Lesson 1

Introduction to Multivariate Modeling

Statistics tell the biggest lies, everybody knows that ...! — However, this is not *necessarily* true. It all depends on the user who is interpreting the results: If the statistical methods are applied appropriately, by somebody who understands their properties, excellent results are often reached. Gaining this understanding is the objective of this report.

1.1 About systems and models

The role of a *model* is to organize information about a given system. There are various questions that need to be answered when a model is being constructed — the selection of the modeling principle being perhaps the most significant, affecting all subsequent analysis. First, one can start from the physical first principles, constructing the model in the bottom-up fashion. However, this modeling style is very knowledge-intensive, and the scarcity of the domain-area experts is an ever increasing problem. And, what is more, the larger the system is, the more probable it is that such a model will *not really be suited for real use,* for analysis of the system behavior or for control design. To understand this, see Fig. 1.1: What is the dimension of this constant-volume system, what is the number of independent system state variables?

The intuition tells us that there are three independent state variables — the concentrations in each tank needs to be represented in the system model. The problem with qualitative analyses is that the relevance of different constructs is not at all judged. For example, assume that one of the tanks is negligible, so that its volume is small as compared to the other two tanks: This tank does not contribute in the system behavior, and the nominal three-state model is unnecessarily complex for practical use. On the extreme, all the system time constants may be negligible as compared to the sampling interval, and, seemingly, the system becomes static with no dynamics whatsoever. On the other hand, assume that the mixing is not perfect: If the tanks are not ideal mixers, the one-state-per-tank approach is no more sufficient, and one ends in

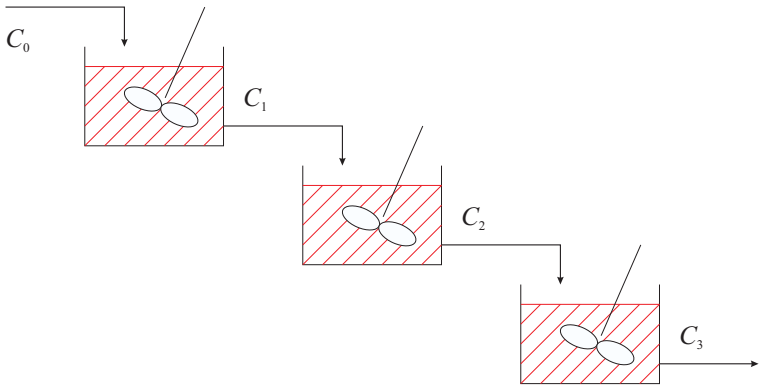

Figure 1.1: What is the dimension of the dynamic system?

a partial differential equation model. How many state variables are needed to reach sufficient accuracy is dependent of the actual system. The *relevant* dimension of the system can be anything between zero and infinity — and, indeed, for partial differential equation models, one can even speak of non-integer dimensions! Appropriate model structure is dependent of the intended use of the model.

As compared to bottom-up methods, the multivariate statistical methods operate in the top-down fashion: Plenty of data is collected and one tries to find the relevant system properties underlying the measurements. Instead of being knowledge-intensive, multivariate methods are *data-intensive.* The field of systems engineering is getting wider and wider, and the systems to be modeled are getting more and more complicated and less structured (see Fig. 1.2). All these trends necessitate data-oriented approaches in both ends of the systems continuum.

Another fact is that totally new branches of research — for example, *data mining* and *microactuators* must be based on massively parallel analysis and modeling methods.

The field of modern multivariate methods is wide and heterogeneous. Researchers in different disciplines typically have different objectives and application fields, and certainly they do have differing terminology and notations. What is more, the methods are still in turmoil and their overall relevance has not yet been generally understood. Some examples:

- Since the Gaussian times, least-squares mapping has been the standard technique in all fields of science. This methodology matured well before any data-orientation became a hot topic, and it seems that it is not typically seen in the wider perspective, in connection to other multivariate methods. It can be assumed that a great number of scientists and engineers suffer from its deficiencies, having to force their data into an unnatural least-squares-conditioned model structure, never getting acquainted with alternatives.

