# Computing Semilinear Sparse Models for Approximately Eventually Periodic Signals

**Fredy Vides** [*]

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

**Abstract:** Some elements of the theory and algorithms corresponding to the computation of sparse semilinear models for discrete-time signals are presented. In this study, we will focus on approximately eventually periodic discrete-time signals, that is, signals that can exhibit an aperiodic behavior for an initial amount of time, and then behave approximately periodic afterwards. The semilinear models considered in this study are obtained by combining sparse representation methods, linear autoregressive models, and GRU neural network models, initially fitting each model independently using some reference data corresponding to some signal under consideration, and then fitting some of the resulting model coefficients again using the reference data previously considered, computing sparse representations of some of the matrix parameters of the resulting model along the process. Some prototypical computational implementations are presented as well.

*Keywords:* Autoregressive model, sparse representation, GRU neural network, Krylov subspace, time series analysis.

## 1. INTRODUCTION

In this document, Some elements of the theory and algorithms corresponding to the computation of sparse semilinear models for discrete-time signals are presented. The study reported in this document is focused on approximately eventually periodic discrete-time signals, that is, signals that can exhibit an aperiodic behavior for an initial amount of time, and then behave approximately periodic afterwards.

The main contribution of the work reported in this document is the application of a *colaborative scheme* involving sparse matrix approximation methods, linear autoregressive type models and GRU neural network models, where each model is first fitted independently using some reference data corresponding to some given signal, and then the resulting model is represented as a linear combination of the previously fitted models, whose mixing coefficients are fitted using the previously considered reference data. Along the process, some of the matrices of parameters of the resulting model are fitted using sparse representation techniques. Some theoretical aspects of the aforementioned process are described in §3. As a byproduct of the work reported in this document, a toolset of Python programs for semilinear sparse signal model computation based on the ideas presented in §3 and §4 has been developed, and is available in Vides (2021a).

The applications of the sparse model identification technology developed as part of the work reported in this document, range from numerical simulation for predictive maintenance of industrial equipement and structures, to geological data analysis. Specific applications in the aforementioned fields will be the subject of future communications.

A prototypical algorithm that applies the ideas presented in §3 for the computation of sparse semilinear autoregressors is presented in §4. Some illustrative computational implementations of the prototypical algorithm presented in §4 are presented in §5.

## 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 $\Sigma_N$, that is determined by the following expression.

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

We will write $I_n$ to denote de identity matrix in $\mathbb{C}^{n\times n}$, and 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.

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

Given any matrix $X \in \mathbb{C}^{m\times n}$, we will write $X^*$ to denote the conjugate transpose $\overline{X}^\top \in \mathbb{C}^{n\times m}$ of $X$. A matrix $P \in \mathbb{C}^{n\times n}$ will be called an orthogonal projector whenever $P^2 = P = P^*$. We will write $\mathbf{S}^1$ to denote the set $\{z \in \mathbb{C} : |z| = 1\}$.

Given a time series $\Sigma = \{x_t\}_{t\geq 1} \subset \mathbb{C}$, a positive integer $L$ and any $t \geq L$, we will write $\mathbf{x}_L(t)$ to denote the vector

$$\mathbf{x}_L(t) = [x_t \ x_{t-1} \ \cdots \ x_{t-L+1}]^\top \in \mathbb{C}^{nL}.$$

Given any matrix $A \in \mathbb{C}^{n\times n}$, we will write $\sigma(A)$ to denote the set of eigenvalues of $A$.

## 3. SEMILINEAR MODELING OF APPROXIMATELY EVENTUALLY PERIODIC SIGNALS

A discrete time signal represented by a times series $\Sigma = \{x_t\}_{t\geq 1}$ is said to be approximately eventually periodic (**AEP**) if it can be aperiodic for an initial amount of time, and then becomes approximately periodic afterwards. In other words, there are $\varepsilon > 0$ and two integers $T, S > 0$ such that $|x_{t+kT} - x_t| \leq \varepsilon$ for each $t \geq S$ and each integer $k \geq 0$. The integer $T$ will be called the approximate period of $\Sigma$.

