# Package: SVMD (via r-universe)

November 16, 2024

**Type** Package

**Title** Spearman Variational Mode Decomposition

**Version** 0.1.0

**Author** Dr. Himadri Shekhar Roy [aut, cre], Dr. Ranjit Kumar Paul [aut], Dr. Chiranjit Mazumder [aut], Dr. Kamalika Nath [aut], Dr. Prakash Kumar [aut]

**Maintainer** Dr. Himadri Shekhar Roy <himadriiasri@gmail.com>

**Description** In practice, it is difficult to determine the number of decomposition modes, K, for Variational Mode Decomposition (VMD). To overcome this issue, this study offers Spearman Variational Mode Decomposition (SVMD), a method that uses the Spearman correlation coefficient to calculate the ideal mode number. Unlike the Pearson correlation coefficient, which only returns a perfect value when X and Y are linearly connected, the Spearman correlation can be calculated without knowing the probability distributions of X and Y. The Spearman correlation coefficient, also called Spearman's rank correlation coefficient, is a subset of a wider correlation coefficient. As VMD decomposes a signal, the Spearman correlation coefficient between the reconstructed and original sequences rises as the mode number K increases. Once the signal has been fully decomposed, subsequent increases in K cause the correlation to gradually level off. When the correlation reaches a specific level, VMD is said to have adequately decomposed the signal. Numerous experiments revealed that a threshold of 0.997 produces the best denoising effect, so the threshold is set at 0.997. This package has been developed using concept of Yang et al. (2021)<doi:10.1016/j.aej.2021.01.055>.

**License** GPL-3

**Encoding** UTF-8

**Imports** VMDecomp, stats

**NeedsCompilation** no

**RoxygenNote** 7.3.1

**Date/Publication** 2024-09-16 14:30:06 UTC
**Repository** https://himadriiasri.r-universe.dev
**RemoteUrl** https://github.com/cran/SVMD
**RemoteRef** HEAD
**RemoteSha** d585dad91441b801f5308e0cb92c234542dff657

# Contents



---

`sVMD` *Spearman Variational Mode Decomposition*

---

## Description

Optimal number of modes of Variational Mode Decomposition (VMD) using Spearman's rank correlation coefficient

## Usage

```
sVMD(
  series,
  alpha = 2000,
  tau = 0,
  DC = FALSE,
  init = 1,
  tol = 1e-07,
  threshold = 0.997,
  max_modes = 10,
  verbose = FALSE
)
```

## Arguments

| | |
|---|---|
| `series` | The input time series signal to be decomposed. |
| `alpha` | The balancing parameter of the data-fidelity constraint. Default is 2000. |
| `tau` | Time-step of the dual ascent (pick 0 for noise-slack). Default is 0. |
| `DC` | If TRUE, the first mode is put and kept at DC (0 frequency). Default is FALSE. |
| `init` | Mode initialization (1 = all omegas start at 0). Default is 1. |
| `tol` | Convergence tolerance criterion. Default is 1e-7. |
| `threshold` | The correlation coefficient threshold to determine the optimal number of modes. Default is 0.997. |
| `max_modes` | The maximum number of modes to consider. Default is 10. |
| `verbose` | Logical, if TRUE, prints detailed messages about the decomposition process. |

## Value

Returns a list containing the optimal number of modes, reconstructed signal, and additional outputs from the VMD process:

- `optimal_K`: The optimal number of modes.
- `reconstructed_signal`: The reconstructed signal from the selected modes.
- `imfs`: Intrinsic Mode Functions (IMFs) obtained from SVMD.
- `u_hat`: Estimated envelopes of the modes.
- `omega`: Frequencies of the modes.

## References

Yang, H., Cheng, Y., and Li, G. (2021). A denoising method for ship radiated noise based on Spearman variational mode decomposition, spatial-dependence recurrence sample entropy, improved wavelet threshold denoising, and Savitzky-Golay filter. Alexandria Engineering Journal, 60(3), 3379-3400

## Examples

```
{
# Example data generation:
# Set the number of observations
N <- 300
# Set a random seed for reproducibility
set.seed(123)
# Generate random uniform values
rand_unif <- runif(n = N, min = 0, max = 1.0)
# Create the components of the time series
sig1 <- 6 * rand_unif
sig2 <- sin(8 * pi * rand_unif)  # Using sine function
sig3 <- 0.5 * sin(40 * pi * rand_unif)  # Using sine function
# Combine the components to form the final signal
signal <- sig1 + sig2 + sig3
# Apply the sVMD function to the signal
result <- sVMD(signal)
}
```

# Index