Ellipsoids and Ellipsoid-Shaped Fuzzy Sets as Natural Multi-Variate Generalization of Intervals and Fuzzy Numbers: How to Elicit Them from Users, and How to Use Them in Data Processing [⋆]

Vladik Kreinovich [∗]

Department of Computer Science, University of Texas at El Paso, El Paso, TX 79968, USA

Jan Beck

Department of Computer Science, University of Texas at El Paso, El Paso, TX 79968, USA, and X-L Synergy, 2000 Wyoming Ave., El Paso, TX 79903, USA

Hung T. Nguyen

Department of Mathematical Sciences, New Mexico State University, Las Cruces, NM 88003, USA

Abstract

In this paper, we show that ellipsoids are natural multi-variate generalization of intervals and ellipsoid-shaped fuzzy sets are a natural generalization of fuzzy numbers. We explain how to elicit them from users, and how to use them in data processing.

Key words: ellipsoids, fuzzy sets, knowledge elicitation, data processing

[⋆] This work was supported in part by Applied Biomathematics, by NASA under cooperative agreement NCC5-209, by NSF grants EAR-0112968, EAR-0225670, and EIA-0321328, by NIH grant 3T34GM008048-20S1, and by the Army Research Lab grant DATM-05-02-C-0046. The author are very thankful to Abe Kandel and Mark Last, Guest Editors of this issue, for their encouragement, and to the anonymous referees for their hard work.

[∗] Corresponding author

Email addresses: `vladik@utep.edu` (Vladik Kreinovich), `janbeck@gmail.com` (Jan Beck), `hunguyen@nmsu.edu` (Hung T. Nguyen).

1 General Formulation of the Problem

Complex systems consist of many interacting subsystems all of which contribute to the main objectives of the system. With respect to each objective, we can usually characterize the system's performance by the corresponding numerical characteristic y. (Sometimes, several numerical characteristics are necessary to describe the system's performance.)

The most natural way to gauge the system's performance is to actually run this system and measure the corresponding characteristic. However, this most natural way is not always possible:

- It is definitely not possible on the design stage, when we do not yet have the working system.
- It is not always possible on the working stage because we may be interested in how the system performs under different circumstances, and it is not possible (or too expensive) to actually test the system under all these conditions.

It is therefore desirable to use computer simulations to gauge the system's performance. For engineering systems, we usually know what the corresponding subsystems do and how they interact with each other. Based on this knowledge, we can design a computer-based model of the system, a model that simulated how the system works and, based on this simulation, computes the value of the desired characteristic y.

Such models are rarely designed for a single system. Usually, we have a *family* of similar systems, and we design a general model f that serves all the systems from this family. This general model contains parameters $x_1, \ldots, x_n$; by specifying the values of these parameters, we can get the models of individual systems. In other words, once we know the exact values of the parameters $x_1, \ldots, x_n$ that describe the system, we can "plug in" these values into the general model f, and get the value $y = f(x_1, \ldots, x_n)$ of the desired characteristic.

This works well in the idealized situation when we know the exact values of the parameters x_i. In real life, we only know these values with some uncertainty. In other words, instead of a single parameter vector $x = (x_1, \ldots, x_n)$, we have several different possible parameters vectors, i.e., several different parameter vectors that are consistent with our knowledge. For different possible parameter vectors, we get different values of y.

It is therefore desirable to find the *range* of possible values of the desired characteristic $y = f(x_1, \ldots, x_n)$ over the set S of all possible parameter vectors $x = (x_1, \ldots, x_n)$.

In many real-life situations, in addition to the set S of all parameter vectors $x = (x_1, \ldots, x_n)$ which are possible, we often also know "narrower" sets $S_\beta \subseteq S$ which contain the actual values of x with different degrees of certainty β. With degree of certainty 1, we can only guarantee that $x \in S$, i.e., we have to take $S_1 = S$. In general, the larger the set, the more certain we are that x belongs to this set. Thus, if $\beta < \beta'$, we have $S_\beta \subseteq S_{\beta'}$. A family of sets with this property is called a *nested family*.

The corresponding family of nested sets can be viewed as a particular case of a *fuzzy set* [15,16]:

- If a nested family is given, then different sets S_β from the nested family can be viewed as α-cuts of a fuzzy set corresponding to different levels of uncertainty $\alpha = 1 - \beta$; this fuzzy set can be described by a characteristic function $\mu(x) = \sup\{\alpha : x \in S_{1-\alpha}\}$.
- Vice versa, if we have a fuzzy set, then its α-cuts $S_\alpha = S_{1-\beta}$ corresponding to $\alpha = 1 - \beta$ form a nested family of sets.

In the fuzzy case, in addition to knowing the range of the characteristic $y = f(x_1, \ldots, x_n)$ over the set S – the guarantee range of values of y, we would also like to know, for different degrees of certainty β, the range of y over the set S_β – the range to which y belongs with certainty β.

It is worth mentioning that these ranges also form a nested family – thus, a 1-D fuzzy set Y corresponding to y.

To compute this fuzzy set, we must be able, for each degree β, to compute the range of y over the corresponding crisp set S_β – the α-cut of the original fuzzy set. Thus, to solve the range problem for fuzzy sets, it is sufficient to solve it for crisp sets corresponding to different α. In view of this comment, in the following text, we will mainly concentrate on the problem of computing the range of a given function over a crisp set.

2 Important Specific Case of the Problem: For Extremely Complex Systems, We Need an Approximate Model

From the mathematical viewpoint, the range of y is an interval $[\underline{y}, \overline{y}]$, where $\underline{y}$ is the smallest possible value of $y = f(x_1, \ldots, x_n)$ under the constraint that $(x_1, \ldots, x_n) \in S$, and $\overline{y}$ is the largest possible value of y under this constraint. Thus, to find the desired range, we must *optimize* the function $f(x_1, \ldots, x_n)$ over the set S of possible parameter vectors.

These exist numerous efficient algorithms for optimizing functions under different constraints. Most of these algorithms are based on the following idea:

- we start with one or several initial tries x;

  for example, in gradient methods, we need to estimate both the value of the function and its partial derivatives $\partial f/\partial x_i$; to estimate each partial derivative

  $$\frac{\partial f}{\partial x_i} \approx \frac{f(x_1,\ldots,x_{i-1},x_i+h,x_{i+1},\ldots,x_n)-f(x_1,\ldots,x_n)}{h},$$

  we must know the value of f at the original point x and at a point $(x_1,\ldots,x_{i-1},x_i+h,x_{i+1},\ldots,x_n)$ at which i-th variable is slightly different; thus, overall, we need to estimate f at $n+1$ different points x;
- we apply the function f to these tries;
- based on the results of this application, we try to predict how the function will behave for other values of the parameter vector x;
- based on this prediction, we produce a new vector x_{new} at which – we hope – the value of f will be better than for the previous tries;
- we then apply f to this new vector x_{new};
- based on the results of this application, we update the previous prediction of f's behavior;
- based on thus modified prediction, we produce yet another vector x, etc.

In short, these methods require that we apply f to numerous combinations of parameters, i.e., in our case, to numerous values of the parameter vector $x=(x_1,\ldots,x_n)$.

For many systems, this can be done, but some systems are so complex than running the model $f(x_1,\ldots,x_n)$ even once may require several hours or even days of computation on an advanced supercomputer. For such extremely complex systems, we can, at best, run the model f approximately $N=$ 100-200 times.

Because of this limitation, when we run an optimization algorithm, we cannot use the *original* function $f(x_1,\ldots,x_n)$ at each step of this algorithm. Therefore, for the optimization algorithm to be effective, we must provide an *approximation* $f_{\text{approx}}(x_1,\ldots,x_n)$ to the function $f(x_1,\ldots,x_n)$ that this optimization algorithm can use instead of the original function $f(x_1,\ldots,x_n)$.

3 How To Construct the Approximate Model? General Idea

In general, how can we get an easy-to-compute approximate model $f_{\text{approx}}(x_1,\ldots,x_n)$ that approximates the original very-hard-to-compute model $f(x_1,\ldots,x_n)$? A natural way it is to fix a family of easy-to-compute approximating models, a family depending on several coefficients, then call f as many times as we can afford and tune these parameters based on the results.

In science in engineering, the most widely used easy-to-compute models are *linear* models

$$f_{\text{approx}}(x_1, \ldots, x_n) = f_0 + f_1 \cdot x_1 + \ldots + f_n \cdot x_n.$$

To uniquely describe a linear model, we must describe the values of $n + 1$ coefficients $f_0, f_1, \ldots, f_n$. To determine the values of these $n + 1$ coefficients, we must know $n + 1$ different values of the function f.

In our practical example, we have $n \approx 50$ variables, and we can perform $N \approx$100-200 calls to the difficult-to-compute model f. Since $N \gg n + 1$, from the results of N runs of f, we can determine more than $n + 1$ coefficients needed for a linear model; thus, we can go beyond linear approximations.

After the linear approximation, the next natural approximation is the quadratic one, in which we approximate a general function $f(x_1, \ldots, x_n)$ by a *quadratic* expression (*response surface*)

$$f_{\text{approx}}(x_1, \ldots, x_n) = f_0 + \sum_{i=1}^{n} f_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot x_i \cdot x_j. \tag{1}$$

To describe a general quadratic expression, we need to know 1 coefficient f_0, n coefficients f_i, and $n(n + 1)/2$ coefficients $f_{i,j} = f_{ji}$ – to the total of $(1/2) \cdot n^2 + (3/2) \cdot n + 1$.