Based on the notion of approximately eventually periodic signal considered on this study, it can be seen that given an AEP signal $\Sigma = \{x_t\}_{t\geq 1}$, there is a positive integer $S$ such that the tail $\{x_t\}_{t\geq S}$ of $\Sigma$ is approximately periodic.

### *3.1 Sparse Semilinear Autogregressors*

Given a time series $\Sigma = \{x_t\}_{t\geq 1} \subset \mathbb{C}$ and a lag value $L > 0$. Let us consider a semilinear signal model of the form:

$$x_{t+1} = \mathcal{L}(\boldsymbol{x}_L(t)) + \mathcal{G}(\boldsymbol{x}_L(t)) + \mathcal{E}(\boldsymbol{x}_L(t)), t \geq L. \quad (3)$$

where $\mathcal{L}$ denotes a linear operation determined by the expression

$$\mathcal{L}(\boldsymbol{x}_L(t)) = c_1 x_t + c_2 x_{t-1} + c_2 x_{t-2} + \cdots + c_L x_{t-L+1}, \quad (4)$$

$\mathcal{G}(\mathbf{x}_L(t))$ represents a linear combination of neural networs of the form (5), and $\mathcal{E}(\boldsymbol{x}_L(t))$ represents some suitable error term. In order to compute models of the form (3), we can combine sparse autoregressive models together with GRU RNN using TensorFlow, Keras and PyTorch whose structures are described as part of Chollet et al. (2015) and Paszke et al. (2019). The GRU based neural network structure is described by the following block diagram.

$$\begin{tikzcd} {} \arrow[r, "\mathbf{x}_L(t)"] & \boxed{\mathbf{GRU}} \arrow[r, "\mathbf{h}(t)"] & \boxed{\mathbf{L}} \arrow[r, "\hat{x}_{t+1}"] & {} \end{tikzcd} \quad (5)$$

For some given an integer $m > 0$, each GRU $j$-cell in the GRU block is described for each $j = 1, \ldots, m$ by the following equations:

$$\begin{aligned} r_j(t) &= \sigma\left(\hat{e}_{j,m}^\top \left(W_{ir}\mathbf{x}(t) + W_{hr}\mathbf{h}(t-1) + b_r\right)\right) \\ z_j(t) &= \sigma\left(\hat{e}_{j,m}^\top \left(W_{iz}\mathbf{x}(t) + W_{hz}\mathbf{h}(t-1) + b_z\right)\right) \\ n_j(t) &= \tanh(\hat{e}_{j,m}^\top \left(W_{in}\mathbf{x}(t) + b_n\right) \\ &\quad + r_j(t)\hat{e}_{j,m}^\top \left(W_{hn}\mathbf{h}(t-1)\right)) \\ h_j(t) &= (1 - z_j(t))n_j(t) + z_j(t)h_j(t-1) \end{aligned}$$

with $\mathbf{x}(t) = \mathbf{x}_L(t)$ and $\mathbf{h}(t) = [h_1(t) \ \cdots \ h_m(t)]^\top$. The GRU layer configuration considered in this document, has been chosen in order to prevent vanishing gradients, by taking advantage of the GRU structure introduced by Cho et al. (2014).

The linear layer $\mathbf{L}$ of the neural network $\mathcal{L}$ described in (5) is determined by the expression

$$\mathbf{L}(\mathbf{h}(t)) = \mathbf{w}_L^\top \mathbf{h}(t) + b_L.$$

An approximate representation

$$\tilde{\mathcal{L}}(\mathbf{x}_L(t)) = \tilde{c}_1 x_t + \tilde{c}_2 x_{t-1} + \tilde{c}_2 x_{t-2} + \cdots + \tilde{c}_L x_{t-L+1},$$

