Sketching

1 Mixed Poisson Distribution

In its simplest form, a Poisson distribution models the probability of a number of events occurring in a fixed interval, given that we know the average rate (λ) of those events. Its probability mass function is:

$$P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$$

A mixed Poisson distribution occurs when the rate parameter itself, λ, is not a fixed number, but rather a random variable. This introduces an additional level of uncertainty.

- **Standard Poisson:** Imagine a set of very specific documents, such as only soccer game summaries. The word "goal" appears, on average, exactly 3 times per summary. The rate $\lambda = 3$ is fixed.
- **Mixed Poisson:** Now, imagine your dataset contains two types of documents: soccer game summaries (where the average rate of "goal" is $\lambda_{\text{soc}} = 3$) and finance articles (where the word "goal" is less common, perhaps $\lambda_{\text{fin}} = 0.5$). If you randomly pick a document without knowing its type, the rate λ is now a random variable (it could be 3 or 0.5, with some probability). The distribution of the word count for "goal" in this random document is a "mixture" of the two Poisson distributions.

The mixed Poisson distribution takes the following form:

$$X \mid Z = z \sim \bigotimes_{i=1}^{d} \text{Poisson}((\mathbf{Az})_i)$$

- X is a vector of counts with d dimensions: $X = [X_1, X_2, \ldots, X_d]^T$.
- Z is a vector of m latent (hidden) random variables that make up the "mixture".
- $\mathbf{A}$ is a weight matrix with d rows and m columns. Its elements are non-negative. Each row i of $\mathbf{A}$ defines how the latent factors in z combine to form the rate for the count X_i.
- $\mathbf{z}$ is a vector of m non-negative latent factors. This is the specific value assumed by the random variable Z.
- $(\mathbf{A}z)_i$ is the Poisson rate λ for the i-th count X_i. It is calculated by multiplying the i-th row of the matrix $\mathbf{A}$ by the vector z. The complete vector of rates is $\lambda = \mathbf{A}z$.
- $\bigotimes_{i=1}^{d}$ Poisson$(\ldots)$ represents the product of independent distributions. This means that, once we know z, the d counts $(X_1, \ldots, X_d)$ are all statistically independent of each other. Each X_i is drawn from its own Poisson distribution with its own rate $(\mathbf{A}z)_i$.

1.1 Example

Let's model the count of $d = 3$ words in documents, based on $m = 2$ latent topics.

- X_1: count of "calculus"
- X_2: count of "football"
- X_3: count of "investment"

- z_1: exposure to Topic 1 ("Academic")
- z_2: exposure to Topic 2 ("Sports & Finance")
- z_3: exposure to both topics

Let our weight matrix $\mathbf{A}$ be defined as:

$$\mathbf{A} = \begin{pmatrix} 10 & 1 \\ 2 & 8 \\ 1 & 9 \end{pmatrix}$$

Interpretation:

- Row 1 ("calculus"): $\begin{bmatrix} 10 & 1 \end{bmatrix}$ - Strongly associated with Topic 1.
- Row 2 ("football"): $\begin{bmatrix} 2 & 8 \end{bmatrix}$ - Strongly associated with Topic 2.
- Row 3 ("investment"): $\begin{bmatrix} 1 & 9 \end{bmatrix}$ - Strongly associated with Topic 2.

Let's generate samples of X for different documents (different z vectors).

Scenario 1: "Academic" Document

We assume a document with high exposure to Topic 1.

$$z_1 = \begin{pmatrix} 5 \\ 0.1 \end{pmatrix}$$

We calculate the rates $\lambda = \mathbf{A}z_1$:

$$\lambda_1 = \begin{pmatrix} 10 & 1 \\ 2 & 8 \\ 1 & 9 \end{pmatrix} \begin{pmatrix} 5 \\ 0.1 \end{pmatrix} = \begin{pmatrix} 50.1 \\ 10.8 \\ 5.9 \end{pmatrix}$$

The samples of X (word counts) will come from $X_1 \sim \text{Poisson}(50.1)$, $X_2 \sim \text{Poisson}(10.8)$, and $X_3 \sim \text{Poisson}(5.9)$. The "calculus" counts are high, as expected.

Scenario 2: "Sports & Finance" Document

We assume a document with high exposure to Topic 2.

$$z_2 = \begin{pmatrix} 0.5 \\ 10 \end{pmatrix}$$

We calculate the rates $\lambda = \mathbf{A}z_2$:

$$\lambda_2 = \begin{pmatrix} 10 & 1 \\ 2 & 8 \\ 1 & 9 \end{pmatrix} \begin{pmatrix} 0.5 \\ 10 \end{pmatrix} = \begin{pmatrix} 15 \\ 81 \\ 90.5 \end{pmatrix}$$

The samples of X will come from $X_1 \sim \text{Poisson}(15)$, $X_2 \sim \text{Poisson}(81)$, and $X_3 \sim \text{Poisson}(90.5)$. The "football" and "investment" counts are high.

Scenario 3: Document with a Balanced Mixture

We assume a document with moderate exposure to both topics.

$$z_3 = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$$

We calculate the rates $\lambda = \mathbf{A}z_3$:

$$\lambda_3 = \begin{pmatrix} 10 & 1 \\ 2 & 8 \\ 1 & 9 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 34 \\ 38 \\ 39 \end{pmatrix}$$

The samples of X will come from $X_1 \sim \text{Poisson}(34)$, $X_2 \sim \text{Poisson}(38)$, and $X_3 \sim \text{Poisson}(39)$. All counts are similar.

