RAIT: the Rational Approximation and Interpolation Toolbox for Matlab, with Experiments on ECG Signals

Péter Kovács, Levente Lócsi

Abstract—There is a wide range of applications of rational function systems. Including in system, control theories and signal processing. A special class of rational functions, the so-called Blaschke functions and the orthonormal Malmquist–Takenaka (MT) systems are effectively used for representing signals especially electrocardiograms.

We present our project on a general MATLAB library for rational function systems and their applications. It contains Blaschke functions, MT systems and biorthogonal systems. We implemented not only the the continuous but the discrete versions as well, since in applications the latter one is needed. The complex and real interpretations are both available. We also built in methods for finding the poles automatically. Also, some interactive GUIs were implemented for visual demonstration that help the users in understanding the roles of certain parameters such as poles, multiplicity etc.

Keywords—Complex rational function systems, Digital signal processing, Discretization, Malmquist–Takenaka system, Matlab toolbox.

I. INTRODUCTION

Rational function systems play an important role in signal processing [1], for instance there is a wide range of their application in system and control theories, see [2]. These systems proved to be appropriate for system identification, furthermore they can be used for denoising and compression. In a wide variety of problems the may also provide a nice and compact representation of the given signal [3]. We have applied our tools mostly for analyzing electrocardiograms (ECG signals).

We present our toolbox for Mathworks MATLAB environment. Our aim was to construct a toolbox that is independent of the particular problem and can be used for processing signals generally.

Many methods and related theoretical results incorporated and implemented in this project. It contains Blaschke functions, Malmquist–Takenaka systems, and also biorthogonal systems to some elementary rational bases [4]. We implemented not only the continuous but the discrete versions as well [5]. Complex and real interpretations are both available. The used systems depend on some parameters, or 'inverse

poles'. The Nelder–Mead simplex method can be used for finding these parameters automatically [6]. Also, some interactive graphical user interfaces (GUI) have been designed. They help the users in understanding how these function systems and methods work, and one can see the effects of certain parameters in action. So the toolbox is also fit for educational purposes.

The toolbox can be downloaded from

http://numanal.inf.elte.hu/~locsi/rait/

The outline of this article is as follows. In Section II we summarise the mathematical methods, and rational function systems used in our tools: Blaschke functions, Malmquist–Takenaka systems, discretization properties, biorthogonal systems and the Nelder–Mead simplex method. In Section III we give a list of possible applications in signal processing. We provide a short reference for the available programs and commands in Section IV, the GUI is demonstrated in Section V. Finally we show two examples for the applications of these tools in the case of real world signals in Section VI.

II. MATHEMATICAL BACKGROUND

In this Section we outline the mathematical concepts behind our toolbox.

Let us denote the set of complex numbers by $\mathbb{C}$, the open unit disc by $\mathbb{D} := \{\, z \in \mathbb{C} : |z| < 1 \,\}$, and the unit circle (or torus) by $\mathbb{T} := \{\, z \in \mathbb{C} : |z| = 1 \,\}$. Furthermore let $\mathbb{N} := \{\, 1, 2, 3, \dots \,\}$.

A. Blaschke functions and products

In this construction the Blaschke functions play an important role. Blaschke functions are complex valued functions of one complex variable, depending on the parameter $a \in \mathbb{D}$. They are defined as follows:

$$B_a(z) := \frac{z - a}{1 - \overline{a}z} \qquad (z \in \mathbb{C} \setminus \{\, 1/\overline{a} \,\}).$$

Blaschke functions have many interesting properties. They are bijections of both $\mathbb{D}$ and $\mathbb{T}$. Furthermore they can be interpreted as the congruencies of the hyperbolic plane in the Poincaré disc model.

Blaschke products are finite products of Blaschke functions. For given $1 < m \in \mathbb{N}$ and $a_1, \ldots, a_m \in \mathbb{D}$ the Blaschke product of order m is defined as follows:

$$B_{a_1,\ldots,a_m} := \prod_{j=1}^{m} B_{a_j}(z).$$

A Blaschke product of order m is an m-fold map on $\mathbb{T}$, i.e. for all $w \in \mathbb{T}$ there exist exactly m values, $z_1, \ldots, z_m \in \mathbb{T}$, that are mapped to w.

The Blaschke functions appear (in connection with the below) when we need to calculate an appropriate discretization on $\mathbb{T}$, and also in the product form of the orthogonal Malmquist–Takenaka functions.

B. Basic rational functions

The elementary building blocks in the toolbox are the complex rational functions of the form

$$r_{a,k}(z) = \frac{1}{(1-\overline{a}z)^k},$$

with $a \in \mathbb{D}, k \in \mathbb{N}$. The parameter a is sometimes referred to as an *inverse pole* (because $1/\overline{a}$ is a pole in the standard sence), r_k is said to be a *basic function of order k*. Using a terminology similar to the trigonometric case, the value $k = 1$ corresponds to the *fundamental tone* and $k > 1$ the *overtones*.

Usually we take the restriction of the functions on the unit circle, which we identify with the $[-\pi, \pi)$ real interval, in our toolbox. To this order we apply the map $t \mapsto e^{it} \in \mathbb{T}$. Then the real part and the imaginary part can both be used for signal processing purposes.

Applying this process to basic rational functions the generated real valued functions shows similarities to certain types of signals. For instance they fit very well to different parts of an ECG signal. We note that this process is very adaptive.

The linear combinations of the $r_{a,k}$ functions can be applied to more general signals. They are of the form

$$\sum_{k=1}^{n} c_k r_{a_k,k} \qquad (c_k \in \mathbb{C}).$$

In our model a signal will be represented as such a linear combination. In order to do that we should find proper inverse poles and coefficients. In the following subsections we will discuss our approach (also implemented in the toolbox) to these problems.

