BMC Bioinformatics

CrossMark

Coding Prony's method in MATLAB and applying it to biomedical signal filtering

A. Fernández Rodríguez, L. de Santiago Rodrigo, E. López Guillén, J. M. Rodríguez Ascariz,
J. M. Miguel Jiménez and Luciano Boquete* [ID]

Abstract

Background: The response of many biomedical systems can be modelled using a linear combination of damped exponential functions. The approximation parameters, based on equally spaced samples, can be obtained using Prony's method and its variants (e.g. the matrix pencil method). This paper provides a tutorial on the main polynomial Prony and matrix pencil methods and their implementation in MATLAB and analyses how they perform with synthetic and multifocal visual-evoked potential (mfVEP) signals.
This paper briefly describes the theoretical basis of four polynomial Prony approximation methods: classic, least squares (LS), total least squares (TLS) and matrix pencil method (MPM). In each of these cases, implementation uses general MATLAB functions. The features of the various options are tested by approximating a set of synthetic mathematical functions and evaluating filtering performance in the Prony domain when applied to mfVEP signals to improve diagnosis of patients with multiple sclerosis (MS).

Results: The code implemented does not achieve 100%-correct signal approximation and, of the methods tested, LS and MPM perform best. When filtering mfVEP records in the Prony domain, the value of the area under the receiver-operating-characteristic (ROC) curve is 0.7055 compared with 0.6538 obtained with the usual filtering method used for this type of signal (discrete Fourier transform low-pass filter with a cut-off frequency of 35 Hz).

Conclusions: This paper reviews Prony's method in relation to signal filtering and approximation, provides the MATLAB code needed to implement the classic, LS, TLS and MPM methods, and tests their performance in biomedical signal filtering and function approximation. It emphasizes the importance of improving the computational methods used to implement the various methods described above.

Keywords: Prony's method, Matrix pencil, Least squares, Total least squares, Multifocal evoked visual potentials, Multiple sclerosis

Background

Prony's method

In 1795, Gaspard de Prony [1] proposed a method to explain the expansion of gases as a linear sum of damped complex exponentials of signals that are uniformly sampled. Prony's method approximates a sequence of $N = 2p$ equally spaced samples to a linear combination of p complex exponential functions with differing amplitudes, damping factors, frequencies and phase angles. The main contribution of this classic method is that it converts a non-linear approximation of exponential sums by solving a set of linear equations and a root-finding problem.

The conventional or polynomial Prony method consists of setting out an autoregressive model of order p that assumes that the value of sampled data x[n] depends linearly on the preceding p values in x. Solving this linear system of equations obtains the coefficients of the characteristic or Prony polynomial $\phi(z)$. The roots of this polynomial yield two of the parameters of the solution (damping factors and frequency) and provide a second system of

* Correspondence: luciano.boquete@uah.es
Grupo de Ingeniería Biomédica, Departamento de Electrónica, Universidad de Alcalá, Plaza de S. Diego, s/n, 28801 Alcalá de Henares, Spain

equations to calculate the amplitude and phase of the p functions.

Prony's original method exactly matched the curve of p exponential terms to a dataset of $N = 2p$ elements. When $N > 2p$, the linear systems of equations are overdetermined and can be approximated by the least squares (LS) method [2]. The conventional least-squares method considers that in the linear system ($\mathbf{A.x} \approx \mathbf{b}$), only $\mathbf{b}$ (observation vector) is contaminated by noise, while $\mathbf{A}$ (coefficient matrix) is noise-free. However, generally both matrix $\mathbf{A}$ and vector $\mathbf{b}$ are noise-perturbed (in Prony's method, $\mathbf{A}$ and $\mathbf{b}$ share the same data source, see below) and, in this case, the total least-squares technique (TLS) [3] can be more advantageous.

In some cases, a problem with the Prony polynomial method is that it can be numerically unstable because of the steps that comprise the algorithm: solving an ill-conditioned matrix equation and finding the roots of a polynomial. When the number of exponentials is relatively high, the sensitivity of roots of the characteristic polynomial to perturbations of their coefficient is likewise high [4] and Prony's method may be unstable.

Another alternative is to use the **matrix pencil method** (MPM). Although similar to Prony's method, it consists of solving an eigenvalue problem rather than following the conventional two-step Prony method. It has been found through perturbation analysis and simulation that for signals with unknown damping factors the MPM is less sensitive to noise than the polynomial method [5].

In recent years, and due to advances in computing systems, Prony's method has been successfully applied in various engineering sectors, such as electric power quality analysis [6], materials science [7], antennae [8], etc. In the biomedical field, the classic Prony method is used in [9] to process multifocal visual-evoked potentials (mfVEPs) to diagnose the early stages of multiple sclerosis (MS). The LS Prony method is used in [10] to estimate the parameters of the single event-related potential; the TLS is used in [11] to discriminate between three cardiac problems, and the MPM is used in [12–14].

Various programming languages are widely used in the scientific field. These languages include Python, a free and open-source high-level programming language [15, 16], and MATLAB®, a proprietary product.

MATLAB® is user-friendly and needs practically no formal programming knowledge [17]. As it implements a wide number and variety of functions (statistics, neural networks, graphics, wavelets, etc.), it is widely accepted as a development platform for numerical software by a significant portion of the computational science and engineering community [18–20]. Its open availability ensures reproducibility and knowledge exchange.

Objectives

This paper presents a tutorial on implementation in MATLAB of two families of Prony methods: the **polynomial method** (classic and extended — LS and TLS) and the **matrix pencil method**. It presents an overview of the mathematical bases of each method and implements them in MATLAB using the functions directly available. The results produced by the different methods when approximating synthetic signals are obtained and filtering of mfVEP records is implemented in the Prony domain. The Discussion section provides information on possible ways of mitigating the ill-conditioning problems associated with several of the resolution phases of the Prony methods.

Implementation
Polynomial method

A data sequence $x[n]$ ($n = 1,...N$) can be represented by the sum of p complex parameters (order p) according to the following expression:

$$
\begin{aligned}
x[n] &= \sum_{k=1}^{p} A_k e^{j\theta_k} \cdot e^{\left(\alpha_k + j2\pi f_k\right)T_s(n-1)} \\
&= \sum_{k=1}^{p} h_k \cdot z_k^{(n-1)}
\end{aligned}
\tag{1}
$$

Approximation of signal $x[n]$ occurs in p components, in which A_k is the initial amplitude in the same units as $x[n]$, α_k is the damping factor in seconds^{-1}, f_k is the frequency in Hertz, T_S is the sampling period (in seconds) of signal $x[n]$ and θ_k is the initial phase in radians. Therefore, signal $x[n]$ is characterized by the parameters A_k, α_k, f_k and θ_k (k = 1,...,p). h_k is the time-independent component and z_k is an exponential and time-dependent component (poles).

Equation 1 is the expression of the general solution of a homogeneous linear difference equation, if the p roots are different [21]. In order to find that equation we have to construct its characteristic equation, which is

$$
\phi(z) = \prod_{k=1}^{p} (z - z_k) = \sum_{k=0}^{p} a[k] z^{p-k}; \qquad a[0] = 1
\tag{2}
$$

where the characteristic roots are the parameters z_k in Eq. 1.

Demonstration of the Prony approximation method is found in [22]. Practical implementation requires performance of the following steps:

Step 1: Solve the linear prediction model constructed by the observed dataset and the obtained coefficients (a [1]...a[p]) of the characteristic polynomial. An autoregressive model of order p considers that the value of x[n] depends linearly on the preceding p values in x. Equation 1 can be rewritten as a linear prediction model according to the matrix system $\mathbf{T}_{pxp}.\mathbf{a}_{px1} = -\mathbf{x}_{px1}$:

$$\begin{pmatrix} x[p] & x[p-1] & \cdots & x[1] \\ x[p+1] & x[p] & \cdots & x[2] \\ \vdots & \vdots & \ddots & \vdots \\ x[2p-1] & x[2p-2] & \cdots & x[p] \end{pmatrix} \begin{pmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{pmatrix}$$
$$= - \begin{pmatrix} x[p+1] \\ x[p+2] \\ \vdots \\ x[2p] \end{pmatrix} \tag{3}$$

Where.

a: Linear prediction coefficients vector.

x: Observation vector.

T: Forward linear prediction matrix (a square Toeplitz matrix).

Solving this linear system (3) reveals that the values of **a** are the coefficients of the characteristic or Prony polynomial $\phi(z)$.

Step 2: Find the roots of the characteristic or Prony polynomial formed from the linear prediction coefficients.