2 Probability-Generating Function (PGF)

For a d-dimensional count vector $\mathbf{X} = (X_1, \ldots, X_d)$, the PGF $G_{\mathbf{X}}(\mathbf{t})$ is defined as the expectation of a product of powers of the components of $\mathbf{t} = (t_1, \ldots, t_d)$:

$$G_{\mathbf{X}}(\mathbf{t}) = \mathbb{E}\left[\prod_{i=1}^{d} t_i^{X_i}\right] = \mathbb{E}\left[e^{\langle \mathbf{X}, \ln(\mathbf{t})\rangle}\right]$$

Considering the expectancy estimation $\hat{\mathbb{E}}[\mathbf{X}] = \frac{1}{n}\sum_{j=1}^{n} \mathbf{X}_j$, where j are the samples of $\mathbf{X}$, it is possible to estimate the PGF from a dataset:

$$\hat{G}_{\mathbf{X}}(\mathbf{t}) = \frac{1}{n}\sum_{j=1}^{n} e^{\langle \mathbf{X}^{(j)}, \ln(\mathbf{t})\rangle}$$

Poisson PGF (Univariate)

For a single univariate Poisson random variable X with a fixed mean (rate) μ, the PGF is defined as $G_X(t) = \mathbb{E}[t^X]$. Using the Poisson probability mass function $P(X = n) = e^{-\mu}\frac{\mu^n}{n!}$, the closed form is:

$$\begin{aligned} G_X(t) &= \sum_{n=0}^{\infty} P(X = n) t^n = \sum_{n=0}^{\infty} e^{-\mu}\frac{\mu^n}{n!} t^n \\ &= e^{-\mu}\sum_{n=0}^{\infty} \frac{(\mu t)^n}{n!} \\ &= e^{-\mu} e^{\mu t} = e^{\mu(t-1)} \end{aligned}$$

Recognizing the Taylor series expansion for the exponential function, $e^a = \sum_{n=0}^{\infty} \frac{a^n}{n!}$ where $a = \mu t$, the PGF simplifies to:

$$G_X(t) = e^{\mu(t-1)}$$

Mixed Poisson PGF (Multivariate)

Let's consider a multivariate observation $\mathbf{X}$ generated by a mixture model, where the conditional distribution of $\mathbf{X}$ given a latent random vector $\mathbf{Z} = \mathbf{z}$ (linked to the rates) is:

$$\mathbf{X}|\mathbf{Z} = \mathbf{z} \sim \bigotimes_{i=1}^{d} \text{Poisson}((\mathbf{A}\mathbf{z})_i)$$

The conditional PGF for the multivariate $\mathbf{X}$ is the product of the independent marginal PGFs, where $\mathbf{A}\mathbf{z}$ is the vector of Poisson rates:

$$G_{\mathbf{X}|\mathbf{Z}=\mathbf{z}}(\mathbf{t}) = \mathbb{E}\left[\prod_{i=1}^{d} t_i^{X_i} \mid \mathbf{Z} = \mathbf{z}\right] = \prod_{i=1}^{d} e^{(\mathbf{A}\mathbf{z})_i (t_i - 1)}$$

This simplifies using vector notation $\langle \cdot, \cdot \rangle$ (and $\mathbf{1}$ as a vector of ones):

$$G_{\mathbf{X}|\mathbf{Z}=\mathbf{z}}(\mathbf{t}) = e^{\langle \mathbf{A}\mathbf{z}, \mathbf{t} - \mathbf{1}\rangle}$$

The mixed Poisson PGF is obtained by taking the expectation of the conditional PGF $G_{\mathbf{X}|\mathbf{Z}=\mathbf{z}}(\mathbf{t})$ over the distribution of the latent mixing variable $\mathbf{Z}$.

Assuming the discrete mixture model defined in the text, where $\mathbf{Z}$ takes one of K fixed values $\mathbf{z}_k$ with probability π_k:

$$p(\mathbf{z}) = \sum_{k=1}^{K} \pi_k \delta_{\mathbf{z}_k}(\mathbf{z})$$

Then,

$$G_{\mathbf{X}}(\mathbf{t}) = \mathbb{E}_{\mathbf{Z}}[G_{\mathbf{X}|\mathbf{Z}=\mathbf{Z}}(\mathbf{t})] = \sum_{k=1}^{K} \pi_k G_{\mathbf{X}|\mathbf{Z}=\mathbf{z}_k}(\mathbf{t})$$

Substituting the conditional PGF yields the closed-form expression for the Mixed Poisson PGF:

$$G_{\mathbf{X}}(\mathbf{t}) = \sum_{k=1}^{K} \pi_k e^{\langle \mathbf{A}\mathbf{z}_k, \mathbf{t}-\mathbf{1} \rangle} = \sum_{k=1}^{K} \pi_k e^{\langle \boldsymbol{\lambda}_k, \mathbf{t}-\mathbf{1} \rangle}$$

where $\boldsymbol{\lambda}_k = \mathbf{A}\mathbf{z}_k$ is the vector of Poisson rates for the k-th mixture component.

