Current problems in information and computational technologies

1

Monografie – Politechnika Lubelska

Lublin University of Technology
Faculty of Electrical Engineering
and Computer Science
ul. Nadbystrzycka 38A
20-618 Lublin

Current problems in information and computational technologies 1

edited by
Waldemar Wójcik and Jan Sikora

Politechnika Lubelska
Lublin 2012

Reviewers:
prof. dr hab. inż. Zbigniew Łukasik
prof. dr hab. inż. Ryszard Romaniuk

Authors:
Mukhtar Junisbekov, (Taraz State University, Kazakhstan) – Chapter 1
Alexandr Khimich, (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 4
Paweł Komada (Lublin University of Technology, Poland) – Chapter 2
Natalya Koshkina (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 2
Andrzej Kotyra (Lublin University of Technology, Poland) – Chapter 4
Iurii Krak, (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 3
Iurii Kryvonos, (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 3
Igor Molchanov, (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 4
Andrzej Smolarz (Lublin University of Technology, Poland) – Chapter 2
Batyrbek Suleimenov (Kazakh National Technical University, Kazakhstan) – Chapter 1
Vadim Tulchinskiy (Glushkov Institute of Cybernetics of NAS of Ukraine) – Chapter 1
Waldemar Wójcik, (Lublin University of Technology, Poland) – Chapters 1,3
Valeriy Zadiraka (Glushkov Institute of Cybernetic of NAS of Ukraine) – Chapter 2

Publication approved by the Rector of Lublin University of Technology

ISBN: 978-83-63569-22-8

Publisher: Lublin University of Technology
ul. Nadbystrzycka 38D, 20-618 Lublin, Poland
Realization: Lublin University of Technology Library
ul. Nadbystrzycka 36A, 20-618 Lublin, Poland
tel. (81) 538-46-59, email: wydawca@pollub.pl
www.biblioteka.pollub.pl
Printed by : TOP Agencja Reklamowa Agnieszka Łuczak
www.agencjatop.pl

Impression: 100 copies

TABLE OF CONTENTS

1. UNDERSAMPLING AND ITS APPLICATIONS

Vadim Tulchinskiy, Waldemar Wójcik, Batyrbek Suleimenov

In recent eight years, a new theory, which suggest that it may be possible to surpass the traditional limits of sampling density inspired more than a thousand papers and pulled in millions of dollars in both international and national research grants. The theory called Compressive Sensing (CS) has attracted considerable attention in applied mathematics, computer science, physics, chemistry, biology, medicine, engineering and geosciences by suggesting. CS builds upon the fundamental fact that we can represent useful signals by just a few non-zero coefficients in a suitable basis or dictionary.

While ideas around CS were deeply studied for at least half a century the key CS concept was discovered by chance [34]. In February 2004, Emmanuel Candès, then a professor at Caltech, now at Stanford, was experimenting with a badly corrupted version of an image called the Shepp-Logan phantom (Fig. 1.1). That image is a standard picture used by computer scientists and engineers to test imaging algorithms to simulate scans of computer tomography/MRI. Candès found that ℓ1 minimization completely restores the image from the noise. Candès, with the assistance of postdoc Justin Romberg, came up with what he considered to be a sketchy and incomplete theory for the observed result. He then presented it on a blackboard to a colleague at UCLA famous mathematician Terence Tao. The next evening, Tao sent a set of notes to Candès about the blackboard session. It was the basis of their first paper together [12] and the basis of what was letter called a compressed (or compressive) sensing (or sampling). In 2006, Candès' work on the topic was rewarded with the $500,000 Waterman Prize, the highest honour of the National Science Foundation. It's not hard to see why. Imagine MRI machines that take seconds to produce images that used to take up to an hour, military software that is vastly better at intercepting an adversary's communications, and sensors that can analyze distant interstellar radio waves.

In this chapter, we provide a brief review of the basic theory underlying CS. After a historical overview of classical sampling theory with attention to cases of recoverable sampling with frequency below the Nyquist rate, we begin with introducing the concept of sparsity. Then we discuss application of low-rate irregular sampling for sparse signals and introduce the thresholding method of sparse signal reconstruction from random and jittered undersampling. We then treat the central question of the CS framework: how to accurately recover a high-dimensional signal from a small set of measurements, and review performance guarantees for a variety of sparse recovery problems. We conclude with a discussion of reconstruction algorithms and applications of the compressive sensing based undersampling. An example of undersampling application for seismic modeling acceleration is examined in details.

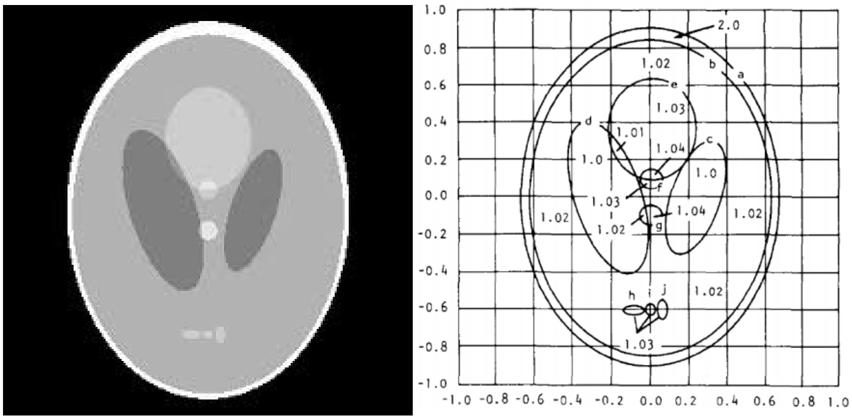


Fig. 1.1. Shepp-Logan phantom (left) and its scheme (right)

1.1. Digital signals and sampling problem

The theoretical foundation of modern Information and Communication Technologies (ICT) is digital signal processing. It is based on pioneering work of Kotelnikov [49], Nyquist [64], Shannon [67], and Whittaker [78] on sampling continuous-time band-limited signals. Their results demonstrate that signals, images, videos, and other data can be exactly recovered from a set of uniformly spaced samples taken at the so-called Nyquist rate of twice the highest frequency present in the signal of interest. On the base of this discovery, much of signal processing has moved from the analog to the digital domain and utilized the power of Moore’s law. Digitization has enabled the creation of sensing and processing systems that are more robust, flexible, cheaper and, consequently, more widely used than their analog counterparts.

An analog signal is said to be band-limited, if it has an identifiable maximum frequency in its spectrum, say, fmax Hz. There are great many real-world signals that are band-limited. For example, speech and music are always band-limited by the human sensing abilities. To process signals that are not band-limited, it is often convenient to deal with their band-limited counterparts by low-pass or band-pass filtering the signals as a pre-processing step. This step is often an integral part of a Digital Signal Processing (DSP) system. In Fig. 1.2, this step is the first function block.

Without the loss of generality, let us now consider a band-limited continuous-time signal $x(t)$ whose spectrum is within the region $0 \le \Omega \le \Omega\text{max}$ where $\Omega\text{max} = 2\pi\cdot\text{fmax}$. Suppose the signal $x(t)$ is defined for $-\infty < t < \infty$ and is sampled uniformly at $t = n\cdot Ts$, where $Ts = 1/fs$ denotes the sampling period in seconds (this means that fs is the sampling frequency) and $-\infty < n < \infty$ are integers.

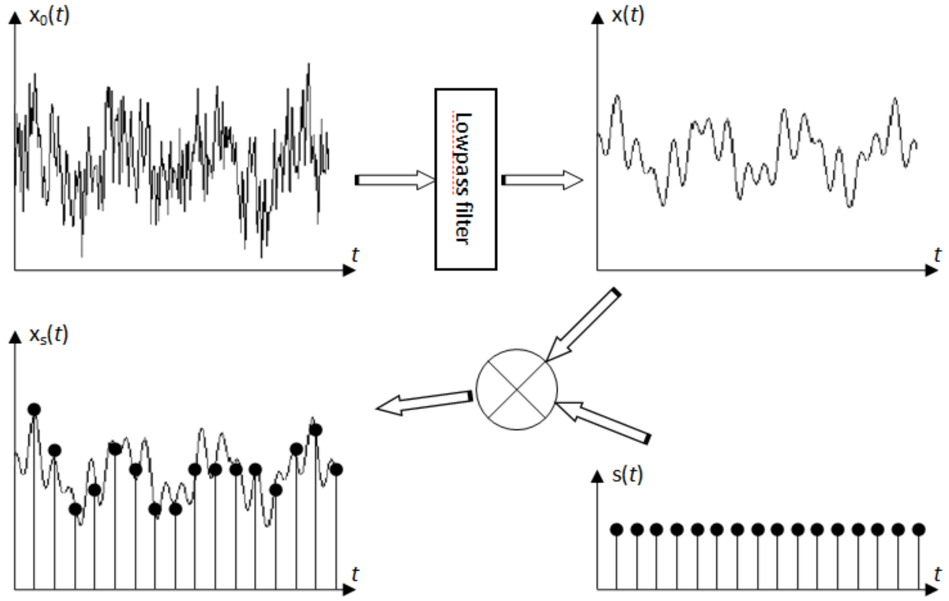


Fig. 1.2. The general regular sampling model

In Fig. 1.2, s(t) is the periodic impulse train

$$\mathrm{s}(t)=\sum_{k=-\infty}^{\infty}\delta(t-kT_s),$$

where δ(t) is the unit impulse function of Dirac, and the sampled signal is obtained by modulating s(t) with x(t) as

$$\mathrm{x}_s(t)=\mathrm{x}(t)\sum_{k=-\infty}^{\infty}\delta(t-kT_s)=\mathrm{x}(t)\cdot\mathrm{s}(t). \tag{1.1}$$

Let analyse the sampling process in the frequency domain. Applying Fourier transform to (1.1) note that the Fourier transform of a product of two functions is equal to the convolution of the Fourier transforms of these functions. The impulse train Fourier transform is known. Therefore,

$$X_s(\Omega)=X(\Omega)*S(\Omega)=X(\Omega)*\left(\frac{1}{T_s}\sum_{k=-\infty}^{\infty}\delta(\Omega-k\Omega_s)\right).$$

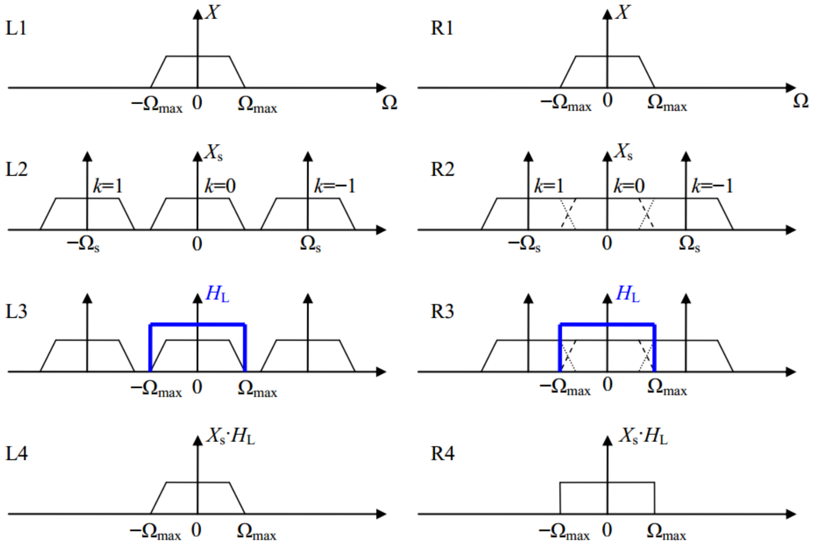


Fig. 1.3. The general scheme of analog signal spectrum reconstruction for regularly sampled model: enough sampling frequency (L), subsampling (R)

Here $\Omega_s = 2\pi/T_s$ is the angular sampling frequency in radians/sec. The formula can be further expressed by convolution between the signal spectrum and Dirac's functions:

$$X_s(\Omega)=\frac{1}{T_s}\sum_{k=-\infty}^{\infty}X(\Omega)*\delta(\Omega-k\Omega_s)=\frac{1}{T_s}\sum_{k=-\infty}^{\infty}\int_{-\infty}^{\infty}\delta(\omega-(\Omega-k\Omega_s))X(\omega)d\omega\,.$$

$$X_s(\Omega)=\frac{1}{T_s}\sum_{k=-\infty}^{\infty}X(\Omega-k\Omega_s). \qquad (1.2)$$

Equation (1.2) is important because it relates explicitly the spectrum of the sampled signal xs(t) to that of the original analog signal x(t). We see that the Fourier transform of xs(t) consists of periodically repeated copies of the Fourier transform of x(t). More specifically, (1.2) says that the copies of X(Ω) are shifted by integer multiples of the sampling frequency and then superimposed to generate the periodic Fourier transform of the impulse train of samples. Two representative cases in terms of the value of Ωmax compared with that of Ωs−Ωmax are shown in Fig. 1.3.The analog signal spectrum X(Ω) can be recovered by multiplying the spectrum of sampled signal Xs(Ω) by rectangular spectrum function

$$H_L(\Omega)=\mathrm{rect}\left(\frac{\Omega}{\Omega_{\max}}\right),\quad \mathrm{rect}(v)=\begin{cases}1, |v|\le 1\\ 0, |v|>1\end{cases}. \qquad (1.3)$$

Let the sampling frequency is sufficiently high (case L of Fig. VT3). It means that Ωs−Ωmax > Ωmax i.e., Ωs > 2Ωmax (the Nyquist threshold). In this case, the replicas of X(Ω) do not overlap. As result they can be easy separated in frequency domain:

$$X(\Omega) = X_s(\Omega) \cdot H_L(\Omega) \tag{1.4}$$

Consequently, the continuous-time signal x(t) can be recovered from the discrete signal xs(t) by ideal lowpass filtering. This is the idea of Shannon-Nyquist (aka Kotelnikov's) Sampling Theorem:

A continuous-time signal x(t) with frequencies no higher than f_{max} (in Hz) can be reconstructed from its samples $x_k = x(kT_s)$, if the samples are taken at a rate $f_s = 1/T_s$ that is greater than 2 f_{max}.

The signal reconstruction technique is known as Whittaker–Shannon interpolation formula. It's easy derived from (1.4) by transforming the spectral domain multiplication to the time domain convolution:

$$\mathrm{x}(t) = \sum_{k=-\infty}^{\infty} \mathrm{x}_s(kT_s) h_L(t - kT_s)$$

.

$$h_L(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} H_L(\omega) e^{j\omega t} d\omega = \frac{1}{2\pi}\int_{-\Omega_{\max}}^{\Omega_{\max}} e^{j\omega t} d\omega = \operatorname{sinc}\left(\frac{t}{T_s}\right), \tag{1.5}$$

$$\operatorname{sinc}(v) = \frac{\sin(\pi v)}{\pi v}.$$

$$\mathrm{x}(t) = \sum_{k=-\infty}^{\infty} \mathrm{x}_s(kT_s) \operatorname{sinc}\left(\frac{t - kT_s}{T_s}\right) = \sum_{k=-\infty}^{\infty} x_k \cdot \operatorname{sinc}\left(\frac{t - kT_s}{T_s}\right). \tag{1.6}$$

The sinc function is shown on Fig. 1.4.

The Shannon-Nyquist Sampling Theorem does not limit sampling frequency for any type of band-limited signals. There are applications where the frequencies of the continuous-time signals fall within a two side limited range $\Omega_{\min} \le \Omega \le \Omega_{\max}$, $\Omega_{\min} > 0$.Such a signal is referred to as a band-pass signal. The signal's bandwidth is defined as $\Delta\Omega = \Omega_{\max} - \Omega_{\min}$. In radio broadcasting, for instance, a relatively low frequency audio signal is modulated by a high-frequency carrier and the modulated audio becomes a band-pass signal with a narrow bandwidth. For convenience, below we assume that the highest frequency contained in a band-pass signal x(t) is a multiple of the bandwidth, i.e. $\Omega_{\max} = M \cdot \Delta\Omega$. Respectively, $\Omega_{\min} = (M - 1)\Delta\Omega$.

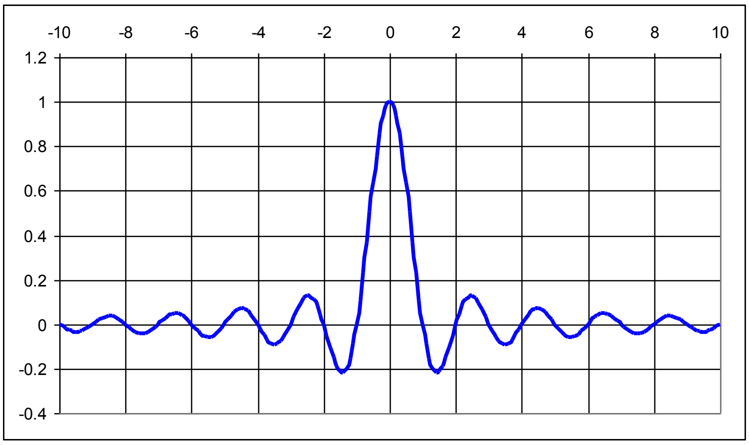


Fig. 1.4. Diagram of sinc function

Such bandpass signal can be completely restored by the sampling frequency twice of the signal's bandwidth $\Omega_s = 2\Delta\Omega$, which is M times smaller than $2\Omega_{\max}$. The idea based on band-pass filtering is expressed by Fig. 1.5.

The formal approach left the same except for another type filter. Analog of (1.3) is

$$H_B(\Omega) = \mathrm{rect}\left(\frac{\Omega}{\Omega_{\max}}\right) - \mathrm{rect}\left(\frac{\Omega}{\Omega_{\min}}\right). \tag{1.7}$$

Convolution is a linear operator. Hence (1.6) is transformed to

$$\mathrm{x}(t) = \sum_{k=-\infty}^{\infty} x_k \left(\mathrm{sinc}(f_{\max} t - k) - \mathrm{sinc}(f_{\min} t - k)\right). \tag{1.8}$$

From Fig. 1.5, we can see that the original band-pass signal x(t) can be recovered by appropriately band-pass filtering sampled signal xs(t). Note that, unlike the situation of Fig. 1.3 R in this case the sampling frequency is below the "formal" Nyquist threshold $2\Omega_{\max}$, because M is an integer greater than 1. An interpretation of this seemingly contradiction is that the bandwidth in the case of low-pass signals is $\Delta\Omega = \Omega_{\max} - 0$, thus the choice of M =1 deduces the Shannon-Nyquist Sampling Theorem in its original formulation. It should be understood that $\Omega_s = 2\Delta\Omega$ is enough for arbitrary signal under condition of both $\Omega_{\max} \le M \cdot \Delta\Omega$ and $\Omega_{\min} > (M-1)\Delta\Omega$ for some positive integer M.

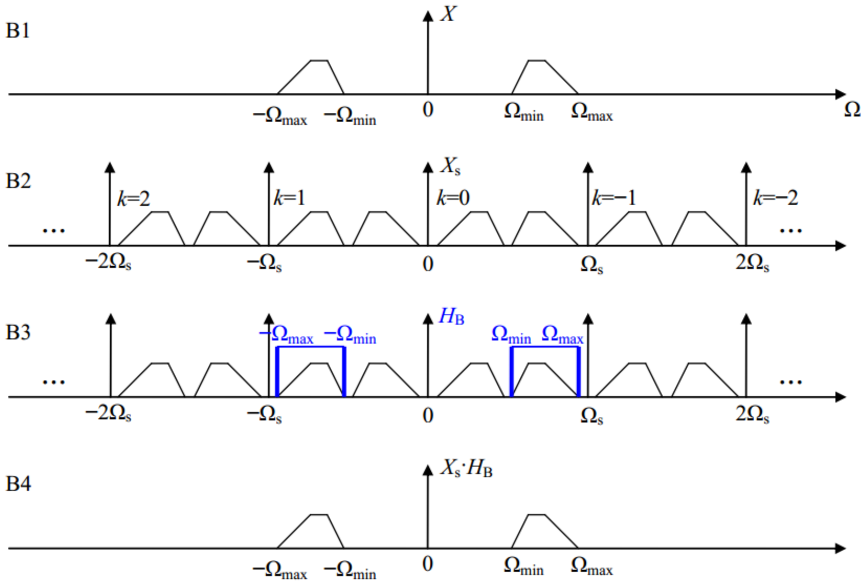

B1
X
$-\Omega_{max}$ $-\Omega_{min}$ 0 Ω_{min} Ω_{max}
Ω
B2
X_s
k=2 k=1 k=0 k=−1 k=−2
…
$-2\Omega_s$ $-\Omega_s$ 0 Ω_s $2\Omega_s$
B3
H_B
$-\Omega_{max}$ $-\Omega_{min}$ Ω_{min} Ω_{max}
$-2\Omega_s$ $-\Omega_s$ 0 Ω_s $2\Omega_s$
B4
$X_s \cdot H_B$
$-\Omega_{max}$ $-\Omega_{min}$ 0 Ω_{min} Ω_{max}

Fig. 1.5. The general scheme of bandpass analog signal spectrum reconstruction for regularly sampled model

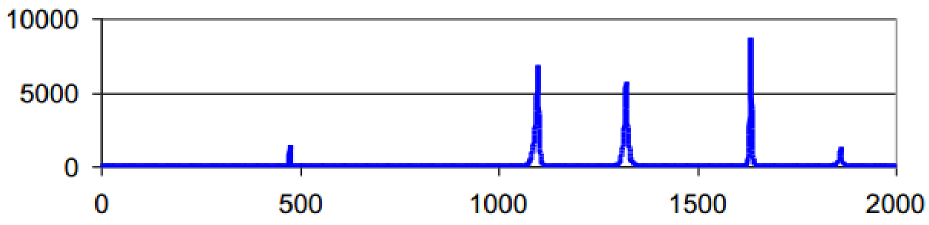


Fig. 1.6. A sparse signal spectrum sample

Is it possible to find better sampling conditions? There are some problems which deal with sets of narrow-band signals. An example is a short pulse radar system. It generates a series of short impulses which are of wide bandwidth in frequency domain. (Such dependency can be illustrated by (1.6): wider rectangular spectrum corresponds to bigger sinc argument, and respectively to shorter impulse. Dirac impulse spectrum is infinite in frequency domain.) The radar system then detects responses (reflected waves) from relatively small remote objects. Each object reflects waves of wavelengths correspondent to its size. They create narrow images in spectral domain. As result the targets can be differentiated by their spectral images for further classification. The modal frequencies of different type targets are different well more then bandwidth of each the target response. As result the reflected signal is presented by a small number of harmonics when almost all other frequencies are zero (Fig. 1.6). Such type spectra are called sparse.

Let a sparse spectrum is band-limited by $\Omega_{\max}$ and consists of N narrow-band signals. Let also the interval 0 – $\Omega_{\max}$ is divided by M equal intervals of length $\Delta\Omega = \Omega_{\max}/M$ such that each the narrow-band signal fits completely in a single interval: $\Omega_{\max}/M$. Denote the interval indices $b_n \in \{1..M\}$. The method of Fig. 1.5 used to decrease sampling frequency of band-pass signal can be applied for sparse signal as well under conditions

$$b_i + mM \neq M - b_j \text{ and } b_i + mM \neq b_j$$

for $\forall i \neq j, m < N/M + 1$. The condition means no two signals share an interval.

Fig. 1.7 illustrates the sparse sampling scheme appropriate for complete reconstruction of a sparse spectrum.

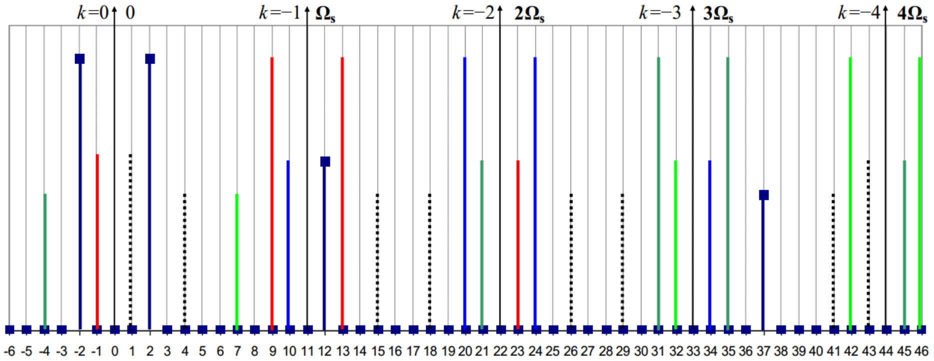


Fig. 1.7. A spectrum of sampled sparse signal: $N = 40$, $M = 11$, $b_n = \{2,12,37\}$. Equal color solid lines represent a (shifted) copy of the input analog signal spectrum. Among them the dark blue marked lines correspond to the target non shifted spectrum

One needs ideal band filter of 3 pairs of rectangles to reconstruct initial frequency limited sparse spectrum from the sampled spectrum of Fig. 1.7. In theory the filtering can be expressed as multiplication by a combination of rect functions in the spectral domain as well as by convolution with a combination of sinc functions in the time domain. In practice, however, the last approach is less suitable for sparse signals than for ordinary wide-band signals because the approximate nature of known filter implementations (see http://en.wikipedia.org/wiki/Electronic_filter) contradicts to high precision requirements.

The above problem can be solved for time-limited signals by a technique known as zero padding. Zero padding in the time domain is used extensively in practice to improve heavily interpolated spectra.

The sampling approach can be applied similarly to the dual frequency space. Let analyze a time limited continues signal which is nonzero only over some finite duration: x(t)=0 for $|t| > T_0$. Let compute a sampled spectrum of whole

Integral Fourier Transform of the signal: $X(\Omega)\cdot\sum\delta(2\pi f_0 n)$. Its inverse Fourier transform produces a periodical signal in the time domain. Under condition of $f_0 > 1/2T_0$ the periodical signal time limiting is enough to reconstruct the initial signal with arbitrary precision. The method is equivalent to use of low-pass filter for reconstruction of a band-limited spectrum in frequency domain according to Fig. 1.3.

Combination of sampling in both time and frequency domains describes useful case of real world digital signal processing: time limited sampled signals:

$$X_k = \sum_n x_n e^{-j\omega_k t_n}, \quad x_n = \frac{1}{N}\sum_k X_k e^{j\omega_k t_n}, \quad \omega_k = \frac{2\pi k}{T_0}, \quad t_n = T_s n. \qquad (1.10)$$

Let assume the number of samples in frequency to be equal to the number of samples in the temporal domain, that is *N*. Let $T_0 = NT_s$. This is not a necessary condition, but it simplifies the notation. In such case the Integral Fourier Transform is reduced to well-known finite Discrete Fourier Transform (DFT):

$$X_k = \sum_{n=0}^{N-1} x_n \exp\left(-j\frac{2\pi kn}{N}\right), \quad x_n = \frac{1}{N}\sum_{n=0}^{N-1} X_k \exp\left(j\frac{2\pi kn}{N}\right). \qquad (1.11)$$

Zero padding technique doubles the number of samples used in DFT formulas of a signal without affecting the signal itself. As result *N* zeros are attached to the signal tail:

$$X_k = \sum_{n=0}^{2N-1} x_n \exp\left(-j\frac{2\pi kn}{2N}\right) = \sum_{n=0}^{N-1} x_n \exp\left(-j\frac{2\pi(k/2)n}{N}\right). \qquad (1.12)$$

Even *k* of (1.12) generates spectral coefficient equal to the coefficient of a similar frequency *k*/2 from (1.11). Odd *k* is high quality interpolation of the spectra in frequency domain. The procedure can be easy iterated or extended to produce *S* times frequency interpolation by attaching *N*(*S*-1) zeros out the real time limits.

It’s incorrect to assume that zero-padding in the time domain yields higher spectral resolution in the frequency domain. Resolution in signal processing refers to the ability to differentiate closely spaced features. The usual way to increase spectral resolution is to take a longer DFT without zero padding – i.e., look at more data. In other words, zero-padding in one domain corresponds to a higher interpolation density in the other domain – not a higher resolution. However, using this approach together with aliasing shift by inserting zero samples at the locations where the interpolated values are desired, Gülünay and Chambers developed a Generalized F-K Trace Interpolation Method [37]. The method computes a frequency domain filter which suppresses linear aliasing artifacts. The Gülünay method is related to prediction error filters (PEF), which can handle aliased events [69]. PEF are designed so that the interpolation error is white noise.

Note that the time-limited assumption directly contradicts to common assumption of periodic extension of a signal. There is no spectral energy, in principle, between the harmonics of a periodic signal, and a periodic signal cannot be time-limited unless it is the zero signal. On the other hand, the interpolation of a time-limited signal's spectrum is nonzero almost everywhere between the original spectral samples. Thus, zero-padding is often used when analyzing data from a non-periodic signal in blocks, and each block, or frame, is treated as a finite-duration signal which can be zero-padded on either side with any number of zeros. In summary, the use of zero-padding corresponds to the time-limited assumption for the data frame, and more zero-padding yields denser interpolation of the frequency samples around the unit circle. Ability to better differentiate narrow-band signals is, hence, limited by the spectrum dissipation because of introduced time-limit.

More fundamental restriction of classical regular sampling approach for sparse signals is the necessity to know frequencies of narrow-band signal components a priory. For the described radar example this means a collection of all possible responses for all possible targets. Moreover, such collection is insufficient for reliable fitting a sparse sampling. One can filter input data (Fig. 1.2) to guarantee the frequency limits for provided band-limited signal. But similar approach is inappropriate for narrow-band signals both because of mentioned implementation obstacles and because the band width is usually comparable to the model uncertainty and observational error. As result the preliminary data filtering can remove or suppress desired signal instead of noise. Otherwise, bypass of the preliminary data filtering results in aliasing.

1.2. Irregular sampling for reconstruction of sparse signals

Modern approach to better sampling of sparse signals is based on irregular (random and jittered) undersampling. It reduces non-aliasing signal reconstruction problem to simple de-noising.

To understand the irregular sampling one need to generalize the definition and the computation of the discrete Fourier transform from the regular sampling to the irregular sampling domain. In the general case, the transform can be similar to DFT given by (1.10), taking into consideration that the samples can be taken at irregular intervals both in time and/or in frequency. The practice, however, enables a more restricted case, which is the case where the samples are irregularly taken in the time domain $t_n \neq nT_s$ but regularly sampled in the frequency domain. That is to say that the samples X_k of the irregular Fourier transform are taken at multiples of a quantity $\Delta\omega = 2\pi/T_0$, which is a fixed quantity in the spectral domain. The extension from regular to irregular sampling, therefore, depends on the duration of the signal x(t) and not on the fact that the samples are taken at regular or irregular intervals. The simplest approximation of Discrete Fourier Transform for irregular samples looks as follows:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j\omega_k t_n} = \sum_{n=0}^{N-1} x_n e^{-jk\Delta\omega t_n} \ . \tag{1.13}$$

Despite obvious analogy to DFT the formula (1.13) does not create conditions for good reconstruction of the sampled signal. This issue can be understood from interpretation of the original reconstruction formula (1.6) as decomposition by a basis of sinc functions:

$$\mathrm{x}(t) = \sum_{k=-\infty}^{\infty} x_s(kT_s)\varphi_k(t), \quad \varphi_k(t) = \mathrm{sinc}\left(\frac{t}{T_s} - k\right).$$

The basis functions φ_k and φ_l are orthogonal as long as k, l are integers:

$$\langle\varphi_k, \varphi_l\rangle = \int_{-\infty}^{\infty} \mathrm{sinc}\left(\frac{t}{T_s} - k\right)\mathrm{sinc}\left(\frac{t}{T_s} - l\right)dt = \mathrm{sinc}(k-l) = \delta_{k,l}\,, \tag{1.14}$$

where $\delta_{k,l}$ is another representation of the Kronecker delta function: $\delta_{k,l} = \delta(k-l)$. The orthogonality property greatly simplifies the data reconstruction. It assures that on a regular grid ($k, l \in \mathrm{Z}$), the sinc function has the weight 1 on the original data position and zero weight on all other integer locations (Fig. 1.4).

The sinc functions also satisfy the unity condition; i.e., for any real number v:

$$\sum_{k=-\infty}^{\infty} \mathrm{sinc}(v-k) = 1\,. \tag{1.15}$$

On an arbitrary irregular grid, the orthogonality condition (1.14) and unity condition (1.15) do not hold.

It is easier to rebuild the unity condition on an irregular grid. In engineering, attention was focused on this unity condition, which does not hold on an irregular grid; i.e., for a general reconstruction base k enumerates the (infinite) index set of irregular data samples N_{p}. Analogy to (1.15) does not hold. Usually, one can normalize the unity condition with some weighting applied to the data:

$$\mathrm{x}(t) = \sum_k w(t,k)\mathrm{x}_s(kT_s)\varphi_k(t) \Big/ \sum_k w(t,k)\varphi_k(t)\,. \tag{1.16}$$

If one introduces a B-spline interpolant for φ_k, equation (1.13) becomes the basic formula for non-uniform rational B-spline (NURBS), first proposed by Versprille [75], and widely applied in computer graphics for representing free-form curves and surfaces.

Less general but common method of normalized Fourier summation rebuilds the unity condition by introducing simple weights, which are proportional to time intervals appeared between serial samples. The weights express the rectangle method (also called the midpoint or mid-ordinate rule) used to compute an approximation to a definite integral by finding the area of a collection of rectangles whose heights are determined by the values of the function. To attenuate the leakage of Fourier coefficients the forward Non-uniform Discrete Fourier Transform (NDFT) is defined:

$$X_k = \sum_{n=0}^{N-1} x_n \Delta t_n e^{-j\Delta\omega k t_n} = \sum_{n=0}^{N-1} x_n \Delta t_n \exp\left(-j\frac{2\pi k t_n}{NT_0}\right), \tag{1.17}$$

where $\Delta t_0 = \Delta t_{N-1} = \frac{1}{2}\left(T_0 - t_{N-1} + t_0\right)$, for other n: $\Delta t_n = \frac{1}{2}\left(t_{n+1} - t_{n-1}\right)$. Inverse NDFT is basically the same as inverse DFT.

The NDFT coefficients equal the DFT coefficients convolved with the NDFT of the sampling weights [33]. This follows from substituting of (1.11), the inverse NDFT, in equation (1.17):

$$\begin{aligned} X_k &= \sum_{n=0}^{N-1}\left(\frac{1}{N}\sum_{m=0}^{N-1}\tilde{X}_m \exp\left(j\frac{2\pi m t_n}{NT_0}\right)\right)\Delta t_n \exp\left(-j\frac{2\pi k t_n}{NT_0}\right) = \\ &= \frac{1}{N}\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\Delta t_n \exp\left(j\frac{2\pi t_n}{NT_0}(m-k)\right)\tilde{X}_m \end{aligned},$$

$$X = \tilde{F} * \tilde{X}\text{, for } \tilde{F}_m = \frac{1}{N}\sum_{n=0}^{N-1}\Delta t_n \exp\left(j\frac{2\pi m t_n}{NT_0}\right). \tag{1.18}$$

Here X_k denotes NDFT spectral coefficients, $\tilde{X}_m$ denotes the correspondent DFT coefficients, and $\tilde{F}_m$ is a point-spread function (PSF). The distortion in the NDFT is determined by the PSF, which in turn is influenced by the weights. As the PSF approaches a delta function, the NDFT becomes more like the DFT.

A reconstruction method based on equation (1.18) satisfies the unity condition and can rebuild a smooth free surface. However, it does not meet the orthogonality condition. Therefore, the reconstructed data do not fit the original measurements on the irregular grid.

Furthermore, the concept of Nyquist frequency does not explicitly exist. For practical implementation, one needs to cut off at some maximum frequency. This will lead to a sinc function centered at each point of the original irregularly sampled grid. The uniform sinc function can take nonzero values at the location of the other samples, which might not be integers. Thus, using the uniform sinc function for irregularly sampled data reconstruction will result in an incorrect interpolation because it violates the orthogonality condition. A simple synthetic test will help us to understand this phenomenon. Recall (1.1) representation of a sampled signal spectrum as the convolution of continues signal spectrum with a sampling train spectrum. The last one determined behavior of the sampled signal. Fig. 1.8 captured from [81] shows randomized train spectra to compare them with uniform one.

In the case of uniform sampling with missing samples, aliases still occur in the Fourier domain, but now the periodicity is determined by the smallest sampling interval (Figure 1.8c–f). Random sampling can be thought of as uniform sampling with missing samples on a very finely sampled underlying grid. The aliases in the Fourier domain are then so widely spaced that the aliasing effect is effectively absent (Figure 1.8g–j). Another effect of non-uniform sampling is that spectrum is no longer a perfect spike train but contains artifacts between its spikes. These artifacts indicate how much a single Fourier coefficient is distorted by shifting from (1.14), (1.15). Convolution with this noisy spike train distorts the Fourier coefficients.

Figures 1.8c and 1.8e show the spectra of corrupted uniform sampling where positions have been omitted such that, respectively, about ¾ and ¼ samples remain. Figures 1.8g and 1.8j show similarly rare spectra for random sampling. Here, the spike series that causes the aliases of the spectrum is absent. In practice, sampling is neither always uniform with missing samples nor fully random. Starting from uniform sampling with or without missing positions, the sampling locations can be perturbed more and more to yield increasingly non-uniform sampling patterns. As sampling of data becomes more non-uniform, the aliasing becomes more and more diffuse until it disappears altogether for random sampling.

The relation between irregularities in data sampling and the non-orthogonality of Fourier basis on the irregular grid identifies the fundamental problem of traditional data regularization [73]. Orthogonalization techniques such as Gram-Schmidt process [80] were used to improve spectra corrupted by irregular sampling. The main contribution of CS is the new light shed on the favorable recovery conditions. By dissipating aliasing spikes the random sampling creates alternative conditions for arbitrary precise reconstruction of sparse signals. The Nyquist threshold disappears and reconstruction procedure of undersampled data win in both simplicity and performance.

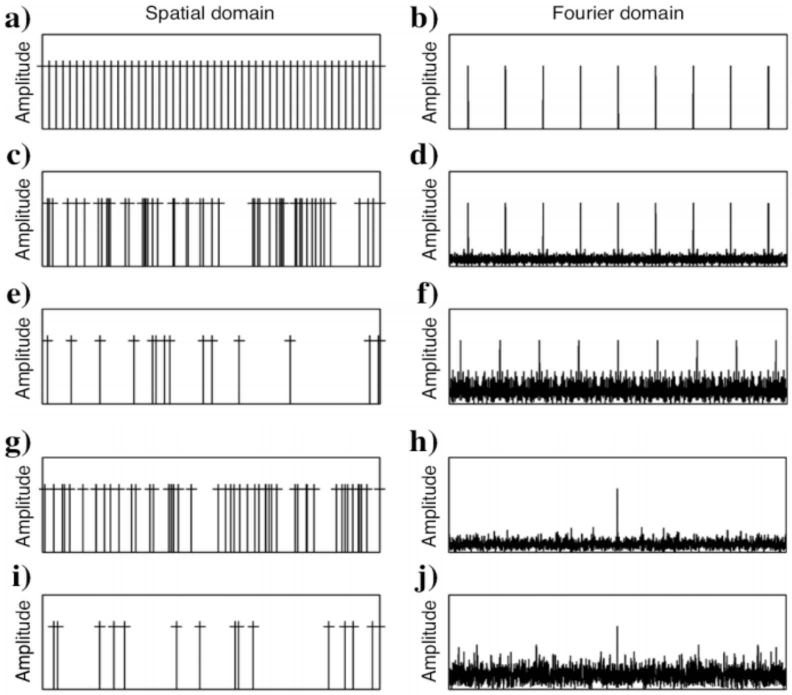


Fig. 1.8. A schematic sampling train in the spatial domain: uniform sampling (a), randomly missed samples (c and e) with 50 and 15 positions; random sampling (g and i) with 50 and 15 positions. The figures in the right column (b, d, f, h, and j) show the NDFT spectra of the figures in the left column. The central spike denotes Ω = 0, and all other spikes are its replications that are the cause of aliasing. Note the lack of these replications for random sampling in (h and j) [81]

Figure 1.9 from [44] shows the superposition of three cosine functions. This signal is sparse in the Fourier domain and is sampled regularly above the Nyquist rate. Its amplitude spectrum is plotted in Figure 1.9b. When the signal is undersampled randomly threefold according to a discrete uniform distribution as in 1.9c, its amplitude spectrum, plotted in Figure 1.9d, is corrupted by artifacts that look like additive incoherent random noise. In this case, the significant coefficients of the to-be-recovered signal remain above the noise level. These coefficients can be detected with a denoising technique that promotes sparsity, e.g., simple thresholding (dashed line in Figure 1.9d and f), and recovered exactly by an amplitude-matching procedure to fit the acquired data.

This experiment illustrates a favorable recovery from severely undersampled data points of a signal that is sparse in the spectral domain. When the original signal is undersampled regularly threefold (Fig. 1.9e), the undersampling artifacts coherently interfere, giving rise to well visible aliases that look like the original signal components and so can’t be easy removed (Fig. 1.9f). In this case, the sparsity-promoting recovery scheme might fail because the to-be-recovered signal components and the aliases are both sparse in the spectral domain. This example suggests that random undersampling according to a discrete uniform distribution is more favorable than regular undersampling for reconstruction algorithms that promote sparsity in the spectral domain. In general terms, these observations hint at undersampling schemes that lead to more favorable recovery conditions.

Consider the following linear forward model for the recovery problem

$$\mathbf{y} = \mathbf{\Phi f}_0 , \tag{1.19}$$

where $\mathbf{y} \in \mathrm{R}^n$ represents real vector of the acquired data; $\mathbf{f}_0 \in \mathrm{R}^N$ with $N >> n$ the unaliased signal to be recovered, i.e., the model; and $\mathbf{\Phi} \in \mathrm{R}^{n \times N}$ the restriction operator that collects the acquired samples from the model.

Assume that $\mathbf{f}_0$ has a sparse representation as a complex vector $\mathbf{x}_0 \in \mathrm{C}^N$ in some known transform domain $\mathbf{\Psi}$. Equation (1.19) can be reformulated as

$$\mathbf{y} = \mathbf{Ax}_0 ,\ \mathbf{A} = \mathbf{\Phi \Psi}^{\mathrm{T}} \tag{1.20}$$

where the superscript T represents the conjugate transpose. As a result, the sparsity of $\mathbf{x}_0$ can be used to overcome the singular nature of $\mathbf{A}$ when estimating $\mathbf{f}_0$ from **y**. After sparsity-promoting inversion, the recovered signal is given by

$$\tilde{\mathbf{f}} = \mathbf{\Psi}^{\mathrm{T}} \tilde{\mathbf{x}} \text{, with } \tilde{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ such that } \mathbf{y} = \mathbf{Ax} . \tag{1.21}$$

In these expressions, the tilde represents estimated quantities, and $\|\mathbf{x}\|_1$ is ℓ_1 norm, i.e. the sum of absolute values of **x**' components x_i. Commonly $\|\mathbf{x}\|_P = \left(\sum |x_i|^P\right)^{1/P}$ denotes ℓ_P norm for $1 \le P \le \infty$ and a quasi-norm for $0 < P < 1$. The concept is generalized for $P = \infty$: $\|\mathbf{x}\|_\infty = \max(|x_i|)$. The term "$\ell_0$ norm" is also widely used despite $\|\mathbf{x}\|_0$ defined as the number of non-zero components x_i is neither norm nor quasi norm. The set of **x**' non-zero components is called support of **x** and denoted by $\mathrm{supp}(\mathbf{x}) = \{i \in \mathrm{Z} | x_i \ne 0 \wedge 0 \le i \le N\}$.

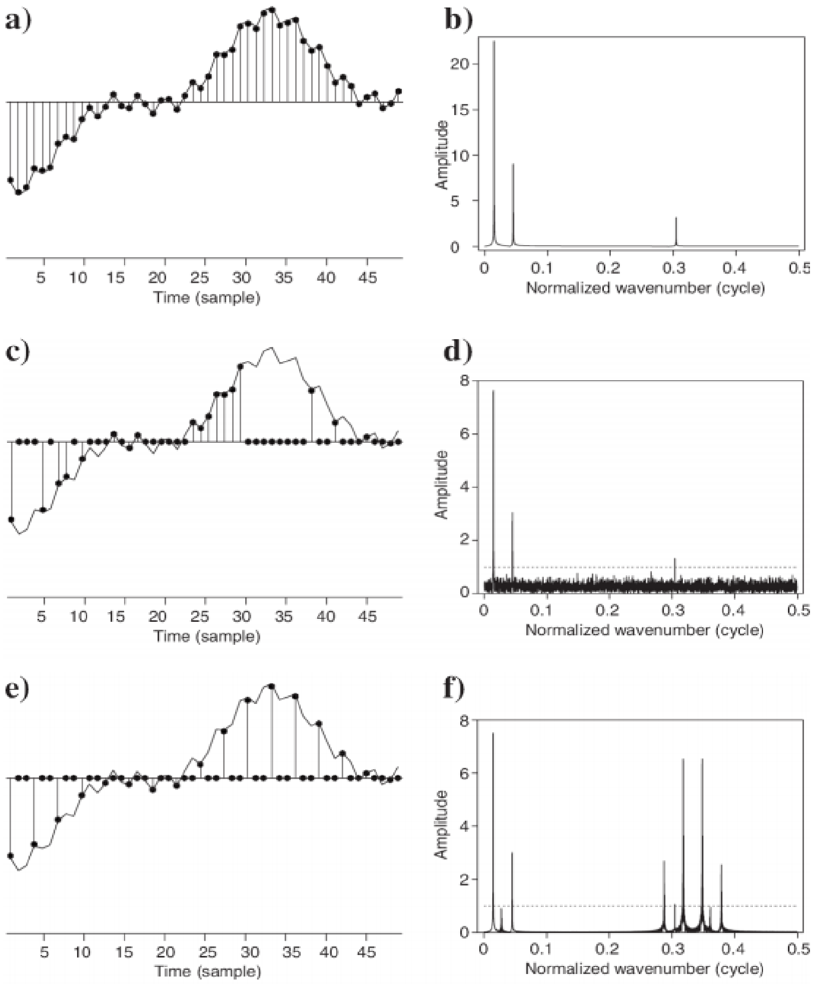


Fig. 1.9. Different sampling schemes for a superposition of three cosine functions: regularly sampling above Nyquist rate (a); random undersamping (c); and regular undersampling (e); the respective amplitude spectra (b, d, and f). Unlike aliases, the undersampling artifacts from random undersampling can be removed easily by using a standard denoising technique that promotes sparsity. E.g., thresholding (dashed line) recovers the original signal [44]

Consider the following linear forward model for the recovery problem

$$\mathbf{y} = \mathbf{\Phi} \mathbf{f}_0 , \tag{1.19}$$

where $\mathbf{y} \in \mathrm{R}^n$ represents real vector of the acquired data; $\mathbf{f}_0 \in \mathrm{R}^N$ with $N >> n$ the unaliased signal to be recovered, i.e., the model; and $\mathbf{\Phi} \in \mathrm{R}^{n \times N}$ the restriction operator that collects the acquired samples from the model.

Assume that $\mathbf{f}_0$ has a sparse representation as a complex vector $\mathbf{x}_0 \in \mathrm{C}^N$ in some known transform domain $\boldsymbol{\Psi}$. Equation (1.19) can be reformulated as

$$\mathbf{y} = \mathbf{A}\mathbf{x}_0,\ \mathbf{A} = \boldsymbol{\Phi}\boldsymbol{\Psi}^{\mathrm{T}} \tag{1.20}$$

where the superscript T represents the conjugate transpose. As a result, the sparsity of $\mathbf{x}_0$ can be used to overcome the singular nature of $\mathbf{A}$ when estimating $\mathbf{f}_0$ from **y**. After sparsity-promoting inversion, the recovered signal is given by

$$\tilde{\mathbf{f}} = \boldsymbol{\Psi}^{\mathrm{T}}\tilde{\mathbf{x}},\ \text{with } \tilde{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ such that } \mathbf{y} = \mathbf{A}\mathbf{x}. \tag{1.21}$$

In these expressions, the tilde represents estimated quantities, and $\|\mathbf{x}\|_1$ is ℓ_1 norm, i.e. the sum of absolute values of **x**' components x_i. Commonly $\|\mathbf{x}\|_P = \left(\sum |x_i|^P\right)^{1/P}$ denotes ℓ_P norm for $1 \le P \le \infty$ and a quasi norm for $0 < P < 1$. The concept is generalized for $P = \infty$: $\|\mathbf{x}\|_\infty = \max(|x_i|)$. The term "$\ell_0$ norm" is also widely used despite $\|\mathbf{x}\|_0$ defined as the number of non-zero components x_i is neither norm nor quasi norm. The set of **x**' non-zero components is called support of **x** and denoted by $\mathrm{supp}(\mathbf{x}) = \{i \in \mathrm{Z} | x_i \ne 0 \wedge 0 \le i \le N\}$.

The strong relation between the ℓ_1 norm and sparsity initially expressed by ℓ_0 is the first and important result of CS theory [11]. Among all possible solutions (n<<N) of the severely underdetermined system of linear equations (1.20), the optimization problem in equation (1.21) finds a sparse or, under certain conditions, the sparsest [29] possible solution that explains the data. Following [74] and [44], we define the matrix $\mathbf{L} = \mathbf{A}^{\mathrm{T}}\mathbf{A} - \alpha\mathbf{I}$ to study the undersampling artifacts $\mathbf{z} = \mathbf{L}\mathbf{x}_0$. The matrix **I** is the identity matrix, and the parameter α is a scaling factor such that diag(**L**)=0. For more general problems and particularly in the field of digital communications, these undersampling artifacts **z** are referred to as multiple-access interference (MAI).

According to the CS theory [8, 26], which will be discussed in the next section, the solution $\tilde{\mathbf{x}}$ in equation (1.21) and $\mathbf{x}_0$ coincide when two conditions are met:

- $\mathbf{x}_0$ is sufficiently sparse, i.e., $\mathbf{x}_0$ has few nonzero entries, and
- the undersampling artifacts are incoherent, i.e., **z** does not contain coherent energy.

The first condition of sparsity requires that the energy of f0 be well concentrated in the sparsifying domain. The second condition of incoherent random undersampling artifacts involves the study of the sparsifying transform $\mathbf{\Psi}$ in conjunction with the restriction operator $\mathbf{\Phi}$. Intuitively, it requires that the artifacts z introduced by undersampling the original signal f0 are not sparse in the $\mathbf{\Psi}$ domain. When this condition on z is not met, sparsity alone is no longer an effective prior information to solve the recovery problem.

When $\mathbf{\Phi}$ keeps all of the data points of f0, i.e., $\mathbf{\Phi} = \mathbf{I}$, the matrix ATA is the identity matrix, and no spectral leakage occurs. This property holds for any orthonormal sparsifying transform including the DFT sampled above Nyquist rate.

When $\mathbf{\Phi}$ corresponds to a regular undersampling scheme, ATA is not diagonal. It also has several nonzero off-diagonals. These off-diagonals create aliases, i.e., undersampling artifacts that are the superposition of circular-shifted versions of the original spectrum. Because x0 is assumed to be sparse, these aliases are sparse as well. Therefore, they are also likely to enter in the solution $\tilde{\mathbf{x}}$ during sparsity-promoting inversion. Because the ℓ1 norm cannot efficiently discriminate the original spectrum from its aliases, regular undersampling is the most challenging case for recovery.

When $\mathbf{\Phi}$ corresponds to a random undersampling according to a discrete uniform distribution, the situation is completely different. The matrix ATA is dense, and the convolution matrix L is a random matrix. As result

$$\mathbf{A}^{\mathrm{T}}\mathbf{y} = \mathbf{A}^{\mathrm{T}}\mathbf{A}\mathbf{x}_0 \approx \alpha\mathbf{x}_0 + \hat{\mathbf{n}}, \tag{1.22}$$

where the spectral leakage is approximated by additive white Gaussian noise $\hat{\mathbf{n}}$.

For infinitely large systems [26], this approximation becomes an equality. Because of this property, the recovery problem turns into a much simpler denoising problem, followed by a correction for the amplitudes. In (1.20) the acquired data **y** are noise free and the noise $\hat{\mathbf{n}}$ in equation (1.22) comes only from the underdeterminedness of the system. In other words, random undersampling according to a discrete uniform distribution spreads the energy of spectral leakage across the spectral domain, turning the noise-free underdetermined problem (1.20) into a noisy well-determined problem (1.22) whose solution can be recovered by solving equation (1.21). This observation first was reported by Donoho et al. in [30] corresponds to Figure 1.9.

As shown, random undersampling according to a discrete uniform distribution creates favorable recovery conditions for a reconstruction procedure that promotes sparsity in the Fourier domain. However, NDFT, which is the basic procedure for random undersampling, looses precision in presence of abnormally big gaps Δt_n (1.17). Consequently, undersampling schemes with control on the maximum gap become more attractive. Hennenfent and Herrmann have proposed the jittered sampling scheme [44] which combines randomness with the maximum gap restriction. The jittered scheme is controlled by two parameters: the undersampling factor γ which defines the cell size of a regular coarse grid and the jitter parameter $0 \le \xi \le \gamma$ which determines the size of the perturbation around the coarse grid cell centers. The perturbation is expressed by the discrete random variables ε_n independently and identically distributed (IID) according to a uniform distribution on the interval between $-\xi/2$ and $\xi/2$:

$$t_n = \left(n\gamma + \frac{\gamma}{2} + \varepsilon_n \right) \frac{T_0}{N}. \tag{1.23}$$

The semi-regular jittered scheme (Fig. 1.10) perturbs regular scheme just by bypassing some positions in the observation grid.

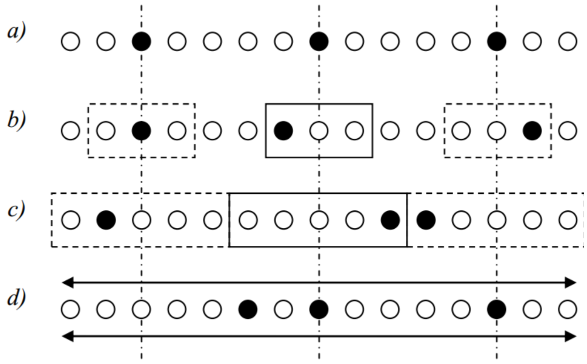


Fig. 1.10. Semi-regular jittering: regular undersampling $\gamma = 5, \xi = 0$ (a); jittered undersampling $\gamma = 5, \xi = 3$ (b); optimally jittered undersampling $\gamma = 5, \xi = 5$ (c); random undersampling $\gamma = 5, \xi = N$ (d). Black circles denote recorded samples, white circles denote omitted samples

Each sample time belongs to $\{mT_s\}$:

$$t_n = \left(n\gamma + \left\lfloor \frac{\gamma}{2} \right\rfloor + \varepsilon_n \right) \frac{T_0}{N}, \; \varepsilon_n \in Z \cap \left[-\frac{\xi}{2}; \frac{\xi}{2} \right] \text{ are uniformly IID.} \tag{1.24}$$

The case of semi-regular jittered sampling was discussed in [44] and its MAI properties have been proven.

Randomly recorded sparse signals create favorable conditions for reconstruction from a fewer number of samples than it is usually assumed. For such cases sampling rate can be kept far below the Nyquist threshold without additional knowledge about the signal structure except it’s sparse in spectral domain. The reconstruction procedure appeared to be unexpectedly simple: see Fig. 1.11 from [54].

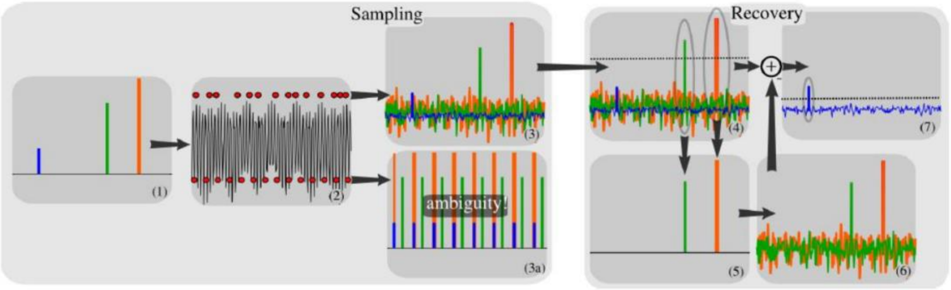


Fig. 1.11. from [54]. Iterative reconstruction procedure. A sparse signal (1) is 8-fold undersampled in time domain (2). Equispaced undersampling results in signal aliasing (3a) that cannot be recovered. Pseudo-random undersampling results in incoherent interference (3). Some strong signal components stick above the interference level, are detected and recovered by thresholding (4 and 5). The interference of these components is computed (6) and subtracted (7), thus lowering the total interference level and enabling recovery of weaker components

Similar improvements in density of observation are obtained for many other sparse representations, e.g., more local windowed Fourier [81], wavelet and curvelet [44] transforms. These wonders are indirect results incited by the new compressive sensing, CS, paradigm. Let discuss the paradigm specifically.

1.3. Introduction in Theory of Compressive Sensing

Many types of signals or images can be well approximated by a sparse expansion in terms of a suitable basis, that is, by only a small number of non-zero coefficients. This is the key to the efficiency of many lossy compression techniques such as JPEG, MP3 etc. based on transform coding. The transform coding schemes exploit signal compressibility by storing only the largest basis coefficients. When reconstructing the signal the non-stored coefficients are simply set to zero.

This is certainly a reasonable strategy when full information of the signal is available. However, when the signal first has to be acquired by a somewhat costly, lengthy or otherwise difficult measurement (sensing) procedure, this seems to be a waste of resources: first, large efforts are spent in order to obtain full information on the signal, and afterwards most of the information is thrown away at the compression stage. This is the fundamental idea behind CS: rather than first sampling at a high rate and then compressing the sampled data, we would like to find ways to directly sense the data in a compressed form, i.e., at a lower sampling rate. The field of CS grew out of the work of Candés, Romberg, and Tao and of Donoho (2005-2006), who showed that a finite-dimensional signal having a sparse or compressible representation can be recovered from a small set of linear, nonadaptive measurements. The design of these measurement schemes and their extensions to practical data models and acquisition systems are central challenges in the field of CS.

While this idea has only recently gained significant attraction in the signal processing community, there have been hints in this direction dating back as far as the eighteenth century. In 1795, baron de Prony proposed an algorithm for the estimation of the parameters associated with a small number of complex exponentials sampled in the presence of noise [65]. The next theoretical leap came in the early 1900's, when Carathéodory showed that a positive linear combination of any *k* sinusoids is uniquely determined by its value at *t*=0 and at any other 2*k* points in time [16]. This represents far fewer samples than the number of Nyquist rate samples when *k* is small and the range of possible frequencies is large. In the 1990's, this work was generalized by George, Gorodnitsky, and Rao, who studied sparsity in biomagnetic imaging [42]. Simultaneously, Bresler and Feng proposed a sampling scheme for acquiring certain classes of signals consisting of *k* components with nonzero bandwidth (as opposed to pure sinusoids) under restrictions on the possible spectral supports, although exact recovery was not guaranteed in general [7]. In the early 2000's Blu, Marziliano, and Vetterli developed sampling methods for certain classes of parametric signals that are governed by only *k* parameters, showing that these signals can be sampled and recovered from just 2*k* samples [76].

The general ℓ_0-problem of finding the sparsest representation / approximation in terms of the given dictionary turns out to be NP-hard [56]. Greedy strategies such as Matching Pursuit algorithms [56], FOCUSS [43] and ℓ_1-minimization [19] were subsequently introduced as tractable alternatives. Multiple references to studies of conditions under which greedy methods recover the sparsest solutions are listed in [35].

The third related research area is Information Based Complexity where Kashin [45], Gluskin and Garnaev [41, 38] sharply bounded both the Gelfand widths and the Kolmogorov widths of ℓ_1-ball in R^n. These widths are related to optimal recovery error which is defined as the maximal reconstruction error for the "best" sampling method and the "best" recovery method.

Ill-conditioned or underdetermined linear inverse problems of type (1.19), (1.20) arise in multiple applications. One must apply additional regularizing constraints in order to obtain interesting or useful solutions. Tykhonov regularization, the classical device for solving linear inverse problems, controls the energy (i.e., the Euclidean norm) of the unknown vector:

$$\begin{aligned}\mathbf{Ax}=\mathbf{y}\rightarrow \text{find}\,\arg\min\left|\mathbf{Ax}-\mathbf{y}\right|^2+\lambda\left|\mathbf{x}-\mathbf{x}_0\right|^2\rightarrow \\ \rightarrow\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}+\lambda\mathbf{I}\right)\mathbf{x}=\mathbf{A}^{\mathrm{T}}\mathbf{y}+\lambda\mathbf{x}_0\end{aligned}, \tag{1.25}$$

where $\mathbf{x}_0$ is a priory target and $\lambda\geq 0$ is regularization parameter. CS refers to so called "sparse approximation problem" where regularization condition is: to find the sparsest **x** such that $\mathbf{Ax}=\mathbf{y}$.

Suppose that $\mathbf{A} \in \mathrm{R}^{n \times N}$ is a real matrix whose N columns have unit Euclidean norm:

$$\left\| a_j \right\|_2 = 1 . \tag{1.26}$$

(The normalization does not compromise generality.) This matrix is often referred to as a dictionary. The columns of the matrix are “entries” in the dictionary, and a column submatrix is called a subdictionary. A vector **x** is called *s*-sparse vector when the number of its non-zero components $\|\mathbf{x}\|_0 \leq s$. We call **x** a representation of the signal **y** with respect to the dictionary $\mathbf{A}$. The most basic problem of CS is to produce a sparsest representation of an observed signal **y**:

$$\arg\min \|\mathbf{x}\|_0 \text{ subject to } \mathbf{Ax} = \mathbf{y} . \tag{1.27}$$

[13] highlights the principle of sparsity by saying that the “information rate” of a continuous-time signal may be much smaller than that suggested by its bandwidth. A discrete-time signal depends on a number of degrees of freedom which is relatively much smaller than its (finite) length. In other words the dictionary is over-complete.

The key notions in the development of current CS theory are sparsity and *incoherence* [13]. Suppose we are given a pair of orthonormal bases $\mathbf{\Phi}, \mathbf{\Psi} \in \mathrm{R}^{N \times N}$. (The first basis $\mathbf{\Phi}$ is then used for sensing the object **y** as in (1.19) and the second is used to represent $\mathbf{f}_0$ as in (1.20), but it does not metter.) The coherence, which is also known as mutual coherence in the literature, between **Φ** and **Ψ** is defined as

$$\mu(\mathbf{\Phi}, \mathbf{\Psi}) = \sqrt{N} \cdot \max_{i,m} \left| \langle \varphi_i, \psi_m \rangle \right| . \tag{1.28}$$

Mutual coherence is so the largest correlation between an atom in basis **Φ** and an atom in basis **Ψ**. Using Cauchy-Schwarz inequality, we get $\mu(\mathbf{\Phi}, \mathbf{\Psi}) \leq \sqrt{n}$. On the other hand, $\mu(\mathbf{\Phi}, \mathbf{\Psi}) \geq 1$ must hold, because otherwise we would have

$$\forall m \left| \langle \varphi_i, \psi_m \rangle \right| < \frac{1}{\sqrt{N}} \Rightarrow \|\varphi_i\|_2^2 = \sum \langle \varphi_i, \psi_m \rangle^2 < 1 ,$$

which contradicts with φ_i being a unit vector. Therefore, the coherence is always in the range [1; $\sqrt{N}$].

We are interested in bases pairs with small coherence and in that case we say the two bases are incoherent. In the case of dictionaries for sparse representation, the incoherence is important because a small μ means small correlation between the two bases, thus a dictionary composed by such a pair of bases has a “richer vocabulary” relative to a dictionary with a larger μ. In the context of CS, the incoherence between the basis involved in signal sensing and the basis involved in signal representation/reconstruction is a crucial feature of an effective CS system.

An important question to address is what kind of matrices **Φ** and **Ψ** should be in order for them to have a low mutual coherence? As $\mathbf{A} = \mathbf{\Phi}\mathbf{\Psi}^{\mathrm{T}}$ is the dictionary with unit atoms then ℓ_2-norm of *k*-th row of **A** is equal to

$$\sum_m \langle \varphi_k, \psi_m \rangle^2 = 1 \cdot$$

Therefore, to possess a small μ, the entries in each row of **A** must not be concentrated but wide spread. For instance, a most concentrated row of **A** would look like $(0,0,...,0,\pm 1,0,...,0)^{\mathrm{T}}$ which gives $\mu(\mathbf{\Phi},\mathbf{\Psi}) = \sqrt{N}$, while a most “spread out” row of **A** would look like $(\pm 1,\pm 1,...,\pm 1)^{\mathrm{T}} / \sqrt{N}$ which would lead to $\mu(\mathbf{\Phi},\mathbf{\Psi}) = 1$ if every row of **A** is like that. Similar claim can be made for the columns of **A**. The same can also be said about **Ψ**: each column of **Ψ** must be spread out in the **Φ** domain in order to have a small mutual coherence.

Let discuss some practical examples. For DFT, **Φ=I** is canonical or spike basis, and **Ψ** is Fourier basis (1.11):

$$\psi_{mk} = \frac{1}{N} \exp\left(j \frac{2\pi km}{N} \right). \qquad (1.29)$$

Since **Ψ** is the sensing matrix, this corresponds to the classical sampling scheme in time. The time-frequency pair obeys $\mu(\mathbf{\Phi},\mathbf{\Psi}) = 1$ and, therefore, we have maximal incoherence. Further, spikes and sinusoids are maximally incoherent not just in one dimension but in any dimension (in two dimensions, three dimensions, etc.) if multidimensional DFT is applied.