We also mention that in some cases the so-called 'modified' basic rational functions are more appropriate. They are of the form

$$\frac{z^{k-1}}{(1-\overline{a}z)^k} \qquad (a \in \mathbb{D}, \ k \in \mathbb{N}).$$

They span the same subspace of the functions analytic on $\overline{\mathbb{D}}$ as the corresponding basic rational functions.

C. Malmquist–Takenaka systems

Both the basic rational functions and their modified versions form a linearly independent set of functions. An orthogonal system however would be more useful in applications, since the coefficients are easy to calculate in this case.

The Malmquist–Takenaka (MT) systems are orthogonal systems of functions. We generate these systems by the Gram–Schmidt orthogonalization applied to a finite set or to a sequence of basic rational functions. The corresponding scalar product used on $\mathbb{T}$ is:

$$\langle F, G \rangle = \frac{1}{2\pi} \int_{-\pi}^{\pi} F(e^{it})\overline{G}(e^{it}) \, \mathrm{d}t \qquad (F, G \in H^2(\mathbb{D})). \quad (1)$$

Naturally we use a discrete approximation of this integral in the toolbox.

A handy property of the MT systems is that their elements can be expressed as Blaschke products. Namely, taking the basic functions for a given $m \in \mathbb{N}$ and $a_1, \ldots, a_m \in \mathbb{D}$ the orthogonalized MT system can be written as:

$$\Phi_k(z) = \frac{\sqrt{1-|a_k|^2}}{1-\overline{a_k}z} \prod_{j=1}^{k-1} B_{a_j}(z),$$

with $1 \le k \le m$.

D. Discretization

In practice the discrete orthogonal systems are very useful. It is known that in the trigonometric case the discrete system is generated by uniform discretization of $\mathbb{T}$. For the MT system we had to find a proper non-uniform sampling of $\mathbb{T}$ and a weight function in the discrete scalar product to obtain a discrete orthogonal system.

In the definition of the set of discrete points, again, Blaschke products play an important role. In this respect the key fact is that B_a is a $\mathbb{T} \to \mathbb{T}$ bijection. This led us to the definition of the *argument function* associated to the Blaschke function B_a $(a \in \mathbb{D})$:

$$\beta_a : [-\pi, \pi) \to \mathbb{R} \qquad \beta_a(t) := \arg B_a(e^{it}).$$

In calculations it is useful to have an explicit form for β_a

$$\beta_a(t) = (\delta + \varphi) + 2\arctan\left(\frac{1+r}{1-r}\tan\frac{t-\varphi}{2}\right),$$

where $a = re^{i\varphi}$.

One can think of $\delta \in [-\pi, \pi)$ as a correction term: we can make sure that β_a will map the interval $[-\pi, \pi)$ onto itself. The function β_a is strictly increasing and invertible.

For Blaschke products we define the argument function in the following way:

$$\beta_{a_1,\ldots,a_m}(t) := \frac{1}{m}\arg\prod_{j=1}^{m} B_{a_j}(z) = \frac{1}{m}\sum_{j=1}^{m}\beta_{a_j}(t).$$

The $1/m$ factor is applied to maintain the $[-\pi, \pi)$ bijection property.

Let us consider the set of $N \in \mathbb{N}$ equidistant points

$$D_0^N := \left\{ -\pi + k \cdot \frac{2\pi}{N} : 0 \le k \le N-1 \right\} \subset [-\pi, \pi).$$

Then for given $1 \leq m \in \mathbb{N}$, and $a_1, \ldots, a_m \in \mathbb{D}$ parameters the set

$$D_{a_1,\ldots,a_m}^N := \left\{ \beta_{a_1,\ldots,a_m}^{-1}(t) \, : \, t \in D_0^N \right\} \subset [-\pi, \pi) \quad (2)$$

is a *non-equidistant N point discretization* of the interval $[-\pi, \pi)$.

We note that when $m = N$ the Malmquist–Takenaka system is also orthogonal with respect to the discrete scalar product on the above defined discretization points:

$$[F, G]_N := \sum_{t \in D_{a_1,\ldots,a_m}^N} F(e^{it})\overline{G}(e^{it})\rho_N(e^{it}), \quad (3)$$

with the weight function $\rho_N(e^{it}) = 1/\beta'_{a_1,\ldots,a_m}(t)$.

In the case of the discrete complex MT system real valued functions have to be extended with its imaginary parts before the MT expansions has been calculated. We should use the discrete real MT systems if we want to compute the MT coefficients directly. Now, let insert a zero pole into the pole vector $\mathbf{a} = \{0, a_1, \ldots, a_m\}$ and consider the following discretization

$$DR_0^N = \left\{ -\pi + k \cdot \frac{\pi}{N} \, : \, 0 \leq k \leq 2N - 2 \right\} \in [-\pi \; \pi) \,.$$

Then for a given pole vector we can define the discretization set on the analogy of (2):

$$DR_{0,a_1,\ldots,a_m}^N = \left\{ \beta_{0,a_1,\ldots,a_m}^{-1}(t) \, : \, t \in DR_0^N \right\} \in [-\pi \; \pi) \,.$$

The real valued MT system that corresponds to the pole vector $\mathbf{a}$ is defined as follows

$$U_0 = 1, \; V_0 = 0, \; U_n = \mathrm{Re}(\Phi_n), \; V_n = \mathrm{Im}(\Phi_n) \; (n \in \mathbb{N}) \,.$$