- In chemical engineering, for example, where calibration of devices is of utmost importance, a method called Partial Least Squares is routinely ap-

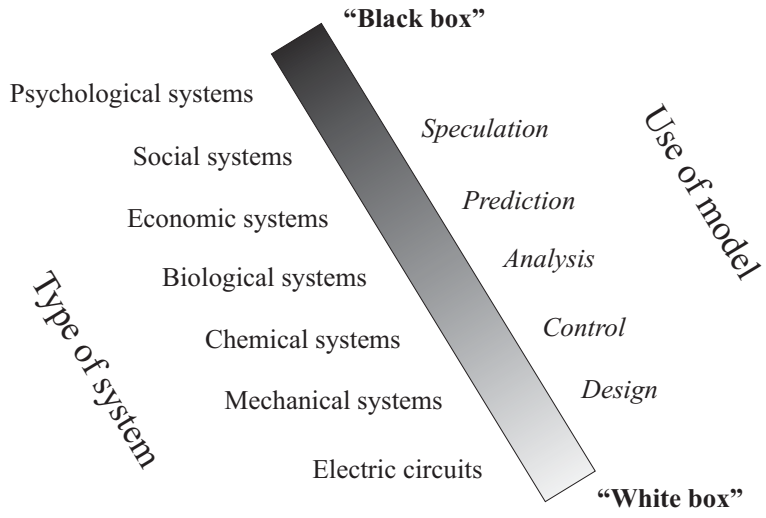

Figure 1.2: Spectrum of systems to be modeled

plied. However, being based on simple matching of correlations, employing unpenetrable algorithms, the uncompromising and ambitious mathematicians and statisticians are not very impressed or interested. Their answer to similar problems is Canonical Correlation Regression — a method that seems to be inaccessible for a practicing engineer. The unfortunate fact is that these mental barriers are caused simply by different terminologies and practices in these communities: The underlying ideas turn out to be closely related.

- The neural networks have become popular in almost all complex data modeling applications. In these research circles there seem to exist prejudices against the "outdated" statistical methods — but it is not only the "postmodernists" to blame: In the traditional school, there exist similar scornful attitudes towards the "heuristic" neural network methods. Again, beyond the surface, there is very much in common.

- Finally, in control engineering, dynamic models are often regarded as the only "interesting" models. However, the dynamic models can often better be understood when the simpler, static models are studied — indeed, dynamic modeling is static modeling with appropriately chosen data. On the other hand, static data samples are seldom independent of each other and dynamic understanding can reveal additional structure beyond that data, so that, again, it would be nice if these two approaches, static and dynamic, could be presented in a consistent setting.

It is difficult to see the underlying relationships among different approaches. An illustration of this heterogeneity in the field of data-oriented modeling is the fact that there seems not to exist an engineering level treatment of all relevant methods in a common framework. This report tries to do that: To offer a homogeneous view over the various disciplines in multivariate analysis.

1.2 About mathematical tools

Its is *mathematics* that is the natural language of nature. To fluently "think" using the syntactical structures defined in that language, the appropriate concepts need to be, not only familiar, but they have to belong to one's *active vocabulary.*

1.2.1 Challenge of high dimensions

In multivariate framework the structural complexity of the system is changed into a problem of high dimensionality: It is assumed that when one includes enough measurement data in the model, arbitrary accuracy concerning the system behavior can be achieved.

One needs to keep in mind the lesson learned in "Flatland" [1]:

> Assume that there existed *two-dimensional* creatures, being only able to recognize their environment in two dimensions. Now, assume that a three-dimensional ball goes through this two-dimensional world — what the creatures perceive, is a point coming from nowhere, expanding into a circle, and finally again contracting into a dot before vanishing. This circle can emerge anywhere in the two-dimensional space. How can the creatures understand what happened?