For our practical case, when $n = 50$, we thus need

$$\frac{1}{2} \cdot 50^2 + \frac{3}{2} \cdot 50 + 1 = 1{,}250 + 75 + 1 = 1{,}326$$

coefficients – but we can only run f at most 100-200 $\ll$ 1,326 times. Thus, in our practical case, not only we cannot meaningfully use 3rd and higher order approximate models – we cannot even use the full quadratic model. The only thing we can do is to use *restricted* quadratic model, in which we select beforehand a limited number of possible non-zero coefficients $f_{i,j}$. Since we can only use N coefficients, and linear terms require n of them, there are only $N - n$ coefficients left to cover the quadratic terms.

The knowledge of which coefficients $f_{i,j}$ are probably 0 and which may be non-zeros can only come from the experts who have designed and/or analyzed the system. How do the corresponding non-zero values affect the system's behavior?

- if $f_{i,i} \neq 0$, this means that the dependence of the desired characteristic y on the corresponding parameter x_i is strongly *non-linear*;

- if $f_{i,j} \neq 0$, this means that the parameters x_i and x_j are *interdependent* in the sense that the degree to which y depends on x_i is different for different values of x_j.

We can therefore ask experts which parameters and pairs of parameters exhibit such behavior. If there too many (more than N) such parameters and pairs, we must limit ourselves only to those parameters which are most non-linear and to the pairs which show the maximum dependence. To be able to do that, we ask the experts not only to list the parameters that have non-linear behavior, but also to rank these parameters by the degree of non-linearity.

We therefore arrive at the following method of constructing the approximate model.

4 How to Construct an Approximate Model: A Methodology

First, we ask an expert (or experts) to list all the parameters x_i for which the dependence of the desired characteristic y on x_i is highly non-linear. Among all these parameters, we ask the expert to provide a ranking, so that these parameters are listed in the order from the ones that exhibit the largest amount of non-linear behavior to the ones that exhibit the smallest amount of non-linearity.

We also ask an expert (or experts) to list all the pairs of parameters (x_i, x_j) which exhibit interdependence, i.e., for which the degree with which the desired characteristic y depends on x_i is different for different values of x_j. Among all these pairs, we ask the expert to provide a ranking, so that these pairs are listed in the order from the ones that exhibit the largest amount of interdependence to the ones that exhibit the smallest amount interdependence.

In addition, we ask the expert to merge their rankings of parameters and pairs into a single ranking. This may be more difficult and somewhat more subjective, but we must have this ranking anyway.

Then, in the joint ranking, we select $N - n$ top parameters and pairs, and consider the model (1) in which:

- the coefficient $f_{i,i}$ can be different from 0 only if x_i is one of the selected parameters (selected for its high non-linearity), and
- the coefficient $f_{i,j}$ $(i \neq j)$ can be different from 0 only if (x_i, x_j) is one of the selected pairs (selected for its high interdependence).

All the other coefficients $f_{i,j}$ are identically 0.

How do we actually determine the values of the non-zero coefficients f_0, f_i, and $f_{i,j}$? There are two possible situations:

- In the ideal case, not only we can run the model f N times, but we can actually decide on which parameter vectors to run it.
- In a more pessimistic (and probably more realistic) situation, we do not have a choice of selecting the parameter vectors. Someone has already selected N different parameter vectors $x^{(k)} = (x_1^{(k)}, \ldots, x_n^{(k)})$, $1 \le k \le N$, ran the model f for these vectors, and got the corresponding N values $y^{(1)}, \ldots, y^{(N)}$.

In the more pessimistic case, we have N linear equations to determine N unknown parameters f_0, f_i, and $f_{i,j}$:

$$f_0 + \sum_{i=1}^{n} f_i \cdot x_i^{(k)} + \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot x_i^{(k)} \cdot x_j^{(k)} = y^{(k)}. \tag{2}$$

In this situation, the only way to find these coefficients is to actually solve this system of N linear equations with N unknowns. On modern computers, this is quite doable.

In the ideal case, we can select the parameter vectors so as to make the reconstruction of the coefficients much easier, without the necessity to solve any large system of linear equations. Namely:

- First, we pick a vector $x^{(0)} = (x_1^{(0)}, \ldots, x_n^{(0)})$ that is inside the zone of possible vectors, and run the model f on this vector. As a result, we get the value $y^{(0)}$.
- For each parameter x_i for which $f_{i,i} = 0$ (i.e., for which there is no non-linear dependence on x_i), we select a deviation h that keeps us within the zone of possible parameter vectors, form a vector

  $$x^{(i)} = (x_1^{(0)}, \ldots, x_{i-1}^{(0)}, x_i^{(0)} + h, x_{i+1}^{(0)}, \ldots, x_n^{(0)}), \tag{3}$$

  and use the model to compute the corresponding value $y^{(i)} = f(x^{(i)})$. Since $f_{i,i} = 0$, from (2), we conclude that $y^{(i)} - y^{(0)} = f_i \cdot h$, hence f_i can be computed as

  $$f_i = \frac{y^{(i)} - y^{(0)}}{h}.$$
- For each parameter x_i with $f_{i,i} \ne 0$, to find the values f_i and $f_{i,i}$, we form two vectors: the vector (3) and the vector

  $$x^{(-i)} = (x_1^{(0)}, \ldots, x_{i-1}^{(0)}, x_i^{(0)} - h, x_{i+1}^{(0)}, x_n^{(0)}).$$

  We then use the model f to compute the corresponding values $y^{(i)} = f(x^{(i)})$ and $y^{(-i)} = f(x^{(-i)})$. Due to (2), we have $y^{(i)} - y^{(-i)} = 2f_i \cdot h$, hence f_i can be computed as

$$f_i = \frac{y^{(i)} - y^{(-i)}}{2h}.$$

Similarly, due to (2), we have $y^{(i)} + y^{(-i)} - 2y^{(0)} = 2f_{i,i} \cdot h^2$, hence $f_{i,i}$ can be computed as

$$f_{i,i} = \frac{y^{(i)} + y^{(-i)} - 2y^{(0)}}{2h^2}.$$

- Finally, for every interdependent pair (i,j), to determine $f_{i,j}$, we form a vector

$$x^{(ij)} = (x_1^{(0)}, \ldots, x_{i-1}^{(0)}, x_i^{(0)} + h, x_{i+1}^{(0)}, \ldots, x_{j-1}^{(0)}, x_j^{(0)} + h, x_{j+1}^{(0)}, \ldots, x_n^{(0)}).$$

Due to (2), we have $y^{(ij)} - y^{(0)} = f_i \cdot h + f_j \cdot h + 2f_{i,j} \cdot h^2$. Since we have already determined f_i and f_j, we can thus determine $f_{i,j}$ as

$$f_{i,j} = \frac{y^{(ij)} - f_i \cdot h - f_j \cdot h}{2h^2}.$$

In the following text, we will assume that we already know the coefficients of the quadratic approximation (2).

5 How to Describe Possible Parameter Vectors? General Idea

In order to find the desired bounds on the value of the quantity y, we must describe the set S of all possible parameter vectors $x = (x_1 \ldots, x_n)$. For many complex systems, this information has to come from experts.

First, for each of the parameters x_i, the experts must provide us with the range of the values of this parameter, i.e., with the interval $[\underline{x}_i, \overline{x}_i]$ of possible values of x_i. Once we know these intervals, we can then guarantee that the possible values of x are inside the box

$$[\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_n, \overline{x}_n]. \quad (4)$$

It does not mean, however, that all the vectors x within this box are indeed possible.

In some cases, there is a "correlation" between the parameters x_i. Here, by a correlation, we do not necessarily mean correlation in the usual statistical sense, we mean correlation in its commonsense meaning. For example, positive correlation between x_1 and x_2 means that, in general, larger values of x_1 correspond to larger values of x_2, and vice versa. In this case, it is highly unprobable that the parameter x_1 attains its largest possible value $\overline{x}_1$, and at the same time the positively related parameter x_2 attains its smallest possible value $\underline{x}_2$.

Even when there is no correlation between the two parameters, i.e., if these parameters are, in this sense, "independent", not all pairs (x_1, x_2) may be possible. It may be possible that the parameter x_1 attains its extreme value $\overline{x}_1$, and it may be possible that the parameter x_2 attains its extreme value $\overline{x}_2$, but often, it is reasonable to believe that both extreme situations cannot occur at the same time.

In both cases (correlation and independence), the set S of possible parameter vectors is a proper subset of the box (4). How can we describe such a subset?

In real life, whenever we have a cluster formed by real-life data points, this cluster has a smooth boundary. This cluster can be a disk (solid circle), a ball (solid sphere in multi-D space), an ellipsoid, or a more complex structure, but it is practically always smooth. The fact that it is smooth means that we can describe its border by an equation $b(x_1, \ldots, x_n) = C$ for some smooth function $b(x_1, \ldots, x_n)$ and for some constant C. As a result, the set S itself can be describe either by the inequality

$$b(x_1, \ldots, x_n) \le C \tag{5}$$

or by the inequality $b(x_1, \ldots, x_n) \ge C$. In the second case, the inequality can be transformed into an equivalent form $b'(x_1, \ldots, x_n) \le C'$, where the function $b'(x_1, \ldots, x_n) = -b(x_1, \ldots, x_n)$ is also smooth, and $C' = -C$. So, without loss of generality, we can assume that the set S is described by the inequality (5), for some smooth function $b(x_1, \ldots, x_n)$.