It can be proved that in the case of $m = N - 1$ the real MT system $(U_n, V_n, 0 \leq n \leq m)$ is also orthonormed system with respect to the discrete real scalar product

$$[F, G]_N^R = \sum_{t \in DR_{0,a_1,\ldots,a_m}^N} F(e^{it})G(e^{it})\rho_N^R(e^{it}) \,,$$

where ρ_N^R denotes the discrete real weight function

$$\rho_N^R(e^{it}) = 1/(2\,\mathrm{Re}\,\beta'_{0,a_1,\ldots,a_m}(t) - 1) \,.$$

E. Biorthogonal systems

As it was mentioned above, signals can be easily represented by using MT systems. We only need to calculate the coefficients of the MT expansions by computing continuous or the discrete scalar products. However in some special cases it can be beneficial to work with the original rational base functions instead of the orthonormal MT systems. For instance, this is the case of system identification when a partial fraction representation of the transfer function is taken, and the poles should be determined [7]. Furthermore, in contrast with the MT systems (Fig. 1), the biorthogonal systems (Fig. 2) are independent from the order of the poles. We take $n + 1$ different poles $a_0, \ldots, a_n$ with multiplicities $m_0, \ldots, m_n$ and the corresponding base functions

$$\phi_{k,i}(z) = \frac{z^{i-1}}{(1 - \overline{a}_k z)^i} \qquad (k = 0, \ldots, n, \; i = 1, \ldots, m_k) \,.$$

(a) Coefficients.

(b) Real part of the MT basic functions.

(c) Coefficients.

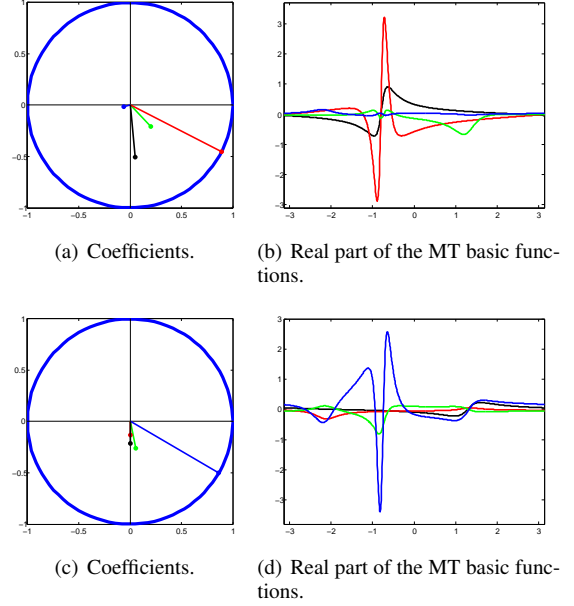(d) Real part of the MT basic functions.

Fig. 1. MT system defined by the same poles but in different order.

For the definition of the biorthogonal system we will need the following functions

$$\Omega_{\ell n}(z) = \frac{1}{(1 - \overline{a}_\ell z)^{m_\ell}} \prod_{i=0, i \neq \ell}^n B_{a_i}^{m_i}(z) \,,$$

$$\omega_{\ell n}(z) = \frac{\Omega_{\ell n}(a_\ell)}{\Omega_{\ell n}(z)}$$

where $(0 \leq \ell \leq n)$.

By Theorem 1 in [4] the functions

$$\Psi_{\ell,j}(z) = \frac{\Omega_{\ell n}(z)(z - a_\ell)^{j-1}}{\Omega_{\ell n}(a_\ell)} \sum_{s=0}^{m_\ell - j} \frac{\omega_{\ell n}^{(s)}(a_\ell)}{s!}(z - a_\ell)^s$$

$(0 \leq \ell \leq n, \; 1 \leq j \leq m_\ell)$ are biorthogonal to $\phi_{k,i}$ with respect to the scalar product defined by (1). As a result the biorthogonal expansion of an $f \in \Re$ function can be easily calculated as follows

$$P_N f = \sum_{k=0}^n \sum_{i=1}^{m_k} \langle f, \Psi_{ki} \rangle \phi_{ki} \,,$$

where $N = m_0 + m_1 + \cdots + m_n$.

Moreover it was shown in Theorem 2 in [4] that the systems $\phi_{k,i}$ and $\Psi_{\ell,j}$ are also biorthogonal with respect to the discrete scalar product defined by (3),

$$[\Psi_{\ell r}, \phi_{ks}]_N = \delta_{k\ell}\delta_{rs} \,,$$

where $(1 \leq r \leq m_\ell, \; 1 \leq s \leq m_k, \; 0 \leq k, \ell \leq n)$.

F. Optimization method

If the poles and the multiplicities, and so the basic rational functions are given then the orthogonal and the biorthogonal expansion can be calculated as described above. The next issue is the choice of the best possible poles and multiplicities.
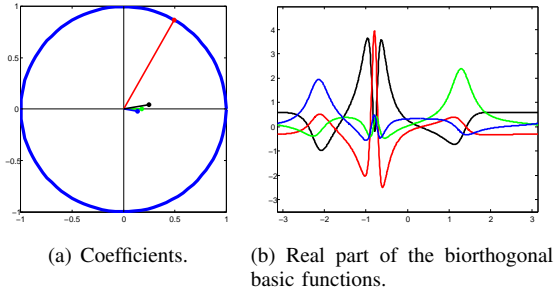
(a) Coefficients.

(b) Real part of the biorthogonal basic functions.

Fig. 2. Biorthogonal system.

(a) Projection onto $\Re_N$.

(b) Approximation error.

Fig. 4. Approximating real signals by taking lead 3 of record *s0301lre* from ptbdb database of Physionet [9].
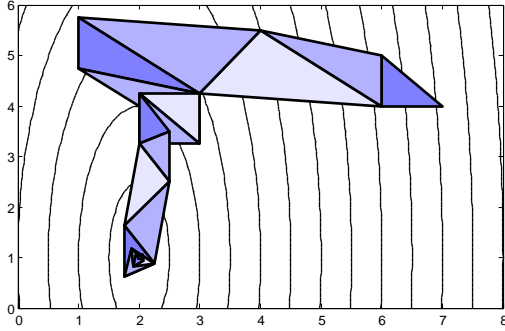

Fig. 3. The steps of the Nelder–Mead simplex algorithm minimizing a quadratic function of two variables.

In other words we want to minimize the distance of the subspace spanned by the basic rational functions with the given parameters from the original signal. For the solution of this optimization problem we have implemented the Nelder–Mead simplex method [8].