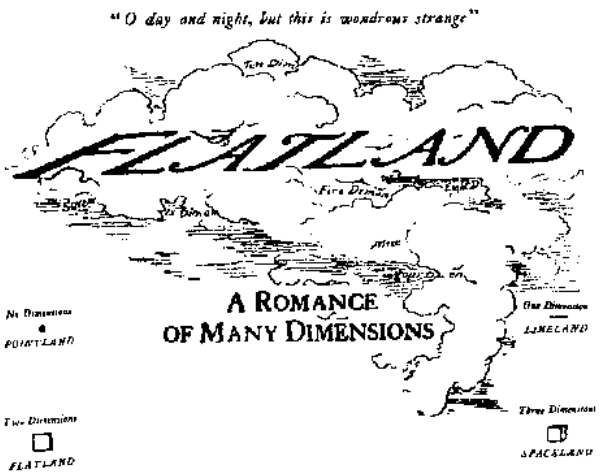

Figure 1.3: Cover page of E. Abbott's "Flatland"

The answer is that this behavior exceeds the capacity of those creatures, there is no way they could really understand it. The point is that *we* are bound to the three-dimensional world — the behavior of higher-dimensional objects similarly exceeds our capacity. Or, as J. Hadamard put it:

> *Give me 100 parameters and I will construct an elephant; give me 101 parameters, and I will make it wag its tail.*

This incomprehensibility is the basic problem when studying multivariate methods: The phenomena emerging in higher dimensions cannot really be visualized as two-dimensional plots (or even three-dimensional wire models). When more sophisticated methods are analyzed, illustrations do not help much, and common sense intuitions are of no use.

How the above problems with high data dimensionality are reflected in practice is perhaps best illustrated by an example: Assume that behaviors of a scalar (one-dimensional) function can be captured along a line; a two-parameter function spans the whole plane, and a three-parameter function spans the three-dimensional space. This means that to reach the same accuracy in each case, to cover the space equelly, the claim for data grows *exponentially.* To master the dimensional complexity, it is evident that one has to make strong assumptions to constrain the model structures.

Mathematics is a robust tool to attack the above challenges, offering stronger concepts and grammar for discussing multivariate phenomena.

The way to reach reasonable restrictions on the model structures, is to assume *linearity.* For linear models, essentially the same methodologies work no matter what is the dimension of the problem. This means that it is *linear algebra* that is the theoretical framework for studying multivariate statistics, and it is *matrix calculus* that is the practical language for implementing models for high-dimensional phenomena. Good understanding of these conceptual tools is vital.

1.2.2 About matrices

When doing multivariate modeling, data is (hopefully) received in huge numbers, and some kind of standardization of representations is necessary. It is assumed here that the only data structure that is employed is a *data matrix,* following the original `Matlab` style course of operation. The matrices will then have different roles: They are used as data storages, but also as frames for vector systems, and as linear operators representing linear mappings. In each case, it is *matrix operations* that are applied to manipulate the data structures.

The principles of matrix calculus are *not* repeated here (for more information, see, for example, [2] or [11]). It is assumed that *matrix inverses,* etc., are familiar; however, let us repeat what are *eigenvalues* and *eigenvectors,* what are *singular values,* and how matrix-form expressions are differentiated and how their extrema can be found. The discussion is restricted to real-valued matrices.

Eigenvalues and eigenvectors

It turns out that, rather astonishingly, most of the major regression methods can be presented in a homogeneous framework; this is the framework of the so called *eigenproblem.* A square matrix M of dimension $n \times n$ generally fulfills the following formula for some ξ and λ:

$$M \cdot \xi = \lambda \cdot \xi. \tag{1.1}$$

Here, λ is a scalar called *eigenvalue* and ξ is a vector called *eigenvector.* This means that the eigenvector directions are, in a sense, "natural" to the matrix M: Vectors in this direction, when multiplied by M, are only scaled, so that their direction is not changed (later, when speaking of linear mappings, this property will be elaborated on further). From the construction, it is clear that if ξ is eigenvecter, then $\alpha\xi$ also is, where α is an arbitrary scalar. For uniqueness, from now on, it will be assumed that the eigenvectors are always normalized, so that $\|\xi\| = \sqrt{\xi^T\xi} = 1$ (this constraint is automatically fulfilled by the eigenvectors that are returned by the `eig` function of `Matlab`, for example).