Solving an equation in finite differences is achieved by finding the roots of the characteristic polynomial. As vector **a** is known from (3), the roots z_k of the polynomial $\phi(z)$ can be computed to obtain the damping factor (α_k) and frequency (f_k).

$$\alpha_k = \frac{\ln|z_k|}{T_s} \tag{4}$$

$$f_k = \frac{\tan^{-1}\left[\dfrac{Im(z_k)}{Re(z_k)}\right]}{2\pi T_s} \tag{5}$$

Step 3: Solve the original set of linear equations to yield the estimates of the exponential amplitude and sinusoidal phase.

First, the initial system of equations ($\mathbf{Z}_{pxp}.\mathbf{h}_{px1} = \mathbf{x}_{px1}$) is solved:

$$\begin{pmatrix} z_1^0 & z_2^0 & \cdots & z_p^0 \\ z_1^1 & z_2^1 & \cdots & z_p^1 \\ \vdots & \vdots & \ddots & \vdots \\ z_1^{p-1} & z_2^{p-1} & \cdots & z_p^{p-1} \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ \vdots \\ h_P \end{pmatrix} = \begin{pmatrix} x[1] \\ x[2] \\ \vdots \\ x[p] \end{pmatrix} \tag{6}$$

The $\mathbf{h_k}$ values yield the amplitude (A_k) and phase (θ_k):

$$A_k = |h_k| \tag{7}$$

$$\theta_k = \tan^{-1}\left[\frac{Im(h_k)}{Re(h_k)}\right] \tag{8}$$

The classic Prony method (N = 2p) obtains an exact fit between the sampled data points and the exponentials if matrices **T** and **Z** are non-singular. However, in many practical cases N > 2p and, in this situation, both systems are overdetermined (more equations than unknowns) and can be approximated using the LS or TLS methods.

Least squares

In general, given the overdetermined linear system: $\mathbf{A}\,\mathbf{x} \approx \mathbf{b}$ with $\mathbf{A} \in \mathbb{C}^{mxn}$, $\mathbf{b} \in \mathbb{C}^{mx1}$, $\mathbf{x} \in \mathbb{C}^{nx1}$, m > n; being **A** the data matrix and **b** the observation vector, the least squares solution $\mathbf{x}_{LS}$ is given by the normal equation:

$$\mathbf{x}_{LS} = \left(\mathbf{A}^H\mathbf{A}\right)^{-1}\mathbf{A}^H\,\mathbf{b} = \mathbf{A}^+\,\mathbf{b} \tag{9}$$

H represents the Hermitian conjugate of a matrix and $\mathbf{A}^+$ is the Moore–Penrose pseudoinverse matrix of **A**. In practice, the normal equation is rarely used, as methods based on QR decomposition or singular value decomposition (SVD), among others, are preferable.

Total least squares

Solution of the system $\mathbf{A}\,\mathbf{x} \approx \mathbf{b}$ by the total least-squares method is a generalization of the LS approximation method when the data matrix **A** and observation vector **b** are contaminated with noise. In Prony's method, eqs. 3 and 6 are constructed from the measured signals. The basic total least-squares algorithm is [3]:

$\mathbf{C} := [\mathbf{A}:\mathbf{b}]$, matrix **A** augmented (expansion by columns) by vector **b** ($\mathbf{C} \in \mathbb{C}^{mx(n+1)}$). SVD of **C** matrix is then performed:

$$\mathbf{C} = \mathbf{U\Sigma V^H} \tag{10}$$

The matrices $\mathbf{U}_{m \times m}$ (left singular vector matrix) and $\mathbf{V}_{(n+1)\times(n+1)}$ (right singular vector matrix) are orthonormal ($\mathbf{U^H U} = \mathbf{U U^H} = \mathbf{I}_m$, $\mathbf{V^H V} = \mathbf{V V^H} = \mathbf{I}_{n+1}$) and $\mathbf{\Sigma}_{m \times (n+1)} = \mathrm{diag}(\sigma_1, \sigma_2, ...\sigma_{\min\ \{m,\ n+1\}}))$ being $\sigma_1 \geq \sigma_2 ... \geq \sigma_{\min\ \{m,\ n+1\}}$ the singular values of **C**.

The structure of **V** is as follows:

$$\mathbf{V} = \begin{bmatrix} v_{1,1} & \cdots & v_{1,(n+1)} \\ \vdots & \ddots & \vdots \\ v_{(n+1),1} & \cdots & v_{(n+1),(n+1)} \end{bmatrix} \quad (11)$$

The TLS solution exists if $v_{(n+1),\,(n+1)} \neq 0$ [23] and, moreover it is unique if $\sigma_n \neq \sigma_{n+1}$:

$$\mathbf{x}_{TLS} = -\frac{1}{v_{(n+1),(n+1)}} \left[v_{1,(n+1)}, v_{2,(n+1)} \cdots v_{n,(n+1)} \right]^T \quad (12)$$

Algorithms in which the solution does not exist or is not unique are considered in detail in [24].

Implementation in MATLAB of the polynomial method

The code presented was developed and tested under MATLAB R2016b. **Code 1** presents implementation in MATLAB of a function to perform the Prony approximation using the three polynomial methods mentioned above. The function is defined as follows:

function   [Amp,alpha,freq,theta] = polynomial_method (x,p,Ts,method)

The sampled data are given in vector x; p is the number of terms to obtain in the approximation, Ts is the sampling time of the signal and classic, LS or TLS indicates the method used to solve the problem. The function returns the parameters (Amp, alpha, freq, theta) resulting from the approximation.

First, the sample length is obtained (**N = length(x)**) and consistency between the parameter method, p and the sample data length is checked.

Step 1.

Coding the linear system of Eq. 2 takes into account that the MATLAB function $T = toeplitz(c,r)$ creates non-symmetrical Toeplitz matrix **T** (dimensions $p \times p$ under the classic method and $(N-p) \times p$ under the over-determined methods), having c as its first column and r as its first row, achieved by the following instruction:

T = toeplitz (x(p:N-1), x(p:-1:1));

The solution of this system of eqs. ($\mathbf{T.a} = -\mathbf{x}$) for the classic and LS methods is obtained in MATLAB using the matrix left division (also known as backslash) operator. If **T** is square and if it is invertible, the backslash operator solves the linear equations using the QR method. With an overdetermined system, LS should be used. The backslash operator is a collection of algorithms used to solve a linear system [25], selected according to the characteristics of matrix **T**. Taking into account that vector **x** is a matrix column:

a = − T \ x(p + 1:N);

In the case of the TLS option, the function **a = tls(T,-x(p + 1:N));** is called (**Code 2**).

Step 2.

The p roots of the polynomial are now obtained:

$$z^p + a[1]z^{p-1} + ... + a[p] = 0$$

The MATLAB instruction $r = roots(c)$ returns a column vector whose elements are the roots of the polynomial c. Row vector c contains the coefficients of a polynomial, ordered in descending powers. If c has n + 1 components, the polynomial it represents is $c_1s^n + ... + c_ns + c_{n+1}$.

The input vector for the **roots** function must be a row vector and must contain the element $a[0] = 1$, which was not obtained in the previous solution. Its implementation is therefore.

c = transpose([1; a]);

r = roots(c);

Based on r, and having defined the acquisition period Ts, it is possible to find the values of the damping factor (α_k) and frequency (f_k):

alpha = log(abs(r))/Ts;

freq = atan2(imag(r),real(r))/(2*pi*Ts);

log is the Napierian logarithm and atan2 returns the four-quadrant inverse tangent.

Step 3: Obtain complex parameters h_k from roots z_k.

The number of equations (**len_vandermonde**) employed for the solution is set (p in classic and N in overdetermined systems) and the data matrix for the system of equations is constructed (6):

Z = zeros(len_vandermonde,p);

for i = 1:length(r).

Z(:,i) = transpose(r(i).^(0:len_vandermonde-1));

End

Finally, the following is solved:

h = Z \ x(1:len_vandermonde);

In the case of the TLS option, the function **h = tls(Z, x(1: len_vandermonde));** (**Code 2**) is called. In the TLS algorithm, SVD is used. The infinite values therefore have to be converted into maximum representative values beforehand otherwise the SVD function will yield an error.

The solutions yield the initial amplitude (A_k) and initial phase (θ_k) values:

Amp = abs(h);

theta = atan2(imag(h),real(h));

The function that solves a linear system using the TLS method (**Code 2**) receives as arguments matrices **A** and **b**, which define the linear system to solve: **Function x = tls(A,b).** The number of columns in matrix **A** is obtained (**[∼,n] = size(A);**) and augmented matrix **C** (**C = [A b]**) is constructed while matrix **V** of the SVD decomposition is obtained via the instruction **[∼,∼,V] = svd(C);** the TLS solution (if it exists) is obtained by applying the formula (12) to matrix **V**.

