Computing Sparse Autoencoders and Autoregressors for Signal Identification

Fredy Vides [*]

[*] *Scientific Computing Innovation Center, Universidad Nacional Autónoma de Honduras, Tegucigalpa, Honduras, (e-mail: fredy.vides@unah.edu.hn)*

Abstract: The theory and algorithms corresponding to the computation of sparse autoencoders and autoregressors for signal modeling and identification are presented. Some prototypical computational implementations are presented as well.

Keywords: Autoencoder, autoregressor, sparse representation, discrete Fourier transform, time series.

1. INTRODUCTION

The theory and algorithms corresponding to the computation of sparse autoencoders and autoregressors for signal modeling and identification are presented. Some prototypical computational implementations are presented as well.

2. PRELIMINARIES

Given $\delta > 0$, let us consider the function defined by the expression

$$H_\delta(x) = \begin{cases} 1, & x > \delta \\ 0, & x \leq \delta \end{cases}.$$

Given a matrix $A \in \mathbb{C}^{m\times n}$ with singular values denoted by the expressions $s_j(A)$ for $j = 1, \ldots, \min\{m, n\}$. We will write $\mathrm{rk}_\delta(A)$ to denote the number

$$\mathrm{rk}_\delta(A) = \sum_{j=1}^{\min\{m,n\}} H_\delta(s_j(A)).$$

Given an ordered sample $\Sigma_N = \{x_t\}_{t=1}^N \subset \Sigma$ from a time series $\Sigma = \{\hat{x}_t\}_{t\geq 1}$, we will write $\mathcal{H}_L(\Sigma_N)$ to denote the Hankel type trajectory matrix corresponding to Σ_N, that is determined by the following expression.

$$\mathcal{H}_L(\Sigma_N) = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_{N-L+1} \\ x_2 & x_3 & x_4 & \cdots & x_{N-L+2} \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ x_L & x_{L+1} & x_{L+2} & \cdots & x_N \end{bmatrix}$$

In this document we will write $\hat{e}_{j,n}$ to denote the matrices in $\mathbb{C}^{n\times 1}$ representing the canonical basis of $\mathbb{C}^n$ (each $\hat{e}_{j,n}$ is the j-column of I_n), that are determined by the expressions

$$\hat{e}_{j,n} = [\delta_{1,j}\ \delta_{2,j}\ \cdots\ \delta_{n-1,j}\ \delta_{n,j}]^\top \quad (1)$$

for each $1 \leq j \leq n$, where $\delta_{k,j}$ is the Kronecker delta determined by the expression.

$$\delta_{k,j} = \begin{cases} 1, & k = j \\ 0, & k \neq j \end{cases} \quad (2)$$

A matrix $P \in \mathbb{C}^{n\times n}$ will be called an orthogonal projector whenever $P^2 = P = P^*$. We will write $\mathbb{S}^1$ to denote the set $\{z \in \mathbb{C} : |z| = 1\}$.

3. DISCRETE FOURIER TRANSFORMS AND SPARSE AUTOENCODERS

In this section we will consider sparse autoencoders, that for the purpose of this study can be considered as models of the form

$$\begin{cases} \mathbf{h} = \boldsymbol{\tau}(W\mathbf{x} + \mathbf{b}) \\ \hat{\mathbf{x}} = W^+\mathbf{h} + \mathbf{c} \end{cases}, \quad (3)$$

for any x in a given submanifold $\mathcal{N} \subset \mathbb{C}^m$, for some positive integer m. We will say that an autoencoder of the form (3) is an autoencoder relative to the manifold $\mathcal{N}$. Schematically the autoencoders considered in this study can by described by the following block diagram.

$$\begin{tikzcd} {} \arrow[r, "x_t"] & \boxed{\mathfrak{E}} \arrow[r, "y_t"] & \boxed{\mathfrak{D}_t} \arrow[r, "\hat{x}_t"] & {} \end{tikzcd}$$

From a topological perspective the autoencoders considered in this document can be describied by the following commutative diagram.

$$\begin{tikzcd} \mathbb{V}^m \arrow[r, dashed, "c"] \arrow[rr, bend left, "id_{\mathbb{V}^m}", "\approx"'] \arrow[dr, hook, "\imath"'] & \mathbf{S}^1 \arrow[r, dashed, "d"] \arrow[d, "e"] & \mathbb{V}^m \arrow[dl, hook', "\imath"] \\ & \mathcal{M}^m & \end{tikzcd}$$

In (3) the matrix W, the vectors $\mathbf{b}, \mathbf{c}$ and the thresholding function $\boldsymbol{\tau}$ are to be determined based on some given sparsity constraints, and W^+ denotes the pseudoinverse of W. In principle, the function defined by the expression $\boldsymbol{\mathcal{T}}(\mathbf{x}) = W^+\boldsymbol{\tau}(W\mathbf{x} + \mathbf{b}) + \mathbf{c}$ needs to provide an approximation of the identity map restricted to the submanifold $\mathcal{N} \subset \mathbb{C}^m$ for some positive integer m.

