Информационные процессы, Том 4, № 1, 2004, стр. 76–116.

MATHEMATICAL MODELS, COMPUTATIONAL METHODS

Synthesis of Continuous-Signal and Digital Filters

M.G.Vitkov*, A.A.Vitkova**

**Moscow Technical University of Communications and Informatics, Moscow, Russia*
***Institute for Information Transmission Problems RAS, Moscow, Russia*
Received January, 16, 2004

Abstract—The software package for analysis and synthesis of continuous-signal and digital filters has been developed. This package consists of 25 "EXE" programs. There are the programs for continuous-signal filter network synthesis and for computation of their impulse response in this package. They allow to synthesize the uneven digital FIR–filters and to compute their frequency responses. There are also the programs for a recursive filter synthesis in this package. Besides, there are the programs for synthesis of the more complete filters, high order filters. All these programs use both our scientific results [1–10] and the results from [11–15]. The article is an application description of the developed software package. The theoretical questions of filtering, of wave digital filters, of z-transforms and the others are discussed in this description. It contains a lot of synthesis examples, too.

INTRODUCTION

Electric filter is the most important component of continuous-signal and digital systems for wire and radio communications. In that article the bases of theory and design for continuous-signal and digital frequency electric filters have been considered jointly. It is easy to see that the discussion of some main theoretical questions is not traditional. Here some new fundamental notions are introduced. For example, we introduce and use new notions for as well the modular for polynomials as the modular for rational functions. Furthermore, we more widely use the notion of the z–representation. We also thought over the notion for a digital signal frequency spectrum. It allowed to simplify the description for some main questions in that filter theory.

The fundamental purpose of this article is to help an engineer truly to formulate the conditions for frequency filter design and to make its synthesis by using our software package. Any add special knowledge are not required. Usual preparation of the designer either in electrical techniques or in communication techniques is adequate.

The article contains the authors' software package description which allows to make synthesis for very large filter class, including very high order filters. The software package, developed by us, is a complex of executed "EXE" files, realized in the TURBO BASIC environment by Borland International company. One can be used those "EXE" programs in the simplest PC, because they require only some hundred of Kb of memory. Yet, the very large filters for known group channels of multi-channel communication systems are easy calculated by using that software package.

Our scientific results [1-10] and the results from [11–15] are used in this article.

1. FILTERS

1.1. Transfer Functions to the Voltage

The filters are often designated the passed two-port networks, distinguishable by their transfer function $H(p)$ or $H(j\omega)$ as indicated in Figure 1-1. The transfer function is usually defined as a ratio of the response, U_2, to the excitation, E or U_1.

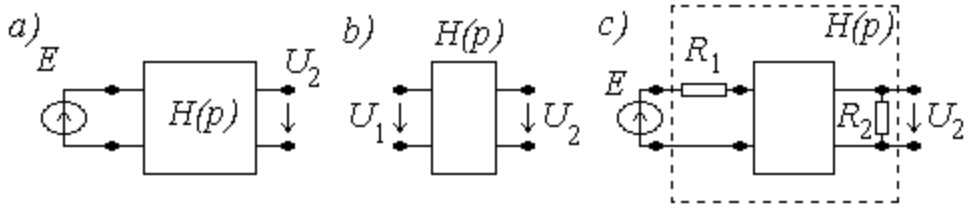

FIGURE 1-1 Passed two-port networks.

The more exactly, that transfer function

$$H(p) = \frac{U_2(p)}{E(p)}, \qquad \text{and} \qquad H(j\omega) = \frac{U_2(j\omega)}{E(j\omega)} \tag{1.1}$$

or

$$H(p) = \frac{U_2(p)}{U_1(p)}, \qquad \text{and} \qquad H(j\omega) = \frac{U_2(j\omega)}{U_1(j\omega)} \tag{1.2}$$

is denominated as the transfer function to the voltage.

The normalized transfer functions are often used for a filter synthesis

$$H(p) = \frac{kU_2(p)}{E(p)}, \qquad \text{and} \qquad H(j\omega) = \frac{kU_2(j\omega)}{E(j\omega)} \tag{1.3}$$

or

$$H(p) = \frac{kU_2(p)}{U_1(p)}, \qquad \text{and} \qquad H(j\omega) = \frac{kU_2(j\omega)}{U_1(j\omega)}. \tag{1.4}$$

Here, the k is a normalizing coefficient. It is chosen so that the greatest value of the transfer function on the $\omega-$axis was equal to 1.

It is often assumed that filters are doubly loaded by the resistors R_1 and R_2. In that case if we have the lossless filter, the normalizing coefficient is looking in the following way

$$k = \sqrt{\frac{4R_1}{R_2}}. \tag{1.5}$$

When we deal with the digital filters then the signals are considered as the signal sample unit sequences. For example, it will be the sequences for input and output voltages in the form of the following z–functions (see Figure 1-2)

$$\hat{E} = \cdots + E_{-3}z^3 + E_{-2}z^2 + E_{-1}z^1 + E_0 + E_1z^{-1} + E_2z^{-2} + E_3z^{-3} + \cdots,$$

$$\hat{U} = \cdots + U_{-3}z^3 + U_{-2}z^2 + U_{-1}z^1 + U_0 + U_1 z^{-1} + U_2 z^{-2} + U_3 z^{-3} + \cdots .$$

In that case the adequate transfer z–function (or system function) is often considered instead of an usual transfer function

$$\hat{H}(z) = \frac{\hat{U}}{\hat{E}} . \tag{1.6}$$

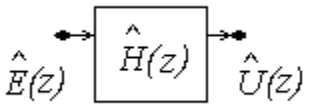

FIGURE 1-2 Transfer z–function of a digital filter.

The transfer z–function $\hat{H}(z)$ for a digital circuit can be always considered as a sequence beginning at the first instant

$$\hat{H}(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} + \cdots$$

or

$$\hat{H}(z) = h(0) + h(1)z^{-1} + h(2)z^{-2} + h(3)z^{-3} + \cdots . \tag{1.7}$$

The $\hat{H}(z)$ is related to the usual transfer functions $H(p)$ and $H(j\omega)$ by a standard z–transform

$$z = e^{pT} \qquad \text{and} \qquad z = e^{j\omega T} .$$

Here, the T is a sampling interval that is related to a sampling frequency F_D by a simple contrary ratio

$$T = \frac{1}{F_D} .$$

The usual transfer functions, in that case, are derived by the formulae

$$H(p) = \hat{H}(e^{pT}) \qquad \text{and} \qquad H(j\omega) = \hat{H}(e^{j\omega T}) .$$

They coincide with the transfer functions if the ideal digital signals take place. For example, if the input voltage will be

$$e(t) = T \sum_{i=-\infty}^{\infty} E_i \, \delta \, (t - iT) .$$

Attention

We are sorry, but sometimes (for example, see [11]) the transfer function is defined by the reciprocal:

$$H(j\omega) = \frac{E(j\omega)}{kU_2(j\omega)} .$$

We will never use that notation here. The authors usually anticipate the readers about it. Please, be attentive when you will read that literature.

1.2. Transfer Functions to the Current

Together with the transfer functions to the voltage, it may be also used the transfer functions to the current (see Figure 1-3):

$$H_I(j\omega) = \frac{k_I I_2(j\omega)}{I(j\omega)}\,.$$

In a case if we consider the filter which is doubly loaded with the resistors R_1 and R_2, the normalizing coefficient for the transfer functions to the current will be, obviously

$$k_I = \sqrt{\frac{4G_1}{G_2}}\,.$$

Here, the $G_1 = 1/R_1$ and the $G_2 = 1/R_2$ are the conduction of corresponding components of an electric filter circuit.

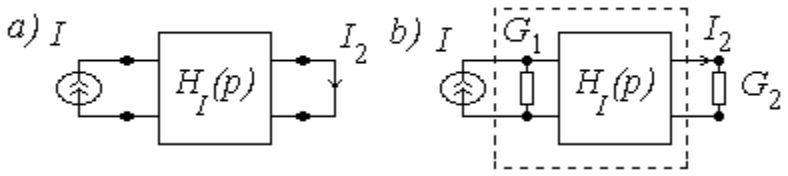

FIGURE 1-3 The transfer functions to the input current.

1.3. LOSS

For a numerical evaluation of the normalized transfer functions the logarithmic value, called as the loss A in decibels (dB), is often used. It has the form

$$A(\omega) = 10\lg\left|1/H^2(j\omega)\right| = 20\lg\left|1/H(j\omega)\right|. \qquad (1.8)$$

As the greatest value of the normalized transfer function is equal to 1 then the corresponding minimum value of A is equal to 0. An usual value of the passband loss is not large. Usually its maximum value is a few hundredth or tenth of 1 dB. But the greatest acceptable value of A is not large than 3 dB. The stopband loss is usually equal to a few decades (40 - 70 dB) and sometimes it reaches even 100 dB. We will denote the least stopband loss as A_S, and the largest acceptable passband loss as A_B. If there are different values of A in some places of both the passband and the stopband then we will denote them as $A_{B1}, A_{B2}, A_{B3}, \ldots$ or $A_{S1}, A_{S2}, A_{S3}, \ldots$. Sometimes the given acceptable values of A in both the passband and the stopband will be denoted as $A_{G1}, A_{G2}, A_{G3}, \ldots$.

2. SYNTHESIS OF DIGITAL FIR-FILTERS

2.1. Digital Filter Transfer Functions

If we deal with a digital filter then the voltages $u_T(t)$ and the currents $i_T(t)$ accord to the periodic transfer functions along the frequency axis. The sampling frequency F_D must be greater in two

times and more than the low processed digital signal frequency. For example, at such sampling frequency the voltage digital signal is described by the following z–function

$$\hat{U}_T(z) = \ldots + u(-2)z^2 + u(-1)z^1 + u(0) + u(1)z^{-1} + u(2)z^{-2} + \ldots .$$

The transfer z–function (or system function) is

$$\hat{H}(z) = \frac{\hat{H}_{2T}(z)}{\hat{E}_T(z)} = h(0) + h(1)z^{-1} + h(2)z^{-2} + \ldots . \tag{2.1}$$

That transfer function may be realized by the FIR-filter, i.e., by the finite impulse response (FIR) filter as it is indicated in Figure 2-1

$$\hat{H}(z) = \frac{\hat{H}_{2T}(z)}{\hat{E}_T(z)} = h(0) + h(1)z^{-1} + h(2)z^{-2} + \ldots + h(n-1)z^{-n+1} . \tag{2.2}$$

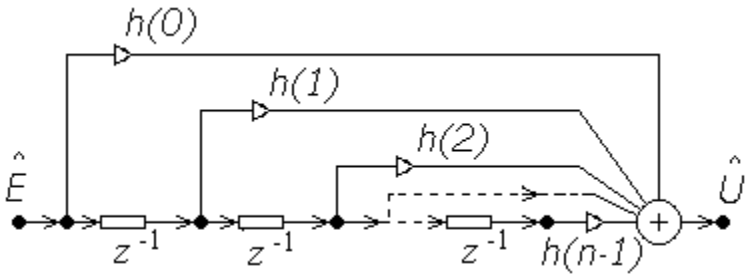

FIGURE 2-1 FIR – filter.

Here, z^{-1} is one-tact signal delay element.

In practice, the passband linear phase FIR-filters are usually used. It allows to avoid phase distortion. For this, the Nyquist's filters with the symmetric function $\hat{H}(z)$, in regard to a middle, are used. For example, in a case with odd filters having an even number of z^{-1}, $n-1 = 2N$, the following sample units of an impulse characteristic must be equal

$$h(N-k) = h(N+k), \qquad k = 1, 2, \ldots, N.$$

The network of that symmetric uneven FIR-filter is shown in Figure 2-2.

Then its transfer function will be

$$\hat{H}(z) = \sum_{i=0}^{N-1} h(i)\left(z^{-i} + z^{-n+1+i}\right) + h(N)z^{-N} . \tag{2.3}$$

Using the standard z–transform $z = e^{pT}$ and $z = e^{j\omega T}$ (T is a sampling interval), we find the following equation for the uneven digital filter amplitude response

$$H(p) = \sum_{i=0}^{N-1} h(i)\left(e^{-ipT} + e^{-(n-1-i)pT}\right) + h(N)e^{-pTN},$$

$$H(j\omega) = \sum_{i=0}^{N-1} h(i)\left(e^{-j\omega Ti} + e^{-j\omega T(n-1-i)}\right) + h(N)e^{-j\omega NT}.$$

Using Euler's formulae, it may be transformed the last equation to the following form

$$H(j\omega) = \left[h(N) + \sum_{l=1}^{N} 2h(N-l)\cos(\omega Tl)\right]e^{-j\omega NT}. \tag{2.4}$$

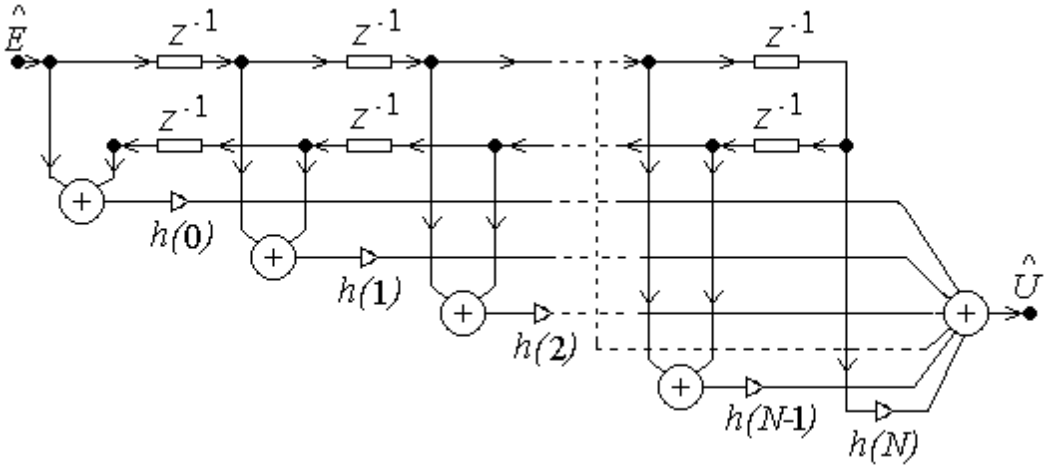

FIGURE 2-2 Odd FIR-filter network.