Fernández Rodríguez *et al. BMC Bioinformatics*   (2018) 19:451

Page 5 of 14

Code 1. Polynomial implementation

```
function [Amp,alfa,freq,theta] = polynomial_method (x,p,Ts,method)
% method: 'classic', 'ls' or 'tls' (case insensitive)
% define the solving methods
CLASSIC = 0;
LS = 1;
TLS = 2;

N = length(x);

if strcmpi(method,'classic')
if N ~= 2*p
disp ('ERROR: length of x must be 2*p samples in classical method.');
Amp = [];
alfa = [];
freq = [];
theta = [];
return;
else
solve_method = CLASSIC;
end
elseif strcmpi(method,'LS')
solve_method = LS;
elseif strcmpi(method,'TLS')
solve_method = TLS;
else
disp ('ERROR: error in parsing the argument "method".');
Amp = [];
alfa = [];
freq = [];
theta = [];
return;
end


%% step 1

T = toeplitz(x(p:N-1),x(p:-1:1));

switch solve_method
case (CLASSIC, LS)
a = - T\x(p+1:N);
case TLS
a = tls(T,-x(p+1:N));
end
%% check for indeterminate forms
indeterminate_form = sum(isnan(a) | isinf(a));
if (indeterminate_form)
Amp = []; alfa = []; freq = []; theta = [];
return;
end

%% step 2
c = transpose([1; a]);
r = roots(c);
alfa = log(abs(r))/Ts;
freq = atan2(imag(r),real(r))/(2*pi*Ts);

% In case alfa equals to +/-Inf the signal will not be recovered for n=0%
(Inf*0 = Nan). Making alfa = +/-realmax that indeterminace will be solved
alfa(isinf(alfa))=realmax*sign(alfa(isinf(alfa)));

%% step 3
switch solve_method
case CLASSIC
len_vandermonde = p; % exact case (N=2p) find h with p samples
case LS
len_vandermonde = N; % overdetermined case (N>2p) find h with N samples
case TLS
len_vandermonde = N; % overdetermined case (N>2p) find h with N samples
end

Z = zeros(len_vandermonde,p);
for i=1:length(r)

Z(:,i) = transpose(r(i).^(0:len_vandermonde-1));
end

rZ = real(Z);
iZ = imag(Z);
% here Inf values are substituted by realmax values
rZ(isinf(rZ))=realmax*sign(rZ(isinf(rZ)));
iZ(isinf(iZ))=realmax*sign(iZ(isinf(iZ)));

Z = rZ+1i*iZ;


switch solve_method
case (CLASSIC,LS)
h = Z\x(1:len_vandermonde);
case TLS
% if exists nan values SVD won't work
indeterminate_form = sum(sum(isnan(Z) | isinf(Z)));
if (indeterminate_form)
Amp = []; alfa = []; freq = []; theta = [];
return;
else
h = tls(Z,x(1:len_vandermonde))
end
end
Amp = abs(h);
theta = atan2(imag(h),real(h));
```

Code 2. Total least square function

```
function x = tls(A,b)

[~,n] = size(A);
C = [A b]; % C: augmented matrix
[~,~,V] = svd(C); % Matrix V (eq. 11)
VXY = V(1:n,1+n:end); % Submatrix: first n rows and the n+1 to last column
VYY = V(1+n:end,1+n:end); % Take the bottom-right block of V.
if VYY == 0
disp ('ERROR: not TLS solution');
x = zeros(n,1);
return;
end
x = -VXY/VYY;
```

Matrix pencil method

Steps 1 and 2 of the polynomial method yield the roots of the characteristic polynomial that coincide with the signal poles z_k. An alternative solution is to use the MPM to find z_k directly by solving a generalized eigenvalue problem.

In general, given two matrices $\mathbf{Y_1} \in \mathbb{C}^{mxn}$, $\mathbf{Y_2} \in \mathbb{C}^{mxn}$, the set of matrices of the form $Y_2 - \lambda Y_1$ ($\lambda \in \mathbb{C}$) is a matrix pencil [26].

In our case, to implement MPM a rectangular Hankel matrix $\mathbf{Y}$ is formed from the signal ($x[n]$, $n = 1,...N$), where, in this method, p is the pencil parameter:

$$\mathbf{Y} = \begin{pmatrix} x[1] & x[2] & \cdots & x[p] & x[p+1] \\ x[2] & x[3] & \cdots & x[p+1] & x[p+2] \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x[N-p] & x[N-p+1] & \cdots & x[N-1] & x[N] \end{pmatrix}_{(N-p)\times(p+1)}$$

(13)

This matrix is used to create matrices $\mathbf{Y}_1$ and $\mathbf{Y}_2$. $\mathbf{Y}_1$ is constructed by eliminating the last column of $\mathbf{Y}$ while $\mathbf{Y}_2$ is constructed by eliminating the first column of $\mathbf{Y}$:

$$\mathbf{Y}_1 = \begin{pmatrix} x[1] & x[2] & \cdots & x[p] \\ x[2] & x[3] & \cdots & x[p+1] \\ \vdots & \vdots & \ddots & \vdots \\ x[N-p] & x[N-p+1] & \cdots & x[N-1] \end{pmatrix}_{(N-p)\times p}$$

(14)

$$\mathbf{Y}_2 = \begin{pmatrix} x[2] & \cdots & x[p] & x[p+1] \\ x[3] & \cdots & x[p+1] & x[p+2] \\ \vdots & \ddots & \vdots & \vdots \\ x[N-p+1] & \cdots & x[N-1] & x[N] \end{pmatrix}_{(N-p)\times p}$$

(15)

Where M is the real number of poles of function $x[n]$, if $M \le p \le (N - M)$ is fulfilled, the poles z_k (k = 1,....p) are the generalized eigenvalues of the matrix pencil $\mathbf{Y}_2 - \lambda\mathbf{Y}_1$ [27]; matrices $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are ill-conditioned and therefore the QZ-algorithm is not stable enough to yield the generalized eigenvalues [5]. It is more efficient to obtain the values of z_k from the following expression:

$$z_k = \text{eigenvalues}\left(\mathbf{Y}_1^+ \mathbf{Y}_2\right) \tag{16}$$

Where $\mathbf{Y}_1^+$ is the Moore–Penrose pseudoinverse matrix of $\mathbf{Y}_1$, defined as:

$$\mathbf{Y}_1^+ = \left[\mathbf{Y}_1^H \mathbf{Y}_1\right]^{-1} \mathbf{Y}_1^H \tag{17}$$

The values z_k yield the parameters α_k and frequency f_k (Equations 5 and 6); The final step coincides with Step 3 of the Prony polynomial method: solving the system $\mathbf{Z}_{pxp}.\mathbf{h}_{px1} = \mathbf{x}_{px1}$ and obtaining A_k and θ_k (Equations 8 and 9).

Coding of the MPM in MATLAB is done in **Code 3**, the function call being.

Function   [Amp,alpha,freq,theta] = matrix_pencil (x,p,Ts)

The first step is to obtain the matrix Y then, based on that, matrices Y_1 and Y_2. To achieve this, the following instruction is employed:

Y = hankel (x(1:end-p), x(end-p:end));
To obtain Y_1, the last column is eliminated.
Y1 = Y (:,1:end-1);
To obtain Y_2, the first column is eliminated.
Y2 = Y (:,2:end);
The eigenvalues are obtained (Equation 16).
l = eig (pinv(Y1)*Y2);

eig (A) is a function that returns the eigenvalues of **A** while **pinv(A)** yields the Moore–Penrose pseudoinverse matrix of **A** which, in this case, corresponds to the expression in Equation 17.

The frequency (f_k) and damping factor (α_k) values are obtained from the eigenvalues in the same way as the roots are obtained in the polynomial method:

alpha = log(abs(l))/Ts;
freq = atan2(imag(l),real(l))/(2*pi*Ts);

To calculate the initial amplitude and phase values (A_k and θ_k), the steps followed are exactly the same as in the polynomial method.