An arbitrary smooth function can be approximated by a polynomial, so, instead of the the general set (5), we can consider the approximating set

$$a(x_1, \ldots, x_n) \le C, \tag{6}$$

where $a(x_1, \ldots, x_n)$ is a polynomial that approximates the smooth function $b(x_1, \ldots, x_n)$.

As we have already mentioned, the simplest possible polynomials are linear polynomials $a(x_1, \ldots, x_n) = a_0 + a_1 \cdot x_1 + \ldots + a_n \cdot x_n$. However, for a linear function $a(x_1, \ldots, x_n)$, the set of all the vectors x for which $a(x) \le C$ is a half-space, i.e., a set that is not bounded in many directions, while we want a set S that is inside the box – and hence, bounded in all directions. Thus, if we restrict ourselves to only linear terms, we do not get a good approximation to the set (5).

To get a reasonable approximation, we must consider quadratic and higher order polynomial approximating functions $a(x_1, \ldots, x_n)$. In particular, for the simplest non-linear polynomials – quadratic polynomials – the approximating set (6) takes the following form:

$$a(x_1,\ldots,x_n) = a_0 + \sum_{i=1}^{n} a_i \cdot x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} a_{i,j} \cdot x_i \cdot x_j \le C. \tag{7}$$

To get an even better description of the actual set (5), we can, in principle, use 3rd, 4th, and higher order polynomials. However, in our problem, we have about $n = 50$ variables. Similarly to the our analysis of the response surface, we can conclude that to describe a general quadratic set (7), we need to know 1,326 coefficients. To describe higher order approximations, we need even more coefficients.

Each coefficient must be elicited from the experts. It is not realistic to ask an expert more than a thousand questions, and expect meaningful answers to all of them. Thus, in our practical case, not only we cannot meaningfully use 3rd and higher order approximating polynomials $a(x_1,\ldots,x_n)$ – we cannot even use the full quadratic model. The only thing we can do is to use *restricted* quadratic model, in which we select beforehand a limited number of possible non-zero coefficients $a_{i,j}$.

For a quadratic function $a(x_1,\ldots,x_n)$, the bounded set of all the values of x for which the inequality (7) holds is an *ellipsoid*. Thus, in our practical problem, we describe the set of all possible parameter vectors x by an ellipsoid.

Comment.

- If we approximate each α-cut of a fuzzy set by an ellipsoid, then we get an ellipsoid-shaped fuzzy set.
- It is worth mentioning that ellipsoids are not only the best approximation that we can afford, they are actually efficiently used in description of uncertainty, e.g., in control; see, e.g., [1–3,5–7,17–19,21]; in fuzzy context, they are used – in 2-D case – in [10,11].
- Also, ellipsoids naturally come from probabilistic uncertainty: if the measurement error is described by a multi-D Gaussian distribution, then the confidence set – described as the set of all the values for which the probability density exceeds a certain threshold – is an ellipsoid.
- Not only ellipsoids work well; it has been experimentally shown that in many practical situations, ellipsoids work better than other families of sets [2,3]. Moreover, it was theoretically proven that under certain reasonable conditions, ellipsoids indeed form the best family [6,13]; for the 2-D fuzzy case, a similar result was proven in [14].
- When we replace the box with a smaller ellipsoid set S, not only we make the range smaller (and thus more realistic), we also make this range easier to compute:

- · it is known that even for a quadratic function $f(x_1, \dots, x_n)$, estimating its range over a box is a provably difficult (NP-hard) problem [12,22];
- · on the other hand, finding the range of a quadratic function over an ellipsoid is a computationally doable problem [12,20,22].

In this paper, we will show, in detail, how this range can be computed.

6 How to Elicit the Ellipsoid from an Expert? General Idea

We have already mentioned that since we cannot ask thousands of questions the experts and expect meaningful answers to all of them, we cannot even use the full quadratic model. Instead, we have to use a restricted quadratic model, in which we select beforehand a limited number of possible non-zero coefficients $a_{i,j}$.

Let us assume that we can only ask E questions to an expert. In this case, we can only have E non-zero coefficients $a_{i,j}$. We can therefore ask experts which pairs of parameters are correlated. If there too many (more than E) such pairs, we must limit ourselves only to those pairs which are most correlated (positively or negatively). To be able to do that, we ask the experts not only to list the correlated pairs, but also to rank these pairs by the degree of correlation, so that the expert will be able to select E most correlated pairs.

Once we select E pairs, what can we ask? In the specific case when the ellipsoid is a confidence set of a general distribution, we can ask about the values of the correlation (in the usual statistical sense) between the variables x_i and x_j. However, in general, an ellipsoid is just a set of possible values of x, with no specific probabilistic meaning, so there is no well-defined statistical correlation, and we cannot ask an expert about its value.

What we can ask is the following. As we described it, dependence between x_i and x_j means that the range of possible values of x_i changes depending on the value of x_j:

- If there is a positive correlation, this means that, in general, when x_j grows, the possible values x_i also become, in general, larger – and the values of x_i are the *largest* when x_j attains its largest possible value $\overline{x}_j$.
- Similarly, if there is a negative correlation, this means that, in general, when x_j grows, the possible values x_i become, in general, smaller – and the values of x_i are the *smallest* when x_j attains its largest possible value $\overline{x}_j$.

Thus, to gauge the degree of (thus defined) dependence between x_i and x_j, we should ask an expert not only to give us the ranges of possible values $[\underline{x}_i, \overline{x}_i]$ and $[\underline{x}_j, \overline{x}_j]$ for the parameters x_i and x_j, but also to describe what values x_i

are possible when x_j attains its largest possible value $\overline{x}_j$.

- If these values x_i are closer to $\overline{x}_i$, this means that we have a positive correlation correlation between x_i and x_j.
- If these value x_i are closer to $\underline{x}_i$, this means that we have a negative correlation between x_i and x_j.

How we can transform this easy-to-elicit information into the precise description of the ellipsoid (7)? To answer this question, let us first simplify the general formula (7).

We start with a very simple step. If we divide both sides of the inequality (7) by the constant C, we get a slightly simpler description of the general ellipsoid, with one fewer coefficient:

$$a(x_1, \ldots, x_n) = a_0 + \sum_{i=1}^{n} a_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot x_i \cdot x_j \le 1. \tag{8}$$

The next simplification step is also based on the known geometric facts. Specifically, it is known that every ellipsoid has a center $\widetilde{x} = (\widetilde{x}_1, \ldots, \widetilde{x}_n)$, and that when we use a coordinate system with the origin at this center – i.e., if we use the differences $\Delta x_i \stackrel{\text{def}}{=} x_i - \widetilde{x}_i$ instead of the original values of x_i – the equation for an ellipsoid takes the following simplified form:

$$a(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot (x_i - \widetilde{x}_i) \cdot (x_j - \widetilde{x}_j) \le 1. \tag{9}$$

In Appendix 1, we describe how we can transform the above easy-to-elicit information into the precise description of the ellipsoid (9). Thus, we arrive at the following algorithm:

7 How to Elicit the Ellipsoid from an Expert? Algorithm

- First, we ask an expert to provide, for each parameter x_i, the range $[\underline{x}_i, \overline{x}_i]$ of possible values of x_i.
- Based on these values, we compute the midpoints $\widetilde{x}_i = (\underline{x}_i + \overline{x}_i)/2$ and the half-width $\Delta_i = (\overline{x}_i - \underline{x}_i)/2$ of the corresponding intervals.
- Then, we ask an expert (or experts) to list all the pairs (x_i, x_j) of correlated parameters. Among all these pairs, we ask the expert to provide a ranking, so that the expert will be able to list E most correlated pairs.
- For each possibly correlated pair (x_i, x_j), we ask the expert: what is the most reasonable value of x_i when $x_j = \overline{x}_j$? We denote the corresponding value by $x_{i,j}$, and compute $\Delta x_{i,j} \stackrel{\text{def}}{=} x_{i,j} - \widetilde{x}_i$.

- For all other (i.e., non-correlated) pairs $i \neq j$, we define $\Delta x_{i,j} \stackrel{\text{def}}{=} 0$, and we also take $\Delta x_{j,j} \stackrel{\text{def}}{=} \Delta_j$.
- Based on the known values $x_{i,j}$, $\widetilde{x}_i$, and Δ_i, we compute the components of the symmetric matrix Z with the components $z_{k,j} \stackrel{\text{def}}{=} \Delta x_{k,j} \cdot \Delta_j$.
- Finally, we invert the matrix Z. The elements $a_{i,j}$ of the inverse matrix $A = Z^{-1}$ are exactly the coefficients that describe the ellipsoid (9).

Algorithmic comment. In principle, to invert the matrix, we can use any matrix inversion algorithm; see, e.g., [9]. However, since our objective is not just to describe the set S of possible values of the parameter vector, but rather to find the range of the quadratic response function $f(x_1, \ldots, x_n)$ over the set S, we will see that a special inversion algorithm is the most appropriate here: an algorithm based on finding the eigenvalues and eigenvectors of the matrix Z.