This algorithm is an iterative method for the minimization of an arbitrary real valued function of several variables. It is widely used in natural and engineering sciences and it is also mathematically studied. Its basic idea is a step-by-step improvement by means of geometric transformations of a simplex (i.e. in 2 dimensions a triangle). For a detailed description we refer to the original article [8], and for a recent application [6].

Fig. 3 shows a simple example for the application of the Nelder–Mead method.

An additional difficulty is that the original Nelder–Mead method is defined in $\mathbb{R}^n$, but in our case the parameters must stay within $\mathbb{D}$. Therefore we used a $\mathbb{R}^n \to \mathbb{D}$ bijective map to adjust the algorithm for our case.

We note that different versions of this algorithm have been implemented according to the function systems, i.e. discrete, continuous, complex, real.

III. Application in signal processing

We provide procedures to handle discrete datasets and signals. If the discrete versions of the systems are used then the values of the signal must be evaluated at non-equidistant discretization points. It means that we have to resample the signal as well. We have implemented this method by using linear interpolation between uniform sampling points.
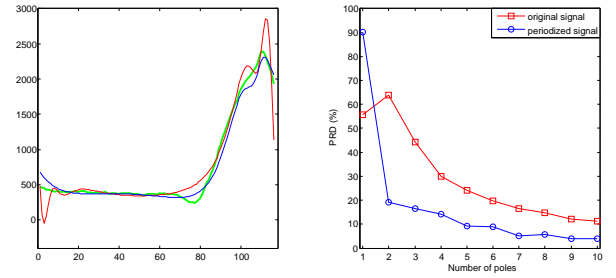
In order to improve the accuracy of the approximation we had to transform the signals to 2π periodic functions. In particular we had to make sure that the values at the end points are equal. In our implementation we wanted to avoid using such filters that distort the original signal. The preprocessing steps were carried out as follows. First we applied Savitzky–Golay FIR smoothing filter by using second order polynomials and width frame size equal to 11 samples. We used this smoothed data to predict the sign of the slopes of the original signal. Then the signal was extended at the two endpoints by keeping the slopes and used vertical shift. Then a Tukey window was applied to generate a 2π periodic signal. The parameters of the window were set so that the constant one part of the window corresponded to the original signal and the transition part must be equal with the length of extension. Fig. 4. shows the difference between the approximations by using the original and the periodized signals. The effect of jump discontinuity at the endpoints can be seen as the red colored function starts to oscillate near to the ends of the interval.

IV. Function reference

In this section we introduce the main parts and the programs of the RAIT toolbox. Our functions can be divided into 6 classes which are enumerated in Table I.

We have three types of rational function systems, namely the biorthogonal, the basic rational and the MT systems. Three functions were implemented for all types of these systems to compute the values, the coefficients and the projection onto $\Re_N$. The names of the functions include the name and the type of the system and the operation that we want to execute. For instance, if we want to calculate the coefficients of a function with respect to the discrete complex MT system, then the abbreviation `mtdr_coeffs` should be used. We introduce the parameterization of these procedures along with the continuous MT system. All functions in the first three classes can be called the same way.

- `mt_system(len,poles)`. Calculates the values of the Malmquist–Takenaka system defined by the `poles` vector at `len` number of uniformly distributed points in the interval $[-\pi, \pi]$.
- `mt_coeffs(v,poles)`. Computes the Fourier coefficients of `v` with respect to the MT system defined by `poles`.

- `mt_generate(len,poles,coeffs)`. Generates a function in the space spanned by the MT system.

The Blaschke class includes the procedures that compute and visualize the values of the Blaschke functions. Additionally, the derivatives and the inverse of the argument function also belong to this directory. Let us consider the most important procedures of this class:

- `arg_fun(a,t)`. Calculate the values of the argument function of a Blaschke product at `t` defined by the pole vector `a`.
- `arg_inv(a,b,eps)`. Compute the inverse images of the argument function at the elements of `b`. In this case, the poles of the Blaschke product is defined by `a` and the accuracy is determined by the variable `eps`.
- `blaschkes_img(image,a,show)`. Transforms an `image` by applying the Blaschke function defined by poles `a`. In addition, the absolute values and the arguments of the projection are also calculated. E.g. Fig. 5 was generated by this procedure.

The problem of calculating the projection onto $\Re_N$ is twofold. On one hand, we need to find the best poles for the selected rational system to minimize the approximation error. On the other, hand we need to calculate the projection. The latter can be easily computed by using functions of the first three classes. The former can be solved by applying different types of simplex algorithms. We have implemented this method for the five types of rational systems so we can choose the one which is the best suited to the given problem.

- `simplex_mt(f,mul,period,init,show,eps)`. Gives the poles of the continuous MT system that best fit the approximation of the function `f`. The parameters `mul` and `period` determine the multiplicities and the periodicity of the pole vector. Initial values are defined by the `init` variable. Furthermore, the predicted poles can be displayed at each steps by setting true the value of the `show` parameter. We can control the accuracy of the process through the `eps` variable. All types of simplex algorithm are called consistently by using the right rational system abbreviation, such as `mtdc`, `mtdr`, `biort` etc.
- `coords2params(k)`. It is an important function that maps coordinates in $\mathbb{R}^2$ to parameters in $\mathbb{D}$. The (x,y) coordinates are given sequentially in the vector `k`.

There are several other procedures that implement operators in $H^2(\mathbb{D})$, perform conversations between systems, visualize rational functions etc. We present some of the most important methods below, but additional examples can be also found in Section VI.