3 Sampling Strategy

We sample the Probability-Generating Function (PGF) along complex directions $\mathbf{u}$, scaled by a factor Δ that is around $1/\lambda_{max}$ (highest poisson rate give by $\mathbf{A}\mathbf{z}$). We define the sampling points $\mathbf{t} \in \mathbb{C}^d$ as:

$$\mathbf{t} = \mathbf{1}_d + j\Delta\mathbf{u}$$

Substituting this into the PGF definition yields the sample vector $\mathbf{y}$:

$$\mathbf{y} = G_{\mathbf{X}}(\mathbf{t}) = \sum_{k=1}^{K} \pi_k e^{j\Delta \langle \boldsymbol{\lambda}_k, \mathbf{u} \rangle}$$

Thus, $\mathbf{y}$ represents a sum of damped complex exponentials. Recovering the frequencies $\{\boldsymbol{\lambda}_k\}$ from the observed samples $\{\mathbf{y}\}$ is precisely the spectral estimation problem that the Joint ESPRIT algorithm is designed to solve.

4 The ESPRIT algorithm

It is logical to first define the one dimensional ESPRIT problem before tackling the multidimensional Joint ESPRIT algorithm.

The Signal Model and Array Manifold

We begin by defining the data received by an array of M sensors. Let's assume D narrowband signals, $s_1(t), \ldots, s_D(t)$, impinge on a uniform linear array (ULA) from directions $\theta_1, \ldots, \theta_D$. The signal received at the m-th sensor is a superposition of all D signals plus additive noise $n_m(t)$.

The full $M \times 1$ received signal vector $\mathbf{x}(t)$ is given by:

$$\mathbf{x}(t) = \sum_{k=1}^{D} \mathbf{a}(\theta_k) s_k(t) + \mathbf{n}(t) = \mathbf{A}\mathbf{s}(t) + \mathbf{n}(t)$$

Where:

- $\mathbf{x}(t) = [x_1(t), \ldots, x_M(t)]^T$ is the $M \times 1$ signal vector measured by the M different sensors.
- $\mathbf{s}(t) = [s_1(t), \ldots, s_D(t)]^T$ is the $D \times 1$ vector of source signals.
- $\mathbf{n}(t) = [n_1(t), \ldots, n_M(t)]^T$ is the $M \times 1$ noise vector.
- $\mathbf{A} = [\mathbf{a}(\theta_1) \cdots \mathbf{a}(\theta_D)]$ is the $M \times D$ array manifold matrix.

For a ULA with inter-element spacing d and signal wavelength λ, the steering vector $\mathbf{a}(\theta_k)$ for the k-th signal is:

$$\mathbf{a}(\theta_k) = \begin{bmatrix} 1 \\ e^{j\phi_k} \\ e^{j2\phi_k} \\ \vdots \\ e^{j(M-1)\phi_k} \end{bmatrix} \quad , \text{ where } \phi_k = \frac{2\pi}{\lambda} d \sin(\theta_k)$$

The full manifold matrix $\mathbf{A}$ is formed by concatenating these steering vectors as columns: $\mathbf{A} = [\mathbf{a}(\theta_1) \cdots \mathbf{a}(\theta_D)]$.

Explicitly, this $M \times D$ matrix has a Vandermonde structure:

$$\mathbf{A} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ e^{j\phi_1} & e^{j\phi_2} & \cdots & e^{j\phi_D} \\ e^{j2\phi_1} & e^{j2\phi_2} & \cdots & e^{j2\phi_D} \\ \vdots & \vdots & \ddots & \vdots \\ e^{j(M-1)\phi_1} & e^{j(M-1)\phi_2} & \cdots & e^{j(M-1)\phi_D} \end{bmatrix}$$

The Core Principle: Rotational Invariance

The ESPRIT algorithm's ingenuity lies in its use of a specific array geometry. The array is partitioned into two identical, overlapping subarrays, which we will call X and Y.

- **Subarray X:** Consists of sensors $1, 2, \ldots, (M-1)$.
- **Subarray Y:** Consists of sensors $2, 3, \ldots, M$.

Subarray Y is a perfect copy of Subarray X, but translationally displaced by the distance d. This physical displacement creates a mathematical relationship between their respective manifold matrices, $\mathbf{A}_X$ and $\mathbf{A}_Y$.

$\mathbf{A}_X$ consists of the first $M-1$ rows of $\mathbf{A}$, and $\mathbf{A}_Y$ consists of the last $M-1$ rows of $\mathbf{A}$. Let's look at the k-th column of $\mathbf{A}_Y$ (the steering vector for θ_k on subarray Y):

$$\mathbf{a}_Y(\theta_k) = \begin{bmatrix} e^{j\phi_k} \\ e^{j2\phi_k} \\ \vdots \\ e^{j(M-1)\phi_k} \end{bmatrix} = e^{j\phi_k} \begin{bmatrix} 1 \\ e^{j\phi_k} \\ \vdots \\ e^{j(M-2)\phi_k} \end{bmatrix} = (e^{j\frac{2\pi}{\lambda} d \sin(\theta_k)}) \cdot \mathbf{a}_X(\theta_k)$$

This shows that the steering vector for Subarray Y is just the steering vector for Subarray X multiplied by a complex phase factor $e^{j\phi_k}$.