Practical comment. In our algorithm, we assume that the set E of all combinations $(x_1, \ldots, x_n)$ that the expert considers possible is an ellipsoid, that for every i, the range $[\underline{x}_i, \overline{x}_i]$ is the exact projection of this ellipsoid, and that for each i and j, the expert produces the value x_i for which $(x_i, \overline{x}_j) \in E$. Thus, the ellipsoid E is inscribed within the box $B \stackrel{\text{def}}{=} [\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_n, \overline{x}_n]$. Under these assumption, our algorithm reconstructs this ellipsoid based on the expert's information.

In reality, the set E of all the combinations $(x_1, \ldots, x_n)$ that the expert consider possible may be close but somewhat different from an ellipsoid. Due to this difference, the (approximate) ellipsoid E_0 produced by our algorithm may be "almost" (but not completely) within the box B.

8 How to Estimate the Range of a Quadratic Function over an Ellipsoid: General Idea

As we described earlier, we must find the maximum and the minimum of the given quadratic function

$$f(x_1, \ldots, x_n) = f_0 + \sum_{i=1}^{n} f_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot x_i \cdot x_j \tag{10}$$

over the given ellipsoid

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot (x_i - \widetilde{x}_i) \cdot (x_j - \widetilde{x}_j) \leq 1. \tag{11}$$

As we have just learned, we can easily extract the values $\widetilde{x}_i$ from the experts; however, we do not directly get the values $a_{i,j}$ from the experts, we only get the matrix $z_{i,j}$ that is the inverse to the matrix $a_{i,j}$.

We will solve this problem in a reasonably straightforward way: we will analyze the problem, figure out the sources of difficulty, and try to overcome these difficulties.

For simplicity, we will start with the assumption that we already know the inverse matrix $a_{i,j}$. After we describe the main idea for this simplified case, we will then discuss what to do in a more realistic case when we only know $z_{i,j}$.

Let us start with the analysis of the constrained optimization problem (10)–(11). In general, the difficulty in solving the constraint optimization problems comes from two sources:

- first, the objective function itself is difficult to optimize;
- second, the constraints are difficult to take into consideration.

In our case, the objective function is quadratic. If we did not have any constraints, then, to find its optima, we could simply differentiate f with respect to each of n variables and equate all n derivatives to 0. The derivative of a quadratic function is a linear function. Thus, in the absence of constraints, we would have a system of n linear equations with n unknown, a system that is easy to solve.

Thus, in contrast to the general case of constrained optimization, in our specific case (10)–(11), the main difficulty lies not in the objective function (10), but in the constraints (11). Therefore, we arrive at the following natural strategy for solving our optimization problem:

- first, to simplify the problem, we will try to simplify the constraints as much as possible;
- only when there is no possibility to further simplify the constraints, we should try to simplify the objective function as well.

The first simplification of the constraint (11) is something that we have already discussed: we introduce the new variables $\Delta x_i = x_i - \widetilde{x}_i$. In these new variables, the constraints (11) take the following simplified form:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot \Delta x_i \cdot \Delta x_j \le 1. \tag{12}$$

To describe the objective function in terms of the new variables Δx_i, we must substitute the expression $x_i = \widetilde{x}_i + \Delta x_i$ into the formula (10). As a result, we arrive at the following expression (detailed derivation of this and other formulas is given in Appendix 2):

$$f(x_1,\ldots,x_n)=f_0'+\sum_{i=1}^n f_i'\cdot\Delta x_i+\sum_{i=1}^n\sum_{j=1}^n f_{i,j}\cdot\Delta x_i\cdot\Delta x_j, \tag{13}$$

where

$$f_0'=f_0+\sum_{i=1}^n f_i\cdot\widetilde{x}_i+\sum_{i=1}^n\sum_{j=1}^n f_{i,j}\cdot\widetilde{x}_i\cdot\widetilde{x}_j, \tag{14}$$

and

$$f_i'=f_i+2\cdot\sum_{j=1}^n f_{i,j}\cdot\widetilde{x}_j. \tag{15}$$

How can we further simplify the expression (14)? This expression contains a general quadratic form $\sum a_{i,j}\cdot\Delta x_i\cdot\Delta x_j$. In the expression (14), we describe each n-dimensional vector $\Delta x=(\Delta x_1,\ldots,\Delta x_n)$ by its coordinates in the standard coordinate system, with respect to the standard basis formed by n mutually orthogonal unit vectors $e^{(1)}=(1,0,\ldots,0)$, $e^{(2)}=(0,1,0,\ldots,0)$, $\ldots$, $e^{(n)}=(0,\ldots,0,1)$:

$$\Delta x=\Delta x_1\cdot e^{(1)}+\ldots+\Delta x_n\cdot e^{(n)}. \tag{16}$$

Using scalar (dot) product of two vectors to multiply both sides of the formula (16) by $e^{(i)}$, and taking into consideration that $e^{(i)}$ is an orthonormal basis, i.e., $e^{(i)}\cdot e^{(j)}=0$ for $i\neq j$ and $e^{(i)}\cdot e^{(i)}=1$, we conclude that $\Delta x_i=\Delta x\cdot e^{(i)}$.

The i-th coefficient Δx_i in the expansion (16) can be therefore described as the results of scalar (dot) product of the vector Δx and the corresponding vector $e^{(i)}$ from the orthonormal basis.

It is known (see, e.g., [9]) that an arbitrary quadratic form can be simplified (namely, it can be *diagonalized*) by using a different basis of n mutually orthogonal unit vectors, namely, by using the basis formed by unit eigenvectors $v^{(1)}=\left(v_1^{(1)},\ldots,v_n^{(1)}\right)$, $\ldots$, $v^{(n)}=\left(v_1^{(n)},\ldots,v_n^{(n)}\right)$ of the matrix A, i.e., vectors for which $A\cdot v^{(k)}=\lambda_k\cdot v^{(k)}$ for some real numbers λ_k (called *eigenvalues*).

It is known that in the generic case, when all the eigenvalues are different, the corresponding eigenvectors are indeed mutually orthogonal.

In the degenerate case, when some eigenvalues coincide, eigenvectors corresponding to the equal eigenvalues are not necessarily orthogonal; however, in this case, we can apply an appropriate orthonomalization procedure and also get mutually orthogonal vector.

In this paper, we use a procedure for computing eigenvalues and eigenvectors that always returns mutually orthogonal eigenvectors. Therefore, in the following text, we will assume that different eigenvectors are orthogonal to each other.

If, to describe a vector Δx, we use its coordinates with respect to the new basic, i.e., the values $y_1, \ldots, y_n$ such that

$$\Delta x = y_1 \cdot v^{(1)} + \ldots + y_n \cdot v^{(n)},$$

then, in terms of the new coordinates, the quadratic form $\sum a_{i,j} \cdot \Delta x_i \cdot \Delta x_j$ turns into a simpler expression $\sum \lambda_k^2 \cdot y_k^2$. Thus, the constraint (14) takes a simplified form:

$$\sum_{k=1}^{n} \lambda_k^2 \cdot y_k^2 \le 1. \tag{17}$$

To describe the objective function (13) in terms of the new variables, we can use the fact (see derivation in Appendix 2) that

$$\Delta x_i = \sum_{k=1}^{n} v_i^{(k)} \cdot y_k. \tag{18}$$

Substituting the expression (18) into the objective function (13), we conclude that

$$f(x_1, \ldots, x_n) = f_0' + \sum_{k=1}^{n} g_k \cdot y_k + \sum_{k=1}^{n} \sum_{l=1}^{n} g_{k,l} \cdot y_k \cdot y_l, \tag{19}$$

where

$$g_k = \sum_{i=1}^{n} f_i' \cdot v_i^{(k)} \tag{20}$$

and

$$g_{k,l} = \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot v_i^{(k)} \cdot v_j^{(l)}. \tag{21}$$

Now, the problem is to optimize the quadratic function (19) under the simplified quadratic constraint (18).

Can we simplify the constraint (18) even further? Yes, if, instead of the original variables y_k, we introduce new variables $z_k = \sqrt{\lambda_k} \cdot y_k$ for which $z_k^2 = \lambda_k \cdot y_k^2$ and therefore, the constraint (17) takes an even simpler form

$$\sum_{k=1}^{n} z_k^2 \le 1. \tag{22}$$

From the geometric viewpoint, this constraint describes a unit ball in n-dimensional space.

Substituting $y_k = \dfrac{z_k}{\sqrt{\lambda_k}}$ into the expression (19) for the objective function, we get the new expression:

$$f(x_1,\ldots,x_n)=f_0'+\sum_{k=1}^n g_k'\cdot z_k+\sum_{k=1}^n\sum_{l=1}^n g_{k,l}'\cdot z_k\cdot z_l, \tag{23}$$

where

$$g_k' \stackrel{\text{def}}{=} \frac{g_k}{\sqrt{\lambda_k}};\ \ g_{k,l}' \stackrel{\text{def}}{=} \frac{g_{k,l}}{\sqrt{\lambda_k}\cdot\sqrt{\lambda_l}}. \tag{24}$$

Now, the constraints have the simplest possible form, so the only way to make the problem easier-to-solve is to simplify the objective function. We already know how to simplify the quadratic form: for that, we will look for the eigenvalues μ_k and eigenvectors $u^{(p)}=\left(u_1^{(p)},\ldots,u_n^{(p)}\right)$ of the matrix $g_{k,l}'$. In terms of the new variables

$$t_p=\sum_{k=1}^n u_k^{(p)}\cdot y_k, \tag{25}$$

the constraint (22) retains its form (see Appendix 2 for details):

$$\sum_{p=1}^n t_p^2\le 1; \tag{26}$$

while the objective function gets the following simplified form:

$$f(x_1,\ldots,x_n)=f_0'+\sum_{p=1}^n h_p\cdot t_p+\sum_{p=1}^n \mu_p\cdot t_p^2, \tag{27}$$

where

$$h_p \stackrel{\text{def}}{=} \sum_{k=1}^n u_k^{(p)}\cdot g_k. \tag{28}$$

After all the above simplification steps, we arrive at a problem of optimizing the objective function (27) under the constraint (26). To solve this problem, we will use the Lagrange multiplier technique. The idea behind the Lagrange multiplier approach to finding the optimum (minimum or maximum) of an objective function f under the constraint $g\le 1$ is as that this optimum is attained either inside the constraint set S, or on its border.

