Revista digital
Matemática, Educación e Internet
https://revistas.tec.ac.cr/index.php/matematica
Vol 26, No 2. Marzo, 2026 – Agosto, 2026

ISSN 1659 -0643
DOI: XXX

Generar una Web Interactica con LaTeX (usando Make4ht e IA)

| Transforming LaTeX Documents into Interactive Websites (with AI & Make4ht) |

| Gerar uma Web Interativa com LaTeX (usando Make4ht e IA) |

Walter Mora Flores[1]
wmora2@gmail.com
Investigador independiente
Alajuela, Costa Rica

Recibido: 2 de mayo de 2025 Aceptado: 1 de agosto de 2025

Resumen: Este documento presenta una plantilla configurada para convertir documentos LaTeX en páginas web interactivas utilizando Make4ht. Se abordan temas como la configuración de Make4ht, la conversión de entornos complejos a imágenes SVG, y con la asistencia de la IA, la inclusión de actividades interactivas con HTML, JavaScript y CSS, y la integración de elementos multimedia como applets de GeoGebra y videos de YouTube. El enfoque bimodal permite generar tanto PDF como HTML, aprovechando las ventajas de cada formato: el PDF para impresión y citación formal, y el HTML para accesibilidad e interactividad en dispositivos digitales.

Palabras Clave: Make4ht, LaTeX, HTML, interactividad, JavaScript, IA

Abstract: This document presents a configured template for converting LaTeX documents into interactive web pages using Make4ht. It addresses topics such as Make4ht configuration, the conversion of complex environments into SVG images and, with the assistance of AI, the inclusion of interactive activities using HTML, JavaScript, and CSS. The template also covers the integration of multimedia elements like GeoGebra applets and YouTube videos. This bimodal approach allows for the generation of both PDF and HTML, leveraging the advantages of each format: PDF for formal printing and citation, and HTML for accessibility and interactivity on digital devices.

Keywords: Make4ht, LaTeX, HTML, interactivity, JavaScript, AI

Resumo: Este documento apresenta um modelo configurado para converter documentos LaTeX em páginas web interativas utilizando o Make4ht. São abordados temas como a configuração do Make4ht, a conversão de ambientes complexos em imagens SVG e, com a assistência da IA, a inclusão de atividades interativas com HTML, JavaScript e CSS, além da integração de elementos multimídia como applets do GeoGebra e vídeos do YouTube. A abordagem bimodal permite gerar tanto PDF quanto HTML, aproveitando as vantagens de cada formato: o PDF para impressão e citação formal, e o HTML para acessibilidade e interatividade em dispositivos digitais.

Palavras-chave: Make4ht, LaTeX, HTML, interatividade, JavaScript, IA

[1]Walter Mora-Flores. Investigador independiente. Dirección postal: San Ramón, Alajuela, Costa Rica. Código postal: 20201. Correo electrónico: wmora2@gmail.com

1. Introducción

Un documento .tex se puede compilar de varias maneras. En nuestro caso, necesitamos una salida bimodal: un documento PDF y un documento HTML (para añadir contenido interactivo).

El formato HTML ofrece accesibilidad, interactividad y un mayor alcance en el entorno digital actual. Generar el HTML desde un archivo LaTeX permite aplicar revisiones y/o modificaciones al archivo .tex sin necesidad de tener que volver a editar manualmente la salida HTML.

Mientras el formato PDF es ideal para impresión y citación formal, el HTML se adapta a cualquier dispositivo y permite integrar elementos dinámicos como visualizaciones, actividades interactivas y contenido multimedia. Este enfoque promueve la ciencia abierta y la educación digital al hacer el contenido más flexible y accesible.

Para obtener un documento PDF, compilamos con PDFLaTeX. Para traducir el documento .tex y obtener un sitio web, con su archivo de estilo .css, compilamos con Make4ht. El archivo .css se modifica y/o se amplia con un archivo personalizado config.cfg. En la Figura 1 se muestra el proceso de manera simplificada.

(**a**) PDFLaTeX (**b**) Make4ht

Figura 1: Compilar con PDFLaTeX o con Make4ht (esquema simplificado). Elaboración propia.

La plantilla .tex (en la que esta basada este documento) está implementado con código que ejecutan tanto PdfLaTeX como Make4ht. Pero hay partes en que solo nos interesa una salida HTML y otras en las que solo nos interesa la salida PDF, para esto, además del texto común, usamos comandos que PDFLaTeX ignora y que Make4ht sí compila y viceversa.

Make4ht se enfoca en la estructura del contenido y, si no hay errores de compilación fatales, entrega un archivo HTML correcto pero con un diseño simple. Este artículo es un ejemplo (y al mismo tiempo es una plantilla) de cómo agregar diseño web moderno y *responsivo*, de manera manual, a través de un archivo de configuración.

Este documento considera las *tareas comunes* en la implementación de material didáctico en matemáticas y agrega un archivo de configuración para la salida moderna y correcta en HTML. La manera de configurar Make4ht para lograr este objetivo se describe en la sección 7.

¿Qué nivel necesitamos? Para usar la plantilla, se requiere ser un usuario normal de LaTeX y posiblemente conocer cosas básicas de HTML y CSS (sino, podríamos empezar, por ejemplo, con W3Schools (s.f.-a) y W3Schools (s.f.-b)). La parte interactiva requiere un conocimiento básico de HTML, CSS y JavaScript.

Además, si queremos modificar la plantilla, necesitamos saber configurar Make4ht. Las lecturas recomendads son la sección 7 y la documentacion de Make4ht (Hoftich, 2024).

La IA se puede usar como asistente para generar código LaTeX, código de estilo CSS y scripts interactivos en JavaScript, con peticiones en *lenguaje natural*. Pero en el estado actual de la IA, para tareas más complejas, hay un aumento significativo en la productividad si hacemos peticiones muy específicas con modelos de código y en lenguaje técnico preciso. También debemos estar atentos a las repues-

tas de la IA, porque además de algún error de lógica, ocasionalmente nos da respuestas basadas en información desactualizada (lo cual nos genera errores de compilación), o al no ser adecuadamente específicos, nos puede dar código fuera de contexto e innecesariamente complicado.

2. Una plantilla.tex configurada

Tenemos un comprimido plantilla.zip que viene con un archivo plantilla.tex, con figuras y un archivo de configración config.cdf (entre otras cosas) que PDFLaTeX compila bien y Make4ht traduce bien a HTML.

Asumimos que tenemos una instalación TeX *completa* y actualizada al 2025 (las más populares son TeXLive, MikeTeX y MacTeX). Esta actualización ya viene con Make4ht. Esto es necesario porque varias cosas funcionan bien con la última actualización, sino podríamos tener errores de compilación o comportamientos inesperados.

En el código 1 podemos ver el inicio de la plantilla.tex.

```
\documentclass[fleqn,oneside]{article} % Puede usar también book, etc.
  \input{Paquetes/WebPreambuloyEntornos.tex}
  \input{Paquetes/ComandosdelUsuario.tex}
  \title{El título}
  \author{El autor}
  \date{}
  \ifdefined\HCode\Configure{HEAD}{% Insertar en el html<hedad></head>
        \HCode{<link rel="stylesheet" href="css/toc-toggle.css" />}
  }\fi
%
\begin{document}
\maketitle
\solopdf{% PDFLaTeX->pdf
             \tableofcontents
         }
\solomk{  % Make4ht -> html
             \webtableofcontents % Botón para el Contenido
       }
\section{Introducción}
%... código LaTeX usual...
\end{document}
```

Código 1: Incio del código de plantilla.tex.

Podemos agregar paquetes y comandos personales, en el archivo ComandosdelUsuario.tex que está en la carpeta "Paquetes". Make4ht soporta una lista amplia de paquetes esenciales: Incluye los paquetes que usualmente usamos. Mantenga los comandos personales "simples".

Hay varios paquetes como Nicematrix, Polynomial, hhline, etc. que no son soportados por Make4ht. Si tenemos código con paquetes no soportados, como veremos más abajo, usamos el comando `\solopdf{...}` para que Make4ht los ignore y, en el .html, se pueden incluir imágenes o podemos usar código equivalente (simplificado) en .tex, que sea soportado por Make4ht y usar el comando `\solomk{...}` (Make4ht lo compila pero PDFLaTeX lo ignora), o insertar directamente un equivalente en código HTML, con el comando `\insertarhtml{...}`. Use la IA como asistente, es buena idea.

La estructura de la carpeta de trabajo se muestra en la Figura 2. Y se explica a continuación.

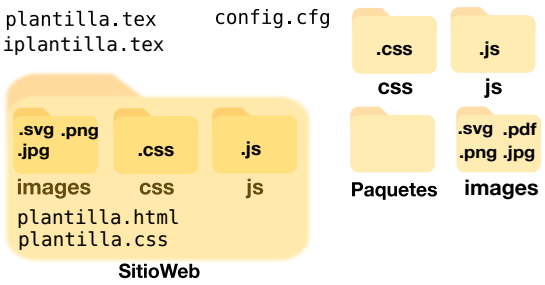


Figura 2: Carpeta de trabajo (plantilla.zip). Elaboración propia.

1. La carpeta images contiene figuras .png, .svg y .pdf. Para HTML son preferibles los formatos vectoriales como SVG (o sino PNG con buena resolución). Esto se puede personalizar un poco, por ejemplo usando Inkscape.

   Algunas figuras generadas con código LaTeX las podemos convertir, de manera automática, con el paquete ltximg (González, 2021). Lo vemos en la sección 3.

2. La carpeta JS contiene los scripts JavaScript (interactivos)

3. La carpeta CSS contiene archivos de estilo .css personalizados y el archivo plantilla.css que genera automáticamente Make4ht

4. La carpeta Paquetes contiene el archivo WebPreambuloyEntornos.tex y el archivo ComandosdelUsuario.tex

5. La carpeta SitioWeb recibe, con la compilación Make4ht , los archivos plantilla.css y plantilla.html (si la carpeta no está, entonces la crea). También recibe algunas figuras necesarias que Make4ht genera.

6. El archivo config.cfg es el archivo de configuración para este documento, mayormente habilita, deshabilita o modifica entornos creados por Make4ht e inserta código CSS para el diseño web de la salida HTML. Este archivo ya viene implementado para esta plantilla. Para comprender y/o modificar algo, es necesario leer la sección 7.

7. El archivo plantilla.tex es una parte de este documento, solo con cosas relevantes, para editar un documento propio. El archivo iplantilla.tex viene con código de las actividades interactivas.

2.1. Compilar esta plantilla

Make4ht ofrece varias opciones de compilación a través de un archivo de configuración y/o con opciones entre comillas. En este documento mayormente compilamos en la terminal, la línea de compilación se muestra en la el código 2.

```
make4ht  --utf8 -c config.cfg  -f html5   -d SitioWeb  plantilla.tex "mathml,
   svg,Gin-dim"
% opción limpiar:  agregar -m clean (y quitar después)
% opción debug:    agregar -a debug
```

Código 2: Compilar en terminal.

En esta plantilla, el archivo config.cfg puede cargar el script MathJax más personalizado, si se ocupan paquetes y comandos adicionales. Ver sección 7.

1. La opción -c config.cfg "inyecta" nuestro código, configuración de entornos y código de estilo CSS, en el archivo plantilla.css, de tal manera que prevalezca sobre el código CSS que genera Make4ht.
2. La opción -d SitioWeb inserta en SitioWeb, en la carpeta de trabajo, los archivos .css y HTML (si no está la carpeta, la crea).

Otras opciones. Se pueden agregar opciones de compilación (máximo 9) en el formato

```
make4ht filename.tex "op1,op2,..,optn"
```

En el capítulo 6 de Hoftich (2024) hay una lista de opciones que se pueden aplicar. Algunas opciones son:

1. Mathml. Usa MathML para las fórmulas.
2. Mathjax. Renderiza fórmulas con MathJax. En nuestro caso, esta opción esta incluida en el archivo config.cfg con una configuración adicional.
3. SVG. Convierte ecuaciones e imágenes a SVG. De acuerdo al sistema (Linux, Windows, macOS) y a las librerías instaladas, las figuras .pdf que se incluyen, se traducen automáticamente a .svg. Sino hay que hacerlo "manualmente".
4. frames. Añade navegación (tabla de contenidos) con marcos. Por ejemplo, si incluimos secciones, compilamos con
   ```
   make4ht -l archivo.tex "mathml,svg,frames,1"
   ```
5. footnotes-in. "Footnotes" al final de cada sección. Por ejemplo,
   ```
   make4ht archivo.tex "3,sec-filename,fn-in
   ```
6. footnotes-at-end. "Footnotes" en una página final. Por ejemplo,
   ```
   make4ht -f html5 archivo.tex "footnotes-at-end"
   ```
7. split-level=1. Divide el HTML por capítulos. Por ejemplo,
   ```
   make4ht -f html5 archivo.tex "split-level=1"
   ```
8. section+. Fuerza división por secciones. Por ejemplo,
   ```
   make4ht -f html5 archivo.tex "section+"
   ```
9. url-enc. Se usa cuando hay url's con caracteres especiales. Por ejemplo,
   ```
   make4ht -f html5 archivo.tex solourl-enc"
   ```
10. charset=utf-8. Usa UTF-8 para codificación. Por ejemplo,
    ```
    make4ht -f html5 archivo.tex "charset=utf-8" o make4ht -u -f html5 archivo.tex
    ```
11. table. Mejora las tablas HTML. Por ejemplo,
    ```
    "make4ht -f html5 archivo.tex "table,longtable"
    ```
12. info. Agrega metadatos.
13. tidy. Limpia y mejora el HTML final. Por ejemplo,
    ```
    make4ht -f html5 archivo.tex "tidy"
    ```
14. clean. Elimina archivos auxiliares tras la compilación. Por ejemplo,
    ```
    make4ht -m clean -f html5 archivo.tex
    ```
15. jats. Salida en XML JATS (para revistas científicas). Por ejemplo,
    ```
    make4ht -f html5 archivo.tex "jats"
    ```

2.1.1. Errores de compilación

Make4ht compila el archivo .tex con htlatex. Esta compilación usa "motores" como PDFTeX o LuaTeX, y soporta una cantidad suficiente de paquetes básicos. La copilación genera un dvi y agrega etiquetas HTML. La primera fuente de errores de compilación están en el código .tex como es usual. Una vez superada esta etapa, podrían haber errores en la traducción a HTML. Hay que evitar que el HTML se "rompa" (evitar que el DOM quede mal formado). por ejemplo por etiquetas HTML que abren, pero no cierran o caracteres extraños que inyecttan otros paquetes o por fórmulas o tablas muy complejas.

En la compilación con Make4ht, una "advertencia" del tipo

```
[WARNING] domfilter: DOM parsing of plantilla.html failed:
[WARNING] domfilter: /home/.../ ... nbalanced Tag (/p) [char=   ]
```

significa que la estructura del documento (DOM) se ha roto y la "advertencia" nos dice que una o varias cosas ya no van a salir bien. Lo normal es corregir el problema directamente o hacer modificaciones. Como en cualquier compilación normal, revisamos el .log y el .html, para tratar de aislar el problema y corregirlo.

El archivo de configuración en plantilla.zip ya incluye la configuración de los entornos que usamos, para evitar la mayoría de este tipo de problemas. Pero los errores puede aparecer si incluimos paquetes no soportados, comandos muy complejos o nuevos entornos que no siguen las reglas de configuración. Ver sección 7.

Caption. Cuando una macro expansiva (como `\caption`) contiene comandos con *expansión frágil*, la pila de parámetros puede desbordarse y aparece el error

```
! TeX capacity exceeded, sorry ...
```

Para resolver el problema agregamos el comando `\protect` antes de los comandos problemáticos (comandos frágiles). Estos comandos puede ser cosas como `\tfrac, \cosh` o comandos personales (en los que no se ha usado `\DeclareRobustCommand`). Por ejemplo deberíamos poner (si tuvieramos problemas)

```
\caption{Texto $ \protect\cosh t $}
\caption{Texto $ \protect\micomando $}
```

Tempa. Un error con algo como "`! Undefined control sequence.l... \tempa`" ocurre a veces porque hemos inyectado código HTML sin "escapar" caracteres, frecuentemente se refiere al caracter # en código `\Css{}` o `\HCode{}`. Por ejemplo

```
\Css{...background-color: #f8f8f8;font-size:83%...}
```

Debería ser: `\Css{...background-color: \#f8f8f8;font-size:83\%...}`

2.2. Comandos importantes.

Como estamos en un ambiente bimodal, algunas cosas se pueden usar tanto para PDFLaTeX como para Make4ht, pero hay otras cosas que son exclusivas para PDFLaTeX y que talvez hay algún equivalente para Make4ht y viceversa.

