

# Locate a Signal in Audio in the Presence of Noise

## ▼ Introduction

This application demonstrates how you can estimate the location of a signal that might exist in a larger signal.

- First, an audio file is first loaded, a small segment is extracted, and random Gaussian noise is added to both
- The [cross-correlation](#) of the full audio and the extract is computed, and the maximum lag computed

The maximum lag is the index at which the extract is predicted to be found in the audio.

> restart:

```
with(SignalProcessing) :
with(ColorTools) :
with(plots) :
with(AudioTools) :

common_plot_opts :=
  axes           = boxed
, axesfont       = [Calibri]
, size           = [800, 400]
, legendstyle    = [font = [Calibri]]
, labeldirections = [horizontal, vertical]
, labelfont      = [Calibri]
, background     = Color("RGB", [218/255, 223/255, 225/255])
, axis          = [gridlines = [5, color = Color("RGB", [1, 1, 1])]]:
```

## ▼ Generate the Full Measurement

```
> fullAud      := AudioTools:-Read(FileTools:-JoinPath(
  [kernelopts(datadir), "audio", "maplesim.wav"]));
N              := numelems(fullAud);
samplingRate   := attributes(fullAud)[1];
```

$$fullAud := \begin{bmatrix} \text{"Sample Rate"} & 11025 \\ \text{"Bit Depth"} & 16 \\ \text{"Channels"} & 1 \\ \text{"Points/Channel"} & 8227 \\ \text{"Duration"} & 0.75 s \end{bmatrix}$$

$N := 8227$   
 $samplingRate := 11025$

(2.1)

Add noise to the audio

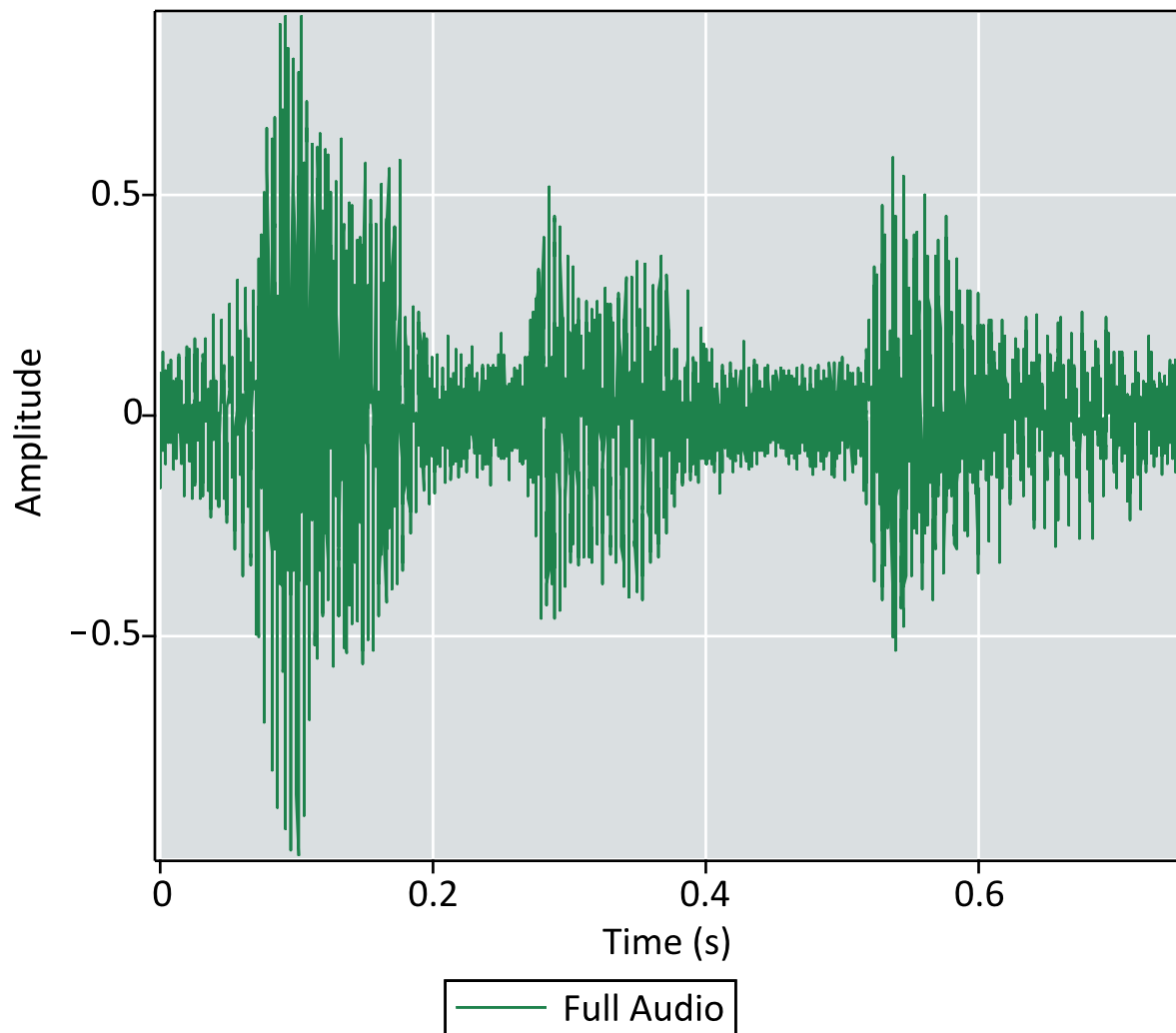
```
> fullAudNoise := fullAud + GenerateGaussian(N, 0, 0.05);
```

