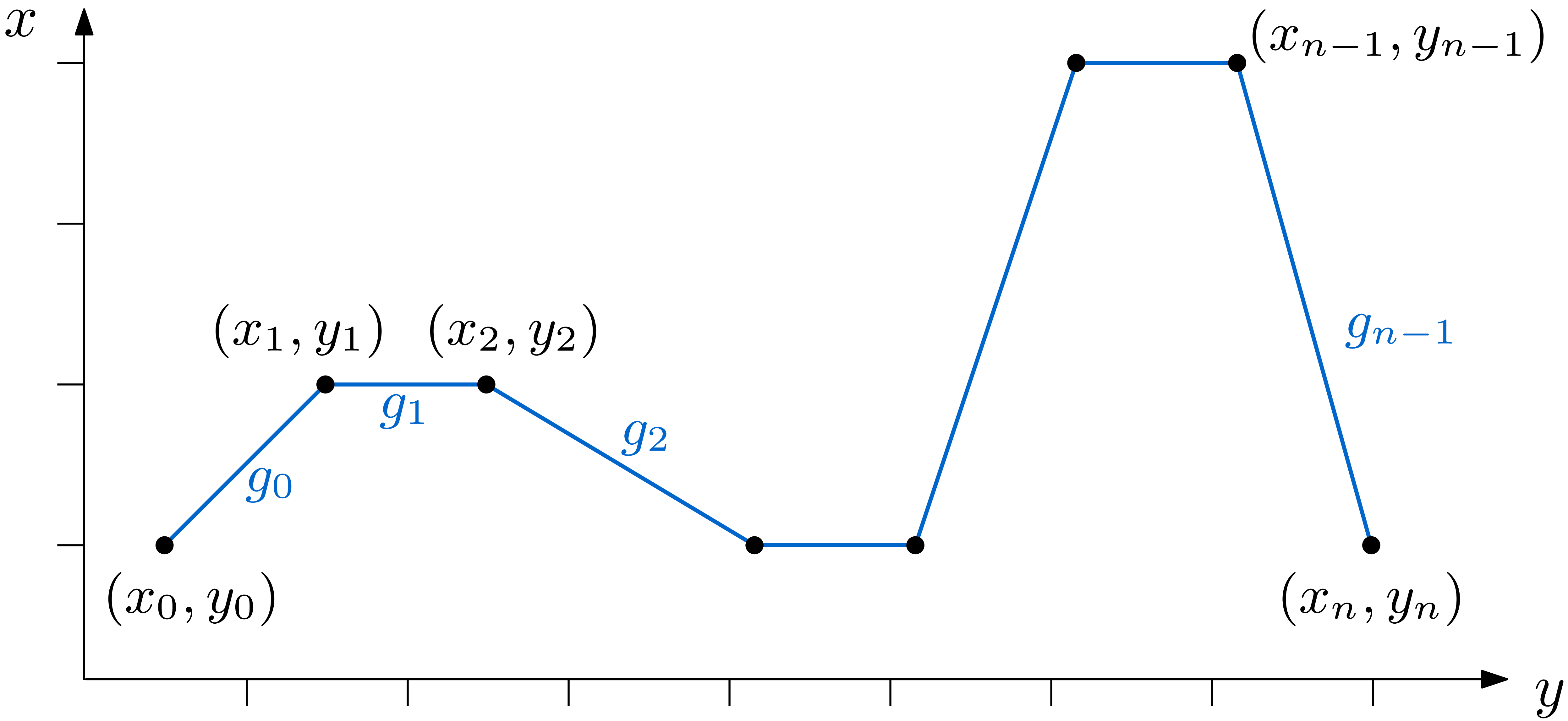
Una spline lineal es el caso más simple, los puntos a interpolar se conectan por segmentos de recta. La curva es continua y lineal a trozos.

Spline lineal



Una spline lineal es el caso más simple, los puntos a interpolar se conectan por segmentos de recta. La curva es continua y lineal a trozos.

Spline lineal

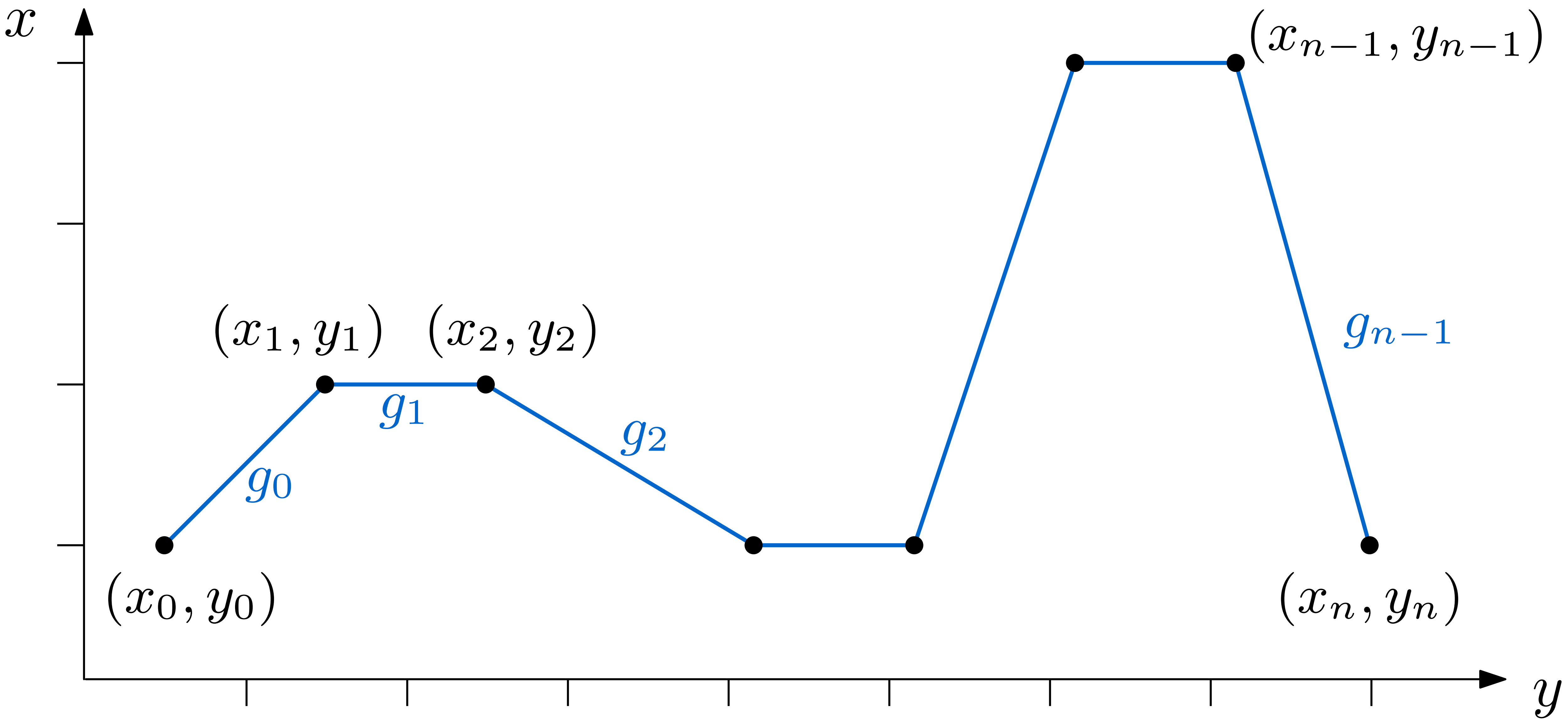


La función de interpolación es $y = g(x)$ con

$g(x) = g_i(x)$,
$x_i \leq x \leq x_{i+1}$,
$0 \leq i \leq n-1$.

Una spline lineal es el caso más simple, los puntos a interpolar se conectan por segmentos de recta. La curva es continua y lineal a trozos.

Spline lineal



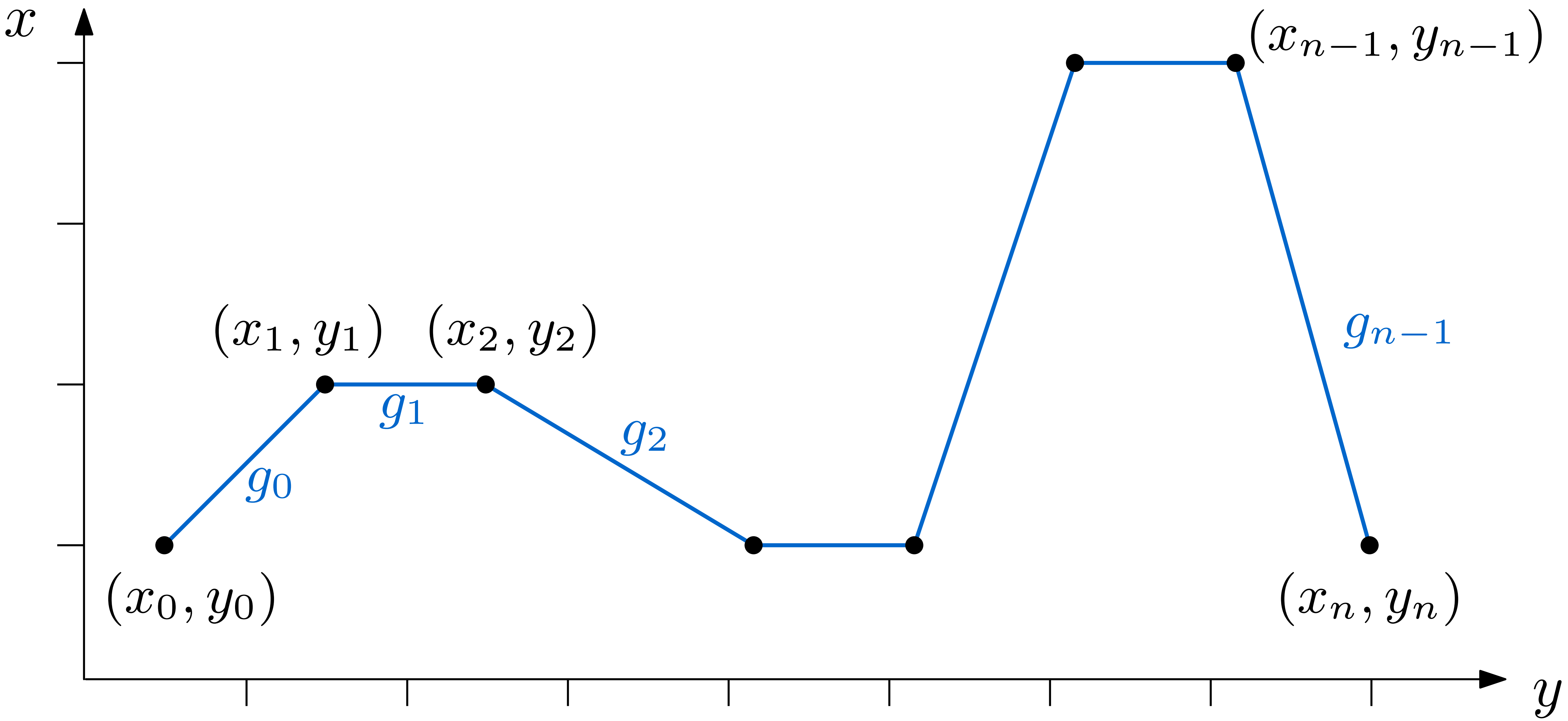
La función de interpolación es $y = g(x)$ con

$g(x) = g_i(x)$,
$x_i \leq x \leq x_{i+1}$,
$0 \leq i \leq n - 1$.

Las g_i son polinomios de grado 1.

Una spline lineal es el caso más simple, los puntos a interpolar se conectan por segmentos de recta. La curva es continua y lineal a trozos.

Spline lineal: cálculo

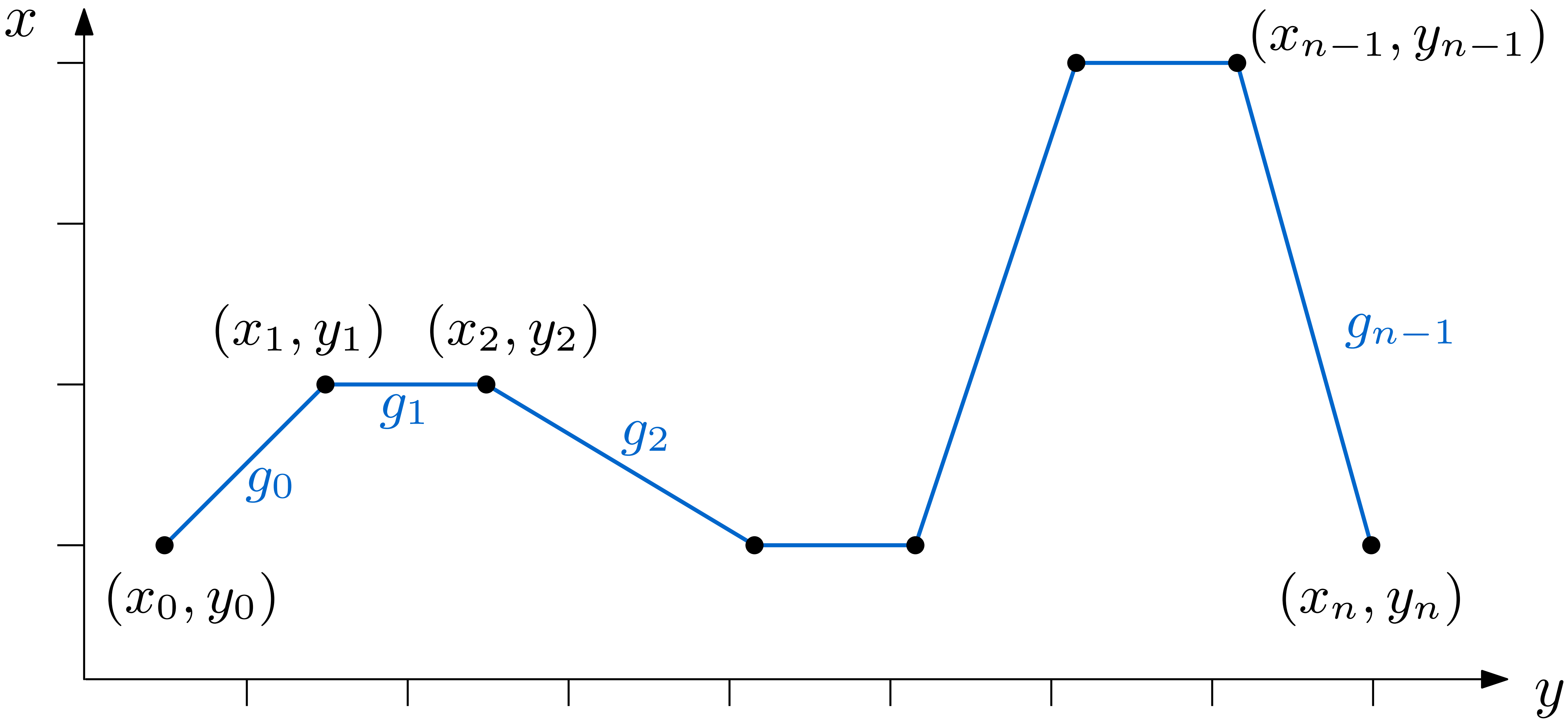


La función de interpolación es $y = g(x)$ con

$g(x) = g_i(x)$,
$x_i \leq x \leq x_{i+1}$,
$0 \leq i \leq n-1$.

Las g_i son polinomios de grado 1.

Spline lineal: cálculo



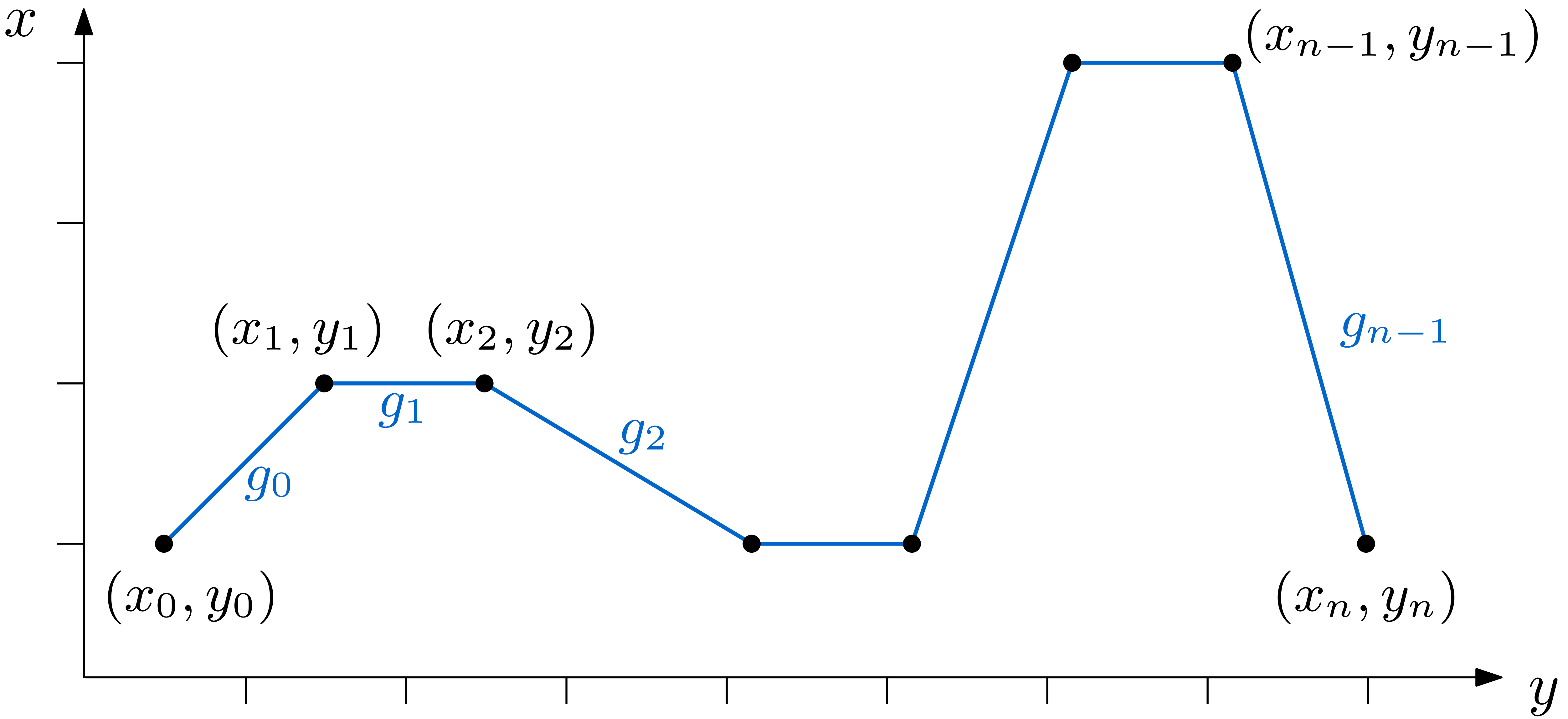
La función de interpolación es $y = g(x)$ con

$g(x) = g_i(x)$, $x_i \leq x \leq x_{i+1}$, $0 \leq i \leq n-1$.

Las g_i son polinomios de grado 1.

$$\boxed{\begin{array}{l} g_i(x) = \\ a_i(x - x_i) + b_i \end{array}}$$

Spline lineal: cálculo



La función de interpolación es $y = g(x)$ con

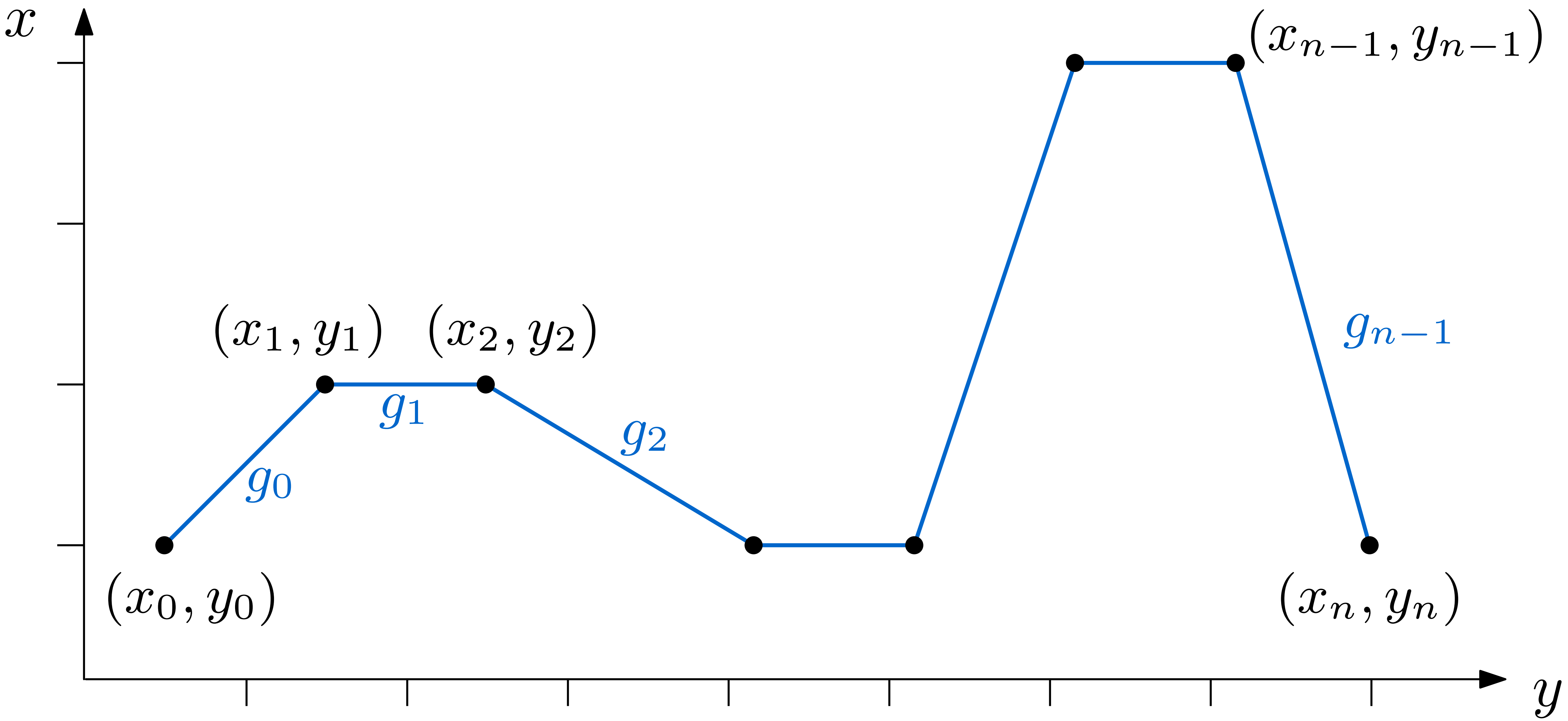
$g(x) = g_i(x)$, $x_i \leq x \leq x_{i+1}$, $0 \leq i \leq n-1$.

Las g_i son polinomios de grado 1.

$$\boxed{g_i(x) = a_i(x - x_i) + b_i}$$

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

Spline lineal: cálculo



La función de interpolación es $y = g(x)$ con

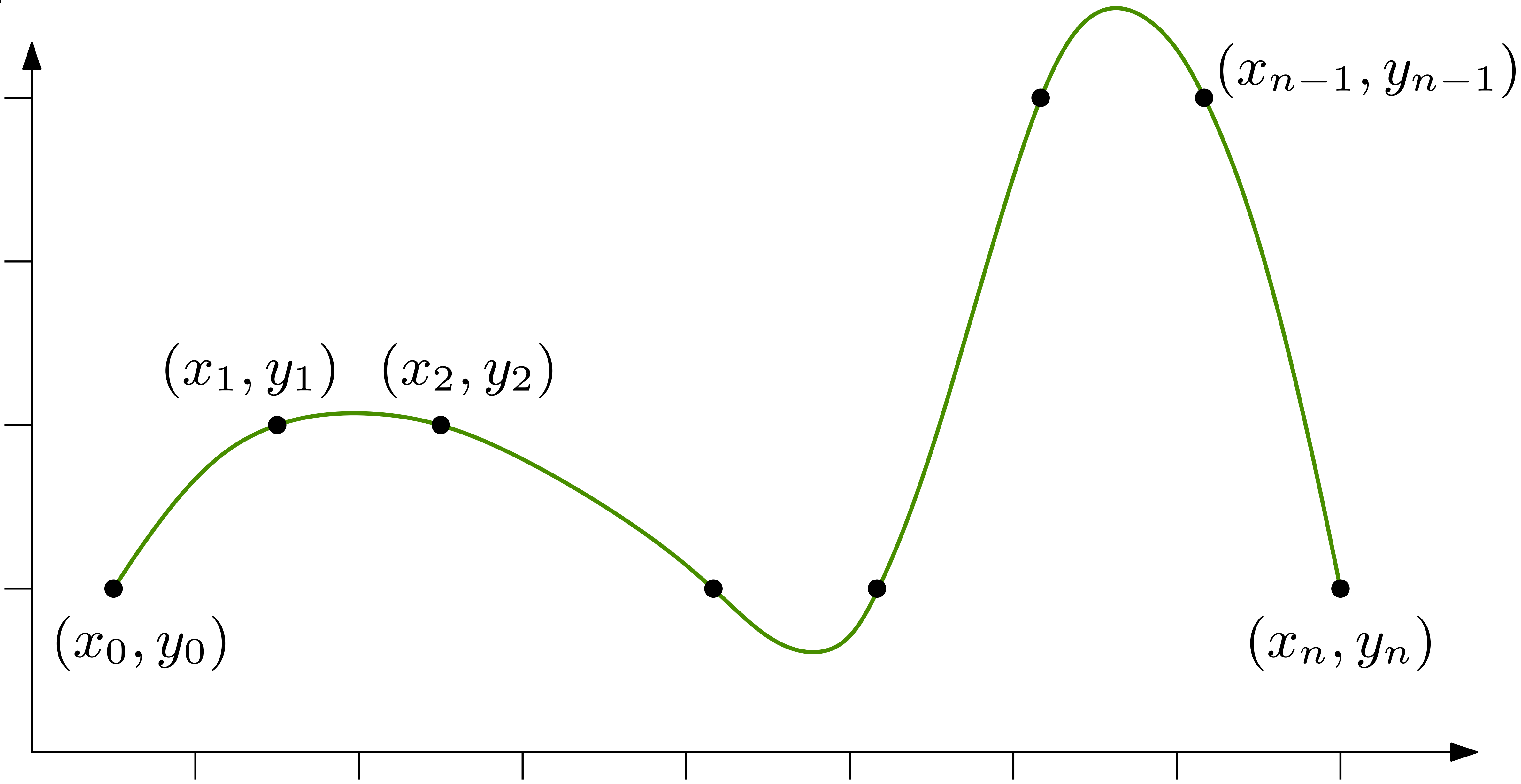
$g(x) = g_i(x)$, $x_i \leq x \leq x_{i+1}$, $0 \leq i \leq n-1$.

Las g_i son polinomios de grado 1.

$$\boxed{g_i(x) = a_i(x - x_i) + b_i}$$

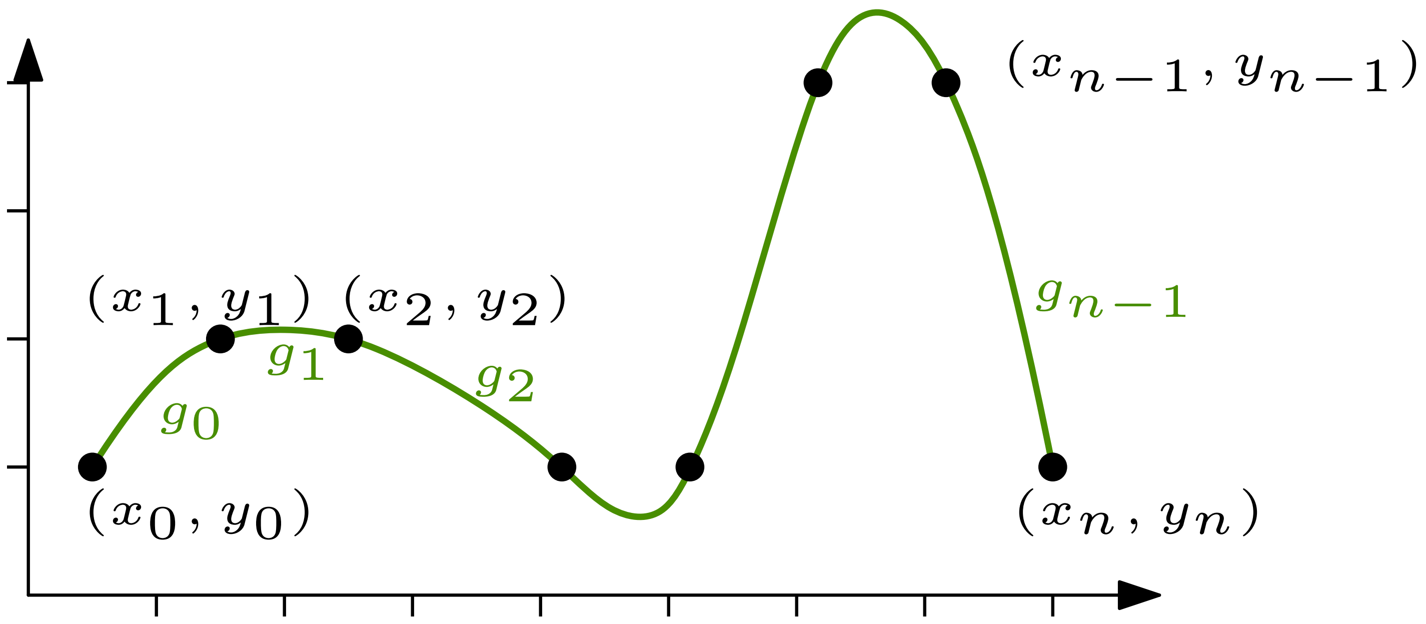
$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \Rightarrow \begin{cases} b_i = y_i \\ a_i = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \end{cases}$$

Spline cúbica



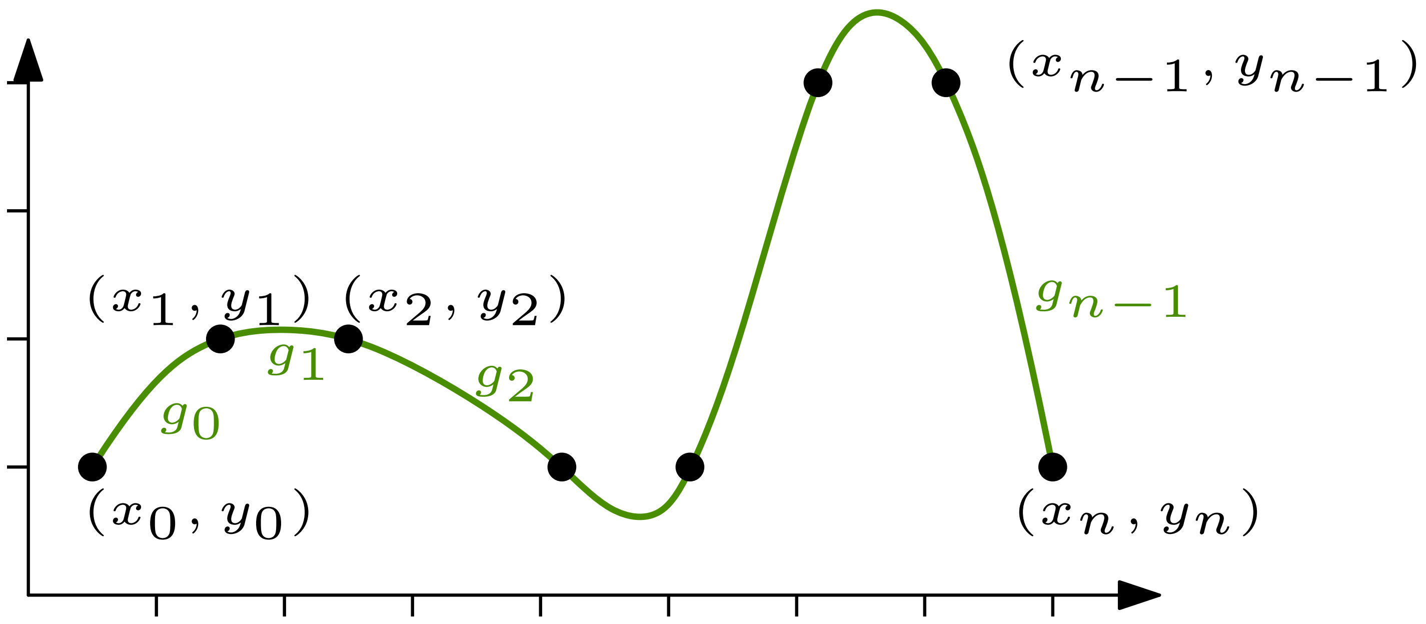
Una spline cúbica es una curva definida por polinomios g_i de grado 3 con continuidad hasta la derivada segunda.