Observe que tanto PDFLaTeX como Make4ht, compilan *todo* el archivo .tex, solo que algunos bloques de código son exclusivos, por razones especiales, para PDFLaTeX y otros para Make4ht.

En esta plantilla tenemos algunos comandos personalizados (Ver Tabla 1) que están en WebPreambuloyEntornos.tex.

Tabla 1: Comandos personalizados de la plantilla. Elaboración propia.

Comando	*Descripción*
`\solopdf{#1}`:	PDFLaTeX ejecuta el código y Make4ht lo ignora.
`\solomk{#1}`:	Make4ht ejecuta el código ... y PDFLaTeX lo ignora.
`\insertarhtml{#1}`:	Código HTML *puro* con Make4ht (PDFLaTeX lo ignora)
`\ifdefined\HCode ...\fi`:	Make4ht ejecuta el código y PDFLaTeX lo ignora.
`\ifdefined\HCode ...\else...\fi`:	PDFLaTeX ejecuta el código después ... de `\else`.
`\linea`:	Una línea gris delgada, que es apropiada para HTML y PDF
`\bigskip`:	Para Make4ht se ha definido como un `<div>` de cierta altura

Comandos adicionales para generar código HTML desde LaTeX se muestran en la sección 5.

Ejemplo. En el código 3 se muestra cómo incluir una figura .pdf con PDFLaTeX e incluir la misma figura en formato .svg en HTML (Make4ht puede hacer la conversión automática de .pdf a .svg, pero depende del sistema, las librerías instaladas y del tipo de archivo .pdf). El resultado se muestra en la Figura 3.

```
\begin{center}
\includegraphics[scale=0.35]{images/sumaRiemann.pdf}
\captionof{figure}{Figura \tt{.pdf}}
\end{center}
```

Código 3: Comandos para incluir una figura.

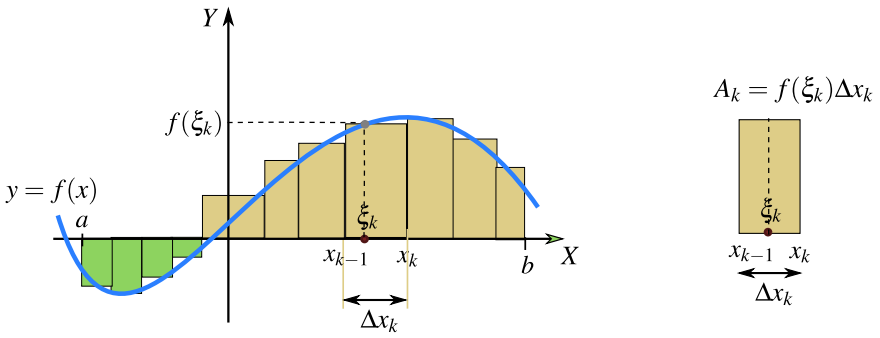


Figura 3: Resultado del código 3. Elaboración propia.

En este caso, la figura sumaRiemann.svg ha sido escala adecuadamente (con Inkscape) para la salida .html. Hay varios programas para escalar figuras.

2.3. Fuentes

En la plantilla.tex, para la compilación con PDFLaTeX usamos las fuentes Palatino, Tgpagella, Eucal, Beramono y Helvetica. Para la salida HTML, Make4ht carga en el archivo de configuración, la fuentes Google Lora, para el texto. Si no declaramos una fuente, el navegador usa una fuente genérica (depende del navegador), por ejemplo Times New Roman o Liberation Serif o Times o Georgia o Apple Garamond.

2.4. Figuras

En HTML, los formatos mejor soportados son PNG, JPG y SVG.

1. **PNG**. Incluir una figura PNG no requiere hacer cambios, compila igual con PDFLaTeX y Make4ht.

   ```
   \includegraphics[width=0.7\textwidth]{images/fig2.png}
   ```

   Para *escalar* una figura PNG o JPG, usamos una opción más adecuada para la salida HTML: La opción, `[width=0.x\textwidth]` (funciona para PDFLaTeX también).

   N Podría pasar que Make4ht no logre escalar una figura PNG si el archivo viene con información que Make4ht no puede manejar (estos archivos no tiene código estándar). En ese caso, se incluye la imagen con medidas absolutas y, de ser necesario se escala manualmente con algún software.

   Estas figuras se inyectan en el HTML con algo como (si hubo éxito escalando)

   ```
   <img alt='PIC' class='includegraphics' src='images/fig2.png' width='310' />.
   ```

   Es decir, también podemos insertar figras PNG desde el archivo .tex. Un ejemplo se muestra en el código 4.

   ```
   \insertarhtml{% Make4ht -> html
      <img alt='PIC' class='includegraphics' src='images/fig3.png'
            width='150' />}
   ```

   Código 4: Insertar figura con código HTML.

2. **SVG**. El formato SVG (Scalable Vector Graphics) se usa para mostrar gráficos vectoriales, escalables y nítidos, en la web.

   Se puede incluir incluir una figura SVG con Make4ht, usando el siguiente código:

   ```
   \solomk{\includegraphics{images/fig1.svg}}\caption{...}\label{...}
   ```

   N En Linux (digamos Ubuntu 24) generalmente (si están instaladas la librerías rsvg-convert y pdf2svg), al compilar con Make4ht, las figuras fig.pdf se convierten automáticamente a fig-.svg y se insertan en el HTML. Aunque hay figuras PDF que por alguna configuración no se logran convertir. Estas librerías están disponibles también en Windows.

   PDFLaTeX puede incluir figuras SVG con el paquete "svg", pero solo funciona para PDFLaTeX.

   Make4ht inyecta la figura .svg en el HTML con algo como:

   ```
   <object class='graphics' data='images/fig1.svg' type='image/svg+xml'></object>
   ```

   a) `<object>...,</object>` es un elemento HTML que incrusta contenido externo (svg, pdf, html, etc.) como un "documento embebido".

b) Estas figuras SVG *no se pueden escalar* con la opción usual `[width=0.x\textwidth]`. El navegador web las inserta con un tamaño absoluto. Esto se debe a que una figura SVG se puede escalar si accedemos a su atributo viewBox (sistema de coordenadas), pero Make4ht no hace eso, solo inserta la figura.

c) Si fuera necesario escalar estas figuras SVG, se pueden usar programas externos.

d) Podemos escalar las figuras SVG visualmente (o con comandos en terminal) por ejemplo con con Inkscape (u otro programa equivalente), como se muestra en la Figura 4.

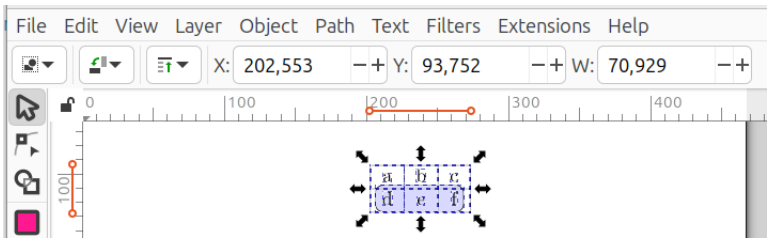


Figura 4: Escalar una figura .svg con Inkscape. Elaboración propia.

Los navegadores no *renderizan* los figuras PDF como imágenes (excepto con el elemento <object>). A veces Make4ht puede convertir estas figuras automáticamente (en la compilación) a PNG o SVG, mediante herramientas externas (como ghostscript o pdf2svg). Esto depende de si en el sistema están instaladas estas herramientas y depende también de la información que Make4ht pueda obtener de la figura (la codificación no es estándar).

2.5. Entorno minipage

La práctica usual de colocar un minipage al lado de otro minpage está configurado en el archivo config.cfg (porque necesitamos manejar de manera correcta las dimensiones de cada minipage).

Es conveniente, para que Make4ht traduzca bien, usar el formato que se muestra en la Figura 5 y en el código 5.

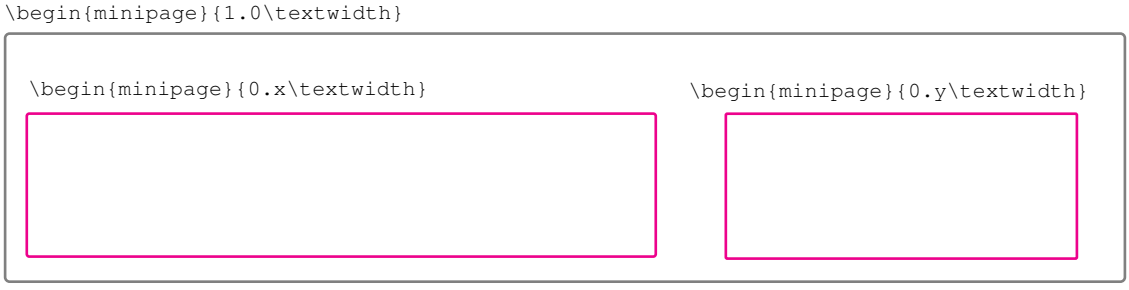


Figura 5: Entorno minipage. Elaboración propia.

```
\begin{minipage}{1.0\textwidth}
        \begin{minipage}{0.x\textwidth}
                % Contenido 1...
        \end{minipage}\hfill\begin{minipage}{0.y\textwidth}
                % Contenido 2...
        \end{minipage}
\end{minipage}
```

Código 5: Editar entorno minipage.

Ejemplo. En el código 6 se muestra cómo colocar, en un entorno minipage, texto a la izquierda y una figura a la derecha, con `\captionof{figure}`. Agregamos un par de líneas horizontales con `\linea` (definida en el preámbulo, correcta para PDFLaTeX y Make4ht).

```
\linea
\begin{minipage}{1.0\textwidth}
        \begin{minipage}{0.6\textwidth}
        \vspace{-2\baselineskip} % solo aplica para pdf
        El espacio Euclidiano es  $\R^2$, con  la métrica
        \[
        ds^2=dx^2+dy^2
        \]
        En este modelo, la distancia más corta entre dos puntos
        es una línea recta
        \end{minipage}\hfill\begin{minipage}{0.35\textwidth}
        \begin{center}
        \includegraphics{images/fig3.png}\captionof{figure}{Camino más corto}
        \end{center}
        \end{minipage}
\end{minipage}
\linea
```

Código 6: Minipage con texto y figura.

El resultado se muestra a continuación:

El espacio Euclidiano es $\mathbb{R}^2$, con la métrica

$$ds^2 = dx^2 + dy^2$$

En este modelo, la distancia más corta entre dos puntos es una línea recta

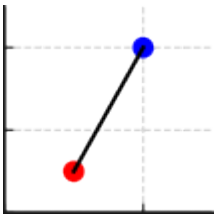


Figura 6: Camino más corto

2.6. Ambiente tikzpicture

Make4ht traduce bien, en general, los ambientes tikzpicture. Con cosas muy complicadas todavía tenemos la opción de recortar la figura con Inkscape u otro software.

Ejemplo. El código 7 muestra un diagrama generado con tikz (Figura 7).

```
\begin{center}
\begin{tikzpicture}
 \matrix (m) [matrix of math nodes, row sep=3.5em,
     column sep=3.5em, nodes={anchor=center}] {A & A[1] \\
    A' & A'[1] \\};
    \path[-stealth]
    (m-1-1) edge node [above] {$x$} (m-1-2)
    (m-2-1) edge node [above] {$y$} (m-2-2)
    (m-1-1) edge node [right] {$z$} (m-2-1)
    (m-1-2) edge node [right] {$t$} (m-2-2);
\end{tikzpicture}
```

```
\end{center}
```

Código 7: Diagrama Tikz.

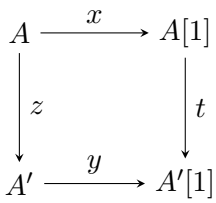


Figura 7: Resultado del código 7. Elaboración propia.

Ejemplo. El código 8 muestra parte del código para generar la Figura 8 con tikz.

```
\begin{tikzpicture}[scale=0.7, x=(-15:1.2), y=(90:1.0), z=(-150:1.0),
        line cap=round, line join=round,
        axis/.style={black, thick,->},
        vector/.style={>=stealth,->}]

        ...     % El código lo puede ver en el archivo "plantilla.tex"

\end{tikzpicture}
```

Código 8: Figura Tikz.

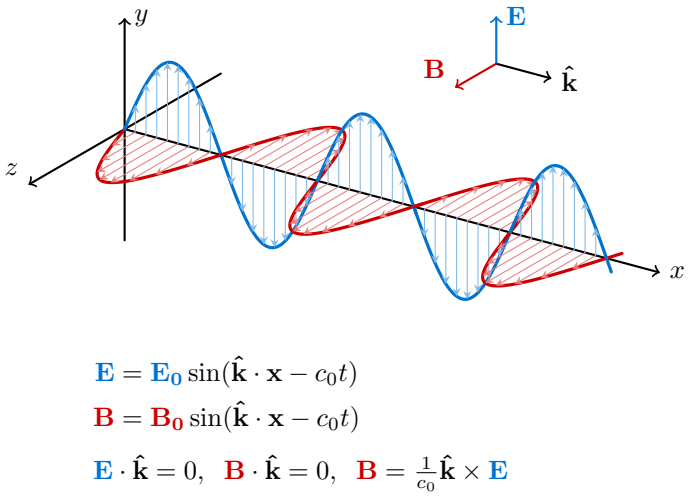


Figura 8: Resultado del código 8. Elaboración propia.

2.7. Matrices, ecuaciones y arreglos

Make4ht traduce estos entornos sin problema. Entornos demasiado complejos, posiblemente sea mejor, insertarlos en el HTML como una imagen SVG, como se indica en la sección 3, o generar el equivalente HTML con IA, si se pudiera.

El código completo de algunos ejemplos, se puede ver en el archivo plantilla.tex

1. El código 9 muestra un ejemplo del entorno equation con underbrace.

```
\begin{equation}
  \norm{f - L}_{\infty} = \overbrace{\abs{\frac{K_2}{2}}}^{\mathclap{\text{
   worst $f''(x)$ }}}\underbrace{\left(\frac{h}{2}\right)^2}_{\mathclap{\
   text{worst (b - a)}}}
\end{equation}
```

Código 9: Ejemplo de entorno `equation` con `underbrace`.

El resultado se muestra a continuación:

$$\|f - L\|_{\infty} = \overbrace{|\frac{K_2}{2}|}^{\text{worst } f''(x)} \underbrace{\left(\frac{h}{2}\right)^2}_{\text{worst (b - a)}} \tag{1}$$

- El código 10 muestra un ejemplo del entorno equation con cases.

```
\begin{equation}\label{eq:cases}
S_{i,t}=
\begin{cases}
 \begin{cases}
  [x_{i,t}=X^*, r_{i,t}=1] & \text{if  $\max\{X_{i,t}\}=X^*$} \\
  [x_{i,t}=\max\{X_{i,t}\}, r_{i,t}=0] & \text{if $\max\{X_{i,t}\} \neq X^*$
   }
 \end{cases}
  &\text{if $\sum_{i=1}^I u_{i,t-1}= \theta^{t-2} X^*$}\\
 \begin{cases}
  [x_{i,t}=1, r_{i,t}=1] & \hspace{\maxmin} \text{if $\min\{X_{i,t}\}=1$} \\
  [x_{i,t}=\min\{X_{i,t}\}, r_{i,t}=0] & \hspace{\maxmin} \text{if $\min\{X_
    {i,t}\} \neq 1$}
 \end{cases}
 &\text{otherwise}
 \end{cases}
\end{equation}
```

Código 10: Ejemplo de entorno `equation` y `cases`.

El resultado se muestra a continuación:

$$S_{i,t} = \begin{cases} \begin{cases} [x_{i,t} = X^*, r_{i,t} = 1] & \text{if } \max\{X_{i,t}\} = X^* \\ [x_{i,t} = \max\{X_{i,t}\}, r_{i,t} = 0] & \text{if } \max\{X_{i,t}\} \neq X^* \end{cases} & \text{if } \sum_{i=1}^{I} u_{i,t-1} = \theta^{t-2} X^* \\ \begin{cases} [x_{i,t} = 1, r_{i,t} = 1] & \text{if } \min\{X_{i,t}\} = 1 \\ [x_{i,t} = \min\{X_{i,t}\}, r_{i,t} = 0] & \text{if } \min\{X_{i,t}\} \neq 1 \end{cases} & \text{otherwise} \end{cases} \tag{2}$$

Hacemos referencia a los entornos en la manera usual.

El código: `En la ecuación \ref{eq:cases}, o el formato \eqref{eq:cases}, se tiene que ...` produce: En la ecuación 2, o el formato (2), se tiene que ...