- `addimag(v)`. Calculates the imaginary part of the function `v` using FFT. It should be used to compute the expansion of `v` with respect to the complex versions of the systems, such as `mt`, `mtdc`, `biort` etc.
- `coeff_conv(l,poles,coeffs,base1,base2)`. It converts the coefficients `coeffs` between the continuous systems `base1` and `base2`. Parameters `l` and `poles` define both of the systems. Furthermore, a similar function called `coeffd_conv` is also available

TABLE I
FUNCTION REFERENCE

Biort_sys	Rat_sys	MT_sys
biort_system	lf_system	mt_system
biort_coeffs	lf_generate	mt_coeffs
biort_generate	mlf_system	mt_generate
biortdc_system	mlf_coeffs	mtdc_system
biortdc_coeffs	mlf_generate	mtdc_coeffs
biortdc_generate	mlfdc_system	mtdc_generate
	mlfdc_coeffs	mtdr_system
	mlfdc_generate	mtdr_coeffs
		mtdr_generate

Blaschke	Simplex	Other
arg_der	simplex_mt	addimag
arg_fun	simplex_mtdc	bisection_order
arg_inv	simplex_mtdr	coeff_conv
arg_inv_anim	simplex_biort	coeffd_conv
argdr_inv	simplex_biortdc	discretize_dc
blaschkes	coords2params	discretize_dr
blaschkes_img	coords2params_all	dotdc
	multiply_poles	dotdr
	periodize_poles	kernel
		multiplicity
		rshow
		subsample
		periodize

to perform conversations between the discrete versions of the systems.

- `discretize_dc(poles,eps)`. Computes the non-equidistant complex discretization on the unit disc that refers to the given `poles`. The accuracy of the method can be set by the variable `eps`.

V. MATLAB GUI

Our aim was to construct a toolbox that is appropriate to a wide range of applications. Furthermore, we also want to help the users in understanding the effects of certain parameters of the systems. Accordingly two additional interactive MATLAB GUIs were implemented to demonstrate the role of poles, multiplicities etc.

The first GUI is called `blaschke_tool`. It can be used to visualize the connection between the position of the pole and the values of the argument function. It is also possible to display grayscale images that are projected onto the unit disc by Blaschke functions. E.g. Fig. 5(a) and Fig. 5(b) show that how the Blaschke functions transform the unit disc and how it depend from the poles. Furthermore, the absolute values and the arguments of the Blaschke function from Fig. 5(b) also can be seen on Fig. 5(c) and on Fig. 5(d).

Additional GUI called `malmquist_tool` were implemented to introduce and to describe the properties of the MT systems. We can determine the positions, the number and the multiplicities of the selected poles. The argument function is also displayed on the unit disc. Furthermore, all the members of the MT system can be visualized related to the defined poles. Additionally, not only the complex case, but the real type of MT expansions were implemented as well. It means that both real and complex discretization can be used for interpolation purposes. In addition, the results and the weight function of the two types of discrete scalar product can be
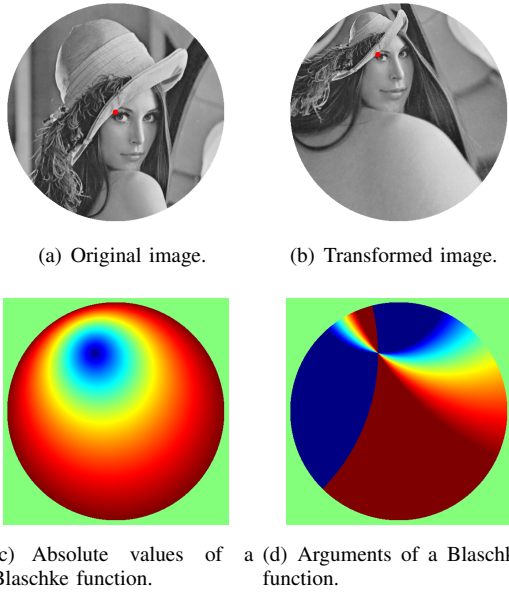
(a) Original image.

(b) Transformed image.

(c) Absolute values of a Blaschke function.

(d) Arguments of a Blaschke function.

Fig. 5. Properties of a Blaschke function.

(a) Poles and multiplicities.

(b) Argument function.

(c) Real part of Φ_6.
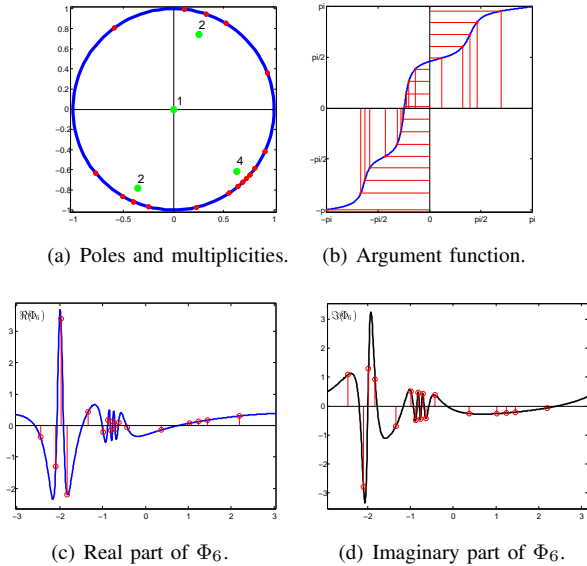

(d) Imaginary part of Φ_6.

Fig. 6. Visualization of the MT systems defined by an appropriate pole vector.

calculated. All of these functions can be animated interactively by moving the poles in the unit circle. Fig. 6 shows some illustrations generated by the toolbox. On Fig. 6(a) one can see the poles and the real discretization on the unit disc determined by the argument function on Fig. 6(b). It illustrates well the fact that the points of discretization tends to be more dense near the pole as it gets closer to the torus. The real and the complex part of the 6th member of the related MT system is showed by Fig. 6(c) and Fig. 6(d).

VI. EXAMPLES

In this section we present some MATLAB instructions from the RAIT toolbox. We will give examples for both continuous and discrete approximations as well.