We can express this relationship for all D signals simultaneously using a $D \times D$ diagonal matrix $\mathbf{\Phi}$:

$$\mathbf{A}_Y = \mathbf{A}_X \mathbf{\Phi} \tag{1}$$

where

$$\mathbf{\Phi} = \operatorname{diag}\left(e^{j\phi_1}, e^{j\phi_2}, \ldots, e^{j\phi_D}\right)$$

This is the rotational invariance property. The DOAs we seek are embedded in the diagonal elements of $\mathbf{\Phi}$.

Subspace Decomposition and Equivalence

The next step is to estimate the signal subspace from the received data. We compute the $M \times M$ covariance matrix $\mathbf{R}_{xx}$, theoretically defined as:

$$\mathbf{R}_{xx} = E\left[\mathbf{x}(t)\mathbf{x}(t)^H\right]$$

In practice, we estimate this matrix from n discrete time samples (snapshots). We form a data matrix $\mathbf{X}$ of size $n \times M$, where each of the n rows contains the M sensor measurements for a given snapshot. The sample covariance matrix $\hat{\mathbf{R}}_{xx}$ is then computed as:

$$\hat{\mathbf{R}}_{xx} = \frac{1}{n}\mathbf{X}^T\mathbf{X}^*$$

Assuming the signals and noise are uncorrelated ($E[\mathbf{s}\mathbf{n}^H] = \mathbf{0}$) and the noise is spatially white ($E[\mathbf{n}\mathbf{n}^H] = \sigma^2\mathbf{I}$), the theoretical covariance matrix simplifies to:

$$\mathbf{R}_{xx} = E\left[(\mathbf{A}\mathbf{s}(t) + \mathbf{n}(t))(\mathbf{A}\mathbf{s}(t) + \mathbf{n}(t))^H\right] = \mathbf{A}E[\mathbf{s}(t)\mathbf{s}(t)^H]\mathbf{A}^H + \sigma^2\mathbf{I} = \mathbf{A}\mathbf{R}_{ss}\mathbf{A}^H + \sigma^2\mathbf{I}$$

We perform an eigendecomposition on our estimated matrix $\mathbf{R}_{xx}$. The matrix $\mathbf{R}_{xx}$ has M eigenvalues.

- D eigenvalues will be $\lambda_k + \sigma^2$, where λ_k are the eigenvalues of $\mathbf{A}\mathbf{R}_{ss}\mathbf{A}^H$.
- $M - D$ eigenvalues will be equal to σ^2.

The D eigenvectors corresponding to the largest eigenvalues form the basis for the **Signal Subspace**, $\mathbf{E}_S$. The other $M - D$ eigenvectors form the **Noise Subspace**, $\mathbf{E}_N$.

Crucially, the eigenvectors of $\mathbf{R}_{xx}$ are the same as the eigenvectors of $\mathbf{A}\mathbf{R}_{ss}\mathbf{A}^H$. The column space of $\mathbf{A}\mathbf{R}_{ss}\mathbf{A}^H$ is identical to the column space of $\mathbf{A}$ (assuming D non-correlated signals). Therefore, the signal subspace and the array manifold span the **same D-dimensional space**.

$$\operatorname{span}(\mathbf{E}_S) = \operatorname{span}(\mathbf{A})$$

This means there exists a unique, $D \times D$ invertible matrix $\mathbf{T}$ that relates them:

$$\mathbf{E}_S = \mathbf{A}\mathbf{T} \tag{2}$$

This $\mathbf{T}$ matrix is unknown, but as we will see, we don't need to find it.

Deriving the Solution

Now we tie everything together. We partition our estimated signal subspace $\mathbf{E}_S$ into two $(M-1) \times D$ matrices, just as we did with the array manifold:

- $\mathbf{E}_X$: First $M - 1$ rows of $\mathbf{E}_S$.
- $\mathbf{E}_Y$: Last $M - 1$ rows of $\mathbf{E}_S$.

Let's substitute $\mathbf{E}_S = \mathbf{A}\mathbf{T}$ (2) into these new matrices.

$$\mathbf{E}_X = \mathbf{J}_X\mathbf{E}_S = (\mathbf{J}_X\mathbf{A})\mathbf{T} = \mathbf{A}_X\mathbf{T} \tag{3}$$

$$\mathbf{E}_Y = \mathbf{J}_Y\mathbf{E}_S = (\mathbf{J}_Y\mathbf{A})\mathbf{T} = \mathbf{A}_Y\mathbf{T} \tag{4}$$

(where $\mathbf{J}_X$ and $\mathbf{J}_Y$ are the selection matrices of 1s and 0s that select the rows).