- Entorno equation con array.

$$\underbrace{\begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix}}_{\boldsymbol{b}} = \underbrace{\begin{pmatrix} s_{11} & s_{12} & s_{13} & \cdots & s_{1n} \\ s_{21} & s_{22} & s_{23} & \cdots & s_{2n} \\ s_{31} & s_{32} & s_{33} & \cdots & s_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ s_{n1} & \cdots & \cdots & \cdots & s_{nn} \end{pmatrix}}_{\boldsymbol{S}} \cdot \underbrace{\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_n \end{pmatrix}}_{\boldsymbol{a}}$$

- Entorno pmatrix con block.

$$\underbrace{\begin{pmatrix} 1 & \underbrace{1\cdots1}_{k} & & & & & \\ & & 1 & \underbrace{1\cdots1}_{k} & & & \\ & & & & \ddots & & \\ & & & & & 1 & \underbrace{1\cdots1}_{k} \end{pmatrix}}_{T}$$

2.8. Paquete subfigure.

Usamos este paquete para colocar dos figuras, una al lado de la otra, con diferentes caption. El archivo de configuración tiene el código para manejar el entorno subfigure en la traducción a HTML. En la Figura 1, al inicio de este documento, usamos un código bimodal, porque usamos figuras .pdf para el PDF y el formato .svg para el HTML. El código 11 muestra parte de la edición al inicio de este documento.

```
\solomk{% Subfiguras con .svg. Make4ht -> html
\begin{figure}[h]
        \centering
        \begin{minipage}{1.0\textwidth}
                \centering
                \begin{subfigure}{0.4\textwidth}
                        \centering
                        \includegraphics{images/fig0a.svg}
                        \caption{PDFLaTeX}
                        \label{fig:fig0a}
                \end{subfigure}%
                ~ % espacio
                \begin{subfigure}{0.4\textwidth}
                        \centering
                        \includegraphics{images/fig0b.svg}
                        \caption{Make4ht}
                        \label{fig:fig0b}
                \end{subfigure}%
                % caption general
                \centering\caption{Compilar con PDFLaTeX o con Make4ht}
                \label{fig:fig0}
        \end{minipage}
\end{figure}}
```

Código 11: Ejemplo del entorno `Subfigure`.

2.9. Tablas

Make4ht traduce correctamente a HTML tablas simples como otras un pco más elaboradas, por ejmplo tablas que utilizan paquetes como multirow, color o booktabs. Sin embargo, cuando se trata de estructuras más complejas –como celdas combinadas de forma irregular, múltiples líneas horizontales o disposiciones no estándar– es necesario recurrir a soluciones alternativas, como las que se ilustran en la sección 3. Además, una vez generado el archivo HTML, es posible agregar interactividad directamente desde este mismo documento .tex (sección 5).

A continuación mostramos algunos ejemplos de tablas que Make4ht traduce bien a HTML.

1. El código 12 muestra una tabla con \multicolumn. La salida es la Tabla 2.

```
\begin{table}[h!!!]
{ % grupo
\arrayrulecolor{lightgray}
\begin{center}
 \begin{tabular}{ | l | c | c | c | c | c | c |}\hline\hline
  Tipo de Instrucción & opcode &  &  & & & \\\hline
  \multicolumn{1}{|r|}{\#bits} & 6 & 5 & 5 & 5 & 5 & 6\\\hline\hline
  Tipo-R & op & rs & rt & rd & shamt & funct\\\hline
  Tipo-I & op & rs & rt & \multicolumn{3}{c|}{dirección/inmediato}\\\hline
  Tipo-J & op & \multicolumn{5}{l|}{dirección objetivo}  \\\hline
 \end{tabular}
\caption{\label{tab:4.1}Tipos de instrucciones de código máquina MIPS}
\end{center}
}
\end{table}
```

Código 12: Tabla con `multicolumn`.

Tabla 2: Tipos de instrucciones de código máquina MIPS. Elaboración propia.

Tipo de Instrucción	opcode					
#bits	6	5	5	5	5	6
Tipo-R	op	rs	rt	rd	shamt	funct
Tipo-I	op	rs	rt	dirección/inmediato		
Tipo-J	op	dirección objetivo				

2. El código 13 muestra una tabla con \multirow y con \rowcolor. La salida es la Tabla 3.

```
\begin{table}[h!!!]
\centering
\begin{tabular}{|l|l|l|l|}
\hline
\rowcolor{gray!30}
\textbf{Categoría} & \textbf{Item} & \textbf{Cantidad}
        & \textbf{Precio} \\ \hline
        \multirow{2}{*}{Frutas} & Manzanas & 10 & \$5.00 \\ \cline{2-4}
        & Naranjas & 20 & \$8.00 \\ \hline
        \multirow{2}{*}{Vegetales} & Zanahorias & 15 & \$4.00 \\ \cline{2-4}
        & Papas& 25 & \$6.00 \\ \hline
\end{tabular}
```

```
        \captionof{table}{Tabla sofisticada. Elaboración propia.}\label{tab
    :4.2}
\end{table}
```

Código 13: Tabla con `multirow` y `rowcolor`

Categoría	Item	Cantidad	Precio
Frutas	Manzanas	10	$5.00
	Naranjas	20	$8.00
Vegetales	Zanahorias	15	$4.00
	Papas	25	$6.00

Tabla 3: Tabla sofisticada. Elaboración propia.

3. El código 14 muestra parte del código de una tablas con color en las filas. Esta tabla se puede traducir como una tabla "fija" o se puede traducir con un efecto "hover" sobre sus filas, como veremos en la sección 5. La salida es la Tabla 4.

```
\begin{table}[h!!!]
\centering
{% grupo
\footnotesize
\renewcommand{\arraystretch}{1.5}
\setlength{\arrayrulewidth}{0.5pt}
\arrayrulecolor{headerblue!80}
\rowcolors{3}{gray}{white}  % Empieza a rayar desde la 3ra fila
\begin{tabular}{cc||ccc||ccc||ccc||ccc}
\toprule
\rowcolor{headerblue!90}
...
\end{tabular}
} % grupo
\caption{Tablas de verdad básicas. Elaboración propia.}
\end{table}
```

Código 14: Tabla con color en las filas.

Negación		Conjunción			Disyunción			Implicación			Equivalencia		
P	$\neg P$	P	Q	$P \wedge Q$	P	Q	$P \vee Q$	P	Q	$P \to Q$	P	Q	$P \leftrightarrow Q$
V	F	V	V	V	V	V	V	V	V	V	V	V	V
V	F	V	F	F	V	F	V	V	F	F	V	F	F
F	V	F	V	F	F	V	V	F	V	V	F	V	F
F	V	F	F	F	F	F	F	F	F	V	F	F	V

Tabla 4: Tablas de verdad básicas. Elaboración propia.

2.10. Listas de ejercicios

El documento plantilla.tex usa el paquete exsheets para manejar listas de ejercicos al compilar con PDFLaTeX. En el preámbulo podremos ver el código que muestra 15.

```
\ifdefined\HCode % Paquete de ejercicios exsheets,no soportado por Make4ht
\else % compila PDFLatex
\usepackage[counter-format=se.qu,counter-within=section]{exsheets}
\DeclareTranslation{english}{exsheets-exercise-name}{\textcolor{tverde}{
   Ejercicio}}
\DeclareTranslation{english}{exsheets-question-name}{\textcolor{tverde}{Pregunta
   }}
 \DeclareTranslation{english}{exsheets-solution-name}{\textcolor{tverde}{Solució
   n}}
\SetupExSheets{
        question/pre-body-hook = {\normalfont},
        solution/pre-hook = {\par\medskip},
}
\fi
```

Código 15: Habilitar `exsheets` para PDFLaTeX.

Cuando compilamos con Make4ht, usamos el mismo código que en PDFLaTeX, solo que en el preámbulo se han definido los comandos para que PDFLaTeX lo interprete de una forma y Make4ht de otra, de tal manera que tengamos listas de ejercicios en ambos ambientes usando el mismo código.

1. `\hejersol{}{}`. Este comando lo ignora PDFLaTeX y Make4ht lo convierte en un enunciado con un botón Ver/Ocultar, para ver la respuesta en el renglón que sigue.
2. `\ejersol{}{}`.

   Este comando funciona para PDFLaTeX y Make4ht. PDFLaTeX lo convierte en un ejercicio normal y, para ver las soluciones, se pone al final del documento

   ```
   \solopdf{ \section*{Respuestas} \printsolutions }{}
   ```

Si compilamos con Make4ht, `\ejersol{}{}` se convierte en `\hejersol{}{}`. La idea es que a veces, un mismo ejercicio tiene un enunciado en el PDF, digamos con una figura fija por ejemplo, mientras que el enunciado en HTML no necesita la figura porque viene con otra cosa, por ejemplo un script.

Ejemplo. En el código 16 se muestra una lista de ejercicios. La salida en PDF corresponde al formato del paquete exsheets, la salida en HTML es una lista de enunciados con un botón para ver/ocultar la respuesta. La plantilla.tex contiene el código para insertar el script que gobierna el botón y su acción.

```
% Ejercicio 1:
\ejersol{Resuelve para $x$: $\frac{2}{3}x + 5 = 10$}{La solución es $x = \frac{
   15}{2}$}\\
% Ejercicio 2:
\ejersol{Calcula el área de un cuadrado de lado 3 cm}{$9\ \text{cm}^2$}\\
% Ejercicio 3:
\ejersol{Deriva $f(x) = 3x^2 + 2x - 1$}{$f'(x) = 6x + 2$}\\
% Ejercicio 4:
\ejersol{Simplifica $\sin^2 \theta + \cos^2 \theta$}{$1$}\\
% Ejercicio 5:  Solo PDFLaTeX
\solopdf{\ejersol{Calcula la velocidad final si $v_0=10\,m/s$,
        $a=2\,m/s^2$ y $t=3\,s$}{$16\,m/s$}
        }\\
% Ejercicio 6: Solo Make4ht
\solomk{\ejersol{Encuentra la energía cinética de un objeto de 5 kg
           moviéndose a 4 m/s}{$40\,J$}
        }\\
% Ejercicio 7:  Solo Make4ht
\hejersol{Halla la corriente si $V=12\,V$ y $R=4\,\Omega$}{$3\,A$}\\
```

```
% Al finaldel documento: \solopdf{ \section*{Respuestas} \printsolutions }{}
```

Código 16: Lista de ejercicios.

Ejercicio 2.1
Resuelve la ecuación $\frac{2}{3}x + 5 = 10$

Ejercicio 2.2
Calcular el área de un cuadrado de lado 3 cm

Ejercicio 2.3
Derivar $f(x) = 3x^2 + 2x - 1$

Ejercicio 2.4
Simplificar $\sin^2\theta + \cos^2\theta$

Ejercicio 2.5
Calcula la velocidad final si $v_0 = 10\,m/s$, $a = 2\,m/s^2$ y $t = 3\,s$

2.11. Tabla de contenidos: Ver/Ocultar

La tabla de contenidos en el PDF se obtiene con `\tableofcontents`. Make4ht compila `\tableofcontents` y genera una tabla de contenidos adecuada para el HTML.

La plantilla incluye un botón *minimalista* que abre y cierra una ventana con el "Contenido" en plantilla.html. Se requiere un script toc-toggle.js y un archivo toc-toggle.css que ya están en las carpetas CSS y JS. El código 17 muestra la manera de habilitar el botón. La Figura 9 es una vista de cómo se ve en el HTML.

```
\begin{document}
\maketitle
\solopdf{\tableofcontents}
\solomk{\webtableofcontents}
...
```

Código 17: Código para habilitar el botón de la tabla de contenidos.

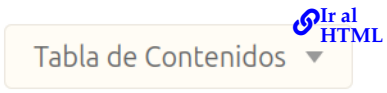


Figura 9: Botón para Abrir/Cerrar la tabla de contenidos. Elaboración propia

Con un poco más de código (*no* incluido en la plantilla), se puede agregar una barra de navegación como se muestra en la Figura 10. En este caso, en vez de un solo .html, usamos las opciones para generar varios archivos .html con las secciones (y capítulos si aplica).

Figura 10: Barra de navegación. Elaboración propia

3. Convertir entornos en imágenes svg

Make4ht *no ofrece* soporte para algunos paquetes que se usan para implementar tablas o arreglos complejos, como las que se implementan con hhline, nicematrix, polynom etc. Podemos convertir estos entornos en imágnes SVG o también, si se presta, se podrían convertir en HTML interactivo (sección 5). También podemos hacer lo mismo con entornos o fórmulas que por su complejidad, no se traducen bien.

1. En el caso de tablas o matrices que no traducen bien, se puede entregar el código LaTeX a la IA para que genere un equivalente en HTML e insertarlo con `\insertarhtml{}`. A veces el resultado, después de varios ajustes, es adecuado.

2. Más generalmente, una opción es abrir el PDF con Inkscape (o un programa equivalente). Inkscape (v. 1.4.1, (Project, 2025)) abre todas las páginas del PDF y solo seleccionamos las que nos interesan (ver Figura 11).

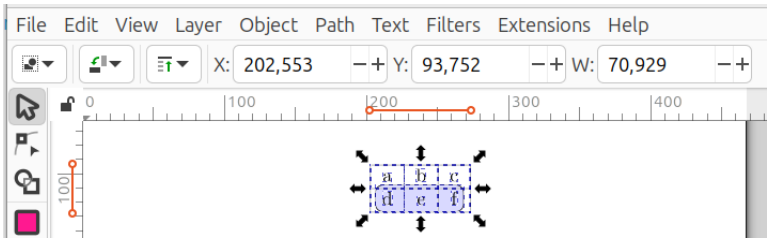


Figura 11: Recortar una figura y guardarla como .svg con Inkscape. Elaboración propia.

Una vez que seleccionamos la página correspondiente, del archivo del PDF, seleccionamos la figura (si hay que desagrupar usamos Crtl-U y a veces hay que más bien "agrupar", con Crtl-G) y la pegamos en otro documento. Recortamos la figura seleccionada con file-Document Properties-Resize to Content y luego guardamos como .svg.

Inkscape es muy conviniente para escalar y/o editar cosas adicionales con la extensión Textext (Extensions-Text-TexText).

3. Otra opción, en presencia de muchos entornos que no traducen bien, es el paquete ltximg (González, 2021). Se ejecuta con un comando *en la terminal*: Marcamos los entornos que queremos convertir a imágenes con `%<*ltximg> %</ltximg>` y la compilación PdfLaTeX convierte estos entornos en imágenes .svg. Ver el Apéndice A.

Ejemplos: Convertir entornos en imágenes.

Vamos a recortar entornos con los paquetes nicematrix, polynomial y hhline. Recuerde que estos paquetes *no son soportados por Make4ht*, pero sí por PDFLaTeX.

El orden de aparición es importante, porque eso define el nombre de la imagen .svg.

1. El código 18 muestra cómo se genera una división de polinomios con el paquete polynom (Figura 12). En este caso abrimos el PDF con Inkscape y seleccionamos la división de polinomios (en este contexto, la división es una imagen), la escalamos y la guardamos con el nombre doc-fig-1.svg.

```
\begin{center}
        \polylongdiv[style=D]{6x^3-2x^2+x+3}{x^2-x+1}
\captionof{figure}{División con el paquete \tt{polynom}. Elaboración propia.
    }
\end{center}
```

Código 18: Recortar entornos como imágenes.

$$\begin{array}{rl|l} 6x^3-2x^2 & +x+3 & x^2-x+1 \\ \cline{3-3} -6x^3+6x^2-6x & & 6x+4 \\ \cline{1-1} 4x^2-5x+3 & & \\ -4x^2+4x-4 & & \\ \cline{1-1} -x-1 & & \end{array}$$

Figura 12: Ejemplo de división con el paquete polynom. Elaboración propia.