First, we start with a real ECG curve from the ptbdb database of Physionet [9]. Second, the simplex algorithm has been performed to find the best poles for the continuous MT system. Finally, we compute the coefficients and the MT expansion. We note that the original signal was extended by its imaginary part. As we shall see later, this step can be skipped if we use the discrete real MT system. This is achieved by the following code:

```
>> s=periodize(ecg,0.1,0);
>> s_im=addimag(s);
>> m=[1 2 1];
>> pr=2;
>> init=zeros(1,6);
>> p=simplex_mt(s,m,pr,init,0,1e-6);
>> mp=periodize_poles(multiply_poles(p,m),pr);
>> mtco=mt_coeffs(s_im,mp);
>> s_mt=mt_generate(length(s),mp,mtco);
```

We can get the reconstructed signal by taking the real part of the variable s_mt. On Fig. 1(d) one can see the poles of the MT system and the approximation. We note that the 6.92% percentage root mean square difference (PRD) was gained by using only 8 complex coefficients and 3 complex poles. The original signal contains about 900 samples per heartbeats.

If we convert the coefficients into the basic rational function system, we can separate the main diagnostic waves of the ECG curve. Using this method to analyse ECG curves is a very obvious intention since these waves express real diagnostic features. On Fig. 9 we can see that the MT basic functions do not reflect any diagnostic properties in contrast with the basic rational functions. On Fig. 9(b) the P, T waves and the QRS complex was generated by sorting the projections of the ECG curve by using the basic functions which were defined by the same pole. The following code separates the main waves of the ECG curve:

```
>> len=length(s);
>> lfco=coeff_conv(len,mp,mtco,'mt','lf');
>> lfs=lf_system(len,mp);
>> P=real(lfco([1,5])*lfs([1,5],:));
>> QRS=real(lfco([2,3,6,7])*lfs([2,3,6,7],:));
>> T=real(lfco([4,8])*lfs([4,8],:));
```

Let us consider the interpolation process of the toolbox. The approximation will be carried out like the continuous case. Now we use a segment of a central venous pressure (CVP) signal. After the best poles were found then the discrete coefficients can be calculated directly. We note that, there is no need to extend the original signal with its imaginary part. On Fig. 10(a) and Fig. 10(b) one can see the reconstructed signal. In the former case the approximation was carried out on 6 poles. The latter figure was obtained by duplicating the number of the poles periodically. Hence the same instructions were executed by setting the variable pr to 2. For instance, Fig. 10 was obtained as follows:

```
>> s=periodize(cvp,0.1,0);
>> len=length(s);
>> m=[1 3 2];
>> pr=1;
>> init=[-0.6,-0.6,0.2,0.7,-0.4,-0.6];
>> p=simplex_mtdr(s,m,pr,init,0,1e-6);
>> mp=periodize_poles(multiply_poles(p,m),pr);
```
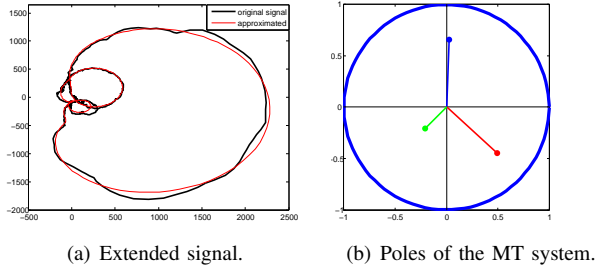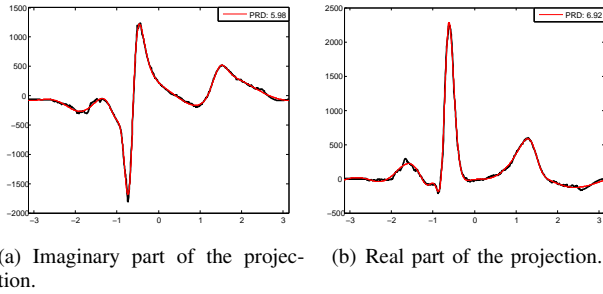
(a) Extended signal.　　　　(b) Poles of the MT system.

Fig. 7. Approximation steps.

(a) Imaginary part of the projection.　　(b) Real part of the projection.

Fig. 8. MT approximation of the record *s0306lrem* from ptbdb database of Physionet.

(a) MT basic functions.　　　(b) LF basic functions.
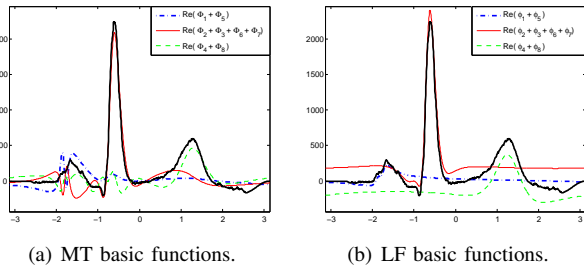
Fig. 9. Basic functions of MT and LF systems.

(a) Approximation of the signal by using 6 poles.　(b) Approximation of the signal by using 12 poles.
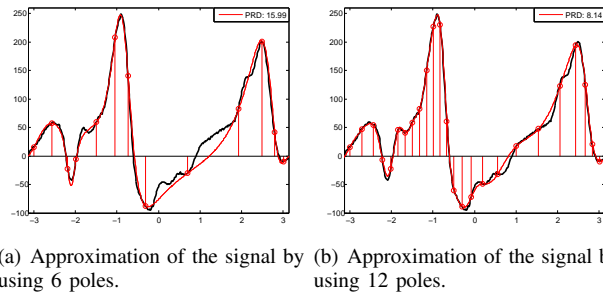
Fig. 10. Real MT approximation of the record *mgh001* from mghdb database of Physionet.

```
>> [cUk,cVk]=mtdr_coeffs(s,mp,1e-6);
>> s_mtdr=mtdr_generate(len,mp,cUk,cVk);
```

VII. EXPERIMENTS

In this section, we want to emphasize the efficiency of rational functions by compressing different types of signals. However, we have already given some examples for the practical applications of the systems, but comparsions have not

been presented yet. For this reason, we have performed experiments to compare the compression ratio and the accuracy of different algorithms in biomedical signal processing. We used the review [10] as a reference of the ECG signal compressing methods. To assess the quality of the results, we performed our tests on the two main classes of the ECG compression algorithms. Namely, we compared the performance of rational functions with wavelet and polynomial approximation based methods.