In this study we will focus on cases where the matrix W in (3) satisfies the constraint $W^+ = W^*$, where W^* denotes the conjugate transpose of W. Of particular interest are the cases where W is corrsponds to a submatrix of a unitary discrete transform matrix, like discrete Fourier transform, for instance.

In cases where the matrix W corresponds to a submatrix of the discrete Fourier transform, we will build on the algorithms presented in Vides (2021b) to obtain *matrix free* algorithms for the computation of autoencoders of the form (3).

3.1 A Projective Approach to DFT Autoencoders

Theorem 1. Given a DFT based sparse autoencoder of the form (3) relative to a submanifold $\mathcal{N} \subset \mathbb{C}^m$, there is an orthogonal projector $P \in \mathbb{C}^{m\times m}$ such that $Px \approx x$ for each y un a small enough neighborhood of $\mathbf{x}$ in $\mathcal{N}$.

Proof. (Sketch.) One just needs to consider the restriction $W = \mathbf{F}_m|_\Gamma$ corresponding to the standard discrete Fourier transform matrix $\mathbf{F}_m$ to the support of $\mathbf{x}$, then form $P = WW^*$.

4. SPARSE AUTOREGRESSORS

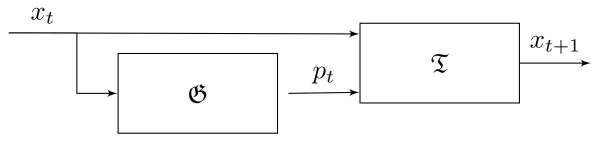
Given a time series $\Sigma = \{x_t\}_{t\geq 1}$ and a lag value $L > 0$, a linear autoregressive model of the form

$$x_{t+1} \approx c_0x_t + c_1x_{t-1} + c_2x_{t-2} + \cdots + c_Lx_{t-L+1}, t \geq L$$

can be computed using a samples $\Sigma_0 = \{x_t\}_{t=1}^{N-1}$ and $\Sigma_1 = \{x_t\}_{t=2}^{N}$ for some suitable $N > L$, by approximately solving the matrix equation

$$\mathcal{H}_L(\Sigma_0)^\top \begin{bmatrix} c_L \\ c_{L-1} \\ \vdots \\ c_1 \\ c_0 \end{bmatrix} = \begin{bmatrix} x_{L+1} \\ x_{L+2} \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix}. \quad (4)$$

Schematically the autoregressors considered in this study can by described by the following block diagram.



4.1 A Projective Approach to Autoregressors

Given a time series $\Sigma = \{x_t\}_{t\geq 1}$ and a lag value $L > 0$, whos dynamical behavior can be approximately identified by a linear autoregressive model of the form

$$x_{t+1} \approx c_0x_t + c_1x_{t-1} + c_2x_{t-2} + \cdots + c_Lx_{t-L+1}, t \geq L.$$

As a consequence of (4), if we consider any sample $\Sigma_M \subset \Sigma$ of suitable size $M = N-1 > L$, such that the states in Σ_M are sucessors of the states in $\Sigma_0 = \{x_t\}_{t=1}^{N-1}$, which means that, for every $\tilde{x}_t \in \Sigma_M$, there is a nonnegative integer S and an element $x_t \in \Sigma_0$ such that $x_{t+S} = \tilde{x}_t$. We will have that there is a matrix $T \in \mathbb{C}^{L\times L}$ such that

$$\mathcal{H}_L(\Sigma_0)^\top T = \mathcal{H}_L(\Sigma_M)^\top. \quad (5)$$

As a consequence of (5), (Vides, 2021b, Theorem 3.6) and (Vides, 2021b, Theorem 4.3) we can obtain the following result.

Theorem 2. Given a time series $\Sigma = \{x_t\}_{t\geq 1}$ and a lag value $L > 0$, whos dynamical behavior can be approximately identified by a linear autoregressive model of the form

$$x_{t+1} \approx c_0x_t + c_1x_{t-1} + c_2x_{t-2} + \cdots + c_Lx_{t-L+1}, t \geq L.$$

There is an orthogonal projector $P \in \mathbb{C}^{L\times L}$ such that $P\mathbf{x}(t) \approx \mathbf{x}(t)$ for each

$$\mathbf{x}(t) = [x_t \; x_{t+1} \; \cdots x_{t+L-1}]^\top$$

and each $t \geq 1$.