2. El código 19 muestra cómo se genera una fórmula con el paquete nicematrix y otros paquetes (Figura 13). En este caso abrimos el PDF con Inkscape y seleccionamos la fórmula (en este contexto, la fórmula es una imagen), la escalamos y la guardamos con el nombre doc-fig-3.svg.

```
\begin{equation*}
   \boldsymbol{\mathcal{F}}^{i} =
   \begin{bNiceMatrix}[margin,nullify-dots]
    ...
       \end{bNiceMatrix}
   \end{equation*}
```

Código 19: Recortar entornos como imágenes.

$$\boldsymbol{\mathcal{F}}^{i} = \begin{bmatrix} \overbrace{\begin{bmatrix} f_{1\,1} & \cdots & f_{1\,\varpi} \\ \vdots & \ddots & \vdots \\ f_{\varpi\,1} & \cdots & f_{\varpi\,\varpi} \end{bmatrix}}^{\mathcal{F}_N^i} & \begin{matrix} f_{1\,(\varpi+1)} & \cdots & f_{1\,\eta} \\ \vdots & & \vdots \\ f_{\varpi\,(\varpi+1)} & \cdots & f_{\varpi\,\eta} \end{matrix} \\ \begin{matrix} f_{(\varpi+1)\,1} & \cdots & f_{(\varpi+1)\,\varpi} \\ \vdots & & \vdots \\ f_{\eta\,1} & \cdots & f_{\eta\,\varpi} \end{matrix} & \begin{matrix} f_{(\varpi+1)\,(\varpi+1)} & \cdots & f_{(\varpi+1)\,\eta} \\ \vdots & \ddots & \vdots \\ f_{\eta\,(\varpi+1)} & \cdots & f_{\eta\,\eta} \end{matrix} \end{bmatrix}$$

Figura 13: Resultado del código 19. Elaboración propia.

4. Entornos definición, teorema, ejemplo, etc.

La plantilla tiene varios entornos ya configurados para que compilen bien con Make4ht. Agregar nuevos entornos posiblemente requiera agregar nuevo código al archivo de configuración config.cfg (ver sección 7). Lo recomendable es usar los modelos que ya están en este archivo. Si usa IA para configurar algún nuevo entorno, lo mejor es indicarle que genere código siguiendo esos modelos ya probados y avalados por la documentacion más reciente.

Para el PDF d ela plantilla, las cosas de diseño de cada entorno, como bordes, color, color de fondo, etc. se puede modificar en WebPreambuloyEntornos.tex.

Modificar el diseño de cada entorno, para la salida HTML, requiere modificar entre otras cosas, el `\Css{...}` del entorno en el archivo config.cfg. Por ejemplo, el código 20 muestra parte del diseño del entorno para las definiciones (además se debe controlar los cierres de etiquetas y los entornos minipage dentro de las definiciones).

```
% Archivo config.cfg
\Css{% No dejar renglones en blanco y "escapar" \# y \%
 .definicion {
   background-color: rgb(251, 251, 250);
   border-left: 4pt solid rgb(255, 20, 147);
   margin: 20pt 0;
   padding: 0pt 4pt 0 4pt;
   overflow: auto;
   clear: both;
        }
  .definicion-title {
   color: rgb(255, 20, 147);
   font-weight: bold;
   margin-bottom: 0.5em;
   display: block;
}
```

Código 20: Parte del diseño para el entorno "definicion"(ya implementado).

A continuación tenemos una lista de entornos con ejemplos.

1. *Definiciones.* El código 21 muestra el formato general de una definición.

```
\label{def:defi1}
\begin{definicion}[nombre-opcional]
 Contenido...
\end{definicion}
```

Código 21: Una definición.

   - Definición con nombre. En estos casos, se puede hacer referencia a la definición con `\ref`. As+i se mostrará como "En la Definición 1..."

> **Definición 1 (Envoltura Convexa).**
> Sea $S \subset \mathbb{R}^n$ un conjunto de puntos. La *envoltura convexa* de S, denotada por $\text{Conv}(S)$, es el conjunto convexo más pequeño que contiene a S. Formalmente:
>
> $$\text{Conv}(S) = \left\{ \sum_{i=1}^{k} \lambda_i x_i \;\middle|\; x_i \in S,\ \lambda_i \geq 0,\ \sum_{i=1}^{k} \lambda_i = 1,\ k \in \mathbb{N} \right\}$$
>
> Es decir, $\text{Conv}(S)$ está formado por todas las combinaciones convexas finitas de puntos de S.

   - La Definición 2 presentan una definición con minipage y una nota `\footnote{...}`.

Definición 2 (Modelo para Geometría Hipérbolica).
Un modelo para la geometría hiperbólica es el semiplano superior

$$\mathbb{H} = \{(x, y) \in \mathbb{R}^2 \,|\, y > 0\}$$

equipado con la métrica

$$ds^2 = \frac{dx^2 + dy^2}{y^2}.$$

Al conjunto $\mathbb{H}$ se le llama semiplano superior de Poincaré.[a]

[a] En honor al gran matemático francés que lo descubrió.

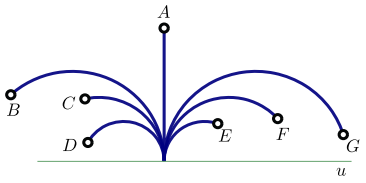


Figura 14: Rectas paralelas

2. *Teoremas*. El código 22 muestra el formato general de un teorema.

```
\label{teo:teo1}
\begin{teorema}[nombre-opcional]
                Contenido...
\end{teorema}
```

Código 22: Un teorema.

- Teorema con nombre (Teorema 2).

Teorema 1 (Parametrización de un triángulo).
Sean $A, B, C \in \mathbb{R}^2$ no colineales entonces existen t, s, w únicos tales que

$$\triangle ABC = \{tA + sB + wC, \quad \text{donde } t, s, w \geq 0, \quad t + s + w = 1\}$$

- El código 23 muestra un teorema con demostración (Teorema 2). Usamos un entorno caja que ya está configurado. Es una caja simple, con fondo blanco pero tiene atributos bien definidos en la configuración para la traducción a HTML.

```
\label{teo:teo2}
\begin{teorema}[La recta es el camino más corto]
...
\begin{caja}[Demostración.]
...
\end{caja}
\end{teorema}
```

Código 23: Un teorema con demostración.

Teorema 2 (La recta es el camino más corto).
En el plano euclidiano $\mathbb{R}^2$ con la métrica usual, el camino más corto entre dos puntos A y B es el segmento de recta que los une.

Demostración. Sea x un número real arbitrario. Sea $\gamma(t) = (x(t), y(t))$ una curva suave que une dos puntos A y B en el intervalo $[a, b]$. Su longitud es:

$$L[\gamma] = \int_a^b \sqrt{x'(t)^2 + y'(t)^2}\, dt.$$

Aplicamos la desigualdad de Cauchy–Schwarz:

$$L[\gamma] \geq \sqrt{(x(b) - x(a))^2 + (y(b) - y(a))^2},$$

con igualdad si y solo si $\gamma'(t)$ es constante y paralelo al vector $B - A$, es decir, si γ es una recta.
Por lo tanto, la longitud mínima se alcanza exactamente cuando γ es el segmento de recta entre A y B. □

3. El código 24 muestra el formato general de un ejemplo ilustrativo.

```
\label{ej:ejemplo1}
\begin{ejemplo}[nombre-opcional]
 % Enunciado
\end{ejemplo}
```

Código 24: Un ejemplo.

- El código 25 muestra el ejemplo 1 que usa el entorno minipage

```
\solopdf{ % PDFLaTeX -> pdf
\label{ej:ejemplo1}
\begin{ejemplo}[Punto dentro del triángulo]
   \begin{minipage}{0.7\textwidth}
          Sean \( A = (1,1) \), \( B = (2,4) \), \( C = (4,0) \).
        ...
   \end{minipage}\hfill\begin{minipage}{0.2\textwidth}
       ...
    \includegraphics[scale=0.7]{images/fig2.pdf}
  \end{minipage}
\end{ejemplo}
 }
 \solomk{  % Make4ht -> html
\label{ej:ejemplo1}
\begin{ejemplo}[Punto dentro del triángulo]
   \begin{minipage}{0.7\textwidth}
          Sean \( A = (1,1) \), \( B = (2,4) \), \( C = (4,0) \).
        ...
   \end{minipage}\hfill\begin{minipage}{0.2\textwidth}
       ...
   \includegraphics[scale=0.7]{images/fig2.svg}
   \end{minipage}
   \end{ejemplo}
```

```
\end{ejemplo}
}
```

Código 25: Un ejemplo con solución.

Ejemplo 1 (Punto dentro del triángulo).

Sean $A = (1,1)$, $B = (2,4)$, $C = (4,0)$. El punto $P = (3,1)$ está dentro del triángulo $\triangle ABC$ pues, como este triágulo es un conjunto comvexo y

$$P = \tfrac{1}{4}A + \tfrac{1}{4}B + \tfrac{1}{2}C.$$

entonces se concluye que $P \in \text{Conv}\{A, B, C\} = \triangle ABC$.

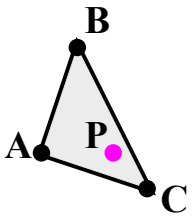


Figura 15: $\triangle ABC$ y P

- Un ejemplo con solución (Ejemplo 2) se puede crear usando el entorno `\begin{caja}[Solución]`.

Ejemplo 2 (Punto fuera del triángulo).

Sean $A = (1,1)$, $B = (2,4)$, $C = (4,0)$. Sea $P = (3,4)$. Verifique que P está fuera del triángulo $\triangle ABC$.

Solución. Resolviendo el sistema 3×3

$$P = tA + sB + wC, \quad t + s + w = 1,$$

obtenemos la única solución $t = -1$, $s = 2$, $w = 0$, y como uno de los coeficientes es negativo, $P \notin \text{Conv}\{A, B, C\} = \triangle ABC$. □

En esta plantilla se han definido más entornos. El diseño se puede modoficar tanto en el archivo WebPreambuloyEntornos.tex(paraPDFLaTeX) como en el archivo de configuración config.cfg (para Make4ht). La lista de entornos es:

1. Entornos:
   - *a*) `\ObjetivoGeneral{...}`
   - *b*) `\ObjetivosEspecificos{...}`
   - *c*) `\resumen{...}`
   - *d*) `\palabrasclave{...}`
   - *e*) `\abstract{...}`
   - *f*) `\keywords{...}`
   - *g*) `\begin{definicion}...\end{definicion}`
   - *h*) `\begin{teorema}...\end{teorema}`
   - *i*) `\begin{ejemplo}...\end{ejemplo}`
   - *j*) `\begin{corolario}...\end{corolario}`
   - *k*) `\begin{proposicion}...\end{proposicion}`
   - *l*) `\begin{lema}...\end{lema}`

m) \begin{caja}...\end{caja}

n) \begin{axioma}...\end{axioma}

ñ) \begin{actividad}...\end{actividad}

o) \begin{notahistorica}...\end{notahistorica}

p) \begin{problema}...\end{problema}

q) \begin{ejercicio}...\end{ejercicio}

r) \begin{proyecto}...\end{proyecto}

s) \begin{aplicacion}...\end{aplicacion}

5. Agregar actividades interactivas

Ahora que ya tenemos una plantilla configurada para traducir LaTeX a HTML (para introducir contenido matemático), pasamos a introducir interactividad. En esta parte, usamos \insertarhtml{} para incrustar el bloque HTML para nuestras actividades interactivas (en el caso de que no queramos que estén en páginas independientes). Asumimos un conocimiento básico de HTML, CSS y JavaScript.

5.1. Scripts con JavaScript

Las actividades interactivas en JavaScript en este documento requieren *separación modular*. Cada actividad puede tener su propia lógica, estilo y comportamiento: Sus componentes de estilo CSS y el código JS, los separamos de las otras actividades. Esto hace que el código se pueda mantener mejor, evita conflictos de estilo entre módulos distintos y se puede reutilizar fácilmente un módulo en otra parte o en otro proyecto.

Usualmente los scripts se comunican con el el sitio web porque el sitio web les provee de campos de texto, botones, lienzos (canvas) para graficar, escribir, etc.

Cada actividad interactiva con JavaScript tiene una introducción teórica y/o práctica e instrucciones. Luego viene el bloque de código HTML de la página. Este códigose comunica con el script: Este código toma el estilo de una archivo .css en la carpeta CSS y el escript .js "vive" en la carpeta JS , como se muestra en al Figura 16.

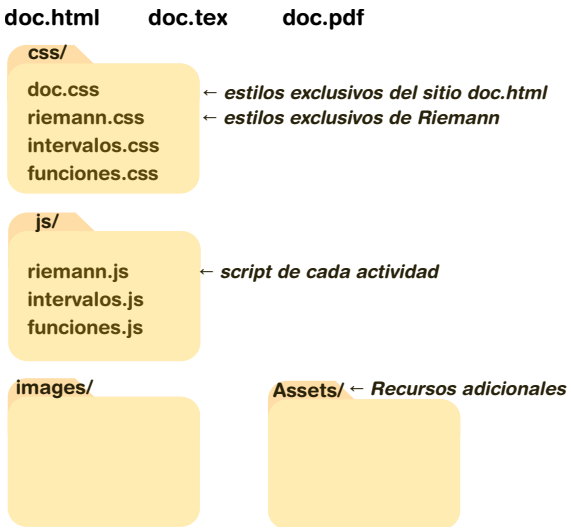


Figura 16: Estructura del sitio web. Elaboración propia.

El código .js y el estilo .css lo implementamos nosotros mismos o, por supuesto, pedimos asistencia de la IA. Podríamos solicitar traducir un program ya implementado en otro lenguaje a un programa equivalente .js o modificar un programa existente o indicar todos los detalles del programa que queremos en *lenguaje natural*.

Para proyectos complejos necesitamos sin duda una especificación con modelos de código y lenguaje técnico para pedir asistencia a la IA.

Si le pedimos a la IA (Copilot, ChatGPT, DeepSeek, etc.) una aplicacion interactiva (no muy compleja), entonces en general, seguimos estos pasos:

1. Pedimos el HTML completo de una página web con la actividad interactiva que solicitamos. Así nos aseguramos que todo funcione bien. También es útil si vamos a usar esta página web de manera independiente.

   - Primero pedimos un diseño básico de lo que queremos, con los detalles pertinentes.
     Hay que tomar en cuenta que en JavaScript hay bibliotecas para graficar en 2D, 3D, derivar en una o varias variables, simplificar expresiones algebraicas, etc.
     Las bibliotecas 3D usuales son three.js y babylon.js, pero posiblemente queramos quedarnos en algo práctico y sencillo como plotly.js, algo fácil de manejar con lenguaje natural.
     Si no tenemos el resultado esperado, a veces es necesario en los ajustes, especificar fórmulas, algoritmos u otros detalles técnicos. Sí, en el estado actual de la IA, seguramente tendremos que pedir ajustes.
   - Después de que la actividad interactiva funcione bien (a veces no funcionan a la primera), empezamos a hacer ajustes. Hacer ajustes o pedir ajustes de diseño, de algoritmos, de fórmulas, de eventos de ratón, etc. Hay que ser muy específico, en proyectos un poco más complejos, la IA a veces arregla una cosa y otras cosas desaprecen!
   - Pedimos que el código de estilo este **encapsulado**, para no afectar otros estilos en el momento que incrustemos la actividad interactiva en el sitio web principal, si fuera el caso. No deben quedar varibales globales en el CSS . Si queremos más encapsulamiento, la petición usual es: Dame un "widget" encapsulado, algo como .nombre-widget, con CSS y JS encapsulados.
   - Si en un script hay que arrastrar puntos, hay que pedir también que en el .js se incluya el manejo de *eventos táctiles* para dispositivos moviles. En realidad puede ser incómodo ver estas cosas en el celular sin un "lápiz", pero lo dejamos preparado.

2. Cuando todo funciona bien, pedimos o manualmente separamos el .css, y el .js y luego insertamos en en nuestro archivo .tex lo que está en el `<body> ...<\body>` con el comando `\insertarhtml{}`. Adicionalmente hay que insertar en el preámbulo de nuestro archivo .tex las líneas que deben ir en el `<head> </head>` del HTML.

   Con más detalle:

   - Las etiquetas `<meta ...>`, `\lstinline` ¡script...¿...+ y/o `<link>...` que tenemos que poner en `<head>...<\head>`.

     Para mantener el HTML de la actividad interactiva sin cambios, cuando modificamos el contenido .tex y complamos de nuevo, lo mejor es inyectar estas etiquetas del encabezado head al inicio de la plantilla.tex. El código 26 muestra como lo hacemos:

```
\documentclass[fleqn,oneside]{article}
  \input{Paquetes/WebPreambuloyEntornos.tex}
  \input{Paquetes/ComandosdelUsuario.tex}
  %-- Actividades interactivas
  \ifdefined\HCode % Solo lo compila  Make4ht -> html
  \Configure{HEAD}
  {
   \HCode{<link rel="stylesheet" href="css/tabla-hover-filas.css">}
   \HCode{<link rel="stylesheet" href="css/riemann.css">}
   %...
  }
\fi
```

Código 26: Insertar etiquetas.