Furthermore, we want to use an objective measure of the level of compression and distortion rate. For this reason, synthesized ECG signals [13] were also used to test these methods. In this case, we can generate electrocardiograms with different diagnostical and geometrical features, but we can get the original signals in analytic forms as well. So, both compression and distortion rate can be measured by using different types of error measures, such as PRD and the so-called weighted diagnostic distortion (WDD) [14].

Wavelets are widely used in signal processing for not only compressing data, but for extracting different features as well. In our tests, a one dimensional discrete wavelet transform (DWT) was applied for ECG signals. It means that we generated 100 different synthetic ECG signals with normal diagnostic parameters by using the statistics of these features from [12]. This is a piecewise polynomial model with at most fifth order segments., therefore we used Daubechies wavelets having six vanishing moments (D6). These functions can be applied efficiently to compress our synthetic signals, because they can supress at least 5 order polynomials. The embedded MATLAB commands wdencmp and wpdencmp were used to perform the tests. The former one is a wavelet coefficient thresholding algorithm. The latter is an efficient implementation of the wavelet packets (WP) method. In contrast with the simple wavelet thresholding algorithms, the WP procedure is an adaptive transformation in the sense of the coefficient management. It means that both the approximation and the detailed coefficients are further decomposed at each level of the multiresulotion analysis. After this step, all the coefficients will be removed which have high entropy. The remaining terms are the 'best' optimal selection in the sense of Shannon entropy measure. We refer to [11] for details.

Now, let us consider the experiment. The aforementioned algorithms were applied to the 100 synthetic ECG signals. During the compression the thresholds were decreased in each step until the desired 5% PRD was reached. Finally, the number of coefficients and the WDD were encountered for each method. Fig. 11 shows our results. One can see that if the MT representation was used then generally we need less number of coefficients than in the D6 wavelet decomposition. In the former case, the diagnostic distortion is better. As it was excepted, the WP algorithm has outperformed the simple wavelet thresholding method. Now, the average number of coefficients is about 30. However it is slightly better, but it is still comparable with the performance of the rational functions. Furthermore, the WDD of the MT approximation is the best among these there methods.

In our second experiment we used an additional compression algorithm. It is a polynomial approximation method where
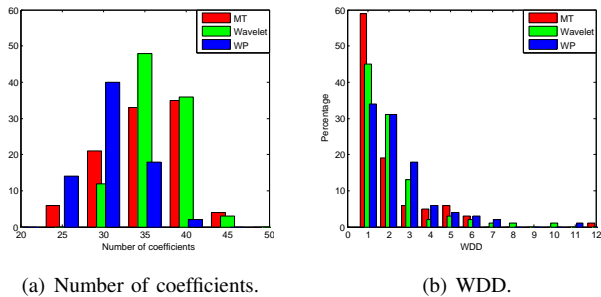
(a) Number of coefficients.   (b) WDD.

Fig. 11.   Statistics of the tested methods on 100 synthetic beats.

Fig. 12.   Statistics of the tested methods on 14 real ECG signals.

the base functions are b-splines. This is a simple procedure which starts with an initial spline approximation given by a large number of knots. The knots are then removed, one by one, until we reach a certain degree of distortion. The key point of this algorithm is the prediction of the error at each of the removed knots. By taking advantage of the fact that the beat to beat changes in ECG signals are generally not significant, the compression ratio of this algorithm can be further improved. But here we will use only the number of coefficients and knots to determine the related compression ratios for each beat. For proper algorithms see e.g. [16].

This test contains 14 real ECG signals of the mitdb database from Physionet [9]. Some preprocessing steps are necessary to deal with noise, but these algorithms are beyond the scope of this paper, we just enumerate the applied techniques, see references for further details. First of all, we segmented the signals into beats by taking into account the heart rate variability and the related statistics of the ECG signals from [12]. Then a simple baseline subtraction algorithm was used from [17] to deal with baseline wandering. Additionally the well known KLT transformation was also applied to filter out high frequency noises. Only the first few KLT basis function were computed by keeping 90% of the whole energy. The series of beats in a signal were aligned along with their R peaks. Furthermore, we calculated the KLT basis by using the SVD decomposition of 60 successive beats. The running SVD algorithm [15] were used to improve the speed of the preprocessing step. For further analysis see Chapters 5 and Appendix 9A from book [12].

In contrast with our previous experiment, now we fixed the number of coefficients that can be used by each of the tested algorithms. On one hand it means that the compression ratio is the same for all of these procedures. On the other hand the error of the approximation is different. The final results can be seen on Fig. 12. The average PRD was assigned to all of the related methods and records. As we expected, the b-spline approximation can not keep step with the other three methods. Otherwise, it is hard to distinguish the average PRDs in the latter case. So it is difficult to choose the algorithm that achieved the best scores. However, the red lines are almost always under the others. Which means that the rational function representation performed slightly better.

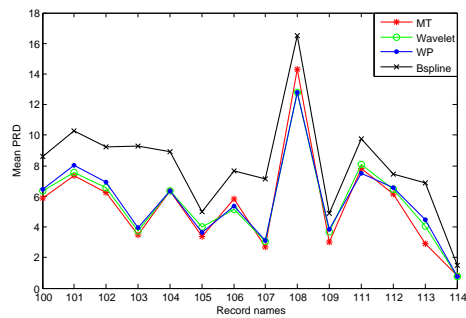Our primary goal was to demonstrate the feasibility and applicability of the method. The potential application area can be extended to other types of datas thats are not necessary to be biomdecical or physiological signals. Although the rational functions systems did perform better than wavelets and wavelet packets on some experiments, but we do not claim that it generally outperforms them. Further research is required to answer these questions.