Another example takes wavelets bases for **Ψ** and noiselets [22] for **Φ**. The coherence between noiselets and Haar wavelets is $\sqrt{2}$, and that between noiselets and Daubechies D4 and D8 wavelets is, respectively, about 2.2 and 2.9 across a wide range of sample sizes *n*. (Noiselets are also maximally incoherent with spikes and incoherent with the Fourier basis.) Noiselets are interesting because they are incoherent with systems providing sparse representations of image data and other types of data, and they come with very fast algorithms (the noiselet transform runs in O(*n*) time, and just like the Fourier transform, the noiselet matrix does not need to be stored to be applied to a vector. This is of crucial practical importance for numerically efficient CS implementations.

Finally, random matrices are known to be incoherent with *any* fixed basis **Ψ**. Random orthonormal basis **Φ** can be compound of *n* vectors sampled independently and uniformly on the unit sphere. For such $\mathbf{\Psi}$ and $\mathbf{\Phi}$, the coherence with high probability is $\mu(\mathbf{\Phi},\mathbf{\Psi}) \approx \sqrt{2\log N}$ [13].

Generally incoherence between a sensing matrix $\mathbf{\Phi}$ and a sparsifying transform basis $\mathbf{\Psi}$ means that

- the test vectors (i.e. the rows in $\mathbf{\Phi}$) must be spread out in the $\mathbf{\Psi}$ domain, just as a spike in the time domain is spread out in the frequency domain;
- in this regard, incoherence extends the duality between time and frequency.

Now let us consider an approach briefly described in previous section (1.19) – (1.21).

The discrete signal $\mathbf{x}$ itself may or may not be sparse in the canonical basis but is sparse or approximately sparse in an appropriate basis $\mathbf{\Psi}$: $\mathbf{x} = \mathbf{\Psi}\boldsymbol{\theta}$. A central idea in the current CS theory is about how a (discrete) signal is acquired: the acquisition of signal $\mathbf{x}$ of length N is carried out by measuring n projections of $\mathbf{x}$ onto sensing (also known as testing) vectors $\{\varphi_i \,|i = 1..n\}$: $y_i = \varphi_i \mathbf{x}$. Using matrix notation the sensing process is described by

$$\mathbf{y} = \hat{\mathbf{\Phi}}\mathbf{x}, \tag{1.30}$$

where $\hat{\mathbf{\Phi}} \in \mathrm{R}^{n\times N}$, $\mathbf{y} \in \mathrm{R}^{n}$, $\mathbf{x} \in \mathrm{R}^{N}$ and for meaningful projections, it is often assumed that the lengths of the projection vectors are unity: $\|\varphi_i\|_2 = 1$. Here are two questions that naturally arise from signal model (1.30).

- What type of matrix $\hat{\mathbf{\Phi}}$ should one choose for the purpose of sensing?
- How many measurements $\{y_i \,|i = 1..n\}$ should one collect so that these measurements will be sufficient to recover signal $\mathbf{x}$?

The following theorem addresses these questions explicitly.

Theorem 1.1 [9]. Given $\mathbf{x} \in \mathrm{R}^{N}$ and suppose $\mathbf{x}$ is s-sparse in basis $\mathbf{\Psi}$. Select n measurements in the $\mathbf{\Phi}$ domain uniformly at random via (1.33) (that is, the n testing vectors $\{\varphi_i \,|i = 1..n\}$ are n rows uniformly randomly selected from matrix $\mathbf{\Phi}$). If

$$n \ge C \cdot \mu^2(\mathbf{\Phi},\mathbf{\Psi}) \cdot s \cdot \log(N/\delta), \tag{1.31}$$

for some positive constant C, then signal $\mathbf{x}$ can be exactly reconstructed with overwhelming probability more then $1-\delta$ by solution of the convex ℓ_1-minimization problem:

$$\mathbf{x} = \mathbf{\Psi} \cdot \operatorname{argmin}\|\boldsymbol{\theta}\|_1 \text{ subject to } \mathbf{y} = \hat{\mathbf{\Phi}}\mathbf{\Psi}\boldsymbol{\theta} = \mathbf{A}\boldsymbol{\theta}. \tag{1.32}$$

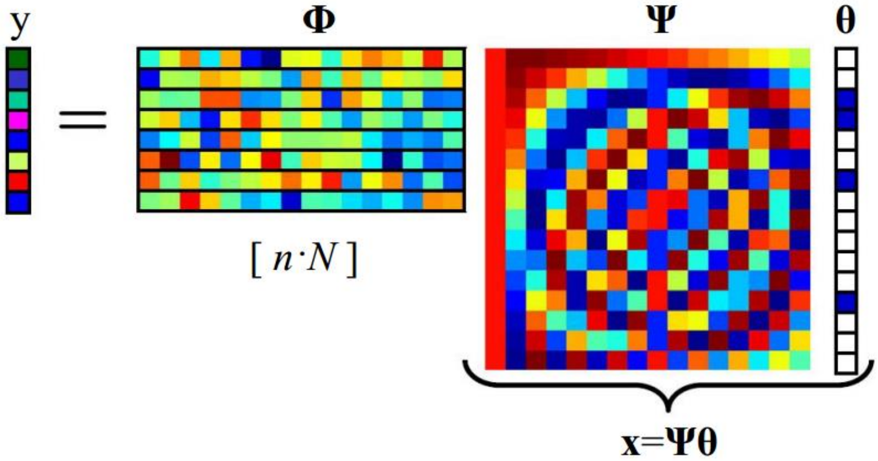


Fig. 1.12. General scheme of compressive sensing problem [3]

We wish to make four comments:

1) The role of the coherence is completely transparent; the smaller the coherence, the fewer samples are needed. The theorem unites sparsity, incoherence and sampling rate in a single formula.

2) One suffers no information loss by measuring just about any set of n coefficients which may be far less than the signal size apparently demands. If $\mu(\mathbf{\Phi}, \mathbf{\Psi}) \approx 1$, then the $O(s \log N)$ samples suffice instead of N. Even better, there is a four-to-one practical rule which says that for exact reconstruction, one needs about four incoherent measurements per unknown nonzero term in **x**:

$$n \geq 4s \tag{1.33}$$

regardless of the signal's dimension N [13].

3) The signal **x** can be exactly recovered from our condensed data set by minimizing a convex functional which does not assume any knowledge about the number of nonzero coordinates of **x**, their locations, or their amplitudes which we assume are all completely unknown a priori. We just run the algorithm and if the signal happens to be sufficiently sparse, exact recovery occurs.

The theorem indeed suggests a very concrete acquisition protocol: sample nonadaptively in an incoherent domain and invoke linear programming after the acquisition step. Following this protocol would essentially acquire the signal in a compressed form. All that is needed is a decoder to "decompress" this data; this is the role of ℓ_1-minimization.

4) Figure 1.13 from [52] offers an intuitive explanation of why minimizing ℓ_1-norm helps obtain a sparse solution, when more common ℓ_2-norm fails to provide sparsity.

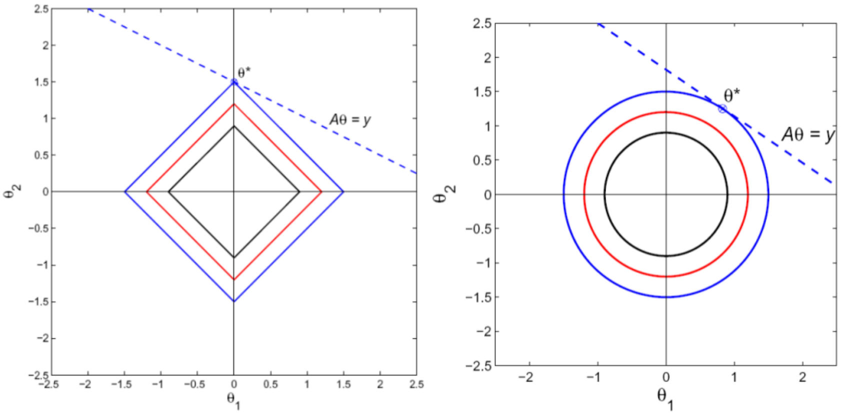


Fig. 1.13. Linear restriction $\mathbf{y} = \mathbf{A}\boldsymbol{\theta}$ is graphically expressed like a target hyper-plane (a straight line in 2D case). Minimizing ℓ_p-norm means search for the smallest radius of hyper-sphere which intersects the hyper-plane. In the ℓ_1 case (left) the osculation points belong to the coordinate axes (lower order subsets) and such way provide sparsity. In the ℓ_2 case (right) the osculation points in most cases are situated anywhere, and thus the solution is not sparse [52]

In practice, signals tend to be *compressible*, rather than sparse. Mathematically, a compressible signal has a representation whose entries decay rapidly when sorted in order of decreasing magnitude. In fact, we can quantify the compressibility by calculating the error incurred by approximating a signal **x** by some *s*-sparse signal **x'**:

$$\sigma_s(\mathbf{x})_p = \min_{\|\mathbf{x}'\|_0 \le s} \|\mathbf{x} - \mathbf{x}'\|_p . \tag{1.34}$$

Compressible signals are well approximated by sparse signals, so the sparse approximation framework applies to this class. In practice, it is usually more challenging to identify approximate representations of compressible signals than of sparse signals. To do it one may relax (1.27) to allow some error tolerance $\varepsilon \ge 0$. Such way both compressible signals and sparse signals observed with noise can be extracted:

$$\operatorname{argmin}\|\mathbf{x}\|_0 \text{ subject to } \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \le \varepsilon . \tag{1.35}$$

It is most common to measure the prediction-observation discrepancy with the Euclidean norm, but other loss functions may also be appropriate. The elements of (1.35) can be combined in several other ways to obtain related problems. For example, we can seek the minimal error possible at a given level of sparsity *s*:

$$\operatorname{argmin}\|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2 \text{ subject to } \|\mathbf{x}\|_0 \le s . \tag{1.36}$$

We can also use a regularization parameter $\lambda \geq 0$ to balance error and sparsity:

$$\arg\min\left(\|\mathbf{Ax}-\mathbf{y}\|_2+\lambda\|\mathbf{x}\|_0\right). \tag{1.37}$$

While Theorem 1.1 provides compete information for sparse signal reconstruction it is not directly generalized to deal with compressible signals. So we need an alternative approach to analysis of the problem (1.27) which can be adopted for problems such as (1.35) – (1.37). The approximation accuracy of a compressed sensing matrix is determined by coherence based on its null space. We now examine a property of the null space which becomes a necessary and sufficient condition for reconstruction. Spark is the most common way to characterize this property.

The spark of a given matrix **A** is the smallest number of columns of **A** that are linearly dependent. $\operatorname{spark}(\mathbf{A})=\min\left\{\|\mathbf{x}\|_0:\mathbf{Ax}=0\wedge\mathbf{x}\neq 0\right\}$. Unlike linear independence which gives the rank, spark gives uniqueness of sparse set solution, measuring how far it is from singularity. Suppose there are two *s*-sparse vectors $\mathbf{x}\neq\mathbf{x}'$ giving the same response **y**. Then

$$\mathbf{Ax}-\mathbf{Ax}'=\mathbf{A}(\mathbf{x}-\mathbf{x}')=0\Rightarrow\operatorname{spark}(\mathbf{A})\leq\|\mathbf{x}-\mathbf{x}'\|_0\leq 2s. \tag{1.38}$$

Theorem 1.2 (Corollary 1 of [27]). For any vector $\mathbf{y}\in\mathrm{R}^n$, there exists at most one *s*-sparse signal $\|\mathbf{x}\|_0\leq s$ such that **y = Ax** if and only if spark(**A**)>2*s*.

It is easy to see that $\operatorname{spark}(\mathbf{A})\in[2;\operatorname{rank}(\mathbf{A})+1]$. If **A** is non-singular then $\operatorname{spark}(\mathbf{A})=n+1$. Therefore, Theorem 1.2 yields the requirement $n\geq 2s$.

When dealing with exactly sparse vectors, the spark provides a complete characterization of when sparse recovery is possible. However, when dealing with approximately sparse signals we must consider somewhat more restrictive conditions on the null space of **A**. Roughly speaking, we must also ensure that it does not contain any too compressible vectors in addition to vectors that are sparse. Let $\Delta:\mathrm{R}^n\to\mathrm{R}^N$ represent our specific recovery method. [21] has shown that

$$\|\Delta(\mathbf{Ax})-\mathbf{x}\|_2\leq C\frac{\sigma_s(\mathbf{x})_1}{\sqrt{s}} \tag{1.39}$$

guarantees exact recovery of all possible *s*-sparse signals, but also ensures a degree of robustness to non-sparse signals that directly depends on how well the signals are approximated by *s*-sparse vectors. The constant *C* is called Null Space Property (NSP) constant. Such guarantees are called instance-optimal since they provide optimal performance for each instance of **x** [21]. This distinguishes them from guarantees that only hold for some subset of possible signals, such as sparse or compressible signals. The quality of the guarantee adapts to the particular choice of **x**. These are also commonly referred to as uniform guarantees since they hold uniformly for all **x**. With latest results [59], the optimal decoder $\Delta_0(\mathbf{y})$ is defined as

$$\Delta_0(\mathbf{y}) \equiv \operatorname{argmin}\{\sigma_s(\mathbf{x})_1 : \mathbf{Ax} = \mathbf{y}\}. \tag{1.40}$$

While the NSP is both necessary and sufficient for establishing guarantees of reconstruction precision for both sparse and compressed signals, these guarantees do not account for noise. When the measurements are contaminated with noise or have been corrupted by some error such as quantization, it will be useful to consider somewhat stronger conditions. Candés and Tao introduced the following isometry condition on matrices **A** and established its important role in CS.

A matrix **A** satisfies the restricted isometry property (RIP) of order *s* if there exists an $\varepsilon_s \in (0;1)$ s. t.

$$(1-\varepsilon_s)\|\mathbf{x}\|_2^2 \le \|\mathbf{Ax}\|_2^2 \le (1+\varepsilon_s)\|\mathbf{x}\|_2^2 \tag{1.41}$$

holds for all *s*-sparse vectors $\mathbf{x} : \|\mathbf{x}\|_0 \le s$. The smallest such ε_s is called the matrix isometry constant of order *s*. When RIP holds for isometry constant close for zero, **A** approximately preserves the Euclidean length of *s*-sparse signals, which in turn implies that *s*-sparse vectors cannot be in the null space of **A**. An equivalent description of the RIP is to say that all subsets of *s* columns taken from **A** are in fact nearly orthogonal. (The columns of A cannot be always exactly orthogonal since there are more columns than rows).

To see the connection between the RIP and CS, imagine we wish to acquire *s*-sparse signals with **A**. Suppose that ε_{2s} is sufficiently less than one. This implies that all pairwise distances between the signals must be well preserved in the measurement space.

That is,

$$(1-\varepsilon_{2s})\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2 \le \|\mathbf{Ax}_1 - \mathbf{Ax}_2\|_2^2 \le (1+\varepsilon_{2s})\|\mathbf{x}_1 - \mathbf{x}_2\|_2^2$$

holds for all *s*-sparse vectors $\mathbf{x}_1$, $\mathbf{x}_2$. This fact guarantees the existence of robust algorithms for discriminating *s*-sparse signals based on their compressive measurements.

Theorem 1.3 [10]. Assume that $\varepsilon_{2s} < \sqrt{2} - 1$. Then

$$\hat{\mathbf{x}} = \operatorname{argmin}\|\mathbf{x}\|_1 \text{ subject to } \mathbf{y} = \mathbf{Ax}$$

obeys

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \le C_0\|\mathbf{x}_s - \mathbf{x}\|_1 / \sqrt{s} \text{ and } \|\hat{\mathbf{x}} - \mathbf{x}\|_1 \le C_0\|\mathbf{x}_s - \mathbf{x}\|_1 \tag{1.42}$$

for some constant C_0, where $\mathbf{x}_s$ is the vector **x** with all but the largest *s* components set to 0.

The conclusions of Theorem 1.3 are stronger than those of Theorem 1.1. If **x** is *s*-sparse, then x = $\mathbf{x}_s$ and, thus, the recovery is exact. But this new theorem deals with all signals. If x is not *s*-sparse, then (1.43) anyway asserts good quality of the recovered signal. It is nearly as good as if one knew a priory the location of the *s* largest components of **x**. In other words, the reconstruction is nearly as good as for usual transform coding of dense recorded signal.

Another striking difference with Theorem 1.1 is that Theorem 1.3 is deterministic; it involves no probability. If we are fortunate enough to hold a sensing matrix **A** obeying the hypothesis of the theorem, we may apply it. And we are then guaranteed to recover all *s*-sparse vectors exactly, and essentially the *s* largest entries of all vectors otherwise; i.e., there is no probability of failure.

However, at this point we still suppose precise measuring while most real signals are observed with noise. To deal with noisy signals one may use minimization conditions of type (1.35) – (1.37). RIP meets this request in the form provided by the next theorem.

Theorem 1.4 [13]. Assume we are given noisy data $\mathbf{y} = \mathbf{A}\mathbf{x} + \hat{\mathbf{n}}$ and **A** satisfies RIP with isometry constant $\varepsilon_{2s} < \sqrt{2} - 1$. Let $\varepsilon \ge \|\hat{\mathbf{n}}\|_2$ bounds the amount of noise in the data. Then

$$\hat{\mathbf{x}} = \arg\min \|\mathbf{x}\|_1 \text{ subject to } \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \le \varepsilon$$

obeys

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \le C_0 \|[\mathbf{x}]_s - \mathbf{x}\|_1 \big/ \sqrt{s} + C_1 \varepsilon \, , \tag{1.43}$$

for some constants C_0 and C_1, where $[\mathbf{x}]_s$ is the vector **x** with all but the largest *s* components set to 0: $[\mathbf{x}]_s = \sum_{i \in \Lambda} x_i \mathbf{e}_i$ where $\Lambda = \arg\max_{\|\mathrm{M}\| \le s} \sum_{i \in \mathrm{M}} |x_i|$ and $\mathbf{e}_m = [\delta_{0m}, \delta_{1m}, \delta_{2m}, \ldots, \delta_{Nm}]^{\mathrm{T}}$ denotes a unit coordinate vector, $\delta_{im} = \delta(i - m)$.

This last result establishes CS as a practical and robust sensing mechanism. It works with all kinds of not necessarily sparse signals, and it handles noise gracefully. Further, the constants C_0 and C_1 are typically small. With ε_{2s} = ¼ for example, $C_0 \le 5.5$ and $C_1 \le 6$ [13].

Finally, if a matrix satisfies the RIP, then it also satisfies the NSP.

Theorem 1.5 [24]. Suppose that **A** satisfies the RIP of order 2*s* with $\varepsilon_{2s} < \sqrt{2} - 1$. Then **A** satisfies the NSP of order 2*s* with constant

$$C = \frac{\varepsilon_{2s}\sqrt{2}}{1 - \varepsilon_{2s}\left(1 + \sqrt{2}\right)} . \tag{1.44}$$

Thus, the RIP is strictly stronger than the NSP.

The RIP condition is equivalent to saying that the symmetric matrix $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ is positive definite with eigenvalues between $(1-\varepsilon_{2s})^2$ and $(1+\varepsilon_{2s})^2$ [13]. Although both the smallest and largest singular values of $\mathbf{A}^{\mathrm{T}}\mathbf{A}$ affect the stability of the reconstruction algorithms, the smaller eigenvalue is dominant for compressed sensing in that it allows distinguishing between sparse vectors from their measurement by **A**.

It is important to note that the RIP conditions are difficult to verify for a given matrix. However, as we mentioned above random matrices are incoherent with any fixed basis. A widely used technique for avoiding checking the RIP directly is to generate the matrix **A** randomly. According to Johnson-Lindenstrauss Lemma [4] randomized sensing matrix **A** preserves the norm of any *s*-sparse input signal to within a small fraction: the pairwise distances are (1 ± δ) preserved when one projects a set of *N* points into a random linear subspace of at least O($\delta^{-2}\log N$) dimensions. The probabilistic distributions fitted to the lemma conditions are called JL favorable ones. Many constructions of random **A** matrices such as Gaussian and Bernoulli ensembles are JL favorable.

Theorem 1.6 [4]. Supposing **A** is drawn from a JL-favorable distribution, then with probability at least $1-e^{-Cn}$, **A** meets the RIP with

$$s \le C\frac{n}{\log(N/n)+1}. \tag{1.45}$$

This type of uniform uncertainties or restricted isometries is sometimes referred to as the Statistical Restricted Isometry Property or STRIP.

1.4. Compressive Signal Recovery Algorithms

If there are no restrictions on the dictionary **A** and the recorded signal **y**, then sparse approximation is at least as hard as a general constraint satisfaction problem. Indeed, for fixed constants $C, K \ge 1$, it is NP-hard to produce a (*Cs*) - sparse approximation whose error lies within a factor *K* of the minimal *s*-term approximation error [Muth05, Sec. 0.8.2]. Nevertheless, over the past decade, researchers have identified many interesting classes of sparse approximation problems that submit to computationally tractable algorithms. In practice, sparse approximation algorithms tend to be slow unless the dictionary **A** admits a fast matrix-vector multiply [72]. However, the cost of these products is only O(*N*log*N*) when **A** is constructed from Fourier or wavelet bases **Ψ**. Fast multiply creates conditions for use of iterative methods for solution of a least-squares problem.

In previous section we have discussed two classes of sparse approximation problems where this property holds. First, many naturally occurring signals are compressible with respect to deterministic dictionaries constructed using principles of harmonic analysis [31] (e.g., wavelet coefficients of natural images). This type of structured dictionary often comes with a fast transformation algorithm. Second, the probabilistic compressive sampling approach typically views **A** as the product of a random observation matrix **Φ** and a fixed orthogonal matrix **Ψ** that determines a basis in which the signal is sparse (Fig. 12). Recently, there have been substantial efforts to incorporate more sophisticated signal constraints into sparsity models. In particular, [2] have studied model-based CS algorithms, which use additional information such as the tree structure of wavelet coefficients to guide reconstruction of signals.

There are five major approaches for solving sparse approximation problems:

- **Greedy pursuit**. Iteratively refine a sparse solution by successively identifying one or more components that yield the greatest improvement in quality.
- **Convex relaxation**. Replace the combinatorial problem with a convex optimization problem. Solve the convex program with algorithms that exploit the problem structure [19].
- **Bayesian framework**. Assume a prior distribution for the unknown coefficients that favors sparsity. Develop a maximum a posteriori estimator that incorporates the observation. Identify a region of significant posterior mass [79] or average over most-probable models [66].
- **Nonconvex optimization**. Relax the ℓ_0 problem to a related nonconvex problem and attempt to identify a stationary point [18].
- **Brute force**. Search through all possible support sets, possibly using cutting-plane methods to reduce the number of possibilities [57].

Here we present just the first approach. Greedy pursuit methods provide both high performance and simple way for development of CS solvers from scratch.

Another fundamental approach for sparse approximation is convex relaxation. It is directly related to Theorems 1.3 and 1.4. But it may take a long time to solve the linear program, even for signals of moderate length. Furthermore, the implementation of convex optimization algorithms may demand significant effort. (Fortunately, there is a wide set of already developed techniques and software tools for convex optimization.)

Other three approaches are not widely used.

Greedy pursuit methods for sparse approximation iteratively improve the current estimate for the target vector $\mathbf{x}$ by modifying one or several coefficients chosen to yield a substantial improvement in approximating the signal. Matching Pursuit (MP) is a computationally simple algorithm proposed in 1993 by Mallat and Zhang:

1. Initialize. Set $\mathbf{x}_0 = 0$, the index set Λ_0 empty, the residual $\mathbf{r}_0 = \mathbf{y}$, and put the counter $k = 1$.
2. Identify. Select the atom that explains most of the energy in the signal. Find a column m of $\mathbf{A}$ that is most strongly correlated with the residual: $m = \arg\max\limits_i \left|\langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_i \rangle\right|$. (Hereafter $\boldsymbol{\alpha}_m$ is *m*-th column of **A**.)
3. Iterate. Update anything: $\Lambda_k = \Lambda_{k-1} \cup \{m\}$, $\mathbf{x}_k = \mathbf{x}_{k-1} + \langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_m \rangle \mathbf{e}_m$, $\mathbf{r}_k = \mathbf{r}_{k-1} - \langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_m \rangle \boldsymbol{\alpha}_m$. Increment *k*: $k = k + 1$. Repeat (2)–(3) until stopping criterion holds.
4. Output. Return the vector $\mathbf{x}_k$.

There are several natural stopping criteria, e.g., break after a fixed number of iterations: *k* = *s*, stop when the residual has small magnitude $\|\mathbf{r}_k\|_2 \le \delta$, or when no column explains a significant amount of energy in the residual: $\max\limits_i \left|\langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_i \rangle\right| \le \delta$. These criteria can all be implemented at minimal cost.

Orthogonal Matching Pursuit (OMP) is one of the earliest methods for sparse approximation. The OMP idea can be traced to 1950s work on variable selection in stepwise regression [72]. It is similar to MP one except the coefficients are completely re-computed at each step:

1. Initialize. Set $\mathbf{x}_0 = 0$, the index set Λ_0 empty, the residual $\mathbf{r}_0 = \mathbf{y}$, and put the counter $k = 1$.
2. Identify. Select the atom that explains most of the energy left in the signal. Find a column m of $\mathbf{A}$ that is most strongly correlated with the residual: $m = \arg\max\limits_i \left|\langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_i \rangle\right|$.
3. Estimate. Find the best coefficients for approximating the signal with the columns chosen so far: $\Lambda_k = \Lambda_{k-1} \cup \{m\}$, $\mathbf{x}_k = \arg\min\limits_{\mathbf{u}} \|\mathbf{y} - \mathbf{A}_{\Lambda_k}\mathbf{u}\|_2$. (Here $\mathbf{A}_{\Lambda_k}$ consists of $\boldsymbol{\alpha}_i : i \in \Lambda_k$.)
4. Iterate. Update the residual: $\mathbf{r}_k = \mathbf{y} - \mathbf{A}\mathbf{x}_k$. Increment *k*: $k = k + 1$. Repeat (2)–(4) until stopping criterion holds.
5. Output. Return the vector $\mathbf{x}_k$.

In a typical implementation of OMP, the identification step is the most expensive part of the computation. The direct approach computes the maximum inner product via the matrix–vector multiplication $\mathbf{Ar}_{k\text{-}1}$, which costs O(*nN*) for an unstructured dense matrix. Nearest-neighbor data structures can be used to perform the identification query more efficiently [25]. In certain applications, such as projection pursuit regression, the columns of **A** are indexed by a continuous parameter, and identification can be posed as a low-dimensional optimization problem [36].

The estimation step requires the solution of a least-squares problem. The most common technique is to maintain a QR factorization of $\mathbf{A}_{\Lambda_k}$, which costs O(*nk*). The new residual $\mathbf{r}_k$ is a by-product of the least-squares problem, so it requires no extra computation.

OMP produces the residual $\mathbf{r}_n = 0$ after *n* steps (provided that the dictionary can represent the signal **y** exactly), but the conditions of efficient finding such way a sparse representation are unfavorable for most practical cases (*n* is supposed to be high for quick convergence). When **A** is sufficiently random, OMP provably recovers *s*-sparse signals when $s<n/(2\log N)$ [70]. Contemporary CS pursuit methods work better in practice and yield essentially optimal theoretical guarantees. These techniques depend on several enhancements to the basic greedy framework:

- selecting multiple columns per iteration;
- pruning the set of active columns at each step;
- solving the least-squares problems iteratively; and
- theoretical analysis using the RIP bound.

CoSaMP [60] was the first algorithm which assembled these ideas to obtain essentially optimal performance guarantees:

1. Initialize. Set $\mathbf{x}_0 = 0$, the index set Λ_0 empty, the residual $\mathbf{r}_0 = \mathbf{y}$, and put the counter $k = 1$.
2. Identify. Select multiple atoms that explains most of the energy in the residual. Find *cs* columns of **A** that are most strongly correlated with the residual: $\Lambda^+ = \arg\max_{\|\mathrm{M}\|<cs} \sum_{i\in\mathrm{M}} \left|\langle \mathbf{r}_{k-1}, \boldsymbol{\alpha}_i \rangle\right|$.
3. Merge. Put the old and new columns into one set: $\Lambda_k = \Lambda_{k-1} \cup \Lambda^+$.
4. Estimate. Find the best coefficients for approximating the signal with the merged column set: $\hat{\mathbf{x}}_k = \arg\min_{\mathbf{u}} \left\|\mathbf{y} - \mathbf{A}_{\Lambda_k}\mathbf{u}\right\|_2$.
5. Prune. Retain the *s* largest coefficients: $\mathbf{x}_k = [\hat{\mathbf{x}}_k]_s$.
6. Iterate. Update the residual: $\mathbf{r}_k = \mathbf{y} - \mathbf{A}\mathbf{x}_k$. Increment *k*: $k = k + 1$. Repeat (2)–(6) until stopping criterion holds.
7. Output. Return the vector $\mathbf{x}_k$.

Both the practical performance and theoretical analysis of CoSaMP require the dictionary **A** to satisfy the RIP of order 2*s* with constant $\varepsilon_{2s} << 1$. According to [72] a heuristic for identifying the maximum sparsity level *s* is

$$s \le \frac{n}{2\log(N/s+1)}. \quad (1.46)$$

Under the RIP hypothesis, each iteration of CoSaMP reduces the approximation error by a constant factor until it approaches its minimal value. Let $\hat{\mathbf{x}}$ is the unknown coefficient vector and $\varepsilon \ge \|\hat{\mathbf{n}}\|_2$ bounds the amount of noise in the data. Then after a sufficient number of iterations, i.e., O(log*N*) [72]:

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \le C_0 \left\| [\hat{\mathbf{x}}]_{s/2} - \hat{\mathbf{x}} \right\|_1 + C_1 \varepsilon. \quad (1.47)$$

The form of this error bound is optimal [21].

In practice, CoSaMP is faster but usually less effective than algorithms based on convex programming. Of course, it can be applied without the RIP, but the behavior is unpredictable.

Other greedy algorithms are described in [61]. They are closely related to iterative thresholding algorithms, which have been studied extensively over the last decade. Among thresholding approaches, iterative hard thresholding (IHT) is the simplest:

1. Initialize. Set $\mathbf{x}_0 = 0$, the residual $\mathbf{r}_0 = \mathbf{y}$, and put the counter $k = 1$.
2. Estimate. Find current residual $\mathbf{r}_k = \mathbf{r}_{k-1} - \mathbf{A}\mathbf{x}_{k-1}$ and use it for approximation of the signal: $\hat{\mathbf{x}}_k = \mathbf{x}_{k-1} + \mathbf{A}^+ \mathbf{r}_k$. Here $\mathbf{A}^+$ denotes pseudoinverse of the matrix **A**.
3. Prune. Retain the *s* largest coefficients: $\mathbf{x}_k = [\hat{\mathbf{x}}_k]_s$.
4. Iterate. Increment *k*: $k = k + 1$. Repeat (2)–(4) until stopping criterion holds.
5. Output. Return the vector $\mathbf{x}_k$.

Blumensath and Davies [6] have established that IHT admits an error guarantee of the form (1.46) under a RIP hypothesis of the form $\varepsilon_{2s} << 1$.

The thresholding techniques of Donoho and Maliki described in the section 1.2 are internally connected to pursuit methods. Stepwise thresholding is reasonably effective for solving sparse approximation problems in practice.

Greedy pursuit methods have often been considered naïve, in part because there are contrived examples where the approach fails spectacularly, e.g. (Sec. 2.3.2) [20]. Some simulations indicate that simple thresholding techniques behave poorly in the presence of noise (Sec. 8) [6]. However, recent research has clarified that greedy pursuits succeed empirically and theoretically in many situations where convex relaxation works. In fact, the boundary between greedy methods and convex relaxation methods is somewhat blurry. The greedy selection technique is closely related to dual coordinate-ascent algorithms, while certain methods for convex relaxation, such as LARS and homotopy, use a type of greedy selection at each iteration [72].

Greedy pursuits, thresholding, and related methods can be quite fast, especially in the ultrasparse regime under low noise. In addition to simplicity they have several special advantages over convex optimization methods. First, the greedy approach is efficiently extended for the dictionaries of infinite size (like in projection pursuit regression). Second, greedy techniques can incorporate constraints that do not fit naturally into convex programming formulations such as tree-like constraints on wavelet coefficients [2].

1.5. Applications of undersampled representations

Considerable efforts have been devoted in recent years by many researchers to adapt the theory of CS to better solve real-world challenges. This has also led to parallel low-rate sampling schemes that combine the principles of CS with the rich theory of sampling such as the finite rate of innovation (FRI) and Xampling frameworks [32]. Just CS encoding and decoding hardware listed in https://sites.google.com/site/igorcarron2/compressedsensinghardware targets wide range of applications from curious devices such as "one-pixel camera" to wideband analog signal receivers, high-speed MRI scanners and high-throughput screening, reduced-cost seismic imaging, X-Ray astronomy camera and ground penetrating radar, audio, video, radio receivers, etc.

CS solves many of the hardware limitations in demodulation in communication systems and movie cameras. For example, many wideband communication signals are comprised of several narrow transmissions modulated at high carrier frequencies. Traditional demodulation, however, requires knowing the exact carrier frequency. In the context of multiband communications, for example, the carrier frequencies may not be known, or may be changing over time. The received signal random projections are utilized in CS to recovery structured analog signals highly above the Nyquist threshold.

Another category, the acquisition hardware is usually limited by construction to measure directly in a transform domain. The most relevant examples are MRI [54] and tomographic imaging [53] where the measurements obtained from the hardware correspond to coefficients of the image's 2-D continuous Fourier transform are incoherent with sparsity/compressibility transforms of the pictured objects such as wavelets, total variation, and the standard canonical representation [32]. Tomographic image sampling can be made in the most dominant (e.g., Fourier) directions to achieve image construction with minimum data. Such smaller sampling would enable real time movements of the organs in medical images [53].

In the case of optical microscopy, the highpass Fourier coefficients are completely lost. To treat the case of recovery from lowpass coefficients, a special purpose sparse recovery method was developed under the name of Nonlocal Hard Thresholding (NLHT) [39]. This technique attempts to allocate the off-support of the sparse signal in an iterative fashion by performing a thresholding step that depends on the values of the neighboring locations.

Let discuss an example of geophysical simulation application which uses undersampling for calculation acceleration purpose.

Synthetic seismograms generated for a given geological model and survey geometry allow one to estimate the practical effect of more expensive prospecting in specific geological conditions. Seismic modeling helps to optimize survey design when an object exploration is planned (Fig. 1.14). Synthetic seismograms are useful also for testing of new algorithms, programs and processing batches. Because the real object geology is not exactly known synthetic seismograms excel field seismograms for quality assessment and comparison of different processing methods. Finally, simulation can significantly reduce the risk of misinterpretation and serves as an argument to justify the interpretation done.

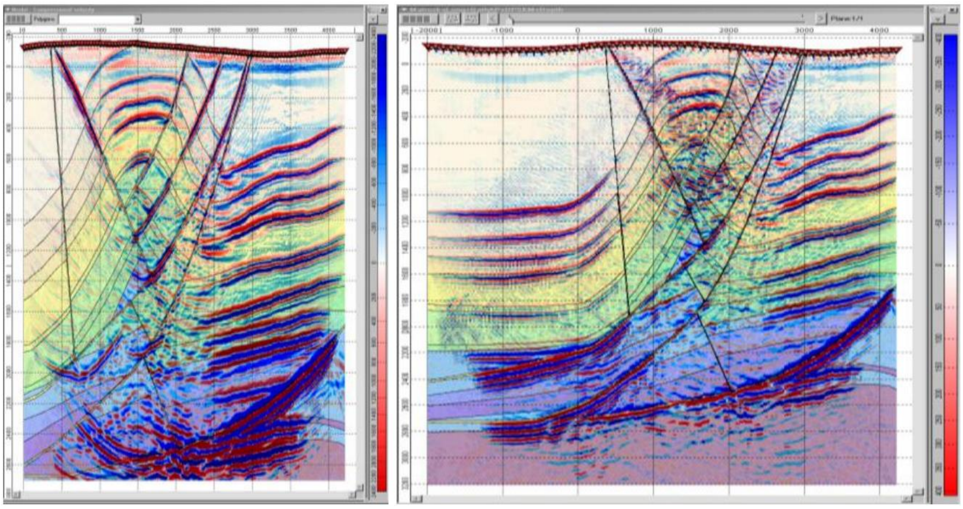


Fig. 1.14. The pre-stack depth migration results before and after 2 km left expansion of the survey line and increasing the record time

Methods based on simplified physical models are widely applied to generate synthetic seismograms for given survey design and velocity model. Ray tracing (RT) is currently the most used approach for 3D survey planning. The RT method is based on a high-frequency approximation and known optics laws of refraction and reflection of compression and shear waves at velocity boundaries and ray propagation laws for gradient media [17]. The method is an efficient tool to estimate target horizon illumination dependent on its geometry, velocities and survey design [40]. Its advantage is the ability to identify each wave type, e.g. calculation of seismograms without multiples and converted waves, or at limited multiplicity. However, the synthetic seismograms of RT are too idealized, as the high-frequency approximation can not properly account inhomogeneities of size comparable with the wavelength. In addition, RT unfits for modeling the reflections that arise in gradient environments as well as to the conditions of sharp edges and points of diffraction (since it uses interpolation of rays and amplitudes). Also RT meets difficulties in synthesis of wave fields which are not completely determined by a zero term of the ray series and require accounting the subsequent terms.

An alternative approach is to simulate the wavefield propagation process in time. The approach is called a full-wave modeling (FWM). It is based on a system of hyperbolic partial differential equations obtained by substitution of Hooke's law of elasticity in the formula of Newton's second law. FWM generates synthetic data almost identical to those observed by a field seismic survey at the conditions of exactly known velocity and density properties of the media. If there is sufficient a priori information about the structure of the upper part of the geological section, the propagation velocities of compression and shear waves, and density of rocks, synthetic seismograms of FWM contain all types of useful and noise waves (reflected, refracted, diffracted, converted, multiple, surface, etc.) that can be expected for field recording. FWM makes possible to consider in advance not only the illumination, but the whole complex of acquisition and processing batch to image target interfaces with highest precision in presence of noise waves, and thus to accelerate solving of the geological problems. Simulation is especially important for 3D prospecting in complex seismological conditions.

The most common approach for the numerical solution of the FWM problem is the explicit finite difference method [1]. Since wide source-receiver offsets and deep penetration of typical seismic acquisition the simulation grid dimensions are measured in hundreds of wavelengths. Preservation of wavelet requires about 10 grid cells per wavelength. Therefore, the finite difference grid in the simulation of 3D seismic data consists of billions of nodes. Given the 3D anisotropic elastic FWM must keep in memory 31 array of at least single-precision numbers (of length 4 bytes) each iteration needs to process hundreds of gigabytes. This value exceeds RAM volume of the majority of modern cluster nodes and multiprocessor servers.

A wide range of methods have been developed to reduce memory size and computation complexity of FWM. Finite-difference schemes of higher order, non-uniform grid, finite element method, spectral, and pseudo-spectral approaches are among the most popular ones [77]. However, there are practical requirements to account inhomogeneities of the medium, which are much smaller than the wavelength. Similarly, the simulation time step is limited by the output seismogram sampling (2-4 ms). As result the acceleration effect of complex techniques for detailed models rarely covers the additional computational cost. An overview of modern finite-difference schemes for 3D anisotropic elastic modeling was presented in [51].

In spite of the long history and diversity of approaches the 3D FWM, until recently, was mostly the field of scientific research. Now there is an opportunity to simulate wavefield propagation within acceptable time due to the progress in high-performance computing, especially on general-purpose graphics cards (GPU). For example, [46] reported reaching 25-50 times acceleration of the anisotropic elastic modeling on NVIDIA GPU-cluster. However, the computational cost of modeling a full set of synthetic 3D seismic data (which includes thousands of wavefields) still substantially exceeds the practical limitations.

3D FWM can be simplified, if all properties of the medium are fixed along one direction (Y). Models of such type (Fig. 1.15) are called "two and a half dimensional” (2.5D). Limitation of 2.5D makes it impossible to synthesize seismograms corresponding to real objects. Consequently, 2.5D does not help to optimize survey design in terms of target horizon illumination (which can do RT). However, 2.5D is useful to investigate the influence of multi-component acquisition and density of survey on interpretation of complex media. 2.5D FWM accounts thin-layering, fracturing and arbitrary anisotropy of models too complex for ray tracing.

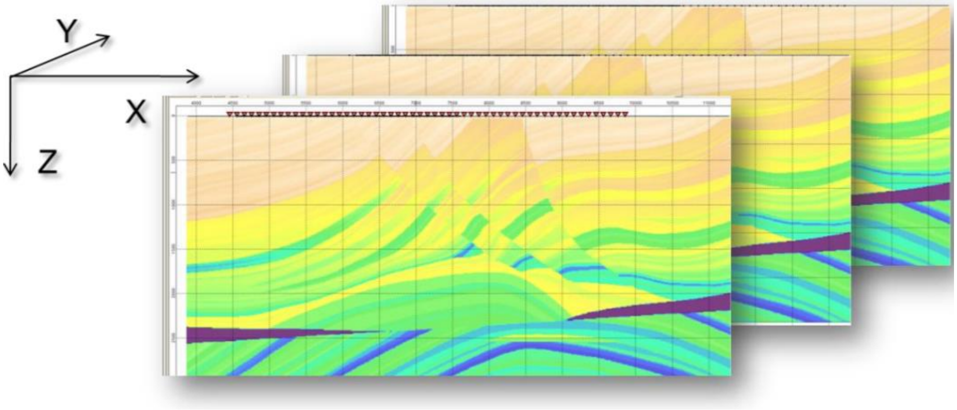


Fig. 1.15. The 2.5D model

Since the 2.5D-seismogram does not change if both the sources and receivers similarly shift along the axis Y, in such a model it is sufficient to simulate only a single shot line. Other shot lines are easy generated by copying traces with change of coordinates in their headers. As result, in the case of 2.5D model the computations can be reduced by tens, sometimes hundreds times, while all the 3D dataset features persist for further processing (Fig. 1.16).

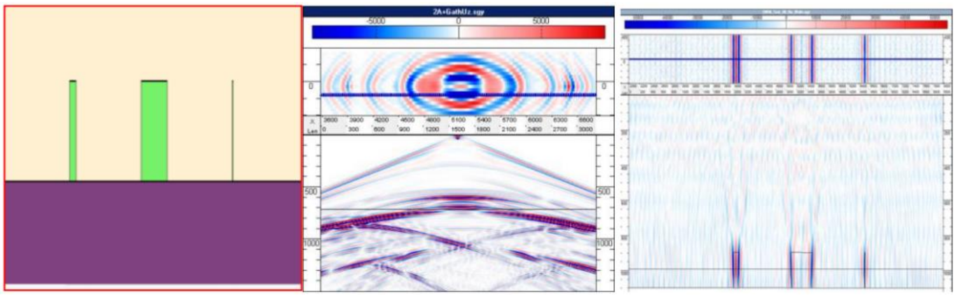


Fig. 1.16. The 2.5D model (left), its 3D synthetic seismogram (middle) and the resulted cube of duplex wave migration which images vertical boundaries (right)

3D elastic anisotropic system of equations expresses Hooke's law combined with Newton's 2nd law:

$$\begin{cases} \rho(X)\dfrac{\partial u_n(X,t)}{\partial t} = \dfrac{\partial \tau_{nk}(X,t)}{\partial x_k} + f_n(X,t) \\ \dfrac{\partial \tau_{mn}(X,t)}{\partial t} = \Lambda_{mnkq}(X)\dfrac{\partial u_k(X,t)}{\partial x_q} + \dfrac{\partial M_{mn}(X,t)}{\partial t} \end{cases} \tag{1.48}$$

where $X = (x_1 \quad x_2 \quad x_3)$ means a point in 3D space, u_n denote displacement velocities, τ_{nm} are components of a stress tensor. Parameters of the geology media are stiffness tensor Λ and density ρ. Source signal is encoded by the vector functions of source forces f_n and moment forces M_{nm}. Both type forces are zero out of source.

In 2.5D case for arbitrary y $\Lambda(x,y,z)=\Lambda(x,0,z)$ and $\rho(x,y,z)=\rho(x,0,z)$. So it is convenient to represent the wavefield in Fourier domain by its decomposition along the axis x_2:

$$u_n(X,t)=\int_{-\infty}^{+\infty}\tilde{u}_n(\tilde{X},t)e^{i\omega y}dy\,,\ \tau_{mn}(X,t)=\int_{-\infty}^{+\infty}\tilde{\tau}_{mn}(\tilde{X},t)e^{i\omega y}dy \tag{1.49}$$

where $\tilde{X} = (x_1 \quad \omega \quad x_3)$. The system (1.47) can be re-written in the space Fourier domain ($q \in \{1,3\}$):

$$\begin{cases} \rho\left(\tilde{X}\right)\dfrac{\partial\tilde{u}_n\left(\tilde{X},t\right)}{\partial t}=\dfrac{\partial\tilde{\tau}_{nq}\left(\tilde{X},t\right)}{\partial x_q}+i\omega\tilde{\tau}_{n2}\left(\tilde{X},t\right)+\tilde{f}_n\left(\tilde{X},t\right) \\ \dfrac{\partial\tilde{\tau}_{mn}\left(\tilde{X},t\right)}{\partial t}=\Lambda_{mnkq}(X)\dfrac{\partial\tilde{u}_k\left(\tilde{X},t\right)}{\partial x_q}+\Lambda_{mnk2}(X)i\omega\tilde{u}_2\left(\tilde{X},t\right)+\dfrac{\partial\tilde{M}_{mn}\left(\tilde{X},t\right)}{\partial t} \end{cases} \tag{1.50}$$

Numeric wave propagation simulation according to (1.50) in the form of the second order central finite-difference scheme on three staggered grids was described in [47].

To avoid numerical dispersion and aliasing for regular sampling one has to use reasonably big space frequency diapason and dense sampling. The equations (1.50) look very similar to (1.48). However, $\tilde{u}_n$ and $\tilde{\tau}_{mn}$ are complex numbers when u_n and τ_{mn} are real. As result implementation of the system (1.49) requires approximately two times more calculations per cell. At the same time, $\tilde{u}_n(x_1 \quad -\omega \quad x_3)$ is complex conjugate to $\tilde{u}_n(x_1 \quad \omega \quad x_3)$, and $\tilde{\tau}_n(x_1 \quad -\omega \quad x_3)$ is complex conjugate to $\tilde{\tau}_n(x_1 \quad \omega \quad x_3)$. Hence we don't need to simulate wave propagation for negative frequencies. Thus calculations for one pair of symmetric frequencies ω and $-\omega$ by (1.50) require approximately the same number of operations and expands approximately the same time as (1.48) for each $y=const$. (The spatial/time samplings are supposed to be equal as determined by stability and dispersion conditions.) Hence acceleration rate due to 2.5D can be estimated as a ratio of the grid size in y direction to the number of non-negative frequencies ω in the wavefield discrete Fourier transform. To avoid numerical dispersion the space sampling

$$\Delta y_{3D}=V_{\min}/Wf_{\max}\ , \tag{1.51}$$

where $W=5..15$ is the minimal number of grid points per wavelength. But for Fourier transform precise reconstruction is provided by regular sampling of Nyquist frequency:

$$\Delta y_{2.5D}\approx V_{\min}/2f_{\max}\Rightarrow\omega_{\max}=\pi/\Delta y\approx 2\pi f_{\max}/V_{\min}\ . \tag{1.52}$$

Y offset from the shot point to the nearest mirror source is $y^i=2\pi/\Delta\omega$. To avoid signals from mirror sources on receivers with offset $y_{\max}$ one must support

$$y^i-y_{\max}>t_{\max}V_{\max}\Rightarrow\Delta\omega<2\pi/\left(y_{\max}+t_{\max}V_{\max}\right). \tag{1.53}$$

$$
\begin{aligned}
N_\omega &= \frac{\omega_{\max}}{\Delta\omega} + 1 = \frac{2\pi f_{\max}}{V_{\min}} \cdot \frac{y_{\max} + t_{\max} V_{\max}}{2\pi} + 1 = \\
&= \frac{y_{\max} f_{\max}}{V_{\min}} + \frac{V_{\max}}{V_{\min}} f_{\max} t_{\max} \approx \frac{N_y}{2W} + \frac{V_{\max}}{W \Delta y_{3D}} t_{\max}
\end{aligned} \quad . \tag{1.54}
$$

The computational complexity cut down because of frequency decomposition (1.50) is

$$
Speedup = \frac{N_y}{N_\omega} \approx W \Big/ \left(\frac{1}{2} + \frac{V_{\max} t_{\max}}{y_{\max} - y_{\min}} \right) \tag{1.55}
$$

Time of recording $t_{\max}$, max velocity $V_{\max}$, diapason of crossline offsets from $y_{\min}$ and $y_{\max}$ are problem dependent. For deep target objects $t_{\max}$ is big and under fixed offsets the theoretical speedup (1.55) becomes smaller than 1. In other words computational complexity can be higher than for full 3D solution of the same problem (Fig. 1.17).

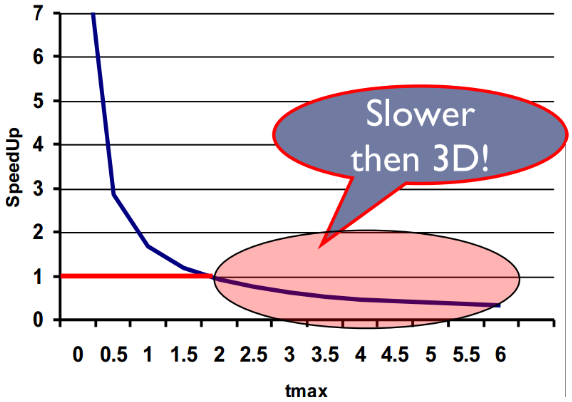


Fig. 1.17. The speedup in terms of number of operations needed for 2.5D simulation of a wavefield relatively to a full 3D method

But the memory requirements are decreased dramatically. The spectral decomposition of a 2.5D problem along the Y axis practically eliminates the lack of memory issue (Table 1.1). It reduces a 3D problem to a set of quasi-2D subproblems. Just 160 bytes per cell of 2D grid is enough to keep in memory both 2.5D model and all the wave field components of a quasi-2D subproblem [50]. This provides conditions to use GPU in the most efficient mode without data reloading. The simulated quasi-2D seismograms are combined to target 3D seismogram by the inverse Fourier transform. Then 3D seismograms of the only simulated shot line are replicated according to the survey design.

Table 1.1. Comparison of 3D and 2.5D elastic anisotropic FWM for an example from [51].

Parameter	3D	2.5D
Simulation grid size	2.5 km × 2.5 km × 2.5 km	
Space sampling	2.5 m	
Number of subproblems	1	581 (for $t_{\max}$ =2 sec)
Required memory volume	216 GB	432 MB

Song and Williamson [68] first time used a similar technique for the 2.5D-acoustic approximation of constant density models. Chao and Grinhalg [15] derived a stability criterion and proposed a method of absorbing boundary implementation. Neto and Costa [62] described a 2.5D method for elastic isotropic and transversely isotropic medium. 2.5D FWM for an arbitrary 3D TTI anisotropy and fracturing was described in [63] and [47].

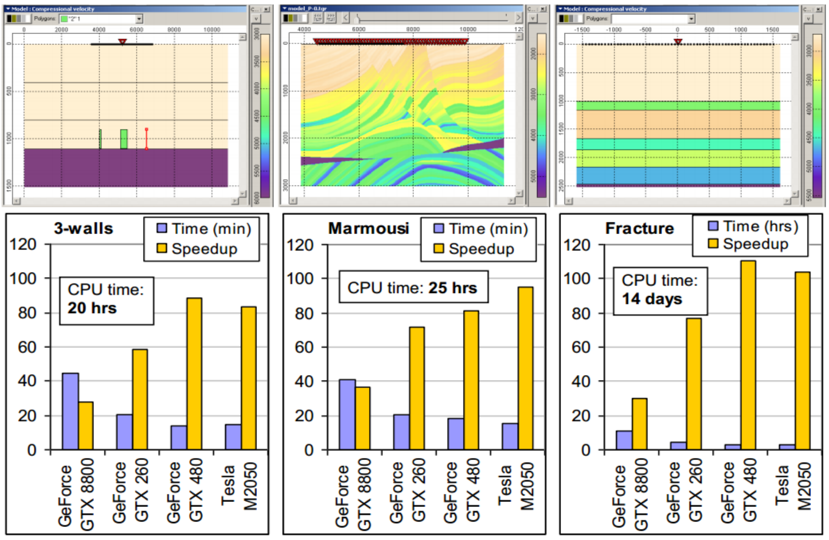


Fig. 1.18. GPU simulation time and speedup over a CPU core for different 2.5D models

The spectral decomposition of the 2.5D FWM problem can be used for calculations on conventional computer clusters [48]. However, practically significant acceleration of the full-wave 3D seismic data synthesis can be achieved with synergy of 2.5D models, spectral decomposition based simulation and application of modern GPU technologies. Experiments (Fig. 1.18) show expected preference of modern GPUs over outdated GTX 8800. Mid level GTX480 unexpectedly approached professional Tesla. Probably it's because of single precision math used. Speedup has been estimated relative to single core of CPU Intel Xeon E5345 (2.33GHz).

Despite high speedup provided by the method, one needs hundreds or even thousands GPU hours to solve an average size multi-shot 2.5D FWM problem. Undersampling is a natural way for further acceleration of the 3D seismogram synthesis.

Synthetic seismograms may be more or less sparse in time/space domain. Their compressibility depends on complexity of the input model. Simple theoretical models such as “3 walls” or “Fracture” from Figure 1.18 contain few reflecting boundaries and correspondent seismograms are sparse. But models which approximate real objects (such as “Marmousi” model from the same picture) usually contain multiple reflecting interfaces. Their seismograms aren’t sparse in time/space domain. However, both simple and complex seismograms are usually compressible in wavelet domain (Fig. 1.19, 1.20).

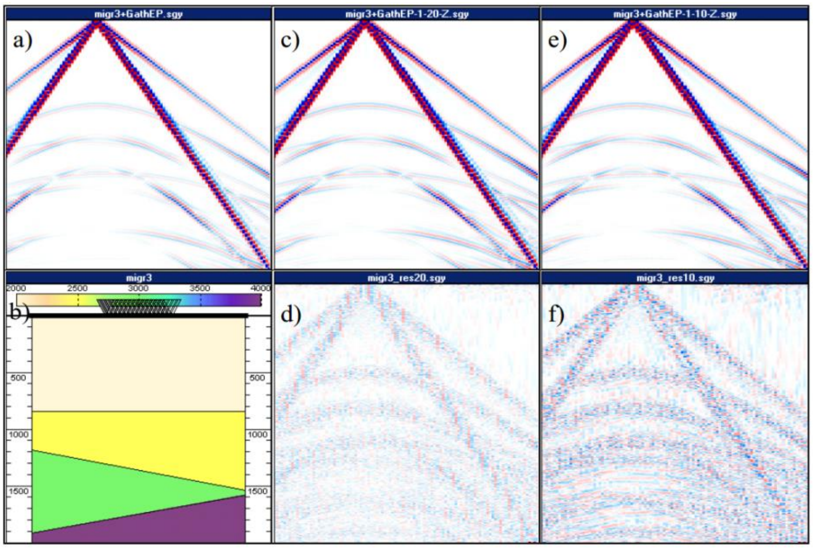


Fig. 1.19. Wavelet compression of the seismogram (a) generated for the model (b); 12% (8.3 times) compressed seismogram (c) and residuals: RMSE=8.85·10 $^{-6}$ (d); 8% (12.5 times) compressed seismogram. (e) and residuals: RMSE=1.05·10 $^{-5}$ (f)

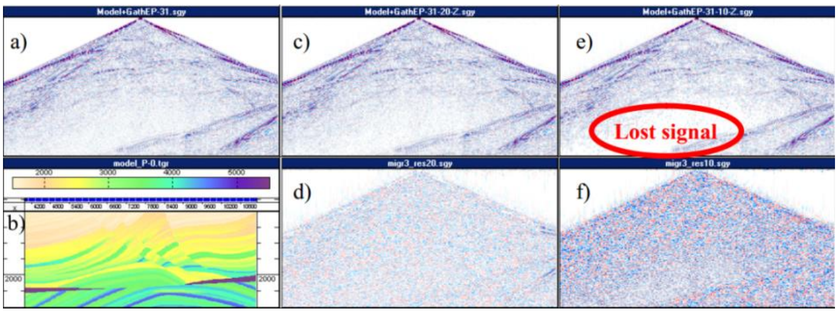


Fig. 1.20. Wavelet compression of the seismogram (a) of the “Marmousi” model (b); 12% (8.3 times) compressed seismogram (c) and residuals: RMSE=1.63·10 $^{-4}$ (d); 8% (12.5 times) compressed seismogram. (e) and residuals: RMSE=8.42·10 $^{-4}$ (f)

Figures 1.19, 1.20 show results of the classical wavelet compression algorithm of 3 steps [28]: Discrete Wavelet Transform (DWT), quantization and coding. Fast 1D algorithm of Mallat [55] has been used for DWT of Daubechies [23]. Noise is significantly highlighted. (See Figure 1.21 to estimate real distortion because of the compression.) Residual RMSE<10^{-4} provided stable high quality of compressed image for both simple and complex models.

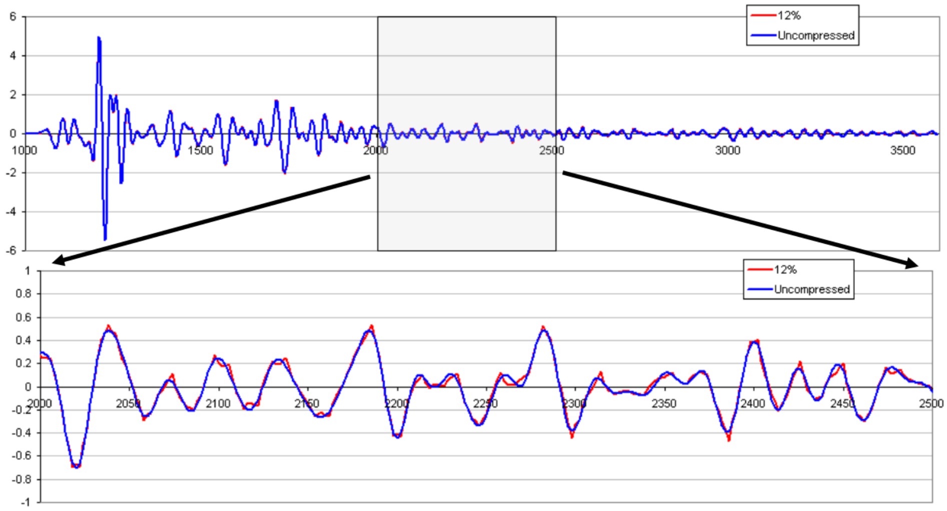


Fig. 1.21. Wavelet compression of "Marmousi" synthetic seismogram: 12% (8.3 times) compressed trace compared to the original one in two scales

Despite the high sparsity promotion of DWT and its known incoherency with DFT it is really a bad choice for this recovery problem. Thin spectra of many DWT atoms are unfavorable for robust approximation of the irregularly sampled signal spectra regardless of the reconstruction algorithm.

Let's first concentrate on simple models. Their compressibility in time/space domain creates conditions for use of irregular sampling in combination with the stepwise thresholding described in the section 1.2. Here irregular sampling means calculation of a smaller number of randomly selected space decomposition frequencies.

Figure 1.22 illustrates application of different undersampling techniques to a simple 2.5D model with 3D TTI anisotropy presented in Table 1.1. The synthetic seismograms were generated by simulation of 2 sec wavefield propagation. The 2.5D simulation complexity of this problem in the correct regular sampling case is about 2 times smaller then for its 3D analogue (but acceleration is much higher).

Results of 2.5D simulation for correct sampling are shown in the column (b) of Figure 1.22. Regular 4 times undersampling (1.22c) results in strong noise signals from two mirror sources. Random 4 times undersampling (1.22d) dissipates the mirror signals. But noise amplitudes left relatively big, especially for small times (top part of the bottom image). Obvious way to decrease the noise level is interpolation of absent frequencies between calculated ones. It is also a method of signal reconstruction valid for smooth functions. Trigonometric local interpolation of spectrum (1.22e) provides significantly lower noise for random 4 times undersampling then other methods. Signal to noise energy ratio above 100 creates conditions for recovery by simple denoising.

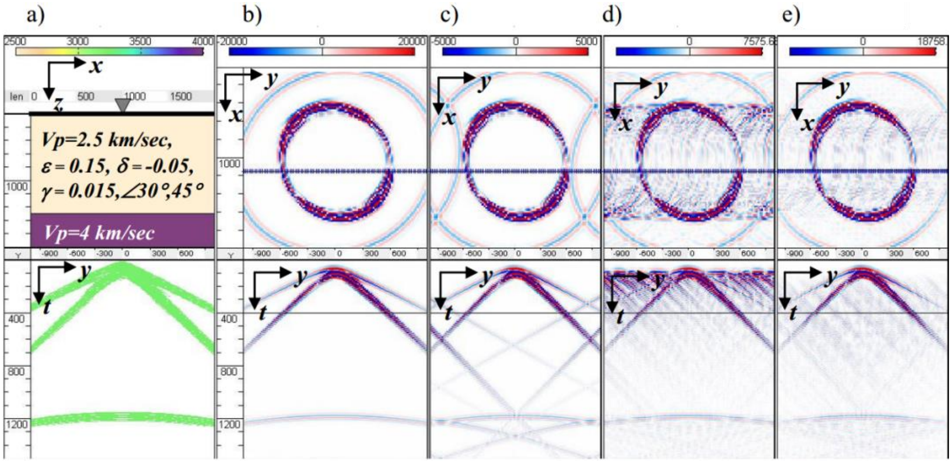


Fig. 1.22. Model and signal mask (a), correct sampling (b), regular undersampling (c), random undersampling (d), random undersampling with spectrum interpolation (e). Top pictures of columns (b), (c), (d), (e) show 400 ms slice of the seismogram cubes; their bottom pictures show the cube vertical sections at X=800 m

Figure 1.23 demonstrates efficiency of random undersampling for various parameters. There are 3 methods of selection space frequencies to simulate: random integer numbers, jittered regular grid and traditional regular grid. We combined them with 3 method of interpolation between frequencies: resampling (lattice diagram), linear (piecewise-linear diagram) and trigonometric interpolation (smooth diagram). Besides, we varied decimation of low and high frequencies. Both linear and trigonometric interpolation of frequencies stably improved the signal to noise energy ratio for about an order.

The diagrams of Figure 1.23 show that additional 3-4 times acceleration of 2.5D FWM can be obtained for simple models by random undersampling with spectra interpolation for the cost of about 1% noise. But simple thresholding is not suitable for reconstruction of seismograms as the wave energy is distributed over the wave front which area is approximately in inverse proportion to time. Non-linear threshold is requested.

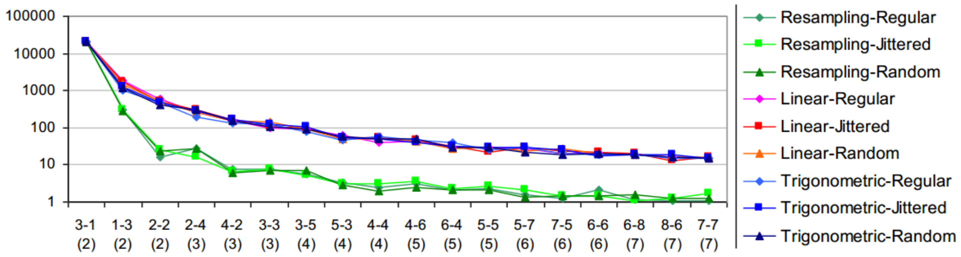


Fig. 1.23. Signal to noise ratio for 3 undersampling and 3 interpolation methods. The X axis is underwrote by the average decimation description: first of low frequencies, then of high frequencies and the average decimation in round brackets below

The problem of recovery of a signal which consists of a finite number of short impulses from partial observation of its Fourier transform is known for decades. In 1938 Beurling proposed a method for extrapolating these observations to determine the entire Fourier transform [5]. The Beurling's approach will correctly recover the entire Fourier transform (of this non-bandlimited signal) from any sufficiently large piece of its Fourier transform. His approach to find the signal with smallest ℓ_1 norm among all signals agreeing with the acquired Fourier measurements bears a remarkable resemblance to some of the modern algorithms used in CS.

1.6. Conclusion

We briefly discussed the area of compressive sampling from the viewpoint of discrete signal acquisition and reconstruction from undersampled records. However, the area of research and application of CS is much wider.

E.g., an efficient undersampled acquisition for a specific class of analog signals having a finite number of harmonics can be implemented by random demodulator [71]. It uses the structurally subsampled matrices (when linear combinations of multiple signal values are sensed instead of separated signal values) to implement compressive Analog to Digital Converters (ADC).

The sparse vector recovery problem we discussed till now is only the first of known CS problems. Among them low-rank matrix recovery and low-rank matrix completion problems are active research areas. Some of the problems are:

- Rank Minimization problem: for sparse $\mathbf{X}\in\mathrm{R}^{m\times n}$, a few linear measurements $\mathbf{y}\in\mathrm{R}^{p}$ and a linear map $F:\mathrm{R}^{m\times n}\rightarrow\mathrm{R}^{p}$ find $\arg\min\mathrm{Rank}(\mathbf{X})$ subject to $F(\mathbf{X})=\mathbf{y}$.
- Trace Minimization problem: for symmetric sparse $\mathbf{X}^{\mathrm{T}}=\mathbf{X}\succ 0$ minimize the sum of eigenvalues $\mathrm{Tr}(\mathbf{X})=\sum\lambda_i(\mathbf{X})$, find $\arg\min\mathrm{Tr}(\mathbf{X})$ subject to $\mathbf{X}\in\Gamma$.