- If the optimum is attained inside the set S, then it is a local (or maybe even a global) optimum or the function f; in particular, the gradient of f is equal to 0 at this point.
- If the optimum is attained on the border of the set S, i.e., at the point where $g=1$, then we look for unconstrained optima of the function $f+\lambda\cdot g$, where the value of the Lagrange multiplier λ should be determined from the condition that $g=1$.

In other words:

- first, we find the unconstrained optima of the objective function f; if one or several of these unconstrained optima satisfies the constraint $g \leq 1$, we consider them;
- we also look for the optima of the function $f + \lambda \cdot g$.

In our case, the objective function is described by the formula (27) and the constraint by the formula (26). Differentiating (27) w.r.t. t_p and equating the resulting derivative to 0, we conclude that $h_p + 2\mu_p \cdot t_p = 0$, i.e., that

$$t_p = -\frac{h_p}{2\mu_p}. \tag{29}$$

Similarly, for the function $f + \lambda \cdot g$, we conclude that $h_p + 2(\mu_p + \lambda) \cdot t_p = 0$, i.e., for $\lambda \neq -\mu_p$, that

$$t_p = -\frac{h_p}{2 \cdot (\mu_p + \lambda)}, \tag{30}$$

in which case the condition (26) turns into the following equation for determining λ:

$$\sum_{p=1}^{n} \frac{h_p^2}{4 \cdot (\mu_p + \lambda)^2} = 1. \tag{31}$$

We can solve this non-linear equation with one unknown λ by applying one of the standard algorithms for solving such equations well described in numerical methods textbooks (see, e.g., [8]) such as bisection, Newton's method, secants method, etc. Once λ is found, we can compute the corresponding values t_p by using the formula (30).

Comment. It is worth mentioning that the equation (30) describes the *general case*, when $h_p \neq 0$ and thus, the case $\lambda = -\mu_p$ is impossible.

In practice, we can also have the *degenerate case*, when for some p, we have $\lambda = -\mu_p$ and $h_p = 0$. For this λ and this p, we can uniquely determine t_q for all $q \neq p$, but for this particular p, the equation $h_p + 2(\mu_p + \lambda) \cdot t_p = 0$ is satisfied for every real number t_p. In this degenerate case, we can find t_p from the condition that $t_1^2 + \ldots + t_n^2 = 1$, as

$$t_p = \pm\sqrt{1 - \sum_{q \neq p} t_q^2}.$$

For each selected combination $t = (t_1, \ldots, t_p)$, we compute the value of the objective function (27). The largest of thus computed values of the objective function is the maximum of f under the constraint, the smallest of these values is the minimum of f under this same constraint.

To complete our description, we must now explain what to do if, instead of the actual matrix A, we only know the inverse matrix $Z = A^{-1}$. In this case, what we suggest to do is to find eigenvalues and eigenvectors of the matrix Z. One can easily check that the matrices A and Z have exactly the same eigenvectors $v^{(k)}$; the only difference is that for each eigenvector, the corresponding eigenvalue $\lambda_k^{(z)}$ of the matrix Z is a reciprocal to the eigenvalue λ_k of the matrix A: $\lambda_k^{(z)} = \frac{1}{\lambda_k}$. Thus, after finding the eigenvalues $\lambda_k^{(z)}$ of the matrix Z, we must then compute the values λ_k as $\lambda_k = \frac{1}{\lambda_k^{(z)}}$.

9 How to Estimate the Range of a Quadratic Function over an Ellipsoid: Algorithm

In this algorithm, we start the coefficients f_0, f_i, and $f_{i,j}$ that describe the quadratic objective function

$$f(x_1, \ldots, x_n) = f_0 + \sum_{i=1}^{n} f_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot x_i \cdot x_j, \tag{10}$$

and with the values $\widetilde{x}_i$ and $z_{i,j}$ that characterize the constraint

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot (x_i - \widetilde{x}_i) \cdot (x_j - \widetilde{x}_j) \le 1, \tag{11}$$

in which the matrix $A = (a_{i,j})$ is the inverse to the matrix $Z = (z_{i,j})$. To find the optima of the function (10) under the constraint (11), we should do the following:

- First, we compute the values f'_0 and f'_i by using the formulas (14) and (15).
- Then, we compute the eigenvalues $\lambda_1^{(z)}, \ldots, \lambda_n^{(z)}$ of the matrix Z and the corresponding eigenvectors

$$v^{(1)} = \left(v_1^{(1)}, \ldots, v_n^{(1)}\right), \ldots, v^{(n)} = \left(v_1^{(n)}, \ldots, v_n^{(n)}\right).$$

- After that, for k from 1 to n, we compute $\lambda_k = \frac{1}{\lambda_k^{(z)}}$.
- Next, we compute the values g_k and $g_{k,l}$ by using the formulas (20) and (21).
- We compute the values g'_k and $g'_{k,l}$ by using the formula (24).
- Then, we compute the eigenvalues $\mu_1, \ldots, \mu_n$ of the matrix $(g'_{k,l})$ and the corresponding eigenvectors

$$u^{(1)} = \left(u_1^{(1)}, \ldots, u_n^{(1)}\right), \ldots, u^{(n)} = \left(u_1^{(n)}, \ldots, u_n^{(n)}\right).$$

- After that, we compute the value h_p by using the formula (28).
- Then, we compute the values t_p by using the formula (29), and check whether $\sum t_p^2 \le 1$. If this inequality is satisfied, we compute the corresponding value (27).
- After that, we solve the equation (31) with the unknown λ and, for each resulting λ, compute the value t_p by using the formula (30), and then the corresponding value of the objective function (27).
- Finally, we compute the smallest and the largest of thus computed values (27):
  - the smallest of these values is the minimum of (10) under the constraint (11); and
  - the largest of these values is the maximum of (10) under the constraint (11).

10 What If We Have an Interval of Correlation?

The above approach to eliciting the ellipsoid from the expert is based on an implicit assumption that an expert knows the relation between different parameters, and thus, the expert can provide us, for each possibly correlated pair (x_i, x_j), the value $x_{i,j}$ of the parameter x_i that is most reasonable to expect when $x_j = \overline{x}_j$.

In real life, it may happen that the expert believes that there is, e.g., a strong positive correlation between x_i and x_j, but still this expert cannot pinpoint a specific value of x_i corresponding to $x_j = \overline{x}_j$. Instead, an expert can provide us with an *interval* $[\underline{x}_{i,j}, \overline{x}_{i,j}]$ of possible values of x_i when $x_j = \overline{x}_j$.

In the case when the ellipsoid comes from the statistical information, as a confidence set of a normal distribution, different shapes of the ellipsoid corresponds to different values of the correlation between the parameters x_i and x_j. The above situation of interval uncertainty would then mean, crudely speaking, that an expert does not know the exact value of the corresponding correlation coefficient; instead, the expert knows an interval of possible values of this coefficient.

In general, in case of such interval uncertainty, inside the box

$$[\underline{x}_1, \overline{x}_1] \times \ldots \times [\underline{x}_n, \overline{x}_n], \tag{4}$$

there is the actual set S of possible parameter vectors, and this set can as well be an ellipsoid. However, in contrast to the case that we discussed before, we do not know which of the possible ellipsoids S from this box is the actual one. By selecting values $x_{i,j}$ from the corresponding intervals $[\underline{x}_{i,j}, \overline{x}_{i,j}]$, we can form different possible ellipsoids S.

Since we do not know which of these ellipsoids is the right one, we must consider all of them when estimating the range of the objective function. In other words, what we need is the range of the objective function $f(x_1, \ldots, x_n)$ over the *union* of all possible ellipsoids.

The main advantage of using ellipsoids is that, since the shape of an ellipsoid is described by a simple formula, it is feasible to compute the range of a quadratic function over an ellipsoid. A union of ellipsoids can have a much more complex shape than an ellipsoid: e.g., an arbitrary open set can be represented as a union of open balls, and a ball is, of course, a particular case of an ellipsoid. We have already mentioned that even for a box, the problem of finding the exact range of a quadratic function over it is, in general, computationally intractable (NP-hard). So, if we try to find the exact range of a quadratic function over an arbitrary union of ellipsoids, the problem becomes computationally intractable.

It is therefore reasonable, instead of considering arbitrary unions, to enclose the union into an appropriate ellipsoid and then estimate the range of a quadratic function over an enclosing ellipsoid.

Once we know the enclosing ellipsoid, we already know how to find the range of a quadratic function over it. The problem is how to describe this enclosing ellipsoid. In this section, we will describe techniques for such a description.