Now, we use the rotational invariance property $\mathbf{A}_Y = \mathbf{A}_X\mathbf{\Phi}$ (1) by substituting it into (4):

$$\mathbf{E}_Y = (\mathbf{A}_X\mathbf{\Phi})\mathbf{T} = \mathbf{A}_X\mathbf{\Phi}\mathbf{T} \tag{5}$$

From (3), we can write $\mathbf{A}_X = \mathbf{E}_X\mathbf{T}^{-1}$ (since $\mathbf{E}_X$ and $\mathbf{T}$ are invertible). Let's substitute this into (5):

$$\mathbf{E}_Y = (\mathbf{E}_X\mathbf{T}^{-1})\mathbf{\Phi}\mathbf{T}$$

By re-grouping the matrices, we get the central ESPRIT equation:

$$\mathbf{E}_Y = \mathbf{E}_X(\mathbf{T}^{-1}\mathbf{\Phi}\mathbf{T})$$

Let us define a new $D \times D$ matrix $\mathbf{\Psi} = \mathbf{T}^{-1}\mathbf{\Phi}\mathbf{T}$. Our equation becomes:

$$\mathbf{E}_Y = \mathbf{E}_X\mathbf{\Psi}$$

Since $\mathbf{E}_X$ and $\mathbf{E}_Y$ are known (estimated from our data), we can solve this overdetermined system for $\mathbf{\Psi}$. This is typically done using a Total Least Squares (TLS) solution, which is robust to noise.

$$\mathbf{\Psi}_{TLS} = \text{argmin}_{\mathbf{\Psi}}\|[\mathbf{E}_X \mid \mathbf{E}_Y]\|_F$$

Finding the DOAs from Eigenvalues

The final step is to retrieve the DOAs. We have our computed matrix $\mathbf{\Psi}$. The equation $\mathbf{\Psi} = \mathbf{T}^{-1}\mathbf{\Phi}\mathbf{T}$ is a **similarity transformation**. A fundamental theorem of linear algebra states that similar matrices have the exact same eigenvalues.

Therefore:

$$\text{eigenvalues}(\mathbf{\Psi}) = \text{eigenvalues}(\mathbf{\Phi}) = \{e^{j\phi_1}, e^{j\phi_2}, \ldots, e^{j\phi_D}\}$$

So, by computing the D eigenvalues of our estimated matrix $\mathbf{\Psi}$, $\{\lambda_1, \ldots, \lambda_D\}$, we have found the D phase factors:

$$\lambda_k = e^{j\phi_k} = e^{j\frac{2\pi}{\lambda}d\sin(\theta_k)}$$

We can now find each DOA θ_k by taking the angle of the corresponding eigenvalue:

$$\arg(\lambda_k) = \phi_k = \frac{2\pi}{\lambda}d\sin(\theta_k)$$

$$\theta_k = \arcsin\left(\frac{\lambda \cdot \arg(\lambda_k)}{2\pi d}\right)$$

This gives us the D direction of arrivals without any computationally expensive spectral search, which is the main advantage of ESPRIT.

5 The Joint ESPRIT Algorithm

Signal Model

We consider a d-dimensional signal $\boldsymbol{y}[\boldsymbol{n}]$ composed of r complex components. The signal at any d-dimensional coordinate $\boldsymbol{n} \in \mathbb{Z}^d$ is:

$$\boldsymbol{y}[\boldsymbol{n}] = \sum_{k=1}^{r} a_k e^{j\langle \boldsymbol{\omega}_k, \boldsymbol{n} \rangle} + \boldsymbol{w}[\boldsymbol{n}]$$

where:

- $a_k \in \mathbb{C}$ is the complex amplitude of component k.
- $\boldsymbol{\omega}_k \in [-\pi, \pi)^d$ is the unknown d-dimensional frequency vector we want to find.
- $\boldsymbol{w}[\boldsymbol{n}]$ is additive noise.

Acquisition Design

Instead of sampling the entire d-Dimensional space, we sample it along a set of M 1D lines, where $M \geq d$.

1. Choose a set of M direction vectors, $\mathcal{U} = \{\boldsymbol{u}_l\}_{l=1}^{M}$, where $\boldsymbol{u}_l \in \mathbb{Z}^d$.
2. For each direction $\boldsymbol{u}_l$, choose S_l random "base points" $\{\boldsymbol{p}_{l,s}\}_{s=1}^{S_l}$, with $S_l \geq r$.
3. For each pair (l, s), sample N_l points along the line $\mathcal{L}_{l,s}$, with $N_l \geq r + 1$.

$$\mathcal{L}_{l,s} = \{\boldsymbol{p}_{l,s} + n\boldsymbol{u}_l : n = 0, \ldots, N_l - 1\}$$

This strategy reduces the d-D problem to a set of M 1D problems, one for each direction $\boldsymbol{u}_l$.

5.1 Step 1: Per-Direction Model (Multi-snapshot 1D)

For a single direction l, we analyze the signal $z_{l,s}[n]$ sampled along the line (l, s). By substituting the line coordinate $\boldsymbol{n}' = \boldsymbol{p}_{l,s} + n\boldsymbol{u}_l$ into the signal model:

$$z_{l,s}[n] = \sum_{k=1}^{r} a_k e^{j\langle \boldsymbol{\omega}_k, \boldsymbol{p}_{l,s} + n\boldsymbol{u}_l \rangle} + w_{l,s}[n]$$

$$z_{l,s}[n] = \sum_{k=1}^{r} \left(a_k e^{j\langle \boldsymbol{\omega}_k, \boldsymbol{p}_{l,s} \rangle} \right) \left(e^{j\varphi_{k,l}} \right)^n + w_{l,s}[n]$$

