**Room Impulse Response Encoding (L08)**

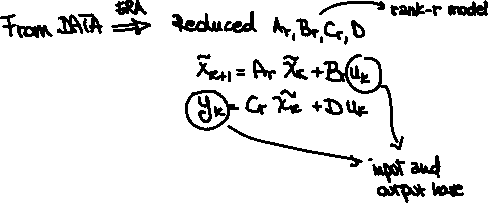
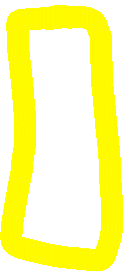
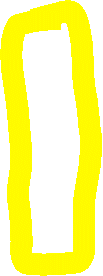
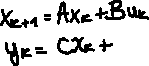
The simulation of acoustic environments needed for, e.g., virtual and augmented reality applications, often relies on room impulse responses (RIRs) measured in real-world reverberant spaces. RIRs typically consist of hundreds or even thousands of coefficients, making them difficult to store, transmit, and apply to an audio signal in real time. Therefore, it is necessary to reduce the RIR's dimension, by employing methods known in the literature as model order reduction (MOR). The key idea of this project is to derive the State-Space Model (SSM) representation of a measured RIR and apply state-ot-the-art MOR techniques to progressively compress them, while trying to preserve the acoustical characteristics of the original reverberation, somewhat akin to what happens, e.g., in speech and video coding.

Requirements: MATLAB/Python, fundamentals of linear algebra.

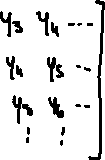
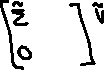
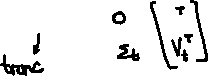
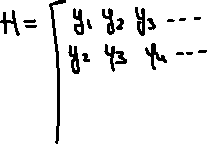
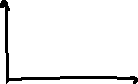
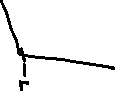
Additional resources:

* <https://www.youtube.com/watch?v=neTO1_RuLdU&ab_channel=SteveBrunton>
* <https://www.youtube.com/watch?v=oulLR06lj_E&list=PLMrJAkhIeNNQkv98vuPjO2X2qJO_UPeWR&ab_channel=SteveBrunton>
* Data-Driven Science and Engineering, Steven L. Brunton, J. Nathan Kutz, Cambridge University Press, 2019

[Space-state representation](https://en.wikipedia.org/wiki/State-space_representation)



[Hankel matrix](https://en.wikipedia.org/wiki/Hankel_matrix) [Singular value decomposition](https://en.wikipedia.org/wiki/Singular_value_decomposition)



MAIN PASSAGES:

Tradeoff between number of parameters (sum of coefficients of A,B,C,D) and accuracy/mean squared error à try ERA with different rank

First: SISO

Next: MIMO

Progressive compression à ERA on the residual for two/three times

Dataset à to be normalized!!!

[Energy Decay Relief](https://ccrma.stanford.edu/~jos/pasp/Energy_Decay_Relief.html)



Consider the residuals and apply again the compression, integrate them to the output y to obtain a better output.

If the results are not good experiment with different concepts before moving on to MIMO (for example try progressive compression first)  
Do a lot of plots to see results

OTHER CONCEPTS:

PCA (Principal Component Analysis) is a linear dimensionality reduction technique where data is linearly transformed into a new coordinate system (change of basis), with directions (principal components) capturing the largest variation in the data. We use it to find the principal directions of distributions and discard the directions with less variation.

Pag. 453 9.3 System Identification ERA

Dataset: **HOMULA-RIR**

* [arXiv (paper](https://arxiv.org/abs/2402.13896))
* [Zenodo (download](https://zenodo.org/records/10479726))

Original ERA paper: <https://ntrs.nasa.gov/api/citations/19850022899/downloads/19850022899.pdf>

Steven Brunton book: [Brunton\_Data\_driven\_science\_engineering.pdf](https://polimi365-my.sharepoint.com/:b:/g/personal/10391311_polimi_it/EVxmo5EUUQpJo3bpXZ0s_8oB3jFCrolrIVKY-ShJbAdIRA?e=CdYfdG)

**Roadmap**

Quando scaricate HOMULA-RIR da Zenodo, vedrete che contiene dati da un uniform microphone array (ULA) e da vari higher-order microphones (HOM). Inizialmente mi concentrerei su questi ultimi.  
  
Un HOM ha 8 capsule (vedi <https://voyage.audio/spatialmic/>). Fissata un sorgente (S1 o S2), a ciascuna di queste capsule è associata una room impulse response (RIR).    
  
In HOMULA-RIR, le varie RIR sono salvate come canali di un **wav file multichannel**. Inizialmente, la metrica di ***memory requirements* di un HOM** è quindi il totale dei valori nella matrice di campioni di queste RIR, ossia **n\_channels x n\_samples**. Chiamiamo questo numero R0. Dato che ci occuperemo di RIR compression/coding, dovrete fare riferimento a questo numero per confrontare l'efficenza dei metodi studiati.

**Scegliete una sorgente**, diciamo S1, **scegliete l'HOM che preferite.  A ogni capsula del HOM corrisponde un canale del wav file e, a sua volta, corrisponde una RIR.**

**NORMALIZZAZIONE**

Iniziate normalizzando la RIR. Ad esempio, potete dividere ogni entry della matrice che contiene tutte le RIR associate a un HOM  per la sua norma di Frobenius.  In MATLAB, questo corrisponde alla seguente riga di codice.

RIR\_matrix = RIR\_matrix ./ norm(RIR\_matrix , "fro")

Tenete a mente che altre scelte di normalizzazione sono possibili.