Code 3 Matrix pencil method

```
Function [Amp,alfa,freq,theta] = matrix_pencil(x,p,Ts)

N = length(x);
Y = hankel(x(1:end-p),x(end-p:end));

Y1 = Y(:,1:end-1); % Y matrix without the last column
Y2 = Y(:,2:end); % Y matrix without the first column

l = eig(pinv(Y1)*Y2); % eigenvalues (eq. 16)

% frequency and damping factor as Prony's method
alfa = log(abs(l))/Ts;
freq = atan2(imag(l),real(l))/(2*pi*Ts);

% complex residues (amplitudes and angle phases) as Prony's method
Z = zeros(N,p);
for i=1:length(l)
Z(:,i) = transpose(l(i).^(0:N-1));
end

rZ = real(Z);
iZ = imag(Z);

rZ(isinf(rZ))=realmax*sign(rZ(isinf(rZ)));
iZ(isinf(iZ))=realmax*sign(iZ(isinf(iZ)));

Z = rZ+1i*iZ;
h = Z\x;
Amp = abs(h);
theta = atan2(imag(h),real(h));
```

Results

The methods described are applied in two situations: a) approximation of synthetic signals and b) filtering of mfVEP signals.

Synthetic functions

1 000 Functions are generated ($g_i[n]$) with $N = 1\ 024$ points each ($i = 1, \ldots 1\ 000$; $n = 0, \ldots 1\ 023$), according to the following expression

$$g_i[n] = \sum_{k=0}^{9} A_k . e^{\alpha_k . n . T_S} . \cos\left(2.\pi.f_k.n.T_S + \theta_k\right) \tag{18}$$

The parameters of the functions have a uniform random distribution at the following intervals: $A_k \in [1, 10]$; $\alpha_k \in [0, -4]$, $f_k \in [1, 31]$, $f_i \ne f_j$; $\theta_k \in [-\pi, \pi]$ and $f_0 = 0$.

Due to the possible existence of numerical errors in the computational approximation of the functions it is advisable to evaluate the error between the original function ($g_i[n]$) and its approximation ($\tilde{g}_i$) using Prony's method. The precision of the approximation

obtained from the normalized root-mean-square error is used:

$$G = 1 - \frac{\|g_i[n] - \widetilde{g_i[n]}\|}{\|g_i[n] - \overline{g_i}\|} \qquad (19)$$

$\|.\|$ indicates the 2-norm and $\bar{g_i}$ is the mean of the reference signal.

If for a certain function $G \geq 0.60$ is fulfilled, the approximation is considered correct. Table 1 shows the number of functions correctly approximated by the Prony LS, Prony TLS and MPM methods and for the two different parameters (N, p).

None of the methods implemented works 100% correctly ($G \geq 0.60$ for the 1000 $g_i[n]$ functions in all the situations tested). If the mean number of functions well-approximated by each method is considered, the best result is obtained with MPM ($\overline{MPM} = 999.55$) and the worst is obtained with TLS ($\overline{TLS} = 677.39$). The LS method yields the correct approximation in 60.52% of cases, the TLS method in 2.63% of cases and the MPM method in 92.10% of cases tested in this experiment.

In general, the results obtained using LS and MPM are very similar, as the MATLAB roots(·) function generates the companion matrix of the polynomial and uses the QR-algorithm to obtain its eigenvalues.

Figure 1 shows the roots obtained using the LS and MPM methods for one of the $g_i[n]$ signals ($N = 256$, $p = 30$). The correct number of roots for signal $g_i[n]$ is $M = 19$; in both examples, $p = 30$ roots are obtained, though with the MPM method 12 roots are equal to 0. This is because in the LS method the range of the companion matrix is always equal to p and, consequently, p roots are obtained. In the MPM method, the range of matrix $(\mathbf{Y}_1^+ \mathbf{Y}_2)$ is less than or equal to p ($r = \text{rank}(\mathbf{Y}_1^+ \mathbf{Y}_2) \leq p$) and r roots other than zero and (p-r) roots equal to 0 are obtained [5]. In the example shown, $r = 18$ is fulfilled. The differences in the results between the two methods are evident in Step 3 and are due to computational errors.

mfVEP filtering

The mfVEP technique [28] can be used to obtain the electrophysiological response of the primary visual cortex to stimuli produced in a large number (e.g. 60) of sectors of the visual field. Generation of the visual stimulus is governed by the same pseudorandom sequence [29] used to

Table 1 Result of approximation of synthetic functions

Number of functions $g_i[n]$ correctly approximated

N	p	LS	TLS	MPM
1024	30	902	811	990
	40	868	499	1000
	50	826	499	1000
	100	997	322	1000
	150	1000	315	1000
	200	1000	375	1000
	250	1000	358	1000
	300	1000	288	1000
	400	1000	224	1000
	500	999	137	1000
512	30	941	741	1000
	40	974	660	1000
	50	996	682	1000
	60	999	618	1000
	70	1000	544	1000
	100	1000	565	1000
	150	1000	622	1000
	200	1000	579	1000
	220	1000	517	1000
	250	999	516	1000
256	30	984	909	1000
	40	998	872	1000
	50	998	855	1000
	60	1000	826	1000
	70	1000	778	1000
	80	1000	862	1000
	90	1000	827	1000
	100	1000	758	1000
	110	1000	733	1000
	120	996	758	1000
128	20	994	995	994
	30	1000	960	1000
	40	1000	956	1000
	50	1000	931	1000
	60	1000	910	1000
64	20	1000	1000	999
	25	1000	969	1000
	30	1000	970	1000
Average value per method		$\overline{LS} = 986.08$	$\overline{TLS} = 677.39$	$\overline{MPM} = 999.55$

Fig. 1 Poles obtained using the polynomial (LS) and MPM methods
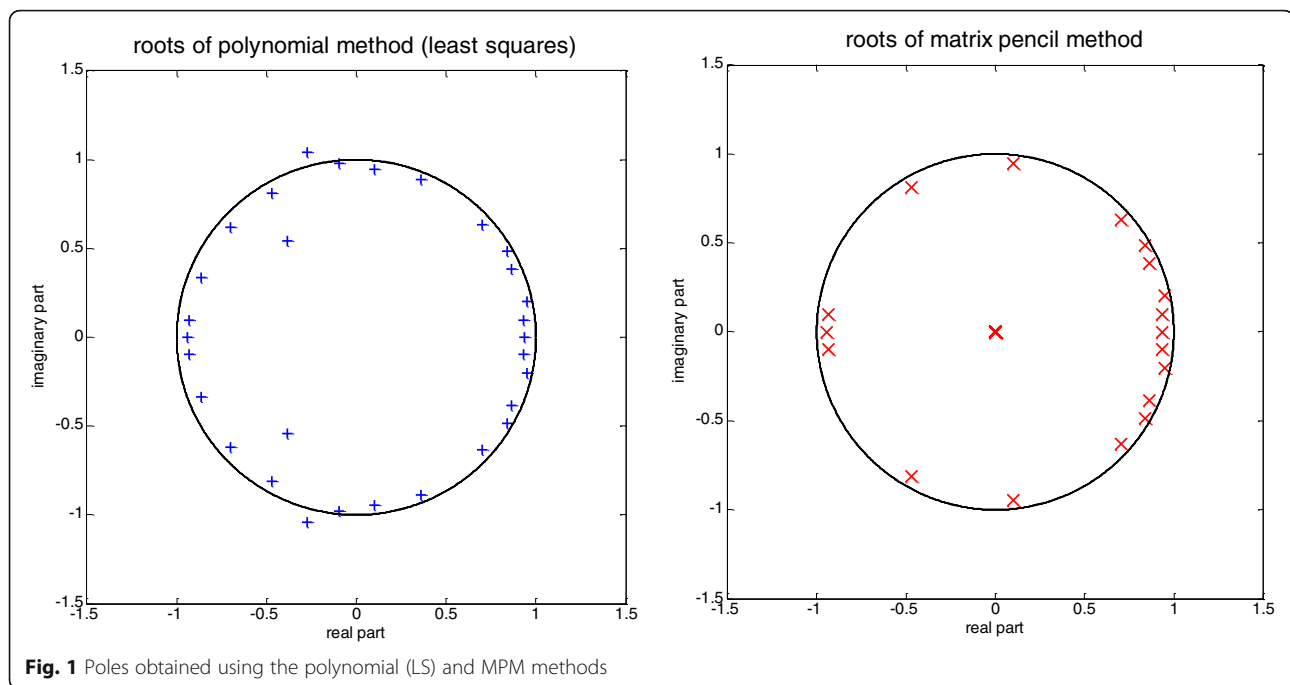
separate the individual responses of each sector from the continual EEG recording obtained using electrodes. Analysis of mfVEP signals is employed in diagnosis and study of patients with glaucoma, amblyopia, nerve optic drusses, optic neuritis, multiple sclerosis and other pathologies.