That equation may be made for N+1 points of $\omega = \omega_k$ (interpolation nodes on the frequency axis). In those points, the utmost interpolation requirements must be satisfied

$$\left|H(j\omega_k)\right| = 1 \tag{2.5}$$

inside the passband, and

$$\left|H(j\omega_k)\right| = 0 \tag{2.6}$$

inside the stopband, and

$$\left.\frac{d\left|H(j\omega)\right|}{d\omega}\right|_{\omega=\omega_k} = 0. \tag{2.7}$$

Here, $\left|H(j\omega)\right|$ is an algebraic expression in brackets of (2.4). If we will find the derivative of (2.4), the developed form of (2.7) will be

$$\frac{d\left|H(j\omega)\right|}{d\omega} = \left[\sum_{l=1}^{N} 2Tlh(N-l)\sin(\omega Tl)\right]. \tag{2.8}$$

Like this, the following equation of (2.5) is satisfied in each interpolation node ω_k which is inside the passband

$$\left[h(N) + \sum_{l=1}^{N} 2h(N-l)\cos(\omega_k Tl)\right] = 1. \tag{2.9}$$

In each interpolation node ω_k inside the stopband the equation (2.6) is satisfied, too

$$\Big[h(N) + \sum_{l=1}^{N} 2h(N-l)\cos(\omega_k T l)\Big] = 0. \tag{2.10}$$

Furthermore, in each mobile interpolation node ω_k inside both the passband and stopband the zero value of a derivative for frequency characteristic of (2.7) must be satisfied

$$\Big[\sum_{l=1}^{N} 2Tlh(N-l)\sin(\omega_k T l)\Big] = 0. \tag{2.11}$$

When $N+1$ points of ω_k have been selected, the last equation set of (2.9 – 2.11) allows to compute $N+1$ coefficients $h(k)$.

Here we will only discuss the odd order FIR-filters. The even FIR-filters have been discussed, for example, in [3]. If the order of an equation set is large then the even order can not be discussed at all. But it is necessary to mark that the number of equations, in that case, may be both even and odd. It depends on both the filtering specifications and the number of the selected mobile and immovable interpolation nodes. This number also depends on requirements pointed out.

2.2. Optimal Synthesis of Odd FIR-Filters

A definition of filter specification is formed in the "data\Kihapf.dat" file. The "Kihzdne.exe" program forms it. This program requires the following input data: 1) the number of approximation ranges No; 2) the sampling frequency $F_D = FD$ (it allows to calculate the sampling interval of that digital signal $T = 1/F_D = 2\pi/\omega_D$). Next, the program prompts to give the sequence of approximated levels for the given approximation ranges $T(k)$ (input 0 for the stopband, and 1 for the passband).

Next, the program prompts the following input data: 3) the number of frequencies N_{FG} in each approximation range. Minimal number of these frequencies is two. If the loss requirements in that range are more composite, N_{FG} may be more than 2. For example, at the FIR-filter synthesis in the "data\Kihapf.dat" file, the case is discussed in which the first approximation range is described by three frequencies: F_{G1}, F_{G2} and F_{G3} with two given limited values of A: A_{G1} and A_{G2}. Next, the program prompts the following input data: 4) the sequences of these frequencies, and 5) the values of the acceptable loss between them. After finishing of keyboard input, the program types the given frequency sequence F_{Gi}.

For example, consider the "data\Kihapf.dat" file containing the writing of filter specification.

"data\Kihapf.dat"

```
2
  1 0
  3 2
  0 .5 1.2 1.8 2.5
  .8 1.2 60
5
```

Here, the first digit we see 2. It is a number of approximation ranges for the digital filter. In next line, we see 1 and 0. In accordance with them, the first range is the passband and the second one is the stopband. Next, the number of frequencies given inside these ranges are 3 and 2. Three frequencies describe the first range and only two frequencies describe the second one. In next line, the values of these frequencies are: 0, 0.5, 1.2, 1.8 and 2.5 MHz, respectively. Next, the values of the loss are: 0.8 and 1.2. The value 0.8 is the loss by dB between the first frequency and second one. The value 1.2 is the loss by dB between second frequency and third one inside the first approximation range. The value 60 is the loss by dB between the limiting frequencies inside the next approximation range. There is a filter sampling frequency in the last line. Note that the frequency dimensionality is not shown. The file for writing the filter specification has also the same form if all the frequencies are equal to 0, 0.5, 1.2, 1.8, and 2.5 kHz or centuries kHz. Please, remember it!

The "Kihapre.exe" program produces an approximation of the given filter specification. At the beginning, the program prompts the file's name for writing the accounting parameters. Note that in practice, the order of adequate equation set usually exceeds some decades or centuries. Thence, the Gauss's method is used in that program. For example, the file's name will be "f". This name must not coincide with the other programs' and files' names. It is recommended to allocate "f" file in the main memory of PC. It is the best of all in the RAM disk.

Then the approximation program recalls the values of the following parameters: 1) the first value of parameter d_1=0.01. It allows to correct an interpolation nodes location for the first time; 2) the change of that parameter d_m=5 when the program will go to the new iteration in the following time, increased the value of $d = d_i$ by d_m ($(d_i = d_m d_{i-1})$); 3) the accuracy of equalization of the weighted errors either inside the different approximation ranges or inside their parts e_m=0.9. The given values of d_i, d_m, and e_m may be saved or changed. It is possible to change these parameters in the end of a computation, too. In many cases e_m may be at once selected as e_m=0.99 or e_m=0.999.

Next, the program prompts the number of free internal interpolation nodes $N(J)$ inside each given approximation range with a number J (J=1, 2, ...). Increased that number in any range, we increase the approximation accuracy of a require level or the transfer function levels: the approximation accuracy of 1 inside the approximation range of variable transmission and the approximation accuracy of 0 inside the range of variable non-transmission to the load. At the beginning of the calculation, we don't usually know these numbers. Thence, at first these numbers are selected at random. Then the calculations are repeated with more accurate values. The repetitions are continued till the more suitable values of $N(J)$ will be found.

Besides the free interpolation nodes whose location is changed at the computation of a problem, the program prompts the order of two interpolation nodes N_o and N_z in the points F_o and F_z where the frequency is equal to 0 and to $F_D/2$, respectively. In these points, the derivative by (2.11) is automatically equal to 0. Thence, here the non-zero value coincides with the order of (2.5) and (or) (2.6). For example, if these values are equal to 0, 0 (i.e., $N_o = 0$, $N_z = 0$) then it is assumed that the additional interpolation nodes are not in these points. When the value is interpolated only in the second additional point, where the frequency coincides with $F_D/2$, then the values of 0, 1 (i.e., $N_o = 0$, $N_z = 1$) are given. If the values of 1, 0 are given then the value is interpolated in the first point. At last, when these values are equal to 1, 1 then the program will take into account the interpolation in both additional points.

At first, the program allocates all the given nodes inside each approximation range onto equal space from each other, i.e., equidistantly. In that case the space from the edges of each range is selected equal to 1/4 of the adequate space or, if it is the first edge of the first range in which the

additional interpolation ($N_o = 1$) takes place, or it is the last range and the last space up to its edge, when the interpolation node is given ($N_z = 1$) inside the last range then the last spaces are equal to 1.

Then the equations (2.9)–(2.11) are solved, the coefficients $h(k)$ (k=0, 1, 2, ...) are found and the amplitude responses are computed.

Then with the initial value of the parameter d=d_1, a try for change of interpolation nodes location inside some approximation range is produced. The amplitude response is again computed. The next value of d=d_2=$d_1\, d_m$ is found. Next, all the procedures are repeated. The repetitions for given approximation range are continued till the equalization of approximation errors is reached by the best. At this, the improvement of the location of the interpolation nodes inside the given range is continued every time with the new value of d=d_i=$d_{i-1}\, d_m$.

After that, we go to the next approximation range and also improve the location of interpolation nodes in it. When that location will improve in all the approximation ranges then we will consider this iteration of an improvement as a finished one. The iteration process is continued till the equalization of the approximation errors $E(J)$ for all the ranges will be better than e_m in the specification, i.e., $E(J) \geq e_m$ in all approximation intervals. After finishing calculation, the results are saved in the "data\Kihkhg.dat" file.

During computation of every variant, at first, the program of approximation calculates the order of a digital filter and also lines in which the calculation is illustrated: the computation element number, the approximation range number, the iteration number, the value of D=d, which allowed to find that solution, the found values of the equalization errors $E(J)$, and the largest normalized approximation error $M(J)$ inside the range with the given number J. The normalized error is selected so that it equals to 1 when the greatest error coincides with the given one of A_G by dB. Thence, inside the stopband the normalized error is selected as the ratio of an usual error to $10^{-0.05A_G}$. But inside the passband, where 1 is approximated, an usual approximation error is divided by $|1 - 10^{-0.05A_G}|$. When all calculations are over then the program types the main reached approximation parameters and, first of all, all the normalized approximation errors inside all the ranges $M(J)$, and also all main approximation parameters display. All computed parameters are saved in the "data\Kihkhg.dat" file.

The found normalized errors $M(J)$ allow to perform an analysis of the values of $N(J)$, N_o and N_z for the next calculation of an approximation. Or they allow to agree with the got version. If the last values of $M(J)$ are less than 1 then the selected numbers of interpolation nodes are sufficient for that problem. If these values are greater than 1 then the approximation requirements are not satisfied. To satisfy the given requirements, one may try to revise the approximation. For that, the value of e_m must be approximate to 1, i.e., we need to give the equalization equals to 0.99 or 0.999. If it does not help then we need to increase the number of interpolation nodes inside the corresponding approximation ranges, i.e., we need to do more composite filter network. If the number of J is equal to 1 and N_o=0 then it may try to increase N_o=1. If the number J is the last one and if the corresponding approximation range reaches the half of the sampling frequency, but N_z=0 then it may try to increase N_z=1. If J=1 and we had N_o=1 then if we increase $N(1)$ by 1, we may try simultaneously to decrease N_o=0. Also it may try to change N_z=1 by N_z=0 if, at this, we increase by 1 the number of mobile interpolation nodes in that approximation range. The computation of approximation is repeated with the new values of the number of interpolation nodes.

If $M(J) << 1$ in the J-th approximation range then we have a large reserve on the required value of an error in it. But it does not need. In that case we can simplify the filter. For this we must decrease $N(J)$, or N_o=1, or N_z=1 absolutely so as in the case when we increased the filter order. Only now these changes can simplify the filter. Thence, the computations of approximation are again repeated till the normalized approximation errors $M(J)$ will be the nearest to 1, but in that case they are less than 1.

In practice, it is often need to compute the large filters. Thence, the computations are very long. For this in practice, a wish can emerge to interrupt the computation and to continue it in the future. Thence, at the computation the other working "data\Kihr.dat" file is formed. It allows to interrupt the calculation and then to continue it again by using the data earlier computed. To interrupt the computation of approximation it is need to put the button "↓". To continue the calculation it is need to run the program of approximation "Kihapre.exe" using 1 for the continuation of computation but not 0 for the beginning of it (see the beginning of that program).

2.3. FIR-Filter Frequency Characteristics

The saved data allow to compute frequency responses. For this, the "Kihafxe.exe" program has been developed by us.

The "Kihafxe.exe" program allows to compute the synthesized FIR-filter frequency characteristics onto an optional frequency interval. This interval contains frequencies from F=$F_D k$, where k is an integer number that is selected by the computer and it corresponds to the zero point on the frequency axis.

Usually, if we deal with the real filters we are interested in their frequency characteristics onto the interval from F=0 up to $F_D/2$. To plot that graph the"Kihafxe.exe" program is used. At the beginning of this program it is need to coincide the points of the initial frequency and zero one. That graph have been plotted by us in Figure 2-3.

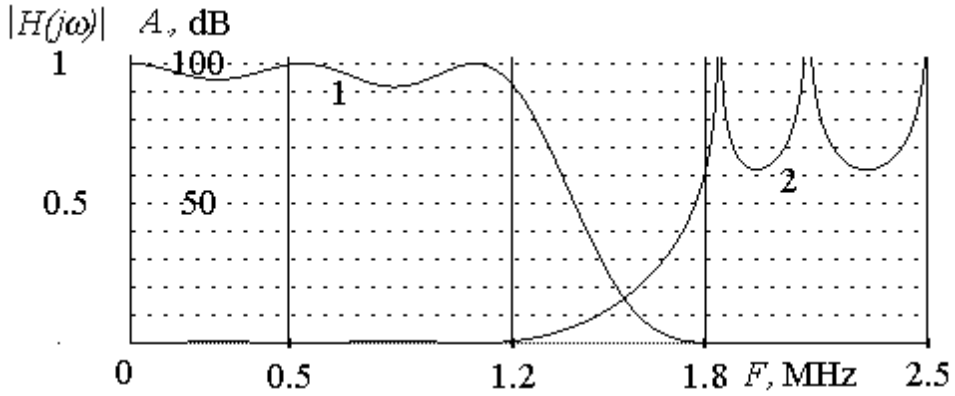

FIGURE 2-3 FIR–filter frequency responses.

The frequency characteristics of the real digital filters are periodically repeated with a period of ω_D along the frequency axis. To see these periodical characteristics it may change an location of the initial, zero and end point onto the frequency axis taken to account that, from the zero point up to the end, one semi-period of frequency response is arranged. It coincides with an interval from $F_D k$ up to $F_D/2$+$F_D k$ where k is the suitable integer.

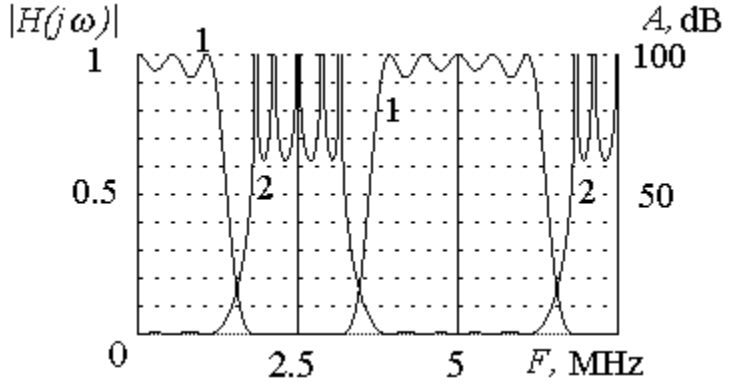

FIGURE 2-4 Periodically repeating frequency responses.

In Figures 2-3, and 2-4 the curve 1 shows the module of transfer function, and the curve 2 shows the loss (if an amplification is equal to 1). But in Figure 2-4 the initial and the first points of frequency are moved aside from each other by F_D and the first and the end points of frequency are moved aside from each other by $F_D/2$. Thence, the periodically repeating frequency response is seen very well in Figure 2-4.

In all programs for the calculation of frequency responses the vertical dimension of one cell, at unit amplification, corresponds to 10 dB. The total dimension, i.e., 10 vertical cells, corresponds to 100 dB. The same dimension corresponds to the value of 1 for the transfer function plotted in the relative units.

