Rastreo de direcciones de arribo en tiempo real

Mario Horacio Garrido Czacki

MUSIC

- Implementado en Python + Numba (Compilador JIT)
- Enfoque a tomar:
  - Realizar análisis cada 2 ventanas.
  - Detectar qué ventanas contienen una señal de interés (magnitud).
  - En esas ventanas, encontrar las frecuencias de interés.
  - Aplicar MUSIC en las frecuencias encontradas.
  - Mostrar resultados.
- Primero se realizó una implementación inicial, y luego se fue iterando para obtener los resultados que se presentarán a continuación.

Funcionamiento

Callback de JACK (productor):

- Solamente se dedica a almacenar las ventanas en una cola circular.

Consumidor:

- “Consume” los elementos en la cola circular, realizando el análisis mientras estén disponibles.
- Trabaja con secuencias de dos ventanas y las unifica en una sola para su análisis.

Selección de ventanas (detección de voz)

Se plantea como una clasificación binaria entre voz (1) y no voz (0).

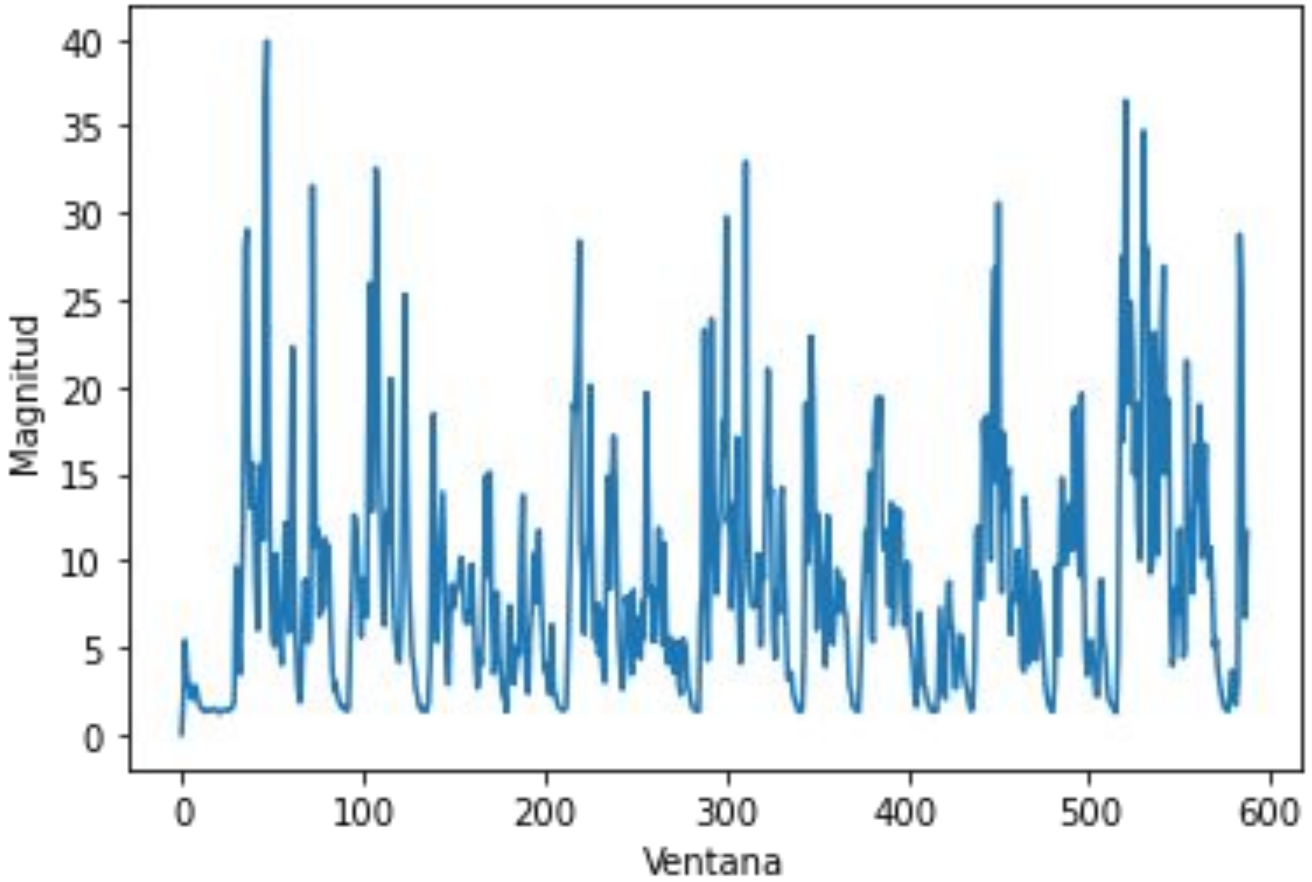
Soluciones intentadas:

- Comparar la magnitud del espectrograma actual con *n* pasados. (Era susceptible a fallar en cambios de volumen de voz)
- Umbral de detección estático. (Variaba entre grabaciones con ruido/sin ruido)

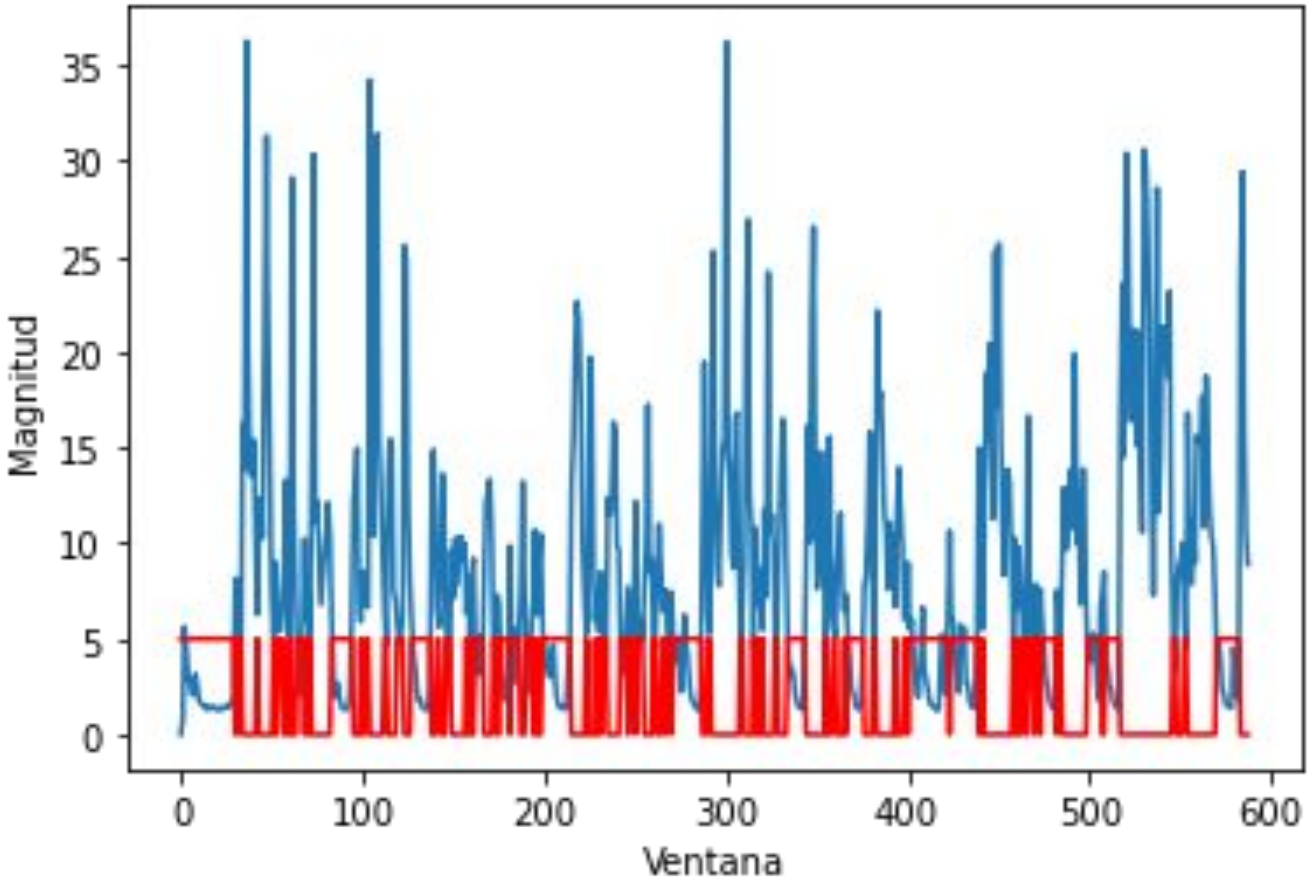
Mejor solución encontrada:

- Perfilado de magnitud de ruido dadas las primeras 10 ventanas (consideradas como ruido).
- Una ventana con magnitud superior a 1.2 veces el perfil del ruido se considera como una ventana con señal de interés.

Magnitud por ventana



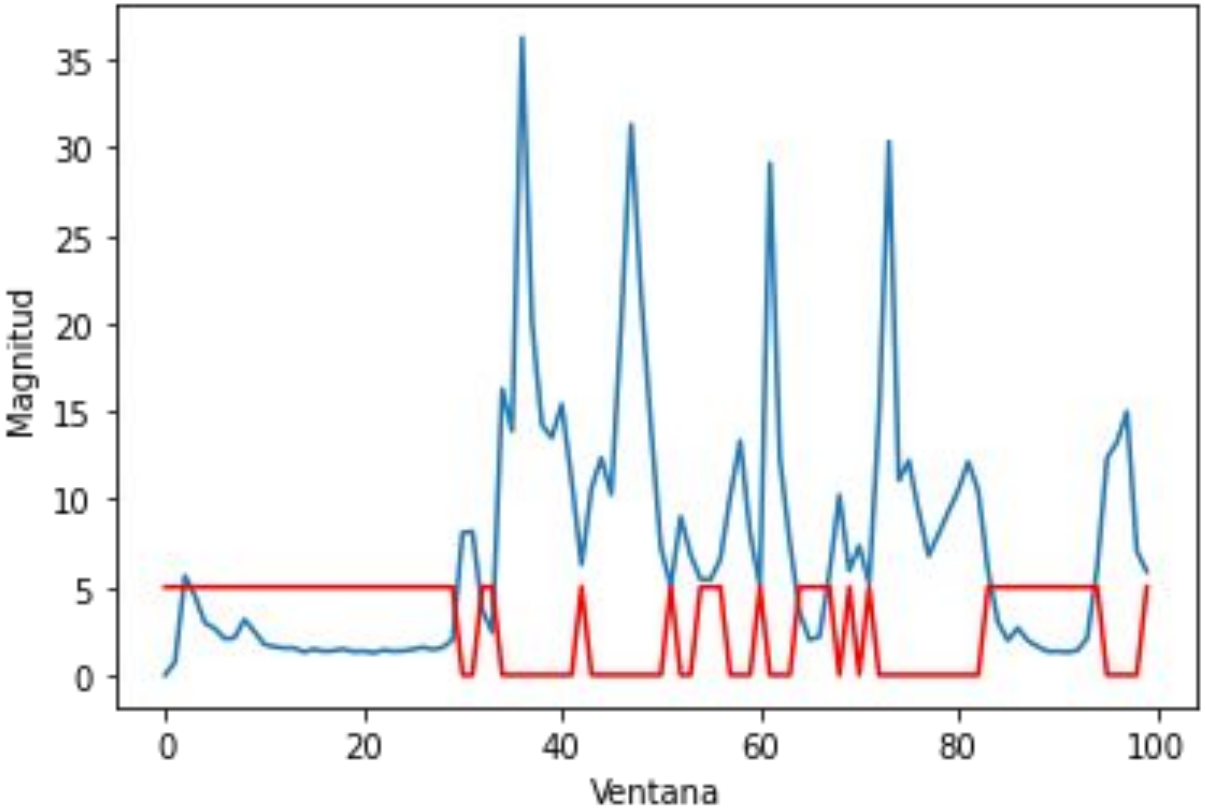
Magnitud por ventana y detección (bajo es voz)