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

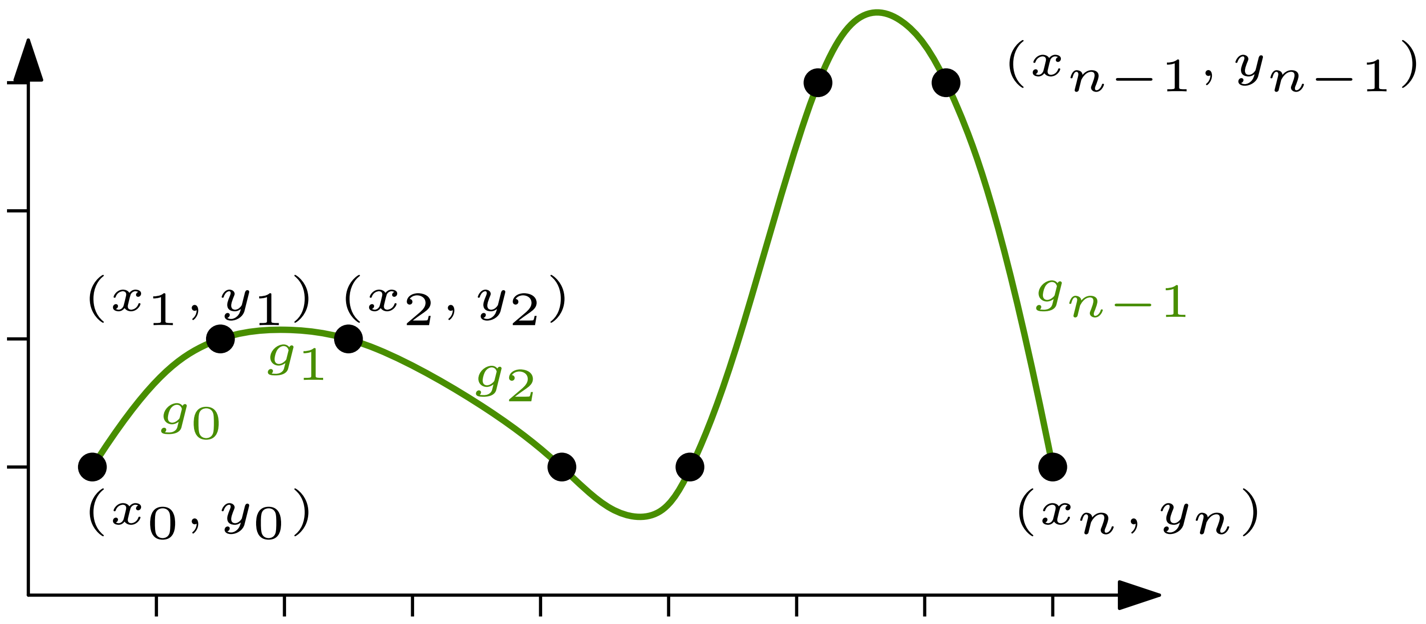
Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

Spline cúbica: restricciones

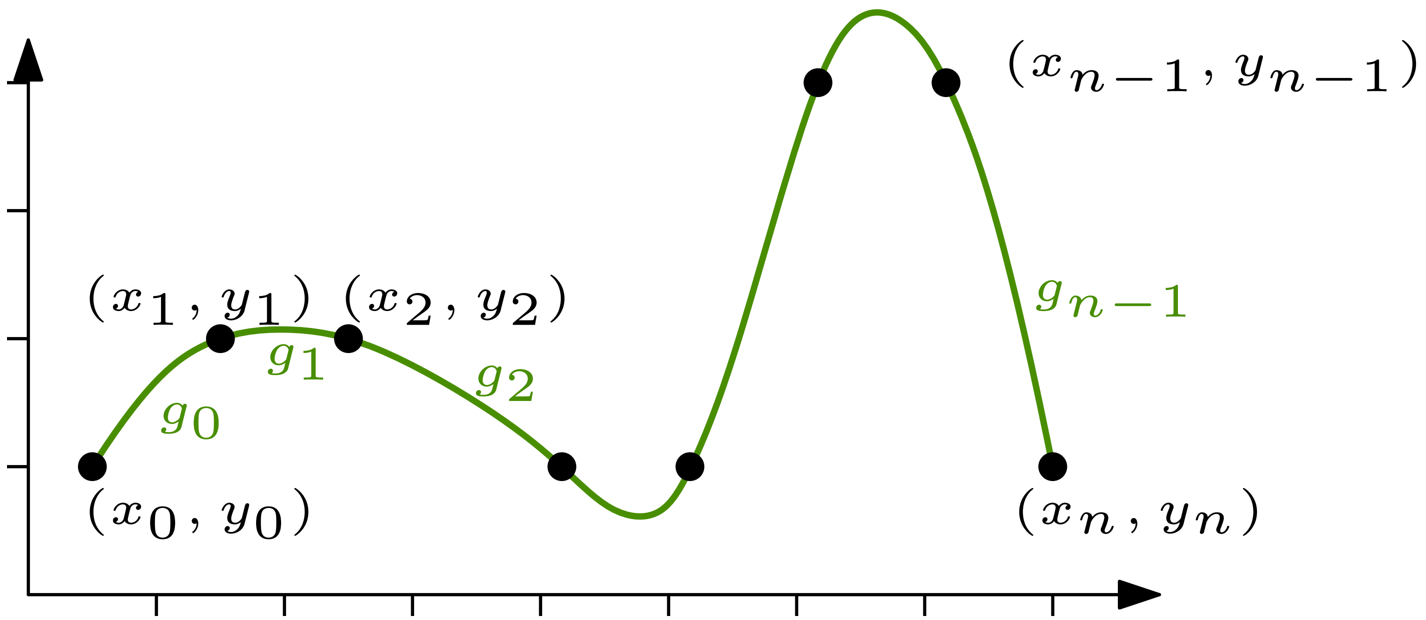


- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

- $4n$ incógnitas

Spline cúbica: restricciones



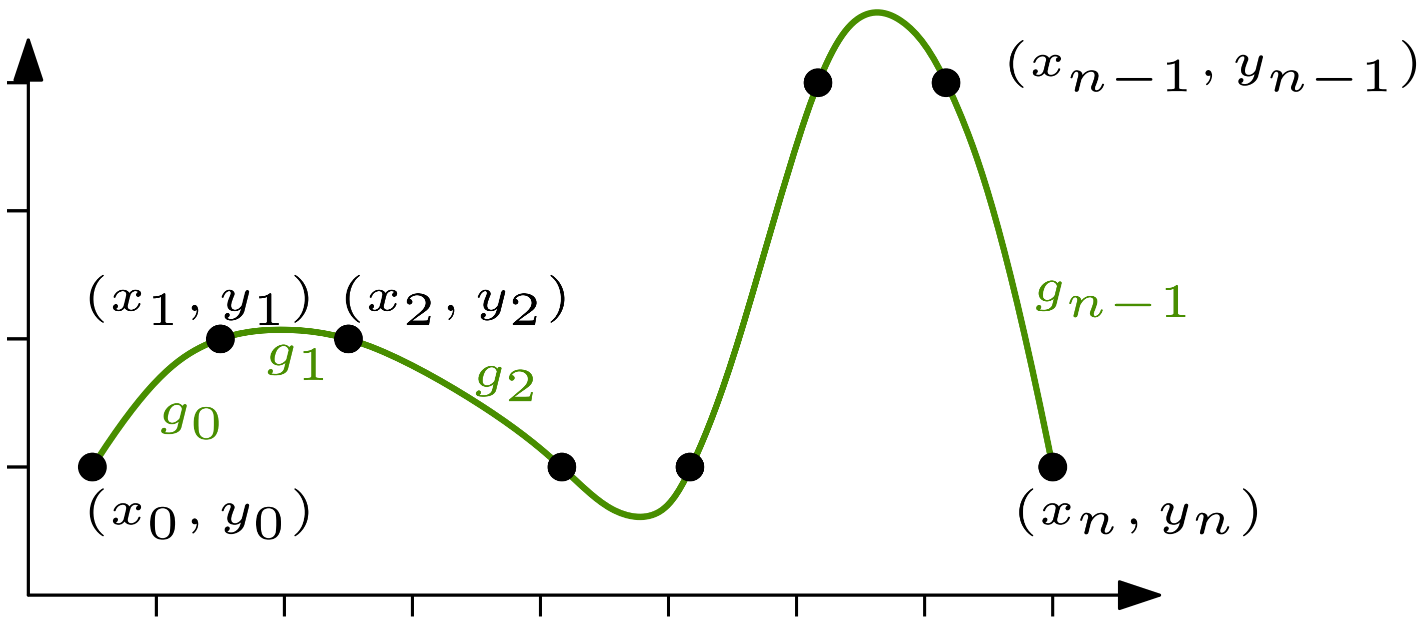
Restricciones:

- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \le i \le n-1$$

- $4n$ incógnitas

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

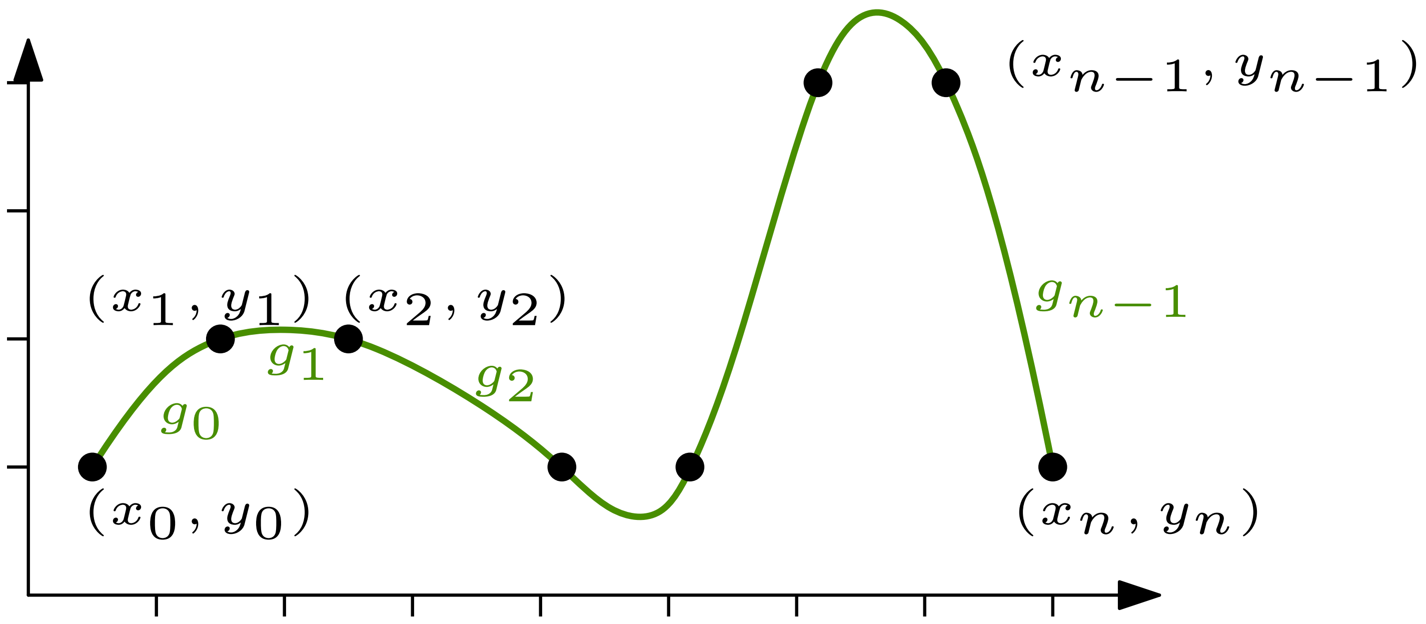
- $4n$ incógnitas

Restricciones:

- Interpolación y continuidad:

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

- $4n$ incógnitas

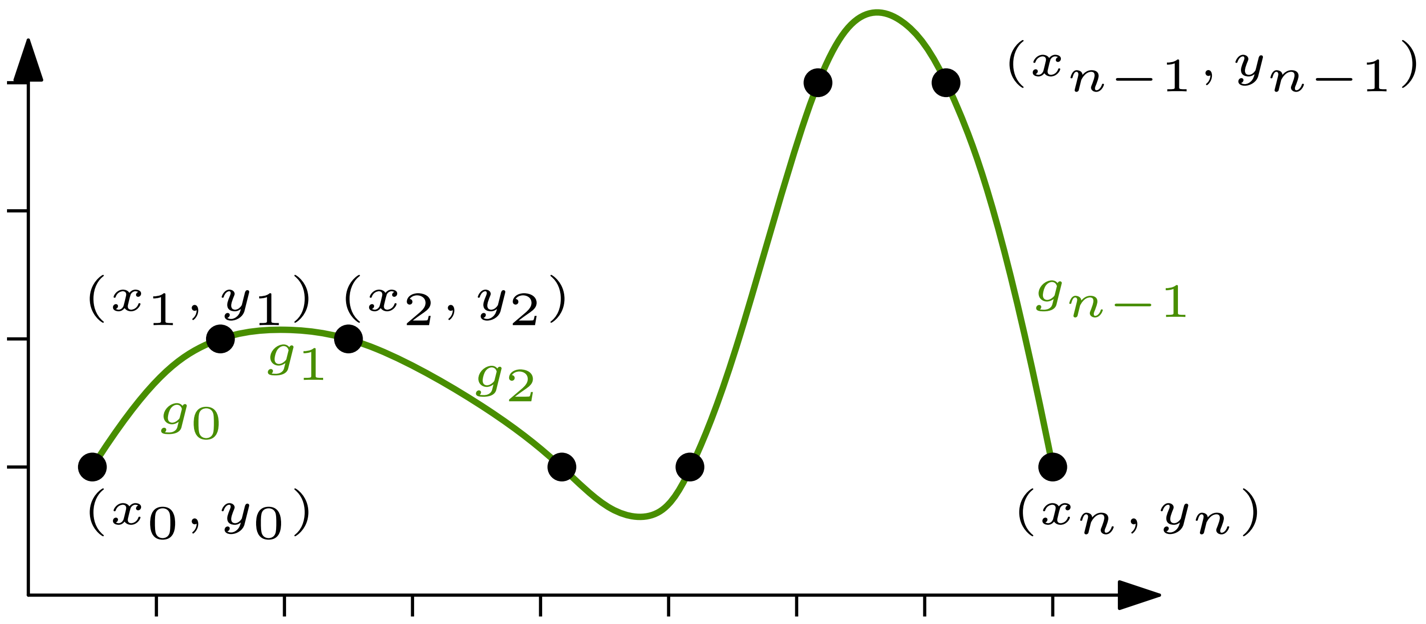
Restricciones:

- Interpolación y continuidad:

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$\rightarrow$ equivalentemente $g_i(x_{i+1}) = g_{i+1}(x_{i+1})$ (pero haría falta una función g_n o añadir la última restricción).

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

- $4n$ incógnitas

Restricciones:

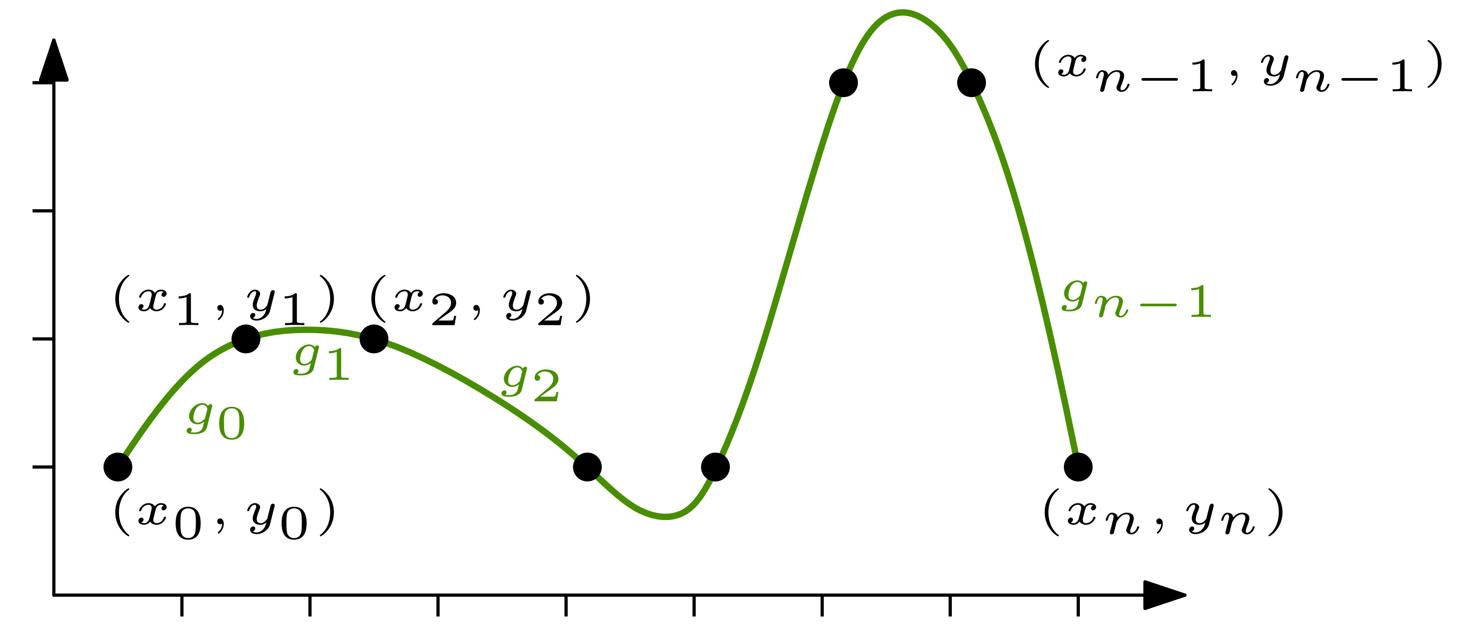
- Interpolación y continuidad:

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones

$$0 \leq i \leq n-1$$

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \le i \le n-1$$

- $4n$ incógnitas

Restricciones:

- Interpolación y continuidad:

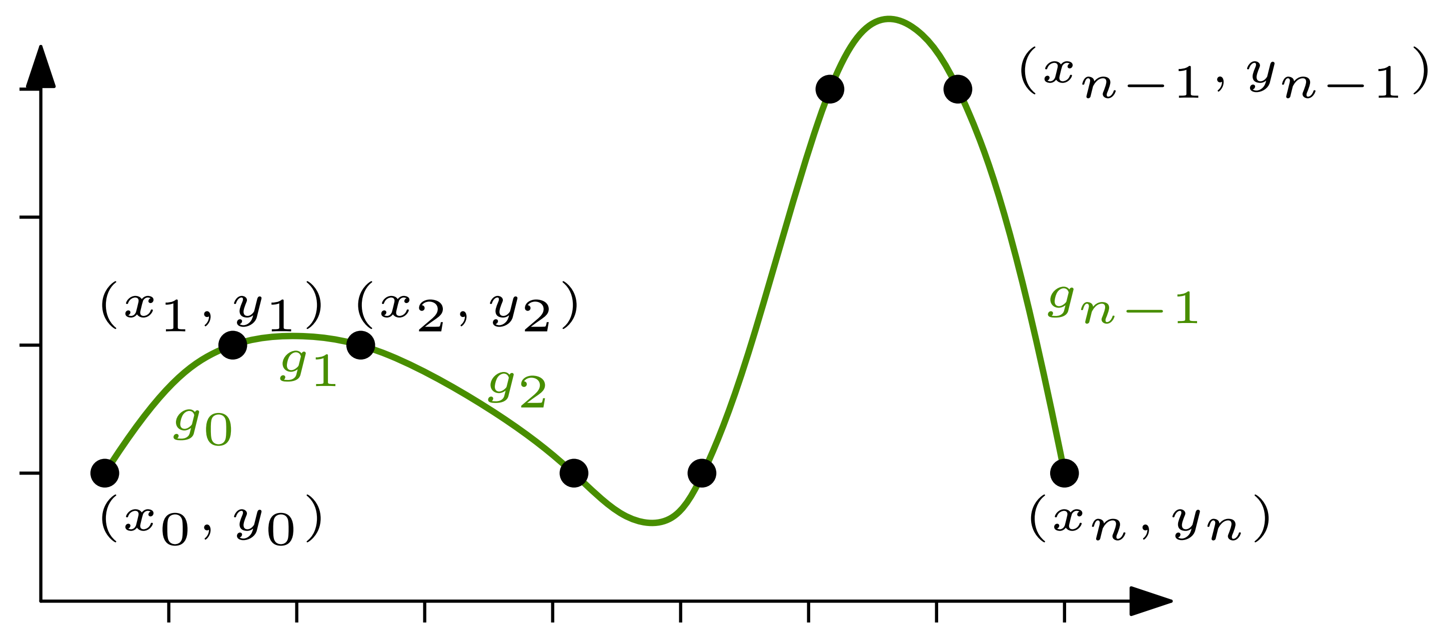
$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones

$$0 \le i \le n-1$$

- Continuidad de la derivada primera: $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \le i \le n-1$$

- $4n$ incógnitas

Restricciones:

- Interpolación y continuidad:

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

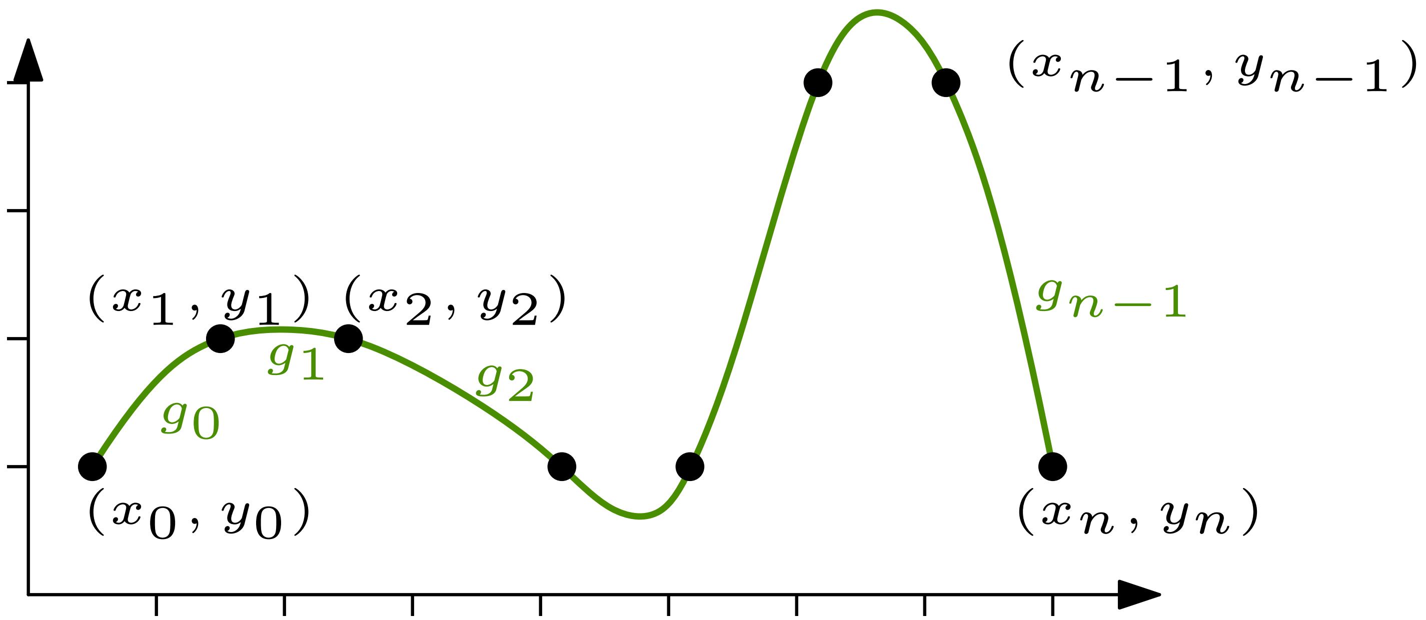
$2n$ restricciones

$$0 \le i \le n-1$$

- Continuidad de la derivada primera: $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $n-1$ restricciones

$$0 \le i \le n-2$$

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

- $4n$ incógnitas

Restricciones:

- Interpolación y continuidad:

$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones

$$0 \leq i \leq n-1$$

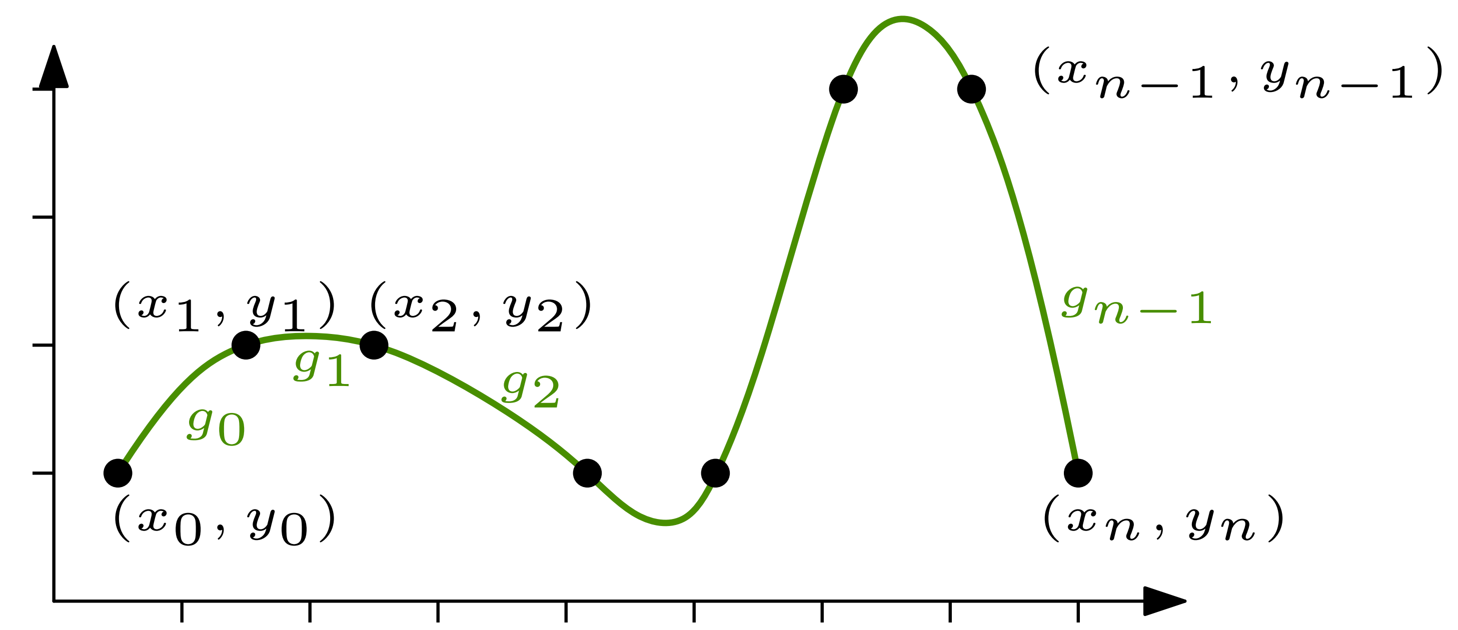
- Continuidad de la derivada primera: $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones

$$0 \leq i \leq n-2$$

- Continuidad de la derivada segunda: $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones

$$0 \leq i \leq n-2$$

Spline cúbica: restricciones



- Las g_i son polinomios de grado 3.
- Continuidad hasta la derivada segunda.

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i$$
$$0 \leq i \leq n-1$$

- $4n$ incógnitas

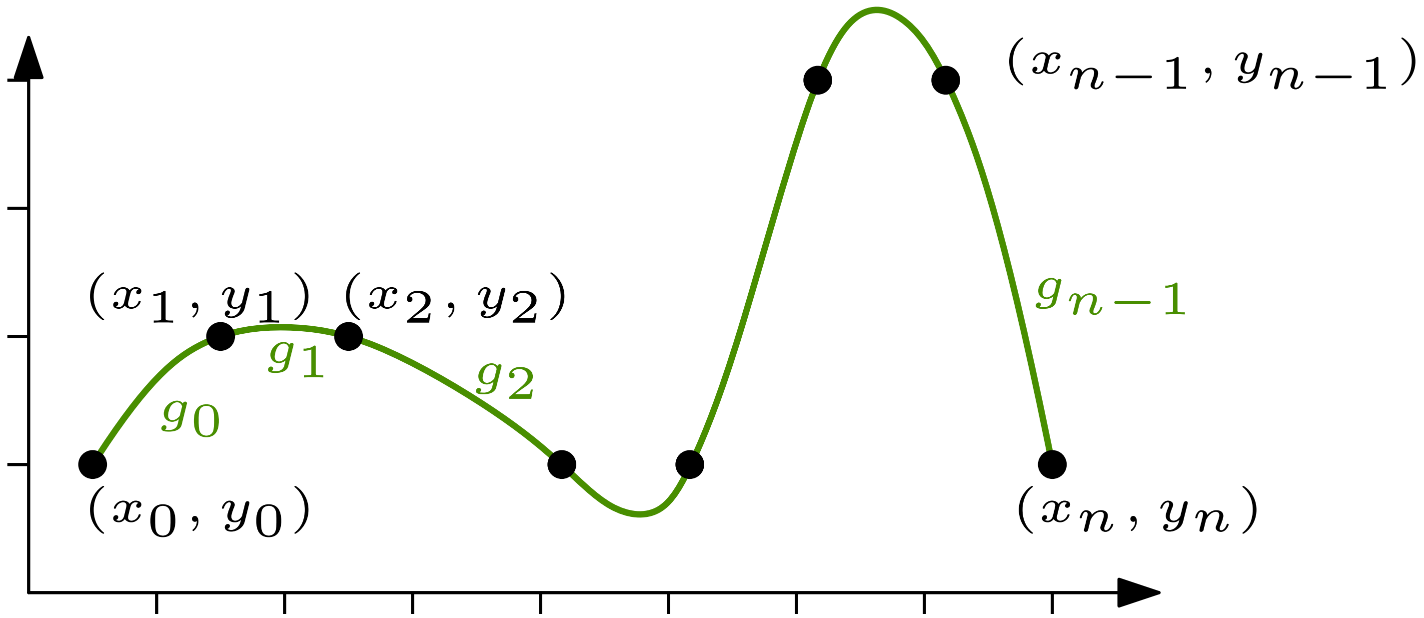
Restricciones:

- Interpolación y continuidad: $\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$ $2n$ restricciones $0 \leq i \leq n-1$
- Continuidad de la derivada primera: $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $n-1$ restricciones
  $0 \leq i \leq n-2$
- Continuidad de la derivada segunda: $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $n-1$ restricciones
  $0 \leq i \leq n-2$

Nos faltan dos restricciones!

Spline cúbica: restricciones adicionales

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

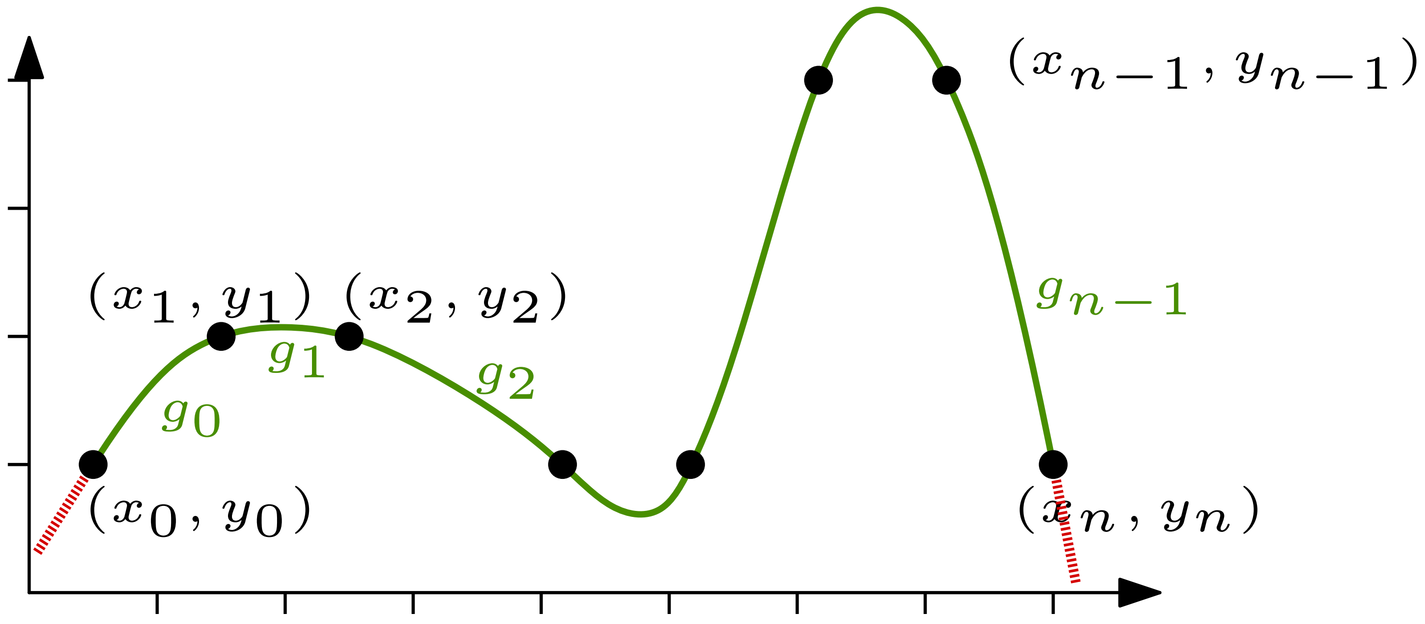
$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad n-1 \text{ restricciones}$$
$$0 \le i \le n-2$$

$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad n-1 \text{ restricciones}$$
$$0 \le i \le n-2$$

Hay varias opciones para las dos restricciones que faltan:

Spline cúbica: restricciones adicionales

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

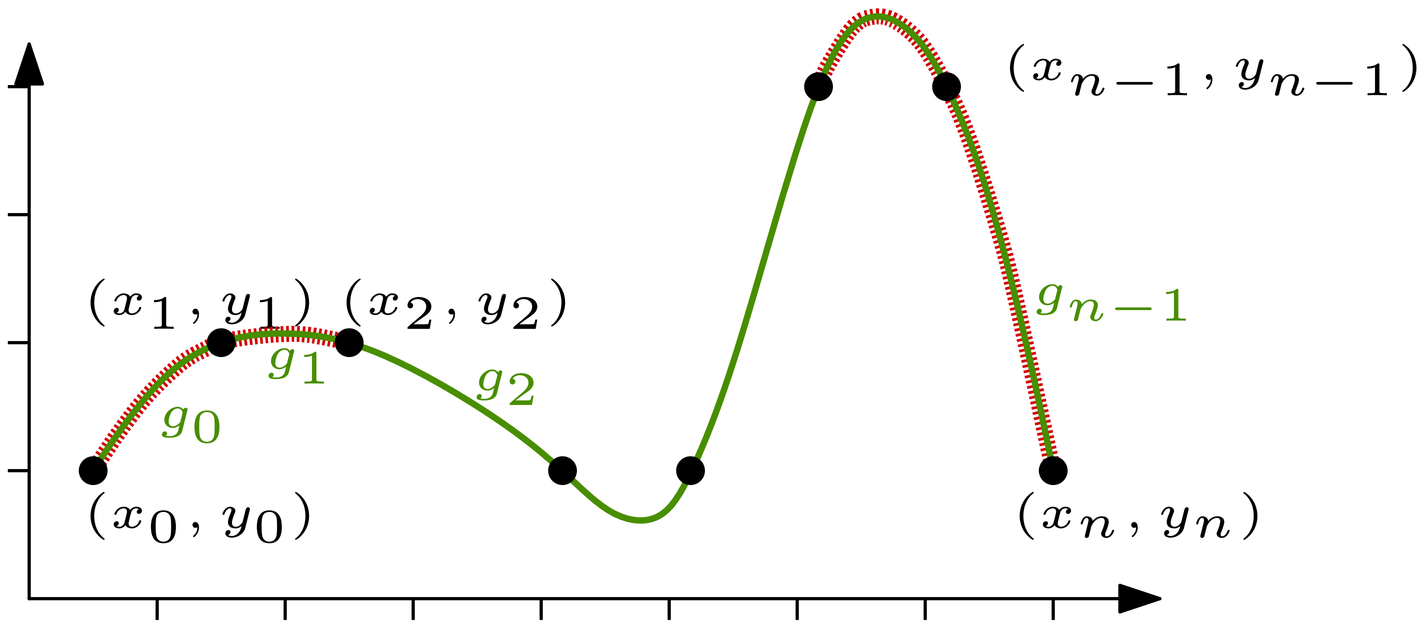
$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad n-1 \text{ restricciones}$$
$$0 \le i \le n-2$$
$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad n-1 \text{ restricciones}$$
$$0 \le i \le n-2$$

Hay varias opciones para las dos restricciones que faltan:

- Spline natural: segundas derivadas en los extremos son 0. $\begin{cases} g_0''(x_0) = 0 \\ g_{n-1}''(x_n) = 0 \end{cases}$

Spline cúbica: restricciones adicionales

$g_i = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad 0 \le i \le n-1$ $\quad 4n$ incógnitas