It is reasonable to assume that the center of the enclosing ellipsoid is at the same point $\widetilde{x} = (\widetilde{x}_1, \ldots, \widetilde{x}_n)$ that is formed by midpoints of the ranges of the corresponding parameters. Thus, the desired ellipsoid has the form

$$a(x_1, \ldots, x_n) \stackrel{\text{def}}{=} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{i,j} \cdot (x_i - \widetilde{x}_i) \cdot (x_j - \widetilde{x}_j) \le 1. \quad (9)$$

We assume that know the values $\widetilde{x}_i$. Thus, to find the shape of the ellipsoid, we must find the coefficients $a_{i,j}$.

For some "correlated" pairs of parameters (x_i, x_j), we know the interval $[\underline{x}_{i,j}, \overline{x}_{i,j}]$ of possible values of x_i when $x_j = \overline{x}_j$. How can we transform this information into the values of the coefficients $a_{i,j}$?

To explain how this can be done, let us start with the simplest 2-D case, when an ellipsoid is simply an ellipse. In this case, we know that the intersection of the enclosing ellipsoid with the line $x_2 = \overline{x}_2$ consists of the interval $[\underline{x}_{1,2}, \overline{x}_{1,2}]$. Thus, the endpoints of this interval should belong to the surface of this ellipsoid. In other words, the surface of the ellipsoid should include the points $(\underline{x}_{1,2}, \overline{x}_2)$ and $(\overline{x}_{1,2}, \overline{x}_2)$. In other words, for these points, the left-hand side of the inequality (9) should be exactly equal to 1. Since $\overline{x}_2 - \widetilde{x}_2 = \Delta_2$, we conclude that:

$$a_{1,1} \cdot (\underline{x}_{1,2} - \widetilde{x}_1)^2 + 2 \cdot a_{1,2} \cdot (\underline{x}_{1,2} - \widetilde{x}_1) \cdot \Delta_2 + a_{2,2} \cdot \Delta_2^2 = 1; \tag{32}$$

$$a_{1,1} \cdot (\overline{x}_{1,2} - \widetilde{x}_1)^2 + 2 \cdot a_{1,2} \cdot (\overline{x}_{1,2} - \widetilde{x}_1) \cdot \Delta_2 + a_{2,2} \cdot \Delta_2^2 = 1. \tag{33}$$

Thus, we get two equations relating the three unknowns $a_{1,1}$, $a_{1,2}$, and $a_{2,2}$.

We can similarly ask what are the possible values of x_2 when $x_1 = \overline{x}_1$; as a result, we will get an interval $[\underline{x}_{2,1}, \overline{x}_{2,1}]$. Based on this information, we can describe two more equations relating the three unknowns:

$$a_{1,1} \cdot \Delta_1^2 + 2 \cdot a_{1,2} \cdot \Delta_1 \cdot (\underline{x}_{2,1} - \widetilde{x}_2) + a_{2,2} \cdot (\underline{x}_{2,1} - \widetilde{x}_2)^2 = 1; \tag{34}$$

$$a_{1,1} \cdot \Delta_1^2 + 2 \cdot a_{1,2} \cdot \Delta_1 \cdot (\overline{x}_{2,1} - \widetilde{x}_2) + a_{2,2} \cdot (\overline{x}_{2,1} - \widetilde{x}_2)^2 = 1. \tag{35}$$

Now, we can either select 3 out of 4 equations, or apply the Least Squares Method to all 4 equations, and get all three desired coefficients $a_{1,1}$, $a_{1,2}$, $a_{2,2}$ that describe our ellipse.

In the general n-dimensional case, we can apply this procedure for every correlated pair of parameters (x_i, x_j), and get a description of the corresponding ellipse

$$a^{(i,j)}_{i,i} \cdot (x_i - \widetilde{x}_i)^2 + 2 \cdot a^{(i,j)}_{i,j} \cdot (x_i - \widetilde{x}_i) \cdot (x_j - \widetilde{x}_j) + a^{(i,j)}_{j,j} \cdot (x_j - \widetilde{x}_j)^2 \le 1. \tag{36}$$

For independent variables (x_i, x_j), we can assume (as we did before) that $x_{i,j} = \widetilde{x}_i$ and hence, the corresponding ellipse takes the form:

$$a^{(i,j)}_{i,i} \cdot (x_i - \widetilde{x}_i)^2 + a^{(i,j)}_{j,j} \cdot (x_j - \widetilde{x}_j)^2 \le 1, \tag{37}$$

where $a^{(i,j)}_{i,i} = \Delta_i^{-2}$ and $a^{(i,j)}_{j,j} = \Delta_j^{-2}$. Thus, for every two parameters x_i and x_j, we know projection of the desired n-dimensional enclosing ellipsoid (9) onto the 2-D plane (x_i, x_j). How can we reconstruct the ellipsoid from its projections?

It is possible to check that if we go from the matrix $a_{k,l}$ to the inverse matrix $z_{k,l}$, then the projection to a 2-D plane corresponds simply to restricting the matrix $z_{k,l}$ to the corresponding variables x_i and x_j:

- The algebraic proof of this fact is similar to our analysis from Appendix 1.
- Due to the fact that for normal distribution, confidence sets are ellipsoids, this fact is also easy to understand in statistical terms: for the normal distribution

$$\rho(x_1, \ldots, x_n) = \frac{1}{(2 \cdot \pi)^{n/2} \cdot \det(A)} \cdot \exp\left(- \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot (x_i - a_i) \cdot (x_j - a_j) \right),$$

the corresponding matrix $a_{i,j}$ is the (half of the) inverse matrix to the covariance matrix $E[(x_i - a_i) \cdot (x_j - a_j)]$. When, instead of considering all n random variables x_i, we only consider some of them, all we have to do is restrict to covariance matrix to the corresponding variables.

Due to this fact, to reconstruct the ellipsoid from its 2-D projections, we can do the following:

- First, for every i and j, we invert the corresponding matrix $a^{(i,j}_{k,l}$ into a matrix $z^{(i,j)}_{k,l}$.
- Then, we combine the values $z^{(i,j)}_{k,l}$ into a single matrix:
  - For $i \neq j$, the value $z_{i,j}$ is only present in $z^{(i,j)}_{i,j}$, so we take $z_{i,j} = z^{(i,j}_{i,j}$.
  - For $i = j$, the value $z_{i,i}$ occurs as $z^{(i,j)}_{i,i}$ for different $j \neq i$.
    We are interested in finding the enclosing ellipsoid, i.e., the ellipsoid that, crudely speaking, contains all projected ellipses. In terms of $a_{i,i}$, the larger the value, the larger the product $a_{i,i} \cdot \Delta x_i^2$ and thus, the more restrictive is the corresponding inequality $a_{i,i} \cdot \Delta x_i^2 + \ldots \leq 1$. Thus, in terms of $a_{i,i}$, if we want to the most enclosing ellipsoid, we must select the *smallest* possible value of $a_{i,i}$. The values $z_{i,j}$ are inverse to $a_{i,j}$.

    Thus, it is reasonable to select the *largest* possible value of $z_{i,i}$:

$$z_{i,i} = \min_{j \neq i} z^{(i,j)}_{i,i}.$$

As a result, we get the matrix $Z = (z_{i,j})$ that is the inverse to the matrix $A = (a_{i,j})$ that describes the desired enclosing ellipsoid.

Knowing Z, we can now use the above-described algorithm to determine the range of a given quadratic function over the corresponding ellipsoid.

11 What If We Have a Nested Family of Intervals of Correlation?

The next natural question is: what if the experts can provide us additional information beyond the intervals?

Often, when an expert can provide us with an interval $[\underline{x}, \overline{x}]$ that is guaranteed to contain the value of some quantity x, this expert can also provide narrower intervals that contain x with different degrees of certainty α.

The corresponding family of nested intervals can be viewed as a particular case of a fuzzy set [15,16]: if a traditional fuzzy set is given, then different intervals from the nested family can be viewed as α-cuts corresponding to different levels of uncertainty α. It can also be viewed as a particular case of a Dempster-Shafer (DS) knowledge base, or as a multi-variate analogue of a p-box [4].

In this case, for each degree of certainty α, we can take the intervals corresponding to this degree of certainty, compute the corresponding ellipsoid, and then follow the above-described algorithm to estimate the range of the given quadratic function over this ellipsoid. As a result, for each α, we get the range estimate corresponding to this α.

In other words, as a result, instead of a single interval range, we get a nested family of interval ranges corresponding to different levels of certainty α. This nested family of intervals can also viewed either as a fuzzy set, or as a DS knowledge base, or as a p-box.

References

[1] G. Belfonte and B. Bona, "An improved parameter identification algorithm for signal with unknown-but-bounded errors", *Proc. 7th IFAC Symposium on Identification and Parameter Estimation*, York, UK, 1985.

[2] F. L. Chernousko, *Estimation of the Phase Space of Dynamical Systems*, Nauka Publ., Moscow, 1988 (in Russian).

[3] F. L. Chernousko, *State Estimation for Dynamical Systems*, CRC Press, Boca Raton, Florida, 1994.

[4] S. Ferson, *RAMAS Risk Calc 4.0*, CRC Press, Boca Raton, Florida, 2002.

[5] A. F. Filippov, "Ellipsoidal estimates for a solution of a system of differential equations", *Interval Computations*, 1992, No. 2(4), pp. 6–17.