- El archivo de estilo de quedar fuertemente **encapsulado**. Este archivo de estilo .css lo colocamos en la carpeta CSS
- El o los scripts .js. Estos archivos los ponemos en la carpeta JS
- El **bloque de código** HTML de cada la actividad interactiva:
  Introducimos este bloque en nuestro .tex para que Make4ht lo inyecte en el HTML de la página web principal. Esto lo hacemos con el comando `\insertarhtml`, como muestra el código 27

```
% Make4ht -> html
\insertarhtml{
<div id="nombre-actividad">
...
</div>
...
<script src="js/nombre-del-script.js"></script>
}
```

Código 27: Insertar HTML.

Si este bloque de código, por alguna razón, lleva texto matemático, se debe incluir en código Mathml. No se debe usar Mathjax para escribir texto de manera interactiva, para no colapsar la página web.

- DeepSeek traduce código 2D de Wolfram Mathematica a JavaScript. El resultado es en general aceptable (si no es código muy complejo) y requiere ajustes adicionales.
- ChatGPT acepta imágenes de partida para generar un script JavaScript.
- Sin duda, si tenemos código en otros lenguajes, se podría ver, en el estado actual de la IA, qué cosas nos genera.

Ejemplo: Un script sencillo en lenguaje natural. Supongamos que queremos implementar un script para visualizar la intersección de dos intervalos. En general, lo mejor es tener una idea gráfica, algo como se muestra en la Figura 17.

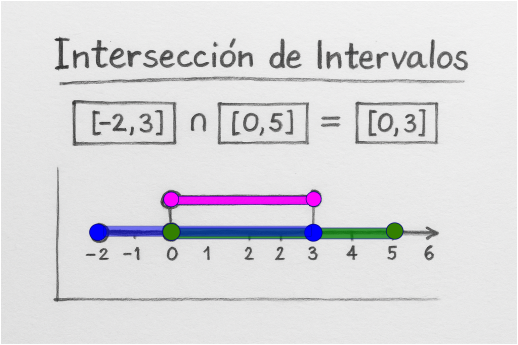


Figura 17: Idea gráfica del script que queremos implementar. Elaboración propia.

Luego debemos pensar en una petición. La IA "sabe" cómo calcular la intersección de dos intervalos, aún así, siempre hay que revisar casos especiales y la lógica general, previniendo errores. Por ejemplo, una primera petición en lenguaje natural podría ser:

> "Crea un HTML completo con un script que muestre dos intervalos arrastrables (azul/-verde) sobre una línea numérica, con discos rellenos/vacíos según sean cerrados/abiertos. Que calcule y muestre su intersección con un segmento (magenta) un poco más arribe, con líneas punteadas hacia abajo. Incluye doble click para cambiar apertura/cierre y redondeo cercano a enteros. *Importante*: El css debe estar encapsulado, o no debe haber ninguna variable ni estilo que afecte afuera de este contenedor."

En este caso, usando Copilot, obtenemos un HTML autocontenido que requiere todavía varios ajustes.

También está la opción de ser mucho más específicos:

Interfaz gráfica:

- Dos intervalos representados como segmentos (azul y verde) sobre una línea numérica (-2 a 6).
- Mostrar tres inputs de texto con la notación (ej: [1, 3]) para la intersección de los dos primeros y el resultado.
- Resultado de la intersección en magenta sobre la línea.

Discos en extremos:

- Rellenos si el intervalo es cerrado ([o]), vacíos si es abierto (] o [).
- Efecto blur al pasar el ratón (solo en los discos azul/verde).

Interacción:

- Arrastrar discos para modificar los extremos.
- Doble clic en discos para alternar entre abierto/cerrado.

Detalles visuales:

- Líneas punteadas magenta desde los extremos de la intersección hasta el segmento inferior.

Encapsulamiento: Dame un widget encapsulado bajo .intintervalos-widget, con CSS y JS encapsulados. No debe haber ninguna variable ni estilo que afecte afuera de ese contenedor.

En este caso, usando nuevamente Copilot, obtenemos un HTML autocontenido más completo.

Después de varios ajustes (peticiones adicionales) el script ya activo se muestra en la subsección 6.1.

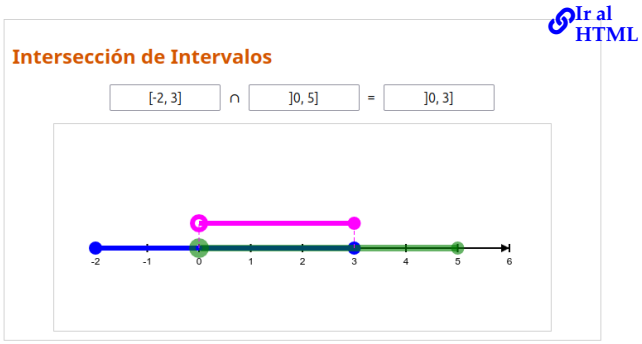


Figura 18: El script en el HTML. Elaboración propia.

Luego, lo recomendable es revisar casos especiales y la lógica en el código del script.

En este caso la IA nos entregó una petición adicional: Los archivos .css, .js y .html, además de una petición especial: que alistara el código `\insertarhtml{...}` para nuestro plantilla.tex.

6. Ejemplos de actividades interactivas

En esta sección aparecen varios ejemplos, expuestos de manera suscinta, con aplicaciones interactivas. En todos estas aplicaciones se usó IA (mayormente Copilot, ChatGPT y DeepSeek), para asistir en la implementación de los scripts interactivos que aparecen más adelante.

No todos los scripts se obtuvieron con peticiones en lenguaje natural, más bien usando a la IA como asistente.

Este documento es acerca de cosas de implementación, por eso no se expone la parte complicada de diseñar la actividad interactiva.

6.1. Intersección de Intervalos

Recordemos que $A \cap B$ (A intersección B) es el conjunto de elementos comunes al conjunto A y al conjunto B.

Dado que los intervalos representan conjuntos, es posible hallar sus intersecciones y uniones. Las representaciones gráficas son útiles en este proceso.

Actividad Interactiva

¿Cómo hallar intersecciones y uniones de dos intervalos?

1. Represente cada intervalo en una recta numérica.

2. Para hallar la intersección de dos intervalos, podemos proceder así:
   - En la misma recta numérica, represento ambos intervalos
   - Toma la porción de la recta numérica que las dos representaciones tienen en común
3. Haga click en los extremos del intervalo para cambiar de abierto a cerrado o viceversa
4. Seleccione y arrastre un extremo de un intervalo

Para insertar el HTML en el sitio web se usó el código 28.

```
% Make4ht -> html
% En el preámbulo se debe agregar la ruta del .css
% \ifdefined\HCode\Configure{HEAD}{
% \HCode{<link rel="stylesheet" href="css/Rinterseccionintervalos.css">}
%    %...
%  }
%\fi
% En la carpeta "css" agregar  interseccionintervalos.css
% En la carpeta "js" agregar  interseccionintervalos.js
\insertarhtml{
<div class="intervalos-container">
  <div class="intervalos-contenedor">
    <h2>Intersección de Intervalos</h2>
    <div class="intervalos-resultado">
      <input type="text" id="intervalos-input1" readonly />
      <span class="intervalos-op">∩</span>
      <input type="text" id="intervalos-input2" readonly />
      <span class="intervalos-op">=</span>
      <input type="text" id="intervalos-resultado" readonly />
    </div>
    <div class="intervalos-canvas-container">
      <canvas id="intervalos-canvas"></canvas>
    </div>
  </div>
</div>
 <script src="js/Rinterseccionintervalos.js"></script>
}
```

Código 28: Intersección de intervalos.

Una imagen del script interactivo, que aparece en el HTML, se ve la Figura 19.

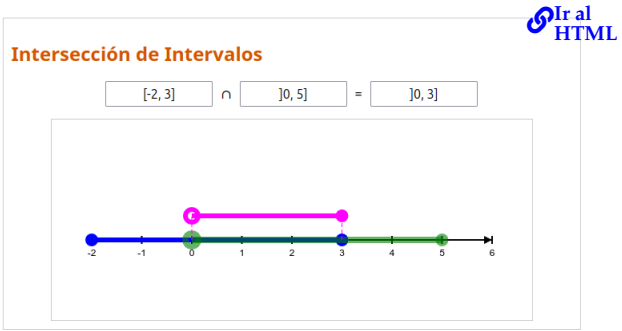


Figura 19: El script en plantilla.html. Elaboración propia.

Script de ejercicios sobre intersección de intervalos.

Actividad Interactiva

En el script que sigue, el botón "Generar Intervalos", genera dos intervalos con extremos aleatorios. Su trabajo consiste en cálcular la intersección, si hubiera, y realizar la representación gráfica. Al final, puede ver la respuesta presionando el botón Respuesta.

El script interactivo que aparece en el HTML se ve la Figura 20.

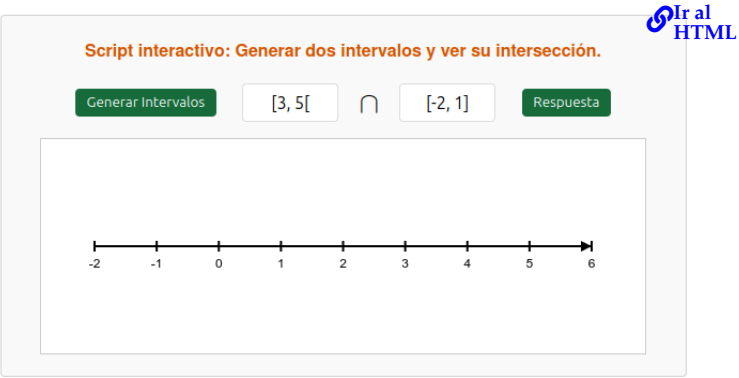


Figura 20: El script en plantilla.html. Elaboración propia.

Implementación: Esta basada en una petición detallada a la IA, especificando lo que contiene la primera fila y la zona gráfica.

El código 29 muestra como insertar el HTML.

```
 % En el preámbulo agregar:
 %\ifdefined\HCode\Configure{HEAD}{%...
 % \HCode{<link rel="stylesheet" href="css/Rejerinterseccionintervalos.css">}
%}
%\fi
% En la carpeta "css" agregar  Rejerinterseccionintervalos.css
% En la carpeta "js" agregar  Rejerinterseccionintervalos.js
\insertarhtml{% Make4ht -> html
  <div class="interseccion-caja">
    <div class="titulo">Script interactivo: Genere dos intervalos y visualice su
    intersección.</div>
    <div class="controls">
        <button id="btnGenerarIntersect">Generar Intervalos</button>
        <input type="text" id="int1Intersect" readonly><span>\( \cap \)</span>
        <input type="text" id="int2Intersect" readonly>
        <button id="btnCalcularIntersect">Respuesta</button>
    </div>
    <div id="resultadoInterseccionIntersect" style="text-align:center; margin-
   bottom:5px; font-size:18px;"></div>
    <canvas id="canvasIntersect" width="565" height="200"></canvas>
  </div>
<script src="js/Rejerinterseccionintervalos.js" defer></script>
}
```

Código 29: Ejercicios sobre intersección de intervalos.

6.2. Interpretación geométrica de la derivada

Una interpretación de la derivada de una función de una variable es que mide la tasa (instántanea) de cambio de la variable dependiente respecto a la variable independiente. La derivada de la función $y = f(x)$ en x es un número (la tasa), si el límite existe.

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

Geométricamente, la derivada de f en x es la pendiente de la recta tangente a f en el punto $(x,\, f(x)\,)$. Y es el límite de las pendientes de las "cuerdas" que pasan por $(x, f(x))$ y $(x+h,\, f(x+h))$ cuando $h \to 0$.

Ecuación de la recta tangente en un punto. Si f es derivable en $x = a$, entonces la ecuación de la recta tangente a f en $x = a$ es $y = f'(a)(x - a) + f(a)$.

Actividad Interactiva

Interpretación geómetrica de la derivada En el siguientes script puede arrastrar el punto rojo para cambiar el punto de tangencia y el punto azul, para empezar el proceso de límite

El código 30 muestra la manera de insertar el HTML.

```
% En el preámbulo agregar:
%\ifdefined\HCode\Configure{HEAD}{
%  \HCode{<link rel="stylesheet" href="css/Rintgeomderivada.css">}
%  \HCode{<script src="https://cdnjs.cloudflare.com/ajax/libs/mathjs/11.8.0/math
   .js"></script>}
%}\fi
% En la carpeta css agregar Rintgeomderivada.css
% En la carpeta "js" agregar  Rintgeomderivada.js
\insertarhtml{ % Make4ht -> html
  <div class="contenedor">
        <div class="grafico-container">
            <svg id="grafico"></svg>
        </div>
        <div class="tabla-valores">
            <table>
                <thead>
                    <tr>
                        <th>x</th>
                        <th>x+h</th>
                        <th>(f(x+h) - f(x)) / h</th>
                    </tr>
                </thead>
                <tbody id="tablaDatos"></tbody>
            </table>
        </div>
    </div>
</div> <script src='js/Rintgeomderivada.js'></script>
}
```

Código 30: Interpretación geómetrica de la derivada.

La salida en el HTML sería algo como lo que se ve la Figura 21

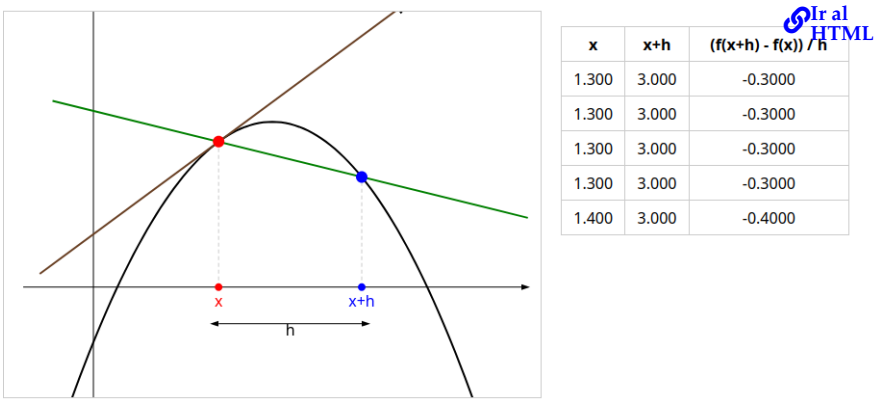


Figura 21: El script en el HTML. Elaboración propia.

Implementación: Esta es un traducción de un archivo .nb (Wofram Mathematica) a JavaScript, con DeepSeek. El archivo está disponible: `Capx-DerivadaDefi.nb`. La traducción requiere algunos ajustes que se pueden hacer a través de peticiones a DeepSeek o manualmente, directamente en el código.

6.3. Un problema de optimización: Ejercicio y respuesta.

El área A del trapecio isósceles inscrito en la parte superior de una circunferencia de ecuación $x^2+y^2 = 16$ se puede escribir en función de la base menor, de longitud $2a$:

$$A(a) = \frac{(B+b)\cdot h}{2} = \frac{(8+2a)\cdot\sqrt{16-a^2}}{2}$$

Si $a = 0$, tendríamos un triángulo de área $8ul^2$. Si nos interesa el valor de a para que el trapecio tenga área máxima, debemos resolver $A'(a) = 0$ y, formalmente, verificar que se trata de un máximo global.

El código 31 muestra como insertar el HTML.

```
% En el preámbulo agregar:
%\ifdefined\HCode\Configure{HEAD}{
%   ...
%  \HCode{<script src="https://d3js.org/d3.v7.min.js"></script>}
%  \HCode{<link rel="stylesheet" href="css/Roptimizaciontrapecio.css">}
%}
%\fi
% En la carpeta css agregar Roptimizaciontrapecio.css
% En la carpeta "js" agregar  Roptimizaciontrapecio.js
\insertarhtml{% No dejar renglones en blanco
    <div class="opt-trap-container">
    ...
     <script src="js/Roptimizaciontrapecio.js" defer></script>
    </div>
}
```

Código 31: Problema de optimización.

En el HTML tendríamos algo como lo que muestra la Figura 22.

Figura 22: Script en plantilla.html. Elaboración propia.

Ahora usamos el script para un ejercicio, con un botón "Ver/Ocultar la respuesta". Recordemos, de la subsección 2.10, que `\hejersol{}{}` es solo para HTML y que `\ejersol{}{}` lo compilan tanto PADFLaTeX como Mak4ht (lo convierte en `\hejersol{}{}`). Pero en este caso, el enunciado de los ejercicios no es igual en ambos contextos, por eso los separamos.