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \le i \le n-1$

$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \quad 0 \le i \le n-2$$

$n-1$ restricciones

$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \quad 0 \le i \le n-2$$

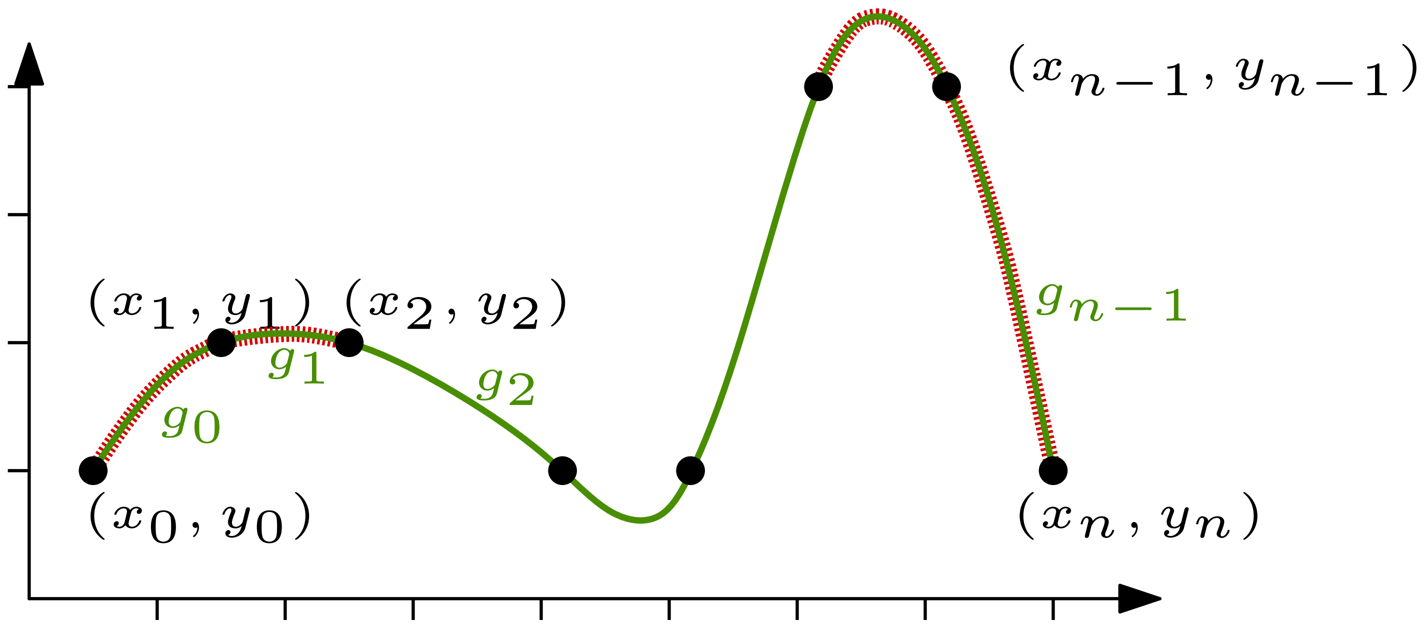
$n-1$ restricciones

Hay varias opciones para las dos restricciones que faltan:

- Spline natural: segundas derivadas en los extremos son 0. $\begin{cases} g_0''(x_0) = 0 \\ g_{n-1}''(x_n) = 0 \end{cases}$
- Not-a-knot (default `spline` en MATLAB):
  $$\begin{cases} g_0(x) = g_1(x) \\ g_{n-2}(x) = g_{n-1}(x) \end{cases}$$

Spline cúbica: restricciones adicionales

$$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n \text{ incógnitas}$$



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad n-1 \text{ restricciones} \quad 0 \le i \le n-2$$

$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad n-1 \text{ restricciones} \quad 0 \le i \le n-2$$

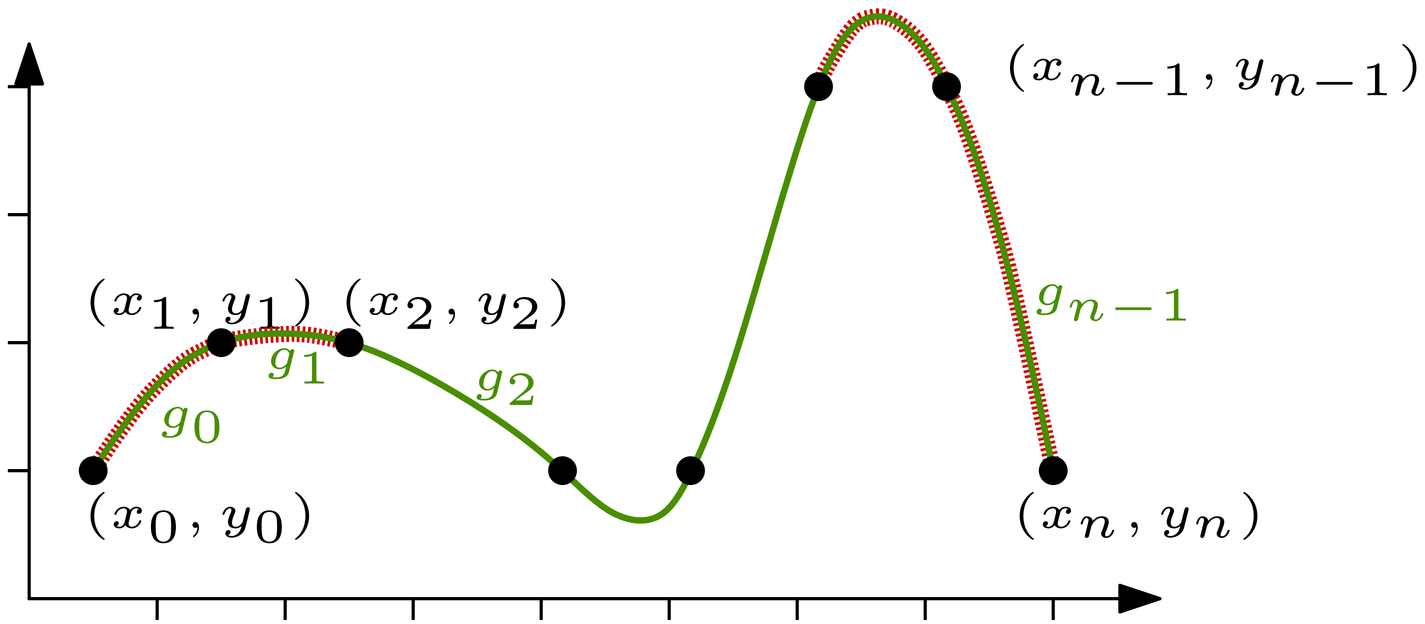
Hay varias opciones para las dos restricciones que faltan:

- Spline natural: segundas derivadas en los extremos son 0. $\begin{cases} g_0''(x_0) = 0 \\ g_{n-1}''(x_n) = 0 \end{cases}$
- Not-a-knot (default `spline` en MATLAB):

$$\begin{cases} g_0(x) = g_1(x) \\ g_{n-2}(x) = g_{n-1}(x) \end{cases} \Leftrightarrow \begin{cases} g_0'''(x_1) = g_1'''(x_1) \\ g_{n-2}'''(x_{n-1}) = g_{n-1}'''(x_{n-1}) \end{cases}$$

Spline cúbica: restricciones adicionales

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad$ $4n$ incógnitas



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$ $2n$ restricciones $\quad 0 \le i \le n-1$

$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \quad 0 \le i \le n-2$$ $n-1$ restricciones

$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \quad 0 \le i \le n-2$$ $n-1$ restricciones

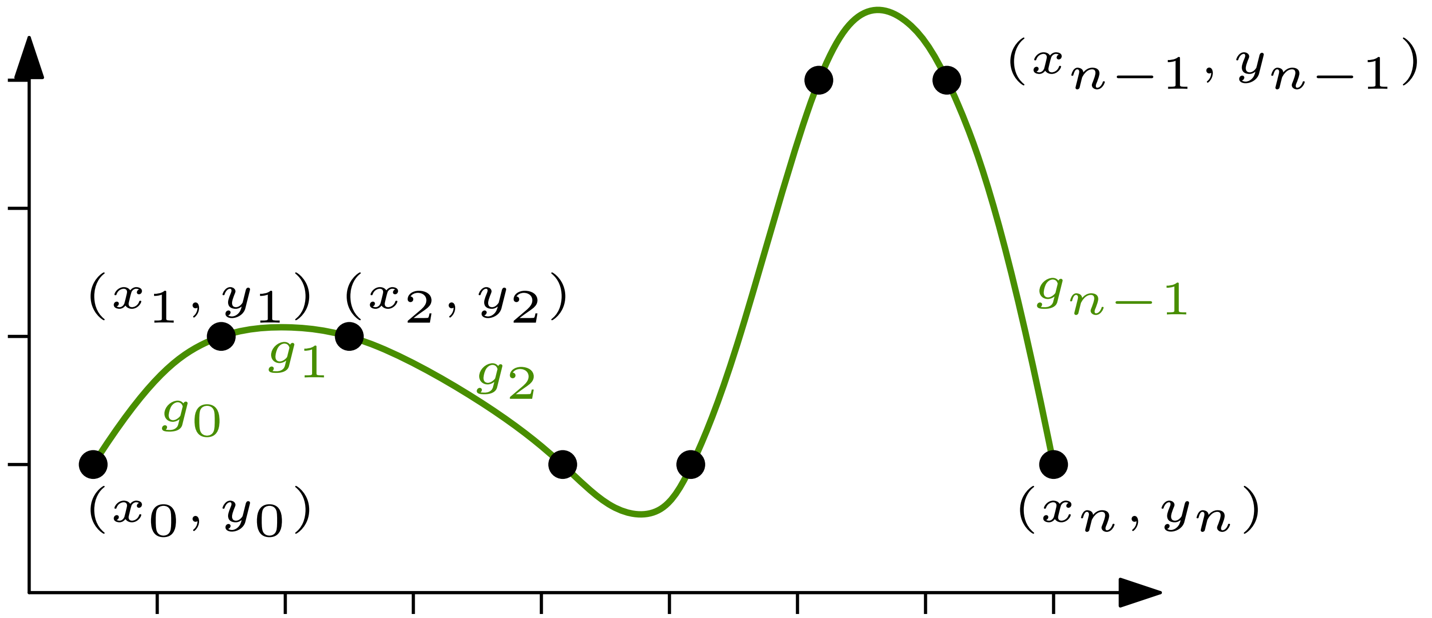
Hay varias opciones para las dos restricciones que faltan:

- Spline natural: segundas derivadas en los extremos son 0. $\begin{cases} g_0''(x_0) = 0 \\ g_{n-1}''(x_n) = 0 \end{cases}$
- Not-a-knot (default `spline` en MATLAB):

$$\begin{cases} g_0(x) = g_1(x) \\ g_{n-2}(x) = g_{n-1}(x) \end{cases} \Leftrightarrow \begin{cases} g_0'''(x_1) = g_1'''(x_1) \\ g_{n-2}'''(x_{n-1}) = g_{n-1}'''(x_{n-1}) \end{cases} \Leftrightarrow \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$$

Spline cúbica: restricciones adicionales

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas



$$\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

$$g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad n-1 \text{ restricciones} \quad 0 \le i \le n-2$$

$$g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad n-1 \text{ restricciones} \quad 0 \le i \le n-2$$

Hay varias opciones para las dos restricciones que faltan:

- Spline natural: segundas derivadas en los extremos son 0. $\begin{cases} g_0''(x_0) = 0 \\ g_{n-1}''(x_n) = 0 \end{cases}$
- Not-a-knot (default `spline` en MATLAB):
- Spline ligado: derivadas en x_0 y en x_n, los valores de $g_0'(x_0)$ y $g_{n-1}'(x_n)$.

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \leq i \leq n-1 \qquad 4n$ incógnitas

$$(0)\ (1) \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \leq i \leq n-1$

$$(2) \quad g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad 0 \leq i \leq n-2$$

$n-1$ restricciones

$$(3) \quad g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad 0 \leq i \leq n-2$$

$n-1$ restricciones

$$(4) \quad \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$$

2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

$$\begin{array}{l} (0) \\ (1) \end{array} \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

$$(2) \quad g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \qquad n-1 \text{ restricciones}$$

$$0 \le i \le n-2$$

$$(3) \quad g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \qquad n-1 \text{ restricciones}$$

$$0 \le i \le n-2$$

$$(4) \quad \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases} \qquad 2 \text{ restricciones}$$

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \leq i \leq n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i;\ \eta_i = y_{i+1} - y_i$

$(4)\ a_0 = a_1;\ \ a_{n-2} = a_{n-1}$

$$(0)\ (1)\ \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \leq i \leq n-1$$

$$(2)\ g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \quad n-1 \text{ restricciones} \quad 0 \leq i \leq n-2$$

$$(3)\ g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \quad n-1 \text{ restricciones} \quad 0 \leq i \leq n-2$$

$$(4)\ \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases} \qquad 2 \text{ restricciones}$$

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i;\ \eta_i = y_{i+1} - y_i$

$(4)\ a_0 = a_1;\ a_{n-2} = a_{n-1}$

$(0)\ d_i = y_i \qquad 0 \le i \le n-1$

$$\begin{matrix}(0)\\(1)\end{matrix}\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1}\end{cases}$$

$2n$ restricciones $\quad 0 \le i \le n-1$

$(2)\ g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones

$0 \le i \le n-2$

$(3)\ g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones

$0 \le i \le n-2$

$$(4)\ \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1}\end{cases}$$

2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \leq i \leq n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

$(4)\ a_0 = a_1;\ a_{n-2} = a_{n-1}$

$(0)\ d_i = y_i \qquad 0 \leq i \leq n-1$

$(1)\ a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i \qquad 0 \leq i \leq n-1$

$$\begin{matrix}(0)\\(1)\end{matrix}\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \leq i \leq n-1$

$(2)\ g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \leq i \leq n-2$

$(3)\ g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \leq i \leq n-2$

$$(4)\ \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$$

2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

(4) $a_0 = a_1;\ a_{n-2} = a_{n-1}$

(0) $d_i = y_i \qquad 0 \le i \le n-1$

(1) $a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i \qquad 0 \le i \le n-1$

(2) $3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1} \qquad 0 \le i \le n-2$

$$\begin{aligned} &(0) \\ &(1) \end{aligned} \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \le i \le n-1$

(2) $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1}) \quad n-1$ restricciones

$0 \le i \le n-2$

(3) $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1}) \quad n-1$ restricciones

$0 \le i \le n-2$

(4) $\begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$ 2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i;\ \eta_i = y_{i+1} - y_i$

(4) $a_0 = a_1;\ a_{n-2} = a_{n-1}$

(0) $d_i = y_i \qquad 0 \le i \le n-1$

(1) $a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i \qquad 0 \le i \le n-1$

(2) $3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1} \qquad 0 \le i \le n-2$

(3) $6a_i h_i + 2b_i = 2b_{i+1} \qquad 0 \le i \le n-2$

$$\begin{array}{l} (0) \\ (1) \end{array} \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases} \qquad 2n \text{ restricciones} \quad 0 \le i \le n-1$$

(2) $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones

$0 \le i \le n-2$

(3) $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones

$0 \le i \le n-2$

$$(4) \quad \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases} \qquad 2 \text{ restricciones}$$

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

(4) $a_0 = a_1$; $a_{n-2} = a_{n-1}$

(0) $d_i = y_i$ $\quad 0 \le i \le n-1$

(1) $a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i$ $\quad 0 \le i \le n-1$

(2) $3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1}$ $\quad 0 \le i \le n-2$

(3) $6a_i h_i + 2b_i = 2b_{i+1}$ $\quad 0 \le i \le n-2$

$0 \le i \le n-1$

$(3) \Rightarrow a_i = \frac{b_{i+1}-b_i}{3h_i}$

$$\begin{aligned}&(0)\\&(1)\end{aligned}\begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1}\end{cases}$$ $2n$ restricciones $\quad 0 \le i \le n-1$

(2) $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

(3) $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

(4) $\begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1}\end{cases}$ $\quad$ 2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i;\ \eta_i = y_{i+1} - y_i$

(4) $a_0 = a_1;\ a_{n-2} = a_{n-1}$

(0) $d_i = y_i$ $\quad 0 \le i \le n-1$

(1) $a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i$ $\quad 0 \le i \le n-1$

(2) $3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1}$ $\quad 0 \le i \le n-2$

(3) $6a_i h_i + 2b_i = 2b_{i+1}$ $\quad 0 \le i \le n-2$

$(3) \Rightarrow a_i = \frac{b_{i+1} - b_i}{3h_i}$ $\quad 0 \le i \le n-1$

$(1) \Rightarrow c_i = \frac{\eta_i}{h_i} - \frac{1}{3} h_i (b_{i+1} + 2b_i)$ $\quad 0 \le i \le n-1$

$$\begin{array}{l} (0) \\ (1) \end{array} \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \le i \le n-1$

(2) $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

(3) $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

$$(4) \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$$

2 restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3+b_i(x-x_i)^2+c_i(x-x_i)+d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

(4) $a_0 = a_1;\ a_{n-2} = a_{n-1}$

(0) $d_i = y_i$ $\quad 0 \le i \le n-1$

(1) $a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i$ $\quad 0 \le i \le n-1$

(2) $3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1}$ $\quad 0 \le i \le n-2$

(3) $6a_i h_i + 2b_i = 2b_{i+1}$ $\quad 0 \le i \le n-2$

(3) $\Rightarrow a_i = \frac{b_{i+1}-b_i}{3h_i}$ $\quad 0 \le i \le n-1$

(1) $\Rightarrow c_i = \frac{\eta_i}{h_i} - \frac{1}{3}h_i(b_{i+1}+2b_i)$ $\quad 0 \le i \le n-1$

(2) $\Rightarrow \frac{1}{3}h_i b_i + \frac{2}{3}(h_i + h_{i+1})b_{i+1} + \frac{1}{3}h_{i+1}b_{i+2} = \frac{\eta_{i+1}}{h_{i+1}} - \frac{\eta_i}{h_i}$ $\quad 0 \le i \le n-2$

$$\begin{array}{l} (0) \\ (1) \end{array} \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$$

$2n$ restricciones $\quad 0 \le i \le n-1$

(2) $g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

(3) $g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

(4) $\begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$ $\quad 2$ restricciones

Spline cúbica (not-a-knot): sistema

$g_i = a_i(x-x_i)^3 + b_i(x-x_i)^2 + c_i(x-x_i) + d_i \qquad 0 \le i \le n-1 \qquad 4n$ incógnitas

Def: $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

$(0)\ \begin{cases} g_i(x_i) = y_i \\ g_i(x_{i+1}) = y_{i+1} \end{cases}$ (1) $\qquad 2n$ restricciones $\quad 0 \le i \le n-1$

$(2)\ g_i'(x_{i+1}) = g_{i+1}'(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

$(3)\ g_i''(x_{i+1}) = g_{i+1}''(x_{i+1})$ $\quad n-1$ restricciones $\quad 0 \le i \le n-2$

$(4)\ \begin{cases} a_0 = a_1 \\ a_{n-2} = a_{n-1} \end{cases}$ $\qquad 2$ restricciones

$(4)\ a_0 = a_1;\ a_{n-2} = a_{n-1}$

$(0)\ d_i = y_i \qquad 0 \le i \le n-1$

$(1)\ a_i h_i^3 + b_i h_i^2 + c_i h_i = \eta_i \qquad 0 \le i \le n-1$

$(2)\ 3a_i h_i^2 + 2b_i h_i + c_i = c_{i+1} \qquad 0 \le i \le n-2$

$(3)\ 6a_i h_i + 2b_i = 2b_{i+1} \qquad 0 \le i \le n-2$

$(3) \Rightarrow a_i = \frac{b_{i+1} - b_i}{3h_i} \qquad 0 \le i \le n-1$

$(1) \Rightarrow c_i = \frac{\eta_i}{h_i} - \frac{1}{3}h_i(b_{i+1} + 2b_i) \qquad 0 \le i \le n-1$

$(2) \Rightarrow \frac{1}{3}h_i b_i + \frac{2}{3}(h_i + h_{i+1})b_{i+1} + \frac{1}{3}h_{i+1}b_{i+2} = \frac{\eta_{i+1}}{h_{i+1}} - \frac{\eta_i}{h_i} \qquad 0 \le i \le n-2$

$(4) \Rightarrow h_1 b_0 - (h_0 + h_1)b_1 + h_0 b_2 = 0;\ h_{n-1}b_{n-2} - (h_{n-2} + h_{n-1})b_{n-1} + h_{n-2}b_n = 0$

Spline cúbica (not-a-knot): resolución

Las b_is forman un sistema (casi) tridiagonal que resolvemos numéricamente.

$$\begin{pmatrix} h_1 & -(h_0+h_1) & h_0 & \dots & 0 & 0 & 0 \\ \frac{1}{3}h_0 & \frac{2}{3}(h_0+h_1) & \frac{1}{3}h_1 & \dots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & \frac{1}{3}h_{n-2} & \frac{2}{3}(h_{n-2}+h_{n-1}) & \frac{1}{3}h_{n-1} \\ 0 & 0 & 0 & \dots & h_{n-1} & -(h_{n-2}+h_{n-1}) & h_{n-2} \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{\eta_1}{h_1} - \frac{\eta_0}{h_0} \\ \vdots \\ \frac{\eta_{n-1}}{h_{n-1}} - \frac{\eta_{n-2}}{h_{n-2}} \\ 0 \end{pmatrix}$$

Spline cúbica (not-a-knot): resolución

Las b_is forman un sistema (casi) tridiagonal que resolvemos numéricamente.

$$\begin{pmatrix} h_1 & -(h_0+h_1) & h_0 & \cdots & 0 & 0 & 0 \\ \frac{1}{3}h_0 & \frac{2}{3}(h_0+h_1) & \frac{1}{3}h_1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{3}h_{n-2} & \frac{2}{3}(h_{n-2}+h_{n-1}) & \frac{1}{3}h_{n-1} \\ 0 & 0 & 0 & \cdots & h_{n-1} & -(h_{n-2}+h_{n-1}) & h_{n-2} \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \\ \boxed{b_n} \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{\eta_1}{h_1} - \frac{\eta_0}{h_0} \\ \vdots \\ \frac{\eta_{n-1}}{h_{n-1}} - \frac{\eta_{n-2}}{h_{n-2}} \\ 0 \end{pmatrix}$$

Spline cúbica (not-a-knot): resolución

Las b_is forman un sistema (casi) tridiagonal que resolvemos numéricamente.

$$\begin{pmatrix} h_1 & -(h_0+h_1) & h_0 & \cdots & 0 & 0 & 0 \\ \frac{1}{3}h_0 & \frac{2}{3}(h_0+h_1) & \frac{1}{3}h_1 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{3}h_{n-2} & \frac{2}{3}(h_{n-2}+h_{n-1}) & \frac{1}{3}h_{n-1} \\ 0 & 0 & 0 & \cdots & h_{n-1} & -(h_{n-2}+h_{n-1}) & h_{n-2} \end{pmatrix} \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_{n-1} \\ b_n \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{\eta_1}{h_1} - \frac{\eta_0}{h_0} \\ \vdots \\ \frac{\eta_{n-1}}{h_{n-1}} - \frac{\eta_{n-2}}{h_{n-2}} \\ 0 \end{pmatrix}$$

Para el resto, calculamos $h_i = x_{i+1} - x_i$; $\eta_i = y_{i+1} - y_i$

$$d_i = y_i$$
$$a_i = \frac{b_{i+1} - b_i}{3h_i} \qquad 0 \le i \le n-1$$
$$c_i = \frac{\eta_i}{h_i} - \frac{1}{3}h_i(b_{i+1} + 2b_i)$$

Splines de Hermite

- En general, las interpolaciones de Hermite satisfacen ciertos valores (dados o calculados) de la derivada en los nudos.

Splines de Hermite

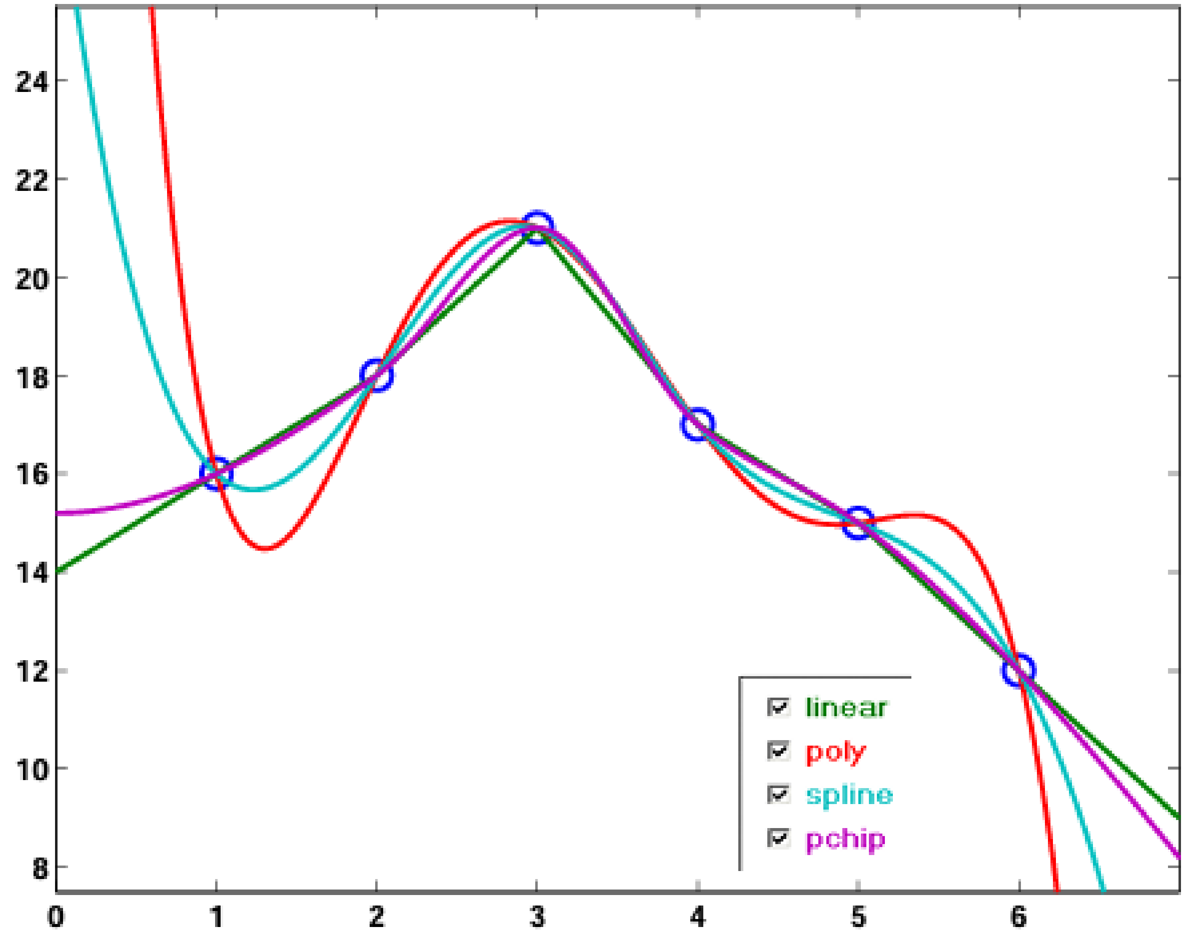
- En general, las interpolaciones de Hermite satisfacen ciertos valores (dados o calculados) de la derivada en los nudos.
- Las splines de Hermite más relevantes son cúbicas a trozos.

Splines de Hermite

- En general, las interpolaciones de Hermite satisfacen ciertos valores (dados o calculados) de la derivada en los nudos.
- Las splines de Hermite más relevantes son cúbicas a trozos.
- Si no se conocen las derivadas, se aproximan a partir de los nudos. Ejemplo: en MATLAB `pchip` (Piecewise Cubic Hermite Interpolating Polynomial), Catmull-Rom, ...

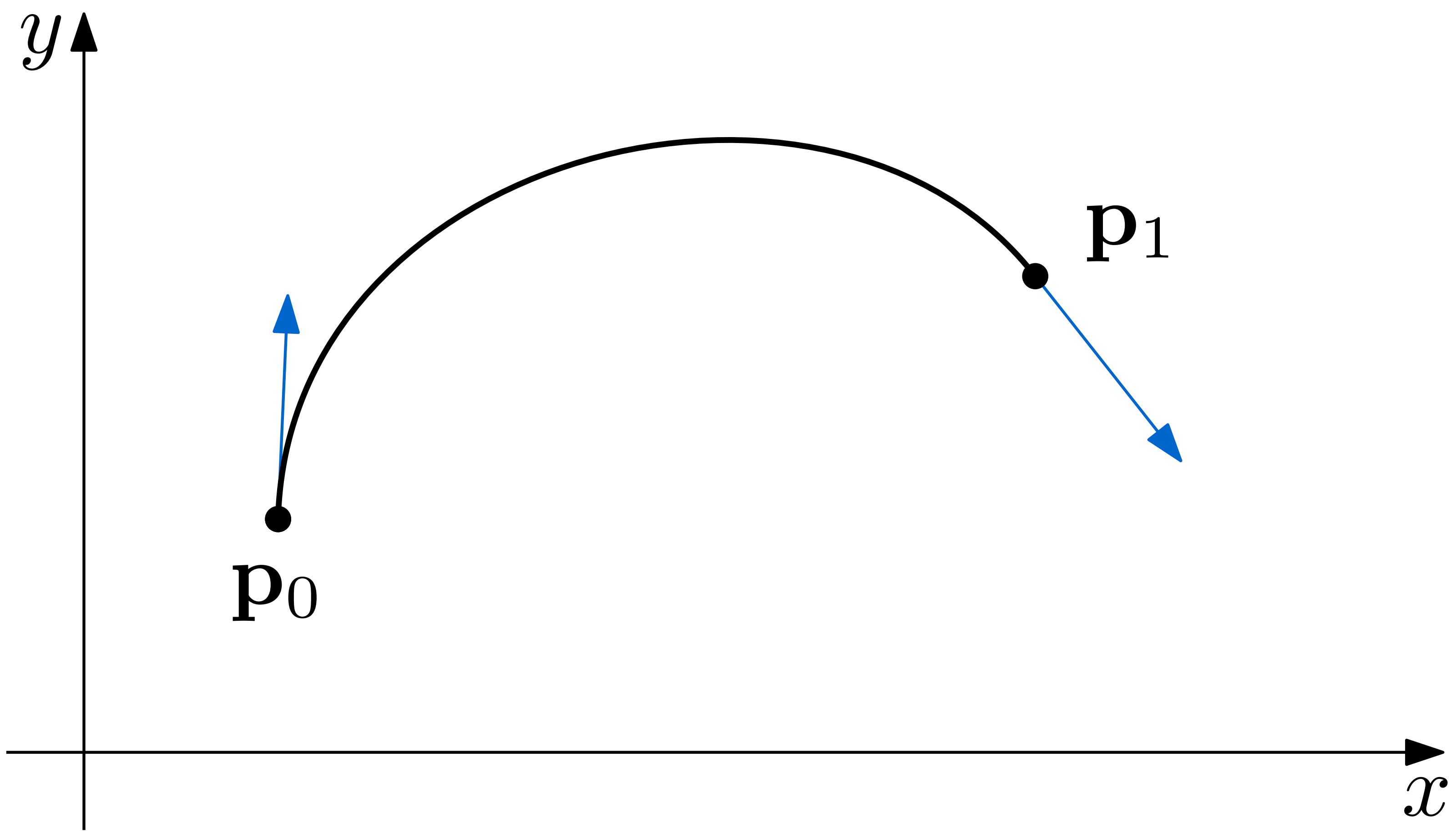
Splines de Hermite

- En general, las interpolaciones de Hermite satisfacen ciertos valores (dados o calculados) de la derivada en los nudos.
- Las splines de Hermite más relevantes son cúbicas a trozos.
- Si no se conocen las derivadas, se aproximan a partir de los nudos. Ejemplo: en MATLAB `pchip` (Piecewise Cubic Hermite Interpolating Polynomial), Catmull-Rom, ...



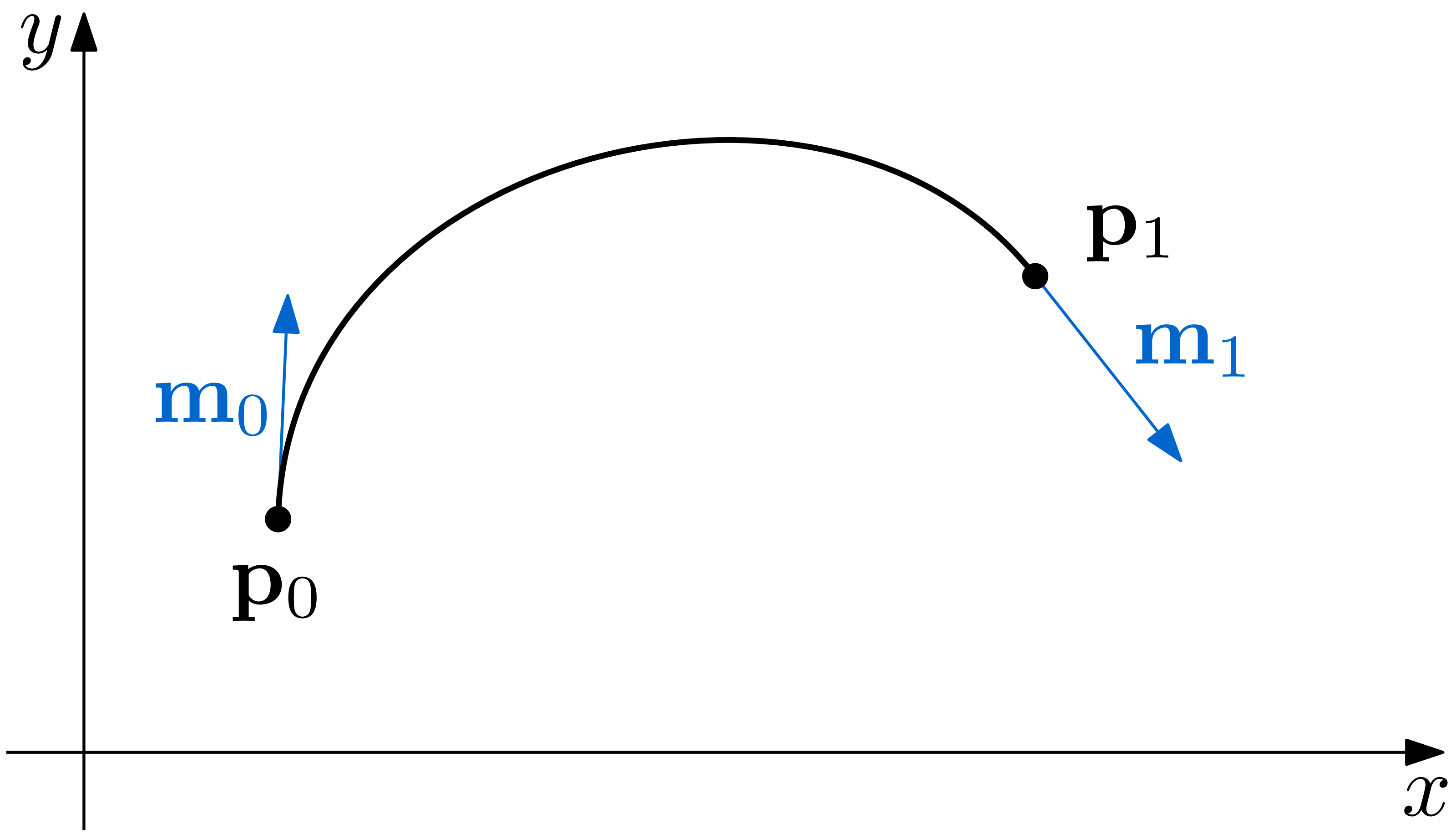
Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.