In those cases that we will study later the matrices will be *non-defective* by construction (however, see the exercise); this means that there will exist n distinct eigenvectors fulfilling the expression (1.1). These eigenvectors ξ_i and corresponding eigenvalues θ_i, where $1 \leq i \leq n$, can be compactly presented as matrices Ξ and Λ, respectively:

$$\Xi = \left(\begin{array}{c|c|c} \xi_1 & \cdots & \xi_n \end{array} \right) \qquad \text{and} \qquad \Lambda = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_n \end{pmatrix}, \tag{1.2}$$

where the dimension of matrices Ξ and Λ is $n \times n$. Using these notations, it is easy to verify that the n solutions to the eigenproblem (1.1) can now be expressed simulateneously in a compact matrix form as

$$M \cdot \Xi = \Xi \cdot \Lambda. \tag{1.3}$$

In those cases that we will be later studying, the eigenvectors are linearly independent, Ξ has full rank and the matrix M is *diagonalizable:* The above expression equals

$$\Xi^{-1} \cdot M \cdot \Xi = \Lambda, \tag{1.4}$$

or

$$M = \Xi \cdot \Lambda \cdot \Xi^{-1}, \tag{1.5}$$

so that M is *similar* to a diagonal matrix consisting of its eigenvalues. One of the specially useful properties of the above *eigenvalue decomposition* is due to the following:

$$\begin{array}{rcl} M^i & = & \left(\Xi \cdot \Lambda \cdot \Xi^{-1}\right)^i \\ & = & \underbrace{\Xi \cdot \Lambda \cdot \Xi^{-1} \cdots \Xi \cdot \Lambda \cdot \Xi^{-1}}_{i \text{ times}} \\ & = & \Xi \cdot \Lambda^i \cdot \Xi^{-1} \\ & = & \Xi \cdot \begin{pmatrix} \lambda_1^i & & 0 \\ & \ddots & \\ 0 & & \lambda_n^i \end{pmatrix} \cdot \Xi^{-1}. \end{array} \tag{1.6}$$

That is, calculation of matrix powers reduces to a set of scalar powers. From this it follows that all matrix functions determined as power series can be calculated using their scalar counterparts after the matrix eigenstructure has been

determined. On the other hand, it is interesting to note that matrix functions *cannot* affect the matrix eigenvectors.

The eigenvalues can be determined also as the roots of the *characteristic equation*

$$\det\{\lambda \cdot I - M\} = 0. \tag{1.7}$$

Even though the eigenvalues should not be calculated this way, the roots of high-order polynomials being numerically badly behaving, some theoretical properties can easily be proven in this framework. For example, if a matrix of a form $q \cdot I$, where q is scalar, is added to the matrix M, all of the eigenvalues are shifted by that amount:

$$\det\{(\lambda - q) \cdot I - M\} = 0. \tag{1.8}$$

The properties of the eigenvalues and eigenvectors will be discussed more when we know more about the properties of the matrix M.