The aim of this test is to evaluate whether mfVEP signal filtering in the Prony domain improves the separation between the signals of control subjects and the signals of patients with MS. It uses the signal-to-noise ratio (SNR) of the records as the parameter. The discrimination factor is evaluated using the area under the ROC curve (AUC). The results achieved by applying the conventional method to mfVEP records are then compared: signals filtered using the discrete Fourier transform (DFT) between 0 and 35 Hz and the signals filtered in the Prony domain.

This experiment uses a database of mfVEP signals captured from 28 patients (age 34.39 ± 10.09 years, 7 males and 21 females) diagnosed with MS according to the McDonald criteria; the signals were obtained from 44 eyes in 22 control subjects (age 30.20 ± 7.55 years, 10 males and 12 females) with normal ophthalmologic and neurological examination results. The study protocol adhered to the tenets of the Declaration of Helsinki and was approved by the local Institutional Review Board (Comité de Ética en Investigación Clínica del Hospital Universitario

Príncipe de Asturias, Alcalá de Henares, Spain). Written informed consent was obtained from all participants.

mfVEP signals were recorded monocularly with VERIS software 5.9 (Electro-Diagnostic Imaging, Inc., Redwood City, CA). The visual stimulus was a scaled dartboard with a diameter of 44.5 degrees, containing 60 sectors, each with 16 alternating checks. The luminance for the white and black checks were 200 and < $3 \, cd/m^2$, respectively. The checks in each sector were reversed in contrast using a pseudorandom sequence (m-sequence) at a frame rate of 75 Hz.

The mfVEP signals were obtained using gold cup electrodes (impedance < 2 KΩ). The reference electrode was positioned on the inion (E_R) and the ground electrode on the forehead. The active electrodes were placed 4 cm above the inion (E_A) and 1 cm above and 4 cm either side of the inion (E_B, E_C). The difference between the signals of the active electrodes was used to obtain channels $CH_1 = E_A$-E_R, $CH_2 = E_B$-E_R and $CH_3 = E_C$-E_R. Three additional derived channels were obtained ($CH_4 = CH_1$-CH_2, $CH_5 = CH_1$-CH_3, $CH_6 = CH_2$-CH_3). Therefore, the data from 6 channels were processed. In the analogue phase, the signals were amplified at a gain of 10^5 at a bandwidth between 3 and 100 Hz. The sampling frequency was 1200 Hz, obtaining 600 samples in each recording (length 500 ms).

The conventional signal-processing method consists of bandpass filtering between 0 and 35 Hz using the fast Fourier transform; these signals are denominated X_{DFT}.

One method for determining the intensity of the mfVEP records is to use the signal-to-noise ratio, defined by the following expression:

$$\text{SNR} = \frac{\text{RMS}_{45-150 \text{ ms}}}{\text{mean}\left(\text{RMS}_{325-430 \text{ ms}}\right)}; \qquad (20)$$

In an mfVEP, the physiological response to the stimulus presents in the 45–150 ms interval (signal window) following onset. In the 325–430 ms interval (noise window) only noise is considered to be recorded. $\text{RMS}_{45-150 \text{ ms}}$ and $\text{RMS}_{325-430 \text{ ms}}$ are the root mean square (RMS) amplitudes in the signal window and noise window, respectively.

Signal processing using Prony's method is carried out in the following steps:

1. The Prony approximation is obtained (X_{LS}, X_{TLS}, X_{MPM}, with $p = 250$, $N = 600$) for the X_{DFT} signals. The number of MS signals is $N_{MS} = 20,160$ ($28 \times 2 \times 60 \times 6$) and the number of control signals is $N_{CONTROL} = 15,840$ ($22 \times 2 \times 60 \times 6$).

2. Correct approximation of the X_{DFT} signal is checked against the expression shown in Equation 19 and considering G ≥ 0.45. Figure 2 shows an example of a signal approximated using the LS method.

3. The correctly approximated signals are bandpass-filtered in the Prony domain, selecting the 10 lowest-frequency components. The MATLAB code is shown in Code 4. Figure 3 shows an example of a filtered signal.

4. The SNR value of the X_{DFT} and Prony-filtered signals (X_{LS_F}, X_{TLS_F}, X_{MPM_F}) is obtained and the discrimination value between the signals of subjects with MS and control subjects is calculated.

Similar to the case of the synthetic signals, the LS method only correctly approximated a low percentage of records (48.79% of the control records and 42.90% of the MS records) (Table 2). The LS and MPM methods yielded the same results, achieving correct approximation in over 99% of cases. The signal intensity value in the control subjects is greater than in the subjects with MS. Filtering the signals using the conventional method yields an AUC value of 0.6538; using the TLS method yields practically the same result (AUC = 0.6472) while using the LS and MPM methods yields a value of 0.7055. This example shows that filtering in the Prony domain can increase the capacity to discriminate between the

signals of control subjects and those of patients with MS.

Code 4 Prony Low Pass filter

```
function yy = prony_lowpass_filter (x,p,Ts,type,method,Ncomp)
% this function calls polynomial_method or matrix_pencil functions to %
analyze the input signal x depending on the argument type.
% Parameters of analysis are p (number of complex exponentials), Ts
% (sampling time) and method (least squares or total least squares)
% for the polynomial method.
% Ncomp are the number of components from which the signal will be
% recovered; Ncomp should be less than p/2.


if strcmpi (type,'mpm')
MPM = 1;
elseif strcmpi (type,'polynomial')
MPM = 0;
end

if MPM == 0
[Amp, alfa, freq, theta] = polynomial_method(x,p,Ts,method);
else
[Amp, alfa, freq, theta] = matrix_pencil (x,p,Ts);
end

[freq,ix] = sort(freq);
Amp = Amp(ix);
alfa = alfa(ix);
theta = theta(ix);

% maximum number of components (positive frequencies)
MAX_COMP = length(find(freq>=0));
ii = find(freq<0,1,'last');

if Ncomp>MAX_COMP
Ncomp = MAX_COMP;
end

% get the higher positive frequency (fcut) and filter from its
% negative value (-fcut) to the positive one (fcut)
fcut = freq(ii+Ncomp); % fcut from 0 Hz to selected component number
ix = find(freq>=-fcut & freq<=fcut);
comp = length(ix); % number of components used to recover the signal

% filtering step
freq_ = freq(ix);
Amp_ = Amp(ix);
alfa_ = alfa(ix);
theta_ = theta(ix);

% recovering temporal filtered signal
N= length(x),

n = 0:N-1;

n = repmat(n,[comp 1]);

k = exp.(repmat(alfa_, [1 N]).*n*Ts);

k(isinf(k))=realmax*sign(k(isinf(k)));

Y = transpose(repmat(Amp_,[1 N]).*k)*...
cos(2*pi*Ts*repmat(freq_, [1 N]).*n + repmat(theta_,[1 N]));

yy = diag(Y);
```
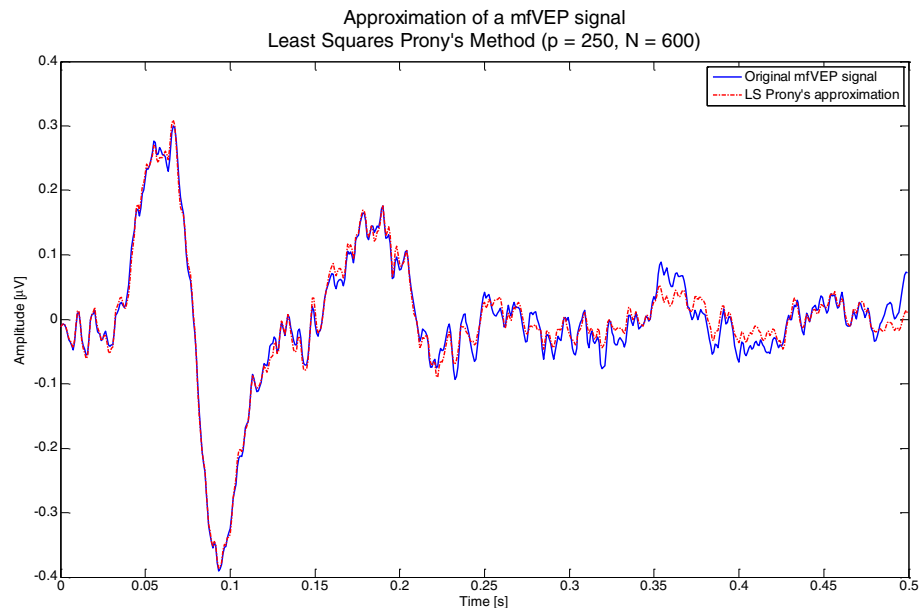
Fig. 2 Example of approximation of an mfVEP signal using Prony's method (LS)

Discussion

In this paper we have used general MATLAB functions to implement the principal methods of function approximation based on the linear combination of exponentials: the polynomial Prony method (classic, LS and TLS) and the matrix pencil method. In the polynomial method, signal poles (frequencies and damping factors) are found as roots of a polynomial while the MPM obtains the poles by finding the eigenvalues of a matrix pencil.