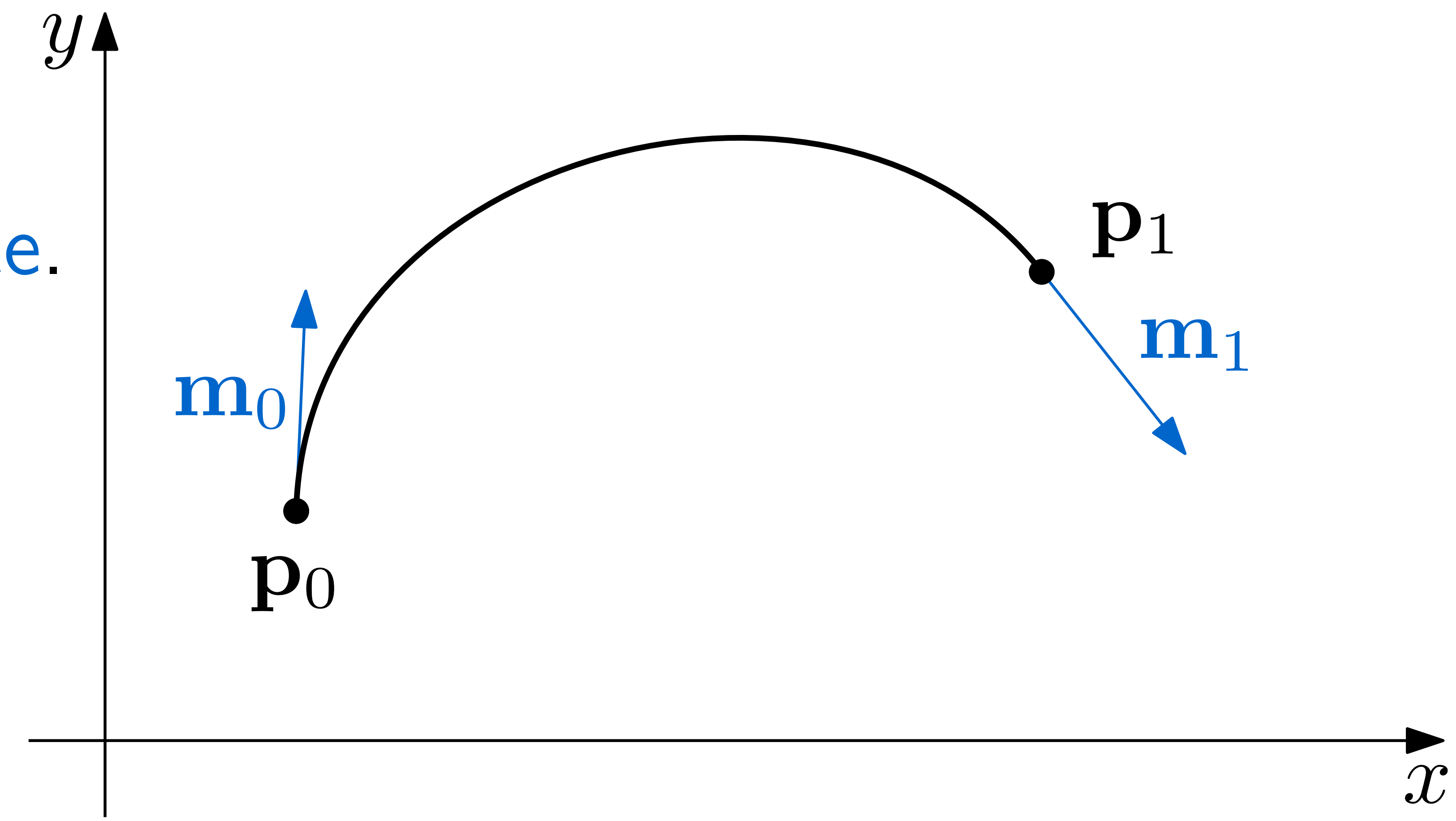
Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.



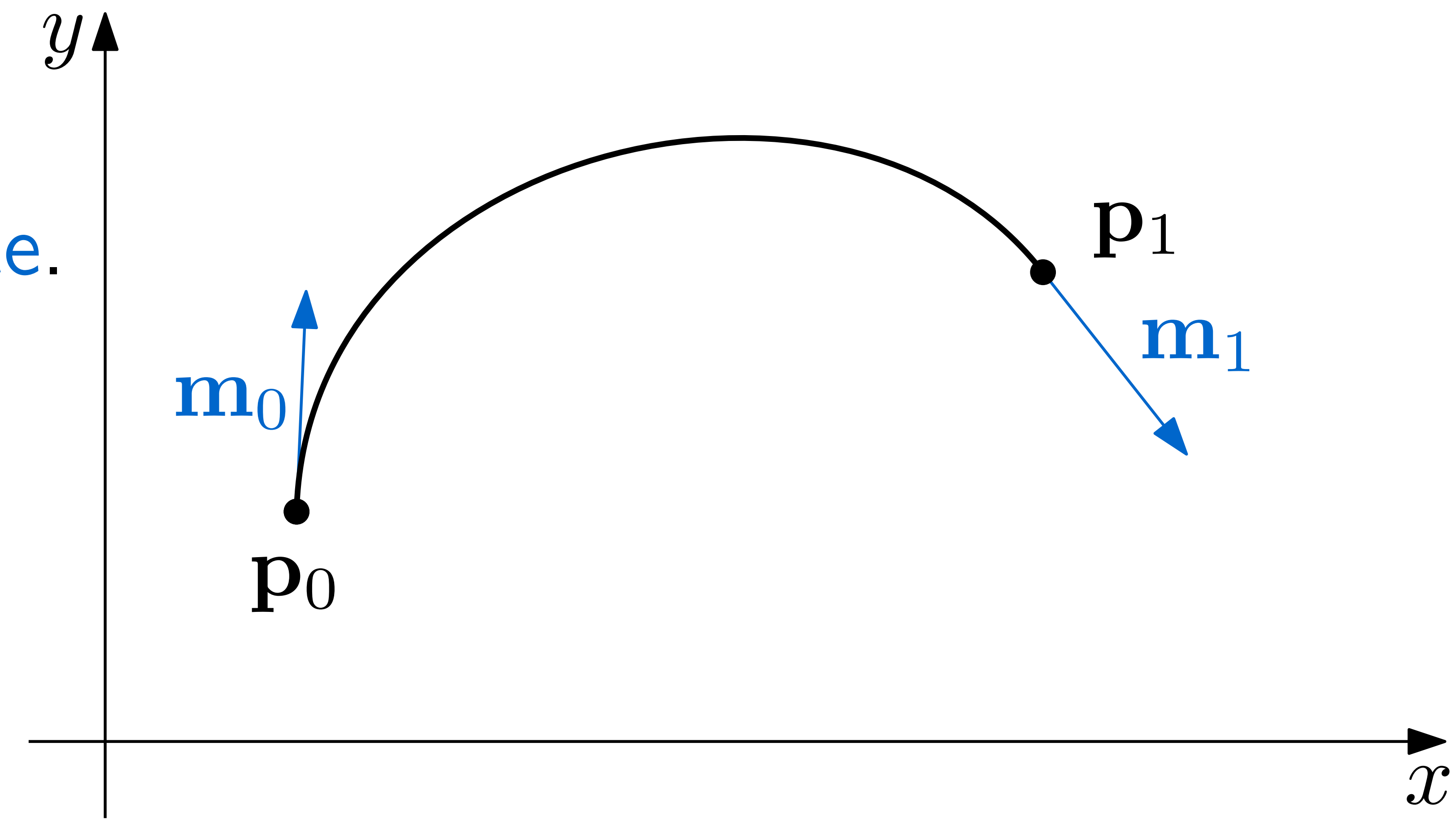
Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.
- Se usa una spline cúbica de Hermite.



Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.
- Se usa una spline cúbica de Hermite.
- Si no se dan las tangentes, hay un parámetro de tensión y distintas opciones para elegirlas.



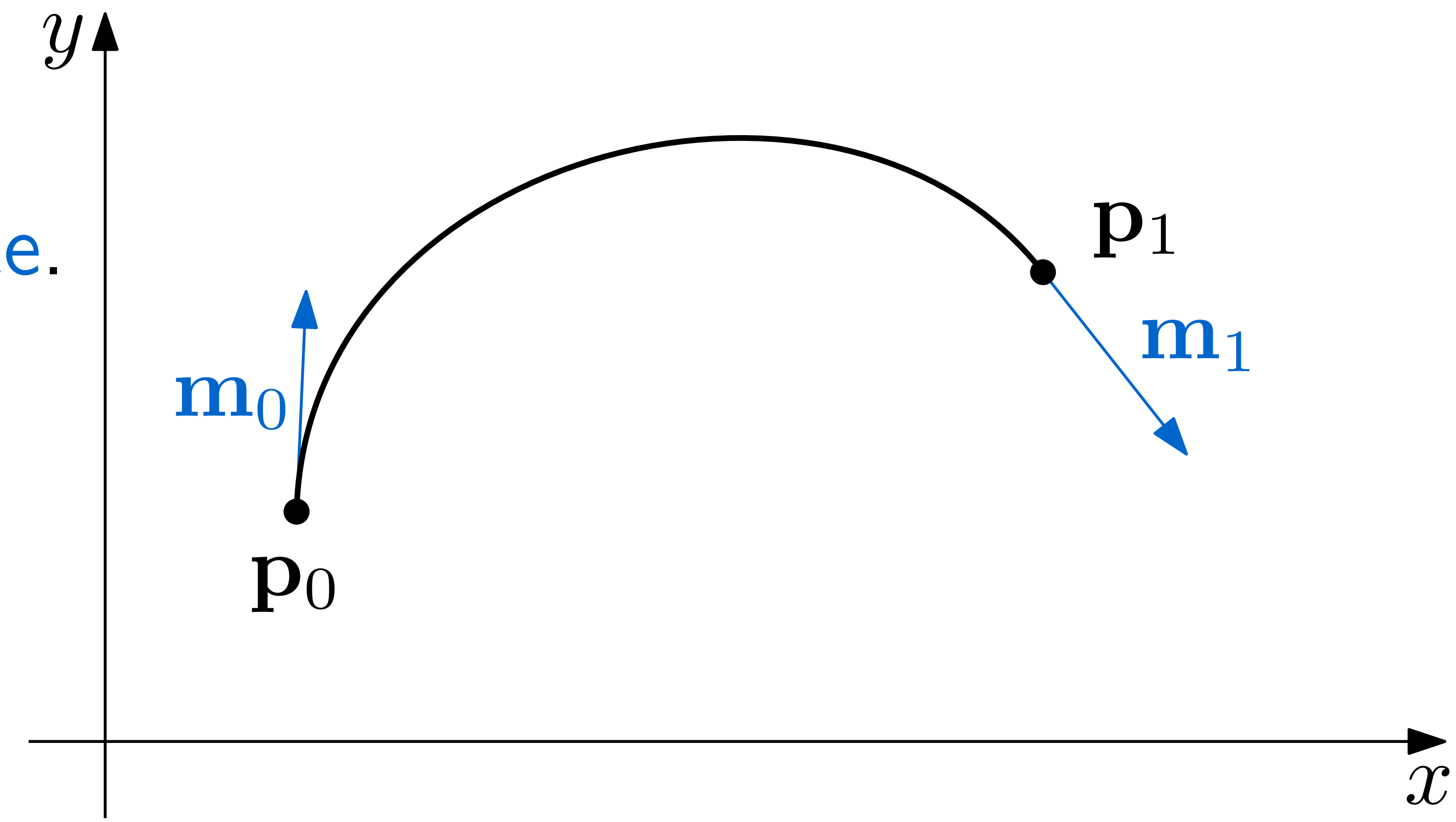
Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.
- Se usa una spline cúbica de Hermite.
- Si no se dan las tangentes, hay un parámetro de tensión y distintas opciones para elegirlas.

  Ejemplo: spline de Catmull–Rom

$$\mathbf{m}_k = \frac{\mathbf{p}_{k+1} - \mathbf{p}_{k-1}}{2}.$$

(Asume un espaciado uniforme y tensión 0.)



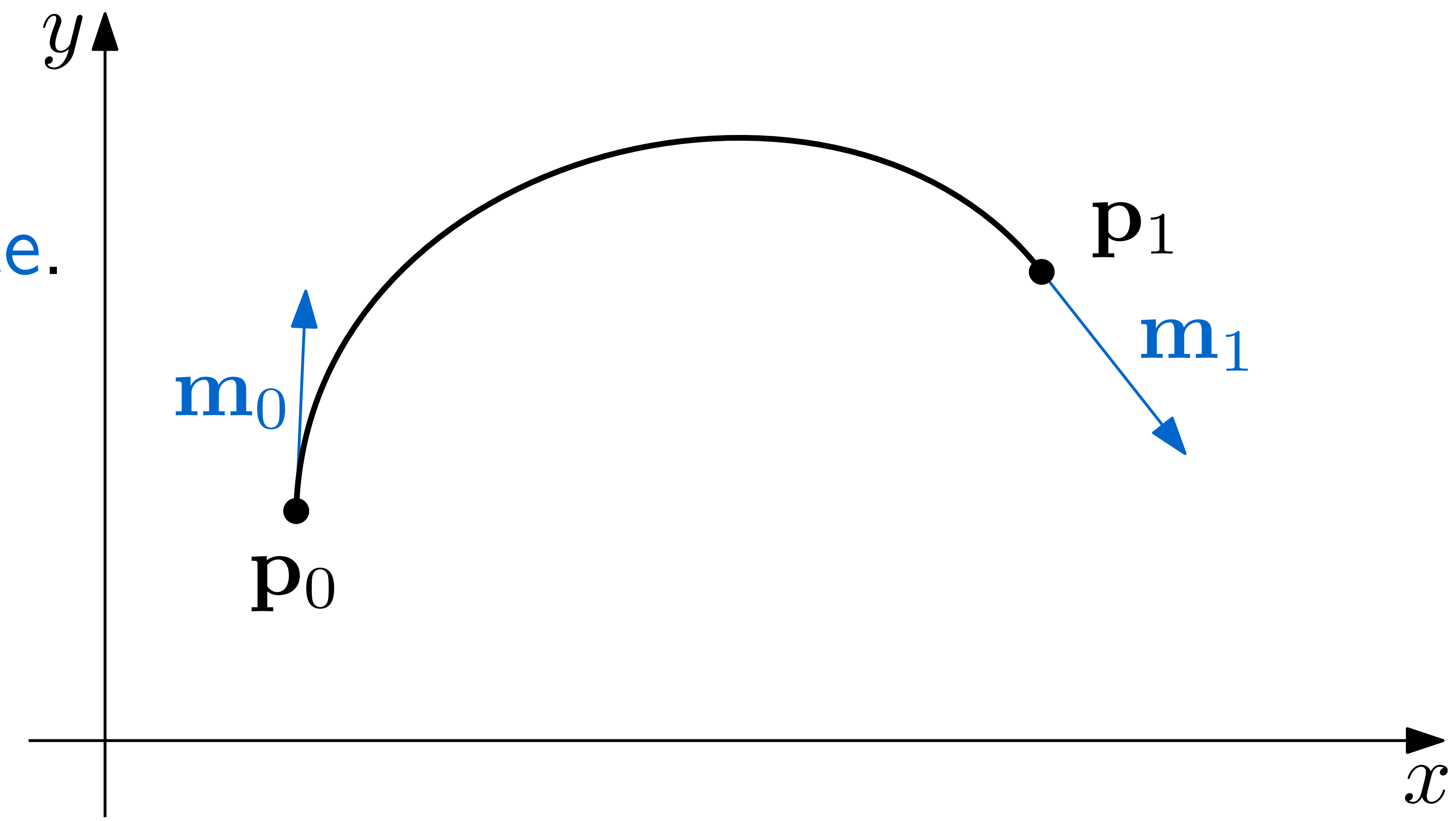
Curvas paramétricas con tangentes: spline cúbica de Hermite

- Muy utilizados en computación gráfica: tangentes dadas $\mathbf{m}_i$.
- Se usa una spline cúbica de Hermite.
- Si no se dan las tangentes, hay un parámetro de tensión y distintas opciones para elegirlas.

  Ejemplo: spline de Catmull–Rom

$$\mathbf{m}_k = \frac{\mathbf{p}_{k+1} - \mathbf{p}_{k-1}}{2}.$$

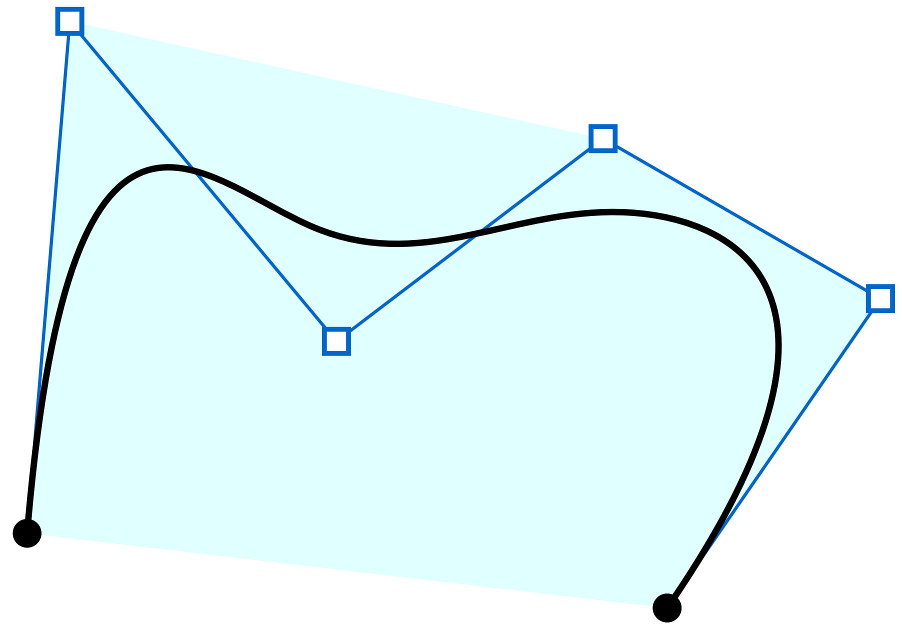
  (Asume un espaciado uniforme y tensión 0.)

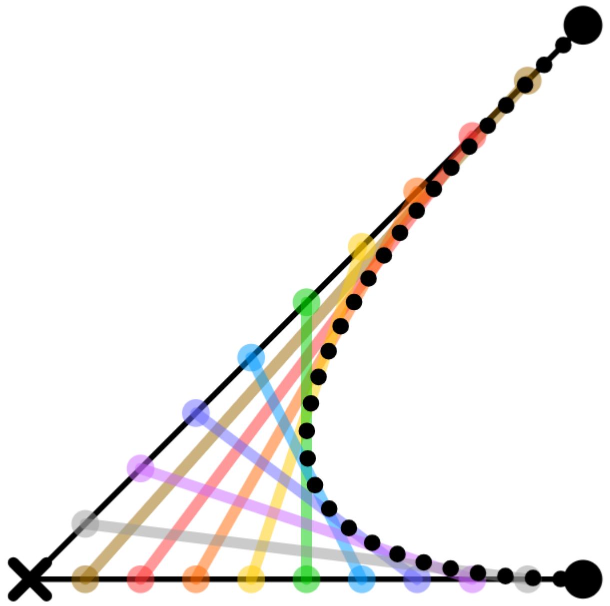


- No es necesario recalcular todo si se decide modificar una parte de la curva.

Curvas de Bézier

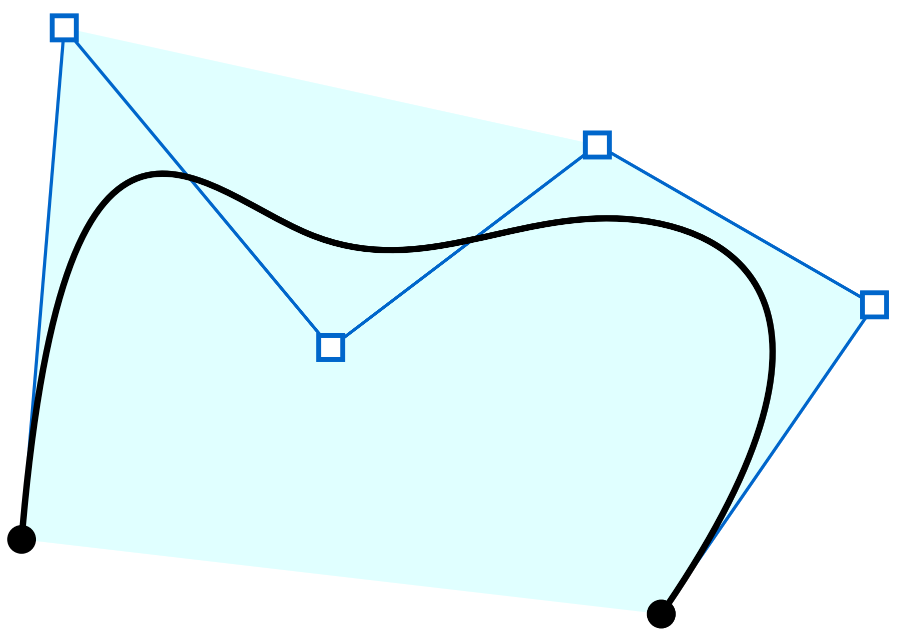
- La curva de Bézier es una curva paramétrica que, a partir de unos puntos de control, permite al usuario modelar a voluntad.

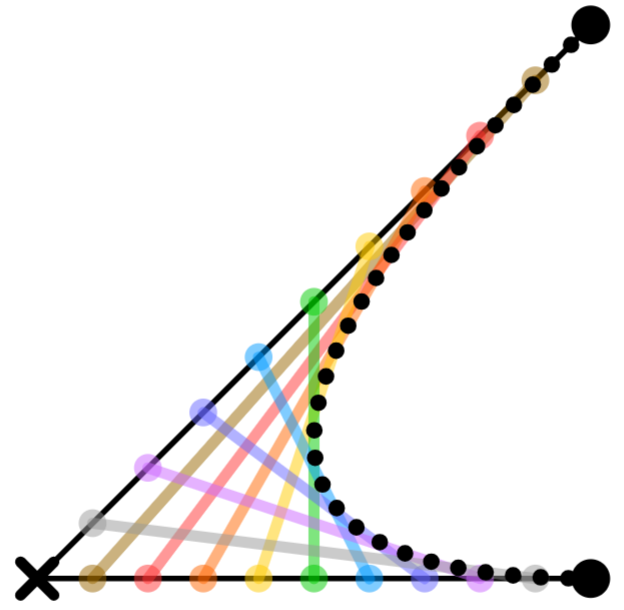




Curvas de Bézier

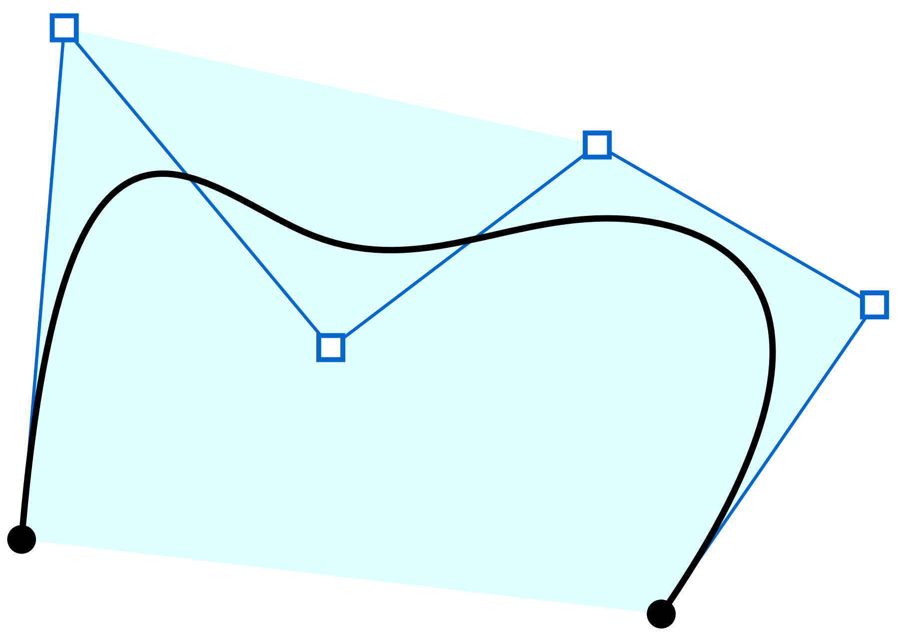
- La curva de Bézier es una curva paramétrica que, a partir de unos puntos de control, permite al usuario modelar a voluntad.
- Su aplicación inicial fue en el diseño de carrocerías de automóviles, barcos, hélices de barcos, etc.

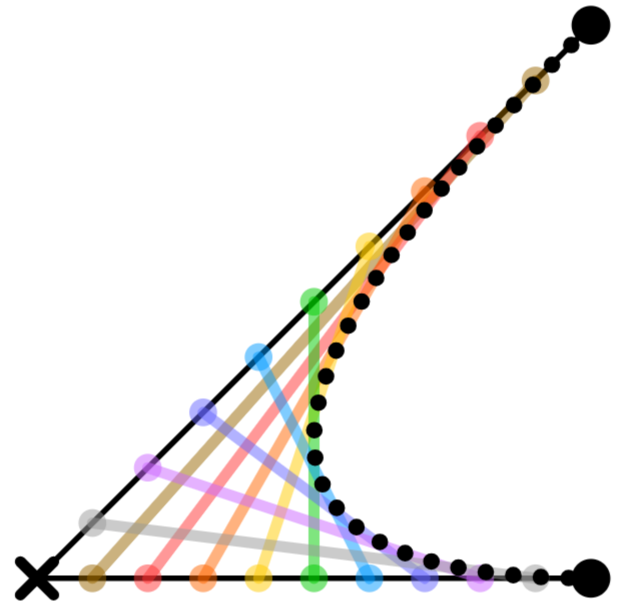




Curvas de Bézier

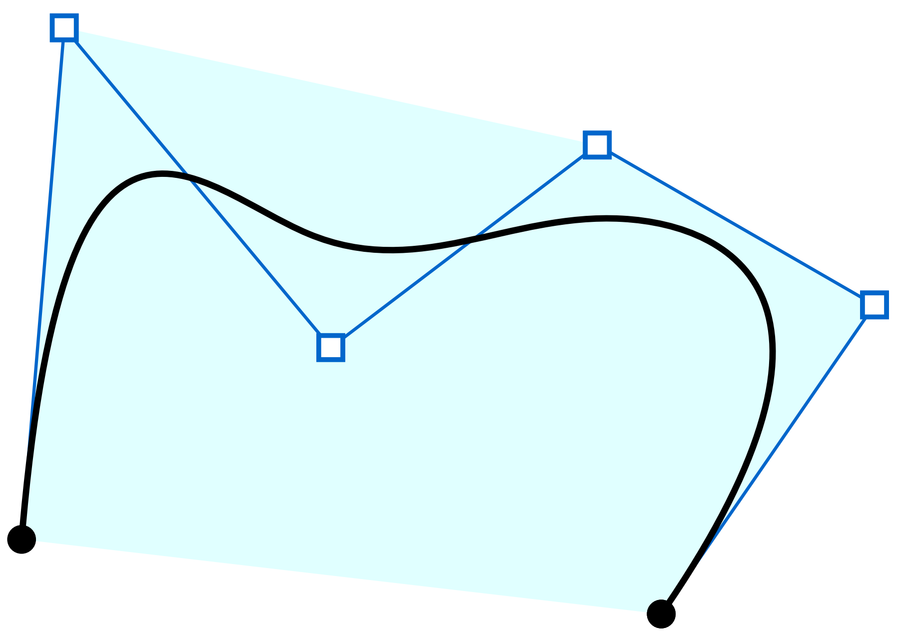
- La curva de Bézier es una curva paramétrica que, a partir de unos puntos de control, permite al usuario modelar a voluntad.
- Su aplicación inicial fue en el diseño de carrocerías de automóviles, barcos, hélices de barcos, etc.
- Los pioneros fueron Pierre Bézier en Renault y Paul de Faget de Casteljau en Citroën.

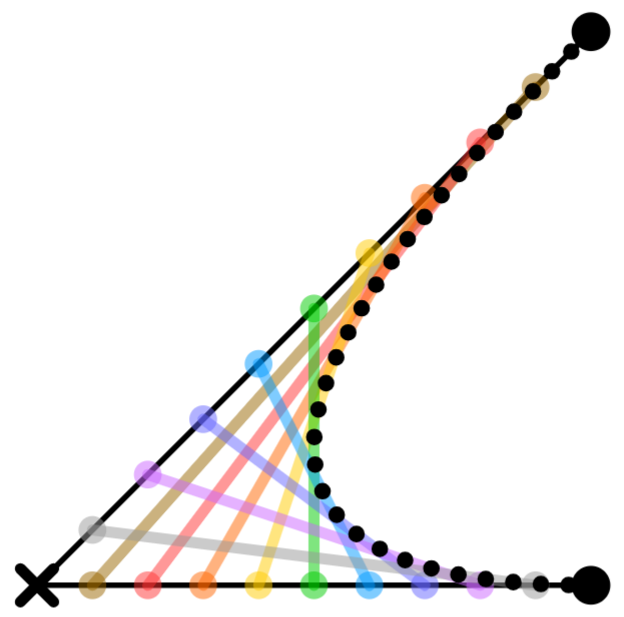


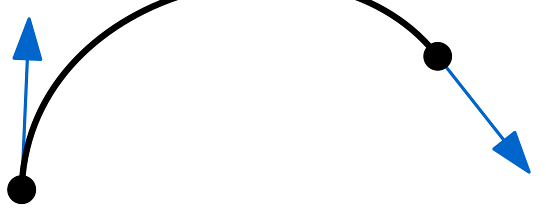


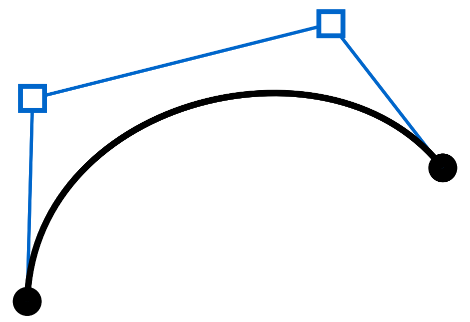
Curvas de Bézier

- La curva de Bézier es una curva paramétrica que, a partir de unos puntos de control, permite al usuario modelar a voluntad.
- Su aplicación inicial fue en el diseño de carrocerías de automóviles, barcos, hélices de barcos, etc.
- Los pioneros fueron Pierre Bézier en Renault y Paul de Faget de Casteljau en Citroën.
- Se puede convertir entre curvas de Bézier cúbicas y splines cúbicas de Hermite.



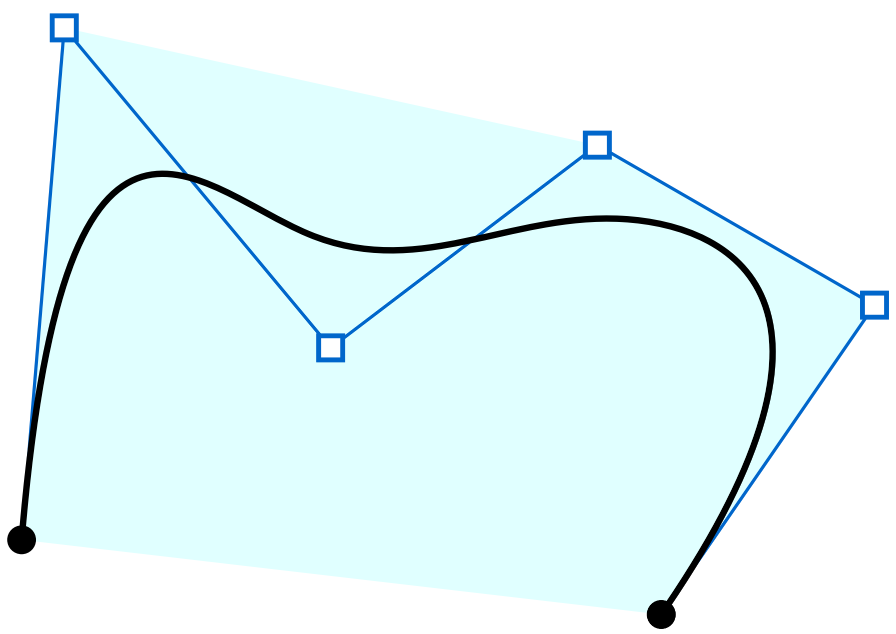


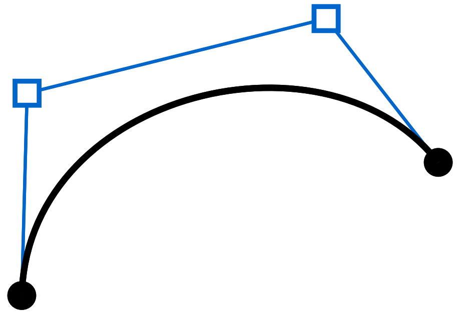


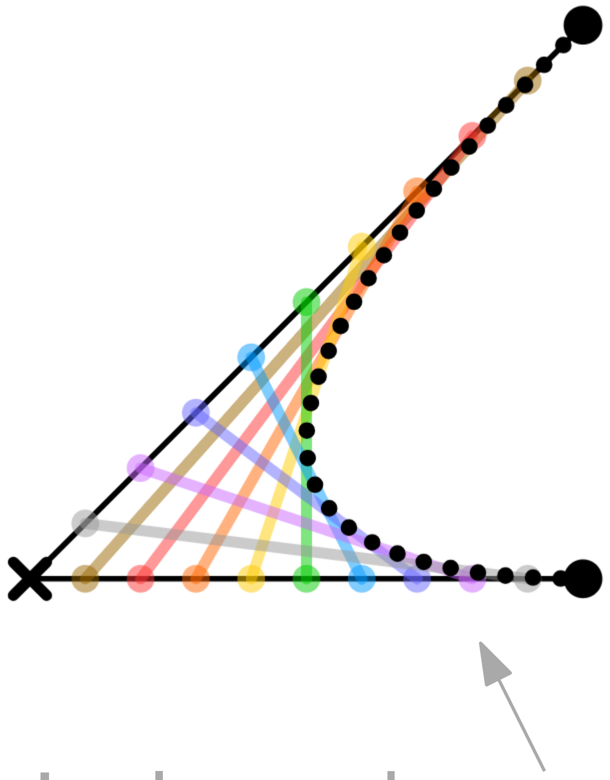


Curvas de Bézier

- La curva de Bézier es una curva paramétrica que, a partir de unos puntos de control, permite al usuario modelar a voluntad.
- Su aplicación inicial fue en el diseño de carrocerías de automóviles, barcos, hélices de barcos, etc.
- Los pioneros fueron Pierre Bézier en Renault y Paul de Faget de Casteljau en Citroën.
- Se puede convertir entre curvas de Bézier cúbicas y splines cúbicas de Hermite.