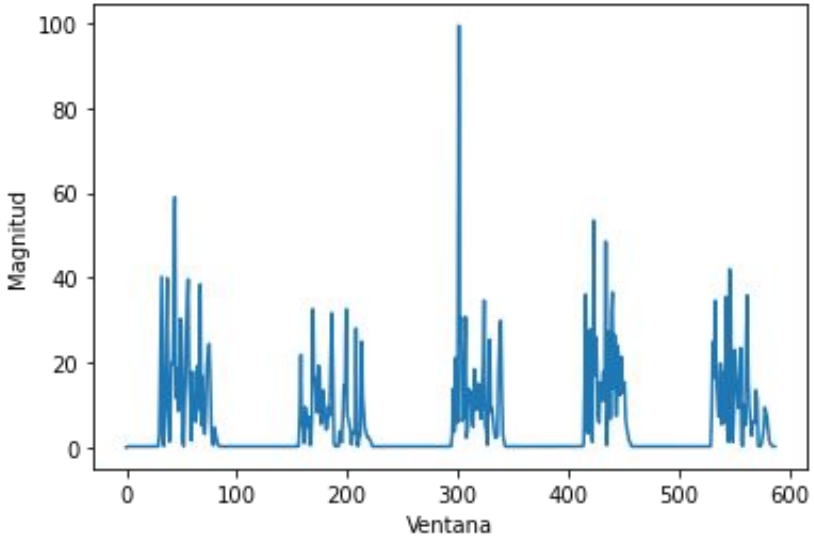
Análisis

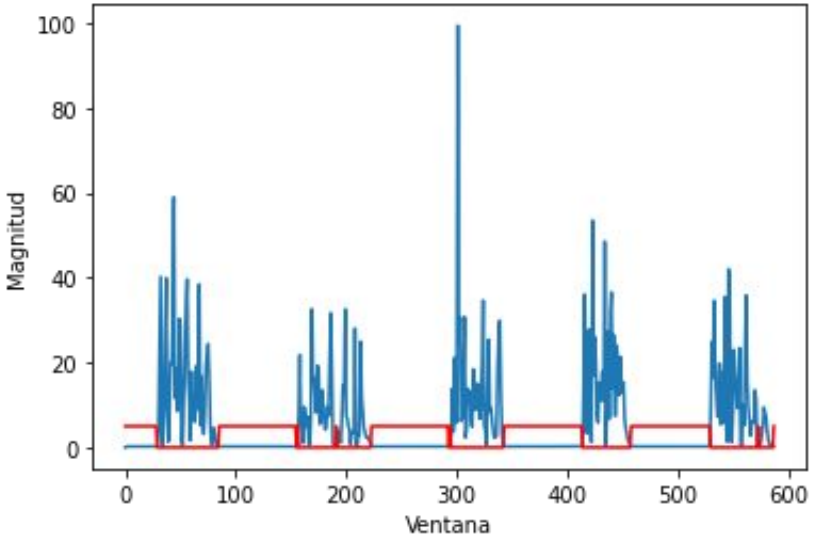
A escala pequeña se nota que esta aproximación minimiza los falsos positivos, pero tiene algunos falsos negativos.

Mejor analizar menos ventanas a analizar ventanas sin señales de interés (ruido).



En señales sin ruido:





Selección de frecuencias de interés

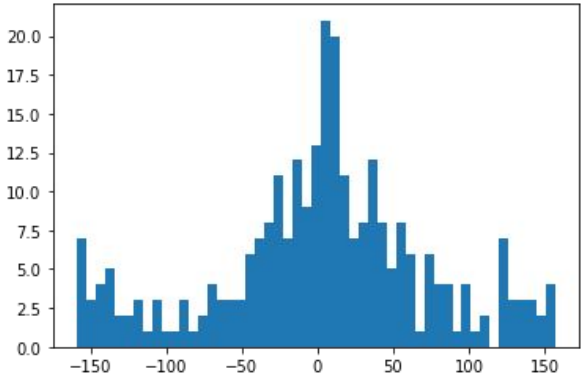
Se intentaron varias aproximaciones:

- *N* frecuencias con magnitud más alta (poco robusto).
- Noise score basado en el Long-term spectral divergence http://ras.papercept.net/images/temp/IROS/files/0192.pdf
- *N* picos en frecuencia, con magnitud más alta.