[6] A. Finkelstein, O. Kosheleva, and V. Kreinovich, "Astrogeometry, error estimation, and other applications of set-valued analysis", *ACM SIGNUM Newsletter*, 1996, Vol. 31, No. 4, pp. 3–25.

[7] E. Fogel and Y. F. Huang, "One the value of information in system identification. Bounded noise case", *Automatica*, 1982, Vol. 18, pp. 229–238.

[8] C. F. Gerald and P. O. Wheatley, *Applied Numerical Methods*, Addison-Wesley, Reading, Massachusetts, 2004.

[9] G. H. Golub and C. F. Van Loan, *Matrix Computations*, John Hopkins University Press, Baltimore, 1996.

[10] A. Kandel, D. Ramot, M. Friedman, and G. Langholz, "Complex Fuzzy Logic", *IEEE Transactions on Fuzzy Systems*, 2003, Vol. 11, No. 4, pp. 450–461.

[11] A. Kandel, D. Ramot, R. Milo, and M. Friedman, “Complex Fuzzy Sets”, *IEEE Transactions on Fuzzy Systems*, 2002, Vol. 10, No. 2, pp. 171–186.

[12] V. Kreinovich, A. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer, Dordrecht, 1998.

[13] S. Li, Y. Ogura, and V. Kreinovich, *Limit Theorems and Applications of Set Valued and Fuzzy Valued Random Variables*, Kluwer Academic Publishers, Dordrecht, 2002.

[14] H. T. Nguyen, A. Kandel, and V. Kreinovich, “Complex Fuzzy Sets: Towards New Foundations”, *Proceedings of the 9th IEEE International Conference on Fuzzy Systems FUZZ-IEEE’2000*, San Antonio, Texas, May 7–10, 2000, Vol. 2, pp. 1045–1048.

[15] H. T. Nguyen and V. Kreinovich, “Nested Intervals and Sets: Concepts, Relations to Fuzzy Sets, and Applications”, In: R. B. Kearfott and Vladik Kreinovich, *Applications of Interval Computations*, Kluwer, Dordrecht, 1996, pp. 245–290

[16] H. T. Nguyen and E. A. Walker, *First Course in Fuzzy Logic*, CRC Press, Boca Raton, Florida, 1999.

[17] F. C. Schweppe, “Recursive state estimation: unknown but bounded errors and system inputs”, *IEEE Transactions on Automatic Control*, 1968, Vol. 13, p. 22.

[18] F. C. Schweppe, *Uncertain Dynamics Systems*, Prentice Hall, Englewood Cliffs, New Jersey, 1973.

[19] S. T. Soltanov, “Asymptotic of the function of the outer estimation ellipsoid for a linear singularly perturbed controlled system”, In: S. P. Shary and S. P. Shokin (eds.), *Interval Analysis*, Krasnoyarsk, Academy of Sciences Computing Center, Publication No. 17, 1990, pp. 35–40 (in Russian).

[20] R. Trejo and V. Kreinovich, “Error Estimations for Indirect Measurements: Randomized vs. Deterministic Algorithms For ‘Black-Box’ Programs”, In: S. Rajasekaran, P. Pardalos, J. Reif, and J. Rolim (eds.), *Handbook on Randomized Computing*, Kluwer, 2001, pp. 673–729.

[21] G. S. Utyubaev, “On the ellipsoid method for a system of linear differential equations”, In: S. P. Shary (ed.), *Interval Analysis*, Krasnoyarsk, Academy of Sciences Computing Center, Publication No. 16, 1990, pp. 29–32 (in Russian).

[22] S. A. Vavasis, *Nonlinear Optimization: Complexity Issues*, Oxford University Press, New York, 1991.

[23] H. M. Wadsworth Jr., *Handbook of statistical methods for engineers and scientists*, McGraw-Hill, N.Y., 1990.

Appendix 1. How to Elicit the Ellipsoid from an Expert? Derivation of the Corresponding Formulas

How can we determine the center $\widetilde{x}$ of the ellipsoid based on the known information? From the formula (9), one can easily see that an ellipsoid is symmetric with respect to the transformation $\Delta x_i \to -\Delta x_i$; in other words, if we start with a point inside the ellipsoid, and we change the signs of all the values Δx_i, then we get the point that is also inside the ellipsoid. So, if a value Δx_i is possible (i.e., occurs for some point within the ellipsoid), its negative $-\Delta x_i$ is also possible. Thus, the set of possible values of $\Delta x_i = x_i - \widetilde{x}_i$ is symmetric w.r.t. 0, and hence, the set of possible values of x_i is symmetric w.r.t. $\widetilde{x}_i$. In other words, the value $\widetilde{x}_i$ is a midpoint of the range of possible values of x_i.

We know the range of possible values of x_i – this range is provided, by an expert, as an interval $[\underline{x}_i, \overline{x}_i]$. Thus, we can determine $\widetilde{x}_i$ as the midpoint of this interval, i.e., as $(\underline{x}_i + \overline{x}_i)/2$.

Let us now look at the desired relation between x_i and x_j. Since the value $\overline{x}_j$ is possible, the ellipsoid has an intersection with the (hyper-)plane $x_j = \overline{x}_j$. As we increase x_j further, to a value $x_j = \overline{x}_j + \varepsilon$ for some small $\varepsilon > 0$, we leave the set of possible values of x_j and therefore, the ellipsoid has no intersection with the corresponding plane $x_j = \overline{x}_j + \varepsilon$. Thus, the plane $x_j = \overline{x}_j$ is a *tangent plane* to our ellipsoid.

It is known that the ellipsoid is strictly convex, so at any given point, it has only one point of intersection with its tangent plane. Thus, when $x_j = \overline{x}_j$, there is only one point in the ellipsoid. Therefore, when we ask the expert – as we intended – about the range of possible values of x_i when $x_j = \overline{x}_j$, we should expect not the range but rather a single value of x_j. In other words, instead of asking for a range, it makes sense to ask, for each correlated values i and j, what is the most reasonable value of x_i when $x_j = \overline{x}_j$. We will denote this "most reasonable value" by $x_{i,j}$.

By eliciting this information from the expert, we can therefore extract E values $x_{i,j}$ corresponding to the E pairs that the expert ranked as most correlated.

In order to find out what are the values $x_{i,j}$ for all other pairs, let us analyze how this value $x_{i,j}$ is related to $\widetilde{x}_i$.

- When there is a positive correlation between the parameters x_i and x_j, then, in accordance with our description of what positive correlation means, we expect x_i to grow with x_j. In other words, since $\overline{x}_j > \widetilde{x}_j$, we expect $x_{i,j} > \widetilde{x}_i$.
- When there is a negative correlation between the parameters x_i and x_j, then, in accordance with our description of what negative correlation means, we expect x_i to decrease when x_j increases. In other words, since $\overline{x}_j > \widetilde{x}_j$, we

expect $x_{i,j} < \widetilde{x}_i$.

So, if there is no correlation, we expect that $x_{i,j}$ is neither larger nor smaller than $\widetilde{x}_i$ – i.e., we expect $x_{i,j} = \widetilde{x}_i$. Thus, for all the pairs of parameters (x_i, x_j) that an expert considers to be (approximately) independent, we take $x_{i,j} = \widetilde{x}_i$.

Now, for every i, we have the values $\underline{x}_i$, $\overline{x}_i$, and $\widetilde{x}_i$, and for all pairs $i \neq j$, we know the value $x_{i,j}$ of x_i when $x_j = \overline{x}_j$. How can we transform this information into the coefficients $a_{i,j}$ of the desired ellipsoid (9)?

We have mentioned that at a point at which $x_j = \overline{x}_j$, the plane $x_j = \overline{x}_j$ is a tangent to the ellipsoid. By definition of the values $x_{i,j}$, this means that at the point $(x_{1,j}, \ldots, x_{j-1,j}, \overline{x}_j, x_{j+1,j}, \ldots, x_n)$, the plane $x_j = \overline{x}_j$ is a tangent to the ellipsoid. To make computations simpler, let us denote $\overline{x}_j$ by $x_{j,j}$. In this new notation, the previous statement takes the following form: the plane $x_j = \overline{x}_j$ is tangent to the ellipsoid at the point with the coordinates $(x_{1,j}, \ldots, x_{n,j})$.

From calculus, it is known that a tangent vector to the surface is orthogonal to its normal vector n, and the normal vector to the surface $a(x_1, \ldots, x_n) = 1$ is proportional to the gradient of f, i.e., to the vector

$$\nabla a = \left(\frac{\partial a}{\partial x_1}, \ldots, \frac{\partial a}{\partial x_n}\right).$$

This gradient vector should be orthogonal to the plane $x_j = \overline{x}_j$; thus, the only non-zero component of this gradient vector is its j-th component, and all other components are 0. In other words, at the point $(x_{1,j}, \ldots, x_{n,j})$, we have $\dfrac{\partial a}{\partial x_i} = 0$ for all $i \neq j$.