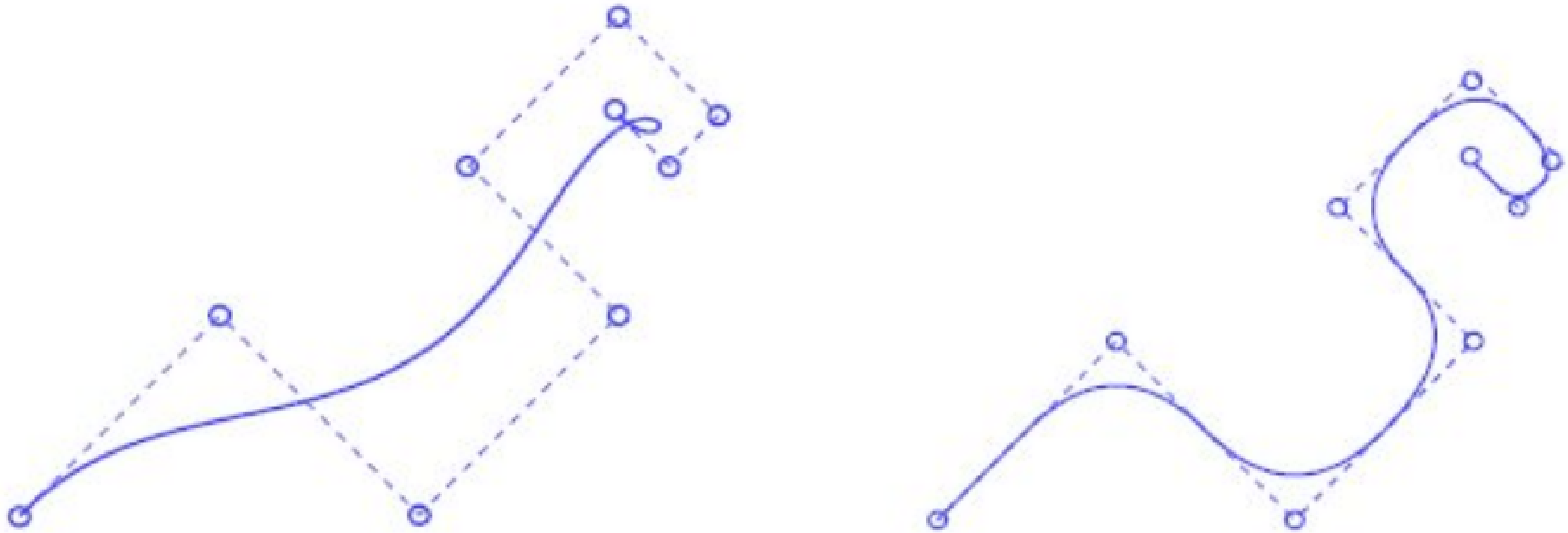




[En Wikipedia, hay ejemplos ilustrativos de curvas de Bézier incluyendo esta.]

B-splines y NURBS

- Las B-splines son la generalización de las curvas de Bézier, que pueden ser generalizadas a NURBS (Non-Uniform Rational B-Splines).



Curva de Bézier y B-spline para puntos de control idénticos

Publi

Curve and Surface Design

Bachelor Degree in Informatics Engineering (GEI)
Departament de Matemàtiques
Facultat d'Informàtica de Barcelona
Universitat Politècnica de Catalunya

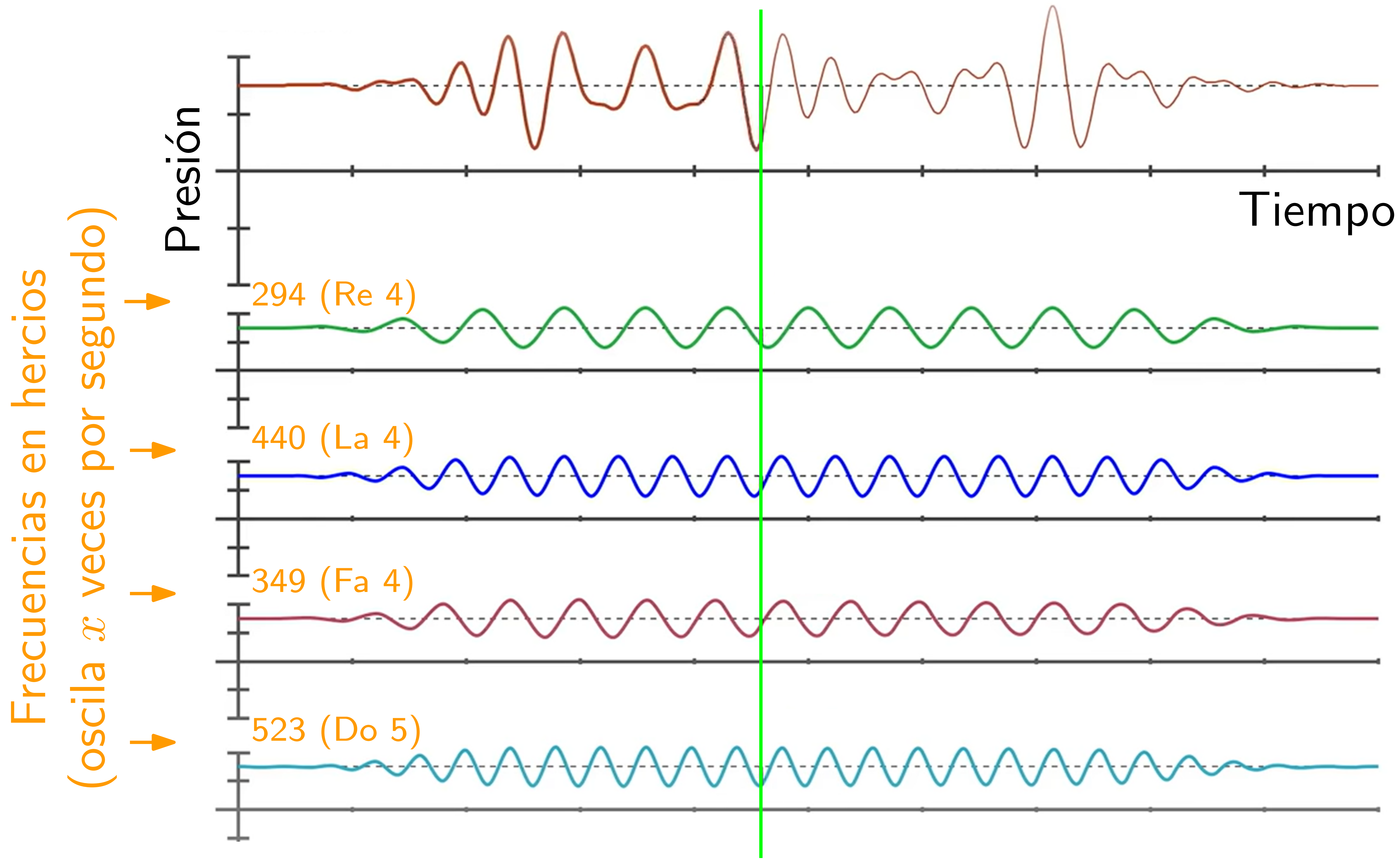
Instructor

- Rodrigo Silveira

Transformada de Fourier

Interpolación trigonométrica

Sonido y presión sonora en el tiempo



Serie de Fourier

Una serie de Fourier es una serie infinita que converge a una función periódica (con periodo T) y continua

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi k}{T}t\right) + b_k \sin\left(\frac{2\pi k}{T}t\right)$$

Serie de Fourier

Una serie de Fourier es una serie infinita que converge a una función periódica (con periodo T) y continua

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi k}{T} t\right) + b_k \sin\left(\frac{2\pi k}{T} t\right) = \sum_{k=-\infty}^{\infty} c_k e^{\frac{2\pi k i}{T} t}$$

con $c_k = \frac{a_k - i b_k}{2}$.

Serie de Fourier

Una serie de Fourier es una serie infinita que converge a una función periódica (con periodo T) y continua

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi k}{T} t\right) + b_k \sin\left(\frac{2\pi k}{T} t\right) = \sum_{k=-\infty}^{\infty} c_k e^{\frac{2\pi k i}{T} t}$$

con $c_k = \frac{a_k - i b_k}{2}$.

Desde una perspectiva aplicada, podemos interpretar la serie de Fourier como:

Serie de Fourier

Una serie de Fourier es una serie infinita que converge a una función periódica (con periodo T) y continua

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi k}{T} t\right) + b_k \sin\left(\frac{2\pi k}{T} t\right) = \sum_{k=-\infty}^{\infty} c_k e^{\frac{2\pi k i}{T} t}$$

con $c_k = \frac{a_k - i b_k}{2}$.

Desde una perspectiva aplicada, podemos interpretar la serie de Fourier como:

- Una descomposición de $f(t)$ ~~en serie de senos y cosenos~~ en frecuencias.

Serie de Fourier

Una serie de Fourier es una serie infinita que converge a una función periódica (con periodo T) y continua

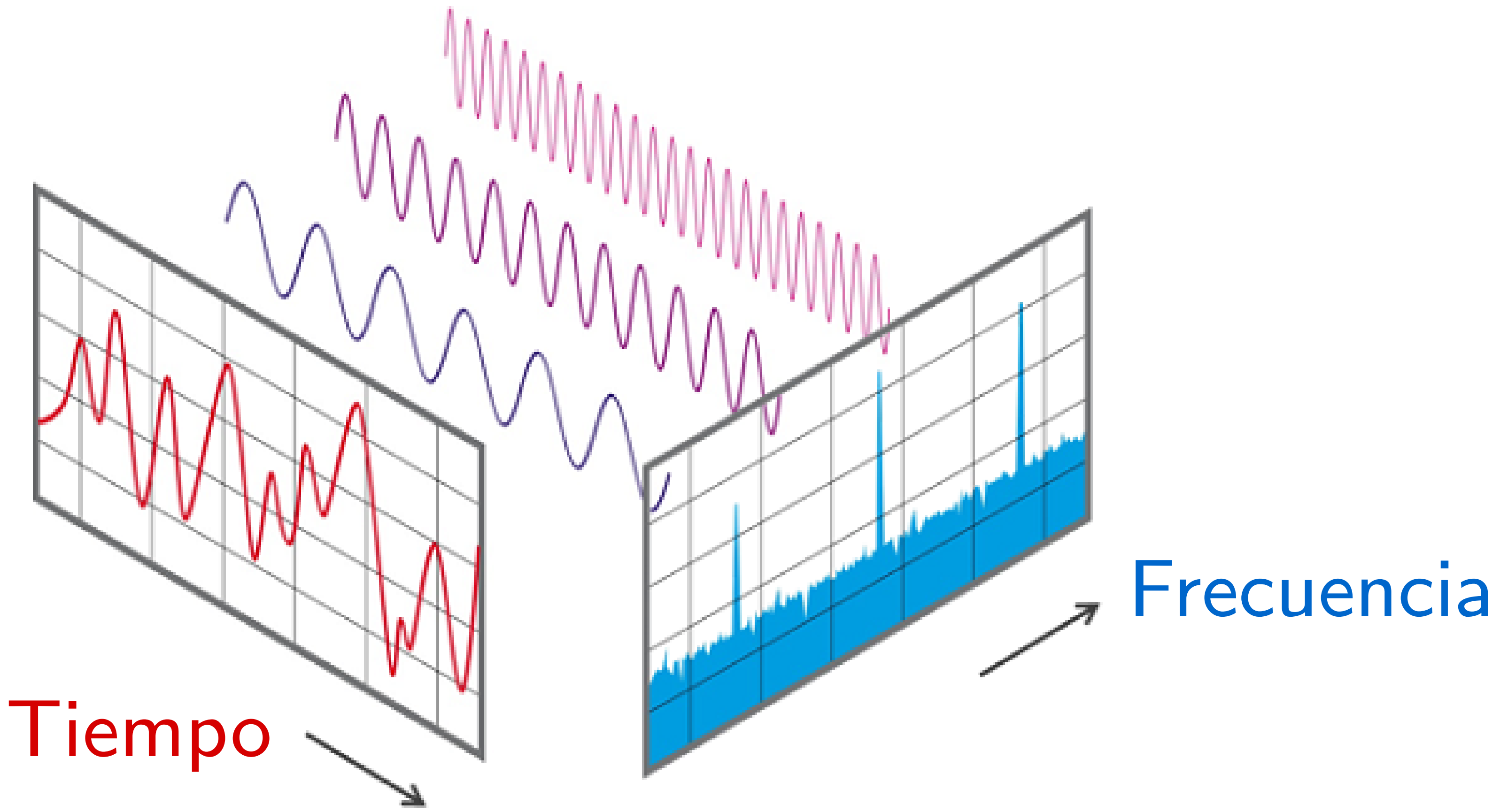
$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} a_k \cos\left(\frac{2\pi k}{T} t\right) + b_k \sin\left(\frac{2\pi k}{T} t\right) = \sum_{k=-\infty}^{\infty} c_k e^{\frac{2\pi k i}{T} t}$$

con $c_k = \frac{a_k - i b_k}{2}$.

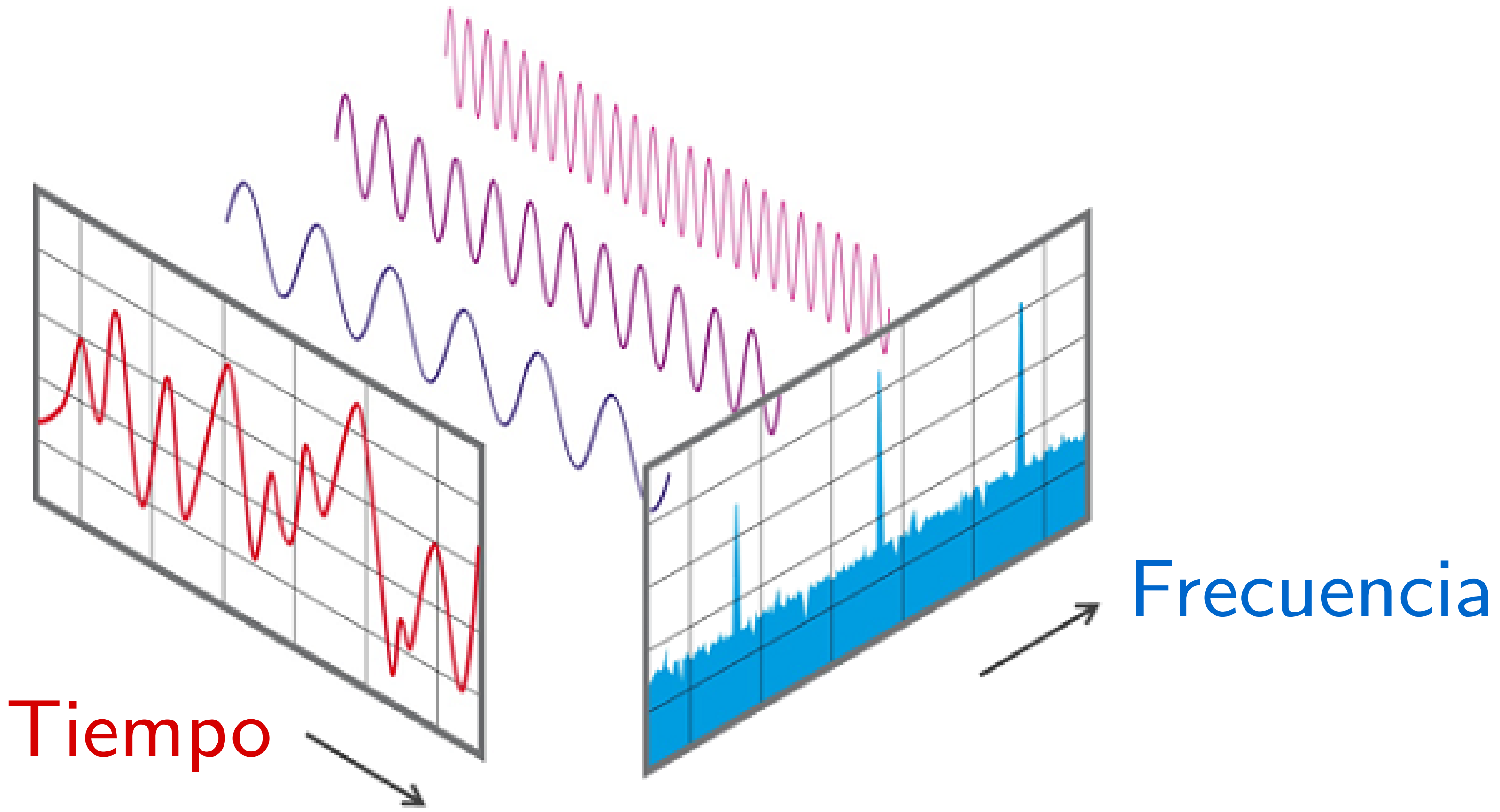
Desde una perspectiva aplicada, podemos interpretar la serie de Fourier como:

- Una descomposición de $f(t)$ ~~en serie de senos y cosenos~~ en frecuencias.
- Como el ajuste trigonométrico de $f(t)$ en serie de senos y cosenos.

Dominio del tiempo y de la frecuencia

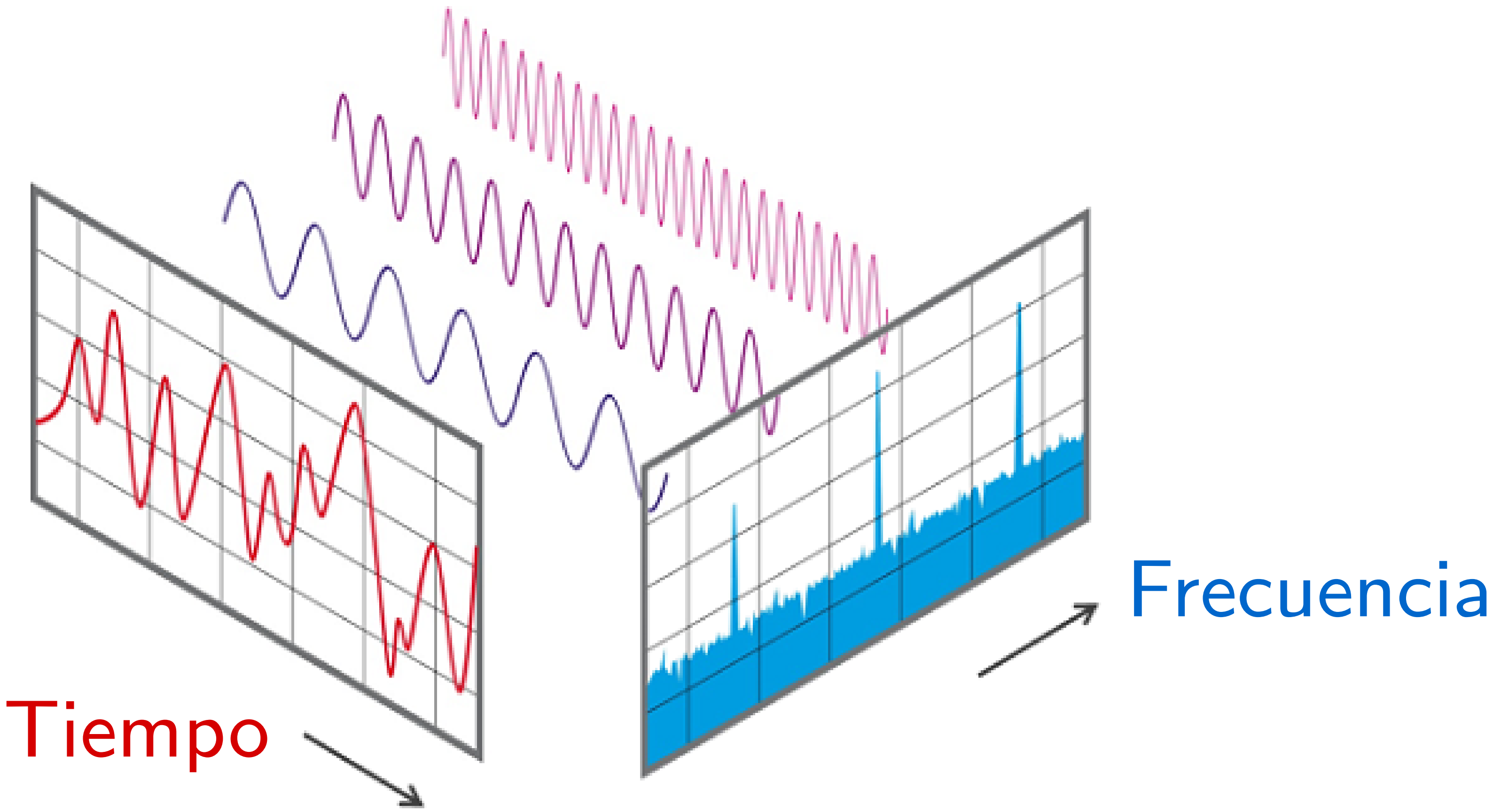


Dominio del tiempo y de la frecuencia



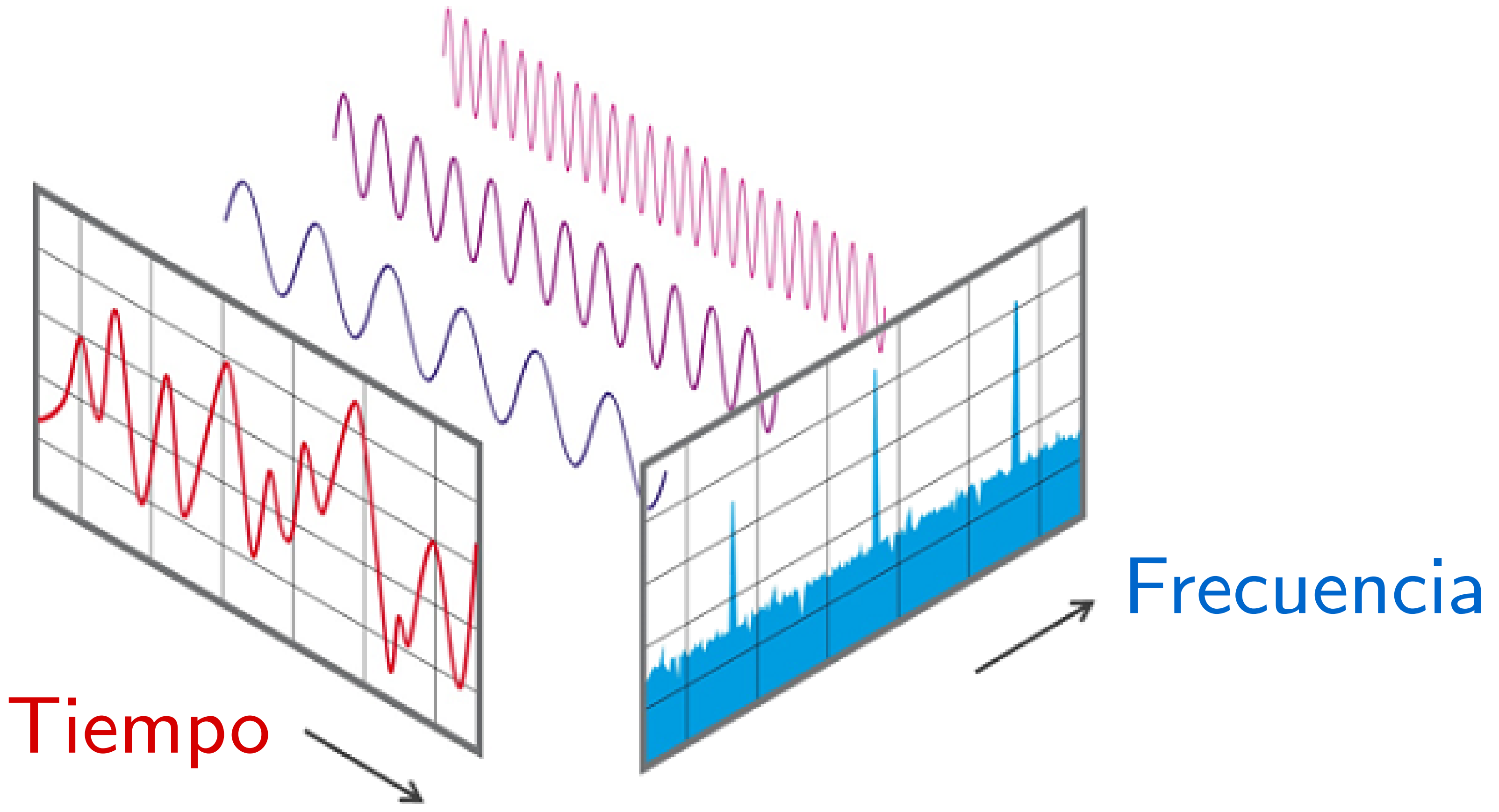
- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.

Dominio del tiempo y de la frecuencia



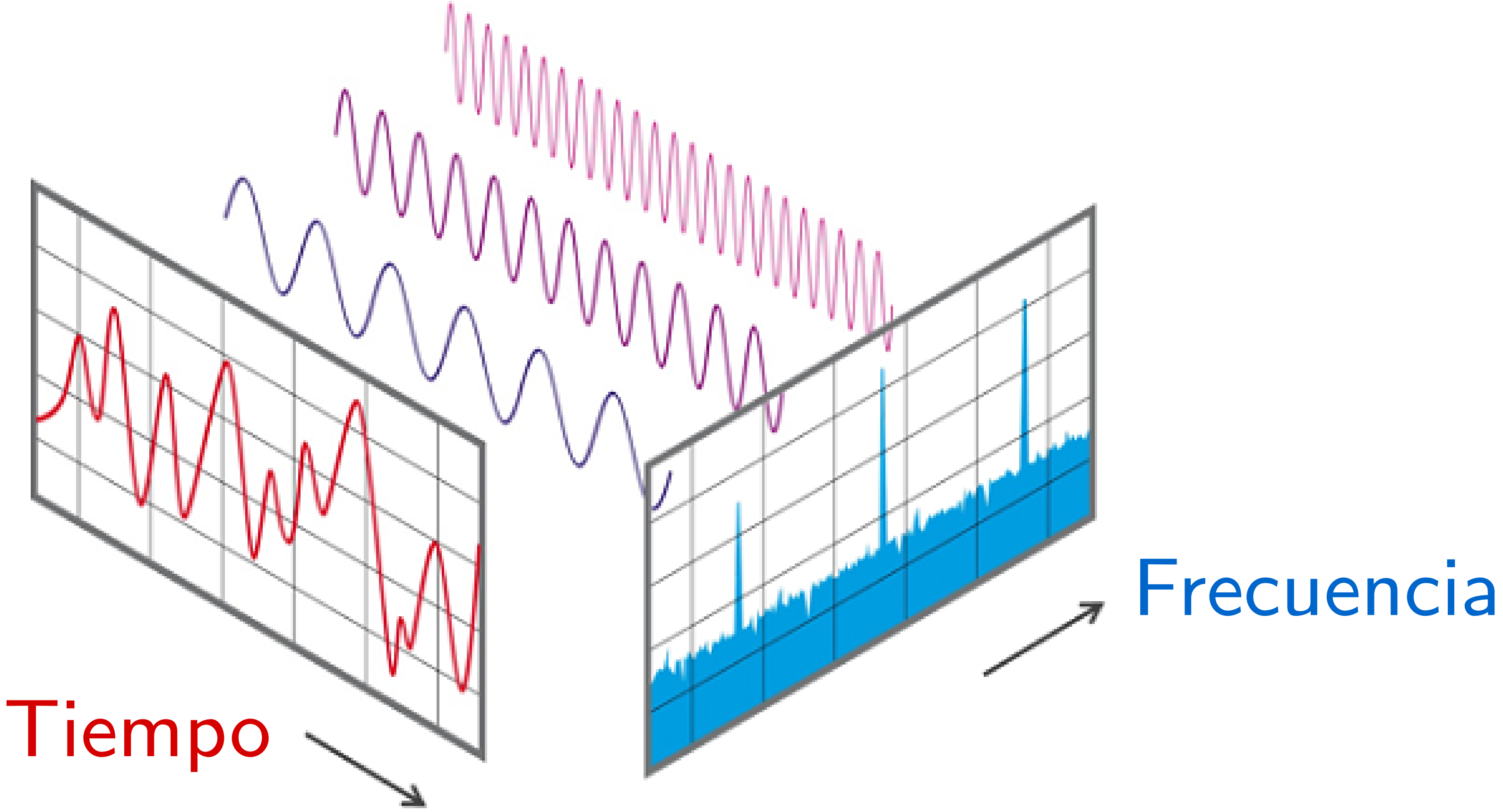
- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.
- El espectro de frecuencias nos muestra la amplitud de cada armónico.

Dominio del tiempo y de la frecuencia



- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.
- El espectro de frecuencias nos muestra la amplitud de cada armónico.
- El el 'peso' de cada coeficiente de la serie se suele medir como $|c_n|^2$.

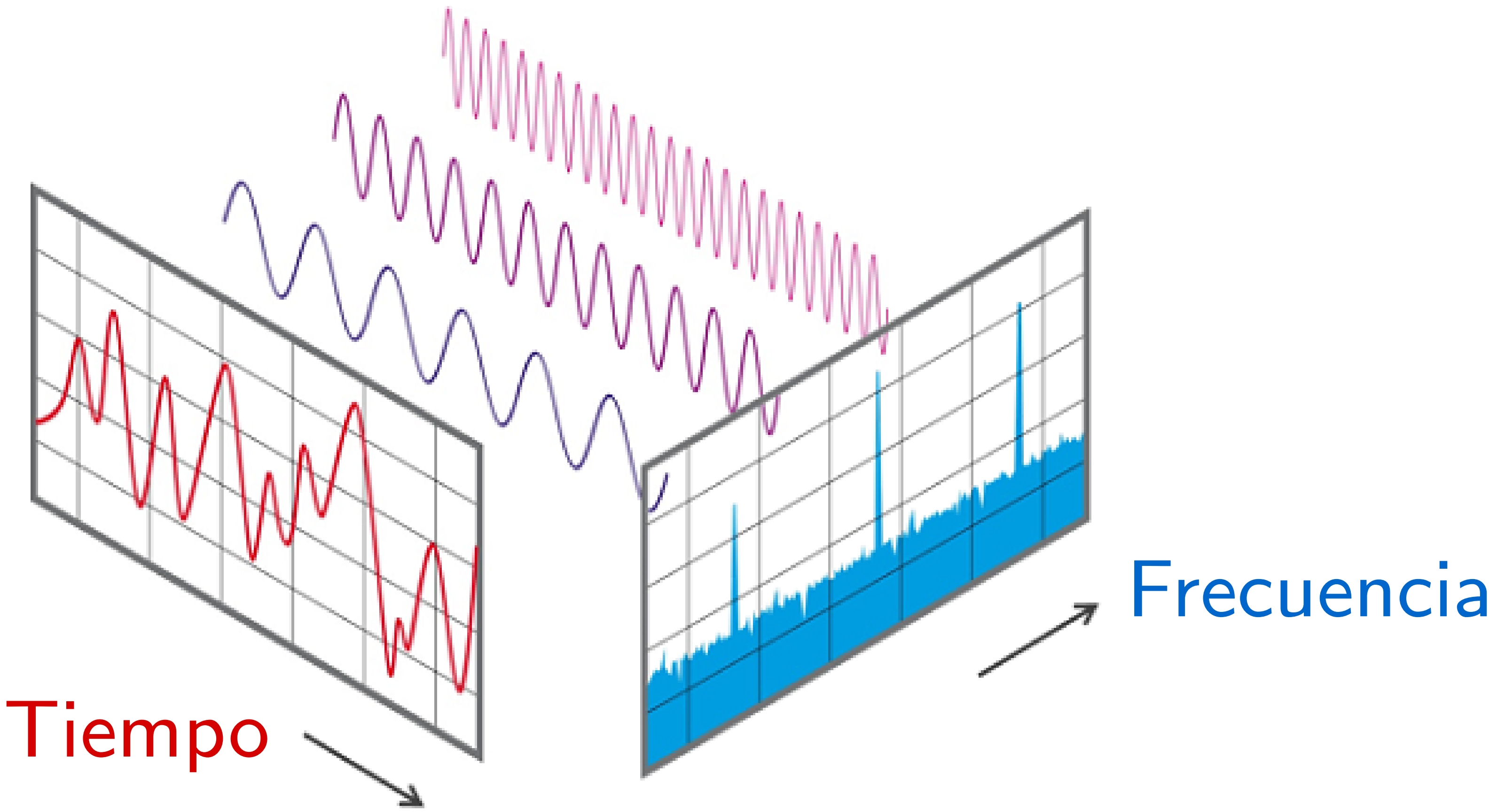
Dominio del tiempo y de la frecuencia



- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.
- El espectro de frecuencias nos muestra la amplitud de cada armónico.
- El el 'peso' de cada coeficiente de la serie se suele medir como $|c_n|^2$.

Applet: https://www.karlsims.com/fft.html

Dominio del tiempo y de la frecuencia

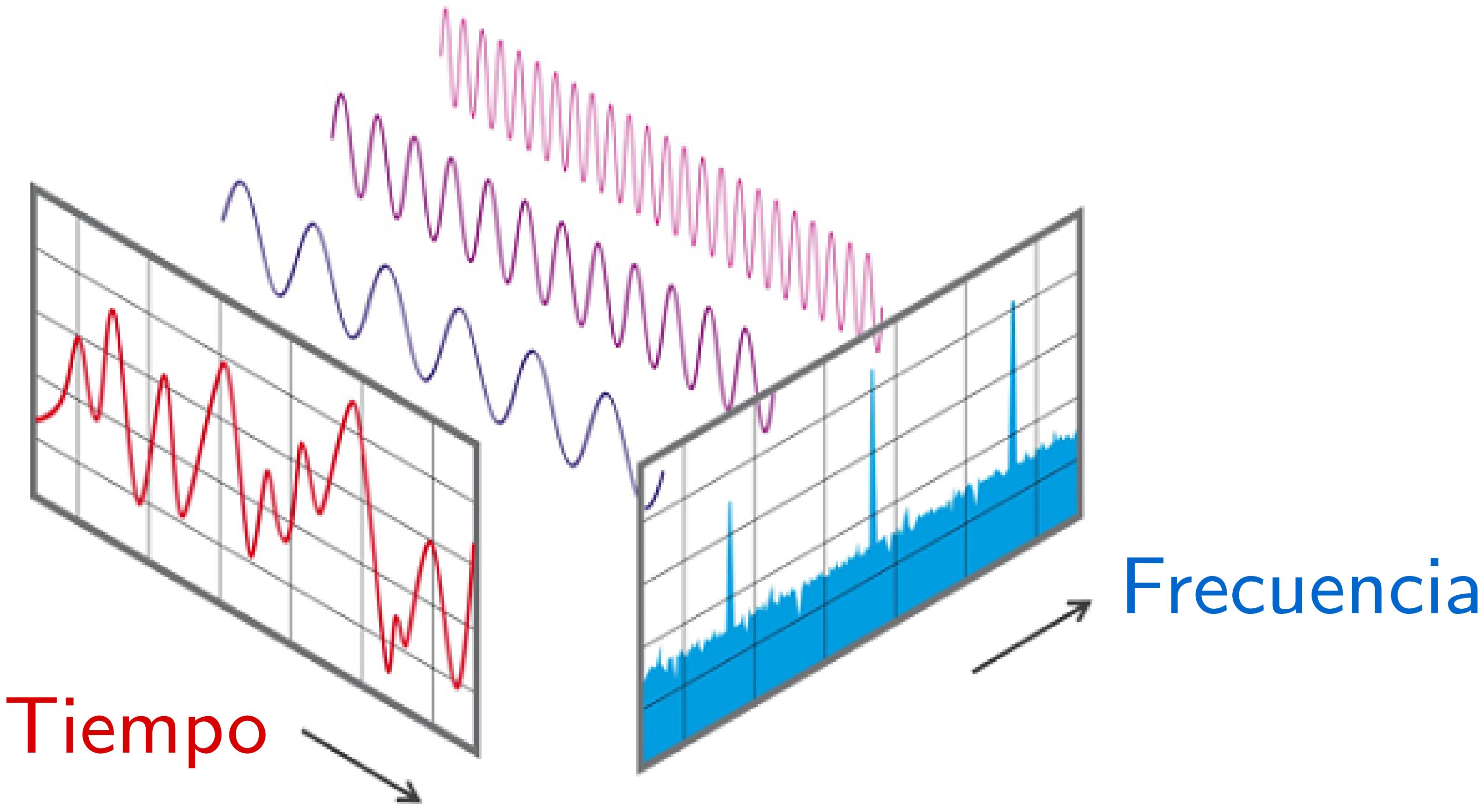


Objetivo: Dada ~~una función $f(t)$~~ una serie de datos, viajar del dominio del tiempo al de la frecuencia (y viceversa).

- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.
- El espectro de frecuencias nos muestra la amplitud de cada armónico.
- El el 'peso' de cada coeficiente de la serie se suele medir como $|c_n|^2$.

Applet: https://www.karlsims.com/fft.html

Dominio del tiempo y de la frecuencia



Objetivo: Dada ~~una función $f(t)$~~ una serie de datos, viajar del dominio del tiempo al de la frecuencia (y viceversa).

¿Cómo? Usando la transformada discreta de Fourier (y su inversa).

- Una función se descompone como una combinación de senos y cosenos de diferentes frecuencias.
- El espectro de frecuencias nos muestra la amplitud de cada armónico.
- El el 'peso' de cada coeficiente de la serie se suele medir como $|c_n|^2$.

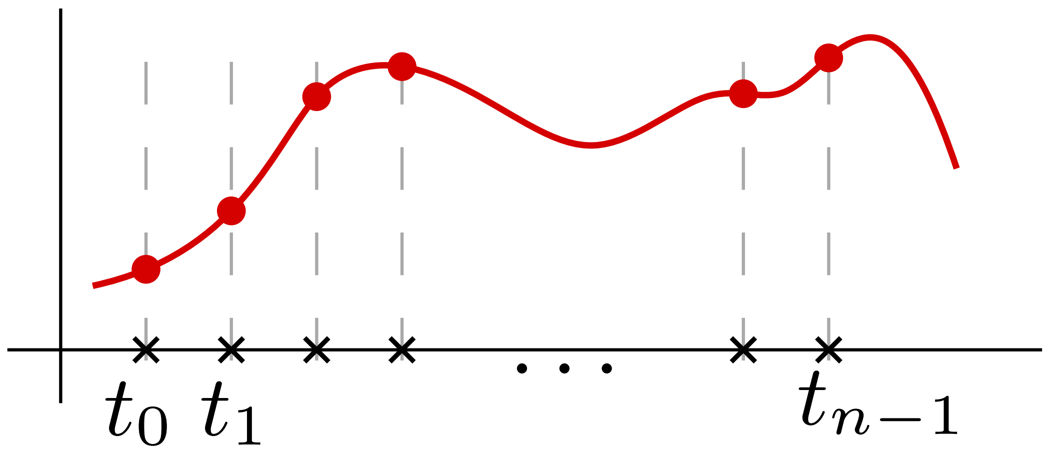
Applet: https://www.karlsims.com/fft.html

Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).

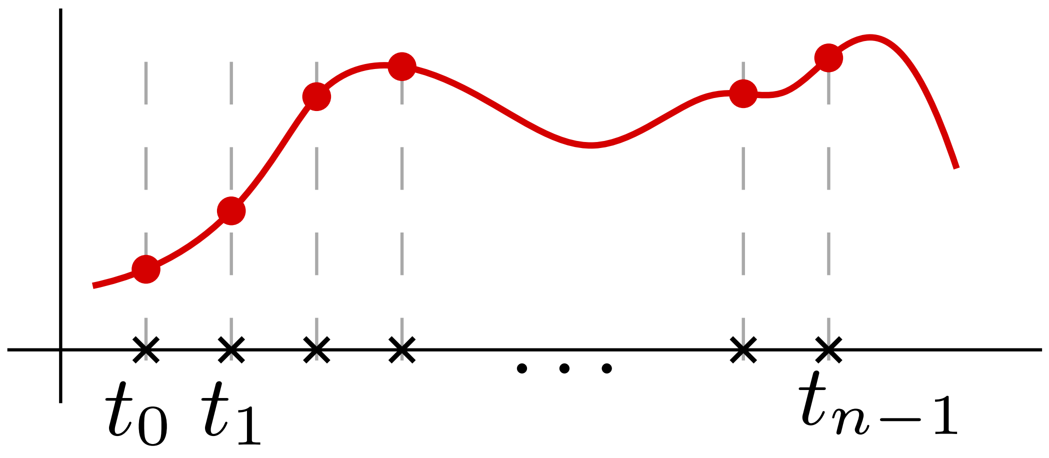
Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).



Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).

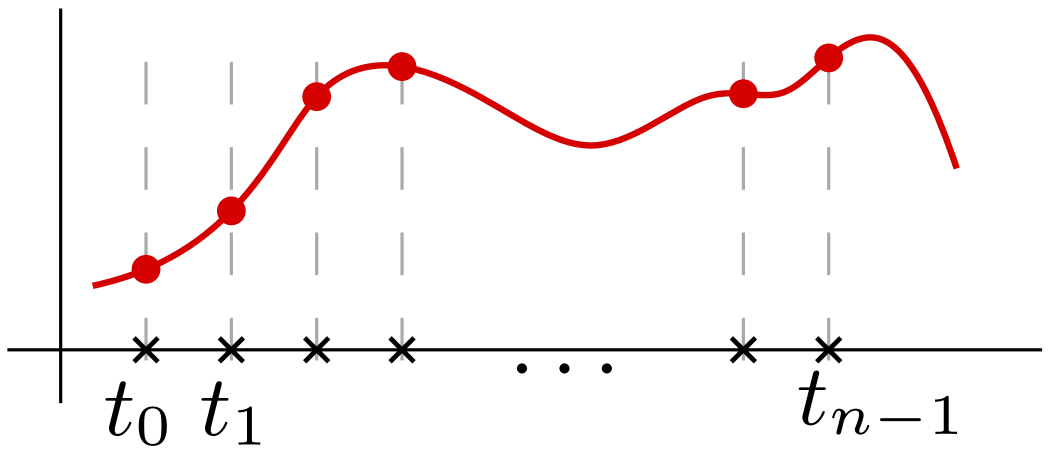


¿Cuántos puntos?

El teorema de Nyquist-Shannon: para reconstruir una señar la tasa de muestreo ha de ser superior al doble de su ancho de banda (rango de frecuencias: max - min).

Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).



¿Cuántos puntos?

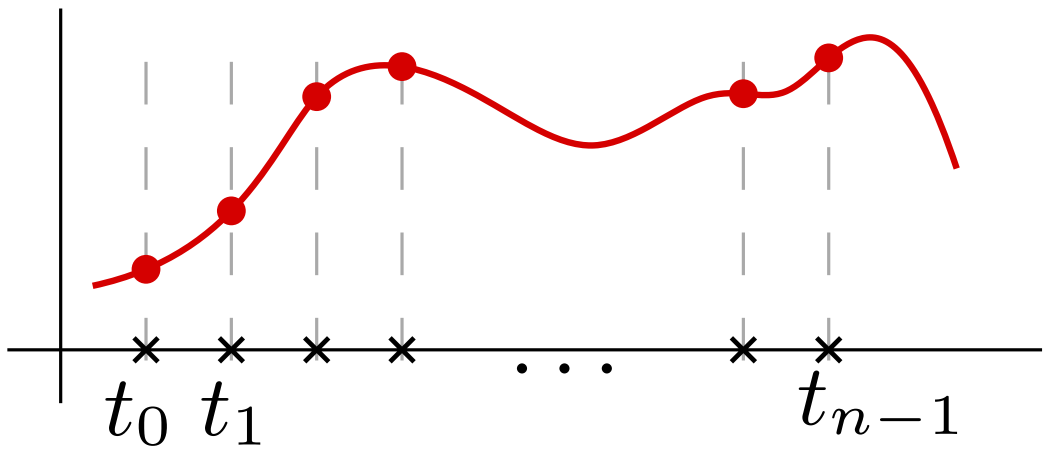
El teorema de Nyquist-Shannon: para reconstruir una señar la tasa de muestreo ha de ser superior al doble de su ancho de banda (rango de frecuencias: max - min).

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}}$$

Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).



¿Cuántos puntos?

El teorema de Nyquist-Shannon: para reconstruir una señar la tasa de muestreo ha de ser superior al doble de su ancho de banda (rango de frecuencias: max - min).

La transformada discreta de Fourier (DFT) se define como

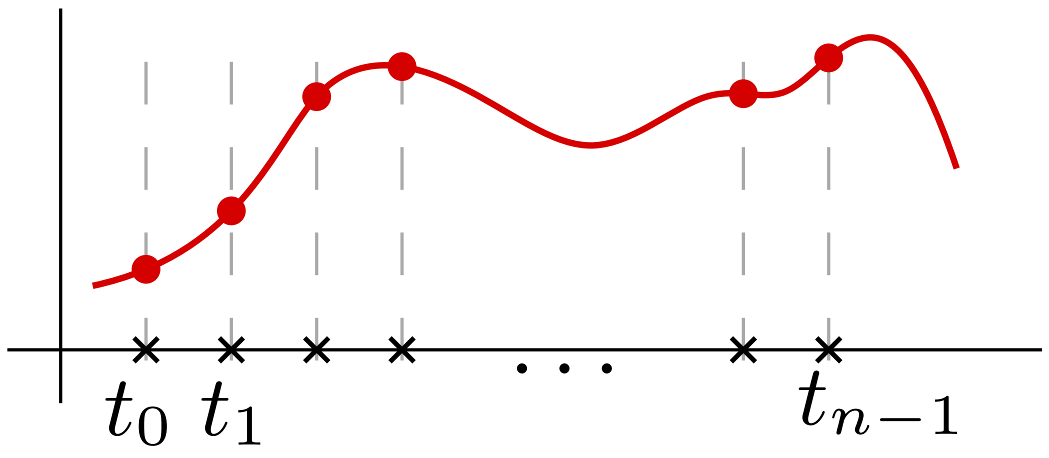
$$\boxed{c_k = \frac{1}{n} \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi k j}{n}}}$$

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi k j}{n}}$$

$$c_k = \frac{1}{T} \int_{t_0}^{t_0+T} f(t) e^{-2\pi i k t/T}\, dt$$

Transformada discreta de Fourier (DFT)

Consideraremos la función T-periódica $f(t)$ evaluada en n puntos equiespaciados en el intervalo $t \in [t_0, T + t_0]$ (de tal forma que $n\Delta t = T$).



¿Cuántos puntos?

El teorema de Nyquist-Shannon: para reconstruir una señar la tasa de muestreo ha de ser superior al doble de su ancho de banda (rango de frecuencias: max - min).

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}}$$

$$f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} \underset{\text{iDFT}}{\overset{\text{DFT}}{\rightleftarrows}} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} & & \\ & \text{Matriz DFT} & \\ & & \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- En general, la frecuencia fundamental es la frecuencia de muestreo dividida entre el número de muestras. Los armónicos son sus múltiplos.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} & & \\ & \text{Matriz DFT} & \\ & & \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \left(\quad \text{Matriz DFT} \quad \right) \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n}\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{pmatrix}\begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{2(n-1)} \\ & & & & \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Cálculo de la DFT

- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{2(n-1)} \\ \vdots & & & & \\ 1 & \xi^{n-1} & \xi^{2(n-1)} & \cdots & \xi^{(n-1)^2} \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi k j}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi k j}{n}}$$

Cálculo de la DFT

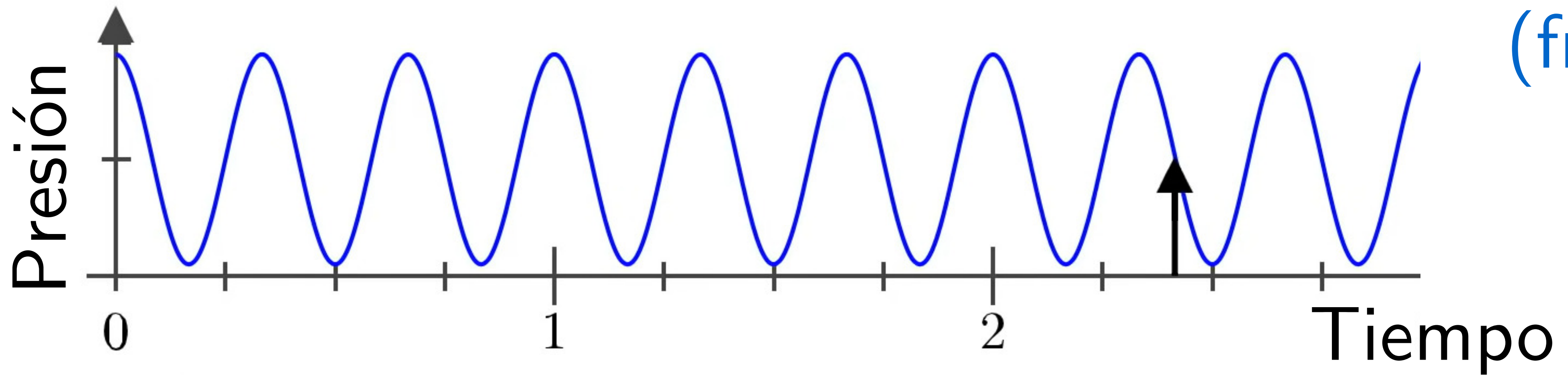
- Definimos $\xi = e^{\frac{-2\pi i}{n}}$.
- Los c_k son complejos. El módulo indica la amplitud; las partes real y compleja dan la fase.

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-1} \end{pmatrix} = \frac{1}{n} \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \xi & \xi^2 & \cdots & \xi^{n-1} \\ 1 & \xi^2 & \xi^4 & \cdots & \xi^{2(n-1)} \\ \vdots & & & & \\ 1 & \xi^{n-1} & \xi^{2(n-1)} & \cdots & \xi^{(n-1)^2} \end{pmatrix} \begin{pmatrix} f(t_0) \\ f(t_1) \\ f(t_2) \\ \vdots \\ f(t_{n-1}) \end{pmatrix}$$

La transformada discreta de Fourier (DFT) se define como

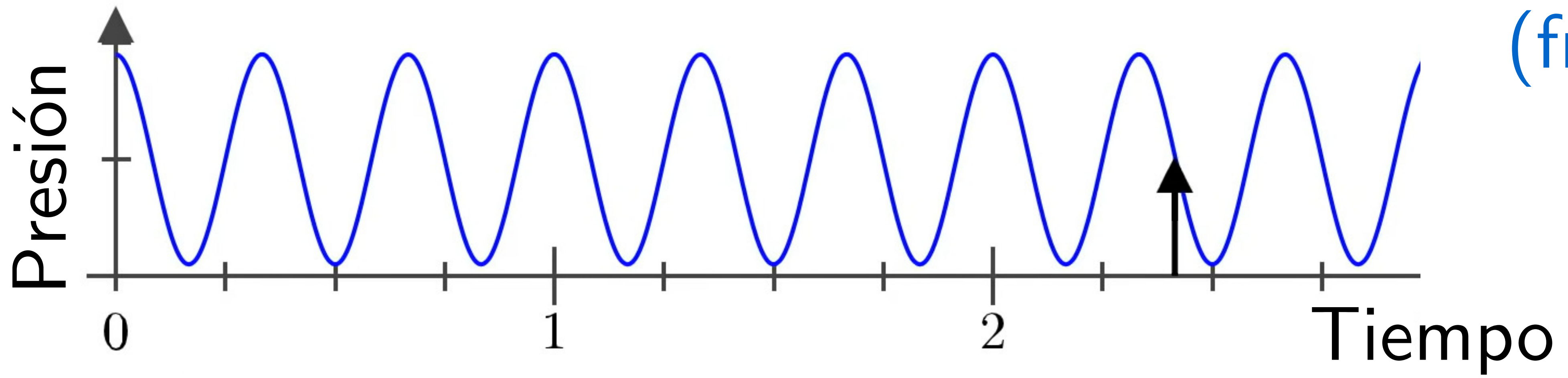
$$\hat{f}_k = n \cdot c_k = \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi kj}{n}} \qquad f(t_k) = \sum_{j=0}^{n-1} c_k e^{i\frac{2\pi kj}{n}}$$

Intuición detrás de la DFT: encontrando la frecuencia



3 pulsos por segundo
(frecuencia $= 3$)

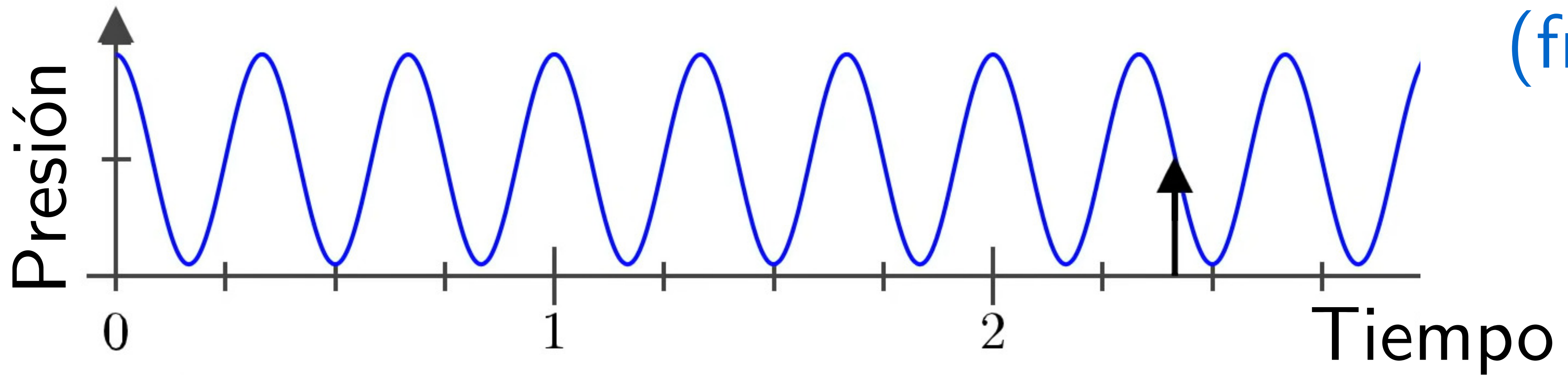
Intuición detrás de la DFT: encontrando la frecuencia



3 pulsos por segundo
(frecuencia $= 3$)

- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.

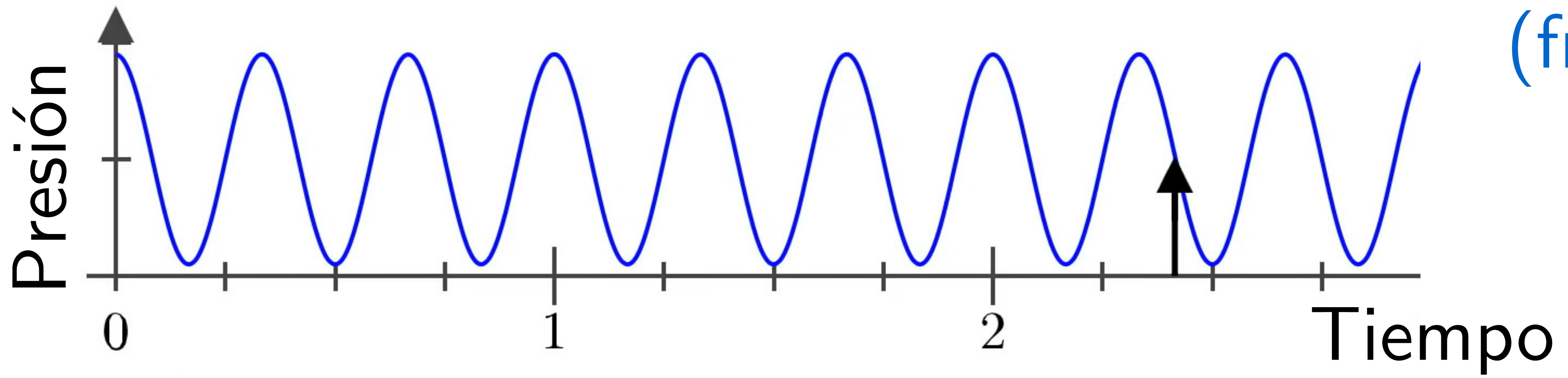
Intuición detrás de la DFT: encontrando la frecuencia



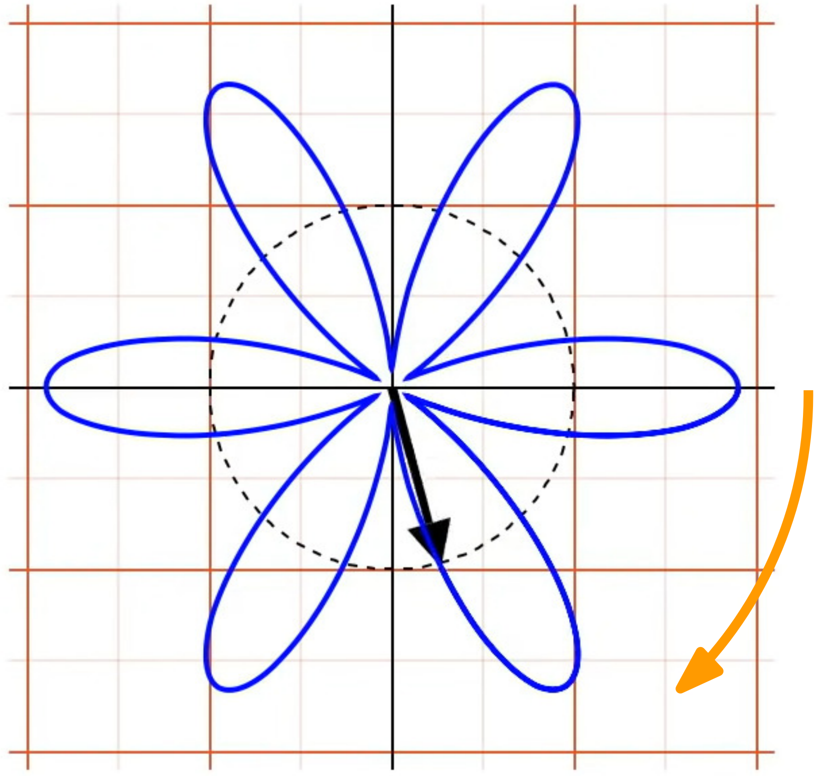
3 pulsos por segundo
(frecuencia $= 3$)

- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 0.5 ciclos por segundo.

Intuición detrás de la DFT: encontrando la frecuencia

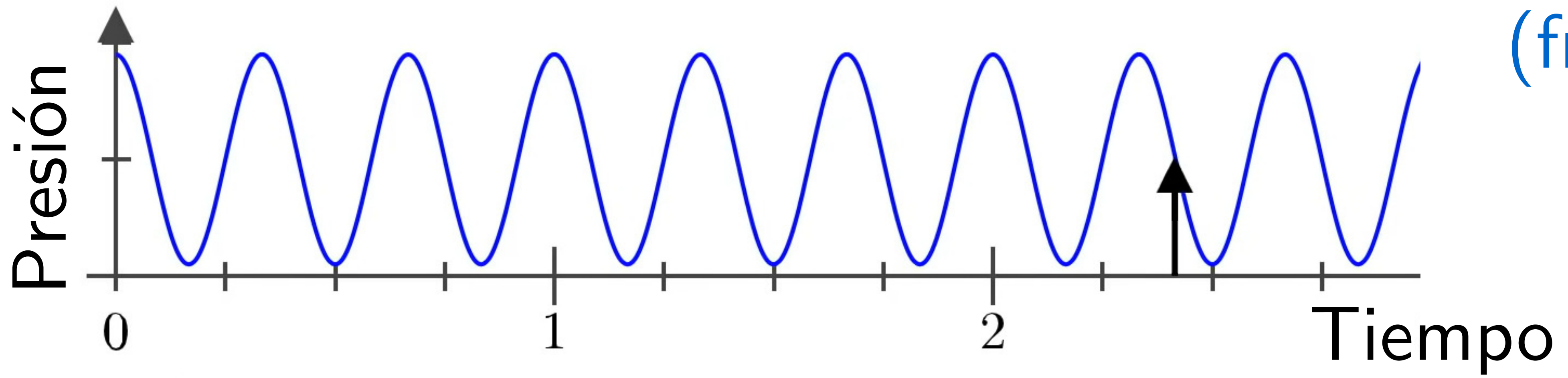


3 pulsos por segundo
(frecuencia $= 3$)

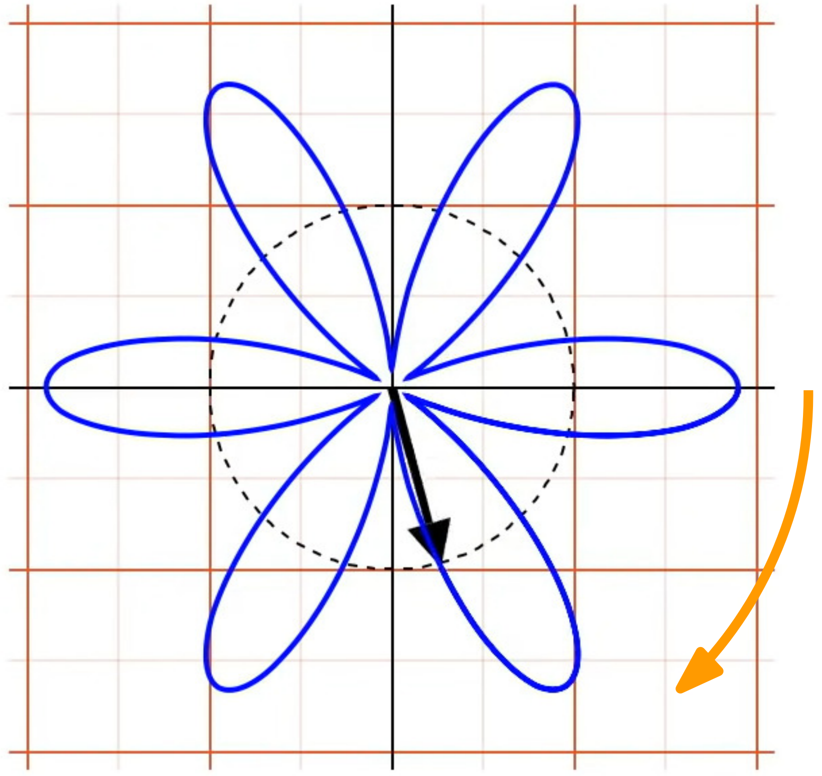


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 0.5 ciclos por segundo.

Intuición detrás de la DFT: encontrando la frecuencia

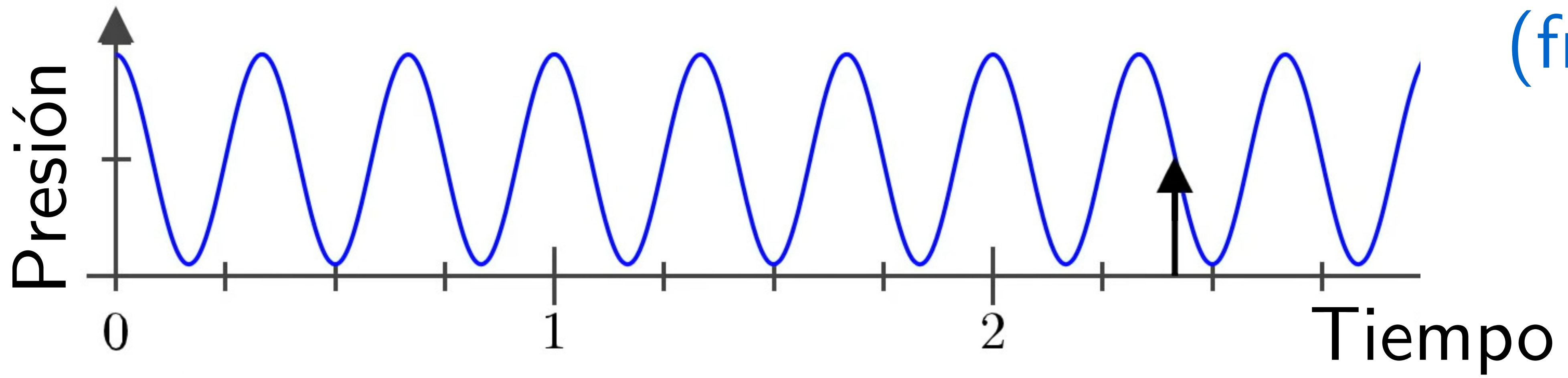


3 pulsos por segundo
(frecuencia $= 3$)

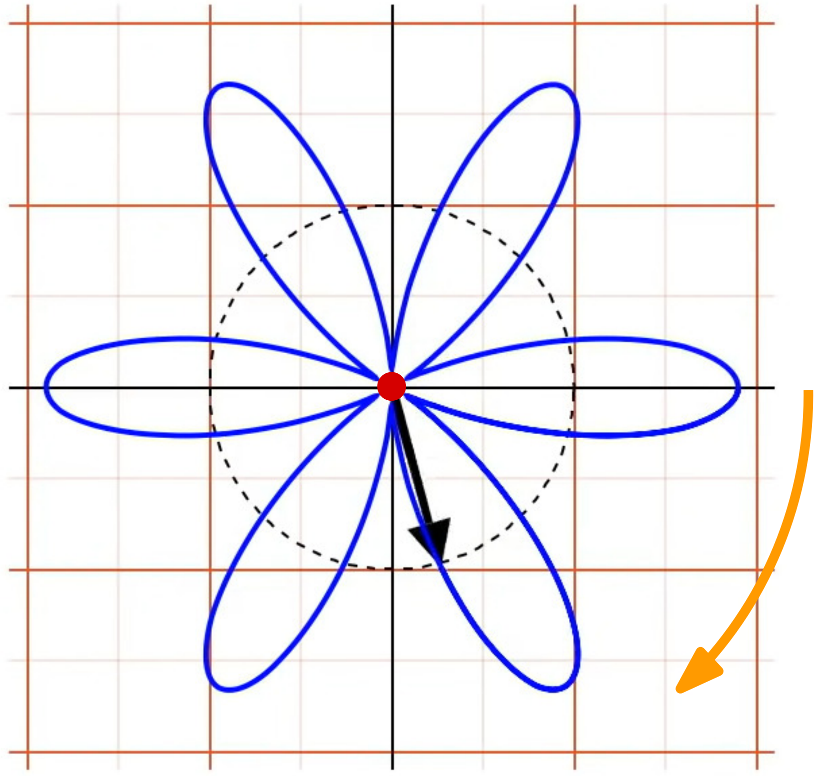


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 0.5 ciclos por segundo.
- Observamos dónde está el centro de masa.

Intuición detrás de la DFT: encontrando la frecuencia

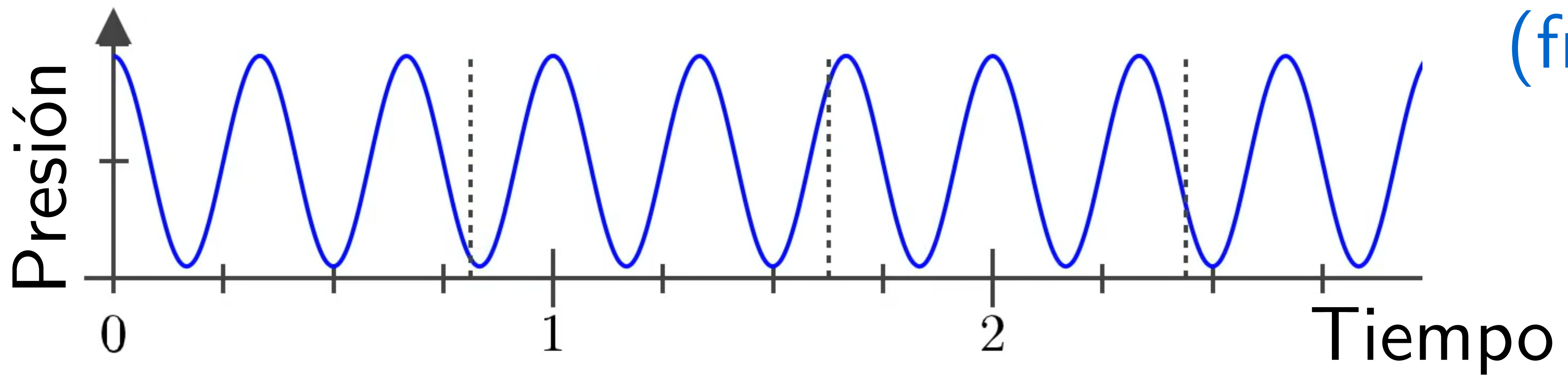


3 pulsos por segundo
(frecuencia $= 3$)

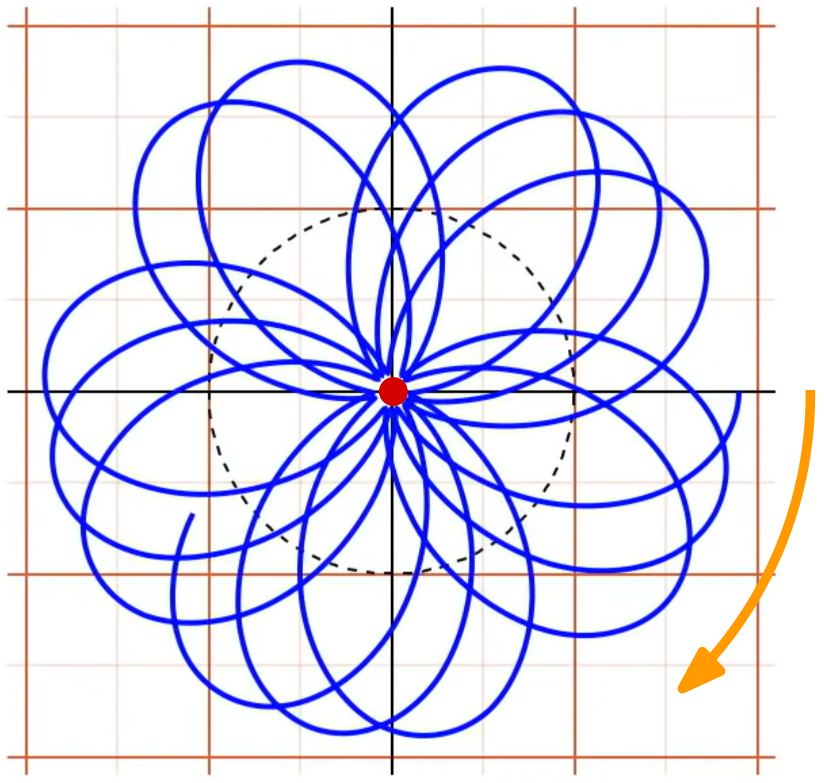


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 0.5 ciclos por segundo.
- Observamos dónde está el centro de masa.