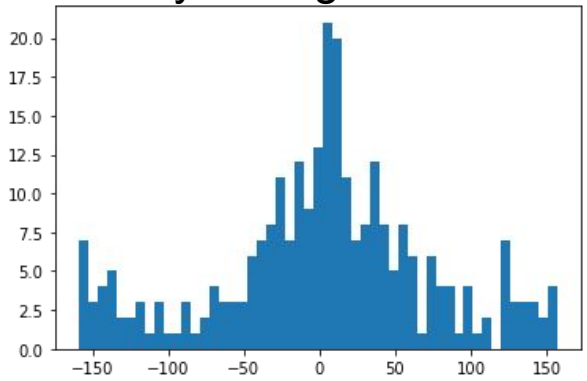
Los resultados de los dos últimos eran indistinguibles entre sí, por lo que opté por los picos en frecuencia ya que eran los más sencillos de calcular.

Análisis sobre 25 segundos de AIRA (noisy-1source):

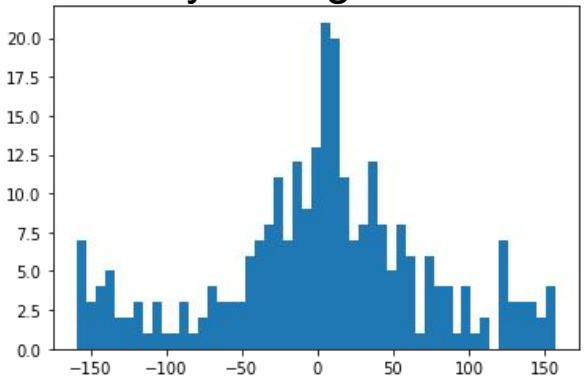
5 frecuencias con mayor magnitud



10 frecuencias con mayor magnitud



50 frecuencias con mayor magnitud

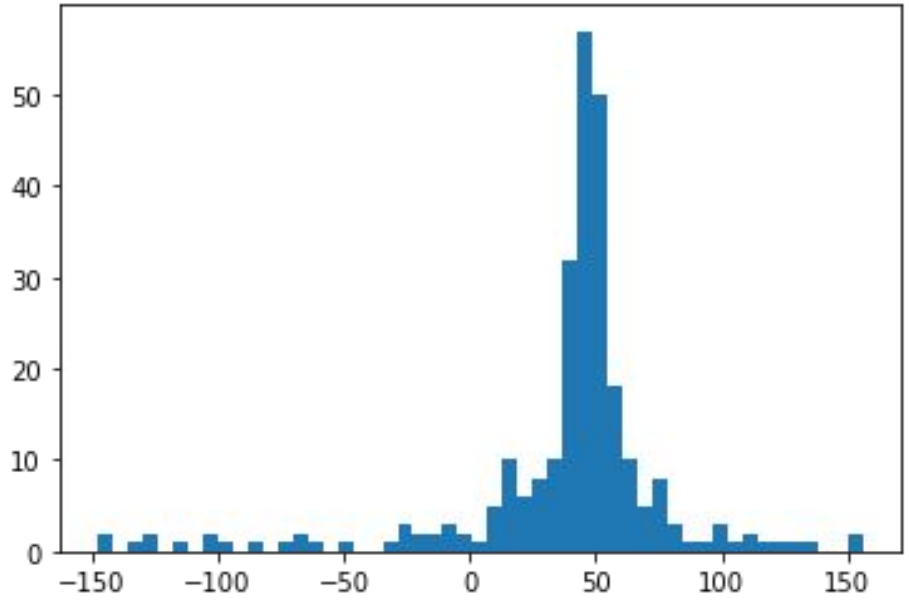


DOA real: 0°

Sin diferencia alguna, por lo que opté por 5.

Se presentan los histogramas de los picos en espectro de frecuencia MUSIC por ventana.