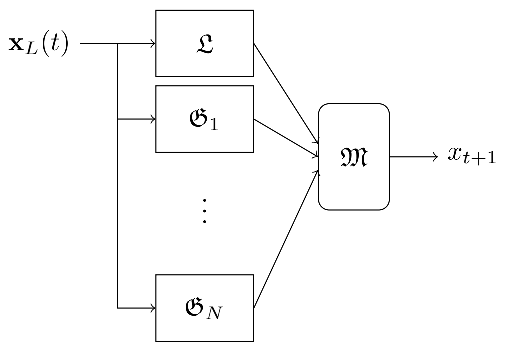
of the linear part of (3) such that

$$x_{t+1} \approx \hat{\mathcal{L}}(\mathbf{x}_L(t)), t \geq L$$

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

$$\mathcal{H}_L(\Sigma_0)^\top \begin{bmatrix} c_L \\ c_{L-1} \\ \vdots \\ c_2 \\ c_1 \end{bmatrix} = \begin{bmatrix} x_{N-L+1} \\ x_{N-L+2} \\ \vdots \\ x_{N-1} \\ x_N \end{bmatrix}. \quad (6)$$

Schematically the autoregressors considered in this study can be described by a block diagram of the form



(7)

where the block $\mathfrak{L}$ is represented by (4), each block $\mathfrak{G}_j$ is represented by (5), and where the block $\mathfrak{M}$ is determined by the expression

$$\mathfrak{M}(y_1(t), \ldots, y_{N+1}(t)) = \sum_{j=1}^{N+1} w_j y_j(t),$$

for some coefficients $w_j$ to be determined and some given $N$, with $y_1(t) = \mathfrak{L}(\mathbf{x}_L(t))$ and $y_{k+1}(t) = \mathfrak{G}_k(\mathbf{x}_L(t))$ for each $k = 1, \ldots, N$ and each $t \geq L$.

The details of the computation of neural network blocks of the form (3) will be omitted for brevity, for details on the theory and computation of GRU models the reader is kindly referred to Cho et al. (2014), Chollet et al. (2015), Paszke et al. (2019) and Vides (2021a).

Several interesting papers have been written on the subject of hybrid time series models that combine ARIMA and

ANN, two important references on this matter are Zhang (2003) and Khandelwal et al. (2015). An important aspect of the modeling approach reported in this document, is that instead of using the GRU RNN components of (7) represented by $\mathcal{G}$ in (3) to approximate the residual $r_t = x_{t+1} - \mathcal{L}(\mathbf{x}_L(t))$, using some training subsets $\Sigma_I, \Sigma_M$ of a given data sample $\Sigma_N$ from an arbitrary AEP signal $\Sigma = \{x_t\}_{t\geq 1}$ under consideration, first the coefficients of the blocks $\mathfrak{L}$, $\mathfrak{G}_1, \cdots, \mathfrak{G}_N$ of (7) are fitted using $\Sigma_I$, and then the coefficients of the block $\mathfrak{M}$ of (7) are fitted using $\Sigma_M$ and some corresponding predicted values generated with $\mathfrak{L}$, $\mathfrak{G}_1, \cdots, \mathfrak{G}_N$.

### *3.2 An Operator Theoretic Approach to Autoregressors*

As a consequence of (6), if we consider any sample $\Sigma_M = \{\tilde{x}_t\}_{t=1}^{M} \subset \Sigma$ of suitable size $M = N-1 > L$, such that the states in $\Sigma_M$ are sucessors of the states in $\Sigma_0 = \{x_t\}_{t=1}^{N-1}$, in the sense that there is some nonnegative integer $S$ such that $\tilde{x}_t = x_{t+S}$, for each $t = 1, \ldots, M$. We will have that the matrix

$$C(L) = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & 0 & 1 \\ c_L & c_{L-1} & \cdots & \cdots & c_2 & c_1 \end{bmatrix} \qquad (8)$$

will satisfy the condition

$$\mathcal{H}_L(\Sigma_0)^\top \left(C(L)^S\right)^\top = \mathcal{H}_L(\Sigma_M)^\top. \qquad (9)$$

Using matrices of the form (8) one can express linear models of the form (4) as follows.

$$\mathcal{L}(\mathbf{x}_L(t)) = \hat{e}_{L,L}^\top C(L)\mathbf{x}_L(t) \qquad (10)$$

One can observe that to each model of the form (4), there corresponds a matrix of the form (8). From here on, a matrix that satisfies the previous conditions will be called the matrix form of a linear model of the form (4).