Intuición detrás de la DFT: encontrando la frecuencia

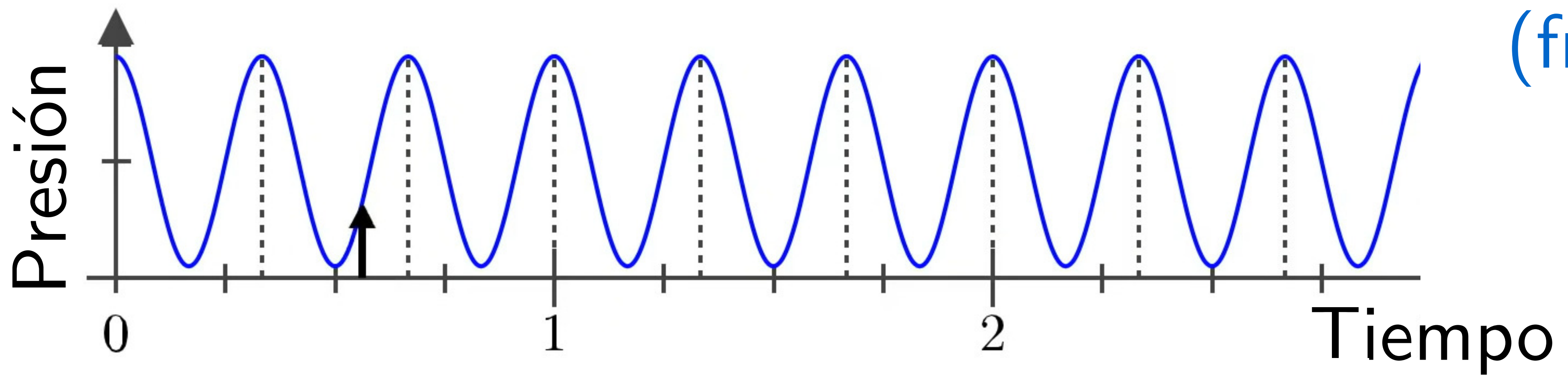


3 pulsos por segundo
(frecuencia $= 3$)

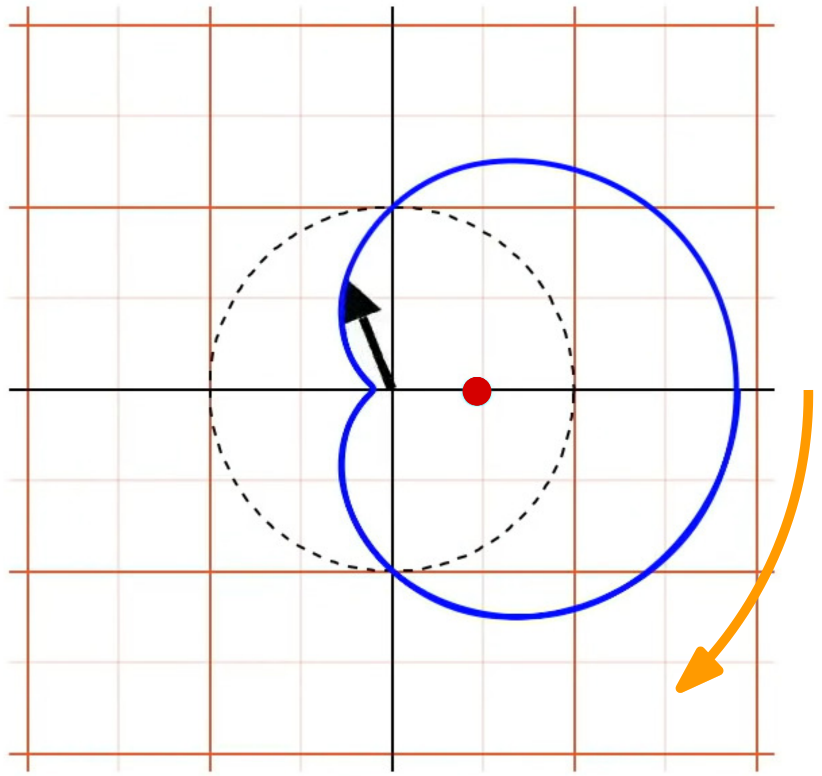


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 1.23 ciclos por segundo.
- Observamos dónde está el centro de masa.

Intuición detrás de la DFT: encontrando la frecuencia



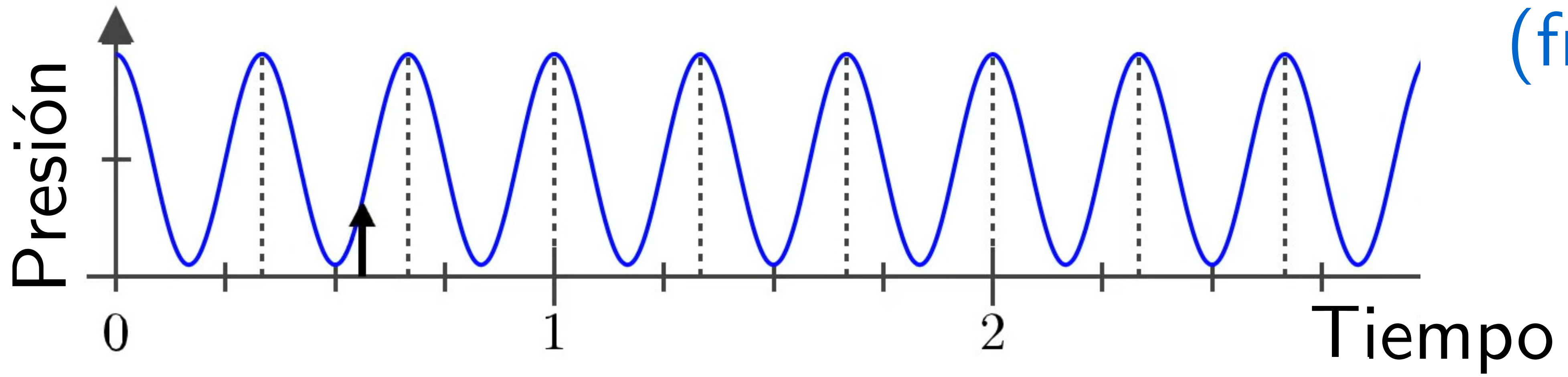
3 pulsos por segundo
(frecuencia $= 3$)

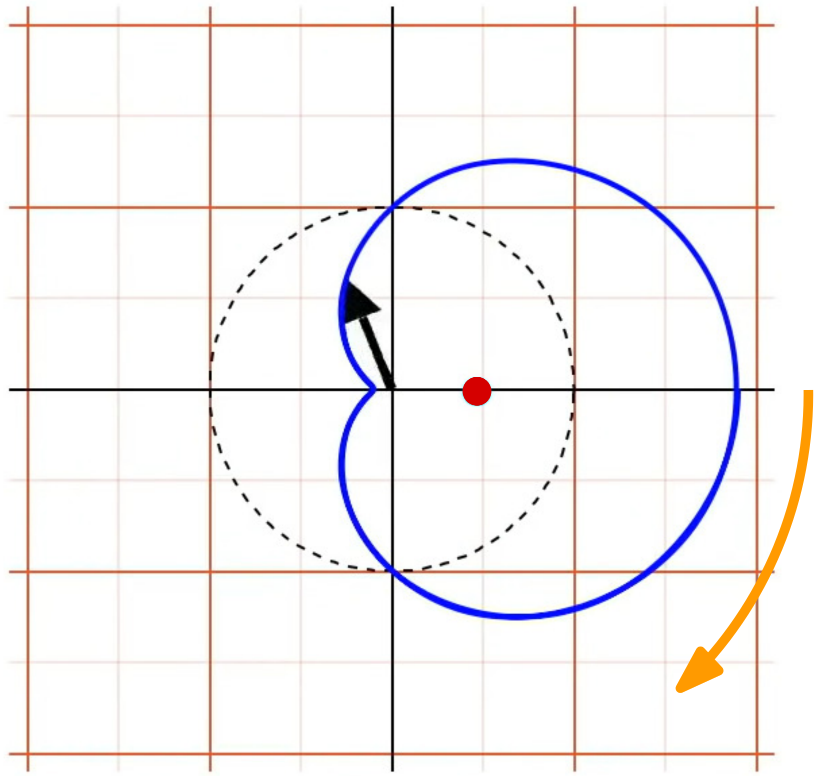


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 3 ciclos por segundo.
- Observamos dónde está el centro de masa.

Intuición detrás de la DFT: encontrando la frecuencia

3 pulsos por segundo
(frecuencia $= 3$)

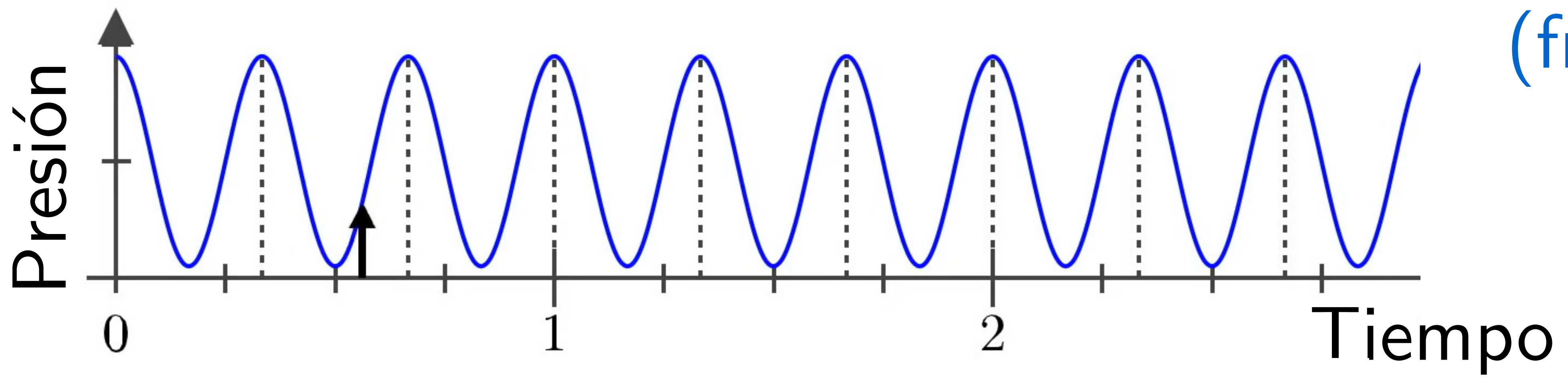




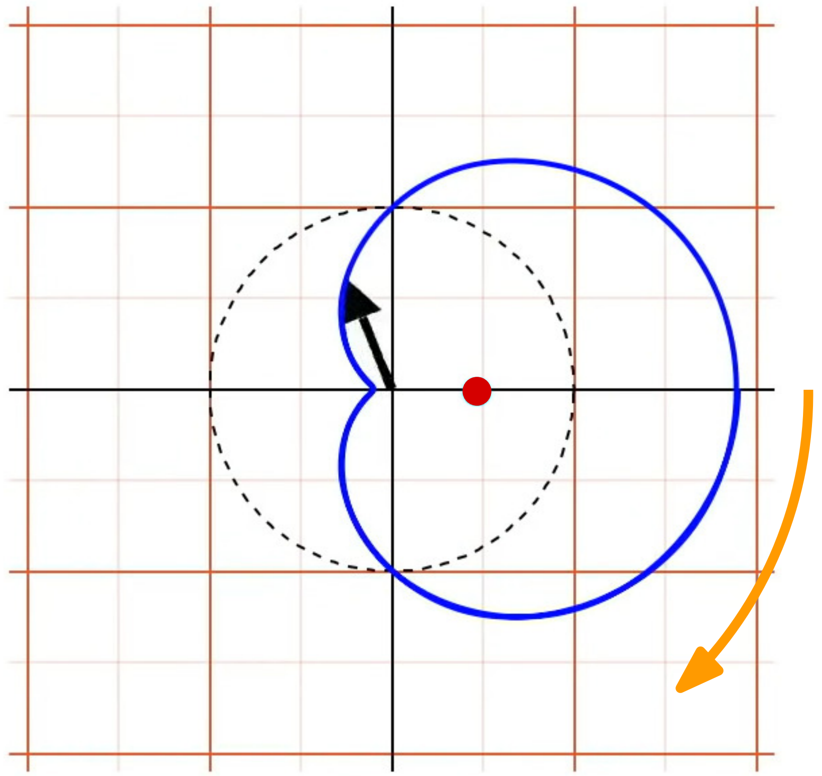
- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 3 ciclos por segundo.
- Observamos dónde está el centro de masa.

La distancia al origen del centro de masas se aleja de cero solo en la frecuencia correcta!

Intuición detrás de la DFT: encontrando la frecuencia



3 pulsos por segundo
(frecuencia $= 3$)

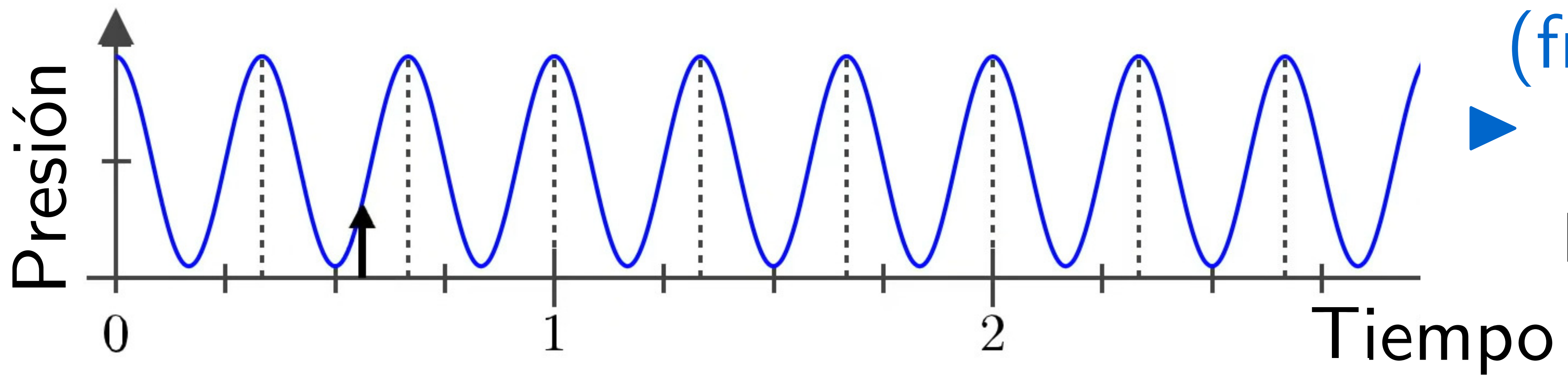


- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 3 ciclos por segundo.
- Observamos dónde está el centro de masa.

La distancia al origen del centro de masas se aleja de cero solo en la frecuencia correcta!

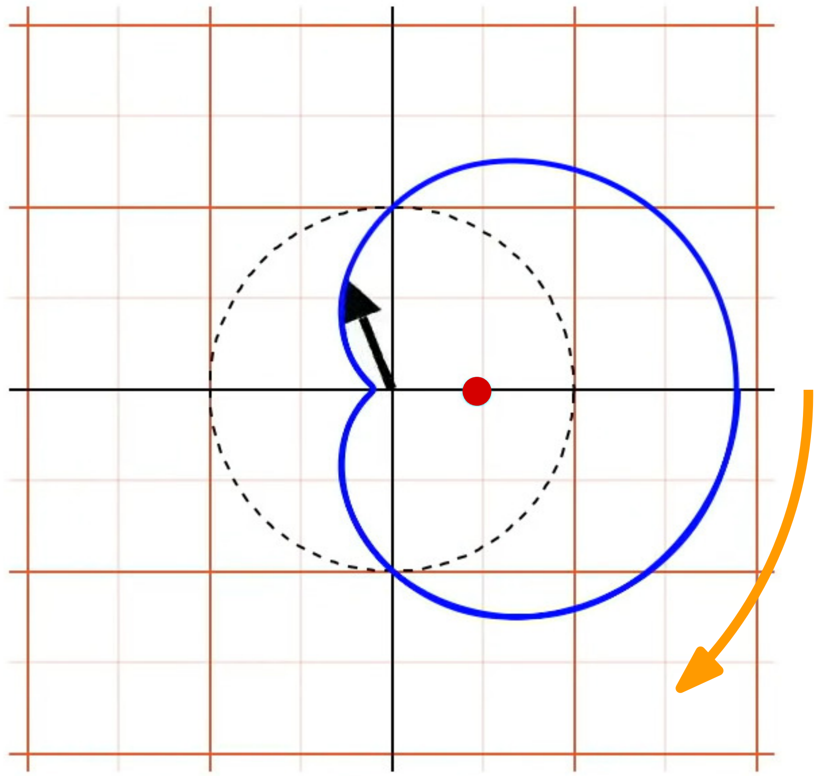
- Casi la DFT! (en la DFT los valores $\hat{f}_i$ están escalados: más prominente).

Intuición detrás de la DFT: encontrando la frecuencia



3 pulsos por segundo
(frecuencia $= 3$)

- Hemos considerado solo presión positiva para hacerlo más claro.

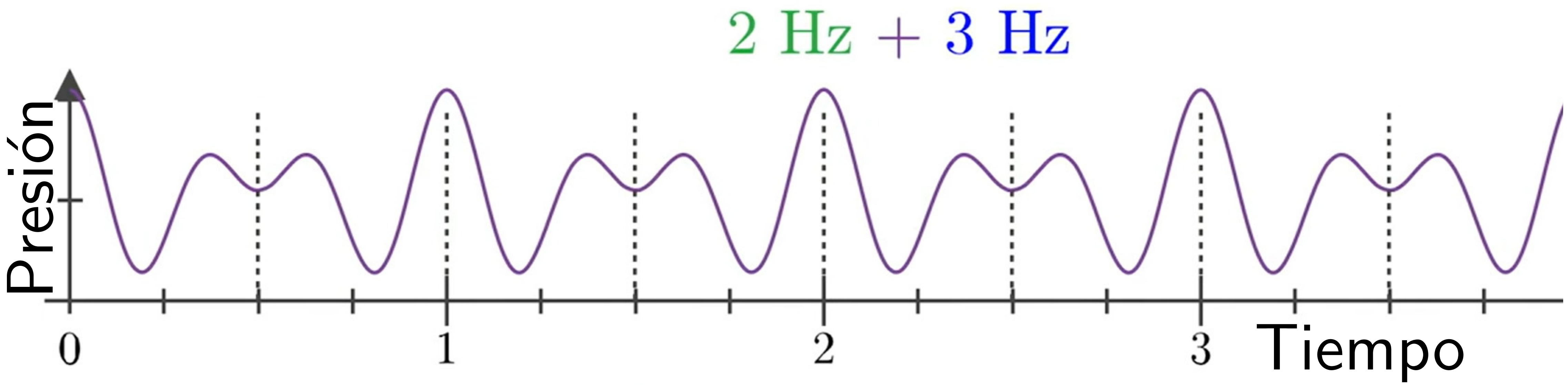


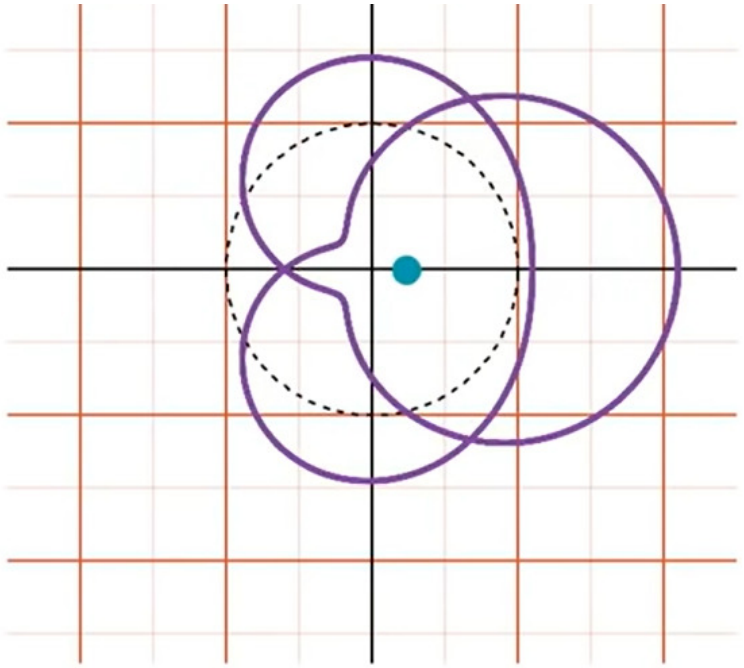
- Seleccionamos cierta frecuencia y queremos ver si es la frecuencia correcta de la señal.
- Probamos con 3 ciclos por segundo.
- Observamos dónde está el centro de masa.

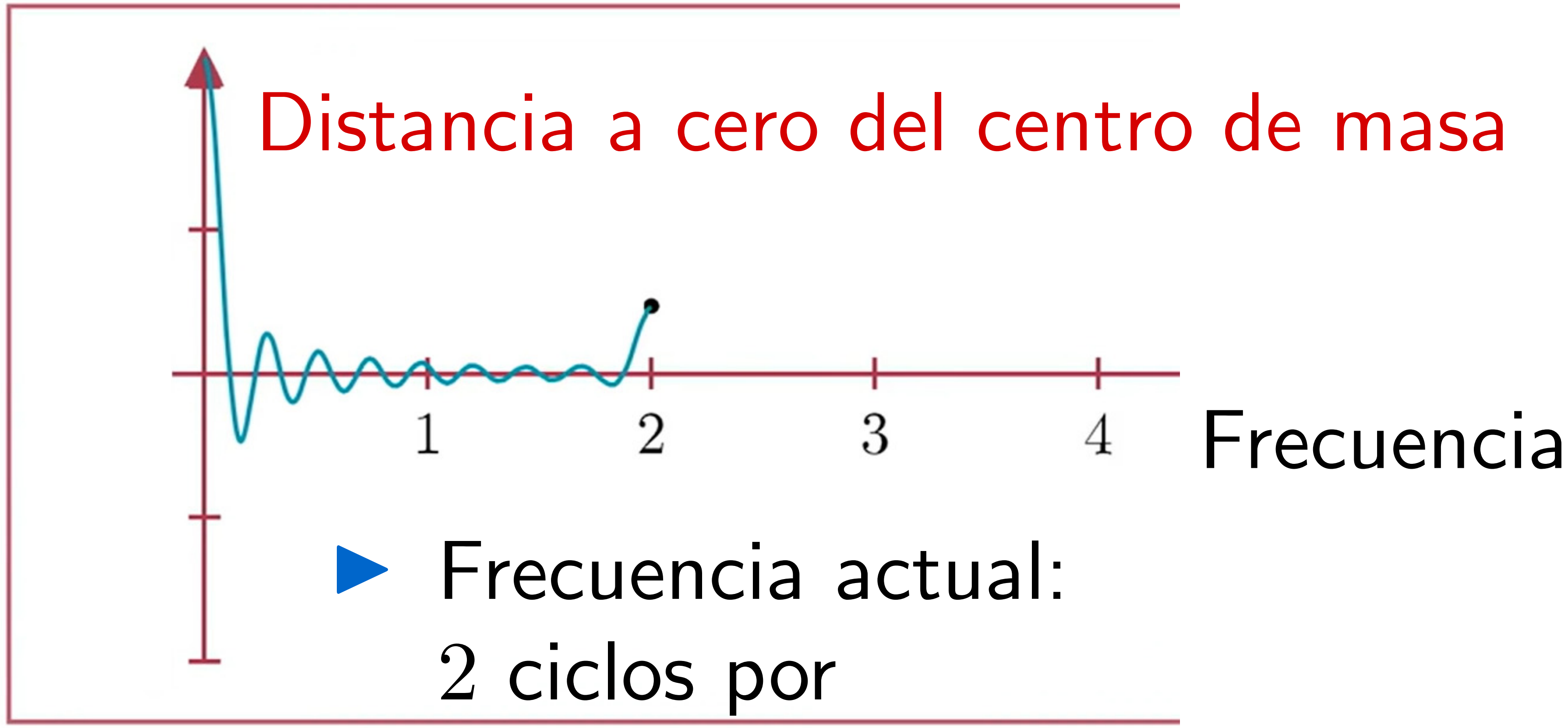
La distancia al origen del centro de masas se aleja de cero solo en la frecuencia correcta!

- Casi la DFT! (en la DFT los valores $\hat{f}_i$ están escalados: más prominente).