- Nuclear Norm Minimization problem: for sparse $\mathbf{X} \in \mathrm{R}^{n \times n}$ minimize the sum of singular values $\|\mathbf{X}\|_* = \sum \sigma_i(\mathbf{X}) = \sum \lambda_i(\mathbf{X}^{\mathrm{T}}\mathbf{X})$, find $\arg\min\|\mathbf{X}\|_*$ subject to $\mathbf{X} \in \Gamma$.

CS stimulates new approaches to traditional application problems such as resolution increasing, acquisition by minimal number of sensors, acquisition by sensors of limited frequency and most difficult problems of signal processing [32]. In machine learning the ability of random orthogonal linear projections to preserve all distances between points opens new horizons for classical problems of artificial intelligence such as classification or distributed representation of holistic structures. Performance benefits of number of sample reduction during a numerical experiment can overcome the enhanced computational effort for the CS reconstruction.

Many other types of problems can be attacked with the compressive sensing techniques for recovery of internally structured data from incomplete and inaccurate samples.

1.7. References

1. Alterman Z., Karal F. C., *Propagation of elastic waves in layered media by finite-difference methods*, “Bulletin of the Seismological Society of America”, 1968 vol. 58, pp. 367-398.
2. Baraniuk R. G., Cevher V., Duarte M., Hegde C., *Model-based compressive sensing*, “IEEE Trans. on Inf. Theory”, 2010 vol. 56(4),pp. 1982-2001.
3. Baraniuk R. G., *Compressive sensing*, “IEEE Signal Processing Magazine”, 2007 vol. 07, pp. 118-124.
4. Baraniuk R., Davenport M., DeVore R., Wakin M., *A Simple Proof of the Restricted Isometry Property for Random Matrices*, “Constructive Approximation”, 2008 vol. 28(3), pp. 253-263.
5. Beurling A., *Sur les int?grales de Fourier absolument convergentes et leur application ? une transformation fonctionelle*, “Proc. Scandinavian Math. Congress”, Helsinki, Finland, 1938.
6. Blumensath T., Davies M., *Iterative hard thresholding for compressed sensing*, “Appl. Comp. Harmonic Anal.”, 2009 vol. 27(3), pp. 265-274.
7. Bresler Y., Feng P., *Spectrum-blind minimum-rate sampling and reconstruction of 2-D multiband signals*, Proc. IEEE Int. Conf. Image Processing (ICIP), Zurich, Switzerland, 1996.
8. Candes E. J., Romberg J., *Quantitative robust uncertainty principles and optimally sparse decompositions*, “Foundations of Computational Mathematics”, 2006 vol. 6, pp. 227-254.
9. Candes E. J., Romberg J., *Sparsity and incoherence in compressive sampling*, “Inverse Prob.”, 2007 vol. 23(3), pp. 969-985.

10. Candes E. J., Romberg J., Tao T., *Stable signal recovery from incomplete and inaccurate measurements*, "Comm. Pure Appl. Math.", 2006 vol. 59(8), pp. 1207-1223.
11. Candes E. J., Tao T., *Decoding by linear programming*, "IEEE Trans. Info. Theory", 2005 vol. 51 (12), pp. 4203-4215.
12. Candes E. J., Tao T.C., *Near Optimal Signal Recovery From Random Projections: Universal Encoding Strategies?* arXiv e-print (arXiv:math/0410542), http://arxiv.org/pdf/math/0410542v3.pdf, 2004-2006.
13. Candes E. J., *The restricted isometry property and its implications for compressed sensing*, "Comp. Rendus Mathematique", 2008 vol. 346(9-10), pp. 589-592.
14. Candes E. J., Wakin M.B., *An introduction to compressive sampling*, "IEEE Signal Processing Magazine", 2008 vol. 25(2), pp. 21-30.
15. Cao S., Greenhalgh S., *2.5D modeling of seismic wave propagation: Boundary condition, stability criterion, and efficiency*, "Geophysics", 1998 vol. 63 (6), pp. 2082-2090.
16. Caratheodory C., *Ueber den Variabilitaetsbereich der Koezienten von Potenzreihen, die gegebene Werte nicht annehmen*, "Math. Ann.", 1907 vol. 64, pp. 95-115.
17. Cerveny V., *Seismic Ray Theory*, "Cambridge University Press", 2001, 722.
18. Chartrand R., *Exact reconstruction of sparse signals via nonconvex minimization*, "IEEE Signal Processing Lett.", 2007 vol. 14(10), pp. 707-710.
19. Chen S. S., Donoho D. L., Saunders M. A., *Atomic decomposition by Basis Pursuit*, "SIAM J. Sci. Comput.", 1999 vol. 20(1), pp. 33-61.
20. Chen S. S., Donoho D. L., Saunders M. A., *Atomic decomposition by Basis Pursuit*, "SIAM Rev.", 2001 vol. 43(1), pp. 129-159.
21. Cohen A., Dahmen W., DeVore R., *Compressed sensing and best k-term approximation*, "J. Amer. Math. Soc.", 2009 vol. 22(1), pp. 211-231.
22. Coifman R., Geshwind F., Meyer Y., *Noiselets*, "Appl. Comput. Harmon. Anal.", 2001 vol. 10(1), pp. 27-44.
23. Daubechies I., *Orthonormal Bases of Compactly Supported Wavelets*, "Communication on Pure Applied Mathematics", 1988 vol. 41, pp. 906-966.
24. Davenport M.A., Duarte M.F., Eldar Y.C., Kutyniok G., *Introduction to Compressed Sensing*, Compressed Sensing: Theory and Applications, Cambridge University Press, 2011.
25. Davis G., Mallat S., Avellaneda M., *Adaptive greedy approximation*, "J. Constr. Approx.", 1997 vol. 13, pp. 57-98.
26. Donoho D.L., *Compressed sensing*, "IEEE Transactions on Information Theory", 2006 vol. 52, pp. 1289-1306.
27. Donoho D.L., Elad M., *Optimally sparse representation in general (nonorthogonal) dictionaries via ?1 minimization*, "Proc. Natl. Acad. Sci.", 2003 vol. 100(5), pp. 2197-2202.

28. Donoho D.L., Ergas R.A., Villasenor J .D., *High performance seismic trace compression*, “Proc. of SEG '95 Meeting”, 1995, pp. 160-163.
29. Donoho D.L., Huo X., *Uncertainty principles and ideal atomic decomposition*, “IEEE Transactions on Information Theory”, 2001vol. 47, pp. 2845-2862.
30. Donoho D.L., Tsaig Y., Drori I., Starck J.-L., *Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit*, Stanford Statistics Department Technical Report TR-2006-2, http://stat.stanford.edu/~idrori/StOMP.pdf, 2006.
31. Donoho D.L., Vetterli M., DeVore R. A., Daubechies I., *Data compression and harmonic analysis*, “IEEE Trans. Info. Theory”, 1998 vol. 44(6), pp. 2433-2452.
32. Duarte M. F., Eldar Y. C., *Structured Compressed Sensing*, From Theory to Applications, http://arxiv.org/pdf/1106.6224.pdf, 2011.
33. Duijndam A.J.W., Schonewille M.A., *Nonuniform fast Fourier transform*, “Geophysics”, 1999 vol. 64, pp. 539-551.
34. Ellenberg J., Fill in the Blanks: *Using Math to Turn Lo-Res Datasets Into Hi-Res Samples*, Wired, 18(3), http://www.wired.com/magazine/18-03/, 2010.
35. Fornasier M., Rauhut H., *Compressive sensing, in Part 2 of the Handbook of Mathematical Methods in Imaging* (O. Scherzer Ed.), Springer, http://www.ricam.oeaw.ac.at/people/page/fornasier/CSFornasierRauhut.pdf, 2011.
36. Friedman J. H., Stuetzle W., *Projection pursuit regressions*, “J. Amer. Statist. Assoc.”, 1981 vol. 76(376), pp. 817-823.
37. G□l□nay N., *Seismic trace interpolation in the Fourier transform domain*, “Geophysics”, 2003 vol. 68(1), pp. 355-369.
38. Garnaev A., Gluskin E., *On widths of the Euclidean ball*, “Dokl. AN SSSR”, Moscow, Russia, 1984, vol. 277, pp. 1048-1052.
39. Gazit S., Szameit A., Eldar Y. C., Segev M., *Super-resolution and reconstruction of sparse sub-wavelength images*, “Opt. Express”, 2009 vol. 17(26), pp. 23920-23946.
40. Gjoystdal H., Iversen E., Laurain L., Lecomte I., Vinje V., Aesteboel K., *Review of ray theory applications in modelling and imaging of seismic data*, “Studia geophysica et geodaetica”, 2002 vol. 46, pp. 113-164.
41. Gluskin E., *Norms of random matrices and widths of finite-dimensional sets*, “Math. USSR-Sb.”, Moscow, Russia, 1984 vol. 48, pp. 173-182.
42. Gorodnitsky I., J. *George, Rao B., Neuromagnetic source imaging with FOCUSS: A recursive weighted minimum norm algorithm*, “Electroencephalography and Clinical Neurophysiology”, 1995 vol. 95(4), pp. 231-251.
43. Gorodnitsky I., Rao B., *Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm*, “IEEE Trans. Signal Processing”, 1997 vol. 45(3), pp. 600-616.

44. Hennenfent G., Herrmann F.J., *Simply denoise: wavefield reconstruction via jittered undersampling,* “Geophysics”, 2008 vol. 73(3), pp. 19-28.
45. Kashin B., *Diameters of some finite-dimensional sets and classes of smooth functions,* “Math. USSR, Izv.”, Moscow, Russia, 1977 vol. 11, pp. 317-333.
46. Komatitsch D., Michea D., Erlebacher G., Goeddeke D., *Running 3D Finite-difference or Spectral-element Wave Propagation Codes 25x to 50x Faster Using a GPU Cluster,* Extended Abstracts of 72th EAGE Conf. & Exhibition, 2010.
47. Kostyukevych A., Marmalevskyi N., Roganov Y., Tulchinsky V., *Anisotropic 2.5D - 3C finite-difference modeling*, “Extended Abstracts of 70th EAGE Conf. & Exhibition”, 2008, 43.
48. Kostyukevych A., Roganov Y., *2.5D forward modeling: a cost effective solution that runs on small computing systems*, “ASEG Extended Abstracts”, 2010, pp. 1-4.
49. Kotelnikov V., *On the carrying capacity of the ether and wire in telecommunications (in Russian),* Izd. Red. Upr. Svyazi RKKA, Moscow, Russia, 1933.
50. Lavreniuk S., Roganov Y., Tulchinsky V., Kolomiets O., *Synergy of 2.5D approach and grid technology for synthesis of realistic 3D/3C seismograms in anisotropic media*, Extended Abstracts of 73rd EAGE Conf. & Exhibition, 2011, 289.
51. Lisitsa V., Vishnevskiy D., *Lebedev scheme for the numerical simulation of wave propagation in 3D anisotropic elasticity,* “Geophysical Prospecting”, 2010 vol. 58, pp. 619-635.
52. Lu Wu-Sheng, *Compressed Sensing and Sparse Signal Processing,* online, http://shmathsoc.org.cn/lu2010/Lecture Notes/Lecture Notes CS LWS Final.pdf, 2010.
53. Lu Y., Zhang X., Douraghy A., Stout D., *Source reconstruction for spectrally-resolved bioluminescence tomography with sparse a priori information*, Optics Express, 2009 vol. 17(10), pp. 8062-8080.
54. Lustig M., Donoho D.L., Santos J.M., Pauly J.M., *Compressed Sensing MRI*, http://www.stanford.edu/~mlustig/CSMRI.pdf, 2007.
55. Mallat S. A., *Theory for multiresolutional signal decomposition: the wavelet representation*, “IEEE Trans. Pattern Analysis and Machine Intelligence”, 1989 vol. 7, pp. 674-693.
56. Mallat S. G., Zhang Z., *Matching pursuits with time-frequency dictionaries*, “IEEE Trans. Signal Process.”, 1993 vol. 41(12), pp. 3397-3415.
57. Miller A. J., *Subset Selection in Regression, 2nd ed. London,* Chapman and Hall, 2002.
58. Muthukrishnan S., *Data Streams,* Algorithms and Applications. Boston: Now Publishers, 2005.

59. Narayanankutty K. A., Soman K. P., *Understanding Theory Behind Compressed Sensing*, “Int. J. Sensing, Computing & Control”, 2011 vol. 1(2), pp. 81-92.
60. Needell D., Tropp J. A., *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples*, “Appl. Comp. Harmonic Anal.”, 2009 vol. 26(3), pp. 301-321.
61. Needell D., Tropp J. A., *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples*, California Inst. Tech., ACM Report 2008-01, 2008.
62. Neto F., Costa J., *2.5D anisotropic elastic finite-difference modeling*, Extended Abstracts of 76th SEG Annual Meeting: 2275-2279, 2006.
63. Neto S .F., Costa J., Novais A., *2.5D Elastic Anisotropic Finite-Difference Modeling*, Extended Abstracts of 10th Int. Congress of the Brazilian Geophysical Society, 2007.
64. Nyquist H., *Certain topics in telegraph transmission theory, Trans*. “AIEE”, 1928 vol. 47, pp. 617-644.
65. Prony (G. Riche, baron de Prony), *Essai expérimental et analytique sur les lois de la Dilatabilité des uides élastiques et sur celles de la Force expansive de la vapeur de l'eau et de la vapeur de l'alkool, à diérentes températures*, “J. de l'École Polytech., Floréal et Prairial III”, 1795 vol. 1(2), pp. 24-76.
66. Schniter P., Potter L. C., Ziniel J., *Fast Bayesian matching pursuit: model uncertainty and parameter estimation for sparse linear models*, submitted to IEEE Trans. on Signal Proc., http://www.ece.osu.edu/~schniter/pdf/-tsp09_fbmp.pdf, 2008.
67. Shannon C., *Communication in the presence of noise*. “Proc. Institute of Radio Engineers”, 1949 vol. 37(1), pp. 10-2.
68. Song Z., Williamson F.R., *Frequency-domain acoustic-wave modeling and inversion of crosshole data: Part 1 - 2.5-D modeling method*, “Geophysics”, 1995 vol. 60(3), pp. 784-795.
69. Spitz S., *Seismic trace interpolation in the F-X domain*, “Geophysics”, 1991 vol. 67, pp. 890-794.
70. Tropp J. A., Gilbert A. C., *Signal recovery from random measurements via orthogonal matching pursuit*,” IEEE Trans. Info. Theory”, 2007 vol. 53(12), pp. 4655-4666.
71. Tropp J. A., Laska J. N., Duarte M. F., Romberg J. K., Baraniuk R. G., *Beyond Nyquist: Efficient sampling of sparse bandlimited signals*, “IEEE Trans. Info. Theory”, 56(1), 520-544, http://users.cms.caltech.edu/~jtropp/papers/TLDRB10-Beyond-Nyquist.pdf, 2010.
72. Tropp J. A., Wright S. J., *Computational Methods for Sparse Solution of Linear Inverse Problems*, “Proc. of the IEEE”, 2010 vol. 98 (6), pp. 948-958.
73. Unser M., *Sampling - 50 years after Shannon*, “Proc. of the IEEE”, 2000 vol. 88, pp. 569-587.
74. Verdu S., *Multiuser detection*, Cambridge University Press, 1998.

75. Versprille K.J., *Computer-aided design applications of the rational B-spline approximation form*, Ph.D. dissertation, Syracuse University, 1975.
76. Vetterli M., Marziliano P., Blu T., *Sampling signals with finite rate of innovation*, “IEEE Trans. Signal Processing”, 2002 vol. 50(6), pp. 1417-1428.
77. Virieux J. (et al.), *Seismic wave modeling for seismic imaging*, “The Leading Edge”, 2009 vol. 28(5), pp. 538-544.
78. Whittaker E., *On the functions which are represented by the expansions of the interpolation theory*, “Proc. Royal Soc. Edinburgh, Sec. A”, 1915 vol. 35, pp. 181-194.
79. Wipf D., Rao B., *Sparse Bayesian learning for basis selection*, “IEEE Trans. Signal Processing”, 2004 vol. 52(8), pp. 2153-2164.
80. Xu S., Zhang Y., Pham D., Lambar G.? *Antileakage Fourier transform for seismic data regularization*, “Geophysics”, 2005 vol. 70(4), pp. 87-95.
81. Zwartjes P.M., Sacchi M.D., *Fourier reconstruction of nonuniformly sampled, aliased seismic data*, “Geophysics”, 2007 vol. 72(1), pp. 21-32.

2. METHODS OF WATERMARKING

Natalya Koshkina, Valeriy Zadiraka, Andrzej Smolarz, Paweł Komada

2.1. Introduction

Digital watermarking is the process of embedding some information (that is called watermark or mark) into a digital multimedia content such way that makes this mark imperceptible and irremovable after some modifications of the carrier. Often, the host signal that carries the watermark is also called a cover object. Digital watermarking systems are usable for a wide range of practical applications, such as noise-robust authentication of audio and visual data (in particular, for the integrity control of CCTV- or telephone conversations recordings), authentication of the data owner (copyright protection), authentication of the data source, broadcasts monitoring, copying control etc.

In general, a digital watermarking system can be considered as a set $\Sigma = (X, W, K, Y, E, D)$ of original cover objects X, watermarks W, keys K, marked cover objects Y; and watermark embedding and extracting transformations which are denoted as E and D accordingly.

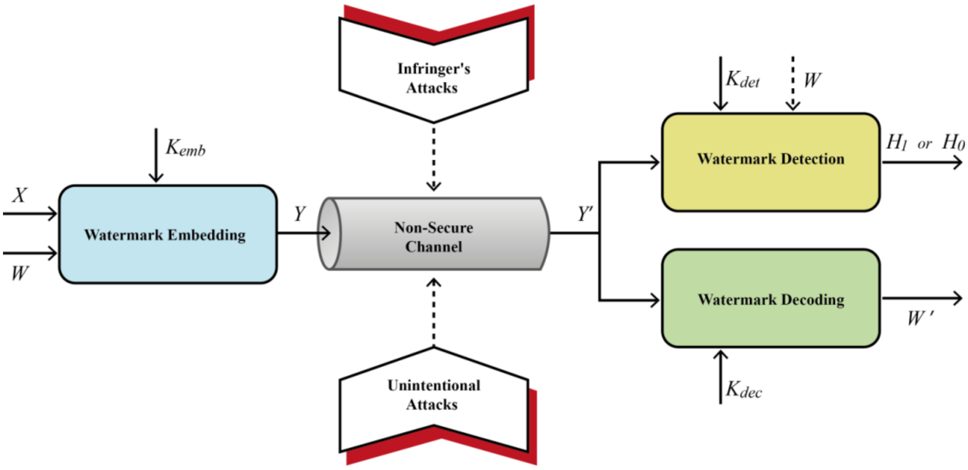


Fig. 2.1. General model of the watermark life-cycle with embedding, attacking, and detecting/decoding transformations

According to Fig. 2.1 that presents general model of the watermark life-cycle, a certain user of watermarking system starts an algorithm of mark *W* embedding into cover object *X* with the key K_{emb}. The marked cover object *Y* is the result of this algorithm which will be transmitted through the non-secure channels. In general, information that is transmitted through the non-secure channels can be distorted by digital signal processing (DSP) procedures with a cover object, which are conscientiously aimed to improve of object quality - so-called unintentional attacks. It is also necessary to consider possible existence of an infringer, who intentionally attacks a watermark or secret parameters of the watermarking system.

The watermark extracting process consists of detecting and/or decoding of the mark. Watermark detection is choice between two hypotheses H_1 or H_0 about the presence or the absence of mark W into the signal Y'. Watermark decoding means restoring of content of the mark. Generally, watermarking systems with watermark detection demand the detector's knowledge not only a secret key, but the watermark that is being checked, too.

Watermarking systems have to satisfy the following requirements:

Imperceptibility: watermark embedding have to save perceptual quality of the original cover object. A watermark should be inaudible for audio signals, and invisible for images.

Robustness: embedded information has to be properly extractable for legal user after some modifications of the original cover, which are caused by conscientious processing operations or by intentional infringer's attacks to the watermark. According to this requirement watermarks are qualified as robust, semi-fragile and fragile.

Security: unauthorized user should not have any possibilities to estimate the secret parameters of watermarking system. According to Kirchhoff's principle, the estimate of level of the watermarking system security is based on efforts that needed to find the secret key.

Reliability (or detection rate of the watermark): It is probable for watermarking systems with a detector that the detector would not detect the watermark; and there is also a probability of its false finding in the empty cover (false alarm). The probabilities of both types of errors should be minimized; however, the reliability of the detector is characterized by the probability of false alarm.

Capacity: capacity is defined as maximum amount of data that can be embedded into the cover with keeping all the requirements to other basic characteristics of the watermarking system.

Speed: watermarking system could be used even in real-time applications, for example, in the audio streaming. The watermark embedding and extracting processes have to be fast enough to meet the requirements of these applications.

There are three basic watermark features, which caused their indispensability in comparison with cryptographic methods that solve the same practical problems. Firstly, watermark is imperceptibility and it does not demand to upsizing of the cover. Secondly, watermark is inseparable from the cover and it cannot be removed without a loss of perceptual quality of marked cover, unlike the use of the digital signature. And the last, cover and watermark are objects of same transformations, which makes available research these transformations even if watermark was distorted or removed. Unlike the digital signature that confirms the authentication of digital information only with its preservation as "bit-in-bit", an authentication watermarks let confirm authenticate data after changing of the file format or, for example, when transmission of the marked cover takes place through a noisy channel.

Watermarking systems are usually designed to provide a certain compromise within the basic requirements, where their relative importance depends on the specific use of this system. Let us review in details the problems of creating robust watermarking methods for such typical cover as audio signals and images. Special attention will be paid to the use of discrete Fourier transform and wavelet transform for the creation of watermarking methods.

2.2. Robust audio watermarking

2.2.1. Fourier and wavelet analysis of digital audio signals

Spectral analysis of digital signals and images is one of the most common tools used for creation of watermarking methods. Traditional mathematical apparatus of spectral analysis is the Fourier transform which represents the signal in the harmonic oscillations basis. Discrete Fourier transform (DFT), especially fast Fourier transform (FFT), short-time Fourier transform (STFT), discrete cosine transform (DCT), modified discrete cosine transform (MDCT) are usually used in digital signal and image processing [18, 32].

Let $f(t_n) = \{f(n)\}_0^{N-1}$ be a signal that is represented by N samples. The DFT coefficients of this signal are calculated by the formula

$$F(r) = \sum_{n=0}^{N-1} f(n) e^{\frac{-2\pi i}{N} rn}, \ r = \overline{0, N-1}. \tag{2.1}$$

For signal restoration the inverse discrete Fourier transform (IDFT) is calculated:

$$f(n) = \frac{1}{N} \sum_{r=0}^{N-1} F(r) \cdot e^{\frac{2\pi i}{N} rn}, \quad n = \overline{0, N-1}. \tag{2.2}$$

If denoting $F(r) = \mathrm{Re}(F) - i\,\mathrm{Im}(F)$, the coefficients for each spectral component in Fourier space can be divided into real and imaginary parts:

$$\mathrm{Re}(F) = \sum_{n=0}^{N-1} f(n) \cos\left(2\pi \frac{rn}{N}\right),$$

$$\mathrm{Im}(F) = \sum_{n=0}^{N-1} f(n) \sin\left(2\pi \frac{rn}{N}\right), \ r = \overline{0, N-1}. \tag{2.3}$$

An alternative is to introduce $F(r)$ as amplitude and phase spectrums:

$$|F(r)| = \sqrt{\mathrm{Re}^2(F) + \mathrm{Im}^2(F)}\,,\ \arg[F(r)] = -arctg\frac{\mathrm{Im}(F)}{\mathrm{Re}(F)}. \tag{2.4}$$

If the signal $f(n)$ is real, then next equalities take place:

$$\mathrm{Re}[X(r)] = \mathrm{Re}[X(N-r)],\ \mathrm{Im}[X(r)] = -\mathrm{Im}[X(N-r)],$$

$$|X(r)| = |X(N-r)|,\ \arg X(r) = -\arg X(N-r)\,. \tag{2.5}$$

The digital processing of audio signals often includes subband-coding that is performed with some analysis/synthesis filter bank [13, 22, 27]. An analysis filter bank is an array of band-pass filters that separates the input signal into several components where each of them is carrier of a single frequency subband. During of analysis, signal is separated into a row of contiguous frequency subbands with the following result decimation. Subband coefficients can be so or otherwise modified - and also by process of watermark embedding. Next, the reconstruction of signal is performed, including an interpolation, filtering by corresponding synthesis filter bank and adding of result.

Segmentation of the signal to the frequency subbands can be realized by applying the multiresolution wavelet decomposition:

$$f(n) = \sum_{m=0}^{N/2^j-1} a_{j,m}\varphi_{j,m}(n) + \sum_{k=1}^{j}\sum_{m=0}^{N/2^k-1} d_{k,m}\psi_{k,m}(n). \tag{2.6}$$

Here N – total number of signal samples, $\varphi(t)$ – scaling function, $\psi(t)$ –wavelet function.

Calculation of the coefficients $a_{j,m}$ and $d_{k,m}$ implies the problem of computing of a large number of integrals with the required accuracy. The problem solution is the algorithm of a fast wavelet transform (FWT) that was proposed by S.Mallat [19]. Each orthogonal wavelet corresponds to the four filters:

- h_n represents the decomposition low-pass filter (LF);
- g_n represents the decomposition high-pass filter (HF);
- $\tilde{h}_n$ represents the reconstruction low-pass filter;
- $\tilde{g}_n$ represents the reconstruction high-pass filter.

The wavelets are not used for the FWT coefficients calculation, but the filters that associated with these wavelets are used instead.

In general, the iterative formulas of FWT look like this:

$$a_{j+1,m} = \sum_n h_n a_{j,2m+n}\ ,\ d_{j+1,m} = \sum_n g_n a_{j,2m+n}. \tag{2.7}$$

Here $a_{0,m} = \int f(t)\varphi(t-m)dt$. For signal that is given as an array of samples, initial decomposition coefficients are usually chosen equal to values of these samples: $a_{0,m} = f(t_n)$.

In a spectral domain, wavelet decomposition leads to octave-band dividing of frequency range signal [27]. For an arbitrary dividing, wavelet packet decomposition is used, when not only approximated coefficients $a_{j,m}$ can be divided, but the detailing coefficients $d_{j,m}$ too. Wavelet packet transform (WPT) contributes to a better frequency localization of the signals and, also, to more effective presentation of the watermark bits embedding area in comparison with DWT.

FWT and WPT signal decomposition/reconstruction can be realized with using of orthogonal (Daubechies wavelets, symlets, koiflets) and biorthogonal (B-splines) wavelets with compact carrier. Exact rules of choosing a basic wavelet do not exist. Daubechies wavelets are the most popular in various applications, particularly in steganography and watermarking techniques [5, 28]. On choosing of wavelet order it is necessary to consider that with increasing of this order, cut-off slope of filters frequency characteristics and the quality of signal decomposition/reconstruction also increase. But at the same time, the computational complexity of implementation increases, too.

2.2.2. Impact of lossy compression operations to audio signals

Analog signals are digitized with using of pulse-code modulation (PCM). The size of the audio signal that is encoded by PCM samples with sufficiently large sampling rate and bit depth is usually unacceptably large for its storage and transmission "as is". Therefore, the different standards of audio compression have been developed. Nowadays, the most common standards are MPEG-1 Layer 3, MPEG-2/4, Ogg Vorbis, and WMA.

Watermark should be robust to possible processing operations of audio signal, firstly to lossy compression. Therefore, a nature of distortions that are made by this operation is advisable to examine.

The human auditory system can be modeled as a frequency analyzer, consisting of a set of band-pass filters, which is implemented in the lossy audio data compression algorithms. The audio signal size is reduced by removing psychoacoustic- and statistical redundancy from the initial signal. Psychoacoustic models (PAM) are used during compression to determine masking thresholds and to adaptive distribution of bits among the signal frequency subbands. They are based on the following characteristics of human auditory system.

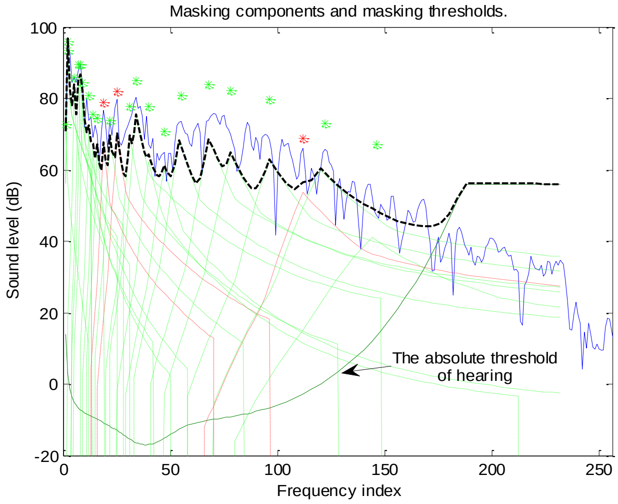


Fig. 2.2. Example of calculation of the masking thresholds for one of psychoacoustics models of standard MPEG-1 Layer 3 (PAM1)

The absolute threshold of hearing. At the same sound pressure level, perceived volume of different frequencies of pure tones is different. Also, the different is the minimal sound pressure when an auditory sense still exists. The threshold of hearing also depends on the experimental conditions. The minimum level of sound pressure when a sound wave with harmonic form could be heard in the absence of other sounds, is called as the absolute hearing threshold or as hearing threshold in quiet (Fig. 2.2). Evidently the spectral signal components that lie below the absolute threshold of hearing can be skipped while encoding and transmitting.

Frequency (simultaneous) masking. The hearing threshold of one signal is changing in the presence of second one. The hearing threshold of one of the sound components in the presence of other is called as a relative hearing threshold.

A weaker but audible sound (the maskee) can be made inaudible in the presence of a louder sound (the masker), that process is called masking. The masking effect depends on the spectral and temporal characteristics of both the maskee and the masker. Masking in the frequency domain is appeared in different ways, depending on the particularities of the audio signals spectrum. In the development of lossy compression algorithms, the masking difference between inside and outside of the critical frequency bands of hearing is taken into account. The masking threshold depends on the frequency, a level of suppress signal, tone or noise characteristics of the maskee and the masker.

Spectral components that are below the relative threshold of hearing are inaudible for the human auditory system (see Fig. 2.2), so they also could be skipped while encoding and transmitting.

Temporal masking. In addition to frequency masking, two time domain phenomena also play an important role in human auditory perception, pre-masking and post-masking. These phenomena describe dynamic properties of hearing, and reflect the changing in time of relative hearing threshold when the maskee and the masker do not sound simultaneously. Pre-masking effects make weaker signals inaudible before a stronger masker is switched on, and post-masking effects make weaker signals inaudible after a stronger masker is switched off. Pre-masking occurs from 5 to 20 ms before the masker appears, while post-masking occurs from 50 to 200 ms after the masker is disappears. Pre-masking duration considerably depends on the individual human features. Therefore, pre-masking effects are not usually taken into account in the PAM.

We should remember that these features and the PAM (generated on their basis) can be used not only for developing for lossy compression methods, but also for watermark embedding.

The critical hearing bands approximation using wavelet packet tree. In the psychoacoustics, 25 critical bands of hearing (Fig. 2.3) are differentiated. The integration of input audio information happens inside them and the frequency masking effect manifests itself the strongest way [1].

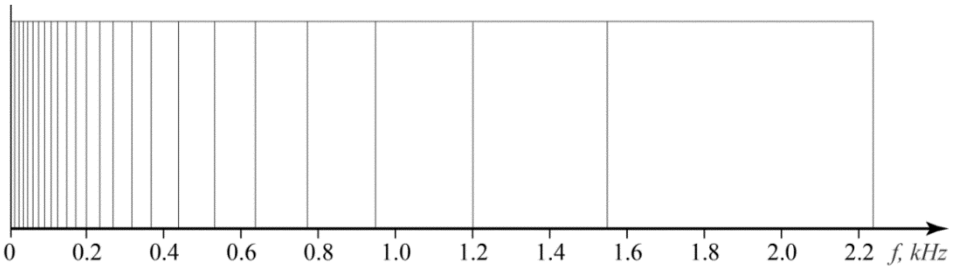


Fig. 2.3. Frequency domain decomposition according to critical bands of hearing

As each of audio compression codec exploits its own signal representation method in a frequency-domain and its PAM, so while resolving the problem of finding invariant to all accepted lossy compression standards, it is appropriate to begin from the most common estimates, particularly the estimates that obtained by the analysis of original and compressed signals spectrograms.

Let us examine the audio compression by different encoders with low bit rate and big losses (Fig. 2.4b-2.4d). As you can see, high frequencies are cut off and middle frequencies are partially suppressed by the encoding due to the standard MPEG-1 Layer 3 audio. The above observation is also correct in the case of WMA encoder, which was used with the settings that bring maximum losses. For Ogg Vorbis encoding relatively more distortion is also observed in high-frequencies domain of the signal.

Another variant of visual analysis was performed for a set of audio signals spectrograms which are received from the original signal caused by sequential compression by fixed encoder with different bit rates, from maximum to minimum possibility for a compression standard and a signal. Overall, this analysis showed that the significant distortions regions will be distributed in the direction from high to low frequencies and in some cases distortion of near-zero frequency domains will be added.

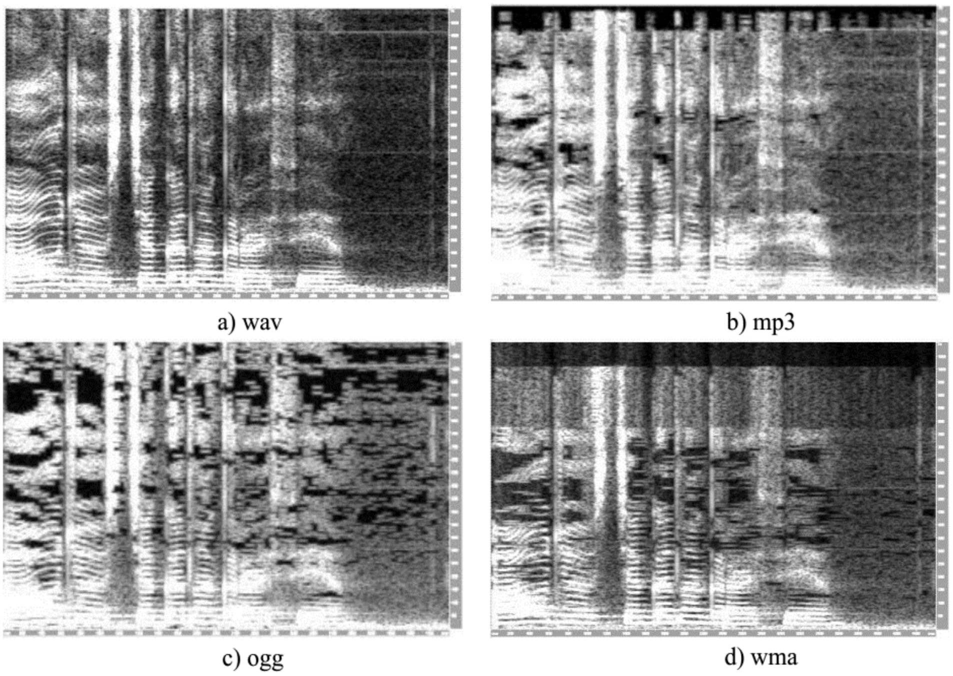


Fig. 2.4. The spectrograms of: a) the initial signal in the format .wav; b) the signal, encoded to the format .mp3 with bit rate 64 kbps; c) the signal, encoded to the format .ogg with bit rate 32 kbps; d) the signal, encoded to the format .wma with bit rate 20 kbps

Let us note that spectral analysis is based on the time $n = 0..N-1$ and frequency $r = 0..N-1$ indexes. It allows using the algorithm to find spectral coefficients without changing the calculation procedure for the different sampling rates. If you want to find matching between indexes and real frequency axis, you should take into account that the frequency spectrum was digitized with sampling increment $\Delta\omega = \frac{2\pi}{N \cdot \Delta t}$ rad/sec, or $\Delta f = \frac{1}{N \cdot \Delta t}$ Hz, where $\Delta t = \frac{1}{F_d}$ – sampling rate in Hz. That is, if you know the sampling rate, then *r*-th spectral sample corresponds to the frequency $\omega = r \cdot \Delta\omega$ rad/sec, or $f = r \cdot \Delta f$ Hz.

2.2.3. Audio watermarking method robust against lossy compression

Here is an example of the audio watermarking method, which implements the scheme with the watermark decoder, and in which watermark can be restored after the lossy compression attacks. The method is based on the fact that the watermark is placed in a signal form which is more stable and predictable parameter of the attacked cover than the individual samples values.

An initial audio signal is divided into equal-length blocks. The length is a question of compromise between high resolution in frequency domain during subsequent spectral analysis and computational complexity of the algorithm. One bit of additional information is embedded in each block.

On the first step of embedding procedure, coefficients of frequency subbands of the signal block, which will serve as an immediate carrier of digital watermark bit, are determined using FWT or PWT. On the next step subband-carrier spreads by a secret key that is pseudo-random uniformly distributed sequence of 1 and -1. The method of slow spread spectrum [8] is used on this step; its influence on the amplitude spectrum is shown in Fig. 2.5a. A simple example of direct and inverse spread is shown in Fig. 2.5b.

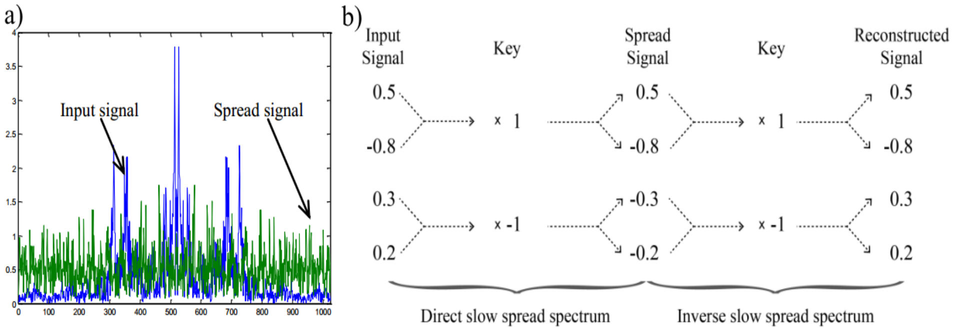


Fig. 2.5. Slow spread spectrum of signal with using of key sequence

The key determines the location of current watermark bit into the current frequency subband, which will be the carrier of this bit. The watermark extraction using another key than the one used for embedding, will lead to the extraction of bits out of the wrong locations, i.e. extract random values. The key is independent of the signal, and its length is equal to the length of subband-carrier.

Then using the Fourier transform, a transition to the frequency representation of the spread subband by key sequence makes; and amplitude spectrum coefficients are considered hereinafter as coefficients for embedding.

The location of current bit into current signal block is determined by the location of maximum coefficient of the amplitude spectrum. This location (with sufficient frequency resolution) is invariant to the lossy compression. Then, the method provides removing (zeroing) of the three coefficients on the left from the maximum if zero-bit value is embedded, or three coefficients on the right from the maximum for one-bit value. Thus, the digital watermark is encoded as relative difference between the samples.

The extraction procedure is identical to embedding procedure until the determination of maximum amplitude location in the Fourier domain of the subband-carrier. Further, if the sum of three coefficients on the left of the maximum is bigger than the sum of three coefficients on the right from it, then a one-bit value is extracted from the current block, and if vice versa - zero-bit value.

This method was implemented using Matlab package. Two-level FWT based on Daubechies wavelet of order 10 was used to determine the subband-carrier. The set of test signals included one-minute music fragments, which were digitized with a sampling rate of 44 kHz and a bit depth of 16 bits. Audiocodec that was used during robustness testing – LameXP.

Robustness was estimated as the relation between number of correctly extracted bits and number of embedded (*Ratio of Correct Bits Recovered*):

$$ROCBR = \frac{100}{Z}\sum_{i=0}^{Z-1}\begin{cases}1, & w_i = w_i', \\ 0, & w_i \neq w_i'.\end{cases} \tag{2.7a}$$

Here w_i is *i*-th bit of embedded watermark, w_i' is *i*-th bit of extracted watermark, Z – total number of bits.

Results of watermark robustness testing against compression according to standards MPEG-1 Layer 3, Ogg Vorbis are given in Fig. 2.6 and Fig. 2.7.

It is possible to improve ROCBR by using redundant digital watermarks. The first way how to get them is successive embedding of input watermark bits by a certain number of times (embedding periods), the second - antinoise coding of input bits. In this case, improving of a robustness will occur at the expense of worsening of a capacity. In the simplest case *z* errors can be compensated by the information repeating 2*z* +1 times. But this variant is the worst for the capacity.

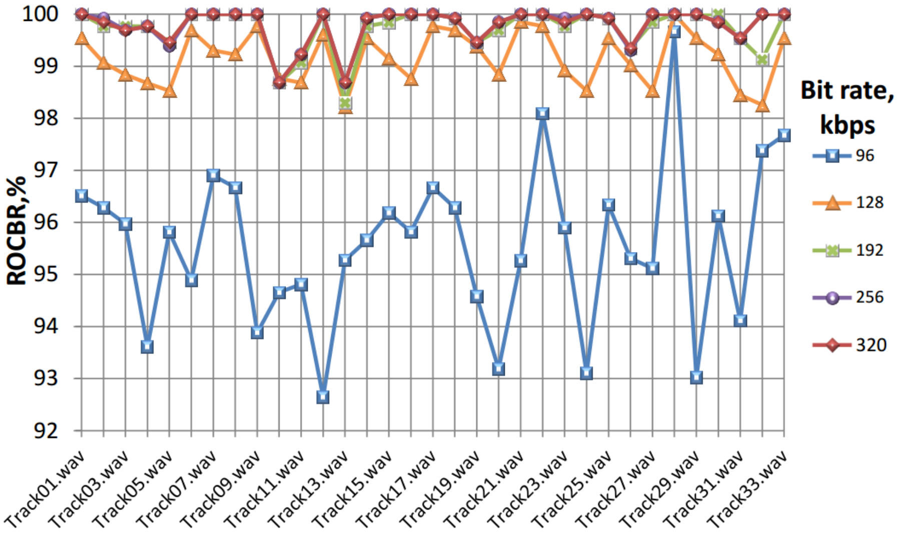


Fig. 2.6. Results of watermark robustness testing against the MPEG-1 Layer 3 compression attack

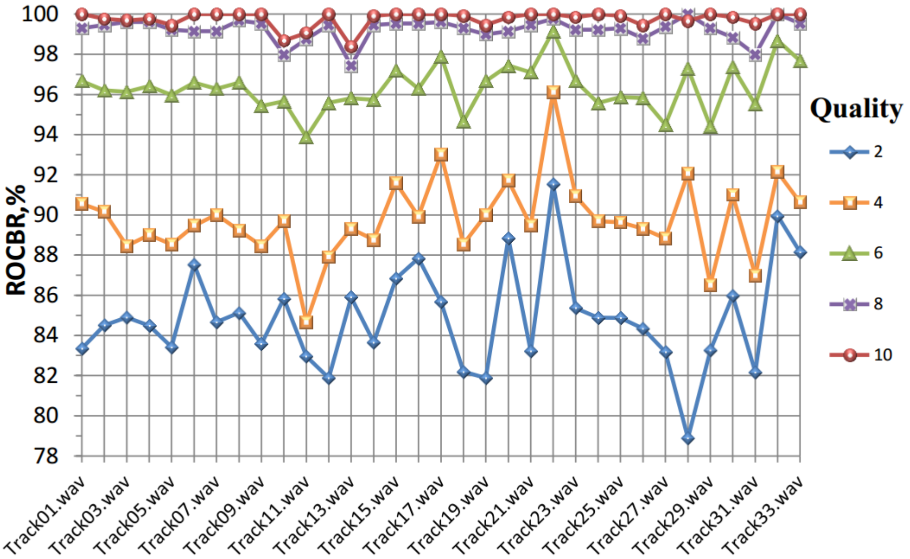


Fig. 2.7. Results of watermark robustness testing against the Ogg Vorbis compression attack

Let's examine the problem of antinoise coding of digital watermark bits before their embedding. To choose the error correction code and its parameters, let analyze the location of errors that appear when watermark is extracted out of a test signal after lossy compression with the worst quality (Fig. 2.8). In Fig. 2.8, the ordinate value is equal to 1, if embedded zero-bit of digital watermark has been extracted as one-bit; -1, if embedded one-bit has been extracted as a zero; 0 for correct extraction.

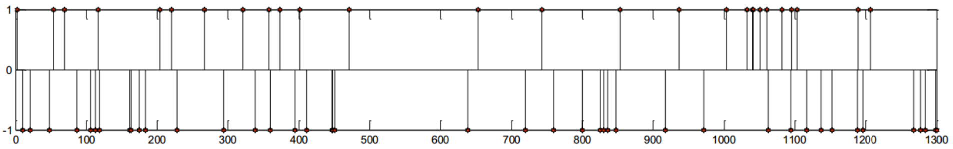


Fig. 2.8. Location of errors when digital watermark is extracted out of the signal after attack by ogg-compression with bit rate ~ 32 kbps

As you can see, there is a necessity to use an error correction code that can correct three or more errors in a code word. There is, e.g., a widely known class of cyclic error-correcting codes that can correct not only single but multiple errors - these are Bose-Chaudhuri-Hocquenghem codes (BCH). These binary codes are beneficial, because they enable choose the lengths of code word and initial data block according to a given number of errors that have to be corrected in a code word, and BCH also have efficient algorithms for encoding-decoding [23].

Results of robustness tests, where digital watermark is encoded by the BCH code before embedding, confirm the usefulness of this code (table 2.1), but task to find new and more efficient error correction codes is also still actual.

Table 2.1. Results of the watermark robustness testing for the version with BCH coding

Test signals (.wav)	Attack by mp3-compression with bit rate 128 kbps						Attack by ogg-compression with quality 6					
	The length of the block signal is equal to 1024			The length of the block signal is equal to 2048			The length of the block signal is equal to 1024			The length of the block signal is equal to 2048		
	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)
	ROCBR, %											
Track01.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track02.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track03.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track04.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track05.wav	99,6	100	100	100	100	100	99,6	100	100	97,15	100	100
Track06.wav	100	100	100	100	100	100	100	100	100	100	100	100

Test signals (.wav)	Attack by mp3-compression with bit rate 128 kbps						Attack by ogg-compression with quality 6					
	The length of the block signal is equal to 1024			The length of the block signal is equal to 2048			The length of the block signal is equal to 1024			The length of the block signal is equal to 2048		
	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)
	ROCBR, %											
Track07.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track08.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track09.wav	100	100	100	100	100	100	100	100	100	99,21	100	100
Track10.wav	100	100	100	100	100	100	99,8	100	100	100	100	100
Track11.wav	99,4	100	100	100	100	100	99,6	100	100	100	100	100
Track12.wav	100	100	100	100	100	100	99,8	100	100	99,21	100	100
Track13.wav	99	99,76	99,65	99,19	100	100	99,4	99,02	98,95	98,78	99,5	99,29
Track14.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track15.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track16.wav	100	100	100	100	100	100	99,4	100	100	100	100	100
Track17.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track18.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track19.wav	100	100	100	100	100	100	100	100	100	99,19	100	100
Track20.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track21.wav	100	100	100	100	100	100	100	100	100	100	100	100

Test signals (.wav)	Attack by mp3-compression with bit rate 128 kbps						Attack by ogg-compression with quality 6					
	The length of the block signal is equal to 1024			The length of the block signal is equal to 2048			The length of the block signal is equal to 1024			The length of the block signal is equal to 2048		
	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)	BCH (31,6)	BCH (63,10)	BCH (63,7)
	ROCBR, %											
Track22.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track23.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track24.wav	100	100	100	100	100	100	99,8	100	100	100	100	100
Track25.wav	99,4	100	100	100	100	100	100	100	100	100	100	100
Track26.wav	99,64	99,85	100	99,28	100	100	99,03	99,71	99,16	99,52	99,12	99,58
Track27.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track28.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track29.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track30.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track31.wav	100	100	100	100	100	100	100	97,62	100	100	100	100
Track32.wav	100	100	100	100	100	100	100	100	100	100	100	100
Track33.wav	100	100	100	100	100	100	100	100	100	100	100	100

2.2.4. Self-synchronised audio watermarking method

The following method implements the watermarking scheme with the detector and this method is built such way that watermark could be correctly extracted out of signal after its displacement on the time axis.

Modifications of marked cover object, which is in the free access, may change its elements values (audio signals samples, image pixels, etc.) and also their locations. The result of a row of typical processing operations is desynchronization of embedded digital watermark in its cover object. For example, during convertion of a file with a marked audio signal from WAV (PCM) format to ACC format using LameXP application (Fig. 2.9b), or to WMA format using WinFF (Fig. 2.9c), the shift (delay) of the signal in time happens, that can lead to extraction of watermark out of incorrect locations (not those where it was embedded). The same problem occurs due to cropping of marked signal. Moreover, the cropping makes troubles not only due to a signal shift on the time axis, but due to a necessity to detect digital watermark in the part of signal.

Except unintentional attacks that occurred during the signal or image processing, a presence of the active offender is also expected when the watermarking system creating. He can purposefully do attack with desynchronization that preserves the signal functionality and, in fact, this desynchronization does not remove the watermark, but breaks correct response of the detector.

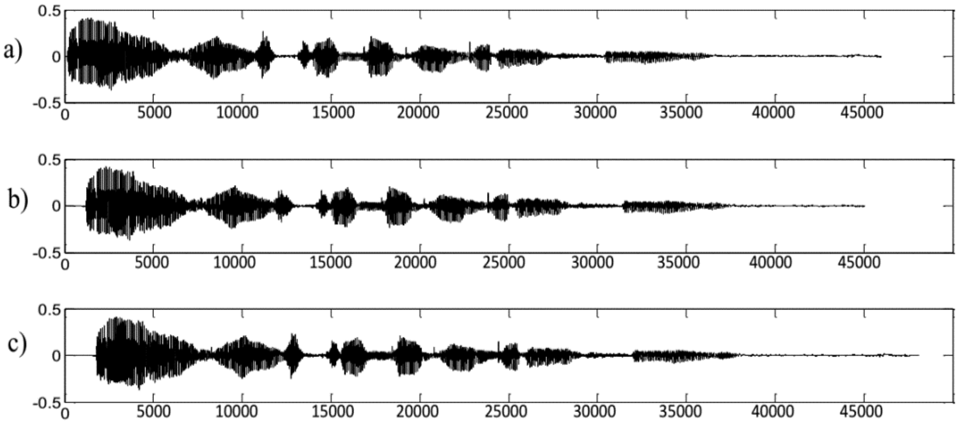


Fig.2.9. Shift audio signal due to its file format changing

Thus, the development of effective digital watermarking system among other requirements has to include the solution of desynchronization problem. In articles [29, 30] was suggested such watermarking method for musical compositions that embeds watermark in signal domain which does not change after the shifting and cropping. There was an interest to explore and adapt this method to solve the problem of digital speech signal source authentication. Speech signals are characterized by lesser frequency range and lower signal-to-noise ratio in comparison to the digital music. In addition, the presence of silence is typical for speech records, and if the modifications formed by embedding of digital watermark touched these areas, they will be heard by the ear as it occurs for digital watermark, that is embedded to the frequency domain of wavelet coefficients of levels 5 or 6 of speech signal decomposition, in applying the method from [29].

So, let us consider the speech watermarking system with automatic synchronization by feature points of the signal and watermark embedding into the spectral domain. Each device that is registered in the watermarking system gets the key *K*, which is used as the starting number for the generator of pseudo-random watermark. Digital watermark is embedded into sensitive to distortion areas, which locations are determined by feature points. Method of correlation analysis is used for watermark detection. User can determine the presence or absence of watermark only if he knows the secret key (or the watermark).

Synchronization of audio signals based on feature points. Qualitative method of extracting feature points should generate approximately the same set of points for the original and distorted by attacks signals. The approach that presented below is characterized by lower computational complexity than synchronization based on a full enumeration of variants, making possible a "blind" detection. Another significant advantage is that, unlike methods based on synchronization templates [9], information about synchronization is not added to the signal, but could be extracted out of it by analyzing the content. While obviously added synchronization template can attract the offender's attention and, as a result, it can be destroyed, synchronization based on feature points does not add any extra distortions into the signal.

One of the variants of feature points of the signal is the points which are extracted as locations, where the signal energy is fast climbing to a peak value. Locations of these points are difficult to shift significantly without bringing audible distortions. Digital watermark should be embedded in significant areas of host signal, such as the regions after the feature points, as they contain high energy.

Thus, the identification method of feature points in the audio signal consists of following steps:

The signal is divided into blocks of a given length. Three-level wavelet decomposition is applied to each block; as a result four subbands are formed: approximation $A_3(n)$ and three details $S_3(n)$, $S_2(n)$, and $S_1(n)$. As human speech is in the frequency range 300 Hz - 3.5 kHz, it often happens that subband $A_3(n)$ contains a small fraction of the energy. For this reason $A_3(n)$ is not included in the embedding area. Subband $S_1(n)$ can also contain little part of the signal energy. In addition, high-frequency components can be significantly distorted by the lossy compression. So, watermark embedding into the $S_1(n)$ is inappropriate. Thereby, further analysis and transformations are performed for subbands $S_3(n)$ and $S_2(n)$ only. This step improves the stability of the resulting set of feature points and watermark robustness to the possible attacks.

For each coefficient of the current embedding subband $S_j(n)$, $j = 2,3$, the total energy of *d* coefficients directly before it and *d* directly after is calculated separately:

$$\begin{cases} E_{before}(n) = \sum_{i=-d}^{-1} S_j^2(n+i), \\ E_{after}(n) = \sum_{i=0}^{d-1} S_j^2(n+i), \;\; n = 1, \ldots N_j. \end{cases} \tag{2.8}$$

The ratio of these two energies values is calculated as:

$$ratio(n) = \frac{E_{after}(n)}{E_{before}(n)}, \;\; n = 1, \ldots N_j. \tag{2.9}$$

Energy fast-climbing points are determined by the conditions:

$$ratio(n) > T_1, \;\; n = 1, \ldots N_j. \tag{2.10a}$$

$$E_{after}(n) > T_2, \;\; n = 1, \ldots N_j. \tag{2.10b}$$

Condition (2.10b) prevents the selection of points that are labeled as energy fast-climbing points because the value $E_{before}(n)$ is near to zero.

The coefficients that satisfy the conditions (2.10a) and (2.10b) are often located in groups. To improve the stability of the resulting set of points, only groups that are consist of more than T_3 consecutive points are kept.

In each retained group the point with maximum value $ratio(n)$ is selected as a feature point. The synchronization in the watermark detector will be performed according to the locations of these points.

As watermark subsequently will embed in 2^p of consecutive samples starting with each selected feature point, in the final set of feature points only points are remained where the distance between them is greater than 2^p.

Digital watermark will be embedded into the signal so many times as feature points were identified within this cover after execution of steps 1-7. Thresholds T1, T2, and T3 are determined in such way as to provide a finding at least 1-2 feature points in one-second duration of an audio signal.

Fig. 2.10 shows an example of the result of finding the feature points locations in the subband S3(n) of one of the original test signals, digitized with 8 kHz sampling rate (locations of feature points are labeled by vertical dotted line with marker). The output of the described algorithm of feature points finding (with parameters r=1024; T1=20; T2=mean(Eafter)/5; T3=5; p=1024) is 14 points in the subband S3(n).

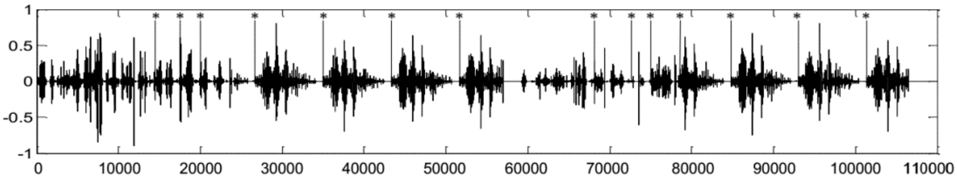


Fig. 2.10. The feature points in subband $S_3(n)$ of one of the test audio signals

Since the feature points depend on the content of the signal, and do not have fixed locations on the time line, their locations are invariant to the shift signal. The representation of stability of the set of feature points to the typical attacks kinds is given in table 2.2 that is based on results from one of the test experiments. In the first column of the table are placed the indexes of feature points defined in the wavelet coefficients of the original signal. Other columns demonstrate the locations of feature points that were identified in the wavelet coefficients of the signal which was attacked by operation that is pointed in the table title, and their shift relatively to the original points. In this case the feature point was considered as stable one if the shift of its location before and after the attack does not exceed 50 samples. Non-stable feature points are shown in bold in the table.

Table 2.2. Results of testing the stability of feature points

Original signal	Attacks on the original signal														
	White noise 40 dB		Low-pass filter 2 kHz		MP3 compr. 32 kbps		OGG comp. 32 kbps		WMA compr. 32 kbps		MPEG-4 Audio compr. 32 kbps		AMR compression		
Locations of feature points in subband $S_3(n)$															
156872	156872	0	156872	0	156872	0	156872	0	156936	-64	156360	512	156886		-14
159887	159887	0	159887	0	159887	0	159887	0	159951	-64	159369	518	159909		-22
162324	162324	0	162324	0	162325	-1	162324	0	162388	-64	161812	512	162333	**164437**	-9 / -
169071	169071	0	169071	0	169071	0	169071	0	169135	-64	168558	513	169083		-12
177397	177397	0	177397	0	177397	0	177397	0	177461	-64	176885	512	177404		-7

Original signal	Attacks on the original signal													
	White noise 40 dB		Low-pass filter 2 kHz		MP3 compr. 32 kbps		OGG comp. 32 kbps		WMA compr. 32 kbps		MPEG-4 Audio compr. 32 kbps		AMR compression	
185718	185718	0	185718	0	185718	0	185718	0	185782	-64	185206	512	185727	-9
194038	194038	0	194038	0	194038	0	194038	0	194102	-64	193527 **203017**	511 / -	194047	-9
210455	210455	0	210455	0	210455	0	210442	13	210519	-64	209933	522	210464	-9
215037	215037	0	215037	0	215037	0	215037	0	215101	-64	214525	512	215046	-9
217421	217421	0	217421	0	217421	0	217421	0	217485	-64	216909	512	217431	-10
220926	220926	0	220926	0	220926	0	220926	0	220990	-64	220414	512	220935	-9
227198	227198	0	227198	0	227198	0	227198	0	227262	-64	226686	512	227206	-8
235390	235390	0	235390	0	235390	0	235390	0	235454	-64	234877	513	235402	-12
243716	243716	0	243716	0	243716	0	243716	0	243780	-64	243203	513	243723	-7
Locations of feature points in subband $S_2(n)$														
301658	301658	0	**287962** 301658	- / 0	301690	-32	301658	0	301789	-131	300621	1037	301676	-18
305337	305335	2	305335	2	305337	0	305335	2	305465	-128	304313	1024	305356	-19
309628	309628	0	309628	0	309628	0	309629	-1	309756	-128	308602	1026	309649	-21

Original signal	Attacks on the original signal														
	White noise 40 dB		Low-pass filter 2 kHz			MP3 compr. 32 kbps		OGG comp. 32 kbps		WMA compr. 32 kbps		MPEG-4 Audio compr. 32 kbps		AMR compression	
319700	319700	0	**319433**		267	319698	2	**319433**	267	319828	-128	318700	1000	**319463**	237
327914	327914	0	327914		0	327914	0	327914	0	328042	-128	326888	1026	327932	-18
333329	333329	0	333329	**371438**	0 /	333327	2	333329	0	333457	-128	332305	1024	**333018**	311
403158	403155	3	**388078**	403159	- / -1	403158	0	403158	0	403286	-128	402120	1038	403176	-18
406625	406625	0	406625		0	406625	0	406625	0	406753	-128	405601	1024	406643	-18
415188	415188	0	415188		0	415188	0	415188	0	415316	-128	414166	1022	**415126**	62
420037	420037	0	419995		42	420029	8	420041	-4	420160	-123	418994	1043	420083	-46
431737	431737	0	**431687**		50	431737	0	431707	30	**431713**	24	430695	1042	431751	-14
434816	434817	-1	434783		33	434816	0	434797	19	434930	-114	433774	1042	434836	-20
441917	441917	0	441915	**454393**	2 / -	441923	-6	441911	6	442056	-139	440893	1024	441942	-25

As we can see, the leader in the number of shifted feature points is the attack of AMR-compression. In this case the quality of the compressed signal is also worse than the quality of the signal that is encoded by the other four standards. Furthermore, it should be explained, that during WMA-compression, the signal delay equal to the 512 samples happened, and consequently the shift of indexes of subband S3 is equal to 64 samples, and subband S2 is equal to 128. A similar indexes shift relative to the content occurs during encoding of MPEG-4 Audio, too.

Watermark embedding in the DFT domain. Although the feature points are defined such way that they has to be the most stable against attacks on the signal, obtaining the enough number of points that are stable against any active attack is impossible. Often it will occur some, even small, point shift before and after the attack (Table 2.2). Shifting of feature points can also occur due to the watermark embedding process. If the embedding will be made directly in the time domain, then any shift will be critical for synchronization. But the problem is simplified if a digital watermark will be embedded in the spectral domain, especially in the Fourier amplitude spectrum.

Let embedding region is defined as $s'(i), i = 1,\ldots 2^p$, where $s'(1)$ is a feature point, and digital watermark will be embedded in the amplitude Fourier spectrum $|S'(r)|, r = 1,\ldots 2^p$. Suppose that after the attack on the signal, feature point was extracted with the shift and the embedding region is identified as $g(i), i = 1,\ldots 2^p$ (Fig. 2.11). One of the DFT properties is the invariance of the amplitude spectrum to the cyclic signal shift, so if the signal $h(i), i = 1,\ldots 2^p$ is formed as it is shown in Fig. 2.11, then

$$|H(r)| = |S'(r)|, r = 1,\ldots 2^p. \tag{2.11}$$

Denote the difference between $h(i)$ and $g(i)$ as $p(i)$

$$p(i) = h(i) - g(i), i = 1,\ldots 2^p. \tag{2.12}$$

Using the linearity property of DFT, we can write

$$|G(r)| = |H(r)| - |P(r)|, r = 1,\ldots 2^p. \tag{2.13}$$

From this equality and taking into account (2.11), we obtain

$$|G(r)| = |S'(r)| - |P(r)|, r = 1,\ldots 2^p. \tag{2.14}$$

That is, for the Fourier amplitude spectrum the desynchronization phenomenon in the time domain replaced by additive noise. If the shift is small relative to the embedding region length 2^p, the energy $|P(r)|$ will be small and $|G(r)| \approx |S'(r)|$.

Digital watermark $W(r)$ is a pseudorandom sequence of 2^{p-1} elements that are distributed normally with zero mean and unit variance. Device key *K* is used as the initial value of the watermark generator.

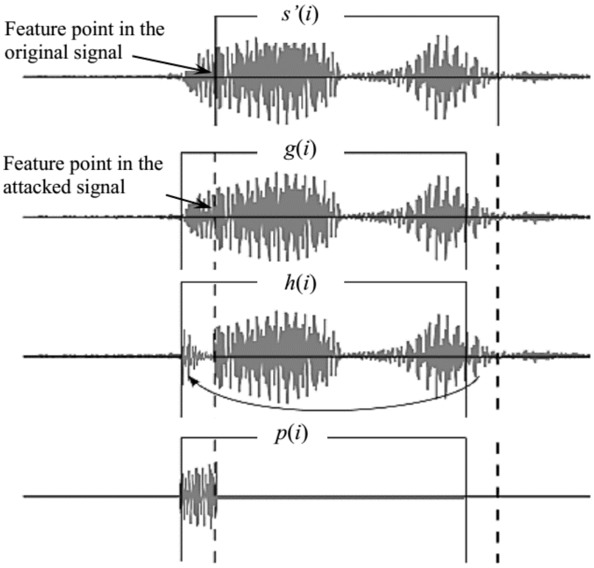

s'(i)
Feature point in the original signal
g(i)
Feature point in the attacked signal
h(i)
p(i)

Fig. 2.11. Influence of the shift of the embedding region on its amplitude spectrum

Let us suppose that during content analysis of audio signal M_j feature points were extracted out of subband $S_j(n)$. Blocks of 2^p coefficients that are denoted as $s'_{j,m}(i)$, $i=1,\dots 2^p$, $m=1,\dots M_j$, where $s'_{j,m}(1)$ are feature points, will be watermark embedding regions. Fourier amplitude spectrum $\left|S'_{j,m}(r)\right|, r=1,\dots 2^p$ is computed for each region.