Given $\delta > 0$, and two matrices $A \in \mathbb{C}^{m\times n}$ and $Y \in \mathbb{C}^{m\times p}$, we will write $AX \approx_\delta Y$ to represent the problem of finding $X \in \mathbb{C}^{n\times p}$, $\alpha, \beta \geq 0$ and an orthogonal projector $Q$ such that $\|AX - Y\|_F \leq \alpha\delta + \beta\|(I_m - Q)Y\|_F$. The matrix $X$ will be called a solution to the problem $AX \approx_\delta Y$.

As a consequence of (9), (Vides, 2021b, Theorem 3.6) and (Vides, 2021b, Theorem 4.3) we can obtain the following result.

*Theorem 1.* Given $\delta > 0$, two integers $L, M > 0$, a sample $\Sigma_N = \{x_t\}_{t=1}^{N}$ from an approximately eventually periodic signal $\Sigma = \{x_t\}_{t\geq 1}$ and a matrix $A \in \mathbb{C}^{M\times L}$. If $r = \mathrm{rk}_\delta(\mathcal{H}_L(\Sigma_N)) > 0$, then there is a sparse matrix $\hat{A} \in \mathbb{C}^{M\times L}$ with at most $Mr$ entries such that $A\mathcal{H}_L(\Sigma_N) \approx_\delta \hat{A}\mathcal{H}_L(\Sigma_N)$.

**Proof.** This result is a consequence of the application of (Vides, 2021b, Theorem 3.6) and (Vides, 2021b, Theorem 4.3) to the matrix equation

$$\mathcal{H}_L(\Sigma_N)^\top \hat{A}^\top = \mathcal{H}_L(\Sigma_N)^\top A^\top.$$

*Remark 2.* Given some AEP signal under consideration $\Sigma = \{x_t\}_{t\geq 1}$ with approximate period $T$, if the corresponding resituals $r_t = |x_{t+1} - \mathcal{L}(\mathbf{x}_L(t))$ are small, then the significative contribution of the linear component $\mathcal{L}$ of (3) to the modeling process of $\Sigma$ would be beneficial for interpretability purposes. Also, if we consider the tail $\tilde{\Sigma} = \{x_t\}_{t\geq S}$ of $\Sigma$, by applying a Krylov space approach along the lines presented in (Saad, 2011, §6.1), and as a consequence of (Vides, 2021b, Theorem 4.3.), one would expect that there are $\varepsilon > 0$ and some matrix $W_k \in \mathcal{C}^{L\times k}$ whose columns form an orthonormal basis of $\mathrm{span}\left(\{x_S, C(L)x_S, C(L)^2x_S, \ldots, C(L)^{T-1}x_S\}\right)$, such that each $z \in \sigma(W_k^* C(L) W_k)$ satisfies the relation $|z^T - 1| \leq \varepsilon$. This interesting *mimetic features* will be furter explored in future communications.

The matrix $W_k^* C(L) W_k$ will be called the $\tilde{\Sigma}$-section of $C(L)$ and will be denoted by $C(L)|_\Sigma$.

*Remark 3.* When a given AEP signal $\Sigma = \{x_t\}_{t\geq 1}$ is well explained by the linear component of a semilinear model, that is, the corresponding residual is relatively small, one would expect that the matrix $C(L)|_\Sigma$ corresponding to the model would *mimic* the approximate periodicity of the tail $\{x_t\}_{t\geq S}$ of $\Sigma$, in the sense that the number $\|(C(L)|_\Sigma)^T - I_n\|$ is relatively small for some suitable matrix norm $\|\cdot\|$ (in the sense of (Saad, 2011, §1.5)). Ideally, when plotting $\sigma((C(L)|_\Sigma)^T)$ one should observe the elements of $\sigma((C(L)|_\Sigma)^T)$ clustering around 1.