Intuición detrás de la DFT: todos los armónicos

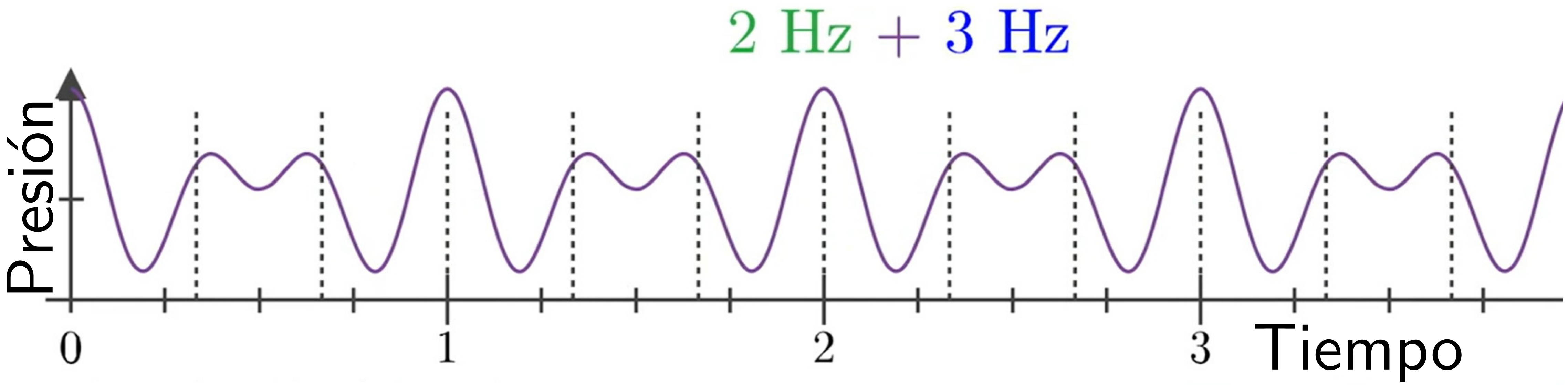


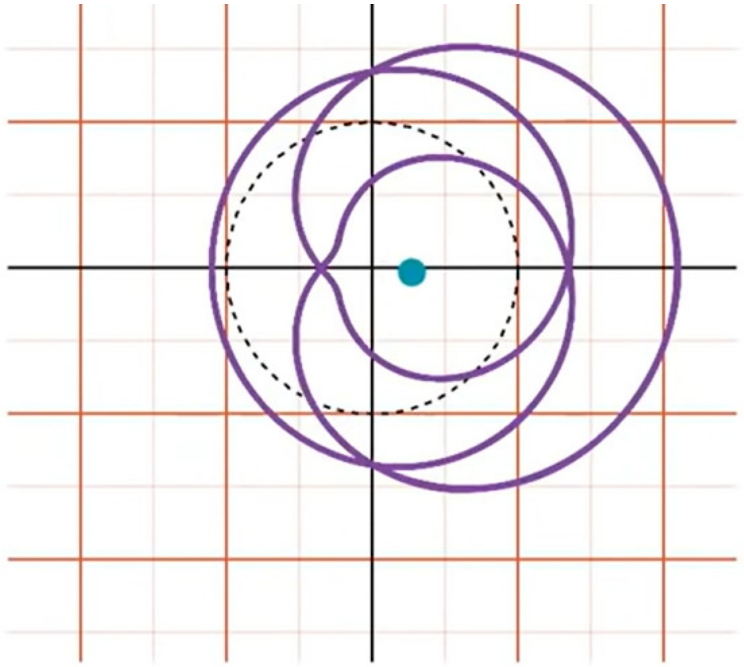


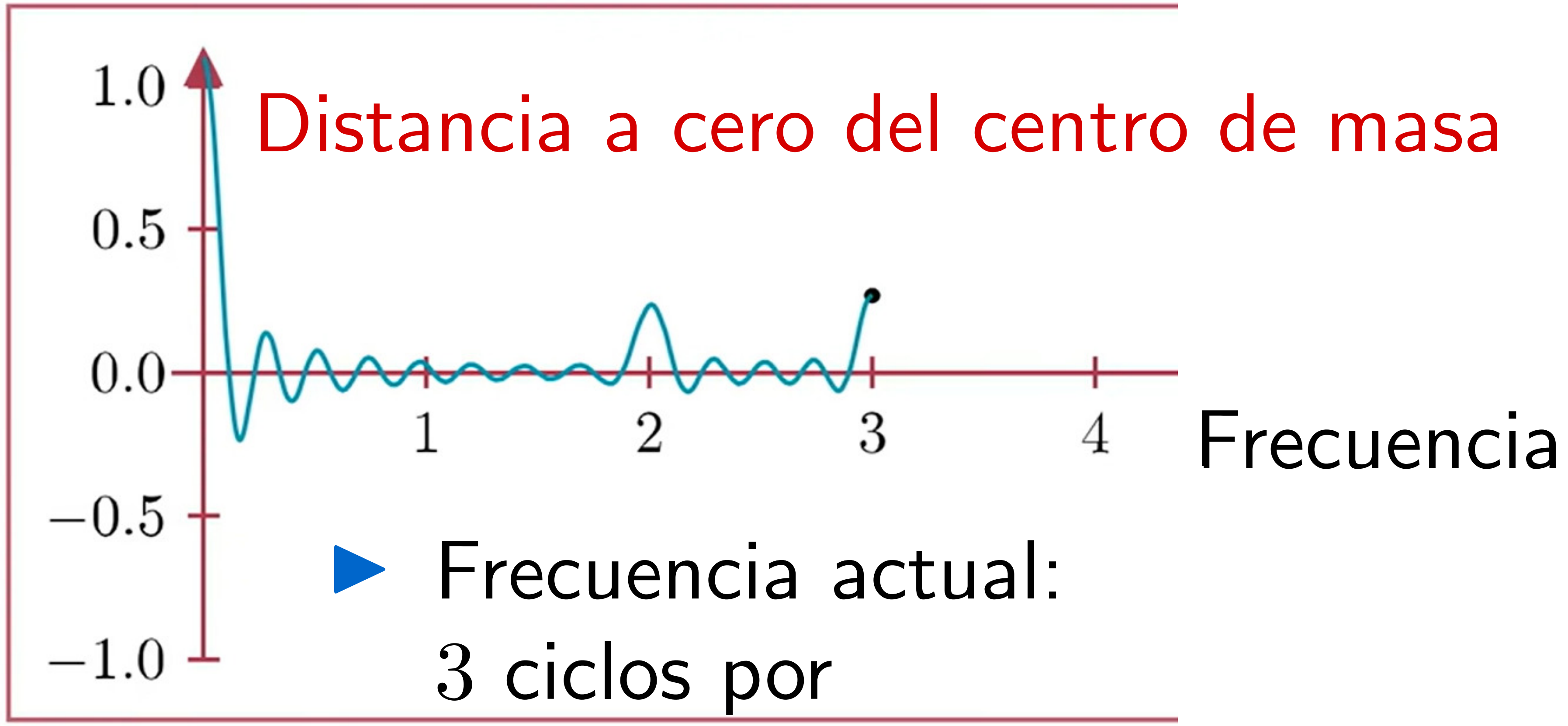


- Frecuencia actual: 2 ciclos por segundo

Intuición detrás de la DFT: todos los armónicos

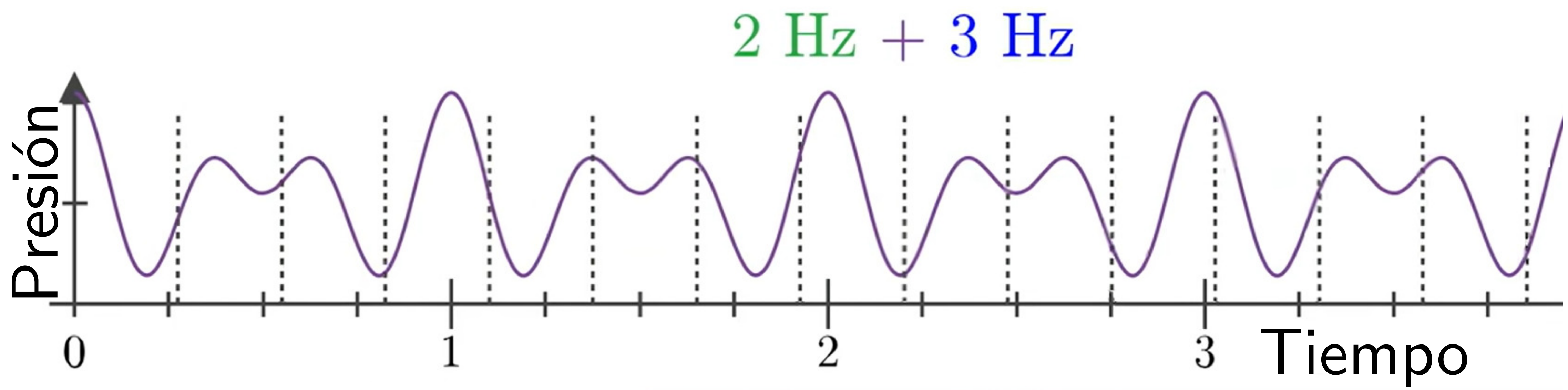


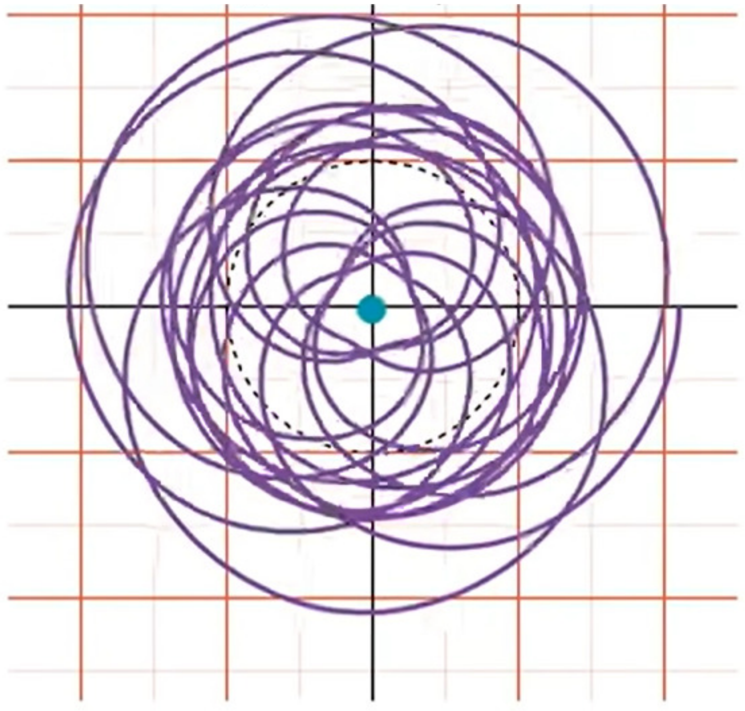


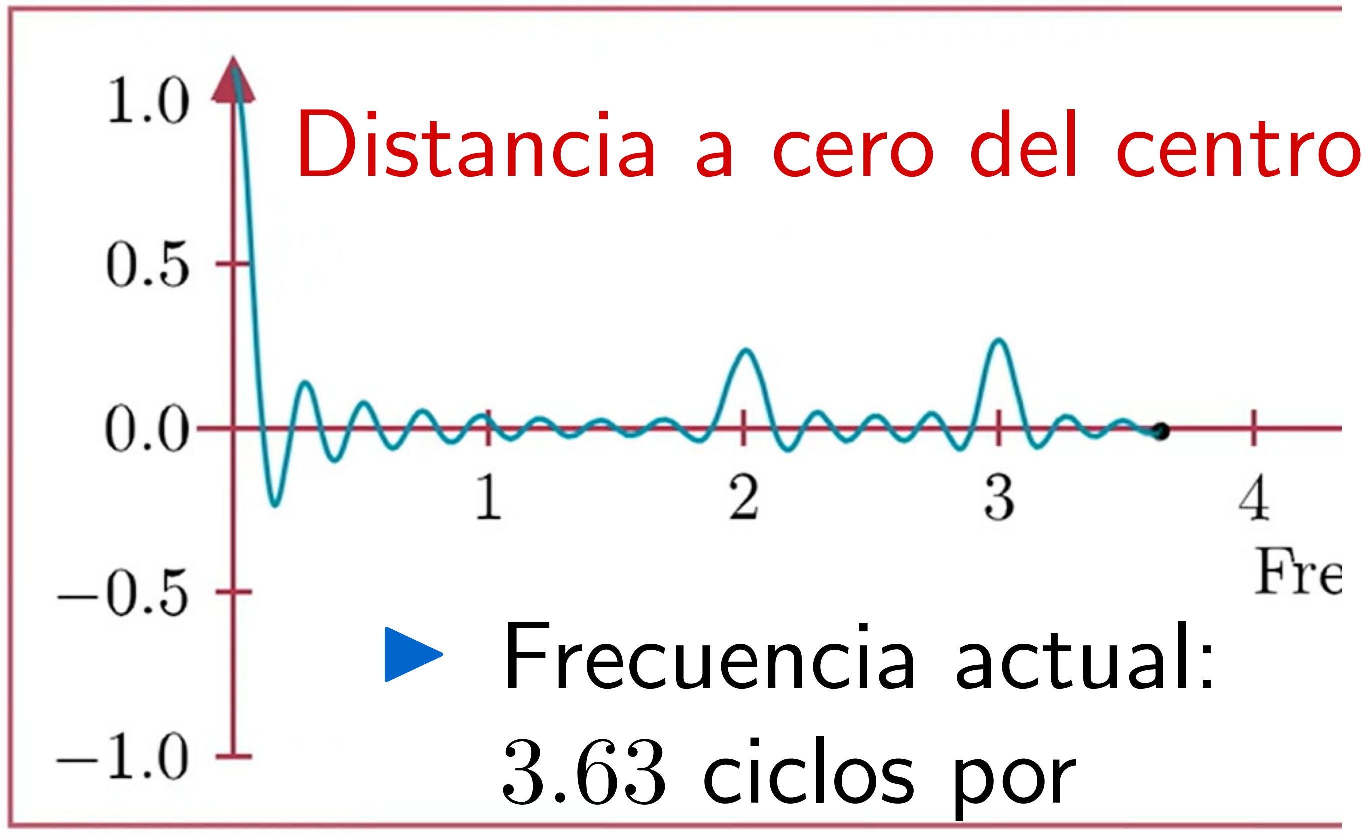


- Frecuencia actual: 3 ciclos por segundo

Intuición detrás de la DFT: todos los armónicos





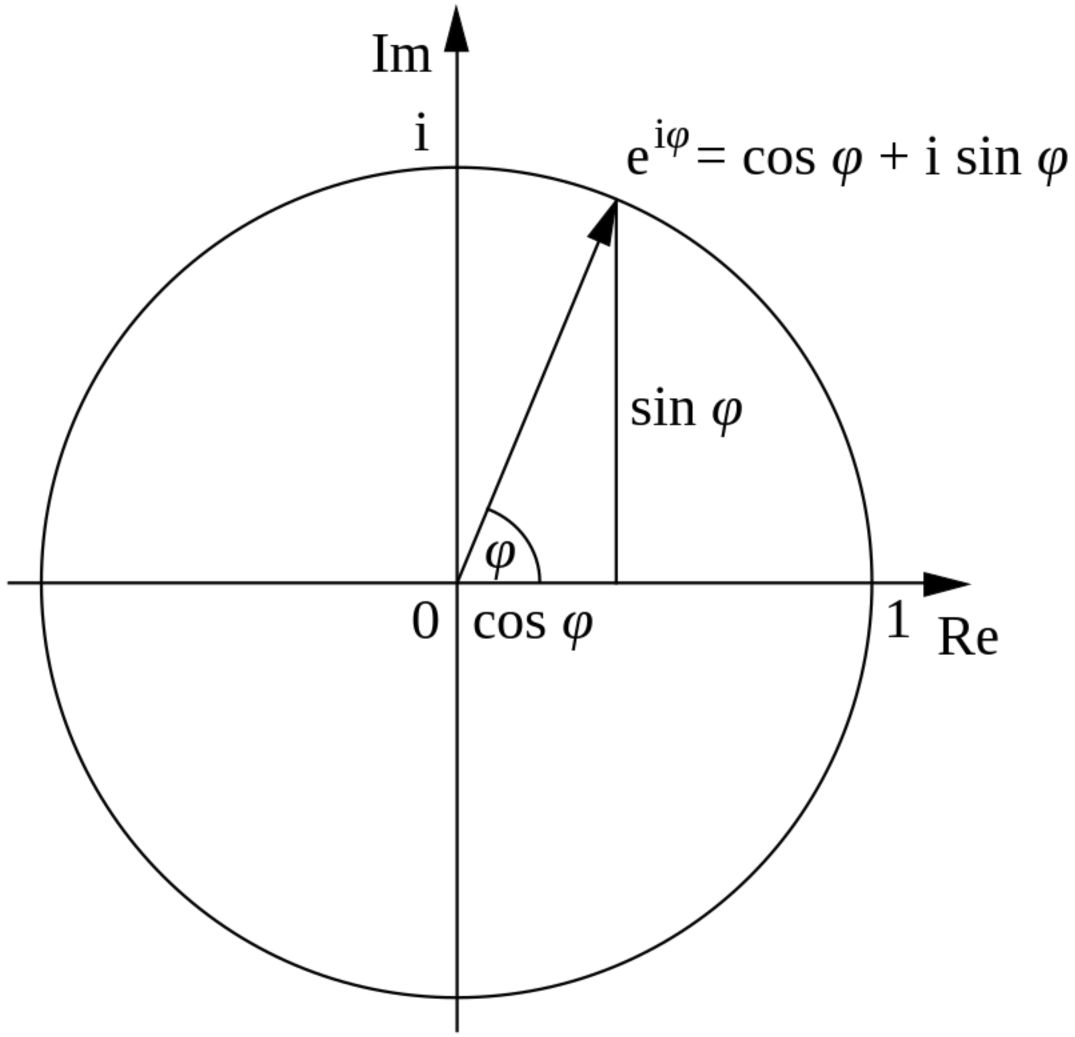


Distancia a cero del centro de masa

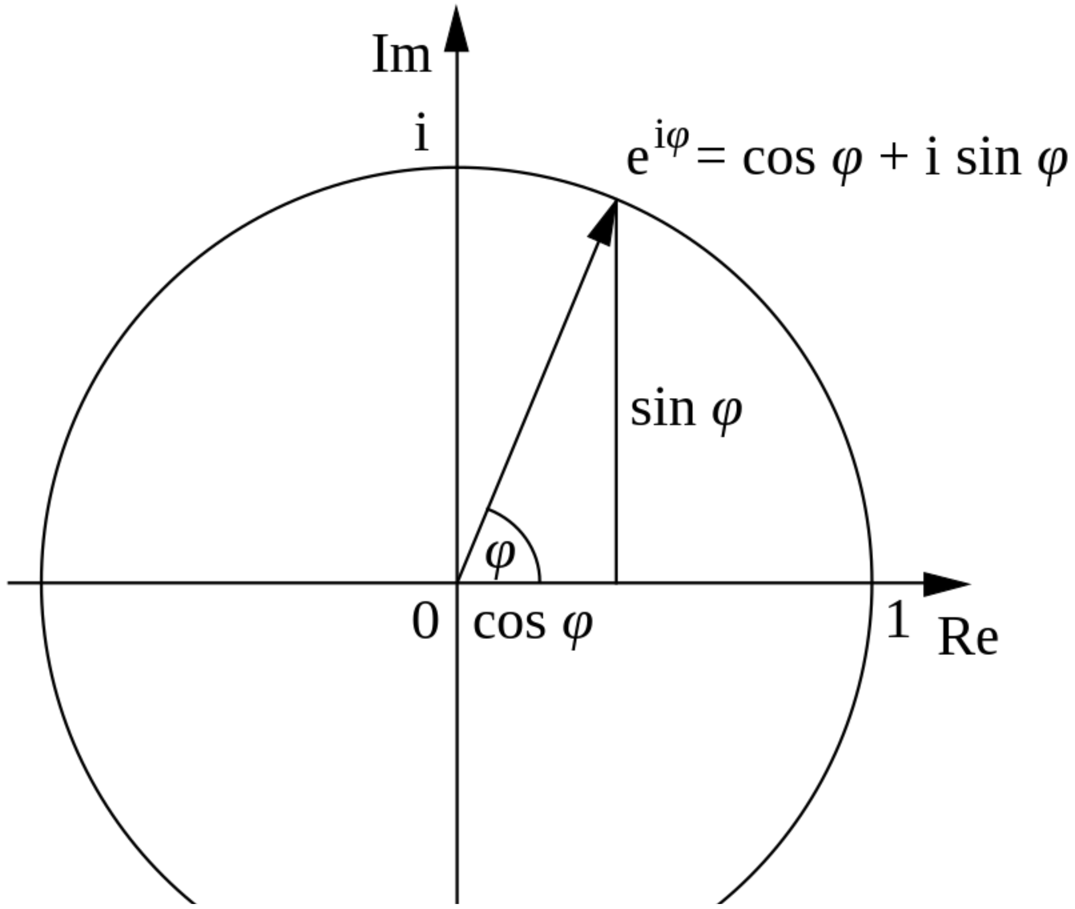
Frecuencia

- Frecuencia actual: 3.63 ciclos por segundo

Intuición detrás de la DFT: fórmula

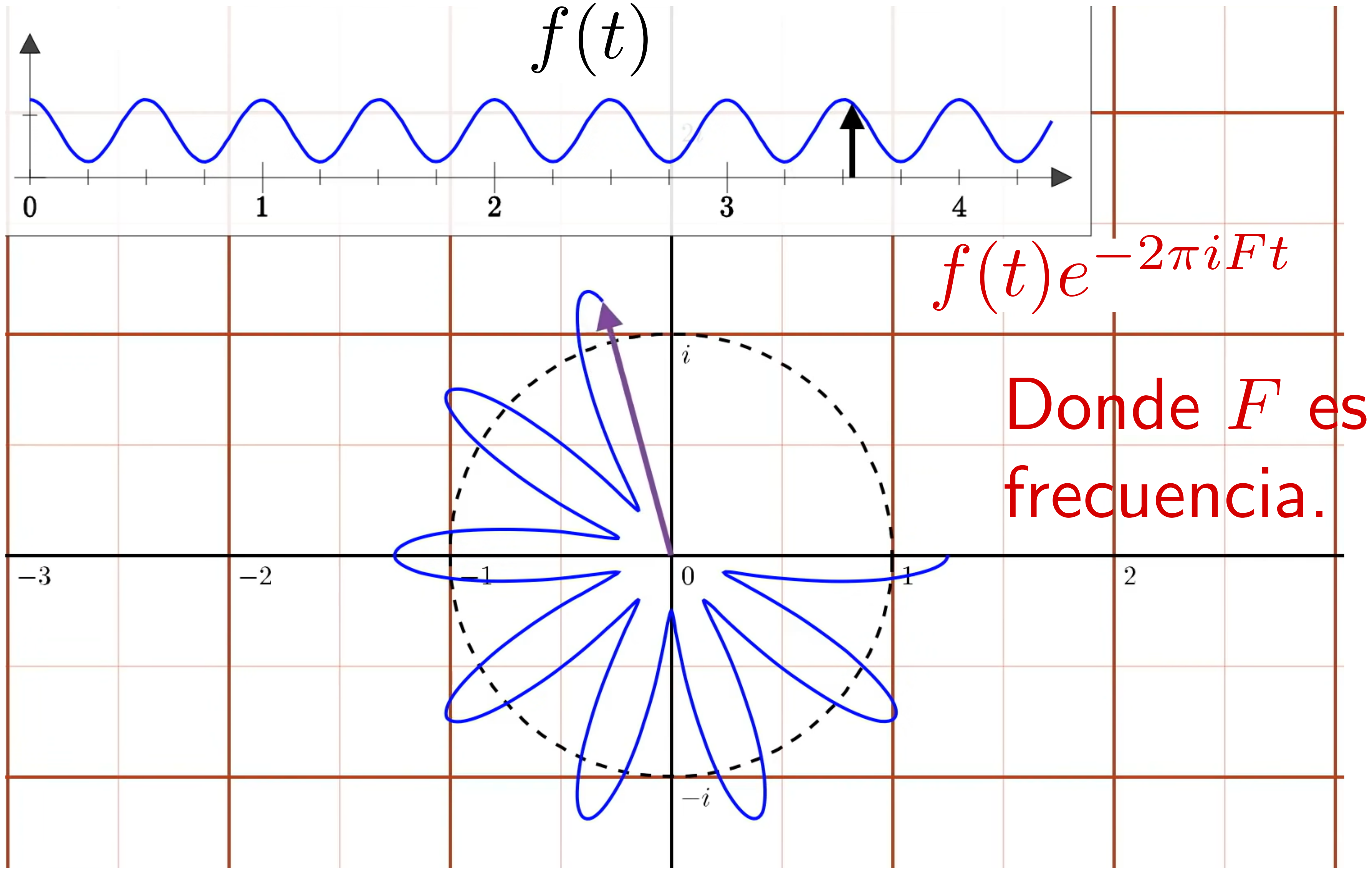


Intuición detrás de la DFT: fórmula



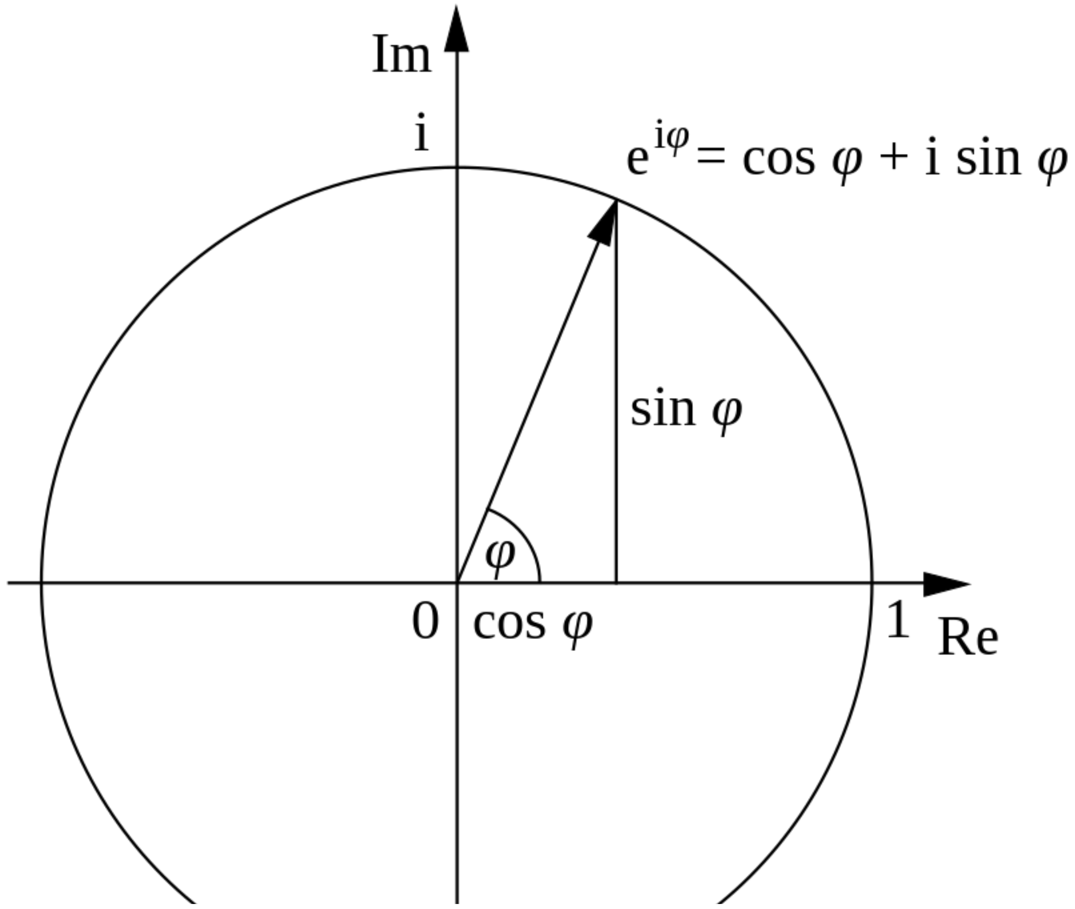
Signo − en el exponente: sentido horario

Intuición detrás de la DFT: fórmula



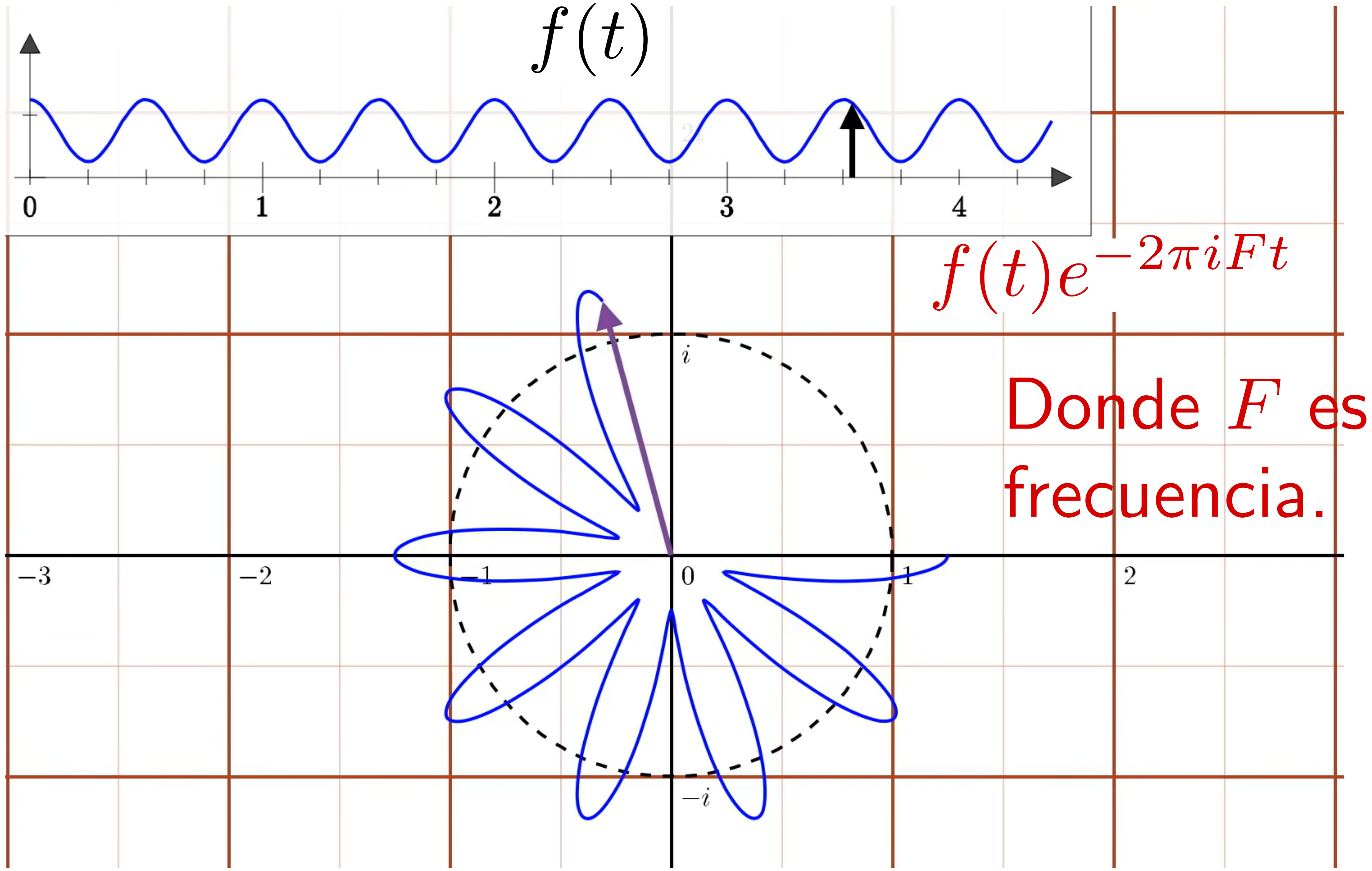
$f(t)e^{-2\pi i F t}$

Donde F es la frecuencia.



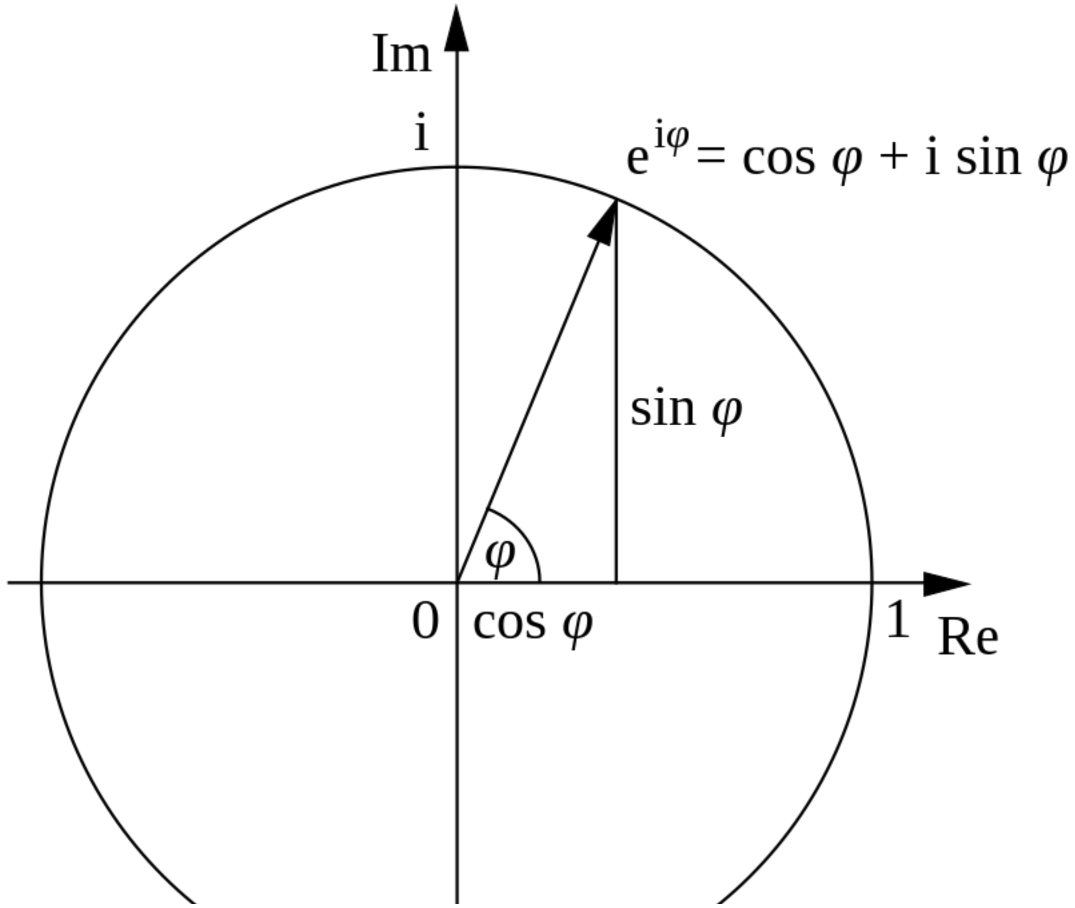
Signo $-$ en el exponente: sentido horario

Intuición detrás de la DFT: fórmula



$f(t)e^{-2\pi i F t}$

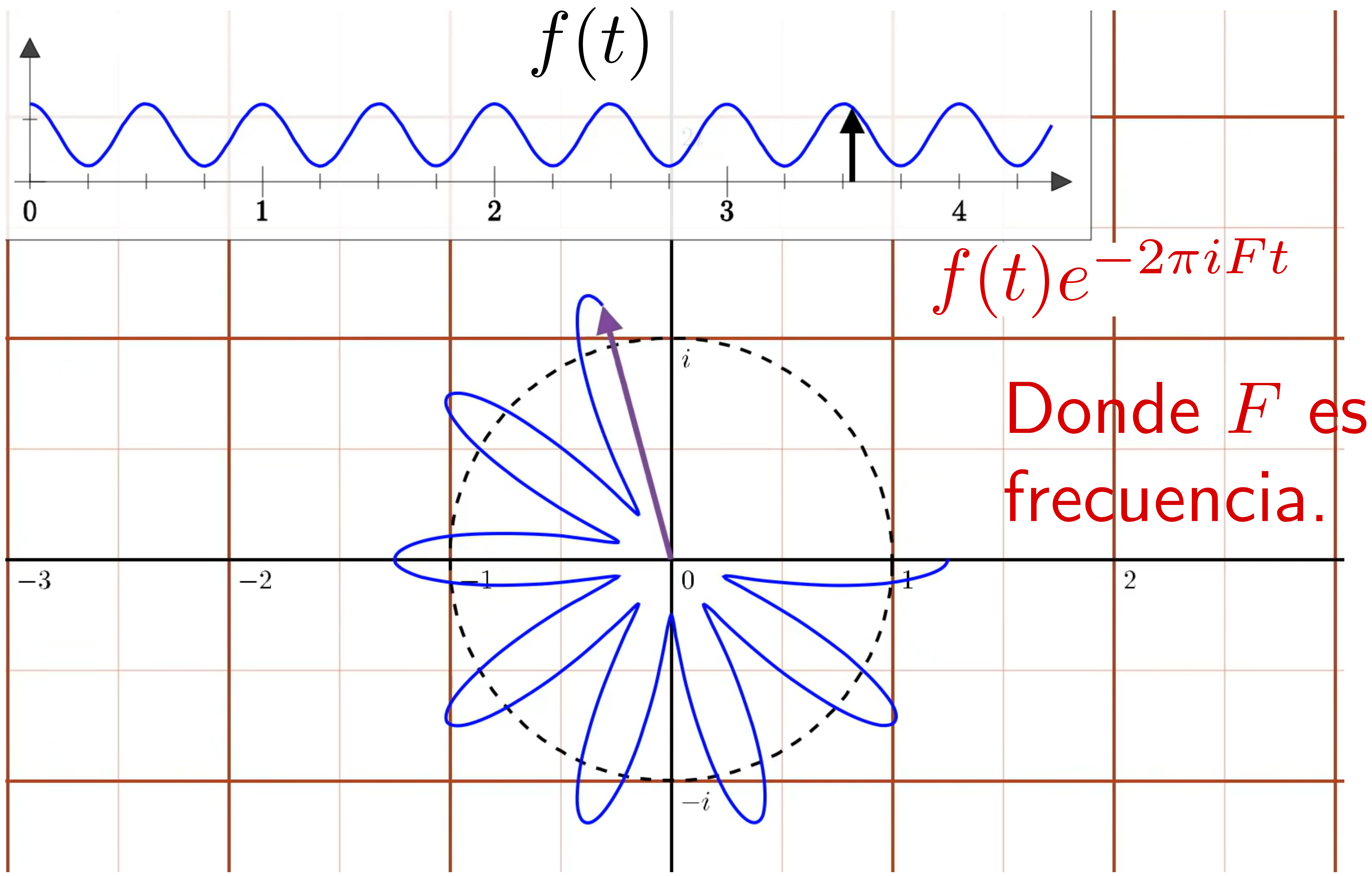
Donde F es la frecuencia.



Signo − en el exponente: sentido horario

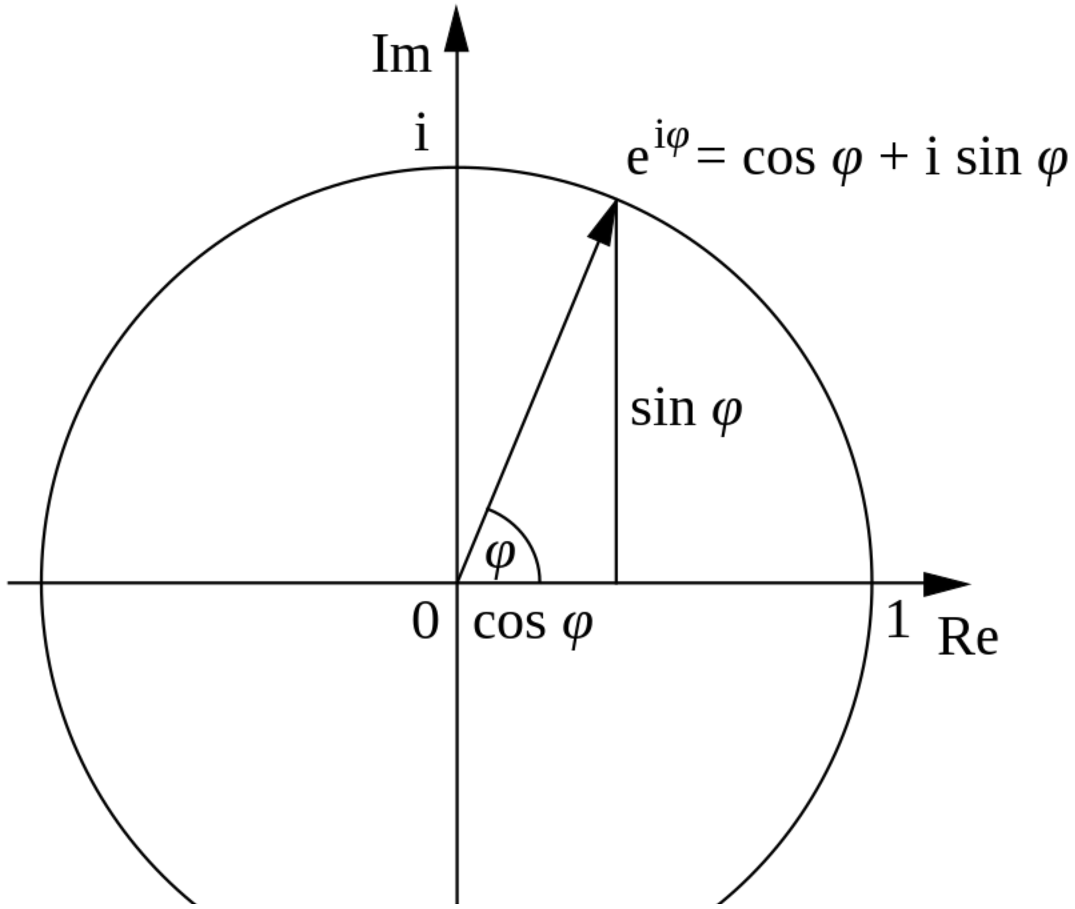
Ejemplo: frecuencia $F = 1/5$ significa que da una vuelta cada 5 segundos.

Intuición detrás de la DFT: fórmula



$f(t)e^{-2\pi i F t}$

Donde F es la frecuencia.



Signo $-$ en el exponente: sentido horario

La transformada discreta de Fourier (DFT) se define como

$$\hat{f}_k = n \cdot c_k = \cancel{\frac{1}{n}} \sum_{j=0}^{n-1} f(t_j) e^{-i\frac{2\pi k j}{n}}$$

$$\hat{f}_k = n \cdot c_k = \cancel{\frac{1}{T}} \int_{t_0}^{t_0+T} f(t) e^{-2\pi i k t/T}\, dt$$

Transformada rápida de Fourier (FFT)

- Es uno de los algoritmos (si no EL algoritmo) más potentes y relevantes del último siglo y quizá de todos los tiempos.

Transformada rápida de Fourier (FFT)

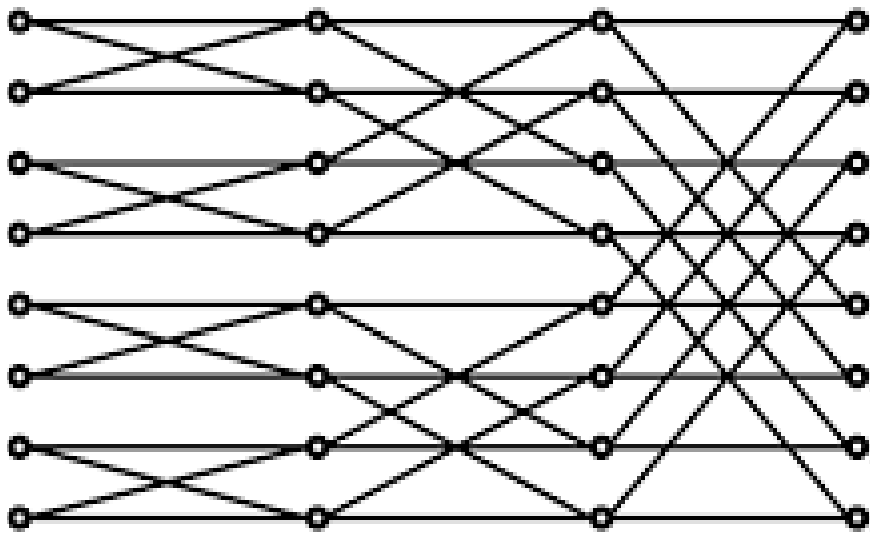
- Es uno de los algoritmos (si no EL algoritmo) más potentes y relevantes del último siglo y quizá de todos los tiempos.
- Tiene muchísimas aplicaciones: compresión de imagen, compresión de sonido, procesamiento de señales, HPC, ...

Transformada rápida de Fourier (FFT)

- Es uno de los algoritmos (si no EL algoritmo) más potentes y relevantes del último siglo y quizá de todos los tiempos.
- Tiene muchísimas aplicaciones: compresión de imagen, compresión de sonido, procesamiento de señales, HPC, ...
- La FFT es una manera muy eficiente de calcular la DFT: $O(n \log n)$ en vez de $O(n^2)$.

Transformada rápida de Fourier (FFT)

- Es uno de los algoritmos (si no EL algoritmo) más potentes y relevantes del último siglo y quizá de todos los tiempos.
- Tiene muchísimas aplicaciones: compresión de imagen, compresión de sonido, procesamiento de señales, HPC, ...
- La FFT es una manera muy eficiente de calcular la DFT: $O(n \log n)$ en vez de $O(n^2)$.
- Este algoritmo aprovecha una serie de simetrías de la matriz DFT para resolver calcular la DFT. En particular, si $n = 2^k$ el aprovechamiento de las simetrías es máximo y, por tanto su velocidad.



Guía de estudio

Guia de estudio

Libro *Càlcul numèric: teoria i pràctica* de M. Grau Sánchez y M. Noguera Batlle.

- Conceptos y ejercicios resueltos: Capítulo 2, apartado 2.3 y capítulo 9.
- Prácticas resueltas y propuestas: 2.5.

Libro *Cálculo numérico* de M. Grau Sánchez y M. Noguera Batlle.

- Conceptos y ejercicios resueltos: Capítulo 2, apartado 2.3.
- Prácticas resueltas y propuestas: 2.5.

Libro *Cálculo científico con MATLAB y Octave* de A. Quarteroni y F. Saleri.

- Conceptos y ejercicios: Capítulo 3, apartados 3.1.3, 3.2 y 3.3.
- Problemas y prácticas propuestos: del 3.5 al 3.14.

Libro *Numerical Computing with MATLAB* de C. Moler

- Capítulo 3, secciones 3.3, 3.4 y 3.5.