Next, the binary perceptual mask $I_{j,m}(r)$ is formed and superimposed on the watermark $W(r)$:

$$W_{j,m}(r)=I_{j,m}(r)W(r), r=1,\dots 2^{p-1}. \tag{2.15}$$

Taking into account that the spectrum of real signal is symmetric, digital watermark that obtained according to (2.15), is symmetrically complemented such way that does not break the symmetry:

$$\begin{cases} W_{j,m}(r)=0, r=1,2^{p-1}+1, \\ W_{j,m}(r)=W_{j,m}(2^p-r+2), r=2^{p-1}+2,\dots 2^p. \end{cases} \tag{2.16}$$

Watermark that is obtained using the expressions (2.15) and (2.16) is embedded in the spectrum as follows:

$$S''_{j,m}(r)=S'_{j,m}(r)\left[1+\alpha W_{j,m}(r)\right], r=1,\dots 2^p. \tag{2.17}$$

Here α is constant that regulates the power of digital watermarking. After marking the block by (2.17), modified coefficients of Fourier amplitude spectrum will be carriers of watermark, but phase spectrum will remain unchanged. Next, IDFT is performed for each marked region and it replaces the corresponding original region in the signal.

Watermark blind detection. The input values of the detector are the test audio signal and also the digital watermark which presence or absence will be checked. The procedure of watermark detecting repeats the embedding procedure until the calculation $\tilde{W}_{j,m}(r)$, which is formed from the original digital watermark by the expressions (2.15) and (2.16). The decision about the watermark presence or absence into signal is taken according to the average value of the correlation coefficient, defined by the formula

$$\Re = \frac{1}{\tilde{M}} \sum_{m=1}^{\tilde{M}} \frac{\sum_{r=1}^{2^p} \left| \tilde{S}'_{j,m}(r) \right| \cdot \tilde{W}_{j,m}(r)}{\sqrt{\sum_{r=1}^{2^p} \left| \tilde{S}'_{j,m}(r) \right|^2} \sqrt{\sum_{r=1}^{2^p} \tilde{W}_{j,m}(r)^2}}, \qquad (2.18)$$

where $\tilde{M}$ – the total number of feature points, that are identified within the test audio signal. If average correlation coefficient is greater than the detection threshold, the decision about the watermark presence in this signal is taken.

During the research was conducted experiment with calculation of correlation coefficients for 1800 original and watermarked audio signals. The experimental results are shown in Fig. 2.12, where histogram H_0 shows the distribution of correlation coefficients for the empty cover signals, and H_1 – for marked covers. According to this statistics, the detection threshold algorithm for embedding parameters r=1024; T_1=20; T_2=mean(E_{after})/5; T_3=5; p=1024; $\alpha = 0.35$ can be set in range between 0.14 and 0.17.

To guarantee the absence of false alarms, it is desirable to set the detection threshold to the maximum high.

Analysis of watermark robustness and system reliability. During the analysis of watermark robustness the same attacks as during the stability testing of feature points set were used. Testing of robustness to the cropping when 10% from the beginning of the signal and 10% from the end were cropped was also performed.

In Fig. 2.13a, the correlation coefficient, obtained as a result of watermark detection in undistorted marked signal with the correct key K=350, is shown in comparison with the correlation coefficients, obtained during watermark detection with 999 other possible keys. Fig. 2.13b-2.13i show the results of similar tests for attacked marked signal.

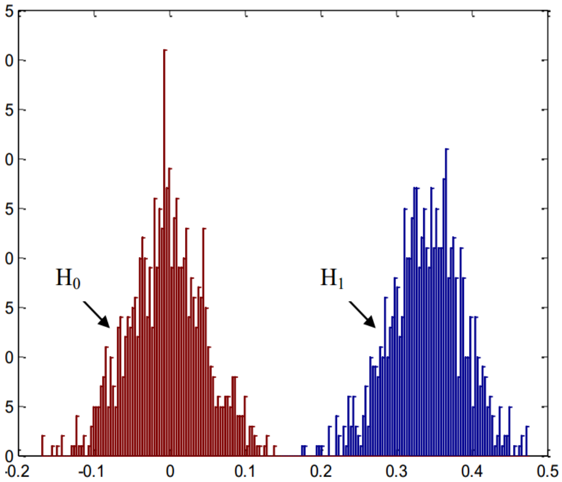
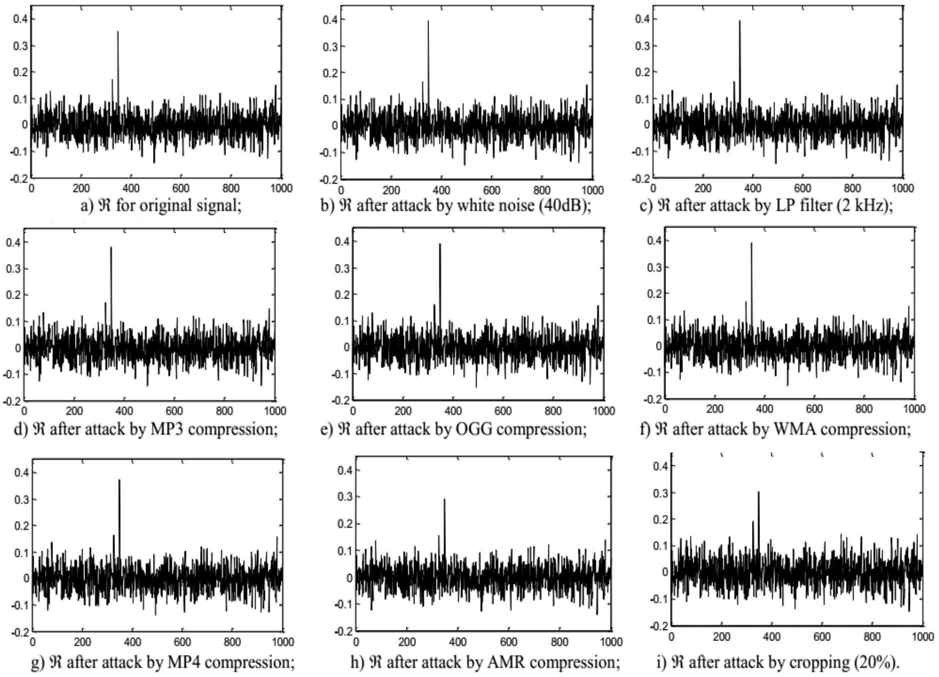


Fig. 2.12. Distribution of correlation coefficients which are calculated for original and marked signals

As we can see, value of the correlation coefficient obtained with the correct key may decrease after the attacks to the marked signal. However, in these tests it is always the highest and significantly higher than the correlation coefficients obtained with the wrong keys.



a) $\Re$ for original signal;

b) $\Re$ after attack by white noise (40dB);

c) $\Re$ after attack by LP filter (2 kHz);

d) $\Re$ after attack by MP3 compression;

e) $\Re$ after attack by OGG compression;

f) $\Re$ after attack by WMA compression;

g) $\Re$ after attack by MP4 compression;

h) $\Re$ after attack by AMR compression;

i) $\Re$ after attack by cropping (20%).

Fig. 2.13. Results of watermark detecting for the original and attacked signals

Reliability of watermarking system can be estimated by analyzing the results of the same test. For this, values of correlation coefficient obtained during watermark detection with the correct key in undistorted and distorted due to attacks marked signals, are written in the first row of Table 2.3. In the second row of this table, the maximum correlation values obtained during watermark detection with the wrong key are written.

According to the first row of the table and the previous experiment about the separability of hypotheses (see Fig. 2.12), the test signal will be recognized as marked after all these attacks. If the detection threshold is fixed as 0.17, the false alarm occurs in 2 cases of 9000 in this experiment: during watermark detection with the key K=327 on signals that were attacked using low-frequency filtering and cropping. This is 0.022% of the total number of tests in the experiment (correlation coefficients obtained with the other 998 keys do not surpass value of 0.17).

Note also that for the values of correlation coefficients that are situated near to the detection threshold, it makes sense to conduct a full search of correlation coefficients for all the keys. This checkup requires more resources, but it will allow interpret the detection result correctly.

Table 2.3. The correlation coefficients obtained with the correct and wrong keys

	$\Re$ for marked signal	$\Re$ for attacked marked signal							
		White noise 40 dB	Low-pass filter 2 kHz	MP3 compression 32 kbps	OGG compression 32 kbps	WMA compression 32 kbps	MPEG-4 Audio compression 32 kbps	AMR compression	White noise 40 dB
1.	0.3932	0.3941	0.3523	0.3794	0.3907	0.3921	0.3734	0.2911	0.303
2.	0.1649	0.1647	0.1703	0.1698	0.1595	0.1658	0.1643	0.1554	0.1915

2.3. Digital watermarks to protect information stored on paper carriers

2.3.1. Classification of natural distortion during print-scan process

Watermarking technologies can be successfully used for protection of not only digital objects, but also those stored on paper or plastic carriers. This will allow increasing the potential user number of watermarking techniques to a great extent. Commercial availability of perceptual qualitative transformation of analogue information into digital form and inversely leads to necessity for search of new ways of counteraction to forgery of important documents: passports, driver's licenses, ID cards, certificates, contracts, plastic cards, etc. For the same reason, efficient methods for protection of author's images which are printed every day in mass media, or, for example, methods for control of facsimile communication safety, etc., are also in demand.

Let us examine how a watermarking system for information protection on paper carriers is functioning. In general case, we have initial digital object $I_0 = f(n,m)$ that is traditionally considered an image. According to certain algorithm, watermark *W* is embedded into I_0 or its identical copy and then the result (marked image) $I_w = f_w(n,m)$ is printed. Resultant paper copy is also considered as marked and it is the object that has to be protected in this task. In case of necessity for confirming its authenticity or settling copyright disputes, the paper object has to be scanned and then the certain synchronizing operations can be performed if they are required. And after that the presence of watermark *W* is detected in reconstructed digital object I'_w, or watermark bits are decoded (fig. 2.14).

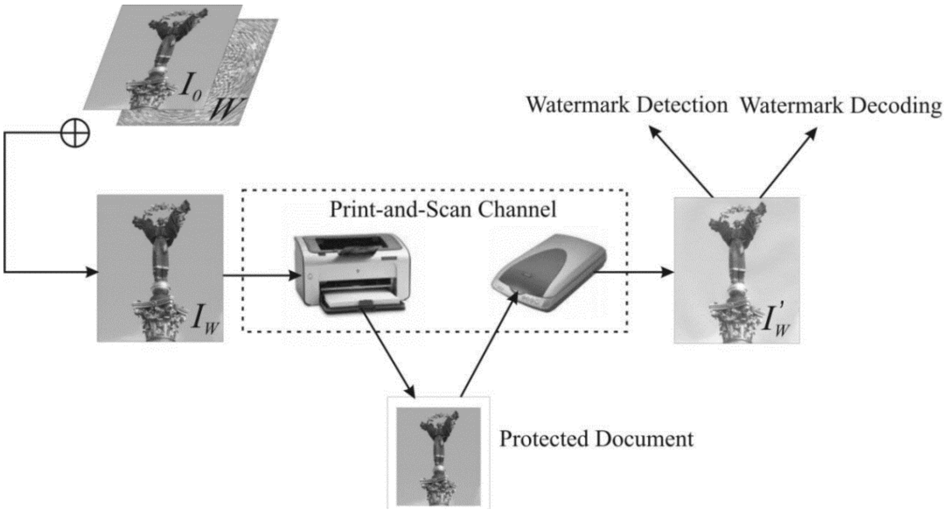


Fig. 2.14. General watermarking scheme for information protection on paper carriers

There is a growing interest to a problem of finding an invariant to the print-scan operation. However, taking into account complexity of the problem (great number of scanner and, especially, printer models, that realize the variety of existing technologies, a lot of setting options of these devices, paper texture characteristics, non-linear nature of transformations during print-scan process), the progress in this area is not very significant.

There are no efficient universal solutions for this problem. However, under certain limitation, it is possible to design analytical model and point out critical components that cause the highest distortions during print-scan.

So, printing process includes the following sub-processes that somehow have influence to the resultant total distortion:

Transformation of initial image sizes to internal image sizes for current printing resolution. On this stage, owing to interpolation, distortion of geometrical sizes inside the image is happening most often. The reason for using interpolation is non-square form of pixels in the screen and non-uniform resolution of printing devices horizontally and vertically (often printers have resolution like 4800 x 1200, i.e. it differs by 4 times). Moreover, e.g. for an image resolution of 96 dpi and a current printer resolution of 300 dpi, there is a fractional coefficient (3.125) for translation of the initial image to the internal image prepared inside the printer driver. Similar distortions can be one of two types: shrinking (and losing of small details) and expanding (appearance of squares).

Using a printer color profile. Hardware cannot guarantee correct and identical color rendition for all exemplars of devices (printers). Therefore, software color distortion is implemented to compensate aberrations of particular device. Aberrations are detected by calibration, which can be automatic, manual or semi-automatic. During image printing, transformation into color space of specific printer model can take place. E.g., RGB color space can be converted into CMYK model, additive, specific to printer and current printer settings, etc. For these conversions, non-linear interpolation inaccuracies arise at level of algorithms (initial values are distorted, similar colors are joined, number of steps for gradient filling is changed, etc.). After color space conversion, increasing of image saturation, contrast, etc. is often done.

Digital halftoning. Any type of printing, except sublimation ones, cannot make halftones. Photograph, obtained as a result of printing with an ensemble of hues and clear detailed representation of colors, is created from separate microscopic points of three, four or six printer’s colors with using halftoning. Halftoning algorithms are classified the following way: 1) algorithms of regular (amplitude-modulated) halftoning or ordered dithering [4], such algorithms are used in laser printers; 2) algorithms of frequency-modulated halftoning, in particular, diffusion algorithms [11], which are often used in ink-jet printers; 3) iterative algorithms, in particular, algorithm of direct binary search [6]. During the raster processing, quantization noise arises that causes appearance of colored noise, i.e. non-uniformity of image colors. Noised image is getting worse during its various serial (one-by-one) conversions.

Dot gain. Printed image suffers from a phenomenon called "dot gain". This phenomenon is that the images tend to appear darker than expected because of the colorant spreading type on the medium, optical edge effect and electrostatic reasons. This distortion is non-linear, but it can be roughly approximated by piecewise-linear curve. Many of digital halftoning algorithms include a model for the dot gain in their design.

Instability of printed copies. Printed copies of even the same digital objects are a little bit different. This distortion is caused, in particular, by moving and rotating of paper sheet during its placing into printer tray, mechanical reasons during operation of traction mechanism, which is not moving continuously but by micro-steps, color mixing, displacement and slight deformation of paper during printing, etc. Such uncertainty during printing leads to appearance of correlated noise. An example of printed copies instability is banding - an artifact, that consists of significant jumps of tone from one level to other, which means horizontal imperfections arising in the printouts.

During scanning, distortions can appear at the following stages of the process:

Matrix noise and scanner interpolation. Scanner matrix noise can be caused by various reasons, e.g. because of photon shot noise, read-out noise, matrix dark noise. This effect leads to randomness of low-order bits in the image pixel values. When scanning with resolution that is not equal to physical resolution of matrix, additional information is obtained not at the expense of image details accounting but with using of data interpolation, i.e. distortions caused by interpolation are arising.

Irregularity of scanning head move. Scanners with mechanical engines that lead matrix (scanner head) have some irregularities in their moving, that is result of inaccuracies in gears, engine controls etc.

Scanner gamma-correction. At the time of scanning, as well as during photo and video fixation, image undergoes gamma-correction by default. This means that during its digitization, non-linearity of human perception is taken into account. For computer, it is quite exactly compensated by exponential luminosity function of monitor. Gamma-correction (linearization) is used to avoid aliasing of digitization in the dark areas of image. At the same time, it decreases accuracy of color reproduction both directly during correction due to round-off errors and during further image processing. These transformations are best visible on the dark image areas.

Digitization. The scanned image must be digitized for storage; this inevitably leads to quantization errors, as a result of internal color depth change, and as a result of changing the image data format too. And since digitization follows previous non-linear distortions, the quantization noise effect may be amplified.

Geometric transformation. As a rule, during scanning the image undergoes next geometrical transformations: cropping, rotation and scaling. But their influence is nonequivalent. Distortions from scaling (during reduction of image data are discarded, during expansion they are added with using of linear, bilinear, or bicubic interpolation algorithms) can be easy compensated by fixing equal size for all processed images in the protection system. Apart from this, rotation can be automatically compensated. For example, Fourier spectrum properties are often used to do this.

Degree of influence of each sub-process on the resulting image depends on peculiarities of the devices and their settings. Analysis of sub-processes described above to the kind of introduced distortion allows make conclusion that all possible image distortions during print-scan can be separated to three groups: non-linear distortions, color noise and geometric transformations [7, 25]. The most present distortions are irreversible. Therefore, to keep watermark it is necessary to embed watermark bits into image areas that are the least sensitive to their influence.

2.3.2. Extraction of invariant to print-scan irreversible distortions in an amplitude spectrum of image

Since watermark robustness, as a rule, is growing when watermark is embedded into frequency domain of cover, let us consider Fourier image spectrum as an area for searching of the invariant and examine what influence non-linear distortions, color noise and geometrical transformations have on it.

Let $f(n,m)$ be an initial image of size $N \times M$. All definitions for Fourier transform that are valid for one-dimensional case (see item 2.1) can be easily translated to two-dimensional case. In particular, DFT of image is calculated as:

$$F(r,d) = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} f(n,m) e^{-2\pi i(\frac{rn}{N}+\frac{dm}{M})}, \; r = \overline{0, N-1}; \; d = \overline{0, M-1}. \quad (2.19)$$

Phases of spectrum are more sensitive to distortions than amplitudes, because they contain more information about the image. This fact is illustrated in Fig. 2.15, where fragment 2.15a shows the initial image, fragment 2.15b – the image for which DFT was done, then phases were discarded by accepting $\arg[F(r,d)] = 0$, and IDFT was done at the end; fragment 2.15c – the image for which DFT was done, then amplitudes were discarded by accepting $|F(r,d)| = 1$, and IDFT was done at the end.

From analysis of print-scan process, next distortions are irreversible for it: non-linear effects, colored noise, and moderate cropping of image. Taking into account the fact that amplitude but not phase image spectrum has more redundancy, let us study what influence the above-mentioned distortions have on it.

The main sources of non-linear distortions are transformation of the initial image sizes to the image sizes inside printer driver, using of printer color profile; dot gain that occur at the printer; and gamma-correction that occurs at the scanner. Non-linear distortions are increased due to following it quantizations: while digital halftoning for printing and while digitization during scanning.

Non-linear effects mostly have an impact on high-frequency and medium-frequency bands of image Fourier spectrum and have much less the impact on low-frequency band. Directly in low-frequency band, coefficients with amplitudes values lesser than adjacent ones, are distorted more than other.

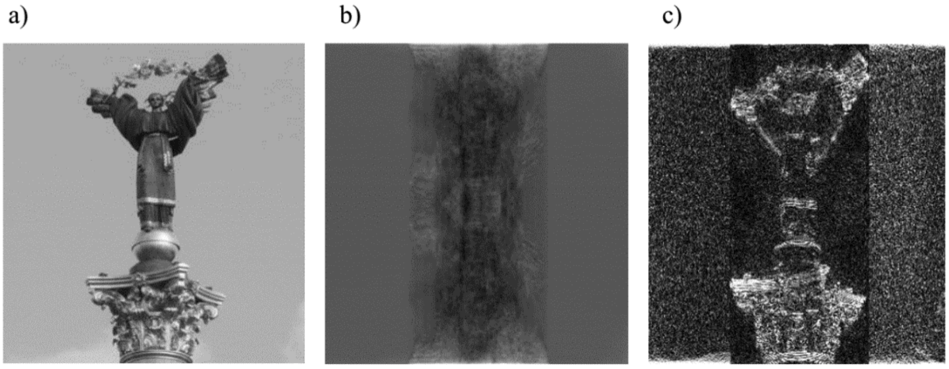


Fig. 2.15. Left to right: a – initial image, b – image reproduced from amplitudes, c – image reproduced from phases

With full access to the devices, i.e. possibility of their control and operation, non-linear distortions can be reduced by calibration of print-scan system. However, when development of copyright protection and electronic commerce systems, it is assumed that control of devices is unavailable. Therefore, protection system must be designed against the worst non-linear distortions.

Color noise is added to image during digital halftoning. Halftoning algorithms tend to place quantization noise at high frequencies, because human vision system is not very sensitive to high-frequency noise. It is possible to reduce the color noise, arising at this stage, with help of inverse halftoning that can lead to smallish softening.

Printing by itself is another source of color noise. Printing uncertainty and resulting instability of printed copies adds correlated noise (Fig. 2.16) that changes for each new printout.

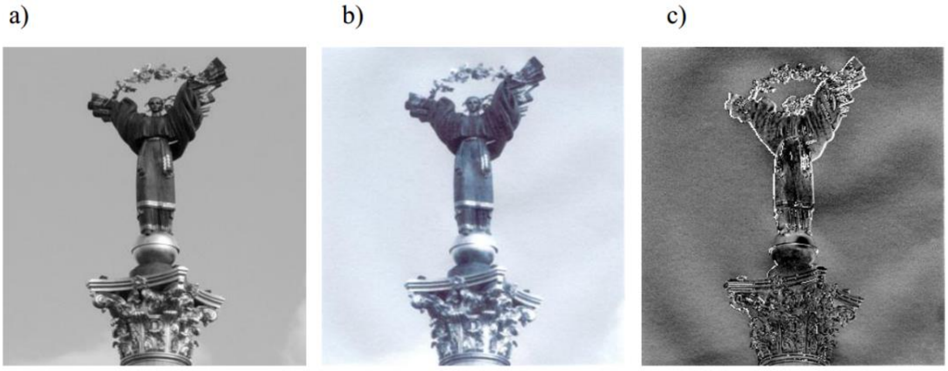


Fig. 2.16. Left to right: a – initial image, b – image after print-scan, c – correlated noise made during print-scan

Thereby according last two points, low-frequency DFT coefficients are more robust to print-scan than high-frequency ones.

Let us analyze how moderate cropping impacts the image.

Cropping process can be simulated as a product of initial image $f(n,m)$ and certain masking window $\phi(n,m)$, $n=\overline{0,N-1}$, $m=\overline{0,M-1}$. Masking window can be written as

$$\phi(n,m)=\begin{cases}1 \text{ if } (V_{1a}\le n\le V_{1b})\wedge(V_{2a}\le m<V_{2b}),\\ 0 \text{ otherwise.}\end{cases} \tag{2.20}$$

Here, V_{1a} and V_{1b} determine the top and bottom cropping, V_{2a} and V_{2b} – left and right edges respectively. So, cropped image sizes are $V_1 \times V_2$, where $V_1 = V_{1a} - V_{1b}$, $V_2 = V_{2a} - V_{2b}$. With that approach, cropped image is represented as

$$f'(n,m) = f(n,m)\cdot\phi(n,m),\ n=\overline{0,N-1},\ m=\overline{0,M-1}. \tag{2.21}$$

Let $F'(r,d)$, $F(r,d)$, and $\Phi(r,d)$ be two-dimensional DFT of cropped, initial images, and masking window respectively. Then, in frequency domain, spectrum of cropped image corresponds to circular convolution of initial image and masking window spectrums:

$$F'(r,d)=\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}F(i,j)\cdot\Phi\big((r-i)_N,(d-j)_M\big), \tag{2.22}$$

where $(\circ)_L$ denotes the modulo L operator.

It is known that DFT of rectangular impulse is a certain *sinc* function, i.e. in our case $\Phi(r,d)$ is two-dimensional *sinc*-like function (Fig. 2.17).

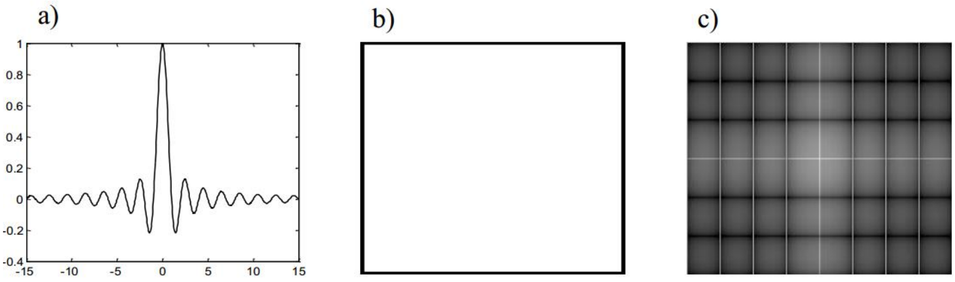


Fig. 2.17. Left to right: a – one-dimensional sinc function, b – example of masking window, c – two-dimensional amplitude spectrum of this window

Cropping has influence on all frequency bands and leads to spectrum blurring. For smallish cropping, most of the energy of $\Phi(r,d)$ is concentrated near (0,0)-coefficient. It corresponds to moderate blurring for coefficients with large amplitudes or similar values with adjacent ones, and to significant increase for coefficients, amplitudes of which are substantially smaller than adjacent ones.

So, DFT coefficients with large amplitudes are more robust to printing and scanning processes than coefficients with small amplitudes.

Conclusion. Digital watermark embedding for information protection on paper carriers can be based on following facts:

1. Low- and medium-frequency amplitude coefficients are kept much better than high-frequency ones. The tendency is traced that the lower the frequency the less distortion which it undergoes during print-scan.
2. During print-scan, in the low- and medium-frequency spectral bands, coefficients with small amplitudes will be increased, where as coefficients with large amplitudes will remain almost unchanged. And it is true for various images, printing techniques, resolutions of image, printer, or scanner. For example, Fig. 2.18 represents results of the experiment where image was printed out via ink-jet printer *Epson Stylus Photo R220* with 720dpi resolution and then scanned through flatbed scanner *hp scanjet 4400c* with 300dpi resolution. In Figs. 18a and 18c, there are amplitude spectra of blocks that extracted from low- and medium-frequency bands of image respectively. Here, the dark pixels correspond to small amplitudes. In Figs. 2.18b and 2.18d, there are differences of amplitude spectra between original and scanned images that correspond to previous fragments. Here, the light pixels correspond to the largest distortions. It is easy to see that the dark pixels in Fig. 2.18a are located in the same places as the light pixels in Fig. 2.18b, and the dark pixels in Fig. 2.18c are located in the same places as the light ones in Fig.2.18d. So, theoretical statements are confirmed by experimental data
3. It was determined by experiments that, if gamma-correction is not changed during print-scan, but its default value is used, coefficients with large amplitudes increase approximately by one. Roughly speaking, if print-scan process is approximated as a linear filter (for large enough coefficients and low enough frequencies), then, after use of standard gamma-correction, we will have predictable changes in above-mentioned amplitudes values.
4. For IDFT, modifications of the lower bits of large amplitude values in low-frequency range will be spread out all over the image and will not lead to significant perception distortion.

Therefore, low-frequency coefficients of Fourier spectrum with large amplitudes are the most robust to non-linear distortions, colored noise and moderate cropping and have some redundancy that can be used for embedding additional information.

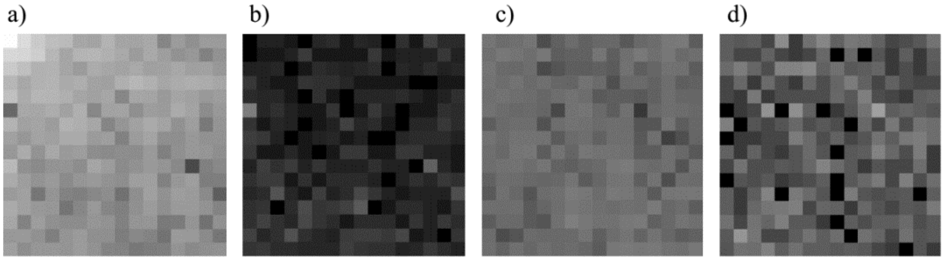


Fig. 2.18. Fragments of low-frequency (a)and medium-frequency (c) spectral bands and difference of spectra of original and scanned images for these fragments

2.3.3. Estimation and compensation of rotation for laser printers on the basis of Fourier spectrum properties

DFT coefficients are formed from Fourier spectrum of infinitely periodically reiterated discrete image that most often has sizeable leaps in colors on the borders of each period. This is manifested in amplitudes with large energy in horizontal and vertical directions from the origin (0,0). This phenomenon is known as "cross artifact". Image rotation always corresponds to rotation of its spectrum by the same angle, but with one refinement: if image was not cropped after rotation, cross artifact is rotating along with other spectral amplitudes, but if the image was rotated and cropped, all coefficients are rotated, except for cross artifact.

If the watermarking system will use regular halftoning algorithms at the stage of printing, this allows to automatically compensate of image rotation irrespective of watermark embedding and extracting, thus reducing list of conditions that watermarking must comply with. For simplicity, let us consider monochrome halftone image. From considerations of perception comfort, tilt angle of just one raster grid of such image for ordered dithering is equal to 45° (for colored image printing in CMYK system, cyan printing form is rotating by 15° or 105°, magenta one – by 75^0 or 15°, black one – by 45° or 135°, and yellow one – by 0° or 90°).

In the amplitude Fourier spectrum due to presence of ordered raster structure after image scanning at quite high resolution, peaks appear that correspond to tilt angle of raster grid. This is shown, for example, in Fig.19, where image was printed via laser printer Kyocera Mita FS-1010 KX with 600dpi resolution and scanned through flatbed scanner hp scanjet 4400c with 300dpi resolution. When increasing resolution, number of such peaks is also increasing.

Slight violation of the image proportions happens during printing. Due to this inconsistency, angle that measured in first or third quadrant of amplitude spectrum, is a bit different from that in second or fourth quadrant. For this reason, on practice, it is expedient to use average value of two non-coinciding angles for estimation of rotation angle.

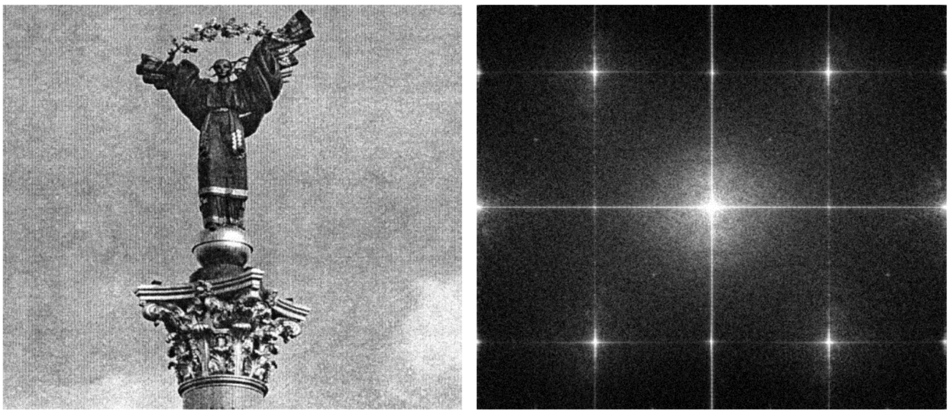


Fig.2.19. Scanned image and its amplitude Fourier spectrum

Taking into account the above-mentioned, method for evaluation and compensation of image rotation after scanning will consist of the following steps:

1. For given initial image $f(n,m)$ with size $N \times M$, it is necessary to calculate maximum value that is a power of two and does not exceed number of pixels the smaller side of image: $R = \max\limits_{r}\left(2^r \le \min(N,M)\right)$. It is expedient to use power of two to reduce computational complexity of the method.
2. If calculated *R* exceeds 1024, *R* = 1024 should be accepted; otherwise, it should be left unchanged (if large image must be processed, data of its small fragment are sufficient for the image rotation angle estimation).
3. Select fragment with size $R \times R$ in the scanned image center. Calculate FFT of this fragment and find maximum peaks in amplitude spectrum for first and second quadrants (without taking into account the cross artifact).
4. Let the angles between the peaks found and cross artifact will be α_1 and α_2 (Fig.2. 20). If the image was not cropped, these angles are determined using coordinate grid with vertical and horizontal axes. Rotation angle is calculated as $\alpha_r = \dfrac{\alpha_1 + \alpha_2}{2} - \dfrac{\pi}{4}$.
5. Image is rotated by angle α_r using, e.g., bicubic interpolation and it is cut from the background by finding edges with the highest difference in values of intensity.

Such geometric transformations as translation, scaling, rotation, cropping, are easy to be done using common software. They do not lead to watermark removal, but they are a reason of watermark desynchronization relative to cover object and, as a result, they make impossible its detection and decoding.

Desynchronization is a common problem of the watermarking systems, i.e. it can appear both as a result of digital-to-analog and analog-to-digital conversion of cover object and as a result of intentional or unintentional attacks on digital cover object.

There are two main approaches to solve the problem of watermark desynchronization in the cover object. First approach consists in estimation and compensation of geometric distortions prior to extracting of watermark. In this case, the watermarking system can use templates [19, 21, 26], self-reference watermarks [2, 10], feature points [3, 15], the Radon transform [24, 31], etc. The rotation compensation algorithm that was described above is also the example of this approach. The second approach consists in embedding of watermark into area that is invariant to geometric transformations. Extraction methods of such invariants can be constructed, e.g. on the basis of Fourier-Mellin transform properties [12, 16, 33]. Comparative analysis of watermark synchronization methods is done in the article [9].

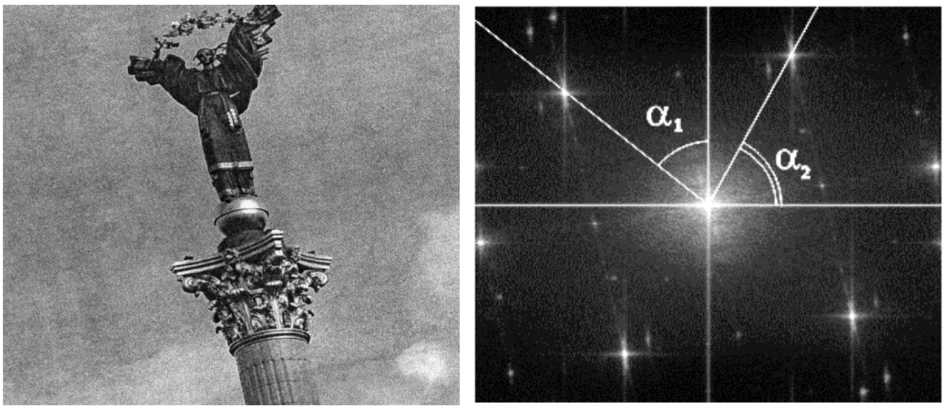


Fig. 2.20. Scanned image with rotation (and cropping) and its amplitude Fourier spectrum

2.4. References

1. Aldoshina, I.A., *Basic of psychoacoustic* (a cycle of articles, part 1-17), http://rus.625-net.ru/audioproducer/1999.
2. Alvarez-Rodriguez M., Perez-Gonzalez F., *Analysis of pilot-based synchronization algorithms for watermarking of still images*, “Signal Processing: Image Communication”, 2002 vol. 17, pp. 611-633.
3. Bas P., Chassery J.M., Macq B., *Geometrically invariant watermarking using feature points*, “Image Processing, IEEE Transactions”, 2002 vol. 9 (11), pp. 1014-1028.
4. Blatner D., Fleishman G., Roth S., *Real world scanning and halftones*, Berkeley, CA: Peachpit Press., 1998.

5. Fu Y., Ma Z., Song G., *A robust audio watermarking algorithm based on wavelet transform*, “Journal of Information and Computational Science 2”, 2005 vol. 5, pp.7-11.
6. Kacker D., Camis T., Allebach J. P., *Electrophotographic process embedded in direct binary search*, “IEEE Trans. on Image Processing”, 2002 vol. 11(3), pp. 243-257.
7. Koshkina N.V., *Invariant extraction for the print-scan process in tasks of computer steganography*, “Control Systems and Machines”, 2007 vol. 1, pp. 30-38.
8. Koshkina N.V., *Survey of Spectral Methods of Embedding of Watermarks into Audio Signals*, “Journal of Automation and Information Sciences”, 2010 vol. 5, pp.132-144.
9. Koshkina, N.V., *Methods of synchronizing digital watermarks*, “Cybernetics and Systems Analysis”, 2008 vol. 1, pp. 180-188.
10. Kutter M., *Watermarking resisting to translation, rotation and scaling*, “Proceedings of the SPIE: Multimedia Systems and Applications”, Boston, USA, 1998 vol. 3528, pp. 423-431.
11. Lau D., Arce G., *Modern Digital Halftoning*, Marcel Dekker, 2001.
12. Lin C.-Y., Chang S.-F., *Distortion Modeling and Invariant Extraction for Digital Image Print-and-Scan Process*, Intl. Symp. on Multimedia Information Processing, Taipei, 1999.
13. Lukin A., *Introduction to digital signal processing* (Mathematical foundations), Moscow, Lomonosov Moscow State University, 2002.
14. Mallat, S.G., *A Theory for Multiresolution Signal Decomposition: The Wavelet Representation*, “IEEE Transactions on Pattern Analysis and Machine Intelligence”, 1989 vol. 7(2), pp. 674-693.
15. Mizuki T., Nozomu H., *Affine Invariant Digital Image Watermarking Using Feature Points*, RISP International Workshop on Nonlinear Circuit and Signal Processing, Hawaii, USA, 2005.
16. O'Ruanaidh J.J.K., Pun T., *Rotation, scale and translation invariant spread spectrum digital image watermarking*, Signal Processing, 1998 vol. 66(3), pp.303-317.
17. Pan, D., *A Tutorial on MPEG Audio Compression*, “IEEE Multimedia”, 1995 vol. 2(2), pp.60-74.
18. Parshin B.J., Zhukov D.O., *Distinction DFT and MDCT at spectral compression of the audioinformation*, “Quality. Innovations. Education”, 2009 vol. 3, pp. 57-61.
19. Pereira S., Pun T., *Fast Robust Template Matching for Affine Resistant Image Watermarks*, Proc. of the Third International Workshop on Information Hiding, Dresden, Germany, Springer Verlag, 1999, pp. 199-210.

20. Petrovsky A.A., *Construction of psychoacoustic model in the wavelet-factors domain for perceptual processing of sound and speech signals*, "Speech technologies", 2008 vol. 4, pp.61-71.
21. Piva A., Barni M., Bartolini F., Cappellini V., Rosa A.D., Orlandi M., *Improving DFT watermarking robustness through optimum detection and synchronisation*, GMD Report 85, Multimedia and Security Workshop at ACM Multimedia, Orlando, FL.
22. Saito S., Furukawa T., Konishi K., *A digital watermarking for audio data using band division based on QMF bank*, "IEEE International Conference on Acoustics, Speech, and Signal Processing", 2002 vol. 4, pp. 3473-3476.
23. Sidelnikov, V.M., *Coding theory. Manual by principles and coding methods*, Moscow, Lomonosov Moscow State University, 2006.
24. Simitopoulos D., Koutsonanos D.E., Strintzis M.G., *Robust Image Watermarking Based on Generalized Radon Transformations*, CirSysVideo, 2003 vol. 8(13), s. 732-745.
25. Solanki K., Madhow U., Manjunath B.S., Chandrasekaran S., El-Khalil I.: *Print and Scan Resilient Data Hiding in Images*, "IEEE Transactions on Information Forensics and Security", 2006 vol. 4(1), pp. 464-478.
26. Solanki K., Madhow U., Manjunath B.S., Chandrasekaran S., *Estimating and undoing rotation for print-scan resilient data hiding*, "ICIP", 2004, pp. 39-42.
27. Vorobev V.I., Gribunin V.G., *Theory and practice of wavelet transform, Saint-Petersburg*, Military University of communication, 1999.
28. Wang Y., Wu S., Huang J., *Audio Watermarking Scheme Robust against Desynchronization Based on the Dyadic Wavelet Transform*, EURASIP Journal on Advances in Signal Processing, Article ID 232616, 2010.
29. Wu C.-P., Su P.-C., Jay K. C.-C., *Robust audio watermarking for copyright protection*, "Proceedings of SPIE, Advanced Signal Processing Algorithms, Architectures, and Implementations IX", 1999 vol. 3807, pp. 387-397.
30. Wu C.-P., Su P.-C., Kuo C.-C.J., *Robust and efficient digital audio watermarking using audio content analysis*, "Proc. of SPIE Electronic Imaging, Security and Watermarking of Multimedia Contents II", 2000 vol. 3971, pp. 382-392.
31. Yao Z., Rajpoot N., *Radon/Ridgelet Signature for Image Authentication*, 43-46 Proc. IEEE ICIP, Singapore, 2004.
32. Zadiraka V.K., *Theory of Fourier transform calculation*, Naukova Dumka, Kiev, 1983.
33. Zheng D., Zhao J., Saddik A.El., *RST-invariant digital image watermarking based on log-polar mapping and phase correlation*, "CirSysVideo", 2003 vol. 8(13), pp. 753-765.

3. MATHEMATICAL METHODS FOR MODELLING OF EMOTIONAL STATES ON HUMAN FACE

Iurii Krak, Iurii Kryvonos, Waldemar Wójcik

3.1. Introduction

Investigation of high-tech technologies passed to the stage of new complex problems decision, mainly socio-economic problems that displaced priorities of scientific and technical direction in the information services, medicine, ecology, transport fields and other aspects of steady development and upgrading life. Obviously these problems will actual during greater part of the XXI century.

From this point of view development of investigations on the human emotions modeling and recognition are very important first of all it applications to practical problems solving. In the last years the human nonverbal, mimic communication became intensive researches which allowed offering the original «formulas» of certain face mimic expressions. It allow done a step on the way of strict experimental research of expression reflection and well-possed problem of various facial expression perception and recognition. The insufficient developed of questions about various perception of facial expression what sufficiently sharply contrasts with practical necessities of these researches [10÷14] (information services, criminalistics, transport, etc.) were are main reasons to initiate given investigations. In this research, taken for basis the formal model of emotions [15] is extended for synthesis and analysis of mimic reflections of the human face emotional states. Results of this research can be the use for the design of human intellectual activity with application in the systems of artificial intelligence, as a constituent at development of algorithms and software tools for computer recognition and modeling (synthesis) and also for high-intelligence multimedia technologies creation.

3.2. Some approaches to mimic displays of emotions modelling

Investigation of emotions mimic expressions began more than 100 years ago. One of the first was paper Ch. Darwin “The Expression of the Emotions in Man and Animals” [3]. The Darwin’s hypothesis are consisted that mimic motions formed of useful effects. It is mean that presently is mimic expression of emotions before was a reaction with the some adaptability value. Directly, mimic motions are: weakened form of these useful motions, or their opposition, or direct expression of emotions. Darwin asserted that mimic reactions were innate and there in close intercommunication with the type of animal.

As marked in paper [6], there is no difference between mimicry of the grown man and child, except for its greater variety for adults. For all of people the same emotions expression are involved equal groups of muscles. Consequently mimic reactions appear innate. If a child does not have any mimic reactions, reason of it is only that her doesn't feeling such emotions.

But if to consider that mimic reactions are fully innate, from it mean out, that everybody must faultlessly "read" emotions on mimicry of other man. This statement afterwards in works of other researchers was refuted. It turned out as a result of their researches, that finding "typical" for all of people mimicry of fear, anger and other emotions is impossible. But also it was proved that in every man there is certain characteristic for him set of mimic reactions which repeat one in different situations. It appeared that the mimic imitation of emotions fit with the generally accepted expression forms, but quite does not match with the natural displays of those emotions, in an experiment tested.

In the paper [17] mimicry names as convention mimicry is given. It mean that, it is proved about the necessity of distinction of involuntary mimic reactions which are the end of the proper reflex, the psychical phenomena complicated, and arbitrary expressive actions which arise up as a result of person deliberately muscles reduction. There are 3 factors what influence on emotions mimic expression forming [17]:

- innate type of species mimic charts correspond the certain emotional states;
- purchased, learned by heart, the socialize methods of emotions display, arbitrarily controlled;
- individual expressive features are provided the specific and social forms of mimic expression specific lines, peculiar only to this individual.

In the paper [8] the system of objective code of mimic displays of basic emotions is developed. C.Izard the human anatomy learning is defined, which one muscles and how take part in the certain expressive changes of human face. Beginning and end of changes was registered in separate parts of human face (area of eyebrows, area of eyes, area of nose and cheeks, area of mouth), a concrete stimulus caused, and on a definite formula an emotion which this mimic image testifies are found. But this methodology in natural terms scarcely valid. C.Izard underlines that in the process of human teaching and socialization basic emotions expressions are modified.

In the paper [5] P.Ekman is fixed that there are seven basic expression of human face – mimicty configurations (charts) what reproduce seven emotions: happiness, surprise, fear, suffering, anger, disgust (contempt) and interest. It was shown that all of people, regardless of nationality and culture which they grew in, with sufficient exactness and co-ordination are interpret these mimic configurations as expressions of the proper emotions.

P.Ekman selected three autonomous areas of human face:

- area of forehead and brows;
- area of eyes (eyes, eyelids, bottom of nose);
- lower part of human face (nose, cheeks, mouth, jaws, chins).

The conducted investigations original «formulas» of mimic expressions what fix characteristic changes in each of three areas of human face are developed and also to construct the photo-standards of mimic expressions of corresponding emotions.

For emotional expresions modeling at first needed to define more detailed their dependences on person muscles movement. In paper [5] the system for modeling of all noticeable movement of human face is described. The system named Facial Action Coding System or FACS is based on enumeration of all of “action units” of human face what draw mimic movement. Some muscles draw anymore one unit of movement, that is why accordance between unit of movement and muscle movement is approximate.

In FACS there are 46 units of movement which changes in human face expression are registered and 12 units which changes head ansd eyes orientation are described.

3.3. A human face emotion synthesis

3.3.1. The formal model of emotions

For emotions formalization, in order to avoid ambiguities at their phenomenological description, it is suggested to pass the study of situations which these emotions arise up in [15]. That, at determination of emotions, a situation which they arise up in the most general view is described. Let will distinguish the name of emotion and its denotation.

Def.1. Under denotation we will mean a vector (Em) (that abstract concept) with the followings features:

$$Em_i^{\eta} = (\xi_1, \xi_2, \xi_3),\ i = \overline{0,7}\,, \tag{3.1}$$

where ξ are binary features which classify emotions:

- ξ_1 is a feature which determines the sign of emotion is a positive (1) emotion or negative (0). Will name an emotion positive, if it arises up in connection with satisfaction of necessity or achievement of purpose, and, accordingly, negative – in connection with dissatisfaction or not achievement;
- ξ_2 is a feature which determines time of origin of emotion in relation to an event (providing (0) for and establishing (1) emotions). Foreseeing emotions arise up to the event of the purpose related to achievement (by not achievement), preceded it;

- ξ_3 is a feature which determines the orientation of emotion. On this sign select emotions sending to itself (1) and sending to the external objects, on the other people (0).

The function of emotions is simplified consists in that emotions prepare an human organism to the certain action in a situation which arises up. Emotions are intended for the decision of universal vital difficulties, narrow circumstances. Every emotion prepares a man to some action. This action can be carried out with an external object or with a man. For example, anger aims at a removal obstacles for achievement of purpose and, thus, directed on an external object. Sadness prepares a man to do without that purpose which it was not succeeded to attain, and directed on itself.

Combining three binary features the 8 different variants are obtained. Let us enter a fourth feature (η).

Def.2. Under emotions on the source of their origin based will mean groups of emotions with the next features:

η=1 – emotions, related to satisfaction (by) of the personal necessities of man;

η=2 – emotions arise up as a result of comparison of some object, itself or the actions for the norms, standards, rules, persuasions;

η=3 – emotions arise up as a result of comparing of object to the public rules and norms;

η=4 – emotions arise up in connection with the necessities of other people;

η=5 – emotions arise up as a result of mutual relationships with other man;

η=6 – emotions arise up on the basis of contempt.

Combination of 4th features elements allow to describe 48 (8*6) high-quality different emotions.

The purpose of further effect of formalization is not definition of emotions, but selection of the names of emotions what most exactly satisfies the set of classifying features. For example, at the analysis of sorrow emotion a conclusion will be done, that it is the establishing negative emotion arises up in connection with the personal necessities directed on itself. It needs to be understood so, that for the establishing, negative, emotion what arises up in connection with the private necessities directed on itself, most exactly approach the name: «sorrow».

After setting every emotion of four classifying features, this set becomes its definition, whereupon under a term «sorrow» is understood not that phenomenological everybody imagines, but emotion what has the indicated set of features.

Thus, after proposed method emotions formalization, they become abstract objects and with them it is possible to operate in accordance with their definition, but not with the personal phenomenological experience.

3.3.2. Basic emotion definition

Farther, the offered features using, will to define emotions for the first group (emotions arise up on the basis of the personal necessities, $\eta = 1$) [15].

Neglect ($Em_0^1 = (0,0,0)$). It is arisen up in presentiment that an object will dissatisfy some our necessity. If a human led itself wrong, we gather a conclusion about its inability to give us that it is needed, and can feel neglect to it.

Fear ($Em_1^1 = (0,0,1)$). Fear arises up as a result of presentiment of throwing away an opportunity satisfaction of some personal necessity.

Anger ($Em_2^1 = (0,1,0)$). Anger arises up as a result of some personal necessity dissatisfaction, which stimulates a human on obstacle overcoming which interferes with its satisfaction.

Sorrow ($Em_3^1 = (0,1,1)$). Under sorrow will understand an emotion which arises up at some values loss.

Interest ($Em_4^1 = (1,0,0)$). Emotion arises up to the object with the help of which a human for to satisfy the necessity provides.

Hope ($Em_5^1 = (1,0,1)$). Emotion hope is arises up as a result of presentiment of satisfaction of the personal necessity.

Satisfaction ($Em_6^1 = (1,1,0)$). Satisfaction arises up as a result of some purpose achievement, related to the personal necessity, and on stopping of operating under achievement of this purpose directed.

Gladness (happiness) ($Em_7^1 = (1,1,1)$). Emotion arises up as a result of satisfaction of some personal requirement in wide sense. A typical situation for gladness emotion will be a situation of the achievement purpose desired.

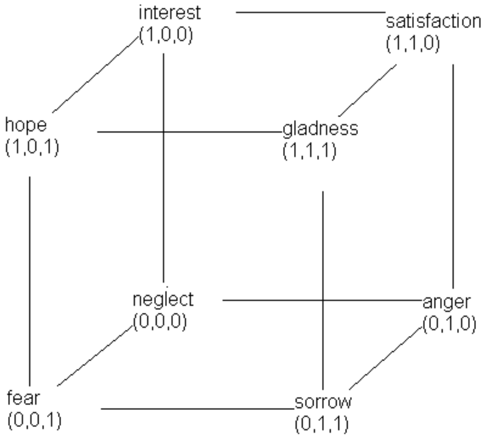


Fig. 3.1. Emotions arise up on the personal necessities basis

For evidently the defined vectors in a cube vertexes displayed (every cube will correspond the different sources of emotions (feature η). The vertical cube edge will correspond of the emotions ξ_1 sign. On an overhead cube face positive emotions will be disposed; on bottom cube face negative emotions will be disposed. The horizontal cube edge will correspond of origin emotion ξ_2 initiation duration. On the left cube face emotions which are preceded an event will be disposed, and on right – establishing. The cube edges are perpendicular a picture plane will be responsible for the emotion ξ_3 direction. On a front cube face there will be emotions directed on itself, and on back cube face there will be emotions directed on an object.

Emotions which arise up on the basis of the personal necessities ($Em_i^1, i = \overline{0,7}$) are represented on the Fig. 3.1.

Def.3. Set of vectors-emotions $Em_i^1, i = \overline{0,7}$ will define as basic (so as any other emotion can be as protuberant combination presented) and there is not a less set of emotions with such properties.

Proposed vectors of emotions $Em_i^1, i = \overline{0,7}$ using will build the mathematical model of emotions on the following chart:

1) will define emotions by various compound of 4th features of situations they arise up in;
2) will put every emotion in accordance some element of vector's space;
3) operation of addition between vectors is entered with the help of definition from the features of situations;
4) operation of multiply by a positive number modeling existence relatively of more strong and more weak identical emotions;
5) operation of multiply by a negative number represents the fact of opposite emotions existence.

In paper [15] it is proved that emotions can be presented as protuberant combination of two emotions from already considered: $Em_i^1, i = \overline{0,7}$.

Using this result any emotion will represent as:

$$Em_i^{\eta} = \alpha Em_k^l + \beta Em_i^1, \tag{3.2}$$

for $\alpha + \beta = 1, \beta > \alpha,\ \eta = \overline{2,6},\ i = \overline{0,7}, k \in [0,\ldots,7], l \in [1,\ldots,6]$.

In the formula (2) Em_i^{η} is mark of emotion number *i* for a cube number η ; Em_k^1 is an emotion from a cube number 1 for shift forming for a cube number η emotions generation; Em_i^1 is an emotion of a cube number 1 which is on a that edge, that and emotion which is generated (it mean that emotion which is generated must have also the same ξ_1, ξ_2 and ξ_3 that emotion from a cube number 1 and it must have greater weight what emotion Em_k^1 (therefore $\beta > \alpha$)).

3.3.3. Emotions arise up on the personal norms and rules

Let us define emotions for the second group (emotions arise up on the personal norms and rules, ($\eta = 2$)). Emotions, related to the personal norms and rules ($\eta = 2$) on the Fig. 3.2 presented. The shift emotion for cube number 1 is an emotion of satisfaction (Em_6^1).

The 8 different emotions are received.

Fault ($Em_3^2 = \alpha Em_6^1 + \beta Em_3^1, \beta > \alpha$) there is Satisfaction (own principles) + Sorrow (from a necessity to carry responsibility for principles violation).

Respect ($Em_6^2 = \alpha Em_6^1 + \beta Em_6^1, \beta > \alpha$) there is Satisfaction + Satisfaction (from that other human corresponding to these principles).

Self-respect ($Em_7^2 = \alpha Em_6^1 + \beta Em_7^1, \beta > \alpha$) there is Satisfaction + Gladness (from accordance itself to these principles).

Contempt ($Em_2^2 = \alpha Em_6^1 + \beta Em_2^1, \beta > \alpha$) there is Satisfaction + Anger (directed on overcoming of these principles disparity situation).

Sympathy ($Em_4^2 = \alpha Em_6^1 + \beta Em_4^1, \beta > \alpha$) there is Satisfaction + Interest (to the human which possibly, will individual necessities satisfy).

Antipathy ($Em_0^2 = \alpha Em_6^1 + \beta Em_0^1, \beta > \alpha$) there is Satisfaction + Neglect (to the human which possibly, will not individual necessity satisfy).

Responsibility ($Em_5^2 = \alpha Em_6^1 + \beta Em_5^1, \beta > \alpha$) there is Satisfaction + Hope (on that an individual will meet the standards).

Irresponsibility ($Em_1^2 = \alpha Em_6^1 + \beta Em_1^1, \beta > \alpha$) there is Satisfaction + Fear (possible disparity to the norms).

Using analogical method it is possible to define emotions for other parameters of η.

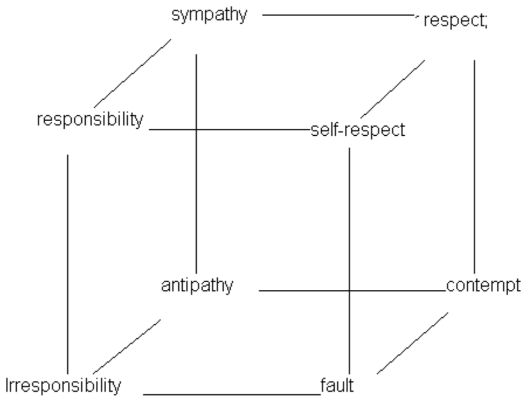


Fig. 3.2. Emotions related to the personal norms and rules

For emotions arise up as a result of accordance (disparities) somebody's or public standards, norms, rules ($\eta = 3$), the shift emotion is an emotion of **respect** (Em_6^2). Emotions which arise up in connection with somebody else's necessities ($\eta = 4$) have the shift emotion is **delightion** (Em_6^3). Emotions which arise up on the basis of mutual relationships with other people $\eta = 5$ have the shift emotion is **gratitude** (Em_6^4). Emotions utilize on the basis of contempt ($\eta = 6$) accordingly, for shift emotion is emotion of **contempt** (Em_2^2) used.

3.4. Modeling of emotions mimics

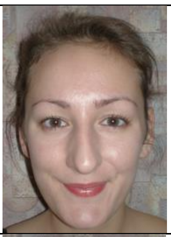
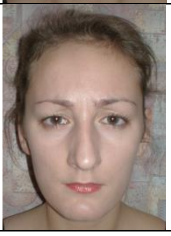
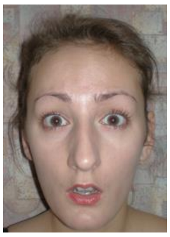
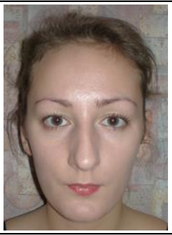
The research will be extended offered in paper [15] the formal specification of the emotional states for the design of mimic expressions of emotions. That the problem of emotions mimic expressions design is considered in obedience to offered at [15] the formal specification of base emotions. For the search of space of characteristic features, construction of base of this space, recreation of the derivative emotional states, with the following application of protuberant combination (2), the following is offered:

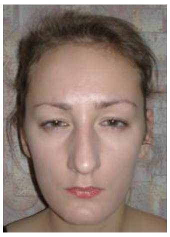
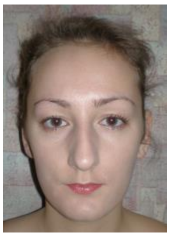
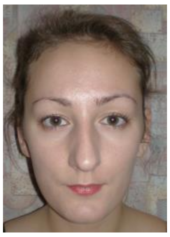
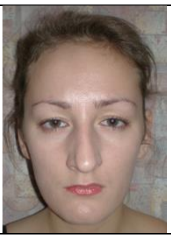
- creation of set of photographic images, on which situations (ξ_1, ξ_2, ξ_3) which base emotions are in will be reproduced actors, and description of mimicry, incident to these emotions;
- analysis of the received images set with the purpose of areas exposure of which contain the emotions characteristic features and description of them (use anatomic features and methodology of FACS);
- creation in the space of characteristic features of basis for a next decomposition on him arbitrary vectors of the emotional states mimic displays (as protuberant combination of the basis emotional states);

- analysis of characteristic features and ranging them for the influencing degree within the framework of the offered basis.

Descriptions of situations, which base emotions displayed, proper those photographic images and description of mimicry, what describe these states are pointed in the Table 3.1.

Table 3.1. Photo etalons of base emotions

Base emotion	Characteristics of the psychological situation that causes an emotion (ξ_1, ξ_2, ξ_3), ξ_1 – negative (0) or positive (1), ξ_2 – concerting future (0) or past (1); ξ_3 – externally cause (0) or inner (1)	Image of emotion	Description of mimics for face zones (1 – upper part; 2 – eyes; 3 – lower part)
Happines	ξ_1 =1 – satisfied need; ξ_2 =1 – the causing event is in the past; ξ_3 =1 – feeling of the (positive) result.		1. Brows and forehead calc; 2. Upper lids calm, lower lids up, wrinkles below lids. 3. Mouth shut lips corners widened and up.
Sorrow	ξ_1 =0 – unsatisfied need; ξ_2 =1 – the (unpleasant) event is in the past; ξ_3 =1 – feeling of loss		1. Inner parts of brows up; 2. Inner parts of lids up; 3. Mouth shut, lips corners lowered, no tension in mouth zone
Fear	ξ_1 =0 – unsatisfied need; ξ_2 =0 – feeling the lost something; ξ_3 =1 – fear for oneself and own need.		1. Brows up and uplift. Wrinkles in the center of forehead. 2. Upper lids up (see sclera), lower lids up and stretch. 3. Mouth open, lips stretched and tensed.
Hope	ξ_1 =1 – opposed to fear; ξ_2 =0 – feeling of the personal necessity satisfaction (feeling of gladness); ξ_3 =1 – directed on itself.		1. Top of brows corners up. 2. Upper lids are little up. 3. –.

Anger	ξ_1 =0 – negative emotion; ξ_2 =1 – it is arisen up after an event which brought necessities over to dissatisfaction; ξ_3 =0 – directed on an object which interferes with purpose achievement.		1. Brows down and uplift. There are vertical wrinkles between eyebrows. 2. Upper lids are stretch; bottom lids are tense and little up. 3. Mouth is closed, lips are clutched.
Satis-faction	ξ_1 =1 – opposite anger on a sign; ξ_2 =1 – arisen up after an event; ξ_3 =0 – it is demonstrated, that a necessity is satisfied with the help of concrete object.		1. –. 2. –. 3. Mouth shut, lips corners are widened and little up.
Interest	ξ_1 =1 – satisfaction of necessity; ξ_2 =0 – feeling of necessity satisfaction; ξ_3 =0 – directed on an object.		1. Eyebrows of little up, wrinkles on the forehead. 2. Eyelids are a bit extended. 3. –.
Neglect	ξ_1 =0 – dissatisfaction of necessity; ξ_2 =0 – feeling of necessity dissatisfaction; ξ_3 =0 – directed on an object.		1. Eyebrows of little up. 2. –; 3. Lips corners are little down. A human face is prolated, a head is elevated, as though a man looks at someone from above; it as though keeps away from an interlocutor.

For the analysis of the photographic images set is received with the purpose of exposure of areas what characteristic features of emotions contain, was taken approach, offered the authors of FACS [5]. A 21 characteristic features combination of which forms the emotions mimic expressions basis was received during research.

That means, emotion mimic expressions (Em) were represented as a vector:

$$Em_i^{\eta} = \left(\mu_1, \cdots, \mu_{21}\right),\ i = \overline{1,8}, \qquad (3.3)$$

where $\mu_j \in [0;1]$ is a characteristic mimic feature (at $\mu = 0$ – there is not a feature, and at the $\mu = 1$ influencing of feature is maximal) (see Table. 3.2).

Table 3.2. Mimic displays for base emotions forming

Features	*Description of emotional mimics on different facial zones*			*States of basic emotions* B_{ij}							
	Facial zone		*Mimics*	*Hap.*	*Sor.*	*Hop.*	*Fea.*	*Sat.*	*Ang.*	*Int.*	*Neg.*
μ_1	Zone of forehead and brows	Forehead	Lots of wrinkles on the forehead center	0	0	0	1	0	0	0	0
μ_2			Single horizontal wrinkle	0	0	0	0	0	0	1	0
μ_3			Single horizontal wrinkle between brows	0	0	0	0	0	1	0	0
μ_4		Brows	Inner parts up and centered	0	1	1	0	0	0	0	0
μ_5			Down and centered	0	0	0	0	0	1	0	0
μ_6			Up	0	0	0	0	0	0	1	1
μ_7			Up and centered	0	0	0	1	0	0	0	0
μ_8	Zone of eyes (eyes, yelashes base of nose)	Up yelashes	Up inner corners	0	1	0	0	0	0	0	0
μ_9			Stretch	0	0	0	0	0	1	0	0
μ_{10}			Up (see sclera)	0	0	0	1	0	0	0	0
μ_{11}			A little up	0	0	1	0	0	0	1	0
μ_{12}		Down yelashes	A little up and not stretch	1	0	0	0	0	0	1	0
μ_{13}			A little up and stretch	0	0	0	1	0	1	0	0
μ_{14}		Wrinkles	“crow's feet” near outside eyes corners	1	0	0	0	0	0	0	0
μ_{15}			Wrinkles under yelashes	1	0	0	0	0	0	0	0
μ_{16}	Inner part of forehead (nose, cheeks, mouth)	Mouth	Close, lips of compresses	0	0	0	0	0	1	0	0
μ_{17}			Opened	0	0	0	1	0	0	0	0
μ_{18}		Lips (line, corners)	Lips corners of stretched and a little up	1	0	0	0	1	0	0	0
μ_{19}			Elongated and stretches	0	0	0	1	0	0	0	0
μ_{20}			Lips corners are let down	0	1	0	0	0	0	0	1
μ_{21}		Wrinkles	Wrinkle from nose to lips corners	1	0	0	0	0	0	0	0

Ensembles of 8th vectors marked thus, forms the basis (B_{ij} , $i=\overline{1,21}$, $j=\overline{1,8}$) of emotional states mimic displays space.

Thus, arbitrary features vector $b=(\mu_1,...,\mu_{21})$, got by image with some emotional state analysis, it is possible to decompose on the basis of B and get description of emotion, as system solution:

$$x = (B^T B)^{-1} B^T b \,. \qquad (3.4)$$

Here B is the emotional states basis matrix of (see table. 2); B^T – the matrix transposition to matrix B; *b* is a vector which describes the mimic display of the arbitrary emotional state; $x = (\alpha_1,...,\alpha_8)$, where $\alpha_1,\ldots,\alpha_8$ are coefficients of protuberant combination $(\sum_{i=1}^{8}\alpha_i = 1, \alpha_i \in [0;1])$ for each with 8th base emotions.

