

## Uso de simulateBMP() v3.0

```
[sensorData, t, dt, pVariable, lx, ly] = ...  
%      simulateBMP(imageFileName, scale, duration, ...  
%      simMedium, simSource, recordVideo);
```

Simula propagación de fuentes acústicas de presión en escenarios 2D definidos por un archivo de imagen. Permite el uso de múltiples sensores de presión y múltiples fuentes con libre definición de su variación temporal

**imageFileName** es el archivo BMP

La versión 3.0 admite formato BMP 256 o BMP 24 bits en forma indistinta.

Conviene seleccionar tamaños de bits que permitan cálculos eficientes. Para ello conviene utilizar previamente `checkFactors(min_number, max_number)`, que es una función del toolbox `kWave` que permite seleccionar dimensiones eficientes.

Los tamaños de tipo  $2^n$  son eficientes (128, 256, 512, 1024, etc.)

Por ejemplo: un archivo BMP de 256x512 es más eficiente que uno de 250x500.

El color blanco representa aire. El color negro representa elementos de reflexión total. El rojo representa fuentes. El verde sensores y el marrón elementos de material absorbente. La versión 3.0 admite absorción pero con un único valor del coeficiente de Sabine para todos los materiales definidos con color marrón.

**scale**

Es la dimensión de cada mínimo salto de reticulado en metros. El reticulado esperado por `simulateBMP` asume que el espaciado es igual tanto en x como en y.

Si se utilizar `scale=0.01` (1 cm) se obtiene una frecuencia de muestreo de simulación de 114 kHz y una frecuencia válida de Nyquist de 17 kHz para las componentes de las ondas que se propagan.

Si se elige otro valor de `scale`, la versión 3.0 informa al comenzar la fNyquist disponible.

**duration**

Indica el tiempo de duración de la simulación en segundos.

En el uso normal `k-Wave` no requiere este parámetro, sino que lo determina en relación con el tiempo máximo que puede tardar una onda en atravesar todo el espacio de simulación. La función `simulateBMP()` pretende simular ondas con duraciones mayores al recorrido total de la imagen. Requiere definirlo en forma obligatoria (no existe ningún valor definido por defecto).

**simMedium**

Es una estructura (struct) que contiene variables relacionadas con el medio de simulación.

En la versión 3.0 es posible realizar simulaciones sin especificar todos los parámetros, ya que incluye valores por defecto para cada uno. Sin embargo, como se trata de un argumento necesario en la función (`simMedium` debe existir), resulta imprescindible especificar al menos uno de los siguientes.

`simMedium.c0` especifica la velocidad de propagación en m/s en el aire (zonas de color blanco en la imagen). Si no existe el campo `c0` en la estructura asume el valor por defecto de 344 m/s

`simMedium.density` especifica la densidad del aire. El valor por defecto es de 1.2 kg/m<sup>3</sup>

`simMedium.alpha` especifica el coeficiente de absorción de Sabine (que no debe confundirse con el coeficiente de absorción de energía que se utiliza en `k-Wave`). Un coeficiente `alpha=1` significaría absorción total. El valor por defecto si no se especifica el campo es `alpha=0.5`. De todas maneras, sólo tiene efecto en la simulación si la imagen contiene puntos en color marrón.

`simMedium.speedRatio` especifica la relación de velocidades de propagación entre el medio absorbente y el aire, considerando `speedRatio= cAbsorb/c0`. Sólo se utiliza cuando existen puntos marrones en la simulación. El valor por defecto es `speedRatio=1.1`.

`simMedium.sign` especifica el signo de la reflexión (su fase). Esto se relaciona con que existe una indeterminación al determinar un coeficiente de Sabine en relación con la fase de la reflexión. Dicho en otras palabras, un coeficiente de Sabine `alpha=0.5` indica que se absorbe la mitad de la energía, pero la

onda que no es absorbida podría reflejarse con la misma fase de presión (más cercano a la reflexión en un tubo cerrado) o con la fase invertida (más cercano a un tubo con extremo abierto). El valor por defecto es  $\text{sign}=1$  (tubo cerrado).

`simMedium.CFL` especifica el valor de coeficiente Courant–Friedrichs–Lewy relacionado con la estabilidad de la solución de las ecuaciones diferenciales. El valor por defecto es  $\text{CFL} = 0.5$ .

`simMedium.showAbsorptionMask` es una variable lógica que especifica mostrar la máscara de material absorbente (como zonas punteadas) en la simulación. El valor por defecto es `true`.

`simMedium.showWallMask` es una variable lógica que especifica mostrar la máscara de material reflectante en la simulación. El valor por defecto es `true`.

`simMedium.showSourceMaskes` una variable lógica que especifica mostrar la máscara de fuentes en la simulación. El valor por defecto es `true`.

`simMedium.showSensorMaskes` una variable lógica que especifica mostrar la máscara de sensores en la simulación. El valor por defecto es `true`.

`simMedium.showLegendMaskes` una variable lógica que especifica mostrar la máscara de leyendas o comentarios (puntos grises puros en el bmp original). El valor por defecto es `true`.