Let us use the expression (9) for the function $a(x_1, \ldots, x_n)$ to find an explicit expression for this partial derivative. For that, let us first separate, in the expression (9), the terms that depend on x_i and the terms that do not depend on x_i. As a result, we arrive at the following formula:

$$a(x_1, \ldots, x_n) = a_{i,i} \cdot (x_i - \widetilde{x}_i)^2 + \sum_{k \neq i} a_{i,k} \cdot (x_i - \widetilde{x}_i) \cdot (x_k - \widetilde{x}_k) +$$

$$\sum_{k \neq i} a_{k,i} \cdot (x_k - \widetilde{x}_k) \cdot (x_i - \widetilde{x}_i) + \sum_{k \neq i} \sum_{l \neq i} a_{k,l} \cdot (x_k - \widetilde{x}_k) \cdot (x_l - \widetilde{x}_l).$$

Differentiating w.r.t. x_i, we conclude that

$$\frac{\partial a}{\partial x_i} = 2a_{i,i} \cdot (x_i - \widetilde{x}_i) + \sum_{k \neq i} a_{i,k} \cdot (x_k - \widetilde{x}_k) + \sum_{k \neq i} a_{k,i} \cdot (x_k - \widetilde{x}_k),$$

i.e., since $a_{i,j}$ is a symmetric matrix, that

$$\frac{\partial a}{\partial x_i} = 2a_{i,i} \cdot (x_i - \widetilde{x}_i) + 2\sum_{k\ne i} a_{i,k} \cdot (x_k - \widetilde{x}_k),$$

and, finally, that

$$\frac{\partial a}{\partial x_i} = 2\sum_{k=1}^{n} a_{i,k} \cdot (x_k - \widetilde{x}_k).$$

At the point with coordinates $x_k = x_{k,j}$, these derivative must be equal to 0, so we conclude that

$$\sum_{k=1}^{n} a_{i,k} \cdot (x_{k,j} - \widetilde{x}_k) = 0. \tag{A1}$$

for every $i \ne j$.

The point with the coordinates $x_{1,j}, \ldots, x_{n,j}$ is on the surface of the ellipsoid (9), so, for this point, we have $a(x_{1,j}, \ldots, x_{n,k}) = 1$. Substituting the values $x_{k,j}$ into the formula (9), we conclude that

$$\sum_{i=1}^{n}\sum_{k=1}^{n} a_{i,k} \cdot (x_{i,j} - \widetilde{x}_i) \cdot (x_{k,j} - \widetilde{x}_k) = 1.$$

This equation can be rewritten as follows:

$$\sum_{i=1}^{n} (x_{i,j} - \widetilde{x}_i) \cdot \left(\sum_{k=1}^{n} a_{i,k} \cdot (x_{k,j} - \widetilde{x}_k)\right) = 1. \tag{A2}$$

Due to the equation (A1), for all $i \ne j$, we get 0, and the only non-zero term is when $i = j$. Thus, in the formula (A2), we can replace the first sum with the j-th term:

$$(x_{j,j} - \widetilde{x}_j) \cdot \left(\sum_{k=1}^{n} a_{j,k} \cdot (x_{k,j} - \widetilde{x}_k)\right) = 1,$$

or, equivalently,

$$\sum_{k=1}^{n} a_{j,k} \cdot (x_{k,j} - \widetilde{x}_k) \cdot \Delta_j = 1, \tag{A3}$$

where we denoted $\Delta_j \overset{\text{def}}{=} x_{j,j} - \widetilde{x}_j = \overline{x}_i - \widetilde{x}_j = (\overline{x}_j - \underline{x}_j)/2$. Multiplying both sides of the equation (A1) by Δ_j, we conclude that

$$\sum_{k=1}^{n} a_{i,k} \cdot (x_{k,j} - \widetilde{x}_k) \cdot \Delta_j = 0 \tag{A4}$$

for all $i \ne j$. Equations (A3) and (A4) can be combined into a single equation

$$\sum_{k=1}^{n} a_{i,k} \cdot (x_{k,j} - \widetilde{x}_k) \cdot \Delta_j = \delta_{i,j}, \tag{A5}$$

where $\delta_{i,j} = 1$ for $i = j$ and $\delta_{i,j} = 0$ for $i \ne j$.

These value of $\delta_{i,j}$ are components of the unit matrix I, and the sum left-hand side of the formula (A5) describes the product $A \cdot X$ of the two following matrices:

- the desired matrix A, with components $a_{i,k}$, and
- the matrix Z, with components $z_{k,j} \overset{\text{def}}{=} (x_{k,j} - \widetilde{x}_k) \cdot \Delta_j$.

Since $A \cdot Z = I$, we thus have $A = Z^{-1}$, so we can obtain A by simply inverting the matrix Z.

Appendix 2. Optimizing Quadratic Function over an Ellipsoid: Derivation of the Corresponding Formulas

Let us first derive the formulas corresponding to replacing the original variables x_i with the new variables Δx_i, i.e., the formulas that are obtained by substituting the expression $x_i = \widetilde{x}_i + \Delta x_i$ into the objective function (10). After substitution, we get the following expression:

$$f = f_0 + \sum_{i=1}^n f_i \cdot (\widetilde{x}_i + \Delta x_i) + \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot (\widetilde{x}_i + \Delta x_i) \cdot (\widetilde{x}_j + \Delta x_j).$$

Opening parentheses, we conclude that

$$f = f_0 + \sum_{i=1}^n f_i \cdot \widetilde{x}_i + \sum_{i=1}^n f_i \cdot \Delta x_i + \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot \widetilde{x}_i \cdot \widetilde{x}_j +$$

$$\sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot \widetilde{x}_i \cdot \Delta x_j + \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot \Delta x_i \cdot \widetilde{x}_j + \sum_{i=1}^n \sum_{j=1}^n a_{i,j} \cdot \Delta x_i \cdot \Delta x_j.$$

Grouping together terms that do not contain Δx_i at all, terms that are linear in Δx_i, and terms that are quadratic in Δx_i, we get the desired formulas (13)–(15).

Let us now derive the formulas (18)–(21). By definition,

$$\Delta x_i = \Delta x \cdot e^{(i)}. \tag{A6}$$

We want to represent Δx_i in terms of the values $y^k = \Delta x \cdot v^{(k)}$. To do that, let us expand the vector $e^{(i)}$ into the base $v^{(1)}, \ldots, v^{(n)}$:

$$e^{(i)} = \sum_{k=1}^n (e^{(i)} \cdot v^{(k)}) \cdot v^{(k)}. \tag{A7}$$

We know the coordinates $v_i^{(k)}$ of each eigenvector $v^{(k)}$ in the standard basis $e^{(i)}$, so we can explicitly compute $e^{(i)} \cdot v^{(k)}$ as $v_i^{(k)}$. Thus, (A7) turns into the following formula:

$$e^{(i)} = \sum_{k=1}^{n} v_i^{(k)} \cdot v^{(k)}. \tag{A8}$$

If we use scalar product to multiply both sides of (A8) by Δx, we conclude that

$$e^{(i)} \cdot \Delta x = \sum_{k=1}^{n} v_i^{(k)} \cdot (v^{(k)} \cdot \Delta x). \tag{A9}$$

We know that $e^{(i)} \cdot \Delta x = \Delta x_i$ and that $v^{(k)} \cdot \Delta x = y_k$, so (A9) takes the following form:

$$\Delta x_i = \sum_{k=1}^{n} v_i^{(k)} \cdot y_k,$$

which is exactly the desired formula (18). Substituting this expression (18) into the formula (13), we conclude that

$$f(x_1, \ldots, x_n) = f_0' + \sum_{i=1}^{n} \sum_{k=1}^{n} f_i' \cdot v_i^{(k)} \cdot y_k + \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} \sum_{l=1}^{n} f_{i,j} \cdot v_i^{(k)} \cdot v_i^{(l)} \cdot y_k \cdot y_l,$$

i.e., that

$$f(x_1, \ldots, x_n) =$$
$$f_0' + \sum_{k=1}^{n} \left(\sum_{i=1}^{n} f_i' \cdot v_i^{(k)} \right) \cdot y_k + \sum_{k=1}^{n} \sum_{l=1}^{n} \left(\sum_{i=1}^{n} \sum_{j=1}^{n} f_{i,j} \cdot v_i^{(k)} \cdot v_i^{(l)} \right) \cdot y_k \cdot y_l,$$

which is exactly the desired formulas (19)–(21).

Let us now derive the formulas (26)–(28). The formula (25) means that $t_p = u^{(p)} \cdot y$. Similarly to the above derivation of the formula (18), we conclude that

$$y_k = \sum_{p=1}^{n} u_k^{(p)} \cdot t_p.$$

Substituting this expression into the formula (19) for the objective function, we conclude – similar to the first time when we used the eigenvectors – that the quadratic part turns into $\sum \mu_p \cdot t_p^2$, and that the the linear part turns into

$$\sum_{k=1}^{n} g_k \cdot \left(\sum_{p=1}^{n} u_k^{(p)} \cdot t_p \right) = \sum_{p=1}^{n} \left(\sum_{k=1}^{n} u_k^{(p)} \cdot g_k \right) \cdot t_p,$$

i.e., exactly to the formulas (27)–(28).

To complete the derivation, we must show that (22) indeed turns into (26). Indeed:

- The values y_k are coordinates of the vector y with respect to the standard orthonormal basis $(1, 0, \ldots, 0)$, $\ldots$, $(0, \ldots, 0, 1)$. The sum $\sum y_k^2$ is thus the square of the length of this vector.

- The values t_p are coordinates of the same vector y with respect to a orthonormal basis: namely, the basis formed by the eigenvectors $u^{(1)}, \dots, u^{(n)}$. The sum $\sum t_p^2$ is thus also equal to the square of the length of this vector.

Thus, the sums $\sum y_k^2$ and $\sum t_p^2$ are always equal, hence the condition (22) is indeed equivalent to the condition (26).