## 4. ALGORITHMS

As an applications of the results in section §3.1 we can obtain a prototypical algorithm.

### *4.1 Autoregressor Algorithm*

The results in §3.1 can be translated into algorithm 1 that relies on (Vides, 2021b, Algorithm 1) and Theorem 1.

**Algorithm 1: SLSpARModel**: Sparse Semilinear 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: $\mathbf{c}, C(L) = \mathbf{SpAutoregressor}(X, \delta, N, \varepsilon)$
  1: Solve (6) applying (Vides, 2021b, Algorithm 1);
  2: Compute $C(L)$ using the elements of the vector $\mathbf{c}$
     computed in step [1] according to (8);
  3: Fit the blocks $\mathfrak{G}_j$ of (7) using data in $\Sigma_N$.
  4: For the GRU layers of each $\mathfrak{G}_j$, compute sparse
     representations of the corresponding input weights
     when appropriate, applying Theorem 1.
  5 : Compute the coefficients $\mathbf{w}_M = (w_1, \ldots, w_{N+1})$ of
     the block $\mathfrak{M}$ of (7) using $\Sigma_N$ and (Vides, 2021b,
     Algorithm 1);
return $\boldsymbol{c}, C(L), \mathfrak{G}_1, \ldots, \mathfrak{G}_j, \mathbf{w}_M$
```

We can apply algorithm 1 to compute the model coefficients needed for the computation of signal models of the form (7).

## 5. NUMERICAL EXPERIMENTS

In this section, some computational implementations of the results and algorithms reported in this document are presented. Most of the numerical experiments documented in this section where performed on a Linux Ubuntu Server

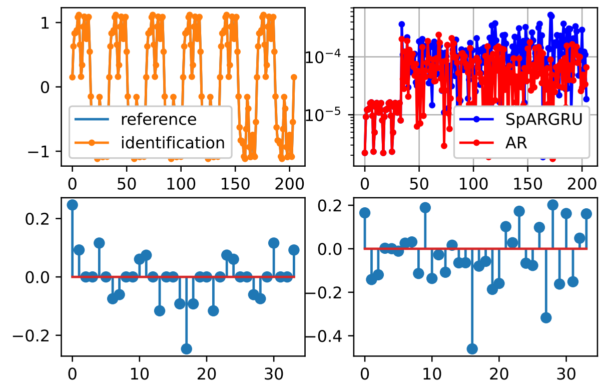


Fig. 1. Reference and approximation signals (top left). Approximation errors (top right). Linear component coefficients of the SpARGRU model with 18 $nz$ (bottom left). Coefficients of the linear component of standard AR model with 34 $nz$ (bottom right).

20.04 workstation equiped with an Intel Core i7 processor with 6 cores and with 8GB RAM. Some numerical experiments where also performed in Google Colab and in IBM Quantum Lab.

The experimental results documented in this section can be replicated using the function `NumericalExperiment.py` or the Jupyter notebook `SLSpAARModelsDemo.ipynb`, that are available in Vides (2021a). The configuration required to replicate the results in this section is available as part of the aforementioned programs.

Since the models considered in this study consit of linear combinations of sparse autoregressive models with GRU RNN based models, we will refer to models of this type as **SpARGRU** models. The signal approximations computed using the SpARGRU models presented in this document are compared with the approximations obtained using standard AR models. The corresponding standard AR models are computed using the python program `Autoreg` included as part of `statsmodels` module. In this section we will write $nz$ to denote nonzero elements.