Proof. (Sketch.) One just needs to apply (Vides, 2021b, Theorem 3.6) and (Vides, 2021b, Theorem 4.3) to (5).

5. ALGORITHMS

As an applications of the results in sections §3 and §4 we can obtain some prototypical algorithms.

5.1 Autoencoder algorithm

We can adapt (Vides, 2021b, Algorithm 1) to obtain algorithm 1.

Algorithm 1: DFTSpSolver: Sparse DFT based linear regression solver algorithm

```
Data: $Y \in \mathbb{C}^{n\times 1}$, $\delta > 0$, $N \in \mathbb{Z}^+$, $\varepsilon > 0$
Result: $J, X = \mathbf{DFTSpSolver}(Y, \delta, N, \varepsilon)$
  1: Set $X_0 = \mathbf{fft}(Y)$;
  2: Set $K = 1$;
  3: Set error $= 1 + \delta$;
  4: Set $c = X_0$;
  5: Set $x_0 = c$;
  6: Set $\hat{c} = [\hat{c}_1 \cdots \hat{c}_n]^\top = \left[|\hat{e}_{1,n}^* c| \cdots |\hat{e}_{n,n}^* c|\right]^\top$;
  7: Compute permutation $\sigma : \{1,\ldots,n\} \to \{1,\ldots,n\}$
     such that: $\hat{c}_{\sigma(1)} \geq \hat{c}_{\sigma(2)} \geq \cdots \geq \hat{c}_{\sigma(n)}$;
  8: Set $N_0 = \max\left\{\sum_{j=1}^n H_\varepsilon\left(\hat{c}_{\sigma(j)}\right), 1\right\}$;
     while $K \leq N$ and error $> \delta$ do
        (1) Set $X = \mathbf{0}_{n,1}$;
            for $k = 1,\ldots,N_0$ do
               Set $x_{\sigma(k)} = \hat{e}_{k,N_0}^* c$;
            end
        (2) Set error $= \|\mathbf{ifft}(X - x_0)\|_\infty$;
        (3) Set $x_0 = X$;
        (4) Set $\hat{c} = [\hat{c}_1 \cdots \hat{c}_n]^\top = \left[|\hat{e}_{1,n}^* X| \cdots |\hat{e}_{n,n}^* X|\right]^\top$;
        (5) Compute permutation
            $\sigma : \{1,\ldots,n\} \to \{1,\ldots,n\}$ such that:
            $\hat{c}_{\sigma(1)} \geq \hat{c}_{\sigma(2)} \geq \cdots \geq \hat{c}_{\sigma(n)}$;
        (6) Set $N_0 = \max\left\{\sum_{j=1}^n H_\varepsilon\left(\hat{c}_{\sigma(j)}\right), 1\right\}$;
        (7) Set $K = K + 1$;
     end
  9: Set $J = \{\sigma(1), \sigma(2), \ldots, \sigma(N_0)\}$;
return $J, X$
```

The results in §3 can be translated into algorithm 2 that relies on algorithm 1.

5.2 Autoregressor algorithm

The results in §4 can be translated into algorithm 3 that relies on (Vides, 2021b, Algorithm 1).

Algorithm 2: SpDFTAutoencoder: DFT based k-Sparse autoencoder algorithm

```
Data: $X \in \mathbb{C}^{n\times 1}$, $\delta > 0$, $N \in \mathbb{Z}^+$, $\varepsilon > 0$
Result: $\hat{X} = \mathbf{SpDFTAutoencoder}(X, \delta, N, \varepsilon)$
  1: Compute $k$-sparse autoencoding $\mathbf{h} = \mathcal{T}(X)$ using
     algorithm 1;
  2: Set $\hat{X} = \mathbf{ifft}(\mathbf{h})$;
return $\hat{X}$
```

Algorithm 3: SpAutoregressor: Autoregressor algorithm

```
Data: $\Sigma_N = \{x_t\}_{t=1}^N \subset \mathbb{C}^{n\times 1}$, $\delta > 0$, $N \in \mathbb{Z}^+$, $\varepsilon > 0$
Result: $\mathbf{c}, P = \mathbf{SpAutoregressor}(X, \delta, N, \varepsilon)$
  1: Solve (4) applying (Vides, 2021b, Algorithm 1);
  2: Use the economy-sized SVD $\mathcal{H}(\Sigma_0)^\top = USV$
     computed as part of (Vides, 2021b, Algorithm 1) to
     compute $P = UU^*$;
return $\hat{X}$
```