## `simSource`

Es una estructura que determina las propiedades de las fuentes utilizadas

`simSource.type` indica el tipo de señal de las fuentes. Dependiendo del tipo de señal seleccionada se tomarán en cuenta otros parámetros de la estructura para definirla completamente, aunque todos los necesarios tienen valores asignados por defecto para hacer más sencillas las simulaciones rápidas.

`simSource.type = 'impulse'` es una señal impulso (solamente un valor unitario en el primer elemento del vector con el resto de la duración igual a cero). Utilizar los parámetros `simSource.amplitude` (amplitud en pascales), y `simSource.mode` (especificando si la fuente será del modo 'additive' (en el cual la presión de la fuente se sumará a lo que el resto de la simulación especifique para dicho punto) o del modo 'dirichlet' (en el cual la presión seleccionada es una condición de contorno fija para el punto en donde está definida la fuente).

`simSource.type = 'nCycles'` es una señal de una sola componente en frecuencia con una cantidad especificada de ciclos emitidos. Lógicamente la cantidad de ciclos especificados debería ser menor que la duración total de la simulación. Si esto no se cumple, la simulación de todas maneras se realiza, pero el resultado sería el de una senoidal pura. Adicionalmente k-Wave en ese caso emite una advertencia (warning) indicando que los datos de fuente son mayores a los tiempos de simulación. Utiliza los siguientes parámetros: `simSource.amplitude` (amplitud en pascales), `simSource.n=2` (cantidad de ciclos), `simSource.f0 = 1000` (frecuencia en Hz), y `simSource.mode`.

`simSource.type = 'whiteNoise'` define el uso de ruido blanco con una duración determinada que si es inferior a la duración total de la simulación permite analizar tiempos de reverberación. Utiliza `simSource.amplitude` (amplitud en pascales), `simSource.duration` (duración del ruido en segundos desde el comienzo de la simulación), y `simSource.mode`.

`simSource.type = '(texto con la ecuación)'` define a la señal de presión utilizando la ecuación especificada entre apóstrofes. La duración de dicha señal es igual a la duración total de la simulación. La ecuación contiene toda la información necesaria por lo que solamente se requiere especificar el modo mediante `simSource.mode`. Ejemplo de uso: `simSource.type = '4*sin(2*pi*1000*t+pi/4)'`;

`simSource.type = 'audio'` define a la señal de presión mediante cualquier variable definida en el Workspacede MATLAB. La variable a utilizar se definirá haciendo que `simSource.p` se igual a dicha variable (por ejemplo, `simSource.p=audioSamples`). Se requiere especificar además `simSource.fs` (frecuencia de muestreo del audio incluido en la variable MATLAB), y `simSource.mode`. Es importante mencionar que en este caso el programa, antes de simular, realiza un remuestreo automático para adecuar la frecuencia de muestreo del audio (`simSource.fs`) a la frecuencia de muestreo de la simulación (`fs`, que usualmente resulta muy superior en valor). Al utilizar este tipo de fuente el programa, una vez finalizada la simulación, remuestrea las señales captadas por los sensores para devolver el resultado en la frecuencia de muestreo del audio. En suma, `sensorData` devuelta por `simulateBMP()` en este caso tendrá la misma frecuencia de muestreo del audio de entrada.

`simSource.fCut` especifica la frecuencia de corte de un filtro antialias. Si no está especificada adopta un valor menor a la frecuencia de Nyquist que puede calcularse como  $f_{Nyquist} = c_{Min}/2/dx$ . Para  $c_{Min} = 344$  m/s y  $dx = 0,01$  m;  $f_{Nyquist} = 17.2$  kHz y  $f_{Cut} = 15$  kHz.

`simSource.order` especifica el orden del filtro antialias generado con la función `butter()`. El valor por defecto es `order=15`.

Valores por defecto de los campos de `simSource` que especifican parámetros

`simSource.mode = 'additive'` (adopta este valor cuando no se especifica `mode`, pero también cuando `mode` se especifica incorrectamente, aunque en ese caso advierte mediante un mensaje que fue seteado el modo `additive`).

`simSource.amplitude = 10`;

`simSource.fs=44100`; (sólo se utiliza con 'audio')

`simSource.f0=1000`;

`simSource.n=3`; (sólo se utiliza con 'nCycles')

`simSource.duration=duration/5`; (sólo se utiliza con 'whiteNoise')

En la versión `simulateBMP()` 3.0 se puede utilizar un arreglo de fuentes con definición individual de la señal de cada fuente. En ese caso la variable `source.p` debe tener una cantidad de filas igual a la cantidad de fuente y una cantidad de columnas igual a la cantidad de muestras de señal de cada fuente.

## Estructura del script para utilizar simulateBMP()

```
% Ejemplo de versión mínima (utilizando valores por defecto)
%% 1-Setup parameters
ImageFile='HelloWorld.bmp';
scale=1e-2; % 1 cm
duration = 1e-3; % 1 ms
simSource.type='nCycles';
recordVideo=false;

%% 2-Call simulateImage256
[sensor_data, t, dt, pVariable, lx, ly] = ...
simulateBMP(imageFileName, scale, duration, ...
simMedium, simSource,recordVideo);

%% 3-Plot the results
tVariable=(0:length(pVariable)-1)*dt;
plot(t,sensor_data)
title('Sensor'); xlabel('t [s]'); ylabel('p [Pa]'); grid on
```