For the experiments documented in this section two GRU RNN blocks were used, the block $\mathfrak{G}_1$ was computed using TensorFlow 2.6.0 and its input weights were replaced by sparse representations computed using (Vides, 2021b, Algorithm 1), and the block $\mathfrak{G}_2$ was computed using PyTorch 1.9.1+cpu and its input weights were left unchanged.

### *5.1 Numerical Experiment 1*

In this section the programs in Vides (2021a) based on algorithms (Vides, 2021b, Algorithm 1) and 1, can be used to compute a sparse model for the signal recorded in the csv file `AlmostPeriodicSignal.csv` in the DataSets folder in Vides (2021a). The graphic representations of the resuls produced by the command sequence `NumericalExperiment(1)` are shown in figures 1 and 2, respectively.

In every figure of the type of figure 2, the red dots represent the points in each considered spectrum, the blue line

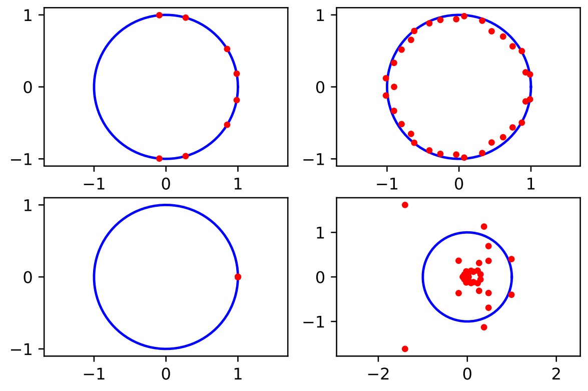


Fig. 2. $\sigma(C(L)|_\Sigma)$ for the linear component of the SpARGRU model (top left). $\sigma(C(L)|_\Sigma)$ for the linear component of the standard AR model (top right). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the SpARGRU model (bottom left). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the standard AR model (bottom right).

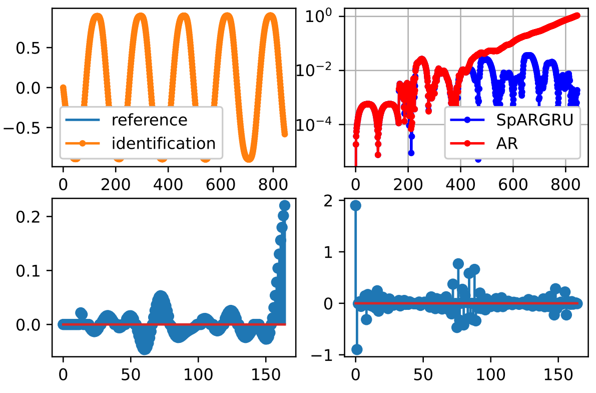


Fig. 3. Reference and approximation signals (top left). Approximation errors (top right). Linear component coefficients of the SpARGRU model with 118 $nz$ (bottom left). Coefficients of the linear component of standard AR model with 165 $nz$ (bottom right).

represents $\mathbf{S}^1$, and the number $T$ represents the estimated approximate period for each signal considered.

### *5.2 Numerical Experiment 2*

In this section the programs in Vides (2021a) based on algorithms (Vides, 2021b, Algorithm 1) and 1, can be used to compute a sparse model for the signal recorded in the csv file `NLOscillatorSignal.csv` in the DataSets folder in Vides (2021a). The graphic representations of the resuls produced by the command sequence `NumericalExperiment(2)` are shown in figures 3 and 4, respectively.

### *5.3 Numerical Experiment 3*

In this section the programs in Vides (2021a) based on algorithms (Vides, 2021b, Algorithm 1) and 1, can