Some random value of amplification may be also given on the vertical axis. In that case the curve amplified by k_{db} times is plotted. The vertical sensitivity of one vertical cell will be less by k_{db} times than 10 dB.

For example, Figure 2-5 shows the curves of transmission of that FIR-filter at the amplification k_{db}=50.

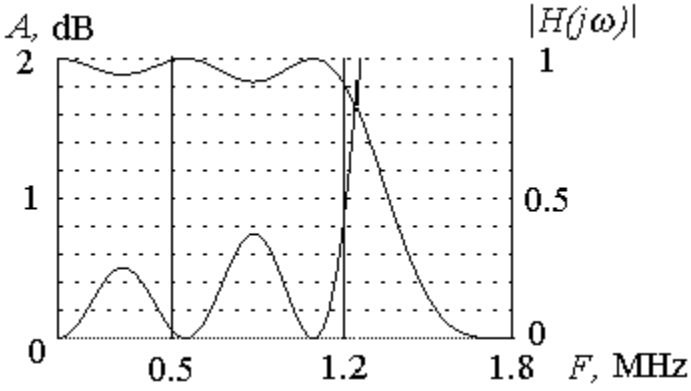

FIGURE 2-5 Frequency responses at amplification k_{db}=50.

The programs for the calculation of frequency responses of continuous-signal and digital recursive filters operate otherwise. In that case if we has given the amplification then both the curve of

frequency response with unit amplification and curve amplified by k_{db} times are typed together. The last curve is elevated up by 4 levels and has other vertical scale.

3. INTERPOLATION NODE CORRECTION

In the filter design process it is need many times to equalize the interpolation errors. In that case the equalization method [2] is used (see Figure 3-1).

In Figure 3-1 (a) the approximation error of the largest unit value inside the low first approximation range (i=1), for 2 mobile interpolation nodes ω_1 and ω_2 and 1 immovable node ω_o, is shown. In these nodes the approximation error is equal to 0 and in other nodes it changes continuously with three largest errors inside that range: M_1=M_{min}, M_2, and M_3=M_{max}. An inaccuracy of equalization of an approximation error inside that range is e=M_{min}/M_{max}. In Figure 3-1 (b) a diagram is shown where at first the points $\omega_{G1}, \omega_1, \omega_2$ and ω_{G2} are along horizontal and vertical axes. Here yet, the line with an inclination angle 45° is shown. It connects the point $(\omega_{G1}, \omega_{G1})$ with the point $(\omega_{G2}, \omega_{G2})$. The initial location of the movable interpolation nodes ω_1 and ω_2 is given onto it.

Using the maximal interpolation errors M_k we compute a broken line of errors in the nodes u_k (if u_0=0) with given value of d

$$u_k = u_{k-1} + \Big(1 + d\frac{M_k}{M_{max}}\Big)(\omega_k - \omega_{k-1}), \qquad k = 1, 2, \ldots, N_i + 1, \quad u_0 = \omega_{G1}.$$

Here, N_i is the common number of mobile interpolation nodes in the i-th approximation range. In that example (see Figure 3-1) N_i=2.

The coefficient of the renormalization for the approximation error will be

$$W_i = \frac{u_{N_i+1}}{\omega_{N_i+1}}.$$

It can help to do a modification of the corrected broken line of the approximation error inside the given range

$$u'_k = \frac{u_k}{W_i}, \qquad k = 1, 2, \ldots, N_i + 1,$$

that has got both the point $(\omega_{G1}, \omega_{G1})$ and the point $(\omega_{G2}, \omega_{G2})$ (see Figure 3-1(b)).

The new values of the mobile interpolation nodes in the i-th range ω'_k (k=1, 2, . . . , N_i) can be easy found from that corrected broken line with given value of d.

If ω_k is between u_l and u_{l+1} then the new value of ω'_k is calculated by

$$\omega'_k = \omega_l + \frac{(\omega_{l+1} - \omega_l)(\omega_k - u'_l)}{u'_{l+1} - u'_l}.$$

For the others approximation ranges the computation is produced by an analogy.

We will produce the computations in many others cases further, too.

In a case when a continuous-signal filter will be designed, we will also use this method of a correction of an interpolation nodes location.

4. CONTINUOUS-SIGNAL FILTERS

4.1. Loaded Reactance Network, Modulars

The continuous-signal filter as the long reactance network consisting of L, C, loaded with the terminations $R = R_1$ and R_2, is widely applied. In that case the common circuit for a signal

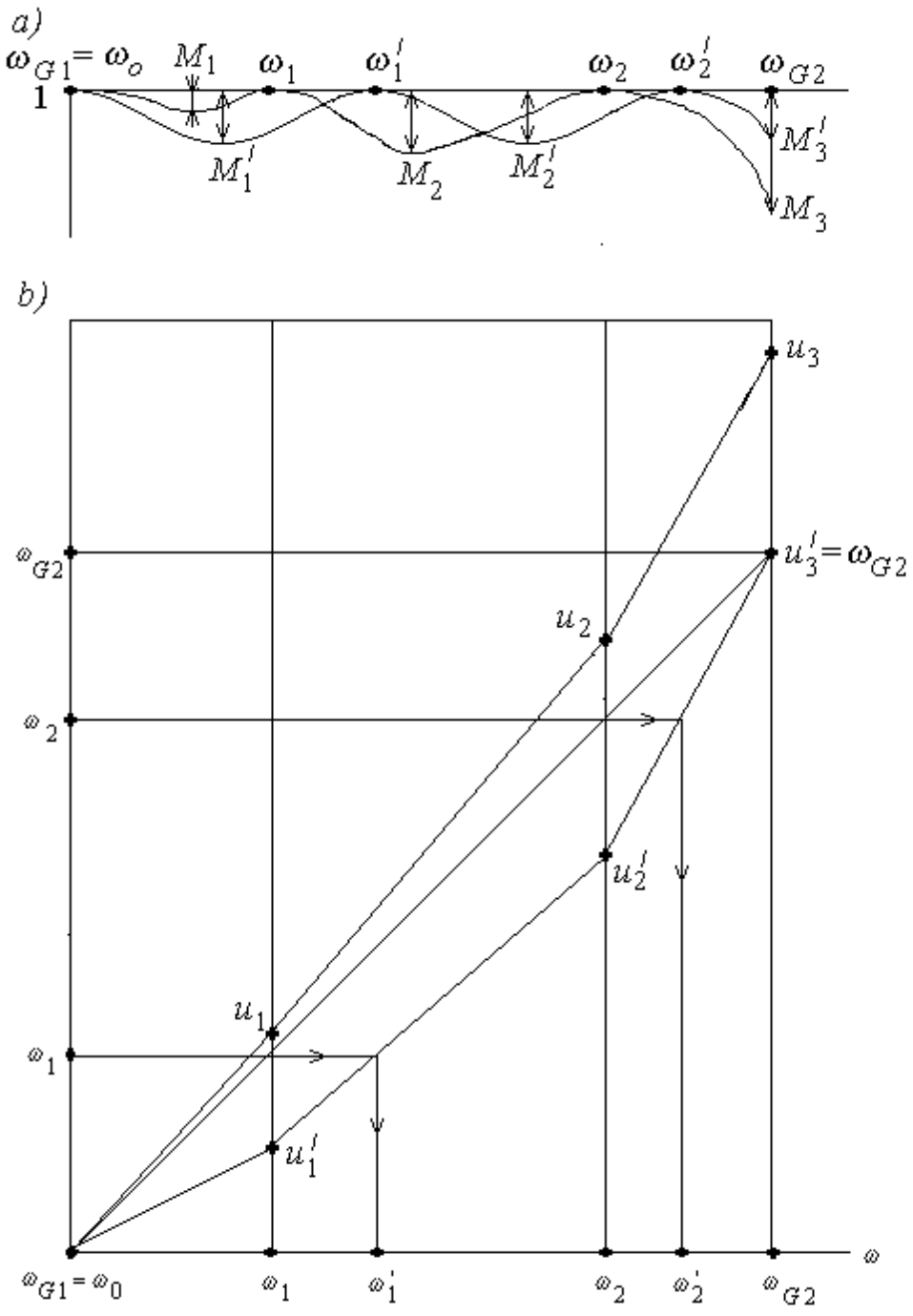

FIGURE 3-1 Diagram of interpolation error equalization method.

transmission coincides with the network in Figure 1-1 (c), in which the internal two-port is that reactance network. The transfer circuit is considered with the normalizing coefficient k, by (1.5). The normalized transfer function is usually computed for R=1 Ohm. In order to compute R_2 and that reactance circuit the Darlington's method is used [12, 13].

For the analysis and synthesis of filters the following functions are widely applied

$$F_{[2]} = F^{[2]} = F(p)F(-p).$$

Denote them as modulars or the second rang modulars for the functions $F(p)$. For the real axis p=$j\omega$ the modulars coincide with a modulus quadrate of the leading function

$$F_{[2]} = F^{[2]} = F(j\omega)F(-j\omega) = \left|F(j\omega)\right|^2.$$

For example, we denominate the functions

$$p_{[2]} = p^{[2]} = -p^2$$

as the modulars for the leading variable. They are equal to a quadrate of frequency on the real frequency axis

$$p_{[2]} = p^{[2]} = \omega^2 .$$

The transfer function modulars (i.e., transmission modulars $H_{[2]}$) are framed from the normalized transfer function

$$H_{[2]} = H^{[2]} = H_{[2]}(-p^2) = H^{[2]}(-p^2) = H(p)H(-p) = \left|H(p)\right|^2,$$

$$H_{[2]}\left(p_{[2]}\right) = H^{[2]}\left(p_{[2]}\right) = H(p)H(-p),$$

$$H_{[2]}\left(\omega^2\right) = H^{[2]}\left(\omega^2\right) = H(j\omega)H(-j\omega) = \left|H(j\omega)\right|^2.$$

It is usually for the majority filters, the transmission modular has the form

$$H_{[2]} = \frac{1}{1 + h\, p_{[2]}^{N_o} \dfrac{\prod\limits_{k=1}^{N_l}\left(p_{[2]} - p_k^{[2]}\right)^2}{\prod\limits_{0k=1}^{N_0}\left(p_{[2]} - p_{0k}^{[2]}\right)^2}}. \tag{4.1}$$

Here, h is a coefficient, N_o is an order of zero, N_l is the number of couples of the mobile interpolation nodes with a value of 1 in the passband of a signal to the termination R_2, and N_0 is the number of couples of the mobile interpolation nodes with a value of 0 in the stopband of one. Here, $p_k^{[2]} = \omega_k^2$ and $p_{0k}^{[2]} = \omega_{0k}^2$ are the modulars of interpolation frequencies of the values of 1 and 0 in the approximation ranges of two transmission values: 1 and 0.

It can be calculated the reflection modular according to the given modular of a transfer function

$$\rho_{[2]} = 1 - H_{[2]} . \tag{4.2}$$

If we know the transmission modulars and reflection modulars then we can find the functions themselves: the normalized transfer function $H(p)$, and reflection function $\rho(p)$. The last function allows also to calculate the input resistance of the two-port loaded with the resistor R_2

$$Z(p) = R_1 \frac{1 - \rho(p)}{1 + \rho(p)} . \tag{4.3}$$

It allows to calculate R_2 and overall reactance filter network.

4.2. Programs for Writing Filter Loss Specification

The “Anfzdne.exe” and “Cfzdne.exe” programs have been developed for design of both the continuous-signal filters and the continuous-signal prototype filters. The first program is used to design the continuous-signal filters. The second one is to design as the continuous-signal prototype filters as digital filters. Thence, it prompts the sampling frequency, too. The common specifications for these programs are saved in the “data\Apf.dat” file.

The additional data for the digital filters are in the end of the “data\Apf.dat” file. They are only used for synthesis of digital circuits: wave digital filters, and both canonical digital filters and cascade these.

For example, consider the “Apf.dat” specification file for the bandpass continuous-signal filter design.

“Apf.dat”

```
3
  0 1 0
  2 4 2
  0 .45 .55 .65 .9 1 1.1 10000
  60 .52 .26 .52 60
1
```

Here, we see 3 in the first line. It is a number of approximation ranges. In the second line 0, 1, 0 are the type of these ranges: the stopband, the passband, and the stopband.

In the third line 2, 4, 2 are the number of frequencies defining the filter specification. The values of these frequencies: 0, .45, .55, .65, .9, 1, 1.1, and 10000 rad/s are in the forth line. In a case if we consider a lowpass and highpass filter, the passband edge which is the most close to the stopband is usually accepted as 1 rad/s. In a case if we consider a bandpass filter, it is usually the highest passband frequency.

The values of the loss are given in the next line. Here, 60 dB is the loss between the first couple of frequencies (0 and 0.45 rad/s) in the first approximation range. In the second range, the values 0.52, 0.26 and again 0.52 dB is between the frequencies in this range: 0.55 and 0.65, 0.65 and 0.9, 0.9 and 1 rad/s. And at last, 60 dB is the loss between the frequencies 1.1 and 10000 rad/s in the third range.

The last character 1 is the frequency which is accepted as 1 at the computation of an continuous-signal filter.

4.3. Programs for Approximation

For solution of filter approximation problem the “Anaprame.exe” program has been developed. It applies the data of the “data\Apf.dat” file and prompts the number of moving frequencies $N(J)$ of approximation nodes inside all the approximation ranges with the different numbers of J. Furthermore, the program prompts the order N_o which defines behavior of the transfer function in zero. If the last coefficient is positive then the more its value the better the accuracy at which the transfer function is equal to 1 at ω=0. If this coefficient is negative then the more $|N_o|$ the better the accuracy at which the transfer function approximates 0 at ω=0. If a coefficient N_o is

equal to 0 then the transfer function is not equal neither 1 nor 0 at ω=0. At the beginning of a computation we do not usually know the values of N_o, $N(1)$, $N(2), \ldots$. In a case when we have a lowpass filter with uniform parameters, it is possible to accelerate this matching if we will use the approximate computations by the "Af_ne.exe" program. Though, this acceleration helps in the large little because the "Af_ne.exe" allows to estimate the order for only simplest Butterworth, and Chebyshev, and Cauer–Zolotarev filters. In more overall cases, the order of an approximation may be chosen by other way. If at the beginning we will arbitrarily give the values of the parameters N_o, $N(1)$, $N(2), \ldots$ then it may quickly to compute the approximation. As a result, the normalized reduced approximation error MR is delivered first of all. This error is less than 1 but it is closely 1 when the choice of approximation parameters is successful. Then the reduced errors $M(J)$ inside all the J approximation ranges are displayed. The most successful list of parameters N_o, $N(1)$, $N(2), \ldots$ will be one at which $MR \leq 1$, but it is closely 1, and besides: 1) when the values of reduced errors $M(J)$ inside all the passband are near, and 2) when the values of reduced errors $M(J)$ inside all the stopband are near, too. Usually the "Anaprame.exe" program allows easy to find that assembly of parameters.