**SILENZIO INIZIALE**

Considerate anche che ogni RIR potrebbe avere un **ritardo iniziale** dovuto alla propagazione dalla sorgente al microfono. A noi questo ritardo non interessa. **Rimuovete quindi il silenzio da tutti i canali del file wav.** Nel farlo, **scegliete il lag di silenzio più corto** e tagliate lo stesso numero di sample da tutti i canali/RIRs.

**Step 1.** L'idea è quella di iniziare da applicare Eigensystem Realization Algorithm **(ERA) a ogni canale della RIR indipendentemente dagli altri.** L'obbiettivo è fittare uno state space model (SSM) per ogni RIR.

**STUDIO HSV**

Per ciascuna capsula/RIR fate uno **studio degli Hankel singular values (HSV) per capire quanti modi sono rilevanti** e vanno conservati nella dimensionality reduction del SSM. la funzione [era](https://it.mathworks.com/help/ident/ref/era.html) di MATLAB permette questa analisi specificando il range di valori di [nx](https://it.mathworks.com/help/ident/ref/era.html" \l "mw_416ff25f-d043-4742-a0fa-a3524dfbbe29" \t "_blank). Salvatevi il plot degli HSV perchè potrebbe essere utile nella stesura del report.  
  
Nel caso date un'occhiata all'argomento "InputDelay"  per compensare i vari delay iniziali dovuti alle diverse posizioni delle capsule. Potete stimar questi delay trovando l'argmax(abs(RIR\_matrix[i, :]) dove i=1,..., 8 è l'indice del canale/capsula. Controllate voi se è opportuno e nel caso come fare.

Ricordate che "the HSV plot shows the relative energy contributions of each state in the system." Nel fare RIR compression, noi dovremo ridurre il numero di stati, **conservando solo gli *N* stati più rilevanti.** Osservando il HSV plot potete vedere dove il contributo degli stati si abbassa e diventa pressochè costante. Questo approccio è chiamato "elbow method" o "**knee method**."

Al che potete fittare **8 diversi single-input single-output (SISO) SSMs, uno per ogni RIR.** L'ordine *N*di ciascun SSM può essere diverso dagli altri e NON è necessario sia costante.

Il *memory requirements*del signolo SSM sarà quindi dato dalla somma delle entries delle matrici A, B, C, D. In questo caso, dato che consideriamo una RIR alla volta, B e C saranno vettori e D è uno scalare. Il *memory requirements* sarà quindi *N*^2 (dimensione di A) + *N* (dimensione di B) + *N* (dimensone di C) + 1 (dimensione di D) = *N*^2 + 2*N* + 1. L'obbiettivo è trovare il minimo *N* che mantenga una buona approssimazione della RIR di partenza.

Dato che fittiamo un SSM per ogni signle-channel RIR, il *memory requirements*globale di questo primo approccio, *R1*, sarà la somma dei *requirements*di tutti e 8 gli SSM.

Potete misurare la bontà di queste approssimazioni con il mean squared error (MSE) tra la single-channel RIR di partenza e la risposta all'impulso dello SSM fittato con ERA. Sarebbe interessante un grafico con diversi ordini *N* sulle x e l'MSE sulle y. I diversi ordini corrispondono a diversi SSM ottenuti con nx diversi che, a loro volta, saranno caratterizzati da un diverso MSE.

**Step 2.**Come secondo step è quello di fittare un multiple-input multiple-output (MIMO) SSM sull'intera matrice delle RIR di size n\_channels x n\_samples. In questo modo avremo un solo SSM che cerca di modellare l'intero HOM. Il suo memory requirments, R2, sarà semplicemente la somma del numero di entries in A, B, C, D (che ora saranno tutte matrici, e non più vettori).

Fatto ciò, andrete a confrontare R0, R1 e R2, oltre che a riportare gli MSE dei metodi SISO e MIMO.

**Bonus step.** Potreste ripetere queste analisi anche sui dati dell'ULA incluso in HOMULA-RIR.

**Step 3.**(in preparation of further steps) Una volta fittati gli SSM nei due step precendenti, potete calcolare il segnale residuo, ossia l'errore tra le RIR di partenza e la risposta all'impulso degli SSMs. l'idea è poi quella di andare ad applicare lo stesso tipo di analisi sui residual signals. Perciò andare a plottare un HSV plot per ogni segnale residuo e vedere se è possibile e conveniente fittare un ulteriore SSM sul segnale residuo stesso.

Nel farlo, vi invito a controllare se gli HSV del residuo corrispondono agli HSV scartati nel ridurre l'ordine del SSM originale.

Naturalmente questo step dipende dai risultati dei due precendenti. Ad esempio, se Step 2 risultasse molto migliore di Step 1, si potrebbe ignorare quest'ultimo e continuare solamente dai risultati di Step 2 (cosa che auspico).

-----------------------------------------------------------------------------------------------------------------------------------

Capisco, processare RIR è difficile proprio per questo motivo. In fondo questo è proprio il motivo per cui è interessante studiarne encoding e compressione.

HOMULA-RIR contiene dati a una frequenza di campionamento di 48 kHz. Potete iniziare farne il resampling a 16 kHz (ed eventualmente a 8 kHz se 16 dovesse essere ancora troppo oneroso).

Inoltre vi consiglio di **stimare il T60 dalla RIR** e **troncare il vettore al numero di samples corrispondente.**

Potete trovare funzioni che calcolano il T60 direttamente su MATLAB Central-File Exchange. Oppure nel modulo experimental di pyroomacoustics se state usando Python.

--------------------------------------------------------------------------------------------------------