VIII. Conclusion

A new MATLAB toolbox has been introduced which can be useful in a wide range of applications. However, we proposed some practical applications for signal processing. For instance, these systems are capable to represent different types of signals. Taking advantage of approximation and interpolation properties not only equidistant but non-uniform discretization can be used as well. Four types of signal representations are available by using different classes of rational function systems. It is worth to examine other properties of these representations, for instance the energy compaction, compression rate, sense of the coefficients and the poles etc. A related open question is that of the filtering property, whether or not we can use these expansions for denoising signals. Two MATLAB GUIs were implemented for educational purposes. However, we want to extend the toolbox with real biorthogonal expansions by analogy with `mtdr` functions. In addition, we are working on a GUI called `biort_tool` similar to `malmquist_tool` that demonstrate the properties of biorthogonal systems.

Further research is required on finding the best poles for a system. We used the Nelder–Mead algorithm to resolve this problem, but it can terminate in local optima. An alternative solution could be applying the simplex algorithm partially. Another way is to use hyperbolic geometry instead of eucledian. Unfortunately, it is hard to generalize the operations in higher dimension such as computing bisectors, intersections, reflections etc. However, Blaschke functions can be useful to resolve this problem. In the near future, we also want to implement a similar toolbox involving hyperbolic operations, simplex algorithm and different models such as Poincaré disk and Cayley–Klein model.

References

[1] A. Soumelidis, F. Schipp, J. Bokor, "Frequency domain representation of signals in rational orthogonal bases," in *Proc. 10th Mediterranean*

Conference on Control and Automation–MED 2002, Lisbon, Portugal, p. on CD.

[2] P. S. C. Heuberger, P. M. J. Van den Hof, B. Wahlberg, *Modelling and Identification with Rational Orthogonal Basis Functions*. London, CA: Springer-Verlag, 2005, pp. 61–102.

[3] S. Fridli, L. Lócsi, F. Schipp, "Rational function system in ECG processing," in *Proc. 13th Int. Conf. Computer Aided Systems Theory– EUROCAST 2011, Part I (eds. R. Moreno-Díaz et al.)*, LNCS 6927, pp. 88–95.

[4] S. Fridli, F. Schipp, "Biorthogonal systems to rational functions," *Annales Univ. Sci. Budapest., Sect. Comp*, vol. 35, no. 1, pp. 95–105, 2011.

[5] L. Lócsi, "Calculating non-equidistant discretizations generated by Blaschke products," *Acta Cybernetica*, vol. 20, no. 1, pp. 111–123, 2011.

[6] L. Lócsi, "Approximating poles of complex rational functions," *Acta Univ. Sapientiae, Mathematica*, vol. 1, no. 2, pp. 169–182, 2009.

[7] A. Soumelidis, M. Pap, F. Schipp, J. Bokor, "Frequency domain identification of partial fraction models," in *Proc. 15th IFAC World Congress*, Barcelona, Spain, 2002, p. on CD.

[8] J. A. Nelder, R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.

[9] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P .Ch. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C. K. Peng, H. E. Stanley. (2000). "PhysioBank, PhysioToolkit, and Physionet: Components of a New Research Resource for Complex Physiologic Signals." *Circulation* [Online]. *101(23)*. pp. 215-220. Available: http://circ.ahajournals.org/cgi/content/full/101/23/e215

[10] P. S. Addison, "Wavelet transforms and ECG: a review," *Computer Journal*, vol. 26, pp. 155–199, 2005.

[11] P. S. Addison, The Illustrated Wavelet Transform Handbook: *Introductory Theory and Applications in Science, Engineering, Medicine and Finance*, 1rd ed.   Naiper University, Edinburgh, UK, 2002.

[12] G.D. Clifford, F. Azuaje and P.E. McSharry, Advanced Methods and Tools for ECG Data Analysis, 1rd ed.   Artech House, London and Boston, 2008.

[13] P. Kovács, "ECG signal generator based on geometrical features," *Annales Univ. Sci. Budapest., Sect. Comp.*, vol. 37, pp. 247–260, 2012.

[14] Y. Zigel, A. Cohen and A. Katz, "The Weighted Diagnostic Distortion (WDD) Measure for ECG Signal Compression," *IEEE Transactions on Biomedical Engineering*, vol. 47, pp. 1422–1430, 2000.

[15] D. Chetverikov, A. Axt, "Approximation-free running SVD and its application to motion detection," *Pattern Recognition Letters*, vol. 31, pp. 891–897, 2010.

[16] M. Karczewicz, M. Gabbouj, "ECG data compression by spline approximation," *Signal Processing*, vol. 59, pp. 43–59, 1997.

[17] G.D. Clifford, "Collection of ECG processing algorithms: Filters, " *http://www.mit.edu/~gari/CODE/FILTERS*, Feb. 22, 2007 [Oct. 26, 2012].

Péter Kovács was born in Budapest, Hungary. He received the BSc degree in Computer Science in 2008, the MSc degree (with honors) in mathematical modelling from Eötvös L. University, Budapest, Hungary, in 2010. Currently he is writing his PhD thesis in signal and image processing at the Department of Numerical Analysis of Eötvös L. University. His current research interests include rational function approximation theory, ECG signal processing, such as filtering, compressing, feature extraction and beat classification.

Levente Lócsi has received his MSc degree (with honors) in Computer Science in 2008 at the Faculty of Informatics, Eötvös L. University, Budapest, Hungary. He is now an assistant lecturer at the Department of Numerical Analysis (same Faculty) since 2011, teaching numerical methods, analysis and Matlab programming. His PhD is in progress with the topic of applying complex rational function systems in signal processing (with special emphasis on ECG signals).