$$fullAudNoise := \begin{bmatrix} \text{"Sample Rate"} & 11025 \\ \text{"Bit Depth"} & 16 \\ \text{"Channels"} & 1 \\ \text{"Points/Channel"} & 8227 \\ \text{"Duration"} & 0.75 s \end{bmatrix}$$

(2.2)

```
> times := Vector(N, i -> (i - 1)/samplingRate, datatype = float
[8]):
```

```
p1 := plot(times, fullAudNoise
,thickness      = 0
,color          = Color("RGB", [30/255, 130/255, 76/255])
,legend         = "Full Audio"
,labels         = ["Time (s)", "Amplitude"]
,common_plot_opts)
```



## ▼ Extract Part of the Full Measurement

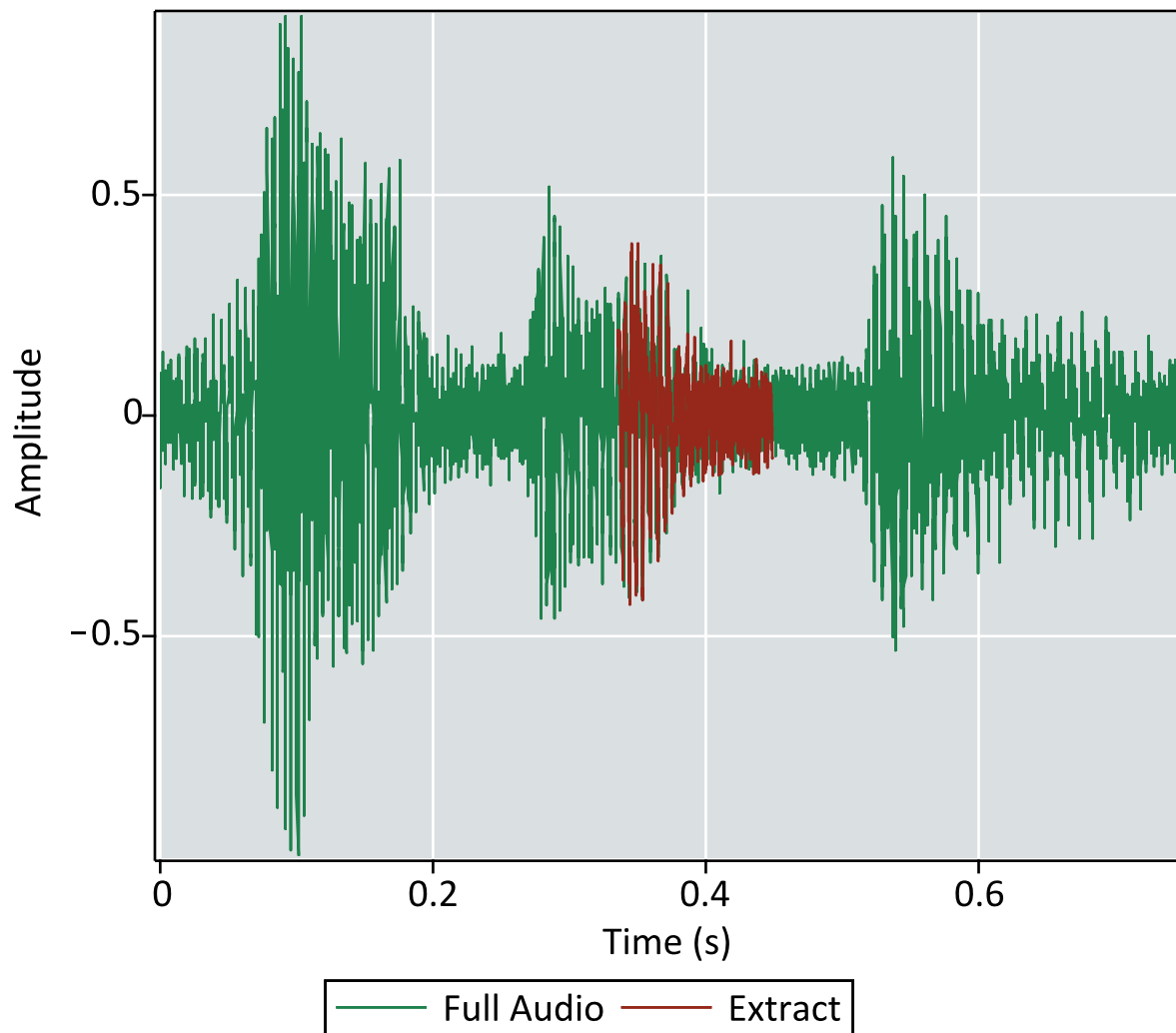
Now extract *delta* elements starting at position *i\_start*, and add noise

```
> delta    := 1250:
   i_start := 3700:
   i_end   := i_start + 1250:

   extract := Vector(fullAud[i_start .. i_end - 1]) +~
   GenerateGaussian(i_end - i_start, 0, 0.05):
   t_extract := Vector(i_end - i_start, i -> (i + i_start - 1)
   /samplingRate, datatype = float[8]):

> p2 := plot(t_extract, extract
,thickness = 0
,color      = Color("RGB",[150/255, 40/255, 27/255])
,legend     = "Extract"
,labels     = ["Time (s)", "Amplitude"]
,common_plot_opts):

display(p1, p2)
```