El código 32 muestra como insertar el HTML.

```
\hejersol{
   % Solo Make4ht -> html: Pregunta con botón ver/ocultar
   Hallar las dimensiones del trapecio isósceles de mayor área que puede
   inscribirse en un semicírculo de radio $4$.
  ...
 }{% respuesta
   La base mayor 8 unid, la menor 4 unid y la altura $2\sqrt{3}$ unid.
}
\solopdf{% PDFLaTeX -> pdf
    \ejersol{
    Hallar las dimensiones del trapecio isósceles de mayor área que puede
   inscribirse
    en un semicírculo de radio $4$.
  \begin{center}
    \includegraphics[width=0.45\linewidth]{images/fig8-appderivadas.pdf}
    \captionof{figure}{Trapecio isósceles inscrito}\label{trapecioisosceles}
    \end{center}
    }{La base mayor 8 unid, la menor 4 unid y la altura $2\sqrt{3}$ unid.} %
   ejersol
  } %  \printsolutions % PDFLaTeX, soluciones  final del documento.
```

Código 32: Un ejercicio.

Ejercicio 6.1

Hallar las dimensiones del trapecio isósceles de mayor área que puede inscribirse en un semicírculo de radio 4. Ver Figura 23.

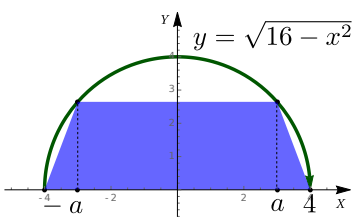


Figura 23: Trapecio isósceles inscrito. Elaboración propia.

Implementación: Esta es un traducción de un archivo .nb (Wofram Mathematica) a JavaScript, con DeepSeek. El archivo está disponible: `Capx-EjemplOptimizacion.nb`. La traducción requiere algunos ajustes que se pueden hacer a través de peticiones a DeepSeek.

6.4. Sumas de Riemann con particiones regulares

Históricamente, la integral de Riemann se define como un "límite" sobre todas las posibles particiones de un intervalo, pero en realidad solo necesitamos definir la integral sobre particiones regulares (Denlinger, 2010).

Como consecuencia, si f es integrable, calculamos la integral con una escogencia de los ξ_i fijos, sobre una familia de particiones regulares (Teorema 3).

Teorema 3.

Sea f integrable en $[a,b]$ y $\mathcal{P}_n = \{x_0, x_1, ..., x_n\}$ con $x_i = a + i\,\frac{b-a}{n}$, $i = 0, 1, 2..., n$. Entonces

$$\int_a^b f(x)\,dx = \lim_{n\to\infty} \sum_{i=0}^{n} f(\xi_i)\frac{b-a}{n}$$

sin importar la escogencia (fija) de los $\xi_i \in [x_i, x_{i+1}]$

El recíproco es falso, si $\lim_{n\to\infty} \sum_{i=0}^{n} f(\xi_i)\frac{b-a}{n}$ existe para una escogencia de los ξ_i fijos, no signifca que f sea integrable.

Ejemplo 3.

$f(x) = x^2 - 4x + 5$ es integrable en $[1,4]$ porque es continua en este intervalo. Por tanto, según el Teorema 3

$$\int_1^4 f(x)dx = 6$$

pues, tomando $\xi_i = 1 + \dfrac{3i}{n}$,

$$\begin{aligned}\int_1^4 f(x)dx &= \sum_{i=1}^{n} f(\xi_i)\frac{3}{n} \\ &= \frac{3}{n}\sum_{i=1}^{n}\left[2 - \frac{6i}{n} + \frac{9i^2}{n^2}\right] \\ &\quad \text{y, usando las fórmulas para estas sumas de } n \text{ términos,} \\ &= 6\end{aligned}$$

Actividad Interactiva

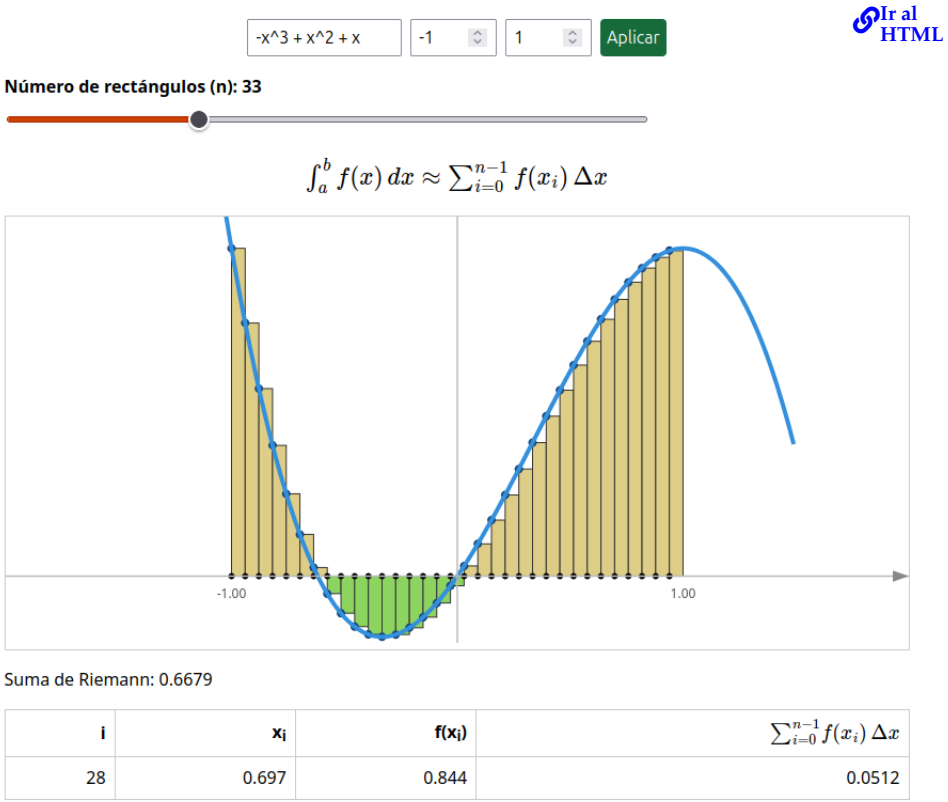
Si f es integrable, entonces podemos tomar $\xi = x_{i-1}$

El código 33 muestra como insertar el HTML.

```
% Make4ht -> html
% En el preámbulo se debe agregar la ruta del riemann.css
% \ifdefined\HCode\Configure{HEAD}{
%\HCode{<link rel="stylesheet" href="css/riemann.css">}  }
%\fi
% En la carpeta css agregar riemann.css
% En la carpeta "js" agregar  riemann.js
\insertarhtml{
  <div class="riemann-widget">
   <h1>Integral de Riemann</h1>
    ...
  </div>
  <script src="js/riemann.js"></script>
}
```

Código 33: Integral de Riemann.

En el HTML aparece el script, como se muestra en la Figura 24.



i	x_i	$f(x_i)$	$\sum_{i=0}^{n-1} f(x_i)\,\Delta x$
28	0.697	0.844	0.0512

Figura 24: El script en plantilla.html. Elaboración propia.

Implementación: Se hizo una petición detallada a DeepSeek. Se especificó que la evaluación de la función se debía hacer con math.js (sino la IA trata de hacer un "parser" demasiado simple) y que no usara "charts" para los rectángulos. Finalmente, cuando el HTML funcionaba correctamente, se pidió a la IA que encapsulara y luego separara riemann.css, también riemann.js y el .html para extraer lo que debemos inyectar en el `<header>...<header>` y para extraer lo que está en `<body>...</body>`, e inyectarlo con `\insertarhtml{}`

6.5. Plano Tangente

Rectas Tangentes y Plano Tangente.

- Si la superficie S tiene ecuación $G(x,y,z) = 0$ con G diferenciable, el plano tangente en $P \in S$ tiene ecuación cartesiana

$$G_x(P)\,x \;+\; G_y(P)\,y \;+\; G_z(P)\,z = \boldsymbol{\nabla G}(P)\cdot P$$

- Si S tiene ecuación $z = f(x,y)$ con f diferenciable, entonces $G(x,y,z) = z - f(x,y)$ y $\boldsymbol{\nabla G}(P) = (-z_x(P),\,-z_y(P),\,1)$, por tanto el plano tangente en $P = (p_0,p_1,p_2) \in S$ tiene ecuación cartesiana

$$\begin{aligned} G_x(P)\,x \;+\; G_y(P)\,y \;+\; G_z(P)\,z \;&=\; \boldsymbol{\nabla G}(P)\cdot P \\ z_x(P)\,x \;+\; z_y(P)\,y \;+\; z \;&=\; (-z_x(P),\,-z_y(P),\,1)\cdot(p_0,p_1,p_2) \end{aligned}$$

  Es decir,

$$z_x(P)\,(x-p_0) + z_y(P)\,(y-p_1) \;=\; z - p_2$$

Rectas tangentes en dirección de $f_x(P)$ y $f_y(P)$.

Si S tiene ecuación $z = f(x,y)$ con f diferenciable, entonces una ecuación paramétrica de la recta tangente a $P = (p_0,p_1,p_2)$ en la dirección del eje X se obtiene tomando $\boldsymbol{v} = (1,0)$ y una ecuación paramétrica de la recta tangente a P en la dirección del eje Y se obtiene tomando $\boldsymbol{v} = (0,1)$

- $\boldsymbol{L_X}(t) = P + t\cdot(1,\,0,\,z_x(P))$ o $\boldsymbol{L_X}(x) = (x, p_1, p_2 + (x-x_0)f_x(p_0,\,p_1))$
- $\boldsymbol{L_Y}(t) = P + t\cdot(0,\,1,\,z_y(P))$ o $\boldsymbol{L_y}(x) = (p_0, y, p_2 + (y-p_1)f_y(p_0,\,p_1))$
- La recta tangente en P, en dirección de $\boldsymbol{v} \in \mathbb{R}^2$ es

$$\boldsymbol{L_v}(t) = P + t\cdot(v_0,\;v_1,\;\boldsymbol{\nabla z}(P)\cdot(v_0,v_1))$$

o, en términos de la derivada direccional, $\boldsymbol{L_v}(t) = P + t\cdot\left(\frac{v_0}{||\boldsymbol{v}||},\;\frac{v_1}{||\boldsymbol{v}||},\;D_{\boldsymbol{v}}z(P)\right)$

En el siguiente script tenemos la superficie $z = 1-(x-1)^2+(y-1)^2$. Arrastrando el punto en la columna a la derecha podemos mover el punto P en la superficie y el plano tangente. Vamos a observar

El código 34 muestra como insertar el HTML.

```
% Make4ht -> html
% En el preámbulo se debe agregar la ruta del planotangente.css y plotly
% \ifdefined\HCode\Configure{HEAD}{%...
%  \HCode{<script src="https://cdn.plot.ly/plotly-2.32.0.min.js"></script>}
%  \HCode{<link rel="stylesheet" href="css/planotangente.css">}}
%  }
```

```
%\fi
% En la carpeta css agregar planotangente.css
% En la carpeta "js" agregar  planotangente.js
\insertarhtml{
<div class="planotangente-widget">
   <div id="plot3d"></div>
   <div id="plot2d-container">
   <div id="controls">
   <label><input type="checkbox" id="toggleTangents" checked> Mostrar rectas
    tangentes</label>
   </div>
   <div id="coords">Coordenadas: (1.05, 0.68, 0.00)</div>
   <div id="plot2d-wrapper">
  <canvas id="plot2d"></canvas>
  </div></div></div>
<script src="js/planotangente.js" defer></script>
}
```

Código 34: Plano tangente.

En el HTML podremos ver el script interactivo de la Figura 25.

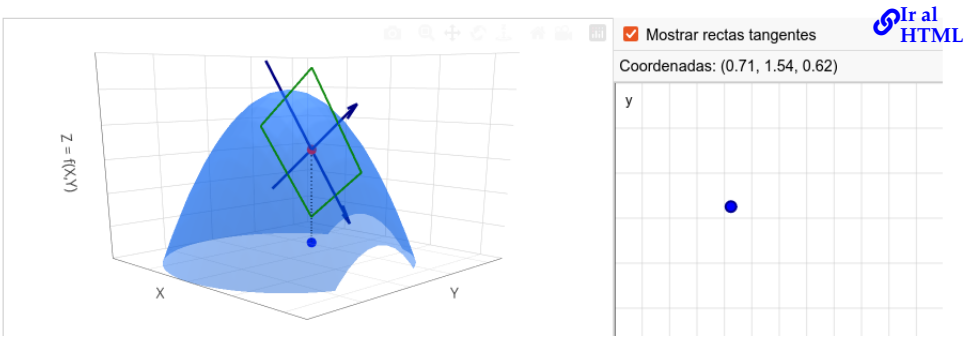


Figura 25: Script en el HTML. Elaboración propia.

Implementación: Se hizo una petición detallada a DeepSeek (Copilot también fue muy eficiente y en menor medida ChatGPT), en este caso hubo que hacer ajustes manuales en el .js para habilitar y deshabilitar cosas y corregir dimensiones. La IA "conoce" las ecuaciones del plano tangente y las ecuaciones de las rectas tangentes solicitadas. Pero debemos especificar que la gráfica de la superficie se implemente con plotly.js y que sea una "constante" (usar three.js o babylon.js requiere demasiados ajustes) y, como plotly.js no ofrece muchos eventos de ratón, solicitamos que el punto $P = (x, y, f(x, y))$ se actualice con las coordenadas (x, y) que provee el punto de arrastre en la segunda columna, que debe ser en JavaScript normal. Es decir, las dos columnas se comunican. Solicitamos encapsulado con .planotangente-widget. Siempre hay que asegurarse que no use variables globales en el CSS , como name {...}, sino .planotangente-widget #name {...}. Finalmente pedimos el .html, el .css y el .js separados. Observe que este script viene con un menú en la parte superior, para trasladar, imprimir, etc.

6.6. Efecto hover en tablas

Queremos aplicar un efecto hover para resaltar o iluminar filas, columnas o elementos de una tabla, cuando el ratón pasa por encima.

Estos efectos se aplican a la etiqueta `<table class="...</table>`, pero Make4ht *no siempre* convierte un entorno tabular en un entorno `<table>...` (esto pasa cuando tabular está en entornos, center, table, minipage, etc.). Además diseños especiales de tablas requieren un manejo especial.

Vamos a implementar un efecto hover en algunos casos.

1. Tablas que Make4ht compila bien.

   En este caso, usamos el diseño de la Tabla 4. El estilo definido ya está en la carpeta CSS y se llama tabla-hover-filas.css. Además ya incluimos el código en el preámbulo de este documento para cargar este estilo. Luego, usamos un entorno tablahoverfilas ya definido en el preámbulo, y envolvemos las tablas con este entorno para envolver el tabular con la etiqueta `<div class='tabla-hover-filas'>`. PDFLaTeX ignora este entorno, es solo para Make4ht.

   El código 35 muestra como insertar el HTML.

```
\begin{tablahoverfilas} %  Make4ht inyecta el efecto :hover en el html
  \begin{table}[h!!!]
   ...
   \begin{tabular}{cc||ccc||ccc||ccc||ccc}
    ...
   \end{tabular}
   \caption{Tabla modelo}\label{tabla-modelo}
  \end{table}
\end{tablahoverfilas}
```

Código 35: Un efecto "hover" en una tabla

Negación		Conjunción			Disyunción			Implicación			Equivale			Ir al HTML
P	$\neg P$	P	Q	$P \wedge Q$	P	Q	$P \vee Q$	P	Q	$P \to Q$	P	Q	$P \leftrightarrow Q$	
V	F	V	V	V	V	V	V	V	V	V	V	V	V	
V	F	V	F	F	V	F	V	V	F	F	V	F	F	
F	V	F	V	F	F	V	V	F	V	V	F	V	F	
F	V	F	F	F	F	F	F	F	F	V	F	F	V	

Tabla 5: Tablas de verdad básicas

2. Tablas que Make4ht no traduce bien.

   En este caso, podemos usar algún generador de tablas "online" y crear algo similar, con código más simple, o también, pedir a la IA un código más simple para la tabla o el código HTML de una tabla equivalente. En este último caso, Make4ht no incluye la referencia de la tabla, porque no la compiló en el archivo .tex, entonces hacemos referencia a esta tabla con

   ```
   \insertarhtml{<a href="\#tabla-modelo">Ver Tabla modelo</a>}
   ```

6.7. Incrustar applets GeoGebra

GeoGebra (GeoGebra, s.f.) es un software interactivo para enseñar y aprender matemáticas mediante construcciones visuales. Aunque su implemetación y/o uso requiere práctica, permite integrar recursos ya creados en páginas web.