So as impossible mimic to define the source of origin of emotion (η) the following set from a 21th emotion will get, which the offered method can be defined (Table. 3.3):

Table 3.3. Curriculum of mimic displays of the emotional states

No	Base states Emotions	Happines(1)	Sorrow(2)	Hope(3)	Fear(4)	Satisfaction(5)	Anger(6)	Interest(7)	Neglect(8)
1	Happines ($\eta = 0$)	***1***	0	0	0	0	0	0	0
2	Sorrow ($\eta = 0$)	0	***1***	0	0	0	0	0	0
3	Hope ($\eta = 0$)	0	0	***1***	0	0	0	0	0
4	Fear ($\eta = 0$)	0	0	0	***1***	0	0	0	0
5	Satisfaction ($\eta = 0$), respect ($\eta = 1$), fascination ($\eta = 2$), gratitude ($\eta = 3$), adoration ($\eta = 4$)	0	0	0	0	***1***	0	0	0
6	Anger ($\eta = 0$)	0	0	0	0	0	***1***	0	0
7	Interest ($\eta = 0$)	0	0	0	0	0	0	***1***	0
8	Neglect ($\eta = 0$)	0	0	0	0	0	0	0	***1***
$\alpha + \beta = 1, \beta > \alpha$									
9	Self-esteem ($\eta = 1$), pride ($\eta = 2$), gladness, for other ($\eta = 3$), fascination ($\eta = 4$)	β	0	0	0	α	0	0	0
10	Guilt ($\eta = 1$), shame ($\eta = 2$), pity ($\eta = 3$), offense ($\eta = 4$)	0	β	0	0	α	0	0	0

No	Base states Emotions	Happines(1)	Sorrow(2)	Hope(3)	Fear(4)	Satisfaction(5)	Anger(6)	Interest(7)	Neglect(8)
11	Sense of responsibility ($\eta = 1$), confidence in itself ($\eta = 2$), generosity ($\eta = 3$), pretentiousness ($\eta = 4$)	0	0	β	0	α	0	0	0
12	Irresponsibility ($\eta = 1$), avidity ($\eta = 3$), bashfulness ($\eta = 2$), prevention ($\eta = 4$)	0	0	0	β	α	0	0	0
13	Contempt ($\eta = 1$), disgust ($\eta = 2$), envy ($\eta = 3$), offended ($\eta = 4$), indignation ($\eta = 5$, $\beta >> \alpha$), complacency ($\eta = 5$, $\alpha >> \beta$)	0	0	0	0	α	β	0	0
14	Liking ($\eta = 1$), approval ($\eta = 2$), goodwill ($\eta = 3$), trustfulness ($\eta = 4$)	0	0	0	0	α	0	β	0
15	Antipathy ($\eta = 1$), indignation ($\eta = 2$), malevolence ($\eta = 3$), suspiciousness ($\eta = 4$)	0	0	0	0	α	0	0	β
$\alpha + \beta + \gamma = 1, \gamma > \beta > \alpha$									
16	Triumph ($\eta = 5$)	γ	0	0	0	α	β	0	0
17	Bitterness ($\eta = 5$)	0	γ	0	0	α	β	0	0
18	Advantage ($\eta = 5$)	0	0	γ	0	α	β	0	0
19	Humility ($\eta = 5$)	0	0	0	γ	α	β	0	0
20	Flatteries ($\eta = 5$)	0	0	0	0	α	β	γ	0
21	Haughtiness ($\eta = 5$)	0	0	0	0	α	β	0	γ

The next problem is analysing the characteristic features and ranked them for the offered basis degrees of influencing within the framework is investigated. For this purpose the singular values of base matrix B_0': $\sigma^0 = \{\sigma_1^o,...,\sigma_8^o\}$ will define. Here B_0' – is the aligned matrix B (it mean that beginning of co-ordinates is carried in the center of characteristic features space).

For determination of the singular values of matrix B_0' will decompose it on three matrices multiplication:

$$B' = UDV^T. \tag{3.5}$$

Here U is orthogonal matrix is formed the matrix $B'B'^T$; V is an orthogonal matrix is eigenmode vectors of matrix $B'^T B'$ formed; D is a diagonal matrix, that elements are singular values matrix B': $\sigma = \{\sigma_1,...,\sigma_8\}$ which equal square roots from the eigenmode values of matrix $B'^T B'$;

Let define a 21 matrix, $B_i', i = 1,\cdots,21$ each of which is formed from a matrix B_0' the way of *i*-th line zeroing. For every got matrix also will define singular values ($\sigma^i = \{\sigma_1^i,...,\sigma_8^i\}$). Will analyse distances in space from σ^0 to σ^i (usual Euclidean distance). These distances influencing each of 21 features will characterize. The results of such calculation on the Fig. 3.5 are presented.

Thus, a formal model, offered in the paper [15] for emotions presentation is extended, as protuberant combination of the base emotional states in case of visualization of these emotions mimic expressions. Space from 21-th of characteristic features and of this space basis is built. It enabled to decompose mimic presentation of arbitrary emotion in space of characteristic features as basis emotions protuberant combination. A set from a 21-th emotions is marked, which can be got as a result will decompose on the basis of characteristic features space. At introduction of origin source of these emotions (η) this set is represented in a set from 48 emotions in paper [15] defined. The analysis of influencing of every characteristic features at a decomposition on basis emotions in the built space is conducted.

A vector each of emotion mimic presentation reproduces of turns out a method, similar FACS offered in a method [5]. It means that a code is people conducted. Technology of flexible templates (as B-spline functions) by means which features space and basis space will be built automatically will be farther considered. It will allow to use the proposed formalism, both for a design and for automatic recognition of mimic displays of the emotional states.

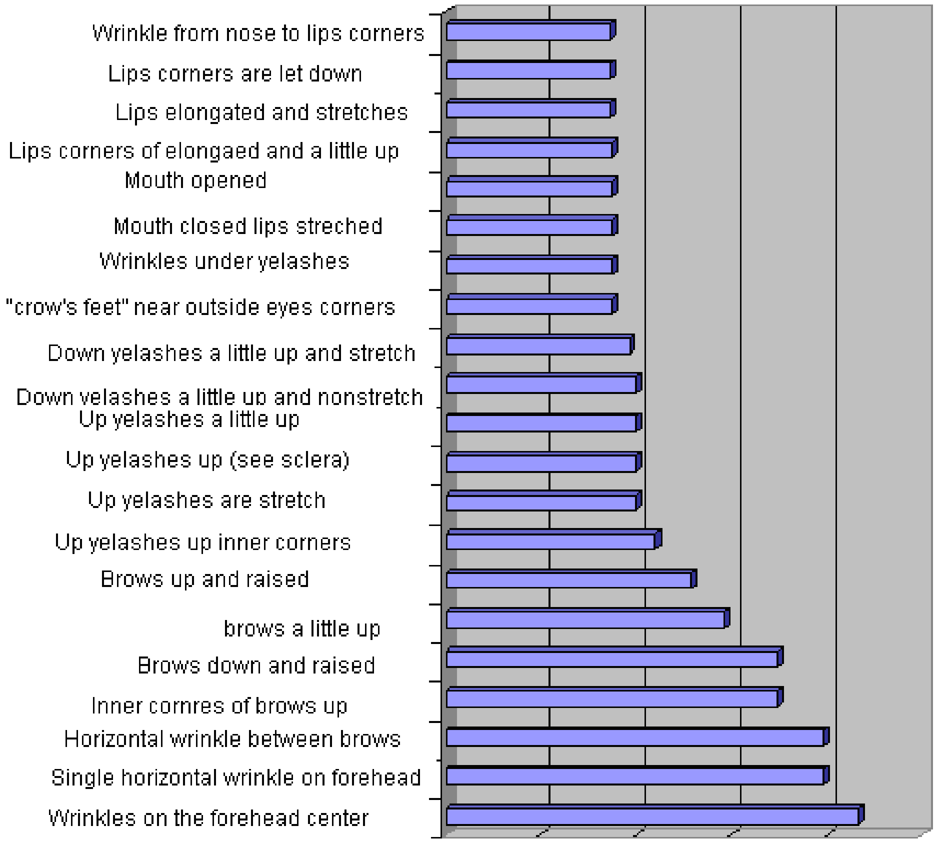


Fig. 3.5. The characteristic features influencing

3.5. A mimic component of emotions recognition

A mimic features of the emotional states of basis space construction was preliminary consider on the using of a priori experience of experimenter. It requires of certain qualification ones and, accordingly, gives an ambiguous result – establishment of the same emotional muscular display unequal for different people. For aim to pass from phenomenological definition of characteristic mimic features to their certain formalization, use own modification of method of deformable models are proposed.

3.5.1. Method of contour models which are deformed

Method of deformable models is got the wide distribution due to that these models have large flexibility (provide presentation of objects with a structure which differentiates strongly) and at the same time enable to build hard limits on the possible changes of objects form which appear.

The deformed curves are used for the features selection on an image is described in paper [9]. A lot of researches were since done in the investigations of perfection of deformable models and application of them in the different areas of images processing [2, 18].

For problem decision of face traits selection on a bitmapped image parametrical deformable models is used. A deformable model will named parametrical in the case when the object form, which model presents fully depends on some quantity of parameters (a bit in comparing to general complication of model). The estimation of accordance of model configuration to representation information on image is determined by means the model energy criterion. Model energy consists of internal energy value of which expresses accordance of configuration of model limitations, given of set experimenter and external energy which measures the criterion of co-ordination of model and image data. The process of model adaptation to the image consists in the search vector of parameters, what will realize global a maximum (minimum) of model energy. A concrete model is described the method of object problem form what is designed and function what calculates model energy.

The models form which are used for the face traits selection are represented by means of the parametrical curves set, with certain limitations, imposed on their possible configurations. Internal energy of models sets additional (less strict conditions) limits on the desired configurations, bringing in a fine in general energy of model at undesirable deformations. The external energy calculation is set coming from features images which meet in the area of selected object. There are features: defined values of brightness of pixels, gradient of brightness, colors of pixels, change of colors of pixels. A model external energy up when a model aims to occupy position on an image, where pixels will form structures, near to the object which is recognized. It can be determined optimization of model energy by means the methods of search local minimum (for example, method of gradient descent), or by the methods of global optimization (for example, genetic algorithms). The choice of optimization method is caused properties of recognized object and its position on an image.

3.5.2. Use of NURBS-curves in models which are deformed

For face traits characteristic features extraction utilize models which are represent with the help of Non-Uniform Rational Base Splines – NURBS-curves is suggested [1, 16].

Let NURBS-curves will be consider and let the array of supporting points (control points) – $p_0,\cdots,p_m$ are given. The problem is: it is needed to find a function, marked on an interval $u_{\min} \le u \le u_{\max}$, such is enough smooth and passes close to the supporting points. Let a sequence of knots (knot-vector) $u_0, u_1,\cdots,u_n$ is, such that:

$$u_{\min} = u_0 \le u_1 \le \cdots \le u_n = u_{\max}. \tag{3.6}$$

At the use of approximation by splines a function $p(u)$ looks like polynomial degree of d on an interval between neighboring knots:

$$p(u)=\sum_{j=0}^{d} c_{jk} u^{j}, \quad u_k < u < u_{k+1} \ . \tag{3.7}$$

Thus, to find a d-th degree spline $p(u)$ it will be to find $n(d+1)$ of three-dimensional vector-coefficients c_{jk} is needed. The equations are need for vector-coefficients c_{jk} finding may be to get, examining different limitations, related to function continuity and closeness to the control points criterion. Such approach to forming of spline is global – it is needed to solve the system from $n(d+1)$ equations relatively $n(d+1)$ unknown vector-coefficients c_{jk}, it mean that every got coefficient will depend on all of control points. Although such method of spline coefficients determination will provide the receipt of smooth curve which passes through the set control points, it not very much well conforms to the specific of computer graphics problems (for example, implementation in the real time objects rendering).

Chosen approach [1, 16] for forming of B-splines are consists with mean a determine spline in the terms of basis functions. Each of the basis functions are different from a zero only on an interval in a few knots. Consequently, it is possible to write down a function $p(u)$ in a form:

$$p(u)=\sum_{i=0}^{m} B_{id}(u) p_i \ . \tag{3.8}$$

Here every function $B_{id}(u)$ is a polynomial degree of d on a few knots interval and equals zero outside this interval.

There are a lot methods for basis functions determination but very important one of them - it C. De Boor's recursive functions method [1]:

$$B_{k,0} = \begin{cases} 1, if \ u_k \le u \le u_{k+1} \\ 0 - \text{in other} \end{cases},$$

$$B_{k,d} = \frac{u-u_k}{u_{k+d}-u_k} B_{k,d-1}(u) + \frac{u_{k+d}-u}{u_{k+d+1}-u_{k+1}} B_{k+1,d-1}(u). \tag{3.9}$$

Every function from the first set – B_{k0} is constant on one interval and equals zero after it ones. Every function from the second set – B_{k1} is linear on two intervals and equals zero after their intervals. Every function from the third set – B_{k2} is the quadratic curve form on three intervals and equals zero after their limits et cetera (see Fig. 3.6).

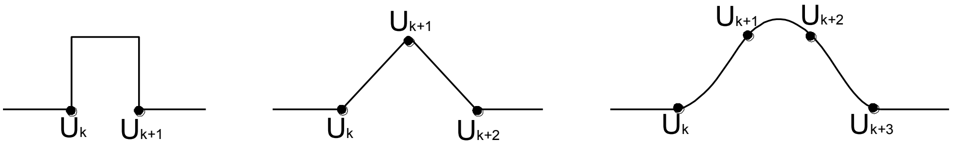


Fig. 3.6. Three basis C. De Boor's functions

For the general case: every function from the set of $d - B_{kd}$ has a different from a zero value on $d+1$ intervals between u_k and u_{k+d+1} control points and is a d-th degree polynomial on each of these intervals. A spline-curve, as a sum of such basis functions is weighed formed will lie into a protuberant polygonal envelope because a condition for basis functions is executed:

$$\sum_{i=0}^{m} B_{id}(u)=1, \quad 0 \le B_{id}(u) \le 1. \tag{3.10}$$

Every function B_{id} is non equal zero only on $d+1$ intervals; it is means that every control point has influence only on that part of total curve which lies into a envelope, by a $d+1$ control points created.

The set of B-spline base functions is determine the spline degree and knots array. But for determination of spline in a range from u_0 to u_{n+1} it is needed to have recursive functions, different from a zero in knots from u_0 to u_{n+d}. Therefore may need yet $d-1$ of addition values in knots, which can be got from limitations, imposed on descriptions of B-spline curve in initial and finite points. Taking into account that at recursive formulas calculation (9) a result from dividing of zero by a zero equals 1 it is possible to have multiple knots (such which repeat oneself). At the knots reiteration the effect of approaching of the formed B-spline appears to the control point with this knot associated. If the multiplies of knot is equal $d+1$ than B-spline degrees d will pass through the proper control point.

Thus, methods of shortage data for forming of spline problem decision consists to raise knots multiplies, proper initial and end points, and to compel a resultant curve to get through these points. In general case, it is possible to do multiple and internal knots, and also to place them nonuniform.

Control points $p_i = [x_i, y_i, z_i]^T$ can be written down in space of homogeneous co-ordinates as follows: $q_i = w_i[x_i, y_i, z_i, 1]^T$. An idea consists in that, to enter coefficients w_i for weight increase or diminishing of concrete control point. These weighed control points for forming of the four measured real-valued B-spline can be used. The three first component of the got spline will look as ordinary B-spline of presentation of the weighed control:

$$q(u) = [x(u), y(u), z(u)]^T = \sum_{i=1}^{n} B_{i,d}(u) w_i p_i \ . \tag{3.11}$$

A component w is a scalar polynomial B-spline, on the sets of weighing coefficients values formed:

$$w(u)=\sum_{i=0}^{n} B_{i,d}(u) w_i \ . \tag{3.12}$$

For fourth-parametrical splines using as of homogeneous co-ordinates components set, value of w, it can't equal **1** that is why in transition to three-dimensional space it to execute perspective transformation is needed:

$$p(u)=\frac{1}{w(u)} q(u)=\frac{\sum_{i=0}^{n} B_{i,d}(u) w_i p_i}{\sum_{i=0}^{n} B_{i,d}(u) w_i} \ . \tag{3.13}$$

Every component of function $p(u)$ is a rational function the parameter u and, as, no limits on the knots location were imposed, this function behaves to the class of NonUniform Rational B-Splines (NURBS).

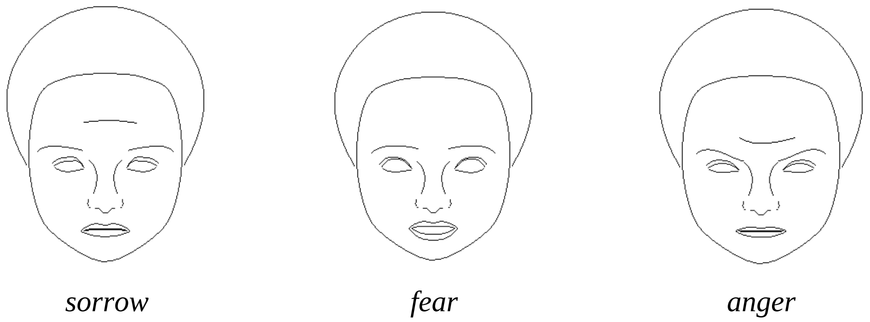
Advantages for human face contour modeling using NURBS-curves are:

- dimension on orders reduced;
- curves deformations (for the imitation of emotions mimic movement) are more smooth – similar to deformations of the real human faces.

The dimension reducing is as result that we pass from pixel space of photographic image (hundreds thousands points) with a help (3.13) to space of control points of NURBS-curves (p_i) (ten of points). Smooth deformation of NURBS-curves is follow out from their properties.

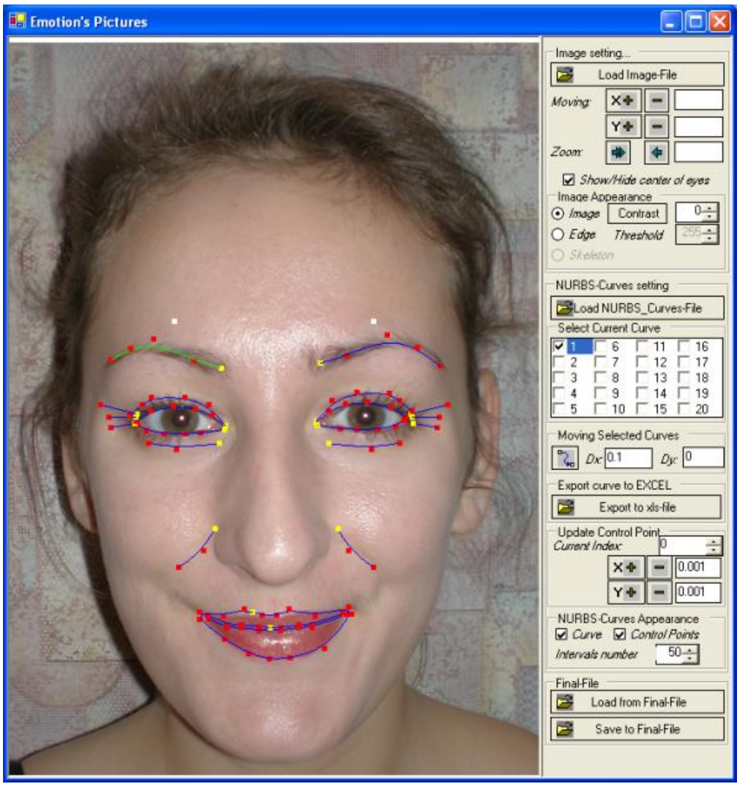
3.5.3. Using NURBS-curves for emotions mimic modelling

Within the framework of the conducted researches, a model was built which consists of the followings flexible templates – NURBS-curves (see Fig. 3.7): eyebrows, eyes, mouth, wrinkles.



sorrow *fear* *anger*

Fig. 3.7. Human face flexible model, built by NURBS-curves



☑ Show/Hide center of eyes

● Image Contrast 0

○ Edge Threshold 255

○ Skeleton

☑ 1 ☐ 6 ☐ 11 ☐ 16
☐ 2 ☐ 7 ☐ 12 ☐ 17
☐ 3 ☐ 8 ☐ 13 ☐ 18
☐ 4 ☐ 9 ☐ 14 ☐ 19
☐ 5 ☐ 10 ☐ 15 ☐ 20

Dx: 0.1 Dy: 0

Current Index: 0

X+ − 0.001

Y+ − 0.001

NURBS-Curves Appearance

☑ Curve ☑ Control Points

Intervals number 50

Fig. 3.8. Model of the emotional state: “Happiness” attachment

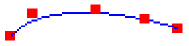
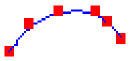
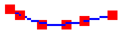
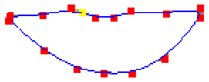
Original software (see Fig. 3.8) which provided necessary functionality for researches was created. Namely:

- display of human face photographic image by possibility of his normalization (on a distance between the pupil’s centers as metrics), moving and rotation;
- possibility of NURBS-model display with its subsequent modification (curves moving on the image surfaces, change of curve form by control points p_i (3.13) modification;
- export of the model ($p(u), B_{i,d}, p_i$) tied to the image in MS Excel (for subsequent researches).

Within the framework of the conducted researches a model what flexible templates consists – NURBS-curves is built. Coming from that the control points of NURBS-curve determine curve simply – by control points vectors consideration only. For the basis construction the next vectors of control points are used (see table. 3.4).

A mathematical model and integral information technology for the arbitrary emotional state on the concrete man face automatic definition as protuberant combination of some base states is offered. For this purpose with the help of mathematical model and original software basis space of concrete man face emotional states is created. In future, the arbitrary emotional display of this man is decomposes out, as protuberant combination of the emotional states of this space. But, in this model, flexible templates tuned in to the display of concrete mimicry with the help of the manual editing of certain quantity parameters – control points of NURBS-curves on the surface of image. Subsequent researches in this direction to modification of contour models which are deformed were sending. For the aims of automatic models deformation it was suggested to use B-spline of curves approximation [16].

Table 3.4. Mimic displays are for forming of base emotions

Features	Description of mimic displays is in the cut of face areas			For eyebrows, eyes and company – vectors of supporting pointsFor wrinkles – $\mu_j \in [0;1]$ (at $\mu = 0$ – there is not a wrinkle, and at a $\mu = 1$ wrinkle is maximal)	
	Human face area		Mimic display		
μ_1	Area of brows and eye-brows	Brow	Wrinkles are in the center of brow		
μ_2			One horizontal wrinkle		
μ_3			Between eyebrows vertical wrinkle		
μ_4		Eyebrows	Heaved up internal corners	$p_i^1, i = \overline{0,4}$	 Left eyebrow
μ_5			Prolapses and erected		
μ_6			Little up		
μ_7			Lifted and erected		
μ_8	Area of eyes (eyes, eyelids, basis of nose)	Overhead eyelids	Heaved up internal corners	$p_i^2,\ i = \overline{0,5}$	 Left overhead eyelid
μ_9			Tense		
μ_{10}			Lifted (see sclera)		
μ_{11}			Little up		
μ_{12}		Lower eyelids	Little up and not tense	$p_i^3,\ i = \overline{0,5}$	 Left lower eyelid
μ_{13}			Little up and tense		
μ_{14}		Wrinkles	"Goose quotation marks" are near external corners		
μ_{15}			A wrinkle is under eyelids		
μ_{16}	Lower part of person (carried, cheeks, mouth)	Mouth	Closed, it is pursed one's lips	$p_i^4,\ i = \overline{0,16}$	 Lips
μ_{17}			Exposed		
μ_{18}		Lips (line, corners)	Lips corners are drawn aside in sides and little up		
μ_{19}			Stretched and tense		
μ_{20}			Lips corners is dropped		
μ_{21}		Wrinkles	A wrinkle is from a nose to lips corners		

3.5.4. A B-spline curves approximation

Will be use easy NURBS-curves property: at $w_i = 1$ a NURBS-curve is a B-spline curve. This simple property is consequence from the identity of control points (p_i) in a homogeneous form and equality of denominator 1. Taking into account that at the design of flexible templates $w_i = 1$ was accepted for simplification of approximation it is possible to pass to B-spline curves.

The problem of B-spline of approximation is the problem of the fitting of B-spline curve from K control points $p = [p_0, \cdots, p_{K-1}]^T$ to the spot points curve $r = [r_0, \cdots, r_{M-1}]^T$, where $M > K$ (usually $M >> K$) for the parameters values $u_0, \cdots, u_{M-1}$.Such approximation problem of results in the redefined system of linear equalizations

$$N \cdot p = r\,, \tag{3.14}$$

or in expanded form:

$$\begin{bmatrix} N_0(u_0) & \cdots & N_{K-1}(u_0) \\ N_0(u_1) & \cdots & N_{K-1}(u_1) \\ \vdots & \ddots & \vdots \\ N_0(u_{M-1}) & \cdots & N_{K-1}(u_{M-1}) \end{bmatrix} \cdot \begin{bmatrix} p_0 \\ \vdots \\ p_{K-1} \end{bmatrix} = \begin{bmatrix} r_0 \\ r_1 \\ \vdots \\ r_{M-1} \end{bmatrix},$$

where $N_i(u)$ is a B-spline basis function.

One of ways of the redefined system of linear equalizations of decision is:

$$N^T N \cdot p = N^T \cdot r\,. \tag{3.15}$$

Whereof the unknown control points parameters $p = [p_0, \cdots, p_{K-1}]^T$ will be defined as:

$$p = \left(N^T N\right)^{-1} \cdot N^T r\,, \tag{3.16}$$

where the next condition will performed:

$$\det\left(\left(N^T N\right)^{-1}\right) > 0\,. \tag{3.17}$$

For application the B-spline of approximation needs to be able to get on an image point curves $r = [r_0, \cdots, r_{M-1}]^T$ which necessary contours meet in order farther to apply transformation (15)-(17).

3.5.5. Technologies for images contouring and frameworking

There are a lot of technologies for a receipt of the image curves of points $r = \left[r_0, \cdots, r_{M-1}\right]^T$, which correspond of eyebrows, eyes, and mouth contours. They are based mainly on a receipt the contour of image as sharp border between image elements (with the help of convolutions, color analysis, and others) with next frameworking (by a receipt the contour of single thickness).

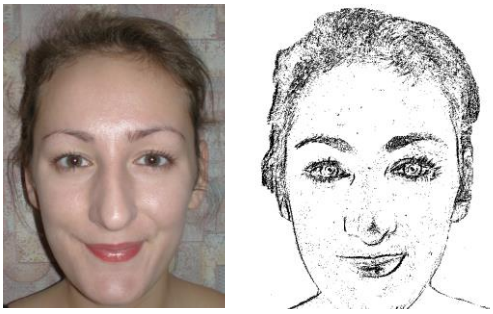


Fig. 3.9. Image: to and after contouring

Imitation of human eye visual receptors work for a image contouring is applied. It is known [4] that an eyeball is in continuous micromovement. Information about these micromovements has ambiguous interpretation. It is possible to provide for, that these micromovements are a necessary condition operating for contours selection on an image. For verification of it will compel the artificial eye retina receptors to fix the offered image, and after will move an image an insignificant rank (for example, on a 1 point) in a side, and again will enable the receptors of eye to fix him. In this moment on the outputs of receptors the relative change of signal will appear. Let take of receptors changes value and will add them to the proper points on an image – receive the contours of image (see Fig. 3.9):

The imitation of eye retina receptors passes as follows. There is an image and direction of micromovement (for example, diagonally on L of points). At first a concrete receptor «sees» a point with co-ordinates (x, y) and after micromovement – with co-ordinates $(x - L, y - L)$. Difference of color planes between an entrance point and point which appeared in her place as a result of micromovement, – it and there is a relative change of input irritating signal (for a concrete receptor).

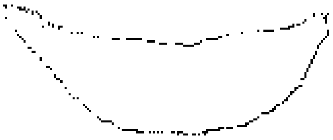


Fig. 3.10. Point curve of lips, obtained after the contour frameworking

The contours over got thus need to be brought to the framework kind. That it is needed to select some middle line which correctly would represent a contour structure. For this purpose the known algorithm of Zhang-Suen [19] will apply. A framework is got it lips contour represented on a Fig. 3.10.

The basic idea of Zhang-Suen's algorithm consists in that at every step, passing on an image a window $3\times3x$, belonging of every pixel is checked up to the set coherent area border. A pixel is withdrawn from an area if the terms of verification are executed. Without regard to the amount of the executed steps, an area will remain linked, in extreme case it in a line in thick in one pixel will degenerate.

3.6. Technology of mimic emotions displays recognition

For the emotions mimic expressions recognition the following integral information technology is offered:

1) for a concrete person face a set from 8 photographic images get on which mimic reaction on a situation correspond of basis emotions is reproduced: happiness, sorrow, hope, fear, satisfaction, anger, interest, neglect;
2) Normalization of photographic images in proper software (as distance between the eyes centers normalization);
3) used technologies of images contouring and frameworking, the contours of next person face parts are received: wrinkles in the areas of brow, eyebrow, overhead eyelids, eyelids, wrinkles «goose quotation marks» near external eyes corners, wrinkles under eyelids, mouth, wrinkles from a nose to the lips corners;
4) useding flexible templates as NURBS-curves and B-spline approximation the set of NURBS-curve control points for templates of each of 8 basis emotions are obtained:

$p_i^{1,(e)} = \left[x_i^{1,(e)}, y_i^{1,(e)}\right]^T, i = \overline{0,4},\ e = \overline{1,8}$ – template of the left eyebrow for 8 emotional states;

$p_i^{2,(e)} = \left[x_i^{2,(e)}, y_i^{2,(e)}\right]^T, i = \overline{0,5},\ e = \overline{1,8}$ – template of the left overhead eyelid for 8 emotional states;

$p_i^{3,(e)} = \left[x_i^{3,(e)}, y_i^{3,(e)}\right]^T, i = \overline{0,5},\ e = \overline{1,8}$ – template of the left bottom eyelid for 8 emotional states;

$p_i^{4,(e)} = \left[x_i^{4,(e)}, y_i^{4,(e)}\right]^T, i = \overline{0,16}, e = \overline{1,8}$ – lips template for 8 emotional states; and for wrinkles description some characteristic mimic features are obtained: $\mu^{(e)} = \left[\mu_1^{(e)}, \mu_2^{(e)}, \mu_3^{(e)}, \mu_{14}^{(e)}, \mu_{15}^{(e)}, \mu_{21}^{(e)}\right]^T, e = \overline{1,8}$ - characteristic mimic features;

6) from 8 sets of control points of NURBS-curves (templates) and vector of characteristic mimic features for wrinkles will build the basis of the emotional states (matrix B) of concrete human face:

$$B = \begin{bmatrix} \mu^{(1)} & \cdots & \mu^{(8)} \\ \mathrm{P}^{1,(1)} & \cdots & \mathrm{P}^{1,(8)} \\ \mathrm{P}^{2,(1)} & \cdots & \mathrm{P}^{2,(8)} \\ \mathrm{P}^{3,(1)} & \cdots & \mathrm{P}^{3,(8)} \\ \mathrm{P}^{4,(1)} & \cdots & \mathrm{P}^{4,(8)} \end{bmatrix}. \tag{3.18}$$

Here $\mathrm{P}^{j,(k)} = \begin{bmatrix} p_0^{j,(k)} \\ \vdots \\ p_{n_j}^{j,(k)} \end{bmatrix}, \ j = \overline{1,4}, \ k = \overline{1,8}, \ n_1 = 4, n_2, n_3 = 5, n_4 = 16$.

The next steps for the analysis of arbitrary image of this man face: let repeat steps 1)-5) for the arbitrary emotion image and vector received:

$$b = \begin{bmatrix} \mu^{(k)} \\ \mathrm{P}^{1,(k)} \\ \mathrm{P}^{2,(k)} \\ \mathrm{P}^{3,(k)} \\ \mathrm{P}^{4,(k)} \end{bmatrix}, \ \mathrm{P}^{j,(k)} = \begin{bmatrix} p_0^{j,(k)} \\ \vdots \\ p_{n_j}^{j,(k)} \end{bmatrix}, \ j = \overline{1,4}, \ k = \overline{1,8} \ n_1 = 4, n_2, n_3 = 5, n_4 = 16. \tag{3.19}$$

We will decompose vector b (3.19) on the built basis of B (3.18):

$$B^T B \cdot x = B^T \cdot b. \tag{3.20}$$

$$x = \left(B^T B\right)^{-1} \cdot B^T b. \tag{3.21}$$

Here the next condition will performed:

$$\det(\left(B^T B\right)^{-1}) > 0. \tag{3.22}$$

Coefficients of vector $x = (\alpha_1, ..., \alpha_8)$ decomposition a concrete contribution each of eight base emotions in the arbitrary emotion of b will specify.

3.7. Research results

The basis set of emotions photographic images was created. From the got images, describe by the proposed methods, need for subsequent processing contours (eyebrows, eyes, lips, and others) are selected. Fig. 3.11 contains the contours of right eyebrow for emotions: happiness, sorrow, hope, fear and satisfaction.

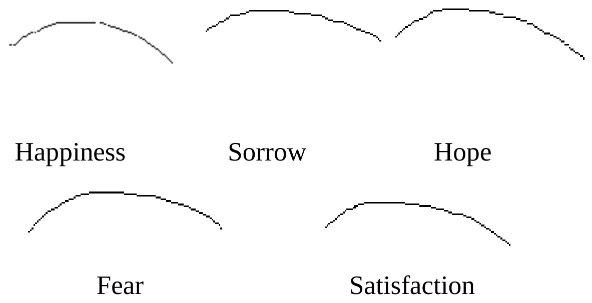


Fig. 3.11. Contours of right eyebrow are for some emotions

On a Fig. 3.12 contours of right eyebrow are given as a point curve.

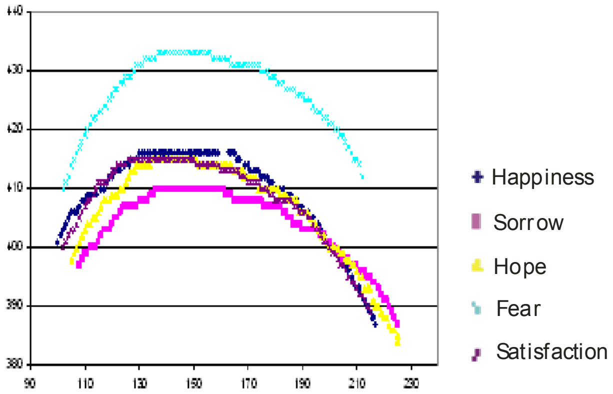


Fig. 3.12. The point curves of right eyebrow for some emotions

On the graph evidently, that the got contours of right eyebrow positions are correspond with mimicry description. It mean that for satisfaction and happiness emotions of special mimicry display it is not, for the emotions of sorrow and hope – internal corners heaved up and for fear emotion – heaved up an eyebrow and erected.

Equactions (3.15) – (3.17) were used to the got contours for the receipt of sets NURBS-curves control points. Fig. 3.13 contains the graph a contour and proper this contour NURBS-curve for position of right eyebrow at the happiness emotion.

A Fig. 3.14 the result of attachment of flexible templates (NURBS-curves) to the proper fragments of human face for all of base emotions are shown.

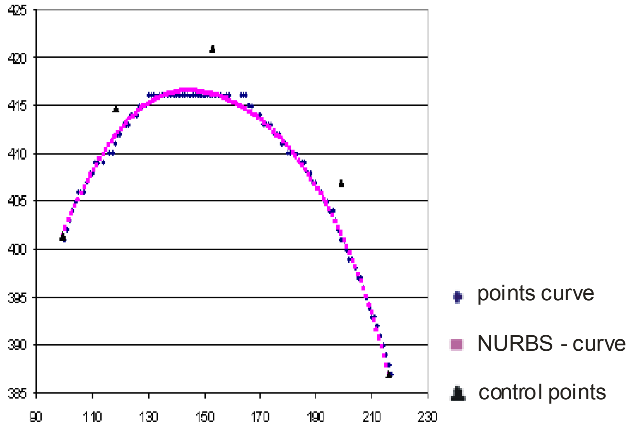
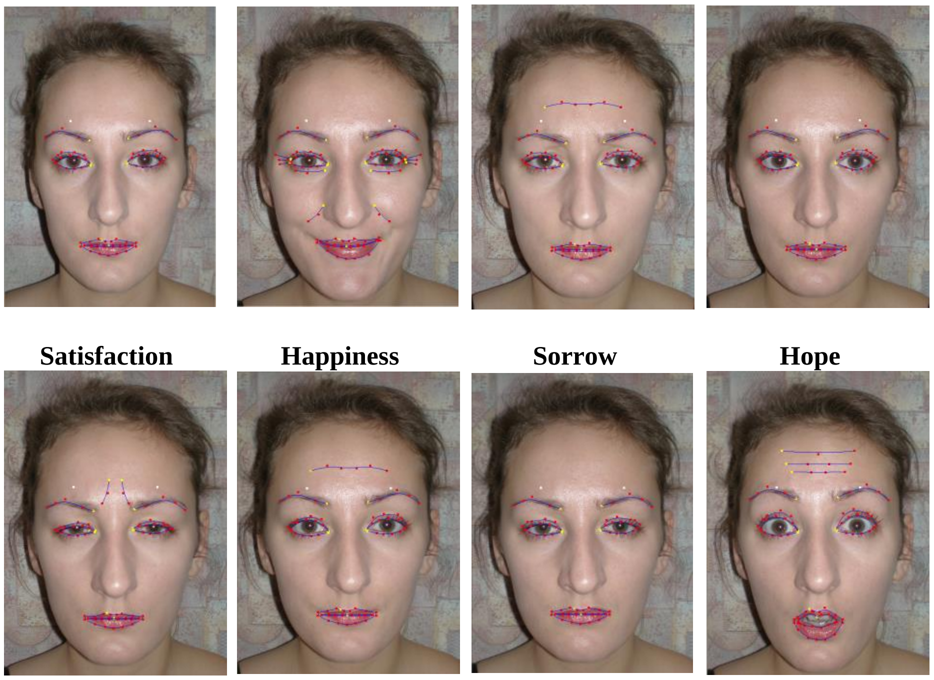


Fig. 3.13. Right eyebrow and a NURBS-curve is proper happiness emotion



Satisfaction **Happiness** **Sorrow** **Hope**

Anger **Interest** **Neglect** **Fear**

Fig. 3.14. Basis emotions in a contour view by NURBS-curves for a concrete actor face presentation

From eight got sets of control points of NURBS-curves (templates) and vector of characteristic mimic features for wrinkles the basis (3.18) of the emotional states (B) of concrete human face was built.

Example Proposed methods using from the photographic image of arbitrary emotion, which reproduces a situation which there feeling of guilt (see Fig. 3.15) appears the proper vector of b was built (3.19).

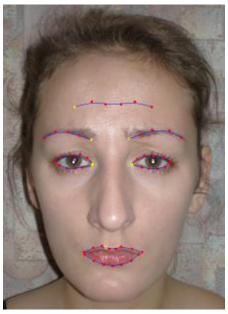


Fig. 3.15. Situation in which arises up emotion of guilt is reproduced

Applying transformation (3.20)-(3.22) vector b is obtained as decomposition on the basis of B.

For the emotional state "guilt" next coefficients of convex combination of the basis states were obtained:

$$\alpha_1 = 0, \alpha_2 = 0.7, \alpha_3 = 0, \alpha_4 = 0, \alpha_5 = 0.3, \alpha_6 = 0, \alpha_7 = 0, \alpha_8 = 0, \sum_{i=1}^{8} \alpha_i = 1 .$$

Here: $\alpha_i, i = \overline{1,8}$ - coefficiets which correspond the next emotional states: happiness, sorrow, hope, fear, satisfaction, anger, interest and neglect. Accordant with [15], state which consists of combinations of satisfaction ($\alpha_5 = 0.3$) and sorrow ($\alpha_2 = 0.7$) emotion "guilt" is corresponds.

3.8. Conclusions

A mathematical model and general information technology for arbitrary emotional state of concrete human face automatic determination as convex combination of some basis states is offered. For this purpose with the help of mathematical model and original software, basis space of emotional states of concrete human face is created. In the future, the arbitrary emotional display of this man is decomposed out as convex combination of the emotional states of this space.

For a basis space emotional states construction the flexible templates of contours of basic areas of human face are utilized. Flexible templates as NURBS-curves are given. Template on the point contour of concrete image settings passes the B-spline of approximation with a help, by the redefined heterogeneous system of linear equations decision.

Technology has a practical value in complex productions, in transport, in the visual checking systems – for the automatic scanning of the emotional state for nonpermanent situations avoidance is offered as example.

3.9. References

1. deBoor C., *A practical guide to splines*, Springer-Verlag, New-York 1978.
2. Cootes T.F., Taylor C.J., *Statistical models of appearance for computer vision*, Technical report, University of Manchester, Manchester M139PT, United Kingdom, September 1999.
3. Darwin Ch., *The expression of the emotions in man and animals,* Sochineniya, Translate by S.L.Sobol, edition by E.N. Pavlovskii, In 8 vol., AN SSSR 1953. T.5 (in Russian).
4. Demidov V.E., *How we see and that we see*, Znanie, 1987 (in Russian).
5. Ekman P., Friesen W. V., *Facial action coding system,* Consulting Psychologists Press Inc., California 1978.
6. Gellershteyn S.G., *Historical significance of Ch. Darwin paper “The Expression of the Emotions in Man and Animals”*, Sochineniya, in 8 vol. M., 1953. T.5 (in Russian).
7. Il'in E.P., *Emotions and sensation,* Spb, Piter, 2001 (in Russian).
8. Izard C.E. *Human emotions*. Plenum Press, New-York 1977, Translated to Russian. M. 1980 (in Russian).
9. Kass M., Witkin A., Terzopoulos D., *Snakes, Active contour models,* “International Journal of Computer Vision”, 1988 vol. 1(4), pp. 321-331.
10. Krak Iu.V., Barmak O.V., Efimov G.M., *Modeling and analysis of human face emotions mimic expressions,* Bulletin of Taras Shevchenko Kiev National University. Cybernetics, 2008 vol. 8, pp. 37-41 (in Ukrainian).
11. Krak Iu.V., Barmak O.V., Efimov G.M., *Information technology of recognition of emotional mimicry on the human face,* “Artificial intelligence”, 2008 vol. 1, pp. 102-109 (in Ukrainian).
12. Krak Iu.V., Barmak O.V., Efimov G.M., *Contour models using for the construction of basis space of emotions mimic expressions*, “Artificial intelligence”, 2007 vol. 4, pp. 288-296 (in Ukrainian).
13. Krak Iu.V., Barmak O.V., Efimov G.M., *Synthesis of mimic expressions of emotions on the basis of formal model*, “Artificial intelligence”, 2007 vol. 2, pp. 22-31 (in Ukrainian).
14. Kryvonos Iu.G., Krak Iu.V., Barmak O.V., *Design and analysis of mimic displays of emotions,* “Reports of National Academy of Science of Ukraine”, 2008 vol. 12, pp. 51-55 (in Ukrainian).
15. Leont'ev V.O., *Emotions classification,* Innovacionno-ipotechnyy center, Odessa 2002 (in Russian).
16. Piegl L., Tiller W., *The NURBS Book*, 2nd Edition, Springer-Verlag, Berlin, 1996.
17. Reykovskiy I., *Experimental psychology of emotions*, 1979 (in Russian).
18. Yuille L., Cohen D., Hallinan P., *Feature Extraction from Faces using deformable templates*, in CVPR, San Diego, 1989.
19. Zhang T.Y., Suen C.Y., *A fast parallel algorithm for thinning digital patterns*, “J. of Commun. ACM”, 1984 vol. 27, no. 3, pp. 236-239.

4. INTELLIGENT NUMERICAL SOFTWARE FOR MIMD-COMPUTER

Alexandr Khimich, Igor Molchanov, Mukhtar Junisbekov, Andrzej Kotyra

4.1. Introduction

For most scientific and engineering problems simulated on computers the solving of problems of the computational mathematics with approximately given initial data constitutes an intermediate or a final stage. Basic problems of the computational mathematics include the investigating and solving of linear algebraic systems, evaluating of eigenvalues and eigenvectors of matrices, the solving of systems of non-linear equations, numerical integration of initial-value problems for systems of ordinary differential equations.

Characteristic feature of mathematical models is a fact that the initial data's specification error should be considered and taken into account along with mathematical equations describing the models and, finally, the reliability of the obtained results should be guaranteed.

A problem of the reliability of computer solutions of mathematical problems possesses two natural aspects: reliability of mathematical models describing the application problem and reliability of the computer solution.

Another, not less important, aspect of practical implementation of the numerical simulation methods is the creation of software at the end user's level – intelligent software providing both communication with computer in terms of the subject area language and automation of all stages in the problem's solving on computer (algorithmization, programming, solving of the problems with approximate initial data together with analysis of the reliability of the obtained computer solutions).

A conception of the intelligent computers intended for the investigating and solving of scientific and engineering problems whose architecture and system software support the intelligent software has been developed at V.M. Glushkov Institute of cybernetics of the National Academy of Sciences of Ukraine. This conception is implemented within the frameworks of Inparcom project jointly performed with State scientific production enterprise "Electronmash".

The lections present results of investigations on the development of parallel algorithms for the solving of basic classes of problems of the computational mathematics: linear algebraic systems, algebraic eigenvalue problem, initial-value problems for systems of ordinary differential equations, non-linear equations and systems as well as software described in monographs [4, 5].

4.2. Composition and engineering characteristics of the intelligent MIMD-computer Inparcom

Intelligent computers Inparcom is a knowledge based computer which in the course of solving of engineering and scientific problems receives information on the characteristics of the problem's computer model and according to these characteristics automatically constructs the solution algorithm, forms a topology based on MIMD-computer's processors and creates a code of the program of parallel computations and, finally, after the completion of the computational process estimates reliability of the obtained results [6, 13].

The intelligent workstation Inparcom includes: host-system and processing unit.

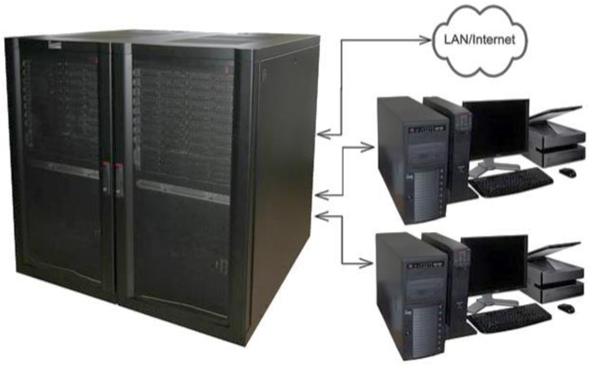


Fig. 4.1. Inparcom system

The host-system carries out the following: control over the utilization of multiprocessor computing resource; all-system monitoring; communication with user's terminal networks; visualization of problem's solving results, realization of that part of computations and data processing which are non- or ill-parallelable.

Processing unit performing the solving of problem with parallel arrangement of computations is a homogeneous scalable structure consisting of a set of high-production processors (with their own operating and disk memories) integrated by network of inter-processor communications.

The Inparcom's software supposes three levels: [8]

1. operating environment supporting the intelligent software;
2. intelligent numerical software intended for investigating and solving problems of the computational mathematics with approximately given initial data;
3. intelligent application software for the classes of applications, for example, for investigating and solving problems on the strength analysis of structures.

The operating environment provides:

- job stacking and run of parallelized program on the computing nodes being chosen;
- monitoring both of the intelligent computer and executable jobs;
- saving and visualization of protocols of parallel computations;
- run of application (executable program code) on the host-computer;
- work via local network and\or Internet (the remote access);
- design of parallel programs;
- management of parts of distributed file system accessible to users.

Intelligent software for each class of problems consists of the following components:

- dialog system,
- library of fundamental modules,
- scheduling/control black, explanatory block.

By means of dialog system the interaction with user is carried out, namely: statement of problem in terms of the subject area language, process of the solving of problem, browsing/analyzing of solution results, delivering of all required information to user, access to glossary of terms for each class of problems, rendering of assistance to user at each workstage.

Functional modules implement both logically computed segments of algorithms and procedures implementing data and information exchanges between processors.

The main purpose of the scheduling control block is to find the most optimal way for the solving formulated problem with information obtained from user and corresponding functional modules.

The explanatory block accumulates information about problem during the computational process for its subsequent output to user. In case in of refusal the solving of problem the user gets detailed explanation of reasons of the refusal as well as recommendations as to the further user's actions.

Engineering characteristics of the intelligent workstations Inparcom are given in Tables 4.1 and 4.2.

Table 4.1. Engineering characteristics of Inparcom intelligent workstations

Model, type and the number of processor cores	Peak productivity, Gflops*	Operating memory, GB**	Disk memory, GB**	Disk storage **	Commu-nication network
INPARCOM32 26 cores	383,04	72	1500	1	Infiniband
INPARCOM64 72 cores	766,08	144	3000	1	Infiniband
INPARCOM128 132 cores	1404, 48	264	5500	4	Infiniband
INPARCOM256 264 cores	2808, 96	526	11000	8	Infiniband

Table 4.2. Details of Inparcom intelligent workstations hardware and software

Hardware			
Title	Computing node	Graphic station	Disk storage
	Computations	Control, input/output of graphic information	Control, storing of data arrays
Processor	2x2xIntel Xeon 53XX	2xIntel Xeon 51XX (53XX)	Intel Xeon 51XX
The number of cores	2x8	4(8)	2
Operating memory	2x16 Gb DDR2-667	16(32) Gbit DDR2-667	2 Gbit DDR2-667
Disk memory	HDD 2x2x250 Gb (2xRAID1)	HDD 2x250Gb (RAID1), FDD, DVD±RW	HDD 10x250Gb (2xRAID1, 8xRAID 0,5,10,50), FDD, DVD-ROM
The number of nodes	Sixteen computing nodes; Two graphic stations (quantity is determined by customer); Two disk storages.		
Computational network	InfiniBand (20 Gbit/s)		
Service network	Gigabit Ethernet, Fast Ethernet (IPMI with KVM)		
System of power supply	IBP 10000VA – 2 items per computing block and IBP 1000VA per each workstation, On-line		
Structure	Computing block – unit 19” / 25U – 2items. Graphic station – system block, monitor, keyboard, mouse, IBP, printer, scanner.		
System software			
Operating system	Linux on the base on Red Hat EL 5, Linux or Windows on the graphic station		
Parallel environment	MPI (OpenFabrics Enterprise Distribution)		
Control system	Program system monitor (management of tasks, monitoring of jobs and hardware of the complex)		
Intelligent software			
Libraries	Libraries of intelligent programs for the solving of problems of the computational mathematics with reliability estimate (Inparlib): Linear algebraic systems; Algebraic eigenvalue problem; Systems of non-linear equations; Systems of ordinary differential equations.		
Interface	Dialog, scheduling and control systems for the solving of problems of the computational mathematics (Inpartool)		
Applications software	Intelligent applications software for investigating and solving of problems on strength analysis of structures (based on NIIASS software Lira 9.4)		

Most problems occurring in engineering and science simulated on computers have as an intermediate of a final stage the solving of problems of the computational mathematics with approximately given initial data. Basic problems of the computational mathematics include: the solving of linear algebraic systems; finding of eigenvalues and eigenvectors of matrices; solving of non-linear algebraic systems; numerical integration of initial-value problems for systems of ordinary differential equations.

It is well known that the efficient solving of mathematical problems with approximately given initial data requires the carrying out the following investigations:

- to reveal the existence of classic or generalized solution;
- to find out an opportunity to determine the unique classic or generalized solution;
- to determine a stability of the solution;
- to find an area within which mathematical solution makes physical sense;
- to estimate an error in the mathematical solution caused by initial data error.

It should be emphasized that due to the initial data error the mathematical problem is to be considered as a problem with a priori unknown characteristics. A machine model of problem to be ultimately implemented on computer is always of the approximate nature with respect to mathematical problem due to the error occurring during input of numerical information about problem into computer.

The error is, in particular, caused by the following:

- a continuum of real numbers in computer is approximated by a finite set of simple fractions (even input of numerical data causes rounding-off errors);
- a phenomenon of “machine zero” gives rise to a number of difficulties during the implementation of computational algorithms (any up-to-date computer possesses the least positive number which can be represented in it; all numbers in modulus less than this number are replaced by zero);
- computer arithmetic operations differ from their mathematical counterparts: associativity and distributivity laws are not valid for any up-to-date computer, while commutativity laws for the floating-point operations are valid only for the correct rounding-off procedure.

So, it is necessary to carry out the computer investigation of mathematical characteristics of computer models of problems, namely:

- to reveal the existence and uniqueness of solution of the problem's computer model;
- to investigate stability of solution within errors in the decimal-to-binary conversion of numbers;

- to determine characteristic features of the computer model of problem for the choice of efficient algorithm for the solving of problem;
- to estimate an inherited error in the mathematical solution;
- to estimate computational error in the obtained solution, i.e. estimate a proximity between the obtained and exact solutions of the computer problem.

To solve problems on MIMD-computers the user is to carry out the following additional work:

- to determine both the optimum number of processors and topology of inter-processor communication for the efficient solving of problem;
- to provide the uniform loading of processors being used for the solving of problem;
- to provide the synchronization of data exchanges between processors;
- to minimize the communicational losses caused by the necessity of inter-processor data exchange.

Such a work requires from users skills in parallel programming, knowledge of mathematical and engineering characteristic features of MIMD-computer, studying of a great deal of the operation instructions for packages and libraries implementing parallel algorithms of programs.

4.3. Composition and architecture of intelligent numerical software

Difficulties occurring during the computer solving of problems of the computational mathematics on MIMD-computers can be overcome by means of the intelligent numerical software: program tools Inpartool and library of intelligent programs Inparlib for the investigating and solving of basic classes of problems occurring in the computational mathematics [5].

Inpartool consists of separate components for investigating and solving problems from the following classes:

- linear algebraic systems;
- algebraic eigenvalue problem;
- non-linear equations and systems;
- ordinary differential equations and systems.

At the level of concepts Inpartool implements the end user's model and represents a set of program and engineering tools providing the investigating and solving of user's problems belonging to the field of numerical methods.

For the linear algebraic systems Inpartool solves the problems with various structure matrices together with reliability elements for the solution, invert matrices, evaluates singular values and matrix ranks, estimates the matrix condition numbers, etc.

For the algebraic eigenvalue problem (standard and generalized) Inpartool solves the both partial and full eigenvalue problems with various structure matrices (general, band or sparse). Inpartool enables to evaluate condition number for the separately taken eigenvalues, condition numbers of eigenvectors, to estimate computational and overall errors in solutions.

For system of non-linear and transcendental equations Inpartool evaluates: the local condition number of the function f(x), the local condition number of the vector-function F(x), termination criteria for the iterative processes, the accuracy of solution with taking into account the approximate nature of the initial data.

For the investigating and solving of initial-value problems for systems of ordinary differential equations Inpartool enables to integrate both common and stiff systems of equations with accuracy of various orders, including any a priori specified accuracy. At user's will Inpartool can carry out the investigation of the stiffness for the systems of ordinary differential equations, the evaluation of Lipschitz constant for them and determination of the accuracy of solution with taking into account approximate nature of the initial data.

At the functional level Inpartool is the software enabling to formulate a problem with approximately given initial data for computer in terms of the subject area language; automatically investigate mathematical characteristics of the problem's computer model; according to the revealed characteristics of the problem construct a solution algorithm; the automatically determine the optimum number of processors and form an efficient topology of the MIMD-computer; distribute the initial data between processors; synthesize a parallel program for solving the problem with taking into account mathematical and engineering characteristics of the computer; solve the problem together with reliability estimates of the solution; explain and visualize the obtained results in terms of the subject area language.

Inpartool implements a conception of knowledge [10]. Its design is based on the synthesis of fundamental achievements in the field of module programming, knowledge bases and databases; it relies on the data processing methods being developed: representation, storage and obtaining of new knowledge, etc.

A subject area for each class of problems involves a wide spectrum of problems, methods, algorithms and computational schemes taking into account approximate nature of the initial data. Special computer methods for investigating mathematical characteristics of their computer models are implemented together with algorithms for the analysis of the obtained computer results. Modular programming principle [9] made it possible to systematize and unify knowledge about subject areas and design special methods of the same type for storing, search, extraction and pressing of data. This made it possible to determine an optimum set of procedures and functions by means of which all problems can be solved. Procedural knowledge is represented by functional modules describing logically completed segments of computer algorithms for the investigating and solving of problems as well as semantics of these algorithms. Each module contains knowledge about its employment, input and output parameters, rules for the initial data distribution between processors, allowable computer topology, required computing resources, etc.

A client-server architecture of Inpartool is represented schematically in fig. 4.2, with the client part consisting only of the dialog system and the server part including systems providing user's access to Inpartool in Internet as well as systems by means of which the investigating and solving of problems with approximately given initial data on parallel computer is implemented.

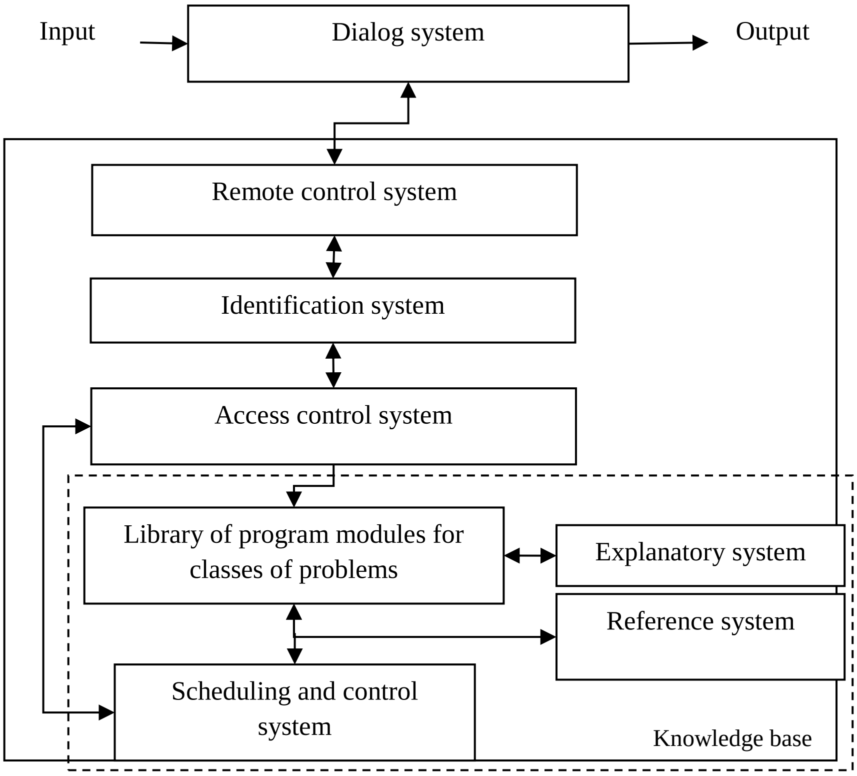


Fig. 4.2. Client-server architecture of Inpartool

Library of program modules enables to automatically construct the required algorithm and synthesize a program for solving the problem from separate functional modules on the basis of revealed problem's characteristics and with efficient employment of MIMD-computer's computing resources. Communication between modules is established both by data and control.

Scheduling and control system is closely associated with formal description of the subject area, knowledge base and dialog system. The principal purpose of the scheduling of computations is to find an optimum way for the solving of the problem.

Principles of the automatic investigating and solving of problems on computer with automatic analysis of the reliability of results impose the following requirements upon the scheduling and control system:

- analysis of user's initial data and their transformation into primary knowledge about the problem;
- possibility of storing and processing knowledge from the subject area during the scheduling of computations;
- arrangement of various ways of employing knowledge both about the problem and subject area for investigating and solving of the problem;
- construction of algorithms and synthesis of programs for investigating and solving problems;
- output and saving of results of investigating and solving the problem for their subsequent clarification and visualization.

In order to implement the solving of problem on the MIMD-computer the system should carry out the following control functions:

- construction of MIMD-computer's virtual topology;
- determination of the number of processors providing efficient solving of the problem;
- distribution of the initial data between processors.

Explanatory system answers the following questions: how was the solution obtained? Why was such a way of investigating the problem's characteristics chosen? It either yields the obtained solution together with reliability estimates or explains reasons of refusal in the producing of the solution. User can manage an extent of working out explanations in detail.

Toward this end various scenarios of explanations, various-level protocols of the computational process and graphic examples for the user have been developed.

Reference system allows the user to get information necessary for the solving of his problem by means of Inpartool: functional potentialities, order of work, input of the initial data, glossary of terms from the subject area being used, etc.

The interaction with user is implemented by means of dialog tools, namely:

- formulating of problem and input of the initial data;
- communication during the process of computations;
- visualization of the obtained results;
- access to explanatory block;
- the obtaining of information-reference data and help at each stage of work.

Dialog scenarios are developed with taking into account a model of the subject area as well as various purposes and level of user’s preparedness to the using of Inpartool. As this takes place, the following requirements are satisfied: communication in terms of the subject area language, suitable forms of information input/output, paper-free form of documentation.

The intelligent interface enables the end user only to formulate the original problem, while a sequence of operations required for the obtaining of the problem’s solution is automatically determined by the software itself by means of including a set of operations carried out by user into the sequence being generated. Forms of communication being used are the following: menu, answer/question, screen forms.

An order of Inpartool’s communication with user is established by main menu. Its structure and basic items are natural and habitual for user since they are inherent in many dialog systems. Various menu selection schemes enable to determine a problem, indicate input (display, data archive) and output (disk, printer) destinations, run the problem, look over glossary of the subject area, etc. In addition, the following operations are provided: browsing, correction, copying and saving of the input and output data and their using in current and subsequent work sessions.

During the input of initial data the user either fills out window forms by means of prompting and instructions or answers the Inpartool’s questions.

The solving of problem can be implemented either automatically when investigating and solving of the problem are carried out without user’s involvement or interactively when user’s participation is possible in all or separate stages of investigating and solving of the problem.

Purpose and composition of the Inparlib

Intelligent programs involved in the library [1] are intended for the investigating and solving of basic problems of the computational mathematics:

- linear algebraic systems;
- algebraic eigenvalue problem;
- non-linear equations and systems;
- systems of ordinary differential equations.

Programs included in the library implement:

- statement of problems with approximately given initial data;
- investigation of characteristics of problem's computer model;
- verification of agreement between characteristics of problem's computer model revealed by computer and chosen solution algorithm;
- construction of topology of Inparcom's processors;
- the obtaining of solution together with reliability estimate which includes both estimate for the inherited error caused by the initial data error and estimate for the computational error.

Program modules implementing finished parts of investigating and solution algorithm are written in C and intended both for the MIMD-architecture computers and parallel programming environment MPI.

As to linear algebraic systems (LAS) program modules included in Inparlib enable to: investigate and solve problems with various structure matrices together with reliability estimates for the solution, invert matrices, evaluate both singular values and matrix ranks well as estimate matrix condition numbers.

As algebraic eigenvalue problems (common and generalized) Inparlib's programs solve both full and partial eigenvalue problem with various structure matrices (dense, band or sparse). By means of programs from Inparlib it is possible to evaluate condition numbers for separately taken eigenvalues, condition numbers for eigenvectors as well as to evaluate estimates for the overall error in solutions.

As to non-linear equations and systems Inparlib's programs enable to: investigate and solve systems of non-linear algebraic and transcendental equations; determine local condition number of the function f(x), local condition number of the vector-function F(x); implement termination criteria for iterative processes guaranteeing the obtaining both of solutions within the given accuracy and solution's errors with taking into account approximate nature of the initial data.

As to systems of ordinary differential equations with initial conditions, Inparlib contains programs enabling to: investigate and solve these systems, integrate both common and stiff systems of equations within accuracy of various orders as well as within any a priori specified accuracy. A user can carry out investigation of the stiffness of SODE, evaluate both the Lipschitz constant and accuracy of the obtained solution with taking into account approximate nature of the initial data.

Functional programs from Inparlib provide: statement of problems with approximately given initial data, investigation of mathematical characteristics of problem's machine models, verification of agreement between the revealed characteristics and application area for the solution algorithm being chosen as well the obtaining of solution together with reliability estimate or a refusal (with indication of reasons) in the solving of problem.

From the end user's point of view programs included in the library are reuse components in the solving of application problems for which problems of the computational mathematics are either intermediate or a final stage.

4.4. Investigating and solving of linear algebraic systems

4.4.1. Functional potentialities of Inpartool on investigating and solving of linear algebraic systems

Linear algebraic systems (LAS) can arise: in data processing problems where linear differential problems are discretized by finite differences or finite elements method; in the solving of linear problems by least squares method; in calculating electric circuits and complicated hydraulic systems, in some models of economic problems and so on.

As this takes place, consider what kinds of problems can be formulated. Thus, in a number of cases it is required to solve LAS with non-singular square n-th order matrix with one vector of free terms (with one right-hand side) or the same system with p right-hand sides In some problems the necessity arises in the evaluation of matrix inverse to the given non-singular matrix of order n. There exist problems where for the given m × n matrix A and vector b consisting of m components it is necessary to evaluate such a vector-column x consisting of n components that the Euclidean norm $\|Ax-b\|$ be the least. Such a vector x is called a solution obtained by least squares method or a generalized solution to the system Ax=b (possibly non-consistent system). If rank of the given system r(n)≠n than there exists an infinite set of vectors x being solutions obtained by least squares method (generalized solutions to LAS). Sometimes it is required to find among such solutions the solution x which possesses the least Euclidean norm $\|x\|$. This vector is always unique and referred to as a normal generalized solution.

As a rule, the solving of application problem starts from the creation of acceptable physical and mathematical models. Various hypothesizes are used for the construction of these models. If these hypotheses are valid (error in hypothesis is absent or sufficiently small) the physical model correctly reflects regularities inherent in application problem. The physical model can be described by mathematical formulas, for example, by some LAS.

Systems of the form:

$$\tilde{A}\tilde{x} = \tilde{b} \qquad (4.1)$$

with accurate initial data are very seldom used in the describing of physical models. The most typical initial data specification has the for

$$Ax = b \qquad (4.2)$$

with indicating error in the initial data:

$$\|\tilde{A} - A\| \equiv \|\Delta A\|, \qquad \|\tilde{b} - b\| \equiv \|\Delta b\|. \qquad (4.3)$$

Thus, a physical model is described by the entire class of equations. As a formal solution to the problem (4.1)–(4.3) one can take any vector which turns equation (4.3) into identity. Note that in the case of rectangular ($m \neq n$) or singular (det$A$=0) matrix of accurate system (4.1) the approximate system (4.2) obtained in computer may turn out to be non-consistent for any accuracy of the initial data specification.