This is a standard 1D multi-snapshot model. The term $\varphi_{k,l} = \langle \boldsymbol{\omega}_k, \boldsymbol{u}_l \rangle$ is the 1D frequency for this direction; it is the scalar projection of the d-D frequency vector $\boldsymbol{\omega}_k$ onto the 1D direction vector $\boldsymbol{u}_l$. We can stack all S_l snapshots (from the S_l base points) into a matrix equation:

$$\mathbf{Z}_l = \mathbf{V}_l \mathbf{C}_l + \mathbf{W}_l$$

where $\mathbf{Z}_l$ is the $N_l \times S_l$ measurement matrix, $\mathbf{V}_l$ is the $N_l \times r$ frequency matrix, $\mathbf{C}_l$ is the $r \times S_l$ amplitude matrix, and $\mathbf{W}_l$ is the noise matrix. Explicitly, the matrices are:

$$\mathbf{Z}_l = \begin{pmatrix} z_{l,1}[0] & z_{l,2}[0] & \cdots & z_{l,S_l}[0] \\ z_{l,1}[1] & z_{l,2}[1] & \cdots & z_{l,S_l}[1] \\ \vdots & \vdots & \ddots & \vdots \\ z_{l,1}[N_l-1] & z_{l,2}[N_l-1] & \cdots & z_{l,S_l}[N_l-1] \end{pmatrix}$$

$$\mathbf{V}_l = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ e^{j\varphi_{1,l}} & e^{j\varphi_{2,l}} & \cdots & e^{j\varphi_{r,l}} \\ e^{j2\varphi_{1,l}} & e^{j2\varphi_{2,l}} & \cdots & e^{j2\varphi_{r,l}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{j(N_l-1)\varphi_{1,l}} & e^{j(N_l-1)\varphi_{2,l}} & \cdots & e^{j(N_l-1)\varphi_{r,l}} \end{pmatrix} \quad \text{where } \varphi_{k,l} = \langle \boldsymbol{\omega}_k, \boldsymbol{u}_l \rangle$$

$$\mathbf{C}_l = \begin{pmatrix} a_1 e^{j\langle \boldsymbol{\omega}_1, \boldsymbol{p}_{1,l}\rangle} & a_1 e^{j\langle \boldsymbol{\omega}_1, \boldsymbol{p}_{2,l}\rangle} & \cdots & a_1 e^{j\langle \boldsymbol{\omega}_1, \boldsymbol{p}_{S_l,l}\rangle} \\ a_2 e^{j\langle \boldsymbol{\omega}_2, \boldsymbol{p}_{1,l}\rangle} & a_2 e^{j\langle \boldsymbol{\omega}_2, \boldsymbol{p}_{2,l}\rangle} & \cdots & a_2 e^{j\langle \boldsymbol{\omega}_2, \boldsymbol{p}_{S_l,l}\rangle} \\ \vdots & \vdots & \ddots & \vdots \\ a_r e^{j\langle \boldsymbol{\omega}_r, \boldsymbol{p}_{1,l}\rangle} & a_r e^{j\langle \boldsymbol{\omega}_r, \boldsymbol{p}_{2,l}\rangle} & \cdots & a_r e^{j\langle \boldsymbol{\omega}_r, \boldsymbol{p}_{S_l,l}\rangle} \end{pmatrix}$$

5.2 Step 2: Subspace Estimation

For each direction l, we compute the spatial covariance matrix and find the signal subspace.

1. Compute covariance: $\hat{\mathbf{R}}_l = \frac{1}{S_l}\mathbf{Z}_l\mathbf{Z}_l^H$.
2. Compute the SVD of $\hat{\mathbf{R}}_l$ and extract the r dominant eigenvectors to form the $N_l \times r$ signal subspace matrix, $\mathbf{U}_{s,l}$.

5.3 Step 3: 1D ESPRIT per Direction

We apply the core ESPRIT rotational invariance principle to each $\mathbf{U}_{s,l}$ independently.

1. Partition $\mathbf{U}_{s,l}$ into two $(N_l - 1) \times r$ matrices:
   - $\mathbf{U}_1 = \mathbf{U}_{s,l}$ with the last row removed.
   - $\mathbf{U}_2 = \mathbf{U}_{s,l}$ with the first row removed.
2. Solve the linear system $\mathbf{U}_2 \approx \mathbf{U}_1\mathbf{\Psi}_l$ (e.g., via least-squares) to find the $r \times r$ "shuffled" matrix $\mathbf{\Psi}_l$.

At the end of this step, we have M calculated, "shuffled" matrices: $\{\mathbf{\Psi}_1, \mathbf{\Psi}_2, \ldots, \mathbf{\Psi}_M\}$. This is the dead end of the naive approach. If we call a standard eigenvalue solver on each $\mathbf{\Psi}_l$, it will return r eigenvalues. Crucially, it returns them in a numerical order (e.g., sorted by magnitude), not in their true physical order (index $k = 1, \ldots, r$). Because the eigenvalues for each $\mathbf{\Psi}_l$ have different magnitudes, this numerical sorting scrambles the component order differently for each direction, and we are left with the "pairing problem".

5.4 Step 4: Joint ESPRIT (Simultaneous Diagonalization)

From the 1D ESPRIT case, we know the relationship between the calculated matrix $\mathbf{\Psi}_l$ and its corresponding diagonal matrix of phase factors $\mathbf{\Phi}_l$ is a similarity transformation $\mathbf{\Psi}_l = \mathbf{T}^{-1}\mathbf{\Phi}_l\mathbf{T}$.