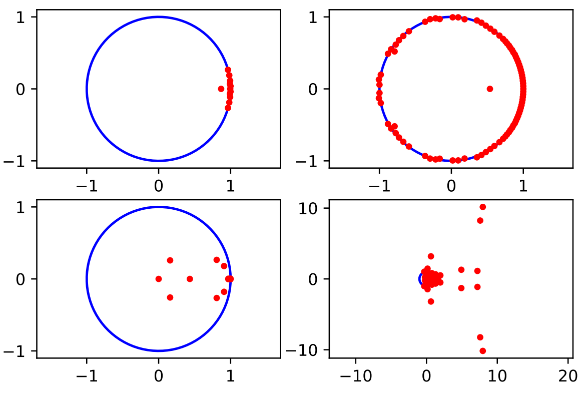


Fig. 4. $\sigma(C(L)|_\Sigma)$ for the linear component of the SpARGRU model (top left). $\sigma(C(L)|_\Sigma)$ for the linear component of the standard AR model (top right). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the SpARGRU model (bottom left). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the standard AR model (bottom right).

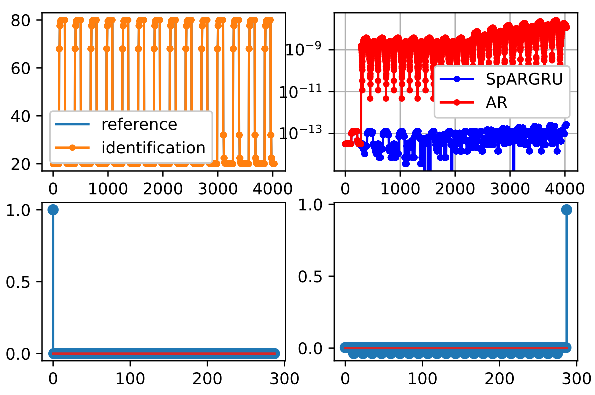


Fig. 5. Reference and approximation signals (top left). Approximation errors (top right). Linear component coefficients of the SpARGRU model with 8 $nz$ (bottom left). Coefficients of the linear component of standard AR model with 288 $nz$ (bottom right).

be used to compute a sparse model for the signals recorded in the csv file `art_daily_no_noise.csv` and `art_daily_small_noise.csv` included as part of Ahmad et al. (2017). The graphic representations of the resuls produced by the command sequence `NumericalExperiment(3)` for the periodic signal without noise are shown in figures 5 and 6, respectively. The graphic representations of the resuls produced by the command sequence `NumericalExperiment(4)` for the periodic signal with noise are shown in figures 7 and 8.

### *5.4 Approximation Errors*

The approximation root mean square errors are sumarized in table 1.

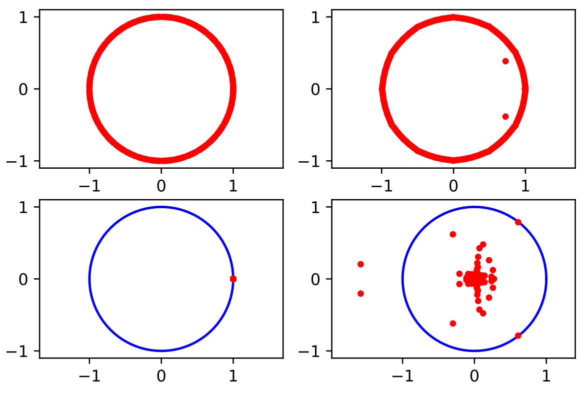


Fig. 6. $\sigma(C(L)|_\Sigma)$ for the linear component of the SpARGRU model (top left). $\sigma(C(L)|_\Sigma)$ for the linear component of the standard AR model (top right). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the SpARGRU model (bottom left). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the standard AR model (bottom right).

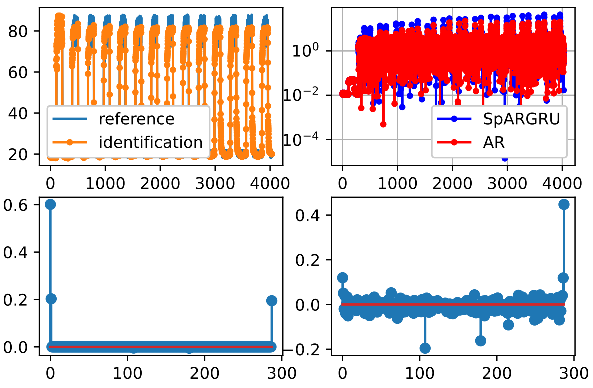


Fig. 7. Reference and approximation signals (top left). Approximation errors (top right). Linear component coefficients of the SpARGRU model with 5 $nz$ (bottom left). Coefficients of the linear component of standard AR model with 288 $nz$ (bottom right).

## 6. CONCLUSION

The results observed in the numerical experiments are consistent with the theoretical elements documented in section §3. In particular, although in some experiments in §5 the corresponding RMSE for the AR and SpARGRU models are similar, the mimetic behavior described in remarks 2 and 3 for the corresponding $\hat{\Sigma}$-sections, can be visualized in figures 2, 4, 6, 8. This mimetic behavior is interesting not just from a theoretica point of view,

Table 1. RMSE

| Model | SpARGRU Model | AR Model |
|---|---|---|
| Experiment 1 | 0.000160358 | 0.0001450925 |