4.4. Approximation Algorithm

The approximation algorithm in the "Anaprame.exe" program mainly coincides with one that was written here in Section 3. At the beginning the interpolation nodes are arranged inside their approximation ranges equidistantly. But inside the last highly long and even infinity approximation range, some difference takes place. Here at the beginning this long range is mapped onto a unit interval. Then the approximation nodes are allocated there. Even the change of an location of nodes is executed in transformed form. The results of equalization of the approximation errors are saved in the "data\Fap.dat" file. These results allow to plot the synthesized filter frequency responses using "Anfafxe.exe" program.

For example, we show the "Fap.dat" file with the results of computation of approximation for that bandpass filter.

"data\Fap.dat"

```
7,2,-2,.9915856668729779,14293442.33416819
.555053578705557,.5980558911963784,.6797379334822582
.7785860020797964,.8743305430950368,.9563224684662942
.9953186984368321,0,0
.4389709689454479,1.110589907375944,0
3
.5158743849414414,.257875953099166,.5158743849414414
```

Here, in the first line we see 7 and 2. They are the number of mobile interpolation nodes inside the passband range (inside the second approximation range of the bandpass filter) and inside the stopband ranges: inside the first and third ranges (one node is in each range). Further, we see the parameter $-2 = N_o$. It is the second degree of approximation of value 0 at zero frequency. The squares of the normalized maximal approximation errors inside both the passband and stopband are there also. The first of these values, that coincides with the largest normalized error inside the passband, must be less than 1 if the conditions of an approximation are satisfied. If its value is more than 1 then it may be tried to correct a computation of approximation. For this it must be input a

larger accuracy of an equalization of errors: e_m=.99, .999, .9999 and even 0.99999. If in that case the result does not satisfy the specification then it may be give the larger values of $N(J)$ or $|N_o|$.

Seven frequencies, in which 1 is interpolated, are in the second, third and fourth lines.

Two frequencies, at which the value of 0 for the transfer function modular (4.1) is interpolated, are in the fifth line.

In the sixth line we see the digit 3. It is the number of the maximal values of the passband loss. These 3 values themselves are in the seventh line.

The synthesized filter loss response is shown in Figure 4-1.

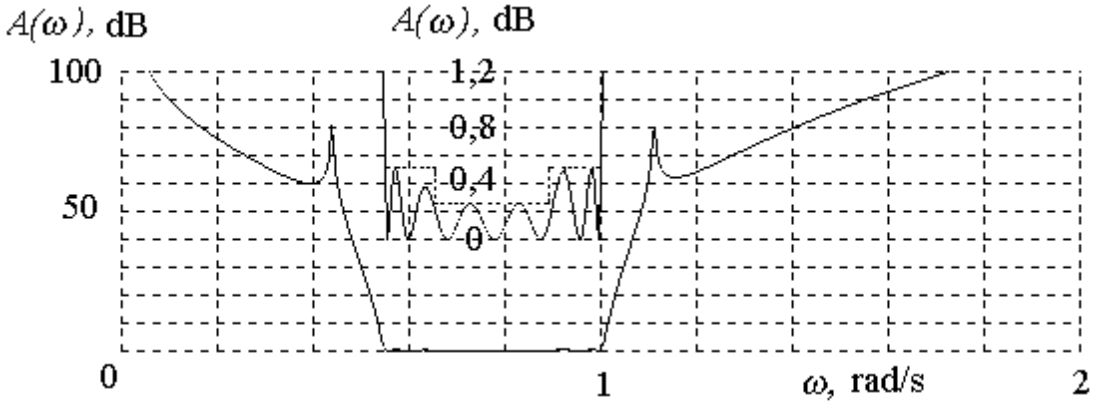

FIGURE 4-1 Frequency response of a bandpass filter.

4.5. Programs for Nature of an Continuous-signal Filter Specification

The “Anadwvte.exe” program serves to continue the continuous-signal filter synthesis. At the beginning, this program supports a passage from the modular parameters to the usual filter parameters. For this a modular equation of a denominator of a transfer function modular is made and solved by Berstou–Khichcoc’s method. If an usual solution does not get then G.C.Temes-H.J.Orchard’s method is used at suitable modification of an modular variable $p_{[2]}$. Usually, if in the case the program prompts $wB1$, $wB2$ then it is need to input: 1, 0.

The results of a computation are saved in the “data\Adw.dat” file. For that example, this file has the following form.

“data\Adw.dat”

13, 14, 7, 2, -2
0,2.199014223850483E-002,1.131077023359328E-002
.2507824318335063,.1049348504363801,1.14307691812088
.371805662123287,2.665003038439922,.6295386457485019
3.356488111623067,.5108733435754953,2.170738237525078
.1595796881599627,.5649419229594665,0
4.27527344014845E-002,2.199014223850483E-002,.5784959574527647
.2507824318335063,3.232726625288943,1.14307691812088
9.663951058156133,2.665003038439922,16.69719283041654

3.356488111623067,16.69709889332935,2.170738237525078
8.966901725796344,.5649419229594665,2
.4389709689454479,1.110589907375944,0
7,14293442.33416819
-1.121363213833725E-002,.548563038095354
-3.667608631455786E-002,.5892913397278411
-6.166524714304781E-002,.6717523554543705
-7.114552633152202E-002,.7792803394164933
-5.796497380724816E-002,.883886277512971
-3.339917687322751E-002,.9633261826862721
-1.040631887179268E-002,1.001305584147396
2.137636720074527E-002,2.199014223850483E-002,.294903363843179
.2507824318335063,1.668830737862662,1.14307691812088
5.017878360139711,2.665003038439922,8.663365738082522
3.356488111623067,8.60398611845242,2.170738237525078
4.563240706978153,.5649419229594665,1

In the first line of this file the degrees of polynomials of numerator and denominator of input resistance of loaded filter are. Then, in this line, the number of mobile interpolation nodes inside both the passband and stopband and also an order of approximation of 0 at ω=0. (Minus denotes an approximation of 0 and a deficiency of it denotes an approximation of 1.

The coefficients of a numerator polynomial $A(k)$ are in the 2– 6th lines. The coefficients of a denominator polynomial for input resistance of a filter loaded by R_2 are in the 7–11th lines.

There are frequencies in which the filter interpolates the zero value of transmission in the twelfth line.

There are the quantity of typed poles (the poles with the zero or positive imaginary value are typed) and square of a maximal approximation error inside the stopband in the thirteenth line. There are a real and imaginary parts of the poles with non-negative imaginary parts in the 14–20th lines.

There are 15 coefficients of a polynomial of a denominator for the transfer function of the synthesized filter in the 21 - 25th lines (from the zero coefficient to the n-th one where n is an filter order).

4.6. Impulse Response

The “data\Adw.dat” file describes a filter transfer function sufficiently well. Thence, the data may allow to organize different methods of filter synthesis. Furthermore, the results in that file may allow to compute different filter characteristics. The “Anfotke.exe” program computes an synthesized filter impulse response.

For example, let us calculate the impulse response of that prototype of bandpass filter (see Figure 4-2). The “Anfotke.exe” program computes the parameters $A(k)$ and $\angle B(k)$ of the impulse response. In this program if a deficiency of $\angle B(k)$ takes place then an exponential addend

$$A(k)e^{\sigma_k t}.$$

is supposed. In a common case when an angle (also and the zero angle) takes place then the response contain an oscillating addend

$$A(k)e^{\sigma_k t}\sin(\omega t + B(k)).$$

In that case there are seven oscillations in that response as it is seen in the following output listing.

```
ANFOTKE - Program of response as a time function
for filter with the simple poles
(C) COPYRIGHT  M.G.VITKOV, A.A.VITKOVA, 1996-2002
Input for a calculation: 1 -Ak, Bk; 2 - f(t); 3 -graphing; 4 -Exit? 1

A( 1 )= 92.47266965844996  < B( 1 )= < -89.52438601885272
A( 2 )= 321.0243132590251  < B( 2 )= <  142.0073069380215
A( 3 )= 587.0633271762824  < B( 3 )= < -88.64412097268909
A( 4 )= 691.5234159733111  < B( 4 )= < -91.36557875745675
A( 5 )= 549.0038246429368  < B( 5 )= <  89.05381953187933
A( 6 )= 290.3807395995207  < B( 6 )= < -90.49446361654543
A( 7 )= 85.54013190739718  < B( 7 )= <  89.9204115537925
```

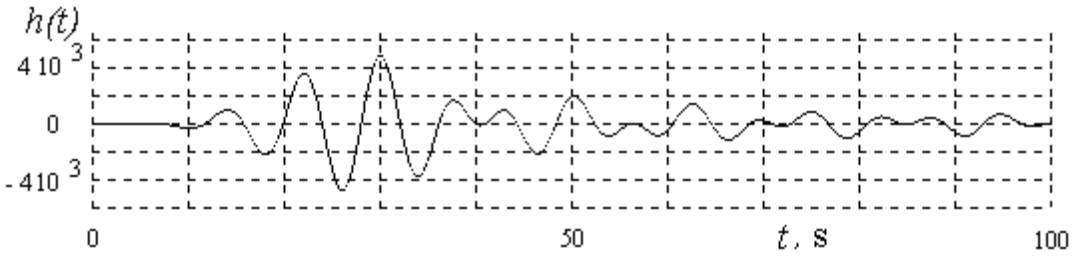

FIGURE 4-2 Computed impulse response.

4.7. Programs for filter network synthesis

For synthesis of filter network the “Ansnte.exe” program has been developed by the authors. This program offers the simplest variants of realization of the calculated network. The numbers of realization in these cases coincide with the numbers of offered elements of realization. Furthermore, some special techniques are possible: synthesis with utilization of Brune’s components or synthesis with negative elements.

For the synthesis will be considered under, the standard networks of realization with variants “6”, “4”,“5” (more exactly “50”), “3” (more exactly “30”), “7” (more exactly “70”), 3 times “1” (more exactly “10”) and “3” (more exactly “30”), once more “1” (more exactly “10”) are used. At the end of a calculation a filter load resistance is realized.

If the “Ansnte.exe” and “Sntnroe.exe” programs are used then we meet with an application of a mask. Before discrimination the next group of components these programs type the calculated with given accuracy coefficients of polynomials of a numerator and denominator of input resistance of a part loaded filter network A_{0j}, A_{1j}, . . ., A_{jj}, D_{0k}, D_{1k}, . . ., D_{kk}. Any whatever of these coefficients A_{0j}, A_{jj}, D_{0k}, D_{kk} really may equal to 0. If we select the mask then it may correct a situation of a calculation. If we select the mask 1, 1, 1, 1 then calculated values of coefficients save their values. If the mask is 0, 1, 1, 1 then the coefficient A_{0j}=0. If the mask is 1, 0, 1, 1 then the coefficient A_{jj}=0. If the mask is 1, 1, 0, 1 then the coefficient D_{0k}=0. If the mask is 1, 1, 1, 0 then the coefficient D_{kk}=0. If we have corrected the need value then we may continue synthesis of filter network.

The synthesis results output in the “data\Tcl.dat” file. For the described filter that file is here.

"data\Tcl.dat"

```
16
41 2 46 3 46 2 3 43 42 3 42 3 42 3 42 1
1 2.782975453302551E-004
.9395559361979255 .9602766374582298 .1026185227109818
7.366909340663519E-002 .6988391327454261
6.675911166357888E-002 7.243719020494208E-003
9.852240555193153E-004
5.404207777156557 4.390896665352455 7.90072229317899
22.88155637179633 2.643847973813101 25.24892478396195
236.9768519156414 1583.356022859957
```

Here, in the first line the quantity of components forming that filter is. There are these components in the second line from the left to the right. Every number accords to one from all components of loaded filter. Our programs use a numeration of filter components in accordance with a network shown in Figure 4-3.

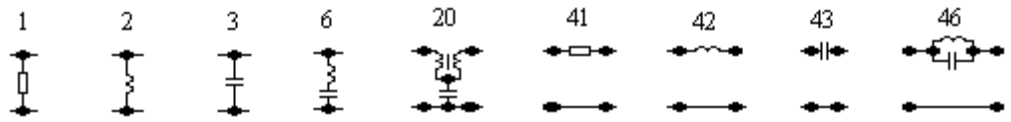

FIGURE 4-3 Numeration of filter components.

Like that, in the second line of that data file the following computed network is described (see Figure 4-4).

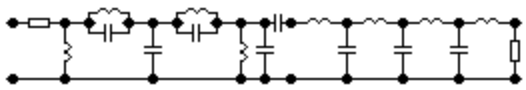

FIGURE 4-4 Accounting network of the designed filter.

In the third line of that file both an input resistor and load one of that filter are. There are inductances in the 4th, 5th, 6th and 7th lines. There are capacitances in the 8th, 9th and 10th lines. All computed values are the components being from the left to the right along the network.

Of course, the described network with unit resistance and unit limiting frequency is not a filter itself but it is a prototype filter. To get a network of real filter it is need to do a denormalization of computed network by using the "Anfshme.exe" program. Next, we show the "data\Tcldn.dat" file containing the data calculated by that denormalization for the last filter when R_1=50 kOhm and f_{B2}=40 kHz.

"data\Tcldn.dat"

```
16
41 2 46 3 46 2 3 43 42 3 42 3 42 3 42 1
```

50000 13.91487789154053
.186918705701828 .191040962934494 2.041530609130859E-002
1.465599983930588E-002 .1390296220779419
1.328130252659321E-002 1.441091997548938E-003
1.960040826816112E-004
4.300531597500878E-010 3.494164402262356E-010
6.287195208898311E-010
1.820856243561764E-009 2.103907326134191E-010
2.009245436696006E-009 1.885801736989379E-008
1.259994633073802E-007
fB = 40000 R1 = 50000

To draw the frequency responses at first it is need to apply the “Ananlzde.exe” program. The program compute polynomial coefficients of numerator and denominator of transfer function and write them in the “data\Anfdn.dat” file. Then it is need to apply the “Abfdndte.exe” program.

In the cases of continuous-signal and recursive filters this program calculates and types the amplitude and phase responses in a scale: 2 vertical cells are equal to $90°$. For example, the synthesized filter phase response is shown (see Figure 4-5).