The core theory of Joint ESPRIT states that, in the absence of noise, this similarity transform $\mathbf{T}$ is common to all M directions. Therefore, all M matrices $\{\mathbf{\Psi}_1, \ldots, \mathbf{\Psi}_M\}$ share a common set of eigenvectors (the columns of $\mathbf{T}^{-1}$).

The "pairing problem" noted in Step 3 arises from this observation: if one applies a standard eigenvalue solver independently to each $\mathbf{\Psi}_l$, the solver will find the correct eigenvalues but return them in a numerical order (e.g., sorted by magnitude). This independent sorting breaks the common physical ordering of the components, making it impossible to pair the 1D frequencies across different directions.

Joint ESPRIT avoids this ambiguity by not finding the eigenvalues of each $\boldsymbol{\Psi}_l$ separately. Instead, it takes the entire set of "shuffled" matrices $\{\boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_M\}$ and searches for the single transformation matrix $\hat{\mathbf{T}}$ that best diagonalizes all of them simultaneously.

We solve the optimization problem:

$$\hat{\mathbf{T}} = \arg\min_{\mathbf{T}} \sum_{l=1}^{M} \left\| \text{offdiag}(\hat{\mathbf{T}}\boldsymbol{\Psi}_l\hat{\mathbf{T}}^{-1}) \right\|_F^2$$

This optimization finds the single $\hat{\mathbf{T}}$ that minimizes the off-diagonal elements across all matrices. This process is a form of simultaneous eigendecomposition and is formally known as **Joint Diagonalization**.

This algorithm finds the single "best-fit" un-shuffling matrix $\hat{\mathbf{T}}$ that is common to all M directions. This $\hat{\mathbf{T}}$ is our estimate for the "true" transformation matrix, and its inverse, $\hat{\mathbf{T}}^{-1}$, contains the correctly-ordered, common eigenvectors as its columns.

5.5 Step 5: Reconstruct d-D Frequencies

Now that we have the common un-shuffler $\hat{\mathbf{T}}$, we can find the paired frequencies.

1. **Un-shuffle the matrices:** We apply the common un-shuffler $\hat{\mathbf{T}}$ to all M of our calculated $\boldsymbol{\Psi}_l$ matrices to get the diagonalized "goal" matrices, $\hat{\boldsymbol{\Phi}}_l$.

$$\hat{\boldsymbol{\Phi}}_l = \hat{\mathbf{T}}\boldsymbol{\Psi}_l\hat{\mathbf{T}}^{-1}$$

2. **Read the paired frequencies:** Because we used the single, common un-shuffler $\hat{\mathbf{T}}$, the diagonal elements of all the $\hat{\boldsymbol{\Phi}}_l$ matrices are now correctly paired. The k-th diagonal element of $\hat{\boldsymbol{\Phi}}_1$ corresponds to the k-th diagonal element of $\hat{\boldsymbol{\Phi}}_2$, and so on. We find the 1D frequencies $\hat{\varphi}_{k,l}$ by taking the angle of the k-th diagonal element of $\hat{\boldsymbol{\Phi}}_l$:

$$\hat{\varphi}_{k,l} = \arg\left((\hat{\boldsymbol{\Phi}}_l)_{kk}\right)$$

3. **Phase Unwrapping:** It is necessary to "unwrap" the $\hat{\varphi}_{k,l}$ values since they were recovered using the arg() function, which only returns a "wrapped" value in the principal range (e.g., $(-\pi, \pi]$). Phase unwrapping algorithms correct these jumps by adding or subtracting multiples of 2π to restore a smooth sequence, $\tilde{\varphi}_k$. This process relies on a key assumption: the true frequency projections $\langle \boldsymbol{\omega}_k, \boldsymbol{u}_l \rangle$ change smoothly (by less than π) between adjacent direction vectors. This assumption is only met if the set of M direction vectors is sufficiently dense, which is why M is chosen to be greater than d ($M > d$). This redundancy provides a dense sampling of the direction space, ensuring the projections can be unwrapped reliably.

4. **Reconstruct $\boldsymbol{\omega}_k$:** For each component k, we now have its vector of M unwrapped 1D frequency projections, $\tilde{\boldsymbol{\varphi}}_k = [\tilde{\varphi}_{k,1}, \ldots, \tilde{\varphi}_{k,M}]^T$. We find the d-D vector $\boldsymbol{\omega}_k$ by solving the linear system $\tilde{\boldsymbol{\varphi}}_k \approx \mathbf{U}\boldsymbol{\omega}_k$. Explicitly, this is:

$$\underbrace{\begin{pmatrix} \tilde{\varphi}_{k,1} \\ \tilde{\varphi}_{k,2} \\ \vdots \\ \tilde{\varphi}_{k,M} \end{pmatrix}}_{\tilde{\boldsymbol{\varphi}}_k \text{ (known)}} \approx \underbrace{\begin{pmatrix} — \boldsymbol{u}_1 — \\ — \boldsymbol{u}_2 — \\ \vdots \\ — \boldsymbol{u}_M — \end{pmatrix}}_{\mathbf{U} \text{ (known)}} \underbrace{\begin{pmatrix} \omega_{k,1} \\ \vdots \\ \omega_{k,d} \end{pmatrix}}_{\boldsymbol{\omega}_k \text{ (unknown)}}$$