Singular value decomposition

The eigenvalue decomposition is defined only for square matrices (and not even all square matrices matrices can be decomposed in such a way). The generalization, the *singular value decomposition (SVD),* on the other hand, is defined[1] for all matrices M:

$$M = \Xi \cdot \Sigma \cdot \Psi^T. \tag{1.9}$$

Here Ξ and Ψ are orthogonal square matrices, so that $\Xi^T\Xi = I$ and $\Psi^T\Psi = I$, and Σ is a diagonal matrix of *singular values.* Note that Σ does not need to be square; if M has dimension ξ times ζ, where $\xi > \zeta$ (and analogous results are found if $\xi < \zeta$), the decomposition looks like

$$\underset{\xi\times\zeta}{M} = \underset{\xi\times\xi}{\Xi} \cdot \left(\begin{array}{ccc} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_\zeta \\ \hline & \mathbf{0} & \end{array}\right) \cdot \underset{\zeta\times\zeta}{\Psi}^T. \tag{1.10}$$

The singular values σ_i are characteristic to a matrix; they are positive real numbers, and it is customary to construct Σ so that they are ordered in descending order. The singular values are close relatives of eigenvalues: Note that, because of the orthogonality of Ξ and Ψ there holds

$$M^TM = \Psi \cdot \Sigma^T\Sigma \cdot \Psi^T = \Psi \cdot \left(\begin{array}{ccc} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_\zeta^2 \end{array}\right) \cdot \Psi^T \tag{1.11}$$

[1]Here, the `Matlab` convention is followed: Matrices Ξ and Psi are kept invertible (square), meaning that generally Σ is non-square. Other ways to define SVD can be found in other contexts

and

$$MM^T = \Xi \cdot \Sigma\Sigma^T \cdot \Xi^T = \Xi \cdot \left(\begin{array}{cc|c} \sigma_1^2 & & 0 \\ & \ddots & \\ 0 & & \sigma_\zeta^2 \\ \hline & \mathbf{0} & \mathbf{0} \end{array}\right) \cdot \Xi^T. \tag{1.12}$$

These are eigenvalue decompositions of M^TM and MM^T, respectively; this means that the (non-zero) eigenvalues of M^TM (or MM^T) are squares of the singular values of M, the corresponding eigenvectors being collected in Ψ (or Ξ, respectively).

Generalization of functions to non-square matrices can be based in the following matrix power definition, following the idea of (1.6):

$$M^i = \Xi \cdot \left(\begin{array}{cc} \sigma_1^i & 0 \\ & \ddots \\ 0 & \sigma_\zeta^i \\ \hline \multicolumn{2}{c}{\mathbf{0}} \end{array}\right) \cdot \Psi^T. \tag{1.13}$$

Matrix differentiation

Corresponding to the differentiation with respect to a scalar, we can differentiate a scalar-valued function $f(\cdot)$ with respect to a vector; the result is a vector called *gradient.* Assume that the function $f : \mathcal{R}^\zeta \to \mathcal{R}$ is being differentiated:

$$\frac{d}{d\,z} f(z) = \begin{pmatrix} \frac{d}{d\,z_1} f(z) \\ \vdots \\ \frac{d}{d\,z_\zeta} f(z) \end{pmatrix}. \tag{1.14}$$

Note that now we choose that gradients to be column vectors (in literature, this is not always the case). Assuming that the matrix M has dimension $\zeta \times \xi$, its row dimension being compatible with z, so that there exists

$$z^TM = \begin{pmatrix} \sum_{i=1}^\zeta z_i M_{i1} & \cdots & \sum_{i=1}^\zeta z_i M_{i\xi} \end{pmatrix}, \tag{1.15}$$

the differentiation can be carried out columnwise:

$$\begin{aligned} \tfrac{d}{d\,z}\left(z^TM\right) &= \begin{pmatrix} \frac{d}{d\,z_1}\sum_{i=1}^\zeta z_i M_{i1} & \cdots & \frac{d}{d\,z_1}\sum_{i=1}^\zeta z_i M_{i\xi} \\ \vdots & \ddots & \vdots \\ \frac{d}{d\,z_\zeta}\sum_{i=1}^\zeta z_i M_{i1} & \cdots & \frac{d}{d\,z_\zeta}\sum_{i=1}^\zeta z_i M_{i\xi} \end{pmatrix} \\ &= \begin{pmatrix} M_{11} & \cdots & M_{1\xi} \\ \vdots & \ddots & \vdots \\ M_{\zeta 1} & \cdots & M_{\zeta\xi} \end{pmatrix} \\ &= M. \end{aligned} \tag{1.16}$$