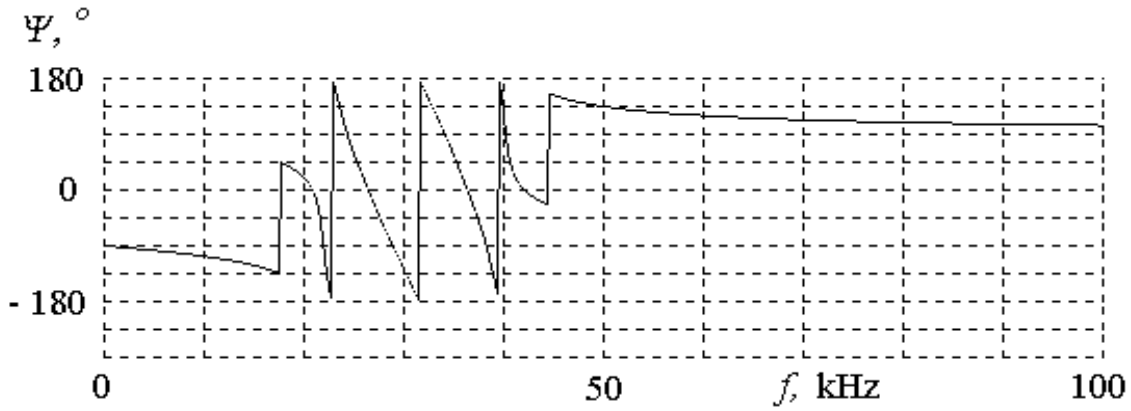

FIGURE 4-5 Phase response of designed filter.

4.8. Other Versions of Filter Realization

If we mutually change data of $A(p)$ and $D(p)$ in the “data \Adw.dat” file then we will get other variants of synthesis. It is may be got, for example, such a network (see Figure 4-6).

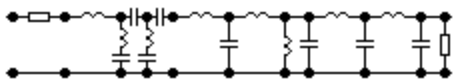

FIGURE 4-6 Other realization of a filter network.

It is may be done for synthesis of highpass filter because at this the number of inductors is often decreased.

The other variant of realization is got if at the beginning we used by synthesis with a Brune's two-point and then by synthesis with a capacitor (see Figure 4-7).

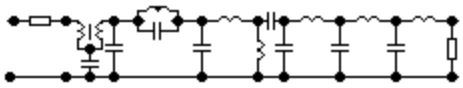

FIGURE 4-7 Brune's realization of a filter network.

The variants of the synthesis allow to simplify the filter network.

5. SYNTHESIS OF RECURSIVE FILTERS

5.1. Computation of Network Parameters

To form a specification for a recursive filter the "Cfzdne.exe" program has been developed. That program transforms given frequencies of a digital filter F_k to the frequencies ω_k of an continuous-signal prototype filter by using bilinear frequency transformation

$$\omega_k = \frac{1}{T_o}\tan\Big(\frac{\pi F_k}{F_D}\Big). \tag{5.1}$$

Here, it is need so to select T_0 that, for a given sampling frequency, one passband limit F_B of a digital filter is transformed to 1, ω_B=1, of the continuous-signal prototype filter. In that case all the data is saved in the "data\Apf.dat" file. There are F_B, F_D and T_0 in the end of that file.

Next, we calculate the parameters of an continuous-signal prototype filter by using the "Anaprame.exe" program. The results of approximation output to the "data\Fap.dat" file. Next, the calculation may be continued by any ways.

At first, by using the "Cfkane.exe" program, we calculate the parameters of canonical recursive filter (see Figure 5-1). The results output to the "data\Cfkangp.dat" file. The transfer z-function of that filter is

$$\hat{H}(z) = \frac{\sum\limits_{k=0}^{n} p_k z^{-k}}{1 + \sum\limits_{k=1}^{n} g_k z^{-k}}. \tag{5.2}$$

Next, by using the "Cfkase.exe" program, we calculate the parameters of cascade recursive filter. The results output to the "data\Cfksoab.dat" file. The transfer z-function of that filter is

$$\hat{H}(z) = \prod_{i=1}^{n_o} \frac{b_{i0} + b_{i1}z^{-1} + b_{i2}z^{-2}}{1 + a_{i1}z^{-1} + a_{i2}z^{-2}}. \tag{5.3}$$

A network of its i-cascade is shown in Figure 5-2.

Those two ways allow to make the digital filters which imitates the transfer functions of continuous-signal prototype filters. Though, there is also the third way. In that case the calculation

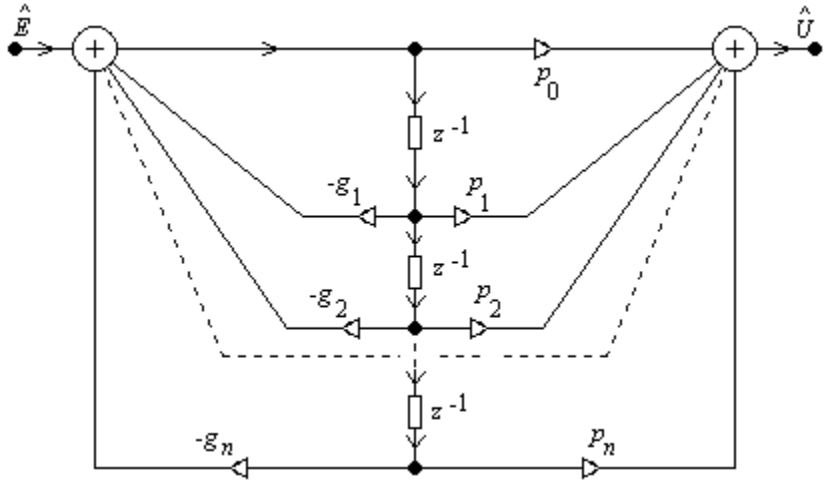

FIGURE 5-1 Canonical recursive filter.

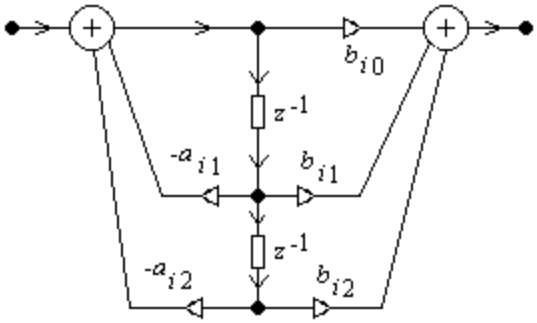

FIGURE 5-2 The i-th cascade of a cascade recursive filter.

of continuous-signal prototype filter is continued by using the "Anadwvte.exe" and "Ansnte.exe" programs for continuous-signal synthesis. At the end of these programs it is offered the way to wave digital filter synthesis by forming the "data\Cf.dat" file. In that case the digital filter imitates yet observed continuous-signal network of a prototype filter. But the components of the networks are noted as new (see Figure 5-3). The results of synthesis of the continuous-signal network components are transformed to according digital networks.

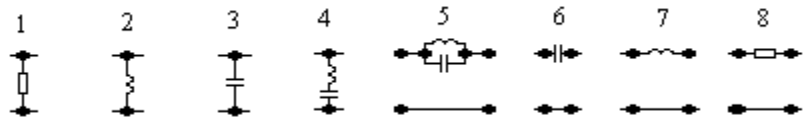

FIGURE 5-3 Circuit elements of filters.

When the wave digital network is formed then the components of a prototype network are accorded the following input wave resistances of a digital network. The resistors R of the 1 and 8 networks (see Figure 5-3) are accorded the wave resistances r=R and wave conductions g=1/R. The wave resistances and wave conductions according to the inductors L of the 2 and 7 networks (see Figure 5-3) accordingly equal to: r=L/T_0 and g=T_0/L. The capacitors of the 3 and 6 networks (see Figure 5-3) are accorded wave resistances and conductions: r=T_0/C and g=C/T_0. The sequential connection of an inductor and capacitor of the 4 network is accorded a wave resistance (see Figure 5-3)

$$r = L/T_0 + T_0/C = (LC + T_0^2)/T_0C = \frac{L}{T_0}\left(1 + (T_0\omega_0)^2\right). \tag{5.4}$$

Here, the wave conduction is

$$g = T_0C/(LC + T_0) = \frac{T_0}{L}\,\frac{1}{\left(1 + (T_0\omega_0)^2\right)}. \tag{5.5}$$

In that case here $\omega_0 = 1/\sqrt{LC}$.

At last, the 5 network of parallel connection of inductor and capacitor is accorded a digital network with wave conduction

$$g = T_0/L + C/T_0 = (LC + T_0^2)/LT_0 = \frac{C}{T_0}\left(1 + (T_0\omega_0)^2\right).$$

Here, wave resistance is

$$r = \frac{T_0}{C}\,\frac{1}{1 + (T_0\omega_0)^2}. \tag{5.6}$$

5.2. Features of Wave Digital Filter Synthesis

In order to supply an imitation of the networks of continuous-signal prototypes, the networks of the wave digital filters very differentiate from usual canonical and cascade networks. The bases of their synthesis has been developed by professor A. Fettweis [4]. Here the economical network of these filters are only used. They have been developed by us in [4]. They have the form of a cascade

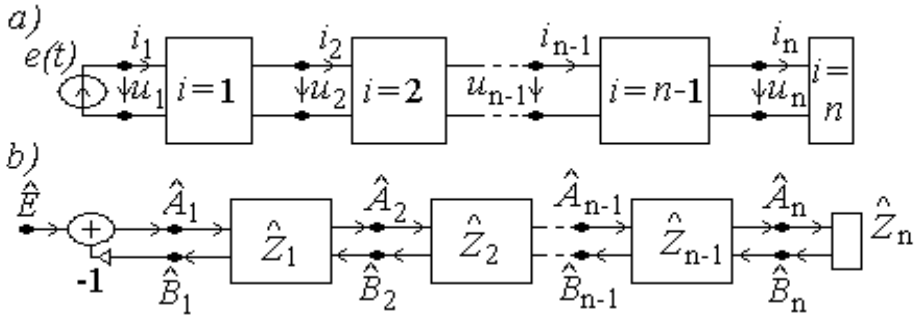

FIGURE 5-4 Economical networks of wave digital filters.

connection from special digital networks as the adapters of in series or parallel connections with the different numbers of i (see Figure 5-4).

The network of imitated continuous-signal filter of an usual long structure is shown in Figure 5-4 (a). Here the $e(t)$ is an e.m.f. at a filter input, the u_i and i_i are the voltages and currents of the different filter places arranged along the filter . These filter places are parallel or in series connections of the different components of an continuous-signal network: resistors, capacitors and inductors. The according wave network of the digital filter (see Figure 5-4 (b)) is described by falling digital waves $\hat{A}_i$, and reflecting waves $\hat{B}_i$. Each connection of the components of an continuous-signal network is realized, for wave digital network, by an adapter and a digital displacing scheme of an according component or their connections in an continuous-signal prototype.

Input e.m.f. of an continuous-signal prototype is exchanged by a collection of input samples, $\hat{E}$=$\hat{E}(z)$. The falling and reflecting waves are connected with the sample units of voltage and current by the wave relations

$$\hat{U}_i(z) = \hat{A}_i(z) + \hat{B}_i(z), \qquad r_i\hat{I}_i(z) = \hat{A}_i(z) - \hat{B}_i(z). \tag{5.7}$$

There is a network at the input of a wave filter (see Figure 5-4 (b)) that accords to the input e.m.f. The according input voltage is equal to sum of (5.7). Thence,

$$\hat{A}_1(z) = \hat{E}(z) - \hat{B}_1(z) \tag{5.8}$$

in the according wave digital network, too.

In an continuous-signal prototype filter, two types of the network ramifications take place: parallel and in series components. At first, consider a parallel cut-in of the network components. These are the networks of an continuous-signal prototype with the numbers 1, 2, 3 and 4 (see Figure 5-3). They accord to the networks of wave filter with an adapter of a parallel type (see Figure 5-5).

The parallel component of an continuous-signal filter is shown in the last Figure in an usual form (see Figure 5-5 (a)) or in a form when this component is connected under (see Figure 5-5 (b)). The according parallel component of a wave digital network will also be shown as connected under (see Figure 5-5 (c), (d)). Here the scheme of an adapter with the number of i of the parallel components of a wave digital network is also shown (see Figure 5-5 (e)).

A coefficient of an adapter m_i is evaluated through the input wave conductions of connected to an adapter network components

$$g_i = g_i' + g_{i+1}, \quad m_i = \frac{g_{i+1}}{g_i}\,. \tag{5.9}$$

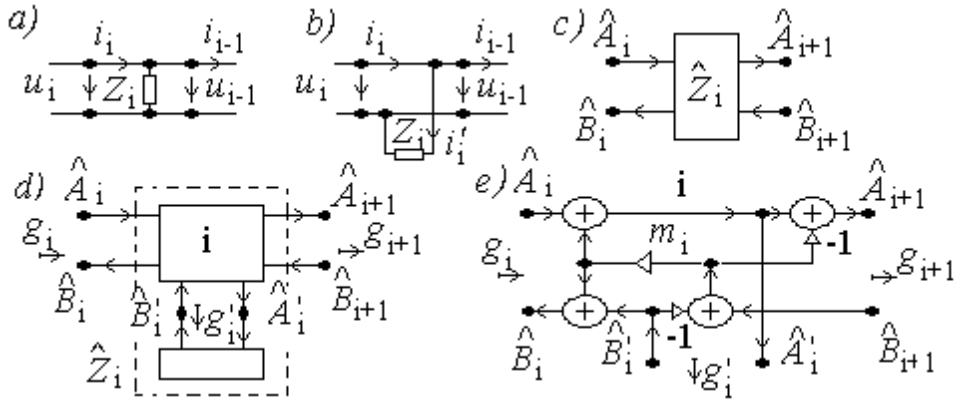

FIGURE 5-5 Wave digital filters with a parallel adapter.

The digital circuits, connected to the adapters, depend on a form of according component of an continuous-signal prototype (see Figure 5-6). In the upper part of this Figure the cut-in of component in an continuous-signal filter is shown but in the under part, the according delineation of this component in a wave digital filter is shown, too.