Análisis sobre 25 segundos de AIRA (clean-1source):



DOA real: ~60°

Estimar el número de señales de interés

- Se intentó un enfoque adaptativo utilizando el “criterio del codo” sobre la varianza explicada en los eigenvalores.
- Idea: Encontrar cuántos de los eigenvalores explican en n% de la varianza.
- En casi todos los casos se detectó que el primer eigenvalor explicaba más del n% de la varianza (se intentó con diversos umbrales).
- 90%, 80%, 50%

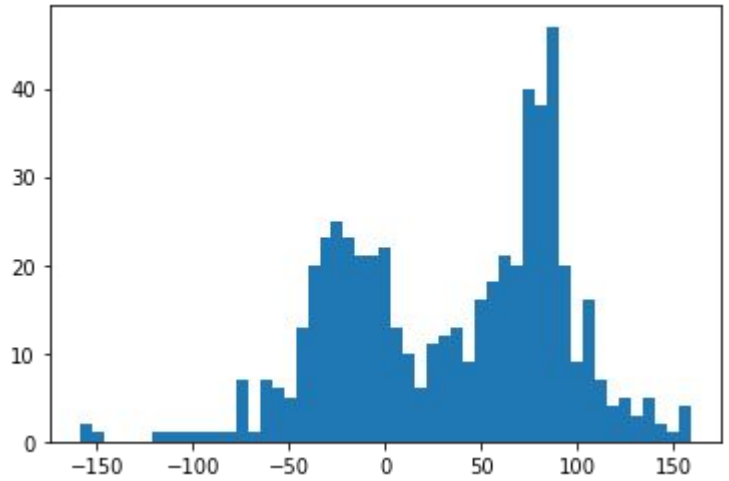
Conclusión:

- No se pudo llegar a mucho en este aspecto.
- El sistema es adaptativo, pero por lo general se encuentra $r = 1$.

No obstante de lo anterior, el funcionamiento es interesante

Comportamiento con 2 fuentes:

clean-2source:

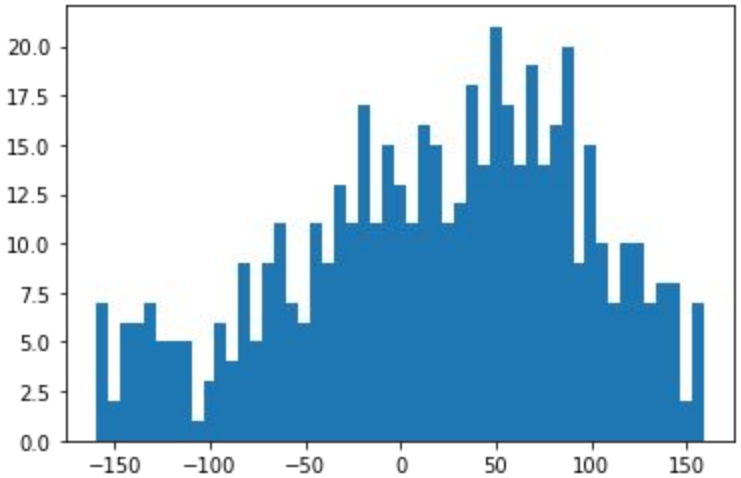


DOAs reales: ~-30° y ~90°

Parece que en todo momento domina una fuente sobre la otra, incluso cuando hay dos al mismo tiempo.

Ahora con ruido:

noisy-2source



DOAs reales: ~-30° y ~90°

Conclusión:

- MUSIC requiere de experimentación y modificaciones constantes para aumentar el desempeño en señales de voz. Originalmente era usado para telecomunicaciones, por lo que se sacó de su dominio.
- El ruido le genera problemas al encontrar vectores ortogonales que no son de interés a lo largo del espectro de MUSIC.
- Podría ayudar agregar otro algoritmo de reducción de ruido/reverberación a la señal antes de pasarla por MUSIC. De igual manera, buscar mejorar el filtrado de frecuencias de búsqueda podría traer mejores resultados.
- Para poder reducir el “ruido” en la salida del algoritmo se podría utilizar un filtro de Kalman/partículas.
- Sigue siendo complicado encontrar el número de fuentes en el audio (0, 1, 2, …)