An error in the solution x caused by inaccurate specification of the initial data is said to be inherited error. Its value depends both on the initial data error and characteristics of the matrix.

A solution to the system of equations (4.2) obtained by some numerical method on computer is called a machine solution of the problem. Because of the error in decimal-to-binary conversion of the initial data, the method error as well as error in the computer implementation of algorithm the obtained machine solution of the problem may differ from the mathematical one.

Thus, in the solving of LAS describing application problems it is necessary to determine a concept of solution to be sought, construct an algorithm for finding this solution, estimate the computer implementation error in the course of solving the problem (i.e. estimate proximity between machine and mathematical solutions) as well as the estimated inherited error in the solution [3,7].

As to the class «Linear algebraic systems» Inpartool involves the solving of the following problems:

- investigation and solving of LAS together with reliability estimates for the obtained results;
- inversion/pseudo-inversion of matrix together with reliability estimates for the obtained results;
- evaluation of estimate for the matrix condition number;
- evaluation of determinant of the matrix;
- evaluation of singular values of the matrix;
- evaluation of matrix rank;
- evaluation of fundamental system of solutions to homogeneous system.

These problems are solved for the following types of matrices:

- dense nonsingular;
- dense symmetric positive definite;
- band symmetric positive semi-definite;
- band symmetric positive definite;
- square singular of arbitrary rank;
- rectangular.

Problems under consideration are covered by small set of solution algorithms but their various modifications take into account all problems and types of matrices. An important requirement is imposed on the set of algorithms intended for the solving of problems with approximately given initial data – to be in accordance with mathematical and engineering characteristic features of computer. During the development of algorithms and programs the questions were investigated related to dependence of problem’s solving time and reliability of results on the following: arrangement of computations, architecture and topology of computer, system software, translator, styles of programming and so on [4, 5].

For the investigating and solving of LAS with dense and band nonsingular matrices various modifications of Gauss algorithm are used. Various modifications of the Cholesky algorithm are employed for the investigating and solving of LAS with dense and band sym-metric positive definite matrices. The least squares method based on the SVD – decomposition of the matrix employed for the solving of LAS with square singular and rectangular matrices of the arbitrary rank.

At the stage of computer solving of LAS with nonsingular matri-ces with approximately given initial data Inpartool provides automatic carrying out of the following:

- investigation of singularity of the matrix within the limits of machine accuracy and within the limits of the initial data error;
- investigation of conditioning of the matrix;
- the solving problem by algorithm corresponding to the revealed characteristics of the problem;
- estimating of the inherited error in the mathematical solution;
- estimating of the proximity between machine and mathematical solutions.

It is common knowledge that basic criterion for the determining of the above-mentioned characteristics of LAS is the condition number $H = \|A\|\|A^{-1}\|$. If H is not large the system's matrix A is called an ill-conditioned or singular within the range of the initial data error. However, the practical evaluation of H in the computer involves the evaluation of the inverse matrix A^{-1} that requires more calculations.

To economize the amount of computations and minimize losses in accuracy one should evaluate an estimate (cond A) instead of evaluating the condition number of the matrix and, in so doing, one should make use of the decomposition of the original matrix A by one of direct methods.

The evaluation of the matrix condition number is implemented by scheme:

$$A \cong L\,U, \qquad \|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}|,$$

$$Uw=e, \qquad L^T y=w, \qquad L\text{v}= y, \qquad Uz=v,$$

$$\operatorname{cond} A = \|A\|_1 \|z\|_1 / \|y\|_1,$$

where $\|z\|_1 = \sum_{i=1}^{n} |z_i|, \qquad \|y\|_1 = \sum_{i=1}^{n} |y_i|.$

If the value cond A in computer satisfies the condition

$$1.0 + 1.0/\text{cond } A = 1.0, \tag{4.4}$$

the matrix is considered to be singular within the limits of machine accuracy. In this case a stable projection of the solution can be found.

If matrix cannot be classified as singular according to (4.4), but $\varepsilon_A \times \text{cond A} \geq 1$, then LAS with approximately given initial data entered to computer is singular within the limits of accuracy of the matrix elements' specification, therefore the reliability of computed solution cannot be guaranteed. Here $\|\Delta A\| / \|A\| \leq \varepsilon_A$ is the maximal relative error in matrix elements. If a user considers the initial data to be given accurately then ε_A is assigned the value *macheps* – the least floating-point number such that condition:

$$1 + macheps > 1$$

is hold in computer.

If the system of equations possesses either a rectangular $m \times n$-matrix or a square singular matrix Inpartool guarantees the obtaining of the generalized solution of the system, i.e. determines a vector minimizing $\|Ax - b\|_E$ over the entire space R^n. A system can possess a set of such solutions. Then a normal solution can be evaluated, i.e. a vector possessing the minimal norm $\|x\|_E$.

In this case the spectral condition number of the matrix is evaluated by means of the singular value decomposition of the matrix:

$$\text{cond}_s A = \|A\| \|A^+\| = \frac{\sigma_1}{\sigma_2},$$

where σ_1 is the largest singular value and σ_2 is the least non-zero singular value.

During the computational process an analysis of the reliability of obtained results is carried out which includes estimating the proximity between machine and mathematical solutions as well as estimating of the inherited error.

The upper bound for the relative inherited error in the solution is determined by formula:

$$\frac{\|x - \tilde{x}\|}{\|\tilde{x}\|} \leq \text{cond} A \times \frac{\varepsilon_A + \varepsilon_b}{1 - \varepsilon_b}$$

where $\tilde{x}$ is the exact solution of the system with accurately given initial data; x is exact solution of the system with approximately given initial data; ε_A, ε_b are maximal relative errors in elements of the matrix and right-hand sides, respectively.

An estimate of the computational error in the solution characterizes the proximity between machine and mathematical solutions. For its evaluation one should employ one step of the solution's iterative refinement procedure.

Let us briefly outline the iterative refinement algorithm for the solving of system with non-singular matrix.

Let x be a solution to the system

$$Ax = b$$

evaluated by some direct method.

The iterative refinement is implemented by scheme

$$\begin{gathered} x^{(0)} = x, \\ r^{(s)} = b - Ax^{(s)}, \\ A \times \Delta x^{(s)} = r^{(s)}, \\ x^{(s+1)} = x^{(s)} + \Delta \mathrm{x}^{(s)}, \\ \mathrm{s} = 0, 1, 2 \ldots \end{gathered}$$

During the evaluation of $\Delta x^{(s)}$ the matrix decomposition is used already obtained by one of algorithms, therefore the iterative refinement procedure doesn't require a lot of extra time. The evaluation of the residual $r_i^{(s)}$ should be carried out with the increased machine word length.

Within Inpartool an estimate for the solution's computational error is determined as follows:

$$Ecomp < \frac{\|\Delta x_1\|}{\|x_2\|}$$

where x_2 is an approximation to the exact solution obtained by one step of the iterative refinement.

Let us outline fundamental conceptual theses of the technological scheme for the solving of LAS by Inpartool:

- possibility of solving problems with approximately given initial data;
- formulating of problems in terms of the subject area language;
- suitable for user forms of the initial data's input;
- automation of the following processes: the computer inve-stigation of mathematical characteristics of the problem's computer model, choice of algorithm and synthesis of program for the solving of problem;
- the solving of problem together with reliability estimates of the obtained computer solutions;
- the obtaining not only of solution to the problem but also a protocol describing the solving of problem together with analysis of its revealed characteristics and reliability of the obtained results;
- implementation of the "hidden parallelism" principle.

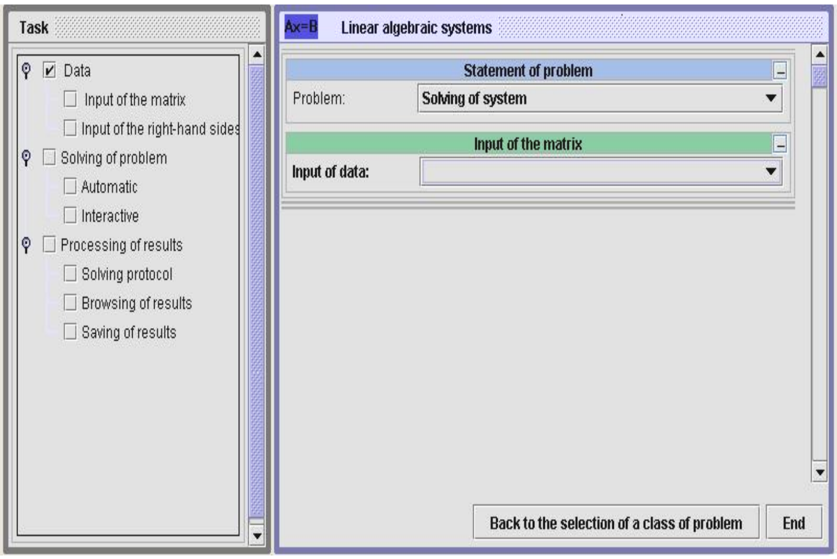
The implementation of the “hidden parallelism” principle involves the following: paralleling of algorithms for the investigating and solving of problems; a choice of the optimum number of processors for the efficient solving of the problem; creation of the computer topology and distribution of the initial data between processors according to the requirements of algorithms; arrangement of data exchanges between processors.

When solving LAS by Inpartool a user takes part only in the formulation of problem while the rest of work stages in the solving of problem are performed automatically.

4.4.2. Technology for investigating and solving of linear algebraic systems

4.4.2.1 Applying to Inpartool for the solving of LAS

The general form of main window **«Linear algebraic systems»** consisting of main menu and two panels is shown in fig. 4.3. The left panel (passive) reflects a sequence of work stages and sub-stages which were already performed, being performed or will be performed.



Problem: Solving of system

Input of data:

Fig. 4.3. The Linear algebraic systems window

Inpartool solves LAS for such matrices:

- dense nonsingular;
- dense symmetric positive definite;
- band symmetric positive semi-definite;
- band symmetric positive definite;
- square singular of arbitrary rank;
- rectangular.

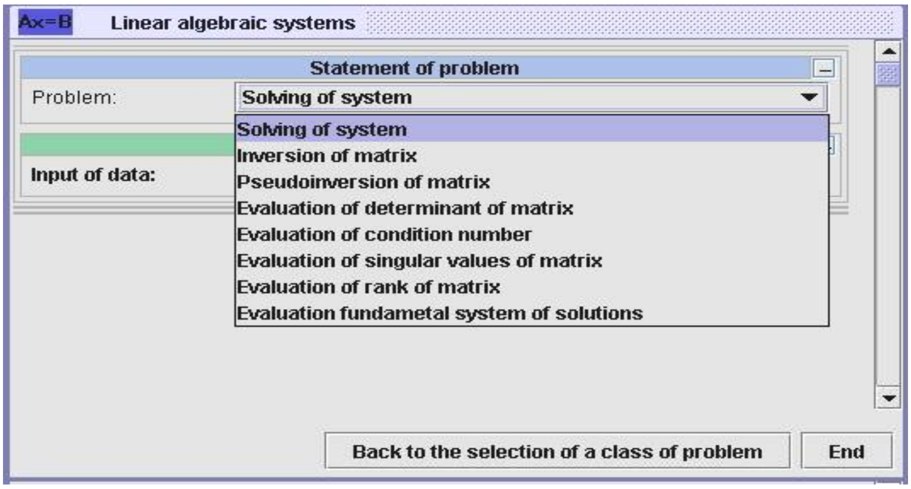
LAS can be solved both for one and many right-hand sides.

To solve the problem a user should carry out the following successive stages of work in the right-hand (active) panel:

- formulate a problem;
- input the problem’s initial data;
- start the problem;
- obtain results.

To formulate a problem the user should click on arrow located to the right of the title «**Problem**». The submenu will appear containing a list of problems from this class of problems which can be solved by means of Inpartool**.**

Now the dialog window has a form shown in fig. 4.4. From the list being proposed user should select a problem to be solved, for example, **«The solving of LAS»**.

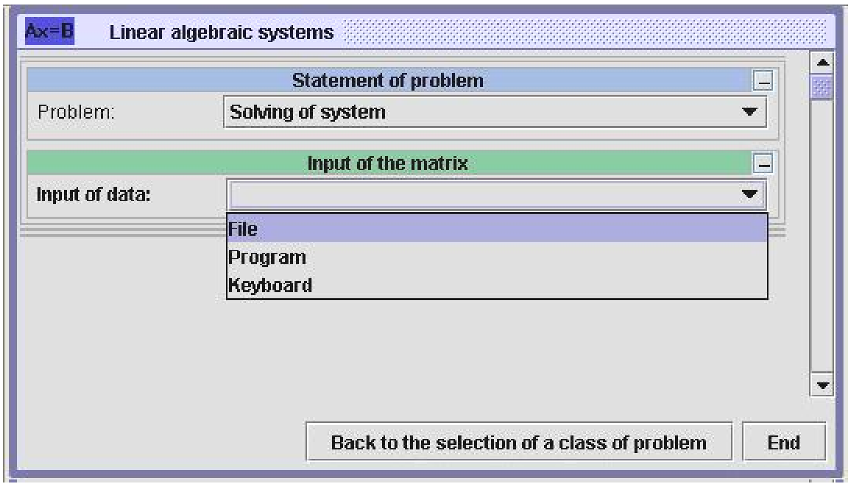


Problem: Solving of system

Ax=B
Linear algebraic systems
Statement of problem
Input of data:
Solving of system
Inversion of matrix
Pseudoinversion of matrix
Evaluation of determinant of matrix
Evaluation of condition number
Evaluation of singular values of matrix
Evaluation of rank of matrix
Evaluation fundametal system of solutions
Back to the selection of a class of problem
End

Fig. 4.4. Selection of problem type list

4.4.2.2 Specification of initial data for the solving of LAS



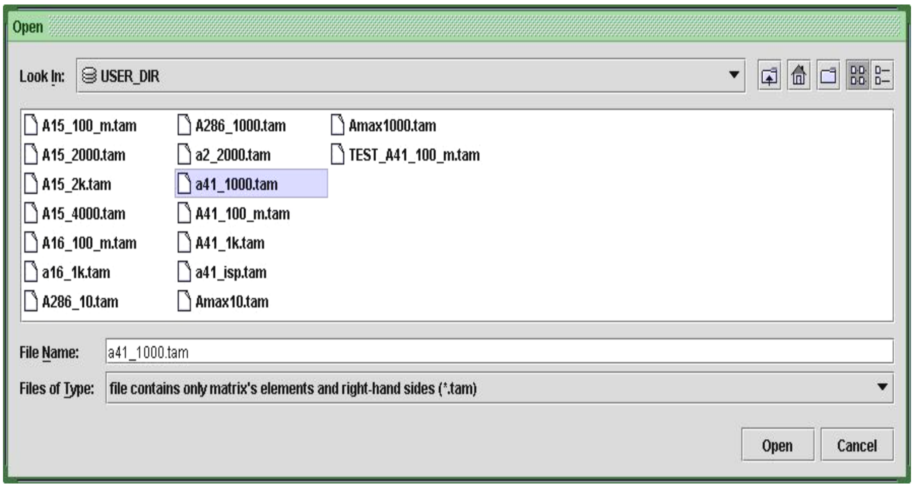
Problem: Solving of system

Input of data:

Fig. 4.5. Input data source selection list

Initial data for the solving of LAS are given by parameters of the problem, i.e. elements of the matrix (the number of rows and columns in the matrix, the number of diagonals for band matrices as well as the number of right-hand sides), matrix elements and their maximum relative errors. The data can be input from the binary file and/or their values can be directly entered into corresponding data fields. This input can also be implemented by program or by formulas (fig. 4.5).

During the data input from file a user can make use of convenient formats provided by Inpartool (fig. 4.6).



Look In: USER_DIR

File Name: a41_1000.tam

Files of Type: file contains only matrix's elements and right-hand sides (*.tam)

Fig. 4.6. Open file window

Among them are the following data formats:

- file contains in the binary form matrix elements and elements of right-hand sides as floating-point numbers the sequence of which is determined by form of the matrix (the file mask is `*.tam`), each number occupying 8 bytes;
- file contains in the binary form only matrix elements as floating point numbers the sequence of which is determined by form of the matrix (the file mask is `*.tam`), each number occupying 8 bytes;
- file contains in the binary form only elements of right-hand side as floating point numbers the sequence of which is determined by form of the matrix (the file mask is `*.tam`), each number occupying 8 bytes;
- file contains in the binary form the number of rows and columns for dense matrices or order of the matrix and the number of diagonals for band matrices, the number of right-hand sides, matrix elements and elements of right-hand sides (the file mask is `*.dat`), each number occupying 8 bytes;
- file contains in the binary form all information about problem: form and structure of the matrix, the number of rows and columns for dense matrices of order of the matrix and the number of diagonals for band matrices, the number of right-hand sides, matrix elements and elements of right-hand sides (the file mask is `*.edat).`

Table 4.3 contains an order in which information about the problem being solved is to be written in file as well as values of parameters to be used during the creation of the initial data file possessing mask`*.edat`.

Table 4.3. `*.edat` file structure

Contents of file	Type	Bytes
Format version (=1)	Integer	4
Matrix structure and type code(= 17)	" – "	4

Code of order in which matrix elements and elements of right-hand sides are written:
0 – by lower diagonals;
1 – by rows;
2 – by columns;
	" – "	4

The number of rows in matrix (for band matrix – order of the matrix)	" – "	4
The number of columns (for band symmetric matrix – half-width of band excluding main diagonal)	" – "	4
The number of right-hand sides in LAS	" – "	4
Relative error in matrix elements	Floating-point number	8
Relative error in elements of right-hand sides	" – "	8
Elements both of matrix and right-hand sides (in the form of sequence of numbers)	" – "	$8\times l_A$ $8\times l_b$

To encode the type of matrix the following formula is used

$$16\, i_1 + i_2,$$

where i_1 is a matrix structure code: $i_1 = 0$ for dense matrix and $i_1 = 1$ for band matrix; i_2 is a matrix code $i_2 = 0$ for general matrix and $i_2 = 1$ for symmetric matrix.

An order in which elements of dense (general and symmetric) and band matrices are entered is given below.

For the dense matrix

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}$$

its elements are entered in the following order:

$$a_{11}\, a_{12} \dots a_{1n}\, a_{21}\, a_{22}\, a_{2n} \dots a_{n1}\, a_{n2} \dots a_{nn}.$$

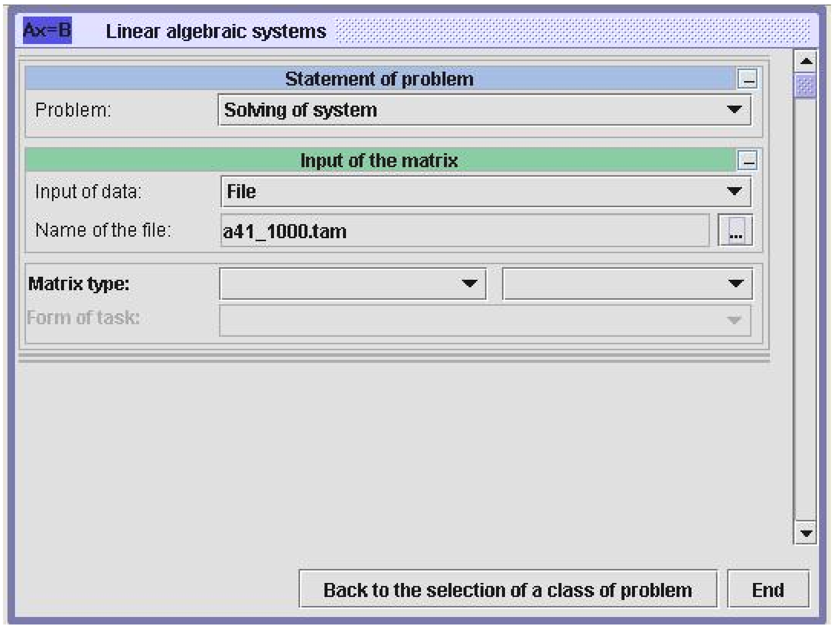
Elements of the band symmetric matrix

$$A_1 = \begin{pmatrix} a_{11} & a_{21} & a_{31} & 0 & 0 & \dots \\ a_{21} & a_{22} & a_{32} & a_{42} & 0 & \dots \\ a_{31} & a_{32} & a_{33} & a_{42} & 0 & \dots \\ 0 & a_{42} & a_{43} & a_{44} & a_{54} & \dots \\ 0 & 0 & a_{53} & a_{54} & a_{55} & \dots \end{pmatrix}$$

are entered in the following order:

$$a_{11}\, a_{22}\, a_{21} a_{33}\, a_{32} a_{31} a_{44}\, a_{43}\, a_{42}\, a_{55} a_{54}\, a_{53} \dots$$

Consider the case where elements both of the symmetric matrix and one right-hand side for the 1000-*th* order LAS are written in succession in binary form in file `A41_1000.tam`. In the window «**Matrix specification**» the user should indicate "File" for data input and enter the file name (fig. 4.7). In the window «**Matrix form**» user should indicate type and structure of the matrix by selecting «**dense, symmetric**» (fig. 4.8).



Linear algebraic systems

Statement of problem

Problem: Solving of system

Input of the matrix

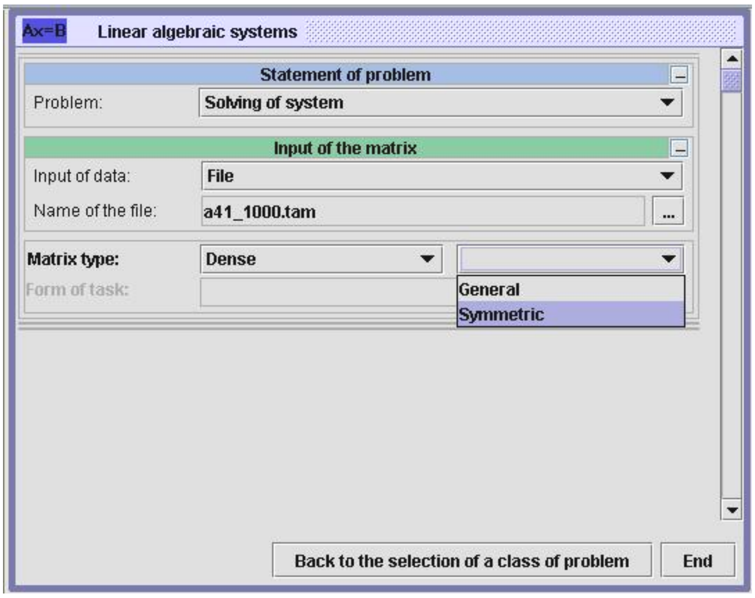
Input of data: File

Name of the file: a41_1000.tam

Matrix type:

Form of task:

Fig. 4.7. Matrix form selection



Linear algebraic systems

Statement of problem

Problem: Solving of system

Input of the matrix

Input of data: File

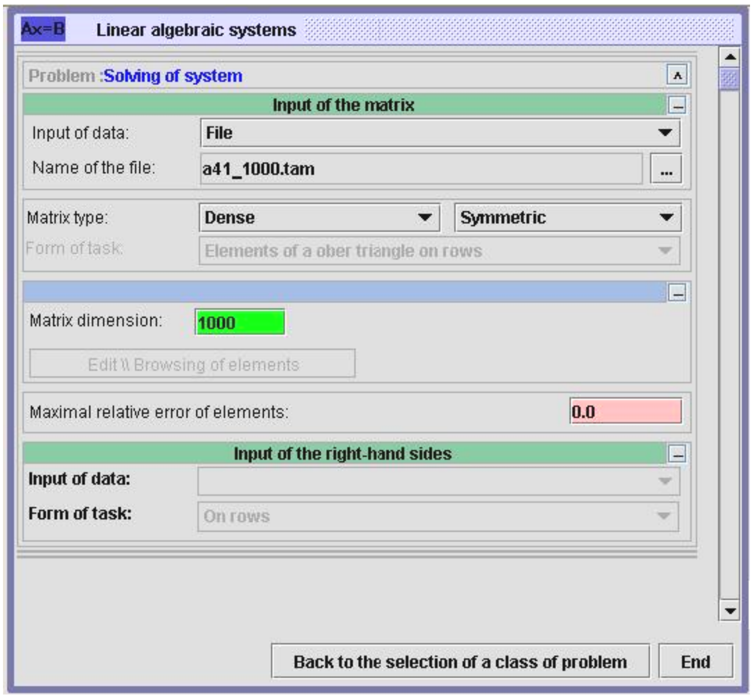
Name of the file: a41_1000.tam

Matrix type: Dense

Form of task:

Fig. 4.8. Matrix type selection

Order of the matrix and maximum relative error in matrix elements can be entered by means of the keyboard (fig. 4.9).


Ax=B
Linear algebraic systems

Problem :Solving of system

Input of the matrix

Input of data: File

Name of the file: a41_1000.tam

Matrix type: Dense Symmetric

Form of task: Elements of a ober triangle on rows

Matrix dimension: 1000

Maximal relative error of elements: 0.0

Input of the right-hand sides

Input of data:

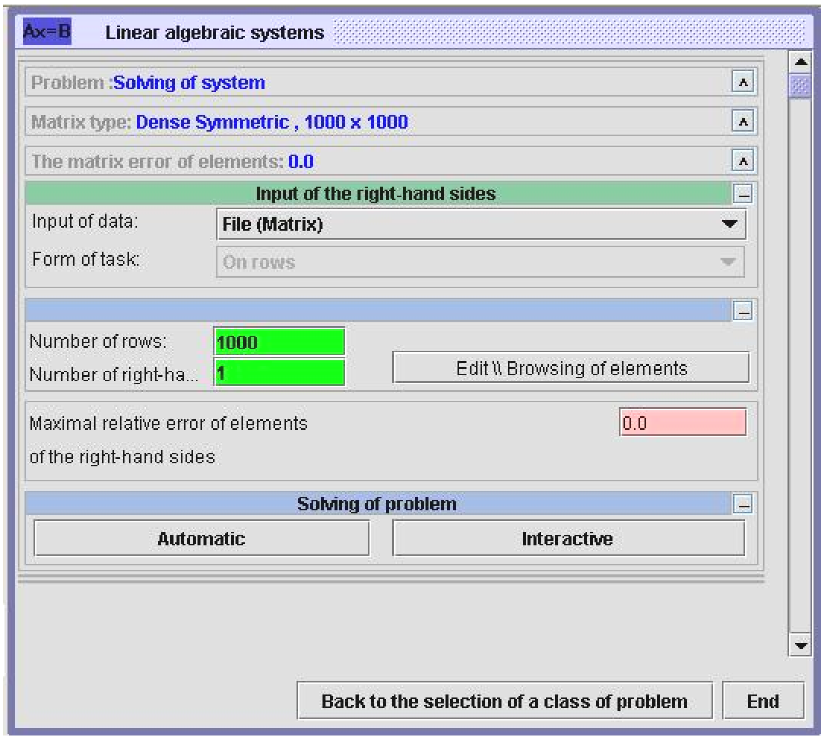
Form of task: On rows

...
Edit \\ Browsing of elements
Back to the selection of a class of problem
End

Fig. 4.9. Matrix dimension selection

If the user considers the initial data to be given accurately he enters a value of the maximum relative error equal to 0.0 (0.0 by de-fault). Already entered values are colored in green, while data to be entered are red.

Information about right-hand sides (fig. 4.10) is entered in the same manner. Elements of the right-hand sides vector are to be written in file which already contains matrix elements, that’s why the user should choose the item **«File, from matrix file»** in the window **«Right-hand sides specification»**. Parameters and maximum relative error in right-hand sides (0.0 by default) are also entered by means of the keyboard.



Ax=B Linear algebraic systems

Problem :Solving of system

Matrix type: Dense Symmetric , 1000 x 1000

The matrix error of elements: 0.0

Input of the right-hand sides

Input of data: File (Matrix)

Form of task: On rows

Number of rows: 1000

Number of right-ha... 1

Edit \\ Browsing of elements

Maximal relative error of elements of the right-hand sides: 0.0

Solving of problem

Automatic

Interactive

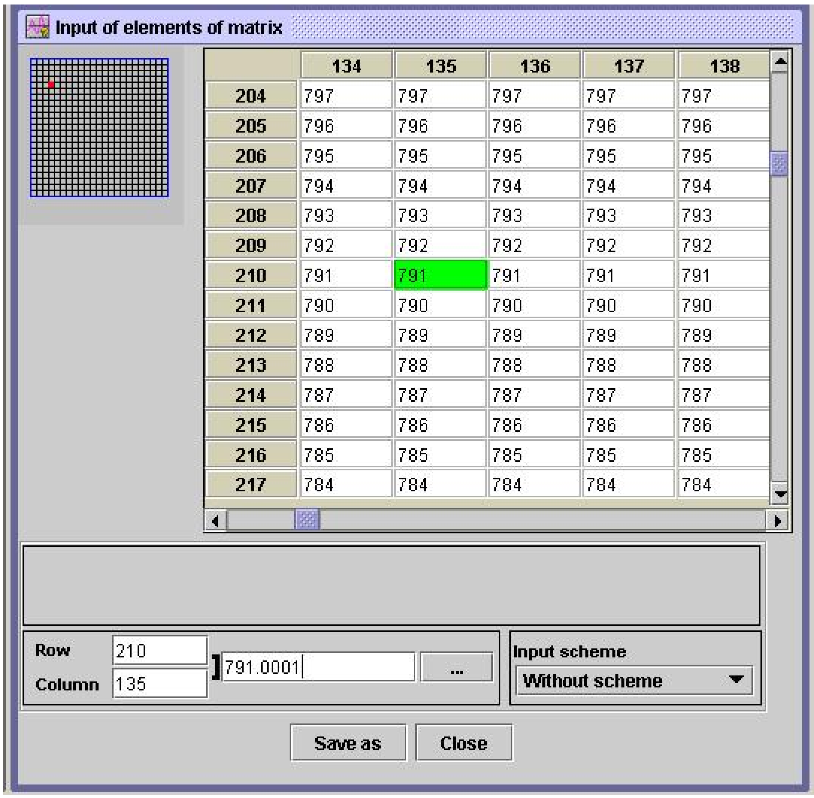
Back to the selection of a class of problem

End

Fig. 4.10. Right-hand sides specification window

Input both of problem’s parameters and maximum relative errors in elements finishes by pressing the **<Enter>** button. In so doing the information is entered and passage to the next data input window takes place.

Elements of matrices of LAS can be edited by pressing Edit \\ Browsing of elements button. In the appeared dialog window (fig. 4.11) the location of marked by user element to be edited is schematically reflected in the left upper corner of the right panel. A red slider can be moved in order to mark matrix segment containing element to be edited. A table to the right represents the matrix segment where the editing is taking place. At the bottom of the panel one can see the numbers of row and column at the crossing of which an element to be edited in separate cell is located. After its editing the corrected matrix is either automatically updated in the old file or saved in another file (when pressing «Save as»). Having pressed «Close» the work of Inpartool can be continued.



Input of elements of matrix

	134	135	136	137	138
204	797	797	797	797	797
205	796	796	796	796	796
206	795	795	795	795	795
207	794	794	794	794	794
208	793	793	793	793	793
209	792	792	792	792	792
210	791	791	791	791	791
211	790	790	790	790	790
212	789	789	789	789	789
213	788	788	788	788	788
214	787	787	787	787	787
215	786	786	786	786	786
216	785	785	785	785	785
217	784	784	784	784	784

Row: 210
Column: 135
791.0001

Input scheme: Without scheme

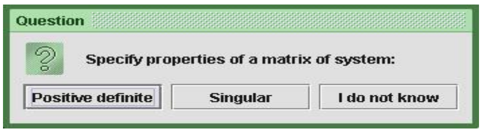
Fig. 4.11. Matrix edition dialog window

4.4.2.3 *The solving of LAS*

Inpartool proposes two ways for the solving of LAS: automatic and interactive. To run the problem one should choose a way for solving the problem in window which will appear after successful input of the data (fig. 4.10).

During the automatic solving of the problem it is investigated. On the basis both of the initial data investigation and characteristics of LAS revealed by computer as well as according to engineering and mathematical potentialities of Inparcom-16 an algorithm for solving the problem is chosen, an efficient topology from the number of pro-cessors optimum for this problem is constructed, initial data are dis-tributed between processors according to the chosen algorithm, the problem is solved and reliability of the obtained results is analyzed.

During the interactive solving of the problem some its characteristics known to user can be indicated, for example, determinacy or singularity of the matrix (fig. 4.12).



Specify properties of a matrix of system:

Fig. 4.12. Matrix known characteristics dialog

Inpartool will construct an algorithm and solution program with taking into account information about problem's characteristics obtained from user and distribute matrix elements between processors. If user was mistaken in the determining of problem's characteristics he will be informed about this by Inpartool and will be proposed to continue the investigating and solving of the problem with taking into account characteristics revealed by computer. The problem will be solved together with reliability and analysis of the obtained results.

During the process of solving the problem a window will appear showing a progress of performing the task (fig. 4.13) which will be closed after completion of the solving of problem.

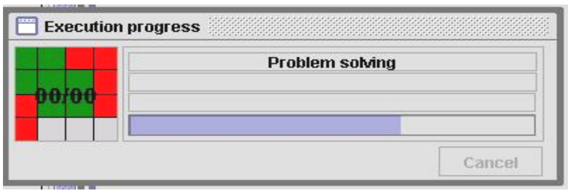


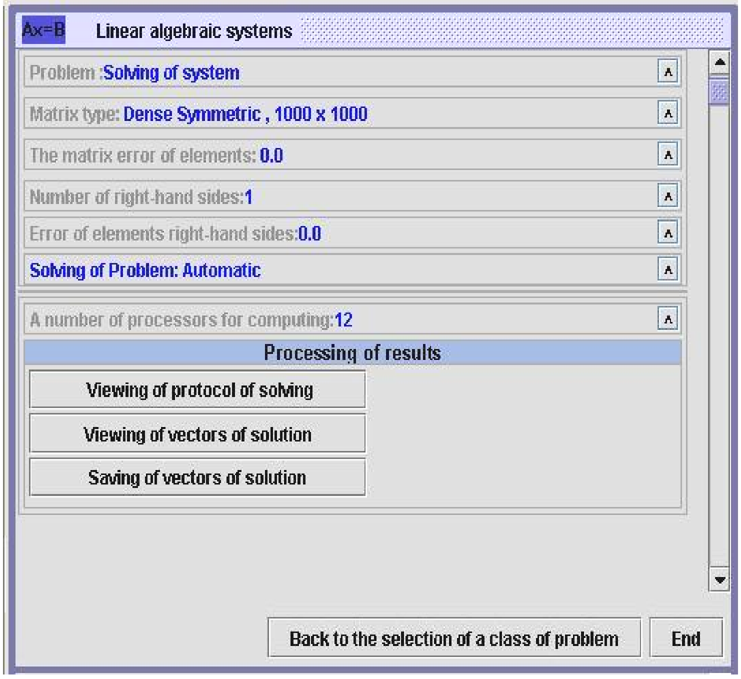
Fig. 4.13. Execution progress window

One may vary the ways of solving the same problem. For example, in order to choose interactive way after automatic solving of the problem one should click an arrow in the line entitled **«The solving of problem: Automatically»** (fig. 4.14) and then in the appeared menu choose **«The solving of problem: Interactively»**.

To solve another problem (with different initial data) from the class of problems under consideration one should click an arrow in the line entitled **«Problem»** (fig. 4.10), choose a problem from the list of problems in the appeared submenu (fig. 4.2) and then perform in succession all stages of work covering input of the initial data and solving of the problem.

4.4.2.4 Results of solving LAS

On the completion of computational process the brief information about problem which was solved appears in the upper part of the right-hand panel. Besides, a popup dialog window **«Processing of results»** (fig. 4.14) appears where user can obtain results of solving the problem.


Ax=B
Linear algebraic systems
Processing of results
Viewing of protocol of solving
Viewing of vectors of solution
Saving of vectors of solution
Back to the selection of a class of problem
End

Problem :Solving of system

Matrix type: Dense Symmetric , 1000 x 1000

The matrix error of elements: 0.0

Number of right-hand sides:1

Error of elements right-hand sides:0.0

Solving of Problem: Automatic

A number of processors for computing:12

Fig. 4.14. Popup dialog window – Processing of results

Results of solving LAS by Inpartool include:

- solution of LAS;
- protocol describing process of investigating and solving LAS.

On the completion of computational process a solution of the problem is automatically saved in a binary file with standard name `result.out.` To look over the obtained solution of the problem (in tabular form), save or print it in the text form one should press «**Browsing of solution**» button. The solution can also be saved in the tabular form), save and print it in the text form one should press binary form for its further using in the solving (by Inpartool**,** Inparlib or some other software) on Inparcom of those problems for which the solving of LAS was intermediate stage. To do this suffice it to press «**Saving of solution vectors**» and indicate the file name.

Protocol describing a process of investigating and solving LAS is presented in the text form. It includes the following descriptions: parameters of the problem, method employed for the investigation of problem in order to choose an efficient algorithm and construct a program for the solving of problem, several control components of the solution, an estimate for the inherited error in the solution, an estimate for the computational error in the solution, an estimate for the matrix condition number and some other characteristics of the problem, the problem's execution time and the number of processors being used.

To look over the protocol a user should press **«Browsing of solutions protocol»** button. The protocol can be printed, saved in text file or deleted. If the current protocol has not been deleted all protocols of problems solved after this problem during this work session will be appended to the already existing protocol.

4.4.2.5 Inpartool's diagnostics during the solving of LAS

During the process of formulating the problem, inputting initial data and solving LAS a user can get:

- referential information;
- help-type message;
- problem's run-time diagnostics.

Having click by right-hand mouse button on the title «Linear algebraic systems» (fig. 4.15) a user can become familiar with functional potentialities of Inpartool concerning the solving of problems belonging to this class of problems as well as with order of work. In similarly the same manner one can get appropriate short information (of the Help-type) at any stage of work by clicking the right-hand mouse button on any menu item, title, inscription or some other control element of interest.

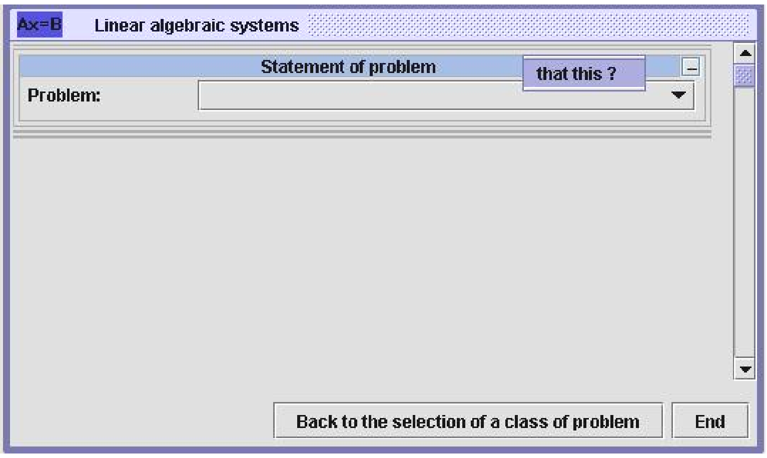
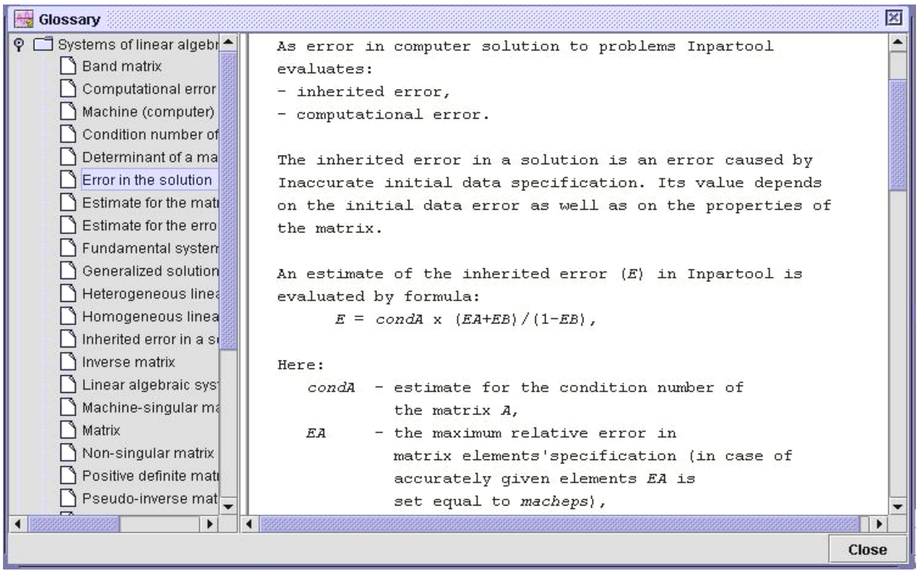


Fig. 4.15. Information pop-up

Some information about terminology related to the linear algebra being used can be obtained by choosing **«Glossary»** menu item in the submenu **«Help»** of the main menu (fig. 4.1). Having chosen a term of interest from the list on the left-hand panel in the appeared window user can get its explanation (fig. 4.16).

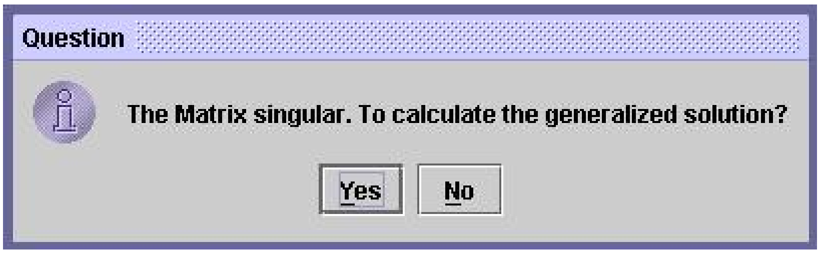


```
As error in computer solution to problems Inpartool
evaluates:
- inherited error,
- computational error.

The inherited error in a solution is an error caused by
Inaccurate initial data specification. Its value depends
on the initial data error as well as on the properties of
the matrix.

An estimate of the inherited error (E) in Inpartool is
evaluated by formula:
      E = condA x (EA+EB)/(1-EB),

Here:
   condA  - estimate for the condition number of
            the matrix A,
   EA     - the maximum relative error in
            matrix elements'specification (in case of
            accurately given elements EA is
            set equal to macheps),
```

Fig. 4.16 Glossary in Help menu

As noted above, after the completion of automatic or interactive solving of the problem by means of Inpartool a user gets (in protocol) information about: process of solving the problem, revealed characteristics of the problem, reliability of the obtained results or reasons for which problem was not solved.

In case of interactive solving of the problem the user can get some run-time information about a process of solving the problem and make a decision as to its further continuation. For example, if user was wrong in determining such characteristics of the matrix as positive definiteness or singularity then, having investigated the problem, Inpartool will deliver appropriate message and provide an opportunity for the user either to continue or interrupt a process of computations (fig. 4.17).



The Matrix singular. To calculate the generalized solution?

Fig. 4.17. Matrix singularity warning dialog

Besides, when working with Inpartool the user receives in case of the necessity various prompts, warnings and error messages.

4.4.3. Examples of solving linear algebraic systems by means of Inpartool

Let us illustrate the computational potentialities of Inpartool for the solving of LAS on the following problems.

Problem 1

Investigate and solve LAS $Ax = b$ by means of Inpartool, where

$$A = (a_{ij}),\ \ i, j = 1 \div n,\ a_{ij} = n + 1 + \max(i, j),$$

$$A = \begin{pmatrix} n & n-1 & n-2 & \dots & 1 \\ n-1 & n-1 & n-2 & \dots & 1 \\ n-2 & n-2 & n-2 & \dots & 1 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \dots \\ 1 \end{pmatrix}.$$

Exact solution of the system is the following:

$$x = (0 \quad 0 \quad \dots \quad -1 \quad 2)^T.$$

The problem was solved on 4 processors. Matrix elements as well as elements of the right-hand sides were input from the binary file `A16_1k.tam`. In dialog windows the user should indicate a type and structure of the matrix: **dense symmetric,** order of the matrix**: 1000**, maximum relative errors in elements: **zero.**

Protcol of solving the Problem 1 in automatic mode

```
   P R O B L E M:
     The solving of the linear algebraic system
     with a symmetric positive definite matrix

   D a t a :
     - matrix dimension                    = 1000
     - number of the right-hand sides
       of the systems                      = 1
     - maximum relative error
       in the matrix elements              = 0.00000e+00
     - maximum relative error
       in elements of the right-hand sides = 0.00000e+00

P r o c e s s   o f   i n v e s t i g a t i n g   a n d
s o l v i n g

   M e t h o d:
     - Cholesky decomposition
   R E S U L T S:
        SOLUTION IS OBTAINED IN FILE result.out

   E s t i m a t e s :
       - inherited error in the solution : 8.93106e-12
       - computational error in the solution:
         8.8817841970012602e-16

   P r o p e r t i e s  :
       - estimate of condition number of the matrix
         2.01110e+04

       Solution (last 10 components)

0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
0.00000 -1.00000 2.00000

Proc Number: 4
```

As one can see in the protocol, a matrix of the system is symmetric and positive definite. For such system a program was constructed for the investigating and solving of the problem together with reliability estimates for solution obtained by Cholesky algorithm. The computed solution (in the protocol we can see 10 last components of it) is of high accuracy that agrees with the given error estimates. Draw your attention to the following peculiarity of estimate for the inherited error: initial data of the problem are accurate (maximum relative errors in their elements an equal to zeros), while estimate for the inherited error in given protocol is non-zero. This can be explained by the fact that all real numbers being input to computer undergo some changes related to their machine representation. An accuracy of the number's representation is characterized by machine epsilon, i.e. the least floating-point number *macheps* Therefore, if user assigns maximum relative errors in elements equal to zeros, these values are replaced by *macheps*=2.220446049250313e-016).

Problem 2

Investigate and solve LAS $Ax = b$ by means of Inpartool, where

$$A = (a_{ij}),\ i, j = 1 \div n,\ n = 3w + 1,\ w = 1, 2, \ldots;\ a_{ii} = n - i,$$

$$a_{ij} = n + 1 - max\,(i, j),$$

$$A = \begin{pmatrix} n-1 & n-1 & n-2 & \ldots & 2 & 1 \\ n-1 & n-2 & n-2 & \ldots & 2 & 1 \\ n-2 & n-2 & n-3 & \ldots & 2 & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 2 & 2 & 2 & \ldots & 1 & 1 \\ 1 & 1 & 1 & \ldots & 1 & 0 \end{pmatrix},$$

$b = \{b\}_1^n$, $b_i = n - i$, if $i \le 2$; $b_i = n + 1 - \mathrm{i}$, if $\mathrm{i} > 2$.

Exact solution of the system is the following:

$$x = (0 \quad 1 \quad 0 \quad \ldots \quad 0)^T.$$

Matrix elements as well as elements of the right-hand side should be input from the binary file `A41_1000.tam`. In dialog windows the user should indicate a type and structure of the matrix: **dense symmetric**, order of the matrix: **1000**, maximum relative errors in elements of the system: **zeros**.

Protocol of solving the Problem 2 in automatic mode

```
  P R O B L E M:
      solving of the linear algebraic system
      with a symmetric positive definite matrix

  D a t a :
      - matrix dimension                   = 1000
     - number of the right-hand sides
       of the systems                      = 1
     - maximum relative error
       in the matrix elements              = 0.00000e+00
     - maximum relative error
       in elements of the right-hand sides = 0.00000e+00

P r o c e s s   o f   i n v e s t i g a t i n g   a n d
s o l v i n g

   M e t h o d:
       - Cholesky decomposition

   R E S U L T S:
      !!! THE MATRIX IS NOT POSITIVE DEFINED !!!
     Number of processors: 4

M e t h o d:
    - Gauss elimination with partial pivoting

   R E S U L T S:
        !!! THE MATRIX IS MACHINE-SINGULAR !!!

     Number of processors: 4

M e t h o d:
        - singular value decomposition of a general
          matrix

   R E S U L T S:

     SOLUTION WAS CALCULATED

     first 4 components of solution (vector 1) are:

   -3.7747582837255322e-010    1.0000000000000031e+000
    3.8857805861880479e-010    3.6489927986770073e-010

The vector(s) of solution are successfully stored in the
file result.out

     Error estimations:   4.99145e-08

     P r o p e r t i e s:
        - estimation of conditional number: 7.49316e+07
        - matrix rank: 999

     Number of processors: 12
```

As one can see in the protocol, since the system's matrix is symmetric, Inpartool has chosen the Cholesky algorithm (most economic algorithm for such matrices) as a trial algorithm for investigating the problem. However, during the process of investigating by this algorithm the matrix turned out to be not-positive definite, and for its further investigation Inpartool has chosen the Gauss algorithm. During the investigating of LAS by Gauss algorithm the matrix turned out to be machine-singular. It is possible to construct a generalized solution for such LAS. In the automatic mode a problem was synthesized for finding the generalized solution of LAS based on the *SVD*-matrix decomposition, the required topology for this algorithm was created from the available processors, data arrays were redistributed between processors and the problem was solved together with reliability estimates for the solution. However, if the user a priori knows that system's matrix is singular he can solve the problem interactively. From the very beginning Inpartool synthesizes a program for the *SVD*-decomposition for finding a generalized solution of LAS and solves the problem together with reliability estimates for the solution.

Protocol of solving Problem 2 in interactive mode

```
 P R O B L E M:
     solving of the linear algebraic system
     with a general matrix

 D a t a :
     - matrix dimension                    = 1000
     - number of the right-hand sides
       of the systems                      = 1
     - maximum relative error
       in the matrix elements              = 0.00000e+00
     - maximum relative error
       in elements of the right-hand sides = 0.00000e+00

P r o c e s s   o f   i n v e s t i g a t i n g   a n d
s o l v i n g

   M e t h o d:
      - singular-value decomposition of a general matrix

     R E S U L T S:   SOLUTION WAS CALCULATED

    first 4 components of solution (vector 1) are:


     -3.7747582837255322e-010    1.0000000000000031e+000

      3.8857805861880479e-010    3.6489927986770073e-010


The vector(s) of solution are successfully stored in the
file result.out

     Error estimations:   4.99145e-08

     P r o p e r t i e s:
       - estimation of conditional number: 7.49316e+07
       - matrix rank: 999

     Number of processors: 12
```

If some absolute error is introduced in the last element of the matrix, namely instead of zero value set: **1.e–8** and set the value of the maximum relative error equal to **1.e–14**, then one can see in the protocol below that the solution of the problem and its error estimate have also changed.

Protocol of solving the Problem 2 with changed value of maximum relative error

```
P R O B L E M:
     solving of the linear algebraic system
     with a general matrix

 D a t a :
     - matrix dimension                     = 1000
     - number of the right-hand sides
       of systems                           = 1
     - maximum relative error
       in the matrix elements               = 1.00000e-14
     - maximum relative error
       in elements of the right-hand sides = 0.00000e+00

P r o c e s s   o f   i n v e s t i g a t i n g   a n d
s o l v i n g

     M e t h o d:
      - singular-value decomposition of a general matrix

     R E S U L T S:SOLUTION WAS CALCULATED

     first 4 components of solution (vector 1) are:

     4.023313522338e-007        1.0000000000018e+000
    -6.407499313354e-007       -5.2154064178466e-007

The vector(s) of solution are successfully stored in the
file result.out

     Error estimations:   1.51527e-06

     P r o p e r t i e s:
        - estimation of conditional number: 7.49316e+07
        - matrix rank: 999

     Number of processors: 12
```

Problem 3

Investigate and solve LAS $Ax = b$ by means of Inpartool, where

$$A = (a_{ij}),\ i, j = 1 \div n,\ a_{ij} = 2,\ j \geq i;\ a_{ij} = 1,\ j < i;$$

$$A = \begin{pmatrix} 2 & 2 & 2 & \dots & 2 \\ 1 & 2 & 2 & \dots & 2 \\ 1 & 1 & 2 & \dots & 2 \\ \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 2 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \dots \\ 0 \end{pmatrix}.$$

Exact solution of the system is:

$$x = \begin{pmatrix} 1 & 0 & 0 & \dots & -0.5 \end{pmatrix}^T .$$

The problem was solved on 40 processors. Matrix elements as well as elements of the right-hand side should be input from the file `A2_2k.tam` in the binary form. In dialog windows the user should indicate a type and structure of the matrix: **dense general**, order of the matrix: **2000**, maximum relative errors in elements of the system: zero.

Protocol of solving the Problem 3 in automatic mode

```
 P R O B L E M:
     solving of the linear algebraic system
     with a general matrix

D a t a :
     - number of matrix's rows              = 2000
     - number of matrix's columns           = 2000
     - number of the right-hand sides
               of the systems               = 1
     - maximum relative error
       in the matrix elements               = 0.00000e+00
     - maximum relative error
       in elements of the right-hand sides = 0.00000e+00

P r o c e s s   o f   i n v e s t i g a t i n g   a n d
s o l v i n g

M e t h o d:
    Gauss elimination with partial pivoting

R E S U L T S :

    SOLUTION IS OBTAINED IN FILE result.out

E s t i m a t e s:
     - inherited error in the solution: 2.96064e-12
     - computational error in the solution: 0.44630e-16
```

```
P r o p e r t i e s:
 - estimate of condition number of the matrix .66678e+03

  Solution for right part number 1

  first 5 components:

1.0000000000000000e+00    0.0000000000000000e+00
0.0000000000000000e+00    0.0000000000000000e+00
0.0000000000000000e+00

  last 5 components:

0.0000000000000000e+00    0.0000000000000000e+00
0.0000000000000000e+00    0.0000000000000000e+00
-5.0000000000000000e-01

Proc number: 40
```

A one can see in the protocol, a matrix of the system is non-singular. For such kind of system Inpartool synthesizes a program for its investigating and solving by Gauss method's algorithm. The value of estimate for the condition number is small therefore the computed solution (5 first and 5 last components of it are presented in the Protocol) possesses the high accuracy that is confirmed by delivered error estimates.

4.5. Investigating and solving of eigenvalue problems

4.5.1. Functional potentialities of Inpartool on investigating and solving of eigenvalue problems

Eigenvalue problems arise in the determining of frequencies and forms of eigen-oscillations of conservative dynamic systems, in investigating of oscillations and stability of objects of mechanical, physical and chemical origin, in factor analysis and as independent mathematical problems.

Algebraic eigenvalue problem (AEVP) consists in finding such numbers λ, for which there exist different from zero solutions of LAS

$$Ax = \lambda Bx\,, \tag{4.5}$$

where A and B are some square matrices of order n. Numbers λ are called eigenvalues of the problem (4.5), while vectors x are called eigenvectors of this problem. If B is an identity n-th order matrix the problem (4.5) is referred to as a standard eigenvalue problem; otherwise – as a generalized problem. In case of standard problem numbers λ and vectors x are also called eigenvalues and eigenvectors of the matrix A.

Problem consisting in finding all eigenvalues and eigenvectors corresponding to them is called a full eigenvalue problem. Problem consisting in finding several eigenvalues and vectors corresponding to them or finding only eigenvalues is referred to a partial eigenvalue problem.

Eigenvalues of either real symmetric matrix or complex-valued Hermitian matrix are real numbers which can be ordered, for example, in increasing order and, then, they can be renumbered. Eigenvectors of real symmetric matrix are real, while eigenvalues of the Hermitian complex-valued matrix are complex-valued.

When solving application problems, AEVPs with accurate initial data

$$\tilde{A}\tilde{x} = \tilde{\lambda}\tilde{B}\tilde{x}\ . \tag{4.6}$$

arise very seldom.

The approximate nature of initial data of the problem (4.5) is caused by the following factors: inaccuracies of measurements performed during the statement of the application problem; accepted simplifications and admissions; errors in the discretization of continuous mathematical model; the using of approximate formulas when forming the initial data, etc.

The most typical specification of the problem (4.5) and error in the initial data is the following:

$$\left\|\tilde{A} - A\right\| \equiv \left\|\Delta A\right\| < \varepsilon_{\mathrm{A}}, \quad \left\|\tilde{B} - B\right\| \equiv \left\|\Delta B\right\| < \varepsilon_{\mathrm{B}} \tag{4.7}$$

The investigation of characteristics of AEVPs with approximately given initial data may include spectrum decomposition of the matrix, construction of the invariant subspaces (for example, eigen- or root-subspaces) of canonical forms (for example, the Jordan's), determination of conditioning of eigenvalues and eigenvectors, investigation of perturbations in solutions depending on errors in the initial data, reliability estimates of the obtained machine solutions, i.e. solutions of the problem (4.5) obtained in computer together with initial data errors (4.7).

A proximity between elements of matrices *A*, *B* and $\tilde{A}, \tilde{B}$**,** respectively, doesn’t always provide proximity between eigenvalues of the problem. When investigating standard eigenvalue problem the following cases should be distinguished [11]:

- perturbation in simple eigenvalue of the matrix possessing linear elementary divisors;
- perturbation in multiple eigenvalue of the matrix possessing linear elementary divisors;
- perturbation in simple eigenvalue of the matrix possessing one or more non-linear elementary divisors;

- perturbation in multiple eigenvalue corresponding to non-linear elementary divisor of the full matrix;
- perturbation in multiple eigenvalues λ_I when more than one elementary divisor with multiplier (λ_I-λ) is available and, at least, one of them is non-linear.

Similar cases may also arise for the generalized eigenvalue problem. When evaluating eigenvectors a problem of estimating the reliability of the obtained solutions arises, as well.

On the basis of the foregoing one can see that the entire class of eigenvalue problems (4.5), (4.7) arises in the describing of physical models. A proximity between solution of problems (4.6) and (4.5), (4.7) is determined, on the one hand, by characteristics of problem's matrices, while, on the other hand, by errors in the initial data specification.

Thus, the computer implementation of methods for finding eigenvalues and eigenvectors of the problem (4.5), (4.7) introduces an error determined by characteristics of the problem's matrix (matrices), by methods for solving AEVPs as well as by characteristic peculiarities of computations. Therefore at the stage of computer solving of the problem the following investigations (which should take into account the above mentioned cases of perturbations in eigenvalues) are carried out:

- to reveal the existence and uniqueness of solution of the machine problem;
- to investigate its stability within the level of errors in the initial data (ε_A, ε_B);
- to choose an algorithm according to the revealed characteristics;
- to estimate inherited and computational errors in the obtained solution, i.e. estimate the proximity between the obtained and exact solutions of the machine problem.

Hence, the investigating of the reliability of the obtained computer solutions to matrix eigenvalue problems includes: the revelation and investigation of characteristics of problems (4.6) and (4.5), (4.7) as well as characteristics of machine problem corresponding to them; estimation of the inherited error in the mathematical solution as well as estimation of the computational error in the obtained machine solution and estimation of the overall error in the solution.

Proceeding from the analysis of practical problems Inpartool includes the solving of AEVPs with following real symmetric matrices: dense, tri-diagonal, band positive definite. For these types of matrices the following AEVPs with approximately given initial data are considered:

- investigate and solve full standard AEVP $Ax = \lambda x$ with tri-diagonal symmetric matrix A;

- investigate and solve full standard AEVP $Ax = \lambda x$ with dense symmetric matrix A;
- investigate and solve partial (finding of some minimum eigenvalues and their corresponding eigenvectors) standard AEVp $Ax = \lambda x$ with band symmetric positive definite matrix A;
- investigate and solve partial generalized AEVP $Ax = \lambda Bx$ with band symmetric positive definite matrices A and B.

As in the case of LAS, these problems can be solved by optimum number of algorithms, various modifications of which take into account all problems and types of matrices under consideration. Thus, the *QL*-algorithm is employed for the evaluation of all eigenvalues both of tri-diagonal and dense symmetric matrix. Algorithm of the iterations' method on the subspace is used for the solving both of standard and generalized AEVP with band symmetric matrices.

Besides, for the solving of AEVP algorithms involved in the investigating and solving of LAS are used. For example, algorithms of the Cholesky method are employed both in the solving of problems by iterations' method on the subspace and in the reliability analysis of the obtained solution to the partial AEVP with band symmetric matrices.

A sequence of orthogonal reflection transformation (the Householder’s method) is used for the reduction of dense symmetric matrix to tri-diagonal symmetric matrix].

If A is a real and symmetric matrix the evaluation of eigenvalues of symmetric matrices is always stable and proximity between problems (4.6) and (4.5), (4.7) is determined only by the initial data error. However, error in the evaluation of eigenvectors also depends on the proximity of egenvalues. The following error estimate for the simple eigenvalue and its corresponding eigenvector is well known [7]:

$$|\Delta\lambda| \le \|\Delta A\|, \qquad \frac{\|\Delta x_i\|}{\|x_i\|} \le \|\Delta A\| \sum_{j=1,\, j\ne i}^{\mathrm{n}} \frac{1}{|\lambda_i - \lambda_j|}.$$

If λ approximates the multiple eigenvalue $\tilde{\lambda}_i$ $(i = p, p+1, \ldots, q)$ of the matrix $\tilde{A}$ and $|\tilde{\lambda}_i - \lambda| \ge s$ $(i \ne p, p+1, \ldots, q)$, while x is an eigenvector of the problem (4.5) corresponding to λ, then there exists vector $f = \alpha_p \tilde{x}_p + \cdots + \alpha_q \tilde{x}_q$ ($\tilde{x}_i$ – eigenvectors of problem (4.6) for which [2]

$$\|f - x\| \le \|\Delta A\| / s.$$

An overall error in the solution of the algebraic eigenvalue problem (4.5), (4.7) can be estimated as follows:

$$\min_j \left|\tilde{\lambda}_j - \hat{\lambda}\right| \le \|\Delta A\| + \|r\|, \qquad \|f - \hat{x}\| \le \left(\|\Delta A\| + \|r\|\right)/s, \tag{4.8}$$

where $\hat{\lambda}$ and vector $\hat{x}$ are approximate eigenvalue and its corresponding eigenvector of the problem (4.5), respectively, while $r = A\hat{x} - \hat{\lambda}\hat{x}$ is the residual vector.

The following estimate delivers relative to error in eigenvalues of the generalized problem (4.5) with symmetric positive definite matrices:

$$\min_j \left|\frac{\tilde{\lambda}_j - \hat{\lambda}}{\tilde{\lambda}_j}\right| \le \sqrt{\frac{\hat{r}^{\mathrm{T}} B \hat{r}}{\hat{x}^{\mathrm{T}} B \hat{x}}}, \quad \hat{r} = A^{-1} r = \hat{x} - \hat{\lambda} A^{-1} B \hat{x}.$$

When solving partial AEVP by method of iterations' on the subspace a loss (non-evaluation) of one or several minimum eigenvalues being evaluated and their corresponding eigenvectors is possible. This phenomenon is caused by orthogonality of each such eigenvectors as well as of the initial subspace being iterated. For a posteriori diagnostics of such phenomenon a property of the Sturm's sequence is used. To this end the LDL^{T}-decomposition of the matrix A-μB is carried out, where shift μ should exceed maximum of the evaluated eigenvalues; then the number of eigenvalues less than μ is equal to the number of negative elements of the diagonal matrix D.

The distinguished features of the Inpartool are the following:

- investigating of characteristics of AEVP;
- possibility of the automatic choice of algorithm and its parameters according to the revealed characteristics of AEVP;
- possibility of the automatic choice of topology (the number of processors) of the parallel computer according to the chosen algorithm and its parameters;
- the solving of AEVP with approximately given initial data;
- investigation of reliability of AEVP's solutions;
- possibility of work with software without preliminary familiarization with it as well as without studying of instructions.

At the level of concepts, Inpartool implements fundamental principles of the information computing technology for the solving of problems. This technology involves: formulation of problem in terms of the subject area language, investigation of characteristics of the problem being solved and automatic choice of algorithm depending on the revealed problem's characteristics, syntheses of the solution program with taking into account mathematical and engineering characteristics of computer, the solving of problems and analyzing the reliability of the obtained results, dialog support and information referential provision for the process of problem's formulating, investigating and solving.

4.5.2. Technology of investigating and solving algebraic eigenvalue problems

4.5.2.1 Applying of Inpartool for the solving of AEVP

The main window «**Algebraic eigenvalue problem**» consists of the main menu and two panels (fig. 4.18).

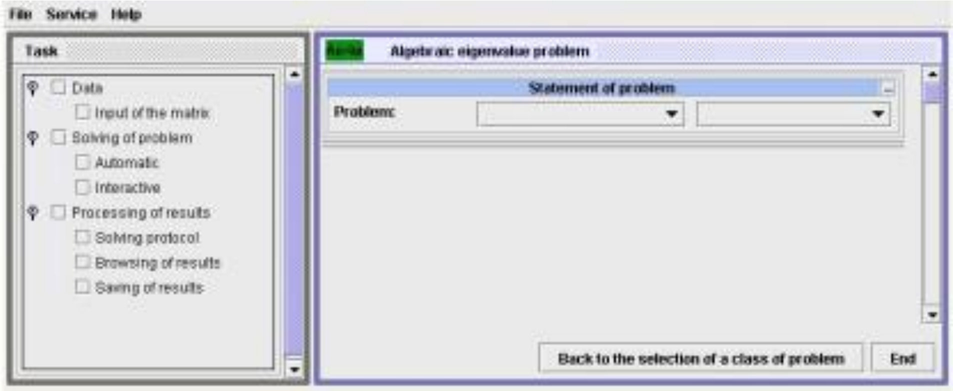


Fig. 4.18 Main Algebraic eigenvalue problem window

The left panel (passive) reflects a sequence of work stages and sub-stages which were already performed, being performed and will be performed.