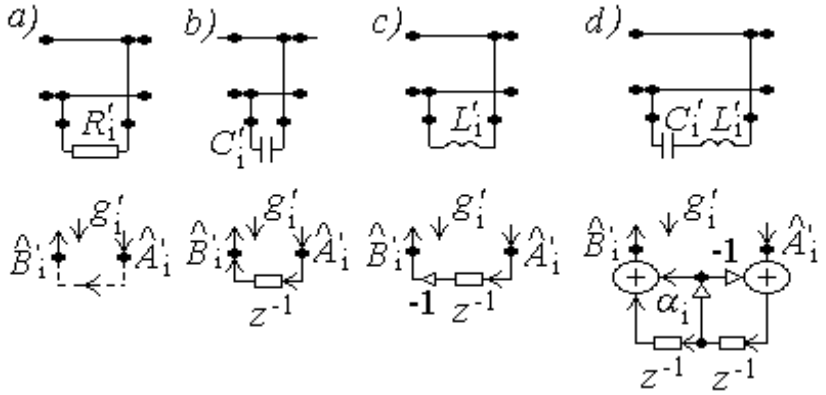

FIGURE 5-6 Interconnection of components in both a continuous-signal and a wave digital filter.

If a resistor was in a prototype circuit then the reflection will not be in a wave digital network: $\hat{B}'_i$=0 (see Figure 5-6 (a)). Here the multiplication is not, i.e., α=0. If a component of an continuous-signal prototype had one capacitor then the wave circuit of a capacitance in a digital network has one delay element (see Figure 5-6 (b)). Here we have multiplication by α=1. If a component of an continuous-signal prototype is an inductor then an according component of a digital circuit will be as a component of one-tact delay as a turn of sign (see Figure 5-6 (c)). Here we have multiplication by $\alpha = -1$. The more composite network with multiplication by α=α_i and one turn of sign accord to a parallel cut-in of an LC–circuit in series in an continuous-signal filter (see Figure 5-6 (d)). In that case we have one turn of sign and

$$g_i = \frac{T_0}{L_i\Big((\omega_i T_0)^2 + 1\Big)}, \quad \alpha_i = \frac{\Big((\omega_i T_0)^2 - 1\Big)}{\Big((\omega_i T_0)^2 + 1\Big)}, \text{ where } \omega_i = \frac{1}{\sqrt{L_i C_i}}. \tag{5.10}$$

The second circuits of the continuous-signal components of a prototype (see Figure 5-7) are connected to its network sequentially.

The network of a digital adapter in series accords to that cut-in (see Figure 5-7 (d)). In this digital circuit in series in the main the parameters are the same as in the scheme of adapter of a parallel connection. Again there is an inversion of sign in two summations. Multiplication by coefficient m_i is realized. A value of m_i depends only on the wave resistances of the connected components:

$$r_i = r_{i+1} + r_i', \quad m_i = \frac{r_{i+1}}{r_i}. \tag{5.11}$$

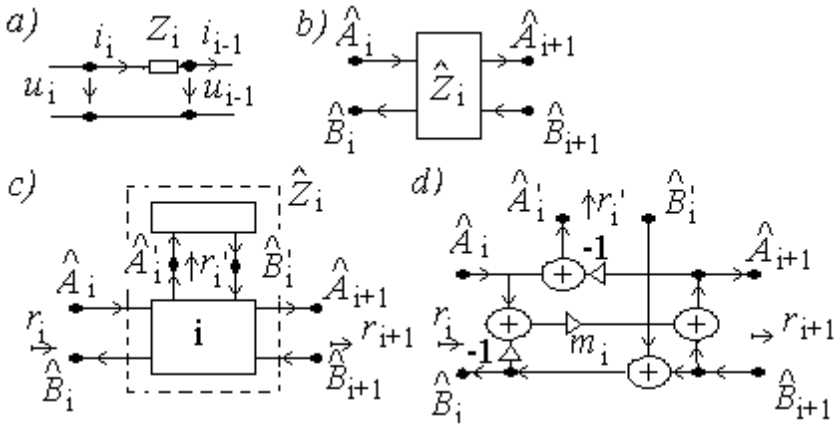

FIGURE 5-7 Network of a serial component.

The networks of the simplest components of continuous-signal prototypes are absolutely analogous to according networks of parallel components R, C, L (see Figure 5-8 (a), (b), (c)). The parameters of a new parallel circuit L and C, connected sequentially, differ a few (see Figure 5-8 (d)). Here a wave resistance r_i and multiplier α_i are

$$r_i = \frac{T_0}{C_i\left(1 + (\omega_i T_0)^2\right)}, \alpha_i = \frac{\left(1 - (\omega_i T_0)^2\right)}{\left(1 + (\omega_i T_0)^2\right)}, \text{ where } \omega_i = \frac{1}{\sqrt{L_i C_i}}. \tag{5.12}$$

All the sequentially connected components of the wave digital filter networks will be shown from above the scheme, i.e., above the schemes of according adapters.

The continuous-signal networks for the synthesis of recursive wave schemes of digital filters are in the "data \Cf.dat" file. These data are used for a synthesis of a scheme of a wave digital filter. It is done by the "Wcfe.exe" program. Besides, it can be calculated a operation of these filters and also the frequency responses of the wave filters. The results of a synthesis of those filters are saved in the "data\Wcf.dat" file. The data about time variables (exciting and resulting voltages of the digital filters) is saved in the "data\Ek.dat" and "data\Uk.dat" files.

5.3. Examples of Recursive Filters Synthesis

Consider synthesis of the digital filters with infinite pulse response. They have F_D=24 kHz and realize a band filtering with the loss less than 0.6 dB inside the range as 4–7.4 kHz and not less than 60 dB inside the ranges as well 0–3.4 kHz as 8–11.4 kHz.

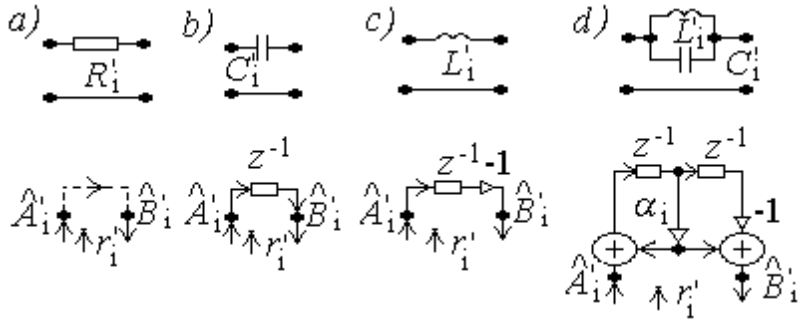

FIGURE 5-8 Networks of the simplest components.

At first, we use the "Cfzdne.exe" program and input the parameters of that filter. Then we compute the transformed frequencies of an continuous-signal prototype ω_k=$w(k)$ by the "Anaprame.exe" program. As a result the following "data\Apf.dat" file is formed.

"data\Apf.dat"

3
0 1 0
2 2 2
0 .3278162013423066 .3968018464575454 1 1.190405539372636
8.732732571266508
60 .6 60
7.4, 24, 1.455009028672445,"$Fo;\ FD;\ To$"

The synthesis of an continuous-signal filter-prototype may satisfy the given requirements if we select $N(1)$=2, $N(2)$=6, $N(3)$=2, $N0 = -2$. In that case the maximal passband loss will be at most 0.56 dB.

If next we will continue the calculation by the program of synthesis of canonical recursive digital filter then under the round-off 5 decades we will get a canonical recursive network of an order n=12. It is shown in Figure 5-1. The resulting data is saved in the "data\Cfkangp.dat" file.

"data\Cfkangp.dat"

12
-.84012 4.1926 -2.9483
8.123699999999999 -4.6315 9.0098
-3.9667 5.9858 -1.8394
2.2481 -.3694 .37346
.72496 -.33978 -.78679
.20164 1.0112 .13814
-1.8988 .13814 1.0112
.20164 -.78679 -.33978
.72496

Here, there are the g_1, g_2, . . ., g_{12} parameters in the 2nd–5th lines but there are the value of p_0, p_1, . . ., p_{12} in the 6th–10th lines. Calculated 500 sample units of a filter response with $\hat{E}(z)$=1, we can get the frequency responses of that digital filter (see Figure 5-9) if an accuracy of a calculation is 16 decades. Approximately the same results were got by using of a cascade synthesis and a wave digital filter.

The cascade scheme was synthesized when the calculation accuracy was 5 decades. The scheme is saved in the "data\Cfksoab.dat" file. That scheme consists of 6 cascades each of which is described by 5 coefficients a_{i1}, a_{i2}, b_{i0}, b_{i1}, b_{i2}.

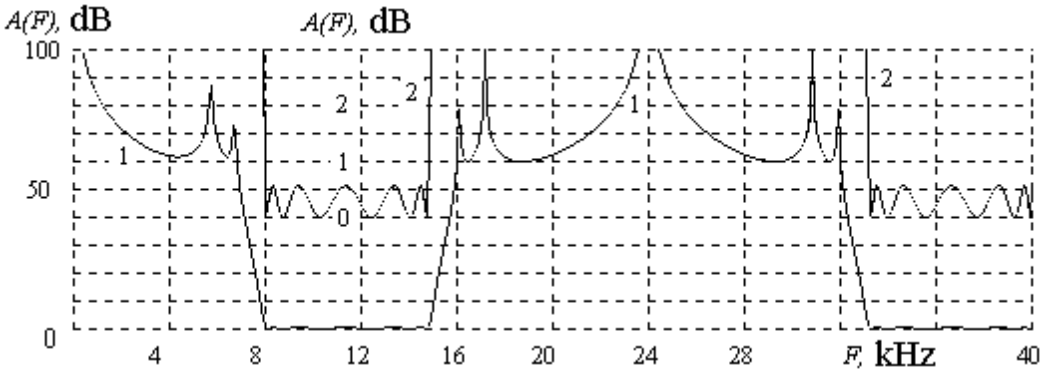

FIGURE 5-9 Computed frequency response of a digital filter.

"data\Cfksoab.dat"

```
6
.12211 .74143 .85705 1.7141 .85705
-.40657 .74534 .53798 -1.076 .53798
.52175 .85492 1.7815 2.2298 1.7815
-.79991 .85959 2.7201 2.7806 2.7201
.70631 .95825 .38198 -.48799 .38198
-.98381 .95966 .84947 -1.2445 .84947
```

By the synthesis of a wave digital filter, at first, the scheme of an continuous-signal prototype is computed (see Figure 5-10). The data of that synthesis is saved in "data\Cf.dat" file. For that synthesis the "Anadwvte.exe" and "Ansnte.exe" programs are used.

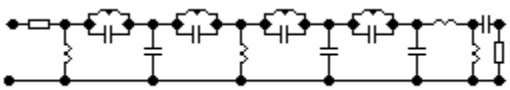

FIGURE 5-10 Network of a continuous-signal prototype of a wave digital filter.

"data\Cf.dat"

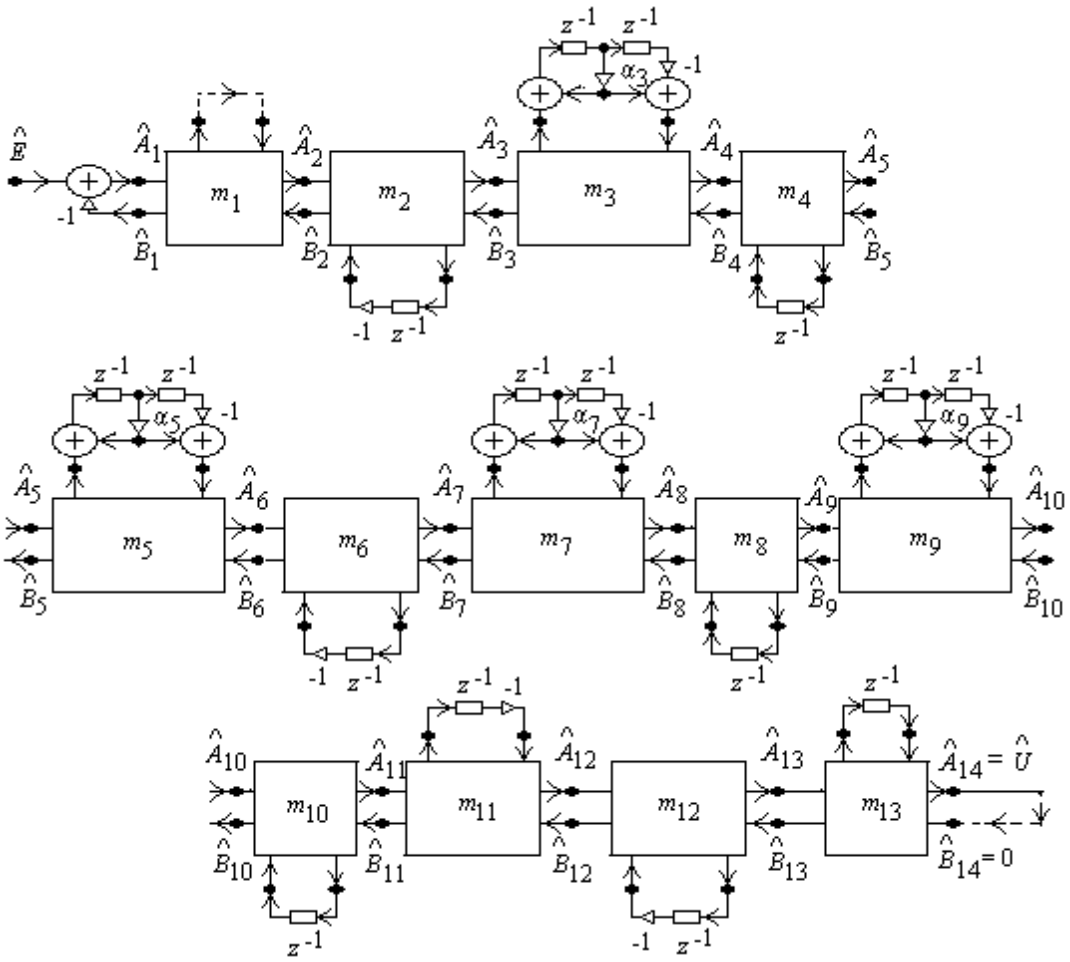

FIGURE 5-11 Synthesized network of a wave digital filter.

14, 8, 8, 2, 7. 4, 24
8 2 5 3 5 2 5 3 5 3 7 2 6 1
1.493768007125467 1.499816662316289 .2245065917582151
.278017867387159 .8061956463885985 5.316177733945213E-002
2.065706145382831E-002 8.717183743475545E-003
6.403520763564617 3.213745881235802 3.050654003462776
17.01063007100337 23.86766227378486 9.164851990832327
81.36201511086514 411.2140284562867
1 2.05052473187388E-003

Here, there is a number of the scheme blocks, a number of the inductors, capacitors, resistors, the passband highest frequency and sampling frequency by kHz in the 1st line. There are the circuit component forms (from the left to the right along by a filter scheme accordingly to the numeration in Figure 5-3) in the 2nd line. There are the values of all inductors in mHn in the 3th–5th lines. There are the values of all capacitors in mF in the 6th–8th lines. There are the values of all resistors in Ohm in the 9th line. Here the components have the numbers in a circuit from the left to the right.