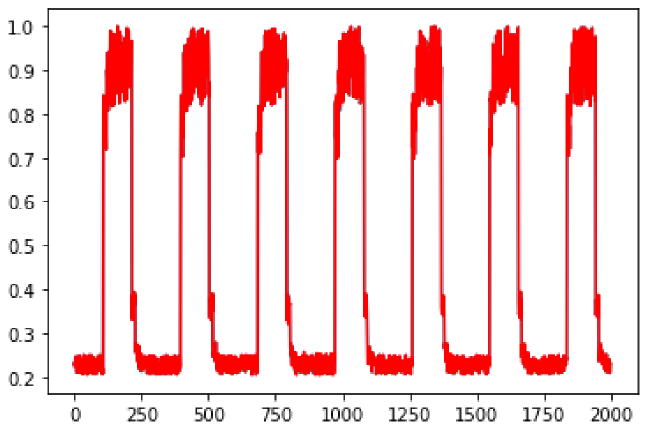


Fig. 1. Reference training segment of the signal for the DFT autoencoder.

6. NUMERICAL EXPERIMENTS

We will consider similar data to the one considered in Shipmon et al. (2017), that is available as part the the Numenta Anomaly Benchmark (NAB).

6.1 Unsupervised Signal Identification via DFT Based k-Sparse Autoencoders

Using the programs `DFTSpSolver.py`, `SpDFTEncoder.py` and `SpDFTDecoder.py` in Vides (2021a) that provide an implmentation of algorithms 1 and 2, we can study the signal recorded in the csv file `signal_with_anomaly.csv`. The reference and testing segments of the signal for the unsupervised training of the autoencoder model are shown in figures 4 and 5, respectively.

The corresponding autoencoder model computations produce the graphical outputs documented in figure 6.

The configuration required to replicate these results is available as part of the program `SpDFTAutoencoderDemo.py`.

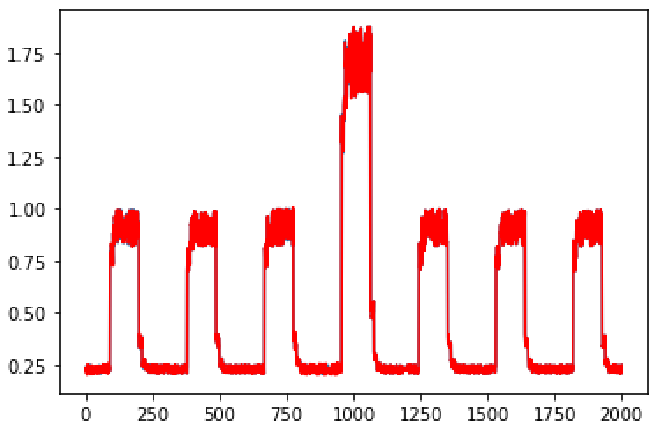


Fig. 2. Testing segment of the signal for anomaly detection via DFT autoencoder.

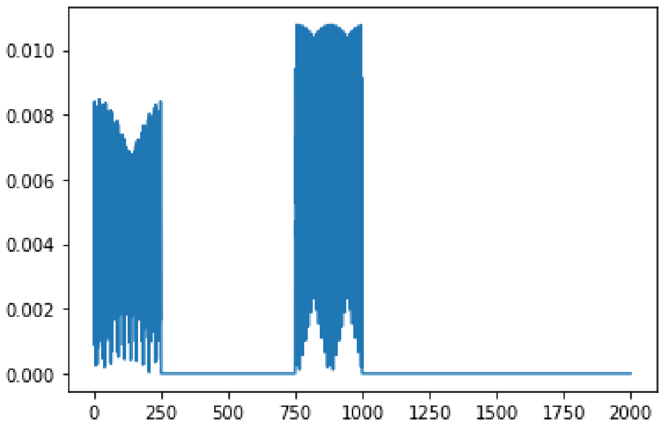


Fig. 3. Anomaly identification pattern generated by the trained DFT autoencoder.

6.2 Unsupervised Signal Identification via Sparse Autoregressors

Using the programs `lsspsolver.py`, `SpAutoRegressor.py` and `SPARPredictor.py` in Vides (2021a) that provide an implmentation of algorithms (Vides, 2021b, Algorithm 1) and 3, we can study the signal recorded in the csv file `signal_with_anomaly.csv`. The reference and testing segments of the signal for the unsupervised training of the autoencoder model are shown in figures 4 and 5, respectively.