Currently, the most common method is Fourier analysis, which represents a signal as a summation of continuous undamped sinusoidal functions with frequency and integer times the fundamental frequency (harmonics). In contrast, the p components of a Prony series may be complex exponentials. In general, the Prony spectrum will be non-uniformly spaced in the frequency scale (as it is one of the estimated parameters), depending on the observed data [30].
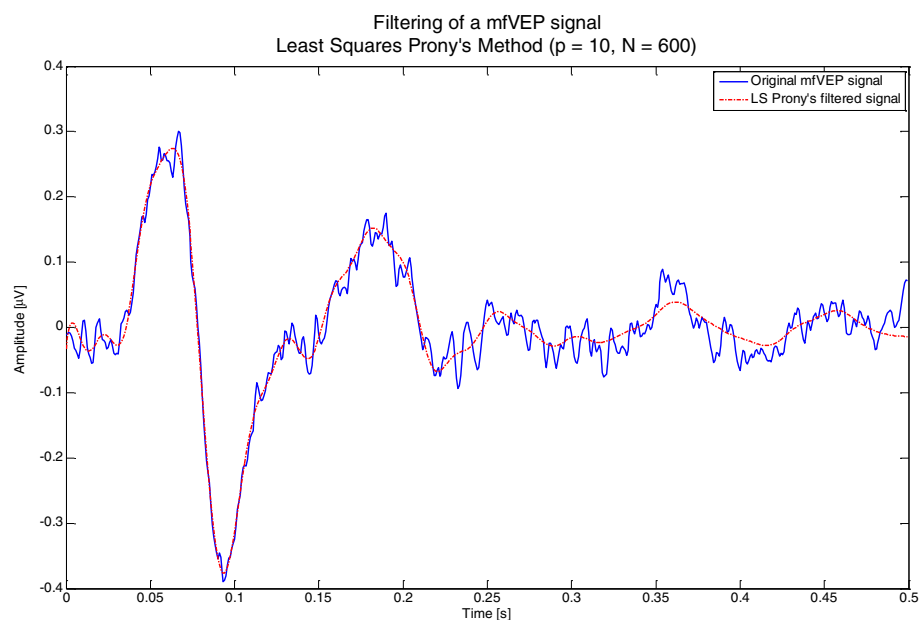

Fig. 3 Example of Prony filtering of an mfVEP signal

Table 2 Results of filtering mfVEP signals (N = 600, p = 250, T_s = 1/1200 *s*)

	Well-approximated control signals (%)	Well-approximated MS signals (%)	SNR$_{CONTROLS}$	SNR$_{MS}$	AUC
DFT	–	–	3.59 ± 2.89	2.44 ± 2.11	0.6538
LS	99.57%	99.91%	4.95 ± 4.19	2.85 ± 2.49	0.7055
TLS	48.79%	42.90%	3.65 ± 2.89	2.54 ± 2.24	0.6472
MPM	99.57%	99.91%	4.95 ± 4.19	2.85 ± 2.49	0.7055

Prony modelling produces higher frequency resolution than DFT methods due to its reliance on autoregressive modelling [31]. Another advantage is that it is a natural transformation for impulse responses since it uses damped sinusoids as a basis and therefore representation is efficient in terms of the number of coefficients required [32].

Not all mathematical signals can be approximated using Prony's method [33] and computational finite arithmetic also generates errors. Consequently, the results of computational implementation of the Prony methods depend on the characteristics and number of points of the signal to be interpolated, on the p number of functions and on the use of computational procedures not susceptible to ill-conditioning issues. Furthermore, these potentially ill-conditioned operations are concatenated, thereby increasing the instability issues. For example, since the second step of Prony's method is an ill-conditioned problem and round-off errors must exist for the linear prediction parameters to be computed in the first step, the estimation of z_k in the second step of Prony's method can contain significant error [34].

In our experimental approximation of synthetic functions, the best result was obtained using the MPM and LS methods, while the effectiveness of the TLS method was shown to be highly dependent on the number of points and on the p number of functions (Table 1). In some cases, when the number of exponentials is relatively high, the sensitivity of roots of the characteristic polynomial to perturbations of their coefficient is likewise high [4] and Prony's method may be unstable.

In a second experiment, we low-pass-filtered mfVEP signals in the Prony domain in order to evaluate the improvement in the capacity to discriminate between signals of control subjects and those of MS patients. Selecting the first 10 components of each record reveals that the AUC value between the signals of healthy subjects and those of MS subjects increases by between 0.3% and 4.7% depending on the method compared. The smallest improvement was obtained with the TLS method and the greatest improvement with the LS and MPM methods.

Coding in MATLAB used the functions directly available in this programming language. However, these evidently have their computational limitations and could be replaced with better alternatives. Various aspects that could improve the code presented in this paper are discussed below.

Solution of linear systems

Solution of the linear systems using the classic and LS methods was implemented with the MATLAB *mldivide* (\\) operator. Although the *mldivide* operator is valid for most cases (it selects between the LU, Cholesky, LDLT or QR-factorization methods, among others, depending on the characteristics of matrix **A** [35]), it may be more efficient to implement other algorithms.

The numerical stability of the solution in linear algebra may be evaluated by the condition number and the numerical rank of matrix **A**. The condition number is defined as: $k_2(\mathbf{A}) = \frac{\sigma_{max}}{\sigma_{min}}$; a low condition number usually means that the system is well-conditioned. The rank (r) of a matrix is the number of linear independent rows (or columns) (r ≤ min {m, n}) and is equal to the number of singular values (σ_i) in the matrix other than zero. When r = min {m, n} the matrix has full range, otherwise it is rank-deficient. If **A** is nearly rank-deficient (σ_{min} is small), then the solution **x** is ill-conditioned and possibly very large. A more robust solution to obtain the effective rank may be to evaluate the number of singular values of $\mathbf{AA^H}$ or $\mathbf{A^HA}$ above a specified tolerance. Analysing the condition number and the rank of a matrix may make it possible to select the best method for system solution.

Least squares

In general, although the normal equation is the fastest method it is not used to solve systems by LS as it yields worse numerical results than other methods. In the normal equation, accuracy depends on $k_2(\mathbf{AA^H}) = k_2^2(\mathbf{A})$, although this method may be used if **A** is well-conditioned [36]. If **A** is rank-deficient, then **x** = **A\B** is not necessarily the minimum norm solution. The more computationally expensive x = *pinv(A)*B* computes the minimum norm least-squares solution. Specifically, the function *pinv(A, tol)* returns the Moore–Penrose pseudoinverse, obtained by SVD decomposition where the values above tolerance (tol) are set to zero; this may be adapted to an ill-conditioned problem (**A** is not of full rank). Another

option to obtain the Moore–Penrose pseudoinverse is proposed in [37], which makes use of QR-factorization and an algorithm based on a reverse order law for generalized inverse matrices; this method was later refined in [38]. An iterative solution to obtain the Moore–Penrose pseudoinverse was published in [39].

TLS

The TLS method implemented is the one that yielded the worst computational performance. This method performs SVD of the augmented matrix $\mathbf{C} := [\mathbf{A} : \mathbf{b}]$; If $\mathbf{C}$ is rank-deficient or nearly rank-deficient (its singular values decay gradually), it may be advisable to truncate its small singular values [40]. [41] presents basic information, references and applications for the TLS method. In [42], an interactive method is proposed which combines orthogonal projections to a sequence of generalized Krylov subspaces of increasing dimensions and Newton's method. The introduction to [43] presents various alternatives to obtaining the solution using the TLS method and the authors present a solution based on randomized algorithms.

Roots

Numerical solution of a polynomial is a classic problem in mathematical research [44]. Methods available with which to obtain the roots of a polynomial include Laguerre [45], Bairstow, Græffe and Müller, Horner, Jenkins and Traub, and Newton [46], etc., with differing performances in terms of accuracy, convergence and speed. The code presented uses the *roots()* function used by the QR-algorithm on the balanced Frobenius companion matrix to compute its eigenvalues.