By those results the "Wcfe.exe" program executes the synthesis of a wave digital filter scheme (see Figure 5-11).

That program allows at first to find a known scheme of a wave digital filter by the data for a prototype scheme. Besides all parameters of that filter are calculated and its coefficients m_i are found. Next, a digital impulse response and frequency responses are computed. The results of that synthesis are saved in the "data\Wcf.dat" file.

"data\Wcf.dat"

m(i)
.19445196 .76487251 .4100686 .71415265
.35666351 .66174134 .2413673 .61328563 .22732712
.51135823 .16920567 .51737065 .36689564
al(i)
0 -1 .63876416 1
-.51113075 -1 .73254269 1 -.62583018
1 -1 -1 1 0

In accordance with notations in Figure 5-6 and 5-8 the synthesized by the found scheme of an continuous-signal prototype wave filter is shown in Figure 5-11. Here, m_1, m_3, m_5, m_7, m_9, m_{11}, m_{13} are the parameters of the adapters of a serial connection of the continuous-signal circuit components and m_2, m_4, m_6, m_8, m_{10}, m_{12} are the parameters of the adapters of a parallel connection. The synthesized filter consists of 13 adapters of the serial and parallel component connections.

The digital pulse response (see Figure 5-12) was computed and plotted in the form of the simplest bar graph by using the "Wcfgrhe.exe" program.

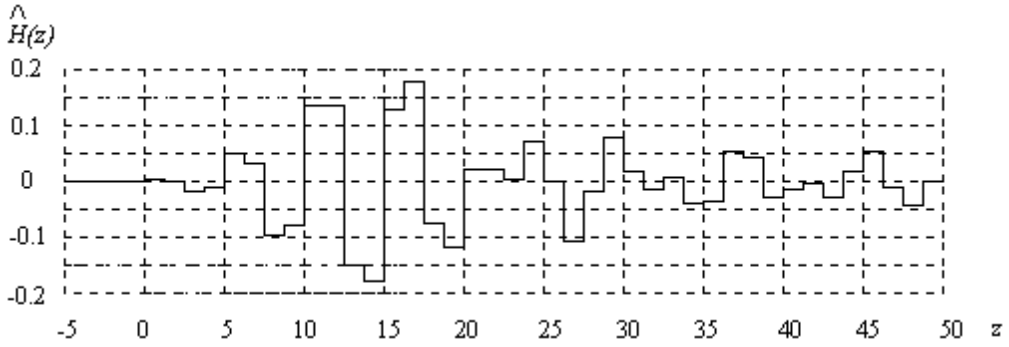

FIGURE 5-12 Digital impulse response.

6. MORE COMPOSITE FILTERS

6.1. High Order Filters

If an order of a synthesized filter increases then the computation is significantly sophisticated. The problems bound with a calculation of algebraic equations and modular polynomials are distinct complexity. To solve these tasks one can use our special programs in which a digit capacity of the computations increases up to 30–45 digits and more. An application of those programs eliminates all difficulties. As a high order filter calculation is a comparatively more rare problem then the data files of these programs have the "rzl" add-in. They are in a main directory together with all the programs.

To increase a computation accuracy we perform every rational number as a assembly of a few numbers: an integer number is a sign of a rational number, an other integer number is an order of that rational number and a certain selected integer number N_R from the rational numbers of an usual extended accuracy (16 decades) that is a number of the digit groups (each group consists of 15 decades) of that rational number. Every group has only 15 digits. The 16th digit does not use. The sign has 2 possible values: +1 and -1. The number of the digit groups N_R may equal to 1, 2, 3 and so on. That number is selected at the beginning of a computation by the "Sntvnroe.exe" program. If a filter of the 15th order is synthesized then an accuracy with 45 decades is adequate, i.e., N_R=3 is adequate.

The algorithms of an arithmetic for so high accuracy of computation have been developed by us. Using that arithmetic one may execute the operations of an addition, deduction, division, multiplication and also evolution of a square root from these multi-digit numbers.

For example, produce the synthesis of a lowpass prototype filter that has at an unit frequency of passband the largest loss is less than 0.1 dB, and has the loss is not lower than 60 dB at the frequency is higher than 1.01 rad/s. The order of that filter is about 15. Thence an approximation of its response executed by the "Anaprame.exe" program required 7 adjusted interpolation nodes inside the passband and 7 adjusted interpolation nodes inside the stopband with 1 fixed interpolation node of an unit value for zero frequency, i.e., N_o=1. Thence for a continuation of that filter synthesis the "Sntvnroe.exe" program was used. In that case 45 decades are used, i.e., N_R=3. The results are saved in the "Mdaunr.rzl" and "Adwnrl.rzl" files.

The "Mdaunr.rzl" file contains the data for modular algebraic equation. Here, there is the value of N_R and the order of that equation n in the 1st line. Next, there are the data of that algebraic

equation coefficients, written with a selected accuracy (a sign of coefficient, its order and its value in the form of three groups with 15 digits, because N_R=3) in the 2nd–17 lines.

"data\Mdaunr.rzl"

```
3,  15
1 -1 184920348661467 575042688046964 689701159442041
-1 0 188947411127293 844949154192083 47354148437853
1 0 827164137709902 510240247419191 981133640569025
-1 1 160657396531424 95899070953508 375075692996445
-1 1 190008780450610 572091512431154 886902437052663
1 2 243100831792800 787753886805291 394321345207908
-1 2 909639521444530 880262580596296 587043022712381
1 3 217379177560789 106622096966599 363987796926628
-1 3 372052026353379 956953019754192 306545499658286
1 3 471359822101084 749483513249041 891572096065607
-1 3 444555288436876 107124629146892 198608162468049
1 3 308753567619005 809405736067380 216967601044140
-1 3 153554052030475 184649331088198 157478923292738
1 2 517928780504019 247071316192802 739631886624743
-1 2 106202565370057 379722218830626 832227793909681
1 1 100000000000000 0 0
```

The data of the "Adwnrl.rzl" file are need for the next synthesis. Here, there are the number of the digit groups for a calculation N_R=3, the number of selected displaced zeroes inside both the passband and stopband (7 and 7) and the order of approximation of an unit value at the zero frequency that equals to 1 in the 1st line. There are the filter coefficients defining its sensitivity in the 2nd–3rd lines. There are the frequencies where a transfer function equals to 1 in the 4th–10th lines. There are the frequencies, where a transfer function equals to 0, in the 11th–17th lines. All the values in the 2nd–17th lines are given accurate to 15 decades.

"data\Adwnr1.rzl"

```
3,  7,  7,  1
1 0 849394866241807 0 0 rrm
1 5 200754305547780 0 0 rm0
1 0 420566688074811 0 0 w1(k)
1 0 715774613804119 0 0 w1(k)
1 0 875445125073526 0 0 w1(k)
1 0 949727219853778 0 0 w1(k)
1 0 981747041980279 0 0 w1(k)
1 0 994854361798482 0 0 w1(k)
1 0 999497435170105 0 0 w1(k)
1 1 101050690686825 0 0 w(k)
1 1 101521980966979 0 0 w(k)
1 1 102877750364363 0 0 w(k)
1 1 106348423401746 0 0 w(k)
```

1 1 115376799059892 0 0 w(k)
1 1 141121338330552 0 0 w(k)
1 1 240186910517109 0 0 w(k)

Using the given coefficients of an algebraic equation in the “Mdaunr.rzl” file, the roots of a filter characteristic equation have calculated with a high accuracy by the “Snaunroe.exe” program. It prepares the “Xynr.rzl” data file containing the roots of that equation. There is the number of the characteristic equation roots with the non-negative values of a imaginary value in the 1st line. Eight such roots are in that example. Their values are in the following lines. For example, in the 2nd line: −1 is a sign of the first root, 0 is an order of a real component, 0 and 0 are a deficiency of an imaginary part of the first root. There are the values of a real part of that root containing 15 decades in the 3–5th lines. In the 6th line −1 and 0 are the sign and the order, respectively, of a real part, but 1 and 0 are the sign and the order of an imaginary part of that root. Then There are the values of both real and imaginary parts of the second root containing 15 decades. They are written by comma in the 7th–9th lines. And so on up to the last.

“data\Xynr.rzl”

```
8
-1,  0,  0,  0
538307272619767,0
892462743832604,0
967338668791849,0
-1,  0,  1,  0
422773834522369,515364568981536
733119531650458,431018778292858
587679718579078,691413911380709
-1,  0,  1,  0
231482862110626,804147176671104
897537363353887,174569805867832
731184228863227,988203891205129
-1,  0,  1,  0
106544794545871,925199968804833
583394724011362,292778147874957
59719710344752,10377895241194
-1,  -1,  1,  0
456608724051602,972928859405440
535373976437654,611643750057788
108958716226441,995102097699149
-1,  -1,  1,  0
188814248929687,991865040099741
778284119004707,660442585033953
601547701304825,373960201146704
-1,  -2,  1,  0
734409055516681,999334497124168
404027748765119,504654047935711
```

170022025753324,154746325825329
-1, -2, 1, 1
191252189577549,100194007093705
349751594294542,993834813461177
86473630256405,848015276713707

The main accounting polynomials of the problem have compiled by using the “Sadwnroe.exe” program. The data for the “Adwnr2.rzl” file have prepared yet. Here, in the 1st line: 3 is N_R, 14 and 15 are the order, respectively, of both numerator and denominator of an input resistance of loaded filter, 7, 7 and 1 are the number of approximating nodes in both passband and stopband and the order of approximation of an unit value at zero frequency and, at last, 8 is the number of the roots of a denominator with non-negative imaginary parts. Then the lines with the values of the coefficients of both a numerator and a denominator of an input resistance of loaded filter follow. Then the lines with the zero value of filter follow. At last, the lines with real and imaginary parts of its roots

$$p_k = x_k + jy_k$$

are following. The filter normalizing coefficients are at the end.

“data\Adwnr2.rzl”

3, 14, 15, 7, 7, 1, 8
0), 1 0 135985421520642 268825275146991 451741461095355
1), 1 0 595833030574113 11642996781520 867073729367935
2), 1 1 227475964911094 758850282480757 374928831504244
3), 1 1 495364459170206 321529813770493 80178992669322
4), 1 2 116555224999772 289314869923848 630581240111700
5), 1 2 162762524333873 461129993259641 374202604443444
6), 1 2 288331088483001 459529371589069 85144892370918
7), 1 2 275023443780082 596113519511838 347173443529994
8), 1 2 394818617031761 877859039965133 163152857406026
9), 1 2 254478658800210 126787932378937 259101523288547
10), 1 2 307360957274046 452985597407324 916859988635309
11), 1 2 123004833599491 807735250273093 753594206247594
12), 1 2 127781026827773 689123995330446 705625641010692
13), 1 1 243652104337150 841842040939767 342727605635119
14), 1 1 220750807447564 699837318781368 729173718099234
a(l)
0), 1 0 135985421520642 268825275146991 451741461095355
1), 1 0 715228963929132 699163773212675 35458912551457
2), 1 1 227475964911094 758850282480757 374928831504244
3), 1 1 651389795843829 326824698308920 595475229426478
4), 1 2 116555224999772 289314869923848 630581240111700
5), 1 2 239752117627701 31425273013690 399212690800976
6), 1 2 288331088483001 459529371589069 85144892370918
7), 1 2 470529457833578 682727455402093 145564040348130

```
8 ), 1 2 394818617031761 877859039965133 163152857406026
9 ), 1 2 537165386223926 734189544161884 422148951817327
10 ), 1 2 307360957274046 452985597407324 916859988635309
11 ), 1 2 358969629489692 701140422255767 266408617384590
12 ), 1 2 127781026827773 689123995330446 705625641010692
13 ), 1 2 130568273925094 82085930676597 374272760563511
14 ), 1 1 220750807447564 699837318781368 729173718099234
15 ), 1 1 200000000000000 0 0
d(l)
1 ), 1 1 101050690686825 0 0
2 ), 1 1 101521980966979 0 0
3 ), 1 1 102877750364363 0 0
4 ), 1 1 106348423401746 0 0
5 ), 1 1 115376799059892 0 0
6 ), 1 1 141121338330552 0 0
7 ), 1 1 240186910517109 0 0
w(l)
-1 0 538307272619767 892462743832604 967338668791849
-1 0 422773834522369 733119531650458 587679718579078
-1 0 231482862110626 897537363353887 731184228863227
-1 0 106544794545871 583394724011362 59719710344752
-1 -1 456608724051602 535373976437654 108958716226441
-1 -1 188814248929687 778284119004707 601547701304825
-1 -2 734409055516681 404027748765119 170022025753324
-1 -2 191252189577549 349751594294542 86473630256405
x(k)
0 0 0 0 0
1 0 515364568981536 431018778292858 691413911380709
1 0 804147176671104 174569805867832 988203891205129
1 0 925199968804833 292778147874957 10377895241194
1 0 972928859405440 611643750057788 995102097699149
1 0 991865040099741 660442585033953 373960201146704
1 0 999334497124168 504654047935711 154746325825329
1 1 100194007093705 993834813461177 848015276713707
y(k)
1 0 849394866241807 0 0 rrm
1 5 200754305547780 0 0 rm0
```

At last, the synthesis of that filter is generated by the “Sntnroe.exe” program (see Figure 6-1). The results output in the “data\Tcl.dat” file and they can be transformed further as the usual data.

“data\Tcl.dat”

```
17
41 3 46 3 46 3 46 3 46 3 46 3 46 3 46 3 1
1 1
```

.455882958854009 .152052778432317 .283003982460178
.403553427135457
.571397318454284 1.21327423351767 1.30243020971256
9.74949048051412E-002 1.64781962041046 .55379436125589
6.44061096527628
.320378933839045 3.42836684239086 .408695493055679
2.34129448575545
.568434555522408 1.54738965898443 1.01981163226641
.413862095677677
1.50911941144489 .13309044190254 .781871008571901
nR, nL, nC= 2 7 15

Here, in the 1th line 17 is a number of a filter components. These components are from the left to the right according Figure 4-3 in the 2nd line. In the 3rd line 1 and 1 are. These are the resistances of the filter loads. There are the values of inductances in the 4–6th lines. There are the values of capacitances in the 7–13th lines. There are the number of resistors, inductors and capacitors of a loaded filter in the 14th line.

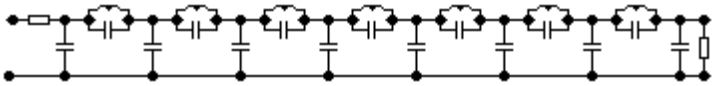

FIGURE 6-1 Synthesis of that filter.