This is how one would expect a linear matrix function to behave. On the other hand, it turns out that if z is multiplied from the other side, the matrix has to be transposed:

$$\frac{d}{d\,z}\left(M^T z\right) = M. \tag{1.17}$$

Thus, using the product differentiation rule,

$$\begin{array}{rcl} \frac{d}{d\,z}\left(z^T M z\right) & = & \left(\frac{d}{d\,z}\left(z^T M \bar{z}\right) + \frac{d}{d\,z}\left(\bar{z}^T M z\right)\right)\big|_{\bar{z}=z} \\ & = & \left(M + M^T\right) z. \end{array} \tag{1.18}$$

Here, M must have dimension $\zeta \times \zeta$; $\bar{z}$ is assumed to be a (dummy) constant with respect to z. For symmetric M the above coincides with $2Mz$, something that looks familiar from scalar calculus.

It turns out that more sophisticated differentiation formulas are not at all needed later in this report.

1.2.3 Optimization

Matrix expressions can be optimized (minimized or maximized) similarly as in the scalar case — set the derivative to zero (this time, "zero" is a vector consisting of only zeros):

$$\frac{d}{d\,z} J(z) = \mathbf{0}. \tag{1.19}$$

For matrix functions having quadratic form (like $x^T Ax$) the minima (maxima) are unique; this is the case in all optimization tasks encountered here. For an extremum point to be maximum, for example, the (hemo) *Hessian matrix* must be negative semidefinite:

$$\frac{d^2}{d\,z^2} J(z) = \frac{d}{d\,z}\left(\frac{d}{d\,z} J(z)\right)^T \leq \mathbf{0}. \tag{1.20}$$

Note the transpose; the other gradient must be written as a row vector, so that the final result would be a square matrix. Here "$<$" has to be interpreted (both sides being matrices) so that

$$\xi^T \cdot \left(\frac{d^2}{d\,z^2} J(z)\right) \cdot \xi \leq 0 \tag{1.21}$$

for any (compatible) vector ξ. However, the above approach only works if there are no constraints to be taken care of in the optimization.

Pseudoinverse (case I)

As an example, study the least squares solution when there does not exist any exact solution.

Assume that there holds $x = \theta z + e$, x and θ being fixed, and one wants to solve z so that the norm of the error vector e is minimized. It is now assumed that the dimension of z is *lower* that that of x; this means that the solution generally has no exact solution. The criterion to be minimized, the square of the norm of e is

$$\begin{aligned} J(z) &= e^T e = (x - \theta z)^T (x - \theta z) \\ &= x^T x - x^T \theta z - z^T \theta^T x + z^T \theta^T \theta z. \end{aligned} \tag{1.22}$$

Differentiating one has

$$\frac{d}{d\,z} J(z) = -\theta^T x - \theta^T x + 2\theta^T \theta z = 0, \tag{1.23}$$

so that one can solve

$$\theta^T \theta z = \theta^T x. \tag{1.24}$$

This is called the *normal equation* — it can further be solved if explicit formula for z is needed:

$$z = \left(\theta^T \theta\right)^{-1} \theta^T x. \tag{1.25}$$

1.2.4 Lagrange multipliers

The method of *Lagrange multipliers* is a generic method for solving constrained optimization problems, assuming that the functions involved are continuously differentiable. It must be recognized that this method gives necessary, not sufficient conditions for optimality.

Assume that one should find the maximum (or minimum) of the function $f(z)$, so that there holds $g(z) = 0$. The idea of the Lagrangian method is visualized in Fig. 1.4: at the optimum point, the gradients of f and g must be parallel. In the case of scalar z this means that the gradient of g at z must be a scalar multiple of the gradient of f. For vector z, analogously, there must exist constant vector λ so that

$$\frac{df(z)}{dz} = \lambda^T \cdot \frac{dg(z)}{dz}, \tag{1.26}$$

or, written in the standard form,

$$\frac{d}{dz}\left(f(z) - \lambda^T \cdot g(z)\right) = \mathbf{0}. \tag{1.27}$$

Pseudoinverse (case II)

As an example, study the least squares solution when there is a multitude of possible solutions available.