Eigenvalues and SVD

The eigenvalues of a square matrix $\mathbf{A}$ are the roots of its characteristic polynomial $\det(\mathbf{A} - \lambda \mathbf{I}) = 0$. However, singular values of $\mathbf{A}$ are non-negative square roots of eigenvalues of $(\mathbf{A}^T\mathbf{A})$, meaning that both methods are related. The general idea is to diagonalize the target matrix as the values of the diagonal are the eigenvalues. All methods to solve the eigenvalue problem are of an iterative nature [47]. The built-in MATLAB function eig($\mathbf{A}$) uses the generalized Schur decomposition method (implemented via the QR-algorithm or its variants), which consists of interactively obtaining an upper triangular matrix $\mathbf{U}$, in which the values of its diagonal are the eigenvalues of $\mathbf{A}$. The QR-algorithm can be adapted to small or moderately large non-symmetrical eigenvalue problems [48]. For large matrices, [49] provides possible alternatives.

Prony-like methods

Other modifications have been made to the Prony method, generally with the intention of improving its numerical stability. If any of the parameters of equation (1) are known, the Prony method makes it easier to find a robust solution. In the polynomial method, small variations in the coefficients of equation (2) due to signal noise can result in large variations in its zeros and, consequently, the frequencies of the approximation will vary greatly. Parametric spectral estimation techniques, such as MUSIC (MUltiple SIgnal Classification) [50], ESPRIT (Estimation of Signal Parameters via Rotational Invariance Techniques) [51] or fast ESPRIT [52] offer an alternative that in many cases make it possible to obtain more robust solutions. [53] presents an algorithm for the factorization of a matrix pencil based on QR decomposition of a rectangular Hankel matrix, which simplifies the ESPRIT method.

The NAFASS (Non-orthogonal Amplitude Frequency Analysis of the Smoothed Signals) approach [54] makes it possible to obtain the set of frequencies that make up strongly correlated random sequences with $N < 1500$. [55] presents the physical interpretation of the Prony decomposition as a linear recording of memory effects that can exist in random sequences in which the Fourier decomposition is a partial case. [56] improves the NAFASS method, presenting a linear recurrence expression that obtains the set of frequencies.

Another way to obtain more robust results is to act on the signals before obtaining their decomposition in a Prony series by using pre-filtering [57]. In the modified instantaneous Prony method [58] the input data used in an application of speech parameter extraction are those derived from the signal x[n] instead of adjacent samples.

Applying the Prony method to a time window that can be moved along the x[n] signal makes it possible to perform time–frequency analysis. One such example could be the short-time matrix pencil method (STMPS) successfully used to obtain antennae responses [59]. The Piecewise Prony Method [60] essentially consists of dividing the signal to be interpolated into windows of variable length and sampling rate.

Conclusions

Decomposition of a signal using Prony's method can be considered a generalization of the Fourier decomposition. Although the method has been known since 1795, its application in engineering has only increased since about the 1970s as computer use has grown. This paper has presented the theoretical bases and their piece-by-piece implementation in MATLAB. It has also shown some of their limitations and the

benefit of improving the quality of the mfVEP signals. With the information provided, readers can begin practical implementation of the most common Prony methods, test the reliability of the results and, if applicable, research other methods more appropriate to their areas of research.

Abbreviations
AUC: Area under the ROC curve; DFT: Discrete Fourier transform; LS: Least squares; mfVEP: Multifocal visual-evoked potential; MPM: Matrix pencil method; MS: Multiple sclerosis; RMS: Root mean square; ROC: Receiver-operating-characteristic; SNR: Signal-to-noise ratio; SVD: Singular value decomposition; TLS: Total least squares

Availability and requirements
Project name: MAT_PRONY.
Project home page: Not applicable.
Operating system(s): Platform independent.
Programming language: Matlab.
Other requirements: Not applicable.
License: Matlab licences.
Any restrictions to use by non-academics: Not applicable.

Availability of data and materials
The Matlab code is included in this published article.

Authors' contributions
AFR and LB conceived and designed the experiments; LdSR and JMRA performed the experiments; ELG and JMMJ revised the mathematics concepts; LB and AFR. wrote the paper; all the authors reviewed and approved the manuscript.
.

Ethics approval and consent to participate
The mfVEP signals adquisition study protocol was approved by the Comité de Ética en Investigación Clínica del Hospital Universitario Príncipe de Asturias, Spain and adhered to the tenets of the Declaration of Helsinki. All participants provided written informed consent.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Publisher's Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References
1. Prony R. Essai expérimental et analytique sur les lois de la dilatabilité des fluides élastiques, et sur celles de la force expansive de la vapeur de l'eau et de la vapeur de l'alkool, á différentes temperatures. J L'école Polytech. 1795; 1:24–76.
2. Householder, AS. On Prony's method of fitting exponential decay curves and multiple-hit survival curves. Oak Ridge National Laborator, 1950.
3. Markovsky I, Van Huffel S. Overview of total least-squares methods. Signal Process. 2007;87:2283–302. https://doi.org/10.1016/j.sigpro.2007.04.004.
4. Guillaume P, Schoukens J, Pintelon R. Sensitivity of roots to errors in the coefficient of polynomials obtained by frequency-domain estimation methods. IEEE Trans Instrum Meas. 1989;38:1050–6. https://doi.org/10.1109/19.46399.
5. Hua Y, Sarkar TK. Matrix pencil method for estimating parameters of exponentially damped/undamped sinusoids in noise. IEEE Trans Acoust. 1990;38:814–24. https://doi.org/10.1109/29.56027.
6. Lobos T, Leonowicz Z, Rezmer J, Schegner P. High-resolution spectrum-estimation methods for signal analysis in power systems. IEEE Trans Instrum Meas. 2006;55:219–25. https://doi.org/10.1109/TIM.2005.862015.
7. Park SW, Schapery RA. Methods of interconversion between linear viscoelastic material functions. Part I—a numerical method based on Prony series. Int J Solids Struct. 1999;36:1653–75. https://doi.org/10.1016/S0020-7683(98)00055-9.
8. Sarkar T, Weiner D, Jain V. Some mathematical considerations in dealing with the inverse problem. IEEE Trans Antennas Propag. 1981;29:373–9. https://doi.org/10.1109/TAP.1981.1142573.
9. Fernández A, de Santiago L, Blanco R, Pérez-Rico C, Rodríguez-Ascariz JM, Barea R, Miguel-Jiménez JM, García-Luque JR, Ortiz del Castillo M, Sánchez-Morla EM, Boquete L. Filtering multifocal VEP signals using Prony's method. Comput Biol Med. 2015;56:13–9. https://doi.org/10.1016/j.compbiomed.2014.10.023.
10. Hansson M, Gänsler T, Salomonsson G. Estimation of single event-related potentials utilizing the Prony method. IEEE Trans Biomed Eng. 1996;43:973–81. https://doi.org/10.1109/10.536898.
11. Chen SW. A two-stage discrimination of cardiac arrhythmias using a total least squares-based Prony modeling algorithm. IEEE Trans Biomed Eng. 2000;47:1317–27. https://doi.org/10.1109/10.871404.
12. Bhuiyan M, Malyarenko EV, Pantea MA, Seviaryn FM, Maev RG. Advantages and limitations of using matrix pencil method for the modal analysis of medical percussion signals. IEEE Trans Biomed Eng. 2013;60:417–26. https://doi.org/10.1109/TBME.2012.2227318.
13. Bauman G, Bieri O. Matrix pencil decomposition of time-resolved proton MRI for robust and improved assessment of pulmonary ventilation and perfusion. Magn Reson Med. 2017;77:336–42. https://doi.org/10.1002/mrm.26096.
14. Gopalakrishnan R, Machado AG, Burgess RC, Mosher JC. The use of contact heat evoked potential stimulator (CHEPS) in magnetoencephalography for pain research. J Neurosci Methods. 2013;220:55–63. https://doi.org/10.1016/j.jneumeth.2013.08.015.
15. Goodman D. Brian: a simulator for spiking neural networks in Python. Front Neuroinform. 2008;2:5. https://doi.org/10.3389/neuro.11.005.2008.
16. Meyer R, Obermayer K. Pypet: a Python toolkit for data management of parameter explorations. Front. Neuroinform. 2016;10. https://doi.org/10.3389/fninf.2016.00038.
17. Sen SK, Shaykhian GA. MatLab tutorial for scientific and engineering computations. Nonlinear Anal Theory, Methods Appl. 2009:e1005–20. https://doi.org/10.1016/j.na.2009.01.069.
18. Ihlen EAF. Introduction to multifractal detrended fluctuation analysis in Matlab. Front Physiol. 2012;3. https://doi.org/10.3389/fphys.2012.00141.
19. Miguel-Jiménez JM, Boquete L, Ortega S, Cordero CA, Barea R, Blanco R. mfERG_LAB: software for processing multifocal electroretinography signals. Comput Methods Prog Biomed. 2012;108:377–87. https://doi.org/10.1016/j.cmpb.2012.02.013.
20. Perakakis P, Joffily M, Taylor M, Guerra P, Vila J. KARDIA: a Matlab software for the analysis of cardiac interbeat intervals. Comput Methods Prog Biomed. 2010;98:83–9. https://doi.org/10.1016/j.cmpb.2009.10.002.
21. Saber E. An introduction to difference equations. New York: Springer-Verlag; 2005. https://doi.org/10.1007/0-387-27602-5.