To solve the problem a user should carry out the following successive stages of work in the right-hand (active) panel:

- formulate the problem;
- input the problem's initial data;
- start the problem;
- obtain results.

Inpartool solves the following AEVPs:

- full AEVP with tri-diagonal symmetric matrix;
- full AEVP with dense symmetric matrix;
- with dense symmetric matrix;
- partial AEVP (evaluation of several minimum eigenvalues and their corresponding eigenvectors);
- standard or generalized AEVP with band symmetric matrices.

To formulate a problem a user should choose the required item from one of two pull-down lists of problems from the given class which can be solved by Inpartool (fig. 4.19, 4.20).

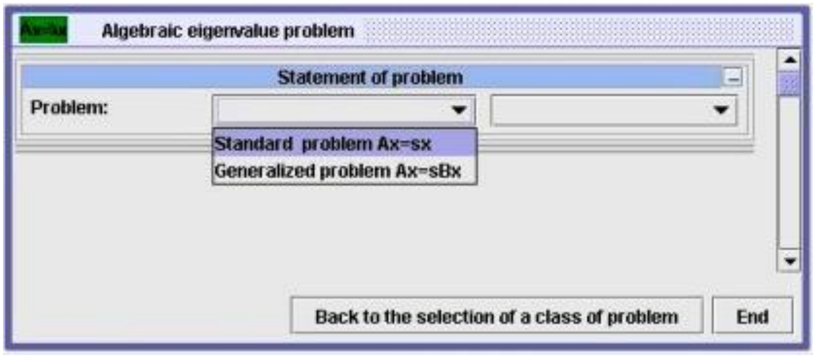
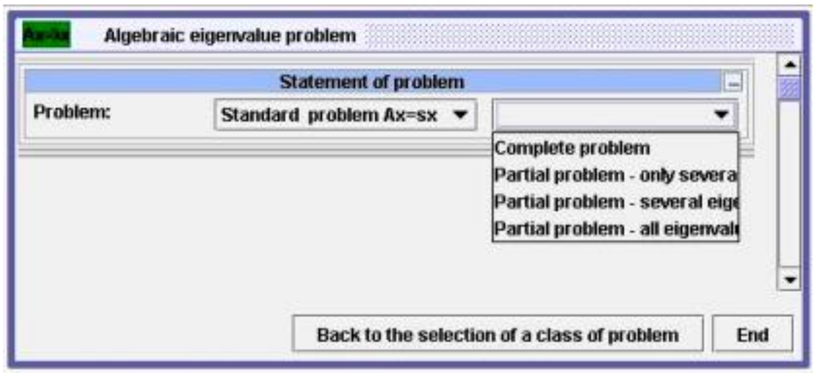

Ax=λx
Algebraic eigenvalue problem
Statement of problem
Problem:
Standard problem Ax=sx
Generalized problem Ax=sBx
Back to the selection of a class of problem
End

Fig. 4.19. Type of problem pull-down list

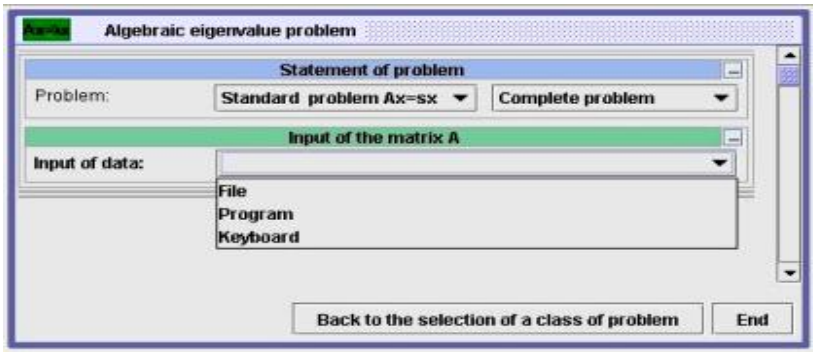

Ax=λx
Algebraic eigenvalue problem
Statement of problem
Complete problem
Partial problem - only severa
Partial problem - several eige
Partial problem - all eigenvalu
Back to the selection of a class of problem
End

Problem: Standard problem Ax=sx

Fig. 4.20. Kind of problem pull-down list

4.5.2.2 *Specification of initial data for the solving of AEVP*

Initial data for the solving of AEVP are given by parameters of the problem being solved, i.e. by the following: order of the matrix (matrices), the number of diagonals for band matrices, number of the first and last eigenvalues to be evaluated for the partial problem, maximum relative error in matrix elements as well as elements of the matrix (matrices).



Problem: Standard problem Ax=sx Complete problem

Fig. 4.21. Initial data input pull-down list

The initial data (fig. 4.21) can be input from the binary file (the input from file) and/or their values can be directly entered by user into corresponding data fields (input by means of the keyboard). In addition it is also possible to form matrix elements by program (functions in C) written by user.

The following data structures (formats) are supported by Inpartool for the data input from file (files):

- file contains only matrix elements in form of floating-point numbers the sequence of which is determined by form of the matrix (the file mask is `*.tam`), each number occupying 8 bytes;
- file contains in binary form parameters and elements of the matrix (the file mask is `*.edat`); Table 4.4 contains structure of such file for the case of band symmetric matrix;
- file contains parameters and elements of two matrices (the file mask is `*.eedat`) required for the solving of generalized AEVP; Table 4.5 contains a structure of such file for the case of band symmetric matrices.

Table 4.4. `*.edat` file structure.

Contents of the file	**Type**	**Bytes**
Format version (=1)	Integer	4
Matrix structure and type code (= 17)	" – "	4
Code of order in which matrix elements are written	" – "	4
Order of the matrix	" – "	4
Band width of the matrix	" – "	4
Relative error in matrix elements	Double precision floating-point number	8
Matrix elements (diagonal and sub-diagonal) in the form of sequence of numbers	" – "	$8\times l_A$

To encode matrix structure and type of the following formula may be used:

$$16\ i_1 + i_2,$$

where $i_1 = 0$ for dense matrix, $i_1 = 1$ for band matrix, $i_1 = 3$ for diagonal matrix, $i_2 = 0$ for general matrix, $i_2 = 1$ for symmetric matrix. Code of the identity matrix is equal to 49.

The bandwidth of the matrix is equal to the number of sub- and over-diagonals plus 1, i. e. in the case of symmetric matrix it is equal to $2m + 1$, where m is the band half-width. Diagonal and sub-diagonal elements of the matrix are entered in succession a row by row, each row beginning from the diagonal element. For example, for the band symmetric matrix the 9-th order with band half-width equal to 2 (the band width is equal to 5) elements of the matrix should be stored in the following order:

$$a_{11},\ a_{22},\ a_{21},\ a_{33},\ a_{32},\ a_{31},\ \ldots,\ a_{99},\ a_{98},\ a_{97}.$$

In so doing the code in which elements of the matrix are written is equal to 1, while the number of matrix elements stored in the file $l_A = 24$.

Table 4.5. `*.eedat` file structure.

Contents of the file	**Type**	**Bytes**
Format version (=2)	Integer	4
Matrix structure and type code for the first (left-hand) matrix (= 17)	" – "	4
Code of order in which elements of the first matrix are written (= 1)	" – "	4
Matrix structure and type code for the second (left-hand) matrix	" – "	4
Code of order in which elements of the second matrix are written (= 1)	" – "	4
Order of problem’s matrices	" – "	4
Band width of the first matrix	" – "	4
Band width of the second matrix	" – "	4
Relative error in elements of first matrix	Double precision floating-point number	8
Relative error in elements of second matrix	" – "	8
Elements of the first matrix	" – "	8×l_A
Elements of the second matrix	" – "	8×l_B

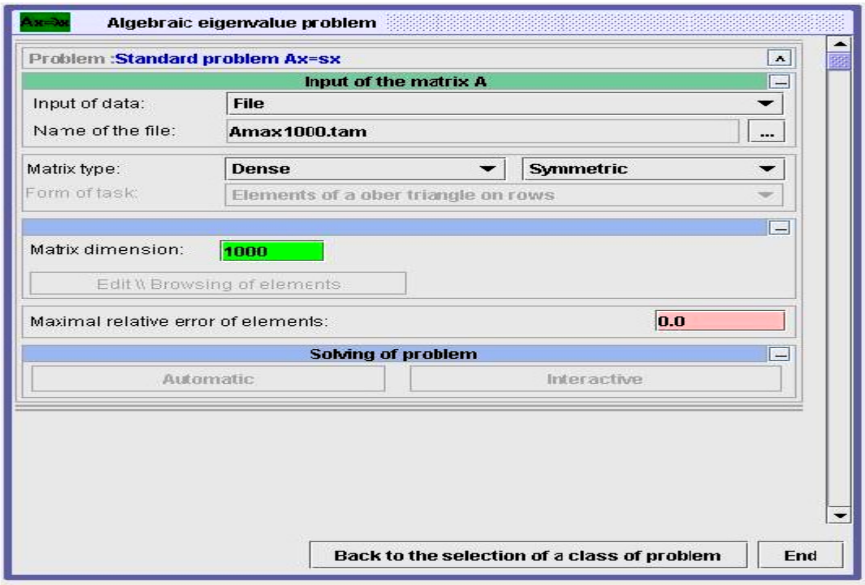
Elements of tri-diagonal symmetric matrix are stored in file in the following order: in succession, beginning from a_{11}, elements of the main diagonal and then elements of the first sub-diagonal.

Elements of the dense symmetric matrix are stored by rows in succession.

As an example, consider a case where the standard full eigenvalue problem with dense symmetric matrix of order $n = 1000$ is solved. Elements of the matrix are written in succession by rows to the file `Amax1000.tam`. A user should choose the file input of the matrix and having indicated data format, enter the file name `Amax1000.tam`. After this in the appeared lists he should choose the matrix type – dense, its structure –symmetric, the matrix order – 1000, and the maximum relative error in its elements are to be entered into data fields (fig. 4.22). By default, matrix elements are considered to be given accurately and value of the maximum relative error is equal to 0.

During the input of numerical parameters the data fields corresponding to them are colored in red if these values are necessarily to be determined (i. e. such parameters are not assigned any values by default). After the input of values the data fields become green. The input of numerical parameters should be finished by pressing `<Enter>` button. In so doing the information is entered and passage to the next data input field or control element takes place. Matrix elements can be edited in the manner described in Chapter 3.

As to the program specification of matrix elements one should enter text of function written in C that forms diagonal and sub-diagonal elements of one row of the matrix (the number of row is an input parameter of this function). User may prepare the text in advance in file or enter it directly and save in a new file. Dialog on the determining of a priori created file containing text of the function is similar to dialog on the determining of the data file. For the direct entering of text or its corrections Inpartool employs a text editor.



Algebraic eigenvalue problem

Problem :Standard problem Ax=sx

Input of the matrix A

Input of data: File

Name of the file: Amax1000.tam

Matrix type: Dense | Symmetric

Form of task: Elements of a ober triangle on rows

Matrix dimension: 1000

Maximal relative error of elements: 0.0

Solving of problem

Ax=λx
Edit \\ Browsing of elements
Automatic
Interactive
Back to the selection of a class of problem
End

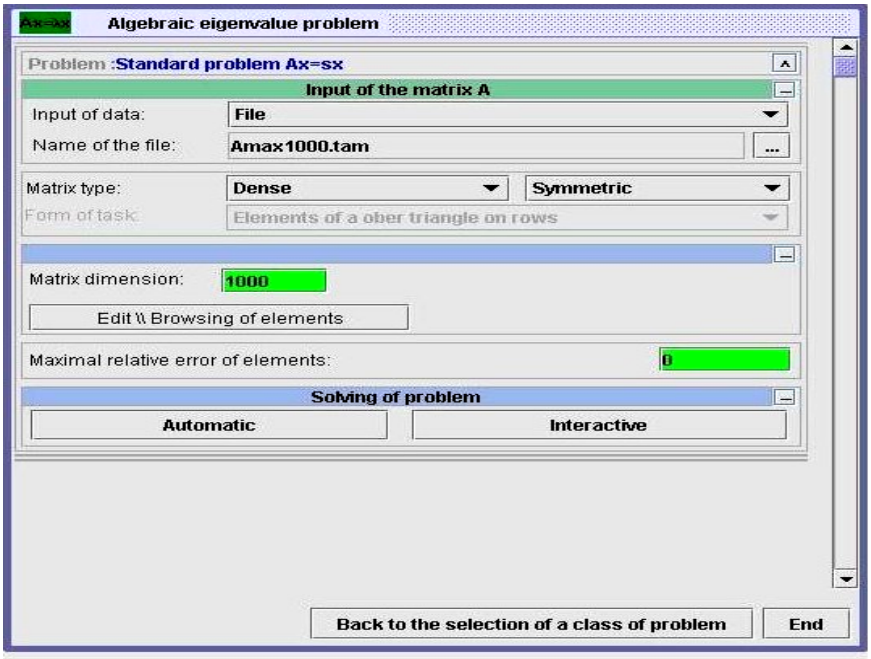
Fig. 4.22. Matrix parameters input window

4.5.2.3 *The solving of AEVP*

The solving of algebraic eigenvalue problem in Inpartool can be carried out either automatically without user’s involvement or interactively in the dialog with user.

To solve the problem one should, after the successful input of the initial data, press appropriate button («**Automatically**» or «**Interactively**»), which appears under the title «**The solving of problem**» (fig. 4.23).

In the case of automatic solving of problem Inpartool first of all investigates the problem. On the basis both of initial data investigation and characteristics of AEVP revealed by Inpartool as well as according to engineering and mathematical potentialities of Inparcom-16 an algorithm for the solving of problem is chosen, a processors' topology is constructed, according to the chosen algorithm the initial data are distributed between processors, the problem is solved and reliability of the obtained results is investigated.



Algebraic eigenvalue problem

Problem :Standard problem Ax=sx

Input of the matrix A

Input of data: File

Name of the file: Amax1000.tam

Matrix type: Dense Symmetric

Form of task: Elements of a ober triangle on rows

Matrix dimension: 1000

Maximal relative error of elements: 0

Solving of problem

Ax=sx
Edit \\ Browsing of elements
Automatic
Interactive
Back to the selection of a class of problem
End

Fig. 4.23. Problem solution mode window

During the iterative solving of the problem a user can specify the number of processors on which the problem is to be solved as well as other parameters of the problem if they can be modified, for example, a block size. Besides, when solving AEVP with band symmetric matrix (matrices) the user can choose a solution algorithm for matrices with narrow band or matrices with wider band. Then it is necessary to press **«Further»**.

During the solving of problem a window appears showing a progress of performing the task which will be closed after the completion of solving the problem. After the completion of computations short information about the problem that was solved appears in upper part of the right-hand panel together with title **«Processing of results»** located below.

4.5.2.4 Results of solving AEVP

Results of solving AEVP by Inpartool are the following:

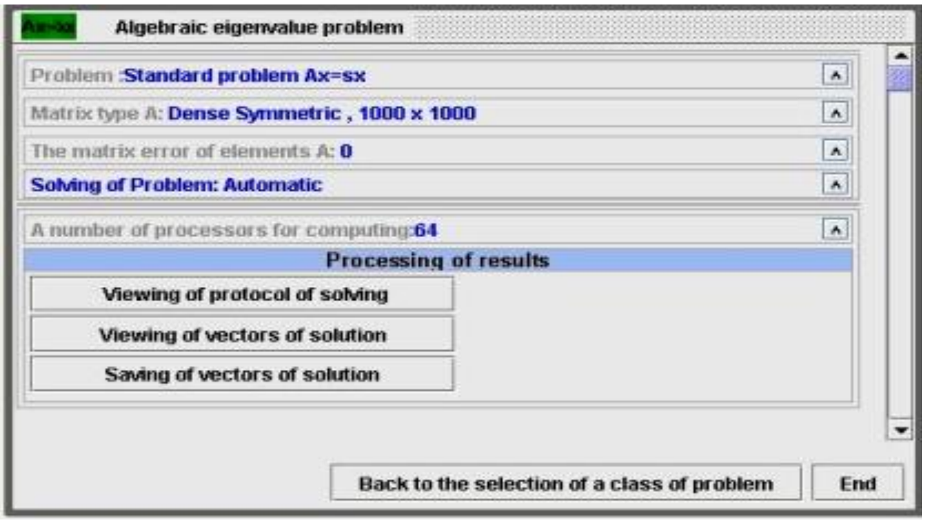
- evaluated eigenvalues;
- evaluated eigenvectors;

- error estimates for eigenvalues;
- error estimates for eigenvectors;
- other information on the reliability of the obtained solutions.

Numerical results (eigen- pairs and their estimates) are saved in a binary file with standard name `result.out`. Some part of numerical results as well as characteristic information about the reliability of the obtained results in the text form are placed in protocol describing the process of investigating and solving of AEVP.

To look over and process the obtained results one should press a button located below the title **«Processing of results»** (fig. 4.24). To look over and process the obtained results saved in the file `result.out`, one should press **«Browsing of results»** button. In the appeared window the results of solving the problem are presented in the form of table (eigenvalues, estimates, eigenvectors). These results or a part of them can be printed or saved in a binary file with unique name for their further using. If it is necessary only to save results suffice it to press **«Saving of solution»** button.

A protocol describing process of investigating and solving AEVP in addition to some results of solving the problem contains description of the problem, name of the method (algorithm) used for the solving of problem, the number of processors being used, the problem's execution time. To look over the protocol the user should press **«Browsing of protocol»** button. By pressing an appropriate button in the browsing window the user can print the protocol, save it in the text file or delete. If current protocol wasn't deleted a protocol of the next problem solved during this work session will be appended to the already existing protocol.



Algebraic eigenvalue problem

Problem :Standard problem Ax=sx

Matrix type A: Dense Symmetric , 1000 x 1000

The matrix error of elements A: 0

Solving of Problem: Automatic

A number of processors for computing:64

Processing of results
Viewing of protocol of solving
Viewing of vectors of solution
Saving of vectors of solution
Back to the selection of a class of problem
End

Fig. 4.24. Processing of results – main window

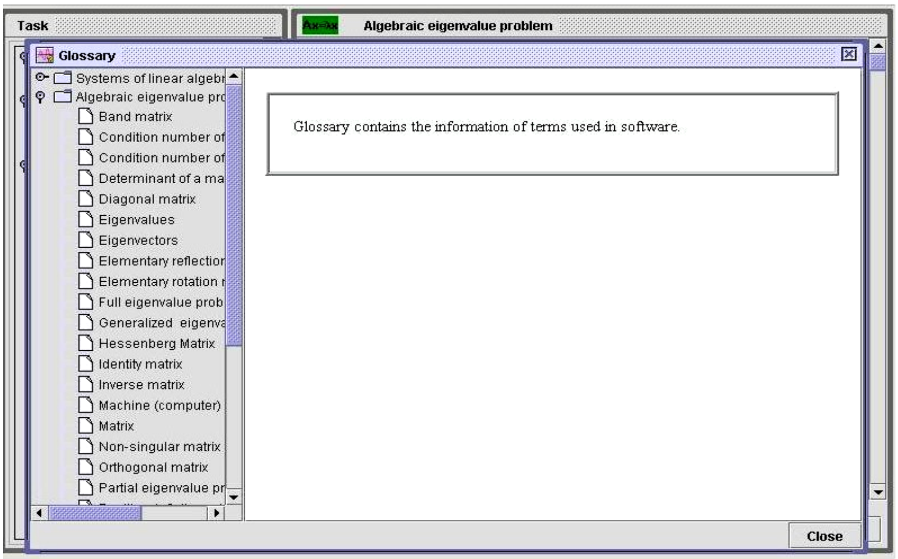
4.5.2.5 Inpartool's diagnostics during the solving of AEVP

During formulating or inputting of the initial data and solving of AEVP a user can get:

- referential information;
- Help-type messages;
- Problem's run-time diagnostics.

Having clicked by right-hand mouse button on the title **«Algebraic eigenvalue problem»** (fig. 4.18) a user can familiarize himself with Inpartool's functional potentialities as to the solving of problems belonging to this class of problems as well as with order of work with Inpartool. In similarly the same manner one can get appropriate short information (of the Help-type) at any stage of work by clicking the right-hand mouse button on any menu item, title, inscription or some other control element of interest.

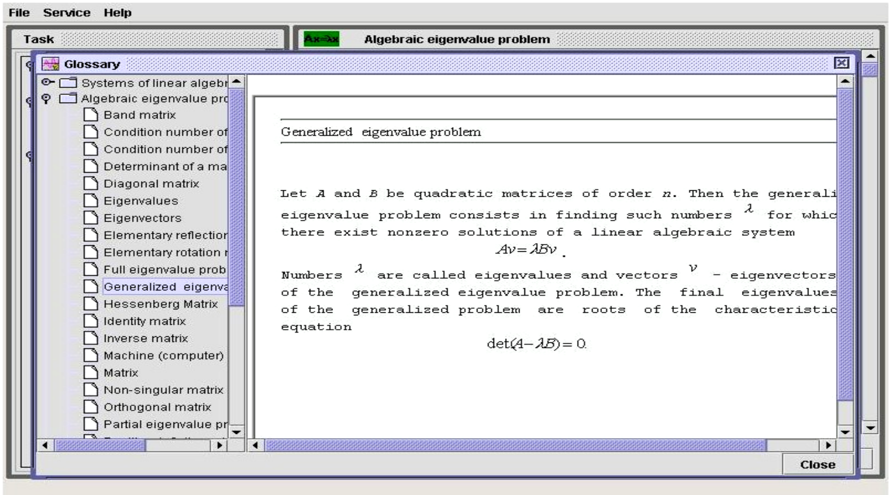
In **«Help»** item of the main menu a user can get information about functional potentialities of Inpartool or terminology related to the linear algebra being used (fig. 4.25).



Glossary contains the information of terms used in software.

Fig. 4.25. Glossary main window

Having chosen the **«Glossary»** item in the **«Help»** submenu and then a term of interest from the list on the left-hand panel in the appeared window a user get its explanation (fig. 4.26). Besides, Inpartool issues to user various prompts, warnings and error messages.



Generalized eigenvalue problem

Let A and B be quadratic matrices of order n. Then the generali eigenvalue problem consists in finding such numbers λ for whic there exist nonzero solutions of a linear algebraic system

$$Av = \lambda Bv\ .$$

Numbers λ are called eigenvalues and vectors v – eigenvectors of the generalized eigenvalue problem. The final eigenvalues of the generalized problem are roots of the characteristic equation

$$\det(A - \lambda B) = 0.$$

Fig. 4.26. Glossary item window

4.5.3. Examples of solving algebraic eigenvalue problems by means of Inpartool

Let us illustrate computational potentialities of Inpartool for the solving of AEVP on the following problems.

Problem 1

Solve the full standard AEVP $Ax = \lambda x$, where A is a tri-diagonal symmetric matrix:

$$A = \begin{pmatrix} 2 & 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & 2 & 1 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & 2 & 1 & \dots & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & 2 \end{pmatrix}$$

Exact solution of the Problem 1 (i,k = 1, 2, …, n) has a form

$$\tilde{\lambda}_k = 4\sin^2\frac{k\pi}{2(n+1)}, \qquad \tilde{x}_{i,k} = \sqrt{\frac{2}{n+1}}\sin\frac{(n+1\text{-}k)i\pi}{n+1}\ .$$

Problem 2

Solve the full standard AEVP $Ax = \lambda x$ with dense symmetric matrix:

$$A=\begin{pmatrix} n-1 & n-1 & n-2 & \ldots & 2 & 1 \\ n-1 & n-2 & n-2 & \ldots & 2 & 1 \\ n-2 & n-2 & n-3 & \ldots & 2 & 1 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 2 & 2 & 2 & \ldots & 1 & 1 \\ 1 & 1 & 1 & \ldots & 1 & 0 \end{pmatrix}$$

Exact solution of the Problem 2 ($i,k = 1, 2, \ldots, n$) has a form

$$\tilde{\lambda}_k = \left(2\sin\frac{(2k-1)\pi}{2(2n+1)}\right)^{-2} - 1, \quad \tilde{x}_{i,k} = \sqrt{\frac{4}{2n+1}}\cos\frac{(2k-1)(2i-1)\pi}{2(2n+1)}\,.$$

Problem 3

Evaluate 8 minimum eigenvalues and their corresponding eigenvectors of the generalized algebraic eigenvalue problem (4.5) with band symmetric positive definite matrices *A* and *B*. Matrices *A* and *B* are obtained during the discretization of eigenvalue problem by finite elements method for the Laplace operator in rectangle one side of which is fixed. In this case matrices are block tri-diagonal:

$$A=\begin{pmatrix} A_1 & A_2 & & & 0 \\ A_2 & 2A_1 & A_2 & & \\ & \ddots & \ddots & \ddots & \\ & & A_2 & 2A_1 & A_2 \\ 0 & & & A_2 & 2A_1 \end{pmatrix}, \quad B=\begin{pmatrix} 2B_1 & B_1 & & & 0 \\ B_1 & 4B_1 & B_1 & & \\ & \ddots & \ddots & \ddots & \\ & & B_1 & 4B_1 & B_1 \\ 0 & & & B_1 & 4B_1 \end{pmatrix},$$

where

$$A_1=\begin{pmatrix} a_2 & f & & & 0 \\ f & 2a_2 & f & & \\ & \ddots & \ddots & \ddots & \\ & & f & 2a_2 & f \\ 0 & & & f & a_2 \end{pmatrix}, \quad A_2=\begin{pmatrix} g & a_1 & & & 0 \\ a_1 & 2g & a_1 & & \\ & \ddots & \ddots & \ddots & \\ & & a_1 & 2g & a_1 \\ 0 & & & a_1 & g \end{pmatrix},$$

$$B_1 = \begin{pmatrix} 2c_0 & c_0 & & & 0 \\ c_0 & 4c_0 & c_0 & & \\ & \ddots & \ddots & \ddots & \\ & & c_0 & 4c_0 & c_0 \\ 0 & & & c_0 & 2c_0 \end{pmatrix}, \qquad \begin{aligned} a_1 &= -(c_1 + c_2), \\ a_2 &= 2(c_1 + c_2), \\ f &= c_1 - 2c_2, \\ g &= c_2 - 2c_1, \end{aligned}$$

$$c_0 = \frac{1}{36 N_x N_y}, \quad c_1 = \frac{N_y}{6N_x}, \quad c_2 = \frac{N_x}{6N_y},$$

N_x, N_y – amounts of partitioning of the rectangular region in horizontal and vertical directions, respectively. The order of square blocks A_1 and A_2, B_1 equals to N_x+1, the number of such blocks in matrices A and B is $N_y \times N_y$. Thus, the order of matrices A and B is $n = (N_x+1)N_y$, while band half-width of these matrices is equal to $m = N_x+2$ (band width is $2m+1 = 2N_x+5$).

Exact solution of the Problem 3 has a form

$$\tilde{\lambda}_{kl} = \frac{1}{c_0}\left(\frac{c_2(1-s_k)}{2+s_k} + \frac{c_1(1-t_l)}{2+t_l} \right),$$

where $s_k = \cos\left(\frac{\pi k}{N_x}\right)$, $t_l = \cos\left(\frac{(l-0{,}5)\pi}{N_y}\right)$, $0 \le k \le N_x$, $1 \le l \le N_y$.

Problem 1 (with $n = 1000$) was solved on 16 processors by *QL*-algorithm. Elements of the matrix are given accurately and written in the file `A286_1000.tam`. In the solution problem the relative error in matrix elements specification is assigned to the least non-zero floating point number *masheps* ≈ 2,22×10^{-16}. A protocol describing the solving of the problem by means of Inpartool in the automatic mode is given below.

```
exact eigenvalues
 9.849886676738e-06 39944968634e-05 8.864839796918e-05
1.575962464286e-04
 2.462423159362e-04  3.545857333380e-04  4.826254314638e-04
6.303601491373e-04
 7.977884311878e-04  9.849086284661e-04

Time mp_esytri  7.794500e-01

 Order of matrix  = 1000
 Number of processor = 16
 Matrix elements error = 2.220446e-16
 FIRST 10 EIGENVALUES
 9.849886674095e-06  3.939944968403e-05  8.864839796700e-05
1.575962464263e-04
 2.462423159338e-04  3.545857333358e-04  4.826254314614e-04
6.303601491352e-04
 7.977884311857e-04  9.849086284639e-04

 ESTIMATION OF FIRST EIGENVALUES
       2.392412e-13        2.392412e-13 2.392413e-13
2.392413e-13
       2.392413e-13        2.392413e-13 2.392413e-13
2.392414e-13
       2.392414e-13        2.392415e-13
 ESTIMATION OF FIRST EIGENVECTORS
       8.096270e-09        8.096270e-09 4.857794e-09
3.469887e-09
       2.698837e-09        2.208176e-09 1.868494e-09
1.619398e-09
       1.428919e-09        1.278544e-09
```

Problem 2 (with n = 1000) was solved on 16 processors. Matrix elements are given accurately and written in the file `Amax1000.tam`. In the solution program the relative error in matrix elements' specification is assigned to the least non-zero floating point number $masheps \approx 2{,}22\times10^{-16}$. During the solving of problem the initial matrix is reduced to tri-diagonal symmetric matrix by means of sequence of two-sided Householder's transformations, and full AEVP for this matrix is solved by *QL*-algorithm. A protocol describing the solving of the problem by means of Inpartool in automatic mode is given below.

```
Calculation of all eigenvalues and eigenvectors

Time 4.270442e+00

Order of matrix = 1000
 Number of row in block = 20
 Number of processor = 16
 Matrix elements error = 2.220446e-16
 FIRST 10 EIGENVALUES
 -7.49999384e-01 -7.49997535e-01 -7.49994454e-01 -
7.49990140e-01 -7.49984594e-01
 -7.49977814e-01 -7.49969802e-01 -7.49960557e-01 -
7.49950078e-01 -7.49938366e-01
 LAST 10 EIGENVALUES
  1.12287869e+03 1.40285538e+03 1.80215054e+03
2.39961659e+03 3.35189425e+03
  5.00760334e+03 8.27847355e+03 1.62266882e+04
4.50757634e+04 4.05689204e+05
```

```
 ERROR ESTIMATION OF FIRST EIGENVALUES
  9.09616314e-11 9.09616314e-11 9.09616314e-11 9.09616314e-
11 9.09616314e-11
  9.09616314e-11 9.09616314e-11 9.09616314e-11 9.09616314e-
11 9.09616314e-11
 ERROR ESTIMATION OF LAST EIGENVALUES
  9.12111271e-11 9.12732944e-11 9.13619558e-11 9.14946199e-
11 9.17060680e-11
  9.20737093e-11 9.27999883e-11 9.45648465e-11 1.00970628e-
10 1.81042897e-10

 ERROR ESTIMATION OF FIRST EIGENVECTORS
  4.92025911e-05 4.92025911e-05 2.95211576e-05 2.10861262e-
05 1.63998857e-05
  1.34176502e-05 1.13529491e-05 9.83876786e-06 8.68081069e-
06 7.76658121e-06
 ERROR ESTIMATION OF LAST EIGENVECTORS
  4.47413616e-13 3.26003185e-13 2.28808073e-13 1.53137771e-
13 9.63018164e-14
  5.56098349e-14 2.83716511e-14 1.18976212e-14 3.49996065e-
15 5.02041457e-16
```

Problem 3 with $N_x = 319$, $N_y = 50$, i. e. $n = 16\,000$, and the band half-width $m = 321$ (total band width is equal to 643) was solved on 16 processors by method of iterations on the subspaces. The problem's initial data are written in the file `Eig16000.eedat`. A protocol describing solving of the problem in the automatic mode is given below.

```
P R O B L E M :
        Solving of Partial Generalized Eigenvalue Problem
        for Band Symmetric Matrices

INPUT PARAMETERS:
        order of matrices                = 16000
        bandwise of matrix A           = 643
        bandwise of matrix B             = 643
        maximal relative errors:
                    of matrix A elements = 0.000e+00
                 of matrix B elements  = 0.000e+00

        number of minimal eigenvalues
                            to calculate  = 8

Exact eigenvalues
    2.467604042554091e+00    1.233728821340764e+01
.222305254921369e+01
```

```
    3.209273672006724e+01    4.194729797545172e+01
6.170274648211132e+01
    6.181196631645549e+01    7.168165048730904e+01
9.130050516997107e+01
    1.012916602493531e+02    1.110559536766307e+02
1.213906838857423e+02
    2.011944685514055e+02    3.015383945680863e+02
1.312603680565959e+02
    2.110641527222591e+02
P r o c e s s   o f   r e s e a r c h   a n d   s o l u t i
o n   of the problem

M e t h o d :    Subspace Iterations
       matrix blocksize          = 10
       number of processors      = 16

pr#15:  (sbpldlt) returns 0   time=2.23644e+00
pr# 0:   conv= 0    Nit=16    time=1.67143e+00
pr# 5:  calc. of errors est.  time=2.78056e+00

problem solving:      total time = 3.43789e+01

R e s u l t s :    SOLUTION WAS CALCULATED
                   by 16 iterations  (mit=32)
    All calculated eigenvalues are minimal

Eigenvalues (calculated)   Estimates of Errors
    2.467604042574461e+00         4.493e-15
    1.233728821342248e+01         2.096e-12
    2.222305254923304e+01         2.674e-15
    3.209273672008100e+01         3.727e-12
    4.194729797546218e+01         5.368e-10
    6.170274648216704e+01         4.187e-07
    6.181196631647476e+01         6.440e-09
    7.168165048952910e+01         2.728e-06

#result=0
pr# 8: (sbpldlt-2) returns 0  time=2.61476e+00
```

An influence of errors in the initial data can be illustrated by results of solving of the following full standard AEVP.

Problem 2 was solved for $n = 2000$ for three different values of error in the specification of the problem's matrix elements: $\varepsilon_A = 0$, $\varepsilon_A = 10^{-10}$, $\varepsilon_A = 10^{-6}$ (remind that during the accurate specification of matrix elements – $\varepsilon_A = 0$ in program is replaced by $\varepsilon_A \approx 2{,}22 \cdot 10^{-16}$). Results of problem's solving for five minimum and five maximum eigenvalues are given in Table 4.6 (overall errors in the eigenvalues' evaluations) and in Table 4.7 (overall errors in the eigenvectors evaluations).

Large values of estimates for errors in evaluation of eigenvectors corresponding to minimum eigenvalues are caused by pathological proximity of these eigenvalues (see estimate (4.8)).

Table 4.6. Results for five minimum and five maximum eigenvalues

I	λ_i	$\Delta\lambda_i$ **with** $\varepsilon_A = 0$	$\Delta\lambda_i$ **with** $\varepsilon_A = 10^{-10}$	$\Delta\lambda_i$ **with** $\varepsilon_A = 10^{-6}$
1	–0.749999846	3.632e–10	2.0036e–07	2.0000004e–03
2	–0.749999383	3.632e–10	2.0036e–07	2.0000004e–03
3	–0.749998613	3.632e–10	2.0036e–07	2.0000004e–03
4	–0.749997534	3.632e–10	2.0036e–07	2.0000004e–03
5	–0.749996147	3.632e–10	2.0036e–07	2.0000004e–03
1996	20 023.1526	3.677e–10	2.0037e–07	2.0000004e–03
1997	33 100.0958	3.706e–10	2.0037e–07	2.0000004e–03
1998	64 877.0677	3.776e–10	2.0038e–07	2.0000004e–03
1999	180 215.707	4.032e–10	2.0040e–07	2.0000004e–03
2000	1 621 948.69	7.234e–10	2.0072e–07	2.0000007e–03

Table 4.7 Results for five minimum and five maximum eigenvectors

I	Δx_i **with** $\varepsilon_A = 0$	Δx_i **with** $\varepsilon_A = 10^{-10}$	Δx_i **with** $\varepsilon_A = 10^{-6}$
1	7.8550e–04	4.3330e–01	4.9786e–01
2	7.8550e–04	4.3330e–01	4.9786e–01
3	4.7130e–04	2.5998e–01	4.9786e–01
4	3.3664e–04	1.8570e–01	4.9786e–01
5	2.6183e–04	1.4443e–01	4.9786e–01
1996	5.5543e–14	3.0269e–11	3.0214e–07
1997	2.8338e–14	1.5322e–11	1.5294e–07
1998	1.1884e–14	6.3057e–12	6.2939e–08
1999	3.4961e–15	1.7375e–12	1.7340e–08
2000	5.0173e–16	1.3922e–13	1.3872e–09

4.6. Investigating and solving of systems of non-linear equations

4.6.1. Functional potentialities of Inpartool on investigating and solving of systems of non-linear equations

Systems of non-linear equations (SNE) often occur in the solving of application problems. These problems may either represent independent problems describing physical processes or may arise in the solving of more complicated mathematical problems at the intermediate stage of their solving. Due to the requirements of source and energy-saving the necessity arises in the mathematical modeling of processes and phenomena with high accuracy and reliability. This fact, in turn, leads to the solving of high-order problems. The speeding-up of the solving of such problems can be gained only by means of using parallel computations.

The solving of SNE is mainly carried out by iterative methods based (to some extent) on the Newton's method. As this takes place, the number of methods requires evaluation of the Jacoby matrix at each iteration and the subsequent solving of LAS.

Paralleling both of the Jacoby matrix evaluation and solving o LAS considerably speeds up a process of finding solutions of SNE. Iterative methods of another type involve iterative evaluation either of the Jacoby matrix or its inverse. In these cases paralleling of computations also considerably reduces time required for the solving of SNE.

The basic information about SNE is contained in vector-functions. By using considerably small increments of the vector-function one can evaluate the system's Jacoby matrix rather accurately. In SNE the real accuracy of the obtained solution (i.e. the reliability of the solution) may be evaluated in the neighborhood of the solution by norm of matrix inverse to the Jacoby matrix. All necessary evaluations can and should be paralleled and thereby the considerable reduction in the problem's execution times can be gained.

Let the system of n non-linear equations

$$f(x) = 0 \tag{4.9}$$

be given,

where $x = (x_1, x_2, \ldots, x_n)^T$, $f(x) = (f_1(x), f_2(x), \ldots, f_n(x))^T$ are vectors containing the solution to be sought and vector-function, respectively. If $H = \left\{ \frac{\partial f_i}{\partial x_j} \right\}_{i,j=1}^{n}$ is the Jacoby matrix of the system (4.9) (or some approximation to it), the iterative process for finding a solution which implements the Newton's method with given initial approximation $x^{(0)}$ can be written in the form

$$H^{(k)}w^{(k)} = -f\left(x^{(k)}\right) \qquad (4.10)$$

where $w^{(k)} = x^{(k+1)} - x^{(k)}$is the correction, and $H^{(k)} = H\left(x^{(k)}\right)$, *k*=0,1, ... is the number of iteration,

$$x^{(k+1)} = x^{(k)} + w^{(k)}. \qquad (4.11)$$

For the solving of such problems in addition to the initial approximation the following information is to be given: an additional region $D = \{a_i \le x_i \le b_i, \quad i = 1, 2, \ldots, n\}$, where the solution is sought and the required accuracy of the obtaining of approximations to the system's solution. In so doing, $x^{(0)} \in D$.

As one can see from formulas (4.10), (4.11), one should solve LAS of the form (4.10) on each iteration by evaluating therewith the value of vector-function and the Jacoby matrix.

When modeling realistic processes on computer by means of systems of non-linear equations one often happens to be concerned with approximately given initial data. The approximate nature of the data may be caused by:

1. errors in system's coefficients since they as results of various measurements cannot be accurate;
2. errors in functions specification; these errors are caused by the fact that non-linear equations are often some approximations to realistic non-linear equations; besides, very often realistic non-linear equations are approximated by more simple ones (which approximate realistic non-linear equations) in order to save arithmetic operations at each evaluation of functions;
3. employment of the numerical method for the solving of problem and rounding off numbers during computations;
4. the obtaining of system of equations by means of discretization of problems of various types by spatial variables

Because of this the approximate nature of the initial data should be taken into account when estimating the accuracy of solutions.

If instead of accurate system (4.9) the approximate system

$$\bar{f}(\bar{x}) = 0,$$

is to be solved for which the following inequality

$$\left\| f(v) - \bar{f}(v) \right\| \le \delta,$$

holds, where *v* is any vector and δ is an error in the vector-function's specification (i.e. SNE), then in case of satisfying the inequality

$\left\| \bar{f}\left(x^{(k)}\right) \right\| \le \frac{\varepsilon}{\left\| \bar{H}^{-1(k)} \right\|}$ in the iterative process of the form (4.10) the accuracy of the obtained solution is estimated by formula

$$\left\| \bar{x}^{(k)} - x \right\| \le \varepsilon + \left\| \bar{H}^{-1} \right\| \delta ,$$

where ε is accuracy of the solution to the problem's computer model specified by user.

On the basis of the foregoing, the intelligent software Inpartool has been created which is intended for the investigating and numerical solving of SNE with approximately given initial data in the specified region and within the required accuracy. Inpartool provides:

- the solving of SNE in the specified region and within the required accuracy;
- investigating of characteristics of the system;
- a choice of classes of methods and solution programs (both automatically and by user);
- dialog tools for the input of information;
- control over the information being input;
- issuing of recommendations as to making a decision in case of interruption;
- reliability of the obtained solution.

The software operates with knowledge obtained during the investigating of problems and on the basis of them makes a decision as to ways and methods for the evaluating of solution within the given accuracy.

Mathematical facilities of the intelligent software include:

- mathematical methods for the computer investigation of characteristic features of SNE;
- algorithms for the solving of SNE;
- tools for the evaluation of the solution and its reliability.
- The initial data are either read from a priori prepared file or entered by means of the keyboard with their further visualization on the display.
- Inpartool possesses the following distinctive characteristics:
- investigation of characteristics of SNE;
- possibility of the automatic choice of a class of methods;
- guarantee of the solution's reliability;
- possibility of work with the software without preliminary familiarization with it and without studying of instructions.

At the conceptual level Inpartool implements the basic principles of the information computing technology for the solving of problems which involves: formulation of problem in terms of the subject area language, investigation of characteristics of the problem being solved and automatic choice of algorithm depending on the revealed characteristics, synthesis of the solution program with taking into account mathematical and engineering characteristics of computer, the solving of problem and reliability analysis of the obtained results as well as dialog support and information referential provision for processes of formulating and solving the problem.

To solve SNE by means of Inpartool the following input data are required:

- order of system of non-linear equations;
- maximum number of iterations being performed;
- accuracy of the obtained solution;
- the initial data specification error;
- vector of initial approximations;
- arrays determining boundaries of the region.

In addition, to solve the SNE by means of Inpartool the user should enter a program for the evaluation of the vector-function. It can be read from the a priori prepared file or entered by means of the keyboard.

Inpartool provides a possibility of automatic mode of investigating and solving of problems under which the problem is investigated in computer without user's involvement and suitable algorithm for solving of problem is chosen on the basis of revealed characteristics of SNE and with taking into account engineering and mathematical characteristics of Inparcom-16, a processor topology is constructed, the initial data are distributed between processors in order required by algorithm, the problem is solved and reliability of the obtained results is estimated. By default, Inpartool constructs a topology from the optimum number of processors. However, the user can choose the required number of processors on his own. All information about process of solving the problem and obtained results are accumulated in protocol.

A possibility of investigating and solving problems in the dialog mode is also provided. In this case characteristics of SNE are investigated first of all. If the Jacoby matrix is symmetric the SNE is solved by Powell's method [4]. If the matrix is non-symmetric one can choose either globally convergent method (Burdakov's method [4]) or locally convergent methods.

In case if user has chosen a class of methods possessing local convergence he can choose one of the following methods:

- Newton's method,
- Dennis-More's method,
- Broyden's method.

Output data for the solving of SNE are the following:

- problem's execution time;
- order of system of non-linear equations;
- the number of iterations performed during the evaluation of solution;

- accuracy of the obtained solution;
- accuracy of the obtained solution with taking into accuracy of the initial data specification;
- array containing norms of the vector-function on the sequence of iterations;
- vector of the solution.

All data listed above are written info file of results.

4.6.2. Technology of investigating and solving systems of non-linear equations

4.6.2.1 Applying of Inpartool for the solving of SNE

Inpartool solves the following problems: finding of roots of one equation and roots of SNE with symmetric and non-symmetric Jacobi matrix.

The main window **«Systems of non-linearequations»** consists of the main menu and two panels. The left panel (passive) reflects a sequence of work stages and sub-stages which were already performed, being performed and will be performed. The view of right-hand panel of the window is shown in fig. 4.27.

To find roots of one equation the following work stages are to be carried out in succession in special windows:

- input of problem’s initial data;
- input of the right-hand side;
- start of the problem;
- obtaining of results of solving the problem.

To solve SNE a user should carry out the following successive work stages in special windows:

- input of problem’s initial data;
- input of the right-hand side;
- a choice of class of SNE;
- a choice of the solution method (in case of interactive solving of the problem)
- start of the problem;
- obtaining of results of solving the problem.

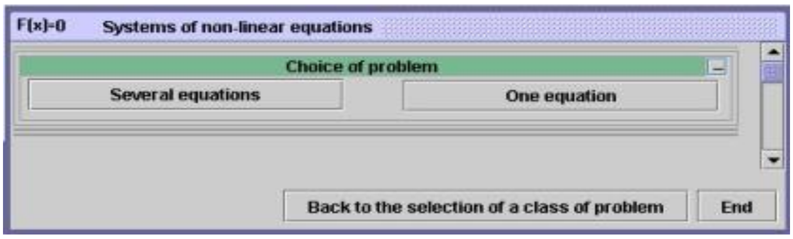

F(x)=0
Systems of non-linear equations
Choice of problem
Several equations
One equation
Back to the selection of a class of problem
End

Fig. 4.27. Right-hand panel of the systems of non-linear equations window

4.6.2.2 Specification of initial data for the solving of LAS

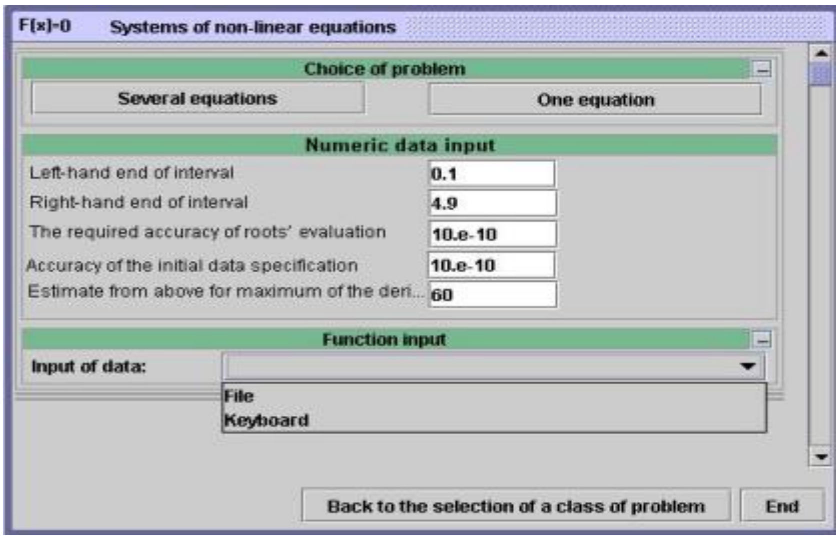
The following input data are required for the finding of roots of one equation by means of Inpartool:

- left-hand end of the interval;
- right-hand end of the interval;
- required accuracy of roots' evaluation;
- accuracy of the initial data specification;
- an estimate from above for maximum of the derivative’s module in the interval.

The solving of SNE by means of Inpartool requires the following input data:

- order of system of non-linear equations;
- the maximum number of the performed iterations;
- accuracy of the obtained solution;
- error in the function’s specification;
- vector of initial approximations;
- arrays determining boundaries of the region.

Numerical data required for finding roots of one equation can be entered by means of the keyboard, while function describing the non-linear equation can be either entered by means of the keyboard or read from the a priori prepared file (fig. 4.28).



F(x)=0 Systems of non-linear equations

Numeric data input

Left-hand end of interval 0.1

Right-hand end of interval 4.9

The required accuracy of roots’ evaluation 10.e-10

Accuracy of the initial data specification 10.e-10

Estimate from above for maximum of the deri... 60

Function input

Input of data:

Fig. 4.28. Input data source selection dialog

The initial data (numerical data and function) can be entered by the means of keyboard (fig. 4.29).

When inputting data from the file (fig. 4.30) the data are read from the binary file `data_nel` in the order implicated above.

When inputting data by means of the keyboard the above-indicated data are entered in special dialog data input window (fig. 4.31). The input of every element ends by pressing the <Enter> button. In so doing the information is entered and passage to the next data input field takes place.

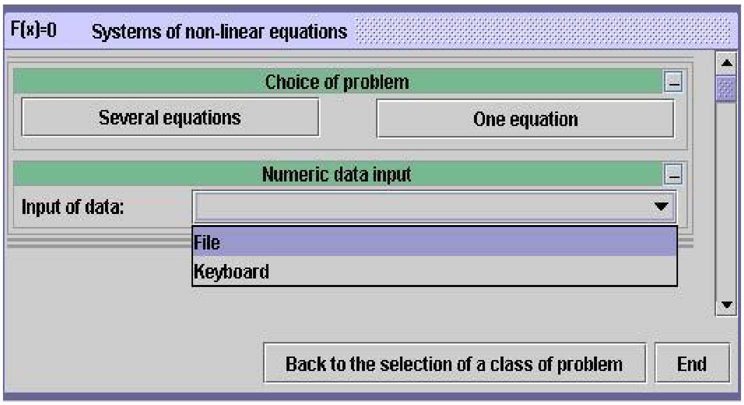
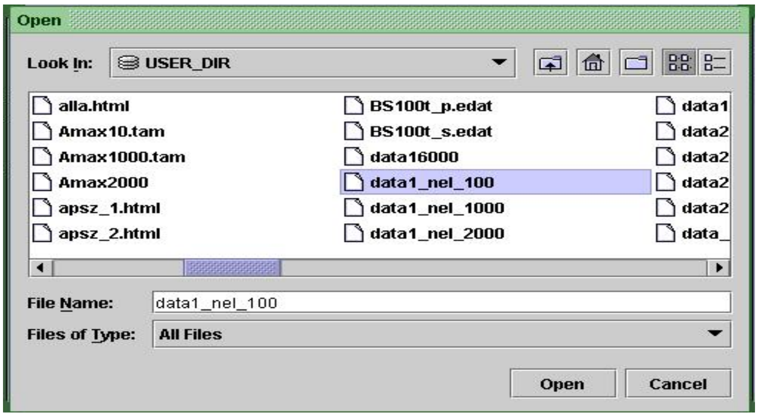
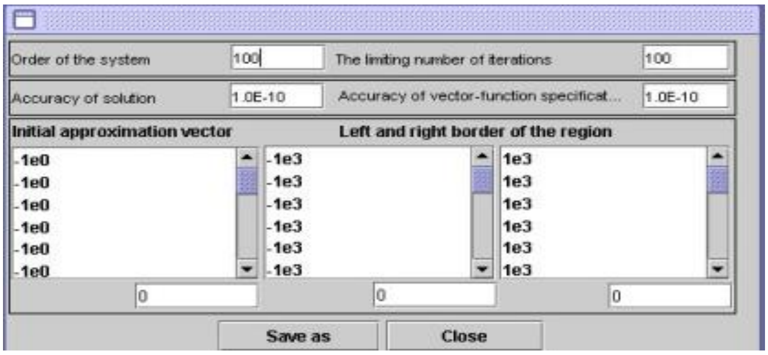


Fig. 4.29 Initial data entering



File Name: data1_nel_100

Fig. 4.30. Input data file selection


Order of the system: 100

The limiting number of iterations: 100

Accuracy of solution: 1.0E-10

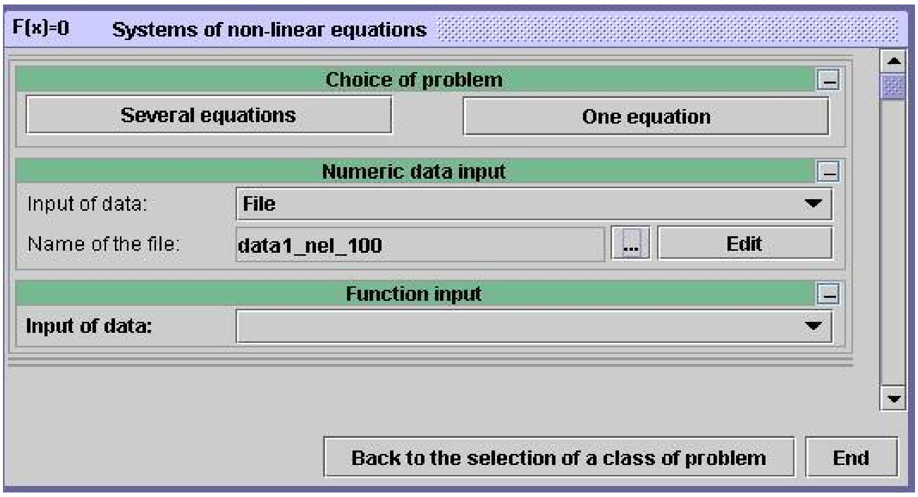
Accuracy of vector-function specificat...: 1.0E-10

Initial approximation vector	Left and right border of the region	
-1e0	-1e3	1e3
-1e0	-1e3	1e3
-1e0	-1e3	1e3
-1e0	-1e3	1e3
-1e0	-1e3	1e3
-1e0	-1e3	1e3
0	0	0

Fig. 4.31. Keyboard data input dialog

All information about current problem can be saved in a single file in «**File**» item of the main menu. To do this one should select «**Save as…**» menu item. This information can be used in next work sessions.

Previously entered data can be edited by pressing «**Edit**» button (fig. 4.32).



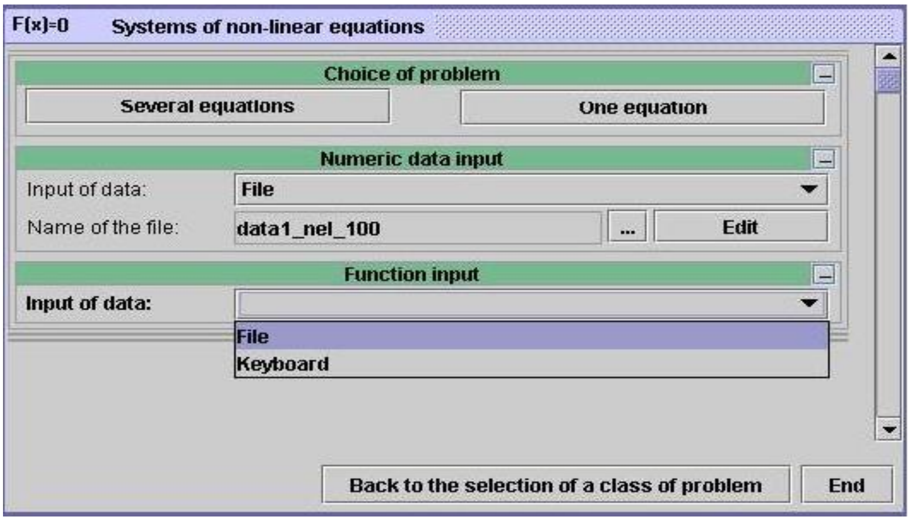
Input of data: File

Name of the file: data1_nel_100

Input of data:

Fig. 4.32. Data editing dialog

Besides, to solve SNE a vector-function is to be entered. It also can be entered either from file or by means of the keyboard (fig. 4.33).



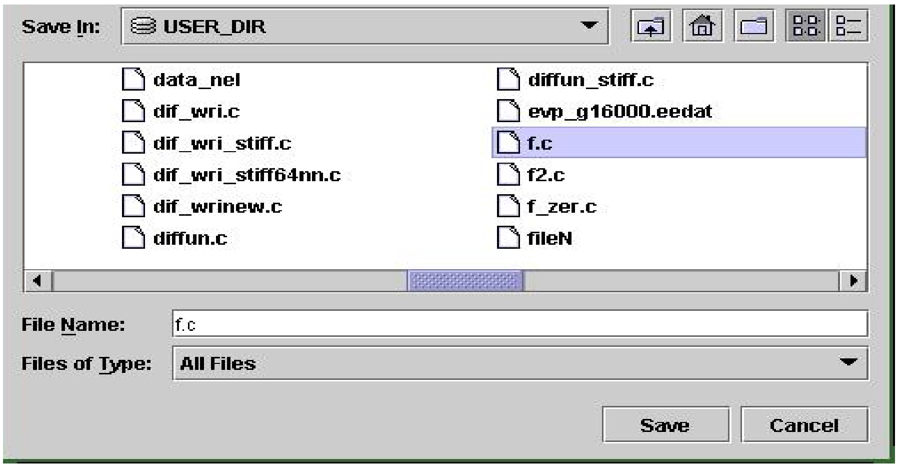
Input of data: File

Name of the file: data1_nel_100

Input of data:

Fig. 4.33. Vector-function source selection

If the user has chosen the input of functions from the file, a window opens containing a list of vector-functions (fig. 4.34) which were used during all work sessions. It remains only to choose the required vector-function (for example, f.c).



Save In: USER_DIR

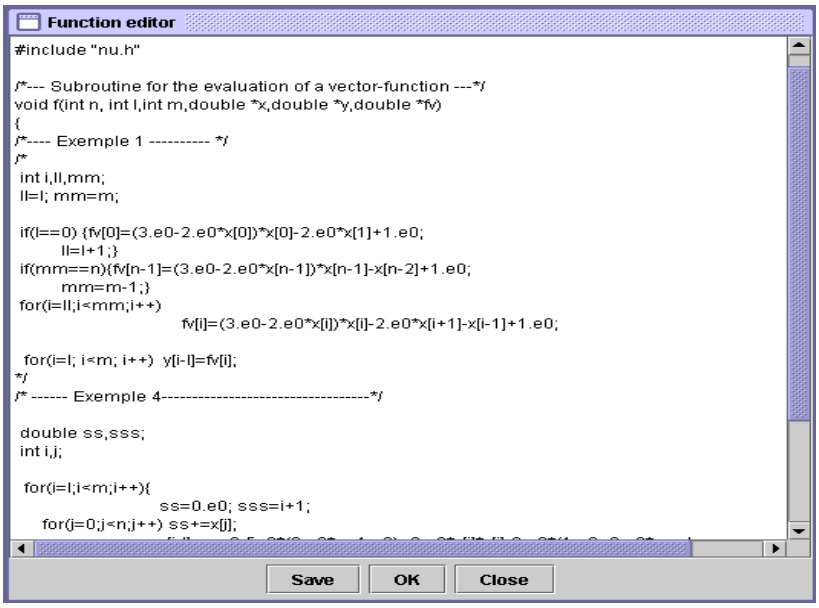
File Name: f.c

Files of Type: All Files

Fig. 4.34. Vector-function file selection

The function's input file contains a program written in C (fig. 4.35). If it is necessary to introduce modifications in the program one should press **«Edit»** button and then save these modifications by pressing **«Save»** button.

When inputting functions by means of the keyboard one should write a program in C for the evaluation of right-hand sides in the appeared window and save it in the file by pressing **«Save»** button.



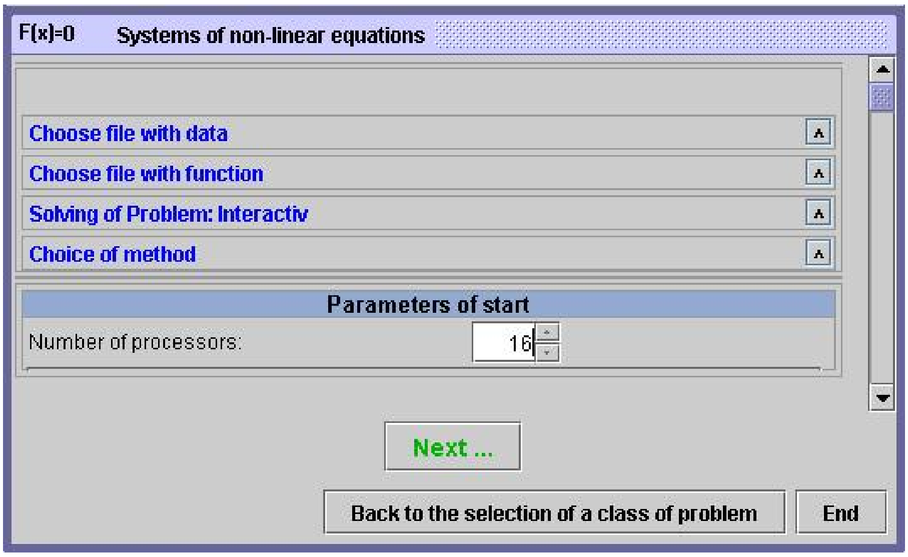
```
#include "nu.h"

/*--- Subroutine for the evaluation of a vector-function ---*/
void f(int n, int l,int m,double *x,double *y,double *fv)
{
/*---- Exemple 1 ---------- */
/*
 int i,ll,mm;
 ll=l; mm=m;

 if(l==0) {fv[0]=(3.e0-2.e0*x[0])*x[0]-2.e0*x[1]+1.e0;
        ll=l+1;}
 if(mm==n){fv[n-1]=(3.e0-2.e0*x[n-1])*x[n-1]-x[n-2]+1.e0;
        mm=m-1;}
 for(i=ll;i<mm;i++)
                        fv[i]=(3.e0-2.e0*x[i])*x[i]-2.e0*x[i+1]-x[i-1]+1.e0;

  for(i=l; i<m; i++)  y[i-l]=fv[i];
*/
/* ------ Exemple 4----------------------------------*/

 double ss,sss;
 int i,j;

  for(i=l;i<m;i++){
                       ss=0.e0; sss=i+1;
    for(j=0;j<n;j++) ss+=x[j];
```

Fig. 4.35. Function editor window

4.6.2.3 The solving of SNE

Inpartool proposes two modes of solving SNE: automatic and interactive. To run the problem a use should choose one of these modes by pressing a button of the appropriate mode in window which appears after the successful input of data.

In case of automatic mode of solving the problem Inpartool constructs a computer topology efficient for the solving of this problem; distributes initial data between processors in the order required by algorithm, solves the problem and estimates the reliability of the obtained results. By default, Inpartool constructs a topology from such number of processes which is optimum for the solving of the given problem. However, the user can choose the number of processors on his own (fig. 4.36).


F(x)=0
Systems of non-linear equations
Choose file with data
Choose file with function
Solving of Problem: Interactiv
Choice of method
Parameters of start
Next ...
Back to the selection of a class of problem
End

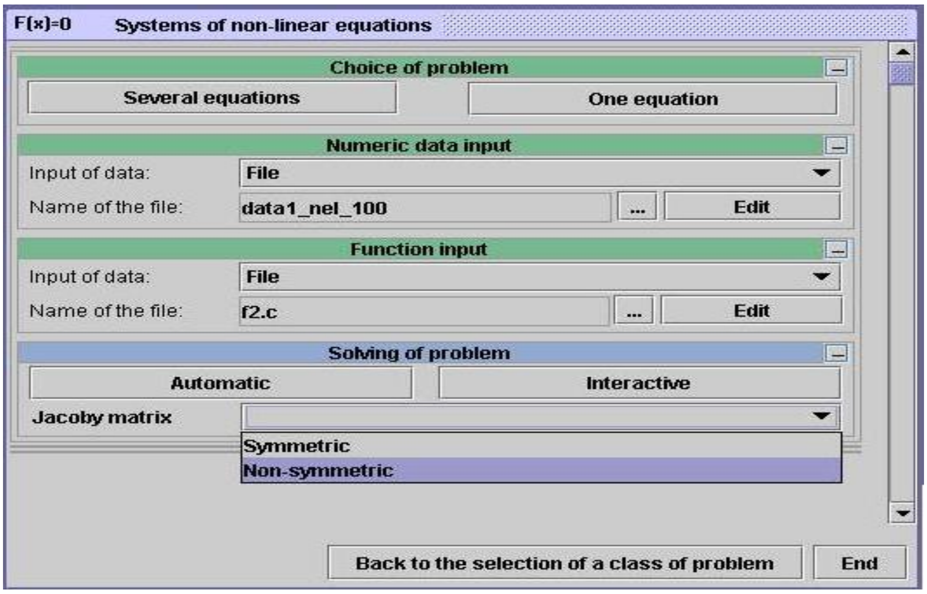
Number of processors: 16

Fig. 4.36. User-defined number of processors

In case of interactive solving of the problem prior to the beginning of problem's solving a user should indicate a type of the Jacoby matrix (symmetric or non-symmetric) (fig. 4.37).

If the matrix is symmetric the user is proposed to solve the problem by Powell method. If the Jacoby matrix is non-symmetric then the user should choose a class of methods first of all (fig. 4.38). If the class **«with global convergence»** has been chosen a user is proposed to solve the problem by Burdakov's method.

If the class **«with local convergence»** has been chosen the user is proposed to solve the problem by one of the following methods: the Newton's, Broyden's or Dennis-More's method's (fig. 4.39).