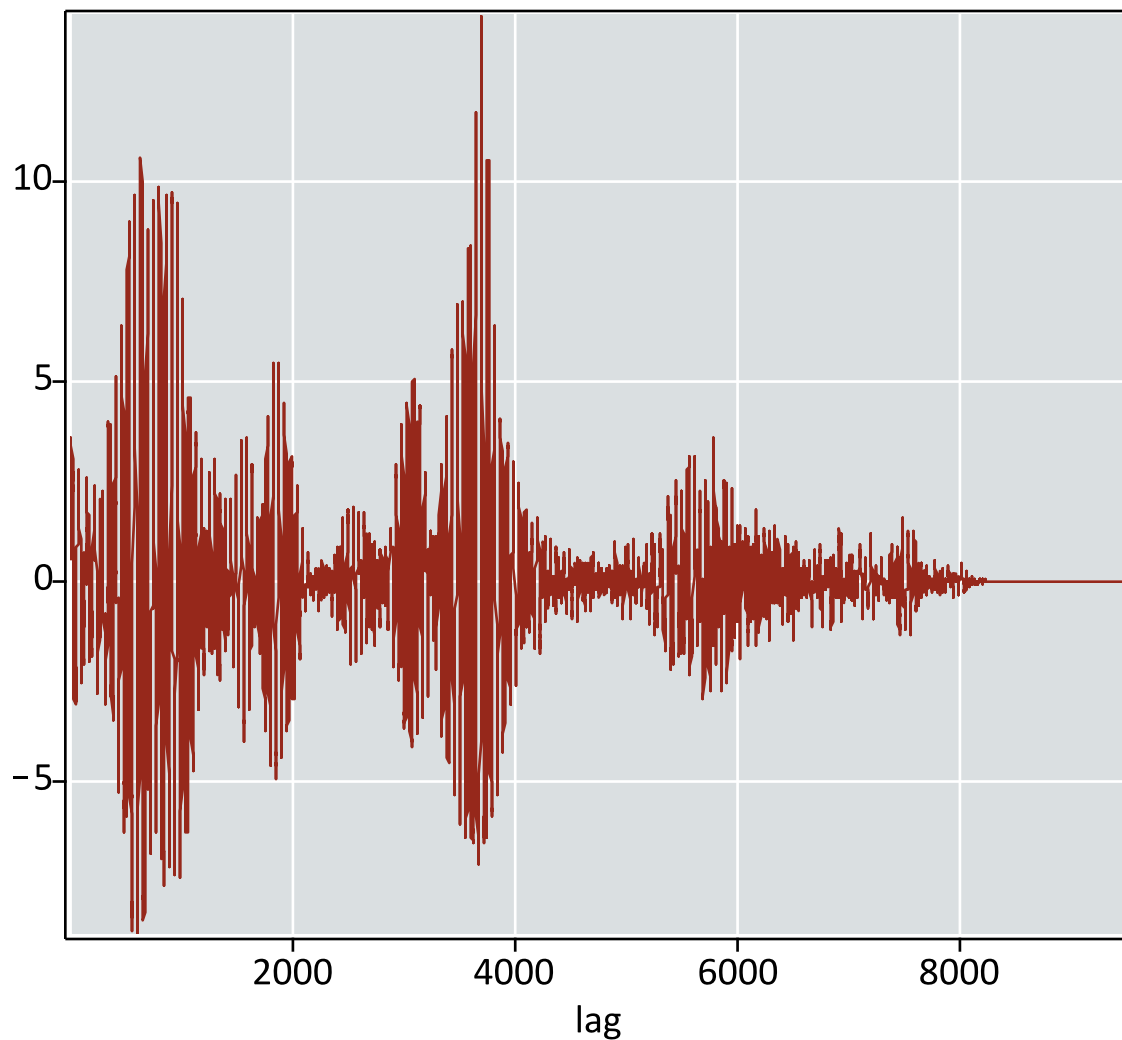


## ▼ Find the Position of the Signal in the Measurement using Cross-Correlation

Now calculate and plot the cross-correlation of the full audio and the extracted signal

```
> cc := CrossCorrelation(extract, fullAudNoise, 1):

dataplot(cc
,style          = line
,thickness      = 0
,color          = Color("RGB",[150/255, 40/255, 27/255])
,labels         = ["lag", ""]
,common_plot_opts)
```



The highest cross-correlation is at this index. This is position that the extracted signal is predicted to start at in the full audio

```
> maxLag := max[index](cc);
   i_end   := maxLag + delta:
                                     maxLag := 3699
(4.1)

> t_extract := Vector(i_end - maxLag, i -> (i + i_start - 1)
  /samplingRate, datatype = float[8]):

p2 := plot(t_extract, extract
,thickness      = 0
,color          = Color("RGB",[150/255, 40/255, 27/255])
,legend         = "Extract"
,labels        = ["Time (s)", "Amplitude"]
,common_plot_opts):

display(p1, p2)
```