| Experiment 2 | 0.0129278690 | 0.3100591516 |
| Experiment 3.1 | 0.0000000000 | 0.0000000074 |
| Experiment 3.2 | 4.2878248107 | 4.0939437825 |

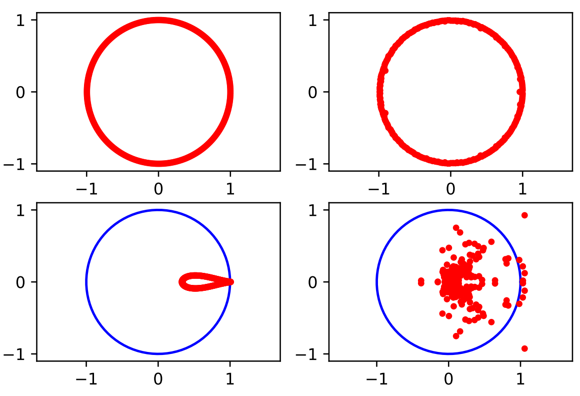


Fig. 8. $\sigma(C(L)|_\Sigma)$ for the linear component of the SpAR-GRU model (top left). $\sigma(C(L)|_\Sigma)$ for the linear component of the standard AR model (top right). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the SpAR-GRU model (bottom left). $\sigma((C(L)|_\Sigma)^T)$ for the linear component of the standard AR model (bottom right).

but also for practical computational reasons, as long term predictions or simulations can be affected when the eigenvalues of the corresponding $\tilde{\Sigma}$-section of the matrix form corresponding to a linear model, lie outside the unit disk.

## 7. DATA AVAILABILITY

The Python programs that support the findings of this study are openly available in the SPAAR repository, reference number Vides (2021a). The time series data used for the experiments 1 and 2 documented in §5 are available as part of Vides (2021a), and the time series data used for experiment 3 in §5 are available as part of the Numenta Anomaly Benchmark (NAB) described in Ahmad et al. (2017).

## 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 research project PI-063-DICIHT. Some experiments were performed on Google Colab and IBM Quantum Lab as well.

## REFERENCES

Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262, 134–147. doi: https://doi.org/10.1016/j.neucom.2017.04.070. Online Real-Time Learning Strategies for Data Streams.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734. Association for Computational Linguistics, Doha, Qatar. doi:10.3115/v1/D14-1179. URL `https://aclanthology.org/D14-1179`.

Chollet, F. et al. (2015). Keras. `https://keras.io`.

Khandelwal, I., Adhikari, R., and Verma, G. (2015). Time series forecasting using hybrid arima and ann models based on dwt decomposition. *Procedia Computer Science*, 48, 173–179. doi: https://doi.org/10.1016/j.procs.2015.04.167. International Conference on Computer, Communication and Convergence (ICCC 2015).

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Saad, Y. (2011). *Numerical Methods for Large Eigenvalue Problems*. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611970739.

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`.

Zhang, G. (2003). Time series forecasting using a hybrid arima and neural network model. *Neurocomputing*, 50, 159–175. doi:https://doi.org/10.1016/S0925-2312(01)00702-0.