The corresponding autoregressor model computations produce the graphical outputs documented in figure 6.

The configuration required to replicate these results is available as part of the program `SpAutoRegressor.py`.

7. CONCLUSION

The results observed in the numerical experiments are consitent with the theoretical elements documented in sections §3 and §4.

8. DATA AVAILABILITY

The programs and data that support the findings of this study are openly available in the SDSI repository, reference number Vides (2021a).

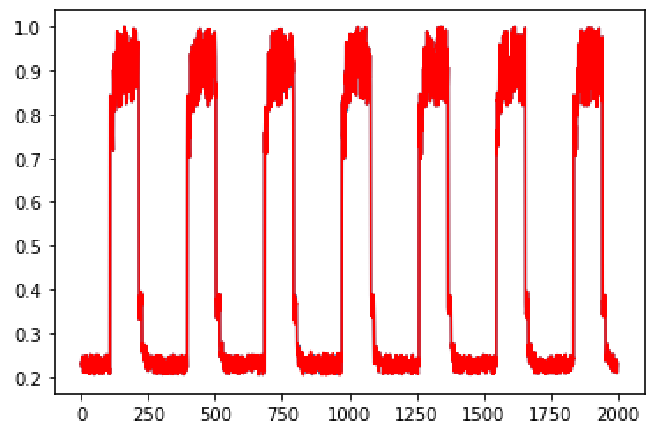


Fig. 4. Reference training segment of the signal for the sparse autoregressor.

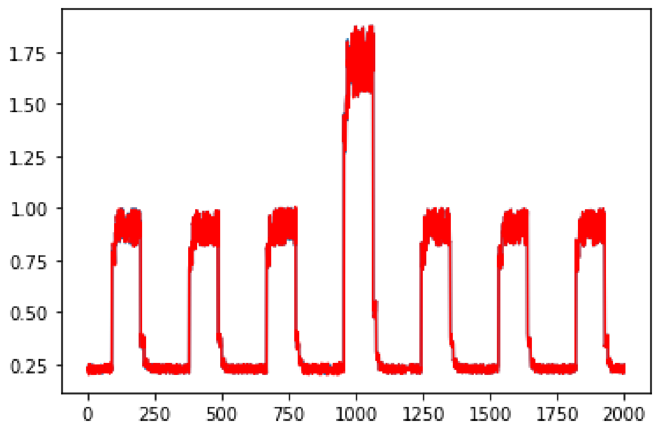


Fig. 5. Testing segment of the signal for anomaly detection via autoregressor.

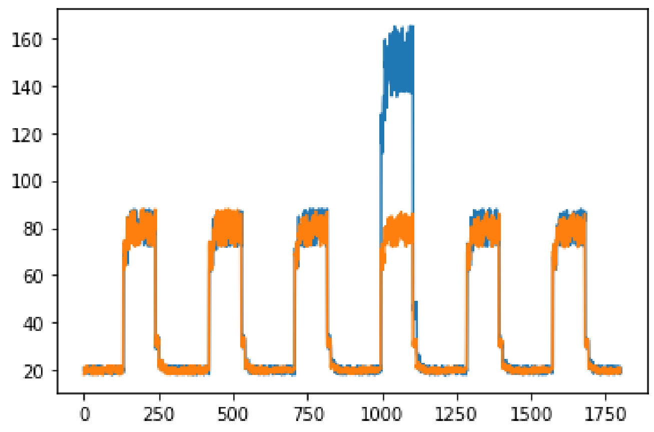


Fig. 6. Anomaly identification combined patterns generated by the trained autoregressor.

ACKNOWLEDGEMENTS

The structure preserving computations corresponding to the numerical experiments documented in this paper were performed using the computational resources of the Scientific Computing Innovation Center of UNAH, as part of the researh projecto PI-063-DICIHT. The time series data were provided by IME Inc. and by the Numenta Anomaly Benchmark (NAB) available available in the address `https://github.com/numenta/NAB`.

REFERENCES

Makhzani, A. and Frey, B.J. (2014). k-sparse autoencoders. In Y. Bengio and Y. LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. URL `http://arxiv.org/abs/1312.5663`.

Shipmon, D.T., Gurevitch, J.M., Piselli, P.M., and Edwards, S.T. (2017). Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data.

Vides, F. (2021a). Spaar: Sparse signal identification python toolset. URL `https://github.com/FredyVides/SPAAR`.

Vides, F. (2021b). Sparse system identification by low-rank approximation. *CoRR*, abs/2105.07522. URL `https://arxiv.org/abs/2105.07522`.