We solve this $M \times d$ system using the unwrapped vector $\tilde{\boldsymbol{\varphi}}_k$:

$$\hat{\boldsymbol{\omega}}_k = (\mathbf{U}^T\mathbf{U})^{-1}\mathbf{U}^T\tilde{\boldsymbol{\varphi}}_k$$

5.6 Step 6: Amplitude Estimation

Now that we have the d-D frequencies $\hat{\boldsymbol{\omega}}_k$ for all components, the only unknowns are the original amplitudes a_k. We solve for them by setting up one final global least-squares problem:

$$\hat{\boldsymbol{a}} = \arg\min_{\boldsymbol{a}} \sum_{\boldsymbol{n} \in \text{all sampled points}} \left| z[\boldsymbol{n}] - \sum_{k=1}^{r} a_k e^{j\langle \hat{\boldsymbol{\omega}}_k, \boldsymbol{n} \rangle} \right|^2$$

This finds the r amplitudes that best fit all the data we collected.

6 Algorithm Improvements and Stability Analysis

6.1 Issue Identification: Component Mixing and Basis Mismatch

During the evaluation of the JESPRIT algorithm, a significant issue was observed where the estimated rates ($\boldsymbol{\lambda}$) exhibited "mass stealing" between components. For example, in a scenario with true rates of 100 and 100, the algorithm would consistently estimate approximately 75 and 126. This systematic error persisted even with a large number of samples, indicating a bias rather than variance.

The root cause was identified in the Joint Diagonalization step. The standard JESPRIT formulation employs a joint diagonalization algorithm (typically based on Jacobi rotations) that explicitly searches for a unitary matrix $\boldsymbol{V}$ (i.e., $\boldsymbol{V}\boldsymbol{V}^H = \boldsymbol{I}$) to simultaneously diagonalize the set of Rotational Invariance Matrices $\boldsymbol{\Psi}_l$.

However, in the general case of mixed Poisson estimation, especially when the steering vectors are not orthogonal or the components are correlated, the true diagonalizing matrix $\boldsymbol{T}$ (which relates the signal subspace to the canonical basis) is not necessarily unitary. Since the standard algorithm is constrained to the manifold of unitary matrices, it cannot recover the correct non-unitary transformation $\boldsymbol{T}$. Instead, it finds the "closest" unitary approximation, which results in a compromised solution that effectively "mixes" the components, leading to the observed bias and "mass stealing".

6.2 Proposed Solution: Brute-Force Eigenvalue Matching

To resolve this, we replaced the joint diagonalization approach with a Brute-Force Eigenvalue Matching strategy. This method acknowledges that while the eigenvectors may differ across directions, the eigenvalues (which contain the frequency information) are intrinsic to the underlying components.

The improved algorithm proceeds as follows:

1. **Independent Diagonalization:** Instead of attempting to find a common basis, we compute the eigendecomposition of each $\boldsymbol{\Psi}_l$ independently:

   $$\boldsymbol{\Psi}_l = \boldsymbol{V}_l \boldsymbol{D}_l \boldsymbol{V}_l^{-1}$$

   where $\boldsymbol{D}_l$ is a diagonal matrix containing the eigenvalues $\mu_{l,k} = \exp(j\delta \boldsymbol{u}_l^\top \boldsymbol{\omega}_k)$. This avoids the error introduced by enforcing a common basis.

2. **Basis Selection:** We select a subset of d linearly independent directions (where d is the dimension of the rate vector) to form a basis. Let these indices be $l_1, \ldots, l_d$.

3. **Permutation Search:** Since the eigenvalues in $\boldsymbol{D}_l$ are arbitrarily ordered for each l, we must determine the correct correspondence between directions. We fix the ordering for the first direction l_1. For the remaining $d-1$ basis directions, we iterate through all possible $r!$ permutations (where r is the rank/number of components).

   For each permutation hypothesis, we construct a candidate frequency vector $\hat{\boldsymbol{\omega}}$ by solving the linear system defined by the basis directions.

4. **Validation:** We validate each candidate $\hat{\boldsymbol{\omega}}$ against the remaining $M-d$ directions. We compute the predicted phase shifts $\delta \boldsymbol{u}_l^\top \hat{\boldsymbol{\omega}}$ and compare them with the observed eigenvalues of $\boldsymbol{\Psi}_l$ (under the best matching permutation for that direction). The hypothesis that minimizes the total angular error across all directions is selected.

5. **Final Estimation:** Once the correct permutations are determined for all directions, we form the "unwrapped" phase matrix and solve for $\boldsymbol{\omega}$ using the full set of M directions via least squares.

6.3 Results

The implementation of this matching strategy yielded a dramatic improvement in estimation accuracy. In the test case where the original algorithm produced a rate error of approximately 11.11, the improved algorithm achieved a rate error of **0.89**. The weight error decreased from 0.12 to **0.005**. Crucially, the "mass stealing" effect was completely eliminated, with the algorithm correctly identifying the distinct components.

This approach is computationally feasible for small to moderate r (since the complexity scales with $(r!)^{d-1}$), which covers most practical applications of this method.