Numeric data input

Input of data: File

Name of the file: data1_nel_100

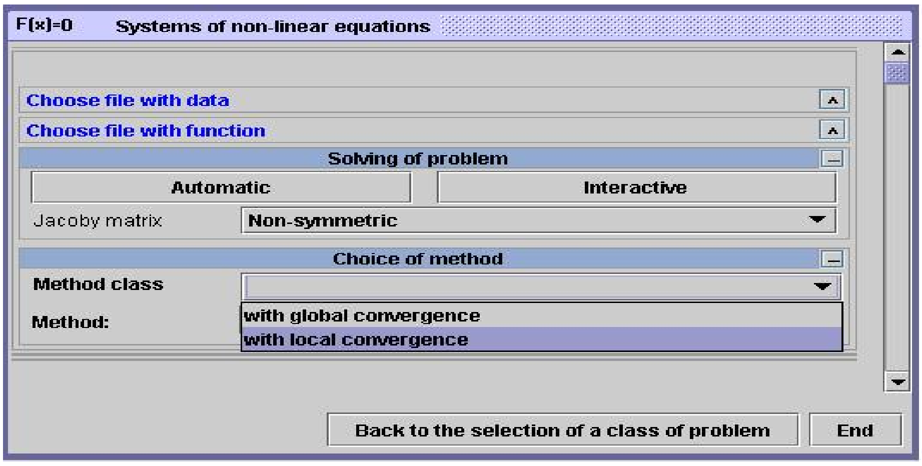
Function input

Input of data: File

Name of the file: f2.c

Jacoby matrix

Fig. 4.37. Type of the Jacoby matrix selection

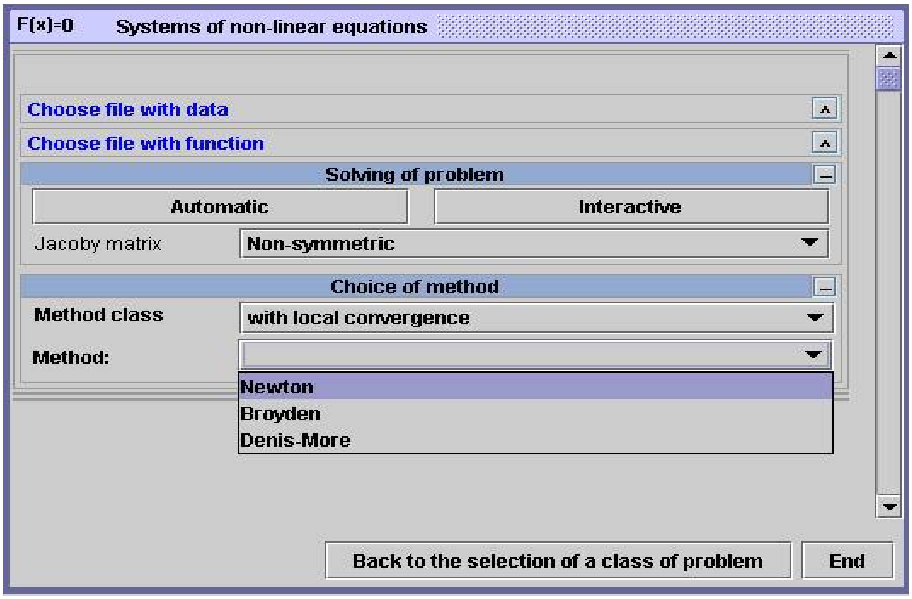


Jacoby matrix: Non-symmetric

Method class

Method:

Fig. 4.38. Local/global convergence selection



Jacoby matrix: Non-symmetric

Method class: with local convergence

Fig. 4.39. Method selection dialog

Differences between methods consist only in the way of construction of approximation to the Jacoby matrix. At each iteration of the Newton's method the Jacoby matrix is constructed by means of using difference approximations. The rest two methods construct the Jacoby matrix by the above described technique only once and further this matrix is improved by one of the iterative methods at each iteration.

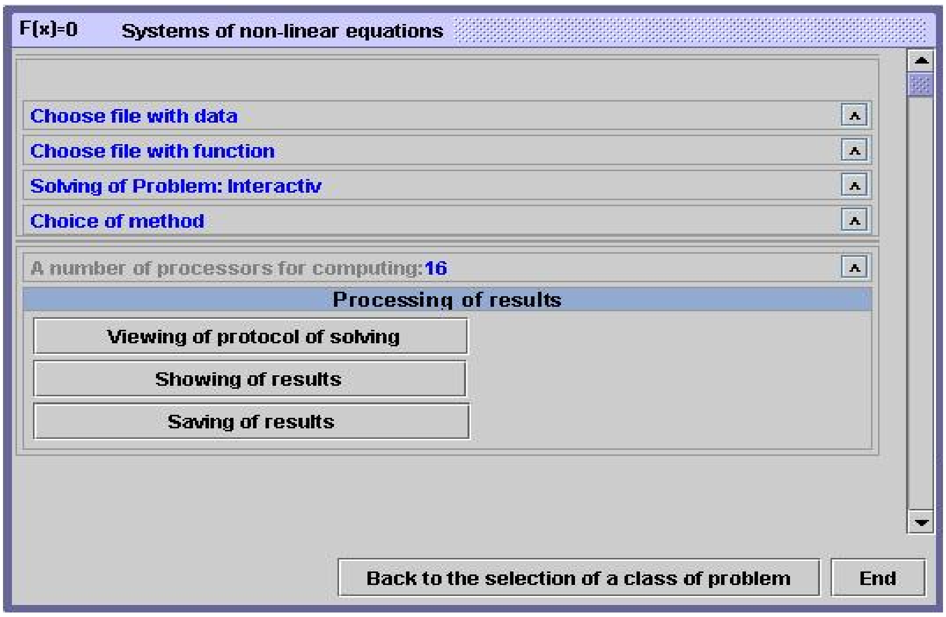
4.6.2.4 Results of solving SNE

Results of solving SNE by Inpartool are the following:

- solution of SNE;
- protocol describing a process of investigation and solving of SNE.

After completion of the computational process information about problem which was solved appears in the upper part of the right-hand panel. Besides, the pop-up window «**Processing of results**» appears, as well (fig. 4.40)

A solution of the problem is saved in the binary form. It can be browsed, saved or printed. A protocol describing a process of investigating and solving of SNE is presented in the text form; it contains parameters of the problem, methods being used, several control components of the solution, problem's run time, and the number of processors being used. The protocol can be saved and printed.



F(x)=0 Systems of non-linear equations
Choose file with data
Choose file with function
Solving of Problem: Interactiv
Choice of method
A number of processors for computing:16
Processing of results
Viewing of protocol of solving
Showing of results
Saving of results
Back to the selection of a class of problem
End

Fig. 4.40. Processing of results – pop-up window

4.6.2.5 Inpartool's diagnostics during the solving of SNE

When solving SNE Inpartool provides:

- referential information;
- Help-type messages at every stage of the user's work;
- problem's run-time diagnostics.

In the «**Help**» item of the main menu a user can get not only information about functional potentialities of Inpartool but also information about terminology being used related to systems of non-linear equations (fig. 4.41).

Having chosen the «**Glossary**» item in the «**Help**» submenu and then a term of interest from the list of terms a user can get its explanation (fig. 4.42).

Having clicked by right-hand mouse button on the title «**Systems of non-linear equations**» a user can familiarize himself with Inpartool'**s** functional potentialities as to the solving of problems belonging of the given class of problems as well as with order of work with Inpartool. In similarly the same manner a user can get appropriate short information at any stage of work by clicking the right-hand mouse button on any menu item or title of interest in the dialog window (fig. 4.43).

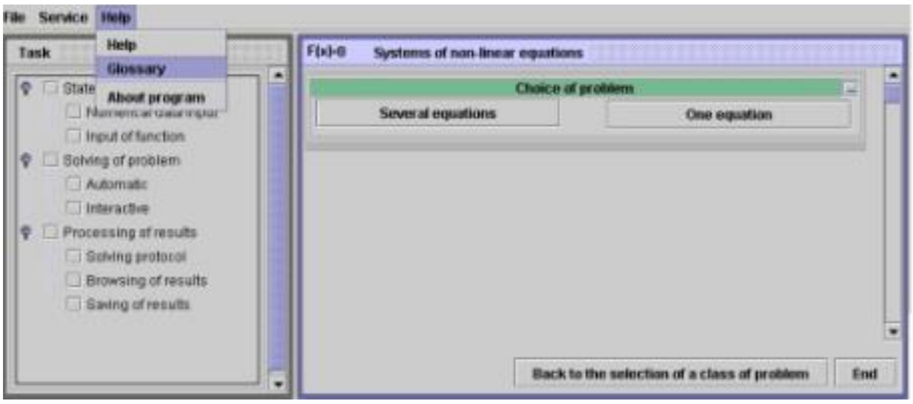
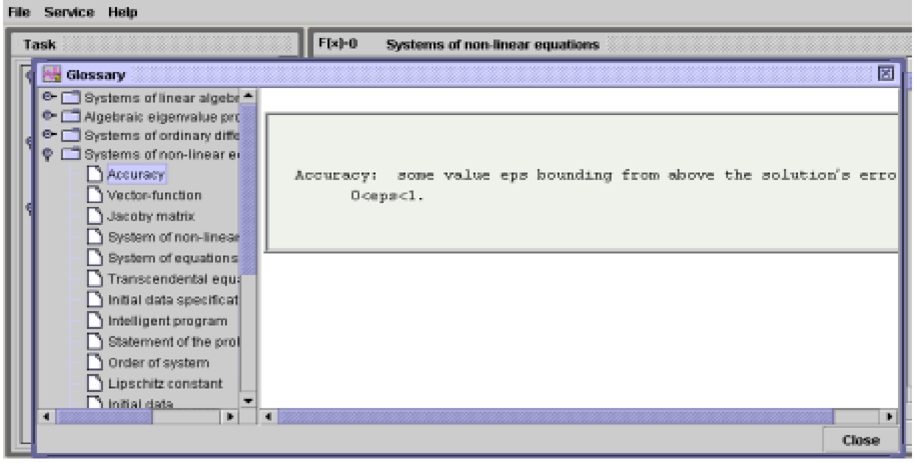
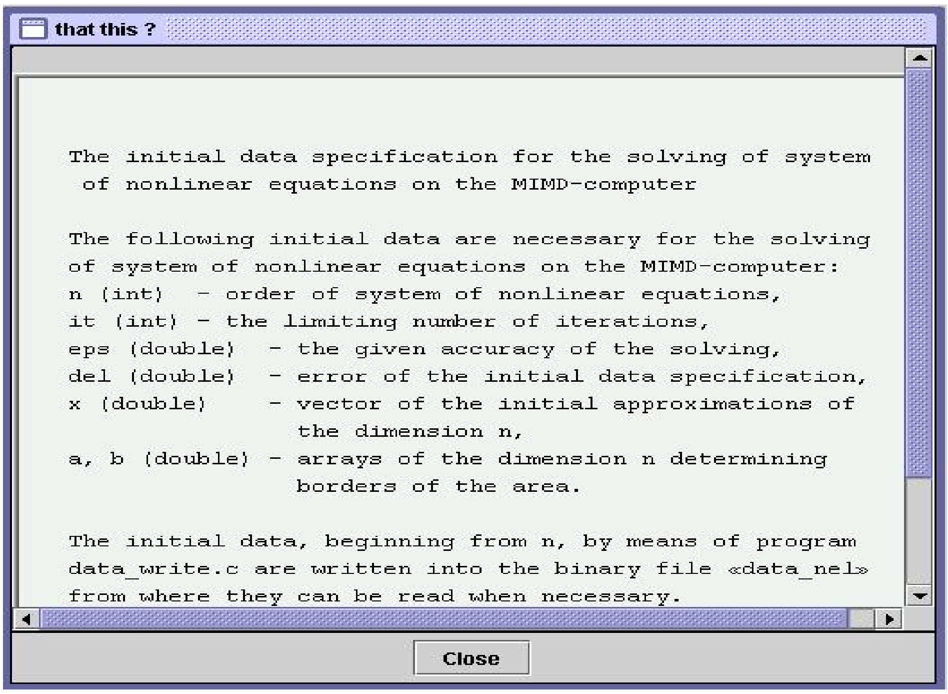


Fig. 4.41. Main help menu



```
Accuracy:  some value eps bounding from above the solution's erro
      0<eps<1.
```

Fig. 4.42. The glossary



The initial data specification for the solving of system of nonlinear equations on the MIMD-computer

```
The following initial data are necessary for the solving
of system of nonlinear equations on the MIMD-computer:
n (int)   - order of system of nonlinear equations,
it (int) - the limiting number of iterations,
eps (double)  - the given accuracy of the solving,
del (double)  - error of the initial data specification,
x (double)    - vector of the initial approximations of
                the dimension n,
a, b (double) - arrays of the dimension n determining
                borders of the area.
```

The initial data, beginning from n, by means of program data_write.c are written into the binary file «data_nel» from where they can be read when necessary.

Fig. 4.43. Glossary item window

4.6.3. Examples of solving systems of non-linear equations by means of Inpartool

Let us illustrate the using of Inpartool for the solving of SNE by the following two problems.

Problem 1

Solve SNE

$$\sum_{j=0}^{n-1} x_j - 0{,}5(3n+1) + 2x_i^2 - 2\left(1 + 2\frac{i}{n} + \left(\frac{i}{n}\right)^2\right) = 0, \quad i = 0,1,2,\ldots,n-1,$$

starting with the initial approximation $1 + 0{,}5\left(\frac{i+1}{n}\right)$ in the region

$$D = \{-1000 \le x_i \le 1000\}, \; i = 0,\, 1,\, 2, \ldots, n-1 \, ,$$

with restriction on the number of iterations it=100, given accuracy eps=$1{,}0\times10^{-10}$, error in the function's specification del=$1{,}0\times10^{-10}$.

Exact solution of the problem is $x_i = 1 + \frac{i+1}{n}$, $i = 0, 1, 2, \ldots, n-1$.

Problem 2
Solve SNE

$$(3-2x_i)x_i - 2x_{i+1} + 1 = 0, \qquad i = 0;$$

$$(3-2x_i)x_i - 2x_{i+1} - x_{i-1} + 1 = 0, \quad i = 1, 2, \ldots, n-2;$$

$$(3-2x_i)x_i - x_{i-1} + 1 = 0, \qquad i = n-1,$$

starting with vector of initial approximation all components of which are equal to −1, in the region $D = \{-1000 \le x_i \le 1000\}$, $i = 0, 1, 2, \ldots, n-1$, with restriction on the number of iterations it=100, given accuracy eps=$1{,}0\times10^{-10}$, error in the specification of the vector-function del=$1{,}0\times10^{-10}$.

Problem 1 for n= 100 was solved on 4 processors by various methods: the Burdakov's, Dennis-More's, Newton's, Broyden's and Powell's. This has resulted in the obtaining of solution (only the first, eleventh and twenty-first components of the solution's vector are presented)

```
Solution
1.0100000000e+00      1.1100000000e+00
1.2100000000e+00
```

Problem 2 for n= 100 was solved on 4 processors by various methods: the Burdakov's, Dennis-More's, Newton's, Broyden's and Powell's. This has resulted in the obtaining of solution (only the first, eleventh and twenty-first components of the solution's vector are presented)

```
Solution
-5.7076119297e-01      -7.0710677535e-01
-7.0710678119e-01
```

For the sake of verification of the obtaining of acceleration on Inparcom Problem 1 was solved with n= 2000 by Dennis-More's method. The following run times (in seconds) were obtained:

on one processor – time= 83.20;
on four processors – time= 6.46.

Thus, the obtained acceleration coefficient is equal to 83,2 / 6,46 = 12,88, while the coefficient of efficiency is equal to 0,8.

The influence of the approximate nature of the initial data can be illustrated by results presented below.

The Problem 2 was solved with n = 100, accuracy of the solution's evaluation eps=$1{,}0\times10^{-10}$, error in the vector-function's specification del= $1{,}0\times10^{-10}$ by Burdakov's method. The following results by have been obtained (only the first, eleventh and twenty-first components of the solution's vector are presented):

```
Accuracy of the obtained solution
```

```
del = 1.0100000000e-08
Solution
-5.7076119297e-01    -7.0710677535e-01
-7.0710678119e-01
```

If for all above values the vector-function's the specification error is equal to del=1,0×10^{-5} (i.e. 1,0×10^{-5} is added to every component of the vector-function) then employment of the Burdakov's method yields the following results:

```
Accuracy of the obtained solution
del  = 1.0000001000e-03
Solution
-5.7076443018e-01    -7.0711031088e-01
-7.0711031671e-001
```

4.7. Investigating and solving of systems od ordinary differential equations

4.7.1. Functional potentialities of Inpartool on investigating and soling systems of ordinary differential equations

Ordinary differential equations and systems of equations with initial conditions may arise at the intermediate stage of the solving of more complicated mathematical problems, for example, as a result of applying the finite elements method for the discretization only by special variables of initial boundary-value problems for systems of partial differential equations. Initial-value problems for systems of ordinary differential equations (SODE) can also arise in the describing of movements, processes and phenomena varying in time. Very often during the solving of problems related to the movement of guided objects a necessity arises to solve SODE faster then in the real-time, moreover the solving of these problems supposes multi-variant calculations. Such problems can be efficiently solved on parallel computers, in particular, on computers possessing MIMD-architecture.

When deriving ordinary differential equations one should abstract himself from some (regarded as secondary) characteristics of properties and processes. Such abstraction may result in the creation of either unstable or stable mathematical model of physically stable process or phenomenon. As a rule, at the next stage a problem arises concerning the construction of numerical solution of the mathematical model. Prior to the construction of numerical solution one should determine whether the mathematical model possesses asymptotically stable or, at least, stable by Lyapunov solution. Having become convinced on the basis of some investigations that mathematical model possesses a stable solution let us turn our attention to such algorithm for the construction of the numerical solution that under the condition of stability of the numerical method at any point of integration the required relative or absolute accuracy can be attained in the minimal run-time.

The solving of the above-mentioned problems in great part can be put on the computer. In so doing all investigations can be carried out in parallel with construction of the numerical solution by using the evaluated values both for the construction of solution and investigation of problem's characteristics.

Initial-value problems in the *n*-th order systems of ordinary differential equations in the interval $[t_0,T]$ will be considered in the form:

$$\frac{dv}{dt} = \varphi(t,v)\,, \tag{4.12}$$

$$v(t_0) = v_0\,, \tag{4.13}$$

where v is *n*-dimensional vector, while $\varphi\ (v)$ is *n*-dimensional function, i.e.

$$v = (v_1,\ v_2,...,v_{\mathrm{n}})^{\mathrm{T}},$$

$$\varphi(v) = (\varphi_1(t,\ v_1,v_2,...,v_{\mathrm{n}}),\ \varphi_2(t,\ v_1,v_2,...,v_{\mathrm{n}}),...,\ \varphi_{\mathrm{n}}(t,\ v_1,v_2,...,v_{\mathrm{n}}))^{\mathrm{T}}\ ,\ t\in[t_0,\ T].$$

Sufficient conditions for the existence and uniqueness of solution to the problem (4.12), (4.13) are the following:

- continuity of components of the right-hand side in the rectangle $D = \{t_0 - a \le t \le t_0 + a,\ v_0 - b \le v \le v_0 + b\}$;
- holding of the Lipschitz condition for all functions φ_j (j = 1, 2, 3, … – the number of vector's component), by all arguments $\left|\varphi_j(t,u_1) - \varphi_j(t,u_2)\right| \le L_j\left|u_1 - u_2\right|,\qquad j = 1, 2, \dots , n,$

in the rectangle *D,* where L_j are the Lipschitz constants.

Under these conditions there exists a unique solution of the problem (4.12), (4.13) on the interval to $t_0 - N \le t \le t_0 + N$, where

$$N < \min_{1\le j\le n}\left(a, \frac{b}{M_j}, \frac{1}{L_j}\right).$$

$$M_j = \max \mid \varphi_j\,(t,\ u) \mid \text{ in } D,$$

Further we will deal with SODE possessing asymptotically stable solutions. In addition, without loss of generality and for the sake of simplifying formulas we will consider the initial-value problem of the form:

$$\frac{dv}{dt} = \varphi(v)\,, \tag{4.14}$$

$$v(t_0) = v_0\,, \tag{4.15}$$

Since the problem (4.12), (4.13) can be reduced to the problem (4.14), (4.15) by introducing both additional differential equation for the time with right-hand side equal to 1 and additional initial condition equal to t_0.

When modeling realistic processes on computer by means of SODE the number of difficulties arises, in particular, one have to deal with problems possessing approximate initial data.

Approximate nature of the initial data may be caused by the following factors:

1. errors in specification of the initial data since the initial conditions being the results of various measurements are inaccurate;
2. errors in the specification of right-hand side; these errors are caused by the fact the right-hand side is some approximation to right-hand side of the realistic differential equation; in particular, this takes place during formulating of problems when neglecting some facts and phenomena which are either unimportant from the viewpoint of user or have a little influence on the development of the process being described; besides the right-hand side is often approximated by more simple functions for the sake of economy in the number of arithmetic operations at each step of integration;
3. in some cases the solution is evaluated from the equivalent equation explicitly unresolved with respect to the solution being sought; then in the process of integration an approximate solution of implicit equation is used instead of the exact solution;
4. applying of the numerical (discrete) method of integration and rounding off numbers during computations;
5. discretization of dynamical problems of various forms by special variables.

The above-mentioned difficulties considerably affect the accuracy estimate for the obtained solution.

If instead of the accurate problem (4.14), (4.15) the problem with approximately given initial data

$$\frac{du}{dt} = f(u), \tag{4.16}$$

$$u(t_0) = u_0 \tag{4.17}$$

is solved with

$$\|v_0 - u_0\| \le \delta, \ \ \|\varphi(w) - f(w)\| \le \Delta \tag{4.18}$$

for the arbitrary functions *w(t)*, then on the obtaining of a solution to the problem (4.16), (4.17) by any numerical method within the accuracy of ε, i.e.

on the attaining of inequality $\|y_{k+1} - u(t_{k+1})\| \le \varepsilon$, where y_{k+1} is the numerical solution to the problem (4.16), 6.(6) at the point t_{k+1}, the error in the solution to problem (4.14), (4.15) is estimated be formula

$$\|y_{k+1} - v(t_k)\| \le \varepsilon + \delta + (t_{k+1} - t_0)\Delta$$

within the accuracy of values of higher order in smallness.

On the basis of the foregoing an intelligent software Inpartool has been created for the investigating and numerical solving of initial-value problems in SODE with approximately given initial data on the given interval and within the required accuracy.

SODE can be common (non-stiff) or stiff. A system is considered to be stiff if the following condition holds:

$$\max_{1 \le i \le n} \mathrm{Re}(-\lambda_i) \times T \ge C,$$

where λ_i are eigenvalues of the Jacoby matrix, n is order of the system and C is a constant depending on performance of computer used for the solving of problem.

As it is known, for mast problems of the type (4.12), (4.13) or (4.16), (4.17) it is impossible to find analytic solution and that's why numerical integration methods are employed for the search of solution. Numerical methods are based on the discretization of differential problems by difference ones. The available well-known methods for the discretization of system (4.12) lead to different classes of methods (one-step and multi-step, explicit and implicit) possessing different orders of accuracy of difference schemes. Problems arise related to the attaining of solution's accuracy provided that conditions of stable computations are fulfilled.

Inpartool provides:

- the solving of initial-value problems on the given interval and within the required accuracy;
- investigation of problem's characteristics (common or stiff);
- a choice of class of methods and solution programs (both automatically and interactively);
- control over integration step size;
- dialog tools for the input of information;
- control of the input data;
- issuing of recommendations as to making decisions in case of interruption;
- reliability of the obtained solutions.

The software works with knowledge obtained during investigation of problems and on the basis of them makes decisions as to ways and methods for the evaluating of solution within the given accuracy.

Mathematical facilities of the intelligent software include:

- mathematical methods for the computer investigation of characteristics of systems;
- algorithms for the solving of systems;
- means for controlling the integration step size based on the accuracy and stability;
- tools for the evaluating of solution and analyzing its error.

The initial data are either read from the a priori prepared file or entered by means of the keyboard with their further displaying on the screen. Both the solution and estimate for the local Lipschitz constant at output points are written to the file.

Inpartool exhibits the following distinctive characteristics:

- investigation of characteristics of SODE;
- possibility of automatic choice of class of methods;
- control over the integration step size based on requirements of accuracy and stability;
- guarantee of the solution's reliability;
- work with software without preliminary familiarization with it as well as without studying of instructions.

At the conceptual level Inpartool implements the following principles of the information computing technology for the solving of problems which involves: formulation of problem in terms of the subject area language, investigation of characteristics of the problem being solved and automatic choice of algorithm depending on the revealed characteristics, syntheses of the solution program with taking into account mathematical and engineering characteristics of computer, the solving of problem and reliability analysis of the obtained results as well as dialog support and information referential provision for processes of formulating and solving the problem.

Inpartool provides a possibility of automatic investigating and solving of problems under which characteristics of SODE are investigated in computer without user's involvement, and on the basis of revealed characteristics as well as with taking into account engineering and mathematical potentialities of Inpartool-16, Inpartool determines whether the problem is common or stiff, a suitable algorithm for the solving of problem is chosen, a processor topology is constructed, the initial data are distributed between processors in order required by algorithm, the problem is solved and reliability of the obtained results is estimated. By default, Inpartool constructs a topology from the optimum number of processors. However, the user can choose the required number of processors on his own. All information about the process of solving the problem and results are accumulated in protocol.

A possibility of investigation and solving of problems in the interactive mode is also provided. In this case characteristics of SODE are investigated first of all and information is issued whether the system is stiff or not. If system is common (non-stiff) the user can choose a class of methods with attaining of either global or local accuracy as well as a method for solving the problem from the list of methods being proposed.

In case if a class of methods with attaining the global accuracy has been chosen the user can solve the problem by explicit Runge-Kutta-type 1-st order method. In case if a class of methods with attaining the local accuracy has been chosen the user can choose the solution method from the following list:

- Adams' methods of the order up to 12 [4];
- 4-th order Runge-Kutta methods;
- 5(6)-th order Runge-Kutta methods;
- Euler-Cauchy method.

If the system is stiff the user is proposed to choose a method for the solving of problem from the following list:

- Gear's methods of order up to 5 [4];
- Rosenbrock method [4].

4.7.2. Technology of investigating and solving systems of ordinary differential equations

4.7.2.1 Applying of Inpartool for the solving of SODE

Inpartool solves common (non-stiff) and stiff systems of ordinary differential equations. The main window **«Systems of ordinary differential equations»** is depicted in fig. 4.44. To solve the problem a user should perform:

- input of problem's initial data;
- input of right-hand side;
- a choice of either interactive or automatic way of solving the problem;
- a choice of class and methods for the solving of problem (in case of interactive solving of problem);
- run of the problem;
- obtaining of results of solving the problem.

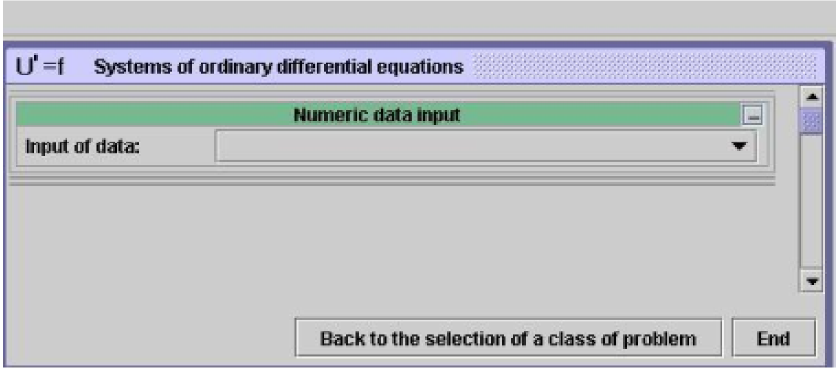

U' =f Systems of ordinary differential equations
Numeric data input
Input of data:
Back to the selection of a class of problem
End

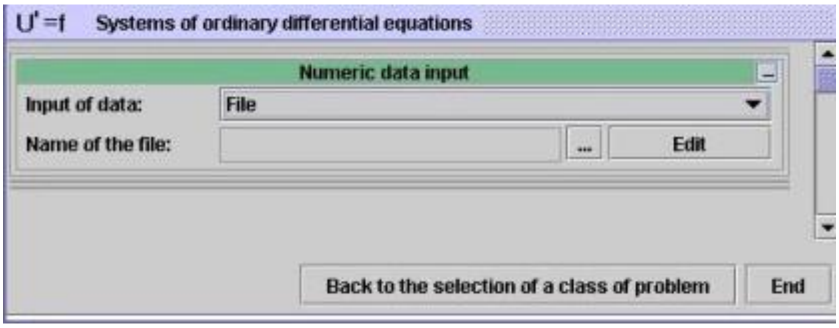
Fig. 4.44. Solving systems of ordinary differential equations main window

4.7.2.2 Specification of initial data for the solving of SODE

The following input data are required for the solving of SODE by means of Inpartool:

- order of SODE;
- the number of output points;
- Starting point of the integration interval (for the first call of function) or a point attained during the integration;
- final point of the integration interval;
- the required accuracy in the solution;
- error in the initial conditions' specification;
- error in the right-hand sides' specification;
- accuracy of the obtained solution with taking into account approximate nature of the initial data;
- array containing either initial values of solution's components (for the first call of the function) or values of solution at the point attained in the process of integration;
- array containing output points of the solution.

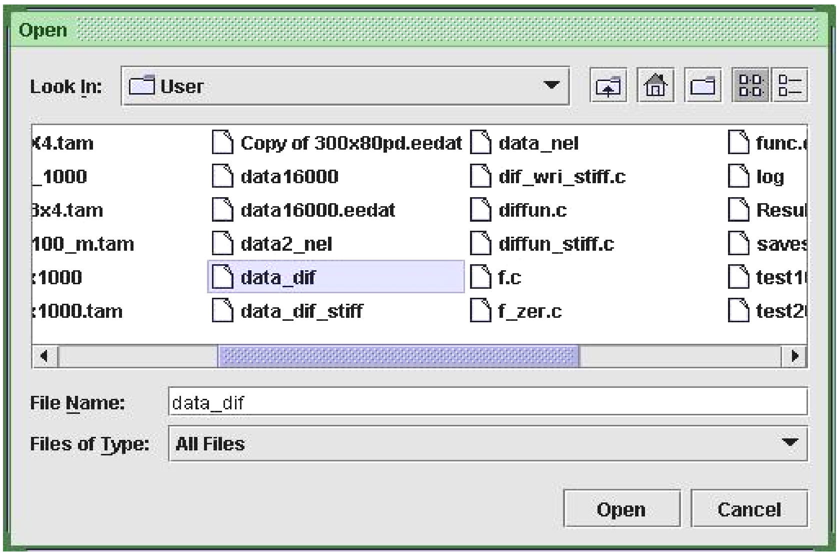
The input data for the solving of SODE can be either read from file or entered by means of the keyboard (fig. 4.45).


Input of data: File

U' =f Systems of ordinary differential equations
Numeric data input
Name of the file:
...
Edit
Back to the selection of a class of problem
End

Fig. 4.45. Input data source selection window

When inputting data from the file (fig. 4.46) they are read from the binary file `data_dif` in the above-indicated order.



User

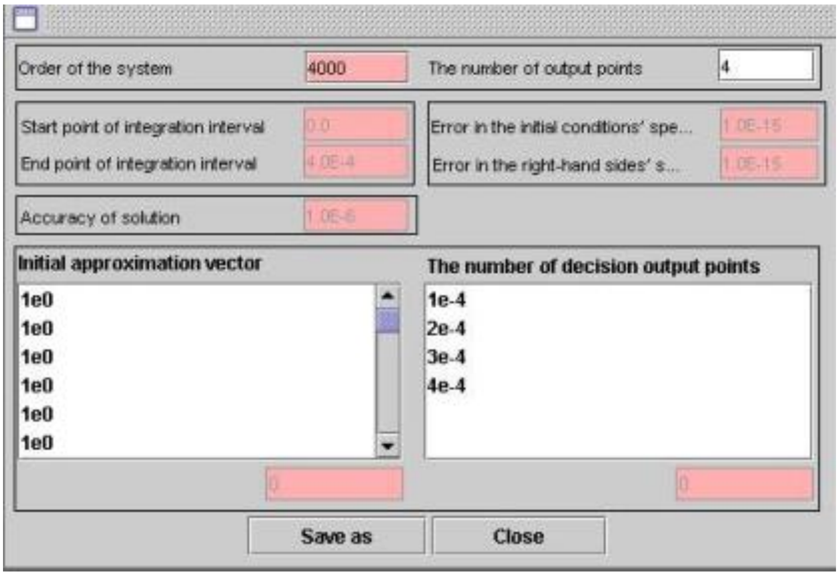
x4.tam	Copy of 300x80pd.eedat	data_nel	func.(
_1000	data16000	dif_wri_stiff.c	log
3x4.tam	data16000.eedat	diffun.c	Resul
100_m.tam	data2_nel	diffun_stiff.c	saves
:1000	data_dif	f.c	test1(
:1000.tam	data_dif_stiff	f_zer.c	test2(

File Name: data_dif

Files of Type: All Files

Fig. 4.46. Input data file selection

When inputting data by means of the keyboard the above-indicated data are entered in special dialog data input window (fig. 4.47). The input of every element ends by pressing the <Enter> button. In so doing the information is entered and passage to the next data input field takes place.



Order of the system 4000

The number of output points 4

Start point of integration interval 0.0

End point of integration interval 4.0E-4

Error in the initial conditions' spe... 1.0E-15

Error in the right-hand sides' s... 1.0E-15

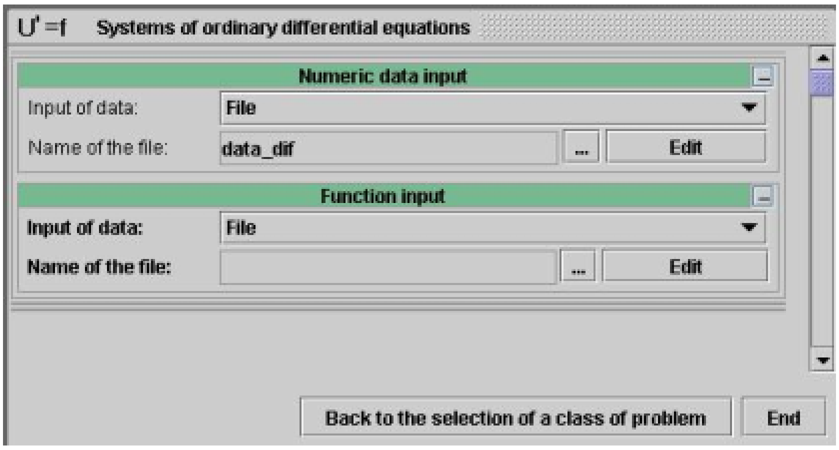
Accuracy of solution 1.0E-6

Initial approximation vector	The number of decision output points
1e0	1e-4
1e0	2e-4
1e0	3e-4
1e0	4e-4
1e0	
1e0	
0	0

Fig. 4.47. Manual data input window

The already entered data can be edited by pressing **«Edit»** button. The current values of the initial data can be saved in file. To do this a user should choose **«Save as…»** in the **«File»** item of the main menu. This information may be used in the next work sessions.

Besides, for the solving of SODE by means of Inpartool the user should input a vector-function for the evaluation of right-hand sides of the system. It also can be entered either from file or by means of the keyboard (fig. 4.48).


U' =f Systems of ordinary differential equations
Numeric data input
...
Edit
Function input
...
Edit
Back to the selection of a class of problem
End

Input of data: File

Name of the file: data_dif

Input of data: File

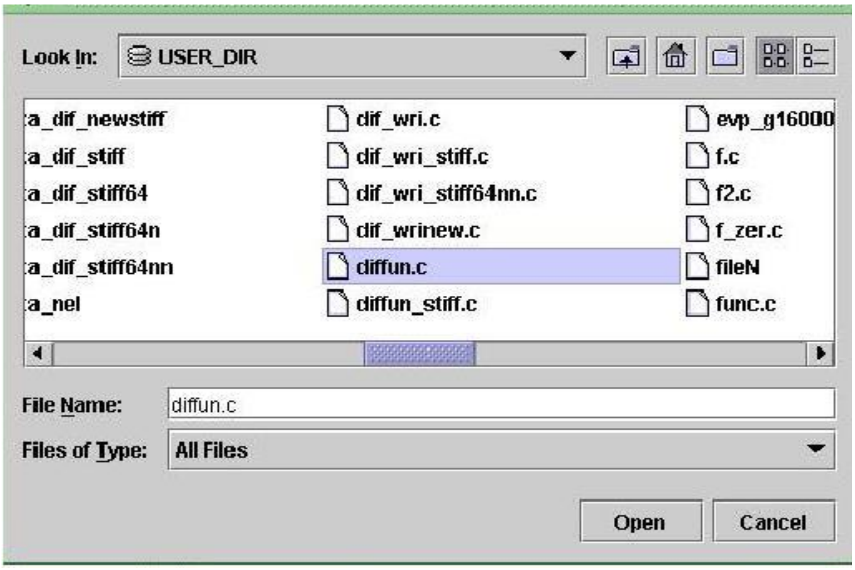
Name of the file:

Fig. 4.48. Function input selection

When inputting functions by means of the keyboard in the appeared window one should write a program in C for the evaluation of right-hand sides and then save it in the file by pressing the **«Save»** button. When user has chosen input of functions from the file, a window opens containing a list of functions (fig. 4.49). It remains only to choose the required function (for example, `diffun.c`).

The file contains a program written in C (fig. 4.50). In case of necessity the user can modify the program and save all changes by pressing **«Save»** button.

When inputting functions by means of the keyboard the user should in the appeared window write a program in C for the evaluation of right-hand sides and then save it in the file with indicated name.

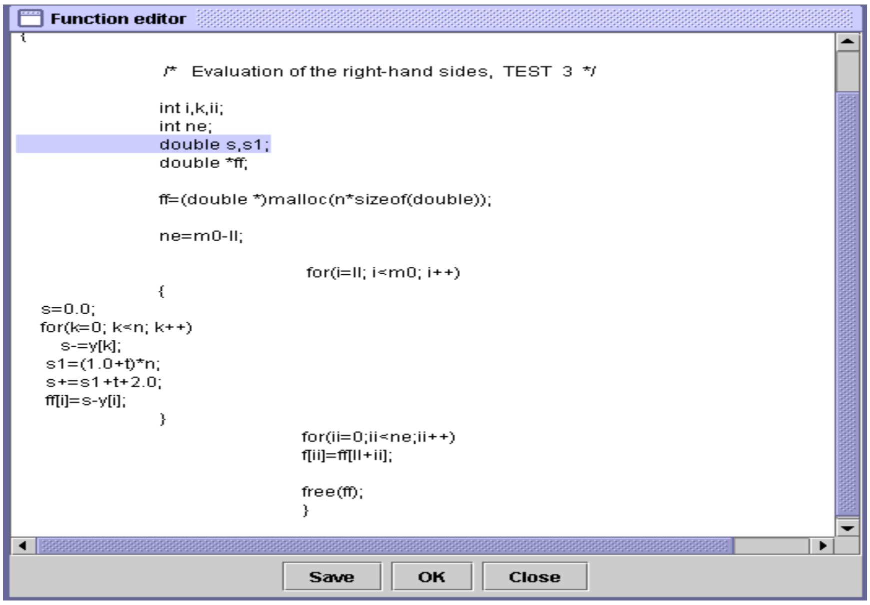


Look In: USER_DIR

File Name: diffun.c

Files of Type: All Files

Fig. 4.49. Function file selection window



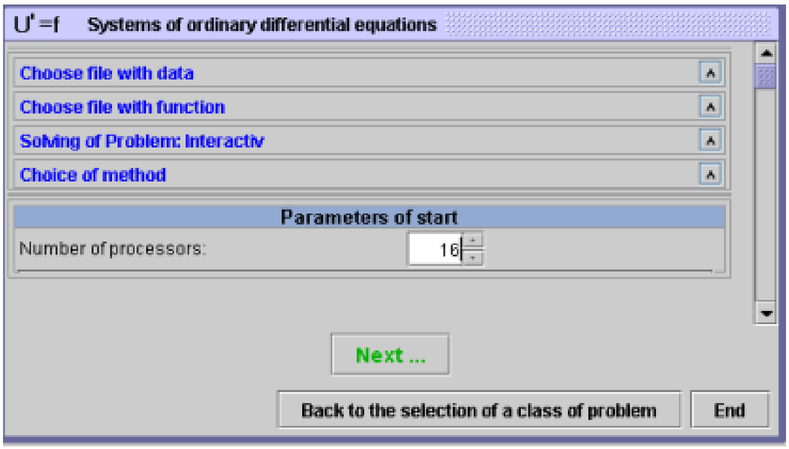
```
{

                    /*   Evaluation of the right-hand sides,  TEST  3  */

                    int i,k,ii;
                    int ne;
                    double s,s1;
                    double *ff;

                    ff=(double *)malloc(n*sizeof(double));

                    ne=m0-ll;

                                        for(i=ll; i<m0; i++)
                    {
  s=0.0;
  for(k=0; k<n; k++)
     s-=y[k];
   s1=(1.0+t)*n;
   s+=s1+t+2.0;
   ff[i]=s-y[i];
                    }
                                       for(ii=0;ii<ne;ii++)
                                       f[ii]=ff[ll+ii];

                                       free(ff);
                                       }
```

Fig. 4.50. Function editor window

4.7.2.3 *The solving of SODE in Inpartool*

Inpartool proposes two modes of solving SODE: automatic and interactive. To run the problem a user should choose one of these modes by pressing a button of the appropriate mode in window which appears after the successful input of data.

In case of automatic mode of solving the problem Inpartool investigates the problem and on the basis of revealed characteristics of SODE and according to engineering and mathematical potentialities of Inparcom-16 determines whether the system is common (non-stiff) or stiff; an appropriate algorithm for solving the problem is chosen; an efficient processor topology is constructed; the initial data distributed between processors in order required by algorithm; the problem is solved and reliability of the obtained results is estimated. By default Inpartool constructs a topology from such number of processes which is optimum for the given problem. However, the user can choose the number of processors on his own (fig. 4.51). All the information about process of solving the problem is accumulated in the protocol.


U' =f Systems of ordinary differential equations
Choose file with data
Choose file with function
Solving of Problem: Interactiv
Choice of method
Parameters of start
Next ...
Back to the selection of a class of problem
End

Number of processors: 16

Fig. 4.51. User-defined number of processors

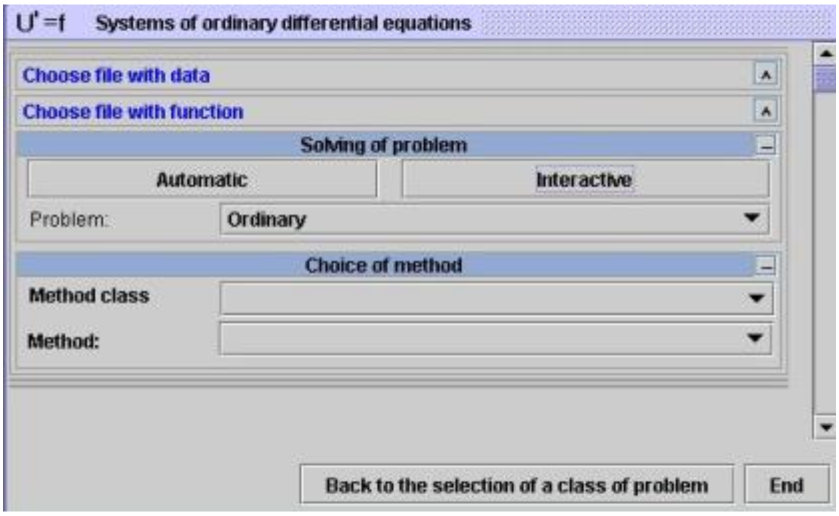
In case of interactive solving of the problem, characteristic of SODE are investigated first of all and user is informed whether it is stiff or common (non-stiff). If system is common (non-stiff) the user can choose a class of methods either **«with attaining of global accuracy»** or **«with attaining of local accuracy** » and then choose a solution method from the list being proposed (fig. 4.52, 6. 4.53).

In case if the class of solution methods **«with attaining of global accuracy»** is chosen, the user can solve the problem by explicit 1-st order Runge-Kutta-type method. If the class of solution methods **«with attaining of local accuracy»** is chosen the user can solve a method from the following list:

- Adams' methods of order up to 12;
- 4-th order Runge-Kutta methods;
- 5(6)-th order Runge-Kutta methods;
- Euler-Cauchy method.

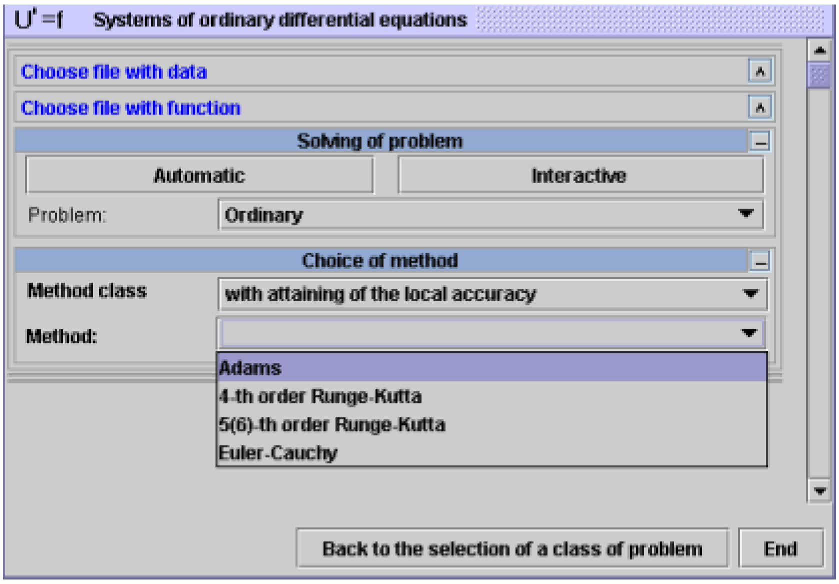
If the system is stiff the user is proposed to choose the solution method from the following list:

- Gear's methods of order up to 5;
- Rosenbrock method.


U' =f Systems of ordinary differential equations
Choose file with data
Choose file with function
Solving of problem
Automatic
Interactive
Choice of method
Method class
Method:
Back to the selection of a class of problem
End

Problem: Ordinary

Fig. 4.52. Interactive solving parameter window


U' =f Systems of ordinary differential equations
Choose file with data
Choose file with function
Solving of problem
Automatic
Interactive
Choice of method
Method:
Adams
4-th order Runge-Kutta
5(6)-th order Runge-Kutta
Euler-Cauchy
Back to the selection of a class of problem
End

Problem: Ordinary

Method class: with attaining of the local accuracy

Fig. 4.53. Solution method selection list

Under the interactive mode of solving the problem the number of processors can be chosen. A process of solving the problem is started after pressing «**Further…**» button.

Similarly as in the case of automatic mode, the problem is solved together with estimating the reliability of the obtained results.

4.7.2.4 *Results of solving SODE*

Results of solving SODE by **Inpartool** are the following:

- solution of SODE;
- protocol describing a process of investigation and solving of SODE.

After completion of the computational process the information about problem which was solved appears in the upper part of the right-hand panel. Besides, the pop-up window **«Processing of results»** appears, as well (fig. 6.11)

A solution of the problem is saved in the binary form. It can be browsed, saved or printed.

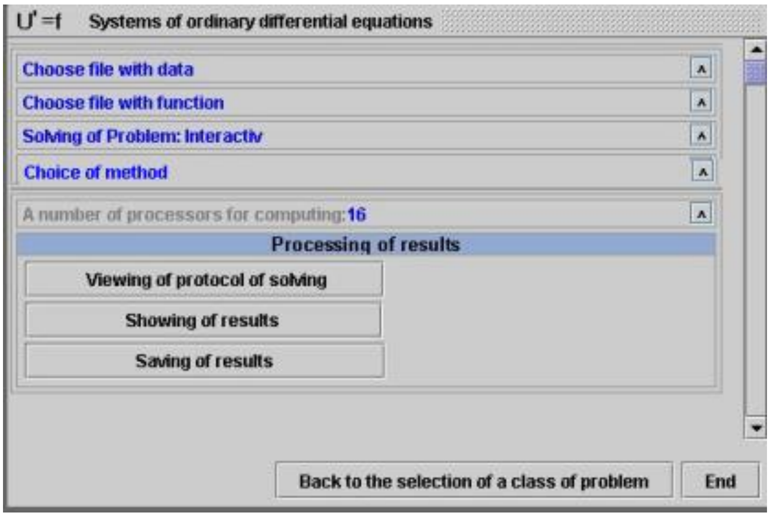
A protocol describing a process of investigating and solving of SODE is presented in the text form. It contains: parameters of the problem, methods used for investigating of problem for the sake of choosing an efficient algorithm and construction of the solution program, several control components of the solution, problem's run time, and the number of processors being used. The protocol can be browsed by pressing **«Show the result»** button, saved or printed.

4.7.2.5 *Inpartool's diagnostics during the solving of SODE*

When solving SODE **Inpartool** provides:

- referential information;
- Help-type messages at every stage of the user's work;
- problem's run-time diagnostics.

In the **«Help»** item of the main menu a user can get not only information about functional potentialities of Inpartool but also information about terminology being used related to systems of ordinary differential equations (fig. 4.54).

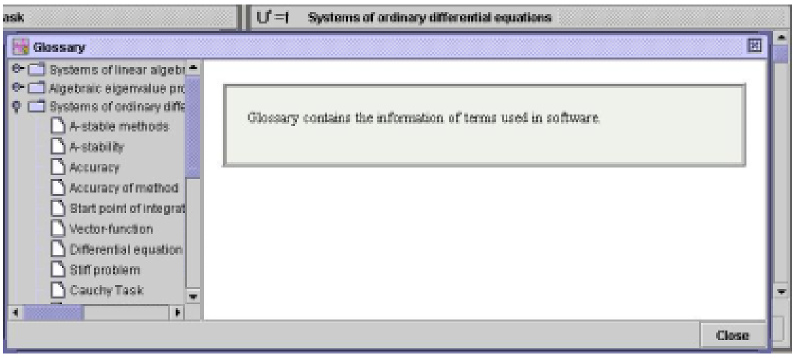


U' =f Systems of ordinary differential equations
Choose file with data
Choose file with function
Solving of Problem: Interactiv
Choice of method
Processing of results
Viewing of protocol of solving
Showing of results
Saving of results
Back to the selection of a class of problem
End

A number of processors for computing:16

Fig. 4.54. SODE processing main window

Having chosen the «**Glossary**» item in the «**Help**» submenu and then a term of interest from the list of terms the user can get its explanation (fig. 4.55).

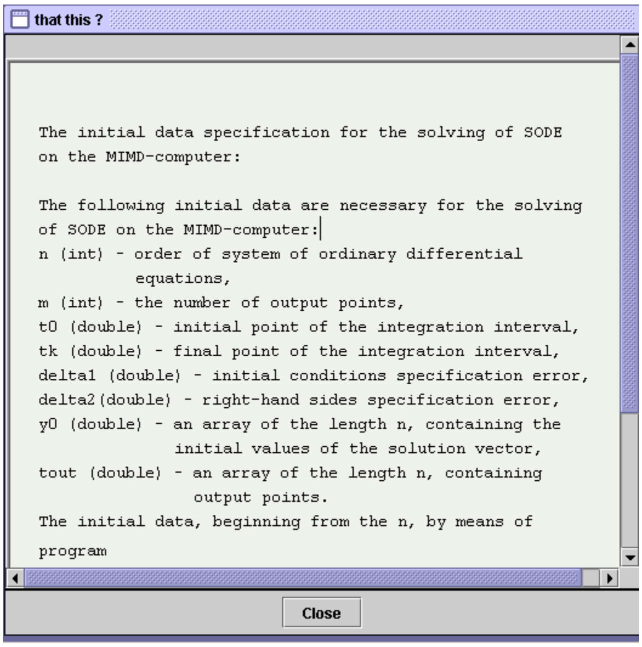


Glossary contains the information of terms used in software.

Fig. 4.55. Glossary help menu.

Having clicked by right-hand mouse button on the title «**Systems of ordinary differential equations**» a user can familiarize himself with Inpartool’s functional potentialities as to the solving of problems belonging to the given class of problems as well as with order of work with Inpartool.

In similarly the same manner a user can get appropriate short information at any stage of work by clicking the right-hand mouse button on any menu item or title of interest in the dialog window (fig. 4.56).



```
The initial data specification for the solving of SODE
on the MIMD-computer:

The following initial data are necessary for the solving
of SODE on the MIMD-computer:
n (int) - order of system of ordinary differential
          equations,
m (int) - the number of output points,
t0 (double) - initial point of the integration interval,
tk (double) - final point of the integration interval,
delta1 (double) - initial conditions specification error,
delta2(double) - right-hand sides specification error,
y0 (double) - an array of the length n, containing the
              initial values of the solution vector,
tout (double) - an array of the length n, containing
                output points.
The initial data, beginning from the n, by means of
program
```

Fig. 4.56. Pop-up help item

4.7.2.6 Examples of solving systems of ordinary differential equations means of Inpartool

Let us illustrate the using of Inpartool for the solving of SODE by the following two problems.

Problem 1

Solve SODE

$$\frac{du_i}{dt} = -\sum_{j=0}^{n-1} u_j - u_i = n(1+t) + 2 + t\,,\ i = 0,1,2,\ \ldots,\ n-1$$

under the initial conditions $u_i(0) = 1$, i = 0, 1, 2,…, n–1 on the interval [0, T].

The problem is common (non-stiff). Accuracy of obtaining the solution is eps=1.0×10^{-6}, error in the initial conditions' specification is delta1=1.0 ×10^{-10}, and error in the vector-function's specification is delta2=1.0 ×10^{-10}.

Exact solution of the problem is $u_i = 1 + t$, i = 0, 1, 2,…, n–1.

Problem 2

Solve SODE

$$\frac{du_i}{dt} = -1000(u_{i+1} + 2.5u_{i+2})u_i - 0.013u_{i+1};$$

$$\frac{du_{i+1}}{dt} = -(1000u_i + 0.013u_{i+1});$$

$$\frac{du_{i+2}}{dt} = -2500u_i\,u_{i+2};$$

under the initial conditions u_i (0) =0,0; u_{i+1} (0) =1,0; u_{i+2} (0) =1,0 on the interval [0, 50], i= 0, 3, 6,…, 3k, k = 0, 1, 2,…, n = 3k+3.

The problem is stiff. Accuracy of obtaining the solution is eps=1.0×10^{-6}, error in the initial conditions' specification is delta1=1.0 ×10^{-10}, error in the vector-function's specification is delta2=1.0 ×10^{-10}. Exact solution of the problem is

$$u_i(50) = -1{,}893\times10^{-6};\ u_{i+1}(50) = 0{,}59765;\ u_{i+2}(50) = 1{,}4023;$$

$$i = 0,\ 3,\ 6,\ \ldots,\ 3k,\ k = 0,\ 1,\ 2,\ \ldots,\ n = 3k+3.$$

Problem 1 was solved on 16 processors with n= 4000, T=0,4 by various methods: Gear's methods of order up to 5; Rosenbrock method, the 4-th order Runge-Kutta method, Adams' methods of order up to 12, explicit 1-st order method with attaining of the local accuracy.

The following solution was obtained (only the first component of it is given)

```
sol[0] = 1.400000e+00.
```

Problem 2 was solved on 16 processors with *n*= 600, *T*=50,0 by Gear's methods of order up to 5. The following solution was obtained (only first 4 components of are given):

```
Solution =
-1.888397e-06  5.964471e-01  1.403653e+00
-1.888397e-06.
```

The obtaining of acceleration was verified on the solving of Problem 1 with n = 4000 and T=4,0×10–4 by Adams' methods of order up to 12. The following run-times were obtained:

for one processor – time= 4.924150e-01;

for four processors – time= 1.443830e-01.

Thus, the obtained acceleration coefficient is equal to 0,49/0,14= 3,5, while the coefficient of efficiency is equal to 0,87.

The influence of the approximate nature of the initial data can be illustrated by following results.

The Problem 1 was solved with *n* = 4000 and *T*=0,4; accuracy of the solution's evaluation is eps=1,0×10^{-6}, error in the initial conditions' specification is delta1=1.0 ×10^{-10}, error in the vector-function's specification is delta2=1.0 ×10^{-10}. The employment of Adam's methods of order up to 12, 4-th order Runge-Kutta method and Euler-Couch method resulted in the following solution(only first 10 components of the solution are given):

```
Solution =    1.400000e+000 1.400000e+000 1.400000e+000
1.400000e+000 1.400000e+000 1.400000e+000 1.400000e+000
1.400000e+000 1.400000e+000 1.400000e+000.
At this point the solution is obtained with
accuracy delta= 1.000000e-006.
```

If for all above values the error in the initial conditions' specification is delta1=1.0 ×10^{-3}, error in the vector-function's specification is delta2=6,0×10^{-3} (i.e. values of initial conditions have been altered, and 6,0×10^{-3} has been added to each component of the vector-function), then employment of, for example, Adam's methods yields the following results:

```
Solution =    1.400059e+000 1.400059e+000 1.400059e+000
1.400059e+000 1.400059e+000 1.400059e+000 1.400059e+000
1.400059e+000 1.400059e+000 1.400059e+000.
At this point the solution is obtained with
accuracy delta= 3.401000e-003.
```

4.8. Library of intelligent programs Inparlib

4.8.1. Purpose and composition of the library

Intelligent programs involved in the library [1] are intended for the investigating and solving of basic problems of the computational mathematics:

- linear algebraic systems;
- algebraic eigenvalue problem;
- non-linear equations and systems;
- systems of ordinary differential equations.

Programs included in the library implement:

- statement of problems with approximately given initial data;
- investigation of characteristics of problem's computer model;
- verification of agreement between characteristics of problem's computer model revealed by computer and chosen solution algorithm;
- construction of topology of Inparcom's processors;
- the obtaining of solution together with reliability estimate which includes both estimate for the inherited error caused by the initial data error and estimate for the computational error.

Program modules implementing finished parts of investigating and solution algorithm are written in C and intended both for the MIMD-architecture computers and parallel programming environment MPI.

As to linear algebraic systems (LAS) program modules included in Inparlib enable to: investigate and solve problems with various structure matrices together with reliability estimates for the solution, invert matrices, evaluate both singular values and matrix ranks well as estimate matrix condition numbers.

As algebraic eigenvalue problems (common and generalized) Inparlib's programs solve both full and partial eigenvalue problem with various structure matrices (dense, band or sparse). By means of programs from Inparlib it is possible to evaluate condition numbers for separately taken eigenvalues, condition numbers for eigenvectors as well as to evaluate estimates for the overall error in solutions.

As to non-linear equations and systems Inparlib's programs enable to: investigate and solve systems of non-linear algebraic and transcendental equations; determine local condition number of the function *f(x)*, local condition number of the vector-function *F(x)*; implement termination criteria for iterative processes guaranteeing the obtaining both of solutions within the given accuracy and solution's errors with taking into account approximate nature of the initial data.

As to systems of ordinary differential equations with initial conditions, Inparlib contains programs enabling to: investigate and solve these systems, integrate both common and stiff systems of equations within accuracy of various orders as well as within any a priori specified accuracy. A user can carry out investigation of the stiffness of SODE, evaluate both the Lipschitz constant and accuracy of the obtained solution with taking into account approximate nature of the initial data.

Functional programs from Inparlib provide: statement of problems with approximately given initial data, investigation of mathematical characteristics of problem's machine models, verification of agreement between the revealed characteristics and application area for the solution algorithm being chosen as well the obtaining of solution together with reliability estimate or a refusal (with indication of reasons) in the solving of problem.

From the end user's point of view programs included in the library are reuse components in the solving of application problems for which problems of the computational mathematics are either intermediate or a final stage.

4.8.2. Library functions' overview

For the investigating and solving of LAS Inparlib contains the following functions:

`PLGESAD` – function for the investigating and solving of LAS with dense non-singular matrix by Gauss method with partial column pivoting within approximately given initial data. The program enables to obtain a solution to LAS together with estimates for the inherited and computational errors;

`PLPPSAD` – function for the investigating and solving of LAS with symmetric positive definite matrix by Cholesky method within approximately given initial data. The program enables to obtain a solution to LAS together with estimates for the inherited and computational errors;

`Slae_bsp_bp` – function for the investigating and solving of LAS with band symmetric positive definite matrix by Cholesky method implementing LDL^T-factorization of the matrix. The program enables to obtain a classic solution to LAS together with its reliability estimate;

`Slae_bss_bp` – function for the investigating and solving of LAS with band symmetric positive semi-definite matrices by three-staged regularization method. The program enables to obtain a pseudo-solution to LAS approximated to the normal solution within the given accuracy;

`Slae_svd_p` – function for the investigating and solving of LAS with rectangular or square singular general matrices by employing the singular-value decomposition of the system's matrix. The program enables to obtain a generalized solution to LAS together with its reliability estimate.

For the investigating and solving of AEVP Inparlib contains the following functions:

`mp_esytri` – function for the investigating and solving of full AEVP for symmetric tri-diagonal matrix with approximately given elements distributed between processors;

`mp_esyqai_bl` – function for the investigating and solving of full AEVP for dense symmetric matrix with approximately given elements distributed between processors;

`Evp_bs_bp` – function for the investigating and solving of partial standard or generalized AEVP for band symmetric positive definite matrix by method of iterations on the subspace. The program evaluates several minimum eigenvalues and eigenvectors corresponding to them as well as estimates the reliability of the obtained solutions.

For the investigating and solving of SNE Inparlib contains the following functions:

`zeroin` – function for the finding roots of non-linear equation by bisection method within specified initial data's error as well as for evaluating of error in the solution (if any);

`bur` – function for the solving of SNE by Burdakov method within approximately given initial data. The method possesses a global convergence and retains the quadratic rate of convergence in the neighborhood of the solution;

`kn` – function for the solving of SNE by Dennis and More's method. It implements quasi-Newton method which during the iterative process approximates the inverse Jacoby matrix with approximately given initial data. The method possesses a super-linear rate of convergence;

`nut` – function for the solving of SNE by Newton method within approximately given initial data. The method retains the quadratic rate of convergence in the neighborhood of the solution;

`fib` – function for the solving of SNE by first Broyden's method within approximately given initial data.. The method possesses a global convergence and retains the quadratic rate of convergence in the neighborhood of the solution;

`paul` – function for the solving of SNE with symmetric Jacoby matrix by Powell method within approximately given initial data. The method possesses a global convergence and retains the quadratic rate of convergence in the neighborhood of the solution.

For the investigating and solving of SODE Inparlib contains the following functions:

`ek_dri` – function for the solving of initial-value problems in SODE by 1-st order explicit Euler-Cauchy method within approximately given initial data;

`rk4_dri` – function for the solving of initial-value problems in SODE by 4-th order explicit Runge-Kutta method within approximately given initial data;

`rk6_dri` – function for the solving of initial-value problems in SODE by 5(6)-th order explicit Runge-Kutta method within approximately given initial data;

`adams_dri` – function for the solving of initial-value problems in SODE by Adams methods of order up to 12-th within approximately given initial data;

`rk1_dri` – function for the solving of initial-value problems in SODE by 1-st order Runge-Kutta-type method within approximately given initial data;

`gear_dri` – function for the solving of initial-value problems in SODE by Gear's methods of order up to 5-th within approximately given initial data;

`ros_dri` – function for the solving of initial-value problems in SODE by 4-th order Rosenbrock method within approximately given initial data.

4.9. References

1. Gerasimova, T.A., Zubatenko, V.S., Molchanov, I.N. et al.: *Library of intelligent parallel programs for the investigating and solving of problems of the computational mathematics with approximately given initial data*: Copyright registration certificate № 17213 as of 11.07.2006 p / State department on the intelligent property (in Russian).
2. Khimich, A.N., *Estimates for overall error in symmetric eigenvalue problem*, "Technology and methods for the solving of some application problems", Kiev: V.M.Glushkov Institute of cyberne-tics, 1991, pp. 85–88 (in Russian).
3. Khimich, A.N A.H., Voitsekhovsky, S.A., Brusnikin, V.N., *On the reliability of linear mathematical models with approximately given initial data*, "Mathematical machines and systems", 2004 vol. 3, pp. 54–62 (in Russian).
4. Khimich A.N., Molchanov I.N., Popov A.V., Yakovlev M.F, Chictyakova T.V., *Parallel algorithms for the solving of problems in the computational mathematics*, Kyiv: Naukova dumka, 2008. p. 247 (in Russian).
5. Khimich A.N., Molchanov I.N., Mova V.I., et al. – *Numerical software for the intelligent MIMD-computer Inparcom* , Kyiv: Naukova dumka, 2007, p. 216 (in Russian).
6. Molchanov, I.N., *Intelligent computers – an efficient tool for the investigating and solving of scientific and engineering problems*, "Cybernetics and system analysis", 2004 vol. 1, pp. 174–179 (in Russian).
7. Molchanov, I.N., *Machine methods for the solving of applied mathematics problems*, "Algebra, approximation of functions, ordinary differential equations", Kiev: Naukova Dumka, 1987, p. 550 (in Russian).
8. Molchanov, I.N. Mova, V.I, Stryuchenko, V.A. *Intelligent computers for investigating and solving of scientific and engineering problems– a new direction in the development of the computational machinery*, "Communications", 2005 vol. 7, pp. 45-46 (in Russian).
9. Parasyuk, I.N., Sergiyenko, I.V. *Packages of programs for the data analysis: development technology*, "Moscow: Finances and statistics", 1988, p. 159 (in Russian).
10. *Representation of knowledge in human-machine and robot systems*: in 3 vol. – Moscow: VINITI, Comp. Center Acad. Scis. USSR, 1984. – vol A, p. 216, vol. B, p. 236, vol. C, p. 378 (in Russian).
11. Wilkinson, J.H., *The algebraic eigenvalue problem*, Clarendon Press, Oxford, 1965.
12. http://www.inparcom.com.