Assume that there holds $x = \theta z$, and we want to solve z when x and θ are fixed. Further, assume that the dimension of z is now *higher* that that of x; this

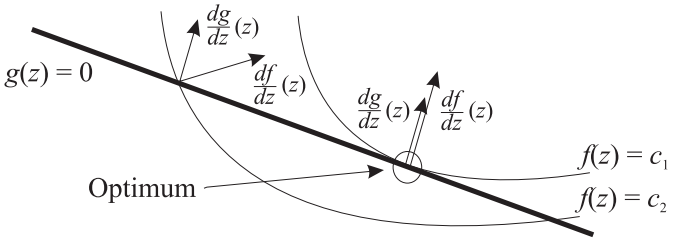

Figure 1.4: Illustration of the optimality condition: in the extremal point the gradients of f and g are parallel

means that the solution is not generally unique. To formulate a well-defined problem, assume that out of the set of candidate solutions, we want to select the "simplest" in the sense of vector size; that is, the criterion to be minimized is $z^T z$.

Now we have the optimality defined in terms of $f(z) = z^T z$ and the constraint function is $g(z) = x - \theta z$, so that

$$\frac{d}{dz}\left(z^T z - \lambda^T \cdot (x - \theta z)\right) = \mathbf{0} \tag{1.28}$$

gives

$$2z + \theta^T \lambda = \mathbf{0}, \tag{1.29}$$

and, further, $z = -\frac{1}{2} \cdot \theta^T \lambda$. From $x = \theta z$ we now get $x = -\frac{1}{2} \cdot \theta\theta^T \lambda$ or $\lambda = (-\frac{1}{2} \cdot \theta\theta^T)^{-1} \cdot x$. Finally, combining this and $z = -\frac{1}{2} \cdot \theta^T \lambda$, one has the least squares solution

$$z = \theta^T \left(\theta\theta^T\right)^{-1} \cdot x. \tag{1.30}$$

In this section, we have found two expressions for the solution of the least-squares problem in different cases. These can be expressed using the so called *pseudoinverse:*

$$\theta^\dagger = \theta^T \left(\theta\theta^T\right)^{-1}, \tag{1.31}$$

or

$$\theta^\dagger = \left(\theta^T\theta\right)^{-1} \theta^T, \tag{1.32}$$

whichever is appropriate; anyway the least-squares solution is given as $z = \theta^\dagger x$.

Note that it is possible that *neither* of the above forms of pseudoinverse is defined, both $\theta^T\theta$ and $\theta\theta^T$ being rank deficient; in such case more general approach to defining pseudoinverse is needed. The *general* pseudoinverse can be calculated utilizing the singular value decomposition (for example, see "`help pinv`" in `Matlab`).

Computer exercises

1. Study the `Matlab` commands `eig` and `svd` (command "`help eig`", etc.). Define the matrix

   $$M = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix},$$

   and construct the eigenvalue and singular value decompositions

   ```
   [XiEIG,Lambda] = eig(M);
   [XiSVD,Sigma,PsiSVD] = svd(M);
   ```

   Study the matrices and explain the results you have when you try

   ```
   XiEIG*Lambda*inv(XiEIG)
   XiSVD*Sigma*inv(PsiSVD)
   ```

   Repeat the above with the matrices M^TM and MM^T. Comment the results in the case where the matrix is defined as

   $$M = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

2. Download the `Regression Toolbox` for `Matlab` from the Internet address `http://saato14.hut.fi/hyotyniemi/publications/01_report125.htm`, and install it following the guidelines presented on page 229. This Toolbox will be used later for illustrating the theoretical derivations.