Incrustar un applet. Para incrustar applets de Geogebra en nuestra página HTML, seguimos las instrucciones que se detallan en Team (s.f.). En la página de Recursos podría ser que haya algo que nos interese, en ese caso abrimos el applet, y a la derecha del nombre del autor hacemos click en los tres puntos y, en el menú que se abre, hacemos click en "Detalles"; en la ventana que se abre, hacemos click en "Compartir" y damos click en "<\>Incrustar". Copiamos el `<iframe....></iframe>` con todo su contenido y eso es lo que vamos a incrustar.

Para incrustar el código HTML del `<iframe...`, usamos un script toggleVentana.js (que ya está en la carpeta JS y que ya se carga con este el archivo plantilla.tex). Produce un botón que *abre u oculta* la ventana en la que se despliega el applet de Geogebra. Pero debemos **numerar** bien la ventana en el data-tarjet="ventanak" y en el identificador id="ventanak", como se muestra en el código 36.

```
<button class="toggle-ventana-btn" data-target="ventanak">
    Ver/Ocultar nombre </button>
<div id="ventanak" class="ventana-container" style="display: none;">
```

Código 36: Botón para applet de Geogebra

Cambiamos ventanak por "ventana1", "ventana2", etc. porque cada applet ocupa un contenedor diferente.

6.7.1. Ejemplos.

Distribución Normal. El código 37 muestra como insertar un applet de Geogebra para visualizar el cálculo de áreas en la Distribución Normal.

```
\insertarhtml{% ventana1
    <button class="toggle-ventana-btn" data-target="ventana1">Ver/Ocultar
   Distribución Normal</button>
    <div id="ventana1" class="ventana-container" style="display: none;">
     <div style="width:100\%; position:relative; padding-bottom:44.58\%;">
        <iframe scrolling="no" title="Distribución Normal" src="https://www.
   geogebra.org/material/iframe/id/rZ7NHCsS/width/1366/height/609/border/888888/
   sfsb/true/smb/false/stb/false/stbh/false/ai/false/asb/false/sri/false/rc/
   false/ld/false/sdz/false/ctl/false" width="900px" height="402px" style="
   border:0px;"></iframe>
</div></div>}
```

Código 37: Distribución Normal

Las dimensiones que nos dan son de `width="1366px" height="609px"`. Pero para nuestro sitio parece mejor (manteniendo las proporciones) `width="900px" height="402px"`.

En el documento plantilla.html aparece el botón para abrir o cerrar la ventana, con el applet de Geogebra, como se muestra en la Figura 26.

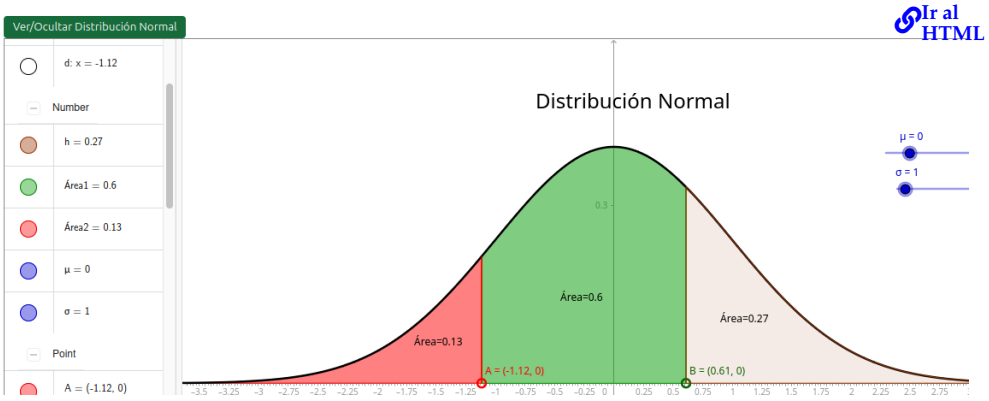


Figura 26: Script en plantilla.html. Elaboración propia.

Tabla de Frecuencias. Este es otro ejemplo que tomamos de la página de Recursos. Es un applet de Geogebra con una actividad interactiva con tablas de frecuencias para el tema de Estadística.

El código 38 muestra como insertar en el HTML.

```
\insertarhtml{ % Numerar ventana!. Vamos por la "ventana2"
    <button class="toggle-ventana-btn" data-target="ventana2">Ver/Ocultar Tabla
   Frecuencias</button>
    <div id="ventana2" class="ventana-container" style="display: none;">
     <iframe scrolling="no" title="Tabla de frecuencias" src="https://www.
   geogebra.org/material/iframe/id/sxG0ihm3/width/900/height/400/border/888888/
   sfsb/true/smb/false/stb/false/stbh/false/ai/false/asb/false/sri/false/rc/true
   /ld/false/sdz/false/ctl/false" width="900px" height="400px" style="border:0px
   ;"> </iframe>
    </div>}
```

Código 38: Tabla de frecuencias.

6.8. Incrustar videos de Youtube

De manera similar a como incrustamos applets de Geogebra, también incrustamos videos de YouTube en la salida HTML de nuestro documento .tex. Usamos el mismo script toggleVentana.js y actualizamos la numeración de la ventana.

En el video de YouTube que nos interesa, hacemos click en "Compartir" y hacemos click en "Incrustar" y tomamos el código `<iframe ...>...</iframe>` y lo insertamos en nuestro HTML.

Ejemplo.

El determinante — Esencia del álgebra lineal, capítulo 5. El canal 3Blue1Brown tiene como meta usar animaciones para aclarar y motivar temas complejos. hemos elegido para este ejemplo, el capítulo 5 sobre determinantes.

El código 39 muestra como insertar el video en el HTML.

```
% Numerar ventana!. Vamos por la "ventana3"
\insertarhtml{
<button class="toggle-ventana-btn" data-target="ventana3">Ver/Ocultar Tabla
   Frecuencias</button>
<div id="ventana3" class="ventana-container" style="display: none;">
<iframe width="560" height="315" src="https://www.youtube.com/embed/yt3eoYvGel0?
   si=sq7SgYaOXnHpZU5j" title="YouTube video player" frameborder="0" allow="
   accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture
   -in-picture; web-share" referrerpolicy="strict-origin-when-cross-origin"
   allowfullscreen></iframe>
</div>
}
```

Código 39: El determinante: Video de YouTube.

7. Personalizar un documento con un archivo de configuración

Tex4ht (implementado por Eitan Gurari (1947–2009)) es un conjunto de scripts que, después de la compilación con htlatex, "traduce" el archivo dvi. Este archivo dvi contiene toda la información sobre

la disposición del texto, gráficos y fórmulas matemáticas en un formato compacto. Esta información se usa para generar documentos HTML, ODT, EPUB, etc.

Make4ht es un sistema de compilación (implementado por Michal Hoftich) para Tex4ht. Make4ht es una herramienta diseñada principalmente para convertir documentos LaTeX en HTML (y otros formatos), enfocándose en la estructura del contenido (texto, tablas, imágenes, fórmulas matemáticas, etc.), pero *no se encarga del diseño web avanzado*. Esto significa que, por defecto, Make4ht no genera estilos CSS complejos *ni diseños responsivos* modernos.

Pero Make4ht permite controlar y configurar el diseño web. Usualmente lo que hacemos es que con un archivo de configuración config.cfg controlamos el diseño HTML de los entornos y otros componentes del código LaTeX.

Esta sección es una aplicación de los elementos de configuración que se pueden leer en Hoftich (2024) y en respuestas a preguntas relacionadas con Make4ht en TeX Stack Exchange. La plantilla plantilla.tex está configurada apropiadamente para que compile bien con Make4ht (parece que no tiene mucho sentido compilar con Make4ht un documento .tex sin implementar la configuración adecuada, excepto que sea un documento muy sencillo).

La IA puede ser útil para que nos ayude generando código tedioso y recomendaciones, pero en el estado actual, no parece estar muy entrenada en Make4ht, por lo que la manera práctica de usar la IA, es dándole instrucciones muy específicas si es posble con modelos de código, para que genere código LaTeX apropiado. Es posible que haya que hacer ajutes para evitar errores de compilación que "rompen" el HTML.

7.1. Algunos comandos de configuración

Hay muchos comandos de configuración, pero vamos a describir y ver ejemplos de aplicación de algunos comandos frecuentes en la configuración de nuestro documento plantilla.tex.

1. `\HCode{output format markup}` se usa para insertar, en este caso HTML "puro".

   Este comando sólo permite la "expansión de macros", antes de enviar su contenido a la salida. En él se puede introducir la instrucción `\Hnewline` para solicitar saltos de línea.

2. `\Css{...}` se usa para introducir estilo CSS desde el archivo de configuración. Como inicialmente compilamos con hlatex (una variante de LaTeX que inserta marcadores HTML), debemos escapar # con `\#` y los porcentajes como font-size: 90 %; debemos "escaparlos" como `font-size : 90\%;`.

3. `\Configure{name}{arg 1}...{arg n}` Configura el comportamiento específico de un elemento en el formato de salida.

   Este comando declara el código que se inserta en los hooks ("ganchos") relacionados con la configuración del "name". Es posible definir hasta nueve hooks, por lo que el número de argumentos es variable.

   Cuando TeX4ht está convirtiendo el documento LaTeX paso a paso, puede haber ciertos "puntos de enganche" o "hooks" donde se puede inyectar código HTML, CSS, JavaScript o incluso instrucciones de procesamiento para TeX4ht. El comando `\Configure` es la forma de decirle a TeX4ht: "Cuando llegues a este punto específico (name), ejecuta estos argumentos."

   Desglose del comando `\Configure{name}`

   - "name" es el identificador de la configuración o del "hook" (punto de enganche) donde se quiere insertar el código. TeX4ht tiene cientos de estos "hooks" predefinidos para casi todo:

encabezados, párrafos, listas, imágenes, entornos específicos, etc. Cada "name" representa un momento o un elemento en el HTML donde se puede intervenir.

- {arg 1}...{arg n} son los argumentos que contienen el código o las instrucciones que TeX4ht insertará en el hook especificado por "name". TeX4ht es flexible con el orden.
- arg 1: Se inserta antes del inicio del elemento HTML.
- arg 2: Se inserta después del inicio del elemento HTML (justo después de la etiqueta de apertura).
- arg 3: Se inserta antes del final del elemento HTML (justo antes de la etiqueta de cierre).
- arg 4: Se inserta después del final del elemento HTML.
- arg 5: Puede ser para atributos HTML (como id, class, style, etc.).
- arg 6 hasta arg 9: Son para usos más específicos o personalizados, dependiendo del "hook". No todos los "hooks" usan los nueve.

Ejemplo. En el código 40 configuramos un párrafo personalizado llamado "mi-parrafo". Esta personalización se aplicaría a la salida HTML de todos los párrafos del documento .tex. Este código usualmente lo ponemos en el archivo config.cfg (ver más abajo).

```
\Configure{p}                          % "p" es el hook: <p></p>
  {\HCode{<div class="mi-parrafo">}}   % arg 1: Abre el div
  {\HCode{</div>}}                     % arg 2: Cierra el div
  % Estilo Css, debemos "escapar" #, por eso ponemos \#
  \Css{.mi-parrafo {
  background-color: \#f0f8ff; /* Azul claro */
  border: 1px solid \#add8e6; /* Borde azul */
  padding: 10px;
  margin-bottom: 15px;
  font-family: sans-serif;
  color: \#333;}
```

Código 40: Configuración de párrafos.

4. \ConfigureEnv{<environmentname>}{before env}{after env}. Personaliza cómo se "renderiza" un entorno específico, definiendo elementos antes y después del entorno y la lista. Este comando puede ser usado para insertar código pertinente en cualquier entorno.

   Cuando definimos entornos con \newenvironment, hay que tener cuidado con las líneas en blanco en el contenido del entorno porque podría provocar que una etiqueta <p> abra pero no cierre del todo o cierre en el lugar equivocado, lo que nos "rompe" el HTML. Entonces debemos "atajar" este problema usando \ConfigureEnv.

   Ejemplo. Supongamos que definimos un entorno test como muestra el código 41.

```
\newenvironment{test}{\itshape}{}
\begin{test}
 Mundo

 Otro texto
\end{test}
```

Código 41: Entorno problemático.

Si solo configuramos el entorno envolviéndolo en un contenedor <div...>

\ConfigureEnv{test}{\HCode{ }}{\HCode{ }},

produciría HTML inválido (nos da un error de compilación), como se muestra en el código 42.

```
<p class="indent" >
<div class="test"> <spanclass="cmti-12">Mundo</span></p>
%
<p class="indent" > % abre <p>
<span  class="cmti-12">Otro texto</span>
</div>
</p>                 % pero cierra después del </div>!!
```

Código 42: HTML problemático.

Tenemos que cerrar `</p>` antes del `</div>`. La configuración correcta es (y este es un modelo que debemos indicarle a al IA, si la estamos usando) la que se muestra en el código 43.

```
\ConfigureEnv{test}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="test">}\par
   }
{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}\par}{}{}
```

Código 43: Configuración correcta.

El desglose es como sigue: El comando `\EndP` inserta la etiqueta de cierre `</p>`. Sin embargo, el nuevo párrafo podría comenzar antes de `<div>` si estamos en *modo vertical*, por lo que también debemos usar el comando `\ifvmode\IgnorePar\fi`. Esto evitará que el nuevo párrafo comience demasiado pronto. Sin embargo, queremos comenzar un nuevo párrafo después de `<div>`, por lo que debemos usar `\par` explícito después de `\HCode`. Se requiere un tratamiento de párrafo similar para cerrar el entorno, por eso el código sigue con `\ifvmode\IgnorePar\fi \EndP\HCode{</div>}\par`. El HTML resultante con esta configuración es correcto.

5. **Ejemplo.** El entorno `definicion` de la plantilla.tex, en el que usamos el paquete `mdframed`, está bien definido para PDFLaTeX, pero la compilación con Make4ht deja un código CSS muy frágil. Así que tenemos que configurarlo de tal manera que no nos de problemas de compilación y que prevalezca sobre el código que inserta Make4ht.

   a) El código 44 muestra como reconfigurar el entorno mdframed.

```
% Elimina la configuración automática de mdframed que inserta Make4ht
\Configure{mdframed}{\IgnorePar\EndP}{\IgnorePar\EndP}{}{}
% Eliminamos los márgenes que vienen por defecto
\Css{.mdframed {
  margin-top: 0pt !important;
  padding-top: 0pt !important;
}}
% Preparar para la configuración personalizada usando \ConfigureEnv con
   \Css
\ConfigureEnv{mdframed}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="
   mdframed">}\par}
{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}\par}{}{}
```

Código 44: Reconfigurar `mdframed`.

En esta configuración de mdframed, se agregan etiquetas `<div>...</div>` alrededor del contenido del cuerpo del documento. Los comandos `\ifvmode\IgnorePar\fi` evitarán la inserción de la etiqueta `<p>` antes de `<div>` si estamos en el modo vertical de TeX. `\EndP` cierra el párrafo abierto, si está abierto. Los comandos `\par\ShowPar` inician un nuevo párrafo después de la etiqueta `<div>` insertada. A veces es necesario iniciar los párrafos explícitamente.

b) Configurar el entorno \begin{definition}...\end{definition}

Podemos agregar diseño con CSS sin que el CSS original intervenga. La idea es que se parezca al diseño del entorno en el plantilla.tex. Para empezar bebemos definir bordes, colores, etc. Más adelante debemos definir otras cosas, como veremos. Recordar que estamos compilando con el motor LaTeX, por lo tanto tenemos que "escapar" caracteres especiales o comentar. Parte del código necesario se muestra en 45

```
\ConfigureEnv{definicion}
{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="definicion">}\par}
{\ifvmode\IgnorePar\fi\EndP\HCode{</div>}\par}
{}{}
\Css{
%/* --- Estilo base para "definicion", bordes, etc. --- */
.definicion {
                background-color: rgb(251, 251, 250);  %/* gris magenta
   */
                border-left: 4pt solid rgb(255, 20, 147);  %/* magenta
   */
                % top right bottom left;
                margin: 20pt 0;
                padding: 0pt 4pt 0 4pt; %/* padding-bottom reducido a 0
   */
                overflow: auto; %/* Contiene floats */
                clear: both; %/* Evita solapamientos */
        }
%/* Título de la definición */
.definicion-title {
                color: rgb(255, 20, 147);
                font-weight: bold;
                margin-bottom: 0.5em;
                display: block;
        }
```

Código 45: Definición.

Este entorno se convierte en una "plantilla" para todos los otros entornos. Adicionalmente, como se puede ver el el archivo `config.cfg`, debemos configurar en el CSS, que los entornos reciban los entornos minipage de manera correcta.