22. Reddy DC. Biomedical signal processing: principles and techniques. New York: McGraw-Hill; 2005.
23. Van Huffer S, Zha H. The total least squares problem. Handb. Stat. 1993;9: 377–408.
24. Van Huffel S, Vandewalle J. The total least squars problem: computational aspects and analysis. Philadelphia: Siam; 1991.
25. Yang WY, Cao W, Chung TS, Morris J. Applied numerical methods using MATLAB. Hoboken: Wiley; 2005.
26. Gantmacher FR. The theory of matrices, vol. 2. Moscow: GITTL; 1953.
27. Sarkar TK, Pereira O. Using the matrix pencil method to estimate the parameters of a sum of complex exponentials. IEEE Antennas Propag Mag. 1995;37:48–55. https://doi.org/10.1109/74.370583.
28. Baseler HA, Sutter EE, Klein SAA, Carney T. The topography of visual evoked response properties across the visual field. Electroencephalogr Clin Neurophysiol. 1994;90:65–81. https://doi.org/10.1016/0013-4694(94)90114-7.
29. Müller PL, Meigen T. M-sequences in ophthalmic electrophysiology. J Vis. 2016;16(1):15.
30. Mitrofanov G, Priimenko V. Prony filtering of seismic data. Acta Geophys. 2015;63:652–78. https://doi.org/10.1515/acgeo-2015-0012.
31. Lander P, Jones D, Berbari E, Lazzara R. Time-frequency structure of the high-resolution ECG. J Electrocardiol. 1994;27:207–12. https://doi.org/10.1016/S0022-0736(94)80093-6.
32. Demiralp T, Ademoglu A, Istefanopulos Y, Gülçür HO. Analysis of event-related potentials (ERP) by damped sinusoids. Biol Cybern. 1998;78:487–93. https://doi.org/10.1007/s004220050.
33. Slivinskas V, Šimonyte V. On the foundation of Prony's method. In: Stoch. Control, Elsevier; 1987. p. 121–6. https://doi.org/10.1016/B978-0-08-033452-3.50025-9.
34. James Hu SL, Yang WL, Li HJ. Signal decomposition and reconstruction using complex exponential models. Mech Syst Signal Process. 2013;40:421–38. https://doi.org/10.1016/j.ymssp.2013.06.037.
35. Davis TA, Duff IS. An unsymmetric-pattern multifrontal method for sparse LU factorization. SIAM J Matrix Anal Appl. 1997;18:140–58. https://doi.org/10.1137/S0895479894246905.
36. Demmel JW. Applied numerical linear algebra, SIAM; 1997. https://doi.org/10.1137/1.9781611971446.
37. Katsikis VN, Pappas D, Petralias A. An improved method for the computation of the Moore–Penrose inverse matrix. Appl Math Comput. 2011;217:9828–34. https://doi.org/10.1016/j.amc.2011.04.080.
38. Ataei A. Improved Qrginv algorithm for computing Moore-Penrose inverse matrices. ISRN Appl Math. 2014;2014:1–5. https://doi.org/10.1155/2014/641706.
39. Petković MD, Stanimirović PS. Two improvements of the iterative method for computing Moore–Penrose inverse based on Penrose equations. J Comput Appl Math. 2014;267:61–71. https://doi.org/10.1016/j.cam.2014.01.034.
40. Fierro RD, Golub GH, Hansen PC, O'Leary DP. Regularization by truncated total least squares. SIAM J Sci Comput. 1997;18:1223–41. https://doi.org/10.1137/S1064827594263837.
41. Markovsky I. Bibliography on total least squares and related methods. Stat Interface. 2010;3:329–34.
42. Lampe J, Voss H. Large-scale Tikhonov regularization of total least squares. J Comput Appl Math. 2013;238:95–108. https://doi.org/10.1016/j.cam.2012.08.023.
43. Xie, P., Wei, Y., Xiang, H. Perturbation analysis and randomized algorithms for large-scale total least squares problems. arXiv preprint arXiv:1401.6832, 2014.
44. Pan VY. Solving a polynomial equation: some history and recent progress. SIAM Rev. 1997;39:187–220. https://doi.org/10.1137/S0036144595288554.
45. Hansen E, Patrick M, Rusnak J. Some modifications of Laguerre's method. BIT. 1997;17:409–17. https://doi.org/10.1007/BF01933450.
46. Madsen K. A root-finding algorithm based on Newton's method. BIT. 1973; 13:71–5. https://doi.org/10.1007/BF01933524.
47. Golub GH, van der Vorst HA. Eigenvalue computation in the 20th century. J Comput Appl Math. 2000;123:35–65.
48. Chandrasekaran S, Gu M, Xia J, Zhu JA. Fast QR algorithm for companion matrices. In: Recent Adv. Matrix Oper. Theory, Birkhäuser Basel, Basel; 2007. p. 111–43. https://doi.org/10.1007/978-3-7643-8539-2_7.
49. Saad Y. Numerical methods for large eigenvalue problems, Society for Industrial and Applied Mathematics; 2011. https://doi.org/10.1137/1.9781611970739.
50. Schmidt R. Multiple emitter location and signal parameter estimation. IEEE Trans Antennas Propag. 1986;34:276–80. https://doi.org/10.1109/TAP.1986.1143830.
51. Roy R, Paulraj A, Kailath T. ESPRIT—A subspace rotation approach to estimation of parameters of cisoids in noise. IEEE Trans Acoust Speech Signal Process. 1986;34:1340–2. https://doi.org/10.1109/TASSP.1986.1164935.
52. Potts D, Tasche M. Fast ESPRIT algorithms based on partial singular value decompositions. Appl Numer Math. 2015;88:31–45. https://doi.org/10.1016/j.apnum.2014.10.003.
53. Potts D, Tasche M. Parameter estimation for nonincreasing exponential sums by Prony-like methods. Linear Algebra Appl. 2013;439:1024–39. https://doi.org/10.1016/j.laa.2012.10.036.
54. Nigmatullin RR, Osokin SI, Toboev VA. NAFASS: discrete spectroscopy of random signals. Chaos, Solitons Fractals. 2011;44:226–40. https://doi.org/10.1016/j.chaos.2011.02.003.
55. Nigmatullin RR, Khamzin AA, Machado JT. Detection of quasi-periodic processes in complex systems: how do we quantitatively describe their properties? Phys Scr. 2013;89:015201. https://doi.org/10.1088/0031-8949/89/01/015201.
56. Nigmatullin RR, Gubaidullin IA. NAFASS: fluctuation spectroscopy and the Prony spectrum for description of multi-frequency signals in complex systems. Commun Nonlinear Sci Numer Simulat. 2018;56:252–69. https://doi.org/10.1016/j.cnsns.2017.08.009.
57. Kumaresan R, Feng Y. FIR prefiltering improves Prony's method. IEEE Trans Signal Process. 1991;39:736–41. https://doi.org/10.1109/78.80860.
58. Azarov E, Vashkevich M, Petrovsky A. Instantaneous harmonic representation of speech using multicomponent sinusoidal excitation. Proc Annu Conf Int Speech Commun Assoc Interspeech. 2013:1697–701.
59. Rezaiesarlak R, Manteghi M. Short-time matrix pencil method for chipless RFID detection applications. IEEE Trans Antennas Propag. 2013;61:2801–6. https://doi.org/10.1109/TAP.2013.2238497.
60. Garoosi V, Jansen BH. Development and evaluation of the piecewise Prony method for evoked potential analysis. IEEE Trans Biomed Eng. 2000;47:1549–54. https://doi.org/10.1109/10.887935.