The frequency response of that filter is computed by the “Ananlize.exe” and “Anabfcle.exe” programs. It is shown near transient range (see Figure 6-2).

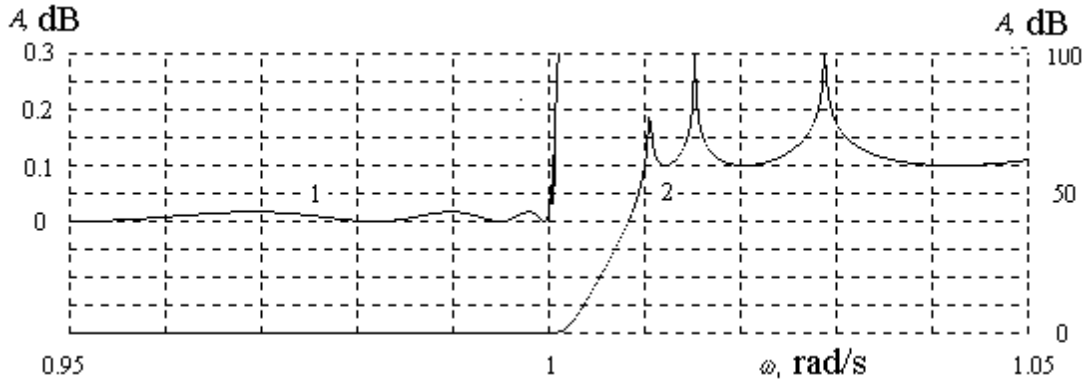

FIGURE 6-2 Frequency response of that filter.

6.2. The Band Transformation

As the prototypes of the bandpass and bandstop filters may be used both the lowpass and highpass filters. Next, one is used the band transformation that increases an order n by two times.

For example, consider a prototype as a lowpass filter. A prototype order is equal to 11. It allows to have a narrow transient range: when $\omega \leq 1$ rad/s, $A < 0.1$ dB, but when $\omega > 1.05$ rad/s, $A \geq 60$ dB (see Figure 6-3 (a)). At first, a denormalization of synthesized lowpass filter is executed by the "Anfshme.exe" program. For example, it may be done by changing of the passband up to 10 kHz.

According bandpass filter (see Figure 6-3 (b)) is deduced by using of a band transformation in the "Anfshme.exe" program. Under, the band transformation is considered when the stopband limiting frequencies are equal to 19.696 and 30.196 kHz .

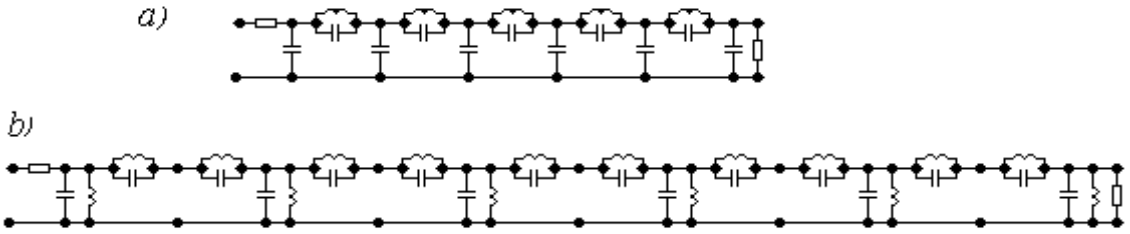

FIGURE 6-3 Lowpass prototype (a) and bandpass filter network (b).

The synthesized filter frequency characteristics, inside the passband neighborhood, have been computed by the "Ananlzde.exe" program for the analysis of denormalized networks. Then the "Abfdndte.exe" program for a computation of frequency responses of the denormalized networks is used (see Figure 6-4).

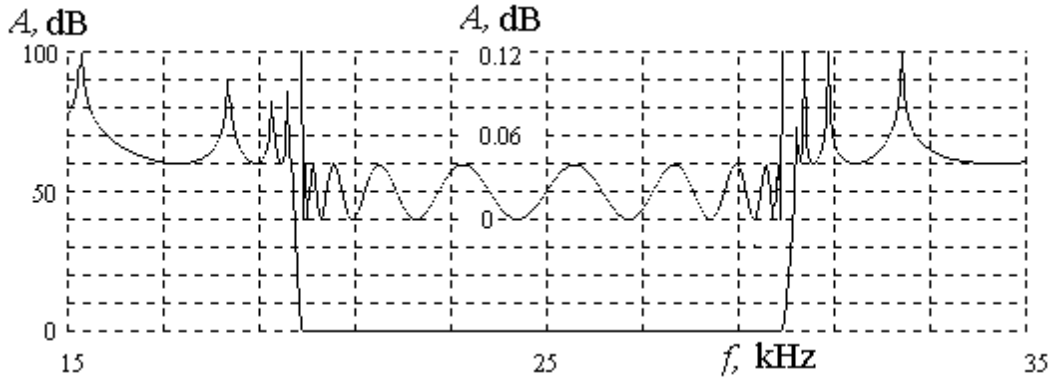

FIGURE 6-4 Frequency responses of that filter.

7. INVERSE Z-TRANSFORM

7.1. The Main Relations

The inverse z-transform is usually called a transition to the original in the form of a certain time dependence

$$\hat{U}(z) = u_0 + u_1 z^{-1} + u_2 z^{-2} + \ldots = \sum_{n=0}^{\infty} u_n z^{-n} \tag{7.1}$$

from z-function as a relation of two polynomials.

$$\hat{U}_T(z) = \frac{\sum\limits_{l=0}^{M}\Big[\Big(B_x(l) + jB_y(l)\Big)z^{-l}\Big]}{\prod\limits_{k=1}^{n_r}\Big[1 - \Big(Z_x(k) + jZ_y(k)\Big)z^{-l}\Big]^{r_k}}. \tag{7.2}$$

Here, $B_x(l)$ and $B_y(l)$ are a real and imaginary components, respectively, of the l-th polynomial coefficient of a numerator, but $Z_x(k))$ and $Z_y(k)$ are a real and imaginary components, respectively, of the roots of a polynomial in denominator which multiplicity is equal to r_k. In that case a number of the differentiating roots is equal to n_r.

As we have shown in [10], in that case the original sums from the following addends.

Suppose that

$$N = \sum_{k=1}^{n_r} r_k. \tag{7.3}$$

If $N > M$ then every addend of an original at time nT is

$$\Big(D_x(k,r) + jD_y(k,r)\Big)n^{r-1}z_k^n \tag{7.4}$$

throughout numbers $r \le r_k$. In that case a real and imaginary components of the coefficients $D_x(k,r)$ and $D_y(k,r)$ may be calculated by the "Obrze.exe" program.

If $N \le M$ then, for the first numbers $n < M-N$ involved n=0, the addend $D_x(0,n)+jD_y(0,n)$ is added. The value of that addend is computed by the "Obrze.exe" program, too.

7.2. Example of Computation of the Inverse z-Transform

Consider the following function as a example of the inverse z-transform.

$$\hat{U}_T(z) = \frac{2 + 6z^{-1} + 8z^{-2} + 10z^{-3} + 8z^{-4} + 5z^{-5} + 1z^{-6}}{\Big(1 - 0.5z^{-1}\Big)^2\Big(1 + (0.09 + j0.8)z^{-1}\Big)\Big(1 + (0.09 - j0.8)z^{-1}\Big)}.$$

That function has a 2-fold real root and 2 simple (non-fold) complex-adjoint roots of a denominator. The data of that problem and its solution by the program is in the following listing.

In that example an order of numerator M is 6, but an order of a denominator N is 4. Consequently, in the $M - N$+1=3 first points the value of D_x(0,n)=DX(0,n), n=0, 1, 2, is added to the inverse transform. (Here it is accounted that an imaginary component of an addition equals to 0).

The fold first pole will answer for an sample unit addend that begins at a moment n=1

$$D_x(1,\,2)nz_1^n = 1.209391762 \cdot 10^2 \cdot n \cdot 0.5^n.$$

Here it is again accounted with the zero value of an imaginary component.

The sample units from all other (1-fold roots) will be created for the moments from n=0 up to infinity. The 1-fold real first pole adds a following real component

$$D_x(1,\,1) \cdot z_1^n = -0.218453713 \cdot 10^3 \cdot 0.5^n.$$

```
Input the numerator order nB and diverse poles number nZ? 6, 3
Bx( 0 ),By( 0 )? 2, 0;
Bx( 1 ),By( 1 )? 6, 0;
Bx( 2 ),By( 2 )? 8, 0;
Bx( 3 ),By( 3 )? 10, 0;
Bx( 4 ),By( 4 )? 8, 0;
Bx( 5 ),By( 5 )? 5, 0;
Bx( 6 ),By( 6 )? 1, 0;
nB= 6
Zx( 1 ),Zy( 1 ),r( 1 )? +.5, 0, 2
Zx( 2 ),Zy( 2 ),r( 2 )? -.09, -.8, 1
Zx( 3 ),Zy( 3 ),r( 3 )? -.09, +.8, 1
V( 0 , 1 )= 1 ; V( 0 , 2 )= 0 ; V( 1 , 2 )= 1 ;
DX( 1 , 2 )= 1.209391762E+02; DY( 1 , 2 )= 0.000000000E+00;
?
DX( 1 , 1 )=-0.218453713E+03; DY( 1 , 1 )= 0.000000000E+00;
?
DX( 2 , 1 )=-0.973731806E+00; DY( 2 , 1 )= 1.493226124E+00;
?
DX( 3 , 1 )=-0.973731806E+00; DY( 3 , 1 )=-0.149322612E+01;
?
DX(0, 0 )= 2.224011768E+02; DY( 0 , 0 )= 0.000000000E+00;
?
DX(0, 1 )= 5.383283495E+01; DY( 0 , 1 )= 0.000000000E+00;
?
DX(0, 2 )= 6.171887054E+00; DY( 0 , 2 )= 0.000000000E+00;
?
Input:0-Exit?
```

The 2nd pole adds the addend

$$\Big(D_x(2,\,1)+jD_y(2,\,1)\Big)\cdot\Big(Z_x(2,\,1)+jZ_y(2,\,1)\Big)^n=$$
$$=(-0.973731806+j1.49322612)\cdot(-0.09-j0.8)^n.$$

The 3rd pole adds a addend that is complex-adjoint with a previous addend

$$\Big(D_x(3,\,1)+jD_y(3,\,1)\Big)\cdot\Big(Z_x(3,\,1)+jZ_y(3,\,1)\Big)^n=$$
$$=(-0.973731806-j1.49322612)\cdot(-0.09+j0.8)^n.$$

Thus, the points of total original for any time moment nT are computed.

The description in Russian has been published in 2003 [15].

REFERENCES

1. Vitkov M.G. K formulam teoremi pazlozheniya operatornogo metoda rascheta perehodnih processov (in Russian). *Radiotehnika*, 1969, no. 1, str. 27-30.

2. Vitkov M.G. *Analiz i sintez peredatochnih funkciy elektricheskih cepey / Novie effektivnie chislennie metodi shemnogo proektirovaniya* [in Russian]. Avtoreferat doktorskoy dissertacii. M.: Institut radiotekhniki i elektroniki RAN (IRE RAN), 1993.

3. Vitkov M.G., Sh'erbina E.G. *Cifrovie fil'tri (vvedenie v teoriyu)* [in Russian]. Preprint. Moskovskiy institut inzhenerov transporta (MIIT). M.: 1999, 52 str.

4. Vitkov M.G. *Tsifrovie fil'tri (volnovie shemi)* [in Russian]. Preprint. Informsvyaz'izdat. M.: 1995, 26 str.

5. Vitkov M.G. Chebishevskie approksimatsii tangensa [in Russian]. *Dokl.Akad. nauk SSSR*. 1974, tom 216, no. 4, str. 721-723. (Čebyšev Approximations of the tangent. *Soviet Math. Dokl*, vol. 15, 1974, no. 3, pp. 859-861).

6. Vitkov M.G., Vitkova A.A. Reshenie algebraicheskih uravneniy metodami Berstou i modulyarnih preobrazovaniy [in Russian]. In: *Algoritmi i programmi*. M.: Gos. fond. Inform. byul. VNTIC, 1988, no. 6, str. 1-2.

7. Vitkov M.G., Vitkova A.A. Povishenie tochnosti resheniya algebraicheskih uravneniy teorii fil'trov pri primenenii modulyarnih preobrazovaniy [in Russian]. *Trudi uchebnih institutov svyazi. Electronnie ustroystva sistem svyazi*, 1988, str. 139-140.

8. Vitkov M.G. Smesh'enie polyusov peredachi v fil'trah iz-za poter' v induktivnih elementah [in Russian]. *Elektrosvyaz'*, 1984, no. 1, str. 56-59.

9. Vitkov M.G. Nailuchshee priblizhenie ogranichivayush'ih urovney ogranichennimi ratsional'nimi funkciyami [in Russian]. *Radiotehnika i elektronika*, 1996, tom. 41, no. 2, str. 210-215. (*Journal of Communications Technology* & *Electronics* , 1996).

10. Vitkov M.G., Vitkova A.A. Svoystva i primeneniya z-preobrazovaniy elementarnih diskretnih signalov pri kratnih polyusah izobrazheniy [in Russian]. *Elektrosvyaz'*, 2002, no. 5, str. 39-42.

11. Zaal' R. *Spravochnik po rachetu fil'trov* [in Russian]. M.: Radio i svyaz' , 1983. (Saal R. *Handbuch zum filterenwurf*. © AEG-Telefunken, 1979).

12. Balabanyan H. *Sintez elektricheskih cepey* [in Russian]. M.-L.: Gosenergoizdat, 1961. (Balabanian N. *Network synthesis*. Prentice-Hall, 1958).

13. Karni S. *Teoriya cepey /Analiz i sintez* [in Russian]. M.: Svyaz', 1973. (Karni S. *Networks theory: analysis and synthesis*. Allyn and Bacon, Inc., Boston, Massachusetts, 1966).

14. Rabiner L.R., Gould B. *Teoriya i primenenie cifrovoy obrabotki signalov* [in Russian]. M.: Mir. 1978. (Rabiner L.R., Gold B. *Theory and Application of Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

15. Vitkov M.G., Vitkova A.A. *Analogovie i tsifrovie fil'tri (osnovi teorii i sintez bez problem)* [in Russian]. Institut problem peredachi informatsii RAN (IPPI RAN) (IITP RAS). M.: 2003, 48 s.

This paper was recommended for publication by V.V.Zyablov, a member of the Editorial Board