7.2. Configuración en el preámbulo

1. La opción [spanish][babel] esta bien para PDFLaTeX, pero para tradución al HTML causa muchos problemas porque redefine muchos caracteres (igual que algunos paquetes de fuentes). Usamos \HCode para indicar qué configuración usará PDFLaTeX y cuál Make4ht, como se muestra en el código 46.

```
\ifdefined\HCode % Make4ht -> html (PDFLaTeX ignora)
\usepackage[spanish,english]{babel}
\else            % PDFLaTeX -> pdf (Make4ht ignora)
    \usepackage[english, spanish, es-noquoting]{babel}
    \usepackage[autostyle, spanish = mexican]{csquotes}
    \MakeOuterQuote{"}
\fi
\usepackage[T1]{fontenc}
```

```
\usepackage[utf8]{inputenc} % LaTeX clásico
```

Código 46: Configuración del idioma.

2. Algunos comandos personalizados para Make4ht.

   - Un efectohover en tablas. El entorno `tablahoverfilas` definido en el preámbulo, hace que Make4ht genere el contenedor `<div class="tabla-hover-filas">`. A este contenedor se le aplica el comportamiento hover: Iluminar filas cuando el cursor pasa. Esto se implementó en la configuración CSS. El código se muestra en 47.

```
\newenvironment{tablahoverfilas}
  {%
    \ifdefined\HCode
      \ifvmode\IgnorePar\fi\EndP
      \HCode{<div class="tabla-hover-filas">}%
      \par
    \fi
  }
  {%
    \ifdefined\HCode
      \ifvmode\IgnorePar\fi\EndP
      \HCode{</div>}%
      \par
    \fi
  }
% \rowcolor{rowgray} tapa el hover, \rowgris para css y da \rowcolor{
   rowgray} para pdf
\ifdefined\HCode
  \newcommand{\rowgris}{\HCode{<tr class="rowgray">}}
\else
  \newcommand{\rowgris}{\rowcolor{rowgray}}
\fi
```

Código 47: Efecto `hover`.

   - Make4ht ignora `\bigskip`, `\medskip`, `\smallskip`, así que lo configuramos para que Make4ht lo interprete con elementos equivalentes en HTML y que PDFLaTeX lo interprete como código LaTeX usual. Las líneas horizontales no quedan bien en HTML, por eso definimos el equivalente `\linea`. El código se muestra en 48.

```
\newcommand{\linea}{\textcolor{lightgray}{\rule{\linewidth}{0.4pt}}\par}
\ifdefined\HCode
\def\linea{\HCode{<hr style="height:1px; border:none; background-color:\
   #ccc;
            margin:20px 0;">}}
\def\bigskip{\HCode{<div style="margin-top:2em;"></div>}}
\def\medskip{\HCode{<div style="margin-top:1.5em;"></div>}}
\def\smallskip{\HCode{<div style="margin-top:1em;"></div>}}
\fi
```

Código 48: Líneas y espacio vertical.

   - Comandos `\solopdf{#1}`, `\solomk` para separar código LaTeX exclusivo para PDF y código LaTeX exclusivo para HTML. `\insertarhtml{}` inserta HTML “puro”, se usa para incrustar Javascript, iframe, object etc. El código se muestra en 49.

```
\ifdefined\HCode
  \long\def\solopdf#1{\protect}  % ignorado por make4ht
  \long\def\solomk#1{\protect#1} % procesado en make4ht
\else
  \long\def\solopdf#1{\protect#1} % procesado en PDF
  \long\def\solomk#1{\protect}    % ignorado por PDF
\fi
```

Código 49: Solo PDFLaTeX o solo Mak4ht.

- La parte interactiva del documento plantilla.tex requiere cargar todos los archivos .css y algunos scripts específicos. Eso lo hacemos antes de `\begin{document}`, como se muestra en el código 50.

```
\documentclass[fleqn,oneside]{article}
\input{Paquetes/WebPreambuloyEntornos.tex}
\input{Paquetes/ComandosdelUsuario.tex}
%% Make4ht -> html:   Inserta en <head>...</head>
\ifdefined\HCode\Configure{HEAD}{
  \HCode{<link rel="stylesheet" href="css/tabla-hover-filas.css">}
  \HCode{<link rel="stylesheet" href="css/riemann.css">}
  \HCode{<link rel="stylesheet" href="css/Rejerinterseccionintervalos.
   css">}
  \HCode{<link rel="stylesheet" href="css/Rinterseccionintervalos.css">}
  \HCode{<link href="css/intervalos.css" rel="stylesheet" type="text/css
   ">}
  \HCode{<link rel="stylesheet" href="css/Rintgeomderivada.css">}
  \HCode{<script src="https://cdnjs.cloudflare.com/ajax/libs/mathjs
   /11.8.0/math.js"></script>}
  \HCode{<link rel="stylesheet" href="css/ejerciciosolucion.css">}
  \HCode{<script src="https://d3js.org/d3.v7.min.js"></script>}
  \HCode{<link rel="stylesheet" href="css/Roptimizaciontrapecio.css">}
  \HCode{<script src="https://cdn.plot.ly/plotly-2.32.0.min.js"></script
   >}
  \HCode{<link rel="stylesheet" href="css/planotangente.css">}
  \HCode{<script src="js/toggleVentana.js" defer></script>}
%\HCode{<link rel="preconnect" href="https://fonts.googleapis.com">}
  }
\fi
...
\begin{document}
```

Código 50: Incrustar en el `head`.

7.3. Archivo (personalizado) de configuración `config.cfg`

El archivo de configuración config.cfg tiene la forma que se muestra en el código 51.

```
\Preamble{xhtml}
%...Configure, ConfigureEnv, etc.
\begin{document}
%...insertar en header (posiblemente no cosas de uso ocasional)
\EndPreamble
```

Código 51: Archivo de configuración

En el documento plantilla.tex configuramos el comportamiento de todos los entornos. El documento plantilla.tex usa el paquete mdframed (tcolorbox inserta las "label" internamente y complica las cosas). Make4ht traduce los entornos mdframe de manera simplificada, con solo cosas básicas. Pero el archivo de estilo plantilla.css gobierna todo, entonces tenemos que anular "ese poder" y modificarlo.

Siempre que configuramos algo, hay que evitar que el HTML se "rompa" (evitar que el DOM se rompa) digamos por etiquetas que abren, pero no cierran como en `<div><p>Contenido</div>` o que cierran a destiempo, o paquetes que introducen caracteres que corrompen el HTML. También el DOM se puede romper por paquetes incompatibles con Make4ht o entornos matemáticos complejos mal interpretados.

En la compilación con Make4ht, una "advertencia" del tipo:

```
domfilter: DOM parsing of test.html failed:
[WARNING] domfilter: /home/.../ ... nbalanced Tag (/p) [char=   ]
```

significa que la estructura del documento (DOM) se ha roto y la "advertencia" nos dice que una o varias cosas ya no van a salir bien. Lo normal es corregir el problema directamente o hacer modificaciones. El código que sigue es muy preciso, para no romper el DOM.

1. Configurar minipage. Queremos que la traducción de dos minipage sea la usual: Una al lado de la otra, respetando las dimensiones. Muchas cosas en Make4ht, se encuentran dispersas por internet. Y la configuración de minipage dichosamente la podemos encontrar en una respuesta en TeX Stack Exchange. Esta configuración nos permite usar minipage en los entornos que han sido configurados apra ello. El código se muestra en refcodeminipage.

```
 %%--configura minipage y figuras----
\makeatletter
% we must refer to minipage from the css file, because tags are beeing
% written before we know dimensions
\newcount\mini@count
\ExplSyntaxOn
% save original minipage
\let\oldiimini\@iiiminipage
% redefine minipage
\def\@iiiminipage#1#2[#3]#4{%
  % calculate minipage dimensions and save it to the CSS file
  \Css{\#minipage\the\mini@count{width:\fp_eval:n{#4/\textwidth*100}\%;}}%
  \global\advance\mini@count by 1\relax%
  \oldiimini{#1}{#2}[#3]{#4}%
}
\ExplSyntaxOff
\ConfigureEnv{minipage}{\ifvmode\IgnorePar\fi\EndP\HCode{<div class="
   minipage" align="center" id="minipage\the\mini@count"> }}
{\ifvmode\IgnorePar\fi\EndP\HCode{</div>\Hnewline}%
  % we must write dimension here to the css file
}{}{}
\makeatother
\Css{div.minipage {
```

```
  float: left;
 }
}
\Css{div.minipage:last-child {
 clear: none;
 float: right;
 }}
\Css{ div.minipage + :not(.minipage) {clear:both;overflow:auto;width:100\%;}
   }
```

Código 52: Configurar `minipage`.

Ahora esto va antes que nuestro entorno definción, por orden, y configuramos este entorno (y todos los otros) para que reciban minipage con dimensiones correctas. El código 53 muestra como se configura el `\Css` de l entorno definicion. Este modelo se usa en el resto de entornos.

```
\Css{ %continuación de la configuración de definición.
%/* --- Compatibilidad con minipages dinámicas --- */
%/* Contenedor padre de minipages (simula 1.0\textwidth) */
       .definicion > .minipage {
              width: 100\% !important; %/* Ancho completo */
              border: none !important; %/* Opcional: elimina borde si no
  lo quieres en el contenedor padre */
              margin-bottom: 0.5em;
              padding-bottom: 0;
       }
%/* Minipages internas (usan width dinámico via tu código \makeatletter) */
       .definicion .minipage .minipage {
              float: left; %/* Tus reglas originales */
              margin-right: 2\%; %/* Espacio entre columnas (ajustable) */
              box-sizing: border-box;
       }
       %/* Última minipage (flotada a la derecha) */
       .definicion .minipage .minipage:last-child {
              float: right !important;
              margin-right: 0 !important;
       }
       %/* Limpiar floats después de minipages anidadas */
       .definicion .minipage::after {
              content: "";
              display: table;
              clear: both;
       }
       %/* Ajuste para párrafos y contenido interno */
       .definicion .minipage p {
              margin-top: 0.3em;
              margin-bottom: 0.3em;
       }
       %/* Captions centrados */
       .definicion .caption {
              text-align: center;
              margin-top: 0.5em;
       }
}
```

Código 53: `minipage` en `definicion`.

2. Además de Mathml, usamos Mathjax. Configurar MathJax puede ser bastante extenso. En la plantilla.tex solo usamos una configuración básica. Si el documento requiere otros paquetes

(que sean soportados) y otros comandos, se pueden agregar. El código 54 muestra como agregar el script y una configuración mínima.

```
% Incrustar en el <head> del html
\Configure{@HEAD}{
\HCode{<script src="https://polyfill.io/v3/polyfill.min.js?features=es6">}\
   HCode{</script>}
\HCode{<script id="MathJax-script" async="" src="https://cdn.jsdelivr.net/
   npm/mathjax@3/es5/tex-mml-chtml.js">}\HCode{</script>}
\HCode{<script>
 MathJax = {
    tex: {
        inlineMath: [['\\(', '\\)']],
        displayMath: [['\\[', '\\]']],
        packages: {'[+]': ['textmacros']}, // Permitir comandos de texto
        processEscapes: true
    },
    options: {
        skipHTMLTags: ['script', 'noscript', 'style', 'textarea', 'pre']
    }
}}\HCode{</script>}}
```

Código 54: Mathjax.

En el preámbulo y en el archivo config.cfg aparecen más configuraciones figure, subfigure, etc. Es un archivo delicado que requiere mucha precisión, así que si lo vamos a modificar, lo mejor es hacer un respaldo.

8. Conclusión

Este artículo muestra cómo combinar LaTeX con herramientas modernas para crear contenido educativo y científico interactivo, ampliando su alcance y utilidad en entornos digitales. La integración de IA facilita la generación de código y la resolución de problemas técnicos, mientras que la modularidad y configuración detallada aseguran resultados profesionales. Este enfoque no solo optimiza el flujo de trabajo, sino que también promueve la ciencia abierta y la educación digital al hacer el contenido más accesible y dinámico. La plantilla proporcionada sirve como punto de partida para adaptaciones futuras, destacando la importancia de la flexibilidad y la innovación en la comunicación académica.

Accesibilidad de datos: Contactar al autor para tener acceso a cualquier información adicional.

Referencias

Denlinger, C. G. (2010). *Elements of real analysis*. Jones & Bartlett Publishers.

GeoGebra. (s.f.). GeoGebra: Plataforma interactiva de matemáticas [s.f.]. https://www.geogebra.org/

González, P. (2021). ltximg: LaTeX environments to image and standalone files (Version 2.1). https://ftp.mpi-inf.mpg.de/pub/tex/mirror/ftp.dante.de/pub/tex/support/ltximg/ltximg-plantilla.pdf

Hoftich, M. (2024). TeX4ht Documentation by TeX4ht Project. https://www.kodymirus.cz/tex4ht-doc/tex4ht-doc.html

Project, I. (2025). Inkscape: Professional vector graphics editor. https://inkscape.org/

Team, G. (s.f.). Incorporación de GeoGebra a páginas web [n.d.]. https://www.geogebra.org/m/bs3cmu4a

W3Schools. (s.f.-a). CSS Tutorial [s.f.]. https://www.w3schools.com/css/

W3Schools. (s.f.-b). HTML Tutorial [s.f.]. https://www.w3schools.com/html/

Apéndice A. Paquete ltximg

En presencia de muchas fórmulas generadas por entornos que Make4ht no soporta o que no traduce bien, el el paquete ltximg (González, 2021), no ayuda a recortar estas fórmulas o entornos, como imágenes. directamente de un PDF local, para cada fórmula.

Se ejecuta con un comando *en la terminal*: Busca los entornos marcados con `%<*ltximg> %</ltximg>` y los convierte en imágenes .svg

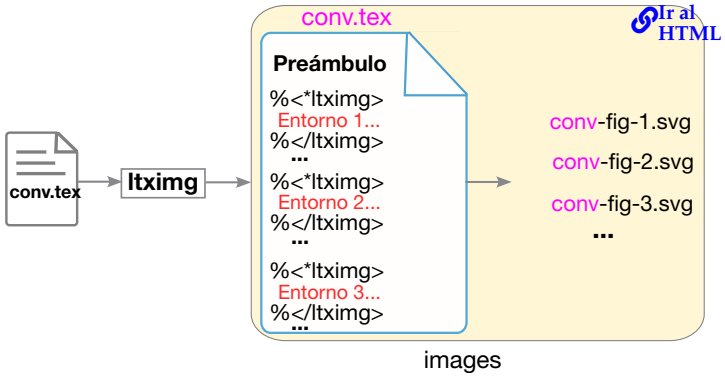


Figura 27: ltximg convierte entornos a .svg. Elaboración propia.

El entorno `%<*ltximg> %</ltximg>` es ingnorado por PDFLaTeX y por Make4ht. Pero ltximg si lo interpreta como un *identificador*. El código 55 muestra cómo marcar los entornos.

```
%<*ltximg>
  Entorno...
%</ltximg>
```

Código 55: `ltximg`.

Preparamos un archivo conv.tex con los entornos y los paquetes y comandos que se necesitan

```
% conv.tex
\documentclass{article}
\usepackage[utf8]{inputenc}
% Preámbulos para cada entorno
\usepackage{polynom}
```

```
\usepackage{nicematrix,tikz}
\usetikzlibrary{fit}
\newcommand{\ff}{\mathtt{f}}

\begin{document}
% conv-fig-1.svg
%<*ltximg>
%<- polynom NO requiere entorno matemático
\polylongdiv[style=D]{6x^3-2x^2+x+3}{x^2-x+1}
%</ltximg>

% conv-fig-2.svg
%<*ltximg>
\begin{NiceTabular}[hvlines]{ccc}
\CodeBefore [create-cell-nodes]
  \tikz \node [draw,fill=blue!15,rounded corners,fit = (2-1) (2-3)] {} ;
\Body
  a & b & c \\
  d & e & f
\end{NiceTabular}
%</ltximg>
...
\end{document}
```

Código 56: Conv.text.

Una vez que tenemos idenficados los entornos que vamos a convertir a imagen SVG, tenemos que *correr en terminal* el comando que recorta y hace las conversiones.

```
ltximg --skipenv 'tikzpicture' --runs 2 --subenv --svg --imgdir images -o conv-out
 conv.tex
```

Funciona así: Este comando recorre el documento conv.tex y localiza los entornos que están envueltos en el entorno `%<*ltximg> %</ltximg>`, luego genera y pega en la carpeta images, un archivo conv.tex con solamente los entornos y el preámbulo del documento, luego genera la imagen .pdf de cada entorno y las convierte a .svg.

Los nombres de las imágenes van numeradas por orden de aparición:

conv-fig-1.svg, conv-fig-2.svg, etc.

Para escalar las figuras SVG podemos usar rsvg-convert o inkscape (o programas equivalentes).